Efficient Multi-authorizer Accredited Symmetrically Private Information Retrieval

Mohamed Layouni[1], Maki Yoshida[2], and Shingo Okamura[2]

[1] School of Computer Science, McGill University, Montreal, Canada
[2] Department of Multimedia Engineering, Graduate School of Information Science and Technology, Osaka University, Osaka, Japan

Abstract. We consider a setting where records containing sensitive personal information are stored on a remote database managed by a *storage provider*. Each record in the database is co-owned by a fixed number of parties called data-subjects. The paper proposes a protocol that allows data-subjects to grant access to their records, to self-approved parties, without the DB manager being able to learn if and when their records are accessed. We provide constructions that allow a Receiver party to retrieve a DB record only if he has authorizations from all owners of the target record (respectively, from a subset of the owners of size greater than a threshold.) We also provide a construction where owners of the same record do not have equal ownership rights, and the record in question is retrieved using a set of authorizations consistent with a general access structure. The proposed constructions are efficient and use a pairing-based signature scheme. The presented protocol is proved secure under the Bilinear Diffie-Hellman assumption.

1 Introduction

Achieving a good quality of service and a high operational efficiency have always been a top priority for governments and businesses alike. Over the years, organizations both from the public and private sectors have experimented with a variety of technical choices and policies to improve the quality of their services. One technical choice that seems to be turning into a trend is the widespread adoption of information technologies and the continuous migration of services from the traditional paper-based world to the electronic world. The latter has a number of advantages, among which we note the greater convenience and speed to access data, which in turn translate into shorter processing delays, less errors, better statistics, higher cost-efficiency, and better auditing and fraud detection mechanisms.

Despite all the above benefits, users are still showing a certain reluctance and skepticism towards newly introduced electronic systems. The reason for this skepticism is mainly attributed to the lack of assurances about the way sensitive user data is handled, and the implications that may result from it on users' privacy.

To reduce this lack of trust, it is important that the new systems be designed in a way that gives users increased control over their data. Research on this topic

L. Chen, M.D. Ryan, and G. Wang (Eds.): ICICS 2008, LNCS 5308, pp. 387–402, 2008.

received a significant attention in the past (e.g., [1,2,3,4,5,6]). More recently, a partial solution that contributes to reinforcing user's control over their data, has been proposed in [7]. This solution, called accredited symmetrically private information retrieval (ASPIR), assumes a setting where sensitive information belonging to users (data-subjects) is stored on a remote database DB managed by a party called a *Sender*. The setting includes an additional party called a *Receiver* who retrieves records from the database. The construction in [7], allows a Receiver to retrieve data owned by the user (data-subject), from a database DB managed by the Sender, such that the following three requirements are satisfied: (1) *Privacy for the data-subject:* the Receiver can retrieve a data record only if he has a valid authorization to do so from the record owner, (2) *Privacy for the Receiver:* the Sender is convinced that the Receiver's query is authorized by the owner of the target DB record, without learning any information about the content of the query, or the identity of the record owner, and (3) *Privacy for the Sender:* the Receiver cannot retrieve information about more than one record per query. For example, the Receiver cannot use an authorization from user U to learn information about database records not belonging to U.

The constructions in [7] cover a setting where each record in the database is owned by a single user. In many applications, data records are the property of several parties simultaneously rather than a single one. For example, in the healthcare domain, a medical procedure is performed by a *doctor* on a *patient* within the premises of a *hospital*. It may be natural in some jurisdictions that all three parties, namely the patient, doctor, and hospital, have a right to the database record documenting the medical procedure. As a result, a Receiver (e.g., a second doctor) who wants to have access to the above record, needs an authorization from all three record owners. With the obtained authorizations, the Receiver should be able to retrieve the target record subject to the following conditions: (1) the Receiver can retrieve the record in question only if he has the approval of all record owners, (2) the Sender is convinced that the Receiver's query is approved by the owners of the target data, without learning any information about the index of the target data, or the identity of the authorizers, and (3) the Receiver cannot retrieve information about records other than the one defined in the submitted query.

The ASPIR constructions of [7] rely on privacy-preserving digital credentials [4] to protect the anonymity of the authorizer with respect to the Sender. The digital credential primitive has been used in addition to hide the index of the retrieved record, and to guarantee the unforgeability of the issued authorizations. While highly versatile, the digital credentials of [4] do require a certain amount of computations from the different participants, especially the authorizers. In addition, the construction in [7] assumes that each record owner possesses a digital credential of the type in [4], and that he is willing to use it to issue authorizations.

In this work, we extend the ASPIR protocol of [7] to a context where each database record can have multiple owners. The protocol we present in this paper has a neater and more generic design, and uses SPIR primitives in a black-box

fashion, unlike the construction in [7] which works specifically for Lipmaa's SPIR scheme [8]. Our construction is more efficient than the one in [7], and uses a lightweight pairing-based signature scheme similar to that in [9] instead of digital credentials. In this work, we also propose a t-out-of-n threshold multi-authorizer ASPIR variant, where records can be privately retrieved by a Receiver as long as he has authorizations from t out of the n owners of the target record.

The paper finally treats a setting where the owners' rights to a record are not necessarily equal. For example one could imagine a setting where an authorization from the patient is sufficient to access his medical record, while authorizations from *both* the doctor and hospital are necessary to access the same record. The latter could be useful in cases of emergency where the patient is unable to grant an authorization.

2 Related Work

The problem of managing personal data according to privacy policies defined by the data owners, has been considered by a number of authors. In [10,11], Bagga *et al.* propose a primitive called policy-based encryption. Policy-based encryption allows a user to encrypt a message with respect to an access policy formalized as a monotone Boolean expression. The encryption is such that only a user having access to a qualified set of credentials, complying with the policy, is able to successfully decrypt the message. The context in [10,11], however, is different from the one in this paper, since the goal there is to allow the user to send a secret message to a designated set of players defined by a policy. In our context, the target data is already stored in a database, and the goal is to allow parties authorized by the data owners to retrieve this data, without the database manager learning which data has been retrieved or the identity of the data owners.

