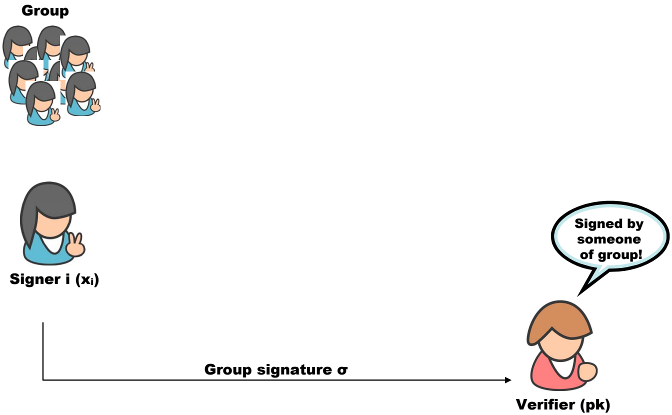
Non-Interactive Plaintext (In-)Equality Proofs and Group Signatures with Verifiable Controllable Linkability

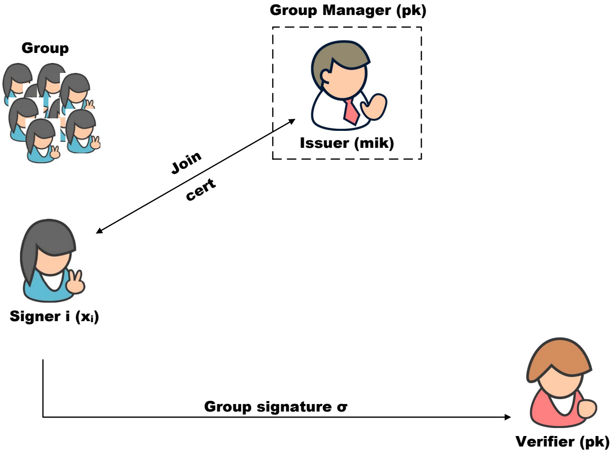
Olivier Blazy[1], David Derler[2], Daniel Slamanig[2], Raphael Spreitzer[2]
[1] **Université de Limoges, XLim, France**
[2] **IAIK, Graz University of Technology, Austria**

CT-RSA 2016, San Francisco, 2nd March 2016

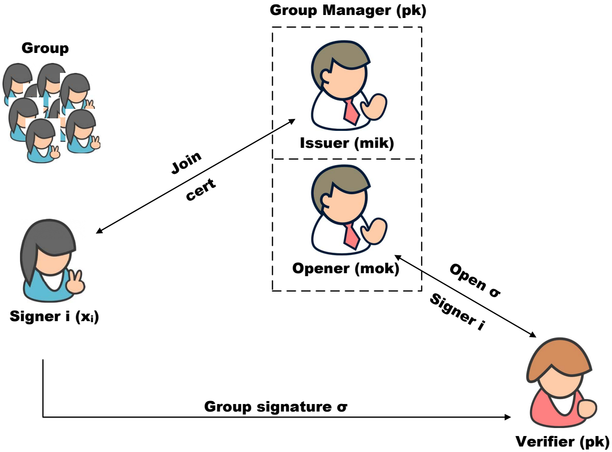
Group Signature Schemes [CvH91]



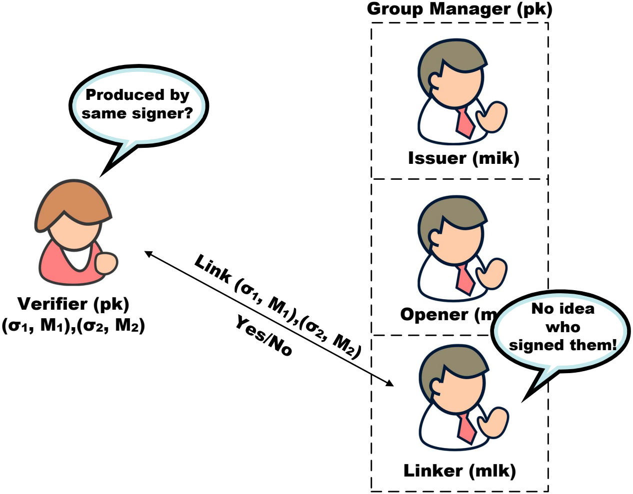
Group Signature Schemes [CvH91]



