Applied Mathematics & Information Sciences
An International Journal

http://dx.doi.org/10.12785/amis/080613

Towards Secure Data Exchange in Peer-to-Peer Data Management Systems

Sk. Md. Mizanur Rahman[1], *Mehedi Masud*[2,*], *Ali N M Noman*[3], *Atif Alamri*[1] *and Mohammad Mehedi Hassan*[1]

[1] College of Computer and Information Sciences, King Saud University, KSA
[2] Computer Science Department, Taif University, KSA
[3] School of Electrical Engineering and Computer Science, University of Ottawa, Ottawa, Canada

Received: 14 Oct. 2013, Revised: 12 Jan. 2014, Accepted: 13 Jan. 2014
Published online: 1 Nov. 2014

Abstract: In a peer-to-peer data management system (P2PDMS) peers exchange data in a pair-wise fashion on-the-fly in response to user queries without any centralized control. When peers exchange highly confidential data over an insecure communication channel, the data might be intercepted and read by intruders. As there is no centralized control for data exchange among peers in a P2PDMS, we cannot assume any central third party security infrastructure (e.g. PKI) to protect confidential data. This paper proposes a security protocol for data exchange in P2PDMSs based on pairing-based cryptography and data exchange policy. The protocol allows the peers to compute their secret session keys dynamically during data exchange session by computing a pairing on an elliptic curve, that is based on the policies between them. We show using a formal verification tool that the proposed protocol is safe, and is robust against different attacks including man-in-the middle, the masquerade, and the reply. Furthermore, the computational and communication overhead of the protocol are analyzed.

Keywords: peer-to-peer, pairing-based cryptography, data exchange, data security and authentication.

1. Introduction

A peer-to-peer data management system (P2PDMS) is a collection of autonomous data sources, called peers. The local data sources on peers are called *peer data sources*. Although peer data sources are created independently, data in one peer may semantically relate with data in another peer. Therefore, each peer specifies pair-wise mappings with other peers to share and exchange related data. In the last few years, significant progress has been made in research on various issues related to P2PDMSs, such as peer data exchange settings, data integration models, mediation methods, coordination mechanisms, and mappings [3,4,5,6] among the peer data sources. However, the aspect of sharing data between trusted or acquainted peers in a secured way is given less attention.

In a peer-to-peer system, we cannot assume a fixed secure channel for data exchange between two peers since peers are dynamic and may leave the network any time, 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. There are some security threats that can occur in a P2PDMS during data exchange. In the following we discuss these threats.

Man in the middle Attack (MITM): In MITM attack, an intruder can establish independent connections with the source and the target and relay messages between them. Source and target believe that they are exchanging data without intervening the data exchange policy between them. But, in reality, intruders are controlling the entire data exchange session. Thus, MITM attack is a severe active attack [18] on data exchange in peer-to-peer data management systems. Once a session is intercepted, the intruder acts as a proxy. Thus the intruder becomes another valid peer on the data exchange channel and is able to read, insert, and modify the data in the intercepted data exchange. The prevention technique of MITM attack for our proposed protocol is discussed in Section 6.1.

Replay Attack: A replay attack is an active attack on data exchange channel in a P2PDMS in which a valid

* Corresponding author e-mail: mmasud@tu.edu.sa

data transmission is maliciously or fraudulently repeated or delayed. Suppose P_j is a target peer who wants to authenticate her identity to a source peer, P_i. For valid identification of P_j, P_i requests her password as a proof of identity, which P_j provides to P_i (possibly after some transformation like a hash function). Meanwhile, an intruder peer, P_{EVE}, is eavesdropping on the conversation and is recording the password. After the verification phase is over, P_{EVE} connects to P_i as P_j. Now, if P_i asks P_{EVE} for proof of identity, P_{EVE} sends P_j's password that is recorded in the verification phase. The replay attack prevention mechanism for the proposed protocol is discussed in Section 6.2.

Masquerade Attack: In this attack, an attacker peer (target) may pretend to be a valid peer (target) of a source by disguising its own identity and publishing the identity of a real target peer. Thus, a malicious peer may gain access to the data of the source. The easiest point of entry for a masquerading peer is provided by a weak authentication between the source and the target. Once the malicious node passes the authentication process, it may be authorized by the source as a target to access its data. Similarly, a malicious peer may falsely act as a source for a target. Therefore, a malicious node may be able to tamper with both exchanged data and the data exchange policy between a source and a target. The prevention technique for masquerade attack is discussed in Section 6.3.

Considering the above security threats, the existing conventional Public Key Infrastructure (PKI) is not suitable to apply since a centralized-trusted control system is needed for the PKI.

For achieving secured data exchange in a P2PDMS system, this paper presents a protocol based on Identity Based Encryption (IBE) and pairing-based cryptography[12,11]. Using pairing-based and IBE 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. In order to verify the security features of our proposed protocol, an automated formal validation tool for internet security protocols, namely AVISPA (Automated Validation of Internet Security Protocols & Applications) is used. AVISPA facilitates a language called HLPSL (High Level Protocol Specification Language) to model any security protocol for the verification [8] purpose. The detail about AVISPA can be found in its official web site [9].

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.

1.1. Our Contribution

In this paper, we present a secure data exchange protocol between peers. In our protocol, peers generate session keys on-the-fly for data exchange based on the requested query. The design of the protocol is based on the cryptographic hardness properties of pairing over elliptic curves. When two peers want to exchange data, each of them generates its secret session key using the shared attributes between them through computing a pairing function over an elliptic curve. Once the generation of the secret session key is complete, one peer sends a challenge to the other peer for its authentication; the other peer then sends a corresponding response as the answer to the challenge. If the challenge and response match then the peers begin the data exchange by encrypting the data with their secret session key. Therefore, no malicious nodes can take part in the communication as they are not authenticated among the peers and cannot self-generate the secret session key. As a result, a man-in-the middle attack, masquerade attack, and replay attacks are prevented. In addition, the protocol does not require other trusted third-party centralized control services for authenticated transactions between source and target. Peers can generate their secret session key on-the-fly as well as authenticating one another.

We also conduct an experiment for formal security verification of our protocol using a High Level Protocol Specification Language tool. We extensively analyze the prevention of different attacks that are provided by our protocol and evaluate the computational and communication complexities of the protocol. A short version [1] (three pages) of this paper is presented in a conference where a specific application (eHealth scenario) was considered and only the basic operations are discussed. In this paper an extensive security analysis is presented.

