Commuting signatures and verifiable encryption

Georg Fuchsbauer

University of Bristol

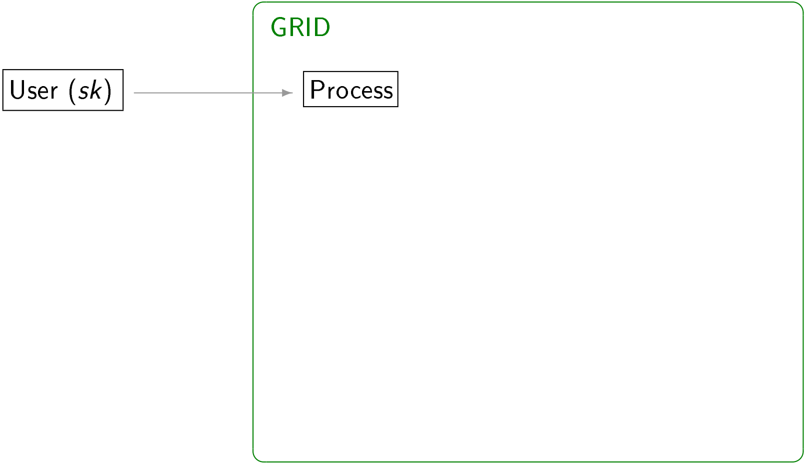
Darmstadt, 21.11.2011

Outline of this talk

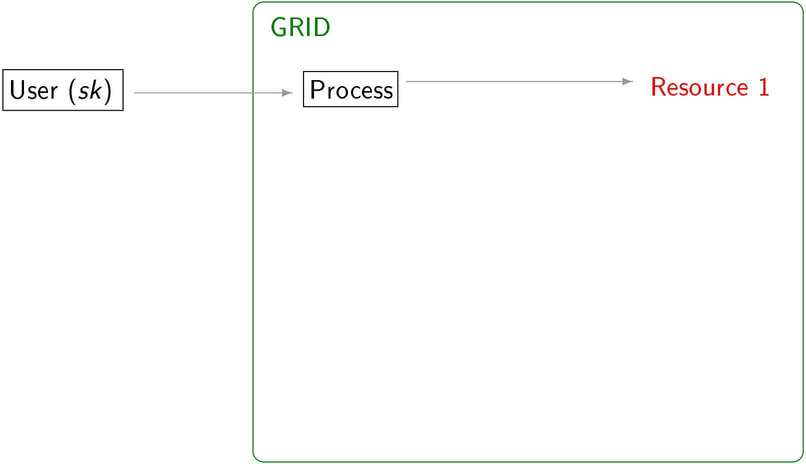
User (sk)

