FOX
Public Key Signature Scheme
NIST PQC Seminar

Gilles Macario-Rat

Orange

April 23, 2024

Table of Contents

FOX is part of the VOX Submission to the NIST call for proposals

Submitters : Benoît Cogliati, Jean-Charles Faugère, Pierre-Alain Fouque, Louis Goubin, Robin Larrieu, Gilles Macario-Rat, Brice Minaud, Jacques Patarin
Official site of VOX submission : `https://vox-sign.com`

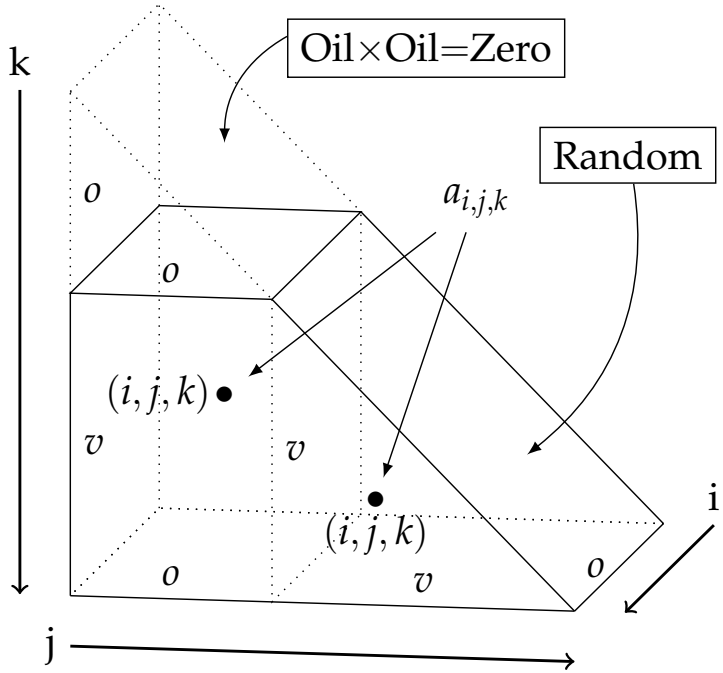
FOX In a nutshell

- FOX is a simplified version of VOX which itself is a multivariate cryptographic signature scheme of the family of UOV.
- FOX keys **Pub** and **Sec** are sets of multivariate polynomials.
- **Pub** and **Sec** are linked by the relation : $\mathbf{Sec} = \mathbf{S} \circ \mathbf{Pub} \circ \mathbf{T}$, where **S** and **T** are two linear bijective mappings (they are also part of the secret key).
- There is a trapdoor that enables to find a solution in x for a given h of $\mathbf{Sec}(x) = h$.
- FOX uses hash and sign paradigm : to sign a message M, apply a hash function H to M, and find a solution σ to the equation $\mathbf{Pub}(x) = H(M)$ using **S**, **T** and the trapdoor, then σ is the signature of M.
- To verify a signature (M, σ), verify that $\mathbf{Pub}(\sigma)$ and $H(M)$ are equal.
- FOX uses two specific techniques in addition to plain UOV.
  - the PBB compression: Petzoldt, Bulygin, and Buchmann 2010
  - the $\hat{+}$ technique: Faugère et al. 2022

