Rainbow

Jintai Ding, Ming-Shing Chen, Matthias Kannwischer, Jacques Patarin, Albrecht Petzoldt, Dieter Schmidt, Bo Yin Yang

3rd NIST Post-Quantum Standardization Conference

07.06.2021
Multivariate Cryptography

Public Key: System of multivariate quadratic polynomials

\[ p^{(1)}(x_1, \ldots, x_n) = \sum_{i=1}^{n} \sum_{j=i}^{n} p_{ij}^{(1)} \cdot x_i x_j + \sum_{i=1}^{n} p_i^{(1)} \cdot x_i + p_0^{(1)} \]

\[ p^{(2)}(x_1, \ldots, x_n) = \sum_{i=1}^{n} \sum_{j=i}^{n} p_{ij}^{(2)} \cdot x_i x_j + \sum_{i=1}^{n} p_i^{(2)} \cdot x_i + p_0^{(2)} \]

\[ \vdots \]

\[ p^{(m)}(x_1, \ldots, x_n) = \sum_{i=1}^{n} \sum_{j=i}^{n} p_{ij}^{(m)} \cdot x_i x_j + \sum_{i=1}^{n} p_i^{(m)} \cdot x_i + p_0^{(m)} \]
Multivariate Cryptography

Public Key: System of multivariate quadratic polynomials

\[ p^{(1)}(x_1, \ldots, x_n) = \sum_{i=1}^{n} \sum_{j=i}^{n} p_{ij}^{(1)} x_i x_j + \sum_{i=1}^{n} p_i^{(1)} x_i + p_0^{(1)} \]

\[ p^{(2)}(x_1, \ldots, x_n) = \sum_{i=1}^{n} \sum_{j=i}^{n} p_{ij}^{(2)} x_i x_j + \sum_{i=1}^{n} p_i^{(2)} x_i + p_0^{(2)} \]

\[ \vdots \]

\[ p^{(m)}(x_1, \ldots, x_n) = \sum_{i=1}^{n} \sum_{j=i}^{n} p_{ij}^{(m)} x_i x_j + \sum_{i=1}^{n} p_i^{(m)} x_i + p_0^{(m)} \]

Security based on the

**Problem MQ:** Given \( m \) multivariate quadratic polynomials \( p^{(1)}(x), \ldots, p^{(m)}(x) \), find a vector \( \bar{x} = (\bar{x}_1, \ldots, \bar{x}_n) \) such that \( p^{(1)}(\bar{x}) = \ldots = p^{(m)}(\bar{x}) = 0 \).
Construction

- Easily invertible quadratic map $\mathcal{F} : \mathbb{F}^n \rightarrow \mathbb{F}^m$
Construction

- Easily invertible quadratic map $\mathcal{F} : \mathbb{F}^n \rightarrow \mathbb{F}^m$
- Two invertible linear maps $S : \mathbb{F}^m \rightarrow \mathbb{F}^m$ and $T : \mathbb{F}^n \rightarrow \mathbb{F}^n$
Construction

- Easily invertible quadratic map $\mathcal{F} : \mathbb{F}^n \to \mathbb{F}^m$
- Two invertible linear maps $S : \mathbb{F}^m \to \mathbb{F}^m$ and $T : \mathbb{F}^n \to \mathbb{F}^n$
- Public key: $\mathcal{P} = S \circ \mathcal{F} \circ T$ supposed to look like a random system
Construction

- Easily invertible quadratic map $\mathcal{F}: \mathbb{F}^n \to \mathbb{F}^m$
- Two invertible linear maps $S: \mathbb{F}^m \to \mathbb{F}^m$ and $T: \mathbb{F}^n \to \mathbb{F}^n$
- **Public key**: $\mathcal{P} = S \circ \mathcal{F} \circ T$ supposed to look like a random system
- **Private key**: $S$, $\mathcal{F}$, $T$ allows to invert the public key
Multivariate Signature Schemes

Signature Generation

\[ d \in \{0, 1\}^* \xrightarrow{H} w \in \mathbb{F}^n \xrightarrow{S^{-1}} x \in \mathbb{F}^n \xrightarrow{F^{-1}} y \in \mathbb{F}^m \xrightarrow{T^{-1}} z \in \mathbb{F}^m \]

Signature Verification

\[ P \]

\[ d \in \{0, 1\}^* \xrightarrow{H} w \in \mathbb{F}^n \]
Multivariate Signature Schemes

Signature Generation

\[ d \in \{0,1\}^* \xrightarrow{\mathcal{H}} w \in \mathbb{F}^n \xrightarrow{S^{-1}} x \in \mathbb{F}^n \xrightarrow{\mathcal{F}^{-1}} y \in \mathbb{F}^m \xrightarrow{\mathcal{T}^{-1}} z \in \mathbb{F}^m \]

\[ \mathcal{P} \]

Signature Verification

**Signature Generation**: Given: document \( d \in \{0,1\}^* \), private key \( S, \mathcal{F}, \mathcal{T} \), compute recursively \( w = \mathcal{H}(d) \), \( x = S^{-1}(w) \), \( y = \mathcal{F}^{-1}(x) \) and \( z = \mathcal{T}^{-1}(y) \)
Multivariate Signature Schemes

