How to Sign using Restricted Errors

Violetta Weger

29th NCM
Nordic Congress of Mathematicians with EMS

July 7, 2023
Motivation

2016 NIST standardization call for post-quantum PKE/KEM and signatures
Motivation

2016 NIST standardization call for post-quantum PKE/KEM and signatures

- PKE/KEM: 1 lattice-based, round 4: 3 code-based
- Signature schemes: 1 hash-based and 2 based on ideal lattices
Motivation

2016 NIST standardization call for post-quantum PKE/KEM and signatures

- PKE/KEM: 1 lattice-based, round 4: 3 code-based
- Signature schemes: 1 hash-based and 2 based on ideal lattices

2022 reopened NIST standardization call for signature schemes
Motivation

2016  NIST standardization call for post-quantum PKE/KEM and signatures

- PKE/KEM: 1 lattice-based, round 4: 3 code-based
- Signature schemes: 1 hash-based and 2 based on ideal lattices

2022  reopened NIST standardization call for signature schemes

- Deadline June 2023: CROSS: signature scheme with restricted errors
Motivation

2016 NIST standardization call for post-quantum PKE/KEM and signatures

- PKE/KEM: 1 lattice-based, round 4: 3 code-based
- Signature schemes: 1 hash-based and 2 based on ideal lattices

2022 reopened NIST standardization call for signature schemes

- Deadline June 2023: CROSS: signature scheme with restricted errors
- Received 50 signature schemes
Motivation

**2016** NIST standardization call for post-quantum PKE/KEM and signatures

- PKE/KEM: 1 lattice-based, round 4: 3 code-based
- Signature schemes: 1 hash-based and 2 based on ideal lattices

**2022** reopened NIST standardization call for signature schemes

- Deadline June 2023: **CROSS**: signature scheme with restricted errors
- Received 50 signature schemes
  - 5 code-based
Motivation

2016 NIST standardization call for post-quantum PKE/KEM and signatures

- PKE/KEM: 1 lattice-based, round 4: 3 code-based
- Signature schemes: 1 hash-based and 2 based on ideal lattices

2022 reopened NIST standardization call for signature schemes

- Deadline June 2023: CROSS: signature scheme with restricted errors
- Received 50 signature schemes
  - 5 code-based
  - 7 MPC in-the-head
Motivation

2016 NIST standardization call for post-quantum PKE/KEM and signatures

- PKE/KEM: 1 lattice-based, round 4: 3 code-based
- Signature schemes: 1 hash-based and 2 based on ideal lattices

2022 reopened NIST standardization call for signature schemes

- Deadline June 2023: CROSS: signature scheme with restricted errors
- Received 50 signature schemes
  - 5 code-based
  - 7 MPC in-the-head
  - 12 others
Coding Theory

Set Up

- Code $C \subseteq \mathbb{F}_q^n$ linear $k$-dimensional subspace
- $c \in C$ codeword
- $G \in \mathbb{F}_q^{k \times n}$ generator matrix $C = \{xG \mid x \in \mathbb{F}_q^k\}$
- $H \in \mathbb{F}_q^{(n-k) \times n}$ parity-check matrix $C = \{c \mid cH^\top = 0\}$
- $s = eH^\top$ syndrome
Coding Theory

Set Up

- Code $C \subseteq \mathbb{F}_q^n$ linear $k$-dimensional subspace
- $c \in C$ codeword
- $G \in \mathbb{F}_q^{k \times n}$ generator matrix  $C = \{xG \mid x \in \mathbb{F}_q^k\}$
- $H \in \mathbb{F}_q^{(n-k) \times n}$ parity-check matrix  $C = \{c \mid cH^\top = 0\}$
- $s = eH^\top$ syndrome
- Decode: find closest codeword
Coding Theory

Set Up

- Code $C \subseteq \mathbb{F}_q^n$ linear $k$-dimensional subspace
- $c \in C$ codeword
- $G \in \mathbb{F}_q^{k \times n}$ generator matrix $C = \{ xG \mid x \in \mathbb{F}_q^k \}$
- $H \in \mathbb{F}_q^{(n-k) \times n}$ parity-check matrix $C = \{ c \mid cH^\top = 0 \}$
- $s = eH^\top$ syndrome
- Decode: find closest codeword
- Hamming metric: $d_H(x, y) = | \{ i \mid x_i \neq y_i \} |$
Coding Theory

**Set Up**

- Code $C \subseteq \mathbb{F}_q^n$ linear $k$-dimensional subspace
- $c \in C$ codeword
- $G \in \mathbb{F}_q^{k \times n}$ generator matrix $C = \{xG \mid x \in \mathbb{F}_q^k\}$
- $H \in \mathbb{F}_q^{(n-k) \times n}$ parity-check matrix $C = \{c \mid cH^\top = 0\}$
- $s = eH^\top$ syndrome
- **Decode**: find closest codeword
- **Hamming metric**: $d_H(x, y) = \left| \{i \mid x_i \neq y_i\} \right|$
- **Minimum distance of a code**:

$$d(C) = \min\{d_H(x, y) \mid x \neq y \in C\}$$
Coding Theory

Set Up

- **Code** $C \subseteq \mathbb{F}_q^n$ linear $k$-dimensional subspace
- $c \in C$ codeword
- $G \in \mathbb{F}_q^{k \times n}$ generator matrix $C = \{xG \mid x \in \mathbb{F}_q^k\}$
- $H \in \mathbb{F}_q^{(n-k) \times n}$ parity-check matrix $C = \{c \mid cH^\top = 0\}$
- $s = eH^\top$ syndrome
- **Decode:** find closest codeword
- **Hamming metric:** $d_H(x, y) = |\{i \mid x_i \neq y_i\}|$
- **minimum distance of a code:**
  \[ d(C) = \min\{d_H(x, y) \mid x \neq y \in C\} \]
- **error-correction capacity:** $t = \lfloor (d(C) - 1)/2 \rfloor$
Hard Problems from Coding Theory

Algebraic structure
(Reed-Solomon, Goppa,..)
→ efficient decoders

random code

→ how hard to decode?

Decoding random linear code is NP-hard

First code-based cryptosystem
R. J. McEliece. "A public-key cryptosystem based on algebraic coding theory",
DSNP Report, 1978

Fastest solvers: ISD, exponential time
A. Becker, A. Joux, A. May, A. Meurer "Decoding random binary linear codes in \(2^{n/20}\): How \(1+1=0\) improves information set decoding",
Eurocrypt, 2012.

Violetta Weger — How to Sign using Restricted Errors 3/15
Hard Problems from Coding Theory

Algebraic structure
(Reed-Solomon, Goppa,..)
→ efficient decoders

→ how hard to decode?

• Decoding random linear code is NP-hard

Hard Problems from Coding Theory

Algebraic structure
(Reed-Solomon, Goppa,...)
→ efficient decoders

scrambling
\[ \varphi \]

Seemingly random code
\[ \langle G \rangle \rightarrow \langle \tilde{G} \rangle \]
→ how hard to decode?

- Decoding random linear code is NP-hard
- First code-based cryptosystem based on this problem


Hard Problems from Coding Theory

Algebraic structure
(Reed-Solomon, Goppa, ...)
→ efficient decoders

scrambling

Seemingly random code
→ how hard to decode?

