Anonymous Proxy Signature with Restricted Traceability

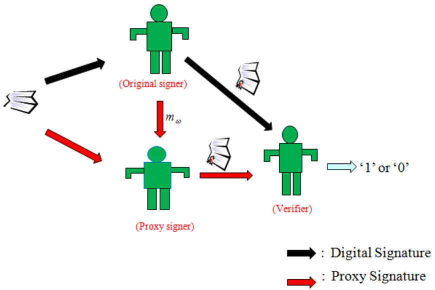
Jiannan Wei

Joined work with Guomin Yang and Yi Mu
University of Wollongong

Outline

- Introduction
- Motivation and Potential Solutions
- Anonymous Proxy Signature with Restricted Traceability
  - Formal Definition
  - Security Requirement
  - Our APSRT Construction
  - Adversary Type
  - Security Model
  - Security Proof
- Conclusion

Introduction



Proxy Signature: When the original signer was not available or leave and he/she will delegate the signing rights to proxy signature.

- Original signer
- Proxy signature
- Verifier

UNIVERSITY OF WOLLONGONG

Motivation

- Signer anonymity: user privacy in many applications
- We study signer anonymity for proxy signature which allows a signer delegate the signing right to another signer, since the proxy signer is the actual signer, we are interested in protecting the proxy signer's identity.

UNIVERSITY OF WOLLONGONG

Ring and Group Signature

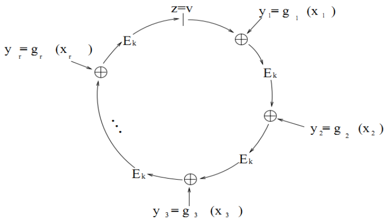


Figure: Ring signature

- Group signature: the group manager is able to reveal the identity of the signer for any valid group signature.
- Ring signature: any ring members can sign messages on behalf of the whole ring without reveal the identity of his/her real identity.

Potential Solutions

In the proxy signature, traceability is an very important property, since the proxy signer may abuse the signing right.

- Combine a group signature with the proxy signature
  - Problem: The group manager has too strong a tracebility that can trace any signature include the signature generated by honest signer.
- Use a ring signature combine with the proxy signature
  - Problem: It has nothing to restrict anonymity and is vulnerable to malicious signers.

Traceable Ring Signature

Traceable Ring Signature: A ring signature with a “gentle” anonymity restriction, which consider “one-more unforgeability” and “double-spending traceability”.

- Unforgeability
- Anonymity
- Publicly Traceability: dishonest signer can be publicly traced by anyone.
- Exculpability: it is against the framing attack, i.e. an honest user cannot be framed by other users in the system.

UNIVERSITY OF WOLLONGONG

Potential Solution

Combine the proxy signature with traceability ring signature

- Advantage: Traceability ring signature is a tag-based signature, a signer can sign messages only once per tag.
- Problem: If the ring members sign messages twice with the same tag, his identity can be *publicly* traced.

Contribution

Our anonymous proxy signature with *restricted traceability*

- Allows the original signer to trace the dishonest signer and at the same time protect the identity of the honest signer even against the original signer.

Formal Definition

APSTR signature scheme consists following algorithms.

- **Parameter generation**($\mathcal{PG}$): $Param \leftarrow \mathcal{PG}(\kappa)$.
- **Key generation**($\mathcal{KG}$): $(Y, x) \leftarrow \mathcal{KG}(Param)$.
- **Delegation signing**($\mathcal{DS}$): $\sigma_0 \leftarrow \mathcal{DS}(m_\omega, x_0)$
- **Delegation verification**($\mathcal{DV}$): $0/1 \leftarrow \mathcal{DV}(Y_0, m_\omega, \sigma_0)$
- **APS generation**($\mathcal{PS}$): $\sigma \leftarrow \mathcal{PS}(m, Y_0, L = (issue, Y_N), x_i, \sigma_0)$, where $Y_N = \{Y_1, \cdots, Y_n\}$.
- **APS verification**($\mathcal{PV}$): $0/1 \leftarrow \mathcal{PV}(m, Y_0, \sigma, L, m_\omega)$
- **Tracing**($\mathcal{TR}$): $indep/linked/Y_i \in Y_N \leftarrow \mathcal{TR}((m, \sigma), (m', \sigma'), L, m_\omega, \sigma_0)$

