Automorphic Signatures in Bilinear Groups

Georg Fuchsbauer

École normale supérieure

UCL, 23.03.2010
1 Motivation: Anonymous Proxy Signatures

2 Groth-Sahai Witness-Indistinguishable Proofs

3 Automorphic Signatures
Motivation: Anonymous Proxy Signatures

Groth-Sahai Witness-Indistinguishable Proofs

Automorphic Signatures
Anonymity

All intermediate delegators and the proxy signer remain anonymous.
Anonymous Consecutive Delegation of Signing Rights

Delegation
A delegator delegates his signing rights to a proxy signer (or delegatee) who can then sign on the delegator’s behalf.

Consecutiveness
A delegatee may re-delegate the received signing rights ⇒ intermediate delegators

Anonymity
All intermediate delegators and the proxy signer remain anonymous.
F, Pointcheval: Anonymous Proxy Signatures [SCN’08]

**Delegation**
A delegator delegates his signing rights to a proxy signer (or delegatee) who can then sign on the delegator’s behalf.

**Consecutiveness**
A delegatee may re-delegate the received signing rights \( \Rightarrow \) intermediate delegators.

**Anonymity**
All intermediate delegators and the proxy signer remain anonymous.
#### Delegation

An **delegatee** may **re-delegate** the received signing rights to **intermediate delegators** for further use.

#### Consecutiveness

After verifying a proxy signature one knows that someone entitled signed but nothing more.

#### Anonymity

All intermediate delegators and the proxy signer remain **anonymous**.
Application: GRID computing

User authenticates herself and starts process which needs to authenticate to resources / start subprocesses

⇒ Delegation and re-delegation of signing rights

No need to know that it was not the user herself to be authenticated

Relation to Other Primitives

Anonymous proxy signatures are a generalization of

- Proxy signatures (consecutive delegation) formalized by [BPW03]
- (Dynamic) group signatures (anonymity) formalized by [BSZ05]

and satisfy the respective security notions.
Application: GRID computing

User authenticates herself and starts process which needs to authenticate to resources / start subprocesses
⇒ Delegation and re-delegation of signing rights

No need to know that it was not the user herself to be authenticated

Relation to Other Primitives

Anonymous proxy signatures are a generalization of

- **Proxy signatures** (consecutive delegation)
  formalized by [BPW03]

- **(Dynamic) group signatures** (anonymity)
  formalized by [BSZ05]

and satisfy the respective security notions.
Application: GRID computing

User authenticates herself and starts process which needs to authenticate to resources / start subprocesses
⇒ Delegation and re-delegation of signing rights
No need to know that it was not the user herself to be authenticated

Relation to Other Primitives

Anonymous proxy signatures are a generalization of

- **Proxy signatures** (consecutive delegation)
  formalized by [BPW03]

- **(Dynamic) group signatures** (anonymity)
  formalized by [BSZ05]

and satisfy the respective security notions.

- more recently: **Delegatable Anonymous Credentials** [BCCKLS09]
(Dynamic) Group Signatures

Group public key: $pk$

Verification: $\text{Verify}(pk, \text{msg}, \sigma) = 1$
Proxy Signatures

Delegator \((pk_D)\)

delegate

Delegatee/Signer

\[\text{Verify}(pk_D, \text{msg}, \sigma)\]

\[\sigma\]
Proxy Signatures, Consecutive Delegations

Delegator \((pk_D)\)

Delegator 2

Delegator 3

Proxy Signer

\(\sigma\)
Proxy Signatures, Consecutive Delegations

Delegator \((pk_D)\)

Delegator 2

Delegator 3

ANONYMOUS

Proxy Signer

\(\sigma\)
Proxy Signatures, Consecutive Delegations

Delegator \((p_k_D)\)

Delegator 2

Delegator 3

ANONYMOUS

Proxy Signer

Opener \((ok)\)

\(\sigma\)

open
$1^\lambda \rightarrow \text{Setup} \rightarrow pp, ik, ok$
Algorithms of Anonymous Proxy Signature Scheme

1^\lambda \rightarrow \text{Setup} \rightarrow pp, ik, ok
Algorithms of Anonymous Proxy Signature Scheme