- Decoding random linear code is NP-hard
- First code-based cryptosystem based on this problem
- Fastest solvers: ISD, exponential time


A. Becker, A. Joux, A. May, A. Meurer “Decoding random binary linear codes in $2^{n/20}$: How $1+1=0$ improves information set decoding”, Eurocrypt, 2012.
Idea of Signature Schemes

Signer

Verifier

2 Approaches for signatures:

- Hash-and-Sign
- Through ZK protocol
Idea of Signature Schemes

**Signer**

- **Key Generation:** P public, S secret

- **Signing:** use S and message m to generate signature \( \sigma \)

**Verifier**

- **Verification:** use P and message m to verify signature \( \sigma \)

2 Approaches for signatures:

- **Hash-and-Sign**
- **Through ZK protocol**
Idea of Signature Schemes

Signer

Verifier

Approaches for signatures:
• Hash-and-Sign
• Through ZK protocol
Idea of Signature Schemes

**Signer**

- **Key Generation:**
  \( P \) public, \( S \) secret

- **Signing:** use \( S \) and message \( m \) to generate signature \( \sigma \)

**Verifier**

- **Verification:** use \( P \) and message \( m \) to verify signature \( \sigma \)
Idea of Signature Schemes

**Signer**
- **Key Generation:**
  \( P \) public, \( S \) secret
- **Signing:** use \( S \) and message \( m \) to generate signature \( \sigma \)

**Verifier**
- **Verification:** use \( P \) and message \( m \) to verify signature \( \sigma \)
  - fast verification

**Approaches for signatures:**
- Hash-and-Sign
- Through ZK protocol
Idea of Signature Schemes

- **Key Generation:**
  - $P$ public, $S$ secret
- **Signing:** use $S$ and message $m$ to generate signature $\sigma$

Verifier

- **Verification:** use $P$ and message $m$ to verify signature $\sigma$
  - fast verification

2 Approaches for signatures:

- **Hash-and-Sign**
- **Through ZK protocol**
Idea of Signature Schemes

Signer

- Key Generation: \( P \) public, \( S \) secret
- Signing: use \( S \) and message \( m \) to generate signature \( \sigma \)

Verifier

- Verification: use \( P \) and message \( m \) to verify signature \( \sigma \)

2 Approaches for signatures:

- Hash-and-Sign
- Through ZK protocol
Hash-and-Sign
Following idea of McEliece

→ start with structured code $H$
→ publish scrambled code $HP$

Hash-and-Sign

Following idea of McEliece

- start with structured code $H$
- publish scrambled code $HP$
- large public key sizes

---

Hash-and-Sign

Following idea of McEliece

→ start with structured code $H$
→ publish scrambled code $HP$
→ large public key sizes

→ $\text{Hash}(m) = eH^\top$, $\text{wt}_H(e) \leq t$
→ signature $\sigma = eP$
Hash-and-Sign

Following idea of McEliece

→ start with structured code $H$
→ publish scrambled code $HP$
→ large public key sizes

→ $\text{Hash}(m) = eH^T$, $\text{wt}_H(e) \leq t$
→ signature $\sigma = eP$
→ slow signing

Hash-and-Sign

Following idea of McEliece

→ start with structured code $H$
→ publish scrambled code $HP$
→ large public key sizes

→ $\text{Hash}(m) = eH^T$, $\text{wt}_H(e) \leq t$
→ signature $\sigma = eP$
→ slow signing

- reduce key sizes:
  → use quasi-cyclic codes
  → use low density generators

Hash-and-Sign

Following idea of McEliece

→ start with structured code $H$
→ publish scrambled code $HP$
→ large public key sizes

→ $\text{Hash}(m) = eH^T$, $\text{wt}_H(e) \leq t$
→ signature $\sigma = eP$
→ slow signing

- reduce key sizes:
  → use quasi-cyclic codes
  → use low density generators
  → statistical attacks

Hash-and-Sign

Following idea of McEliece

→ start with structured code $H$
→ publish scrambled code $HP$
→ large public key sizes

→ $\text{Hash}(m) = eH^T$, $\text{wt}(e) \leq t$
→ signature $\sigma = eP$
→ slow signing

- reduce key sizes:
  → use quasi-cyclic codes
  → use low density generators
  → statistical attacks

Advertisement:


Idea of ZK Protocol

**Prover**
- $\mathcal{S}$: secret
- $\mathcal{P}$: related public key
- $c$: commitments to secret
- $r_b$: response to challenge $b$

**Verifier**
- $\mathcal{P},c \rightarrow b$:
- Recover $c$ from $r_b$ and $\mathcal{P}$

---

Idea of ZK Protocol

**Prover**

\[ S: \text{secret} \]
\[ \mathcal{P}: \text{related public key} \]
\[ c: \text{commitments to secret} \]
\[ r_b: \text{response to challenge } b \]

**Verifier**

\[ \mathcal{P}, c \]

\[ b \]

\[ r_b \]

\[ b: \text{challenge} \]

Recover \( c \) from \( r_b \) and \( \mathcal{P} \)

Idea of ZK Protocol

**Prover**

\[ S: \text{secret} \]
\[ \mathcal{P}: \text{related public key} \]
\[ c: \text{commitments to secret} \]
\[ r_b: \text{response to challenge} b \]

**Interaction**

\[ \mathcal{P}, c \rightarrow b \rightarrow r_b \]

**Verifier**

\[ b: \text{challenge} \]
Recover \( c \) from \( r_b \) and \( \mathcal{P} \)
Idea of ZK Protocol

Prover

- $S$: secret
- $P$: related public key
- $c$: commitments to secret
- $b$: Hash of message, $c$
- $r_b$: response to challenge $b$

Fiat-Shamir

$P, (b, r_b) \rightarrow$

Verifier

Recover $c$ from $r_b$ and $P$
Verify $b = \text{Hash}(m, c)$

---

Idea of ZK Protocol

**Prover**

- $S$: secret
- $\mathcal{P}$: related public key
- $c$: commitments to secret
- $b$: Hash of message, $c$
- $r_b$: response to challenge $b$

**Verifier**

$\mathcal{P}, (b, r_b) \rightarrow$

- Recover $c$ from $r_b$ and $\mathcal{P}$
- Verify $b = \text{Hash}(m, c)$

- $\alpha$ cheating probability, $\lambda$ bit security level
- **Rounds**: have to repeat ZK protocol $N$ times: $2^\lambda < (1/\alpha)^N$

---

**Code-based ZK Protocols**


### Syndrome Decoding Problem

Given parity-check matrix $H$, syndrome $s$, weight $t$, find $e$ s.t.

1. $s = eH^\top$
2. $\text{wt}_H(e) \leq t$

**Prover**

- $S$: $e$ of weight $t$,
- $P$: random $H$, $s = eH^\top$, $t$
- $c_1$: commitment to syndrome equation 1.
- $c_2$: commitment to weight 2.
- response: $r_1 = \varphi$, $r_2 = \varphi(e)$

**Verifier**

- $P, c_1, c_2$ −−−−→
- $b$ ←−−−−
- $r_b$ −−−−→
- recover $c_b$ from $r_b$ and $P$
Code-based ZK Protocols


Syndrome Decoding Problem

Given parity-check matrix $H$, syndrome $s$, weight $t$, find $e$ s.t.

1. $s = eH^\top$
2. $\text{wt}_H(e) \leq t$

**Prover**

$S$: $e$ of weight $t$,

$\mathcal{P}$: random $H$, $s = eH^\top$,

$c_1$: commitment to $s$,

$c_2$: commitment to weight $2$.

response: $r_1 = \varphi$, $r_2 = \varphi(e)$

**Verifier**

Problem: large cheating probability $\rightarrow$ big signature sizes

recover $c_b$ from $r_b$ and $\mathcal{P}$
Performance of Classical Approach

Classical CVE

- $\lambda = 128$ bit security level $\rightarrow N = 135 \quad \rightarrow$ public key size: 832 b
- $q = 31, n = 256, k = 204 \quad \rightarrow$ signature size: 43 kB
## Performance of Classical Approach

<table>
<thead>
<tr>
<th>Classical CVE</th>
</tr>
</thead>
<tbody>
<tr>
<td>• $\lambda = 128$ bit security level $\rightarrow N = 135$ $\rightarrow$ public key size: 832 b</td>
</tr>
<tr>
<td>• $q = 31$, $n = 256$, $k = 204$ $\rightarrow$ signature size: 43 kB</td>
</tr>
</tbody>
</table>

for a long time not been considered practical
Performance of Classical Approach

Classical CVE

- $\lambda = 128$ bit security level $\rightarrow N = 135$ $\rightarrow$ public key size: 832 b
- $q = 31$, $n = 256$, $k = 204$ $\rightarrow$ signature size: 43 kB

for a long time not been considered practical

Recent improvements through in-the-head computations
$\rightarrow$ smaller signature sizes $\sim 15$ kB


Performance of Classical Approach

Classical CVE

- $\lambda = 128$ bit security level $\rightarrow N = 135$ $\rightarrow$ public key size: 832 b
- $q = 31$, $n = 256$, $k = 204$ $\rightarrow$ signature size: 43 kB

for a long time not been considered practical

Recent improvements through in-the-head computations
$\rightarrow$ smaller signature sizes $\sim 15$ kB


Problem of Classical Approach

Classical CVE (1 round)

- public key size: seed of $H$, $s$; $\log_2(q) (n - k) < 0.1$ kB
- signature size: $\text{Hash}(m, c)$ and response: transformation $\varphi$ or $\varphi(e)$
Problem of Classical Approach

Classical CVE (1 round)

- public key size: seed of $H$, $s$; $\log_2(q)(n-k) < 0.1$ kB
- signature size: $\text{Hash}(m,c)$ and response: transformation $\varphi$ or $\varphi(e)$

Which $\varphi$ are allowed?
Problem of Classical Approach

Classical CVE (1 round)

- public key size: seed of \( H, s; \log_2(q)(n - k) < 0.1 \text{ kB} \)
- signature size: Hash\((m, c)\) and response: transformation \( \varphi \) or \( \varphi(e) \)

Which \( \varphi \) are allowed?