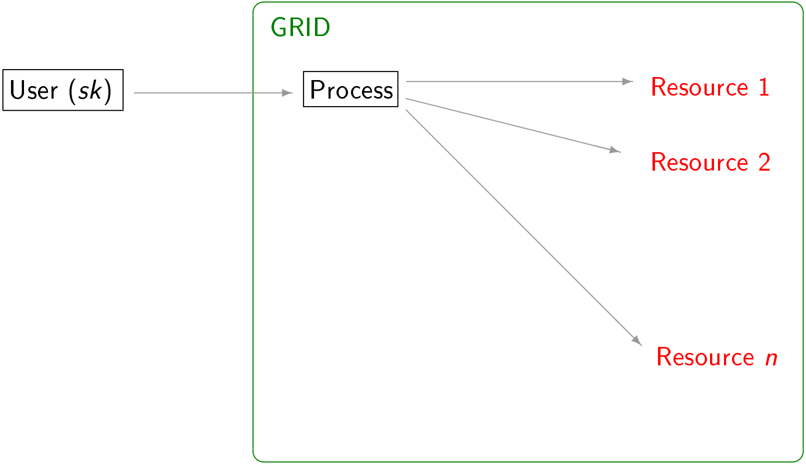
GRID

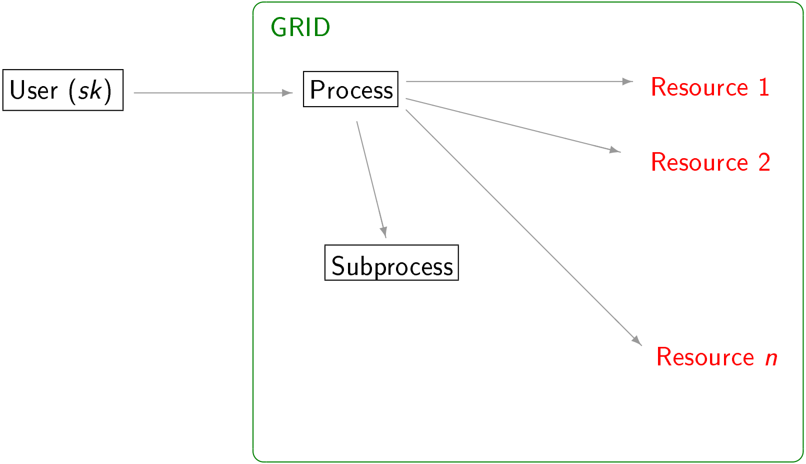
Motivation : GRID Computing

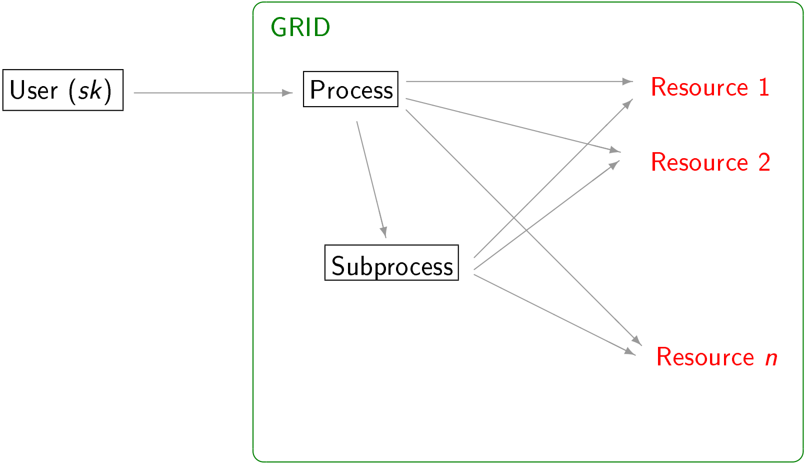


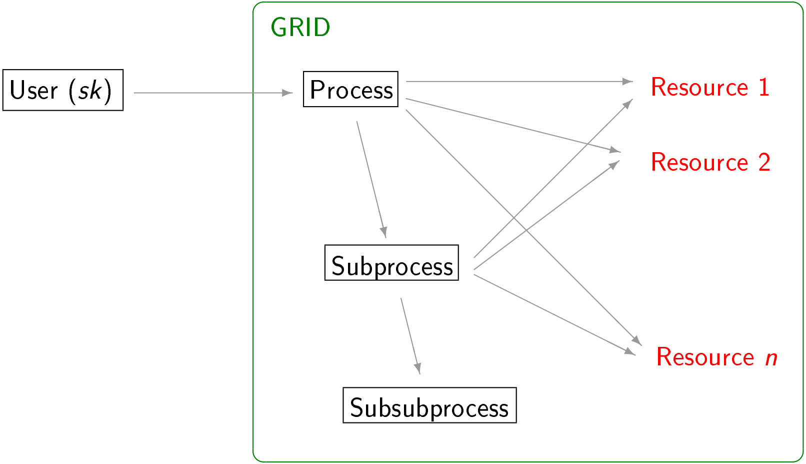
Motivation : GRID Computing

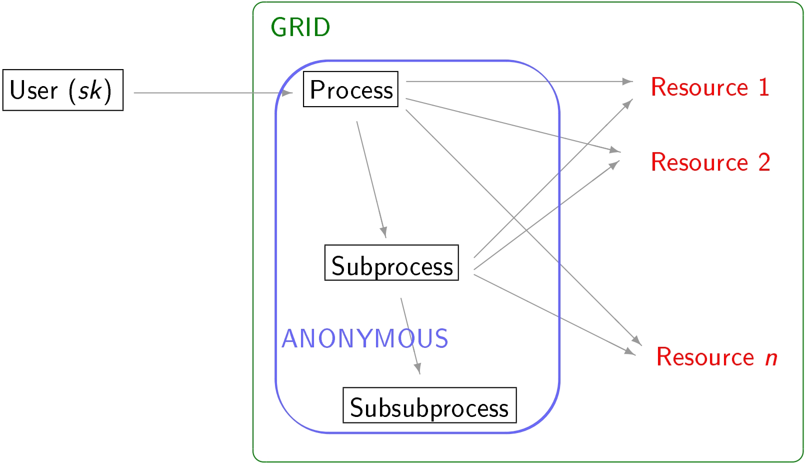


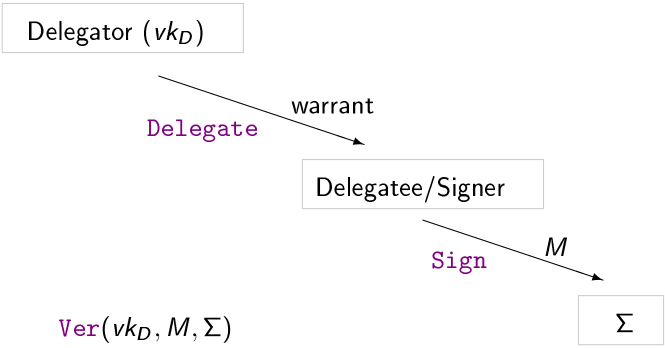

GRID
User (sk)
Process
Resource 1
Resource 2
Resource n

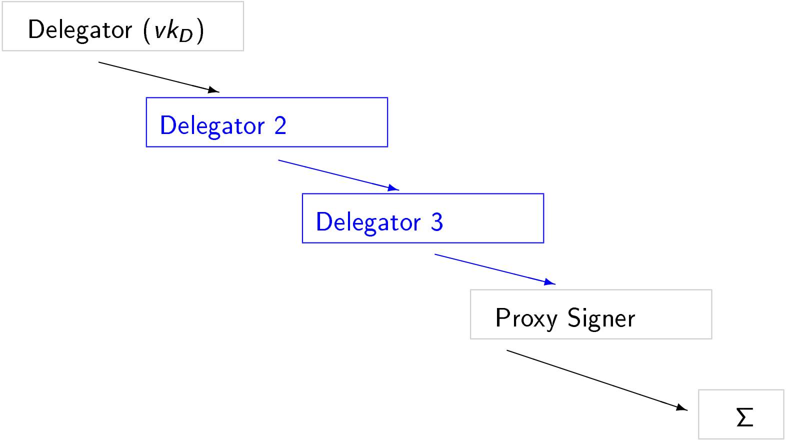

GRID
User (sk)
Process
Subprocess
Resource 1
Resource 2
Resource n

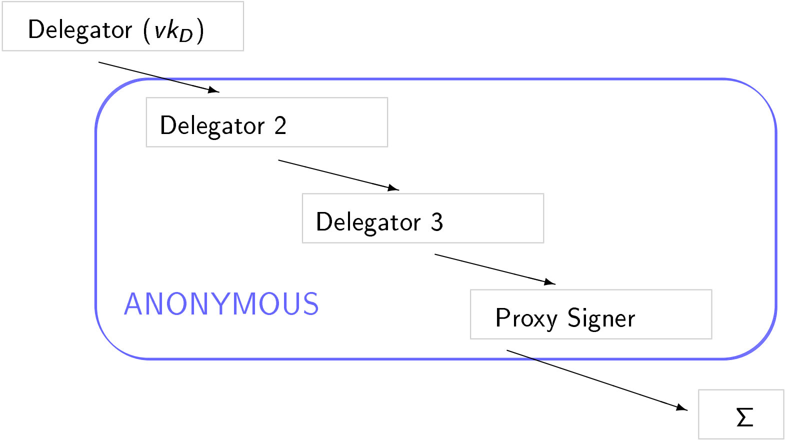

User (sk)
GRID
Process
Subprocess
Resource 1
Resource 2
Resource n

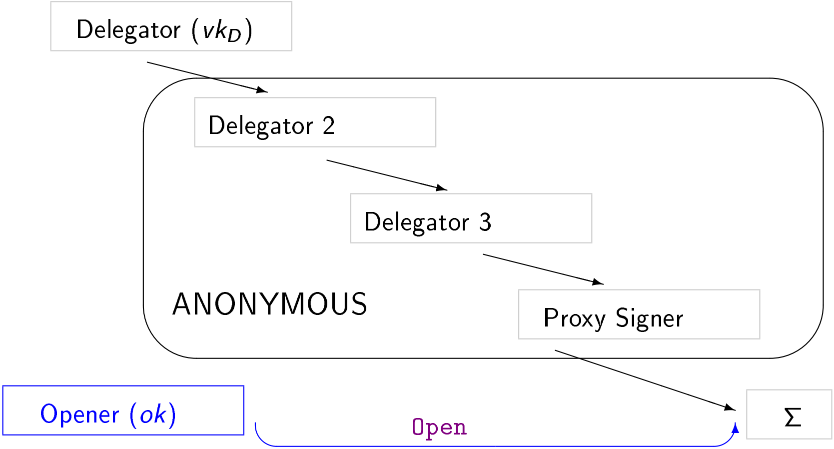

User (sk)
GRID
Process
Resource 1
Resource 2
Subprocess
Resource n
Subsubprocess


GRID
User (sk)
Process
Subprocess
Subsubprocess
ANONYMOUS
Resource 1
Resource 2
Resource n

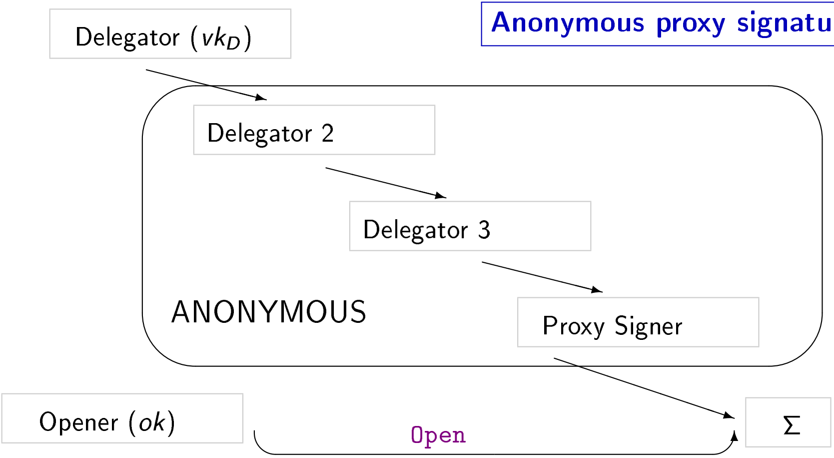

Delegator (vk_D)
warrant
Delegate
Delegatee/Signer
M
Sign
Σ
$\mathtt{Ver}(vk_D, M, \Sigma)$


Delegator (vk_D)
Delegator 2
Delegator 3
Proxy Signer
Σ


Delegator (vk_D)
Delegator 2
Delegator 3
ANONYMOUS
Proxy Signer
Σ


Delegator (vk_D)
Delegator 2
Delegator 3
ANONYMOUS
Proxy Signer
Opener (ok)
Open
Σ

Anonymous proxy signatures



Ingredients

- Digital signatures
- Public-key encryption

Ingredients

- Digital signatures
- Public-key encryption
- Non-interactive zero-knowledge proofs (NIZK)

  ...allow us to prove validity of a statement

  without revealing anything else

[simplified version]

Setup Generate decryption key for opening authority

[simplified version]

Setup Generate decryption key for opening authority
System parameters : [. . .]

Anonymous proxy signatures : Generic construction II

[simplified version]

Setup Generate decryption key for opening authority
System parameters : [. . .]

Delegate Sign delegatee's verification key → warrant
Re-delegate Additionally forward received warrant(s)

$$vk_0 \xrightarrow{\Sigma_1} vk_1 \xrightarrow{\Sigma_2} \bullet\bullet\bullet \xrightarrow{\Sigma_n} vk_n$$

Anonymous proxy signatures : Generic construction II

[simplified version]

Setup Generate decryption key for opening authority
System parameters : [. . .]

Delegate Sign delegatee's verification key → warrant
Re-delegate Additionally forward received warrant(s)

Proxy-sign Sign message

$$vk_0 \xrightarrow{\Sigma_1} vk_1 \xrightarrow{\Sigma_2} \bullet\bullet\bullet \xrightarrow{\Sigma_n} vk_n \xrightarrow{\Sigma_M} M$$

Anonymous proxy signatures : Generic construction II

[simplified version]

Setup Generate decryption key for opening authority
System parameters : [. . .]

Delegate Sign delegatee's verification key → warrant
Re-delegate Additionally forward received warrant(s)

Proxy-sign Sign message
Encrypt

- delegators' verification keys
- warrants
- signature on message

$$vk_0 \xrightarrow{\boxed{\Sigma_1}} \boxed{vk_1} \xrightarrow{\boxed{\Sigma_2}} \cdots \xrightarrow{\boxed{\Sigma_n}} \boxed{vk_n} \xrightarrow{\boxed{\Sigma_M}} M$$

[simplified version]

Setup Generate decryption key for opening authority
System parameters : [. . .]

Delegate Sign delegatee's verification key → warrant
Re-delegate Additionally forward received warrant(s)

Proxy-sign Sign message
$\boxed{\text{Encrypt}}$

- delegators' verification keys
- warrants
- signature on message

Prove correctness

$$vk_0 \xrightarrow[\pi_1]{\boxed{\Sigma_1}} \boxed{vk_1} \xrightarrow[\pi_2]{\boxed{\Sigma_2}} \cdots \xrightarrow[\pi_n]{\boxed{\Sigma_n}} \boxed{vk_n} \xrightarrow[\pi_M]{\boxed{\Sigma_M}} M$$

[simplified version]

Setup Generate decryption key for opening authority
System parameters : [. . .]

Delegate Sign delegatee's verification key → warrant
Re-delegate Additionally forward received warrant(s)

Proxy-sign Sign message
Encrypt
- delegators' verification keys
- warrants
- signature on message

Prove correctness

Proxy signature :
- ciphertexts
- proofs

$$vk_0 \xrightarrow[\pi_1]{\boxed{\Sigma_1}} \boxed{vk_1} \xrightarrow[\pi_2]{\boxed{\Sigma_2}} \cdots \xrightarrow[\pi_n]{\boxed{\Sigma_n}} \boxed{vk_n} \xrightarrow[\pi_M]{\boxed{\Sigma_M}} M$$

Anonymous proxy signatures : Generic construction II

[simplified version]

Setup Generate decryption key for opening authority
System parameters : [. . .]

Delegate Sign delegatee's verification key → warrant
Re-delegate Additionally forward received warrant(s)

Proxy-sign Sign message
$\boxed{\text{Encrypt}}$

- delegators' verification keys
- warrants
- signature on message

Prove correctness

Proxy signature :
- $\boxed{\text{ciphertexts}}$
- proofs

$$vk_0 \xrightarrow[\pi_1]{\boxed{\Sigma_1}} \boxed{vk_1} \xrightarrow[\pi_2]{\boxed{\Sigma_2}} \cdots \xrightarrow[\pi_n]{\boxed{\Sigma_n}} \boxed{vk_n} \xrightarrow[\pi_M]{\boxed{\Sigma_M}} M$$

Verify Verify proofs

Anonymous proxy signatures : Generic construction II

[simplified version]

Setup Generate decryption key for opening authority
System parameters : [. . .]

