Efficient Receipt-Freeness for e-Voting

David Pointcheval

Joint work with Olivier Blazy, Georg Fuchsbauer and Damien Vergnaud

Ecole normale supérieure, CNRS & INRIA

Chinacrypt – Beijing – China

October 17th, 2010

Outline

1 Introduction
2 Cryptographic Tools
3 Electronic Voting: State-of-the-Art
4 Signatures on Randomizable Ciphertexts

Electronic Voting

Dessert Choice

If one wants to get preferences for the desserts, one asks people to vote for

- Chocolate Cake
- Cheese Cake
- Ice Cream
- Apple

with e.g., possibly 2 choices

After collection of the ballots, one counts the number of choices:

<table>
<thead>
<tr>
<th>Dessert</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>Chocolate Cake</td>
<td>243</td>
</tr>
<tr>
<td>Cheese Cake</td>
<td>111</td>
</tr>
<tr>
<td>Ice Cream</td>
<td>167</td>
</tr>
<tr>
<td>Apple</td>
<td>52</td>
</tr>
</tbody>
</table>

→

1 Chocolate Cake
2 Ice Cream
3 Cheese Cake
4 Apple
Electronic Voting: Basic Properties

**Authentication**
- Only people authorized to vote should be able to vote
- Voters should vote only once

**Anonymity**
- Votes and voters should be unlinkable

**Main Approaches**
- Blind Signatures
- Homomorphic Encryption ← the most promising

---

Homomorphic Encryption & Signature

- The voter generates his vote \( v \in \{0, 1\} \) (for each □)
- The voter encrypts \( v \) to the server \( c = \varepsilon_{pk}(v; r) \)
- The voter signs his vote \( \sigma = S_{usk}(c; s) \)

Such a pair \((c, \sigma)\) is a ballot

- unique per voter, because it is signed by the voter
- anonymous, because the vote is encrypted

Counting: granted homomorphic encryption, anybody can compute

\[
C = \prod c = \prod \varepsilon_{pk}(v_i; r_i) = \varepsilon_{pk}(\sum v_i; \sum r_i) = \varepsilon_{pk}(V; R)
\]

The server decrypts the tally \( V = D_{sk}(C) \), and proves it

---

**Weaknesses**

- **Anonymity**: the server can decrypt any individual vote
  \( \rightarrow \) use of distributed decryption (threshold decryption)
- **Receipt**: if a voter wants to sell his vote, \( r_i \) is a proof
  (a coercer can also provide a modified voting client system
  in order to generate a receipt or even receive it directly)
  \( \rightarrow \) re-randomization of the ciphertext

Distributed decryption is easy (ElGamal, Linear, etc),
while re-randomization of the ciphertext requires more work!

**Receipt-Freeness**

Our goal is to prevent receipts
\( \rightarrow \) receipt-free electronic system
Introduction Cryptographic Tools State-of-the-Art Signatures on Ciphertexts

Outline

1 Introduction

2 Cryptographic Tools
   • Computational Assumptions
   • Signature & Encryption
   • Security
   • Groth-Sahai Methodology

3 Electronic Voting: State-of-the-Art

4 Signatures on Randomizable Ciphertexts

David Pointcheval – 9/43

Assumptions: Diffie-Hellman

Definition (The Computational Diffie-Hellman problem (CDH))

\( \mathbb{G} \) a cyclic group of prime order \( p \).
The CDH assumption in \( \mathbb{G} \) states:
for any generator \( g \leftarrow \mathbb{G} \), and any scalars \( a, b \leftarrow \mathbb{Z}_p^* \),
given \( (g, g^a, g^b) \), it is hard to compute \( g^{ab} \).

Definition (The Decisional Diffie-Hellman problem (DDH))

\( \mathbb{G} \) a cyclic group of prime order \( p \).
The DDH assumption in \( \mathbb{G} \) states:
for any generator \( g \leftarrow \mathbb{G} \), and any scalars \( a, b, c \leftarrow \mathbb{Z}_p^* \),
given \( (g, g^a, g^b, g^c) \), it is hard to decide whether \( c = ab \) or not.