**Signature Generation**

\[ d \in \{0, 1\}^* \xrightarrow{\mathcal{H}} \mathbf{w} \in \mathbb{F}^n \xrightarrow{S^{-1}} \mathbf{x} \in \mathbb{F}^n \xrightarrow{F^{-1}} \mathbf{y} \in \mathbb{F}^m \xrightarrow{T^{-1}} \mathbf{z} \in \mathbb{F}^m \]

**Signature Verification**

**Signature Generation**: Given: document \( d \in \{0, 1\}^* \), private key \( S, F, T \) compute recursively \( \mathbf{w} = \mathcal{H}(d) \), \( \mathbf{x} = S^{-1}(\mathbf{w}) \), \( \mathbf{y} = F^{-1}(\mathbf{x}) \) and \( \mathbf{z} = T^{-1}(\mathbf{y}) \)

**Signature Verification**: Given: document \( d \in \{0, 1\}^* \), signature \( \mathbf{z} \in \mathbb{F}^m \), public key \( \mathcal{P} \) check if \( \mathcal{P}(\mathbf{z}) = \mathcal{H}(d) \)
The Rainbow Signature Scheme

- finite field $\mathbb{F}$ with $q$ elements, integers $0 < v_1 < v_2 < \cdots < v_u < v_{u+1} = n$
The Rainbow Signature Scheme

- finite field $\mathbb{F}$ with $q$ elements, integers $0 < v_1 < v_2 < \cdots < v_u < v_{u+1} = n$
- set $V_i = \{1, \ldots, v_i\}$ and $O_i = \{v_i + 1, \ldots, v_{i+1}\}$ ($i = 1, \ldots, u$) $\Rightarrow |V_i| = v_i$, $|O_i| = v_{i+1} - v_i := o_i$
finite field $\mathbb{F}$ with $q$ elements, integers $0 < v_1 < v_2 < \cdots < v_u < v_{u+1} = n$

set $V_i = \{1, \ldots, v_i\}$ and $O_i = \{v_i + 1, \ldots, v_{i+1}\}$ ($i = 1, \ldots, u$) $\Rightarrow |V_i| = v_i$, $|O_i| = v_{i+1} - v_i := o_i$

central map $\mathcal{F}$ consists of $m := n - v_1$ polynomials $f^{(v_1+1)}, \ldots, f^{(n)}$ of the form

$$f^{(k)}(x_1, \ldots, x_n) = \sum_{i,j \in V_\ell} \alpha_{ij}^{(k)} x_i x_j + \sum_{i \in V_\ell, j \in O_\ell} \beta_{ij}^{(k)} x_i x_j + \sum_{i \in V_\ell \cup O_\ell} \gamma_i^{(k)} x_i + \delta^{(k)},$$

where $\ell$ is the only integer such that $k \in O_\ell$. 
The Rainbow Signature Scheme

- finite field $\mathbb{F}$ with $q$ elements, integers $0 < v_1 < v_2 < \cdots < v_u < v_{u+1} = n$
- set $V_i = \{1, \ldots, v_i\}$ and $O_i = \{v_i + 1, \ldots, v_{i+1}\}$ ($i = 1, \ldots, u$) $\Rightarrow |V_i| = v_i$, $|O_i| = v_{i+1} - v_i := o_i$
- central map $\mathcal{F}$ consists of $m := n - v_1$ polynomials $f^{(v_1+1)}, \ldots, f^{(n)}$ of the form

$$f^{(k)}(x_1, \ldots, x_n) = \sum_{i,j \in V_{\ell}} \alpha^{(k)}_{ij} x_i x_j + \sum_{i \in V_{\ell}, j \in O_{\ell}} \beta^{(k)}_{ij} x_i x_j + \sum_{i \in V_{\ell} \cup O_{\ell}} \gamma^{(k)}_i x_i + \delta^{(k)},$$

where $\ell$ is the only integer such that $k \in O_{\ell}$.
- two invertible linear maps $S : \mathbb{F}^m \rightarrow \mathbb{F}^m$ and $T : \mathbb{F}^n \rightarrow \mathbb{F}^n$
The Rainbow Signature Scheme

- finite field $\mathbb{F}$ with $q$ elements, integers $0 < v_1 < v_2 < \cdots < v_u < v_{u+1} = n$
- set $V_i = \{1, \ldots, v_i\}$ and $O_i = \{v_i + 1, \ldots, v_{i+1}\}$ ($i = 1, \ldots, u$) $\Rightarrow |V_i| = v_i$, $|O_i| = v_{i+1} - v_i := o_i$
- central map $\mathcal{F}$ consists of $m := n - v_1$ polynomials $f^{(v_1+1)}, \ldots, f^{(n)}$ of the form
  
  $$f^{(k)}(x_1, \ldots, x_n) = \sum_{i,j \in V_\ell} \alpha_{ij}^{(k)} x_i x_j + \sum_{i \in V_\ell, j \in O_\ell} \beta_{ij}^{(k)} x_i x_j + \sum_{i \in V_\ell \cup O_\ell} \gamma_i^{(k)} x_i + \delta^{(k)},$$

  where $\ell$ is the only integer such that $k \in O_\ell$.
- two invertible linear maps $S : \mathbb{F}^m \to \mathbb{F}^m$ and $T : \mathbb{F}^n \to \mathbb{F}^n$
- **Public Key**: $\mathcal{P} = S \circ \mathcal{F} \circ T : \mathbb{F}^n \to \mathbb{F}^m$
The Rainbow Signature Scheme

