Efficient Two-Move Blind Signatures in the Common Reference String Model

E. Ghadafi  N.P. Smart

Department of Computer Science, University of Bristol

Information Security Conference – ISC 2012
Outline

1. Blind Signatures
2. Security Model
3. Related Work
4. Our Construction
5. Efficiency Comparison
6. Open Problems
OUTLINE

1. Blind Signatures
2. Security Model
3. Related Work
4. Our Construction
5. Efficiency Comparison
6. Open Problems
OUTLINE

1 Blind Signatures
2 Security Model
3 Related Work
4 Our Construction
5 Efficiency Comparison
6 Open Problems

Efficient Two-Move Blind Signatures
OUTLINE

1. Blind Signatures
2. Security Model
3. Related Work
4. Our Construction
5. Efficiency Comparison
6. Open Problems
OUTLINE

1. Blind Signatures
2. Security Model
3. Related Work
4. Our Construction
5. Efficiency Comparison
6. Open Problems
OUTLINE

1. Blind Signatures
2. Security Model
3. Related Work
4. Our Construction
5. Efficiency Comparison
6. Open Problems
(Two-Move) Blind Signatures

**User**

**Signer**

pk

sk
(Two-Move) Blind Signatures
(Two-Move) Blind Signatures
(Two-Move) Blind Signatures
Example applications:

- **E-Cash**: A bank signs a coin without learning its serial number (provides unlinkability between withdrawal and spend transactions).

- **E-Voting**: Authority certifies a ballot without learning its content. The client cannot vote for more than one candidate.

- **Many other applications** where anonymity/privacy or unlinkability are required (Anonymous Access Control, etc.).
ALGORITHMS OF A BLIND SIGNATURE

- **Setup**
  \[ \text{crs}_{BS} \leftarrow \text{Setup}_{BS}(1^\lambda) \]

- **Key Generation**
  \[ (\text{sk}_{BS}, \text{pk}_{BS}) \leftarrow \text{KeyGen}_{BS}(\text{crs}_{BS}) \]

- **Signing**
  \[ (\bot, \sigma) \leftarrow \langle \text{Request}_{BS}(\text{pk}_{BS}, m), \text{Issue}_{BS}(\text{sk}_{BS}) \rangle \]

- **Verification**
  \[ 1/0 \leftarrow \text{Verify}_{BS}(\text{pk}_{BS}, m, \sigma) \]
**Blindness [JLO97,PS00]:** The Signer does not learn what message he is signing nor can he link a signature to its sign request.

The adversary wins if $b^* = b$.

- **Malicious Keys [Oka06]:** The adversary generates the keys.
**Security of Blind Signatures**

- **Blindness [JLO97,PS00]:** The Signer does not learn what message he is signing nor can he link a signature to its sign request.

The adversary wins if $b^* = b$.

- **Malicious Keys [Oka06]:** The adversary generates the keys.
(Weak) Unforgeability [JLO97,PS00]: The User cannot output more signatures than the number of interactions with the signer.

\[
\text{pk}_{\text{BS}} \quad \downarrow \\
\text{Issue}_{\text{BS}}(\text{sk}_{\text{BS}}) \quad \text{(n times)} \\
(m_1,\sigma_1), \ldots, (m_{n+1},\sigma_{n+1}) \\
\]

The adversary wins if all \(\sigma_i\) verify and the messages are distinct.
Some previous two-move constructions:

- Fischlin 2006: generic construction (CRS).
- MSF 2010: using Waters signatures in composite-order groups (CRS).
We follow the Blind-Unblind paradigm ...

\[
\begin{align*}
\text{pk} & \rightarrow m \\
& \downarrow \quad m' \leftarrow \text{Blind}(m, r) \\
\text{USER} & \quad \downarrow \\
& \quad \sigma \leftarrow \text{Unblind}(\sigma', r) \\
\text{sk} & \leftarrow \sigma' \leftarrow \text{Sign}(\text{sk}, m')
\end{align*}
\]

However, we dispense with the need for random oracles by requiring a common reference string.
(Prime-Order) Bilinear Groups

$\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T$ are finite cyclic groups of prime order $q$, where $\mathbb{G}_1 = \langle P_1 \rangle$ and $\mathbb{G}_2 = \langle P_2 \rangle$.

