Lattice-Based Ring Signature Scheme

Carlos Aguilar Melchor[1] Slim Bettaieb [1]
Xavier Boyen[2] Laurent Fousse[3] Philippe Gaborit[1]

[1]XLIM-DMI, Université de Limoges, (France)

[2]Palo Alto Research Center, (États-Unis)

[3]Google Inc, (État-Unis)

Journées Codage et Cryptographie - 12th October 2012

Outline

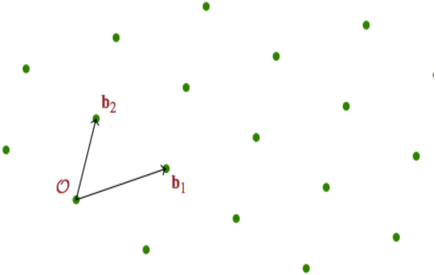
Lattices

Lattice

Let $b_1, b_2, ..., b_n \in \mathbb{R}^n$, n linearly independent vectors, the lattice generated by them is

$$\mathcal{L} := \{\sum_i x_i b_i \mid x_i \in \mathbb{Z}\}.$$

A lattice in $\mathbb{R}^2$



Lattice-based Cryptography

- 90s: Strong security reductions
- Belived to resist quantum computer attacks

Efficient Lattices

Let $\mathcal{R} = \mathbb{Z}[x] / \langle x^n + 1 \rangle$, with n a power of 2.

Ideal Lattice

- Let $\mathcal{I} \subseteq \mathcal{R}$ be an ideal.
  - Polynomials in $\mathcal{I}$ can be seen as vectors.
  - $\mathcal{I}$ corresponds to a sublattice of $\mathbb{Z}^n$

An ideal lattice is a sublattice of $\mathbb{Z}^n$ that correspond to an ideal $\mathcal{I} \subseteq \mathcal{R}$.

Properties

- More efficient (SWIFFT a candidate to NIST SHA-3 competition)
- Security reduction still holds.

Group Signatures

- Introduced by Chaum and Van Heyst in 1991.
- A group manager
- There is an anonymity revocation mechanism

Ring Signatures

- Introduced by Rivest, Shamir, and Tauman in 2001.
- Allow to leak a secret anonymously
- A user sign a message on behalf of a set of members (that include himself)
- [Ad-hoc] The signer can choose any ring and sign messages without the permission or asistance of its members
- [Anonymity] The signature gives a proof that the message was signed by a member of some entity, but it does not give any information about the real signer.

Group Signatures

- Introduced by Chaum and Van Heyst in 1991.
- A group manager
- There is an anonymity revocation mechanism

Ring Signatures

- Introduced by Rivest, Shamir, and Tauman in 2001.
- Allow to leak a secret anonymously
- A user sign a message on behalf of a set of members (that include himself)
- [Ad-hoc] The signer can choose any ring and sign messages without the permission or asistance of its members
- [Anonymity] The signature gives a proof that the message was signed by a member of some entity, but it does not give any information about the real signer.

Group Signatures

- Introduced by Chaum and Van Heyst in 1991.
- A group manager
- There is an anonymity revocation mechanism

Ring Signatures

- Introduced by Rivest, Shamir, and Tauman in 2001.
- Allow to leak a secret anonymously
- A user sign a message on behalf of a set of members (that include himself)
- [Ad-hoc] The signer can choose any ring and sign messages without the permission or asistance of its members
- [Anonymity] The signature gives a proof that the message was signed by a member of some entity, but it does not give any information about the real signer.

Group Signatures

- Introduced by Chaum and Van Heyst in 1991.
- A group manager
- There is an anonymity revocation mechanism

Ring Signatures

- Introduced by Rivest, Shamir, and Tauman in 2001.
- Allow to leak a secret anonymously
- A user sign a message on behalf of a set of members (that include himself)
- [Ad-hoc] The signer can choose any ring and sign messages without the permission or asistance of its members
- [Anonymity] The signature gives a proof that the message was signed by a member of some entity, but it does not give any information about the real signer.

Group Signatures

- Introduced by Chaum and Van Heyst in 1991.
- A group manager
- There is an anonymity revocation mechanism

Ring Signatures

- Introduced by Rivest, Shamir, and Tauman in 2001.
- Allow to leak a secret anonymously
- A user sign a message on behalf of a set of members (that include himself)
- [Ad-hoc] The signer can choose any ring and sign messages without the permission or asistance of its members
- [Anonymity] The signature gives a proof that the message was signed by a member of some entity, but it does not give any information about the real signer.

