Recent Advances in Code-based Signatures

Violetta Weger

CAST Workshop:
Quantentechnologie und Quantencomputer-resistente Sicherheit

September 7, 2023
Motivation

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

Standardized:
- Signatures: Dilithium, FALCON, SPHINCS+
- PKE/KEM: KYBER

4th round:
- PKE/KEM: Classic McEliece, BIKE, HQC based on structured lattices

Hash-based

Code-based

2023 NIST additional call for signature schemes → This talk
Motivation

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

Standardized:
- Signatures: Dilithium, FALCON, SPHINCS+
- PKE/KEM: KYBER

4th round:
- PKE/KEM: Classic McEliece, BIKE, HQC
## Motivation

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

<table>
<thead>
<tr>
<th>Standardized:</th>
<th>Signatures:</th>
<th>PKE/KEM:</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Dilithium, FALCON, SPHINCS+</td>
<td>KYBER</td>
</tr>
</tbody>
</table>

4th round:

<table>
<thead>
<tr>
<th>PKE/KEM:</th>
<th>Classic McEliece, BIKE, HQC</th>
</tr>
</thead>
</table>

- **based on structured lattices**
- **Hash-based**
- **Code-based**
Motivation

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

Standardized:
Signatures: Dilithium, FALCON, SPHINCS+
PKE/KEM: KYBER

4th round:
PKE/KEM: Classic McEliece, BIKE, HQC

2023 NIST additional call for signature schemes → This talk
Outline

1. **Code-based Cryptography**
   - Introduction to Coding Theory
   - Hard Problems from Coding Theory

2. **Code-based Signature Schemes**
   - What is a Signature Scheme
   - Techniques to Construct Signatures
   - Our Scheme: CROSS

3. **Round 1 Submissions**
   - Survivors after 2 months of cryptanalysis
   - Efficiency and Performance
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
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
**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: $d_H(x, y) = |\{i \mid x_i \neq y_i\}|$
- Minimum distance of a code:

$$d(C) = \min\{d_H(x, y) \mid x \neq y \in C\}$$
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: $t = \lceil(d(C) - 1)/2\rceil$
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 based on this problem.
R. J. McEliece. "A public-key cryptosystem based on algebraic coding theory",

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.
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

Hard Problems from Coding Theory

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

scrambling

Seemingly random code

- 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

• 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**

- **Key Generation:** Public, Private
- **Signing:** Use Private and message $m$ to generate signature $\sigma$

**Verifier**

- **Verification:** Use Public and message $m$ to verify signature $\sigma$

Approaches for signatures:
- Hash-and-Sign
- ZK Protocol
- ZK + MPC
Idea of Signature Schemes

Signer

Verifier

Approaches for signatures:
- Hash-and-Sign
- ZK Protocol
- ZK + MPC
Idea of Signature Schemes

Signer

Verifier

Approaches for signatures:
• Hash-and-Sign
• ZK Protocol
• ZK + MPC

Violetta Weger — Recent Advances in Code-based Signatures
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 \( \sigma \)

Approaches for signatures:

- Hash-and-Sign
- ZK Protocol
- ZK + MPC
Idea of Signature Schemes

**Signer**

- **Key Generation:**
  \[ \mathcal{P} \text{ public, } \mathcal{S} \text{ secret} \]

- **Signing:** use \[ \mathcal{S} \] and message \[ m \] to generate signature \[ \sigma \]

**Verifier**

- **Verification:** use \[ \mathcal{P} \] and message \[ m \] to verify signature \[ \sigma \]

Approaches for signatures:

- Hash-and-Sign
- ZK Protocol
- ZK + MPC
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 \( \sigma \)

Approaches for signatures:

- Hash-and-Sign
- ZK Protocol
- ZK + MPC
Hash-and-Sign

First introduced in
Following idea of McEliece

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


Hash-and-Sign

First introduced in

Following idea of McEliece


→ start with structured code $H$

→ publish scrambled code $HP$

→ large public key sizes
Hash-and-Sign

First introduced in
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 $\sigma = eP$


→ 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

First introduced in

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 $\sigma = eP$
→ slow signing
→ reduce key sizes:
→ use quasi-cyclic codes
→ use low density generators
Hash-and-Sign

