Improved Cryptanalysis of SNOVA

Ward Beullens

IBM Research - Europe, Zürich, Switzerland

Abstract. SNOVA is a multivariate signature scheme submitted to the NIST project for additional signature schemes by Cho, Ding, Kuan, Li, Tseng, Tseng, and Wang. With small key and signature sizes and good performance, SNOVA is one of the more efficient schemes in the competition, which makes SNOVA an important target for cryptanalysis.

In this paper, we observe that SNOVA implicitly uses a structured version of the "whipping" technique developed for the MAYO signature scheme. We show that the extra structure makes the construction vulnerable to new forgery attacks. Concretely, we formulate new attacks that reduce the security margin of the proposed SNOVA parameter sets by a factor between 2^8 and 2^{39}. Furthermore, we show that large fractions of public keys are vulnerable to more efficient versions of our attack. For example, for SNOVA-37-17-2, a parameter set targeting NIST's first security level, we show that roughly one out of every 500 public keys is vulnerable to a universal forgery attack with bit complexity 2^{97}, and roughly one out of every 143000 public keys is even breakable in practice within a few minutes.

1 Introduction

Cryptographic signature schemes are crucial for ensuring the authenticity and integrity of digital communications. Due to the rise of quantum computing, which threatens the security of cryptographic systems based on the hardness of factoring and solving discrete logarithms, the search for efficient quantum-safe signature schemes has become increasingly important. The U.S. National Institute for Standards and Technology (NIST) has standardized quantum-safe digital signature schemes based on the hardness of computational problems involving lattices, and based on the security of cryptographic hash functions. However, in 2022, NIST initiated a new project to standardize additional post-quantum signature schemes, to diversify the set of digital signature schemes further. Many of the signature schemes submitted to the new standardization project are based on the hardness of finding solutions to a system of multivariate quadratic equations. Most of these signature schemes are variants of the Oil and Vinegar signature scheme, introduced in 1997 by Patarin [10]. Oil and Vinegar is a signature scheme whose security is relatively well understood. It has small signature sizes and good signing and verification performance. The main drawback is that it has relatively large public keys, for example, 43 KB public keys for NIST security level I parameters [2]. To solve this problem, structured variants of the Oil and Vinegar signature scheme have been proposed that significantly reduce the public key size, at the cost of new security assumptions. One of these digital signature schemes is called SNOVA.

SNOVA has small key and signature sizes coupled with good performance, making it a seemingly promising candidate for standardization. However, the scheme uses a new, ad-hoc design without a security proof that reduces some well-established computational problem to breaking the security of SNOVA. Therefore, more cryptanalysis is needed to increase the confidence in the security of SNOVA.

Related Work. The security analysis in the original SNOVA paper was quite limited and widely overestimated the security of SNOVA against key recovery attacks. Two papers with an improved analysis of key-recovery attacks against SNOVA appeared on ePrint in January 2024. The first paper, authored by Ikematsu and Akiyama [7], observes that a SNOVA public key with parameters (q, n, m, l) contains the structure of an Oil and Vinegar public key with parameters (q, ln, lm). They show that known attacks against Oil and Vinegar therefore apply to SNOVA, and some attacks can be made even more efficient because (unlike in the case of Oil and Vinegar) the bilinear forms in a SNOVA public key are not symmetric. This shows that some of the parameter sets in the SNOVA submission do not meet the claimed NIST security levels. The second paper, authored by Li and Ding [8], appeared on ePrint three days later and has very similar observations and results. In response to this new cryptanalysis, the SNOVA team updated some of their parameter sets, so that the proposed parameter sets meet the required security levels again.

Contributions. In this work, we observe that SNOVA has some similarities with the "whipping" technique of MAYO. The idea behind this is to "whip up" a public Oil-and-Vinegar multivariate quadratic map $\mathcal{P} : \mathbb{F}_q^n \to \mathbb{F}_q^m$ into a map $\mathcal{P}^* : \mathbb{F}_q^{kn} \to \mathbb{F}_q^m$ with a factor k more variables. The base map vanishes on a secret vector space O of dimension o, and the whipped map $\mathcal{P}^*$ is constructed in such a way that it vanishes on $O^k = \{(\mathbf{o}_1, \ldots, \mathbf{o}_k) \mid \mathbf{o}_1, \ldots, \mathbf{o}_k \in O\}$ which is a vector space of dimension ko. In MAYO, the map $\mathcal{P}^*$ is defined as