Syndrome Decoding Problem

Given \( H \in \mathbb{F}_q^{(n-k)\times n}, s \in \mathbb{F}_q^{n-k} \), weight \( t \), find \( e \in \mathbb{F}_q^n \) such that \( s = eH^\top \) and \( \text{wt}_H(e) \leq t \).

\[ e \begin{bmatrix} 0 & 0 & 0 & 0 \end{bmatrix} \xrightarrow{\varphi} \begin{bmatrix} 0 & \_ & \_ & 0 \end{bmatrix} e' \]
Problem of Classical Approach

Classical CVE (1 round)

- public key size: seed of $H$, $s$; $\log_2(q)(n - k) < 0.1$ kB
- signature size: $\text{Hash}(m, c)$ and response: transformation $\varphi$ or $\varphi(e)$

Which $\varphi$ are allowed?

Syndrome Decoding Problem

Given $H \in \mathbb{F}_q^{(n-k) \times n}$, $s \in \mathbb{F}_q^{n-k}$, weight $t$, find $e \in \mathbb{F}_q^n$ such that $s = eH^\top$ and $w_{H}(e) \leq t$.

$e \begin{array}{cccc} 0 & 0 & 0 & 0 \end{array} \xrightarrow{\varphi} \begin{array}{cccc} 0 & 0 & 0 & 0 \end{array} = e'$

$\rightarrow \varphi$: linear isometries of Hamming metric:
permutation + scalar multiplication
Problem of Classical Approach

Classical CVE (1 round)

- Public key size: seed of $H$, $s$; $\log_2(q)(n-k) < 0.1$ kB
- Signature size: $\varphi(e) : t \log_2(q-1) + t \log_2(n)$ or $\varphi : n \log_2(q-1) + n \log_2(n)$

Which $\varphi$ are allowed?

Syndrome Decoding Problem

Given $H \in \mathbb{F}_q^{(n-k) \times n}$, $s \in \mathbb{F}_q^{n-k}$, weight $t$, find $e \in \mathbb{F}_q^n$ such that $s = eH^\top$ and $\text{wt}_H(e) \leq t$.

$\varphi :$ linear isometries of Hamming metric:
permutation + scalar multiplication

$e$ 0 0 0 0 0 0 0 0 0 0 $\varphi$ 0 0 0 0 0 0 0 0 0 0 $e'$

$\rightarrow \varphi :$ linear isometries of Hamming metric:
permutation + scalar multiplication
Restricted Errors

Syndrome Decoding Problem

Given $H \in \mathbb{F}_q^{(n-k) \times n}$, $s \in \mathbb{F}_q^{n-k}$, weight $t$, find $e \in \mathbb{F}_q^n$ such that $s = eH^\top$ and $\text{wt}(e) \leq t$.

Can we avoid permutations?
Restricted Errors

Syndrome Decoding Problem

Given $H \in \mathbb{F}_q^{(n-k) \times n}$, $s \in \mathbb{F}_q^{n-k}$, weight $t$, find $e \in \mathbb{F}_q^n$ such that $s = eH^\top$ and $\text{wt}(e) \leq t$.

Can we avoid permutations - but keep the hardness of the problem?

↓

Syndrome Decoding Problem

Given $H \in \mathbb{F}_q^{(n-k) \times n}$, syndrome $s \in \mathbb{F}_q^{n-k}$, find $e \in \mathbb{F}_q^n$ such that $s = eH^\top$. 
Restricted Errors

Syndrome Decoding Problem

Given $H \in \mathbb{F}_q^{(n-k) \times n}$, $s \in \mathbb{F}_q^{n-k}$, weight $t$, find $e \in \mathbb{F}_q^n$ such that $s = eH^\top$ and $\text{wt}(e) \leq t$.

Can we avoid permutations - but keep the hardness of the problem?

Restricted Syndrome Decoding Problem

Given $H \in \mathbb{F}_q^{(n-k) \times n}$, syndrome $s \in \mathbb{F}_q^{n-k}$, $E \subseteq \mathbb{F}_q^*$, find $e \in E^n$ such that $s = eH^\top$. 

$\varphi$
Restricted Errors

Syndrome Decoding Problem

Given \( H \in \mathbb{F}_q^{(n-k) \times n} \), \( s \in \mathbb{F}_q^{n-k} \), weight \( t \), find \( e \in \mathbb{F}_q^n \) such that \( s = eH^\top \) and \( \text{wt}(e) \leq t \).

Can we avoid permutations - but keep the hardness of the problem?

\[ \begin{array}{c c c c c c c c c}
\text{e} & 0 & 0 & 0 & 0 \\
\phi & \Downarrow \end{array} \begin{array}{c c c c c c c c c}
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & e' \\
\end{array} \]

Restricted Syndrome Decoding Problem

Given \( H \in \mathbb{F}_q^{(n-k) \times n} \), syndrome \( s \in \mathbb{F}_q^{n-k} \), \( E \subseteq \mathbb{F}_q^\star \), find \( e \in E^n \) such that \( s = eH^\top \).

How to choose \( E \)?
Restricted Errors


\[(E, \cdot) < (\mathbb{F}_q^*, \cdot) \rightarrow g \in \mathbb{F}_q^* \text{ of prime order } z \rightarrow E = \{g^i \mid i \in \{1, \ldots, z\}\}\]

\[
q = 13 \quad \rightarrow \quad g = 3 \quad \text{order } z = 3 \quad \rightarrow \quad E = \{1, 3, 9\}
\]
Restricted Errors


\[(\mathbb{E}, \cdot) \leq (\mathbb{F}_q^*, \cdot) \rightarrow g \in \mathbb{F}_q^* \text{ of prime order } z \rightarrow \mathbb{E} = \{g^i \mid i \in \{1, \ldots, z\}\}\]

\[q = 13 \rightarrow g = 3 \text{ order } z = 3 \rightarrow \mathbb{E} = \{1, 3, 9\}\]

\[(\mathbb{E}^n, \cdot)\]

\[
\begin{array}{c}
q = 13 \\
g = 3 \\
z = 3 \\
\mathbb{E} = \{1, 3, 9\}
\end{array}
\]

\[(\mathbb{F}_z^n, +)\]

\[\ell \rightarrow\]
Restricted Errors


\[(\mathbb{E}, \cdot) < (\mathbb{F}_q^*, \cdot) \rightarrow g \in \mathbb{F}_q^* \text{ of prime order } z \rightarrow \mathbb{E} = \{g^i | i \in \{1, \ldots, z\}\}\]

\[q = 13 \rightarrow g = 3 \text{ order } z = 3 \rightarrow \mathbb{E} = \{1, 3, 9\}\]

\[\begin{align*}
(\mathbb{E}^n, \star) \\
\bullet & e = (1, 9, 3, 3) \in \{1, 3, 9\}^4 \\
& \ell(e) = (0, 2, 1, 1) \in \mathbb{F}_3^4
\end{align*}\]

\[\begin{align*}
(\mathbb{F}_z^n, +) \\
\bullet & \ell(e) = (0, 2, 1, 1) \in \mathbb{F}_3^4
\end{align*}\]
Restricted Errors


\[
(\mathbb{E}, \cdot) < (\mathbb{F}_q^*, \cdot) \rightarrow g \in \mathbb{F}_q^* \text{ of prime order } z \rightarrow \mathbb{E} = \{g^i \mid i \in \{1, \ldots, z\}\}
\]