Security requirement

1. Unforgeability
2. Anonymity
3. Restricted Traceability: If the proxy signer is dishonest, no outsider is able to link signatures generated by the dishonest proxy signer.
4. Exculpability

Anonymous Proxy Signature with Restricted Traceability(APSRT):

General idea:

- Challenge problem: develop new techniques that could disallow outsiders to perform the trace operation.
  - We try to randomize each proxy signature so that only the original signer or another proxy signer who has also been delegated the same signing right has the secret to de-randomize the proxy signature.
- Our APSRT which can guarantee the anonymity against original signer and the restricted traceability against outsider.

Our Construction of APSRT(1)

- **Parameter generation.** Taking κ as input, outputs (G, q, P), G is a cyclic group of prime order q and P is a generator of G. Let $H_0 : \{0,1\} * G \to \mathbb{Z}_q$, $H : \{0,1\}^* \to G$, $H' : \{0,1\}^* \to G$ and $H'' : \{0,1\}^* \to \mathbb{Z}_q$. $Param = (G, q, P, H_0, H, H', H'')$.
- **Key generation.** User i randomly selects $x_i \in \mathbb{Z}_q$ and computes $Y_i = x_i P$. (x_i, Y_i) is the key pair of the user.
- **Delegation sign.** The original signer first generates a warrant m_ω. Original signer randomly chooses $\alpha \in \mathbb{Z}_q$ and computes $W_o = \alpha P$.

UNIVERSITY OF WOLLONGONG

Our Construction of APSRT(2)

- **Delegation sign.** Proxy signer picks another random $r \in \mathbb{Z}_q$ and computes $R = rP$, $s = r + x_0 H_0(m_\omega, R, W_o) \ mod \ q$. Finally, the proxy signer sends (m_ω, α, R, s) to all the proxy signers in $\mathcal{U} = \{u_1, u_2, \ldots, u_n\}$ via a secure channel.
- **Delegation verification.** Upon receiving (m_ω, α, R, s), the proxy signer u_i checks if $sP = R + H_0(m_\omega, R, W_o = \alpha P)Y_0$. If it holds, the proxy signer u_i computes his/her proxy signing secret key $psk_i = s + x_i H_0(m_\omega, R, W_o) = r + H_0(m_\omega, R, W_o)(x_0 + x_i) \ mod \ q$. For simplicity, let $pk_i = psk_i P = R + H_0(m_\omega, R, W_o)(Y_0 + Y_i)$ denote the corresponding proxy signing public key.
- **Proxy Sign** To sign $m \in \{0, 1\}^*$ with respect to a tag $L = (issue, Y_N)$ where Y_N are public keys of some proxy signers described in the the warrant m_ω, the real proxy signer u_i proceeds as follows:

UNIVERSITY OF WOLLONGONG

Our Construction of APSRT(3)

- **Proxy Sign**
  - Randomly choose $\beta \in \mathbb{Z}_q$, compute $F = H(L)$, $W_p = (W_{p1}, W_{p2}) = (\alpha\beta P, \beta F)$ and $\sigma_i = \alpha\beta F + psk_i F = (\alpha\beta + psk_i)F$.
  - Set $A_0 = H'(L, m)$ and $A_1 = \frac{1}{i}(\sigma_i - A_0)$.
  - For all $j \neq i$, compute $\sigma_j = A_0 + jA_1 \in G$. Note that every $(j, \log_F(\sigma_j))$ is on the line defined by $(0, \log_F(A_0))$ and $(i, psk_i + \alpha\beta)$.
  - Generate (c_N, z_N) based on a (non-interactive) zero-knowledge proof of knowledge for the language

    $$\mathcal{L} = \{(L, F, \sigma_N) \mid \exists i \in N \ s.t. \log_P(pk_i') = \log_F(\sigma_i)\}$$

    where $\sigma_N = (\sigma_1, \sigma_2, \ldots, \sigma_n)$ and $pk_i' = W_{p1} + pk_i$ as follows:

Our Construction of APSRT(4)

- **Proxy Sign**
  - Generate (c_N, z_N) based on a NIZK proof of knowledge for the language $\mathcal{L}$ as follows:
    1. Pick random $\omega_i \leftarrow \mathbb{Z}_q$ and set $a_i = \omega_i P$, $b_i = \omega_i F \in G$.
    2. Pick random $z_j, c_j \leftarrow \mathbb{Z}_q$, and set $a_j = z_j P + c_j pk'_j$, $b_j = z_j F + c_j \sigma_j \in G$ for every $j \neq i$.
    3. Set $c = H''(L, m, A_0, A_1, a_N, b_N)$ where $a_N = (a_1, \ldots, a_n)$ and $b_N = (b_1, \ldots, b_n)$.
    4. Set $c_i = c - \Sigma_{j \neq i} c_j \ mod \ q$ and $z_i = \omega_i - c_i(\alpha\beta + psk_i) \ mod \ q$.
    5. Return (c_N, z_N), where $c_N = (c_1, \ldots, c_n)$ and $z_N = (z_1, \ldots, z_n)$, as a proof for $\mathcal{L}$.

Our Construction of APSRT(5)

- **Proxy Sign**
  - Perform another (non-interactive) zero-knowledge proof of knowledge for

$$\mathcal{L}' = \{(F, W_{p2}, W_o, W_{p1}) \mid \log_{W_o} W_{p1} = \log_F W_{p2}\}$$

    as follows

    1. Pick random $\omega \leftarrow \mathbb{Z}_p$ and set $\tilde{a} = \omega W_o$, $\tilde{b} = \omega F \in G$.
    2. Set $\tilde{c} = H''(L, m, A_0, A_1, \tilde{a}, \tilde{b})$.
    3. Set $\tilde{z} = \omega - \tilde{c}\beta$.
    4. Return $(\tilde{c}, \tilde{z})$ as a proof for $\mathcal{L}'$.
  - Return $\sigma = (A_1, R, W_o, W_p, c_N, z_N, \tilde{c}, \tilde{z})$ as the signature on (L, m).

Our Construction of APSRT(6)

- **Verification** To verify a proxy signature $\sigma = (A_1, R, W_o, W_p, c_N, z_N, \tilde{c}, \tilde{z})$ on message m and tag L, check the following:
  1. Parse L as $(issue, Y_N)$, and compute $pk'_i = W_{p1} + pk_i = W_{p1} + R + H_0(m_\omega, R, W_o)(Y_0 + Y_i)$ for all $i \in N$.
  2. Set $F = H(L)$ and $A_0 = H'(L, m)$, and compute $\sigma_i = A_0 + iA_1 \in G$ for all $i \in N$.
  3. Compute $a_i = z_i P + c_i pk'_i$, $b_i = z_i F + c_i \sigma_i$, for all $i \in N$.
  4. Check that $H''(L, m, A_0, A_1, a_N, b_N) = \Sigma_{i \in N} c_i \ mod \ q$, where $a_N = (a_1, \ldots, a_n)$ and $b_N = (b_1, \ldots, b_n)$.
  5. Compute $\tilde{a} = \tilde{z} W_o + \tilde{c} W_{p1}$, $\tilde{b} = \tilde{z} F + \tilde{c} W_{p2}$.
  6. Check if $H''(L, m, A_0, A_1, \tilde{a}, \tilde{b}) = \tilde{c}$.
  7. If all the above checks are successful, outputs accept; otherwise, outputs reject.

