Strongly-Optimal Structure Preserving Signatures from Type II Pairings: Synthesis and Lower Bounds

Gilles Barthe[2] **Edvard Fagerholm**[1] Dario Fiore[2]
Andre Scedrov[1] Benedikt Schmidt[2] Mehdi Tibouchi[3]

[1]University of Pennsylvania [2]IMDEA Software Institute
[3]NTT Secure Platform Laboratories

March 31, 2015

Computer-Aided Cryptography

- Motivation for synthesis:
  - Find new and improved schemes
  - Improve theoretical understanding through exhaustive search
  - Prove theorems and make conjectures

Computer-Aided Cryptography

- Motivation for synthesis:
  - Find new and improved schemes
  - Improve theoretical understanding through exhaustive search
  - Prove theorems and make conjectures
- Two approaches to synthesis:
  - Transformational synthesis: modify existing schemes; security relies on security of original construction
  - Full synthesis: generate schemes and automatically prune insecure ones

Computer-Aided Cryptography

- Motivation for synthesis:
  - Find new and improved schemes
  - Improve theoretical understanding through exhaustive search
  - Prove theorems and make conjectures
- Two approaches to synthesis:
  - Transformational synthesis: modify existing schemes; security relies on security of original construction
  - Full synthesis: generate schemes and automatically prune insecure ones
- Transformational synthesis only useful for finding new schemes

Computer-Aided Cryptography cont.

- Generic Group Analyzer (GGA) of CRYPTO 2014:
  - Automated verification tool, starting point for this work
  - Language for expressing assumptions in the generic group model
  - Language supports oracles, complex winning conditions
  - Expressive enough for (s)EUF-CMA allowing analysis of e.g. SPS and SPS-EQ

Computer-Aided Cryptography cont.

- Generic Group Analyzer (GGA) of CRYPTO 2014:
    - Automated verification tool, starting point for this work
    - Language for expressing assumptions in the generic group model
    - Language supports oracles, complex winning conditions
    - Expressive enough for (s)EUF-CMA allowing analysis of e.g. SPS and SPS-EQ
- Main challenges of current work:
    - Extend GGA to handle Laurent polynomials as input
    - Extend GGA to handle group elements as oracle parameters
    - Prove results conjectured as a result of extensive search
    - Narrow down search spaces through flexible template system

From Automated Verification to Synthesis

- From verification to synthesis:
  - Generate large amount of potential candidate schemes
  - Prune insecure candidates with verification tool
  - Analyze remaining schemes

From Automated Verification to Synthesis

- From verification to synthesis:
  - Generate large amount of potential candidate schemes
  - Prune insecure candidates with verification tool
  - Analyze remaining schemes
- Previous method has its challenges:
  - Need fast verification in comparison to search space size
  - Might require manual analysis in order to cut search space a priori

From Automated Verification to Synthesis

- From verification to synthesis:
  - Generate large amount of potential candidate schemes
  - Prune insecure candidates with verification tool
  - Analyze remaining schemes
- Previous method has its challenges:
  - Need fast verification in comparison to search space size
  - Might require manual analysis in order to cut search space a priori
- In practice though:
  - Combined workflow of manual analysis and automated search is fast and reduces the tedious part of the work
  - Not just useful for synthesis, but also for improving theoretical understanding as well as making conjectures

Current Work

Current Work

- Synthesis of SPS schemes in the Type II setting

Current Work

- Synthesis of SPS schemes in the Type II setting
- Uses the "verifier to synthesizer" method from the previous slides

Current Work

- Synthesis of SPS schemes in the Type II setting
- Uses the "verifier to synthesizer" method from the previous slides
- Leverages the Generic Group Analyzer as the verification tool

Current Work

- Synthesis of SPS schemes in the Type II setting
- Uses the "verifier to synthesizer" method from the previous slides
- Leverages the Generic Group Analyzer as the verification tool
- Develop template system to specify candidate SPS schemes

Current Work

- Synthesis of SPS schemes in the Type II setting
- Uses the “verifier to synthesizer” method from the previous slides
- Leverages the Generic Group Analyzer as the verification tool
- Develop template system to specify candidate SPS schemes
- Find improvements on previously known SPS schemes