Group Signatures

- Introduced by Chaum and Van Heyst in 1991.
- A group manager
- There is an anonymity revocation mechanism

Ring Signatures

- Introduced by Rivest, Shamir, and Tauman in 2001.
- Allow to leak a secret anonymously
- A user sign a message on behalf of a set of members (that include himself)
- [Ad-hoc] The signer can choose any ring and sign messages without the permission or asistance of its members
- [Anonymity] The signature gives a proof that the message was signed by a member of some entity, but it does not give any information about the real signer.

Group Signatures

- Introduced by Chaum and Van Heyst in 1991.
- A group manager
- There is an anonymity revocation mechanism

Ring Signatures

- Introduced by Rivest, Shamir, and Tauman in 2001.
- Allow to leak a secret anonymously
- A user sign a message on behalf of a set of members (that include himself)
- [Ad-hoc] The signer can choose any ring and sign messages without the permission or asistance of its members
- [Anonymity] The signature gives a proof that the message was signed by a member of some entity, but it does not give any information about the real signer.

Group Signatures

- Introduced by Chaum and Van Heyst in 1991.
- A group manager
- There is an anonymity revocation mechanism

Ring Signatures

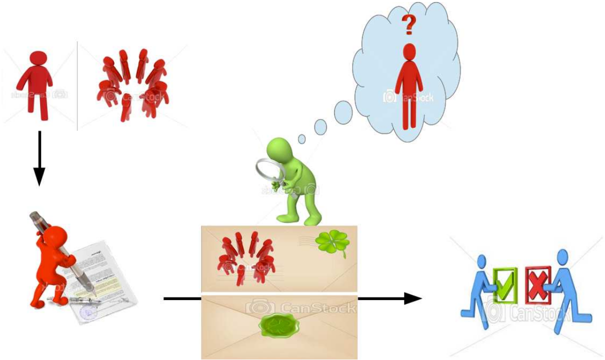
- Introduced by Rivest, Shamir, and Tauman in 2001.
- Allow to leak a secret anonymously
- A user sign a message on behalf of a set of members (that include himself)
- [Ad-hoc] The signer can choose any ring and sign messages without the permission or asistance of its members
- [Anonymity] The signature gives a proof that the message was signed by a member of some entity, but it does not give any information about the real signer.

Group Signatures

- Introduced by Chaum and Van Heyst in 1991.
- A group manager
- There is an anonymity revocation mechanism

Ring Signatures

- Introduced by Rivest, Shamir, and Tauman in 2001.
- Allow to leak a secret anonymously
- A user sign a message on behalf of a set of members (that include himself)
- [Ad-hoc] The signer can choose any ring and sign messages without the permission or asistance of its members
- [Anonymity] The signature gives a proof that the message was signed by a member of some entity, but it does not give any information about the real signer.



Applications

- Allow to leak a secret anonymously
- Designated-verifier signatures

Security Model

Unforgeability property

- Infeasibility of signing on behalf of a ring without knowing one of the secret keys

Anonymity of the signer

- It is not possible to know which secret key was used

Strong Security Model (Unforgeability)

Unforgeability w.r.t insider corruption

- Signing query can be done with respect to any ring
- Attacker is allowed to choose any subring to corrupt (i.e, the attacker can see the secret keys of this subring).

Unforgeability Game

1. Forger is given
   - $S = \{pk_i\}_{i=1}^{\ell}$
   - access to OSign()
   - access to a set of secret keys denoted C.
2. The forger outputs $(\sigma^\star, \mu^\star, R^\star)$ and succeeds if
   - the forger never queried $(\cdot, \mu^\star, R^\star)$ to OSign().
   - $R^\star \subseteq S \setminus C$

Strong Security Model (Unforgeability)

Unforgeability w.r.t insider corruption

- Signing query can be done with respect to any ring
- Attacker is allowed to choose any subring to corrupt (i.e, the attacker can see the secret keys of this subring).

Unforgeability Game

1. Forger is given
   - $S = \{pk_i\}_{i=1}^{\ell}$
   - access to OSign()
   - access to a set of secret keys denoted C.
2. The forger outputs $(\sigma^\star, \mu^\star, R^\star)$ and succeeds if
   - the forger never queried $(\cdot, \mu^\star, R^\star)$ to OSign().
   - $R^\star \subseteq S \setminus C$

Strong Security Model (Anonymity)

Anonymity against chosen setting attacks

Attacker create its own users, thus he knows all the secret keys.