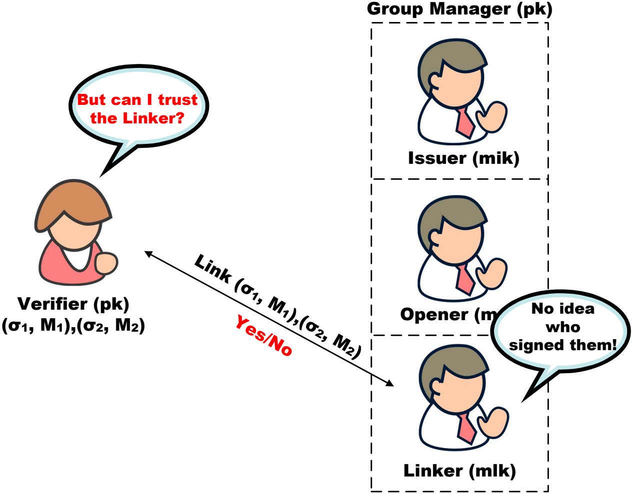
Group Signature Schemes [CvH91]



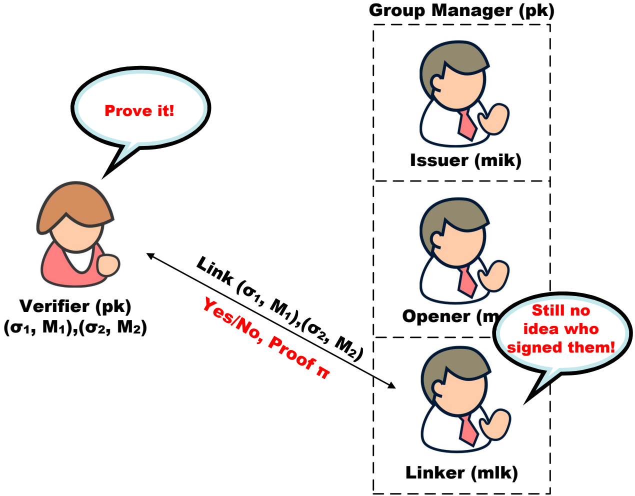
Controllable Linkability [HLhC+11, SSU14]



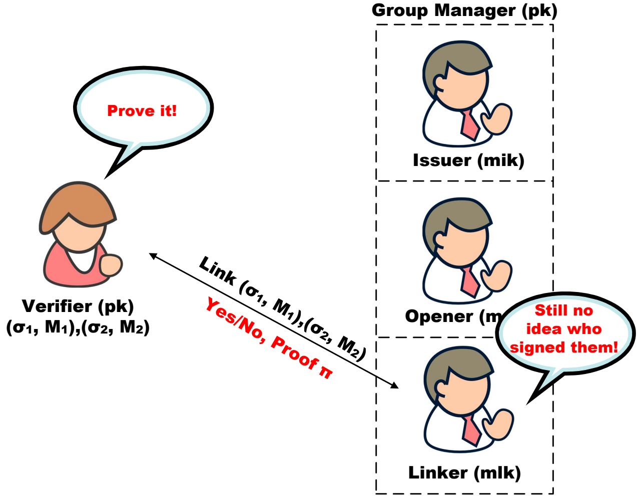
Controllable Linkability [HLhC+11, SSU14]



Verifiable Controllable Linkability



Verifiable Controllable Linkability



Primitive to prove plaintext (in-)equality

Contributions

1. Model generic proof system for plaintext (in-)equality

Contributions

1. Model generic proof system for plaintext (in-)equality
2. Efficient instantiation of this proof system

Contributions

1. Model generic proof system for plaintext (in-)equality
2. Efficient instantiation of this proof system
3. Group signatures with verifiable controllable linkability

Contributions

1. Model generic proof system for plaintext (in-)equality
2. Efficient instantiation of this proof system
3. Group signatures with verifiable controllable linkability
4. Extend GSs with verifiable controllable linkability (VCL)

Sign-Encrypt-Prove Paradigm

Basic building blocks

- $\mathcal{DS} = (\mathsf{KG_s}, \mathsf{Sign}, \mathsf{Verify})$

Sign-Encrypt-Prove Paradigm

Basic building blocks

- $\mathcal{DS} = (\mathsf{KG_s}, \mathsf{Sign}, \mathsf{Verify})$
- $\mathcal{AE} = (\mathsf{KG_e}, \mathsf{Enc}, \mathsf{Dec})$

Sign-Encrypt-Prove Paradigm

Basic building blocks

- $\mathcal{DS} = (\mathsf{KG_s}, \mathsf{Sign}, \mathsf{Verify})$
- $\mathcal{AE} = (\mathsf{KG_e}, \mathsf{Enc}, \mathsf{Dec})$
- Signature of Knowledge