**Pairing** ($e : \mathbb{G}_1 \times \mathbb{G}_2 \rightarrow \mathbb{G}_T$):

The function $e$ must have the following properties:

- **Bilinearity:** $\forall Q_1 \in \mathbb{G}_1, Q_2 \in \mathbb{G}_2, x, y \in \mathbb{Z}$, we have

  $$e([x]Q_1, [y]Q_2) = e(Q_1, Q_2)^{xy}.$$

- **Non-Degeneracy:** The value $e(P_1, P_2) \neq 1$ generates $\mathbb{G}_T$.

- **The function $e$ is efficiently computable.**

**Type-3 [GPS08]**: $\mathbb{G}_1 \neq \mathbb{G}_2$ and no efficiently computable isomorphism between $\mathbb{G}_1$ and $\mathbb{G}_2$. 
**Intractability Assumptions**

**Definition (LRSW Assumption [LRSW99])**

Given \((X \leftarrow [x]P_2, Y \leftarrow [y]P_2)\) and access to an oracle \(O_{X,Y}(\cdot)\) that, on input \(f_i \in \mathbb{Z}_q\) outputs \((A_i, B_i, C_i) \leftarrow (A_i, [y]A_i, [x + f_i \cdot x \cdot y]A_i)\), for some random \(A_i \in \mathbb{G}_1\), it is hard to output \((f^*, A^*, B^*, C^*)\) where \(f^* \notin \{f_i\} \cup \{0\}\).

**Definition (B-LRSW Assumption [CMS09])**

Given \((X \leftarrow [x]P_2, Y \leftarrow [y]P_2)\) and access to an oracle \(O^B_{X,Y}(\cdot)\) that, on input \(F_i = [f_i]P_1 \in \mathbb{G}_1\) outputs \((A_i, B_i, C_i) \leftarrow (A_i, [y]A_i, [x + f_i \cdot x \cdot y]A_i)\), for some random \(A_i \in \mathbb{G}_1\), it is hard to output \((f^*, A^*, B^*, C^*)\) where \([f^*]P_1 \notin \{F_i\} \cup \{0_{\mathbb{G}_1}\}\).
**Intractability Assumptions**

**Definition (LRSW Assumption [LRSW99])**

Given \( (X \leftarrow [x]P_2, Y \leftarrow [y]P_2) \) and access to an oracle \( \mathcal{O}_{X,Y}(\cdot) \) that, on input \( f_i \in \mathbb{Z}_q \) outputs \( (A_i, B_i, C_i) \leftarrow (A_i, [y]A_i, [x + f_i \cdot x \cdot y]A_i) \), for some random \( A_i \in \mathbb{G}_1 \), it is hard to output \( (f^*, A^*, B^*, C^*) \) where \( f^* \notin \{ f_i \} \cup \{ 0 \} \).

**Definition (B-LRSW Assumption [CMS09])**

Given \( (X \leftarrow [x]P_2, Y \leftarrow [y]P_2) \) and access to an oracle \( \mathcal{O}^B_{X,Y}(\cdot) \) that, on input \( F_i = [f_i]P_1 \in \mathbb{G}_1 \) outputs
\( (A_i, B_i, C_i) \leftarrow (A_i, [y]A_i, [x + f_i \cdot x \cdot y]A_i) \), for some random \( A_i \in \mathbb{G}_1 \), it is hard to output \( (f^*, A^*, B^*, C^*) \) where \( [f^*]P_1 \notin \{ F_i \} \cup \{ 0_{\mathbb{G}_1} \} \).
Intractability Assumptions

Definition (E-LRSA Assumption (holds in GGM))

Given $X \leftarrow [x]P_2$, $Y \leftarrow [y]P_2$, $Z \leftarrow [z]P_1$ and access to an oracle $O^E_{X,Y,Z}(\cdot)$ that on input $F_i = [f_i]P_1 \in \mathbb{G}_1$ outputs $(A_i, B_i, C_i, D_i) \leftarrow (A_i, [y]A_i, [x + f_i \cdot x \cdot y]A_i, [x \cdot y \cdot z]A_i)$, for some random $A_i \in \mathbb{G}_1$, it is hard to output $(f_i, A_i, B_i, C_i)_{i=1}^{n+1}$ where $f_i \neq 0$ are distinct after interacting with $O^E$ $n$ times.
Building Blocks

- **CL Signatures [CL04]**

  Given the description of bilinear groups $\mathcal{P} \leftarrow \text{Setup}_{\text{Grp}}(1^\lambda)$.