Anonymity Game

1. Attacker sends to the challenger
   - a ring $R = \{pk_i\}_{i=1}^{\ell}$, two dististinct indices i_0, i_1
   - two secret keys sk_{i_0}, sk_{i_1} and a message μ
2. Challenger,
   - pick random $b \in \{0, 1\}$
   - send $\sigma_b \leftarrow Sign(\mu, sk_{i_b})$ to attacker
3. Attacker outputs a bit b' and wins the game if $b' = b$

Strong Security Model (Anonymity)

Anonymity against chosen setting attacks

Attacker create its own users, thus he knows all the secret keys.

Anonymity Game

1. Attacker sends to the challenger
   - a ring $R = \{pk_i\}_{i=1}^{\ell}$, two dististinct indices i_0, i_1
   - two secret keys sk_{i_0}, sk_{i_1} and a message μ
2. Challenger,
   - pick random $b \in \{0, 1\}$
   - send $\sigma_b \leftarrow Sign(\mu, sk_{i_b})$ to attacker
3. Attacker outputs a bit b' and wins the game if $b' = b$

Related Work

Brakerski-Kalai (eprint 2010/086):

- Hash-and-sign / bonsai-tree approach
- Chosen subring attack unforgeability
- No insider corruption proof

Wang-Sun (ICICS '11):

- Hash-and-sign / bonsai-tree approach
- Insider corruption proof works only for log-sized rings

Cayrel-Linder-Rückert-Silva (Latincrypt '10):

- Threshold ring signature scheme (hard feature to obtain)
- Uses Stern's construction for signature schemes
- No insider corruption proof

Sets and notations

- Let $\mathcal{R} = \mathbb{Z}_p[x] / \langle x^n + 1 \rangle$
- Let

$$C^{27} := \{\hat{s} = (s_1, \ldots, s_{27}) : s_i \in \mathcal{R}, \|s_i\|_\infty \leq 1\}$$

$$D^{27} := \{\hat{y} = (y_1, \ldots, y_{27}) : y_i \in \mathcal{R}, \|y_i\|_\infty \leq 3 \cdot 10^6\}$$

- For any $\hat{a} = (a_1, \ldots, a_{27}) \in \mathcal{R}^{27}$, we have the following hash function

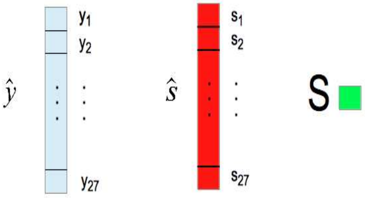
$$\begin{array}{llll} h_{\hat{a}}: & D^{27} & \to & \mathcal{R} \\ & \hat{y} & \mapsto & \hat{y} \odot \hat{a} = \sum_{i=1}^{27} a_i \cdot y_i \end{array}$$

- Let $\hat{v}, \hat{w} \in \mathcal{R}^{27}$ and $x \in \mathcal{R}$

$$h_{\hat{a}}(x \cdot \hat{v} + \hat{w}) = x \cdot h_{\hat{a}}(\hat{v}) + h_{\hat{a}}(\hat{w})$$

Key generation

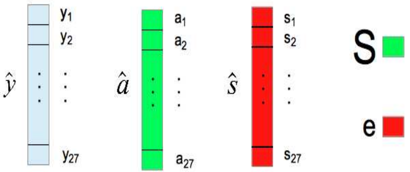
Signing key: $\hat{s} \in C^{27}$
Verification key: $h_{\hat{a}}$ and $S = h_{\hat{a}}(\hat{s})$



Key generation

Signing key: $\hat{s} \in C^{27}$

Verification key: $h_{\hat{a}}$ and $S = h_{\hat{a}}(\hat{s})$



$$H : \quad \{0,1\}^* \quad \rightarrow \quad \{g \in \mathcal{R} : \|g\|_\infty \leq 1\}$$

Signing algorithm

Sign(μ)

- Pick a random $\hat{y}$
- Compute $Y = h_{\hat{a}}(\hat{y})$
- Compute $e = H(Y\|\mu)$
- $\hat{z} = e \cdot \hat{s} + \hat{y}$
- Output $(\hat{z}, e)$