$$\mathcal{P}^*(\mathbf{x}_1, \ldots, \mathbf{x}_k) := \sum_{i=1}^{k} \mathbf{E}_i \mathcal{P}(\mathbf{x}_i) + \sum_{i=1}^{k} \sum_{i=1}^{k} \mathbf{E}_{i,j} \mathcal{P}'(\mathbf{x}_i, \mathbf{x}_j) \,,$$

where, for security-related reasons, the "emulsifier" matrices $\mathbf{E}_i, \mathbf{E}_{i,j} \in \mathbb{F}_q^{m \times m}$ are chosen in such a way that all non-trivial linear combinations of them have full rank. We observe that a SNOVA public map has the same structure, except that in this case the matrices $\mathbf{E}_i, \mathbf{E}_{i,j} \in \mathbb{F}_q^{m \times m}$ can have linear combinations with rank deficiencies, which leads to more efficient forgery attacks. Moreover, it turns out that in SNOVA the $\mathbf{E}_i, \mathbf{E}_{i,j}$ matrices are block diagonal matrices with ℓ identical blocks on the diagonal. Compared to random matrices, matrices of this form are much more likely to have large rank deficiencies, which makes the forgery attack even more efficient.

We develop this attack against SNOVA, and we estimate its cost, showing that it significantly reduces the security margin of SNOVA (see Table 3). Moreover, we show that large fractions of public keys are vulnerable to more efficient versions of our attack, which further reduces the security of SNOVA in a setting where an attacker is interested in forging messages for one out of a sufficiently large set of public keys. We implement the most efficient version of our weak-key attack, which is the case of the SNOVA-37-17-2 parameter set. For this parameter set we demonstrate that for roughly one out of every 143000 honestly generated public keys, it is possible to forge signatures for arbitrary messages within only a few minutes using modest computational resources.[1] The source code for our experiments is publicly available at the following link.

`https://github.com/WardBeullens/BreakingSNOVA`

2 Preliminaries

2.1 Notation.

Let $\mathbb{F}_q$ be the finite field of order q. We denote vectors over $\mathbb{F}_q$ by bold lowercase letters, e.g., $\mathbf{x}$, and matrices over $\mathbb{F}_q$ by bold uppercase letters, e.g., $\mathbf{M}$. For a matrix $\mathbf{M} \in \mathbb{F}_q^{\ell\times\ell}$ and a positive integer n, we denote by $\mathbf{M}^{\otimes n} \in \mathbb{F}_q^{n\ell\times n\ell}$ the Kronecker product of the identity matrix of order n with $\mathbf{M}$, i.e., $\mathbf{M}^{\otimes n} = \mathbf{1}_n \otimes \mathbf{M}$ is the block diagonal matrix with n copies of $\mathbf{M}$ on the block diagonal.

2.2 Multivariate quadratic maps and their polar forms.

We say a sequence of m multivariate quadratic polynomials in n variables $\mathcal{P} = (p_1, \dots, p_m)$ is a multivariate quadratic map, and we identify it with the function $\mathcal{P} : \mathbb{F}_q^n \to \mathbb{F}_q^m : \mathcal{P}(\mathbf{x}) := (p_1(\mathbf{x}), \dots, p_m(\mathbf{x}))$ given by evaluation of the component polynomials.

For any multivariate quadratic polynomial $p(\mathbf{x})$ we define the corresponding *polar form* as $p'(\mathbf{x}, \mathbf{y}) := p(\mathbf{x} + \mathbf{y}) - p(\mathbf{x}) - p(\mathbf{y}) + p(\mathbf{0})$, which is a symmetric

[1] The author's laptop.

bilinear form. We denote by $\mathcal{P}'(\mathbf{x},\mathbf{y}) : \mathbb{F}_q^n \times \mathbb{F}_q^n \to \mathbb{F}_q^m$ the function of evaluating the polar forms associated to $\mathcal{P}$, which clearly satisfies $\mathcal{P}'(\mathbf{x},\mathbf{y}) = \mathcal{P}(\mathbf{x}+\mathbf{y}) - \mathcal{P}(\mathbf{x}) - \mathcal{P}(\mathbf{y}) + \mathcal{P}(\mathbf{0})$.