Current Work

- Synthesis of SPS schemes in the Type II setting
- Uses the "verifier to synthesizer" method from the previous slides
- Leverages the Generic Group Analyzer as the verification tool
- Develop template system to specify candidate SPS schemes
- Find improvements on previously known SPS schemes
- Prove new minimality results on Type II SPS schemes derived from conjectures based on search results

Structure-Preserving Cryptography

Typical mathematical structure in cryptography:

- A finite cyclic group $\mathbb{G}$
- A bilinear group $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$

Structure-Preserving Cryptography

Typical mathematical structure in cryptography:

- A finite cyclic group $\mathbb{G}$
- A bilinear group $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$
  - Type I if $\mathbb{G}_1 = \mathbb{G}_2$
  - Type II if there is an efficient isomorphism $\phi : \mathbb{G}_2 \to \mathbb{G}_1$, but none $\mathbb{G}_1 \to \mathbb{G}_2$
  - Type III if no known efficient isomorphism between $\mathbb{G}_1$ and $\mathbb{G}_2$

Structure-Preserving Cryptography

Typical mathematical structure in cryptography:

- A finite cyclic group $\mathbb{G}$
- A bilinear group $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$
  - Type I if $\mathbb{G}_1 = \mathbb{G}_2$
  - Type II if there is an efficient isomorphism $\phi : \mathbb{G}_2 \to \mathbb{G}_1$, but none $\mathbb{G}_1 \to \mathbb{G}_2$
  - Type III if no known efficient isomorphism between $\mathbb{G}_1$ and $\mathbb{G}_2$

Structure-preserving cryptography is a design philosophy:

- Use only generic group operations
- All transmitted data consists of tuples of group elements
- Allows composability and modular design

Structure-Preserving Signatures

Definition (Pairing-Product Equation)

Given bilinear group $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ a *pairing-product equation* (PPE) is an equation

$$\prod_i \prod_j e(X_i, Y_j)^{a_{ij}} = 1,\ X_i \in \mathbb{G}_1,\ \ Y_j \in \mathbb{G}_2, a_{ij} \in \mathbb{Z},$$

if in the Type II setting, we can also have $X_i = \phi(Y_j)$.

Structure-Preserving Signatures

Definition (Pairing-Product Equation)

Given bilinear group $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ a *pairing-product equation* (PPE) is an equation

$$\prod_i \prod_j e(X_i, Y_j)^{a_{ij}} = 1,\ X_i \in \mathbb{G}_1,\ \ Y_j \in \mathbb{G}_2, a_{ij} \in \mathbb{Z},$$

if in the Type II setting, we can also have $X_i = \phi(Y_j)$.

Definition (Structure-Preserving Signature Scheme)

Signature scheme given bilinear group $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e)$, such that

- Verification key consist of elements of $\mathbb{G}_1, \mathbb{G}_2$.
- Messages/Signatures consist of elements of $\mathbb{G}_1, \mathbb{G}_2$.
- Verification algorithm checks PPEs in the bilinear group.

Previous Work on SPS schemes

Previous Work on SPS schemes

Much of previous work has concentrated on proving minimality results:

Previous Work on SPS schemes

Much of previous work has concentrated on proving minimality results:

- Minimal number of verification and signing keys

Previous Work on SPS schemes

Much of previous work has concentrated on proving minimality results:

- Minimal number of verification and signing keys
- Minimal number of group elements in signature

Previous Work on SPS schemes

Much of previous work has concentrated on proving minimality results:

- Minimal number of verification and signing keys
- Minimal number of group elements in signature
- Minimal number of PPEs for verification

Previous Work on SPS schemes

Much of previous work has concentrated on proving minimality results:

- Minimal number of verification and signing keys
- Minimal number of group elements in signature
- Minimal number of PPEs for verification

Setting	Signature	Verification Key	PPEs
Type I	3	3	2
Type II	2	2	1
Type III	3	2	2

Previous Work on SPS schemes

Much of previous work has concentrated on proving minimality results:

- Minimal number of verification and signing keys
- Minimal number of group elements in signature
- Minimal number of PPEs for verification

Setting	Signature	Verification Key	PPEs
Type I	3	3	2
Type II	2	2	1
Type III	3	2	2

- For all types we know how to simultaneously minimize all parameters

Previous Work on SPS schemes

Much of previous work has concentrated on proving minimality results:

- Minimal number of verification and signing keys
- Minimal number of group elements in signature
- Minimal number of PPEs for verification

Setting	Signature	Verification Key	PPEs
Type I	3	3	2
Type II	2	2	1
Type III	3	2	2

- For all types we know how to simultaneously minimize all parameters
- Also minimality result for schemes based on noninteractive assumptions

A Closer Look at the Type II Case

Achieving the lower bounds for a Type II group $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$?