Our Construction of APSRT(7)

- **Tracing** To check the relation between (m, σ) and (m', σ') under the same warrant m_ω and the same tag L where $\sigma = (A_1, R, W_o, W_p, c_N, z_N, \tilde{c}, \tilde{z})$ and $\sigma' = (A'_1, R, W_o, W'_p, c'_N, z'_N, \tilde{c}', \tilde{z}')$, check the following:
  1. Parse L as $(issue, Y_N)$. Set $F = H(L)$ and $A_0 = H'(L, m)$. Compute $\sigma_i = A_0 + iA_1 \in G$ for all $i \in N$. Since $W_p = (\alpha\beta P, \beta F)$, with the secret α, the original signer or any proxy signer specified in the warrant m_ω can compute $\widehat{\sigma_i} = \sigma_i - \alpha\beta F = psk_i F \in G$ for all $i \in N$. Do the same operation for σ' to get $\widehat{\sigma'_i}$ for all $i \in N$.
  2. For all $i \in N$, if $\widehat{\sigma_i} = \widehat{\sigma'_i}$, store pk_i in **TList**, where **TList** is initially empty.
  3. Output pk if pk is the only entry in **TList**; "linked" if **TList** $= Y_N$; "indep" otherwise.

Our Construction of APSRT(8)

Correcness:

$$
\begin{aligned}
&z_iP + c_ipk_i' \\
=&\Big[\omega_i - c_i\big[\alpha\beta + (r + H_0(m_\omega, R, W_o)(x_0 + x_i))\big]\Big]P + c_ipk_i' \\
=&\omega_iP - c_i\big[\alpha\beta + r + H_0(m_\omega, R, W_o)(x_0 + x_i)\big]P + \\
&c_i[\alpha\beta P + R + H_0(m_\omega, R, W_o)(Y_0 + Y_i)] \\
=&\omega_iP \\
=&a_i
\end{aligned}
$$

$$
\begin{aligned}
&z_iF + c_i\sigma_i \\
=&\Big[\omega_i - c_i\big[\alpha\beta + (r + H_0(m_\omega, R, W_o)(x_0 + x_i))\big]\Big]F \\
&+ c_i(\alpha\beta F + psk_iF) \\
=&\omega_iF - c_i\big(\alpha\beta + (r + H_0(m_\omega, R, W_o)(x_0 + x_i))\big)F + c_i\alpha\beta F \\
&+ c_iF(r + H_0(m_\omega, R, W_o)(x_0 + x_i)) \\
=&\omega_iF \\
=&b_i
\end{aligned}
$$

Our Construction of APSRT(9)

Correcness:

$$\tilde{z}W_o + \tilde{c}W_{p1}$$
$$= (\omega - \tilde{c}\beta)\alpha P + \tilde{c}\alpha\beta P$$
$$= \omega\alpha P$$
$$= \tilde{a}$$

$$\tilde{z}F + \tilde{c}W_{p2}$$
$$= (\omega - \tilde{c}\beta)F + \tilde{c}\beta F$$
$$= \omega F$$
$$= \tilde{b}$$

Adversary Type

- Type I: Outsider. $(Y_0, Y_1, \ldots Y_n)$
- Type II: Adversary is a proxy signer. $(Y_0, Y_1, \ldots Y_n, x_1, \ldots x_n)$
- Type III: Adversary is the original signer. $(Y_0, Y_1, \ldots Y_n, x_0)$

UNIVERSITY OF WOLLONGONG

Security Model

- Unforgeability
  - Unforgeability against type II adversary
  - Unforgeability against type III adversary
- Restricted Traceability
  - Tag-linkability
  - Untraceability against outsider
- Anonymity against original signer
- Exculpability

UNIVERSITY OF WOLLONGONG

Unforgeability against proxy signer

- Setup: $\mathcal{C}$ runs the algorithm to obtain the secret key and public key pairs (x_0, Y_0), (x_1, Y_1), ..., (x_n, Y_n) of the original signer and n proxy signers. $\mathcal{C}$ then sends $(Y_0, Y_1, \ldots, Y_n, x_{i_k})$ $(i_k) \in \{1, \ldots, n\}$ to the adversary $\mathcal{A}_{II}$.
- Delegation signing query: $\mathcal{A}_{II}$ can request a signature on a warrant he chooses. In response, $\mathcal{C}$ outputs a signature σ on m_ω.
- Output: Finally, $\mathcal{A}_{II}$ outputs a target warrant m_ω^* and σ^* such that
  - σ^* is a valid delegation signature on m_ω^*.
  - m_ω^* has never been requested in delegation signature queries.