In [12], Song *et al.* present a scheme allowing keyword search on encrypted data. Their setting consists of a user, and a server storing encrypted data owned by the user. The server can process search queries on the user's stored ciphertext, only if given proper authorization from the user. The scheme in [12] also supports hidden user queries, where the server conducts the search without learning anything about the content of the query. Although related to our context, it is not clear how the work in [12] can be applied to the problem we describe in this paper, since delegating querying capabilities to a third party (e.g., a Receiver) may require the user to reveal his encryption key, and thus share all of his past and future secrets. Besides, it is not clear how the scheme in [12] can hide the identity of the data-owner from the server, or how it can impose restrictions (e.g., wrt. time or usage) on the search capabilities delegated to a third party.

Finally, in [13] Aiello *et al.* consider a scenario where users privately retrieve data from a database containing a set of priced data items. The proposed protocol is called priced oblivious transfer, and allows a user U, who made an initial deposit, to buy different data items, without the database manager learning which items U is buying, subject to the condition that U's balance contains

sufficient funds. We believe the construction in [13] is the first to consider imposing additional requirements on oblivious transfer protocols. While interesting in their own right, the added requirements do not address the issue of protecting the identity of the data owners.

3 Summary of Contribution and Paper Organization

We propose a multi-authorizer accredited SPIR scheme where data records stored on a Sender's database can be retrieved by a Receiver only if (1) the latter has authorizations to do so from the target record owners, and (2) without the Sender learning information about the index of the retrieved record or the identity of any of the record owners. In addition, the proposed scheme allows record owners to encode, in the issued authorizations, any privacy policy they want to enforce on their data, including the Receiver's identity, an expiry date etc. The paper also proposes a variant scheme for t-out-of-n threshold access, where a Receiver is able to retrieve a data record only if it has authorizations from at least t out of the n owners of the record. Finally, the paper treats a setting where owners of a record have unequal rights. In this setting, records are retrieved in accordance with a general access structure reflecting the non-uniformity of owners' rights.

In Section 4, we introduce few definitions, and describe the SPIR primitive which we use as a building block in our construction. In Section 5, we present our main multi-authorizer ASPIR construction. In Section 6, we evaluate the security and privacy of the proposed scheme. In Section 8, we briefly describe an extension to t-out-of-n threshold access, and treat the more general case where owners have unequal rights in Section 9. We conclude in Section 11.

4 Preliminaries

The construction we present uses a pairing-based signature scheme similar to [9], and relies on the hardness of the Bilinear Diffie-Hellman Problem (BDH). We first introduce bilinear maps, and BDH, and describe the pairing-based signature and SPIR building blocks.

Definition 1 (Admissible bilinear pairings). *Let $(\mathbb{G}_1, \times)$ and $(\mathbb{G}_2, \times)$ be multiplicative groups of the same prime order q. Assume that the discrete logarithm problem in $\mathbb{G}_1$ or $\mathbb{G}_2$ is hard, an admissible bilinear pairing is a map $e : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$ satisfying the following properties:*

- *Bilinearity: For all $P, Q \in \mathbb{G}_1$, and $\alpha, \beta \in \mathbb{Z}_q^*$, $e(P^\alpha, Q^\beta) = e(P, Q)^{\alpha\beta}$.*
- *Non-degeneracy: There exists $P, Q \in \mathbb{G}_1$ such that $e(P, Q) \neq 1_{\mathbb{G}_2}$.*
- *Computability: Given $P, Q \in \mathbb{G}_1$, there is an efficient algorithm to compute $e(P, Q)$.*

Definition 2 (Bilinear Diffie-Hellman Problem). *Let $e : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$ be an admissible bilinear map, and let P be a generator of $\mathbb{G}_1$. For $a, b, c \in \mathbb{Z}_q^*$, given the tuple (P, P^a, P^b, P^c) output $e(P, P)^{abc}$.*

4.1 Pairing-Based Signature Scheme

Let $e : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$ be an admissible bilinear map, and let P be a generator of $\mathbb{G}_1$. Assume the signer has a private key $sk := x \in \mathbb{Z}_q^*$, and a corresponding public key $pk := P^x$. To sign a message m, the signer computes $\sigma := H(m)^x$, where $H : \{0,1\}^* \rightarrow \mathbb{G}_1$ is a public collision-resistant one-way function. The verifier accepts σ' as a valid signature on m' with respect to pk, only if $e(\sigma', P) = e(H(m'), pk)$ holds.

4.2 Symmetrically Private Information Retrieval

A private information retrieval scheme or PIR for short, involves two players: a Sender and a Receiver. The Sender manages a database DB, and answers queries on DB submitted by the Receiver. The main goal of PIR schemes is to allow the Receiver to retrieve a DB record of his choice without the Sender learning the content of his query, and without resorting to the trivial and inefficient method where the Sender just returns the whole database back to the Receiver. The property of hiding the content of the Receiver's query from the Sender is called *Privacy for the Receiver*.

PIR schemes are mainly concerned with providing *Privacy for the Receiver*. There are settings however, where the Sender too is interested in controlling access to his database. For example, the Sender could be a multimedia provider with a business model based on charging a fee for every piece of content accessed in his database. A solution to this type of settings can be obtained by using Symmetrically Private Information Retrieval schemes or SPIR for short.

A SPIR scheme allows a Receiver to efficiently retrieve records from the Sender's database such that the following two properties are assured:

- *Privacy for the Receiver*: the sender does not learn any information about the index of the target record
- *Privacy for the Sender*: the Receiver does not learn any information on the database content, other than the target record.

The above properties, namely Privacy for the Receiver, and Privacy for the Sender can be either perfect, statistical or computational. For example, Lipmaa proposes in [8] a SPIR scheme that is computationally private for the Receiver and perfectly private for the Sender.

A significant number of PIR and SPIR schemes can be found in the literature (e.g., [14,15,16,8,17]) with various performance levels, and a multitude of features such as :

- Single-DB (e.g., [15]) vs. multiple-DB Senders (e.g., [14].)
- Use of algebraic properties (e.g., homomorphic encryption [8] and ϕ-assumption [16]) vs. non-algebraic properties (e.g., existence of one-way trapdoor permutation [15].)
- Index-based (e.g., [8,16]) vs. keyword-based queries (e.g., [18].)

More information on these and other differences can be found in [19,20]. For the purpose of this paper however, we do not discuss these features any further, and use SPIR schemes in a *black-box* fashion.

