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>Choice</th>
<th>Votes</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 $c = E_{pk}(v; r)$
- 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

$$C = \prod c = \prod E_{pk}(v_i; r_i) = E_{pk}(\sum v_i; \sum r_i) = E_{pk}(V; R)$$

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

**Security**
- uniqueness per voter: the voter signs his vote
- anonymity: the voter encrypts his vote

**Universal Verifiability**
- identity of voter: proof of identity = signature
- validity of the vote: proof of bit encryption + more
- decryption: proof of decryption

All the steps (voting + counting) can be checked afterwards
Helios is from this family: the IACR e-voting process

**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
**Assumptions: Diffie-Hellman**

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

\[ \mathbb{G} \text{ a cyclic group of prime order } p. \]

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))**

\[ \mathbb{G} \text{ a cyclic group of prime order } p. \]

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))**

\[ \mathbb{G} \text{ a cyclic group of prime order } p. \]

The DLin assumption states:

- for any generator \( g \leftarrow \mathbb{G} \), and any scalars \( a, b, x, y, c \leftarrow \mathbb{Z}_p^* \)
- given \( (g, g^x, g^y, g^{ax}, g^{by}, g^c) \)
- 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 \( (U = u^a = g^{xa}, V = v^b = g^{by}, T = g^c) \),
decide whether \( T = g^{a+b} \) or not (that is \( c = a + b \)).

**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 \( F = F(M) \), for any function \( F\), one extends the above definitions: \( \text{Sign}(sk, (F, F, \Pi_M); s) \) and \( \text{Verif}(vk, (F, F, \Pi_M), \sigma) \) where \( F \) 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 \( F \).
In a group $G$ of order $p$, with a generator $g$, and a bilinear map $e : G \times G \to G_T$

**Waters Signature**

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

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

**Encryption: Example**

In a group $G$ of order $p$, with a generator $g$:

**Linear Encryption**

- **EKeyGen**: $dk = (x_1, x_2) \sim \mathbb{Z}_p^2, pk = (X_1 = g^{x_1}, X_2 = g^{x_2})$;
- **Encrypt**(pk = $(X_1, X_2), m; (r_1, r_2)$), for $m \in G$ and $(r_1, r_2) \sim \mathbb{Z}_p^2$
  → $c = c_1 = X_1^{r_1}, c_2 = X_2^{r_2}, c_3 = g^{r_1+r_2} \cdot m$
- **Decrypt**(dk = $(x_1, x_2), c = (c_1, c_2, c_3)$) → $m = c_3 / c_1^{x_1} c_2^{x_2}$.

**Homomorphism**

$(\oplus_M = x, \otimes_C = x, R = +)$-homomorphism

With $m = g^M$ → $(\oplus_M = +, \otimes_C = x, R = +)$-homomorphism

**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.

**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 e0 1 -30(yption)-278(Cf)-278(oen)-278(Cf)JT 0 -13.549 Td [(thm(),-2 oen
e0 1 -30(ypedA)]JT 0 g 0 GET1 0 0 1 14.173726.634 cm1 0 0 1 2000.02!

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

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

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

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

Group Element Commitment

To commit a group element \(X \in \mathbb{G}\), one chooses random coins \(s_1, s_2, s_3 \in \mathbb{Z}_p\) and sets

\[C(X) := (1, 1, X) \quad u_1^{s_1} \quad u_2^{s_2} \quad u_3^{s_3} \quad u_{1,1} \cdot u_{2,2} \cdot u_{3,2} \cdot X \cdot g^{s_1+s_2} \cdot u_{3,3}^2\]

Scalar Commitment

To commit a scalar \(x \in \mathbb{Z}_p\), one chooses random coins \(\gamma_1, \gamma_2 \in \mathbb{Z}_p\) and sets

\[C(x) := (u_{3,1}^x, u_{3,2}^x, (u_{3,3}g)^x) \quad u_1^{\gamma_1} \quad u_3^{\gamma_2} \quad u_{3,1}^x \cdot u_{3,2}^{x+\gamma_2} \cdot u_{3,3}^{x+\gamma_1} \cdot g^{x+\gamma_1}\]