2.3 Oil and Vinegar trapdoors.

The Oil and Vinegar digital signature scheme relies on a simple but powerful procedure for sampling preimages for a multivariate quadratic map $\mathcal{P} : \mathbb{F}_q^n \to \mathbb{F}_q^m$, given knowledge of a linear subspace $O \subset \mathbb{F}_q^n$ of dimension at least m on which $\mathcal{P}$ evaluates to zero. Given a target $\mathbf{t} \in \mathbb{F}_q^m$, the procedure first samples a vector $\mathbf{v} \in \mathbb{F}_q^n$ uniformly at random, and then solves for $\mathbf{o} \in O$ such that $\mathcal{P}(\mathbf{v}+\mathbf{o}) = \mathbf{t}$. The latter can be done efficiently because it reduces to solving a system of linear equations in $\mathbf{o}$ by using the polar form:

$$\mathcal{P}(\mathbf{v}+\mathbf{o}) = \underbrace{\mathcal{P}'(\mathbf{o},\mathbf{v})}_{\text{linear in } \mathbf{o}} + \underbrace{\mathcal{P}(\mathbf{o})}_{=\mathbf{0}} + \underbrace{\mathcal{P}(\mathbf{v}) - \mathcal{P}(\mathbf{0})}_{\text{fixed by choice of } \mathbf{v}} = \mathbf{t}\,. \tag{1}$$

For properly chosen parameters (m sufficiently large, and n sufficiently larger than $2m$), it is assumed hard to sample preimages for $\mathcal{P}$ without knowledge of O, so this gives a cryptographic trapdoor function.

2.4 The MAYO trapdoor.

The Oil and Vinegar signature scheme uses an almost uniformly random subspace O [2], and a map $\mathcal{P}$ which is chosen uniformly at random among the maps vanishing on O. Representing such maps requires $O(m^3)$ coefficients, which unfortunately means that the Oil and Vinegar signature scheme has large public key sizes. To solve this problem, more structured families of multivariate quadratic maps have been proposed, which still vanish on large hidden subspaces, but which require fewer than $O(m^3)$ coefficients to represent, reducing the public key size dramatically. Examples from the NIST standardization project for additional signatures include MAYO, QR-UOV, and SNOVA [1,4,14].

We briefly explain the MAYO trapdoor, since it will be relevant for the analysis of SNOVA. MAYO starts from a "base" multivariate quadratic map $\mathcal{P} : \mathbb{F}_q^n \to \mathbb{F}_q^m$ that vanishes on a subspace O. The only difference with an Oil and Vinegar public key is that in the Oil and Vinegar signature scheme we have $\dim(O) = m$,

[2] In typical implementations, the subspace O is the row span of a matrix $(\mathbf{1}_m\ \mathbf{O})$, where the submatrix $\mathbf{O} \in \mathbb{F}_q^{m\times(n-m)}$ is chosen uniformly at random.

whereas in the case of MAYO the dimension of O is smaller. This means that the procedure from section 2.3 no longer works, since the linear system (1) has more equations than variables, and is thus unlikely to have solutions. To increase the size of the subspace, MAYO "whips up" the base map $\mathcal{P}$ into a larger map $\mathcal{P}^* : \mathbb{F}_q^{kn} \to \mathbb{F}_q^m$ that has a factor k more variables, where k is a parameter of the scheme. This is done by letting the polynomials of $\mathcal{P}^*(\mathbf{x}_1, \ldots, \mathbf{x}_k)$ be linear combinations of $\mathcal{P}(\mathbf{x}_i)$ and $\mathcal{P}'(\mathbf{x}_i, \mathbf{x}_j)$ for $i, j \in \{1, \ldots, k\}$, where each $\mathbf{x}_i$ is a sequence of n variables. More concretely, this means we have

$$\mathcal{P}^*(\mathbf{x}_1, \ldots, \mathbf{x}_k) := \sum_{i=1}^{k} \mathbf{E}_i \mathcal{P}(\mathbf{x}_i) + \sum_{i=1}^{k} \sum_{j=i+1}^{k} \mathbf{E}_{i,j} \mathcal{P}'(\mathbf{x}_i, \mathbf{x}_j)\,,$$

where $\mathbf{E}_i$ and $\mathbf{E}_{i,j}$ are fixed m-by-m matrices whose rows determine what linear combination of $\mathcal{P}(\mathbf{x}_i)$ and $\mathcal{P}'(\mathbf{x}_i, \mathbf{x}_j)$ each polynomial of $\mathcal{P}^*$ consists of.