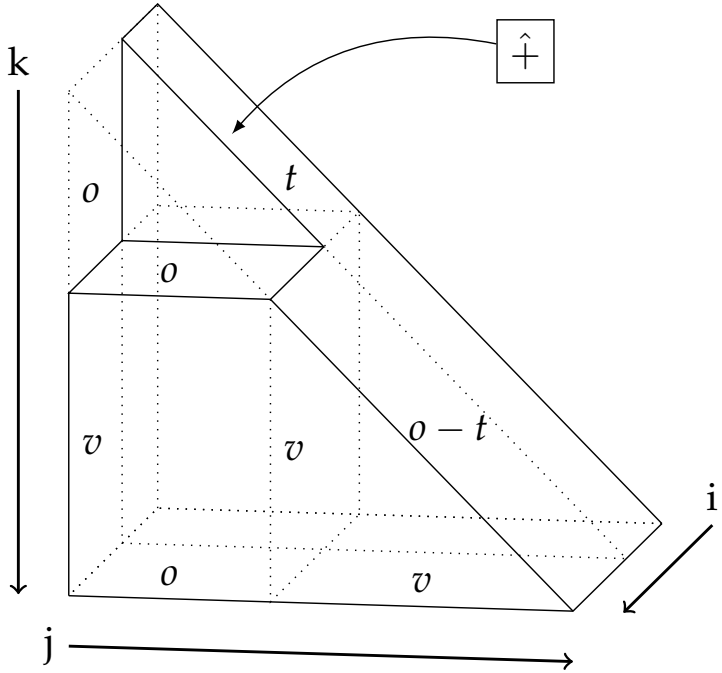
Design

Secret key UOV vs. FOX: $\hat{+}$

$$\mathbf{Sec} = \left\{ \sum_{1 \le k \le j \le o+v} a_{i,j,k} x_j x_k \quad i = 1, \ldots, o \right\} \quad o \text{ oil and } v \text{ vinegar variables}$$



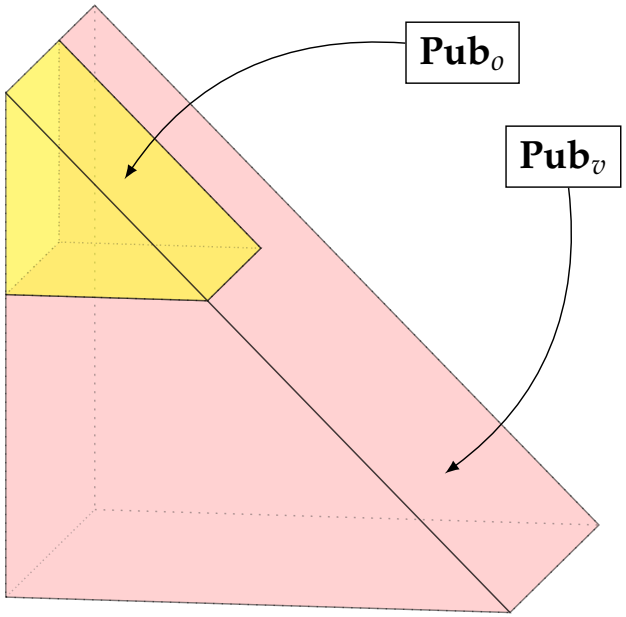
(a) UOV Secret Key



(b) FOX Secret Key

Design

Public key FOX – PBB



The trapdoor:

$$\mathbf{Sec} = \mathbf{S} \circ \mathbf{Pub} \circ \mathbf{T}, \quad \mathbf{Sec}_o = 0.$$

Sufficient way to mix oil and vinegar

$$\mathbf{S} = \begin{pmatrix} I_t & \mathbf{S}'_{t\times(o-t)} \\ 0 & I_{o-t} \end{pmatrix}$$

$$\mathbf{T} = \begin{pmatrix} I_o & \mathbf{T}'_{o\times v} \\ 0 & I_v \end{pmatrix}$$

As a consequence, the PBB property :

$$\mathbf{Pub}_o = -(\mathbf{S} \circ \mathbf{Pub}_v \circ \mathbf{T})_o.$$

$\mathbf{S}$, $\mathbf{T}$ and $\mathbf{Pub}_v$ can be derived from a seed and a DRBG

Design

Inversion of **Sec**

$$\text{Solve} \quad \mathbf{Sec}_i(x_1, \ldots, x_o, x_{o+1}, \ldots, x_{o+v}) = y_i, \quad i = 1, \ldots, o.$$

- Set the vinegar variables with random values : call it V.
- The last $o - t$ equations form a linear system L in o variables. Solve : the set of solutions S_O is an affine vector space of dimension t, it can be expressed as affine expressions in t free variables, $z_1, \ldots, z_t$, with probability $\approx 1 - 1/q^{t+1}$

$$S_O = \left\{(x_1, \ldots, x_o) = A_0 + z_1 A_1 + \cdots + z_t A_t, (z_1, \ldots, z_t) \in \mathbb{F}_q^t\right\}.$$

- Replace these expressions in the first t equations : we get a system Q of t non homogeneous quadratic equations in t variables. Solve, then pick at random one solution $(z_1, \ldots, z_t)$ if any.
- Evaluate the solution of S_O with $(z_1, \ldots, z_t)$, this gives O.
- Concatenate O with V, this gives one solution.

