Efficient Implementations of Rainbow and UOV using AVX2

[1]Kyung-Ah Shim, [1]Sangyub Lee, and [2]**Namhun Koo**

[1]National Institute for Mathematical Sciences, Republic of Korea
[2]Institute for Mathematical Sciences, Ewha Womans University, Republic of Korea

CHES 2022
19 September, 2022

Contents

MQ-PKC (1/5)

Key Generation

- $\mathcal{P} = \mathcal{S} \circ \mathcal{F} \circ \mathcal{T}$: public key
  1. $\mathcal{F}$: easily invertible quadratic map
  2. $\mathcal{S}$, $\mathcal{T}$: invertible affine(or linear) maps
- $(\mathcal{F}, \mathcal{S}, \mathcal{T})$: secret key

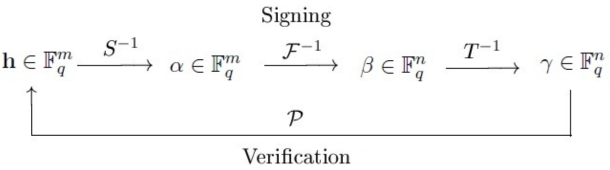

Figure: Structure of MQ Signature Scheme

MQ-PKC (2/5)

Signature Generation or Decryption

For given message M and hash value $\mathbf{h} = H(M)$, compute $\mathbf{s}$ with $\mathbf{h} = \mathcal{P}(\mathbf{s})$ so that $\mathbf{s} = (\mathcal{T}^{-1} \circ \mathcal{F}^{-1} \circ \mathcal{S}^{-1})(\mathbf{h})$. Then $\mathbf{s}$ is a signature for message M.

Signature Verification or Encryption

Compute $\mathcal{P}(\mathbf{s}) = \mathbf{h'}$. If $\mathbf{h} = \mathbf{h'}$ accept it. If not, reject it.

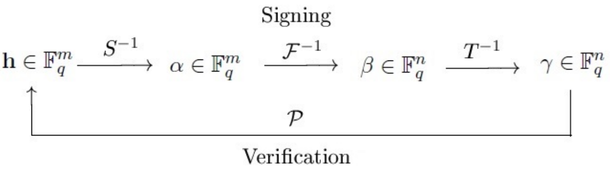

Figure: Structure of MQ Signature Scheme

MQ-PKC (3/5)

How to construct easily invertible $\mathcal{F}$: Single Field Type - OV map

- Notations
  - $x_1, \cdots, x_v$: Vinegar variables (v : the number of Vinegar variables)
  - $x_{v+1}, \cdots, x_{v+o}$: Oil variables (o : the number of Oil variables, the number of equations in $\mathcal{F}$)
  - $n = v + o$: the number of variables in $\mathcal{F}$
- Each component function $\mathcal{F}^{(k)}$ of $\mathcal{F} = (\mathcal{F}^{(1)}, \mathcal{F}^{(2)}, \cdots, \mathcal{F}^{(o)})$ is of the form

$$\mathcal{F}^{(k)}(x_1, \cdots, x_n) = \sum_{i=1}^{v} \sum_{j=i}^{v} \alpha_{ij}^{(k)} x_i x_j + \sum_{i=1}^{v} \sum_{j=v+1}^{n} \alpha_{ij}^{(k)} x_i x_j + \sum_{i=1}^{n} \beta_i^{(k)} x_i + \gamma^{(k)}$$

where $\alpha_{ij}^{(k)}, \beta_i^{(k)}, \gamma^{(k)} \in \mathbb{F}_q$.

MQ-PKC (4/5)

How to construct easily invertible $\mathcal{F}$: Single Field Type - OV map

- How to invert $\mathcal{F}$
  - Choose Vinegar variables at random and plug them into each $\mathcal{F}^{(k)}$

$$\sum_{i=1}^{v}\sum_{j=i}^{v}\alpha_{ij}^{(k)}x_i x_j + \sum_{i=1}^{v}\sum_{j=v+1}^{n}\alpha_{ij}^{(k)}x_i x_j + \sum_{i=1}^{v}\beta_i^{(k)}x_i + \sum_{i=v+1}^{n}\beta_i^{(k)}x_i + \gamma^{(k)}$$

    Then the red parts are converted to constants in above equation.
  - since there are no quadratic terms with oil variables in each $\mathcal{F}^{(k)}$ with $1 \leq k \leq o$, we obtain a linear equation with Oil variables $x_{v+1}, \cdots, x_n$ for each $\mathcal{F}^{(k)}$.
  - In other words, we obtain a linear equation system with o equations with o variables which can be easily solvable.
  - If this linear equation system is not solvable, choose Vinegar variables and try again.
  - Note that the obtained linear equation system would be solvable with very high probability.