Since $\mathcal{P}$ vanishes on the subspace O, it follows that $\mathcal{P}^*$ vanishes on $O^k = \{\,(\mathbf{o}_1, \ldots, \mathbf{o}_k) \mid \mathbf{o}_1, \ldots, \mathbf{o}_k \in O\}$, which is a subspace of $\mathbb{F}_q^{nk}$ of dimension $k \dim(O)$. The parameters are chosen such that $k \dim(O) \geq m$, so that the Oil and Vinegar procedure from section 2.3 can be used to efficiently sample preimages for $\mathcal{P}^*$.

The matrices $\mathbf{E}_i, \mathbf{E}_{i,j}$ are chosen in such a way that all non-trivial linear combinations of them have full rank [3], since linear combinations with low rank would lead to more efficient forgery attacks [1]. We will see that SNOVA can be interpreted as a variant of MAYO, where the matrices $\mathbf{E}_i, \mathbf{E}_{i,j}$ and the base quadratic map $\mathcal{P}$ have additional structure. It is an open question if the additional structure of $\mathcal{P}$ can lead to more efficient key-recovery attacks, but it is clear that linear combinations of the $\mathbf{E}_i$ and $\mathbf{E}_{i,j}$ matrices can have low rank, which leads to forgery attacks.

3 SNOVA

SNOVA is a multivariate digital signature scheme based on Oil and Vinegar submitted to the NIST standardization process of additional post-quantum signatures by Wang, Chou, Ding, Kuan, Li, Tseng, Tseng, and Wang [14]. The description of the SNOVA cryptosystem is rather complicated. To keep this paper understandable, we describe SNOVA only to the level of detail that is necessary to describe our attack. For more details on SNOVA, we refer to the submission document [13].

[3] To ensure that non-trivial linear combinations have full rank these matrices are chosen to represent multiplication by $\mathbb{F}_q$-linearly independent elements of an extension field $\mathbb{F}_{q^m}$.

3.1 SNOVA public keys

A public key for the SNOVA cryptosystem is the description of a structured multivariate quadratic map $\mathcal{P}$. The structure of the map is described in terms of the ring $\mathcal{R} = \mathrm{Mat}_{\ell\times\ell}(\mathbb{F}_q)$ of ℓ by ℓ matrices. The SNOVA scheme is parameterized by the following parameters:

- q the size of the finite field,
- ℓ the size of the matrix ring $\mathcal{R} = \mathrm{Mat}_{\ell\times\ell}(\mathbb{F}_q)$ used by the scheme,
- m the number of matrix equations in a public map,
- n the number of matrix variables in a public map,
- $\mathbf{S}$ a symmetric ℓ by ℓ matrix with an irreducible characteristic polynomial.

The latest parameter sets proposed by the SNOVA designers are given in table 1. Note that some of the parameter sets are different from the parameters in the NIST submission in response to the attacks of [7,8].

Security level	(q, ℓ, m, n)	Signature size	Public key size	best attack
I	(16,2,17,54)	124 B	9.6 KB	151
	(16,3,8,33)	164.5 B	2.3 KB	159
	(16,4,5,29)	248 B	1.0 KB	175
III	(16,2,25,81)	178 B	31 KB	215
	(16,3,11,60)	286 B	5.8 KB	213
	(16,4,8,45)	376 B	4.0 KB	271
V	(16,2,33,108)	232 B	70 KB	279
	(16,3,15,81)	380.5 B	15 KB	285
	(16,4,10,70)	576 B	7.8 KB	335

Table 1. The parameter sets proposed by the SNOVA designers in [14]. The last column gives an estimate of the bit-cost ($\log_2$ of the number of bit operations) of the most efficient attack against the parameter set.

A SNOVA public key contains the following information:

- Two sequences of ℓ^2 invertible matrices $\mathbf{A}_1, \dots, \mathbf{A}_{\ell^2}, \mathbf{B}_1, \dots, \mathbf{B}_{\ell^2}$ of size $\ell \times \ell$, chosen uniformly at random during key generation.