$\lambda \rightarrow \text{Setup} \rightarrow pp, ik, ok$

$sk_x, pk_y \rightarrow \text{Del} \rightarrow \text{warr}_{x \rightarrow y}$
Algorithms of Anonymous Proxy Signature Scheme

\[
\begin{array}{c}
\text{Issuer (} ik \text{)} \quad \cdots \quad \text{Reg} \\
\quad pk \quad \cdots \quad \text{User} \\
\end{array}
\]

\[
\begin{align*}
1^\lambda & \quad \rightarrow \quad \text{Setup} \quad \rightarrow \quad pp, ik, ok \\
\sk_x, \left[ \text{warr}_{\xrightarrow{x}} \right] \pk_y & \quad \rightarrow \quad \text{Del} \quad \rightarrow \quad \text{warr}_{\xrightarrow{\cdot}} x \xrightarrow{\cdot} y \\
\end{align*}
\]
Algorithms of Anonymous Proxy Signature Scheme

Issuer (\(ik\)) \(\xrightarrow{\ldots} pk\) Reg \(\xleftarrow{\ldots} pk, sk\) User

\[1^\lambda \rightarrow \text{Setup} \rightarrow pp, ik, ok\]
\[sk_x, [\text{warr}_{\rightarrow x},] pk_y \rightarrow \text{Del} \rightarrow \text{warr}_{\rightarrow x}^{\rightarrow y}\]
\[sk_y, \text{warr}_{x\rightarrow \ldots \rightarrow y}, M \rightarrow \text{PSig} \rightarrow \sigma\]
Algorithms of Anonymous Proxy Signature Scheme

Issuer (ik) \[\cdots\] Reg \[\cdots\] User

\[pk\] \[pk, sk\]

\[1^\lambda \rightarrow \text{Setup} \rightarrow pp, ik, ok\]

\[sk_x, [warr\rightarrow_x,] \rightarrow pk_y \rightarrow \text{Del} \rightarrow warr[\rightarrow]_{x\rightarrow y}\]

\[sk_y, warr_{x\rightarrow \cdots \rightarrow y}, M \rightarrow \text{PSig} \rightarrow \sigma\]

\[pk_x, M, \sigma \rightarrow \text{PVer} \rightarrow b \in \{0, 1\}\]
Algorithms of Anonymous Proxy Signature Scheme

\[ 1^\lambda \rightarrow \text{Setup} \rightarrow pp, ik, ok \]

\[ sk_x, [\text{warr}_x \rightarrow y] \rightarrow \text{Del} \rightarrow \text{warr}_{x \rightarrow y} \]

\[ sk_y, \text{warr}_x \rightarrow \ldots \rightarrow y, M \rightarrow \text{PSig} \rightarrow \sigma \]

\[ pk_x, M, \sigma \rightarrow \text{PVer} \rightarrow b \in \{0, 1\} \]

\[ ok, M, \sigma \rightarrow \text{Open} \rightarrow \text{a list of users or \( \perp \) (failure)} \]
## Security

<table>
<thead>
<tr>
<th>Security Feature</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Anonymity</strong></td>
<td>Intermediate delegators and proxy signer remain anonymous</td>
</tr>
<tr>
<td><strong>Traceability</strong></td>
<td>Every valid signature can be traced to its intermediate delegators and proxy signer</td>
</tr>
<tr>
<td><strong>Non-Frameability</strong></td>
<td>No one can produce a signature that, when opened, wrongfully reveals a delegator or signer</td>
</tr>
</tbody>
</table>
# Security for Anonymous Proxy Signatures

## Security

<table>
<thead>
<tr>
<th>Anonymity</th>
<th>Intermediate delegators and proxy signer remain anonymous</th>
</tr>
</thead>
<tbody>
<tr>
<td>Traceability</td>
<td>Every valid signature can be traced to its intermediate delegates and proxy signer</td>
</tr>
<tr>
<td>Non-Frameability</td>
<td>No one can produce a signature that, when opened, wrongfully reveals a delegator or signer</td>
</tr>
</tbody>
</table>
Security

**Anonymity**
Intermediate delegates and proxy signer remain anonymous.

**Traceability**
Every valid signature can be traced to its intermediate delegates and proxy signer.

**Non-Frameability**
No one can produce a signature that, when opened, wrongfully reveals a delegator or signer.
Generic Construction: Ingredients

using

- Digital signatures (EUF-CMA)
- Public-key encryption (IND-CPA)
- Non-interactive zero-knowledge proofs
Generic Construction: Ingredients

using

- Digital signatures (EUF-CMA)
- Public-key encryption (IND-CPA)
- Non-interactive zero-knowledge proofs

(Existence follows from trapdoor permutations)
**Generic Construction: Overview**

**Setup**
Generates decryption key for opening authority; signing key for issuer
Parameters: resp. public keys, crs for NIZK