Delegate Sign delegatee's verification key → warrant
Re-delegate Additionally forward received warrant(s)

Proxy-sign Sign message
$\boxed{\text{Encrypt}}$

- delegators' verification keys
- warrants
- signature on message

Prove correctness

Proxy signature :
- $\boxed{\text{ciphertexts}}$
- proofs

$$vk_0 \xrightarrow[\pi_1]{\boxed{\Sigma_1}} \boxed{vk_1} \xrightarrow[\pi_2]{\boxed{\Sigma_2}} \cdots \xrightarrow[\pi_n]{\boxed{\Sigma_n}} \boxed{vk_n} \xrightarrow[\pi_M]{\boxed{\Sigma_M}} M$$

Verify Verify proofs

Open Decrypt ciphertext → verification keys

[simplified version]

Setup Generate decryption key for opening authority
System parameters : [. . .]

Delegate Sign delegatee's verification key → warrant
Re-delegate Additionally forward received warrant(s)

Proxy-sign Sign message

Showed theoretic feasability
but can we instantiate them practically ?

Prove correctness

Proxy signature :
- ciphertexts
- proofs

$$vk_0 \xrightarrow[\pi_1]{\boxed{\Sigma_1}} \boxed{vk_1} \xrightarrow[\pi_2]{\boxed{\Sigma_2}} \cdots \xrightarrow[\pi_n]{\boxed{\Sigma_n}} \boxed{vk_n} \xrightarrow[\pi_M]{\boxed{\Sigma_M}} M$$

Verify Verify proofs

Open Decrypt ciphertext → verification keys

Bilinear group : $(p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, G, H)$ with

Groups : $\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T$ cyclic groups of prime order p

Pairing : $e \colon \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ bilinear, ie

$e(X^a, Y^b) = e(X, Y)^{ab}$ for all $X \in \mathbb{G}_1; Y \in \mathbb{G}_2; a, b \in \mathbb{Z}$

Generators : $\mathbb{G}_1 = \langle G \rangle$, $\mathbb{G}_2 = \langle H \rangle$, $\mathbb{G}_T = \langle e(G, H) \rangle$

Bilinear group : $(p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, G, H)$ with

Groups : $\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T$ cyclic groups of prime order p

Pairing : $e\colon \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ bilinear, ie

$e(X^a, Y^b) = e(X, Y)^{ab}$ for all $X \in \mathbb{G}_1; Y \in \mathbb{G}_2; a, b \in \mathbb{Z}$

Generators : $\mathbb{G}_1 = \langle G \rangle$, $\mathbb{G}_2 = \langle H \rangle$, $\mathbb{G}_T = \langle e(G, H) \rangle$

Pairing-product equation (PPE)

over variables $X_1, \ldots, X_m \in \mathbb{G}_1$, $Y_1, \ldots, Y_n \in \mathbb{G}_2$

$$\prod_{i=j}^{n} e(A_j, Y_j) \prod_{i=1}^{m} e(X_i, B_i) \prod_{i=1}^{m} \prod_{j=1}^{n} e(X_i, Y_j)^{\gamma_{i,j}} = \mathbf{t} \ , \qquad \text{(E)}$$

defined by $A_i \in \mathbb{G}_1, B_i \in \mathbb{G}_2$, $\gamma_{i,j} \in \mathbb{Z}_p$ and $\mathbf{t} \in \mathbb{G}_T$

Pairing-product equation (PPE)

over variables $X_1, \ldots, X_m \in \mathbb{G}_1$, $Y_1, \ldots, Y_n \in \mathbb{G}_2$

$$\prod_{i=j}^{n} e(A_j, Y_j) \prod_{i=1}^{m} e(X_i, B_i) \prod_{i=1}^{m} \prod_{j=1}^{n} e(X_i, Y_j)^{\gamma_{i,j}} = \mathbf{t} \,, \tag{E}$$

defined by $A_i \in \mathbb{G}_1, B_i \in \mathbb{G}_2$, $\gamma_{i,j} \in \mathbb{Z}_p$ and $\mathbf{t} \in \mathbb{G}_T$

Pairing-product equation (PPE)

over variables $X_1, \ldots, X_m \in \mathbb{G}_1,\ Y_1, \ldots, Y_n \in \mathbb{G}_2$

$$\prod_{i=j}^{n} e(A_j, Y_j) \prod_{i=1}^{m} e(X_i, B_i) \prod_{i=1}^{m} \prod_{j=1}^{n} e(X_i, Y_j)^{\gamma_{i,j}} \;=\; \mathbf{t} \;, \qquad \text{(E)}$$

defined by $A_i \in \mathbb{G}_1, B_i \in \mathbb{G}_2,\ \gamma_{i,j} \in \mathbb{Z}_p$ and $\mathbf{t} \in \mathbb{G}_T$

Pairing-product equation (PPE)
over variables $X_1, \ldots, X_m \in \mathbb{G}_1,\ Y_1, \ldots, Y_n \in \mathbb{G}_2$

$$\prod_{i=j}^{n} e(A_j, Y_j) \prod_{i=1}^{m} e(X_i, B_i) \prod_{i=1}^{m}\prod_{j=1}^{n} e(X_i, Y_j)^{\gamma_{i,j}} = \mathbf{t}\ , \qquad \text{(E)}$$

defined by $A_i \in \mathbb{G}_1, B_i \in \mathbb{G}_2,\ \gamma_{i,j} \in \mathbb{Z}_p$ and $\mathbf{t} \in \mathbb{G}_T$

Pairing-product equation (PPE)
over variables $X_1, \ldots, X_m \in \mathbb{G}_1$, $Y_1, \ldots, Y_n \in \mathbb{G}_2$

$$\prod_{i=j}^{n} e(A_j, Y_j) \prod_{i=1}^{m} e(X_i, B_i) \prod_{i=1}^{m} \prod_{j=1}^{n} e(X_i, Y_j)^{\gamma_{i,j}} = \mathbf{t} \,, \qquad \text{(E)}$$

defined by $A_i \in \mathbb{G}_1, B_i \in \mathbb{G}_2$, $\gamma_{i,j} \in \mathbb{Z}_p$ and $\mathbf{t} \in \mathbb{G}_T$

Groth–Sahai proofs [GS08]
Efficient non-interactive zero-knowledge proofs :

1. Encrypt X_i's and Y_j's
2. Make proof π that encrypted values satisfy E w/o revealing anything else

Pairing-product equation (PPE)
over variables $X_1, \ldots, X_m \in \mathbb{G}_1$, $Y_1, \ldots, Y_n \in \mathbb{G}_2$

$$\prod_{j=1}^{n} e(A_j, Y_j) \prod_{i=1}^{m} e(X_i, B_i) \prod_{i=1}^{m}\prod_{j=1}^{n} e(X_i, Y_j)^{\gamma_{i,j}} = t_T \qquad \text{(E)}$$

Have a *proof system* for very specific language
but can we combine it with signatures ?

Groth–Sahai proofs [GS08]
Efficient non-interactive zero-knowledge proofs :

1. Encrypt X_i's and Y_j's
2. Make proof π that encrypted values satisfy E w/o revealing anything else

- Groth-Sahai proofs allow us to
  - encrypt group elements and to
  - prove that they satisfy PPEs

- Groth-Sahai proofs allow us to
  - encrypt group elements and to
  - prove that they satisfy PPEs

WANTED Signature scheme s.t.

- able to sign its own verification keys
- messages and signatures are group elements
- verification by PPE
- unforgeable (under chosen-message attacks)

- Groth-Sahai proofs allow us to
  - encrypt group elements and to
  - prove that they satisfy PPEs

WANTED Signature scheme s.t.

- able to sign its own verification keys
- messages and signatures are group elements
- verification by PPE
- unforgeable (under chosen-message attacks)

- Groth-Sahai proofs allow us to
  - encrypt group elements and to
  - prove that they satisfy PPEs

WANTED Signature scheme s.t.

- able to sign its own verification keys
- messages and signatures are group elements
- verification by PPE
- unforgeable (under chosen-message attacks)

- Groth-Sahai proofs allow us to
  - encrypt group elements and to
  - prove that they satisfy PPEs

$\boxed{\text{WANTED}}$ Signature scheme s.t.

- able to sign its own verification keys
- messages and signatures are group elements
- verification by PPE
- unforgeable (under chosen-message attacks)

- Groth-Sahai proofs allow us to
  - encrypt group elements and to
  - prove that they satisfy PPEs

WANTED Signature scheme s.t.

- able to sign its own verification keys
- messages and signatures are group elements
- verification by PPE
- unforgeable (under chosen-message attacks)

- Groth-Sahai proofs allow us to
  - encrypt group elements and to
  - prove that they satisfy PPEs

Signature scheme s.t.

$\boxed{\text{Automorphic signatures}}$

- able to sign its own verification keys
- messages and signatures are group elements
- verification by PPE
- unforgeable (under chosen-message attacks)

- Groth-Sahai proofs allow us to
  - encrypt group elements and to
  - prove that they satisfy PPEs

Signature scheme s.t.

