Outline

* Introduction
* Preliminaries
* The Proposed Scheme
  * The withdrawal protocol
  * The payment protocol
  * Anonymity control
* Security
  * Unforgeability
* Conclusion
Introduction

* Real life E-Cash example:
  * DigiCash: Payer anonymous system
  * Mondex: Smartcard-Based system, small value
* Our proposed scheme is aimed at the payer anonymous system
* The anonymity control is needed!
  * Privacy revoking
  * Coin tracing
Introduction

✱ Real life E-Cash example:
  ✴ DigiCash: Payer anonymous system
  ✴ Mondex: Smartcard-Based system, small value

✱ Our proposed scheme is aimed at the payer anonymous system

✱ The anonymity control is needed!
  ✴ Privacy revoking
  ✴ Coin tracing

http://www.mondex.com/
Introduction

* Real life E-Cash example:
  * DigiCash: Payer anonymous system
  * Mondex: Smartcard-Based system, small value
* Our proposed scheme is aimed at the payer anonymous system
* The anonymity control is needed!
  * Privacy revoking
  * Coin tracing

http://www.mondex.com/
Introduction

- Real life E-Cash example:
  - DigiCash: Payer anonymous system
  - Mondex: Smartcard-Based system, small value

- Our proposed scheme is aimed at the payer anonymous system

- The anonymity control is needed!
  - Privacy revoking
  - Coin tracing

http://www.mondex.com/
Introduction

* Real life E-Cash example:
  * DigiCash: Payer anonymous system
  * Mondex: Smartcard-Based system, small value

* Our proposed scheme is aimed at the payer anonymous system

* The anonymity control is needed!
  * Privacy revoking
  * Coin tracing

http://www.mondex.com/
Introduction

- Real life E-Cash example:
  - DigiCash: Payer anonymous system
  - Mondex: Smartcard-Based system, small value
- Our proposed scheme is aimed at the payer anonymous system
- The anonymity control is needed!
  - Privacy revoking
  - Coin tracing

http://www.mondex.com/
Preliminaries

- David Chaum’s blind signature
- Chameleon hash function
- Trusted Platform Module (TPM)
Preliminaries

David Chaum’s Blind Signature

\[ \alpha = a^e H(m) \mod n \] (Blinding)

\[ s = ta^{-1} \mod n \] (Unblinding)

\[ (s, m) \]

Verification:
\[ s^e \equiv H(m) \pmod{n} \]

\[ t = \alpha^d \mod n \] (Signing)

- \( n = pq \)
- \((e, n)\): public key
- \(d\): secret key
Krawczyk and Rabin’s scheme

- Collision Resistance
- Trapdoor Collisions: $h_{HK}(m, r) = h_{HK}(m', r')$

**Construction:**
- prime number $p$
- $q : p = 2q + 1$
- $g \in \mathbb{Z}_p^*$, $x \in \mathbb{Z}_q^*$, $y = g^x \mod p$
- Hash key: $(p, q, g, y)$
- Trapdoor key: $x$
- Hash function: $h_{HK}(m, r) = g^m y^r \mod p$
- Collision: $r' = m + rx - m'x \mod q$
  \[ g^m y^r \equiv g^m y^{r'} \pmod{p} \]
Krawczyk and Rabin’s scheme

- Collision Resistance
- Trapdoor Collisions: \( h_{HK}(m, r) = h_{HK}(m', r') \)

**Construction:**
- prime number \( p \)
- \( q : p = 2q + 1 \)
- \( g \in \mathbb{Z}_p^*, x \in \mathbb{Z}_q^*, y = g^x \mod p \)
- Hash key: \((p, q, g, y)\)
- Trapdoor key: \( x \)
- Hash function: \( h_{HK}(m, r) = g^m y^r \mod p \)
- Collision: \( r' = m + rx - m' x \mod q \)
  \[ g^m y^r \equiv g^{m'} y^{r'} \pmod{p} \]