Sign-Encrypt-Prove Paradigm

Basic building blocks

- $\mathcal{DS} = (\mathsf{KG_s}, \mathsf{Sign}, \mathsf{Verify})$
- $\mathcal{AE} = (\mathsf{KG_e}, \mathsf{Enc}, \mathsf{Dec})$
- Signature of Knowledge

Keys

- $gpk \leftarrow (pk_e, pk_s)$,

Sign-Encrypt-Prove Paradigm

Basic building blocks

- $\mathcal{DS} = (\mathsf{KG_s}, \mathsf{Sign}, \mathsf{Verify})$
- $\mathcal{AE} = (\mathsf{KG_e}, \mathsf{Enc}, \mathsf{Dec})$
- Signature of Knowledge

Keys

- $gpk \leftarrow (pk_e, pk_s)$, $gmsk \leftarrow sk_e$,

Sign-Encrypt-Prove Paradigm

Basic building blocks

- $\mathcal{DS} = (\mathsf{KG_s}, \mathsf{Sign}, \mathsf{Verify})$
- $\mathcal{AE} = (\mathsf{KG_e}, \mathsf{Enc}, \mathsf{Dec})$
- Signature of Knowledge

Keys

- $gpk \leftarrow (pk_e, pk_s)$, $gmsk \leftarrow sk_e$, $gmik \leftarrow sk_s$

Sign-Encrypt-Prove Paradigm

Basic building blocks

- $\mathcal{DS} = (\mathsf{KG_s}, \mathsf{Sign}, \mathsf{Verify})$
- $\mathcal{AE} = (\mathsf{KG_e}, \mathsf{Enc}, \mathsf{Dec})$
- Signature of Knowledge

Keys

- $gpk \leftarrow (pk_e, pk_s)$, $gmsk \leftarrow sk_e$, $gmik \leftarrow sk_s$

Join

- User's secret: x_i

Sign-Encrypt-Prove Paradigm

Basic building blocks

- $\mathcal{DS} = (\mathsf{KG_s}, \mathsf{Sign}, \mathsf{Verify})$
- $\mathcal{AE} = (\mathsf{KG_e}, \mathsf{Enc}, \mathsf{Dec})$
- Signature of Knowledge

Keys

- $gpk \leftarrow (pk_e, pk_s)$, $gmsk \leftarrow sk_e$, $gmik \leftarrow sk_s$

Join

- User's secret: x_i
- Issuer computes: $cert \leftarrow \mathsf{Sign}(gmik, f(x_i))$

Sign-Encrypt-Prove Paradigm I

Sign

- $T \leftarrow \mathrm{Enc}(pk_e, cert)$

Sign-Encrypt-Prove Paradigm I

Sign

- $T \leftarrow \mathrm{Enc}(pk_e, cert)$
- $\pi \leftarrow SoK\{(x_i, cert) : cert = \mathrm{Sign}(sk_s, f(x_i)) \wedge T = \mathrm{Enc}(pk_e, cert))\}(m)$

Sign-Encrypt-Prove Paradigm I

Sign

- $T \leftarrow \mathrm{Enc}(pk_e, cert)$
- $\pi \leftarrow SoK\{(x_i, cert) : cert = \mathrm{Sign}(sk_s, f(x_i)) \wedge T = \mathrm{Enc}(pk_e, cert)\}(m)$
- $\sigma \leftarrow (T, \pi)$

Sign-Encrypt-Prove Paradigm I

Sign

- $T \leftarrow \mathrm{Enc}(pk_e, cert)$
- $$\begin{aligned}\pi \leftarrow SoK\{(x_i, cert) : cert &= \mathrm{Sign}(sk_s, f(x_i)) \wedge \\ T &= \mathrm{Enc}(pk_e, cert))\}(m)\end{aligned}$$
- $\sigma \leftarrow (T, \pi)$

Verify

- "verification of π"

Sign-Encrypt-Prove Paradigm I

Sign

- $T \leftarrow \mathrm{Enc}(pk_e, cert)$
- $$\begin{aligned}\pi \leftarrow SoK\{(x_i, cert) : cert &= \mathrm{Sign}(sk_s, f(x_i)) \wedge \\ T &= \mathrm{Enc}(pk_e, cert))\}(m)\end{aligned}$$
- $\sigma \leftarrow (T, \pi)$

Verify

- "verification of π"

Open

- $cert \leftarrow \mathrm{Dec}(sk_e, T)$

Controllable Linkability

Public key encryption with equality tests [Tan12, SSU14]

- Conventional public key encryption scheme

Controllable Linkability

Public key encryption with equality tests [Tan12, SSU14]

- Conventional public key encryption scheme

+ Com algorithm for equality tests using trapdoor

Controllable Linkability

Public key encryption with equality tests [Tan12, SSU14]