Automorphic signatures

- able to sign its own verification keys
- messages and signatures are group elements
- verification by PPE
- unforgeable (under chosen-message attacks)

Combined with Groth-Sahai proofs :

- encrypt keys, messages, and signatures
- prove validity of encryptions

  $\Rightarrow$ verifiably encrypt certificate chain

Applications of automorphic signatures

- Efficient *anonymous proxy signatures*
- *Non-frameable group signatures* with concurrent join
- First efficient *round-optimal blind signatures*

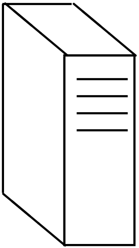
Applications of automorphic signatures

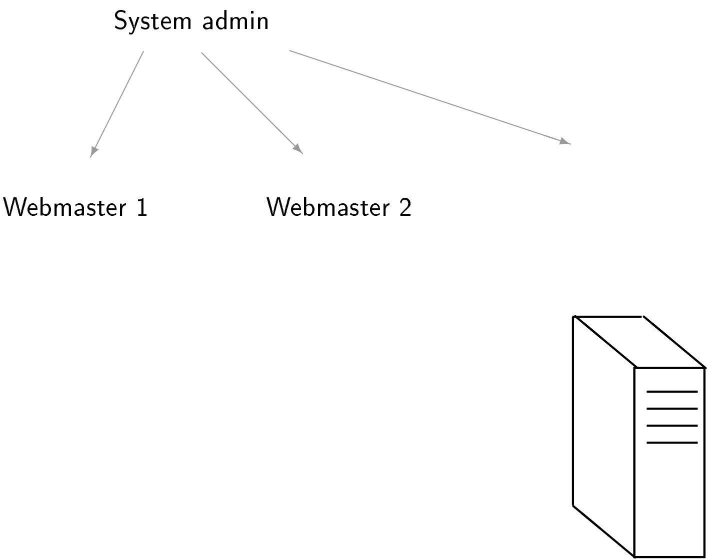
- Efficient *anonymous proxy signatures*
- *Non-frameable group signatures* with concurrent join
- First efficient *round-optimal blind signatures*
- Commuting signatures and verifiable encryption

Applications of automorphic signatures

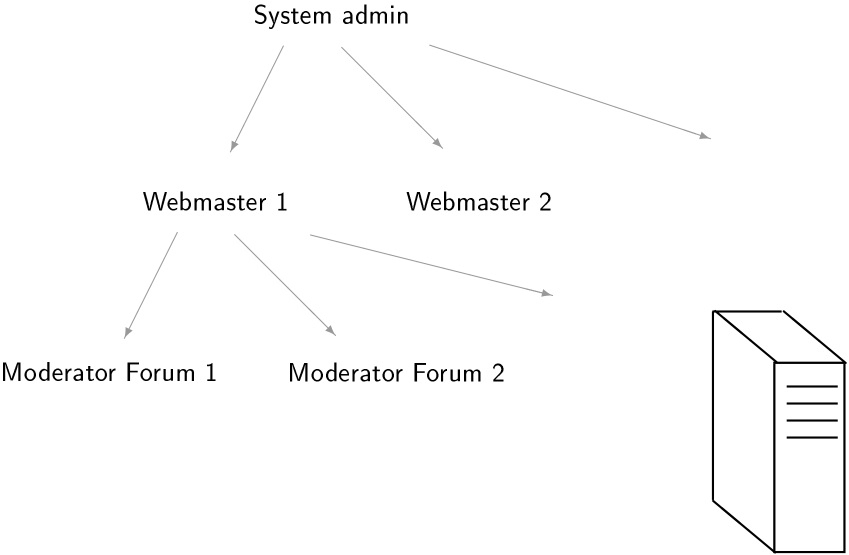
- Efficient *anonymous proxy signatures*
- *Non-frameable group signatures* with concurrent join
- First efficient *round-optimal blind signatures*
- Commuting signatures and verifiable encryption