Verification algorithm

Check if

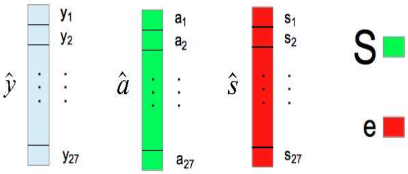
$$e \stackrel{?}{=} H(h_{\hat{a}}(\hat{z}) - e \cdot S\|\mu)$$

since

$$h_{\hat{a}}(\hat{z}) = e \cdot S + h_{\hat{a}}(\hat{y})$$

Key generation

Signing key: $\hat{s} \in C^{27}$
Verification key: $h_{\hat{a}}$ and $S = h_{\hat{a}}(\hat{s})$



$$H : \quad \{0,1\}^* \quad \to \quad \{g \in \mathcal{R} : \|g\|_\infty \leq 1\}$$

Signing algorithm

Sign(μ)

- Pick a random $\hat{y}$
- Compute $Y = h_{\hat{a}}(\hat{y})$
- Compute $e = H(Y\|\mu)$
- $\hat{z} = e \cdot \hat{s} + \hat{y}$
- Output $(\hat{z}, e)$

Verification algorithm

Check if

$$e \stackrel{?}{=} H(h_{\hat{a}}(\hat{z}) - e \cdot S\|\mu)$$

since

$$h_{\hat{a}}(\hat{z}) = e \cdot S + h_{\hat{a}}(\hat{y})$$

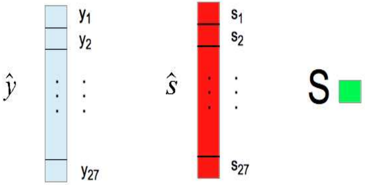
Toy Example: a ring with 3 members (First attempt)

Ring-KeyGen($\mathcal{R}$)

Secret keys: $\hat{s}_1, \hat{s}_2, \hat{s}_3 \in C^{27}$

Public keys: $h_{\hat{a}_1}, h_{\hat{a}_2}, h_{\hat{a}_3} := h_1, h_2, h_3.$

$$S_1 = h_1(\hat{s}_1), S_2 = h_2(\hat{s}_2), S_3 = h_3(\hat{s}_3)$$



Reminder

- Let $\mathcal{R} = \mathbb{Z}_p\,[x]\,/\,\langle x^n + 1\rangle$
- Let

$$C^{27} := \{\hat{s} = (s_1, \ldots, s_{27}) : s_i \in \mathcal{R}, \|s_i\|_\infty \le 1\}$$

$$h_{\hat{a}}(\hat{s}) = \hat{s} \odot \hat{a} = \sum_{i=1}^{27} a_i \cdot s_i$$

- Let $\hat{v}, \hat{w} \in \mathcal{R}^{27}$ and $x \in \mathcal{R}$

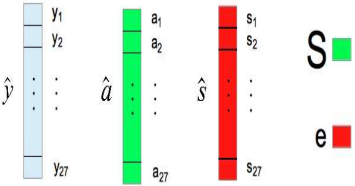
$$h_{\hat{a}}(x \cdot \hat{v} + \hat{w}) = x \cdot h_{\hat{a}}(\hat{v}) + h_{\hat{a}}(\hat{w})$$

Toy Example: a ring with 3 members (First attempt)

Ring-KeyGen

Secret keys: $\hat{s}_1, \hat{s}_2, \hat{s}_3 \in C^{27}$
Public keys: $h_{\hat{a}_1}, h_{\hat{a}_2}, h_{\hat{a}_3} := h_1, h_2, h_3.$

$$S_1 = h_1(\hat{s}_1), S_2 = h_2(\hat{s}_2), S_3 = h_3(\hat{s}_3)$$



Ring-Sign($\mu, \hat{s}_1$)

- Pick random $\hat{y}_1, \hat{y}_2, \hat{y}_3$
- Compute $Y = h_1(\hat{y}_1) + h_2(\hat{y}_2) + h_3(\hat{y}_3)$
- Compue $e = H(Y \| \mu)$
- $\hat{z}_1 = e \cdot \hat{s}_1 + \hat{y}_1$
- $\hat{z}_2 = \hat{y}_2$
- $\hat{z}_3 = \hat{y}_3$

Send $\sigma = (\hat{z}_1, \hat{z}_2, \hat{z}_3, e)$

Ring-Sign($\mu, \hat{s}_3$)