- finite field $\mathbb{F}$ with $q$ elements, integers $0 < v_1 < v_2 < \cdots < v_u < v_{u+1} = n$
- set $V_i = \{1, \ldots, v_i\}$ and $O_i = \{v_i + 1, \ldots, v_{i+1}\}$ ($i = 1, \ldots, u$) $\Rightarrow |V_i| = v_i$, $|O_i| = v_{i+1} - v_i := o_i$
- central map $\mathcal{F}$ consists of $m := n - v_1$ polynomials $f^{(v_1+1)}, \ldots, f^{(n)}$ of the form

$$f^{(k)}(x_1, \ldots, x_n) = \sum_{i,j \in V_\ell} \alpha^{(k)}_{ij} x_i x_j + \sum_{i \in V_\ell, j \in O_\ell} \beta^{(k)}_{ij} x_i x_j + \sum_{i \in V_\ell \cup O_\ell} \gamma^{(k)}_{i} x_i + \delta^{(k)}$$

where $\ell$ is the only integer such that $k \in O_\ell$.
- two invertible linear maps $S : \mathbb{F}^m \rightarrow \mathbb{F}^m$ and $T : \mathbb{F}^n \rightarrow \mathbb{F}^n$
- **Public Key**: $\mathcal{P} = S \circ \mathcal{F} \circ T : \mathbb{F}^n \rightarrow \mathbb{F}^m$
- **Private Key**: $S$, $\mathcal{F}$, $T$
Signature Generation

Given a document \( d \in \{0,1\}^* \) to be signed, perform the following steps:

1. Use a hash function \( H : \{0,1\}^* \rightarrow F_m \) to compute \( w = H(d) \).
2. Compute \( x = S^{-1}(w) \in F_m \).
3. Choose the Vinegar variables \( y_1, \ldots, y_v \) at random and substitute them into the polynomials \( f(v_1+1), \ldots, f(n) \).
4. For \( i = 1, \ldots, u \) do:
   a. Solve the linear system provided by \( f(v_i+1), \ldots, f(v_i+1) \) to get the values of \( y_{v_i+1}, \ldots, y_{v_i+1} \) and substitute them into the polynomials \( f(v_i+1+1), \ldots, f(n) \).
   b. (if \( i < u \)) Substitute the values of \( y_{v_i+1}, \ldots, y_{v_i+1} \) into the polynomials \( f(v_i+1+1), \ldots, f(n) \).
5. Set \( y = (y_1, \ldots, y_n) \in F_n \).
6. Compute the signature \( z \in F_n \) by \( z = T^{-1}(y) \).
Signature Generation

Given a document $d \in \{0, 1\}^*$ to be signed, perform the following steps:

1. Use a hash function $\mathcal{H} : \{0, 1\}^* \rightarrow \mathbb{F}^m$ to compute $w = \mathcal{H}(d)$. 

2. Compute $x = S^{-1}(w) \in \mathbb{F}^m$.

3. Choose the Vinegar variables $y_1, \ldots, y_v$ at random and substitute them into the polynomials $f(v_1+1), \ldots, f(n)$.

4. For $i = 1, \ldots, u$ do:
   - Solve the linear system provided by $f(v_i+1), \ldots, f(v_i+1)$ to get the values of $y_{v_i+1}, \ldots, y_{v_i+1}$.
   - Substitute the values of $y_{v_i+1}, \ldots, y_{v_i+1}$ into the polynomials $f(v_i+1+1), \ldots, f(n)$.
   - (if $i < u$) Substitute the values of $y_{v_i+1}, \ldots, y_{v_i+1}$ into the polynomials $f(v_i+1+1), \ldots, f(n)$.

5. Set $y = (y_1, \ldots, y_n) \in \mathbb{F}^n$.

6. Compute the signature $z \in \mathbb{F}^n$ by $z = T^{-1}(y)$.
Signature Generation

Given a document $d \in \{0, 1\}^*$ to be signed, perform the following steps:

1. Use a hash function $H : \{0, 1\}^* \rightarrow \mathbb{F}^m$ to compute $w = H(d)$.
2. Compute $x = S^{-1}(w) \in \mathbb{F}^m$. 
Signature Generation

Given a document $d \in \{0,1\}^*$ to be signed, perform the following steps:

1. Use a hash function $H : \{0,1\}^* \rightarrow \mathbb{F}^m$ to compute $w = H(d)$.
2. Compute $x = S^{-1}(w) \in \mathbb{F}^m$.
3. Choose the Vinegar variables $y_1, \ldots, y_{v_1}$ at random and substitute them into the polynomials $f^{(v_1+1)}, \ldots, f^{(n)}$. 
Signature Generation