- Conventional public key encryption scheme
+ Com algorithm for equality tests using trapdoor
- $\Rightarrow$ Link: $1/0 \leftarrow \mathrm{Com}(T, T', gmlk)$

Controllable Linkability

Public key encryption with equality tests [Tan12, SSU14]

- Conventional public key encryption scheme
+ Com algorithm for equality tests using trapdoor
- $\Rightarrow$ Link: $1/0 \leftarrow \mathrm{Com}(T, T', gmlk)$

- Semantic security without trapdoor

Controllable Linkability

Public key encryption with equality tests [Tan12, SSU14]

- Conventional public key encryption scheme
+ Com algorithm for equality tests using trapdoor
- ⇒ Link: $1/0 \leftarrow \mathrm{Com}(T, T', gmlk)$

- Semantic security without trapdoor
- One-way security for trapdoor holders

Controllable Linkability

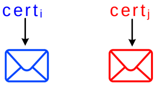
Public key encryption with equality tests [Tan12, SSU14]

- Conventional public key encryption scheme
+ Com algorithm for equality tests using trapdoor
- $\Rightarrow$ Link: $1/0 \leftarrow \mathrm{Com}(T, T', gmlk)$

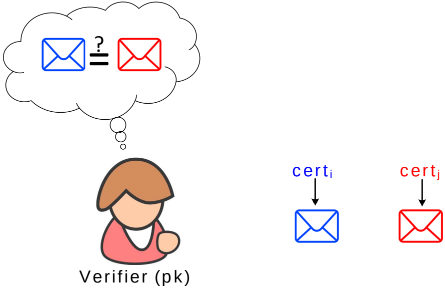
- Semantic security without trapdoor
- One-way security for trapdoor holders

$\Rightarrow$ ZK proof of knowledge of trapdoor for VCL

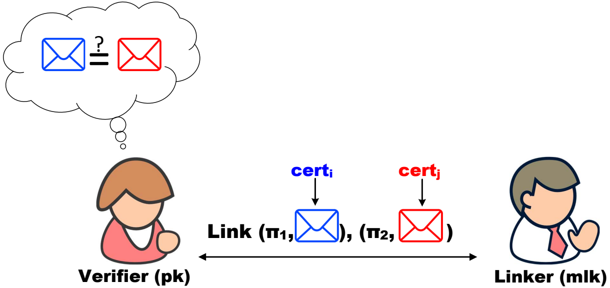
Setting



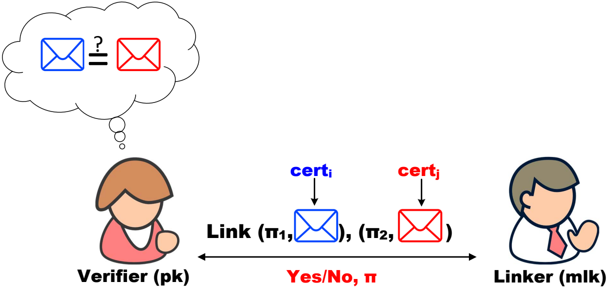
Setting



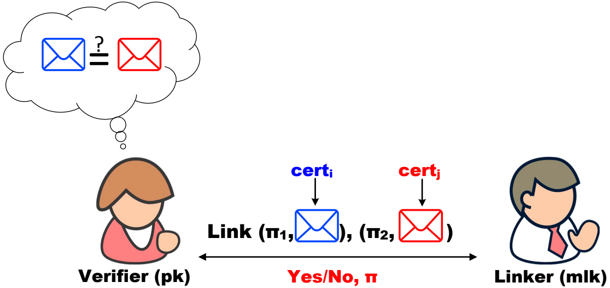
Setting



Setting



Setting



Non-interactive plaintext (in-)equality proofs

Non-Interactive Plaintext (In-)Equality Proofs

Given any $\mathcal{PKEQ}$ and ciphertexts T and T' under pk

Proof system Π

Non-Interactive Plaintext (In-)Equality Proofs

Given any $\mathcal{PKEQ}$ and ciphertexts T and T' under pk

Proof system Π

1. Prove knowledge of trapdoor tk

Non-Interactive Plaintext (In-)Equality Proofs

Given any $\mathcal{PKEQ}$ and ciphertexts T and T' under pk

Proof system Π

1. Prove knowledge of trapdoor tk
2. $\mathsf{Com} = 1$ (membership) or $\mathsf{Com} = 0$ (non-membership)

Non-Interactive Plaintext (In-)Equality Proofs

Given any $\mathcal{PKEQ}$ and ciphertexts T and T' under pk

Proof system Π

1. Prove knowledge of trapdoor tk
2. $\mathsf{Com} = 1$ (membership) or $\mathsf{Com} = 0$ (non-membership)
3. Without revealing trapdoor tk