[KPG99] A. Kipnis, J. Patarin, and L. Goubin, *Unbalanced Oil and Vinegar Signature Schemes*, Eurocrypt 1999, LNCS Vol 1592, pp.206-222, Springer.

MQ-PKC (5/5)

MQ signarure schemes using OV map

- UOV(Unbalanced Oil and Vinegar) - "unbalanced' means $v > o$.
- Rainbow - multi-layered UOV : a Finalist of NIST PQC standardization
- There are many other variants of UOV and Rainbow

An important remark about Rainbow

- Recently Beullens proposed a simple attack on Rainbow[Beu22] : an equivalent key of Rainbow with security level 1 can be recovered in 53 hours by a laptop.
- Because of this attack, Rainbow team announced that they replace the Rainbow level 1 parameters with their level 3 parameters and level 3 with level 5 parameters.

[Beu22] W. Beullens, *Breaking Rainbow Takes a Weekend on a Laptop*, IACR ePrint Archive 2022/214 (presented at Crypto 2022), 2022.

Implementations of UOV and Rainbow (1/4)

Our parameter selection

- UOV
  - Recently, Beullens proposed the intersection attack [Beu21] which is suitable when $v < 2o$. As a result, UOV parameters frequently used in the past, for example $(o, v) = (44, 59)$ for Security Level I, are vulnerable under the intersection attack.
  - So we suggested new UOV parameters considering the complexity of the intersection attack (see a table in next slide).
  - Very recently, the Rainbow team (collaborated with Beullens) suggested new UOV parameters after the simple attack had been proposed. Note that their parameters are slightly different from our parameters.
- Rainbow
  - Our implementation of Rainbow is based on source codes of Rainbow team which are submitted for NIST PQC Standardization Round 3. So our paremeters in our paper are same with NIST PQC Round 3.
  - In this presentation, we apply parameter changes of Rainbow due to the attacks proposed in [Beu22].

[Beu21] W. Beullens, *Improved Cryptanalysis of UOV and Rainbow*, Eurocrypt 2021, LNCS 12696, pp. 348-373, 2021.
[Beu22] W. Beullens, *Breaking Rainbow Takes a Weekend on a Laptop*, IACR ePrint Archive 2022/214 (presented at Crypto 2022), 2022.

Implementations of UOV and Rainbow (2/4)

Scheme	Security Category	I	III	V
UOV	λ (Gates)	146	212	274
	(o, v)	(46, 70)	(72, 109)	(96, 144)
	Direct Attack	144.05	212.05	274.847
	Intersection Attack	166.87	236.36	291.501
Rainbow	λ (Gates)	177	226	-
	(v, o_1, o_2)	(68, 32, 48)	(96, 36, 64)	-
	Direct Attack	234	285	-
	Intersection Attack	177	226	-

Table: Suggested Parameters of UOV/Rainbow at Three SCs.

Note that we select $q = 256$ for all our parameters above.

Implementations of UOV and Rainbow (3/4)

Detail of our implementations

- Similarly with Rainbow, we apply $\mathcal{T} = \begin{pmatrix} I & T' \\ 0 & I \end{pmatrix}$ for UOV.
- Intel(R) Core(TM) i9-10900X CPU running at the constant clock frequency of 3.70GHz.
- Each result is an average of 10,000 measurements for each function using the C programming language with GNU GCC version 10.1.0 compiler on Centos 7.9.2009. Hyperthreading and Turbo Boost are switched off.