**Register**
Issuer signs user’s public key → certificate

**Delegate**
Sign delegatee’s public key → warrant
Re-delegate: additionally forward received warrants

**Proxy-Sign**
Sign message, encrypt
- interim. delegates’ verification keys and certificates
- warrants
- signature on message

**Output**
- ciphertext
- NIZK proof that plaintext contains valid signatures

**Verify**
Verify NIZK proof

**Open**
Decrypt ciphertext
Generic Construction: Overview

**Setup**  Generates decryption key for opening authority; signing key for issuer
Parameters: resp. public keys, crs for NIZK

**Register**  Issuer signs user’s public key → *certificate*

**Delegate**  Sign delegatee’s public key → *warrant*
Re-delegate: additionally forward received warrants

**Proxy-Sign**  Sign message, encrypt
- interm. delegators’ verification keys and certificates
- warrants
- signature on message

**Output**
- ciphertext
- NIZK proof that plaintext contains valid signatures

**Verify**  Verify NIZK proof

**Open**  Decrypt ciphertext
Generic Construction: Overview

**Setup**  Generates decryption key for opening authority; signing key for issuer

Parameters: resp. public keys, \( crs \) for NIZK

**Register**  Issuer signs user’s public key \( \rightarrow \) certificate

**Delegate**  Sign delegatee’s public key \( \rightarrow \) warrant

**Re-delegate**: additionally forward received warrants

**Proxy-Sign**  Sign message, encrypt

- interm. delegatores’ verification keys and certificates
- warrants
- signature on message

**Output**  

- ciphertext
- NIZK proof that plaintext contains valid signatures

**Verify**  Verify NIZK proof

**Open**  Decrypt ciphertext
Generic Construction: Overview

**Setup**  Generates decryption key for opening authority; signing key for issuer
Parameters: resp. public keys, \(crs\) for NIZK

**Register**  Issuer signs user’s public key → *certificate*

**Delegate**  Sign delegatee’s public key → *warrant*
**Re-delegate:** additionally forward received warrants

**Proxy-Sign**  Sign message, encrypt
- interim. delegates’ verification keys and certificates
- warrants  • signature on message

**Output**
- ciphertext
- NIZK proof that plaintext contains valid signatures

**Verify**  Verify NIZK proof

**Open**  Decrypt ciphertext
Generic Construction: Overview

**Setup**  Generates decryption key for opening authority; signing key for issuer
Parameters: resp. public keys, crs for NIZK

**Register**  Issuer signs user’s public key → *certificate*

**Delegate**  Sign delegatee’s public key → *warrant*
Re-delegate: additionally forward received warrants

**Proxy-Sign**  Sign message, encrypt
- interm. delegates’ verification keys and certificates
- warrants
- signature on message

**Output**
- ciphertext
- NIZK proof that plaintext contains valid signatures

**Verify**  Verify NIZK proof

**Open**  Decrypt ciphertext
Generic Construction: Overview

**Setup** Generates decryption key for opening authority; signing key for issuer
Parameters: resp. public keys, \( crs \) for NIZK

**Register** Issuer signs user’s public key \( \rightarrow \) certificate

**Delegate** Sign delegatee’s public key \( \rightarrow \) warrant
**Re-delegate:** additionally forward received warrants

**Proxy-Sign** Sign message, encrypt
- interm. delegators’ verification keys and certificates
- warrants
- signature on message

**Output**
- ciphertext
- NIZK proof that plaintext contains valid signatures

**Verify** Verify NIZK proof

**Open** Decrypt ciphertext
F, Pointcheval: Proofs on Encrypted Values in Bilinear Groups and an Application to Anonymity of Signatures. [PAIRING ’09]

- Encryption and proofs based on a generalization of techniques of Boyen-Waters Group Signatures [PKC’07] based on Subgroup Decision Assumption
- Signature scheme inefficient due to bit-by-bit techniques
1. Motivation: Anonymous Proxy Signatures

2. Groth-Sahai Witness-Indistinguishable Proofs

3. Automorphic Signatures
Non-Interactive Witness-Indistinguishable Proofs

An NP language $\mathcal{L}$ is defined by relation $R$ as $\mathcal{L} := \{x \mid \exists w : (x, w) \in R\}$.

A NIWI for $\mathcal{L}$ consists of Setup, Prove and Verify.

- Setup outputs a common reference string $crs$
- $\text{Prove}(crs, x, w)$ outputs a proof $\pi$
- $\text{Verify}(crs, x, \pi)$ and outputs 1 or 0
Non-Interactive Witness-Indistinguishable Proofs

An NP language \( \mathcal{L} \) is defined by relation \( R \) as \( \mathcal{L} := \{ x \mid \exists w : (x, w) \in R \} \).