- Pick random $\hat{y}_1, \hat{y}_2, \hat{y}_3$
- Compute $Y = h_1(\hat{y}_1) + h_2(\hat{y}_2) + h_3(\hat{y}_3)$
- Compue $e = H(Y \| \mu)$
- $\hat{z}_1 = \hat{y}_1$
- $\hat{z}_2 = \hat{y}_2$
- $\hat{z}_3 = e \cdot \hat{s}_3 + \hat{y}_3$

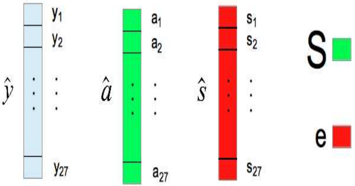
Send $\sigma = (\hat{z}_1, \hat{z}_2, \hat{z}_3, e)$

Toy Example: a ring with 3 members (First attempt)

Ring-KeyGen

Secret keys: $\hat{s}_1, \hat{s}_2, \hat{s}_3 \in C^{27}$
Public keys: $h_{\hat{a}_1}, h_{\hat{a}_2}, h_{\hat{a}_3} := h_1, h_2, h_3.$

$$S_1 = h_1(\hat{s}_1), S_2 = h_2(\hat{s}_2), S_3 = h_3(\hat{s}_3)$$



Ring-Sign$(\mu, \hat{s}_1)$

- Pick random $\hat{y}_1, \hat{y}_2, \hat{y}_3$
- Compute $Y = h_1(\hat{y}_1) + h_2(\hat{y}_2) + h_3(\hat{y}_3)$
- Compue $e = H(Y \| \mu)$
- $\hat{z}_1 = e \cdot \hat{s}_1 + \hat{y}_1$
- $\hat{z}_2 = \hat{y}_2$
- $\hat{z}_3 = \hat{y}_3$

Send $\sigma = (\hat{z}_1, \hat{z}_2, \hat{z}_3, e)$

Ring-Sign$(\mu, \hat{s}_3)$

- Pick random $\hat{y}_1, \hat{y}_2, \hat{y}_3$
- Compute $Y = h_1(\hat{y}_1) + h_2(\hat{y}_2) + h_3(\hat{y}_3)$
- Compue $e = H(Y \| \mu)$
- $\hat{z}_1 = \hat{y}_1$
- $\hat{z}_2 = \hat{y}_2$
- $\hat{z}_3 = e \cdot \hat{s}_3 + \hat{y}_3$

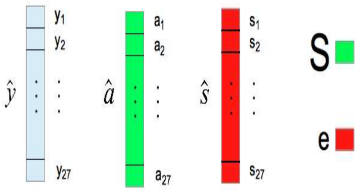
Send $\sigma = (\hat{z}_1, \hat{z}_2, \hat{z}_3, e)$

Toy Example: a ring with 3 members (First attempt)

Ring-KeyGen

Secret keys: $\hat{s}_1, \hat{s}_2, \hat{s}_3 \in C^{27}$
Public keys: $h_{\hat{a}_1}, h_{\hat{a}_2}, h_{\hat{a}_3} := h_1, h_2, h_3.$

$$S_1 = h_1(\hat{s}_1), S_2 = h_2(\hat{s}_2), S_3 = h_3(\hat{s}_3)$$



Ring-Sign$(\mu, \hat{s}_2)$

- Pick random $\hat{y}_1, \hat{y}_2, \hat{y}_3$
- Compute
  $Y = h_1(\hat{y}_1) + h_2(\hat{y}_2) + h_3(\hat{y}_3)$
- Compue $e = H(Y \| \mu)$
- $\hat{z}_1 = \hat{y}_1$
- $\hat{z}_2 = e \cdot \hat{s}_2 + \hat{y}_2$
- $\hat{z}_3 = \hat{y}_3$

Send $\sigma = (\hat{z}_1, \hat{z}_2, \hat{z}_3, e)$

Ring-Verif(μ, σ)

$$e \stackrel{?}{=} H\left(\sum_{i=1}^{3} h_i(\hat{z}_i) - e \cdot S_2 \| \mu\right)$$

since

$$h_1(\hat{z}_1) + h_2(\hat{z}_2) + h_3(\hat{z}_3)$$

$$= h_1(\hat{y}_1) + e \cdot S_2 + h_2(\hat{y}_2) + h_3(\hat{y}_3)$$

Toy Example: a ring with 3 members (First attempt)

Ring-Sign($\mu, \hat{s}_1$)