Implementation

Small System Solver

- The deliberate choice of FOX is to compute Gröbner basis of "regular systems" only, i.e. systems that behave as the generic system (system with symbolic coefficients)
- Advantages : a pre-computation is possible, thus faster execution ; algorithm is deterministic thus avoid side channel attacks like timing attacks.
- Requires only three steps
  1. Computation of a "Graded Reverse Lexicographic" Gröbner basis
  2. Conversion to a "Lexicographic" Gröbner basis
  3. Solving a univariate polynomial over $\mathbb{F}_q$ (degree is 2^t)

Implementation
Random Generation

- FOX uses a deterministic random byte generator base on `shake256`
- Elements of the finite field are uniformly generated using sampling and rejection over $[0, q-1]$.

Implementation

Sizes & Performances

Table. Parameter sets and corresponding compressed public key and signature sizes for the FOX signature scheme.

Variant	Security Level	q	o	v	t	Signature	Public Key
FOX-I	1	251	49	75	8	124 B	50,241 B
FOX-III	3	4093	70	106	8	264 B	231,121 B
FOX-V	5	65521	91	140	8	462 B	694,892 B

Table. Times in ms for the FOX signature scheme. The second values (*) are obtained with a pre-computed key, public or secret. Processor : Intel(R) Core(TM) i5-1145G7 @ 2.60GHz

Variant	Security Level	Key Generation	Sign	Sign (*)	Verify	Verify (*)
FOX-I	1	9.96	4.14	4.04	1.09	0.06
FOX-III	3	39.15	6.14	3.97	5.59	0.14
FOX-V	5	166.38	29.28	23.51	10.40	0.59

Security Analysis

Known attacks

- **Attacks related to the signature protocol and/or hash function** i.e finding collisions of the hash function.
- **Direct attacks** Inverting the system issued from the public key : Best solving algorithm is Hybrid-F5.
- **Key recovery attacks** Trying to recover an equivalent description of the secret key, or the secret "Oil space"
  - "Rectangular Minrank Attack" Solving by "Support Minors Modeling" or finding simultaneously a linear combination of matrices issued from the public key and its kernel
  - "Kipnis-Shamir", Complexity $\approx q^{v-o+t}$
  - "UOV distinguishing attacks", prevents from guessing at least two equations without $\hat{+}$ perturbation : security margin of q^{2t}

Thank you !

Questions ?

PROV

Short Post-Quantum Multivariate Signatures from Minimal Assumptions

Benoît Cogliati, Jean-Charles Faugère, Pierre-Alain Fouque, Louis Goubin, Robin Larrieu,
Gilles Macario-Rat, Brice Minaud, Jacques Patarin, Jocelyn Ryckeghem

https://prov-sign.github.io/

NIST online seminar, 23 April 2024

Why FOX *and* PROV?

Why UOV? Enables short post-quantum signatures.

Why VOX/FOX? reinforce UOV security (heuristically).

Why PROV? add provable security to UOV, from minimal assumptions.

Roadmap:

1. What are we proving, what's the point.
2. PROV design and performance.
3. Recent updates.

Hash-and-Sign paradigm

Hash-and-Sign signature Given: H = hash function.

Public key: **P** = one-way function *with trapdoor*.

Secret key: trapdoor **T** of **P**, allows to compute $\mathbf{P}^{-1}$.

- **Sign(*m*):** $\sigma = \mathbf{P}^{-1}(H(m))$, computed using **T**.
- **Verify(*m*,σ):** check $\mathbf{P}(\sigma) = H(m)$.

For UOV:

P ≈ random system of quadratic equations.

Inverting such a system = average-case **MQ problem.**

UOV trapdoor

For UOV:

P ≈ random system of quadratic equations over $\mathbb{K}^n$.

We also need a trapdoor **T**.

For UOV:

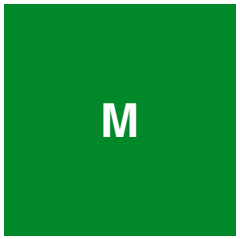
$\exists$ large subspace $S \subseteq \mathbb{K}^n$: $\forall s \in S$, **P**$(s) = 0$.