- Two sequences of ℓ^2 invertible matrices $\mathbf{Q}_{1,1}, \ldots, \mathbf{Q}_{\ell^2,1}, \mathbf{Q}_{1,2}, \ldots, \mathbf{Q}_{\ell^2,2}$ of size $\ell \times \ell$, chosen uniformly from $\mathbb{F}_q[\mathbf{S}] \setminus \{0\}$, where $\mathbb{F}_q[\mathbf{S}]$ is the ring generated by $\mathbf{S}$. [4] [5]
- A set of m matrices $\mathbf{P}_1, \cdots, \mathbf{P}_m \in \mathbb{F}_q^{n\ell \times n\ell}$, generated in some way which is not relevant for us.

Together, these matrices describe a multivariate quadratic map $\mathcal{P} : \mathbb{F}_q^{n\ell^2} \to \mathbb{F}_q^{m\ell^2}$. The $n\ell^2$ variables are grouped in a matrix $\mathbf{U}$ of height $n\ell$ and width ℓ, each entry being a separate variable for $n\ell^2$ variables in total. The polynomials of $\mathcal{P}$ are then the entries of

$$p_i(\mathbf{U}) := \sum_{\alpha=1}^{\ell^2} \mathbf{A}_\alpha \cdot \mathbf{U}^T \cdot \mathbf{Q}_{\alpha,1}^{\otimes n} \cdot \mathbf{P}_i \cdot \mathbf{Q}_{\alpha,2}^{\otimes n} \cdot \mathbf{U} \cdot \mathbf{B}_\alpha \tag{2}$$

for $i \in \{1, \ldots, m\}$, where $\mathbf{Q}_{\alpha,b}^{\otimes n}$ denotes the $n\ell$ by $n\ell$ block matrix with n copies of $\mathbf{Q}_{\alpha,b}$ on the block diagonal and zeros elsewhere. Each of the ℓ^2 entries of $p_i(\mathbf{U})$ is a homogeneous quadratic polynomial in the entries of $\mathbf{U}$, for a total of $m\ell^2$ quadratic polynomials in total.

3.2 SNOVA signing and signature verification

A SNOVA signature consists of a vector $\mathbf{s} \in \mathbb{F}_q^{n\ell^2}$ and a random bit string salt. The signature $\sigma = (\mathbf{s}, \mathsf{salt})$ is valid for a message $M \in \{0,1\}^*$ if $\mathcal{P}(\mathbf{s}) = \mathcal{H}(M||\mathsf{salt})$, where $\mathcal{P} : \mathbb{F}_q^{n\ell^2} \to \mathbb{F}_q^{m\ell^2}$ is the map described by the SNOVA public key as explained in the previous section, and where $\mathcal{H} : \{0,1\}^* \to \mathbb{F}_q^{m\ell^2}$ is a cryptographic hash function that outputs elements of $\mathbb{F}_q^{m\ell^2}$. It turns out that $\mathcal{P}$ vanishes on a subspace of dimension $m\ell^2$, known only to the entity that generated the public key. Therefore, the signer can generate signatures for $\mathcal{P}$ efficiently, using the Oil and Vinegar procedure explained in section 2.3.

Clearly, if an attacker can efficiently find $\mathbf{s}$ such that $\mathcal{P}(\mathbf{s}) = \mathbf{t}$ for arbitrary $\mathbf{t} \in \mathbb{F}_q^{m\ell^2}$, then SNOVA would be vulnerable to a universal forgery attack. This is because the attacker can sample a random bit string salt, compute $\mathbf{s}$ such that $\mathcal{P}(\mathbf{s}) = \mathcal{H}(M||\mathsf{salt})$, and output the forgery $\sigma = (\mathbf{s}, \mathsf{salt})$.

---

[4] Note that, since the characteristic polynomial of $\mathbf{S}$ is irreducible, the ring $\mathbb{F}_q[\mathbf{S}]$ is a finite field. We have $\mathbb{F}_q[x]/(p_{\mathbf{S}}(x)) \simeq \mathbb{F}_q[\mathbf{S}]$, where $p_{\mathbf{S}}(x)$ is the characteristic polynomial of $\mathbf{S}$.