Scheme	SC	I	III	V
UOV	KeyGen.	29,077,126	98,870,925	161,016,435
	Sign	201,834	707,959	1,486,775
	Verify	125,312	222,012	485,344
Rainbow	KeyGen.	65,099,975	214,977,689	—
	Sign	322,799	807,309	—
	Verify	151,466	395,259	—

Implementations of UOV and Rainbow (4/4)

Detail of our Implemtations on Signing process

Scheme	Layer	Operations	I	III	V
UOV	1	Vinegar Value Substitutions	58.09 %	48.37 %	56.60 %
		Computation of LS_V^{-1}	36.38 %	47.98 %	41.71 %
		Etc.	5.53 %	3.65 %	1.69 %
Rainbow	1	Vinegar Value Substitutions	18.58 %	34.37 %	—
		Computation of $LS_{V,1}^{-1}$	11.37 %	8.03 %	—
		Etc.	1.26 %	0.68 %	—
	2	Vinegar Value Substitutions	39.54 %	29.07 %	—
		Computation of $LS_{V,2}^{-1}$	25.38 %	25.34 %	—
		Etc.	3.88 %	2.51 %	—

Run-time of UOV/Rainbow signing is mostly dominated by the two operations.

- Substitution of Vinegar Values into the Central Polynomials
- Solving Obtained Linear System

Efficient Implementations of UOV and Rainbow

The most dominated part of signing of UOV and Rainbow

- **Substitution of Vinegar Values into the Central Polynomials** - computing the coefficient matrix LS_V and constant term of the linear system which we will obtain.
- **Solving Obtained Linear System** - we require to compute the inverse matrix LS_V^{-1} of the coefficient matrix obtained above.

Key idea of our efficient implementations of UOV and Rainbow

- Block Matrix Inversion - we replace an inversion of $m \times m$ matrix into two inversions of $m/2 \times m/2$ matrices when m is even.
- Precomputation - the above two parts can be precomputed, and then signing process can be significantly improved.

Block Matrix Inversion (1/6)

Theorem 1

Let $R = \begin{pmatrix} A & B \\ C & D \end{pmatrix}$ be a matrix partitioned into 2×2 blocks.

(i) Assume A is nonsingular. Then the matrix R is invertible if and only if the Schur complement $(D - CA^{-1}B)$ of A is invertible and

$$R^{-1} = \begin{pmatrix} A^{-1} + A^{-1}B(D - CA^{-1}B)^{-1}CA^{-1} & -A^{-1}B(D - CA^{-1}B)^{-1} \\ -(D - CA^{-1}B)^{-1}CA^{-1} & (D - CA^{-1}B)^{-1} \end{pmatrix}.$$

(ii) Assume D is nonsingular. Then the matrix R is invertible if and only if the Schur complement $(A - BD^{-1}C)$ is invertible and

$$R^{-1} = \begin{pmatrix} (A - BD^{-1}C)^{-1} & -(A - BD^{-1}C)^{-1}BD^{-1} \\ -D^{-1}C(A - BD^{-1}C)^{-1} & D^{-1} + D^{-1}C(A - BD^{-1}C)^{-1}BD^{-1} \end{pmatrix}$$

$\Rightarrow$ It requires two inversions and six matrix multiplications of the half-sized matrices.

Block Matrix Inversion (2/6)

Theorem 2

For a nonsingular $k \times k$ matrix R in the above, $R^{-1} \cdot \alpha$ requires two inversions, two matrix multiplications of the half-sized block matrices and four block matrix-vector products, where k is even and $\alpha = (\alpha_1, \cdots, \alpha_{k/2})^{\mathtt{T}}$.

Sketch of Proof. A nonsingular square matrix R of 2 $\times$ 2 blocks is represented by the LDU decomposition of block matrices based on the Schur complement as

$$R = \begin{pmatrix} A & B \\ C & D \end{pmatrix} = \begin{pmatrix} I & O \\ CA^{-1} & I \end{pmatrix} \begin{pmatrix} A & O \\ 0 & D - CA^{-1}B \end{pmatrix} \begin{pmatrix} I & A^{-1}B \\ 0 & I \end{pmatrix}$$
$$= L \cdot D_{Sc} \cdot U.$$

So, we have $R^{-1} = U^{-1} \cdot D_{Sc}^{-1} \cdot L^{-1}$.

