Multivariate Signature Schemes and Cryptanalysis of Early Proposals

Pierre Pébereau
Sorbonne Université, LIP6, CNRS, Thales SIX

July 10, 2023

Signature Schemes

Public Key Signature Schemes [Diffie, Hellman, 1976]

Alice wants to convince Bob that she wrote the message he received, without trading **secrets** beforehand.

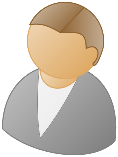
Signature Schemes

Public Key Signature Schemes [Diffie, Hellman, 1976]

Alice wants to convince Bob that she wrote the message he received, without trading **secrets** beforehand.

$\mathcal{S}$: Secret Key

Alice



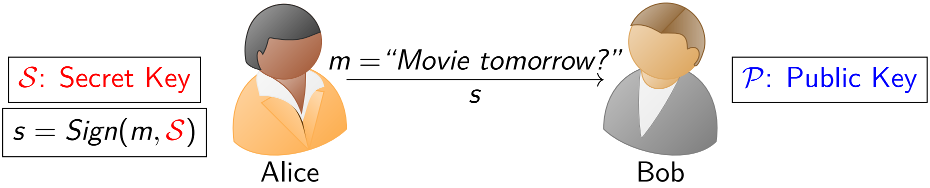
Bob

$\mathcal{P}$: Public Key

Signature Schemes

Public Key Signature Schemes [Diffie, Hellman, 1976]

Alice wants to convince Bob that she wrote the message he received, without trading **secrets** beforehand.



Signature Schemes

Public Key Signature Schemes [Diffie, Hellman, 1976]

Alice wants to convince Bob that she wrote the message he received, without trading **secrets** beforehand.

$\mathcal{S}$: Secret Key

$s = Sign(m, \mathcal{S})$

Alice

$$\xrightarrow[s]{m = \text{“Movie tomorrow?”}}$$

Bob

$\mathcal{P}$: Public Key

$Verify(m, s, \mathcal{P})$

Signature Schemes

Traditional Solutions

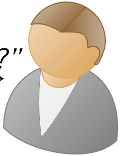
- Discrete logarithm (DSA, ElGamal, ECDSA, ...)
- Factoring (RSA)

$\boxed{\mathcal{S}\text{: Secret Key}}$

$\boxed{s = Sign(m, \mathcal{S})}$

Alice

$$\xrightarrow[s]{m = \text{``}Movie\ tomorrow?\text{''}}$$



Bob

$\boxed{\mathcal{P}\text{: Public Key}}$

$\boxed{Verify(m, s, \mathcal{P})}$

Signature Schemes

Traditional Solutions

- Discrete logarithm (DSA, ElGamal, ECDSA, ...)
- Factoring (RSA)

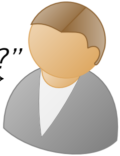
$\rightarrow$ Polynomial for a quantum computer [Shor 94]

$\boxed{\mathcal{S}\text{: Secret Key}}$

$\boxed{s = Sign(m, \mathcal{S})}$

Alice

$\xrightarrow[s]{m = \text{"Movie tomorrow?"}}$



Bob

$\boxed{\mathcal{P}\text{: Public Key}}$

$\boxed{Verify(m, s, \mathcal{P})}$

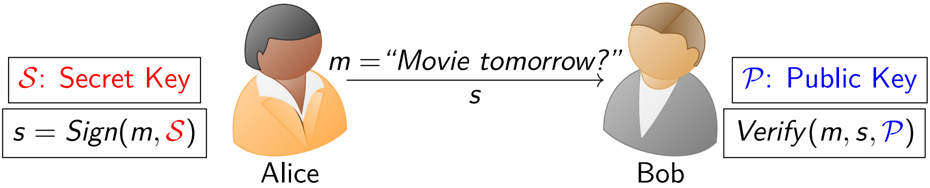
Signature Schemes

Traditional Solutions

- Discrete logarithm (DSA, ElGamal, ECDSA, ...)
- Factoring (RSA)

$\rightarrow$ Polynomial for a quantum computer [Shor 94]

Post-quantum signature schemes?



Multivariate Signature Scheme

Unbalanced Oil and Vinegar, informally [Kipnis, Patarin, Goubin, 1999]

- The legitimate signer solves a linear system to **sign**. EASY

Multivariate Signature Scheme

Unbalanced Oil and Vinegar, informally [Kipnis, Patarin, Goubin, 1999]

- The legitimate signer solves a linear system to **sign**. EASY
- An adversary solves a quadratic system to **forge** a signature. HARD

Multivariate Signature Scheme

Unbalanced Oil and Vinegar, informally [Kipnis, Patarin, Goubin, 1999]

- The legitimate signer solves a linear system to **sign**. EASY
- An adversary solves a quadratic system to **forge** a signature. HARD
- The receiver evaluates a quadratic map to verify a signature. EASY

Multivariate Signature Scheme

Unbalanced Oil and Vinegar, informally [Kipnis, Patarin, Goubin, 1999]

- The legitimate signer solves a linear system to **sign**. EASY
- An adversary solves a quadratic system to **forge** a signature. HARD
- The receiver evaluates a quadratic map to verify a signature. EASY

Multivariate vs Post-Quantum standards

- Multivariate: UOV, Rainbow, GeMSS, MAYO, VOX, ...