(Non-)Membership Proofs

$\mathsf{Com} = 1$ defines L for membership $((x, w) \in L_R)$

- Witnessed by trapdoor tk
- Standard techniques [GS08]

(Non-)Membership Proofs

$\mathsf{Com} = 1$ defines L for membership $((x, w) \in L_R)$

- Witnessed by trapdoor tk
- Standard techniques [GS08]

$\mathsf{Com} = 0$ defines L for non-membership $((x, w) \notin L_R)$

(Non-)Membership Proofs

$\mathsf{Com} = 1$ defines L for membership $((x, w) \in L_R)$

- Witnessed by trapdoor tk
- Standard techniques [GS08]

$\mathsf{Com} = 0$ defines L for non-membership $((x, w) \notin L_R)$

- Idea [BCV15]
  - Π_1: Failing membership proof for L_R

(Non-)Membership Proofs

$\mathsf{Com} = 1$ defines L for membership ($(x, w) \in L_R$)

- Witnessed by trapdoor tk
- Standard techniques [GS08]

$\mathsf{Com} = 0$ defines L for non-membership ($(x, w) \notin L_R$)

- Idea [BCV15]
  - Π_1: Failing membership proof for L_R
  - Π_2: Proof that Π_1 has been computed honestly

(Non-)Membership Proofs

$\mathsf{Com} = 1$ defines L for membership ($(x, w) \in L_R$)

- Witnessed by trapdoor tk
- Standard techniques [GS08]

$\mathsf{Com} = 0$ defines L for non-membership ($(x, w) \notin L_R$)

- Idea [BCV15]
  - Π_1: Failing membership proof for L_R
  - Π_2: Proof that Π_1 has been computed honestly
- Efficient instantiations (GS and SPHFs)

(Non-)Membership Proofs

Com $= 1$ defines L for membership $((x, w) \in L_R)$

- Witnessed by trapdoor tk
- Standard techniques [GS08]

Com $= 0$ defines L for non-membership $((x, w) \notin L_R)$

- Idea [BCV15]
  - Π_1: Failing membership proof for L_R
  - Π_2: Proof that Π_1 has been computed honestly
- Efficient instantiations (GS and SPHFs)
- Technicalities: m, r must be known [BCV15]

Smooth Projective Hash Functions (SPHFs)

$w : (x, w) \in R$

Hashing key: hk

Projection key: hp

Verifier

Statement: $x \in L_R$

Prover

$$\xrightarrow{\quad hp \quad}$$

$$H \leftarrow ProjHash(hp, x, w)$$

$$\xleftarrow{\quad H \quad}$$

$$H \stackrel{?}{=} Hash(hk, x)$$

If $x \in L_R : Hash(hk, x) \stackrel{!}{=} ProjHash(hp, x, w)$

(Correctness)

Construction - Non-Membership Proof

Hashing key: hk

Projection key: hp

Verifier

Statement: $x \in L_R$

Prover

$$\xrightarrow{\quad hp \quad}$$

$$H \leftarrow ProjHash(hp, x, w)$$

$$\xleftarrow{\quad H \quad}$$

$$H \stackrel{?}{\neq} Hash(hk, x)$$

Construction - Non-Membership Proof

Hashing key: hk

Projection key: hp

Verifier

Statement: $x \in L_R$

Prover

$$\text{Verifier} \xrightarrow{hp} \text{Prover}$$

$$H \leftarrow ProjHash(hp, x, w)$$

$$\pi_1 = Proof((hp, x, H), w)$$

$$\text{Verifier} \xleftarrow{H} \text{Prover}$$

$$H \stackrel{?}{\neq} Hash(hk, x)$$

Construction - Non-Membership Proof

Hashing key: hk

Projection key: hp

Verifier

Statement: $x \in L_R$

Prover

$$\xrightarrow{\quad hp \quad}$$

$$H \leftarrow ProjHash(hp, x, w)$$

$$\xleftarrow{\quad H, \pi_1 \quad}$$

$$\pi_1 = Proof((hp, x, H), w)$$

$$H \stackrel{?}{\neq} Hash(hk, x)$$

Blazy, Derler, Slamanig, Spreitzer

CT-RSA 2016, San Francisco, 2nd March 2016

Construction - Non-Membership Proof

Hashing key: *hk*
Projection key: *hp*

Verifier

Statement: $x \in L_R$

Prover

$$\xrightarrow{\quad hp \quad}$$

$$H \leftarrow ProjHash(hp, x, w)$$

$$\xleftarrow{\quad H, \pi_1 \quad} \qquad \pi_1 = Proof((hp, x, H), w)$$

$$H \overset{?}{\neq} Hash(hk, x) \ \wedge \ Verify((hp, x, H), \pi_1) \overset{?}{=} 1$$