Organization of The Paper: The next section introduces the primitives of cryptography and a formal verification tool that are necessary to describe our protocol. Section 3 describes how the data exchange policy/mapping is established between two peers and the threats that can occur when peers exchange their data in an unsecured channel. In Section 4, we present our cryptographic solution and describe the protocol for exchanging data between peers. In section 5, we discuss issues of cryptographic implementation and security analysis of the protocol. In section 6, we discuss prevention of different attacks that is provided by the protocol. Section 7 describes related work, and finally Section 8 concludes and points out avenues for further research.

2. Cryptographic Primitives and Tool

In this section, we describe some basic cryptographic primitives and mathematical properties which are useful

to understand the protocol. The security strength, computational and communication complexities of the protocol also depend on these primitives. A tool for formal security analysis of internet security protocol is discussed as well, in fact this tool is used to verify the security strength of the the protocol.

2.1. Elliptic Curves

Elliptic curves are considered interesting primarily as an alternative group structure. In regard to implement of common cryptographic protocols, certain advantages come with the elliptic curve families, $E(\mathbb{F}_q) : y^2 = x^3 + Ax + B$ [10]. As there is no known polynomial-time algorithm for the discrete logarithm (DL) problem for the great majority of such curves, much smaller keys can be used. This is one of the major advantages of using these curve families. Given a point P on the curve E defined over a finite field $\mathbb{F}_q$ where $q = p^m$ is the size of the finite field and p is said to be the characteristic of $\mathbb{F}_q$, if p is a large prime then it is computationally difficult to determine "a" for some given "aP". In most circumstances the points on such curve form a simple cyclic group which yield flexible deployment of pairing-based cryptography on such a curve.

At the foundation of every public key cryptosystem there is a hard mathematical problem that is computationally infeasible to solve. The DL problem is the basis for the security of many cryptosystems, including the elliptic curve cryptosystem. More specifically, the ECC relies upon the difficulty of the elliptic curve discrete logarithm problem (ECDLP).

2.2. Bilinear Maps

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 $\mathbb{F}_q$, and G_2 as a subgroup of the multiplicative group of a finite field $\mathbb{F}_{q^k}$ for some $k \in \mathbb{Z}_q^*$, where $\mathbb{Z}_q^* = \{\xi | 1 \leq \xi \leq q-1\}$. 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 $a, b \in \mathbb{Z}_q^*$. 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 [11] and Tate Pairing [12] are examples of cryptographic bilinear maps.

2.2.1. Mathematics of Bilinear Maps

A cryptographic pairing is a bilinear map between two groups in which the discrete logarithm problem is hard and it is used to construct cryptographic protocol (e.g., key exchange, identity-based encryption, short digital signatures, etc.). In practice pairings are based on the Weil and Tate pairings on elliptic curves over finite fields. These pairings are bilinear maps from an elliptic curve group $E(\mathbb{F}_q)$ to the multiplicative group of some extension field $\mathbb{F}^*_{q^k}$. The parameter k is called the embedding degree of the elliptic curve. The pairing is considered to be secure if taking discrete logarithms in the groups $E(\mathbb{F}_q)$ and $\mathbb{F}^*_{q^k}$ are both computationally infeasible. The reduced Tate pairing of order l is the map $e_l : E(\mathbb{F}_q)[l] \times E(\mathbb{F}_{q^k})[l] \longrightarrow \mathbb{F}^*_{q^k}$ which can be defined as $e_l(P,Q) = f_P(\mathscr{B}_Q)^{q^k-1/l}$, where f is a function defined as $f : E(\mathbb{F}_{q^k}) \longrightarrow \mathbb{F}_{q^k}$ [29]. For more details about the pairing mathematics readers can go through the references [13] and [29].

Now we just give a brief overview of the existing algorithms for computing pairing functions which are useful to implement our proposed protocol. Miller first introduced the algorithm for computing Tate pairing in [13] and Duursma *et al.*, formulated for computing Tate pairing for hyperelliptic curves in [14]. Later, Barreto *et al.*, proposed a faster variant of the Tate pairing algorithm for hyperelliptic curves, namely η_T pairing [15]. Finally, in 2007, a faster algorithm for computing η_T pairing over finite fields of characteristic three was proposed by Beuchat *et al.* [17].

2.3. Diffie-Hellman Problems

The group G_1 represents the group of points on the elliptic curve E. 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 $a, b \in \mathbb{Z}_q^*$, 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 $a, b, c \in \mathbb{Z}_q^*$, 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 $a, b, c \in \mathbb{Z}_q^*$, compute $e(P,P)^{abc}$.

Groups where the *CDH* problem is hard but the *DDH* problem is easy are called GAP Diffie-Hellman (*GDH*) groups. Details about *GDH* groups can be found in [11].

2.4. AVISPA: A Verification Tool for Formal Security Analysis

AVISPA tool is one of the well-known automated formal security analysis tools that does not only verify whether a security protocol is 'really' secure or not, but also is able to show all possible attack traces if the security protocol is not secure. Furthermore it is publicly available and comparatively easy to model any security protocol using AVISPA. AVISPA facilitates an extremely expressive and intuitive language called High Level Protocol Specification Language (HLPSL) to the protocol verifier for writing a protocol specification. HLPSL draws its semantic roots from Lamport's Temporal Logic of Actions (TLA) [20]. It allows complicated flow patterns and data structures to be defined and expressed. It supports "Alice-Bob" notation to show how communication takes place between agents. In this language the specification of protocols is written as different roles where roles are played by agents. Each agent has to perform its task as a basic role. The basic role follows event-action based transition: when an event occurs, the agent moves from one state to another after the completion of certain actions. Moreover, an event or action of any agent (i.e. one role) is related to an event or action of one of the remaining agents (i.e. another role); to be more specific, when an agent sends or receives something, there is always another agent who acts as a receiver or sender respectively for that action. There is another type of role, known as a composed role. The composed role instantiates basic roles for modeling the entire protocol or create a session of multiple agents. When the composed role instantiates or runs the entire protocol it is called the main role (also known as environment role). After defining the environment role, there is a need to define the security goals in HLPSL. Once a protocol is modeled in HLPSL it can be executed by AVISPA verification back-ends (e.g. OFMC, CL-AtSe) to check its security goals.

AVISPA uses Dolev-Yao intruder model [21] which assumes that an intruder has all means to interfere with the network and can capture as much traffic as required for analysis. In addition it is also possible to define intruder knowledge in the HLPSL model.

3. Secure Data Exchange Setup

In this section, we introduce the concept of data exchange settings between peers in a P2PDMS and then discuss different security threats that can happen during the exchange of data between peers through an unsecured channel.

3.1. Data Exchange Policy

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.