**Trapdoor key reveal:**
- Given \((m', r'), (m'', r'')\)
  \[ h_{HK}(m, r) = h_{HK}(m'', r'') \]
  \[ \rightarrow g^{m'} y^{r'} \equiv g^{m''} y^{r''} \pmod{p} \]
  \[ \rightarrow g^{m' + xr'} \equiv g^{m'' + xr''} \pmod{p} \]
  \[ \rightarrow m' + xr' \equiv m'' + xr'' \pmod{q} \]
  \[ \rightarrow x = \frac{m' - m''}{r'' - r'} \pmod{q} \]
Preliminaries

TPM - Trusted Platform Module

- Tamper-Resistance Device

The Proposed Scheme

- High Level Description
- Initialization
- Withdrawal Protocol
- Payment Protocol
- Anonymity Control
  - Double-Spending (revoke the identity of the spender without TTP)
  - Revocability
  - Traceability
The Proposed Scheme

High Level Description

User \rightarrow \alpha \rightarrow Bank

User \leftarrow t \leftarrow Bank
The Proposed Scheme

High Level Description

Blinding

Unblinding

User

Bank

α

t

Signing

David Chaum’s Blind Signature

In 1983, Chaum proposed a blind signature scheme that contains one-way hash function and private keys. The algorithm will output the public and private keys.

\[ \text{Signing: } \text{The signer computes } s = \text{Chaum's blind signature scheme} \]

\[ \text{Unblinding: } \text{Then the user sends the signer } \alpha \text{ back to the user.} \]

\[ \text{Verifying: } \text{By checking if } m \equiv s \mod n \text{ is true or not.} \]
The Proposed Scheme
High Level Description

Blinding

Chameleon Hash Function
using user’s identity as trapdoor key

User

Bank

α

Unblinding

Signing

David Chaum’s Blind Signature

Friday, November 27, 2009
The Proposed Scheme

High Level Description

- $(\Sigma, y, m, r, \delta)$
- $\Sigma^e_b \equiv h_{HK}(m, r)H(\delta||y)$
The Proposed Scheme

High Level Description

\[ \Sigma, y, m, r, \delta \]
\[ \Sigma^{e_b} \equiv h_{HK}(m, r)H(\delta||y) \]

\[ h_{HK}(m, r) = h_{HK}(m', r') \]

Friday, November 27, 2009
The Proposed Scheme
High Level Description

- \( (\Sigma, y, m, r, \delta) \)
- \( \Sigma^b = h_{HK}(m, r) H(\delta || y) \)
- \( h_{HK}(m, r) = h_{HK}(m', r') \)
- \( h_{HK}(m, r) = h_{HK}(m'', r'') \)

- \( m' \)
- \( r' \)
- \( m'' \)
- \( r'' \)

- \( \alpha \)
- \( t \)

- \( \text{Double-Spending!} \)

- \( ID = \frac{m' - m''}{r'' - r'} \mod q \)
The Proposed Scheme

**Initialization**

Bank

\((n_b, e_b, p, q, g, H, d)\)

Judge

\((pk_j, sk_j)\)
The Proposed Scheme

Initialization

Bank

Judge

$(n_b, e_b, p, q, g, H, d)$
The Proposed Scheme

Withdrawal Protocol

Bank

User

Judge device

\( E_{pk-j}(k, m, r) \)

\( k \in_R \{0, 1\}^{l_k} \)

\( \mu = ID_u \)

\( (E_{pk-j}(k, m, r), \mu) \)

\( (\beta, E_k(x, c, k, \delta)) \)

\( (t, E_k(x, c, k, \delta)) \)

- Decrypt \( E_{pk-j}(k, m, r) \)
- \( r_1, r_2 \in_R \{0, 1\}^{l_r}, c \in_R Z_{n_b}^* \)
- \( x = (\mu||r_1) \in Z_q^* \)
- \( y = g^x \mod p, \delta = E_{pk-j}(\mu||r_2) \)
- \( \beta = c^{\mu} h_{HK}(m, r) H(\delta||y) \mod n_b \)
  = \( c^{\mu} (g^m y^r \mod p) H(\delta||y) \mod n_b \)

- Compute \( t = \beta^{d_b} \mod n_b \)