Given a document \( d \in \{0, 1\}^* \) to be signed, perform the following steps

1. Use a hash function \( \mathcal{H} : \{0, 1\}^* \rightarrow \mathbb{F}_m \) to compute \( w = \mathcal{H}(d) \).
2. Compute \( x = S^{-1}(w) \in \mathbb{F}_m \).
3. Choose the Vinegar variables \( y_1, \ldots, y_{v_1} \) at random and substitute them into the polynomials \( f(v_1+1), \ldots, f(n) \).
4. For \( i = 1, \ldots, u \) do
   - Solve the linear system provided by \( f(v_i+1), \ldots, f(v_i+1) \) to get the values of \( y_{v_i+1}, \ldots, y_{v_i+1} \) and substitute them into the polynomials \( f(v_i+1+1), \ldots, f(n) \).
Signature Generation

Given a document $d \in \{0,1\}^*$ to be signed, perform the following steps

1. Use a hash function $\mathcal{H} : \{0,1\}^* \rightarrow \mathbb{F}^m$ to compute $w = \mathcal{H}(d)$.

2. Compute $x = S^{-1}(w) \in \mathbb{F}^m$.

3. Choose the Vinegar variables $y_1, \ldots, y_{v_1}$ at random and substitute them into the polynomials $f(v_1+1), \ldots, f(n)$.

4. For $i = 1, \ldots, u$ do
   
   ▶ Solve the linear system provided by $f(v_i+1), \ldots, f(v_i+1)$ to get the values of $y_{v_i+1}, \ldots, y_{v_i+1}$. and substitute them into the polynomials $f(v_{i+1}+1), \ldots, f(n)$.
   
   ▶ (if $i < u$) Substitute the values of $y_{v_i+1}, \ldots, y_{v_i+1}$ into the polynomials $f(v_{i+1}+1), \ldots, f(n)$.
Signature Generation

Given a document $d \in \{0, 1\}^*$ to be signed, perform the following steps

1. Use a hash function $\mathcal{H} : \{0, 1\}^* \to \mathbb{F}^m$ to compute $w = \mathcal{H}(d)$.
2. Compute $x = S^{-1}(w) \in \mathbb{F}^m$.
3. Choose the Vinegar variables $y_1, \ldots, y_{v_1}$ at random and substitute them into the polynomials $f^{(v_1+1)}, \ldots, f^{(n)}$.
4. For $i = 1, \ldots, u$ do
   - Solve the linear system provided by $f^{(v_i+1)}, \ldots, f^{(v_i+1)}$ to get the values of $y_{v_i+1}, \ldots, y_{v_i+1}$ and substitute them into the polynomials $f^{(v_i+1+1)}, \ldots, f^{(n)}$.
   - (if $i < u$) Substitute the values of $y_{v_i+1}, \ldots, y_{v_i+1}$ into the polynomials $f^{(v_i+1+1)}, \ldots, f^{(n)}$.
5. Set $y = (y_1, \ldots, y_n) \in \mathbb{F}^n$. 
Signature Generation

Given a document \( d \in \{0, 1\}^* \) to be signed, perform the following steps

1. Use a hash function \( \mathcal{H} : \{0, 1\}^* \rightarrow \mathbb{F}^m \) to compute \( w = \mathcal{H}(d) \).
2. Compute \( x = S^{-1}(w) \in \mathbb{F}^m \).
3. Choose the Vinegar variables \( y_1, \ldots, y_{v_1} \) at random and substitute them into the polynomials \( f^{(v_1+1)}, \ldots, f^{(n)} \).
4. For \( i = 1, \ldots, u \) do
   - Solve the linear system provided by \( f^{(v_i+1)}, \ldots, f^{(v_i+1)} \) to get the values of \( y_{v_i+1}, \ldots, y_{v_{i+1}} \) and substitute them into the polynomials \( f^{(v_{i+1}+1)}, \ldots, f^{(n)} \).
   - (if \( i < u \)) Substitute the values of \( y_{v_{i+1}}, \ldots, y_{v_{i+1}} \) into the polynomials \( f^{(v_{i+1}+1)}, \ldots, f^{(n)} \).
5. Set \( y = (y_1, \ldots, y_n) \in \mathbb{F}^n \).
6. Compute the signature \( z \in \mathbb{F}^n \) by \( z = \mathcal{T}^{-1}(y) \).
Signature Verification

Given a document $d \in \{0, 1\}^*$ and a signature $z \in \mathbb{F}^n$, compute

$$w' = P(z) \in \mathbb{F}^m \text{ and } w = H(d) \in \mathbb{F}^m.$$ 

If $w' = w$ holds, the signature is accepted; otherwise it is rejected.
Signature Verification

Given a document $d \in \{0, 1\}^*$ and a signature $z \in \mathbb{F}^n$, compute

- $w' = P(z) \in \mathbb{F}^m$ and
Signature Verification

Given a document $d \in \{0, 1\}^*$ and a signature $z \in \mathbb{F}^n$, compute

- $w' = \mathcal{P}(z) \in \mathbb{F}^m$ and
- $w = \mathcal{H}(d) \in \mathbb{F}^m$.