Multivariate Signature Scheme

Unbalanced Oil and Vinegar, informally [Kipnis, Patarin, Goubin, 1999]

- The legitimate signer solves a linear system to **sign**. EASY
- An adversary solves a quadratic system to **forge** a signature. HARD
- The receiver evaluates a quadratic map to verify a signature. EASY

Multivariate vs Post-Quantum standards

- Multivariate: UOV, Rainbow, GeMSS, MAYO, VOX, ...
- NIST Standards: Dilithium, Falcon, SPHINCS+ (**Lattices & Hash**)

Multivariate Signature Scheme

Unbalanced Oil and Vinegar, informally [Kipnis, Patarin, Goubin, 1999]

- The legitimate signer solves a linear system to **sign**. EASY
- An adversary solves a quadratic system to **forge** a signature. HARD
- The receiver evaluates a quadratic map to verify a signature. EASY

Multivariate vs Post-Quantum standards

- Multivariate: UOV, Rainbow, GeMSS, MAYO, VOX, ...
- NIST Standards: Dilithium, Falcon, SPHINCS+ (**Lattices & Hash**)
- Shorter signatures: suited for **low bandwidth applications**

UOV: Original formulation

Unbalanced Oil and Vinegar [Kipnis, Patarin, Goubin, 1999]

Private Key: - **structured** symmetric matrices $F = (F_1, \dots, F_k)$ in $(\mathbb{F}_q^{n\times n})^k$
- $A \in GL_n(\mathbb{F}_q)$ random change of variables

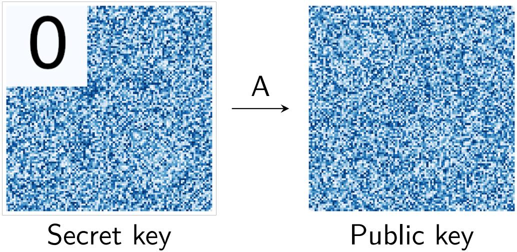


Figure: UOV Key Pair in $\mathbb{F}_{257}$

UOV: Original formulation

Unbalanced Oil and Vinegar [Kipnis, Patarin, Goubin, 1999]

Private Key: - **structured** symmetric matrices $F = (F_1, \dots, F_k)$ in $(\mathbb{F}_q^{n\times n})^k$

- $A \in GL_n(\mathbb{F}_q)$ random change of variables

Public Key: symmetric matrices $G = F \circ A$

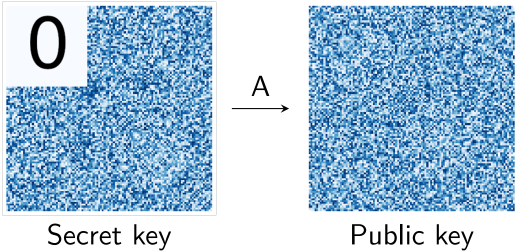


Figure: UOV Key Pair in $\mathbb{F}_{257}$

UOV: Original formulation

Unbalanced Oil and Vinegar [Kipnis, Patarin, Goubin, 1999]

Private Key: - **structured** symmetric matrices $F = (F_1, \ldots, F_k)$ in $(\mathbb{F}_q^{n \times n})^k$
- $A \in GL_n(\mathbb{F}_q)$ random change of variables

Public Key: symmetric matrices $G = F \circ A$

Link with standard multivariate cryptography

Private key polynomials: k quadratic forms $\boldsymbol{x}^T F_i \boldsymbol{x}$ **linear** in $x_1, \ldots, x_k$
Public key polynomials: k quadratic forms $\boldsymbol{x}^T G_i \boldsymbol{x}$ in n variables.

UOV: Original formulation

Unbalanced Oil and Vinegar [Kipnis, Patarin, Goubin, 1999]

Private Key: - **structured** symmetric matrices $F = (F_1, \ldots, F_k)$ in $(\mathbb{F}_q^{n\times n})^k$
- $A \in GL_n(\mathbb{F}_q)$ random change of variables

Public Key: symmetric matrices $G = F \circ A$

Link with standard multivariate cryptography

Private key polynomials: k quadratic forms $\boldsymbol{x}^T F_i \boldsymbol{x}$ **linear** in $x_1, \ldots, x_k$

Public key polynomials: k quadratic forms $\boldsymbol{x}^T G_i \boldsymbol{x}$ in n variables.