Construction - Non-Membership Proof

Hashing key: hk
Projection key: hp $\longrightarrow$

Verifier

Statement: $x \in L_R$

Prover

$$\xrightarrow{\quad hp \quad}$$

$$H \leftarrow ProjHash(hp, x, w)$$

$$\xleftarrow{\quad H, \pi_1 \quad} \qquad \pi_1 = Proof((hp, x, H), w)$$

$$H \overset{?}{\neq} Hash(hk, x) \;\wedge\; Verify((hp, x, H), \pi_1) \overset{?}{=} 1$$

Construction - Non-Membership Proof

Hashing key: hk

Projection key: hp

Verifier

Statement: $x \in L_R$

Prover

$$\xrightarrow{\quad hp \quad}$$

$$H \leftarrow ProjHash(hp, x, w)$$

$$\xleftarrow{\quad H, \pi_1 \quad} \qquad \pi_1 = Proof((hp, x, H), w)$$

$$H \overset{?}{\neq} Hash(hk, x) \;\wedge\; Verify((hp, x, H), \pi_1) \overset{?}{=} 1$$

Construction - Non-Membership Proof

Hashing key: hk
Projection key: hp

Verifier

Statement: $x \in L_R$

Prover

$$H \leftarrow ProjHash(hp, x, w)$$

$$\xleftarrow{\quad H, \pi_1 \quad} \qquad \pi_1 = Proof((hp, x, H), w)$$

$$H \overset{?}{\neq} Hash(hk, x) \ \wedge \ Verify((hp, x, H), \pi_1) \overset{?}{=} 1$$

Construction - Non-Membership Proof

Hashing key: hk
Projection key: hp

Verifier

Statement: $x \in L_R$

Prover

$H \leftarrow ProjHash(hp, x, w)$

$\pi_1 = Proof((hp, x, H), w)$

$\xleftarrow{H, \pi_1}$

$H \overset{?}{\neq} Hash(hk, x) \ \wedge \ Verify((hp, x, H), \pi_1) \overset{?}{=} 1$

Construction - Non-Membership Proof

Hashing key: hk

Projection key: hp

Verifier

Statement: $x \in L_R$

Prover

$H' \leftarrow Hash(hk, x)$

$H \leftarrow ProjHash(hp, x, w)$

H, π_1 ←

$\pi_1 = Proof((hp, x, H), w)$

Construction - Non-Membership Proof

Hashing key: $\color{red}{hk}$

Projection key: $\color{blue}{hp}$

Verifier

Statement: $x \in L_R$

Prover

$H' \leftarrow Hash(\color{red}{hk}, x)$

$H \leftarrow ProjHash(\color{blue}{hp}, x, w)$

$\pi_1 = Proof((\color{blue}{hp}, x, H), w)$

$$\xleftarrow{\quad H, H', \color{blue}{hp}, \pi_1 \quad}$$

Construction - Non-Membership Proof

Hashing key: hk

Projection key: hp

Verifier

Statement: $x \in L_R$

Prover

$H' \leftarrow Hash(hk, x)$

$H \leftarrow ProjHash(hp, x, w)$

$\pi_1 = Proof((hp, x, H), w)$

$$\xleftarrow{\quad H, H', hp, \pi_1 \quad}$$

$$H \overset{?}{\neq} H' \ \wedge \ Verify((hp, x, H), \pi_1) \overset{?}{=} 1$$

Construction - Non-Membership Proof

Hashing key: hk

Projection key: hp

Verifier

Statement: $x \in L_R$

Prover

$H' \leftarrow Hash(hk, x)$

$\pi_2 \leftarrow Proof((H' \wedge hp), hk)$

$H \leftarrow ProjHash(hp, x, w)$

$\xleftarrow{H, H', hp, \pi_1}$

$\pi_1 = Proof((hp, x, H), w)$

$$H \overset{?}{\neq} H' \wedge Verify((hp, x, H), \pi_1) \overset{?}{=} 1$$

Construction - Non-Membership Proof

Hashing key: $\color{red}{hk}$
Projection key: $\color{blue}{hp}$

Verifier

Statement: $x \in L_R$

Prover

$H' \leftarrow Hash(\color{red}{hk}, x)$

$\pi_2 \leftarrow Proof((H' \wedge \color{blue}{hp}), \color{red}{hk})$

$H \leftarrow ProjHash(\color{blue}{hp}, x, w)$

$\pi_1 = Proof((\color{blue}{hp}, x, H), w)$

$\xleftarrow{H, H', \color{blue}{hp}, \pi_1, \pi_2}$

$H \stackrel{?}{\neq} H' \;\wedge\; Verify((\color{blue}{hp}, x, H), \pi_1) \stackrel{?}{=} 1$