Generally, in data exchange settings [2], source-to-target data exchange policies are constituted by a set of assertions. 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 1.Consider a family physician database (FDDBS) with the schema S consisting of two relations R_1(OHIP, DOB, Name, Address, Tel, Illness) and R_2(OHIP, TestName, Result, Date). Also consider a database in a medical research cell (MRCDBS) with the schema T consisting of a relation R_3(OHIP, Name, Illness, DOB, TestName, Result). Assume the following policy is assigned between S and T.

$$\forall_{ohip,name,illness,dob,testname,result} \exists_{name,address}$$
$$R_1(ohip, name, address, illness, dob),$$
$$R_2(ohip, testname, result, date)$$
$$\rightarrow R_3(ohip, illness, dob, testname, result)$$

The policy expresses that patients' data (ohip, name, illness, dob, testname, result) are exchanged from FDDBS to MRCDBS. It also shows that the attributes $\{Ohip, Illness, DOB, TestName, Result\}$ are shared between FDDBS and MRCDBS. Although the attributes are shared for MRCDBS, they also contain some confidential attributes e.g. $\{Ohip, DOB\}$ 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 $\{TestName, Result\}$, 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 define the shared attributes, confidential attributes, non-confidential attributes, and private attributes.

Definition 1(Shared attributes). *Consider two peers P_i and P_j in a P2PDMS. 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 $att(\Sigma_{st})$ denotes the set of attributes exposed by P_i using the policy Σ_{st}. Therefore, the shared attributes, denoted by SA, are $SA \subseteq U_s = att(\Sigma_{st})$.*

Definition 2(Confidential attributes). *Consider a data sharing policy $\Sigma_{st} = q_S \rightarrow q_T$ between two peers P_i and P_j. 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 $\Sigma_{st} = q_S \rightarrow q_T$ between two peers P_i and P_j. 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 $\Sigma_{st} = q_S \rightarrow q_T$ 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 2.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.

We now describe a scenario to justify the need of a protocol that secures confidential information of shared attributes during exchange of data in an unsecured channel.

Assume that a user at RDB submits the following query *q*.
SELECT ohip, name, dob, illness, result
FROM R_3
WHERE testname="whitebloodcount"
Since RDB is connected with FDB, the query is forwarded to *RDB* after transformation with respect to the schema of FDB. Suppose the transformed query for FDB is as follows:
SELECT ohip, name, dob, illness, result
FROM R_1, R_2
WHERE (R_1.ohip=R_2.ohip) *and (testname="whitebloodcount")*
When the query is received by FDB, it realizes that the target is requesting some confidential data, for example $\{ohip, dob\}$. It is now the responsibility of FDB to provide the requested data in a secured way because FDB is the "trusted" or "authoritative" source according the data exchange setting. As we discussed in Section 1, there are several security threats that can occur during data exchange from a source to a target.

4. Description of the Protocol

In a P2PDMS, a peer may act as a source and/or a target. For secure data exchange, source and target peers are responsible to generate the secret session key using a pairing function 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. To this end, an "on-the-fly" security setup is needed between the source P_i and the target P_j, based on the query. Since there is no established security mechanism between them, there could be an attack on the communication, which we discussed before in the section 1.

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:

$$\text{Confidential attributes (CA)} = \{CA_1, CA_2, \cdots, CA_m\}$$
$$\text{Non-confidential attributes (NCA)} = \{NCA_1, NCA_2, \cdots, NCA_p\}$$

The purpose of the security protocol is to ensure secure data exchange when P_j requests data from P_i through a query Q that contains confidential attributes as well as non-confidential attributes. Assume a query Q_t at any time instance t is requested from P_j to P_i. Before forwarding the query Q_t, 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_i, peer P_j 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 γ 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_t 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_t:

Confidential attributes in Q_t, denoted by $CA_{Q_t} = \{QCA_1, QCA_2, \cdots, QCA_m\} \subseteq CA$

Non-confidential attributes in Q_t, denoted by $NCA_{Q_t} = \{QNCA_1, QNCA_2, \cdots, QNCA_p\} \subseteq NCA$

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_t^R. 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 $\tilde{e} : G_1 \times G_1 \rightarrow G_2$.
–H_2, H_3, two collision resistant cryptographic hash functions. $H_2 : \{0,1\}^{n-k} \times \{0,1\}^k \rightarrow Z_q^*$, where $Z_q^* = \{\mu | 1 \leq \mu \leq q-1\}$. $H_3 : \{0,1\}^* \rightarrow \{0,1\}^{\lambda}$; a mapping from arbitrary-length strings to λ -bit fixed length string.

Session parameters:

–An ID $ID_{P_i} = H_1(P_i^{\zeta}) \in G_1$, where, ζ 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_{S_i} and authentication code Aut_0 using its own parameters and the parameters of P_j. The generation and purpose of K_{S_i} 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 σ based on the query attribute sets CA_{Q_t} and NCA_{Q_t} as follows:

$$\sigma = H_2(NCA_{Q_t} \times CA_{Q_t}) \in Z_q^*$$

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_{Q_t} as follows:

$$ID_{SP} = H_1(CA_{Q_t}) \in G_1$$

Depending on the query attributes, session key K_{S_i} for each session is generated by the source P_i as follows:

$$K_{S_i} = \tilde{e}(ID_{P_i} + ID_{P_j}, \sigma ID_{SP}) = \tilde{e}(ID_{P_i}, ID_{SP})^{\sigma} \tilde{e}(ID_{P_j}, ID_{SP})^{\sigma}$$

Source P_i also generates authentication code Aut_0 as follows:

$$Aut_0 = H_3(K_{S_i} || ID_{P_i} || ID_{P_j} || R_{i-SESSION} || 0)$$

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 $< G_2, \tilde{e}, H_2, H_3 >$ including the session parameters $< ID_{P_i}, R_{i-SESSION}, Aut_0 >$ to the target P_j.

After receiving the system parameters as well as session parameters from the source P_i, target P_j generates σ and ID_{SP}. Finally target P_j computes a session key K_{S_j} as follows:

$$K_{S_j} = \tilde{e}(ID_{P_j} + ID_{P_i}, \sigma ID_{SP}) = \tilde{e}(ID_{P_i}, ID_{SP})^{\sigma} \tilde{e}(ID_{P_j}, ID_{SP})^{\sigma} = K_{S_i}$$

Target also computes the verification code Ver_0 as follows:

$$Ver_0 = H_3(K_{S_j} || ID_{P_i} || ID_{P_j} || R_{i-SESSION} || 0)$$

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 generates another authentication code Aut_1 as follows:

$$Aut_1 = H_3(K_{S_j} || ID_{P_i} || ID_{P_j} || R_{j-SESSION} || R_{i-SESSION} || 1)$$

where $R_{j-SESSION}$ 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 $< Aut_1, R_{j-SESSION} >$ to source P_i.