First introduced in
Following idea of McEliece


→ start with structured code $H$
→ publish scrambled code $HP$
→ large public key sizes
→ Hash($m$) = $eH^T$, wt$_H$(e) ≤ t
→ signature $\sigma = eP$
→ slow signing
→ reduce key sizes:
→ use quasi-cyclic codes
→ use low density generators
→ statistical attacks
Idea of ZK Protocol

Prover

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

Verifier

- $\mathcal{P}$, $c$ → $b$←
- $r_b$ →

$b$: challenge

Recover $c$ from $r_b$ and $\mathcal{P}$

Idea of ZK Protocol

**Prover**

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

**Interaction**

$\xrightarrow{\mathcal{P}, c}$

$\leftarrow b$

$\xrightarrow{r_b}$

**Verifier**

- $b$: challenge
- Recover $c$ from $r_b$ and $\mathcal{P}$

Idea of ZK Protocol

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

**Prover**

**Verifier**

Recover \( c \) from \( r_b \) and \( P \)

Verify \( b = \text{Hash}(m, c) \)

Idea of ZK Protocol

\[ N \rightarrow \]

**Prover**

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

**Verifier**

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

- \( \alpha \) cheating probability, \( \lambda \) bit security level
- **Rounds**: have to repeat ZK protocol \( N \) times: \( 2^{\lambda} < (1/\alpha)^N \)
- Signature size: communication within all \( N \) rounds

---

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: transformation, e.g. permutation

$r_1 = \varphi$, or transformed secret $r_2 = \varphi(e)$

Verifier

$P, c_1, c_2 \rightarrow$

$b \in \{1, 2\}$

$b \rightarrow$

$rb \rightarrow$

recover $c_b$ from $rb$ 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 $t$, $P$: random $H$, $s = eH^\top$

$c_1$: commitment to syndrome equation 1.
$c_2$: commitment to weight 2.

response: transformation, e.g. permutation $r_1 = \varphi$, or transformed secret $r_2 = \varphi(e)$

Verifier

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

CVE $\lambda = 128$ bit security $\rightarrow$ signature size: 43 kB

recover $c_b$ from $r_b$ and $P$
1. Solution: Multiparty Computation (MPC) in-the-head


**Prover**

- Split secret $S$ into $N$ shares $s_i$
- Commitments $c_i$ to $s_i$
- Compute $\varphi(s_i) = \alpha_i$
- Response: all shares but $\ell$

**Verifier**

- Challenge $\ell \in \{1, \ldots, N\}$
- Check $\alpha_i, c_i$ from $s_i$

→ New cheating probability: $1/N$
1. Solution: Multiparty Computation (MPC) in-the-head


**Problem: complex implementation**

Verification and signing is slow

- Compute $\varphi(s_i) = \alpha_i$
- Response: all shares but $\ell$
- New cheating probability: $1/N$
Code-Based ZK Protocols

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

\[ e \begin{array}{cccc} 0 & 0 & 0 & 0 \end{array} \xrightarrow{\varphi} \begin{array}{cccc} 0 & \_ & \_ & 0 \end{array} e' \]
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 $\text{wt}_H(e) \leq t$ and $s = eH^\top$.

Which $\varphi$ are allowed?
Code-Based ZK Protocols

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

Which $\varphi$ are allowed?

$\rightarrow \varphi$: linear isometries of Hamming metric:
- permutation + scalar multiplication
Code-Based ZK Protocols

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 $\text{wt}_{H}(e) \leq t$ and $s = eH^\top$.

Which $\phi$ are allowed?