\[q = 13 \rightarrow g = 3 \text{ order } z = 3 \rightarrow \mathbb{E} = \{1, 3, 9\}\]

\[
\begin{array}{c}
(\mathbb{E}^n, \ast) \\
\bullet e = (1, 9, 3, 3) \in \{1, 3, 9\}^4 \\
\bullet \text{trans.: } \varphi : \mathbb{E}^n \rightarrow \mathbb{E}^n, e \leftrightarrow e \ast e' \\
\end{array}
\]

\[
\begin{array}{c}
(\mathbb{F}_z^n, +) \\
\bullet \ell(e) = (0, 2, 1, 1) \in \mathbb{F}_3^4 \\
\bullet \ell(\varphi) \in \mathbb{F}_z^n
\end{array}
\]
Restricted Errors


\[(\mathbb{E}, \cdot) < (\mathbb{F}_q^*, \cdot) \rightarrow g \in \mathbb{F}_q^* \text{ of prime order } z \rightarrow \mathbb{E} = \{g^i \mid i \in \{1, \ldots, z\}\}\]

\[q = 13 \rightarrow g = 3 \text{ order } z = 3 \rightarrow \mathbb{E} = \{1, 3, 9\}\]

\[(\mathbb{E}^n, \ast)\]
- \(e = (1, 9, 3, 3) \in \{1, 3, 9\}^4\)
- trans.: \(\varphi : \mathbb{E}^n \rightarrow \mathbb{E}^n, e \mapsto e \ast e'\)
- \(\varphi : e' = (3, 9, 1, 3) \in \mathbb{E}^n\)

\[(\mathbb{F}_z^n, +)\]
- \(\ell(e) = (0, 2, 1, 1) \in \mathbb{F}_3^4\)
- \(\ell(\varphi) \in \mathbb{F}_z^n\)
- \(\ell(\varphi) : \ell(e') = (1, 2, 0, 1) \in \mathbb{F}_3^4\)
Restricted Errors


\[(\mathbb{E}, \cdot) \leq (\mathbb{F}_q^\star, \cdot) \rightarrow g \in \mathbb{F}_q^\star \text{ of prime order } z \rightarrow \mathbb{E} = \{g^i \mid i \in \{1, \ldots, z\}\}\]

\[q = 13 \quad \rightarrow \quad g = 3 \text{ order } z = 3 \quad \rightarrow \quad \mathbb{E} = \{1, 3, 9\}\]

\[
\begin{array}{|c|}
\hline
(\mathbb{E}^n, \ast) \\
\hline
- e = (1, 9, 3, 3) \in \{1, 3, 9\}^4 \\
- \text{trans.: } \varphi : \mathbb{E}^n \rightarrow \mathbb{E}^n, e \mapsto e \ast e' \\
- \varphi : e' = (3, 9, 1, 3) \in \mathbb{E}^n \\
- \varphi(e) = e \ast e' \in (\mathbb{E}^n, \ast) \\
\hline
\end{array}
\]

\[
\begin{array}{|c|}
\hline
(\mathbb{F}_z^n, +) \\
\hline
- \ell(e) = (0, 2, 1, 1) \in \mathbb{F}_3^4 \\
- \ell(\varphi) \in \mathbb{F}_z^n \\
- \ell(\varphi) : \ell(e') = (1, 2, 0, 1) \in \mathbb{F}_3^4 \\
- \ell(e) + \ell(e') \in (\mathbb{F}_z^n, +) \\
\hline
\end{array}
\]
Restricted Errors


\((\mathbb{E}, \cdot) < (\mathbb{F}_q^*, \cdot) \rightarrow g \in \mathbb{F}_q^* \text{ of prime order } z \rightarrow \mathbb{E} = \{g^i \mid i \in \{1, \ldots, z\}\}\)

\(q = 13 \rightarrow g = 3 \text{ order } z = 3 \rightarrow \mathbb{E} = \{1, 3, 9\}\)

\((\mathbb{E}^n, \ast)\)

- \(e = (1, 9, 3, 3) \in \{1, 3, 9\}^4\)
- trans.: \(\varphi : \mathbb{E}^n \rightarrow \mathbb{E}^n, e \mapsto e \ast e'\)
- \(\varphi : e' = (3, 9, 1, 3) \in \mathbb{E}^n\)
- \(\varphi(e) = e \ast e' \in (\mathbb{E}^n, \ast)\)
- \(\varphi(e) = (1, 9, 3, 3) \ast (3, 9, 1, 3)\)

\((\mathbb{F}_z^n, +)\)

- \(\ell(e) = (0, 2, 1, 1) \in \mathbb{F}_3^4\)
- \(\ell(\varphi) \in \mathbb{F}_z^n\)
- \(\ell(\varphi) : \ell(e') = (1, 2, 0, 1) \in \mathbb{F}_3^4\)
- \(\ell(e) + \ell(e') \in (\mathbb{F}_z^n, +)\)
- \((0, 2, 1, 1) + (1, 2, 0, 1)\)

Can do even better
Restricted Errors


\((\mathbb{E}, \cdot) \triangleleft (\mathbb{F}_q^*, \cdot) \rightarrow g \in \mathbb{F}_q^* \text{ of prime order } z \rightarrow \mathbb{E} = \{g^i \mid i \in \{1, \ldots, z\}\}\)

\(q = 13 \rightarrow g = 3 \text{ order } z = 3 \rightarrow \mathbb{E} = \{1, 3, 9\}\)

\(\left(\mathbb{E}^n, \ast\right)\)

- \(e = (1, 9, 3, 3) \in \{1, 3, 9\}^4\)
- trans.: \(\varphi : \mathbb{E}^n \rightarrow \mathbb{E}^n, e \mapsto e \ast e'\)
- \(\varphi : e' = (3, 9, 1, 3) \in \mathbb{E}^n\)
- \(\varphi(e) = e \ast e' \in (\mathbb{E}^n, \ast)\)
- \(\varphi(e) = (1, 9, 3, 3) \ast (3, 9, 1, 3)\)

\(\left(\mathbb{F}_z^n, +\right)\)

- \(\ell(e) = (0, 2, 1, 1) \in \mathbb{F}_3^4\)
- \(\ell(\varphi) \in \mathbb{F}_z^n\)
- \(\ell(\varphi) : \ell(e') = (1, 2, 0, 1) \in \mathbb{F}_3^4\)
- \(\ell(e) + \ell(e') \in (\mathbb{F}_z^n, +)\)
- \((0, 2, 1, 1) + (1, 2, 0, 1)\)

new size: before: \(n \log_2((q - 1)n)\)
fast arithmetic: before: \((\mathbb{F}_q^n, \cdot)\)

new: \(n \log_2(z)\)
new: \((\mathbb{F}_z^n, +)\)
Restricted Errors

M. Baldi, S. Bitzer, A. Pavoni, P. Santini, A. Wachter-Zeh, **V.W.** “Zero knowledge protocols and signatures from the restricted syndrome decoding problem”, Preprint, 2023

$$(\mathbb{E}, \cdot) < (\mathbb{F}_q^*, \cdot) \rightarrow g \in \mathbb{F}_q^* \text{ of prime order } z \rightarrow \mathbb{E} = \{ g^i \mid i \in \{1, \ldots, z\} \}$$

$q = 13 \rightarrow g = 3 \text{ order } z = 3 \rightarrow \mathbb{E} = \{1, 3, 9\}$

\[
\begin{align*}
\mathbb{E}^n, \ast & \\
& \text{• } e = (1, 9, 3, 3) \in \{1, 3, 9\}^4 \\
& \text{• } \text{trans.: } \varphi : \mathbb{E}^n \rightarrow \mathbb{E}^n, e \mapsto e \ast e' \\
& \text{• } \varphi : e' = (3, 9, 1, 3) \in \mathbb{E}^n \\
& \text{• } \varphi(e) = e \ast e' \in (\mathbb{E}^n, \ast) \\
& \text{• } \varphi(e) = (1, 9, 3, 3) \ast (3, 9, 1, 3)
\end{align*}
\]

\[
\begin{align*}
\mathbb{F}_z^n, + & \\
& \text{• } \ell(e) = (0, 2, 1, 1) \in \mathbb{F}_3^n \\
& \text{• } \ell(\varphi) \in \mathbb{F}_z^n \\
& \text{• } \ell(\varphi) : \ell(e') = (1, 2, 0, 1) \in \mathbb{F}_3^n \\
& \text{• } \ell(e) + \ell(e') \in (\mathbb{F}_z^n, +) \\
& \text{• } (0, 2, 1, 1) + (1, 2, 0, 1)
\end{align*}
\]

Can do even better

\[
\begin{align*}
\text{new size:} & \quad \text{before: } n \log_2((q - 1)n) \\
\text{fast arithmetic:} & \quad \text{before: } (\mathbb{F}_q^n, \cdot) \\
\text{new:} & \quad n \log_2(z) \\
\text{new:} & \quad (\mathbb{F}_z^n, +)
\end{align*}
\]
Restricted-G SDP

Restricted Syndrome Decoding Problem

Given $H \in \mathbb{F}_q^{(n-k) \times n}$, $s \in \mathbb{F}_q^{n-k}$, $\mathbb{E} \subseteq \mathbb{F}_q^*$, find $e \in \mathbb{E}^n$ s.t. $s = eH^\top$.