Trapdoor: T = S.

Graphically

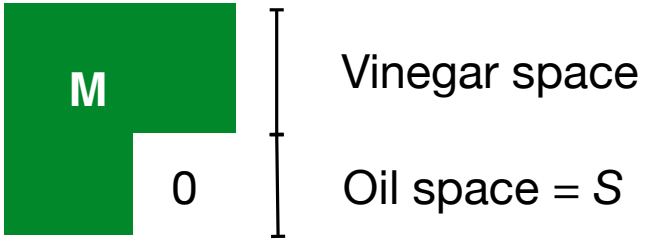
One quadratic equation **F**(x) for $x \in \mathbb{K}^n$...

... = one matrix



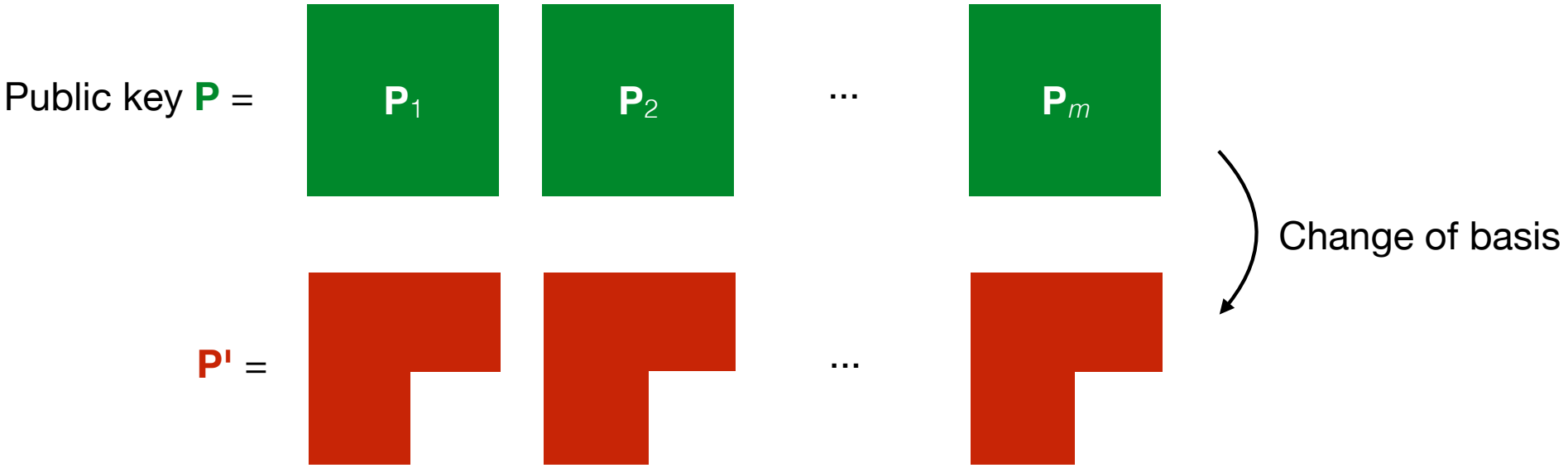
letting $\mathbf{F}(x) = x^T\mathbf{M}x$.

A matrix *with trapdoor* =



Public key

UOV public key = m quadratics equations, with dim(S) = m.



Using the trapdoor

Want to compute $\mathbf{P}^{-1}(\mathrm{H}(m))$. Compute $\mathbf{P'}^{-1}$ instead.

1. Sample vinegar v.
2. Observe $\mathbf{P'}_i(v + o) = (v + o)^T\mathbf{P'}_i(v + o)$

$$= v^T\mathbf{P'}_i v + v^T\mathbf{P'}_i o + o^T\mathbf{P'}_i v + o^T\mathbf{P'}_i o$$

This is linear in oil variable $o \in S \Rightarrow$ solve in o. **Done.**

Key point: m equations, m degrees of freedom.

What about security?

Need hardness of **average-case MQ**.

Need hardness of **UOV indistinguishability assumption**.

UOV indistinguishability assumption

A random system of m quadratic equations in n variables

is indistinguishable from:

A random system of m quadratic equations in n variables,
with trapdoor suspace of dimension m.