A NIWI for \( \mathcal{L} \) consists of Setup, Prove and Verify.

- **Setup** outputs a common reference string \( crs \)
- **Prove**\((crs, x, w)\) outputs a proof \( \pi \)
- **Verify**\((crs, x, \pi)\) and outputs 1 or 0

It satisfies

- **completeness**
- **soundness**
- **witness indistinguishability**
Bilinear Groups and the Decision Linear Assumption [BBS04]

- Bilinear group \((p, G, G_T, e, G)\)
  - \((G, +)\) and \((G_T, \cdot)\) cyclic groups of prime order \(p\)
  - \(e: G \times G \to G_T\) bilinear, i.e. \(\forall X, Y \in G, \forall a, b \in \mathbb{Z}: e(aX, bY) = e(X, Y)^{ab}\)
  - \(G = \langle G \rangle, G_T = \langle e(G, G) \rangle\)
Bilinear Groups and the Decision Linear Assumption [BBS04]

- Bilinear group \((p, G, G_T, e, G)\)
  - \((G, +)\) and \((G_T, \cdot)\) cyclic groups of prime order \(p\)
  - \(e : G \times G \rightarrow G_T\) bilinear, i.e. \(\forall X, Y \in G, \forall a, b \in \mathbb{Z}:\)
    \[ e(aX, bY) = e(X, Y)^{ab} \]
  - \(G = \langle G \rangle, G_T = \langle e(G, G) \rangle\)

- Given \((U, V, G, \alpha U, \beta V, \gamma G)\) it is hard to decide whether \(\gamma = \alpha + \beta\).
Bilinear Groups and the Decision Linear Assumption [BBS04]

- Bilinear group \((p, G, G_T, e, G)\)
  - \((G, +)\) and \((G_T, \cdot)\) cyclic groups of prime order \(p\)
  - \(e : G \times G \to G_T\) bilinear, i.e. \(\forall X, Y \in G, \forall a, b \in \mathbb{Z}: \)
    \[ e(aX, bY) = e(X, Y)^{ab} \]
  - \(G = \langle G \rangle, G_T = \langle e(G, G) \rangle\)

- Given \((U, V, G, \alpha U, \beta V, \gamma G)\) it is hard to decide whether \(\gamma = \alpha + \beta\).

PPE

A *pairing-product equation* is an equation over variables \(X_1, \ldots, X_n \in G\) of the form

\[
\prod_{i=1}^{n} e(A_i, X_i) \prod_{i=1}^{n} \prod_{j=1}^{n} e(X_i, X_j)^{\gamma_{i,j}} = t_T, \tag{E}
\]

determined by \(A_i \in G, \gamma_{i,j} \in \mathbb{Z}_p\) and \(t_T \in G_T\), for \(1 \leq i, j \leq n\).
Bilinear Groups and the Decision Linear Assumption [BBS04]

- Bilinear group \((p, G, G_T, e, G)\)
  - \((G, +)\) and \((G_T, \cdot)\) cyclic groups of prime order \(p\)
  - \(e : G \times G \rightarrow G_T\) bilinear, i.e. \(\forall X, Y \in G, \forall a, b \in \mathbb{Z} : e(aX, bY) = e(X, Y)^{ab}\)
- \(G = \langle G \rangle, G_T = \langle e(G, G) \rangle\)

Given \((U, V, G, \alpha U, \beta V, \gamma G)\) it is hard to decide whether \(\gamma = \alpha + \beta\).

PPE

A pairing-product equation is an equation over variables \(X_1, \ldots, X_n \in G\) of the form

\[
\prod_{i=1}^{n} e(A_i, X_i) \prod_{i=1}^{n} \prod_{j=1}^{n} e(X_i, X_j)^{\gamma_{i,j}} = t_T,
\]

(E)

determined by \(A_i \in G, \gamma_{i,j} \in \mathbb{Z}_p\) and \(t_T \in G_T\), for \(1 \leq i, j \leq n\).