Notations. In the remainder of this paper we assume that we have a SPIR scheme denoted SPIR. Let s be the secret index of the record the Receiver is interested in. The Receiver uses the public information, and possibly his private information to compute a SPIR query encoding s. We denote by $\mathsf{Q}_{\mathsf{SPIR}}$ the query the Receiver submits to the Sender. Let $\mathsf{R}_{\mathsf{SPIR}}$ be the Sender's answer to the Receiver's query. The Receiver then uses his private information and s, to recover DB$[s]$ from $\mathsf{R}_{\mathsf{SPIR}}$.

5 Protocol Description

The multi-authorizer accredited SPIR protocol we propose relies on the two building blocks described above. We start by describing a first construction in section 5.2, and then present a more efficient one in section 5.3. We assume the public parameters of the above building blocks are already known to all parties: the Sender, the Receiver, and the Authorizers.

5.1 Settings

We assume that multiple parties play the Authorizer role, as opposed to one single party as in [7]. Without loss of generality, we assume that we have *three* types of Authorizers $\mathcal{A}, \mathcal{B}$, and $\mathcal{C}$. For example, $\mathcal{A}$ could represent the Patients, $\mathcal{B}$ the Doctors, and $\mathcal{C}$ the Hospitals. In addition, our setting contains a database DB of size N managed by the Sender. Each record in DB belongs to a triplet of parties (A, B, C) from the set $\mathcal{A} \times \mathcal{B} \times \mathcal{C}$. The owners (A, B, C) of a given record may or may not have the same rights (depending on the privacy laws in place.) Section 9 treats the case where owners have unequal rights.

Next we assume that each party has an identifier *ID*, and that each record in the database is labeled with the identity of its owners, e.g., (ID_A, ID_B, ID_C). We also assume the existence of a publicly known one-to-one correspondence between *ID* triplets and the indexes of DB record, denoted $index : \mathcal{A} \times \mathcal{B} \times \mathcal{C} \rightarrow [1, N]$. Finally we assume that each DB record indexed by j, and corresponding to identity triplet $(ID_{j,1}, ID_{j,2}, ID_{j,3})$, contains a field with the owners' public keys $(pk_{j,1}, pk_{j,2}, pk_{j,3}) := (P^{x_{j,1}}, P^{x_{j,2}}, P^{x_{j,3}})$ stored in it.

5.2 First Construction

Let (A, B, C) be a tuple of owners who are willing to authorize a Receiver *RecID*, to retrieve their record indexed by $s := index(ID_A, ID_B, ID_C)$, according to a usage policy $\mathcal{P}$. Each of the owners first provides the Receiver with a signature $\sigma_i(P_m) := (P_m)^{x_i}$, for $P_m := H(s, RecID, \mathcal{P})$. Next, the Receiver prepares a SPIR query $\mathsf{Q}_{\mathsf{SPIR}}$ for index s, and submits *RecID*, $\mathcal{P}$, and $\mathsf{Q}_{\mathsf{SPIR}}$ to the Sender. Upon receiving this information, the Sender first authenticates[1] *RecID* and

[1] The receiver can be authenticated using conventional X.509 public key certificates for example. In case the identity of the receiver needs to be protected, then privacy-preserving credential systems (e.g., [4,5,6]) can be used instead.

verifies that the submitted query is compliant with usage policy $\mathcal{P}$.[2] If one of these checks fails the Sender aborts, else it proceeds with query. Next, for every Authorizer type[3], the Sender chooses a random blinding factor $\delta_i \in \mathbb{Z}_q^*$, (for the purpose of our description we have $i \in [1,3]$.) For each record DB$[j]$, the Sender computes $P_{mj} := H(j, RecID, \mathcal{P})$ and $\mathrm{DB}'[j] := \mathrm{DB}[j] \times \left(\prod_{i=1}^{3} e((P_{mj})^{\delta_i}, pk_{j,i}) \right)$.

The Sender then executes the SPIR scheme on $\mathsf{Q}_{\mathsf{SPIR}}$ and DB$'$, and returns the response $\mathsf{R}_{\mathsf{SPIR}}$ to the Receiver along with $(P)^{\delta_1}$, $(P)^{\delta_2}$, and $(P)^{\delta_3}$. The Receiver first recovers DB$'[s]$ from $\mathsf{R}_{\mathsf{SPIR}}$, and then computes

$$
\begin{aligned}
\mathrm{DB}_0[s] &= \mathrm{DB}'[s] \;/ \prod_{i=1}^{3} e\left(\sigma_i(P_m), (P)^{\delta_i}\right) \\
&= \mathrm{DB}[s] \times \prod_{i=1}^{3} e\left((P_{m,s})^{\delta_i}, pk_{s,i}\right) / \prod_{i=1}^{3} e\left((P_m)^{x_i}, (P)^{\delta_i}\right) \\
&= \mathrm{DB}[s] \times \left(\prod_{i=1}^{3} e\left((P_{m,s})^{\delta_i}, P^{x_{s,i}}\right) / e\left((P_m)^{x_i}, P^{\delta_i}\right) \right) \\
&\overset{(*)}{=} \mathrm{DB}[s] \times \left(\prod_{i=1}^{3} e\left((P_m)^{\delta_i}, P^{x_i}\right) / e\left((P_m)^{x_i}, P^{\delta_i}\right) \right) \\
&= \mathrm{DB}[s]
\end{aligned}
$$

$(*)$: the equality holds because for $s = index(ID_A, ID_B, ID_C)$, the keys $x_{s,i}$ are no other than the secret keys x_i of owners (A, B, C). Similarly $P_{m,s} = P_m$.

In the above solution, the Sender is required to (1) make a number of pairings linear in the number of authorizer types (to compute each $e((P_{mj})^{\delta_i}, pk_{j,i})$, $i \in [1,n]$), and (2) return $(P)^{\delta_i}$ for each authorizer type. This results in computational and communication complexities linear in the number of authorizer types. We improve these complexities in the next section.