Construction - Non-Membership Proof

Hashing key: hk

Projection key: hp

Verifier

Statement: $x \in L_R$

Prover

$H' \leftarrow Hash(hk, x)$

$\pi_2 \leftarrow Proof((H' \wedge hp), hk)$

$H \leftarrow ProjHash(hp, x, w)$

$\pi_1 = Proof((hp, x, H), w)$

$$\xleftarrow{\quad H,\, H',\, hp,\, \pi_1,\, \pi_2 \quad}$$

$$H \overset{?}{\neq} H' \;\wedge\; Verify((hp, x, H), \pi_1) \overset{?}{=} 1 \;\wedge\; Verify((H' \wedge hp), \pi_2) \overset{?}{=} 1$$

Example of Efficient Instantiation

ElGamal with equality tests (as in [SSU14])

- Keypair: $(sk, pk) \leftarrow (x, g^x) \in \mathbb{Z}_p \times \mathbb{G}_1$
- Trapdoor: $(\hat{r}, \hat{r}^x) \in \mathbb{G}_2 \times \mathbb{G}_2$
- Encryption of m: $(g^r, m \cdot g^{x \cdot r}) \in \mathbb{G}_1 \times \mathbb{G}_1$

Example of Efficient Instantiation

ElGamal with equality tests (as in [SSU14])

- Keypair: $(sk, pk) \leftarrow (x, g^{x}) \in \mathbb{Z}_p \times \mathbb{G}_1$
- Trapdoor: $(\hat{r}, \hat{r}^{x}) \in \mathbb{G}_2 \times \mathbb{G}_2$
- Encryption of m: $(g^{r}, m \cdot g^{x \cdot r}) \in \mathbb{G}_1 \times \mathbb{G}_1$

Pairing based equality test:

- Ciphertexts: $(g^{r}, m \cdot g^{x \cdot r}), (g^{r'}, m' \cdot g^{x \cdot r'})$

$$m = m' \iff \frac{e(m \cdot g^{x \cdot r}, \hat{r})}{e(g^{r}, \hat{r}^{x})} = \frac{e(m' \cdot g^{x \cdot r'}, \hat{r})}{e(g^{r'}, \hat{r}^{x})}$$

Instantiation of $\Pi_\in$

Com = 1: plaintext equality proof

$$((g^{r}, m \cdot g^{x \cdot r}), (g^{r'}, m' \cdot g^{x \cdot r'}), g^{x}) \in L_{\in} \iff$$

$$\frac{e(m \cdot g^{x \cdot r}, \hat{r})}{e(g^{r}, \hat{r}^{x})} = \frac{e(m' \cdot g^{x \cdot r'}, \hat{r})}{e(g^{r'}, \hat{r}^{x})} \quad \wedge$$

$$e(g, \hat{r}^{x}) = e(g^{x}, \hat{r})$$

Instantiation of $\Pi_\in$

Com = 1: plaintext equality proof

$$((g^{r}, m \cdot g^{x \cdot r}), (g^{r'}, m' \cdot g^{x \cdot r'}), g^{x}) \in L_\in \iff$$

$$\frac{e(m \cdot g^{x \cdot r}, \hat{r})}{e(g^{r}, \hat{r}^{x})} = \frac{e(m' \cdot g^{x \cdot r'}, \hat{r})}{e(g^{r'}, \hat{r}^{x})} \wedge$$

$$e(g, \hat{r}^{x}) = e(g^{x}, \hat{r})$$

$$\prod_{i=1}^{2} e(A_i, \underline{\hat{Y}_i}) = \frac{e(m \cdot g^{x \cdot r} \cdot (m' \cdot g^{x \cdot r'})^{-1}, \hat{r})}{e(g^{r} \cdot g^{-r'}, \hat{r}^{x})} = 1_{\mathbb{G}_T}$$