- Decrypt \( E_k(x, c, k, \delta) \)
- Parse the 3rd parameter as \( k' \)
- Check if \( k' = k \)
- \( \Sigma = c^{-1} t \mod n_b \)
- Check if \( \Sigma^{e_b} = h_{HK}(m, r) H(\delta||y) \mod n_b \) is true
- E-cash: \( (\Sigma, y, m, r, \delta) \)
The Proposed Scheme

Withdrawal Protocol

Judge device

Bank

User

Chaum’s Blind Signature

- Decrypt $E_{pk,j}(k, m, r)$
- $r_1, r_2 \in_R \{0, 1\}^{l_r}, c \in_R \mathbb{Z}_{n_b}^*$
- $x = (\mu || r_1) \in \mathbb{Z}_q^*$
- $y = g^x \mod p$, $\delta = E_{pk,j}(\mu || r_2)$
- $\beta = c^{eb} h_{HK}(m, r) H(\delta || y) \mod n_b$
  $= c^{eb}(g^{m^* y} \mod p) H(\delta || y) \mod n_b$

$(\beta, E_k(x, c, k, \delta))$

- Compute $t = \beta^{db} \mod n_b$

$(t, E_k(x, c, k, \delta))$

- Decrypt $E_k(x, c, k, \delta)$
- Parse the 3rd parameter as $k'$
- Check if $k' = k$
- $\Sigma = c^{-1} t \mod n_b$
- Check if $\Sigma^{eb} \equiv h_{HK}(m, r) H(\delta || y) \mod n_b$ is true
- E-cash: $(\Sigma, y, m, r, \delta)$
The Proposed Scheme

Withdrawal Protocol

Judge device

Bank

User

\( E_{pk,j}(k, m, r) \)

\( \mu = ID_u \)

\( k \in_R \{0, 1\}^l \)

Chameleon Hash Function

- Decrypt \( E_{pk,j}(k, m, r) \)
- \( r_1, r_2 \in_R \{0, 1\}^l, c \in_R \mathbb{Z}_{n_b}^* \)
- \( x = (\mu || r_1) \in \mathbb{Z}_q^* \)
- \( y = g^x \mod p, \delta = E_{pk,j}(\mu || r_2) \)
- \( \beta = c^{eb} h_{HK}(m, r) H(\delta || y) \mod n_b \)
  - \( = c^{eb} (g^m y^e \mod p) H(\delta || y) \mod n_b \)

\( (\beta, E_k(x, c, k, \delta)) \)

- Compute \( t = \beta^{eb} \mod n_b \)

\( (t, E_k(x, c, k, \delta)) \)

- Decrypt \( E_k(x, c, k, \delta) \)
- Parse the 3rd parameter as \( k' \)
- Check if \( k' = k \)
- \( \Sigma = c^{-1} t \mod n_b \)
- Check if \( \Sigma^{eb} = h_{HK}(m, r) H(\delta || y) \mod n_b \) is true
- E-cash: \( (\Sigma, y, m, r, \delta) \)
The Proposed Scheme

Payment Protocol

Bank

\( m' = ID_S || r_s \)

Shop

\( m' \)

\( (\Sigma, y, m', r', \delta) \)

User

\( r' = x^{-1}(m + xr - m') \mod q \)

\( (\Sigma, r', y, \delta) \)

- Check if \( \Sigma^{eb} \equiv h_{HK}(m', r')H(\delta||y) \mod n_b \) is true
- Store the e-cash \((\Sigma, y, m', r', \delta)\)
- Deposit the e-cash later

- Check if \( \Sigma^{eb} \equiv h_{HK}(m', r')H(\delta||y) \mod n_b \) is true
- Check if \((\Sigma, y, \delta)\) exists in database
- If not, store \((\Sigma, y, m', r', \delta)\) in the database
The Proposed Scheme

Payment Protocol

Bank

Shop

User

• $m' = ID_S || r_s$

• $r' = x^{-1}(m + xr - m') \mod q$

• Check if $\Sigma^{eb} \equiv h_{HK}(m', r') H(\delta || y)$ (mod $n_b$) is true