In practice: $m \approx n/3$.

Are we done?

Need hardness of **average-case MQ**.

Need hardness of **UOV indistinguishability**.

Theorem?

If these two assumptions hold, UOV is secure.

That doesn't work...

EUF-CMA security: adversary has access to a signature oracle.

Is proving UOV fixable?

Expectation: **no**.

Reason: “Not provable because not true” — UOV signatures leak information.

- Far away in statistical distance from what we can simulate [CFGM24, app. B].
- Attacker can use signatures to learn *e.g.* complementary spaces of the secret oil space.

Timeline

[SSH11] Koichi Sakumoto, Taizo Shirai, and Harunaga Hiwatari. *On provable security of UOV and HFE signature schemes against chosen-message attack*. PQCrypto 2011.

Proposes a tweak to achieve provable security, but flawed proof. Solution proposed in:

[CDP22] Sanjit Chatterjee, M. Prem Laxman Das, and Tapas Pandit. *Revisiting the security of salted UOV signature*. INDOCRYPT 2022.

Proposes fix, but requires very large field with $1/\mathbb{K} \approx 2^{-\lambda} \Rightarrow$ impractical parameters.

[KX24] Haruhisa Kosuge and Keita Xagawa. *Probabilistic hash-and-sign with retry in the quantum random oracle model*. PKC 2024.

Quantum proof of the SSH11 design with QROM. Somewhat loose bounds. Variability in signing time.

[CFGM24] Benoît Cogliati, Pierre-Alain Fouque, Louis Goubin, Brice Minaud. *New Security Proofs and Techniques for Hash-and-Sign with Retry Signature Schemes*. ePrint 2024/609 (yesterday!).

Sharp bounds, but only in the classical setting (for now).

Recent results

New Security Proofs and Techniques for Hash-and-Sign with Retry Signature Schemes

Benoît Cogliati[1], Pierre-Alain Fouque[2], Louis Goubin[3], Brice Minaud[4]

[1]Thales DIS France SAS [2]Université de Rennes [3]Laboratoire de Mathématiques de Versailles, UVSQ, CNRS, Université Paris-Saclay, France [4]École Normale Supérieure, PSL University, CNRS, Inria, France

Abstract. Hash-and-Sign with Retry is a popular technique to design efficient signature schemes from code-based or multivariate assumptions. Contrary to Hash-and-Sign signatures based on preimage-sampleable functions as defined by Gentry, Peikert and Vaikuntanathan (STOC 2008), trapdoor functions in code-based and multivariate schemes are not surjective. Therefore, the standard approach uses random trials. Kosuge and Xagawa (PKC 2024) coined it the Hash-and-Sign with Retry paradigm.

As many attacks have appeared on code-based and multivariate schemes, we think it is important for the ongoing NIST competition to look at the security proofs of these schemes. The original proof of Sakumoto, Shirai, and Hiwatari (PQCrypto 2011) was flawed, then corrected by Chatterjee, Das and Pandit (INDOCRYPT 2022). The fix is still not sufficient, as it only works for

Investigates provable security of UOV variants in detail, including MAYO, PROV, [SSH11]-modified UOV. (By subset of PROV authors.)

[SSH11] solution

1.Sample vinegar v.

2. Get linear system in oil variable $o \in S \Rightarrow$ solve in o.

Key point: m equations, m degrees of freedom.

[SSH11] modification: resample salt instead of resampling vinegar.

Caveat: variable signing time.

PROV solution

Recall: inverting **P** using **T** ⇔ solving system of m equations in $\dim(S) = m$ variables

PROV achieves provable security by combining two ideas:

1. [SSH11] idea: resample salt instead of vinegar.

2. Slighly larger oil space: $m + \delta$ variables for m equations
   ⇒ Probability of resampling dramatically decreases.

PROV: $\delta = 8$ ⇒ probability of resampling $\approx 2^{-71}$.

End result: summary of PROV

Theorem

PROV is secure in the standard EUF-CMA model based **only** on:

average-case MQ, UOV indistinguishability assumption

(and the security of symmetric cryptography).

True in the **classical** *and* **quantum** settings [KX24,CFGM24].

Core reason: PROV signatures leak *no* information.