Block Matrix Inversion (3/6)

Repeated BMI

- An inversion of $m \times m$ matrix can be replaced by 2 inversions of $m/2 \times m/2$ matrices. In a similar manner, each of 2 inversions of $m/2 \times m/2$ matrices can be replaced by 2 inversions of $m/4 \times m/4$ matrices if m is a multiple of 4.
- Like this, for $k = 2^l \cdot k'$, we can apply the BMI l times. We define the number of these iterations of the BMI as a depth. We cannot expect that l iterations will always be effective, because 2^l inversions of $k/2^l \times k/2^l$ matrices are required.

Block Matrix Inversion (4/6)

Matrix Size	GE	BMI (Depth 1)	BMI (Depth 2)
46	94,033	72,302	—
48	101,498	75,136	72,479
50	142,243	82,768	—
56	175,195	103,357	99,445
64	225,081	87,150	70,091
68	322,315	187,947	170,788
72	355,173	208,355	190,480
96	713,462	252,538	226,627
100	923,489	453,441	391,747

Table: Gaussian Elimination (GE) vs. Block Matrix Inversion Technique in CPU Cycles.

- The larger the size, the greater the performance improvement.
- Especially excellent improvements in the case of 64 and 96 are due to the fact that the multiples of 32 are optimal parameters which are suitable for the AVX2 vectorization.

Appling BMI to UOV and Rainbow Signing

- After obtaining LS_V from the Vinegar value substitution, we set

$$LS_V = \begin{pmatrix} A & B \\ C & D \end{pmatrix}$$

  and apply the BMI on LS_V.
- If A or $[D - CA^{-1}B]$ is not invertible then we choose another Vinegar values.
- Note that the probability that the matrices are invertible is $\left(1 - \frac{1}{q}\right)^2 \approx 99.22\%$.
- Details of our proposed algorithm applying BMI on the signing of Rainbow and UOV are given in Algorithm 7 and 8 in our paper, respectively.

Block Matrix Inversion (6/6)

- The below table describes our implementation results on our proposed BMI method in CPU Cycles.
- Compared to UOV implemented with Gaussian elimination, by using the BMI with the depth 1, we obtain speedups of 12.36%, 20.41%, and 32.42% at the three security categories, respectively.

Scheme	SC	I	III	V
UOV	G.E.	201,834	707,959	1,486,775
	BMI (Depth 1)	176,884	563,519	1,004,704
	BMI (Depth 2)	—	535,660	981,351
Rainbow	G.E.	322,799	807,309	—
	BMI (Depth 1)	270,731	650,400	—
	BMI (Depth 2)	271,986	639,965	—

Table: Implementation Results of BMI on the Intel at Three SCs, in CPU cycles.

Precomputation (1/3)

The general idea of using an offline/online phase was first introduced by Even, Goldreich, and Micali [EGM90].

Offline/Online Signing of UOV.

- Offline phase
  - After choosing random Vinegar values $s_V = (s_1, \cdots, s_v) \in \mathbb{F}_q^v$, substitute s_V into o equations $\mathcal{F}^{(k)}$ $(1 \le k \le o)$ to get the linear system LS_V of o equations and o unknowns and a constant vector $c_V = (c_1, \cdots, c_m)$.
  - Compute LS_V^{-1}. If LS_V is not invertible then go back to the first step.
  - Store $< s_V, c_V, LS_V^{-1} >$ as the precomputed values.
- Online phase
  - Choose a random salt r and compute $h = \mathcal{H}(\mathcal{H}(\mathtt{m})||r)$ for a message $\mathtt{m}$.
  - From $< s_V = (s_1, \cdots, s_V)$, $c_V = (c_1, \cdots, c_m)$, $LS_V^{-1} >$, compute $LS_V^{-1} \cdot h_V^{\mathtt{T}} = \alpha$, where $h_V = (h_1 - c_1, \cdots, h_m - c_m)$ and $h = (h_1, \cdots, h_m)$.
  - Compute $T^{-1} \cdot (S_V, \alpha)^{\mathtt{T}} = \sigma$ and output $\tau = (\sigma, r)$ as a signature on $\mathtt{m}$.

[EGM90] S. Even, O. Goldreich, and S. Micali, *On-line/off-line digital schemes*, Crypto '89, LNCS 435, pp. 263-275, 1990.