5.3 Improved Construction

Let (A, B, C) be a tuple of owners who are willing to authorize a Receiver *RecID*, to retrieve their record indexed by $s := index(ID_A, ID_B, ID_C)$, according to a usage policy $\mathcal{P}$. Each of the owners first provides the Receiver with a signature $\sigma_i(P_m) := (P_m)^{x_i}$, for $P_m := H(s, RecID, \mathcal{P})$. The Receiver aggregates the σ_i's into one single signature $Sig(P_m) := \prod_{u \in \{A,B,C\}} \sigma_u(P_m)$. He then prepares a SPIR query $\mathsf{Q}_{\mathsf{SPIR}}$ for index s, and submits *RecID*, $\mathcal{P}$, and $\mathsf{Q}_{\mathsf{SPIR}}$ to the Sender as in the first construction. The Sender processes the Receiver's query as in the first construction, except that here it chooses a single random blinding factor $\delta \in \mathbb{Z}_q^*$,

[2] The policy $\mathcal{P}$ can be any Boolean statement of the form: "Receiver should be a practicing surgeon accredited by the College of Physicians **AND** Retrieval date prior to 31 July 2009" for instance. The policy can be encoded using state of the art XML format for example.

[3] As noted earlier, to keep the description simple we assumed *three* types $\mathcal{A}, \mathcal{B}$, and $\mathcal{C}$.

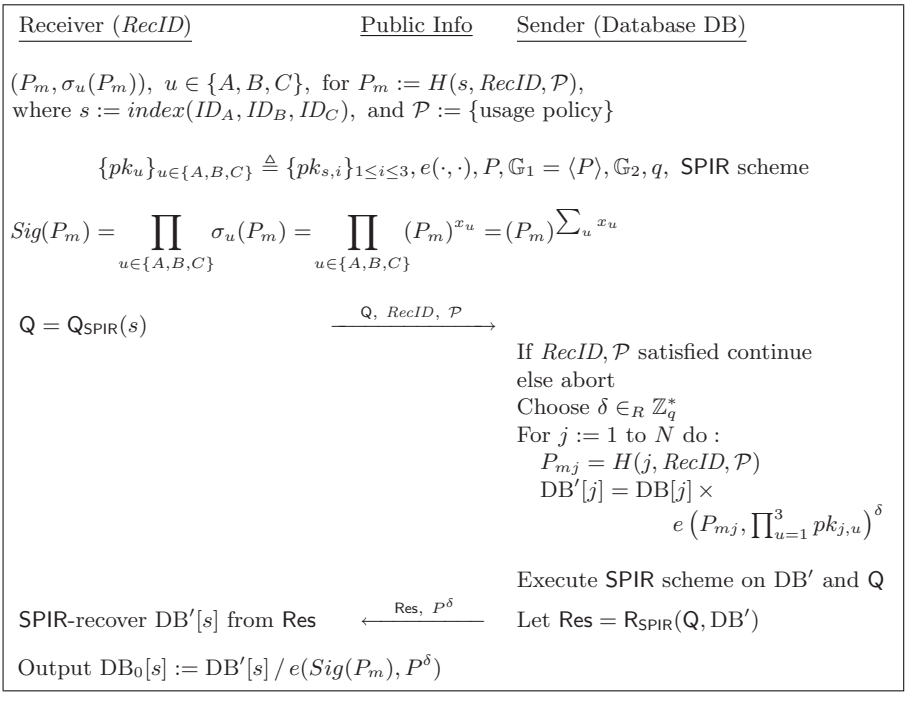

Fig. 1. Multi-Authorizer ASPIR scheme (improved construction)

and for each $1 \le j \le N$, computes $\mathrm{DB}'[j] := \mathrm{DB}[j] \times e\left(P_{mj}, \prod_{u=1}^{3} pk_{j,u}\right)^{\delta}$. The use of a single blinding factor δ for all types of Authorizers will reduce the Sender's computational complexity from linear in the number of Authorizer types to constant. A similar reduction is achieved in the size of the Sender's response which passes from linear in the number of Authorizer types to constant.

Finally, the Sender executes the SPIR scheme on $\mathsf{Q}_{\mathsf{SPIR}}$ and DB′, and returns the response $\mathsf{R}_{\mathsf{SPIR}}$ to the Receiver along with δP. The Receiver then recovers DB′[s] from $\mathsf{R}_{\mathsf{SPIR}}$, and computes $\mathrm{DB}_0[s] = \mathrm{DB}'[s] \,/ e(Sig(P_m), P^\delta)$, thereby using the aggregate signature $Sig(P_m)$ as if it was a "decryption key". This approach of using signatures as decryption keys is of general interest, and could be useful in the wider context of access control. A summary of the whole protocol is given in Figure 1.

It can be easily checked that $\mathrm{DB}_0[s]$ computed by the Receiver is the desired record DB[s].

$$\begin{aligned}
\mathrm{DB}_0[s] &= \mathrm{DB}'[s] \,/\, e(Sig(P_m), P^\delta) \\
&= \mathrm{DB}[s] \times e(P_m, \prod_{u=1}^{3} pk_u)^\delta \,/\, e((P_m)^{\sum_{u=1}^{3} x_u}, P^\delta) \\
&= \mathrm{DB}[s] \times e(P_m, P^{\sum_{u=1}^{3} x_u})^\delta \,/\, e((P_m)^{\sum_{u=1}^{3} x_u}, P)^\delta \\
&= \mathrm{DB}[s]
\end{aligned}$$

Remark. The usage policy $\mathcal{P}$ encoded in P_m can be any privacy policy the owners want enforced on their record. This may include usage limitations such as an expiry date, a description of what is considered an acceptable usage scenario etc. Note that by binding authorizations to a specific Receiver exclusively, the protocol is able to prevent pooling attacks[4].