System admin

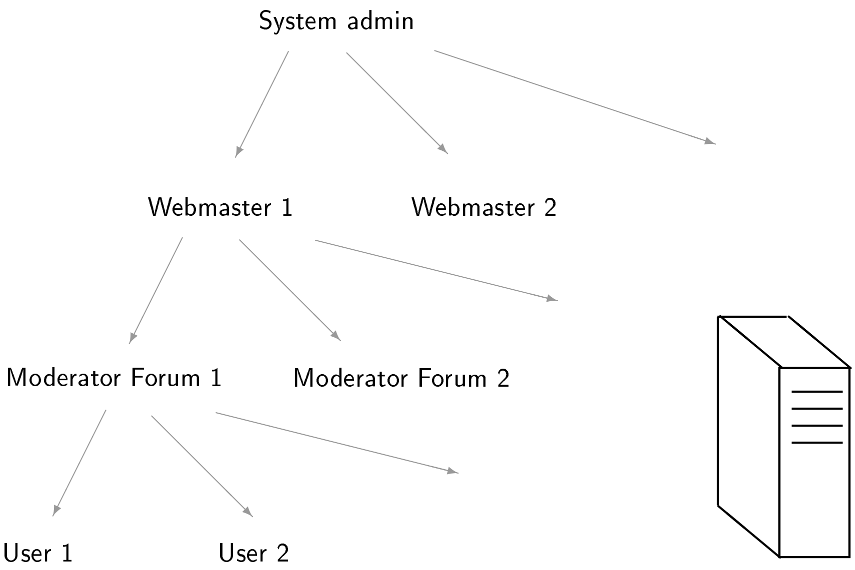



System admin
Webmaster 1
Webmaster 2

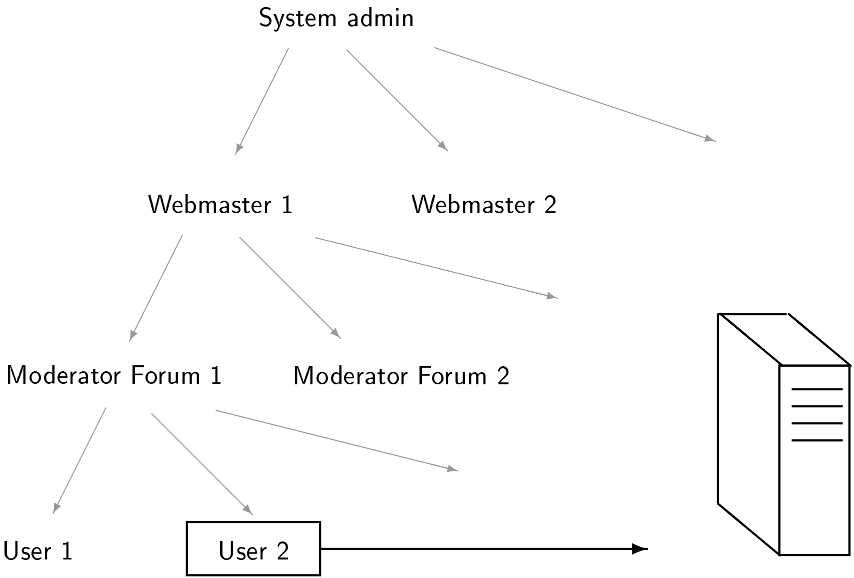
Motivation : Delegatable credentials



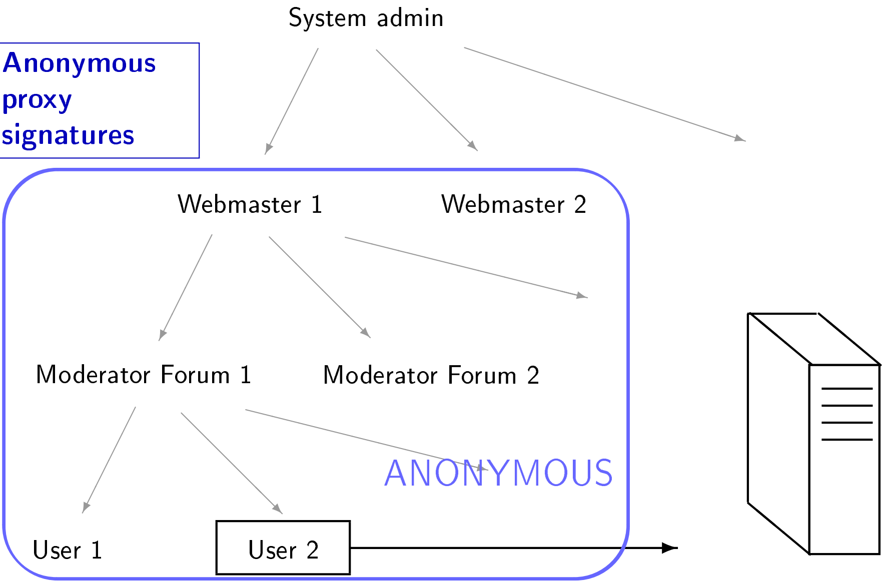
Motivation : Delegatable credentials

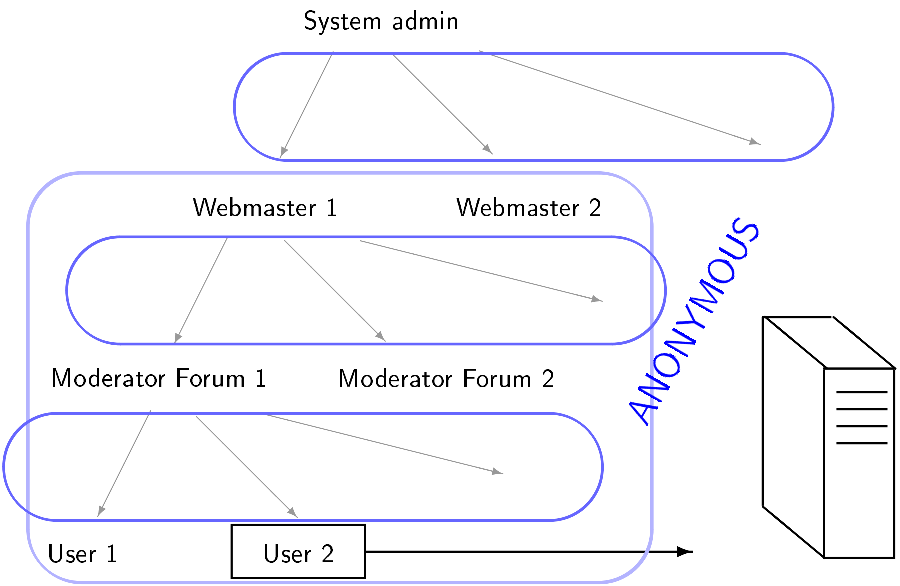


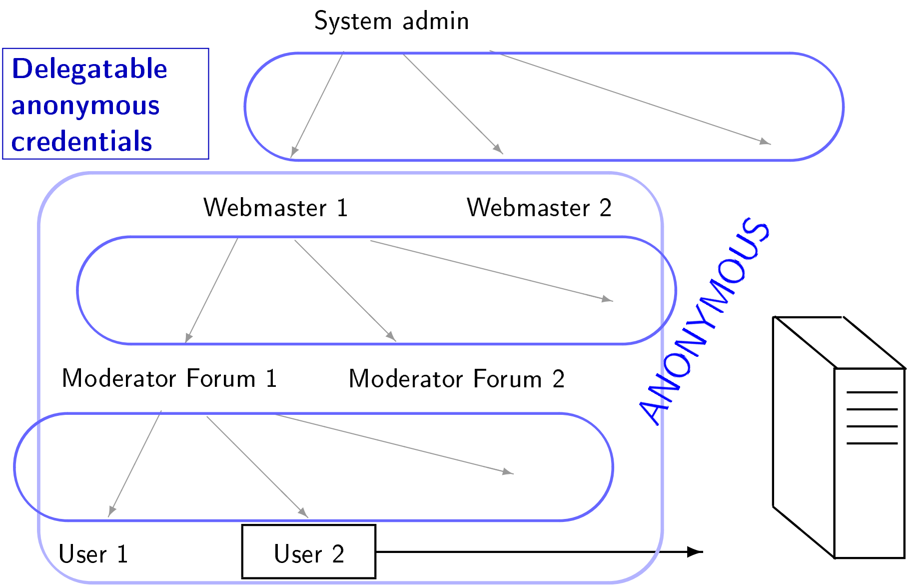
Motivation : Delegatable credentials



Anonymous proxy signatures




System admin
Webmaster 1
Webmaster 2
Moderator Forum 1
Moderator Forum 2
User 1
User 2
ANONYMOUS


System admin
Delegatable anonymous credentials
Webmaster 1
Webmaster 2
ANONYMOUS
Moderator Forum 1
Moderator Forum 2
User 1
User 2

Delegatable anonymous credentials [BCCKLS09]

- Users can *prove* to hold credential w/o revealing their identity
- Credentials can be issued/delegated and obtained anonymously

Delegatable anonymous credentials [BCCKLS09]

- Users can *prove* to hold credential w/o revealing their identity
- Credentials can be issued/delegated and obtained anonymously

BCCKLS model

- Each user holds a secret key and can
- . . . produce arbitrarily many (unlinkable) pseudonyms from it

Delegatable anonymous credentials [BCCKLS09]

- Users can *prove* to hold credential w/o revealing their identity
- Credentials can be issued/delegated and obtained anonymously

BCCKLS model

- Each user holds a secret key and can
- . . . produce arbitrarily many (unlinkable) pseudonyms from it
- . . . can publish pseudonym as *public key* for a credential
- . . . run *interactive protocol* to issue/delegate credentials to other users

Delegatable anonymous credentials [BCCKLS09]

- Users can *prove* to hold credential w/o revealing their identity
- Credentials can be issued/delegated and obtained anonymously

BCCKLS model

- Each user holds a secret key and can
- . . . produce arbitrarily many (unlinkable) pseudonyms from it
- . . . can publish pseudonym as *public key* for a credential
- . . . run *interactive protocol* to issue/delegate credentials to other users
- . . . prove to hold credentials for every pseudonym

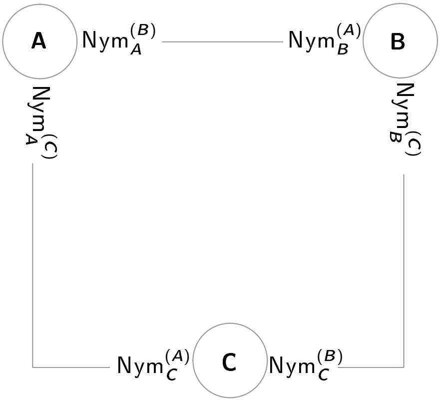
Delegatable anonymous credentials [BCCKLS09]

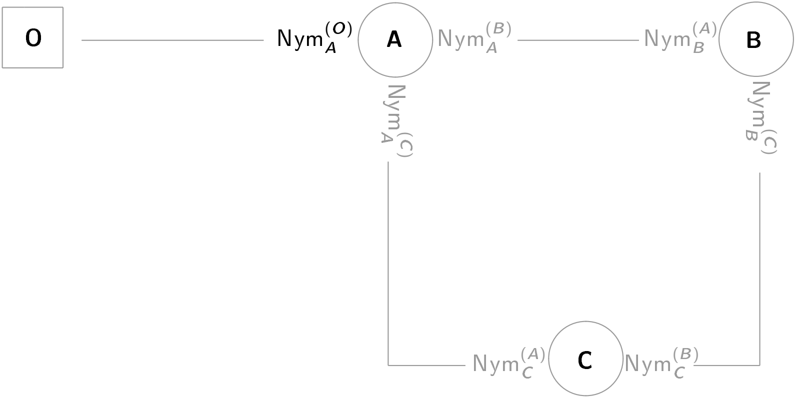
- Users can *prove* to hold credential w/o revealing their identity
- Credentials can be issued/delegated and obtained anonymously

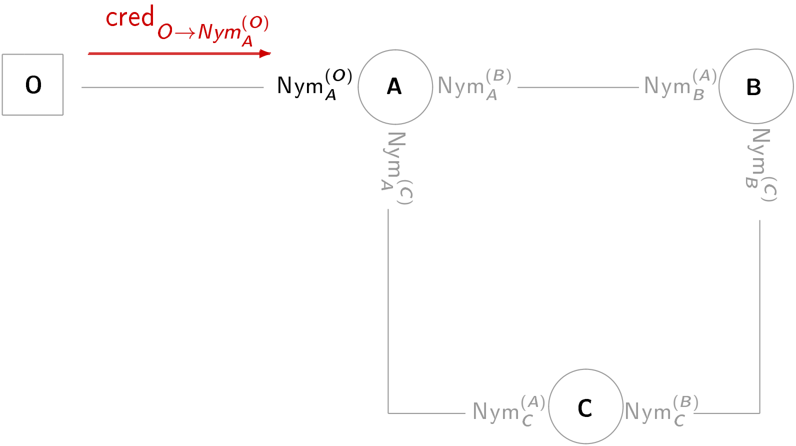
BCCKLS model

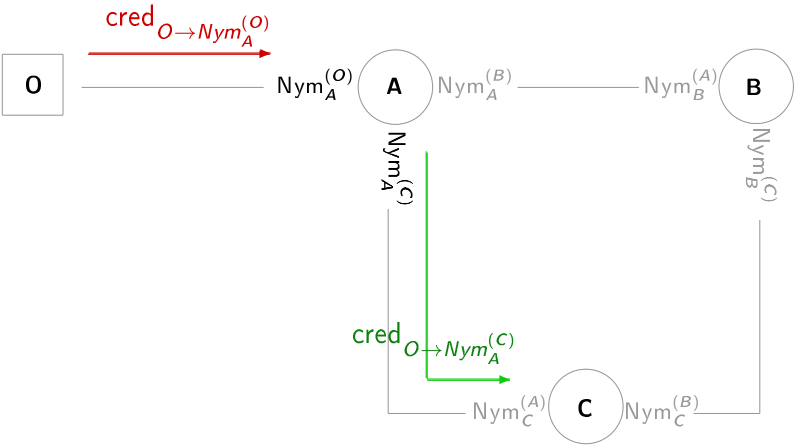
- Each user holds a secret key and can
- . . . produce arbitrarily many (unlinkable) pseudonyms from it
- . . . can publish pseudonym as *public key* for a credential
- . . . run *interactive protocol* to issue/delegate credentials to other users
- . . . prove to hold credentials for every pseudonym

Our scheme : *Non-interactive* issuing & delegation

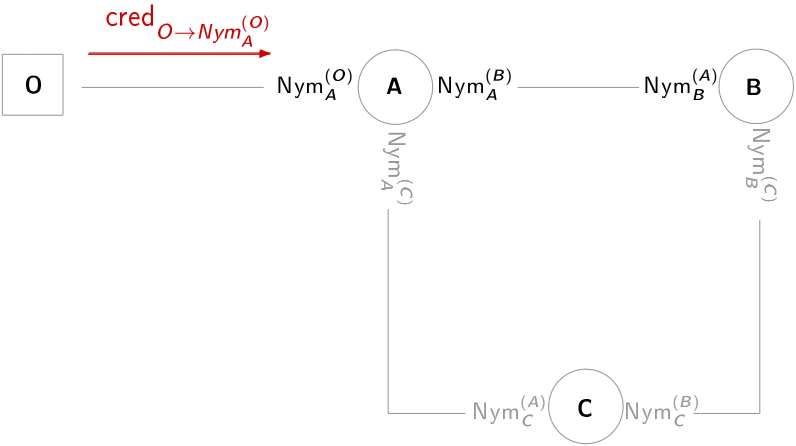

A
$\mathrm{Nym}_A^{(B)}$
$\mathrm{Nym}_B^{(A)}$
B
$\mathrm{Nym}_A^{(C)}$
$\mathrm{Nym}_B^{(C)}$
$\mathrm{Nym}_C^{(A)}$
C
$\mathrm{Nym}_C^{(B)}$


O
$\text{Nym}_A^{(O)}$
A
$\text{Nym}_A^{(B)}$
$\text{Nym}_B^{(A)}$
B
$\text{Nym}_A^{(C)}$
$\text{Nym}_B^{(C)}$
$\text{Nym}_C^{(A)}$
C
$\text{Nym}_C^{(B)}$