Groth, Sahai: NIWI proof of satisfiability of PPE
Setup on input the bilinear group output a commitment key $ck$

Com on input $ck$, $X \in \mathbb{G}$, randomness $\rho$ output commitment $c_X$ to $X$

Prove on input $ck$, $(X_i, \rho_i)_{i=1}^n$, equation $E$ output a proof $\phi$

Verify on input $ck$, $\vec{c}$, $E$, $\phi$, output 0 or 1
**Setup** on input the bilinear group output a **commitment key** $ck$

**Com** on input $ck$, $X \in \mathbb{G}$, randomness $\rho$ output commitment $c_X$ to $X$

**Prove** on input $ck$, $(X_i, \rho_i)_{i=1}^n$, equation $E$ output a **proof** $\phi$

**Verify** on input $ck$, $\vec{c}$, $E$, $\phi$, output 0 or 1

**Correctness** Honestly generated proofs are accepted by **Verify**

**Soundness** ExtSetup outputs $(ck, ek)$ s.t.

given $\vec{c}$ and $\phi$ s.t. $\text{Verify}(ck, \vec{c}, E, \phi) = 1$ then $\text{Extract}(ek, \vec{c})$

returns $\vec{X}$ that satisfies $E$

**Witness-Indistinguishability** WISetup outputs $ck^*$ indist. from $ck$ s.t.

- Com($ck^*$, ·, ·) produces statistically hiding commitments i.e.
  \[ \forall c \forall X \exists \rho : \text{Com}(ck^*, X, \rho) = c \]

- Given $(X_i, \rho_i)_i$, $(X'_i, \rho'_i)_i$ s.t. $c_i = \text{Com}(ck^*, X_i, \rho_i) = \text{Com}(ck^*, X'_i, \rho'_i)$ and $(X_i)_i$ and $(X'_i)_i$ satisfy $E$ then
  \[ \text{Prove}(ck^*, (X_i, \rho_i)_i, E) \sim \text{Prove}(ck^*, (X'_i, \rho'_i)_i, E) \]
Groth-Sahai I

**Setup** on input the bilinear group output a commitment key $ck$

**Com** on input $ck$, $X \in \mathbb{G}$, randomness $\rho$ output commitment $c_X$ to $X$

**Prove** on input $ck$, $(X_i, \rho_i)_{i=1}^n$, equation $E$ output a proof $\phi$

**Verify** on input $ck$, $\vec{c}$, $E$, $\phi$, output 0 or 1

---

**Correctness** Honestly generated proofs are accepted by Verify

**Soundness** ExtSetup outputs $(ck, ek)$ s.t.

- given $\vec{c}$ and $\phi$ s.t. Verify$(ck, \vec{c}, E, \phi) = 1$ then Extract$(ek, \vec{c})$

returns $\tilde{X}$ that satisfies $E$

**Witness-Indistinguishability** WISetup outputs $ck^*$ indist. from $ck$ s.t.

- $\text{Com}(ck^*, \cdot, \cdot)$ produces statistically hiding commitments i.e.
  $$\forall c \ \forall X \ \exists \rho : \text{Com}(ck^*, X, \rho) = c$$

- Given $(X_i, \rho_i)_i$, $(X'_i, \rho'_i)_i$ s.t. $c_i = \text{Com}(ck^*, X_i, \rho_i) = \text{Com}(ck^*, X'_i, \rho'_i)$
  and $(X_i)_i$ and $(X'_i)_i$ satisfy $E$ then
  $$\text{Prove}(ck^*, (X_i, \rho_i)_i, E) \sim \text{Prove}(ck^*, (X'_i, \rho'_i)_i, E)$$
Groth-Sahai II

Setup  on input the bilinear group output a commitment key $ck$

Com  on input $ck$, $X \in \mathbb{G}$, randomness $\rho$ output commitment $c_X$ to $X$

Prove on input $ck$, $(X_i, \rho_i)_{i=1}^n$, equation $E$ output a proof $\phi$

Verify on input $ck$, $\vec{c}$, $E$, $\phi$, output 0 or 1

**Correctness**    Honestly generated proofs are accepted by Verify

**Soundness** ExtSetup outputs $(ck, ek)$ s.t.
given $\vec{c}$ and $\phi$ s.t. $\text{Verify}(ck, \vec{c}, E, \phi) = 1$ then $\text{Extract}(ek, \vec{c})$
returns $\vec{X}$ that satisfies $E$

**Witness-Indistinguishability** WISetup outputs $ck^*$ indist. from $ck$ s.t.

- $\text{Com}(ck^*, \cdot, \cdot)$ produces statistically hiding commitments i.e.
  \[ \forall c \ \forall X \ \exists \rho : \text{Com}(ck^*, X, \rho) = c \]