[5] Since the matrices $\mathbf{A}_i, \mathbf{B}_i, \mathbf{Q}_{ib}$ are sampled at random, they can be generated from a short seed, so they can be represented very compactly. The public key size is therefore dominated by the representation of the $\mathbf{P}_i$ matrices.

4 SNOVA has the whipping structure of MAYO.

Given the symmetric matrix $\mathbf{S}$, we can associate to each of the matrices $\mathbf{P}_1, \ldots, \mathbf{P}_m$ in a SNOVA public key ℓ^2 bilinear forms

$$B_i^{(a,b)} : \mathbb{F}_q^{n\ell} \times \mathbb{F}_q^{n\ell} \to \mathbb{F}_q : B_i^{(a,b)}(\mathbf{x}, \mathbf{y}) := \mathbf{x}^t (\mathbf{S}^a)^{\otimes n} \mathbf{P}_i (\mathbf{S}^b)^{\otimes n} \mathbf{y}\,,$$

for $(a,b) \in \{0, \ldots, \ell-1\}^2$. In other words, $B_i^{(a,b)}(\mathbf{x}, \mathbf{y})$ is the usual bilinear form associated to $\mathbf{P}_i$, after multiplying each ℓ-block of the inputs $\mathbf{x}$ and $\mathbf{y}$ by $\mathbf{S}^a$ and $\mathbf{S}^b$ respectively. Let $\mathbf{u}_0, \ldots, \mathbf{u}_{\ell-1}$ be the columns of the matrix of variables $\mathbf{U}$. The reason for introducing the bilinear forms $B_i^{(a,b)}$ is the following observation:

Lemma 1. *The entries of $p_i(\mathbf{U})$ are linear combinations of $B_i^{(a,b)}(\mathbf{u}_j, \mathbf{u}_k)$ for $a, b, j, k \in \{0, \ldots, \ell-1\}$. More precicely, there is a linear map $T : \mathbb{F}_q^{\ell^4} \to \mathbb{F}_q^{\ell^2}$, determined by $\{\mathbf{A}_\alpha, \mathbf{B}_\alpha, \mathbf{Q}_{\alpha,1}, \mathbf{Q}_{\alpha,2}\}_{\alpha \in \{1,\ldots,\ell^2\}}$, such that for all $1 \leq i \leq m$*

$$p_i(\mathbf{U}) = T\left(\{B_i^{(a,b)}(\mathbf{u}_j, \mathbf{u}_k)\}_{a,b,j,k \in \{0,\ldots,\ell-1\}}\right)\,.$$

Proof. The $\mathbf{Q}_{\alpha,1}$ and $\mathbf{Q}_{\alpha,2}$ matrices are drawn from $\mathbb{F}_q[\mathbf{S}]$, so by construction they are linear combinations of powers of $\mathbf{S}$, meaning that the (j,k)-th entry of $\mathbf{U}^t \mathbf{Q}_{\alpha,1}^{\otimes n} \cdot \mathbf{P}_i \cdot \mathbf{Q}_{\alpha,2}^{\otimes n} \mathbf{U}$ is a linear combination of the $B_i^{(a,b)}(\mathbf{u}_j, \mathbf{u}_k)$ for $j, k \in \{0, \ldots, \ell-1\}$. Multiplying this matrix from the left by $\mathbf{A}_\alpha$ and from the right by $\mathbf{B}_\alpha$ and summing over all α from 1 to ℓ^2 is a linear operation, so it follows that the entries of $p_i(\mathbf{U})$ are indeed linear combinations of $B_i^{(a,b)}(\mathbf{u}_j, \mathbf{u}_k)$, as claimed. □

It follows that $\mathcal{P}$ has a structure very similar to that of a MAYO public map.

Corollary 1. *Let $\mathcal{B} : \mathbb{F}_q^{n\ell} \times \mathbb{F}_q^{n\ell} \to \mathbb{F}_q^{m\ell^2}$ be the bilinear map defined as $\mathcal{B}_{i,a,b}(\mathbf{x}, \mathbf{y}) := B_i^{(a,b)}(\mathbf{x}, \mathbf{y})$. There exist matrices $\mathbf{E}_{j,k} \in \mathbb{F}_q^{m\ell^2 \times m\ell^2}$ for j, k from 0 to $\ell-1$, whose entries only depend on the matrices $\{\mathbf{A}_\alpha, \mathbf{B}_\alpha, \mathbf{Q}_{\alpha,1}, \mathbf{Q}_{\alpha,2}\}_{\alpha \in \{1,\ldots,\ell^2\}}$ in the public key, such that the SNOVA public map can be written as*

$$\mathcal{P}(\mathbf{U}) = \sum_{j=0}^{\ell-1} \sum_{k=0}^{\ell-1} \mathbf{E}_{j,k} \cdot \mathcal{B}(\mathbf{u}_j, \mathbf{u}_k)\,. \tag{3}$$