$x_1, \ldots, x_k \to$ **oil variables**

$x_k, \ldots, x_n \to$ **vinegar variables**

UOV: Original formulation

Unbalanced Oil and Vinegar [Kipnis, Patarin, Goubin, 1999]

Private Key: - **structured** symmetric matrices $F = (F_1, \dots, F_k)$ in $(\mathbb{F}_q^{n\times n})^k$
- $A \in GL_n(\mathbb{F}_q)$ random change of variables

Public Key: symmetric matrices $G = F \circ A$

Link with standard multivariate cryptography

Private key polynomials: k quadratic forms $\boldsymbol{x}^T F_i \boldsymbol{x}$ **linear** in $x_1, \dots, x_k$

Public key polynomials: k quadratic forms $\boldsymbol{x}^T G_i \boldsymbol{x}$ in n variables.

$x_1, \dots, x_k \to$ **oil variables**

$x_k, \dots, x_n \to$ **vinegar variables**

In practice: $\boldsymbol{n \leq 3k}$

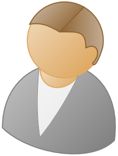
UOV: Signing process

Signing

A **signature** for the message $\boldsymbol{m} \in \mathbb{F}_q^k$ is a vector $\boldsymbol{x} \in \mathbb{F}_q^n$ such that $1 \leq i \leq k$, $G_i(\boldsymbol{x}) = m_i$

$\boxed{\text{(A,F)}}$

Alice



Bob

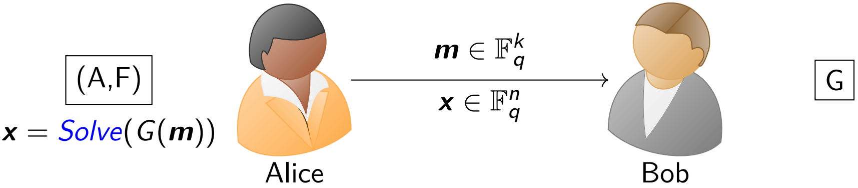
UOV: Signing process

Signing

A **signature** for the message $\boldsymbol{m} \in \mathbb{F}_q^k$ is a vector $\boldsymbol{x} \in \mathbb{F}_q^n$ such that

$$1 \leq i \leq k,\ G_i(\boldsymbol{x}) = m_i$$

- Alice signs: $\boldsymbol{x}$ solution of a linear system in $x_1, \ldots, x_k$.



UOV: Signing process

Signing

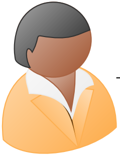
A **signature** for the message $\boldsymbol{m} \in \mathbb{F}_q^k$ is a vector $\boldsymbol{x} \in \mathbb{F}_q^n$ such that

$$1 \leq i \leq k,\ G_i(\boldsymbol{x}) = m_i$$

- Alice signs: $\boldsymbol{x}$ solution of a linear system in $x_1, \ldots, x_k$.
- Bob verifies: checks that for $1 \leq i \leq k$, $G_i(\boldsymbol{x}) = m_i$.

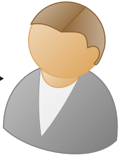
$\boxed{\text{(A,F)}}$

$\boldsymbol{x} = \mathit{Solve}(G(\boldsymbol{m}))$



Alice

$$\xrightarrow[\boldsymbol{x} \in \mathbb{F}_q^n]{\boldsymbol{m} \in \mathbb{F}_q^k}$$



Bob

$\boxed{\text{G}}$

$G(\boldsymbol{x}) \stackrel{?}{=} \boldsymbol{m}$

UOV: Signing process

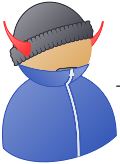
Signing

A **signature** for the message $\boldsymbol{m} \in \mathbb{F}_q^k$ is a vector $\boldsymbol{x} \in \mathbb{F}_q^n$ such that

$$1 \leq i \leq k, G_i(\boldsymbol{x}) = m_i$$

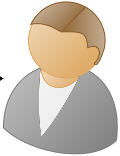
- Alice signs: $\boldsymbol{x}$ solution of a linear system in $x_1, \ldots, x_k$.
- Bob verifies: checks that for $1 \leq i \leq k$, $G_i(\boldsymbol{x}) = m_i$.
- Eve forges: $\boldsymbol{x}$ solution of a polynomial system in $x_1, \ldots, x_n$.