- Given $(X_i, \rho_i)_i$, $(X'_i, \rho'_i)_i$ s.t. $c_i = \text{Com}(ck^*, X_i, \rho_i) = \text{Com}(ck^*, X'_i, \rho'_i)$
  and $(X_i)_i$ and $(X'_i)_i$ satisfy $E$ then
  \[ \text{Prove}(ck^*, (X_i, \rho_i)_i, E) \sim \text{Prove}(ck^*, (X'_i, \rho'_i)_i, E) \]
1 Motivation: Anonymous Proxy Signatures

2 Groth-Sahai Witness-Indistinguishable Proofs

3 Automorphic Signatures
Motivation

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

Opener’s public and decryption key: \((ck, ek) \leftarrow \text{ExtSetup}\)

To instantiate generic construction, we need signature scheme s.t.
- signatures are group elements
- verification by PPE
- able to sign public keys
- EUF-CMA
Motivation

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

Opener’s public and decryption key: \((ck, ek) \leftarrow \text{ExtSetup}\)

- To instantiate generic construction, we need signature scheme s.t.
  - signatures are group elements
  - verification by PPE
  - able to sign public keys
  - EUF-CMA
Motivation

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

Opener’s public and decryption key: \((ck, ek) \leftarrow \text{ExtSetup}\)

To instantiate generic construction, we need signature scheme s.t.
- signatures are group elements
- verification by PPE
- able to sign public keys
- EUF-CMA
Motivation

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

Opener’s public and decryption key: \((ck, ek) \leftarrow \text{ExtSetup}\)

- To instantiate generic construction, we need signature scheme s.t.
  - signatures are group elements
  - verification by PPE
  - able to sign public keys
  - EUF-CMA
Motivation

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

  Opener’s public and decryption key: \((ck, ek) \leftarrow \text{ExtSetup}\)

To instantiate generic construction, we need signature scheme s.t.
- signatures are group elements
- verification by PPE
- able to sign public keys
- EUF-CMA
Groth-Sahai proofs allow us to
- commit to (encrypt) group elements and to
- prove that they satisfy PPEs

Opener’s public and decryption key: $(ck, ek) \leftarrow \text{ExtSetup}$

To instantiate generic construction, we need signature scheme s.t.
- signatures are group elements
- verification by PPE
- able to sign public keys
- EUF-CMA
Motivation

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

Opener’s public and decryption key: \((ck, ek) \leftarrow \text{ExtSetup}\)

- To instantiate generic construction, we need signature scheme s.t.
  - signatures are group elements
  - verification by PPE
  - able to sign public keys
  - EUF-CMA

Automorphic Signatures
Boneh-Boyen Signatures

The \( q \)-Strong Diffie-Hellman Problem (SDH) [BB04]

Given \( (G, xG, x^2G, \ldots, x^q G) \in \mathbb{G}^{q+1} \) for \( x \leftarrow \mathbb{Z}_p^* \), output \( \left( \frac{1}{x+c} G, c \right) \in \mathbb{G} \times \mathbb{Z}_p \).

Boneh-Boyen Weak Signatures

Given \( G, xG \in \mathbb{G} \) and \( q-1 \) distinct pairs \( \left( \frac{1}{x+c_i} G, c_i \right) \in \mathbb{G} \times \mathbb{Z}_p \), output a new pair \( \left( \frac{1}{x+c} G, c \right) \in \mathbb{G} \times \mathbb{Z}_p \).

Boneh-Boyen Short Signatures

- Secret key \( (x, y) \in \mathbb{Z}_p^2 \), public key \( X = xG, \ Y = yG \)
- Sign \( m \in \mathbb{Z}_p \): choose \( r \leftarrow \mathbb{Z}_p \); signature: \( (A = \frac{1}{x+m+ry} G, r) \)
- Verify \( (A, r) \) on \( m \) under \( (X, Y) \) by checking \( e(A, X + mG + rY) = e(G, G) \)
The $q$-Strong Diffie-Hellman Problem (SDH) [BB04]

Given $(G, xG, x^2G, \ldots, x^qG) \in \mathbb{G}^{q+1}$ for $x \leftarrow \mathbb{Z}_p^*$, output $(\frac{1}{x+c}G, c) \in \mathbb{G} \times \mathbb{Z}_p$.

Boneh-Boyen Weak Signatures

Given $G, xG \in \mathbb{G}$ and $q - 1$ distinct pairs $(\frac{1}{x+c_i}G, c_i) \in \mathbb{G} \times \mathbb{Z}_p$, output a new pair $(\frac{1}{x+c}G, c) \in \mathbb{G} \times \mathbb{Z}_p$.