Precomputation (2/3)

Offline/Online Signing of Rainbow

- The offline phase of the first layer of Rainbow is similar with UOV.
- But in the second layer precomputation is limited - Some Vinegar variables $x_{v+1}, \cdots, x_{v+o_1}$ of the second layer are detemined depending on the (hashed) message h.
- So precomputable values are :
  - $LS_{V,1}^{-1}$ - the inverse of the coefficient matrix $LS_{V,1}$ of the linear system obtained in the first layer
  - $C_{V,1}$ - a vector of constant terms in $\left(\mathbb{F}^{(1)(s_V)}, \cdots, \mathbb{F}^{(o_1)(s_V)}\right)$
  - $\left(\mathbb{F}^{(o_1+1)}(s_V), \cdots, \mathbb{F}^{(o_1+o_2)}(s_V)\right)$ - linear terms and constant terms when s_V is substituted into central polynomials in second layer

Precomputation (3/3)

Scheme	Security Category	Unit	I	III	V
UOV	Sign w/o Precomp.	cycle	201 834	707 959	1 486 775
	Precomp. (offline)	cycle	189 224	690 586	1 460 168
	Sign w/ Precomp. (online)		11 788	19 439	23 133
	Total (offline + online)		201 012	710 025	1 483 301
	Precomp. Memory Cost per Sig.	byte	2 256	5 402	9 504
Rainbow	Sign w/o Precomp.	cycle	68 203	322 799	807 309
	Precomp. (offline)	cycle	37 212	173 204	508 890
	Sign w/ Precomp. (online)		31 973	142 179	278 511
	Total (offline + online)		69 185	315 383	787 401
	Precomp. Memory Cost per Sig.	byte	2 152	4 792	5 648

Resilence against Leakage or Reuse of Precomputed Values

Store $< s_V, c_V, LS_V^{-1} >$ **Securely.** The precomputed values $< s_V, c_V, LS_V^{-1} >$ should be stored securely.

Theorem 3

If $(n+1)$ tuples $< \mathtt{m}^{(i)}, \tau^{(i)}, s_V^{(i)},\ c_V^{(i)}, LS_V^{(i)-1} >$ are given such that the $n \times n$ matrix $(\sigma^{(1)\mathtt{T}}\ \sigma^{(2)\mathtt{T}}\ \cdots\ \sigma^{(n)\mathtt{T}})$ is invertible then the secret key of UOV is completely recovered in polynomial-time.

Theorem 4

If $(n+1)$ tuples $< \mathtt{m}^{(i)}, \tau^{(i)}, s_V^{(i)},\ c_V^{(i)}, (LS_V^{(i)})^{-1} >$ are given such that the $n \times n$ matrix $(\sigma^{(1)\mathtt{T}}\ \sigma^{(2)\mathtt{T}}\ \cdots\ \sigma^{(n)\mathtt{T}})$ is invertible then an equivalent key of Rainbow is completely recovered in polynomial-time.

Resilence against Leakage or Reuse of Precomputed Values

Do not Reuse $< s_V, c_V, LS_V^{-1} >$. The precomputed value $< s_V, c_V, LS_V^{-1} >$ should not be reused in signing.

Theorem 5 [SK20]

If $(m+1)$ signatures generated by the reused Vinegar values are given then

- the equivalent key of UOV is completely recovered in polynomial time,
- the complexity of the KRAs using good keys on Rainbow is determined by solving a multivariate system of m quadratic equations with o_1 variables.

Theorem 6

If (o_2+1) signatures generated by reusing the precomputed values then an equivalent key of Rainbow is recovered in polynomial-time with high probability.

[SK20] K. -A. Shim, and N. Koo, *Algebraic Fault Analysis of UOV and Rainbow with the Leakage of Random Vinegar Values*, IEEE Transactions on Information Forensics and Security, Vol. 15, pp.2429-2439, 2020.

Conclusion

We presented two efficient implementation methods to improving signing of UOV and Rainbow, and gave implementation results on our methods.

- The Block Matrix Inversion - improves the process to solve the linear system
  - 10-40% faster than Gaussian elimination

- Precomputation - improves the process substituting Vinegar values and solving the linear system
  - For UOV, about 17, 36, and 64 times improvements for Security Level 1, 3, and 5, respectively.

This is the end of my presentation

Thank you for your attention.