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

2. Problem: permutations are costly $\rightarrow \phi : n \log_{2}(q - 1) + n \log_{2}(n)$
Code-Based ZK Protocols

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

Can we avoid permutations - but keep the hardness of the problem?
### Code-Based ZK Protocols

#### 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 \( \text{wt}_H(e) \leq t \) and \( s = eH^\top \).

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 \( s = eH^\top \).
2. Solution: Restricted Errors


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

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

2. Solution: Restricted Errors


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

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

\[ (\mathbb{E}^n, \ast) \quad \xrightarrow{\ell} \quad (\mathbb{F}_z^n, +) \]

- \[e = (1, 9, 3, 3) \in \{1, 3, 9\}^4 \]

- \[\ell(e) = (0, 2, 1, 1) \in \mathbb{F}_3^4 \]
2. Solution: Restricted Errors


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

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

\((E^n, \star) \quad \xrightarrow{\ell} \quad (F_z^n, +)\)

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

- \(\ell(e) = (0, 2, 1, 1) \in F_3^4\)
- \(\ell(\varphi) \in F_z^n\)
- \(\ell(\varphi) : \ell(e') = (1, 2, 0, 1) \in F_3^4\)
Restricted Errors

2. Solution: Restricted Errors


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

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

\[
\mathbb{E}^n = \{ (1, 9, 3, 3), (3, 9, 1, 3) \}^4
\]

\[
\text{trans.: } \varphi : \mathbb{E}^n \rightarrow \mathbb{E}^n, e \mapsto e \ast e'
\]

\[
\varphi : e' = (3, 9, 1, 3) \in \mathbb{E}^n
\]

\[
\varphi(e) = e \ast e' \in (\mathbb{E}^n, \ast)
\]

\[
\varphi(e) = (1, 9, 3, 3) \ast (3, 9, 1, 3)
\]

\[
\ell(e) = (0, 2, 1, 1) \in \mathbb{F}_3^4
\]

\[
\ell(e) + \ell(e') \in (\mathbb{F}_3^n, +)
\]

\[
(0, 2, 1, 1) + (1, 2, 0, 1)
\]
Restricted Errors

2. Solution: Restricted Errors


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

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

\[(E^n, \star) \xrightarrow{\ell} (F_z^n, +)\]

\[\rightarrow \text{Smaller sizes: } n \log_2(z) \text{ instead of } n \log_2((q - 1)n)\]

\[\rightarrow \text{Faster arithmetic: ops. in } (F_z^n, +) \text{ instead of } (F_q^n, \cdot)\]
<table>
<thead>
<tr>
<th>Basis</th>
<th>Optimizations</th>
<th>Security</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Restricted SDP</td>
<td>• Merkle trees</td>
<td>• no trapdoor needed</td>
</tr>
<tr>
<td>• ZK + Fiat-Shamir</td>
<td>• unbalanced challenges</td>
<td>• EUF-CMA security</td>
</tr>
<tr>
<td>→ compact</td>
<td>→ efficient</td>
<td>→ secure</td>
</tr>
</tbody>
</table>
## CROSS

### Basis
- Restricted SDP
- ZK + Fiat-Shamir
→ compact

### Optimizations
- Merkle trees
- unbalanced challenges
→ efficient

### Security
- no trapdoor needed
- EUF-CMA security
→ secure

Sizes in bytes, times in MCycles

<table>
<thead>
<tr>
<th>Level</th>
<th>pk</th>
<th>sign</th>
<th>$t_{sign}$</th>
<th>$t_{verify}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>I</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>fast</td>
<td>38</td>
<td>8’,665</td>
<td>3.08</td>
<td>2.11</td>
</tr>
<tr>
<td>short</td>
<td>38</td>
<td>7’,625</td>
<td>11.04</td>
<td>7.81</td>
</tr>
<tr>
<td>III</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>fast</td>
<td>56</td>
<td>21’,697</td>
<td>4.91</td>
<td>3.23</td>
</tr>
<tr>
<td>short</td>
<td>56</td>
<td>17’,429</td>
<td>18.06</td>
<td>12.24</td>
</tr>
<tr>
<td>V</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>fast</td>
<td>77</td>
<td>37’,924</td>
<td>11.05</td>
<td>7.49</td>
</tr>
<tr>
<td>short</td>
<td>77</td>
<td>31’,696</td>
<td>29.08</td>
<td>19.44</td>
</tr>
</tbody>
</table>

No optimized implementation
Codes & Restricted Objects Signature Scheme
http://cross-crypto.com/
Round 1 Submissions

Submitted: 50  \rightarrow  Complete & Proper: 40

- Multivariate: 12
- Code-based: 11
- Lattice-based: 7
- Symmetric: 4
- Other: 5
- Isogeny-based: 1