- $(\mathbb{E}^n, \ast) \cong (\mathbb{F}_z^n, +)$
- $e = (1, 9, 3, 3) \in \mathbb{E}^4 = \{1, 3, 9\}^4$
**Restricted-G Syndrome Decoding Problem**

Given $H \in \mathbb{F}_q^{(n-k) \times n}$, $s \in \mathbb{F}_q^{n-k}$, $E \subseteq \mathbb{F}_q^*$, $G = \langle x_1, \ldots, x_m \rangle \leq E^n$ find $e \in G$ s.t. $s = eH^\top$.

- $(\mathbb{E}^n, \ast) \cong (\mathbb{F}_z^n, +)$

  $\Rightarrow$ Subgroup $(G, \ast) \leq (\mathbb{E}^n, \ast)$
  
  $G = \langle x_1, \ldots, x_m \rangle$

  $\Rightarrow$ $e' = \prod_{i=1}^{m} x_i^{u_i} \in G$

- $e = (1, 9, 3, 3) \not\in G$

  - $x_1 = (9, 1, 9, 1), x_2 = (9, 9, 1, 9), x_3 = (1, 9, 9, 3)$
  - $e' = x_1^2 \ast x_2^1 \ast x_3^0 = (1, 9, 3, 9) \in G$
Restricted-G Syndrome Decoding Problem

Given \( H \in \mathbb{F}_q^{(n-k) \times n} \), \( s \in \mathbb{F}_q^{n-k} \), \( E \subseteq \mathbb{F}_q^* \), \( G = \langle x_1, \ldots, x_m \rangle \leq E^n \) find \( e \in G \) s.t. \( s = eH^\top \).

- \((\mathbb{F}^n, \ast) \cong (\mathbb{F}_z^n, +)\)

\rightarrow \text{Subgroup } (G, \ast) \leq (\mathbb{F}^n, \ast)
\[ G = \langle x_1, \ldots, x_m \rangle \]
\[ e' = \prod_{i=1}^m x_i^{u_i} \in G \]
- \( M_G = [\ell(x_i)] \in \mathbb{F}_z^{m \times n} \)
- \( \ell(e') = yM_G, \ y \in \mathbb{F}_z^m \)
- \( \ell(e') = (0, 2, 1, 2) = (2, 1, 0)M_G \)

- \( e = (1, 9, 3, 3) \notin G \)

- \( x_1 = (9, 1, 9, 1), x_2 = (9, 9, 1, 9), x_3 = (1, 9, 9, 3) \)

- \( e' = x_1^2 \ast x_2^1 \ast x_3^0 = (1, 9, 3, 9) \in G \)

- \( M_G = \begin{pmatrix} 2 & 0 & 2 & 0 \\ 2 & 2 & 0 & 2 \\ 0 & 2 & 2 & 1 \end{pmatrix} \)
Restricted-G SDP

**Restricted-G Syndrome Decoding Problem**

Given \( H \in \mathbb{F}_q^{(n-k) \times n} \), \( s \in \mathbb{F}_q^{n-k} \), \( E \subseteq \mathbb{F}_q^* \), \( G = \langle x_1, \ldots, x_m \rangle \leq E^n \) find \( e \in G \) s.t. \( s = eH^\top \).

- \((E^n, \ast) \cong (\mathbb{F}_z^n, +)\)
- Subgroup \((G, \ast) \leq (E^n, \ast)\)
- \( G = \langle x_1, \ldots, x_m \rangle \)
- \( e' = \prod_{i=1}^m x_i^{u_i} \in G \)
- \( M_G = \llbracket \ell(x_i) \rrbracket \in \mathbb{F}_z^{m \times n} \)
- \( \ell(e') = yM_G, \ y \in E^m \)
- fast arithmetic

smaller sizes: \( n \log_2((q - 1)n) \) → rest.: \( n \log_2(z) \) → rest.-G: \( m \log_2(z) \)

- \( e = (1, 9, 3, 3) \not\in G \)
- \( x_1 = (9, 1, 9, 1), x_2 = (9, 9, 1, 9), x_3 = (1, 9, 9, 3) \)
- \( e' = x_1^2 \ast x_2^1 \ast x_3^0 = (1, 9, 3, 9) \in G \)
- \( M_G = \begin{pmatrix} 2 & 0 & 2 & 0 \\ 2 & 2 & 0 & 2 \\ 0 & 2 & 2 & 1 \end{pmatrix} \)
- \( \ell(e') = (0, 2, 1, 2) = (2, 1, 0)M_G \)
Restricted Syndrome Decoding Problem

Given $H \in \mathbb{F}_q^{(n-k) \times n}$, $s \in \mathbb{F}_q^{n-k}$, $E \subseteq \mathbb{F}_q^*$, find $e \in E^n$ s.t. $s = eH^\top$. 

NP hard for $E < \mathbb{F}_q^*$
## Restricted Syndrome Decoding Problem

Given \( H \in \mathbb{F}_q^{(n-k) \times n} \), \( s \in \mathbb{F}_q^{n-k} \), \( E \subseteq \mathbb{F}_q^* \), find \( e \in \mathbb{E}^n \) s.t. \( s = eH^\top \).

\[
\rightarrow \text{NP hard for } E < \mathbb{F}_q^*
\]
Restricted Syndrome Decoding Problem

Given $H \in \mathbb{F}_q^{(n-k) \times n}$, $s \in \mathbb{F}_q^{n-k}$, $E \subseteq \mathbb{F}_q^*$, find $e \in \mathbb{F}_q^n$ s.t. $s = eH^\top$.

→ NP hard for $E < \mathbb{F}_q^*$

Information set decoding?

Restricted Syndrome Decoding Problem

Given $H \in \mathbb{F}_q^{(n-k) \times n}$, $s \in \mathbb{F}_q^{n-k}$, $E \subseteq \mathbb{F}_q^*$, find $e \in \mathbb{E}^n$ s.t. $s = eH^\top$.

→ NP hard for $E < \mathbb{F}_q^*$

Information set decoding?

- Restricted errors first introduced: $g = -1 \rightarrow z = 2$


Is this Safe?

Restricted Syndrome Decoding Problem

Given $H \in \mathbb{F}_q^{(n-k) \times n}$, $s \in \mathbb{F}_q^{n-k}$, $E \subseteq \mathbb{F}_q^*$, find $e \in \mathbb{E}^n$ s.t. $s = eH^T$.

→ NP hard for $E < \mathbb{F}_q^*$

Information set decoding?

- Restricted errors first introduced: $g = -1 \rightarrow z = 2$
- several proposals for small $z$
e.g. $z = 4, 6$

→ additive structure on $E$ not safe


Is this Safe?

→ additive structure on $E$ not safe
Is this Safe?

→ additive structure on $E$ not safe
Is this Safe?

→ additive structure on $\mathbb{E}$ not safe

\[
-g \quad -g + 1
\]

\[
-1 \quad 0 \quad 1
\]

\[
\begin{align*}
&g - 1 \\
&g
\end{align*}
\]
Is this Safe?

$\rightarrow$ additive structure on $\mathbb{E}$ not safe
→ additive structure on $\mathbb{E}$ not safe
Is this Safe?

→ additive structure on $E$ not safe
Is this Safe?