- Two verification key elements: $V, W \in \mathbb{G}_1$ or $V \in \mathbb{G}_1$, $W \in \mathbb{G}_2$
- Signing key: the discrete logarithms
- Message $M \in \mathbb{G}_2$

A Closer Look at the Type II Case

Achieving the lower bounds for a Type II group $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$?

- Two verification key elements: $V, W \in \mathbb{G}_1$ or $V \in \mathbb{G}_1$, $W \in \mathbb{G}_2$
- Signing key: the discrete logarithms
- Message $M \in \mathbb{G}_2$

What can be said about the verification equation?

A Closer Look at the Type II Case

Achieving the lower bounds for a Type II group $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$?

- Two verification key elements: $V, W \in \mathbb{G}_1$ or $V \in \mathbb{G}_1$, $W \in \mathbb{G}_2$
- Signing key: the discrete logarithms
- Message $M \in \mathbb{G}_2$

What can be said about the verification equation?

- Pairings expensive to compute

A Closer Look at the Type II Case

Achieving the lower bounds for a Type II group $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$?

- Two verification key elements: $V, W \in \mathbb{G}_1$ or $V \in \mathbb{G}_1$, $W \in \mathbb{G}_2$
- Signing key: the discrete logarithms
- Message $M \in \mathbb{G}_2$

What can be said about the verification equation?

- Pairings expensive to compute
- Efficient verification requires few pairings in PPE

A Closer Look at the Type II Case

Achieving the lower bounds for a Type II group $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$?

- Two verification key elements: $V, W \in \mathbb{G}_1$ or $V \in \mathbb{G}_1$, $W \in \mathbb{G}_2$
- Signing key: the discrete logarithms
- Message $M \in \mathbb{G}_2$

What can be said about the verification equation?

- Pairings expensive to compute
- Efficient verification requires few pairings in PPE
- **Want to minimize number of pairings in PPE**

A Closer Look at Pairings

- A Type II SPS is defined by a tuple (**Setup**, **KeyGen**, **Sign**, **Verify**)
- **Setup** returns the *public parameters*
- **KeyGen** returns the *signing/verification key pair*

A Closer Look at Pairings

- A Type II SPS is defined by a tuple (**Setup**, **KeyGen**, **Sign**, **Verify**)
- **Setup** returns the *public parameters*
- **KeyGen** returns the *signing/verification key pair*
- Data returned by **Setup** and **KeyGen** can be reused

A Closer Look at Pairings

- A Type II SPS is defined by a tuple (**Setup**, **KeyGen**, **Sign**, **Verify**)
- **Setup** returns the *public parameters*
- **KeyGen** returns the *signing/verification key pair*
- Data returned by **Setup** and **KeyGen** can be reused
- A pairing in the PPE is called *offline* if it only depends on the public parameters as well as the verification key elements

A Closer Look at Pairings

- A Type II SPS is defined by a tuple (**Setup**, **KeyGen**, **Sign**, **Verify**)
- **Setup** returns the *public parameters*
- **KeyGen** returns the *signing/verification key pair*
- Data returned by **Setup** and **KeyGen** can be reused
- A pairing in the PPE is called *offline* if it only depends on the public parameters as well as the verification key elements
- Other pairings are called *online*

A Closer Look at Pairings cont.

An example SPS scheme

- $PP = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, \psi, G, H) \leftarrow \textbf{Setup}(1^k)$
- $(VK, SK) \leftarrow \textbf{KeyGen}(PP)$, $SK = (v, w) \leftarrow \mathbb{Z}_p^2$, $VK = (G^v, G^w)$
- $(R, S) = (H^r, (M^v H^w)^{1/r}) \leftarrow \textbf{Sign}(PP, SK, M)$, $M \in \mathbb{G}_2$, $r \leftarrow \mathbb{Z}_p^*$
- $\textbf{Verify}(PP, VK, M, (R, S))$: accept if $e(\psi(R), S) = e(V, M)e(W, H)$

A Closer Look at Pairings cont.