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:

\[e(A_j, X_j)^{\alpha_{i,j}} e(Y_i, B_i)^{\beta_{i,j}} e(x_i, Y_j)^{\gamma_{i,j}} = t,\]

where the \(A_j, B_i\), and \(t\) are constant group elements, \(\alpha_{i,j}, \beta_{i,j}\), 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 \(g_{1,j}^{x_i}\) or \(g_{2,i}^{y_i}\), 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
   - General Process
   - Receipt-Freeness
4. Signatures on Randomizable Ciphertexts

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): $\sum v_i \in \{0, 1, 2\}$

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 $S = (\text{Setup}, \text{SKGen}, \text{Sign}, \text{Verif})$ that is EF-CMA, e.g., Waters Signature;
- Homomorphic enc. $E = (\text{Setup}, \text{EKeyGen}, \text{Encrypt}, \text{Decrypt})$ 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 \( V_i \)

\[
\begin{align*}
    c_i &= Encrypt(pk, v_i; r_i) \\
    \sigma_i &= Sign(usk_i, c_i; s_i) \\
    \Pi_c &= \text{Proof of bit encryption}
\end{align*}
\]

\( \Pi_c \rightarrow c_i, \sigma_i, \Pi_c \rightarrow \Sigma_i \)

\( \Sigma_i = Sign(sk, c_i; s'_i) \)

- from \((\sigma_i, \Pi_c)\): authorization and uniqueness of a voter
- from \(c_i\): privacy for the voter
- because distributed decryption of the tally only
- with \(\Sigma_i\): a voter can complain if his vote is not in the ballot-box

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 \(V\)

- Everything is verifiable: \textit{universal verifiability}

**Weakness: Receipt**

To sell his vote, the voter reveals his random coins \(r_i\) as a receipt

\textbf{Receipt-freeness:} the voter should not know the random coins \(r_i\)!

Receipt-Freeness

**Re-Randomization**

**Voting Phase**

Voter \( V_i \)

\[
\begin{align*}
    c_i &= Encrypt(pk, v_i; r_i) \\
    \sigma_i &= Sign(usk_i, c_i; s_i) \\
    \Pi_c &= \text{Proof of bit encryption}
\end{align*}
\]

\( \Pi_c \rightarrow c_i, \sigma_i, \Pi_c \rightarrow \Sigma_i \)

\( \Sigma_i = Sign(sk, c_i; s'_i) \)

\( c'_i = Random(c_i; r'_i) \)

\( \sigma_i \equiv c_i \)

\( \Sigma_i = Sign(sk, c_i; s'_i) \)

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**

- \textit{re-randomization:} the voter no longer knows the random coins
- \textit{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 \textit{universal verifiability}, the proof should be adapted

Possible with Groth-Sahai methodology

**Weakness: interactions**

Interactive proof: \textit{2-round} voting (at best!)

**Non-Interactive Receipt-Freeness**

Our goal: \textit{non-interactive} receipt-freeness
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

Voting Phase

Voter \( V_i \)
\[ c_i = Encrypt(pk, v_i; r_i) \]
\[ \sigma_i = Sign(usk, c_i; s_i) \]
\[ \Pi_c = \text{Proof of bit encryption} \]
\[ (c'_i, \sigma'_i, \Pi'_c) = Random(c_i, \sigma_i, \Pi_c; r'_i) \]
\[ c'_i, \Pi'_c, \Sigma_i \]
\[ \Sigma = Sign(sk, (c'_i, \Pi'_c); s'_i) \]

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 \(Random\): receipt-freeness (unknown random coins \(r_i + r'_i\))

Linear Encryption

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

- \( \text{EKeyGen: } dk = (x_1, x_2) \leftarrow \mathbb{Z}_p^2, pk = (X_1 = g^{x_1}, X_2 = g^{x_2}) \)
- \( Encrypt(pk = (X_1, X_2), m; (r_1, r_2)), \) for \( m \in \mathbb{G} \) and \((r_1, r_2) \leftarrow \mathbb{Z}_p^2 \)
  \[ c = c_1 = X_1^{r_1}, c_2 = X_2^{r_2}, c_3 = g^{r_1 + r_2} \cdot m; \]
- \( Decrypt(dk = (x_1, x_2), c = (c_1, c_2, c_3)) \rightarrow m = c_3 / c_1^{1/x_1} c_2^{1/x_2}. \)