In some pairing-friendly groups, the latter assumption is wrong.

Assumptions: Linear Problem

Definition (Decision Linear Assumption (DLin))

\( \mathbb{G} \) a cyclic group of prime order \( p \).
The DLin assumption states:
for any generator \( g \leftarrow \mathbb{G} \), and any scalars \( a, b, x, y, c \leftarrow \mathbb{Z}_p^* \),
given \( (g, g^x, g^y, g^{xa}, g^{yb}, g^c) \), it is hard to decide whether \( c = a + b \) or not.

Equivalently, given a reference triple \( (u = g^x, v = g^y, g) \) and a new triple \( (U = u^a = g^{xa}, V = v^b = g^{yb}, T = g^c) \), decide whether \( T = g^{a+b} \) or not (that is \( c = a + b \)).

General Encryption Tools: Signature

Definition (Signature Scheme)

\( S = (\text{Setup}, \text{SKeyGen}, \text{Sign}, \text{Verif}) \):
- \( \text{Setup}(1^k) \rightarrow \) global parameters \( \text{param} \);
- \( \text{SKeyGen}(\text{param}) \rightarrow \) pair of keys \( (sk, vk) \);
- \( \text{Sign}(sk, m; s) \rightarrow \) signature \( \sigma \), using the random coins \( s \);
- \( \text{Verif}(vk, m, \sigma) \rightarrow \) validity of \( \sigma \)

If one signs \( F = \mathcal{F}(M) \), for any function \( \mathcal{F} \), one extends the above definitions: \( \text{Sign}(sk, (\mathcal{F}, F, \Pi_M); s) \) and \( \text{Verif}(vk, (\mathcal{F}, F, \Pi_M), \sigma) \) where \( \mathcal{F} \) details the function that is applied to the message \( M \) yielding \( F \), and \( \Pi_M \) is a proof of knowledge of a preimage of \( F \) under \( \mathcal{F} \).
Introduction Cryptographic Tools State-of-the-Art Signatures on Ciphertexts

Signature: Example

In a group $G$ of order $p$, with a generator $g$, and a bilinear map $e: G \times G \rightarrow G_T$

Waters Signature

For a message $M = (M_1, \ldots, M_k) \in \{0, 1\}^*$,
we define $F(M) = u_0 \prod_{i=1}^k u_i^{M_i}$, where $u = (u_0, \ldots, u_k) \ll G^{k+1}$.
For an additional generator $h \ll G$.

- $\text{SKeyGen}: vk = X = g^x$, $sk = Y = h^x$, for $x \ll \mathbb{Z}_p$;
- $\text{Sign}(sk = Y, M; s)$, for $M \in \{0, 1\}^*$ and $s \ll \mathbb{Z}_p$
  $\rightarrow \sigma = (\sigma_1 = Y \cdot F(M)^s, \sigma_2 = g^{-s})$;
- $\text{Verif}(vk = X, M, \sigma = (\sigma_1, \sigma_2))$ checks whether
  $e(g, \sigma_1) \cdot e(F(M), \sigma_2) = e(X, h)$.

General Tools: Encryption

Definition (Encryption Scheme)

$E = (\text{Setup}, \text{EKeyGen}, \text{Encrypt}, \text{Decrypt})$:
- $\text{Setup}(1^k) \rightarrow \text{global parameters param}$;
- $\text{EKeyGen}(\text{param}) \rightarrow \text{pair of keys (pk, dk)}$;
- $\text{Encrypt}(pk, m; r) \rightarrow \text{ciphertext c, using the random coins r}$;
- $\text{Decrypt}(dk, c) \rightarrow \text{plaintext, or } \perp \text{if the ciphertext is invalid}$.