An example SPS scheme

- $PP = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, \psi, G, H) \leftarrow \textbf{Setup}(1^k)$
- $(VK, SK) \leftarrow \textbf{KeyGen}(PP)$, $SK = (v, w) \leftarrow \mathbb{Z}_p^2$, $VK = (G^v, G^w)$
- $(R, S) = (H^r, (M^v H^w)^{1/r}) \leftarrow \textbf{Sign}(PP, SK, M)$, $M \in \mathbb{G}_2$, $r \leftarrow \mathbb{Z}_p^*$
- $\textbf{Verify}(PP, VK, M, (R, S))$: accept if $e(\psi(R), S) = e(V, M)e(W, H)$

Classifying pairings in example SPS scheme

A Closer Look at Pairings cont.

An example SPS scheme

- $PP = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, \psi, G, H) \leftarrow \textbf{Setup}(1^k)$
- $(VK, SK) \leftarrow \textbf{KeyGen}(PP)$, $SK = (v, w) \leftarrow \mathbb{Z}_p^2$, $VK = (G^v, G^w)$
- $(R, S) = (H^r, (M^v H^w)^{1/r}) \leftarrow \textbf{Sign}(PP, SK, M)$, $M \in \mathbb{G}_2$, $r \leftarrow \mathbb{Z}_p^*$
- $\textbf{Verify}(PP, VK, M, (R, S))$: accept if $e(\psi(R), S) = e(V, M)e(W, H)$

Classifying pairings in example SPS scheme

- $e(\psi(R), S)$ online: depends on (R, S)

A Closer Look at Pairings cont.

An example SPS scheme

- $PP = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, \psi, G, H) \leftarrow \textbf{Setup}(1^k)$
- $(VK, SK) \leftarrow \textbf{KeyGen}(PP)$, $SK = (v, w) \leftarrow \mathbb{Z}_p^2$, $VK = (G^v, G^w)$
- $(R, S) = (H^r, (M^v H^w)^{1/r}) \leftarrow \textbf{Sign}(PP, SK, M)$, $M \in \mathbb{G}_2$, $r \leftarrow \mathbb{Z}_p^*$
- $\textbf{Verify}(PP, VK, M, (R, S))$: accept if $e(\psi(R), S) = e(V, M)e(W, H)$

Classifying pairings in example SPS scheme

- $e(\psi(R), S)$ online: depends on (R, S)
- $e(V, M)$ online: depends on M

A Closer Look at Pairings cont.

An example SPS scheme

- $PP = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, \psi, G, H) \leftarrow \textbf{Setup}(1^k)$
- $(VK, SK) \leftarrow \textbf{KeyGen}(PP)$, $SK = (v, w) \leftarrow \mathbb{Z}_p^2$, $VK = (G^v, G^w)$
- $(R, S) = (H^r, (M^v H^w)^{1/r}) \leftarrow \textbf{Sign}(PP, SK, M)$, $M \in \mathbb{G}_2$, $r \leftarrow \mathbb{Z}_p^*$
- $\textbf{Verify}(PP, VK, M, (R, S))$: accept if $e(\psi(R), S) = e(V, M)e(W, H)$

Classifying pairings in example SPS scheme

- $e(\psi(R), S)$ online: depends on (R, S)
- $e(V, M)$ online: depends on M
- $e(W, H)$ offline: depends on verification keys and public parameters

A Closer Look at Pairings cont.