• Check if $(\Sigma, y, \delta)$ exists in database

• If not, store $(\Sigma, y, m', r', \delta)$ in the database

(\Sigma, y, m', r', \delta)

$\Sigma, r', y, \delta$

Chameleon

Hash Function
The Proposed Scheme
Anonymity Control - On Double-Spending

\[(\Sigma_1, y_1, m_1, r_1, \delta_1) \quad (\Sigma_2, y_2, m_2, r_2, \delta_2)\]

\[\Sigma_1 = \Sigma_2, \ y_1 = y_2, \ \delta_1 = \delta_2, \ (m_1, r_1) \neq (m_2, r_2)\]

\[x = \frac{m_1 - m_2}{r_2 - r_1} \mod q\]

\[x = (ID \| r_j)\]

- The bank can revoke the identity of the double spender by itself.
The Proposed Scheme

Anonymity Control - On Illegal Transaction

• When an e-cash has been reported to an illegal transaction the judge can revoke the anonymity directly.

E-Cash: \((\Sigma, y, m, r, \delta)\)

\[\delta = E_{pk_j}(ID || r'_j)\]
The Proposed Scheme

Anonymity Control - Traceability

Judge device

Bank

User

\( E_{pk} (k, m, r) \)

\( \mu = ID_u \)

\( (E_{pk} (k, m, r), \mu) \)

\( (\beta, E_k (x, c, k, \delta)) \)

\( (t, E_k (x, c, k, \delta)) \)

\( \bullet \) Decrypt \( E_{pk} (k, m, r) \)
\( \bullet \) \( r_1, r_2 \in R \{0, 1\}^{lr}, c \in R \mathbb{Z}_{n^b}^* \)
\( \bullet \) \( x = (\mu || r_1) \in \mathbb{Z}_q^* \)
\( \bullet \) \( y = g^x \mod p, \delta = E_{pk} (\mu || r_2) \)
\( \bullet \) \( \beta = c^{eb} h_{HK} (m, r) H(\delta || y) \mod n_b \)
\( = c^{eb} (g^m y^r \mod p) H(\delta || y) \mod n_b \)

\( (\beta, E_k (x, c, k, \delta)) \)

\( \bullet \) Compute \( t = \beta^{db} \mod n_b \)

\( (t, E_k (x, c, k, \delta)) \)

\( \bullet \) Decrypt \( E_k (x, c, k, \delta) \)
\( \bullet \) Parse the 3rd parameter as \( k' \)
\( \bullet \) Check if \( k' = k \)
\( \bullet \) \( \Sigma = c^{-1} t \mod n_b \)
\( \bullet \) Check if \( \Sigma^{eb} \equiv h_{HK} (m, r) H(\delta || y) \mod n_b \) is true
\( \bullet \) E-cash: \( (\Sigma, y, m, r, \delta) \)
The Proposed Scheme

Anonymity Control - Traceability

- Bank knows the identity of the user. The bank encrypts the ID of the user. The user then decrypts the encrypted ID to get the ID of the user.

- After receiving the encrypted data from the bank, the judge device can reveal k to the police.

- Then, the police can obtain δ and trace the user via δ.
Security

- Theorem
- Unforgeability Game
- Unforgeability Game Simulation
- One-More E-Cash Forgery -> One-More Signature Forgery
For any attacker $A$ forging an e-cash in the proposed $EDREC$ scheme, there exists a forger $F$ attacking Chaum’s blind signature scheme such that

$$Adv^EDREC_A(k) \leq Adv^RSA-OMF_F(k)$$

and the time-complexity of $F$ is polynomial in the time-complexity of $A$ where $Adv^EDREC_A(k)$ is the probability of $A$ forging an e-cash in the proposed scheme.
Security
Unforgeability Game

For any attacker $A$ forging an e-cash in the proposed EDREC scheme, there exists a forger $F$ attacking Chaum's blind signature scheme such that $\text{Adv}^{\text{EDREC}}_A(k) \leq \text{Adv}^{\text{RSA-OMF}}_F(k)$ and the time-complexity of $F$ is polynomial in the time-complexity of $A$ where $\text{Adv}^{\text{EDREC}}_A(k)$ is the probability of $A$ forging an e-cash in the proposed scheme.