- Pick random $\hat{y}_1, \hat{y}_2, \hat{y}_3$
- Compute $Y = h_1(\hat{y}_1) + h_2(\hat{y}_2) + h_3(\hat{y}_3)$
- Compue $e = H(Y \| \mu)$
- $\hat{z}_1 = e \cdot \hat{s}_1 + \hat{y}_1$
- $\hat{z}_2 = \hat{y}_2$
- $\hat{z}_3 = \hat{y}_3$

Send $\sigma = (\hat{z}_1, \hat{z}_2, \hat{z}_3, e)$

Ring-Verif(μ, σ)

$$e \stackrel{?}{=} H(\sum_{i=1}^{3} h_i(\hat{z}_i) - e \cdot S_1 \| \mu)$$

since

$$h_1(\hat{z}_1) + h_2(\hat{z}_2) + h_3(\hat{z}_3)$$

$$= e \cdot S_1 + h_1(\hat{y}_1) + h_2(\hat{y}_2) + h_3(\hat{y}_3)$$

To obtain anonymity

S is the same for all users !

$$S = S_1 = S_2 = S_3$$

Toy Example: a ring with 3 members (First attempt)

Ring-Sign($\mu, \hat{s}_1$)

- Pick random $\hat{y}_1, \hat{y}_2, \hat{y}_3$
- Compute
  $Y = h_1(\hat{y}_1) + h_2(\hat{y}_2) + h_3(\hat{y}_3)$
- Compue $e = H(Y \| \mu)$
- $\hat{z}_1 = e \cdot \hat{s}_1 + \hat{y}_1$
- $\hat{z}_2 = \hat{y}_2$
- $\hat{z}_3 = \hat{y}_3$

Send $\sigma = (\hat{z}_1, \hat{z}_2, \hat{z}_3, e)$

Ring-Verif(μ, σ)

$$e \stackrel{?}{=} H(\sum_{i=1}^{3} h_i(\hat{z}_i) - e \cdot S_1 \| \mu)$$

since

$$h_1(\hat{z}_1) + h_2(\hat{z}_2) + h_3(\hat{z}_3)$$

$$= e \cdot S_1 + h_1(\hat{y}_1) + h_2(\hat{y}_2) + h_3(\hat{y}_3)$$

To obtain anonymity

S is the same for all users !

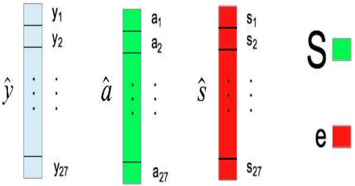
$$S = S_1 = S_2 = S_3$$

Toy Example: a ring with 3 members

Ring-KeyGen($\mathcal{R}$, S)

Secret keys: $\hat{s}_1, \hat{s}_2, \hat{s}_3 \in C^{27}$
Public keys: $h_{\hat{a}_1}, h_{\hat{a}_2}, h_{\hat{a}_3} := h_1, h_2, h_3.$

$$S = h_1(\hat{s}_1) = h_2(\hat{s}_2) = h_3(\hat{s}_3)$$



Ring-Sign($\mu, \hat{s}_2$)

- Pick random $\hat{y}_1, \hat{y}_2, \hat{y}_3$
- Compute
  $Y = h_1(\hat{y}_1) + h_2(\hat{y}_2) + h_3(\hat{y}_3)$
- Compue $e = H(Y \| \mu)$
- $\hat{z}_1 = \hat{y}_1$
- $\hat{z}_2 = e \cdot \hat{s}_2 + \hat{y}_2$
- $\hat{z}_3 = \hat{y}_3$

Send $\sigma = (\hat{z}_1, \hat{z}_2, \hat{z}_3, e)$

Ring-Verif(μ, σ)

Check if

$$e \stackrel{?}{=} H\left(\sum_{i=1}^{3} h_i(\hat{z}_i) - e \cdot S \| \mu\right)$$

since

$$h_1(\hat{z}_1) + h_2(\hat{z}_2) + h_3(\hat{z}_3)$$

$$= h_1(\hat{y}_1) + e \cdot S + h_2(\hat{y}_2) + h_3(\hat{y}_3)$$

Contributions

More efficient scheme

Unforgeability in the insider corruption setting

- proof that works even if the ring is of polynomial size .

We present a modification which can be applied to other schemes to provide the unforgeability in the insider corruption setting.

Anonymity against chosen setting attacks

Thanks !