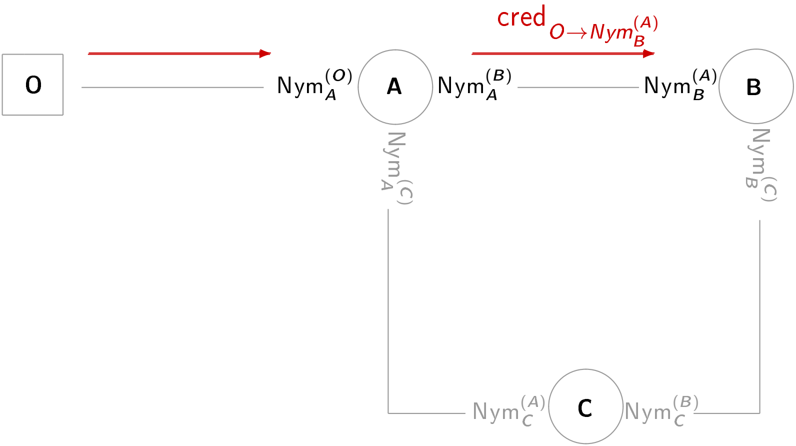

$\mathrm{cred}_{O \to Nym_A^{(O)}}$
O
$\mathrm{Nym}_A^{(O)}$
A
$\mathrm{Nym}_A^{(B)}$
$\mathrm{Nym}_B^{(A)}$
B
$\mathrm{Nym}_A^{(C)}$
$\mathrm{Nym}_B^{(C)}$
$\mathrm{Nym}_C^{(A)}$
C
$\mathrm{Nym}_C^{(B)}$

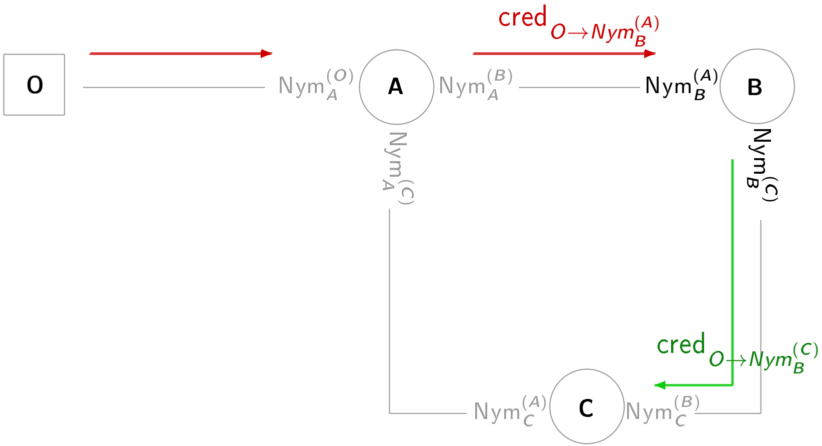


$\mathrm{cred}_{O \to Nym_A^{(O)}}$
O
$\mathrm{Nym}_A^{(O)}$
A
$\mathrm{Nym}_A^{(B)}$
$\mathrm{Nym}_B^{(A)}$
B
$\mathrm{Nym}_A^{(C)}$
$\mathrm{Nym}_B^{(C)}$
$\mathrm{cred}_{O \to Nym_A^{(C)}}$
$\mathrm{Nym}_C^{(A)}$
C
$\mathrm{Nym}_C^{(B)}$


$\mathsf{cred}_{O \to Nym_A^{(O)}}$
O
$\mathsf{Nym}_A^{(O)}$
A
$\mathsf{Nym}_A^{(B)}$
$\mathsf{Nym}_B^{(A)}$
B
$\mathsf{Nym}_A^{(C)}$
$\mathsf{Nym}_B^{(C)}$
$\mathsf{Nym}_C^{(A)}$
C
$\mathsf{Nym}_C^{(B)}$

Non-interactively delegatable anonymous credentials





$\mathsf{cred}_{O \to Nym_B^{(A)}}$
O
$\mathsf{Nym}_A^{(O)}$
A
$\mathsf{Nym}_A^{(B)}$
$\mathsf{Nym}_B^{(A)}$
B
$\mathsf{Nym}_A^{(C)}$
$\mathsf{Nym}_B^{(C)}$
$\mathsf{cred}_{O \to Nym_B^{(C)}}$
$\mathsf{Nym}_C^{(A)}$
C
$\mathsf{Nym}_C^{(B)}$

Commuting signatures and verifiable encryption I

- Signature $\quad M \xrightarrow{sk} \Sigma \quad$ Verification : vk, M, Σ

Commuting signatures and verifiable encryption I

- Signature $\quad M \xrightarrow{sk} \Sigma \qquad$ Verification : $vk,\ M,\ \Sigma$
- Verifiable encryption

$$vk, M, \Sigma \longrightarrow \begin{cases} \longrightarrow \boxed{\Sigma}, \widetilde{\pi} \\ \longrightarrow \boxed{M}, \bar{\pi} \\ \longrightarrow \boxed{M}, \boxed{\Sigma}, \pi \\ \longrightarrow \boxed{vk}, \boxed{M}, \boxed{\Sigma}, \widehat{\pi} \end{cases}$$

Verification : $vk,\ M,\ \boxed{\Sigma}, \widetilde{\pi}$

Verification : $vk, \boxed{M},\ \Sigma,\ \bar{\pi}$

Verification : $vk, \boxed{M}, \boxed{\Sigma}, \pi$

Verification : $\boxed{vk}, \boxed{M}, \boxed{\Sigma}, \widehat{\pi}$

- Signature $\quad M \xrightarrow{sk} \Sigma \qquad$ Verification : $vk,\ M,\ \Sigma$
- Verifiable encryption

$$vk, M, \Sigma \longrightarrow \begin{cases} \longrightarrow \boxed{\Sigma}, \widetilde{\pi} \\ \longrightarrow \boxed{M}, \bar{\pi} \\ \longrightarrow \boxed{M}, \boxed{\Sigma}, \pi \\ \longrightarrow \boxed{vk}, \boxed{M}, \boxed{\Sigma}, \widehat{\pi} \end{cases}$$

Verification : $vk,\ M,\ \boxed{\Sigma}, \widetilde{\pi}$

Verification : $vk, \boxed{M},\ \Sigma,\ \bar{\pi}$

Verification : $vk, \boxed{M}, \boxed{\Sigma}, \pi$

Verification : $\boxed{vk}, \boxed{M}, \boxed{\Sigma}, \widehat{\pi}$

- Commuting signature and verifiable encryption

Proof adaptation : $\qquad \left.\begin{matrix} \widetilde{\pi} \\ \bar{\pi} \end{matrix}\right\} \longleftrightarrow \pi \longleftrightarrow \widehat{\pi}$

- Signature $\quad M \xrightarrow{sk} \Sigma \quad$ Verification : $vk,\ M,\ \Sigma$
- Verifiable encryption

$$vk, M, \Sigma \longrightarrow \begin{cases} \longrightarrow \boxed{\Sigma}, \widetilde{\pi} \\ \longrightarrow \boxed{M}, \bar{\pi} \\ \longrightarrow \boxed{M}, \boxed{\Sigma}, \pi \\ \longrightarrow \boxed{vk}, \boxed{M}, \boxed{\Sigma}, \widehat{\pi} \end{cases}$$

Verification : $vk,\ M,\ \boxed{\Sigma}, \widetilde{\pi}$

Verification : $vk, \boxed{M}, \Sigma, \bar{\pi}$

Verification : $vk, \boxed{M}, \boxed{\Sigma}, \pi$

Verification : $\boxed{vk}, \boxed{M}, \boxed{\Sigma}, \widehat{\pi}$

- Commuting signature and verifiable encryption

Proof adaptation : $\left.\begin{matrix} \widetilde{\pi} \\ \bar{\pi} \end{matrix}\right\} \longleftrightarrow \pi \longleftrightarrow \widehat{\pi}$

Sign M given $\boxed{M}$: $\quad \boxed{M} \xrightarrow{sk} \Sigma$

- Signature $\quad M \xrightarrow{sk} \Sigma \qquad$ Verification : $vk,\ M,\ \Sigma$
- Verifiable encryption

$$vk, M, \Sigma \longrightarrow \begin{cases} \longrightarrow \boxed{\Sigma}, \widetilde{\pi} \\ \longrightarrow \boxed{M}, \bar{\pi} \\ \longrightarrow \boxed{M}, \boxed{\Sigma}, \pi \\ \longrightarrow \boxed{vk}, \boxed{M}, \boxed{\Sigma}, \widehat{\pi} \end{cases}$$

Verification : $vk,\ M,\ \boxed{\Sigma}, \widetilde{\pi}$

Verification : $vk,\ \boxed{M},\ \Sigma,\ \bar{\pi}$

Verification : $vk,\ \boxed{M},\ \boxed{\Sigma}, \pi$

Verification : $\boxed{vk}, \boxed{M}, \boxed{\Sigma}, \widehat{\pi}$

- Commuting signature and verifiable encryption

Proof adaptation : $\quad \left.\begin{matrix} \widetilde{\pi} \\ \bar{\pi} \end{matrix}\right\} \longleftrightarrow \pi \longleftrightarrow \widehat{\pi}$

Sign M given $\boxed{M}$: $\quad \boxed{M} \ \cancel{\xrightarrow{sk}}\ \Sigma$

- Signature $\quad M \xrightarrow{sk} \Sigma \qquad$ Verification : vk, M, Σ
- Verifiable encryption

$$vk, M, \Sigma \longrightarrow \begin{cases} \longrightarrow \boxed{\Sigma}, \widetilde{\pi} \\ \longrightarrow \boxed{M}, \bar{\pi} \\ \longrightarrow \boxed{M}, \boxed{\Sigma}, \pi \\ \longrightarrow \boxed{vk}, \boxed{M}, \boxed{\Sigma}, \widehat{\pi} \end{cases}$$