6 Security and Privacy Evaluation

Definition 3 (Valid Authorization). *Let (A, B, C) be the owners of a record in the Sender's DB, indexed by $s = index(ID_A, ID_B, ID_C)$. For a given usage policy $\mathcal{P}$, a Receiver is said to have a valid authorization under $\mathcal{P}$, from owner $O \in \{A, B, C\}$, if and only if the Receiver has a valid signature from O on $P_m = H(s, ReceiverID, \mathcal{P})$, and $\mathcal{P}$ is satisfied at the time the authorization is used.*

Definition 4 (Secure ASPIR protocols). *An ASPIR protocol is said to be secure if (1) the protocol satisfies the "privacy for Receiver" and "privacy for Sender" properties usually provided by conventional SPIR schemes, and (2) a Receiver cannot retrieve a given record with non-negligible probability unless he has authorizations from all owners of that record. For the special cases of threshold ASPIR (resp., ASPIR with unequal ownership rights), we require the Receiver to have authorizations from a subset of the owners of size greater than a threshold (resp., a subset that is part of a given access structure.)*

Theorem 1. *Assuming the Bilinear Diffie-Hellman problem is hard and the* SPIR *primitive secure, the protocol of Figure 1 is a secure ASPIR protocol.*

Proof. The protocol of Figure 1 is by assumption based on a secure SPIR primitive. By examining the exchange of messages, it is easy to see that the protocol of Figure 1 satisfies the "privacy for Receiver" and "privacy for Sender" properties already provided by the underlying $SPIR$ primitive. In the following we examine the second security criterion of definition 4.

We show that if an Adversary $\mathcal{A}_{ASPIR}$ can retrieve a record that $\mathcal{A}_{ASPIR}$ is not authorized to obtain then the Bilinear Diffie-Hellman problem can be solved. In other words, we show how to construct an Adversary $\mathcal{A}_{BDH}$ that uses $\mathcal{A}_{ASPIR}$ to solve the Bilinear Diffie-Hellman problem.

Let s be the index of the record targeted by the Adversary $\mathcal{A}_{ASPIR}$ playing the role of a malicious Receiver. Let (ID_A, ID_B, ID_C) be the identity tuple of the corresponding owners, i.e., $s = index(ID_A, ID_B, ID_C)$. The Adversary $\mathcal{A}_{ASPIR}$ submits a query and retrieves record DB[s] from the Sender's response without having all required authorizations from owners tuple (A, B, C). In the absence of authorizations from owners tuple (A, B, C), the best scenario for the adversary is to have valid signatures from two (out of the three) owners. Without loss of generality, assume he has signatures from A and B.

[4] Pooling attacks occur when different receivers combine their authorizations in order to gain access to records they were not able to get access to, each on his/her own.

For any given instance $(P', (P')^a, (P')^b, (P')^c)$ of the BDH problem, the Adversary $\mathcal{A}_{BDH}$ obtains $(abc) \cdot e(P', P')$ by interacting with $\mathcal{A}_{ASPIR}$ and playing the role of the owners A and B, and the Sender as follows.

1. $\mathcal{A}_{BDH}$ chooses random elements x_A, x_B of $\mathbb{Z}_q^*$ and sets $P = P'$, $pk_A = (P')^{x_A}$, $pk_B = (P')^{x_B}$, and $pk_C = (P')^c$.
2. $\mathcal{A}_{BDH}$ gives P and $\{pk_i\}_{i\in\{A,B,C\}}$ to $\mathcal{A}_{ASPIR}$.
3. $\mathcal{A}_{BDH}$ sets $P_m = (P')^b$ for the parameters s, *RecID*, and $\mathcal{P}$ (the hash function H is assumed as a random oracle in this proof).
4. $\mathcal{A}_{BDH}$ computes signatures $\sigma_A(P_m) = (P_m)^{x_A}$ and $\sigma_B(P_m) = (P_m)^{x_B}$, and gives them to $\mathcal{A}_{ASPIR}$, along with s, *RecID*, and usage policy $\mathcal{P}$.
5. $\mathcal{A}_{ASPIR}$ submits $\mathsf{Q} := \mathsf{Q}_{\mathsf{SPIR}}(s)$, *RecID*, and usage policy $\mathcal{P}$ to $\mathcal{A}_{BDH}$.
6. $\mathcal{A}_{BDH}$ sets :
   - $\mathrm{DB}_0[j] := e\left(P_m, ((P')^a)^{(x_A+x_B)}\right)$ for all j.
   - $P^\delta := (P')^a$

   $\mathcal{A}_{BDH}$ then executes SPIR on DB_0 and Q and returns $\mathsf{Res} = \mathsf{R}_{\mathsf{SPIR}} = \mathsf{SPIR}(\mathrm{DB}_0, \mathsf{Q})$ and P^δ to $\mathcal{A}_{ASPIR}$.
7. $\mathcal{A}_{ASPIR}$ computes (this step could be done earlier)

$$Sig(P_m) := \prod_{i\in\{A,B,C\}} \sigma_i(P_m) := (P')^{b(x_A+x_B+c)}$$

8. $\mathcal{A}_{ASPIR}$ recovers $\mathrm{DB}_0 = \mathrm{DB}_0[s]$ from Res and computes

$$\begin{aligned}
\mathrm{DB} &= \mathrm{DB}_0 \,/\, e(Sig(P_m), P^\delta) \\
&= e\left(P_m, (P')^{a(x_A+x_B)}\right) \,/\, e((P')^{b(x_A+x_B+c)}, (P')^a) \\
&= e\left((P')^b, (P')^{a(x_A+x_B)}\right) \,/\, e((P')^{b(x_A+x_B+c)}, (P')^a) \\
&= e\,(P', P')^{ab(x_A+x_B)} \,/\, e(P', P')^{ab(x_A+x_B+c)} \\
&= e\,(P', P')^{(ab(x_A+x_B)-ab(x_A+x_B+c))} \\
&= e\,(P', P')^{-(abc)}
\end{aligned}$$

9. $\mathcal{A}_{BDH}$ outputs $\mathrm{DB}^{-1} = e\,(P', P')^{abc}$

$\mathcal{A}_{BDH}$ can solve the BDH problem using $\mathcal{A}_{ASPIR}$. Therefore, assuming the BDH problem is hard, computing a record without all the required valid authorizations is unfeasible. ∎

The above proof can be straightforwardly generalized to the case where records belong to n owners, for n arbitrary. Similar theorems can be proved for the protocol variants of Sections 8, and 9.

7 Performance Analysis

In this analysis we focus mainly on exponentiation operations; group operations such as multiplications are significantly cheaper. A pairing operation can be reduced to a single exponentiation of size less than the group order (as noted in [21]), and is therefore considered as a small-size exponentiation.

It is worth noting at this point that all SPIR schemes require $\Omega(|\mathrm{DB}|)$ computations form the Sender; if this is not the case, then the Sender will not touch at least one record in the database, and thus can safely infer that the untouched records are not being sought in the Receiver's query, thereby violating the Receiver's privacy. As a result of this observation, the Sender's overall computations cannot be expected to drop below this linear lower bound.

