Extended Private Information Retrieval and its Application in Biometrics Authentications

J. Bringer and H. Chabanne — D. Pointcheval and Q. Tang

Sagem Sécurité, France — Ecole normale supérieure, France

CANS 2007 – December 2007

Outline

Outline

Authentication

Authentication Modes

An authentication protocol usually involves a user and a server, where the user tries to prove his identity to the server with

- the knowledge of a password;
- the knowledge of a private key related to a public key;
- the possession of a device (that securely stores the above private key);
- a biometric feature.

The server needs to apply the protocol with a specific reference, related to the actual user.
$\Longrightarrow$ Privacy concern!

Privacy vs. Authentication

Privacy: What about checking whether a user is authorized, without knowing who he is?

- the knowledge of a private key
  the possession of a device
  $\Longrightarrow$ use of anonymous credentials.
- the knowledge of a password
  a biometric feature
  $\Longrightarrow$ not that simple!

Biometric Authentication

Biometric Template

The biometric template

- cannot be chosen by the user;
- cannot be modified if compromised;
- is slightly different each time.

How to combine biometric authentication with privacy?

Anonymous Biometric Authentication

Anonymous Biometric Authentication

In order to combine both, we want to play the following game:

- the server owns a database with $\{$ID : biometric_reference$\}$
- the user *id* owns an ephemeral biometric template T
- the server wants to check whether T matches to the biometric reference of the user with real identity *id*

for privacy reasons:

- the server should not learn anything about *id* nor T
- a user that claims *id*, but with wrong T, should not learn anything else than *Reject*

Outline

PIR: Private Information Retrieval

Definition (PIR [Chor-Kushilevitz-Goldreich-Sudan '98]**)**

A PIR (Private Information Retrieval) protocol enables a user to retrieve a bit from a bit-database.
When user asks for bit i to the database,

- Soundness: the user actually retrieves the bit i;
- User-Privacy: the database learns nothing about which bit the user has retrieved.

Definition (Symmetric Private Information Retrieval)

An SPIR is a PIR that furthermore provides

- Database-Privacy: the user learns nothing about other bits in the database.

PBR: Private Block Retrieval

Definition (PBR [Chor-Kushilevitz-Goldreich-Sudan '98]**)**

A PBR (Private Block Retrieval) protocol enables a user to retrieve a block from a block-database.

- on the high residuosity [Lipmaa '05]
- on the subgroup decision assumption [Gentry-Ramzan '05]

Notations

We generalize the PIR/PBR setting:

- the database $\mathcal{DB}$ contains a list of N blocks

$$(R_1, R_2, \cdots, R_N)$$

- a user $\mathcal{U}$ can run a protocol to retrieve R_i for any $1 \leq i \leq N$.

EPIR: Extended Private Information Retrieval

A particular case to Secure Function Evaluation can be, for a common function f

- $\mathcal{DB}$ owns $(R_1, \ldots, R_N)$
- $\mathcal{U}$ owns some index i, and an input x

$\mathcal{U}$ wants to learn $f(R_i, x)$, so that

- User-Privacy: $\mathcal{DB}$ learns nothing about the index i, nor the input x
- Database-Privacy: $\mathcal{U}$ learns nothing else than $f(R_i, x)$

This is an extension to PIR: with $f(R_i, x) = R_i$, EPIR=SPIR.

Outline

User-Privacy

The adversary $\mathcal{A}$ plays the role of the database, and tries to learn some information from the user. The function f is fixed:

Definition (User-Privacy)

1. $\mathcal{A}_1$ generates the database: $(R_1, R_2, \cdots, R_N)$;
2. $\mathcal{A}_2$ outputs (i_0, i_1, x_0, x_1);
3. The challenger randomly chooses $b \in \{0, 1\}$
   and issues a *retrieve*-query on input (i_b, x_b) with $\mathcal{A}_3$;
4. $\mathcal{A}_4$ outputs a guess b'.

Database-Privacy

The adversary $\mathcal{A}$ plays the role of the user, and tries to distinguish between the execution with an actual database, from the execution with a simulator. The function f is fixed:

Definition (Database-Privacy)

1. The challenger randomly chooses $b \in \{0, 1\}$.
   If $b = 0$ then $\mathcal{A}$ will interact with an actual database.
   If $b = 1$ then $\mathcal{A}$ will interact with a simulator $\mathcal{S}$ that,
   for a *retrieve*-query on input (i, x), only knows $f(R_i, x)$.
2. The attacker $\mathcal{A}_1$ generates the database: $(R_1, R_2, \cdots, R_N)$.
3. The attacker $\mathcal{A}_2$ issues *retrieve*-queries
   (with either the actual database, or the simulator).
   Then, $\mathcal{A}_2$ outputs a guess b'.

Secure EPIR

An EPIR protocol must satisfy

- Soundness: if both $\mathcal{U}$ and $\mathcal{DB}$ follow the protocol, then *retrieve*(i, x) provides $\mathcal{U}$ with the correct value of $f(R_i, x)$ (at least with an overwhelming probability).
- User-Privacy: any attacker has only negligible advantage in guessing b in the *User-Privacy* attack game.
- Database-Privacy: any attacker has only negligible advantage in guessing b in the *Database-Privacy* attack game.