If $w' = w$ holds, the signature is accepted; otherwise it is rejected.
Signature Verification

Given a document $d \in \{0, 1\}^*$ and a signature $z \in \mathbb{F}^n$, compute

- $w' = P(z) \in \mathbb{F}^m$ and
- $w = H(d) \in \mathbb{F}^m$.

If $w' = w$ holds, the signature is accepted; otherwise it is rejected.
Conventions

- We restrict to Rainbow schemes with two layers

Our conventions don't weaken the security of the scheme and enable us to speed up the key generation process drastically.
Conventions

- We restrict to Rainbow schemes with two layers
- We restrict to homogeneous maps $S, F$ and $T$

Our conventions don't weaken the security of the scheme and enable us to speed up the key generation process drastically.
Conventions

- We restrict to Rainbow schemes with two layers
- We restrict to homogeneous maps $S, F$ and $T$
- The linear maps $S$ and $T$ are represented by matrices $S, T$ of the form

$$S = \begin{pmatrix} I_{o_1} & S'_{o_1 \times o_2} \\ 0_{o_2 \times o_2} & I_{o_2} \end{pmatrix} \quad T = \begin{pmatrix} I_{v_1} & T^{(1)}_{v_1 \times o_1} & T^{(2)}_{v_1 \times o_2} \\ 0_{o_1 \times v_1} & 1_{v_1} & T^{(3)}_{o_1 \times o_2} \\ 0_{o_2 \times v_1} & 0_{o_2 \times o_1} & I_{o_2} \end{pmatrix}.$$
Conventions

- We restrict to Rainbow schemes with two layers
- We restrict to homogeneous maps $S$, $F$ and $T$
- The linear maps $S$ and $T$ are represented by matrices $S$, $T$ of the form

$$S = \begin{pmatrix} I_{o_1} & S'_{o_1 \times o_2} \\ 0_{o_2 \times o_2} & I_{o_2} \end{pmatrix} \quad T = \begin{pmatrix} I_{v_1} & T^{(1)}_{v_1 \times o_1} & T^{(2)}_{v_1 \times o_2} \\ 0_{o_1 \times v_1} & 1_{v_1} & T^{(3)}_{o_1 \times o_2} \\ 0_{o_2 \times v_1} & 0_{o_2 \times o_1} & I_{o_2} \end{pmatrix}.$$ 

- Our conventions don’t weaken the security of the scheme and enable us to speed up the key generation process drastically
Generate the blue parts of the public key as well as the linear maps $S$ and $T$ using a PRNG.

Compute the remaining parts of the public key as well as the central map $F$.

Drastical Reduction of the public key size (up to 75%)

We propose also a compressed version in which the central map is generated from a seed, too.
Generate the blue parts of the public key as well as the linear maps \( S \) and \( T \) using a PRNG.
Generate the blue parts of the public key as well as the linear maps $S$ and $T$ using a PRNG.

Compute the remaining parts of the public key as well as the central map $F$.

$\Rightarrow$ Drastical Reduction of the public key size (up to 75%).
Generate the blue parts of the public key as well as the linear maps $S$ and $T$ using a PRNG.

Compute the remaining parts of the public key as well as the central map $F$.

⇒ Drastical Reduction of the public key size (up to 75%).

We propose also a compressed version in which the central map is generated from a seed, too.
# Key and Signature Sizes

<table>
<thead>
<tr>
<th>NIST security category</th>
<th>parameters ((q, v_1, o_2, o_2))</th>
<th>signature size (bit)</th>
</tr>
</thead>
<tbody>
<tr>
<td>I</td>
<td>((16,36,32,32))</td>
<td>528</td>
</tr>
<tr>
<td>III</td>
<td>((256,68,32,48))</td>
<td>1,312</td>
</tr>
<tr>
<td>V</td>
<td>((256,96,36,64))</td>
<td>1,696</td>
</tr>
</tbody>
</table>
### Key and Signature Sizes

<table>
<thead>
<tr>
<th>NIST security category</th>
<th>parameters ((q, v_1, o_2, o_2))</th>
<th>signature size (bit)</th>
</tr>
</thead>
<tbody>
<tr>
<td>I</td>
<td>((16,36,32,32))</td>
<td>528</td>
</tr>
<tr>
<td>III</td>
<td>((256,68,32,48))</td>
<td>1,312</td>
</tr>
<tr>
<td>V</td>
<td>((256,96,36,64))</td>
<td>1,696</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>NIST security category</th>
<th>standard Rainbow</th>
<th>CZ-Rainbow</th>
<th>compressed Rainbow</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>(</td>
<td>pk</td>
<td>)</td>
</tr>
<tr>
<td>I</td>
<td>157.8</td>
<td>101.2</td>
<td>58.8</td>
</tr>
<tr>
<td>III</td>
<td>861.4</td>
<td>611.3</td>
<td>258.4</td>
</tr>
<tr>
<td>V</td>
<td>1,885.4</td>
<td>1,375.7</td>
<td>523.6</td>
</tr>
</tbody>
</table>
## Performance