Homomorphic Encryption

For some group laws: $\oplus$ on the plaintext, $\otimes$ on the ciphertext, and $\ominus$ on the randomness

$\text{Encrypt}(pk, m_1; r_1) \otimes \text{Encrypt}(pk, m_2; r_2) = \text{Encrypt}(pk, m_1 \oplus m_2; r_1 \ominus r_2)$

$\text{Decrypt}(sk, \text{Encrypt}(pk, m_1; r_1) \otimes \text{Encrypt}(pk, m_2; r_2)) = m_1 \oplus m_2$

Security

Security Notions: Signature

Signature: EF-CMA

Existential Unforgeability under Chosen-Message Attacks
An adversary should not be able to generate a new valid message-signature pair even if it is allowed to ask signatures on any message of its choice

Impossibility to forge signatures

Waters signature reaches EF-CMA under the CDH assumption
Security Notions: Encryption

**Encryption: IND-CCA**

Indistinguishability under Chosen-Plaintext Attacks

An adversary that chooses two messages, and receives the encryption of one of them, should not be able to decide which one has been encrypted.

**Impossibility to learn any information about the plaintext**

The Linear Encryption reaches IND-CPA under the DLin assumption.

---

**Groth-Sahai Commitments**

Under the DLin assumption, the commitment key is:

\[(u_1 = (u_{1,1}, 1, g), u_2 = (1, u_{2,2}, g), u_3 = (u_{3,1}, u_{3,2}, u_{3,3})) \in (\mathbb{G}^3)^3\]

**Initialization**

\[u_3 = u_1^\lambda \odot u_2^\mu = (u_{3,1} = u_{1,1}^\lambda, u_{3,2} = u_{2,2}^\mu, u_{3,3} = g^{\lambda + \mu})\]

with \(\lambda, \mu \in \mathbb{Z}_p^*,\) and random elements \(u_{1,1}, u_{2,2} \leftarrow \mathbb{G}.\)

It means that \(u_3\) is a linear tuple w.r.t. \((u_{1,1}, u_{2,2}, g).\)

---

**Groth-Sahai Proofs**

- If \(u_3\) a linear tuple, these commitments are perfectly binding.
- With the initialization parameters, the committed values can even be extracted \(\rightarrow\) extractable commitments.
- Using pairing product equations, one can make proofs on many relations between scalars and group elements:

\[\prod_j e(A_j, X_j)^{\alpha_j} \prod_i e(Y_i, B_i)^{\beta_i} \prod_{i,j} e(X_i, Y_j)^{\gamma_{i,j}} = t,\]

where the \(A_j, B_i,\) and \(t\) are constant group elements, \(\alpha_i, \beta_i,\) and \(\gamma_{i,j}\) are constant scalars, and \(X_j\) and \(Y_i\) are either group elements in \(\mathbb{G}_1\) and \(\mathbb{G}_2,\) or of the form \(g_1^{X_j}\) or \(g_2^{Y_i},\) respectively, to be committed.
- The proofs are perfectly sound.
Groth-Sahai Methodology

**Groth-Sahai Proofs**

- If \( u_3 \) a linear tuple, these commitments are perfectly binding
  - The proofs are perfectly sound
- If \( u_3 \) is a random tuple, the commitments are perfectly hiding
  - The proofs are perfectly witness hiding
- Under the \( DLin \) assumption, with a correct initialization, proofs are witness hiding

Can be used for any **Pairing Product Equation**

If one re-randomizes the commitments, the proof can be adapted

---

**Outline**

1. **Introduction**
2. **Cryptographic Tools**
3. **Electronic Voting: State-of-the-Art**
   - General Process
   - Receipt-Freeness
4. **Signatures on Randomizable Ciphertexts**

---

**General Process**

**Dessert Choice**

A ballot consists of one or two crosses in

- Chocolate Cake
- Cheese Cake
- Ice Cream
- Apple

Each box is thus expressed as a bit: \( v_i \in \{0, 1\} \), for \( i = 1, 2, 3, 4 \)

With the additional constraint (at most 2 choices): \( \sum_i v_i \in \{0, 1, 2\} \)

In the following, we focus on one box only:

- \( V_i \) is the \( i \)-th voter
- \( v_i \) is the value of the box for this voter: 0 or 1

**Voting Procedure**

**Cryptographic Primitives**

- Signature \( S = (Setup, SKeyGen, Sign, Verif) \)
  - that is EF-CMA, *e.g.*, Waters Signature;
- Homomorphic enc. \( \mathcal{E} = (Setup, EKeyGen, Encrypt, Decrypt) \)
  - that is IND-CPA, *e.g.*, ElGamal or Linear Encryption