→ additive structure on $\mathbb{E}$ not safe

→ our $\mathbb{E}$ has no additive structure
## Performance of Restricted-$G$ Signatures

<table>
<thead>
<tr>
<th>Restricted CVE</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Classical</strong>: $q = 31$, $n = 256$, $k = 204$</td>
</tr>
<tr>
<td><strong>Rest.</strong>: $q = 127$, $z = 7$, $n = 2$, $k = 127$</td>
</tr>
<tr>
<td><strong>Rest.-$G$</strong>: $q = 509$, $z = 127$, $m = 24$, $n = 2$, $k = 42$</td>
</tr>
</tbody>
</table>
# Performance of Restricted-$G$ Signatures

## Restricted CVE

<table>
<thead>
<tr>
<th>Description</th>
<th>Parameters</th>
<th>Signature Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>Classical</td>
<td>$q = 31$, $n = 256$, $k = 204$</td>
<td>$43$ kB</td>
</tr>
<tr>
<td>In-the-head computations</td>
<td></td>
<td>$15$ kB</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Description</th>
<th>Parameters</th>
<th>Signature Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rest. $G$</td>
<td>$q = 509$, $z = 127$, $m = 24$, $n = 2$, $k = 42$</td>
<td>$7$ kB</td>
</tr>
</tbody>
</table>

## Conclusion

- Can replace classical SDP with Restricted SDP/Restricted-$G$ SDP in any code-based ZK protocol.
- Achieve smaller signature sizes, smaller running times.
## Performance of Restricted-G Signatures

<table>
<thead>
<tr>
<th>Restricted CVE</th>
</tr>
</thead>
<tbody>
<tr>
<td>• classical: $q = 31, n = 256, k = 204$ → signature size: 43 kB</td>
</tr>
<tr>
<td>• in-the-head computations → signature size: 15 kB</td>
</tr>
<tr>
<td>• rest.: $q = 127, z = 7, n = 2k = 127$ → signature size: 10 kB</td>
</tr>
</tbody>
</table>
## Performance of Restricted-G Signatures

<table>
<thead>
<tr>
<th>Restricted CVE</th>
<th>classical: $q = 31, n = 256, k = 204$</th>
<th>$\rightarrow$ signature size: 43 kB</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>in-the-head computations</td>
<td>$\rightarrow$ signature size: 15 kB</td>
</tr>
<tr>
<td></td>
<td>rest.: $q = 127, z = 7, n = 2k = 127$</td>
<td>$\rightarrow$ signature size: 10 kB</td>
</tr>
<tr>
<td></td>
<td>rest.-G: $q = 509, z = 127, m = 24, n = 2k = 42$</td>
<td>$\rightarrow$ signature size: 7 kB</td>
</tr>
</tbody>
</table>

Conclusion

- Can replace classical SDP with Restricted SDP/Restricted-G SDP in any code-based ZK protocol.
- Achieve smaller signature sizes, smaller running times.
Performance of Restricted-$G$ Signatures

Restricted CVE

- classical: $q = 31, n = 256, k = 204$ → signature size: 43 kB
- in-the-head computations
- rest.: $q = 127, z = 7, n = 2k = 127$ → signature size: 15 kB
- rest.-$G$: $q = 509, z = 127, m = 24, n = 2k = 42$ → signature size: 10 kB
- rest.-$G$: $q = 509, z = 127, m = 24, n = 2k = 42$ → signature size: 7 kB

Conclusion

- Can replace classical SDP with Restricted SDP/Restricted-$G$ SDP in any code-based ZK protocol.
- Achieve smaller signature sizes, smaller running times
Questions?

CROSS

Codes & Restricted Objects Signature Scheme
http://cross-crypto.com/

Thank you!
Running times

Running time given in kCycles, CROSS has only PoC, no optimization, parallelization

<table>
<thead>
<tr>
<th>Scheme</th>
<th>Key gen.</th>
<th>Signature gen.</th>
<th>Verification</th>
</tr>
</thead>
<tbody>
<tr>
<td>SPHINCS</td>
<td>1794</td>
<td>5802</td>
<td>6506</td>
</tr>
<tr>
<td>Dilitium</td>
<td>49</td>
<td>140</td>
<td>61</td>
</tr>
<tr>
<td>CROSS</td>
<td>19</td>
<td>187</td>
<td>184</td>
</tr>
</tbody>
</table>
$G = \langle x_1, \ldots, x_m \rangle$: use generators?

No: $\prod_{i=1}^{m} x_i^{u_i} H^\top = s$

→ not compatible  unlike $\sum_{i=1}^{m} \lambda_i x_i H^\top = s$
Solving Restricted SDP in subgroup $G$

- we want $q, z$ such that $E$ has no additive structure
- Publicly known: $x_1, \ldots, x_m$ generators of multiplicative group $G$
- $x_\ell = (g^{i_1,\ell}, \ldots, g^{i_n,\ell})$
- define $M_G \in \mathbb{F}_z^{m \times n}$ having rows $(i_1,\ell, \ldots, i_n,\ell)$

\[
M_G = \begin{bmatrix}
i_1,\ell & \cdots & i_n,\ell
\end{bmatrix} 
\]

$m' \geq \min \left\{ |J|, \frac{\lambda}{\log_2(z)} \right\}$ → no improvement over enumerating all possible errors in these positions
## Comparison

<table>
<thead>
<tr>
<th>Scheme</th>
<th>Public Key size</th>
<th>Signature size</th>
<th>Total size</th>
<th>Variant</th>
</tr>
</thead>
<tbody>
<tr>
<td>SPHINCS+</td>
<td>&lt;0.1</td>
<td>16.7</td>
<td>16.7</td>
<td>Fast</td>
</tr>
<tr>
<td></td>
<td>&lt;0.1</td>
<td>7.7</td>
<td>7.7</td>
<td>Short</td>
</tr>
<tr>
<td>Falcon</td>
<td>0.9</td>
<td>0.6</td>
<td>1.5</td>
<td>-</td>
</tr>
<tr>
<td>Dilitiumium</td>
<td>1.3</td>
<td>2.4</td>
<td>3.7</td>
<td>-</td>
</tr>
<tr>
<td>CROSS</td>
<td>0.1</td>
<td>7.7</td>
<td>7.8</td>
<td>Fast</td>
</tr>
<tr>
<td></td>
<td>0.1</td>
<td>7.2</td>
<td>7.3</td>
<td>Short</td>
</tr>
<tr>
<td>GPS</td>
<td>0.1</td>
<td>24.0</td>
<td>24.1</td>
<td>Fast</td>
</tr>
<tr>
<td></td>
<td>0.1</td>
<td>19.8</td>
<td>19.9</td>
<td>Short</td>
</tr>
<tr>
<td>FJR</td>
<td>0.1</td>
<td>22.6</td>
<td>22.7</td>
<td>Fast</td>
</tr>
<tr>
<td></td>
<td>0.1</td>
<td>16.0</td>
<td>16.1</td>
<td>Short</td>
</tr>
<tr>
<td>SDItH</td>
<td>0.1</td>
<td>11.5</td>
<td>11.6</td>
<td>Fast</td>
</tr>
<tr>
<td></td>
<td>0.1</td>
<td>8.3</td>
<td>8.4</td>
<td>Short</td>
</tr>
<tr>
<td>Ret. of SDItH</td>
<td>0.1</td>
<td>12.1</td>
<td>12.1</td>
<td>Fast, V3</td>
</tr>
<tr>
<td></td>
<td>0.1</td>
<td>5.7</td>
<td>5.8</td>
<td>Shortest, V3</td>
</tr>
</tbody>
</table>
## Comparison