Boneh-Boyen Short Signatures

- Secret key $(x, y) \in \mathbb{Z}_p^2$, public key $X = xG$, $Y = yG$
- Sign $m \in \mathbb{Z}_p$: choose $r \leftarrow \mathbb{Z}_p$; signature: $(A = \frac{1}{x+m+ry}G, r)$
- Verify $(A, r)$ on $m$ under $(X, Y)$ by checking $e(A, X + mG + rY) = e(G, G)$
Boneh-Boyen Signatures

The *q*-Strong Diffie-Hellman Problem (SDH) [BB04]

Given \((G, xG, x^2G, \ldots, x^qG) \in \mathbb{G}^{q+1}\) for \(x \leftarrow \mathbb{Z}^*_p\), output \((\frac{1}{x+c}G, c) \in \mathbb{G} \times \mathbb{Z}_p\).

Boneh-Boyen Weak Signatures

Given \(G, xG \in \mathbb{G}\) and \(q - 1\) distinct pairs \((\frac{1}{x+c_i}G, c_i) \in \mathbb{G} \times \mathbb{Z}_p\), output a new pair \((\frac{1}{x+c}G, c) \in \mathbb{G} \times \mathbb{Z}_p\).

Boneh-Boyen Short Signatures

- Secret key \((x, y) \in \mathbb{Z}_p^2\), public key \(X = xG, Y = yG\)
- Sign \(m \in \mathbb{Z}_p\): choose \(r \leftarrow \mathbb{Z}_p\); signature: \((A = \frac{1}{x+m+ry}G, r)\)
- Verify \((A, r)\) on \(m\) under \((X, Y)\) by checking \(e(A, X + mG + rY) = e(G, G)\)
Boneh-Boyen Signatures

The $q$-Strong Diffie-Hellman Problem (SDH) [BB04]

Given $(G, xG, x^2G, \ldots, x^qG) \in \mathbb{G}^{q+1}$ for $x \leftarrow \mathbb{Z}_p^*$, output $(\frac{1}{x+c}G, c) \in \mathbb{G} \times \mathbb{Z}_p$.

Boneh-Boyen Weak Signatures

Given $G, xG \in \mathbb{G}$ and $q - 1$ distinct pairs $(\frac{1}{x+c_i}G, c_i) \in \mathbb{G} \times \mathbb{Z}_p$, output a new pair $(\frac{1}{x+c}G, c) \in \mathbb{G} \times \mathbb{Z}_p$.

Boneh-Boyen Short Signatures

- Secret key $(x, y) \in \mathbb{Z}_p^2$, public key $X = xG$, $Y = yG$
- Sign $m \in \mathbb{Z}_p$: choose $r \leftarrow \mathbb{Z}_p$; signature: $(A = \frac{1}{x+m+ry}G, r)$
- Verify $(A, r)$ on $m$ under $(X, Y)$ by checking $e(A, X + mG + rY) = e(\frac{1}{x+m+ry}G, (x + m + ry)G) = e(G, G)$
Variants of Boneh-Boyen

Boneh-Boyen Weak Signatures

Given $G, X := xG \in \mathbb{G}$ and $q - 1$ distinct pairs

\((\frac{1}{x+c_i} G, c_i) \in \mathbb{G} \times \mathbb{Z}_p\),

output a new pair \((\frac{1}{x+c} G, c) \in \mathbb{G} \times \mathbb{Z}_p\).
Variants of Boneh-Boyen

The Hidden SDH [BW07]

Given \( G, H, X := xG \in \mathbb{G} \) and \( q - 1 \) distinct triples
\[
\left( \frac{1}{x + c_i} G, c G, c_i H \right) \in \mathbb{G}^3,
\]
output a new triple \( \left( \frac{1}{x + c} G, c G, c H \right) \in \mathbb{G}^3 \).
The Hidden SDH [BW07]

Given $G, H, X := xG \in \mathbb{G}$ and $q - 1$ distinct triples

$\left( \frac{1}{x+c_i} G, c_i G, c_i H \right) \in \mathbb{G}^3$, output a new triple $\left( \frac{1}{x+c} G, cG, cH \right) \in \mathbb{G}^3$.