+ distributed decryption, as Linear Encryption scheme allows

**Initialization**

- The authority owns a signing/verification key-pair \((sk, vk)\)
- The ballot-box owns an encryption key \( pk \), which decryption capability is distributed among the board members
- Each voter \( V_i \) owns a signing/verification key-pair \((usk_i, uvk_i)\)
**Voting Procedure**

**Voting Phase**

- **Voter** $V_i$
- $c_i = Encrypt(pk, v_i; r_i)$
- $\sigma_i = Sign(usk_i, c_i; s_i)$
- $\Pi_c = \text{Proof of bit encryption}$
- $\Sigma = \text{Sign}(sk, c_i; s'_i)$

**Counting Procedure**

**Counting Phase**

- Anybody can check all the votes $(c_i, \sigma_i, \Pi_c)$
- Anybody can compute
  
  $C = \prod c_i = \prod E_{pk}(v_i; r_i) = E_{pk}(\sum v_i; \sum r_i) = E_{pk}(V; R)$

  - The board members decrypt $C$ in a distributed and verifiable way, into $V$

**Receipt-Freeness**

- Non-transferable proof of $c'_i \equiv c_i$: verifier-designated proof
  
  - Proof of knowledge of $[r'_i$ such that $c'_i = Random(c_i, r'_i)]$ or $[usk_i]$

**Security**

**Re-Randomization**

- re-randomization: the voter no longer knows the random coins
  
  - designated-verifier proof: voter convinced and non-transferable proof

  The initial proof $\Pi_c$ can be verified on $c$ by the server only

  To get universal verifiability, the proof should be adapted

  Possible with Groth-Sahai methodology

**Weakness: interactions**

Interactive proof: 2-round voting (at best!)

**Non-Interactive Receipt-Freeness**

Our goal: non-interactive receipt-freeness
Introduction

Cryptographic Tools

State-of-the-Art

Signatures on Ciphertexts

Outline

1 Introduction

2 Cryptographic Tools

3 Electronic Voting: State-of-the-Art

4 Signatures on Randomizable Ciphertexts

- Our Full Primitive
- Example
- Security Notions

Signatures on Randomizable Ciphertexts

Our Full Primitive

Example

Security Notions

David Pointcheval – 29/43

Voting Phase

Voter $V_i$

$$c_i = \text{Encrypt}(pk, v_i; r_i)$$

$$\sigma_i = \text{Sign}(usk_i, c_i; s_i)$$

$$\Pi_c = \text{Proof of bit encryption}$$

$$\rightarrow c_i, \sigma_i, \Pi_c \rightarrow (c_i', \sigma_i', \Pi'_c) = \text{Random}(c_i, \sigma_i, \Pi_c; r'_i)$$

$$\leftarrow c_i', \Pi'_c, \Sigma_i \rightarrow \Sigma_i = \text{Sign}(sk_i, (c_i', \Pi'_c); s'_i)$$

The server not only adapts the proof, but the signature too!

- from $(\sigma_i, \Pi_c)$: authorization and uniqueness of a voter
- from $c_i$: privacy for the voter
- from $\text{Random}$: receipt-freeness (unknown random coins $r_i + r'_i$)

David Pointcheval – 30/43

Linear Encryption

In a group $\mathbb{G}$ of order $p$, with a generator $g$, and a bilinear map $e : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$

**Linear Encryption**

- $E\text{KeyGen} : dk = (x_1, x_2) \xleftarrow{\$} \mathbb{Z}_p^2$, $pk = (X_1 = g^{x_1}, X_2 = g^{x_2})$

- $\text{Encrypt}(pk = (X_1, X_2), m; (r_1, r_2))$, for $m \in \mathbb{G}$ and $(r_1, r_2) \xleftarrow{\$} \mathbb{Z}_p^2$

  $$\rightarrow c = (c_1 = X_1^{r_1}, c_2 = X_2^{r_2}, c_3 = g^{r_1 + r_2} \cdot m)$$