  - **KeyGen($\mathcal{P}$):** Select $x, y \leftarrow \mathbb{Z}_q$. Set $X \leftarrow [x]P_2$ and $Y \leftarrow [y]P_2$.
    
    $\text{sk} \leftarrow (x, y) \in \mathbb{Z}_p \times \mathbb{Z}_p$ and $\text{pk} \leftarrow (X, Y) \in \mathbb{G}_2^2$.

  - **Sign($\text{sk}, m$):** To sign a message $m \in \mathbb{Z}_q$, select $a \leftarrow \mathbb{Z}_q$, and set
    
    $A \leftarrow [a]P_1$, $B \leftarrow [y]A$, and $C \leftarrow [x + m \cdot x \cdot y]A$.

    Output $\sigma \leftarrow (A, B, C) \in \mathbb{G}_1^3$.

  - **Verify($\text{pk}, m, \sigma$):** Output 1 iff
    
    $$e(A, Y) = e(B, P_2) \quad \text{and} \quad e(C, P_2) = e(A, X)e(B, X)^m$$

  - **Existentially unforgeable** $\Rightarrow$ the LRSW assumption.
  - **Randomizable signatures:** To randomize a signature $\sigma$, select
    
    $t \leftarrow \mathbb{Z}_q$ and compute $\sigma' \leftarrow [t]\sigma$. 
Blog Signatures Security Model Related Work Our Construction Efficiency Comparison Open Problems

Building Blocks

- Pedersen Commitment [Ped91]

  - Setup: Let $\mathbb{G} = \langle P \rangle$ be a group of prime order $q$. Select $Q \leftarrow \mathbb{G}$. Set $pk \leftarrow (P, Q)$.

  - Commit($m$): To commit to a message $m \in \mathbb{Z}_q$, select $r \leftarrow \mathbb{Z}_q$, and set $C \leftarrow [m]P + [r]Q$.

  - Opening: to open a commitment $C$ just reveal $m$ and $r$. the correctness can be checked by verifying that $C = [m]P + [r]Q$.

- Security:
  - Information theoretically hiding.
  - Computationally binding $\Rightarrow$ DL assumption.
 OUR SCHEME

► The Idea:

1. The User sends a Pedersen commitment $C_0$ to his message to the signer.
2. The Signer issues an E-LRSW tuple $(A, B, C, D)$ on the commitment $C_0$.
3. Using the randomness used in $C_0$, the user recovers a CL signature $(A, B, C)$ on $m$ and re-randomizes it.

Security of the scheme:

Blindness $\Rightarrow$
- The perfect hiding property of Pedersen commitments.
- The re-randomizability of CL signatures.

Unforgeability $\Rightarrow$
- the hardness of the E-LRSW assumption.
Our Scheme

The Idea:

1. The User sends a Pedersen commitment $C_0$ to his message to the signer.
2. The Signer issues an E-LRSW tuple $(A, B, C, D)$ on the commitment $C_0$.
3. Using the randomness used in $C_0$, the user recovers a CL signature $(A, B, C)$ on $m$ and re-randomizes it.

Security of the scheme:

- **Blindness**
  1. The perfect hiding property of Pedersen commitments.
  2. The re-randomizability of CL signatures.
- **Unforgeability**
  the hardness of the E-LRSW assumption.
**Our Scheme**

Setup\(_{BS}(1^\lambda)\):
- \(\mathcal{P} \leftarrow \text{Setup}_{\text{Grp}}(1^\lambda)\).
- \(z \in \mathbb{Z}_q\).
- \(Z \leftarrow [z]P_1\).
- \(\mathcal{M} := \mathbb{Z}_q^\times\).
- \(\text{crs}_{BS} \leftarrow (\mathcal{P}, Z, \mathcal{M})\).
- Output \(\text{crs}_{BS}\).

Request\(_{BS}^0(m, \text{pk}_{BS})\):
- \(r \leftarrow \mathbb{Z}_q\).
- \(\text{Co} \leftarrow [m]P_1 + [r]Z\).
- \(\rho \leftarrow \text{Co}, \text{St} \leftarrow (m, r)\).
- Output \((\rho, \text{St})\).