Verification : $vk, M, \boxed{\Sigma}, \widetilde{\pi}$

Verification : $vk, \boxed{M}, \Sigma, \bar{\pi}$

Verification : $vk, \boxed{M}, \boxed{\Sigma}, \pi$

Verification : $\boxed{vk}, \boxed{M}, \boxed{\Sigma}, \widehat{\pi}$

- Commuting signature and verifiable encryption

Proof adaptation : $\left.\begin{matrix} \widetilde{\pi} \\ \bar{\pi} \end{matrix}\right\} \longleftrightarrow \pi \longleftrightarrow \widehat{\pi}$

Sign M given $\boxed{M}$: $\quad \boxed{M} \xrightarrow{sk} \boxed{\Sigma}, \pi \qquad$ Verification : $vk, \boxed{M}, \boxed{\Sigma}, \pi$

- Signature $\quad M \xrightarrow{sk} \Sigma \qquad$ Verification : $vk,\ M,\ \Sigma$
- Verifiable encryption

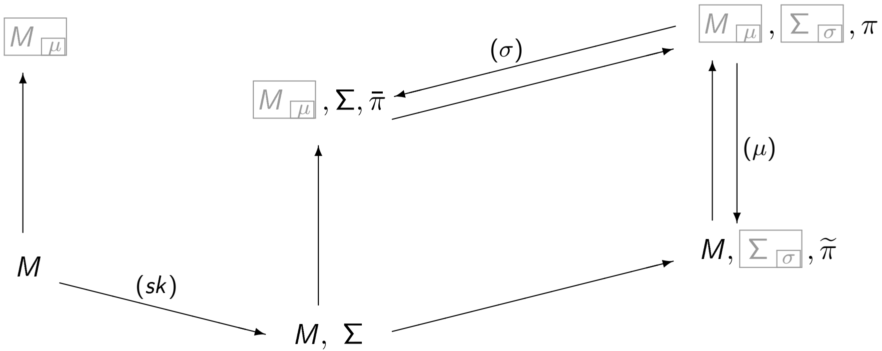
$$vk, M, \Sigma \longrightarrow \begin{cases} \longrightarrow \boxed{\Sigma}, \widetilde{\pi} & \text{Verification} :\ vk,\ M,\ \boxed{\Sigma}, \widetilde{\pi} \\ \longrightarrow \boxed{M}, \bar{\pi} & \text{Verification} :\ vk,\ \boxed{M},\ \Sigma,\ \bar{\pi} \\ \longrightarrow \boxed{M}, \boxed{\Sigma}, \pi & \text{Verification} :\ \boxed{vk, \boxed{M}, \boxed{\Sigma}, \pi} \\ \longrightarrow \boxed{vk}, \boxed{M}, \boxed{\Sigma}, \widehat{\pi} & \text{Verification} :\ \boxed{vk}, \boxed{M}, \boxed{\Sigma}, \widehat{\pi} \end{cases}$$

- Commuting signature and verifiable encryption

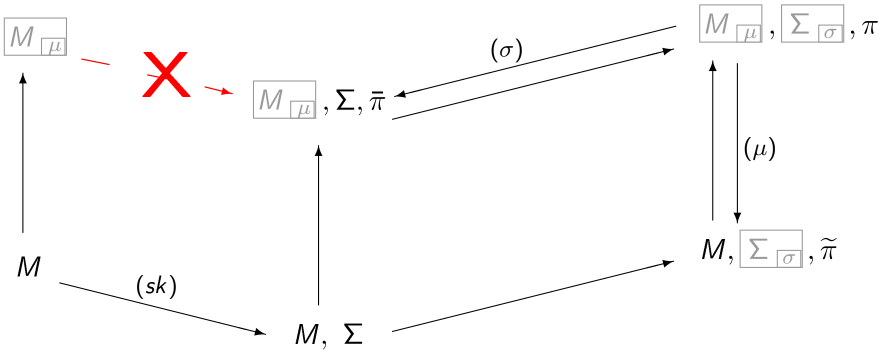
Sign plaintext then encrypt $\Longleftrightarrow$ encrypt then sign plaintext

Sign M given $\boxed{M}$: $\quad \boxed{M} \xrightarrow{sk} \boxed{\Sigma}, \pi \qquad$ Verification : $\boxed{vk, \boxed{M}, \boxed{\Sigma}, \pi}$

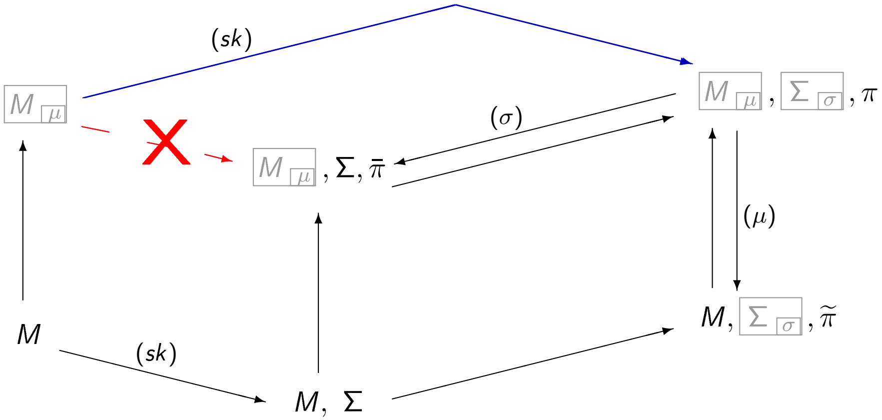
$$
\begin{tikzcd}
& \boxed{M_{\boxed{\mu}}}, \boxed{\Sigma_{\boxed{\sigma}}}, \pi \arrow[dl, shift right, "(\sigma)"'] \arrow[d, shift left, "(\mu)"] \\
\boxed{M_{\boxed{\mu}}}, \Sigma, \bar{\pi} \arrow[ur, shift right] & M, \boxed{\Sigma_{\boxed{\sigma}}}, \widetilde{\pi} \arrow[u, shift left] \\
M, \Sigma \arrow[u] \arrow[ur] &
\end{tikzcd}
$$


$\boxed{M_{\boxed{\mu}}}$
$\boxed{M_{\boxed{\mu}}}, \Sigma, \bar{\pi}$
$\boxed{M_{\boxed{\mu}}}, \boxed{\Sigma_{\boxed{\sigma}}}, \pi$
(σ)
(μ)
M
(sk)
M, Σ
$M, \boxed{\Sigma_{\boxed{\sigma}}}, \widetilde{\pi}$

Commuting signatures and verifiable encryption II



Commuting signatures and verifiable encryption II



In a nutshell

- Pseudonym : encryption of user verification key
- Credential : verifiably encrypted signature
- Non-interactive delegation : commuting signature

- Delegation of signing rights

Black-box instantiation of NIDAC

- Delegation of signing rights $\quad vk_0 \xrightarrow{\Sigma_1} vk_1$

Black-box instantiation of NIDAC

- Delegation of signing rights $\quad vk_0 \xrightarrow{\Sigma_1} vk_1 \xrightarrow{\Sigma_2} \cdots \xrightarrow{\Sigma_n} vk_n$

Black-box instantiation of NIDAC

- Delegation of signing rights

$$vk_0 \xrightarrow{\Sigma_1} vk_1 \xrightarrow{\Sigma_2} \cdots \xrightarrow{\Sigma_n} vk_n$$

Signatures

credential

- Delegation of signing rights

  Signatures

$$vk_0 \xrightarrow{\Sigma_1} vk_1 \xrightarrow{\Sigma_2} \cdots \xrightarrow{\Sigma_n} vk_n$$

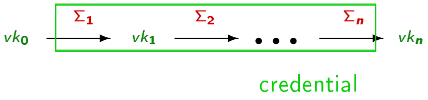
credential

- Anonymous show

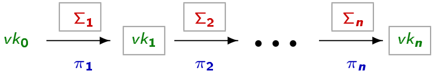
$$vk_0 \xrightarrow{\Sigma_1} vk_1 \xrightarrow{\Sigma_2} \cdots \xrightarrow{\Sigma_n} vk_n$$

- Delegation of signing rights

  Signatures



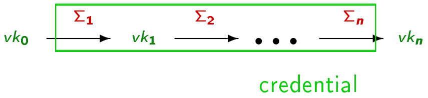
- Anonymous show



Black-box instantiation of NIDAC

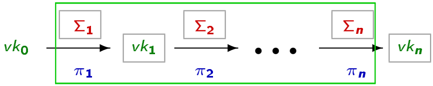
- Delegation of signing rights

  Signatures



- Anonymous show

  Verifiable encryption



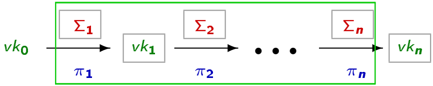
- Delegation of signing rights

  Signatures

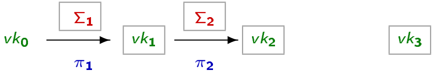
credential

- Anonymous show

  Verifiable encryption



- Anonymous delegation

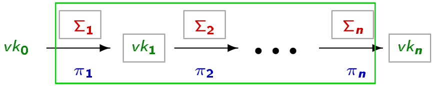


- Delegation of signing rights

Signatures

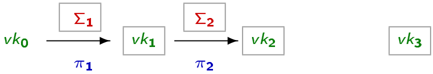
credential

- Anonymous show



Verifiable encryption

- Anonymous delegation



Commuting signatures

- Sign encrypted value $\boxed{vk_3}$ $\Rightarrow$ (vk_2, $\boxed{\Sigma_3}$, $\boxed{vk_3}$, π'_3)

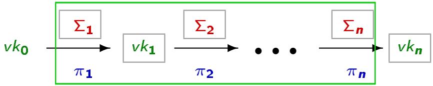
- Delegation of signing rights

  Signatures

credential

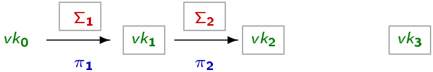
- Anonymous show

  Verifiable encryption



- Anonymous delegation

  Commuting signatures



- Sign encrypted value $\boxed{vk_3}$ $\Rightarrow$ $(vk_2, \boxed{\Sigma_3}, \boxed{vk_3}, \pi'_3)$
- Adapt proof for $\boxed{vk_2}$ $\Rightarrow$ $(\boxed{vk_2}, \boxed{\Sigma_3}, \boxed{vk_3}, \pi_3)$