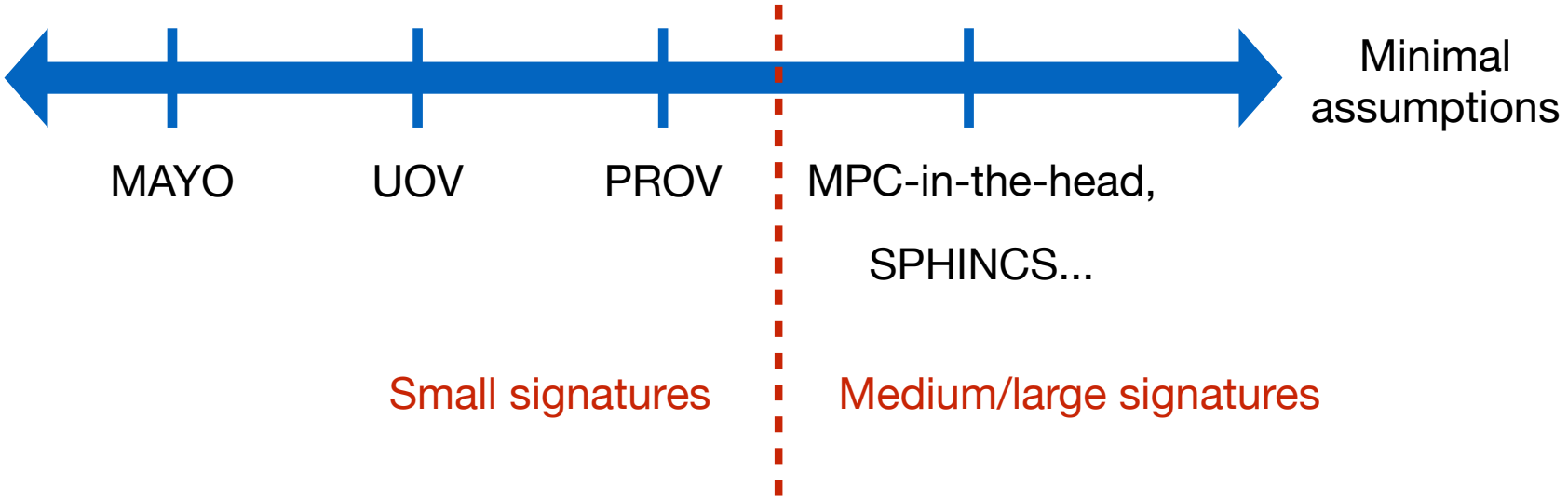
Proof is tight enough to be used to derive actual parameters.

+ additional guarantees via BUFF framework.

Comparison with UOV-based schemes

MAYO	UOV	PROV
• Proof based on MQ, UOV-ind, and new assumption. • Small PK.	• No proof. • Large PK.	• Proof based on MQ, UOV-ind. • Large PK.

Can you go further?



MPC-in-head has proof based on *only* symmetric assumptions. (Or *e.g.* MQ + sym.)

Absolute minimum for signatures!

PROV = furthest we can go towards minimal assumptions, with short signatures.

Parameters and performance

Sizes

	Signature	Public key	Secret key	Exp. Sec. key
Level I	166 b	81 Kb	48 b	237 Kb
Level III	238 b	252 Kb	72 b	752 Kb
Level V	310 b	589 Kb	96 b	1.75 Mb

Speed

	KeyGen	Sign	Verify	Exp. Sign
Level I	1.22 ms	0.136 ms	0.0544 ms	0.0418 ms
Level III	4.57 ms	0.371 ms	0.171 ms	0.110 ms
Level V	13.2 ms	0.780 ms	0.389 ms	0.228 ms

Measured on Intel Core i3 @3.6GHz, Coffee Lake, with AVX2, no Turbo boost, using (public) optimized implementation.

News

- Feb. 2024: PROV 1.1 fixed a bug in the specification.
- April 2024: PROV 1.2 is released (today!):

  `https://prov-sign.github.io/`

  1. New optimized implementation on AVX2.
  2. Sharper proofs based on [CFGM24].
  3. (Other minor changes: AES for seed expansion, etc.)

What's next?

- Sharper proof in quantum setting.
- Optimized implementation for Haswell (scheduled for May/June).

Thank you!