An example SPS scheme

- $PP = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, \psi, G, H) \leftarrow$ **Setup**(1^k)
- $(VK, SK) \leftarrow$ **KeyGen**(PP), $SK = (v, w) \leftarrow \mathbb{Z}_p^2$, $VK = (G^v, G^w)$
- $(R, S) = (H^r, (M^v H^w)^{1/r}) \leftarrow$ **Sign**(PP, SK, M), $M \in \mathbb{G}_2$, $r \leftarrow \mathbb{Z}_p^*$
- **Verify**$(PP, VK, M, (R, S))$: accept if $e(\psi(R), S) = e(V, M)e(W, H)$

Classifying pairings in example SPS scheme

- $e(\psi(R), S)$ online: depends on (R, S)
- $e(V, M)$ online: depends on M
- $e(W, H)$ offline: depends on verification keys and public parameters

Point is that $e(W, H)$ needs to be computed once, if we must verify multiple messages signed by the same key

Search Templates

- Search space defined by polynomials and guards on coefficients

Search Templates

- Search space defined by polynomials and guards on coefficients
- Uses the standard discrete logarithm notation

Search Templates

- Search space defined by polynomials and guards on coefficients
- Uses the standard discrete logarithm notation
- Example: schemes of the form
  $\mathbf{Sign}(PP, SK, M) = (r, \frac{vf(r,m)+g(w,r,m)}{r}) = (R, S)$ with guards
  - f, g of degree one
  - f must contain a term containing m
  - g must contain a term containing w
  - Coefficients of f, g in the set $\{-1, 0, 1\}$

Search Templates

- Search space defined by polynomials and guards on coefficients
- Uses the standard discrete logarithm notation
- Example: schemes of the form
  $\mathbf{Sign}(PP, SK, M) = (r, \frac{vf(r,m)+g(w,r,m)}{r}) = (R, S)$ with guards
  - f, g of degree one
  - f must contain a term containing m
  - g must contain a term containing w
  - Coefficients of f, g in the set $\{-1, 0, 1\}$
- To provide an input for GGA tool, we need the verification equation

Search Templates

- Search space defined by polynomials and guards on coefficients
- Uses the standard discrete logarithm notation
- Example: schemes of the form
  $\textbf{Sign}(PP, SK, M) = (r, \frac{vf(r,m)+g(w,r,m)}{r}) = (R, S)$ with guards
  - f, g of degree one
  - f must contain a term containing m
  - g must contain a term containing w
  - Coefficients of f, g in the set $\{-1, 0, 1\}$
- To provide an input for GGA tool, we need the verification equation
  - Same completion procedure as in GGA tool used to compute a basis for all linear relations in the group $\mathbb{G}_T$ given V, W, R, S, M

Search Templates

- Search space defined by polynomials and guards on coefficients
- Uses the standard discrete logarithm notation
- Example: schemes of the form
  $\mathbf{Sign}(PP, SK, M) = (r, \frac{vf(r,m)+g(w,r,m)}{r}) = (R, S)$ with guards
  - f, g of degree one
  - f must contain a term containing m
  - g must contain a term containing w
  - Coefficients of f, g in the set $\{-1, 0, 1\}$
- To provide an input for GGA tool, we need the verification equation
  - Same completion procedure as in GGA tool used to compute a basis for all linear relations in the group $\mathbb{G}_T$ given V, W, R, S, M
  - Dimension of relation space determines number of PPEs needed

Search Templates

- Search space defined by polynomials and guards on coefficients
- Uses the standard discrete logarithm notation
- Example: schemes of the form
  $\mathbf{Sign}(PP, SK, M) = (r, \frac{vf(r,m)+g(w,r,m)}{r}) = (R, S)$ with guards
  - f, g of degree one
  - f must contain a term containing m
  - g must contain a term containing w
  - Coefficients of f, g in the set $\{-1, 0, 1\}$
- To provide an input for GGA tool, we need the verification equation
  - Same completion procedure as in GGA tool used to compute a basis for all linear relations in the group $\mathbb{G}_T$ given V, W, R, S, M
  - Dimension of relation space determines number of PPEs needed
  - May extract PPEs from basis

Search Templates

- Search space defined by polynomials and guards on coefficients
- Uses the standard discrete logarithm notation
- Example: schemes of the form
  $\mathbf{Sign}(PP, SK, M) = (r, \frac{vf(r,m)+g(w,r,m)}{r}) = (R, S)$ with guards
  - f, g of degree one
  - f must contain a term containing m
  - g must contain a term containing w
  - Coefficients of f, g in the set $\{-1, 0, 1\}$
- To provide an input for GGA tool, we need the verification equation
  - Same completion procedure as in GGA tool used to compute a basis for all linear relations in the group $\mathbb{G}_T$ given V, W, R, S, M
  - Dimension of relation space determines number of PPEs needed
  - May extract PPEs from basis
- Given extracted verification equation(s), generate EUF-CMA winning condition for analysis by interactive solver of GGA tool