$\boldsymbol{x} = Solve(G(\boldsymbol{m}))$



Eve

$$\xrightarrow[\boldsymbol{x} \in \mathbb{F}_q^n]{\boldsymbol{m} \in \mathbb{F}_q^k}$$



Bob

$\boxed{G}$

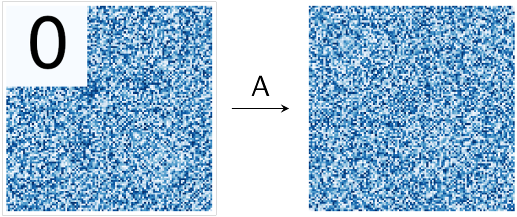
$G(\boldsymbol{x}) \stackrel{?}{=} \boldsymbol{m}$

UOV: Alternative formulation

Equivalent characterisation of the trapdoor [Beullens 2020]

Trapdoor: subspace $\mathcal{O}$ of dimension k such that

$$\forall (\boldsymbol{x}, \boldsymbol{y}) \in \mathcal{O}^2, \quad \boldsymbol{x}^T G_1 \boldsymbol{y} = \ldots = \boldsymbol{x}^T G_k \boldsymbol{y} = 0$$



Secret key Public key

Figure: UOV Key Pair in $\mathbb{F}_{257}$

UOV: Alternative formulation

Equivalent characterisation of the trapdoor [Beullens 2020]

Trapdoor: subspace $\mathcal{O}$ of dimension k such that

$$\forall(\boldsymbol{x},\boldsymbol{y}) \in \mathcal{O}^2, \quad \boldsymbol{x}^T G_1 \boldsymbol{y} = \ldots = \boldsymbol{x}^T G_k \boldsymbol{y} = 0$$

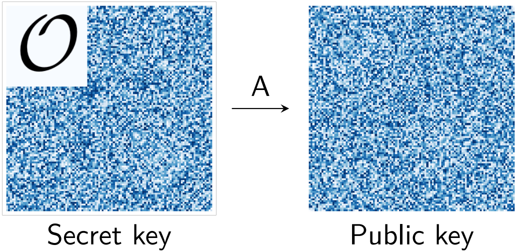


Figure: UOV Key Pair in $\mathbb{F}_{257}$

Cryptanalysis

Forgery

Goal: Find **a** signature $\boldsymbol{x} \in \mathbb{F}_q^n$ for a **single** message $M \in \mathbb{F}_q^k$.

$$V(M) = \{\boldsymbol{x} \in \mathbb{F}_q^n \quad | \quad \forall i \leq k, G_i(\boldsymbol{x}) = M_i\}$$

Cryptanalysis

Forgery

Goal: Find **a** signature $\boldsymbol{x} \in \mathbb{F}_q^n$ for a **single** message $M \in \mathbb{F}_q^k$.

$$V(M) = \{\boldsymbol{x} \in \mathbb{F}_q^n \quad | \quad \forall i \leq k, G_i(\boldsymbol{x}) = M_i\}$$

Computational problem: Find a point in a **variety of dimension** $n - k$

Cryptanalysis

Forgery

Goal: Find **a** signature $\boldsymbol{x} \in \mathbb{F}_q^n$ for a **single** message $M \in \mathbb{F}_q^k$.

$$V(M) = \{\boldsymbol{x} \in \mathbb{F}_q^n \quad | \quad \forall i \leq k, G_i(\boldsymbol{x}) = M_i\}$$

Computational problem: Find a point in a **variety of dimension** $n - k$

Key recovery

Goal: find an equivalent secret key to sign **any** message.

$$\mathcal{O} \subset \{\boldsymbol{x} \in \mathbb{F}_q^n \quad | \quad \forall i \leq k, G_i(\boldsymbol{x}) = 0\}$$

Cryptanalysis

Forgery

Goal: Find **a** signature $\boldsymbol{x} \in \mathbb{F}_q^n$ for a **single** message $M \in \mathbb{F}_q^k$.

$$V(M) = \{\boldsymbol{x} \in \mathbb{F}_q^n \quad | \quad \forall i \leq k, G_i(\boldsymbol{x}) = M_i\}$$

Computational problem: Find a point in a **variety of dimension** $n - k$

Key recovery

Goal: find an equivalent secret key to sign **any** message.

$$\mathcal{O} \subset \{\boldsymbol{x} \in \mathbb{F}_q^n \quad | \quad \forall i \leq k, G_i(\boldsymbol{x}) = 0\}$$

Computational problem: Find a **linear subspace of dimension** k **in** $V(0)$

Contribution

Input $G, v \in \mathcal{O}$ $\xrightarrow{O(kn^\omega)}$ **Output** $\mathcal{O}$

Main result [P. 2023]

- Polynomial-time algorithm that takes as input **one vector** in $\mathcal{O}$ and the public key G, and returns a basis of $\mathcal{O}$.