Let n be the number of owners of each record in the Sender's database, and let N be the database size. In addition to the basic operations required by the underlying SPIR scheme, our protocol requires : (a) each owner of the target record to perform one pre-computable exponentiation in $\mathbb{G}_1$, (b) the Receiver to perform a pre-computable n-point multiplication in $\mathbb{G}_1$ (to compute $Sig(P_m)$), and one pairing, and (c) the Sender to perform N exponentiations and N pairings. Despite the increase in functionalities, the protocol we propose does not lead to higher computational cost compared to that of the underlying SPIR scheme (which is linear in N.) Similarly, our communication performance is equivalent to that of the underlying SPIR scheme, since we increase the amount of exchanged data only by a small constant. This is negligible, since the best known communication complexity for SPIR achieved so far is $\mathcal{O}\big(\log^2(N)\big)$ [17,8].

8 Extension to Threshold Access

In some applications it may be useful to provide a mechanism to allow a Receiver to privately recover a certain record as long as he has authorizations from t out of the n record owners. As in the basic case, the Sender should not learn the identity of the Authorizers or the index of the retrieved record. We do this using ideas similar to those in [22]. In the following, we only point out the changes from the basic protocol of section 5.

Assume the record owners jointly select a master secret key $MSK := x \in \mathbb{Z}_q^*$, and distribute it verifiably among themselves in a (t, n)-secret sharing scheme. We note that there is no need for a third party in the secret sharing procedure. The n record owners can generate secret key MSK, and privately distribute the shares among themselves without help from a trusted third party using protocols such as [23,24]. The secret generation is such that no shareholder knows MSK individually. Due to space limitations we do not expose the details of those schemes here. Let $x_u,\ u \in [1, n]$ be the n secret shares, and $(sk_u, pk_u) := (x_u, P^{x_u}),\ u \in [1, n]$ the private/public key pairs of the record owners. The master secret key x can be written as a Lagrange interpolation of any subset of shares x_u, of size greater or equal to t. Let $MPK := P^x$ be the corresponding master public key. Finally we assume that each DB record indexed by j, and

corresponding to identity triplet $(ID_{j,1}, \cdots, ID_{j,n})$, contains a field with the master public key MPK_j stored in it. Note that given the owners' public keys $(pk_{j,1}, \cdots, pk_{j,n})$, anyone can reconstruct the corresponding master public key MPK_j by simple Lagrange interpolation.

A Receiver holding authorizations $(P_m, \sigma_u(P_m))$ from at least t record owners $\{u_1, \cdots, u_t\}$, can reconstruct a signature on P_m with respect to the master public key MPK, by computing $Sig(P_m) = \prod_{v=1}^{t} \sigma_{u_v}(P_m)^{L_{u_v}} = (P_m)^{\sum_{v=1}^{t} L_{u_v} x_{u_v}} = (P_m)^x$, where L_{u_v} denote the appropriate Lagrange coefficients[5]. The Receiver then proceeds with the protocol as in the basic case, and submits $\mathsf{Q}_{\mathsf{SPIR}}$, RecID, and usage policy $\mathcal{P}$ to the Sender.

The Sender checks the consistency of the submitted query with the Receiver's identity and usage setting, and chooses a random blinding factor $\delta \in \mathbb{Z}_q^*$. For each record in the database indexed by j, the Sender computes P_{mj}, and $\mathrm{DB}_0[j] = \mathrm{DB}[j] \times e\,(P_{mj}, MPK_j)^{\delta}$. The rest of the protocol is similar to the one in Section 5.

9 Extension to Authorizers with Unequal Rights

Up to this point we have assumed that the owners of a given record all have equal rights. In other words, if a record belongs to (A, B, C) then an authorization from A is worth exactly the same as one from B or C. In some settings however, owners of a record do not have equal rights. For instance in the healthcare context, a medical record belonging to (patient A, doctor B, hospital C) should be accessible only if authorizations are provided, say from A alone, or B and C together. Authorizations from B or C alone are not sufficient. More generally, for a record R owned by a set $O = \{A_1, \cdots, A_n\}$, we denote by $\mathcal{A} \subset 2^O$ the subsets of O whose authorizations are sufficient to access R. The set $\mathcal{A}$ is called a generalized access structure. In the following we show how secret sharing with a generalized access structure [25] can be used to realize multi-authorizer ASPIR in a context where owners have unequal rights to their record.

Consider a database record R, and assume R's owners agree on a generalized access structure $\mathcal{A}$. Using a method similar to that of Section 8, R's owners jointly select a master secret key $MSK := x \in \mathbb{Z}_q^*$, and split it into shares among themselves, according to the access structure $\mathcal{A}$. The secret generation and distribution are such that no shareholder knows MSK individually, and no help from a secret sharing dealer is needed. More details on how this is done are given in the example below. Each owner ends up with a share of information on MSK, that he uses as a signing key. The master public key MPK corresponding to MSK is stored in a field within record R, as in the threshold construction of Section 8. A Receiver then obtains signatures from a subset of owners as in the threshold case. Next, the Receiver combines the partial signatures using Lagrange interpolation in order to recover a valid signature with respect to master key MPK. Recovering this signature is possible only if the Receiver obtains partial signatures from a set of owners that is part of the access structure $\mathcal{A}$.

[5] The values of the L_{u_v}'s depend only on the values of the u_v's.

Example. Let R be a record belonging to (A_1, A_2, A_3, A_4), who agree on access structure $\mathcal{A} = \{\{A_1, A_2, A_3\}, \{A_1, A_4\}, \{A_2, A_4\}, \{A_3, A_4\}\}$. Let $x \in \mathbb{Z}_q^*$ be the master secret key *MSK* that (A_1, A_2, A_3, A_4) select jointly. Let (x_1, x_2, x_3, x_4) be shares of x in a $(4, 4)$-threshold secret sharing scheme. Assume we have a mechanism to securely distribute share tuples (x_2, x_4) to A_1, (x_3, x_4) to A_2, (x_1, x_4) to A_3, and (x_1, x_2, x_3) to A_4. It can be easily seen that the distributed share tuples do satisfy the access structure $\mathcal{A}$. Further details on how share tuples are determined in the general case, can be found in [25].