\[
\text{E-Cash}(s) = \{(\Sigma_i, y_i, m_i, r_i, \delta_i) | 1 \leq i \leq \lambda \} \cup (\Sigma, y, m, r, \delta)
\]
E-Cash(s) = \{((\Sigma, y_i, m_i, r_i, \delta_i)|1 \leq i \leq \lambda} \cup (\Sigma, y, m, r, \delta)\}

\lambda + 1
Security
Unforgeability Game Simulation

$$E_{pk_j}(k_i, m_i, r_i)$$
$$t_i, E_{k_i}(x_i, c_i, k_i, \delta_i)$$

$E$-Cash$(s) = \{ \{ (\Sigma_i, y_i, m_i, r_i, \delta_i) | 1 \leq i \leq \lambda \} \cup (\Sigma, y, m, r, \delta) \}$
Security
Unforgeability Game Simulation

E-Cash(s) = \{\{(\Sigma_i, y_i, m_i, r_i, \delta_i)|1 \leq i \leq \lambda\} \cup (\Sigma, y, m, r, \delta)\}

\alpha = a^{eb} h_{HK}(m, r) \quad \beta = b^{eb} H(\delta||y)

\Sigma^{eb} \equiv h_{HK}(m, r)H(\delta||y) \pmod{n_b}
Security

One-More E-Cash Forgery → One-More Signature Forgery

\[
E\text{-Cash}(s) = \{(\Sigma_i, y_i, m_i, r_i, \delta_i) | 1 \leq i \leq \lambda \} \cup (\Sigma, y, m, r, \delta)
\]
Security
One-More E-Cash Forgery \rightarrow One-More Signature Forgery

\[ E-Cash(s) = \{(\Sigma_i, y_i, m_i, r_i, \delta_i) | 1 \leq i \leq \lambda \} \cup (\Sigma, y, m, r, \delta) \]

\[ \Sigma \equiv h_{HK}(m, r)^{d_b} H(\delta||y)^{d_b} \pmod{n_b} \]
Security

One-More E-Cash Forgery → One-More Signature Forgery

\[ \lambda + 1 \]

\[ 2\lambda + 1 \]

\[ \Sigma \equiv h_{HK}(x) \cdot q^b H(\delta||y)^{db} \pmod{n_b} \]
Comparisons

Features

<table>
<thead>
<tr>
<th></th>
<th>AC</th>
<th>nTS</th>
<th>RWT</th>
<th>TP</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Revokability</td>
<td>Traceability</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ours</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>[12]</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>[13]</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>[14]</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
</tbody>
</table>

**AC**: Anonymity Control  
**nTS**: non-TTP-Storing, **RWT**: Revokability without the help of TTP  
**TP**: Theoretical Proof on unlinkability and unforgeability

Computation Cost

<table>
<thead>
<tr>
<th></th>
<th>Revokability</th>
<th>Payment</th>
<th>Withdrawal</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>B</td>
<td>R</td>
<td>U</td>
</tr>
<tr>
<td>ours</td>
<td>1E</td>
<td>0%</td>
<td>&lt;1E</td>
</tr>
<tr>
<td>[12]</td>
<td>2E</td>
<td>≈ 50%</td>
<td>&lt;1E</td>
</tr>
<tr>
<td>[13]</td>
<td>3E</td>
<td>≈ 67%</td>
<td>22E</td>
</tr>
<tr>
<td>[14]</td>
<td>4E</td>
<td>≈ 75%</td>
<td>8E</td>
</tr>
</tbody>
</table>

**E**: Modular Exponentiation  
**U**: User, **B**: Bank, **S**: Shop  
**R**: Computation reduction percentage: \((1 - \frac{A}{C}) \times 100\%\) where \(A\) is the cost of our scheme and \(C\) is the cost of another scheme
Conclusion

- We have proposed an efficient off-line e-cash scheme.
  - Revoking on double spender without the participation of TTP.
- We also provided the formal proofs on Unlinkability and Unforgeability.
Thanks for your attention