Contribution

Input $G, v \in \mathcal{O}$ $\xrightarrow{O(kn^{\omega})}$ **Output** $\mathcal{O}$

Main result [P. 2023]

- Polynomial-time algorithm that takes as input **one vector** in $\mathcal{O}$ and the public key G, and returns a basis of $\mathcal{O}$.
- Polynomial-time algorithm that takes as input a vector $x \in \mathbb{F}_q^n$ and the public key G, and that answers the question "$x \in \mathcal{O}$?".

Contribution

Input $G, v \in \mathcal{O}$ $\xrightarrow{O(kn^\omega)}$ **Output** $\mathcal{O}$

Main result [P. 2023]

- Polynomial-time algorithm that takes as input **one vector** in $\mathcal{O}$ and the public key G, and returns a basis of $\mathcal{O}$.
- Polynomial-time algorithm that takes as input a vector $x \in \mathbb{F}_q^n$ and the public key G, and that answers the question "$x \in \mathcal{O}$?".

Consequence for the security of UOV

- An attacker needs to find a single vector in $\mathcal{O}$ to retrieve the **secret key** up to equivalence. This is enough to sign **any** message.

Contribution

Input $G, v \in \mathcal{O}$ → $O(kn^{\omega})$ → **Output** $\mathcal{O}$

Main result [P. 2023]

- Polynomial-time algorithm that takes as input **one vector** in $\mathcal{O}$ and the public key G, and returns a basis of $\mathcal{O}$.
- Polynomial-time algorithm that takes as input a vector $x \in \mathbb{F}_q^n$ and the public key G, and that answers the question "$x \in \mathcal{O}$?".

Consequence for the security of UOV

- An attacker needs to find a single vector in $\mathcal{O}$ to retrieve the **secret key** up to equivalence. This is enough to sign **any** message.
- Finding a vector of $\mathcal{O}$ remains challenging.

State-of-the-art of Key Recovery Attacks

Reconciliation [Ding, Yang, Chen, Chen, Cheng 2008], [Beullens 2020/21]

Key recovery attacks benefit from knowledge of some vectors of $\mathcal{O}$: additional equations in quadratic system.

State-of-the-art of Key Recovery Attacks

Reconciliation [Ding, Yang, Chen, Chen, Cheng 2008], [Beullens 2020/21]

Key recovery attacks benefit from knowledge of some vectors of $\mathcal{O}$: additional equations in quadratic system. $\rightarrow$ Reconciliation

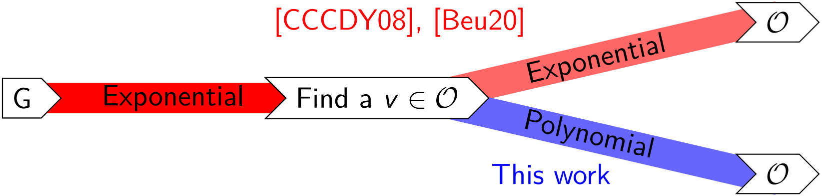
State-of-the-art of Key Recovery Attacks

Reconciliation [Ding, Yang, Chen, Chen, Cheng 2008], [Beullens 2020/21]

Key recovery attacks benefit from knowledge of some vectors of $\mathcal{O}$: additional equations in quadratic system. $\rightarrow$ Reconciliation

This work

Any vector in $\mathcal{O}$ characterizes it. $\rightarrow$ Polynomial reconciliation



Contribution: The algorithm

Equivalent characterisation of the trapdoor [Beullens 2020]

Trapdoor: subspace $\mathcal{O}$ of dimension k such that

$$\forall(\boldsymbol{x}, \boldsymbol{y}) \in \mathcal{O}^2, \quad \boldsymbol{x}^T G_1 \boldsymbol{y} = \cdots = \boldsymbol{x}^T G_k \boldsymbol{y} = 0$$

Contribution: The algorithm

Equivalent characterisation of the trapdoor [Beullens 2020]

Trapdoor: subspace $\mathcal{O}$ of dimension k such that

$$\forall(\boldsymbol{x},\boldsymbol{y}) \in \mathcal{O}^2, \quad \boldsymbol{x}^T G_1 \boldsymbol{y} = \cdots = \boldsymbol{x}^T G_k \boldsymbol{y} = 0$$

Reformulation

$$\forall \boldsymbol{x} \in \mathcal{O}, \quad \mathcal{O} \subset J(\boldsymbol{x}) := \ker(\boldsymbol{x}^T G_1) \cap \ldots \cap \ker(\boldsymbol{x}^T G_k)$$

Contribution: The algorithm

Equivalent characterisation of the trapdoor [Beullens 2020]

Trapdoor: subspace $\mathcal{O}$ of dimension k such that

$$\forall (\boldsymbol{x}, \boldsymbol{y}) \in \mathcal{O}^2, \quad \boldsymbol{x}^T G_1 \boldsymbol{y} = \cdots = \boldsymbol{x}^T G_k \boldsymbol{y} = 0$$

Reformulation

$$\forall \boldsymbol{x} \in \mathcal{O}, \quad \mathcal{O} \subset J(\boldsymbol{x}) := \ker(\boldsymbol{x}^T G_1) \cap \ldots \cap \ker(\boldsymbol{x}^T G_k)$$

Observation

$J(\boldsymbol{x})$ is of dimension $n - k$.