The received x_i's are used by the owners as private signing keys to issue authorizations. For example, a Receiver authorized by $\{A_1, A_4\} \in \mathcal{A}$, obtains $(P_m, \sigma_2(P_m), \sigma_4(P_m))$ from A_1, and $(P_m, \sigma_1(P_m), \sigma_2(P_m), \sigma_3(P_m))$ from A_4, where $\sigma_i(P_m) = (P_m)^{x_i}$ for $1 \leq i \leq 4$. The Receiver then computes the signature on P_m with respect to master key *MPK*, by interpolating the σ_i's as follows : $\sigma(P_m) = \prod_{v=1}^{4} (\sigma_v(P_m))^{L_v}$, where L_v denote the appropriate Lagrange coefficients. The reconstructed signature is later used by the Receiver to "decrypt" $\mathrm{DB}'[s]$ as in the original ASPIR protocol of Section 5. The rest of the protocol remains the same as in the threshold case.

Now we give a brief overview on how the master secret x is jointly selected by (A_1, A_2, A_3, A_4), and how the shares are generated and distributed. For $1 \leq i \leq 4$, owner A_i chooses $s_i \in_R \mathbb{Z}_q^*$, and generates a random 3^{rd}-degree polynomial in $\mathbb{Z}_q$, $f_i(X) = s_i + \sum_{j=1}^{3} a_{ij} X^j$. Let $f(X) = \sum_{i=1}^{4} f_i(X)$. If we set $x = \sum_{i=1}^{4} s_i$, then $\{x_j = f(j),\ 1 \leq j \leq 4\}$ is valid set of (4,4)-threshold shares of x. Note that x is uniquely determined at this point, and yet unknown to any of the A_i's individually. Next, the share tuples are distributed as follows. Consider for instance the share tuple (x_2, x_4) intended for A_1. For $2 \leq i \leq 4$, owner A_i sends $(f_i(2), f_i(4))$ to A_1. Next, A_1 obtains the desired shares by computing $x_j = \sum_{i=1}^{4} f_i(j)$, for $j \in \{2, 4\}$. The remaining share tuples for A_2, A_3, and A_4 are distributed in the same way. The share distribution above can be made verifiable using the technique of [23].

10 The Case of an Owner Tuple Possessing Multiple Records

So far, we have assumed that each tuple (A, B, C) could own at most one single record. In the following we briefly discuss the case where a tuple of owners may possess $k \geq 1$ records. The goal now is to allow these owners to issue an authorization to the Receiver so that he can retrieve their k records. One trivial way to do this is as follows. First, add one argument to the $index(\ldots)$ function, specifying the rank of record. For example, $s_i = index(A, B, C, i)$ will now denote the index of the i^{th} record (among k) belonging to (A, B, C). The owners now give the Receiver an authorization for each $\mathrm{DB}[s_i]$, and the retrieval proceeds as in the basic case.

To avoid the issuing of multiple authorizations, we can use the following method. The value of P_m in the authorization issued to the Receiver is now computed as $P_m = H(ID_A, ID_B, ID_C, RecID, \mathcal{P})$, and each of the owners provides the

Receiver with a signature $\sigma_i(P_m) := (P_m)^{x_i}$. The Receiver then aggregates the σ_i's into one single signature $Sig(P_m) := \prod_{u\in\{A,B,C\}}\sigma_u(P_m)$ as in section 5.3. A similar modification is required on the Sender's side as well. For $j \in [1, N]$, the Sender computes the P_{mj}'s rather as $P_{mj} = H(ID_{j,1}, ID_{j,2}, ID_{j,3}, RecID, \mathcal{P})$. Note that the identities $ID_{j,u}$, $u \in [1, 3]$, of the record owners are readily available to the Sender along with the corresponding public keys $pk_{j,u}$, $u \in [1, 3]$. The Sender then computes $\mathrm{DB}'[j]$ from $\mathrm{DB}[j]$ as in section 5.3 using the new value of P_{mj} instead. As a result of the above modifications, we note that for all indexes $s_i = index(ID_{j,1}, ID_{j,2}, ID_{j,3}, i)$, $i \in [1, k]$, referencing the records belonging owner tuple $(ID_{j,1}, ID_{j,2}, ID_{j,3})$, the value of P_{mj} is the same, and the entries $\mathrm{DB}[s_i]$ are all encrypted with the same "key": $e(P_{mj}, \prod_{u=1}^{3} pk_{j,u})^{\delta}$. The Receiver finally SPIR retrieves the entries $\mathrm{DB}'[s_i]$ one by one, and decrypts them using his aggregate signature $Sig(P_m)$ as in the basic case.

In the above scheme, the Receiver SPIR retrieves the the $\mathrm{DB}'[s_i]$'s separately. This can be improved using a method based on the hybrid encryption paradigm [26]. First we modify the setting to include two databases DB_1 and DB_2. Each entry in DB_1 is used to store a key corresponding to a triplet of owners. The database DB_2 on the other hand, is used to store the actual owners' records encrypted under the keys kept in DB_1, using some data encapsulation mechanism (DEM)[6][26]. DB_2 is such that the records belonging to a given tuple of owners are all encrypted under the *same* key. In order to grant access to their records, the owners (A, B, C) give the Receiver an authorization to retrieve their encryption key from DB_1 (using the construction of section 5.3.) And using this key, the Receiver decrypts all the DB_2 records belonging to (A, B, C). Note that if DB_2 can be made public, the Receiver does not need to run the SPIR scheme again to retrieve the encrypted records.

11 Conclusion

The paper presents a special access control protocol for databases containing sensitive personal data. In particular, the described constructions allow a Receiver to retrieve a record in the database, if and only if (a) he has authorizations from all (resp. a threshold portion of) the target record owners, and (b) the context in which the database is queried, is consistent with a usage policy chosen by the owners of the target record, and embedded in authorizations issued to the Receiver. The above is achieved without the database manager being able to learn any information about the index of the target record or the identity of its owners. The proposed construction is proved secure under the BDH assumption. The paper also presents a construction where the owners of a record do not have equal ownership rights. The protocol we propose in this paper is more efficient than the one in [7] and can be constructed with any SPIR primitive. Despite the increase in functionality, the presented protocol does not lead to a complexity higher than that of the underlying SPIR.