Upon receiving $< Aut_1, R_{j-SESSION} >$ from the target P_j, source P_i generates another verification code Ver_1 as follows, and compares it with Aut_1.

$$Ver_1 = H_3(K_{S_i} || ID_{P_i} || ID_{P_j} || R_{j-SESSION} || R_{i-SESSION} || 1)$$

If Ver_1 matches Aut_1 , *i.e* $(Ver_1 = Aut_1)$ then source peer sends the data of the query result Q_t^R by encrypting it with the private session key K_{S_i}.

For distinguishing computation and communication between the source and the target, “0” and “1”are used.

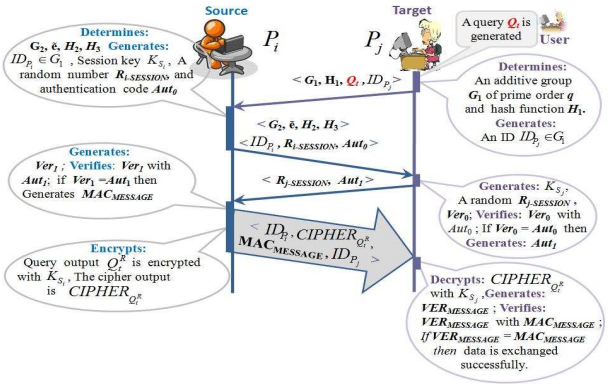


Figure 1 Illustration of proposed protocol for key agreement and secure data exchange in peer-to-peer data management systems.

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_t^R, which is computed as $MAC_{MESSAGE} = H_3(Q_t^R)$. The source also encrypts Q_t^R with its secret session key K_{S_i}, denoted by $CIPHER_{Q_t^R}$, which is computed as $CIPHER_{Q_t^R} = E_{K_{S_i}}(Q_t^R)$, where $E_{K_{S_i}}$ means encryption using the session key K_{S_i}. Finally, P_i sends the following packet to P_j.

$$< ID_{P_i}, CIPHER_{Q_t^R}, MAC_{MESSAGE}, ID_{P_j} >$$

After receiving the packet, P_j decrypts $CIPHER_{Q_t^R}$ with the session key K_{S_j} denoted as $D_{K_{S_j}}(CIPHER_{Q_t^R})$ and generates the verification message authentication code, denoted by $VER_{MESSAGE}$, which is computed as follows:

$$VER_{MESSAGE} = H_3(D_{K_{S_j}}(CIPHER_{Q_t^R}))$$

Finally, P_j 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:

The step-by-step procedure of the proposed protocol goes as follows:

STEP 1: A query Q_t is generated at the target P_j.
STEP 2: Target P_j determines group G_1, hash function H_1 and performs the following steps:
- 2.a: Generates an ID ID_{P_j}.
- 2.b: Sends $< G_1, H_1, Q_t, ID_{P_j} >$ to the source P_i.

STEP 3: Source P_i executes the query Q_t on its local database and performs the following steps:
- 3.a: Determines group G_2, bilinear mapping function $\tilde{e}$, and cryptographic hash functions H_2 and H_3.
- 3.b: Generates an ID ID_{P_i}, a random number$R_{i-SESSION}$.
- 3.c: Generates secret session key K_{S_i}, authentication code Aut_0.
- 3.d: Sends $< G_2, \tilde{e}, H_2, H_3, ID_{P_i}, R_{i-SESSION}, Aut_0 >$ to the target P_j.

STEP 4: Target P_j generates secret session key K_{S_j}, verification code Ver_0.
- 4.a: Generates random $R_{j-SESSION}$.
- 4.b: Compares Ver_0 with Aut_0
  if $Ver_0 = Aut_0$ *then* generates Aut_1.
- 4.c: Sends $< R_{j-SESSION}, Aut_1 >$ to the source P_i.

STEP 5: Source P_i generates verification code Ver_1.
- 5.a: Compares Ver_1 with Aut_1
  if $Ver_1 = Aut_1$ *then* generates *message authentication code* $MAC_{MESSAGE}$.
- 5.b: Encrypts query result Q_t^R, by using the session key K_{S_i} denoted as $CIPHER_{Q_t^R}$.
- 5.c: Sends $< ID_{P_i}, CIPHER_{Q_t^R}, MAC_{MESSAGE}, ID_{P_j} >$ to the target P_j.

STEP 6: Target decrypts $CIPHER_{Q_t^R}$ with session key K_{S_j}; 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 Security Analysis

In this section we discuss a cryptographic implementation of the proposed protocol. To this end, in the following subsections we discuss a suitable choice of key lengths and finite fields for the implementation of the pariing-based cryptosystem.

5.1. Choosing Key Length for a Desired Security Level

RSA Security Systems evaluated the equivalence between the symmetric key systems and RSA Security systems: 1024-bit RSA keys are equivalent in strength to 80-bit symmetric keys, 2048-bit RSA keys to 112-bit symmetric keys and 3072-bit RSA keys to 128-bit symmetric keys [26]. Furthermore NIST [27] key management guidelines suggests that 15360-bit RSA keys are equivalent in strength to 256-bit symmetric keys. For achieving different security levels, NIST has evaluated a

Table 1 Comparable Security Strengths

Security Strength (in bits)	Integer factorization cryptography (IFC) (in bits)	Size of extension field $\mathbb{F}_{q^k}$(in bits)	ECC: Group size of $E(\mathbb{F}_q)[l]$ (in bits)
**	e.g.,RSA	(e.g., DSA, D-H)	e.g., ECDSA
80	1024	1024	160 – 223
112	2048	2048	224 – 255
128	3072	3072	256 – 383
192	7680	7680	384 – 511
256	15360	15360	512+

comparable security strengths among different crypto systems, which is given in Table 1. Column one of the Table 1 indicates the number of bits of security provided by the algorithms and the key sizes in the particular row. Due to the computational advantages of the attackers on the security algorithms, the bits of security is not necessarily the same as the key sizes for the algorithms.

The security of pairing-based cryptosystems is mainly dependent on two basic problems: (i) ECDLP: elliptic curve discrete logarithm problem in the elliptic curve group and (ii) the logarithm problem in the extension field $\mathbb{F}_{q^k}$ [16]. Hence, choosing the size of the group and the extension field are the important factors for the implementation of the proposed protocol. According to the desired level of security which we want to be available for our proposed protocol based on the Table 1, we have to select the size of the extension field and the size of the group. As an example we are considering 80-bit security level; therefore in the next subsection we discuss choosing an elliptic curve with a corresponding appropriate finite field.