Instantiation of $\Pi_{\notin}$

Com $=0$: plaintext inequality proof

$$((g^{r}, m \cdot g^{x \cdot r}), (g^{r'}, m' \cdot g^{x \cdot r'}), g^{x}) \in L_{\notin} \iff$$

$$\frac{e(m \cdot g^{x \cdot r}, \hat{r})}{e(g^{r}, \hat{r}^{x})} \neq \frac{e(m' \cdot g^{x \cdot r'}, \hat{r})}{e(g^{r'}, \hat{r}^{x})} \quad \wedge$$

$$e(g, \hat{r}^{x}) = e(g^{x}, \hat{r})$$

Instantiation of $\Pi_{\notin}$

Com $= 0$: plaintext inequality proof

$$((g^{r}, m \cdot g^{x \cdot r}), (g^{r'}, m' \cdot g^{x \cdot r'}), g^{x}) \in L_{\notin} \iff$$

$$\frac{e(m \cdot g^{x \cdot r}, \hat{r})}{e(g^{r}, \hat{r}^{x})} \neq \frac{e(m' \cdot g^{x \cdot r'}, \hat{r})}{e(g^{r'}, \hat{r}^{x})} \quad \wedge$$

$$e(g, \hat{r}^{x}) = e(g^{x}, \hat{r})$$

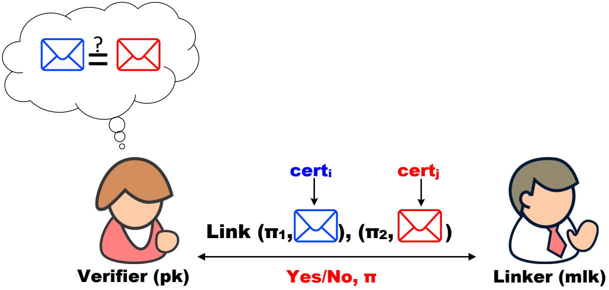
$\Rightarrow$ Our construction for non-membership proofs

NIPEI Proof System

Proof system $\Pi = (\Pi_{\in}, \Pi_{\notin})$

NIPEI Proof System

Proof system $\Pi = (\Pi_{\in}, \Pi_{\notin})$



Blazy, Derler, Slamanig, Spreitzer

GSSs with Verifiable Controllable Linkability

Extended security model for VCL-GS

- Algorithms: Link and $\mathsf{Link}_{\mathsf{Judge}}$
- Property: linking soundness

GSSs with Verifiable Controllable Linkability

Extended security model for VCL-GS

- Algorithms: Link and $\text{Link}_{\text{Judge}}$
- Property: linking soundness

Instantiation based on NIPEI

- Link: Π.Proof
- $\text{Link}_{\text{Judge}}$: Π.Verify

Take-Home Message

- Proposed generic approach for (in-)equality proof

Take-Home Message

- Proposed generic approach for (in-)equality proof
- Efficient instantiation in the pairing setting

Take-Home Message

- Proposed generic approach for (in-)equality proof
- Efficient instantiation in the pairing setting
- Also works for DLIN and CCA-secure ElGamal variants
  - Compatible with Naor-Yung paradigm (for free)

Take-Home Message

- Proposed generic approach for (in-)equality proof
- Efficient instantiation in the pairing setting
- Also works for DLIN and CCA-secure ElGamal variants
  - Compatible with Naor-Yung paradigm (for free)
- Novel application
  - GSSs with verifiable controllable linkability

Non-Interactive Plaintext (In-)Equality Proofs and Group Signatures with Verifiable Controllable Linkability

Olivier Blazy[1], **David Derler**[2], **Daniel Slamanig**[2], **Raphael Spreitzer**[2]
[1] **Université de Limoges, XLim, France**
[2] **IAIK, Graz University of Technology, Austria**

CT-RSA 2016, San Francisco, 2nd March 2016

Bibliography I

[BCV15] Olivier Blazy, Céline Chevalier, and Damien Vergnaud. Non-Interactive Zero-Knowledge Proofs of Non-Membership. In *CT-RSA*, 2015.

[CvH91] David Chaum and Eugène van Heyst. Group Signatures. In *EUROCRYPT*, 1991.

[GS08] Jens Groth and Amit Sahai. Efficient Non-interactive Proof Systems for Bilinear Groups. In *EUROCRYPT*, 2008.

[HLhC+11] Jung Yeon Hwang, Sokjoon Lee, Byung ho Chung, Hyun Sook Cho, and DaeHun Nyang. Short Group Signatures with Controllable Linkability. In *LightSec*. IEEE, 2011.

[SSU14] Daniel Slamanig, Raphael Spreitzer, and Thomas Unterluggauer. Adding Controllable Linkability to Pairing-Based Group Signatures for Free. In *ISC*, 2014.

[Tan12] Qiang Tang. Public Key Encryption Supporting Plaintext Equality Test and User-Specified Authorization. *Security and Communication Networks*, 5(12), 2012.