Search Results

We find e.g. the following *randomizable* EUF-CMA secure Type II scheme:

- **Setup**(1^k): return $PP = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, \psi, G, H)$
- **KeyGen**(PP): choose random $v, w \leftarrow \mathbb{Z}_p$, return $VK = (G^v, G^w)$, $SK = (v, w)$
- **Sign**(PP, SK, M): given $M \in \mathbb{G}_2$, choose $r \leftarrow \mathbb{Z}_p^*$ and return $(R, S) = (H^r, (M^v H^w)^{1/r})$
- **Verify**$(PP, VK, M, (R, S))$: accept if and only if $M, R, S \in \mathbb{G}_2$ and

$$e(\psi(R), S) = e(V, M)e(W, H)$$

- **Rerand**$(PP, VK, M, (R, S))$: choose $\alpha \leftarrow \mathbb{Z}_p^*$, return $(R', S') = (R^\alpha, S^{1/\alpha})$

Conjectures Based on Search Results

There exists no secure Type II SPS with the following properties

- minimal signature size,
- minimal number of verification keys,
- one PPE in verification equation,

such that the verification equation requires less than 3 pairings. For any verification equation requiring 3 pairings at least two of the pairings must be online.

From Conjecture to Theorem

- We prove that conjecture is true

From Conjecture to Theorem

- We prove that conjecture is true
- Proof technique depends on classifying all possible verification equations with two pairings

From Conjecture to Theorem

- We prove that conjecture is true
- Proof technique depends on classifying all possible verification equations with two pairings
- Verification equations ruled out case-by-case

From Conjecture to Theorem

- We prove that conjecture is true
- Proof technique depends on classifying all possible verification equations with two pairings
- Verification equations ruled out case-by-case

There are three cases for proving the theorem:

From Conjecture to Theorem

- We prove that conjecture is true
- Proof technique depends on classifying all possible verification equations with two pairings
- Verification equations ruled out case-by-case

There are three cases for proving the theorem:

- $V, W \in \mathbb{G}_2$

From Conjecture to Theorem

- We prove that conjecture is true
- Proof technique depends on classifying all possible verification equations with two pairings
- Verification equations ruled out case-by-case

There are three cases for proving the theorem:

- $V, W \in \mathbb{G}_2$
- $V, W \in \mathbb{G}_1$

From Conjecture to Theorem

- We prove that conjecture is true
- Proof technique depends on classifying all possible verification equations with two pairings
- Verification equations ruled out case-by-case

There are three cases for proving the theorem:

- $V, W \in \mathbb{G}_2$
- $V, W \in \mathbb{G}_1$
- $V \in \mathbb{G}_1,\ W \in \mathbb{G}_2$
  - Can be found in full version on eprint
  - Complicated case distinction
  - Gröbner basis computations for ruling out cases

From Conjecture to Theorem

- We prove that conjecture is true
- Proof technique depends on classifying all possible verification equations with two pairings
- Verification equations ruled out case-by-case

There are three cases for proving the theorem:

- $V, W \in \mathbb{G}_2$
- $V, W \in \mathbb{G}_1$
- $V \in \mathbb{G}_1,\ W \in \mathbb{G}_2$
  - Can be found in full version on eprint
  - Complicated case distinction
  - Gröbner basis computations for ruling out cases

From our proofs, we extract a minimal EUF-RMA secure Type II SPS

Converting Type II Schemes to Type III

- There is a simple heuristic for converting Type II to Type III schemes
  - For each use of ψ, we “copy” the argument from $\mathbb{G}_2$ to $\mathbb{G}_1$
  - Replace any $Y \in \mathbb{G}_2$ and $\psi(Y)$ in verification equation by a fresh $Y' \in \mathbb{G}_1$
  - Add PPE $e(Y', H) = e(G, Y)$ to verification equation
  - In signature algorithm add corresponding elements to $\mathbb{G}_1$