5.2. Choosing Elliptic Curves and Finite Fields

Choosing an elliptic curve that is suitable for pairing-based cryptography, there are two options available (i) supersingular curves or (ii) non-supersingular curves of prime characteristic. One of the basic requirements for the selected elliptic curve is that it should have a small embedding degree, or security multiplier [19]. As we are considering 80-bit security strength, the smallest subgroup order of $E(\mathbb{F}_q)[l]$ should be 160 bits long and the size of the extension field $\mathbb{F}_{q^k}$ should be 1024 bits long. Thus the embedding degree k should be close to 6.4.

Supersingular elliptic curves can be constructed on different fields such as prime fields $\mathbb{F}_p$, binary fields $\mathbb{F}_{2^m}$ and fields of characteristic three $\mathbb{F}_{3^m}$. The embedding degree is different for different underlying fields. Table 2 shows some pairing-friendly supersingular elliptic curves, and their required field sizes for achieving 80-bit security level. Considering implementation, the memory required for storing an element in $\mathbb{F}_{3^{97}}$ is less than that for storing an element in $\mathbb{F}_{3^{239}}$ or $\mathbb{F}_{512}$. Furthermore, fields of characteristic three uses the least memory for storing elliptic curve points (base field elements) compared to prime fields and binary fields, though the the extension field size among these three choices of base fields is the same, around 1024 bits long [16].

The curve $E(\mathbb{F}_{p^m}) : y^2 = x^3 + Ax + B$, can be either supersingular or non-supersingular. For supersingular it has an embedding degree of k = 2, and for non-supersingular it has any finite embedding degree with m = 1. There are available efficient algorithms for some non-supersingular elliptic curves to compute pairing, as an example MNT curves [22]. The embedding degree of some MNT curves is also 6, but for Tate pairing computation on such curves it is needed to take inputs from $E(\mathbb{F}_q)[l]$ and $E(\mathbb{F}_{q^k})[l]$, to have an output in $\mathbb{F}^*_{q^k}$. Furthermore, the size of $E(\mathbb{F}_{q^k})[l]$ is very large. On the other hand, there exists a *distortion map* that maps a point from $E(\mathbb{F}_q)[l]$ to a point in $E(\mathbb{F}_{q^k})[l]$ for supersingular elliptic curves. The *distortion map* saves a lot of memory for point storage, and also helps for point computation on supersingular elliptic curves [16]. Thus, for the implementation of our proposed protocol the candidate finite field can be $\mathbb{F}_{3^{97}}$ on the supersingular elliptic curves $y^2 = x^3 - x + 1$ or $y^2 = x^3 - x - 1$.

In the following analysis, we will use the parameter values given above, resulting in the elements in G_1 and G_2 to be roughly 160-bit and 1024-bit, respectively. We further assume SHA-1 [24] is used to compute the keyed-hash message authentication code (AUT_0, AUT_1), which yields a 160-bit output.

5.3. 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_j is two packets that are as follows: (i) First packet with size = (Number of bits to describe Group G_1 + No. of bits to describe H_1 + 160 bit + No. of bits for description of the query), which can be stated as (*Descriptor Packet for* G_1 + *Descriptor Packet for* H_1 + $|G_1|$ *element* + *Descriptor Packet for* Q_t) and the (ii) Second packet with size = (160 bit + 160 bit), which can be stated as ($< Aut_1, R_{j-SESSION} >$)=(160*bit HMAC output* + 160*bit random number*).

Communication overhead for the source peer P_i is two packets that are as follows: (i) First packet ($< G_2, \tilde{e}, H_2, H_3 >$), which can be stated as (*Descriptor Packet for* G_2 + *Descriptor Packet for* $\tilde{e}$ + *Descriptor Packet for* H_2 + *Descriptor Packet for* H_3) and the (ii) Second Packet ($< ID_{P_i}, R_{i-SESSION}, Aut_0 >$) = ($|G_1|$*element* + 160*bit random number* + 160*bit HMAC output*)

Table 2 Some Pairing-friendly Supersingular Elliptic Curves with 80-bit Security Strength

Elliptic Curve Equation	Finite Field	Curve Order	Embedding Degree	Field Size (in bits)
$E(\mathbb{F}_{p^m}) : y^2 = x^3 + Ax + B$	$\mathbb{F}_p$	$p+1$	2	512
$E(\mathbb{F}_{2^m}) : y^2 + y = x^3 + x + b; b \in 0,1$	$\mathbb{F}_2$	$2^m + 1 \pm 2^{(m+1)/2}$	4	239
$E(\mathbb{F}_{3^m}) : y^2 = x^3 - x + b; b \in -1,1$	$\mathbb{F}_3$	$3^m + 1 \pm 3^{(m+1)/2}$	6	$97 \times 2 = 194$

5.4. Computational Cost

The protocol setup involves 1 pairing operation, 1 point addition, 1 point multiplication (for deriving the symmetric key), 2 hash evaluations on H_1, 1 hash evaluation on H_2, 2 hash evaluations on H_3, and 1 random number generation for the source peer P_i as well for the target peer P_j. Hence, 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. The computation tasks for peers include pairing operations (basic pairing and finite field exponentiation), point multiplications and additions, hash operations, etc., among which pairing operations are undoubtedly the most time-consuming task. An example can be found in Tables 3.3, 4.3 and 5.2 of [28]. If the Tate pairing is used for the basic pairing operation, it is shown in [29] that the time taken for computing a Tate pairing is 26.2 ms, in the underlying base field of $F_{3^{97}}$. Tate pairing computation on elliptic curves of characteristic 2 and 3 has been significantly improved [15], which is more realistic in security applications for pairing-based cryptosystems. From this discussion we can conclude that the real-time computation intensity in our protocol is quite acceptable.

5.5. Formal Verification

In this section we discuss formal security analysis of our proposed protocol. To conduct the experiment, we use the AVISPA tool which is discussed in section 2.4. To be more precise, our objective is to check whether the session key generated (separately) by each peer will remain a secret between them and thus an adversary cannot retrieve the query reply. To this end, we model our proposed protocol using HLPSL.