Outline

ElGamal-based EPIR

One uses the additive variant of ElGamal:

$$sk = x \quad pk = y = g^x \quad \mathcal{E}(m) = \mathcal{E}(m, r) = (g^r, y^r g^m).$$

$\mathcal{U}$ wants to retrieve the value $f(R_i, m) \stackrel{def}{=} (R_i \stackrel{?}{=} m)$:

1. $\mathcal{U}$ generates an ElGamal key pair (pk, sk);
2. $\mathcal{U}$ first sends pk and $c = \mathcal{E}(i||m)$;
3. $\mathcal{DB}$ generates a randomized database:

$$C_j = \left(c/\mathcal{E}(j||R_j)\right)^{r_j} = \mathcal{E}\left(\left(i||m - j||R_j\right) \times r_j\right)$$

4. $\mathcal{U}$ and $\mathcal{DB}$ run a PIR protocol to retrieve C_i: $\mathcal{U}$ then decrypts C_i. it decrypts to 0 iff $m = R_i$.

Security Analysis

Security

- Soundness: PIR is *sound* $\Longrightarrow$ EPIR is *sound*.
- User-Privacy: PIR achieves *user-privacy* + DDH $\Longrightarrow$ EPIR achieves *user-privacy*.
- Database-Privacy: EPIR unconditionally achieves *database-privacy*.

- the PIR does not need to be an SPIR for the *Database-Privacy*: all the fields, except the i-th, are random;
- Any homomorphic encryption scheme can be used.

Weighted Hamming Distance

$\mathcal{U}$ wants to compute the Weighted Hamming Distance between a string S chosen by itself and a block R_i from $\mathcal{DB}$:

- Notation: for an ℓ-bit string S, $S^{(k)}$ is the k-th bit of S.
- Weights: the weight vector is $(w_1, w_2, \cdots, w_\ell)$, where w_k are integers ($1 \leq k \leq \ell$).
- Function:

$$f(R_i, S) = \sum_{k=1}^{\ell} w_k \times (R_i^{(k)} \oplus S^{(k)}).$$

With $w_k = 1\ \forall k$, one obtains the usual Hamming Distance.

BGN Encryption

[Boneh-Goh-Nissim '05]

BGN Parameters

Parameters: $n = pq, G, \mathbb{G}^T, \hat{e}, g, h, G, H$.

- $\mathbb{G}, \mathbb{G}^T$ are groups of order n
- $\hat{e} : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}^T$ is an admissible bilinear map.
- $g \in \mathbb{G}$, $G = \hat{e}(g, g) \in \mathbb{G}^T$ are generators
- $h \in \mathbb{G}$, $H = \hat{e}(g, h) \in \mathbb{G}^T$ are of order p

BGN Encryption Scheme

- Keys: $pk = (n = pq, \mathbb{G}, g, h)$, and $sk = p$.
- Encryption: $\mathcal{E}(m, r) = g^m h^r$, for $m \in \mathbb{Z}_q$
- Decryption of c: compute $c^p = (g^m h^r)^p = (g^p)^m$, then extract the discrete logarithm in base g^p in $\mathbb{G}$.

BGN Encryption Schemes in $\mathbb{G}$ and in $\mathbb{G}^T$

BGN Encryption Scheme in $\mathbb{G}^T$

- Keys: $pk = (n = pq, \mathbb{G}^T, G, H)$, and $sk = p$.
- Encryption: $\mathcal{E}'(m, r) = G^m H^r$, for $m \in \mathbb{Z}_q$
- Decryption of C, compute $C^p = (G^m H^r)^p = (G^p)^m$, Then extract the discrete logarithm in base G^p, in $\mathbb{G}^T$.

Properties

- additively homomorphic: $\mathcal{E}$ in $\mathbb{G}$, and $\mathcal{E}'$ in $\mathbb{G}^T$;
- multiplicatively homomorphic into $\mathbb{G}^T$;
  $\Longrightarrow$ applies once only
- non-interactive zero-knowledge proofs of encryption of 0/1

[Groth-Ostrovsky-Sahai '06]

BGN-based EPIR

$\mathcal{U}$ wants to retrieve $f(R_i, X)$:

1. $\mathcal{U}$ encrypts/sends $c = \mathcal{E}(i)$ and $c_k = \mathcal{E}(X^{(k)})$, with NIZK.
2. $\mathcal{DB}$ checks validity, computes C_j, for every $1 \leq j \leq N$:

$$C_j = \hat{e}(c/\mathcal{E}(j), g)^{r_j} \times \prod m_{j,k}^{w_k}$$

where, for every $1 \leq k \leq \ell$,

$$m_{j,k} = \hat{e}(c_k g^{R_j^{(k)}}, g) \times \hat{e}(c_k, g^{R_j^{(k)}})^{-2} = \mathcal{E}'(X^{(k)} \oplus R_j^{(k)})$$

Then, $C_j = \mathcal{E}'\left(r_j \times (i - j) + \sum w_k \times (X^{(k)} \oplus R_j^{(k)})\right)$

3. $\mathcal{U}$ and $\mathcal{DB}$ run a PIR: $\mathcal{U}$ retrieves C_i, and extracts $f(R_i, X)$.

Outline

Conclusion

We have proposed a new generic primitive:

Extended Private Information Retrieval

- this is a generalization of PIR/SFE
- it allows *private computation of* $f(R_i, x)$ for a client $\mathcal{U}$
  - for fields $(R_1, \ldots, R_N)$, private to $\mathcal{DB}$
  - for an input x and an index i, private to $\mathcal{U}$

with concrete examples for biometric authentication

- equality test (ElGamal): with the use of secure sketches
- Hamming distance (BGN): for iris biometrics