[Link to all schemes and their performances: https://pqshield.github.io/nist-sigs-zoo/]

Violetta Weger — Recent Advances in Code-based Signatures
## Round 1 Submissions

<table>
<thead>
<tr>
<th>Submitted: 50</th>
<th>→</th>
<th>Complete &amp; Proper: 40</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Cryptanalysis</th>
<th>→</th>
<th>Survivors: 29</th>
</tr>
</thead>
<tbody>
<tr>
<td>- Multivariate: 12</td>
<td>→ 9</td>
<td></td>
</tr>
<tr>
<td>- Code-based: 11</td>
<td>→ 9</td>
<td></td>
</tr>
<tr>
<td>- Lattice-based: 7</td>
<td>→ 5</td>
<td></td>
</tr>
<tr>
<td>- Symmetric: 4</td>
<td>→ 4</td>
<td></td>
</tr>
<tr>
<td>- Other: 5</td>
<td>→ 1</td>
<td></td>
</tr>
<tr>
<td>- Isogeny-based: 1</td>
<td>→ 1</td>
<td></td>
</tr>
</tbody>
</table>
## Round 1 Submissions

<table>
<thead>
<tr>
<th>Submitted: 50</th>
<th>(\rightarrow)</th>
<th>Complete &amp; Proper: 40</th>
</tr>
</thead>
</table>

## Cryptanalysis

<table>
<thead>
<tr>
<th>(\rightarrow)</th>
<th>Survivors: 29</th>
</tr>
</thead>
<tbody>
<tr>
<td>Multivariate: 12</td>
<td>9</td>
</tr>
<tr>
<td>Code-based: 11</td>
<td>9</td>
</tr>
<tr>
<td>Lattice-based: 7</td>
<td>5</td>
</tr>
<tr>
<td>Symmetric: 4</td>
<td>4</td>
</tr>
<tr>
<td>Other: 5</td>
<td>1</td>
</tr>
<tr>
<td>Isogeny-based: 1</td>
<td>1</td>
</tr>
</tbody>
</table>

→ all of the schemes and their performances:

https://pqshield.github.io/nist-sigs-zoo/
Round 1 Submissions

Submitted: 50  \rightarrow  Complete & Proper: 40

Cryptanalysis  \rightarrow  Survivors: 29

- Multivariate: 12  \rightarrow  9
- Code-based: 11  \rightarrow  9
- Lattice-based: 7  \rightarrow  5
- Symmetric: 4  \rightarrow  4
- Other: 5  \rightarrow  1
- Isogeny-based: 1  \rightarrow  1

\rightarrow  all of the schemes and their performances:
https://pqshield.github.io/nist-sigs-zoo/
# Code-Based Round 1 Submissions

## MPC in-the-head
- SDitH: SDP
- RYDE: Rank SDP
- MIRA/MiRitH: matrix rank SDP
- PERK: permuted kernel

## ZK Protocol
- LESS: code equivalence
- CROSS: restricted SDP
- MEDS: matrix rank CE

## Hash & Sign
- FuLeeca: Lee SDP
- WAVE: \((U, U + V)\)
- Enh. pqsigRM: Reed-Muller large weight SDP
## Code-Based Round 1 Submissions

### MPC in-the-head

- SDitH: SDP
- RYDE: Rank SDP
- MIRA/MiRitH: matrix rank SDP
- PERK: permuted kernel

→ slow signing and verification

### ZK Protocol

- LESS: code equivalence
- CROSS: restricted SDP
- MEDS: matrix rank CE

→ large signatures

### Hash & Sign

- FuLeeca: Lee SDP
- WAVE: $(U, U + V),$ large weight SDP
- Enh. pqsigRM: Reed-Muller

→ attacked

→ large public keys

---

*Violetta Weger — Recent Advances in Code-based Signatures*
Performance

NIST Category I, all sizes in bytes

sign vs. pk
Performance

NIST Category I, all sizes in bytes

- SPHINCS+
- Dilithium
- Falcon
- FuLeeca
- LESS
- MIRACL
- MiRiTH
- E.pqsigRM
- PERK
- RYDE
- SDitH
- SPHINCS+
- WAVE