HLPSL model of the proposed protocol: It is comparatively easy to model a protocol when it is represented in ”Alice-Bob Notation” because it gives a clear picture of how communication takes place among agents. That is why, prior to writing the HLPSL model of the proposed protocol, we first represent our proposed protocol in ”Alice-Bob” notation as follows:

$1.Peer_j \rightarrow Peer_i : < G_1, H_1, Q_t, ID_{P_j} >$
$2.Peer_i \rightarrow Peer_j :$
$< G_2, \tilde{e}, H_2, H_3, ID_{P_i}, R_{i-SESSION}, Aut_0 >$
$3.Peer_j \rightarrow Peer_i : < R_{j-SESSION}, Aut_1 >$
$4.Peer_i \rightarrow Peer_j :$
$< ID_{P_i}, CIPHER_{Q_t^R}, MAC_{MESSAGE}, ID_{P_j} >$

From the above Alice-Bob notation of our proposed protocol, it is easy to see that our HLPSL model will have two basic roles for two principals, namely $Peer_i$ and $Peer_j$. Figure 2 shows our HLPSL model of the proposed secure data exchange protocol (shown in Figure 1) in automata format where the state transitions of all basic roles (i.e $Peer_j$ and $Peer_i$; played by agent A and B respectively) have been clearly shown. Since HLPSL is an event-action based model, the words ”event” and ”action” are attached with each transition. Due to space limitations, it is not possible to include the original HLPSL model (i.e., HLPSL code of the proposed protocol) in this paper. Even though the HLPSL model shown in Figure 2 and our proposed protocol (shown in Figure 1) are the same semantically, we find it important to discuss few issues regarding the HLPSL language for the better understanding of our HLPSL model. First of all, in our HLPSL model (Figure 2), the keywords ”RCV” and ”SND” are used to represent receiving and sending message to or from another agent respectively. Secondly, the HLPSL language facilitates a default signal/word called ”start” to show the initiator of the protocol. For instance, in our model $Peer_j$ initiates the communication by receiving a ”special” signal RCV(start). Furthermore, it is important to note that unlike in our proposed model, neither role (i.e. $Peer_j$ and $Peer_i$) sends G_1, G_2, H_1, H_2 and $\tilde{e}$ as a part of its message since the HLPSL language facilitates a role to share/have some prior knowledge. Moreover, in HLPSL, an agent can check the secrecy of a secret as follows: after creating the secret (values or variable) in the basic role, he will write a statement where he specifies the agents to whom it remains a secret. An example of the original HLPSL syntax is given as follows: ***secret(Qt_reply, qt_result_id,*** $\{$***A,B***$\}$***)***. Here ***QT_reply*** is the secret, ***A*** and ***B*** are the agents to whom it remains a secret and ***qt_result_id*** is the *protocol id* which will be invoked from the security goal section of our HLPSL model to check the secrecy of ***Qt_reply***. If no one other than those specified agents (i.e. an intruder) can learn the secret then the protocol will be called safe when

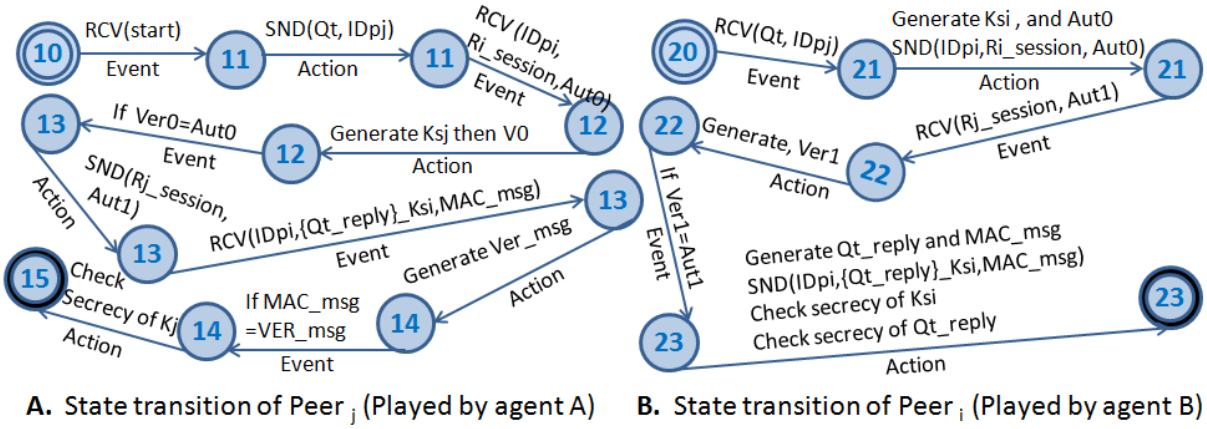


Figure 2 HLPSL model of the proposed secure data exchange protocol for P2PDMS.

it is be executed by AVISPA back-ends; otherwise (i.e. when an intruder can learn the secret) it will be marked as unsafe and the corresponding attack trace will be shown by the AVISPA back-ends.

We execute our HLPSL model of the proposed protocol using the OFMC and CL-AtSe back-ends of AVISPA in order to check two secrecy goals: secrecy of the session key and secrecy of the query-reply. Both of the AVISPA back-ends mark our protocol as safe.

6. Prevention of attacks

In this section we discuss prevention of different attacks by the proposed protocol.

6.1. Man-in-the-middle Attack

In this section we discuss the prevention of man-in-the-middle (MITM) attack provided by our proposed protocol.

In our proposed protocol the secret keys K_{S_i} and K_{S_j} are generated based on a *shared secret parameter*, σ, and a *shared secret identity*, ID_{SP}. The *Shared secret parameter* and *shared secret identity* are computed based on confidential and non-confidential attributes that are only shared between the source and the target peers. Moreover, there are no public parameters associated with σ and ID_{SP}, used to generate session keys K_{S_i}, and K_{S_j}. Hence, by copying public parameters, 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 prevented in our proposed protocol.

6.2. Replay Attack

In our proposed protocol, a malicious peer cannot pass the authentication process. We use an example to illustrate the situation. Consider a scenario with two peers P_i as a source and P_j as a target in a P2PDMS, and a malicious peer P_m wants to mount a replay attack. Suppose that P_j sends a query Q_t to P_i for data exchange and the session/system parameters generated during the data exchange session are $< G_1, H_1, ID_{P_j} >$, $< G_2, \tilde{e}, H_2, H_3 >$, $< ID_{P_i}, R_{i-SESSION}, Aut_0 >$, and $< Aut_1, R_{j-SESSION} >$. The generation of parameters is discussed in Section 4. Assume that when P_j sends Q_t to P_i, P_m makes a copy of Q_t and the session/system parameters during the data exchange session for a replay attack. Later, P_m sends the query Q_t to the source by using the last session parameters $< G_1, H_1, ID_{P_j} >$ for the replay attack. After receiving these parameters, P_i generates a new session and system parameters, and sends them to P_m. Now the random number $R_{i-SESSION}$ is newly generated by source P_i to compute a new authentication code Aut_0 denoted as Aut_0^{new} and a new verification code Ver_1 denoted as Ver_1^{new}. Note that after the session is over P_i and P_j do not store Aut_0, Aut_1, Ver_0, and Ver_1. Since $Ver_1^{new} \neq Aut_1$, where Aut_1 is the old authentication code stored by P_m, P_i does not send the query result Q_t^R to P_m.