- Delegation of signing rights

  Signatures

$$vk_0 \xrightarrow{\Sigma_1} vk_1 \xrightarrow{\Sigma_2} \bullet\bullet\bullet \xrightarrow{\Sigma_n} vk_n$$

credential

- Anonymous show

  Verifiable encryption

$$vk_0 \xrightarrow[\pi_1]{\boxed{\Sigma_1}} \boxed{vk_1} \xrightarrow[\pi_2]{\boxed{\Sigma_2}} \bullet\bullet\bullet \xrightarrow[\pi_n]{\boxed{\Sigma_n}} \boxed{vk_n}$$

- Anonymous delegation

  Commuting signatures

$$vk_0 \xrightarrow[\pi_1]{\boxed{\Sigma_1}} \boxed{vk_1} \xrightarrow[\pi_2]{\boxed{\Sigma_2}} \boxed{vk_2} \qquad \boxed{vk_3}$$

- Sign encrypted value $\boxed{vk_3}$ $\Rightarrow (vk_2 , \boxed{\Sigma_3}, \boxed{vk_3}, \pi'_3)$
- Adapt proof for $\boxed{vk_2}$ $\Rightarrow (\boxed{vk_2}, \boxed{\Sigma_3}, \boxed{vk_3}, \pi_3)$

Send credential $\left(\underset{\pi_1}{\boxed{\Sigma_1}}, \boxed{vk_1}, \underset{\pi_2}{\boxed{\Sigma_2}}, \boxed{vk_2}, \underset{\pi_3}{\boxed{\Sigma_3}} \right)$

- Groth-Sahai (GS) proofs
  (shown to be randomizable [BCCKLS09])
- Automorphic signatures
  (allow signing of verification keys)

- Groth-Sahai (GS) proofs
  (shown to be randomizable [BCCKLS09])
- Automorphic signatures
  (allow signing of verification keys)

GS proofs + automorphic signatures = verifiably encrypted signatures

- Groth-Sahai (GS) proofs
  (shown to be randomizable [BCCKLS09])
- Automorphic signatures
  (allow signing of verification keys)

GS proofs + automorphic signatures = verifiably encrypted signatures

Algebraic properties of GS proofs $\Longrightarrow$ instantiation of new functionalities

- Groth-Sahai (GS) proofs
  (shown to be randomizable [BCCKLS09])
- Automorphic signatures
  (allow signing of verification keys)

GS proofs + automorphic signatures = verifiably encrypted signatures

Algebraic properties of GS proofs $\Longrightarrow$ instantiation of new functionalities

- Proof adaptation $\left.\begin{matrix}\widetilde{\pi}\\ \bar{\pi}\end{matrix}\right\} \longleftrightarrow \pi \longleftrightarrow \widehat{\pi}$
- Sign encrypted messages : $\boxed{M} \xrightarrow{sk} \boxed{\Sigma}, \pi$ (where Σ is signature on M)

Properties of Groth-Sahai proofs

$$\pi : \quad e(A_1, \boxed{Y_1}) \cdots e(\, A_n, \boxed{Y_n}) \cdot \cdots \cdots e(\boxed{X_i}, B_i) \cdot \cdots \cdots e(\boxed{X_m}, \boxed{Y_n})^{\gamma_{m,n}} \quad = \quad \mathbf{t}$$

Independence

Proofs do not depend on $\mathbf{t}$

Properties of Groth-Sahai proofs

$$\pi : \quad e(A_1, \boxed{Y_1}) \cdots e(\,A_n, \boxed{Y_n}) \cdots\cdots e(\boxed{X_i}, B_i) \quad = \quad \mathbf{t}$$

Independence

Proofs do not depend on $\mathbf{t}$

Some proofs ($\gamma_{ij} = 0$) do not depend on encrypted values

$$\pi : \quad e(A_1, \boxed{Y_1}) \cdots e(\, A_n, \boxed{Y_n}) \cdots\cdots e(\boxed{X_i}, B_i) \cdots\cdots e(\boxed{X_m}, \boxed{Y_n})^{\gamma_{m,n}} \quad = \quad \mathbf{t}$$

Independence

Proofs do not depend on $\mathbf{t}$

Some proofs ($\gamma_{ij} = 0$) do not depend on encrypted values

Adapting

Proofs can be adapted when constants are turned into variables or vice versa

Properties of Groth-Sahai proofs

$$\pi : \quad e(A_1, \boxed{Y_1}) \cdots e(\boxed{A_n}, \boxed{Y_n}) \cdots\cdots e(\boxed{X_i}, B_i) \cdots\cdots e(\boxed{X_m}, \boxed{Y_n})^{\gamma_{m,n}} \quad = \quad \mathbf{t}$$

Independence

Proofs do not depend on $\mathbf{t}$

Some proofs ($\gamma_{ij} = 0$) do not depend on encrypted values

Adapting

Proofs can be adapted when constants are turned into variables or vice versa

Properties of Groth-Sahai proofs

$$\pi: \quad e(A_1, \boxed{Y_1}) \cdots e(\, A_n, \boxed{Y_n}) \cdots\cdots e(\boxed{X_i}, B_i) \cdots\cdots e(\boxed{X_m}, \boxed{Y_n})^{\gamma_{m,n}} \quad = \quad \mathbf{t}$$

Independence

Proofs do not depend on $\mathbf{t}$

Some proofs ($\gamma_{ij} = 0$) do not depend on encrypted values

Adapting

Proofs can be adapted when constants are turned into variables or vice versa

$$\pi: \quad e(A_1, \boxed{Y_1}) \cdots e(\ A_n, Y_n\ ) \cdots\cdots e(\boxed{X_i}, B_i) \cdots\cdots e(\boxed{X_m}, \boxed{Y_n})^{\gamma_{m,n}} \quad = \quad \mathbf{t}$$

Independence

Proofs do not depend on $\mathbf{t}$

Some proofs ($\gamma_{ij} = 0$) do not depend on encrypted values

Adapting

Proofs can be adapted when constants are turned into variables or vice versa

$$\pi: \qquad e(A_1, \boxed{Y_1}) \cdots e(\ A_n, \boxed{Y_n}) \cdots\cdots e(\boxed{X_i}, B_i) \cdots\cdots e(\boxed{X_m}, \boxed{Y_n})^{\gamma_{m,n}} \quad = \quad \mathbf{t}$$

Independence

Proofs do not depend on $\mathbf{t}$

Some proofs ($\gamma_{ij} = 0$) do not depend on encrypted values

Adapting

Proofs can be adapted when constants are turned into variables or vice versa

$$\pi: \quad e(A_1, \boxed{Y_1}) \cdots e(\ A_n, \boxed{Y_n}) \cdots\cdots e(\boxed{X_i}, B_i) \cdots\cdots e(\boxed{X_m}, \boxed{Y_n})^{\gamma_{m,n}} = \mathbf{t}$$

$$\pi': \quad e(A'_1, \boxed{Y'_1}) \cdots e(A'_n, \boxed{Y'_n}) \cdots\cdots e(\boxed{X'_i}, B'_i) \cdots\cdots e(\boxed{X'_m}, \boxed{Y'_n})^{\gamma'_{m,n}} = \mathbf{t}'$$

Independence

Proofs do not depend on $\mathbf{t}$

Some proofs ($\gamma_{ij} = 0$) do not depend on encrypted values

Adapting

Proofs can be adapted when constants are turned into variables or vice versa

Homomorphic

The product of 2 proofs is a proof for the product of the 2 equations

$$\begin{array}{lllr}
\pi & e(A_1, \boxed{Y_1}) \cdots e(A_n, \boxed{Y_n}) \cdots\cdots e(\boxed{X_i}, B_i) \cdots\cdots e(\boxed{X_m}, \boxed{Y_n})^{\gamma_{m,n}} & & \mathbf{t} \\
\bullet\ : & \bullet & = & \bullet \\
\pi' & e(A'_1, \boxed{Y'_1}) \cdots e(A'_n, \boxed{Y'_n}) \cdots\cdots e(\boxed{X'_i}, B'_i) \cdots\cdots e(\boxed{X'_m}, \boxed{Y'_n})^{\gamma'_{m,n}} & & \mathbf{t}'
\end{array}$$

Independence

Proofs do not depend on $\mathbf{t}$

Some proofs ($\gamma_{ij} = 0$) do not depend on encrypted values

Adapting

Proofs can be adapted when constants are turned into variables or vice versa

Homomorphic

The product of 2 proofs is a proof for the product of the 2 equations

Conclusion

New primitives

- Automorphic signatures (First efficient "Groth-Sahai compatible" signatures)
- Commuting signatures (Toolbox for privacy-preserving primitives)

Conclusion

New primitives

- Automorphic signatures (First efficient "Groth-Sahai compatible" signatures)
- Commuting signatures (Toolbox for privacy-preserving primitives)

Applications

- First efficient *anonymous proxy signatures*
- First efficient *round-optimal blind signatures*
- First *anonymous credentials* that are non-interactively delegatable

& no more complex 2-party protocols ; size halved

Conclusion

New primitives

- Automorphic signatures (First efficient "Groth-Sahai compatible" signatures)
- Commuting signatures (Toolbox for privacy-preserving primitives)

Applications

- First efficient *anonymous proxy signatures*
- First efficient *round-optimal blind signatures*
- First *anonymous credentials* that are non-interactively delegatable

  & no more complex 2-party protocols ; size halved
- Receipt-free e-voting [BFPV11]
- Fully anonymous transferable e-cash [BCFGST11]

Thank you ! ☺