<table>
<thead>
<tr>
<th>Scheme</th>
<th>Public Key size</th>
<th>Signature size</th>
<th>Total size</th>
<th>Variant</th>
</tr>
</thead>
<tbody>
<tr>
<td>WAVE</td>
<td>3200</td>
<td>2.1</td>
<td>3202</td>
<td>-</td>
</tr>
<tr>
<td>Durandal</td>
<td>15.2</td>
<td>4.1</td>
<td>19.3</td>
<td>-</td>
</tr>
<tr>
<td>Ideal Rank BG</td>
<td>0.5</td>
<td>8.4</td>
<td>8.9</td>
<td>Fast</td>
</tr>
<tr>
<td></td>
<td>0.5</td>
<td>6.1</td>
<td>6.6</td>
<td>Short</td>
</tr>
<tr>
<td>MinRank Fen</td>
<td>18.2</td>
<td>9.3</td>
<td>27.5</td>
<td>Fast</td>
</tr>
<tr>
<td></td>
<td>18.2</td>
<td>7.1</td>
<td>25.3</td>
<td>Short</td>
</tr>
<tr>
<td>Rank SDP Fen</td>
<td>0.9</td>
<td>7.4</td>
<td>8.3</td>
<td>Fast</td>
</tr>
<tr>
<td></td>
<td>0.9</td>
<td>5.9</td>
<td>6.8</td>
<td>Short</td>
</tr>
<tr>
<td>Beu</td>
<td>0.1</td>
<td>18.4</td>
<td>18.5</td>
<td>Fast</td>
</tr>
<tr>
<td></td>
<td>0.1</td>
<td>12.1</td>
<td>12.2</td>
<td>Short</td>
</tr>
<tr>
<td>PKP BG</td>
<td>0.1</td>
<td>9.8</td>
<td>9.9</td>
<td>Fast</td>
</tr>
<tr>
<td></td>
<td>0.1</td>
<td>8.8</td>
<td>8.9</td>
<td>Short</td>
</tr>
<tr>
<td>FuLeeca</td>
<td>1.3</td>
<td>1.1</td>
<td>2.4</td>
<td>-</td>
</tr>
</tbody>
</table>
## Hash-and-Sign: CFS

<table>
<thead>
<tr>
<th>PROVER</th>
<th>VERIFIER</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>KEY GENERATION</strong></td>
<td></td>
</tr>
<tr>
<td>( S = H ) parity-check matrix</td>
<td></td>
</tr>
<tr>
<td>( \mathcal{P} = (t, HP) ) permuted ( H )</td>
<td></td>
</tr>
<tr>
<td><strong>SIGNING</strong></td>
<td></td>
</tr>
<tr>
<td>Choose message ( m )</td>
<td></td>
</tr>
<tr>
<td>( s = \text{Hash}(m) )</td>
<td></td>
</tr>
<tr>
<td>Find ( e ): ( s = eH^\top = eP(HP)^\top ), and ( \text{wt}(e) \leq t )</td>
<td></td>
</tr>
<tr>
<td>( m, eP \rightarrow )</td>
<td></td>
</tr>
<tr>
<td><strong>VERIFICATION</strong></td>
<td></td>
</tr>
<tr>
<td>Check if ( \text{wt}(eP) \leq t ) and ( eP(HP)^\top = \text{Hash}(m) )</td>
<td></td>
</tr>
</tbody>
</table>
Hash-and-Sign: CFS

<table>
<thead>
<tr>
<th>PROVER</th>
<th>VERIFIER</th>
</tr>
</thead>
<tbody>
<tr>
<td>KEY GENERATION</td>
<td></td>
</tr>
<tr>
<td>$S = H$ parity-check matrix</td>
<td></td>
</tr>
<tr>
<td>$\mathcal{P} = (t, HP)$ permuted $H$</td>
<td></td>
</tr>
<tr>
<td>SIGNING</td>
<td></td>
</tr>
<tr>
<td>Choose message $m$</td>
<td></td>
</tr>
<tr>
<td>$s = \text{Hash}(m)$</td>
<td></td>
</tr>
<tr>
<td>Find $e$: $s = eH^\top = eP(HP)^\top$, and $\text{wt}(e) \leq t$</td>
<td></td>
</tr>
<tr>
<td>$m, eP$</td>
<td></td>
</tr>
<tr>
<td>VERIFICATION</td>
<td></td>
</tr>
<tr>
<td>Check if $\text{wt}(eP) \leq t$ and $eP(HP)^\top = \text{Hash}(m)$</td>
<td></td>
</tr>
</tbody>
</table>

Problem: Distinguishability
# Hash-and-Sign: CFS

<table>
<thead>
<tr>
<th>PROVER</th>
<th>VERIFIER</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>KEY GENERATION</strong></td>
<td></td>
</tr>
<tr>
<td>( S = H ) parity-check matrix</td>
<td></td>
</tr>
<tr>
<td>( \mathcal{P} = (t, HP) ) permuted ( H )</td>
<td></td>
</tr>
<tr>
<td><strong>SIGNING</strong></td>
<td></td>
</tr>
<tr>
<td>Choose message ( m )</td>
<td></td>
</tr>
<tr>
<td>( s = \text{Hash}(m) )</td>
<td></td>
</tr>
<tr>
<td>Find ( e ): ( s = eH^\top = eP(HP)^\top ), and ( \text{wt}(e) \leq t )</td>
<td></td>
</tr>
<tr>
<td>( m, eP )</td>
<td></td>
</tr>
</tbody>
</table>

\[ \text{VERIFICATION} \]

Check if \( \text{wt}(eP) \leq t \)
and \( eP(HP)^\top = \text{Hash}(m) \)

Not any \( s \) is syndrome of low weight \( e \)
PROVER

commitments $c_0, c_1$

response $r_b$

VERIFIER

$\leftarrow b$

$b \in \{0, 1\}$

$\rightarrow r_b$

Verify $c_b$ using $r_b, P$

SIGNING

Choose message $m$

Construct signature $s$ from $S, m$

$\rightarrow m, s$

Verify signature $s$ using $P, m$

Signature Scheme
<table>
<thead>
<tr>
<th>PROVER</th>
<th>VERIFIER</th>
</tr>
</thead>
<tbody>
<tr>
<td>commits $c_0, c_1$</td>
<td>$c_0, c_1$</td>
</tr>
<tr>
<td>response $r_b$</td>
<td>$b$</td>
</tr>
<tr>
<td>$r_b$</td>
<td></td>
</tr>
<tr>
<td>Fiat-Shamir</td>
<td></td>
</tr>
</tbody>
</table>

**Signing**

- Choose message $m$
- Construct signature $s$ from $S, m$

**Verification**

- Verify signature $s$ using $\mathcal{P}, m$
Fiat-Shamir

<table>
<thead>
<tr>
<th>PROVER</th>
<th>VERIFIER</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>KEY GENERATION</strong></td>
<td></td>
</tr>
<tr>
<td>Given $\mathcal{P}, S$ of some ZKID and message $m$</td>
<td></td>
</tr>
<tr>
<td><strong>SIGNING</strong></td>
<td></td>
</tr>
<tr>
<td>Choose commitment $c$</td>
<td></td>
</tr>
<tr>
<td>$b = \text{Hash}(m, c)$</td>
<td></td>
</tr>
<tr>
<td>Compute response $r_b$</td>
<td></td>
</tr>
<tr>
<td>Signature $s = (b, r_b)$</td>
<td>$m,s$</td>
</tr>
</tbody>
</table>

**VERIFICATION**

Using $r_b$, $\mathcal{P}$ construct $c$
check if $b = \text{Hash}(m, c)$
<table>
<thead>
<tr>
<th>PROVER</th>
<th>VERIFIER</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>KEY GENERATION</strong></td>
<td></td>
</tr>
<tr>
<td>Choose $e$ with $\text{wt}(e) \leq t$</td>
<td>$H$ parity-check matrix</td>
</tr>
<tr>
<td>Compute $s = eH^\top$</td>
<td>$\mathcal{P} = (H, s, t)$</td>
</tr>
</tbody>
</table>

**VERIFICATION**

Choose $u \in \mathbb{F}_q^n$, $\sigma \in S_n$
Set $c_1 = \text{Hash}(\sigma, uH^\top)$
Set $c_2 = \text{Hash}(\sigma(u), \sigma(e))$

Choose $z \in \mathbb{F}_q^\times$
Choose $b \in \{1, 2\}$
Set $y = \sigma(u + ze)$
$r_1 = \sigma$
$r_2 = \sigma(e)$