Contribution: The algorithm

Public key: $G \in (\mathbb{F}_q^{n \times n})^k$ Secret vector: $\boldsymbol{x} \in \mathbb{F}_q^n$ $\dim(J(\boldsymbol{x})) = n - k$

Reduction

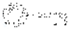
Restriction $G_{|J(\boldsymbol{x})} \to$ UOV instance with **smaller parameters** and one secret vector.

Contribution: The algorithm

Public key: $G \in (\mathbb{F}_q^{n \times n})^k$ Secret vector: $\boldsymbol{x} \in \mathbb{F}_q^n$ $\dim(J(\boldsymbol{x})) = n - k$

Reduction

Restriction $G_{|J(\boldsymbol{x})} \rightarrow$ UOV instance with **smaller parameters** and one secret vector.



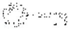
$\longrightarrow \boxed{G, \boldsymbol{x}}$

Contribution: The algorithm

Public key: $G \in (\mathbb{F}_q^{n\times n})^k$ Secret vector: $\boldsymbol{x} \in \mathbb{F}_q^n$ $\dim(J(\boldsymbol{x})) = n - k$

Reduction

Restriction $G_{|J(\boldsymbol{x})} \rightarrow$ UOV instance with **smaller parameters** and one secret vector.



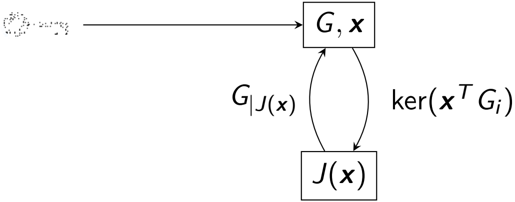
$$\begin{tikzcd} \boxed{G, \boldsymbol{x}} \arrow[d, bend left, "\ker(\boldsymbol{x}^T G_i)"] \\ \boxed{J(\boldsymbol{x})} \end{tikzcd}$$

Contribution: The algorithm

Public key: $G \in (\mathbb{F}_q^{n\times n})^k$ Secret vector: $\boldsymbol{x} \in \mathbb{F}_q^n$ $\dim(J(\boldsymbol{x})) = n - k$

Reduction

Restriction $G_{|J(\boldsymbol{x})} \to$ UOV instance with **smaller parameters** and one secret vector.

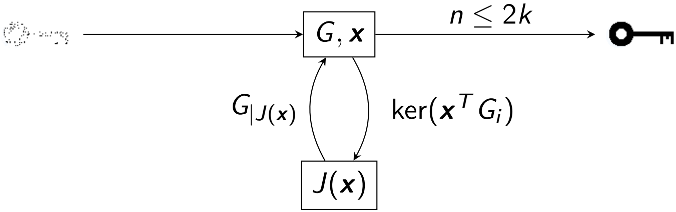


Contribution: The algorithm

Public key: $G \in (\mathbb{F}_q^{n\times n})^k$ Secret vector: $\boldsymbol{x} \in \mathbb{F}_q^n$ $\dim(J(\boldsymbol{x})) = n - k$

Reduction

Restriction $G_{|J(\boldsymbol{x})} \to$ UOV instance with **smaller parameters** and one secret vector.



Concluding the attack

$n \leq 2k \to$ broken in **polynomial time**. [Kipnis, Shamir 1998]

Contribution: Complexity analysis

Public key: $G \in (\mathbb{F}_q^{n\times n})^k$ Secret vector: $\boldsymbol{x} \in \mathbb{F}_q^n$ $\dim(J(\boldsymbol{x})) = n - k$

Complexity of the attack

1. Computing B, a basis of $J(\boldsymbol{x})$ $O(n^\omega)$ and $2 \leq \omega \leq 3$

Contribution: Complexity analysis

Public key: $G \in (\mathbb{F}_q^{n\times n})^k$ Secret vector: $\boldsymbol{x} \in \mathbb{F}_q^n$ $\dim(J(\boldsymbol{x})) = n - k$

Complexity of the attack

1. Computing B, a basis of $J(\boldsymbol{x})$ — $O(n^\omega)$ and $2 \leq \omega \leq 3$
2. Computing the restrictions: $G_{i|J(\boldsymbol{x})} = B^T G_i B$ — $O(kn^\omega)$