Violetta Weger — Recent Advances in Code-based Signatures
Performance

NIST Category I, all sizes in bytes
Performance

NIST Category I, all sizes in bytes

- CROSS
- Dilithium
- E.pqsigRM
- Falcon
- FuLeeca
- LESS
- MEDS
- MIRA
- MiRitH
- PERK
- RYDE
- SDitH
- SPHINCS+
- WAVE
Performance

NIST Category I, all sizes in bytes

sign

10^3

10^4

10^2  10^3  10^4  10^6  pk

CROSS  MIRA
Dilithium  MiRitH
E.pqsigRM  PERK
Falcon  RYDE
FuLeeca  SDitH
LESS  SPHINCS+
MEDS  WAVE
Performance

NIST Category I, all sizes in bytes

- CROSS
- Dilithium
- E.pqsigRM
- Falcon
- FuLeeca
- LESS
- MEDS
- MIRA
- MiRitH
- PERK
- RYDE
- SDitH
- SPHINCS+
- WAVE
Performance

NIST Category I, all sizes in MCycles

- CROSS
- MIRA
- Dilithium
- MiRitH
- E.pqsigRM
- PERK
- Falcon
- RYDE
- FuLeea
- SDitH
- LESS
- SPHINCS+
- MEDS
- WAVE

Log-log plot showing the verification and signing times for different code-based signature schemes.
Performance

NIST Category I, all sizes in MCycles

signing

verification

10^{-1}

10^0

10^1

10^2

10^3

CROSS
Dilithium
E.pqsigRM
Falcon
FuLeeca
LESS
MIRA
MiRitH
PERK
RYDE
SDiTH
SPHINCS+
MEDS
WAVE

Dilithium
SPHINCS+
Falcon
Performance

NIST Category I, all sizes in MCycles

Verification vs. Signing

- CROSS
- MIRA
- Dilithium
- MiRtH
- E.pqsigRM
- PERK
- Falcon
- RYDE
- FuLeeca
- SDitH
- LESS
- SPHINCS
- MEDS
- WAVE

WAVE
FuLeeca
Performance

NIST Category I, all sizes in MCycles
Performance

NIST Category I, all sizes in MCycles
Performance

NIST Category I, all sizes in MCycles

- CROSS
- MIRA
- Dilithium
- MiRitH
- E.pqsigRM
- PERK
- Falcon
- RYDE
- FuLeeca
- SDitH
- LESS
- SPHINCS
- MEDS
- WAVE
Questions?

What’s next?

- Cryptanalysis continues
- Improvements?
- How many rounds?

Thank you!

Slides
# Code-Based Submissions

All sizes in bytes, times in MCycles.

| Scheme     | Based on             | Technique | | Pk | | Sig | | Sign | | Verify |
|------------|----------------------|-----------|-----|----|-----|-----|------|------|-------|
| CROSS      | Restricted SDP       | ZK        | 32  | 7’625 | 11 | 7.4 |
| Enh. pqsigRM | Reed-Muller          | Hash & Sign | 2’000’000 | 1’032 | 1.3 | 0.2 |
| FuLeeca    | Lee SDP              | Hash & Sign | 1’318 | 1’100 | 1’846 | 1.3 |
| LESS       | Code equiv.          | ZK        | 13’700 | 8’400 | 206 | 213 |
| MEDS       | Matrix rank equiv.   | ZK        | 9’923 | 9’896 | 518 | 515 |
| MIRA       | Matrix rank SDP      | MPC       | 84   | 5’640 | 46’8 | 43’9 |
| MiRitH     | Matrix rank SDP      | MPC       | 129  | 4’536 | 6’108 | 6’195 |
| PERK       | Permuted Kernel      | MPC       | 150  | 6’560 | 39  | 27  |
| RYDE       | Rank SDP             | MPC       | 86   | 5’956 | 23.4 | 20.1 |
| SDitH      | SDP                  | MPC       | 120  | 8’241 | 13.4 | 12.5 |
| WAVE       | Large wt \((U, U + V)\) | Hash & Sign | 3’677’390 | 822 | 1’160 | 1.23 |

⚠️ Not all schemes have optimized implementations → Numbers may change