$b = 1$: $c_1 = \text{Hash}(\sigma, \sigma^{-1}(y)H^\top - zs)$
$b = 2$: $\text{wt}(\sigma(e)) = t$
and $c_2 = \text{Hash}(y - z\sigma(e), \sigma(e))$
<table>
<thead>
<tr>
<th>PROVER</th>
<th>VERIFIER</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>KEY GENERATION</strong></td>
<td><strong>VERIFICATION</strong></td>
</tr>
<tr>
<td>Choose ( e ) with ( \text{wt}(e) \leq t ) ( H ) parity-check matrix</td>
<td>Recall SDP: [(1) \ s = eH^\top \ (2) \ \text{wt}(e) \leq t ]</td>
</tr>
<tr>
<td>Compute ( s = eH^\top )</td>
<td>( P = (H,s,t) )</td>
</tr>
</tbody>
</table>

Choose \( u \in \mathbb{F}_q^n \), \( \sigma \in S_n \)
- Set \( c_1 = \text{Hash}(\sigma, uH^\top) \)
- Set \( c_2 = \text{Hash}(\sigma(u), \sigma(e)) \)

\[
\begin{align*}
\text{Set } y &= \sigma(u + ze) \\
r_1 &= \sigma \\
r_2 &= \sigma(e)
\end{align*}
\]

Choose \( b \in \{1, 2\} \)
- \( b = 1: c_1 = \text{Hash}(\sigma, \sigma^{-1}(y)H^\top - zs) \)
- \( b = 2: \ \text{wt}(\sigma(e)) = t \)
- and \( c_2 = \text{Hash}(y - z\sigma(e), \sigma(e)) \)
### PROVER

#### KEY GENERATION

Choose $e$ with $\text{wt}(e) \leq t$

$H$ parity-check matrix

Compute $s = eH^\top$

- **Problem:** big signature sizes

### VERIFIER

- $P = (H, s, t)$

### VERIFICATION

Choose $u \in \mathbb{F}_q^n$, $\sigma \in S_n$

Set $c_1 = \text{Hash}(\sigma, uH^\top)$

Set $c_2 = \text{Hash}(\sigma(u), \sigma(e))$

Choose $z \in \mathbb{F}_q^\times$

Set $y = \sigma(u + ze)$

$r_1 = \sigma$

$r_2 = \sigma(e)$

$b = 1$: $c_1 = \text{Hash}(\sigma, \sigma^{-1}(y)H^\top - zs)$

$b = 2$: $\text{wt}(\sigma(e)) = t$

and $c_2 = \text{Hash}(y - z\sigma(e), \sigma(e))$
Cheating Probability

- Cheating probability = Probability of impersonator getting accepted
- For security level $2^\lambda$ want cheating probability $2^{-\lambda}$
- If cheating probability $\delta$, with $N$ rounds $\rightarrow$ cheating probability $\delta^N$
Cheating Probability

- Cheating probability = Probability of impersonator getting accepted
- For security level $2^{\lambda}$ want cheating probability $2^{-\lambda}$
- If cheating probability $\delta$, with $N$ rounds $\rightarrow$ cheating probability $\delta^N$
- might need many rounds: large communication cost
Cheating Probability

- Cheating probability = Probability of impersonator getting accepted
- For security level $2^\lambda$ want cheating probability $2^{-\lambda}$
- If cheating probability $\delta$, with $N$ rounds $\rightarrow$ cheating probability $\delta^N$
- might need many rounds: large communication cost
- solution: compression technique
- do not send $c_i^0, c_i^1$ in each round $i$
- before 1. round send $c = \text{Hash}(c_0^1, c_1^1, \ldots, c_0^N, c_1^N)$
- $i$th round: receiving challenge $b$ prover sends $r_i^b, c_{1-b}^i$
- end: verifier checks $c = \text{Hash}(c_0^1, c_1^1, \ldots, c_0^N, c_1^N)$

Cheating Probability

- Cheating probability = Probability of impersonator getting accepted
- For security level $2^\lambda$ want cheating probability $2^{-\lambda}$
- If cheating probability $\delta$, with $N$ rounds $\rightarrow$ cheating probability $\delta^N$
- might need many rounds: large communication cost
- other solution: MPC in the head
- third party: trusted helper sends commitments $\rightarrow \delta = 0$
- instead prover sends seeds of commitment: not ZK $\rightarrow$ cut and choose
- $x < N$ times send response, $N - x$ times send the seed of commitment
- to compress: use Merkle root or seed tree

### Comparison

<table>
<thead>
<tr>
<th></th>
<th>ZKID</th>
<th>Hash-and-Sign</th>
</tr>
</thead>
<tbody>
<tr>
<td>reduction to NP-hard</td>
<td></td>
<td></td>
</tr>
<tr>
<td>low public key size</td>
<td></td>
<td></td>
</tr>
<tr>
<td>low signature size</td>
<td></td>
<td></td>
</tr>
<tr>
<td>fast verification</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
## Comparison

<table>
<thead>
<tr>
<th>Feature</th>
<th>ZKID</th>
<th>Hash-and-Sign</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reduction to NP-hard</td>
<td>✓</td>
<td>×</td>
</tr>
<tr>
<td>Low public key size</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Low signature size</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Fast verification</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
# Comparison

<table>
<thead>
<tr>
<th></th>
<th>ZKID</th>
<th>Hash-and-Sign</th>
</tr>
</thead>
<tbody>
<tr>
<td>reduction to NP-hard</td>
<td>✓</td>
<td>×</td>
</tr>
<tr>
<td>low public key size</td>
<td>✓</td>
<td>×</td>
</tr>
<tr>
<td>low signature size</td>
<td></td>
<td></td>
</tr>
<tr>
<td>fast verification</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Violetta Weger — How to Sign using Restricted Errors
Comparison

<table>
<thead>
<tr>
<th>Feature</th>
<th>ZKID</th>
<th>Hash-and-Sign</th>
</tr>
</thead>
<tbody>
<tr>
<td>reduction to NP-hard</td>
<td>✓</td>
<td>×</td>
</tr>
<tr>
<td>low public key size</td>
<td>CVE: 70 B</td>
<td>WAVE: 3 MB</td>
</tr>
<tr>
<td>low signature size</td>
<td>NIST: 3 KB</td>
<td></td>
</tr>
<tr>
<td>fast verification</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
## Comparison

<table>
<thead>
<tr>
<th>Feature</th>
<th>ZKID</th>
<th>Hash-and-Sign</th>
</tr>
</thead>
<tbody>
<tr>
<td>reduction to NP-hard</td>
<td>✓</td>
<td>×</td>
</tr>
<tr>
<td>low public key size</td>
<td>CVE: 70 B</td>
<td>WAVE: 3 MB</td>
</tr>
<tr>
<td>low signature size</td>
<td>~</td>
<td>✓</td>
</tr>
<tr>
<td>fast verification</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
## Comparison

<table>
<thead>
<tr>
<th>Feature</th>
<th>ZKID</th>
<th>Hash-and-Sign</th>
</tr>
</thead>
<tbody>
<tr>
<td>reduction to NP-hard</td>
<td>✓</td>
<td>×</td>
</tr>
<tr>
<td>low public key size</td>
<td>CVE: 70 B</td>
<td>WAVE: 3 MB</td>
</tr>
<tr>
<td>low signature size</td>
<td>CVE: 43 KB</td>
<td>WAVE: 1 KB</td>
</tr>
<tr>
<td>fast verification</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
## Comparison

<table>
<thead>
<tr>
<th>Feature</th>
<th>ZKID</th>
<th>Hash-and-Sign</th>
</tr>
</thead>
<tbody>
<tr>
<td>reduction to NP-hard</td>
<td>✓</td>
<td>×</td>
</tr>
<tr>
<td>low public key size</td>
<td>CVE: 70 B</td>
<td>WAVE: 3 MB</td>
</tr>
<tr>
<td>low signature size</td>
<td>CVE: 43 KB</td>
<td>WAVE: 1 KB</td>
</tr>
<tr>
<td>fast verification</td>
<td>~</td>
<td>✓</td>
</tr>
</tbody>
</table>
Statistical Attacks

- Low Hamming weight generators will produce low Hamming weight signatures
- Observing many signatures reveals the support of the secret low Hamming weight generators
Statistical Attacks

- Low Hamming weight generators will produce low Hamming weight signatures
- Observing many signatures reveals the support of the secret low Hamming weight generators

- Low Lee weight generators: 
  \[ \text{supp}_L(x) = (\text{wt}_L(x_1), \ldots, \text{wt}_L(x_n)) \]
- Signatures have low Lee weight
- Recovering Lee support of secret generators: much harder