Converting Type II Schemes to Type III

- There is a simple heuristic for converting Type II to Type III schemes
  - For each use of ψ, we "copy" the argument from $\mathbb{G}_2$ to $\mathbb{G}_1$
  - Replace any $Y \in \mathbb{G}_2$ and $\psi(Y)$ in verification equation by a fresh $Y' \in \mathbb{G}_1$
  - Add PPE $e(Y', H) = e(G, Y)$ to verification equation
  - In signature algorithm add corresponding elements to $\mathbb{G}_1$
- For each variable that the isomorphism is applied to we need to add a new group element to the corresponding Type III scheme as well as a new PPE

Converting Type II Schemes to Type III

- There is a simple heuristic for converting Type II to Type III schemes
  - For each use of ψ, we "copy" the argument from $\mathbb{G}_2$ to $\mathbb{G}_1$
  - Replace any $Y \in \mathbb{G}_2$ and $\psi(Y)$ in verification equation by a fresh $Y' \in \mathbb{G}_1$
  - Add PPE $e(Y', H) = e(G, Y)$ to verification equation
  - In signature algorithm add corresponding elements to $\mathbb{G}_1$
- For each variable that the isomorphism is applied to we need to add a new group element to the corresponding Type III scheme as well as a new PPE
- Using this method we can translate our new randomizable Type II SPS to a Type III scheme

Converting Type II Schemes to Type III

- There is a simple heuristic for converting Type II to Type III schemes
  - For each use of ψ, we "copy" the argument from $\mathbb{G}_2$ to $\mathbb{G}_1$
  - Replace any $Y \in \mathbb{G}_2$ and $\psi(Y)$ in verification equation by a fresh $Y' \in \mathbb{G}_1$
  - Add PPE $e(Y', H) = e(G, Y)$ to verification equation
  - In signature algorithm add corresponding elements to $\mathbb{G}_1$
- For each variable that the isomorphism is applied to we need to add a new group element to the corresponding Type III scheme as well as a new PPE
- Using this method we can translate our new randomizable Type II SPS to a Type III scheme
- One may even think of Type II synthesis as a search of Type III schemes of a certain form

Criticism of Type II/III by Chatterjee and Menezes

- Care is needed when comparing schemes using currently available instantiations of Type II and Type III pairings
  - Denote pairing by $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$
  - Group operations and pairings of roughly equivalent complexity
  - In the Type II setting, membership testing in $\mathbb{G}_2$ (currently) requires two pairings and is much slower

Criticism of Type II/III by Chatterjee and Menezes

- Care is needed when comparing schemes using currently available instantiations of Type II and Type III pairings
  - Denote pairing by $e : \mathbb{G}_1 \times \mathbb{G}_2 \rightarrow \mathbb{G}_T$
  - Group operations and pairings of roughly equivalent complexity
  - In the Type II setting, membership testing in $\mathbb{G}_2$ (currently) requires two pairings and is much slower
- Concrete implementations of the optimal schemes found in the paper have to compute the additional pairings required for group membership testing

Criticism of Type II/III by Chatterjee and Menezes

- Care is needed when comparing schemes using currently available instantiations of Type II and Type III pairings
  - Denote pairing by $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$
  - Group operations and pairings of roughly equivalent complexity
  - In the Type II setting, membership testing in $\mathbb{G}_2$ (currently) requires two pairings and is much slower
- Concrete implementations of the optimal schemes found in the paper have to compute the additional pairings required for group membership testing
- However, a Type II scheme with fewer pairings will still need fewer pairings once group membership testing is accounted for

Conclusions and Future Work

- Contributions of paper:

Conclusions and Future Work

- Contributions of paper:
  - Template system for specifying SPS candidates

Conclusions and Future Work

- Contributions of paper:
  - Template system for specifying SPS candidates
  - Tool to prune SPS schemes insecure in the GGM

Conclusions and Future Work

- Contributions of paper:
  - Template system for specifying SPS candidates
  - Tool to prune SPS schemes insecure in the GGM
  - A new minimal bandwidth rerandomizable EUF-CMA-secure Type II SPS with optimal efficiency