[6] Note that DEM could be any symmetric-key encryption scheme (e.g., AES.)

Acknowledgments. The first author thanks the IWT-SBO project (ADAPID) "Advanced Applications for Electronic Identity Cards in Flanders", and the Universitary Mission of Tunisia in North America for their support.

References

1. Golle, P., McSherry, F., Mironov, I.: Data collection with self-enforcing privacy. In: ACM Conference on Computer and Communications Security, pp. 69–78 (2006)
2. Ateniese, G., de Medeiros, B.: Anonymous e-prescriptions. In: WPES, pp. 19–31 (2002)
3. Yang, Y., Han, X., Bao, F., Deng, R.H.: A smart-card-enabled privacy preserving e-prescription system. IEEE Transactions on Information Technology in Biomedicine 8(1), 47–58 (2004)
4. Brands, S.: Rethinking Public Key Infrastructures and Digital Certificates: Building in Privacy. MIT Press, Cambridge (2000)
5. Camenisch, J., Lysyanskaya, A.: Efficient non-transferable anonymous multi-show credential system with optional anonymity revocation. In: Pfitzmann, B. (ed.) EUROCRYPT 2001. LNCS, vol. 2045, pp. 93–118. Springer, Heidelberg (2001)
6. Camenisch, J., Lysyanskaya, A.: Signature schemes and anonymous credentials from bilinear maps. In: Franklin, M. (ed.) CRYPTO 2004. LNCS, vol. 3152, pp. 56–72. Springer, Heidelberg (2004)
7. Layouni, M.: Accredited symmetrically private information retrieval. In: Miyaji, A., Kikuchi, H., Rannenberg, K. (eds.) IWSEC 2007. LNCS, vol. 4752, pp. 262–277. Springer, Heidelberg (2007)
8. Lipmaa, H.: An oblivious transfer protocol with log-squared communication. In: Zhou, J., López, J., Deng, R.H., Bao, F. (eds.) ISC 2005. LNCS, vol. 3650, pp. 314–328. Springer, Heidelberg (2005)
9. Boneh, D., Lynn, B., Shacham, H.: Short signatures from the weil pairing. In: Boyd, C. (ed.) ASIACRYPT 2001. LNCS, vol. 2248, pp. 514–532. Springer, Heidelberg (2001)
10. Bagga, W., Molva, R.: Policy-based cryptography and applications. In: S. Patrick, A., Yung, M. (eds.) FC 2005. LNCS, vol. 3570, pp. 72–87. Springer, Heidelberg (2005)
11. Bagga, W., Molva, R.: Collusion-free policy-based encryption. In: Katsikas, S.K., López, J., Backes, M., Gritzalis, S., Preneel, B. (eds.) ISC 2006. LNCS, vol. 4176, pp. 233–245. Springer, Heidelberg (2006)
12. Song, D.X., Wagner, D., Perrig, A.: Practical techniques for searches on encrypted data. In: Proceedings of the 2000 IEEE Symposium on Security and Privacy, pp. 44–55. IEEE Computer Society, Los Alamitos (2000)
13. Aiello, W., Ishai, Y., Reingold, O.: Priced oblivious transfer: How to sell digital goods. In: Pfitzmann, B. (ed.) EUROCRYPT 2001. LNCS, vol. 2045, pp. 119–135. Springer, Heidelberg (2001)
14. Chor, B., Kushilevitz, E., Goldreich, O., Sudan, M.: Private information retrieval. J. ACM 45(6), 965–981 (1998)
15. Kushilevitz, E., Ostrovsky, R.: Replication is not needed: Single database, computationally-private information retrieval. In: FOCS, pp. 364–373 (1997)
16. Cachin, C., Micali, S., Stadler, M.: Computationally private information retrieval with polylogarithmic communication. In: Stern, J. (ed.) EUROCRYPT 1999. LNCS, vol. 1592, pp. 402–414. Springer, Heidelberg (1999)

17. Gentry, C., Ramzan, Z.: Single-database private information retrieval with constant communication rate. In: Caires, L., Italiano, G.F., Monteiro, L., Palamidessi, C., Yung, M. (eds.) ICALP 2005. LNCS, vol. 3580, pp. 803–815. Springer, Heidelberg (2005)
18. Chor, B., Gilboa, N., Naor, M.: Private information retrieval by keywords. Cryptology ePrint Archive, Report 1998/003 (1998)
19. Ostrovsky, R., Skeith III, W.E.: A survey of single-database private information retrieval: Techniques and applications. In: Public Key Cryptography, pp. 393–411 (2007)
20. Gasarch, W.I.: A survey on private information retrieval (column: Computational complexity). Bulletin of the European Association for Theoretical Computer Science 82, 72–107 (2004)
21. Boyen, X.: A promenade through the new cryptography of bilinear pairings. In: IEEE Information Theory Workshop—ITW 2006, pp. 19–23. IEEE Press, Los Alamitos (2006)
22. Boldyreva, A.: Threshold signatures, multisignatures and blind signatures based on the gap-diffie-hellman-group signature scheme. In: Desmedt, Y.G. (ed.) PKC 2003. LNCS, vol. 2567, pp. 31–46. Springer, Heidelberg (2002)
23. Pedersen, T.P.: Non-interactive and information-theoretic secure verifiable secret sharing. In: Feigenbaum, J. (ed.) CRYPTO 1991. LNCS, vol. 576, pp. 129–140. Springer, Heidelberg (1991)
24. Ingemarsson, I., Simmons, G.J.: A protocol to set up shared secret schemes without the assistance of a mutually trusted party. In: Damgård, I.B. (ed.) EUROCRYPT 1990. LNCS, vol. 473, pp. 266–282. Springer, Heidelberg (1991)
25. Ito, M., Saito, A., Nishizeki, T.: Secret sharing scheme realizing general access structure. Electronics and Communications in Japan (Part III: Fundamental Electronic Science) 72(9), 56–64 (1989)
26. Shoup, V.: A proposal for an ISO standard for public key encryption. Cryptology ePrint Archive, Report2001/112 (2001), `http://eprint.iacr.org/`