If $R_{i-SESSION}$ is generated repeatedly by the source P_i and all the previous session parameters are copied by P_m, still P_m cannot decrypt the query result Q_t^R. Because P_m cannot compute *secret session key* K_{S_i} or K_{S_j}, it cannot complete the authentication process. Hence, the proposed protocol is robust against a replay attack.

6.3. Masquerade Attack

In our 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 $< R_{i-SESSION}, Aut_0, Aut_1, R_{j-SESSION} >$ are completely different in each session. Hence, by storing these session parameters and using these parameters in challange/reponse 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 not effective in our proposed protocol.

7. Related Work

To the best of the knowledge of the authors, our proposal is the first work for query-based secure session key generation for secure data exchange between peers in P2PDMS. There is not enough available research work directly related to the secure data exchange in P2PDMS. The only work that is close to the proposal is the work of [23], where the authors claim secure data propagation among multiple nodes by using pre-existing friendship relationships among the nodes in the network. It is assumed that the nodes are friends with each other in real life, thus they have a pre-existing trust relationship and have secure keys beforehand. This assumption is not realistic, and therefore it is eliminated with no required pre-existing security agreement between the peer nodes, and the security setup is completely based on query, initiated by a target peer in P2PDMS.

8. Conclusion

In this paper, we have extended the protocol in [1] for secure data exchange in a P2PDMS 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. A rigorous formal security analysis is given to prove the security strength of the protocol. The protocol prevents replay attack and masquerade attack, and is robust against a man-in-the middle attack which is extensively analyzed.

Acknowledgement

The authors would like to extend their sincere appreciation to the Deanship of Scientific Research at King Saud University for its funding of this research through the Research Group Project no RGP- VPP-281.

References

[1] Sk. Md. M. Rahman, Md. M. Masud, C. Adams, H. Mouftah and A. Inomata, "Session-wise Private Data Exchange in eHealth Peer-to-Peer Database Management Systems," IEEE International Conference on Intelligence and Security Informatics (ISI2011), July 9-12, Beijing, China, 204-206 (2011).

[2] A. Fuxman, P. G. Kolaitis, R. J. Miller, and W. C. Tan, "Peer Data Exchange," *ACM Trans. Database System*, **31**, 1454-1498 (2005).

[3] P. Rodriguez-Gianolli, M. Garzetti, L. Jiang, A. Kementsietsidis, I. Kiringa, M. Masud, R. Miller, and J. Mylopoulos, "Data Sharing in the Hyperion Peer Database System," *In Proc. of the Int'l Conf. on Very Large Data Bases (VLDB)*, 1291-1294 (2005).

[4] L. Serafini, F. Giunchiglia, J. Molopoulos, and P. Bernstein, "Local Relational Model:A Logocal Formalization of Database Coordination," Technical Report, Informatica e Telecomunicazioni, University of Trento, (2003).

[5] P. Rodriguez-Gianolli, M. Garzetti, L. Jiang, A. Kementsietsidis, I. Kiringa, M. Masud, R. Miller, and J. Mylopoulos, "Data Sharing in the Hyperion Peer Database System," *Proc. of the Int'l Conf. on Very Large Data Bases (VLDB)*, 1291-1294 (2005).

[6] A. Kementsietsidis, M. Arenas, and R.J. Miller, "Mapping Data in Peer-to-Peer Systems: Semantics and Algorithmic Issues," *Proc. of the Int'l Conf. on the Management of Data (ACMSIGMOD)*, 325-336 (2003).

[7] M. Masud and I. Kiringa, "Acquaintance Based Consistency in an Instance-Mapped P2P Data Sharing System During Transaction Processing," *In Proc. of the Int'l Conf. on Cooperative Information Systems (CoopIS)*, 169-187 (2007).

[8] A. Armando, D. Basin, Y. Boichut, Y. Chevalier, L. Compagna, J. Cuellar, et al., "The Avispa Tool for the automated validation of internet security protocols and applications," in Proc. CAV 2005, LNCS 3576, Springer Verlag, (2005).

[9] "AVISPA Project:," available at http://www.avispa-project.org.

[10] "The Tate Pairing," available at http://www.computing.dcu.ie/~~mike/tate.html

[11] D. Boneh and M. Franklin, "Identity-based Encryption from the Weil Pairing," *Proc. CRYPTO 2001, LNCS 2139*, Springer-Verlag, Berlin Heidelberg, Santa Barbara, CA, USA, 213-229 (2001).

[12] R. Sakai, K. Ohgishi, and M. Kasahara, "Cryptosystems Based on Pairing," *Proc. The 2000 Symposium on Cryptography and Information Security (SCIS2000)*, Okinawa, Japan, 26-28 (2000).

[13] V. S. Miller, "Short programs for functions on curves," npublished manuscript available at http://crypto.standford.edu/miller/miller.pdf

[14] I. Duursma and H.S. Lee, "Tate pairing implementation for hyperelliptic cruves $y^2 = x^p - x + d$," *ASIACRYPT 2003*, Taipei, Taiwan, (2003).

[15] P. Barreto, S. Galbraith, C.Ó' hÉigeartaigh, and M. Scott, "Efficient pairing computation on supersingular Abelian varieties," *Designs, Codes and Cryptography*, **42**, 239-271 (2007).

[16] Xiaokang Xiong, Duncan S. Wong, Xiaotie Deng, "TinyPairing: Computing Tate Pairing on Sensor Nodes with Higher Speed and Less Memory," *8th IEEE International Symposium on Network Computing and Applications* (NCA 2009), Cambridge, MA, USA, 187-194 (2009).

[17] J. Beuchat, M. Shirase, T. Takagi, and E. Okamoto, "An algorithm for the η_t pairing calculation in characteristic three and its hardware implementation," *18th IEEE International Symposium on Computer Arithmetic*, Montpellier, France, 97-104 (2007).

[18] Sk. Md. M. Rahman, A. Inomata, M. Mambo and E. Okamoto, "Anonymous On-Demand Position-based Routing in Mobile Ad-hoc Networks," IPSJ (Information Processing Society of Japan) Journal, Japan, ISSN 0387-5806, **47**, August, 2396-2408 (2006).