Conclusions and Future Work

- Contributions of paper:
  - Template system for specifying SPS candidates
  - Tool to prune SPS schemes insecure in the GGM
  - A new minimal bandwidth rerandomizable EUF-CMA-secure Type II SPS with optimal efficiency
  - Tight bounds for the efficiency of Type II SPS of minimal bandwidth

Conclusions and Future Work

- Contributions of paper:
  - Template system for specifying SPS candidates
  - Tool to prune SPS schemes insecure in the GGM
  - A new minimal bandwidth rerandomizable EUF-CMA-secure Type II SPS with optimal efficiency
  - Tight bounds for the efficiency of Type II SPS of minimal bandwidth
  - A new minimal bandwidth EUF-RMA-secure Type II SPS with optimal efficiency

Conclusions and Future Work

- Contributions of paper:
  - Template system for specifying SPS candidates
  - Tool to prune SPS schemes insecure in the GGM
  - A new minimal bandwidth rerandomizable EUF-CMA-secure Type II SPS with optimal efficiency
  - Tight bounds for the efficiency of Type II SPS of minimal bandwidth
  - A new minimal bandwidth EUF-RMA-secure Type II SPS with optimal efficiency
- Future work:

Conclusions and Future Work

- Contributions of paper:
  - Template system for specifying SPS candidates
  - Tool to prune SPS schemes insecure in the GGM
  - A new minimal bandwidth rerandomizable EUF-CMA-secure Type II SPS with optimal efficiency
  - Tight bounds for the efficiency of Type II SPS of minimal bandwidth
  - A new minimal bandwidth EUF-RMA-secure Type II SPS with optimal efficiency
- Future work:
  - Our template system supports Type I and III settings out-of-the-box

Conclusions and Future Work

- Contributions of paper:
  - Template system for specifying SPS candidates
  - Tool to prune SPS schemes insecure in the GGM
  - A new minimal bandwidth rerandomizable EUF-CMA-secure Type II SPS with optimal efficiency
  - Tight bounds for the efficiency of Type II SPS of minimal bandwidth
  - A new minimal bandwidth EUF-RMA-secure Type II SPS with optimal efficiency
- Future work:
  - Our template system supports Type I and III settings out-of-the-box
  - However, more group elements in signature/verification key implies larger search spaces

Conclusions and Future Work

- Contributions of paper:
  - Template system for specifying SPS candidates
  - Tool to prune SPS schemes insecure in the GGM
  - A new minimal bandwidth rerandomizable EUF-CMA-secure Type II SPS with optimal efficiency
  - Tight bounds for the efficiency of Type II SPS of minimal bandwidth
  - A new minimal bandwidth EUF-RMA-secure Type II SPS with optimal efficiency
- Future work:
  - Our template system supports Type I and III settings out-of-the-box
  - However, more group elements in signature/verification key implies larger search spaces
  - Need more manual guidance as well as ideas on what to look for

Conclusions and Future Work

- Contributions of paper:
  - Template system for specifying SPS candidates
  - Tool to prune SPS schemes insecure in the GGM
  - A new minimal bandwidth rerandomizable EUF-CMA-secure Type II SPS with optimal efficiency
  - Tight bounds for the efficiency of Type II SPS of minimal bandwidth
  - A new minimal bandwidth EUF-RMA-secure Type II SPS with optimal efficiency
- Future work:
  - Our template system supports Type I and III settings out-of-the-box
  - However, more group elements in signature/verification key implies larger search spaces
  - Need more manual guidance as well as ideas on what to look for
  - Find better SPS under other constraints, e.g. automorphic signatures?

Conclusions and Future Work

- Contributions of paper:
  - Template system for specifying SPS candidates
  - Tool to prune SPS schemes insecure in the GGM
  - A new minimal bandwidth rerandomizable EUF-CMA-secure Type II SPS with optimal efficiency
  - Tight bounds for the efficiency of Type II SPS of minimal bandwidth
  - A new minimal bandwidth EUF-RMA-secure Type II SPS with optimal efficiency
- Future work:
  - Our template system supports Type I and III settings out-of-the-box
  - However, more group elements in signature/verification key implies larger search spaces
  - Need more manual guidance as well as ideas on what to look for
  - Find better SPS under other constraints, e.g. automorphic signatures?
  - Other cryptographic constructions with security games expressible in GGA tool language?

Questions?