- \( \text{Randomization: } pk = (X_1, X_2), c = (c_1, c_2, c_3; (r'_1, r'_2)), \) for \((r'_1, r'_2) \leftarrow \mathbb{Z}_p^2 \)
  \[ c' = c'_1 = c_1 \cdot X_1^{r'_1}, c'_2 = c_2 \cdot X_2^{r'_2}, c'_3 = c_3 \cdot g^{r'_1 + r'_2}. \]
Waters Signature

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

For a message $M = (M_1, \ldots, M_k) \in \{0, 1\}^k$, we define $F = F(M) = u_0 \oplus_{i=1}^k u_i^{M_i}$, where $u = (u_0, \ldots, u_k) \xleftarrow{\$} \mathbb{G}^{k+1}$.

For an additional generator $h \in \mathbb{G}$,

- **SKGen**: $vk = X = g^x$, $sk = Y = h^x$, for $x \xleftarrow{\$} \mathbb{Z}_p$;
- **Sign**: $(sk = Y, F, s)$, for $M \in \{0, 1\}^k$, $F = F(M)$, and $s \xleftarrow{\$} \mathbb{Z}_p$.

We get $\sigma = (\sigma_1, \sigma_2, \sigma_3) = (g^s, X_1^s, X_2^s)$, and the verification is $e(g, \sigma_1) \cdot e(F, \sigma_2) = e(X, h)$.

Waters Signature [Waters, 2005]

Re-randomization of Ciphertext

$\sigma = (\sigma_1, \sigma_2, \sigma_3)$

$\sigma' = (\sigma_1', \sigma_2', \sigma_3')$

$\sigma_1' = \sigma_1 \cdot \sigma_3^{r_1} \cdot \sigma_3^{r_2}$

$\sigma_2' = \sigma_2 \cdot \sigma_3^{r_1} \cdot \sigma_3^{r_2}$

$\sigma_3' = \sigma_3 \cdot g^{r_1 + r_2}$

where $(r_1, r_2) \xleftarrow{\$} \mathbb{Z}_p^2$

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 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.
Unforgeability

From a valid ciphertext-signature pair:
\[ c = c_1 = X_1^{r_1}, c_2 = X_2^{r_2}, c_3 = g^{\alpha_1 + \beta_1} \cdot F \]
\[ \sigma = \sigma_1 = Y \cdot c_3^s, \sigma_2 = (c_1^s, c_2^s), \sigma_3 = (g^s, X_1^s, X_2^s) \]

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 \((M, \Sigma)\)
\[ \rightarrow \text{needs of a proof of knowledge } \Pi_M \text{ of } M \text{ in } F = F (M) \]
\[ \text{bit-by-bit commitment of } M \text{ and Groth-Sahai proof} \]

Chosen-Message Attacks

From a valid ciphertext \( c = c_1 = X_1^{r_1}, c_2 = X_2^{r_2}, c_3 = g^{\alpha_1 + \beta_1} \cdot F \)
\[ \text{and the additional proof of knowledge of } M, \]
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 \( c \)
\[ \rightarrow \text{needs of a proof of knowledge } \Pi_r \text{ of } r_1, r_2 \text{ in } c \]
\[ \text{bit-by-bit commitment of } r_1, r_2 \text{ and Groth-Sahai proof} \]

From the random coins \( r_1, r_2 \) (and the decryption key):
\[ \sigma = \sigma_1 = \Sigma_1 \cdot (\Sigma_2^{r_1} \Sigma_2^{r_2}), \sigma_2 = (\Sigma_2^{r_1}, \Sigma_2^{r_2}), \sigma_3 = (\Sigma_2, \Sigma_2, \Sigma_2) \]
\[ = Y \cdot c_3^s, \]
\[ = (c_1^s, c_2^s), \]
\[ = (g^s, X_1^s, X_2^s) \]
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 ($\ell$ bit-long)
- $r_1$ and $r_2$ are random in $\mathbb{Z}_p$

$\Rightarrow$ $C$ is large!

d David Pointcheval – 41/43

**Efficiency**

We can improve efficiency: with a variant of Waters Signature

$\Rightarrow$ shorter signatures: $9\ell + 33$ group elements

d David Pointcheval – 43/43

**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 $\mathbb{G}_1$

and $6\ell + 7$ group elements in $\mathbb{G}_2$