UNIVERSITY OF WOLLONGONG

Untraceability against outsider

- Setup: $\mathcal{C}$ runs the algorithm to obtain the (x_1, Y_1), ..., (x_n, Y_n) representing the keys of n proxy signers, which will be send to adversary $\mathcal{A}$.
- Key selection: The adversary $\mathcal{A}$ outputs (Y_i, Y_j) as the two target proxy signer's public keys, let $b \in \{i, j\}$ be a random hidden bit. $\mathcal{A}$ sends the (Y_i, Y_j) to $\mathcal{C}$.
- Proxy signing query: $\mathcal{A}$ may access 3 signing oracles: Sig_{psk_b}, Sig_{psk_i}, Sig_{psk_j} for the warrant m_ω and the tag, where
  - Sig_{psk_b} is the signing oracle with respect to proxy signer $b(b \in \{i, j\})$ who has a valid proxy signing key psk_b;
  - Sig_{psk_i}(resp. Sig_{psk_j}) is the signing oracle with respect to proxy signer i who has a valid proxy signing key psk_i.
- Output: Finally, $\mathcal{A}$ outputs a bit b', $\mathcal{A}$ wins the game if $b' = b$.

UNIVERSITY OF WOLLONGONG

Theorems

Theorem

If there exists a type II adversary $\mathcal{A}_{II}$ which can break the proposed APSRT scheme, then we can construct another adversary $\mathcal{B}$ who can use $\mathcal{A}_{II}$ to solve DL problem.

Theorem

If there exists an adversary $\mathcal{D}$ who can correctly guess b with an non-negligible advantage ϵ, we can construct another algorithm $\mathcal{B}$ that can solve DDH problem.

Security Proof of Unforgeability 1

Proof. Given (P, x^*P) for some unknown $x^* \in \mathbb{Z}_p$ as an instance of DL problem. $\mathcal{B}$ can solve the DL problem with the help of $\mathcal{A}_{\mathcal{II}}$. $\mathcal{B}$ sets original signer's public key $Y_0 = Y^* = x^*P$, and generate the keys for proxy signers honestly. Then $\mathcal{B}$ sends $(Y_0, Y_1, \ldots Y_n, x_1, \ldots x_n)$ to adversary $\mathcal{A}$.

H_0 **hash query:** $\mathcal{A}_{\mathcal{II}}$ send the query (m_ω, R, W_o), $\mathcal{B}$ will check the H_0 list.

- If query tuple $((m_\omega, R, W_o), h_i)$ in the H_o list, $\mathcal{B}$ returns h_i to $\mathcal{A}_{\mathcal{II}}$.
- Otherwise, $\mathcal{B}$ choose a random number $h_i \in \mathbb{Z}_p$. Add $((m_\omega, R, W_o), h_i)$ to H_o list, return h_i to $\mathcal{A}_{\mathcal{II}}$.

Security Proof of Unforgeability 2

Delegation signing queries: $\mathcal{A}_{\mathcal{II}}$ send a query of $m_{\omega i}$, $\mathcal{B}$ performs the following:

- Randomly choose $c, s, \alpha \in \mathbb{Z}_q^*$ and compute $R = sP - cY^*$.
- Set $W_o = \alpha P$, $H_0 = (m_\omega, R, W_o) = c$ and store $((m_\omega, R, W_o), c)$ into the hash list H_0.
- Return $\sigma_0 = (\alpha, R, s)$ as the delegation signing key for m_ω.