<table>
<thead>
<tr>
<th>NIST security category</th>
<th>standard Rainbow</th>
<th></th>
<th>CZ-Rainbow</th>
<th></th>
<th>compressed Rainbow</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>keygen</td>
<td>sign</td>
<td>verify</td>
<td>keygen</td>
<td>sign</td>
<td>verify</td>
</tr>
<tr>
<td>I</td>
<td>9.9M</td>
<td>67k</td>
<td>34k</td>
<td>10.7M</td>
<td>67k</td>
<td>3.5M</td>
</tr>
<tr>
<td>III</td>
<td>52M</td>
<td>285k</td>
<td>132k</td>
<td>64M</td>
<td>285k</td>
<td>20M</td>
</tr>
<tr>
<td>V</td>
<td>192M</td>
<td>739k</td>
<td>392k</td>
<td>235M</td>
<td>739k</td>
<td>47M</td>
</tr>
</tbody>
</table>

Linux / Skylake (using AVX2 instructions)

See also the talk by M. Kannwischer on Implementing Rainbow on Cortex-M4
Security

- new attacks by Beullens (Intersection and New MinRank)
new attacks by Beullens (Intersection and New MinRank)
similar to other candidates (NTRUprime, Crystals) we distinguish between “free” and “real” cost of the attacks
new attacks by Beullens (Intersection and New MinRank)
similar to other candidates (NTRUprime, Crystals) we distinguish between “free” and “real” cost of the attacks
the “real” cost considers the cost of memory access
new attacks by Beullens (Intersection and New MinRank)
similar to other candidates (NTRUprime, Crystals) we distinguish between “free” and “real” cost of the attacks
the “real” cost considers the cost of memory access
cost of one multiplication
\[ \approx (\text{bit length memory transitions}) \times \sqrt{\#\text{bits non sequentially accessed}}/2^5 \]
Security

- new attacks by Beullens (Intersection and New MinRank)
- similar to other candidates (NTRUprime, Crystals) we distinguish between “free” and “real” cost of the attacks
- the “real” cost considers the cost of memory access
- cost of one multiplication
  \[ \approx (\text{bit – length memory transitions}) \times \sqrt{\# \text{bits non sequentially accessed}}/2^5 \]
- For Wiedemann over \((\mathbb{F}_{2^k})^V\) the cost of one multiplication is \(\lg V \sqrt{kV}/2^5\) “gates”
Our parameter proposals meet the NIST requirements.

<table>
<thead>
<tr>
<th>security category</th>
<th>parameters ((q, v_1, o_1, o_2))</th>
<th>Intersection</th>
<th>New MinRank</th>
<th>target cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>I</td>
<td>((16,36,32,32))</td>
<td>(2^{134.3})</td>
<td>(2^{162.1})</td>
<td>(2^{143})</td>
</tr>
<tr>
<td>III</td>
<td>((256,68,32,48))</td>
<td>(2^{105.4})</td>
<td>(2^{248.3})</td>
<td>(2^{207})</td>
</tr>
<tr>
<td>V</td>
<td>((256,96,36,64))</td>
<td>(2^{254.5})</td>
<td>(2^{309.9})</td>
<td>(2^{272})</td>
</tr>
</tbody>
</table>
Our parameter proposals meet the NIST requirements
Speeding up the Verification Process

Verification Process for Multivariate Signature Schemes

Given: message \( d \in \{0, 1\}^\star \), signature \( z = (z_1, \ldots, z_n) \in F_n \), public key \( (p(1), \ldots, p(m)) \)

1. Compute the hash value \( w = (w_1, \ldots, w_m) = H(m) \)

2. Check, for \( i = 1, \ldots, m \), if \( p(i)(z) = w_i \) holds.

Accept the signature, if and only if all the tests are fulfilled.

First Observation: We can stop, as soon as we find an \( i \) with \( p(i)(z) \neq w_i \).
Speeding up the Verification Process

Verification Process for Multivariate Signature Schemes

Given: message $d \in \{0, 1\}^\star$, signature $z = (z_1, \ldots, z_n) \in F^n$, public key $(p(1), \ldots, p(m))$

1. Compute the hash value $w = (w_1, \ldots, w_m) = H(m)$
2. Check, for $i = 1, \ldots, m$, if $p(i)(z) = w_i$ holds.

Accept the signature, if and only if all the tests are fulfilled.