- $\text{Decrypt}(dk = (x_1, x_2), c = (c_1, c_2, c_3)) \rightarrow m = c_3 / c_1^{1/x_1} c_2^{1/x_2}$

**Re-Randomization**

- $\text{Random}_e(pk = (X_1, X_2), c = (c_1, c_2, c_3); (r'_1, r'_2))$, for $(r'_1, r'_2) \xleftarrow{\$} \mathbb{Z}_p^2$

  $$\rightarrow c' = (c'_1 = c_1 \cdot X_1^{r'_1}, c'_2 = c_2 \cdot X_2^{r'_2}, c'_3 = c_3 \cdot g^{r'_1 + r'_2})$$

David Pointcheval – 31/43
Waters Signature

In a group $G$ of order $p$, with a generator $g$, and a bilinear map $e : G \times G \to G_T$

**Waters Signature**

[Waters, 2005]

For a message $M = (M_1, \ldots, M_k) \in \{0, 1\}^k$, we define $F = \mathcal{F}(M) = u_0 \prod_{i=1}^k u_i^M$, where $\sigma = (u_0, \ldots, u_k) \in \mathbb{Z}_{p}^{k+1}$. For an additional generator $h \in \mathbb{G}$,

- **KeyGen**: $vk = X = g^x$, $sk = Y = h^x$, for $x \in \mathbb{Z}_p$
- **Sign**($sk = Y, F, s$), for $M \in \{0, 1\}^k$, $F = \mathcal{F}(M)$, and $s \in \mathbb{Z}_p$
  
  $\sigma = (\sigma_1 = Y \cdot F^s, \sigma_2 = g^{-s})$

- **Verif**($vk = X, M, \sigma = (\sigma_1, \sigma_2)$) checks whether
  
  $e(g, \sigma_1) \cdot e(F, \sigma_2) = e(X, h)$.

Re-Randomization of Ciphertext

For a valid ciphertext-signature pair, so that the plaintext is different from all the plaintexts in the ciphertexts sent to the signing oracle

$\sigma_3$ is needed for ciphertext re-randomization

Unforgeability under Chosen-Ciphertext Attacks

The adversary is allowed to ask any valid ciphertext of his choice to the signing oracle

Because of the re-randomizability of the ciphertext-signature, we cannot expect resistance to existential forgeries, but we should allow a restricted malleability only:

**Forgery**

A valid ciphertext-signature pair, so that the plaintext is different from all the plaintexts in the ciphertexts sent to the signing oracle

\[ c = (c_1 = X_{1}^{r_1}, c_2 = X_{2}^{r_2}, \quad c_3 = g^{r_1+r_2} \cdot F) \]

\[ \sigma = (\sigma_1 = Y \cdot c_3^s, \quad \sigma_2 = (c_1^s, c_2^s), \quad \sigma_3 = (g^s, X_1^s, X_2^s) \]

\[ c' = (c_1' = c_1 \cdot X_1^{r_1'}, c_2' = c_2 \cdot X_2^{r_2'}, \quad c_3' = c_3 \cdot g^{r_1'+r_2'} \]

\[ \sigma' = (\sigma_1' = \sigma_{3,0}^{r_1'+r_2'}, \quad \sigma_2' = (\sigma_{2,0} \cdot \sigma_{3,1}^{r_1'}, \sigma_{2,1} \cdot \sigma_{3,2}^{r_2'}), \quad \sigma_3' = \sigma_3 \]

Anybody can publicly re-randomize $c$ into $c'$ with additional random coins $(r_1', r_2')$, and adapt the signature $\sigma$ of $c$ into $\sigma'$ of $c'$.
Unforgeability

From a valid ciphertext-signature pair:
\[ c = (c_1 = X_1^1, c_2 = X_2^2, c_3 = g^{r_1+r_2} \cdot F) \]
\[ \sigma = (\sigma_1 = Y \cdot c_3^s, \sigma_2 = (c_1^s, c_2^s), \sigma_3 = (g^s, X_1^1, X_2^2)) \]

and the decryption key \((x_1, x_2)\), one extracts
\[ F = c_3 / (c_1^{1/x_1} c_2^{1/x_2}) \]
\[ \Sigma = (\Sigma_1 = \sigma_1 / (\sigma_2^{1/x_1} \sigma_2^{1/x_2}), \Sigma_2 = \sigma_3) \]
\[ = (Y \cdot F^s, g^s) \]