Contribution: Complexity analysis

Public key: $G \in (\mathbb{F}_q^{n\times n})^k$ Secret vector: $\boldsymbol{x} \in \mathbb{F}_q^n$ $\dim(J(\boldsymbol{x})) = n - k$

Complexity of the attack

1. Computing B, a basis of $J(\boldsymbol{x})$ — $O(n^\omega)$ and $2 \leq \omega \leq 3$
2. Computing the restrictions: $G_{i|J(\boldsymbol{x})} = B^T G_i B$ — $O(kn^\omega)$
3. Kipnis-Shamir attack or kernel computations — $O(kn^\omega)$
4. Total cost: $\boldsymbol{O(kn^\omega)}$

Contribution: Experimental results

	NIST SL	n	m	$\mathbb{F}_q$	|pk| (bytes)	|sk| (bytes)	|cpk| (bytes)	|sig+salt| (bytes)
ov-Ip	1	112	44	$\mathbb{F}_{256}$	278 432	237 912	43 576	128
ov-Is	1	160	64	$\mathbb{F}_{16}$	412 160	348 720	66 576	96
ov-III	3	184	72	$\mathbb{F}_{256}$	1 225 440	1 044 336	189 232	200
ov-V	5	244	96	$\mathbb{F}_{256}$	2 869 440	2 436 720	446 992	260

Figure: Modern UOV[Beullens, Chen, Hung, Kannwischer, Peng, Shih, Yang 2023]

Contribution: Experimental results

	NIST SL	n	m	$\mathbb{F}_q$	|pk| (bytes)	|sk| (bytes)	|cpk| (bytes)	|sig+salt| (bytes)
ov-Ip	1	112	44	$\mathbb{F}_{256}$	278 432	237 912	43 576	128
ov-Is	1	160	64	$\mathbb{F}_{16}$	412 160	348 720	66 576	96
ov-III	3	184	72	$\mathbb{F}_{256}$	1 225 440	1 044 336	189 232	200
ov-V	5	244	96	$\mathbb{F}_{256}$	2 869 440	2 436 720	446 992	260

Figure: Modern UOV[Beullens, Chen, Hung, Kannwischer, Peng, Shih, Yang 2023]

n	112	160	184	244
Time	1.7s	4.4s	5.7s	13.3s

Figure: Implementation of our attack with native **sagemath** functions on a laptop

Contribution: Experimental results

	NIST SL	n	m	$\mathbb{F}_q$	|pk| (bytes)	|sk| (bytes)	|cpk| (bytes)	|sig+salt| (bytes)
ov-Ip	1	112	44	$\mathbb{F}_{256}$	278 432	237 912	43 576	128
ov-Is	1	160	64	$\mathbb{F}_{16}$	412 160	348 720	66 576	96
ov-III	3	184	72	$\mathbb{F}_{256}$	1 225 440	1 044 336	189 232	200
ov-V	5	244	96	$\mathbb{F}_{256}$	2 869 440	2 436 720	446 992	260

Figure: Modern UOV[Beullens, Chen, Hung, Kannwischer, Peng, Shih, Yang 2023]

n	112	160	184	244
Time	1.7s	4.4s	5.7s	13.3s

Figure: Implementation of our attack with native **sagemath** functions on a laptop

Reminder

This is the time it takes, **given** one vector in $\mathcal{O}$, to retrieve a basis of $\mathcal{O}$.

Gap between key recovery and forgery

Key recovery versus forgery

- Experimentally, observe large gap between forgery attacks and key recovery attacks.

Gap between key recovery and forgery

Key recovery versus forgery

- Experimentally, observe large gap between forgery attacks and key recovery attacks.
- Key size: $G \in (\mathbb{F}_q^{n\times n})^k, n = \lceil 2.5k \rceil$

Gap between key recovery and forgery

Key recovery versus forgery

- Experimentally, observe large gap between forgery attacks and key recovery attacks.
- Key size: $G \in (\mathbb{F}_q^{n\times n})^k, n = \lceil 2.5k \rceil$

Gap between key recovery and forgery

Key recovery versus forgery

- Experimentally, observe large gap between forgery attacks and key recovery attacks.
- Key size: $G \in (\mathbb{F}_q^{n\times n})^k, n = \lceil 2.5k \rceil$

k	9	10	11	12	13	14	15	16	17
Forgery	0.1s	0.3s	1s	4s	20s	144s	930s	2h	14h
Recovery	40s	1h	2h	>11000h					

Figure: CPU-time in $\mathbb{F}_{31}$ with **msolve** [Berthomieu, Eder, Safey el Din, 2021]

Gap between key recovery and forgery

Key recovery versus forgery

- Experimentally, observe large gap between forgery attacks and key recovery attacks.
- Key size: $G \in (\mathbb{F}_q^{n\times n})^k, n = \lceil 2.5k \rceil$

k	9	10	11	12	13	14	15	16	17
Forgery	0.1s	0.3s	1s	4s	20s	144s	930s	2h	14h
Recovery	40s	1h	2h	>11000h					

Figure: CPU-time in $\mathbb{F}_{31}$ with **msolve** [Berthomieu, Eder, Safey el Din, 2021]

Key Recovery

This is the time it takes to retrieve **one** vector in $\mathcal{O}$.