First Observation: We can stop, as soon as we find an $i$ with $p(i)(z) \neq w_i$. 
Verification Process for Multivariate Signature Schemes

Given: message \( d \in \{0,1\}^* \), signature \( z = (z_1, \ldots, z_n) \in \mathbb{F}^n \), public key \( (p^{(1)}, \ldots, p^{(m)}) \)
Speeding up the Verification Process

Verification Process for Multivariate Signature Schemes

Given: message $d \in \{0, 1\}^*$, signature $z = (z_1, \ldots, z_n) \in \mathbb{F}^n$, public key $(p^{(1)}, \ldots, p^{(m)})$

1. Compute the hash value $w = (w_1, \ldots, w_m) = H(m)$
Verification Process for Multivariate Signature Schemes

Given: message $d \in \{0, 1\}^*$, signature $z = (z_1, \ldots, z_n) \in \mathbb{F}^n$, public key $(p^{(1)}, \ldots, p^{(m)})$

1. Compute the hash value $w = (w_1, \ldots, w_m) = \mathcal{H}(m)$
2. Check, for $i = 1, \ldots, m$, if
   \[ p^{(i)}(z) = w_i \]
   holds.

Accept the signature, if and only if all the tests are fulfilled.

First Observation: We can stop, as soon as we find an $i$ with $p^{(i)}(z) \neq w_i$. 
Verification Process for Multivariate Signature Schemes

Given: message $d \in \{0, 1\}^*$, signature $z = (z_1, \ldots, z_n) \in \mathbb{F}^n$, public key $(p^{(1)}, \ldots, p^{(m)})$

1. Compute the hash value $w = (w_1, \ldots, w_m) = \mathcal{H}(m)$

2. Check, for $i = 1, \ldots, m$, if

$$p^{(i)}(z) = w_i$$

holds.

Accept the signature, if and only if all the tests are fulfilled.
Verification Process for Multivariate Signature Schemes

Given: message \( d \in \{0, 1\}^* \), signature \( \mathbf{z} = (z_1, \ldots, z_n) \in \mathbb{F}^n \), public key \( (p^{(1)}, \ldots, p^{(m)}) \)

1. Compute the hash value \( \mathbf{w} = (w_1, \ldots, w_m) = H(m) \)
2. Check, for \( i = 1, \ldots, m \), if
   \[
   p^{(i)}(\mathbf{z}) = w_i
   \]
   holds.

Accept the signature, if and only if all the tests are fulfilled.

First Observation: We can stop, as soon as we find an \( i \) with \( p^{(i)}(\mathbf{z}) \neq w_i \).
Checking a Fixed Subset of Equations

- we don’t have to check all equations \( p^{(i)}(z) = w_i \ (i = 1, \ldots, m) \)
Checking a Fixed Subset of Equations

- we don’t have to check all equations $p^{(i)}(z) = w_i$ ($i = 1, \ldots, m$)
- we only have to check $k$ equations, where $k$ is the smallest number such that

$$\text{compl}_{\text{direct}}(q, k, n) \geq 2^\lambda$$
Checking a Fixed Subset of Equations

- we don’t have to check all equations \( p^{(i)}(z) = w_i \ (i = 1, \ldots, m) \)
- we only have to check \( k \) equations, where \( k \) is the smallest number such that

\[
\text{compl}_{\text{direct}}(q, k, n) \geq 2^\lambda
\]

<table>
<thead>
<tr>
<th>Security category</th>
<th>parameters ((q, v_1, o_1, o_2))</th>
<th># equations (m)</th>
<th># equations to be checked</th>
<th>compl. of sig. forgery</th>
</tr>
</thead>
<tbody>
<tr>
<td>I</td>
<td>(16, 36, 32, 32)</td>
<td>64</td>
<td>55</td>
<td>145</td>
</tr>
<tr>
<td>III</td>
<td>(256, 68, 32, 48)</td>
<td>80</td>
<td>72</td>
<td>212</td>
</tr>
<tr>
<td>V</td>
<td>(256, 96, 36, 64)</td>
<td>100</td>
<td>98</td>
<td>280</td>
</tr>
</tbody>
</table>

Albrecht Petzoldt
Rainbow
07.06.2021 15 / 19
Checking a Randomly Chosen Subset of Equations

- the signer / adversary does not know which $k$ of the $m$ equations will be checked
Checking a Randomly Chosen Subset of Equations

- the signer / adversary does not know which $k$ of the $m$ equations will be checked
- the number $k$ of equations we have to check is the smallest number such that
  \[
  \min_{k \leq \ell \leq m} \left( \binom{m}{k} \cdot \text{comp}_{\text{direct}}(q, \ell, n) \right) \geq 2^\lambda
  \]
the signer / adversary does not know which $k$ of the $m$ equations will be checked

the number $k$ of equations we have to check is the smallest number such that

$$\min_{k \leq \ell \leq m} \left( \binom{m}{k} \cdot \text{comp}_{\text{direct}}(q, \ell, n) \right) \geq 2^\lambda$$

selecting $\ell$ of the $m$ public equations is costly, but
Checking a Randomly Chosen Subset of Equations

- the signer / adversary does not know which $k$ of the $m$ equations will be checked
- the number $k$ of equations we have to check is the smallest number such that
  \[
  \min_{k \leq \ell \leq m} \left( \frac{m}{k} \cdot \binom{\ell}{k} \cdot \text{comp}_{\text{direct}}(q, \ell, n) \right) \geq 2^\lambda
  \]
- selecting $\ell$ of the $m$ public equations is costly, but
- if the public key is known beforehand (e.g. SecureBoot), this step can be done offline
Checking a Randomly Chosen Subset of Equations