Request\(_{BS}^1(\beta, \text{St}, \text{pk}_{BS})\):
- Parse \(\beta\) as \((A, B, C, D)\).
- Parse \(\text{St}\) as \((m, r)\).
- \(C \leftarrow C - [r]D\).
- If \(\text{Verify}_{BS}(m, (A, B, C), \text{pk}_{BS}) = 0\)
  - Return \(\bot\).
- \(t \leftarrow \mathbb{Z}_q\).
- \(A \leftarrow [t]A, B \leftarrow [t]B, C \leftarrow [t]C\).
- \(\sigma \leftarrow (A, B, C)\).
- Output \(\sigma\).

KeyGen\(_{BS}(\mathcal{P})\):
- \(x, y \leftarrow \mathbb{Z}_q\).
- \(X \leftarrow [x]P_2\).
- \(Y \leftarrow [y]P_2\).
- \(\text{sk}_{BS} \leftarrow (x, y), \text{pk}_{BS} \leftarrow (X, Y)\).
- Output \((\text{pk}_{BS}, \text{sk}_{BS})\).

Issue\(_{BS}(\rho, \text{sk}_{BS})\):
- Parse \(\rho\) as \(\text{Co}\).
- \(a \leftarrow \mathbb{Z}_q\).
- \(A \leftarrow [a]P_1\)
- \(B \leftarrow [a \cdot y]P_1\)
- \(C \leftarrow [a \cdot x]P_1 + [a \cdot x \cdot y]Co\)
- \(D \leftarrow [a \cdot x \cdot y]Z\).
- \(\beta \leftarrow (A, B, C, D)\).
- Output \(\beta\).

Verify\(_{BS}(m, \sigma, \text{pk}_{BS})\):
- Parse \(\sigma\) as \((A, B, C)\).
- If \(A = 0\) or \(e(A, Y) \neq e(B, P_2)\) or \(e(C, P_2) \neq e(A, X) \cdot e(B, X)^m\)
  - Return 0.
- Return 1.
Our Scheme

- **The Pros:**
  - Standard final signatures (i.e. we do not hide the signature).
  - Round-optimal signing protocol (i.e. two-move).
  - No proofs of knowledge used.
  - Short signatures of size $G_1^3$.
  - Very low communication overhead: user sends one element in $G_1$, whereas the signer sends $G_1^4$.
  - Short public key of size $G_2^2$.
  - Minimal reference string which is one element in $G_1$. 
**The Cons:**

- Blindness holds w.r.t. honestly generated keys.
  - Can be overcome by requiring the signer to prove knowledge of the secret key.

- The E-LRSW assumption is interactive and is thus unfalsifiable [Naor03].
### Efficiency Comparison

<table>
<thead>
<tr>
<th>Scheme</th>
<th>Signature</th>
<th>Communication User</th>
<th>Communication Signer</th>
<th>CRS</th>
<th>Key</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fuchsbauer09</td>
<td>$G_1^{18} \times G_2^{16}$</td>
<td>$G_1^{17} \times G_2^{16}$</td>
<td>$G_1^3 \times G_2^2$</td>
<td>$G_1^7 \times G_2^4$</td>
<td>$G_1 \times G_2$</td>
</tr>
<tr>
<td>AHO10</td>
<td>$G_1^{12} \times G_2^{14}$</td>
<td>$G_2^3$</td>
<td>$G_1^2 \times G_2^5$</td>
<td>$G_1^{10} \times G_2^5$</td>
<td>$G_1^4 \times G_2^7$</td>
</tr>
<tr>
<td>MSF10†</td>
<td>$G_2^2$</td>
<td>$G_2^2 \cdot |m|$</td>
<td>$G_2^3$</td>
<td>$G_2^2 \cdot |m| + 2$</td>
<td>$G_T$</td>
</tr>
<tr>
<td>Ours</td>
<td>$G_1^3$</td>
<td>$G_1^1$</td>
<td>$G_1^4$</td>
<td>$G_1^1$</td>
<td>$G_2^2$</td>
</tr>
</tbody>
</table>

† Uses composite-order groups. At 80-bit symmetric-key security, the size of elements of $G$ is 1024 bits compared to 128 and 256 bits for $G_1$ and $G_2$ in prime-order groups.
Open Problems

- Achieving blindness w.r.t. maliciously chosen keys without degrading the efficiency or increasing the number of rounds.
- Constructions with similar efficiency based on falsifiable (non-interactive) intractability assumptions.
The End.

Questions?