Forgery attacks are key-recovery attacks

Forgery

Goal: forge **a** signature $\boldsymbol{x} \in \mathbb{F}_q^n$ for a **single** message $M \in \mathbb{F}_q^k$.

$$V(M) = \{\boldsymbol{x} \in \mathbb{F}_q^n \quad | \quad \forall i \leq k, G_i(\boldsymbol{x}) = M_i\}$$

Reminder: $\mathcal{O} \subset V(O)$

Forgery attacks are key-recovery attacks

Forgery

Goal: forge **a** signature $\boldsymbol{x} \in \mathbb{F}_q^n$ for a **single** message $M \in \mathbb{F}_q^k$.

$$V(M) = \{\boldsymbol{x} \in \mathbb{F}_q^n \quad | \quad \forall i \leq k, G_i(\boldsymbol{x}) = M_i\}$$

Reminder: $\mathcal{O} \subset V(O)$

Key recovery from forgery

Attempt to forge a signature $\boldsymbol{x}$ for the message 0 until $\boldsymbol{x}$ belongs to $\mathcal{O}$.

Forgery attacks are key-recovery attacks

Forgery

Goal: forge **a** signature $\boldsymbol{x} \in \mathbb{F}_q^n$ for a **single** message $M \in \mathbb{F}_q^k$.

$$V(M) = \{\boldsymbol{x} \in \mathbb{F}_q^n \quad | \quad \forall i \leq k, G_i(\boldsymbol{x}) = M_i\}$$

Reminder: $\mathcal{O} \subset V(O)$

Key recovery from forgery

Attempt to forge a signature $\boldsymbol{x}$ for the message 0 until $\boldsymbol{x}$ belongs to $\mathcal{O}$.

n	112	160	184	244
Time	0.2s	0.5s	0.7s	1.5s

Figure: Implementation of our test $\boldsymbol{x} \in \mathcal{O}$? on a laptop

Perspectives

Contribution [P. 2023]

- One secret vector $\rightarrow$ equivalent easy UOV instance.
- Forgery attack $\rightarrow$ key recovery attack.

Perspectives

Contribution [P. 2023]

- One secret vector $\rightarrow$ equivalent easy UOV instance.
- Forgery attack $\rightarrow$ key recovery attack.

New directions

- Improve forgery attacks to improve key recovery attacks

Perspectives

Contribution [P. 2023]

- One secret vector $\rightarrow$ equivalent easy UOV instance.
- Forgery attack $\rightarrow$ key recovery attack.

New directions

- Improve forgery attacks to improve key recovery attacks
- Key recovery attacks: **two** vectors in $\mathcal{O}$ at once [Beullens 2020]

Perspectives

Contribution [P. 2023]

- One secret vector $\rightarrow$ equivalent easy UOV instance.
- Forgery attack $\rightarrow$ key recovery attack.

New directions

- Improve forgery attacks to improve key recovery attacks
- Key recovery attacks: **two** vectors in $\mathcal{O}$ at once [Beullens 2020]
- Can we find only **one** vector faster than **two**?

Perspectives

Contribution [P. 2023]

- One secret vector $\rightarrow$ equivalent easy UOV instance.
- Forgery attack $\rightarrow$ key recovery attack.

New directions

- Improve forgery attacks to improve key recovery attacks
- Key recovery attacks: **two** vectors in $\mathcal{O}$ at once [Beullens 2020]
- Can we find only **one** vector faster than **two**?
- Side-channel attacks [Aulbach, Campos, Kramer, Samardjiska, Stottinger]

Perspectives

Contribution [P. 2023]

- One secret vector $\rightarrow$ equivalent easy UOV instance.
- Forgery attack $\rightarrow$ key recovery attack.

New directions

- Improve forgery attacks to improve key recovery attacks
- Key recovery attacks: **two** vectors in $\mathcal{O}$ at once [Beullens 2020]
- Can we find only **one** vector faster than **two**?
- Side-channel attacks [Aulbach, Campos, Kramer, Samardjiska, Stottinger]

Paper

Preprint to be released, stay tuned!

Thank you for your attention!