- the signer / adversary does not know which $k$ of the $m$ equations will be checked
- the number $k$ of equations we have to check is the smallest number such that
  \[ \min_{k \leq \ell \leq m} \left( \binom{m}{k} \cdot \text{comp}_{\text{direct}}(q, \ell, n) \right) \geq 2^{\lambda} \]
- selecting $\ell$ of the $m$ public equations is costly, but
- if the public key is known beforehand (e.g. SecureBoot), this step can be done offline

<table>
<thead>
<tr>
<th>Security category</th>
<th>parameters $(q, v_1, o_1, o_2)$</th>
<th># equations $m$</th>
<th># equations to be checked</th>
<th>compl. of sig. forgery</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>classical</td>
</tr>
<tr>
<td>I 16,36,32,32</td>
<td>64 64</td>
<td>32 32</td>
<td>143 125</td>
<td></td>
</tr>
<tr>
<td>II 256,68,32,48</td>
<td>80 80</td>
<td>48 48</td>
<td>212 192</td>
<td></td>
</tr>
<tr>
<td>III 256,96,36,64</td>
<td>100 100</td>
<td>72 72</td>
<td>276 245</td>
<td></td>
</tr>
</tbody>
</table>
we check $k$ equations of the form $\tilde{p}^{(i)} = \sum_{j=1}^{m} \left( \alpha^{(i)}_{j} p^{(j)} \right) (z) = \sum_{j=1}^{m} \alpha^{(i)}_{j} w_{j} = \tilde{w}_{i}$
we check $k$ equations of the form $\tilde{p}^{(i)} = \sum_{j=1}^{m} (\alpha_j^{(i)} p^{(j)}) (z) = \sum_{j=1}^{m} \alpha_j^{(i)} w_j = \tilde{w}_i$

we have $\alpha_j^{(i)} \in \{0, 1\}$ and $\sum_{i=1}^{m} \alpha_j^{(i)} = 1$
Checking Randomly Chosen Linear Combinations

- we check $k$ equations of the form $\tilde{p}^{(i)} = \sum_{j=1}^{m} (\alpha_{j}^{(i)} p^{(j)}) (z) = \sum_{j=1}^{m} \alpha_{j}^{(i)} w_{j} = \tilde{w}_{i}$
- we have $\alpha_{j}^{(i)} \in \{0, 1\}$ and $\sum_{i=1}^{m} \alpha_{j}^{(i)} = 1$
- the adversary can either solve the whole public system or try to guess $z$ in such a way that the $k$ equations are fulfilled
we check $k$ equations of the form $\tilde{p}^{(i)} = \sum_{j=1}^{m} \left( \alpha_{j}^{(i)} p^{(j)} \right) (z) = \sum_{j=1}^{m} \alpha_{j}^{(i)} w_{j} = \tilde{w}_{i}$

we have $\alpha_{j}^{(i)} \in \{0, 1\}$ and $\sum_{i=1}^{m} \alpha_{j}^{(i)} = 1$

the adversary can either solve the whole public system or try to guess $z$ in such a way that the $k$ equations are fulfilled

if the public key is known beforehand, we can generate the system $\tilde{P}_i$ offline
we check \( k \) equations of the form
\[
\tilde{p}^{(i)} = \sum_{j=1}^{m} \left( \alpha_j^{(i)} p^{(j)} \right) (z) = \sum_{j=1}^{m} \alpha_j^{(i)} w_j = \tilde{w}_i
\]
we have \( \alpha_j^{(i)} \in \{0, 1\} \) and \( \sum_{i=1}^{m} \alpha_j^{(i)} = 1 \)
the adversary can either solve the whole public system or try to guess \( z \) in such a way that the \( k \) equations are fulfilled
if the public key is known beforehand, we can generate the system \( \tilde{P}_i \) offline

<table>
<thead>
<tr>
<th>Security category</th>
<th>parameters ((q, v_1, o_1, o_2))</th>
<th># equations (m)</th>
<th># variables (n)</th>
<th># equations to be checked</th>
</tr>
</thead>
<tbody>
<tr>
<td>I</td>
<td>(16,36,32,32)</td>
<td>64</td>
<td>100</td>
<td>32</td>
</tr>
<tr>
<td>III</td>
<td>(256,68,32,48)</td>
<td>80</td>
<td>148</td>
<td>24</td>
</tr>
<tr>
<td>V</td>
<td>(256,96,36,64)</td>
<td>100</td>
<td>196</td>
<td>32</td>
</tr>
</tbody>
</table>
Rainbow: Advantages and Disadvantages

Advantages

- short signatures
- fast key generation
- very fast signature generation
- very fast signature verification
Rainbow: Advantages and Disadvantages

Advantages

- short signatures
- fast key generation
- very fast signature generation
- very fast signature verification

Disadvantages

- large key sizes
- no security proof
Thank you for your attention

The Team:
Jintai Ding, Ming-Shing Chen, Matthias Kannwischer, Jacques Patarin, Albrecht Petzoldt, Dieter Schmidt, BoYin Yang

www.pqcrainbow.org