Security of Waters signature is for a pair \((M, \Sigma)\)
→ needs of a proof of knowledge \(\Pi_M\) of \(M\) in \(F = \mathcal{F}(M)\)
bit-by-bit commitment of \(M\) and Groth-Sahai proof

Chosen-Message Attacks

From a valid ciphertext \(c = (c_1 = X_1^1, c_2 = X_2^2, c_3 = g^{r_1+r_2} \cdot F)\), and the additional proof of knowledge of \(M\), one extracts \(M\) and asks for a Waters signature:
\[ \Sigma = (\Sigma_1 = Y \cdot F^s, \Sigma_2 = g^s) \]

In this signature, the random coins \(s\) are unknown, we thus need to know the coins in \(c\)

\[ \rightarrow \text{ needs of a proof of knowledge } \Pi_r \text{ of } r_1, r_2 \text{ in } c \]
bit-by-bit commitment of \(r_1, r_2\) and Groth-Sahai proof

From the random coins \(r_1, r_2\) (and the decryption key):
\[ \sigma = (\sigma_1 = \Sigma_1 \cdot \Sigma_2^{r_1+r_2}, \sigma_2 = (\Sigma_2^{r_1}, \Sigma_2^{r_2}), \sigma_3 = (\Sigma_2, \Sigma_2^{r_1}, \Sigma_2^{r_2})) \]
\[ = Y \cdot c_3^s, (c_1^s, c_2^s), (g^s, X_1^1, X_2^2) \]

Chosen-Ciphertext Attacks

A valid ciphertext \(C = (c_1, c_2, c_3, \Pi_M, \Pi_r)\) is a

- ciphertext \(c = (c_1, c_2, c_3)\)
- a proof of knowledge \(\Pi_M\) of the plaintext \(M\) in \(F = \mathcal{F}(M)\)
- a proof of knowledge \(\Pi_r\) of the random coins \(r_1, r_2\)

From such a ciphertext and the decryption key \((x_1, x_2)\), and a Waters signing oracle, one can generate a signature on \(C\)

Forgery

From a valid ciphertext-signature pair \((C, \sigma)\), where \(C\) encrypts \(M\), one can generate a Waters signature on \(M\)

Security

- From the Waters signing oracle, we answer Chosen-Ciphertext Signing queries
- From a Forgery, we build a Waters Existential Forgery

Security Level

Since the Waters signature is EF-CMA under the CDH assumption, our signature on randomizable ciphertext is Unforgeable against Chosen-Ciphertext Attacks under the CDH assumption
Properties

Proofs
Since we use the Groth-Sahai methodology for the proofs $\Pi_M$ and $\Pi_r$:
- in case of re-randomization of $c$, one can adapt $\Pi_M$ and $\Pi_r$
- because of the need of $M$, but also $r_1$ and $r_2$ in the simulation, we need bit-by-bit commitments:
  - $M$ can be short (`bit-long)
  - $r_1$ and $r_2$ are random in $\mathbb{Z}_p$
  $\rightarrow$ $C$ is large!

Efficiency
We can improve efficiency: with a variant of Waters Signature
$\rightarrow$ shorter signatures: $9\ell + 33$ group elements

Conclusion
Extractable Randomizable Signature on Randomizable Ciphertexts

Various Applications
- non-interactive receipt-free electronic voting scheme
- (fair) blind signature

Security relies on the $CDH$ and the $DLin$ assumptions
For an `$\ell$-bit message, ciphertext-signature:
$9\ell + 33$ group elements

A more efficient variant with asymmetric pairing
on the $CDH^*$ and the $SXDH$ assumptions
Ciphertext-signature: $6\ell + 15$ group elements in $G_1$
and $6\ell + 7$ group elements in $G_2$