[19] Michael Scott, Neil Costigan and Wesam Abdulwahab, "Implementing Cryptographic Pairings on Smartcards," *Cryptographic Hardware and Embedded Systems* (CHES 2006), Lecture Notes in Computer Science, **4249/2006**, 134-147 (2006).

[20] L. Lamport, "The temporal logic of actions," *ACM Transactions on Programming Language and Systems*, **16**, 872-923 (1994).

[21] D. Dolev and A. Yao. "On the Security of Public-Key Protocols," *IEEE Transactions on Information Theory*, **29**, (1983).

[22] A. Miyaji, M. Nakabayashi, and S. Takano, "New explicit conditions of elliptic curve traces for FR-reduction," *IEICE Transactions on Fundamentals*, **E84-A**, 1234- 1243 (2001).

[23] Bogdan C. Popescu, Bruno Crispo, Andrew S. Tanenbaum, "Safe and Private Data Sharing with Turtle: Friends Team-Up and Beat the System," *Proc. of the 12th Cambridge Intl. Workshop on Security Protocols 2004*,Lecture Notes in Computer Science(LNCS 3957), Springer-Verlag Berlin Heidelberg 2006; B. Christianson et al. (Eds.): Security Protocols, **2004**, 213-220 (2006).

[24] NIST, "Digital Hash Standard," *Federal Information Processing Standards (FIPS) Publication 180-1*, Apr. (1995).

[25] M. Masud, I. Kiringa, and H. Ural, "Update processing in instance-mapped P2P data sharing systems," *Intl. Journal of Cooperative Information Systems*, **18**, 339-379 (2009).

[26] Burt Kaliski, "TWIRL and RSA Key Size," *RSA Laboratories*, *Revised* May 6, 2003, available at http://www.rsa.com/rsalabs/node.asp?id=2004.

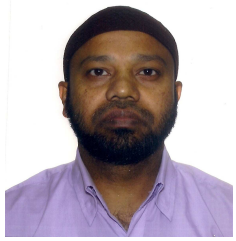
[27] NIST, "Recommendation for key management Part 1: General (Revised) page 63]," 2007. Available at:http://csrc.nist.gov/publications/nistpubs/800-57/sp800-57-Part1-revised2_Mar08-2007.pdf.

[28] H. W. Lim, "On the Application of Identity-Based Cryptography in Grid Security," *Ph.D thesis*, University of London, (2006).

[29] P. S. L. M. Barreto, H. Y. Kim, B. Lynn, and M. Scott, "Efficient algorithms for pairing-based cryptosystems," *CRYPTO 2002*, Springer-Verlag, LNCS, **2442**, 354-368, (2002).

[30] A. N. El-Kassar and R. A. Haraty, "ElGamal Public-key Cryptosystem Using Reducible Polynomials over a Finite Field," Proc. of the 13th International Conference on Intelligent and Adaptive Systems and Software Engineering (IASSE-2004), France, (2004).

[31] R. A. Haraty, A. El-Kassar and B. Shebaro, "A Comparative Study of RSA-based Digital Signature Algorithms," Journal of Mathematics and Statistics, **2**, (2006).



Sk. Md. M. Rahman is an assistant professor in Information System Department in the College of Computer and Information Sciences at King Saud University, KSA. Prior to his current appointment, he worked for several years in cryptography and security engineering in the high-tech industry in Ottawa, Canada. He also worked as a postdoctoral researcher for several years in University of Ottawa, University of Ontario Institute of Technology (UOIT), and University of Guelph, Canada. Information Processing Society Japan (IPSJ) awarded Dr. Rahman with IPSJ Digital Courier Funai Young Researcher Encouragement Award for his excellent contribution in IT security research. He completed a Ph.D. in Risk Engineering in the Laboratory of Cryptography and Information Security, Department of Risk Engineering, University of Tsukuba, Japan, on March 2007. He completed an M.Sc. and a B.Sc. (Hons) in Computer Science, securing first class first with distinction marks in both the programs and awarded with Gold Medal for the result of excellence. Primary research interest of Dr. Rahman is on Cryptography, Software Security, Information Security, Privacy Enhancing Technology and Network Security. He has applied for a patent on white-box cryptography and published over 50 peer-reviewed journal and international conference research papers and book chapters.

Mehedi Masud received his PhD in Computer Science from the University of Ottawa, Canada. He is an Assistant Professor at the Department of Computer Science, Taif University, KSA. His research interests include issues related to P2P and networked data management, query processing and optimization, eHealth, and information security. He has published several research papers at international journals and conferences. He has served as a member of the technical committees of several international conferences and workshops. He is on the editorial board of some journals including Journal of Internet and Information Systems (JIIS), Journal of Engineering and Computer Innovations, and Journal of Software (JWS). He served as a guest editor for Journal of Computer Science and Information Science (ComSIS).

Ali Niman

is a PhD student at University of Ottawa conducting research in the field of cryptography and network security in general and cloud computing security in particular. He received his M.Sc. degree in Information and Communication Systems Security from Royal Institute of Technology (KTH), Sweden in 2007. He had the opportunity to conduct research in security domain at several universities (e.g., University of Trento, Staffordshire University, University of Ulster) in Europe. He has about 6 years of experience in doing formal verification of internet security protocols using AVISPA and has authored/co-authored 7 international conference papers and 3 journal papers.

Atif Alamri

is the Chair of Pervasive and Mobile Computing (CPMC) at King Saud University, Riyadh, Saudi Arabia. He is also an Assistant Professor at the College of Computer and Information Sciences, King Saud University. His research interest includes multimedia assisted health systems, ambient intelligence, and service-oriented architecture. Mr. Alamri is a Guest Associate Editor of the IEEE TRANSACTIONS ON INSTRUMENTATION AND MEASUREMENT, a Cochair of the first IEEE International Workshop on Multimedia Services and Technologies for E-health, a Technical Program Cochair of the 10th IEEE International Symposium on Haptic Audio Visual Environments and Games, and serves as a Program Committee Member of many conferences in multimedia, virtual environments, and medical applications.

M. Mehedi Hassan

received his PhD in Computer Engineering from the Kyung Hee University, Global Campus, South Korea. At present he is an Assistant Professor in Information Systems Department, involved also in Research with Chair of Pervasive and Mobile Computing, College of Computer and Information Sciences, King Saud University, Riyadh, KSA. His main research interests include Cloud computing, Sensor-Cloud integration, Body Sensor Network, publish/subscribe system, event matching algorithm, Grid information retrieval system, storage and processing. He has published several research papers at international journals and conferences and has authored/co-authored 24 international conference papers and 15 journal papers.