Moreover, the matrices $\mathbf{E}_{j,k}$ have a block diagonal structure, with m idential blocks on the diagonal, i.e., $\mathbf{E}_{j,k} = \tilde{\mathbf{E}}_{j,k}^{\otimes m}$ for some matrix $\tilde{\mathbf{E}}_{j,k} \in \mathbb{F}_q^{\ell^2 \times \ell^2}$.

Proof. Lemma 1 says that each coefficient of $p_i(\mathbf{U})$ is a linear combination of $B_i^{(a,b)}(\mathbf{u}_j, \mathbf{u}_k)$, and the coefficients of the linear combination are the same for all $i \in \{1, \dots, m\}$. By collecting the coefficients of the linear combinations in the rows of the matrices $\tilde{\mathbf{E}}_{j,k}$ we obtain equation (3) with $\mathbf{E}_{j,k} = \tilde{\mathbf{E}}_{j,k}^{\otimes m}$. □

This is very similar to the "whipping" structure of a MAYO public map, with the difference that in the case of MAYO the role of the base map is symmetric, so the terms involving $\mathcal{B}(\mathbf{u}_i, \mathbf{u}_j)$ and $\mathcal{B}(\mathbf{u}_j, \mathbf{u}_i)$ can be combined, whereas in SNOVA they need to be kept separate because the $\mathbf{P}_i$ matrices in the SNOVA public key are not symmetric. Recall that, to prevent forgery attacks, in MAYO the $\mathbf{E}_{i,j}$ matrices were chosen so that non-trivial linear combinations of them have full rank. This is not the case in SNOVA, so we expect this to cause problems. The fact that $\mathbf{E}_{i,j} = \tilde{\mathbf{E}}_{i,j}^{\otimes m}$ even amplifies the problem, because if there is a linear combination of the $\tilde{\mathbf{E}}_{i,j}$ matrices with a rank defect d, then the corresponding linear combination of the $\mathbf{E}_{i,j}$ has rank defect dm. In the next subsections, we will see that this allows for practical forgery attacks in some cases.

5 Our forgery attack against SNOVA

The basic idea behind the attack was already introduced by Beullens [1, Section 5], who realized that it is possible to more efficiently sample preimages for MAYO-like maps (i.e. multivariate quadratic maps of the form (3)) if one can find certain linear combinations of the $\mathbf{E}_{j,k}$ matrices with low rank. This was taken into account in the design of MAYO, which ensures that such linear combinations do not exist.

Basic attack. The idea is for the attacker to first choose random vectors $\mathbf{v}_1, \dots, \mathbf{v}_{\ell-1} \in \mathbb{F}_q^{n\ell}$ and then solve for a solution $\mathbf{U} = (\mathbf{u}_0, \dots, \mathbf{u}_{\ell-1})$ satisfying $\mathbf{u}_i = \alpha_i \mathbf{u}_0 + \mathbf{v}_i$ for all $i \in \{1, \dots, \ell-1\}$, for some scalars $\alpha_1, \dots \alpha_{\ell-1} \in \mathbb{F}_q$. The motivation for doing this is that we have