- All components are group elements
- Validity of a triple $(A, C, D)$ is verifiable by PPEs:

$$e(A, X + C) = e(G, G)$$
$$e(C, H) = e(G, D)$$
F, Pointcheval, Vergnaud: Transferable Constant-Size Fair E-Cash [CANS’09]

SDH implies hardness of the following:

Given $G, K, X := xG \in \mathbb{G}$ and $q - 1$ triples

\[
\frac{1}{x+c_i}(K+v_i G), c_i, v_i \in \mathbb{G} \times \mathbb{Z}_p^2, \text{ output a new triple}
\]

\[
\frac{1}{x+c}(K+v G), c, v \in \mathbb{G} \times \mathbb{Z}_p^2.
\]
SDH implies hardness of the following:

Given $G, K, X := xG \in \mathbb{G}$ and $q - 1$ triples

$$\left( \frac{1}{x+c_i} (K + v_i G), c_i, v_i \right) \in \mathbb{G} \times \mathbb{Z}_p^2$$

output a new triple

$$\left( \frac{1}{x+c} (K + vG), c, v \right) \in \mathbb{G} \times \mathbb{Z}_p^2$$

Asymm. Double Hidden SDH (ADHSDH)

Given $G, K, F, H, X := xG, Y := xH \in \mathbb{G}$ and $q - 1$ tuples

$$\left( \frac{1}{x+c_i} (K + v_i G), c_i F, c_i H, v_i G, v_i H \right)$$

output a new tuple

$$\left( \frac{1}{x+c} (K + vG), cF, cH, vG, vH \right)$$
Assumptions II

Verification

$(A, C, D, V, W)$ satisfies

- $e(A, Y + D) = e\left(\frac{1}{x+c}(K + vG), xH + cH\right) = e(K + V, H)$,
- $e(C, H) = e(cF, H) = e(F, D)$
- $e(V, H) = e(vG, H) = e(G, W)$
Assumptions II

Verification

(A, C, D, V, W) satisfies

- $e(A, Y + D) = e\left(\frac{1}{x+c}(K + vG), xH + cH\right) = e(K + V, H)$,
- $e(C, H) = e(cF, H) = e(F, D)$
- $e(V, H) = e(vG, H) = e(G, W)$

(Weak) Flexible CDH (WFCDH)

Given $(G, aG, bG) \in \mathbb{G}^3$, output $(R, aR, bR, abR) \in \mathbb{G}^4$ with $R \neq 0$. 

Automorphic Signature

- Parameters: \((G, K, F, H, T) \leftarrow \mathbb{G}^5\), which define the message space as \(\mathcal{D}H := \{(mG, mH) | m \in \mathbb{Z}_p\}\).
- KeyGen: secret key \(x \leftarrow \mathbb{Z}_p\), public key \((X := xG, Y := yH)\)
- Sign \((M, N) \in \mathcal{D}H\): choose \(c, r \leftarrow \mathbb{Z}_p\), set
  \[
  (A := \frac{1}{x + c}(K + rT + M), C := cF, D := cH, R := rG, S := rH)
  \]
- A signature on a message \((M, N) \in \mathcal{D}H\) is valid iff
  \[
  e(A, Y + D) = e(K + M, H)e(T, S) \\
  e(C, H) = e(F, D) \\
  e(R, H) = e(G, S)
  \]
Automorphic Signature

- **Parameters:** $(G, K, F, H, T) \leftarrow \mathbb{G}^5$, which define the message space as $\mathcal{DH} := \{(mG, mH) \mid m \in \mathbb{Z}_p\}$.
- **KeyGen:** secret key $x \leftarrow \mathbb{Z}_p$, public key $(X := xG, Y := yH)$
- **Sign** $(M, N) \in \mathcal{DH}$: choose $c, r \leftarrow \mathbb{Z}_p$, set

  $$(A := \frac{1}{x+c}(K+rt+M), C := cF, D := cH, R := rG, S := rH)$$

- A signature on a message $(M, N) \in \mathcal{DH}$ is valid iff

  $$e(A, Y + D) = e(K + M, H) e(T, S) \quad e(C, H) = e(F, D) \quad e(R, H) = e(G, S)$$

The above scheme is EUF-CMA under ADHSDH and WFCDH.
Applications

Efficiency

- Messages and public keys in $\mathbb{G}^2$, signatures in $\mathbb{G}^5$
- Verification: 7 pairing evaluations
- Also instantiable in *asymmetric* bilinear groups

In combination with Groth-Sahai proofs, automorphic signatures enable efficient instantiations of generic concepts.
Applications

Efficiency

- Messages and public keys in $G^2$, signatures in $G^5$
- Verification: 7 pairing evaluations
- Also instantiable in asymmetric bilinear groups

In combination with Groth-Sahai proofs, automorphic signatures enable efficient instantiations of generic concepts.

- Round-Optimal Blind Signatures
- Group Signatures
- Anonymous Proxy Signatures with new features:
  - Delegator anonymity (by randomizing Groth-Sahai proofs)
  - Blind delegation (using blind signatures)
Thank you! 😊