Output: $\mathcal{A}$ output $\sigma_0 = (\alpha^*, R^*, s^*)$ which is a valid delegation signing key for warrant m_ω^*. m_ω^* should not have been queried before. Forking lemma: rewinding $\mathcal{A}_{\mathcal{II}}$, $\mathcal{B}$ can obtain $s_1^* = r + c_1^* x_0^*$, $s_2^* = r + c_2^* x_0^*$. c_1^* and c_2^* are two hash outputs of H_0. $\mathcal{B}$ can output

$$x_0^* = \frac{s_1^* - s_2^*}{c_1^* - c_2^*} \; mod \; q$$

as the value of x^* and the solution of DL problem.

UNIVERSITY OF WOLLONGONG

Security Proof of Untraceability 1

Proof. If $\mathcal{D}$ can correctly guess b with ϵ, we can construct $\mathcal{B}$ who can solve DDH problem.
Setup: $\mathcal{B}$ generate all the public and private keys by running the key generation algorithm. $\mathcal{B}$ sends all the public keys to the adversary $\mathcal{D}$.
Key selection: $\mathcal{D}$ outputs a m_ω, a tag L and two target proxy signer's public keys (Y_i, Y_j), $i, j \in N$. $\mathcal{B}$ then sets $W_o = aP$ and $F = H(L) = bP$, randomly selects $r \in \mathbb{Z}_q$ and computes $R = rP$ and $s = r + x_0 H_0(m_\omega, R, W_o)$. $\mathcal{B}$ also randomly selects $b \in \{i, j\}$, and answer $\mathcal{D}$'s queries as follows.

UNIVERSITY OF WOLLONGONG

Security Proof of Untraceability 2

Hash queries: All the hash queries made by $\mathcal{D}$ are answered as in the previous proof where $\mathcal{B}$ maintains a hash table for each hash oracle.
Proxy signing queries: When $\mathcal{D}$ makes a proxy signing query to Sig_{psk_i} on message m, $\mathcal{B}$ randomly selects $\beta \in \mathbb{Z}_q$, and computes $W_p = (\beta W_o, \beta F)$ and $\sigma_i = \beta z P + psk_i F$. β generate A_0, A_1 and $\sigma_j (j \neq i)$ by following the proxy signing algorithm. $\mathcal{B}$ also simulates the NIZK proof for language $\mathcal{L}$ using the following simulator.

Security Proof of Untraceability 3

NIZK Simulator:

1. For all $i \in N$, uniformly pick up at random $z_i, c_i \in \mathbb{Z}_p$, and compute $a_i = z_i P + c_i pk'_i$, $b_i = z_i F + c_i \sigma_i \in G$.
2. Set $H''(L, m, A_0, A_1, a_N, b_N)$ as c:$= \sum_{i \in N} c_i$ where $a_N = (a_1, a_2, \ldots, a_n)$, $b_N = (b_1, b_2, \ldots, b_N)$.
3. Output (c_N, z_N), where $c_N = (c_1, c_2, \ldots, c_N)$ and $z_N = (z_1, z_2, \ldots, z_N)$.

$\mathcal{B}$ also simulates the NIZK proof $(\tilde{c}, \tilde{z})$ for language $\mathcal{L}'$ honestly using the knowledge of β. Finally, $\mathcal{B}$ returns $\sigma = (A_1, R, W_o, W_p, c_N, z_N, \tilde{c}, \tilde{z})$ to $\mathcal{D}$.

Output: Finally, $\mathcal{D}$ outputs b'. If $b' = b$, $\mathcal{B}$ outputs 1; Otherwise, $\mathcal{B}$ outputs 0.

Conclusion

- We put forward to the notion of APSRT.
- We proposed a new concrete APSRT scheme which ensure the requirement of anonymity against the original signer, restricted untraceability against outsider.
- We also provided formal security proof to demonstrate that our APSRT is provable secure.

Thanks.
Any questions?