$$\begin{aligned}\mathcal{P}(\mathbf{U}) &= \sum_{j=0}^{\ell-1}\sum_{k=0}^{\ell-1} \mathbf{E}_{j,k} \cdot \mathcal{B}(\mathbf{u}_j, \mathbf{u}_k) = \sum_{j=0}^{\ell-1}\sum_{k=0}^{\ell-1} \mathbf{E}_{j,k} \cdot \mathcal{B}(\alpha_j \mathbf{u}_0 + \mathbf{v}_j, \alpha_k \mathbf{u}_0 + \mathbf{v}_k) \\ &= \sum_{j=0}^{\ell-1}\sum_{k=0}^{\ell-1} \mathbf{E}_{j,k} \cdot \left[\underbrace{\mathcal{B}(\alpha_j \mathbf{u}_0, \alpha_k \mathbf{u}_0)}_{\text{quadratic in } \mathbf{u}_0} + \underbrace{\mathcal{B}(\mathbf{v}_j, \alpha_k \mathbf{u}_0) + \mathcal{B}(\alpha_j \mathbf{u}_0, \mathbf{v}_k)}_{\text{linear in } \mathbf{u}_0} + \underbrace{\mathcal{B}(\mathbf{v}_j, \mathbf{v}_k)}_{\text{constant}} \right] .\end{aligned}$$

We see that the quadratic part of $\mathcal{P}(\mathbf{U})$ is just $\mathbf{E}_{\boldsymbol{\alpha}} \cdot \mathcal{B}(\mathbf{u}_0, \mathbf{u}_0)$, where

$$\mathbf{E}_{\boldsymbol{\alpha}} = \sum_{j=0}^{\ell-1}\sum_{k=0}^{\ell-1} \alpha_j \alpha_k \mathbf{E}_{j,k} .$$

Given a target $\mathbf{t} \in \mathbb{F}_q^{m\ell^2}$, one can find $\mathbf{U} \in \mathbb{F}_q^{n\ell^2}$ such that $\mathcal{P}(\mathbf{U}) = \mathbf{t}$ with the following two-step prodecure:

1. Find $\alpha_1, \ldots, \alpha_{\ell-1} \in \mathbb{F}_q$, such that $\mathbf{E}_{\boldsymbol{\alpha}}$ has small rank mr, i.e., $\mathbf{E}_{\boldsymbol{\alpha}} = \tilde{\mathbf{E}}_{\boldsymbol{\alpha}}^{\otimes m}$ with $\tilde{\mathbf{E}}_{\boldsymbol{\alpha}}$ of rank r. Finding these α_i is a generalization of the MinRank problem, where given an ℓ^2 by ℓ^2 matrix whose entries are quadratic polynomials in α_i, the problem is to find an assignment to the α_i such that the evaluation of the matrix at those α_i has rank $\leq r$. This can be done with algebraic methods, as explained by Faugère, Safey El Din, and Spaenlehauer [3], or in practice one can just do a brute force search over all $\boldsymbol{\alpha} \in \mathbb{F}_q^{\ell-1}$, since ℓ and q are very small in the proposed parameter sets.
2. Pick $\mathbf{v}_1, \ldots, \mathbf{v}_{\ell-1} \in \mathbb{F}_q^{n\ell}$ uniformly at random. Use a generic system solving algorithm, to solve for $\mathbf{u}_0$ such that $\mathcal{P}(\mathbf{U}) = \mathbf{t} = \mathcal{H}(M||\mathsf{salt})$, where $\mathbf{u}_i = \alpha_i \mathbf{u}_0 + \mathbf{v}_i$ for all $i \in \{1, \ldots, \ell-1\}$. Since $\mathbf{E}_{\boldsymbol{\alpha}}$ has rank mr, this is equivalent to solving a system of mr quadratic equations and $m(\ell^2 - r)$ linear equations in $n\ell$ variables. Or, after using the linear equations to eliminate some variables this becomes a system of mr quadratic equations in $n\ell - m(\ell^2 - r)$ variables.

Exploiting the $\mathbb{F}_q[\mathbf{S}]$-structure. In the case of SNOVA, we can do slightly better than the strategy outlined above. Instead of setting $\mathbf{u}_i = \alpha_i \mathbf{u}_0 + \mathbf{v}_i$ with scalar α_i we can set $\mathbf{u}_i = \mathbf{R}_i^{\otimes n} \mathbf{u}_0 + \mathbf{v}_i$ for $\mathbf{R}_i \in \mathbb{F}_q[\mathbf{S}]$. Indeed, let $\mathbf{R}, \mathbf{R}' \in \mathbb{F}_q[\mathbf{S}]$, then we have $B_i^{(a,b)}(\mathbf{R}^{\otimes n}\mathbf{x}, \mathbf{R}'^{\otimes n}\mathbf{y}) = B_i^{(0,0)}((\mathbf{S}^a\mathbf{R})^{\otimes n}\mathbf{x}, (\mathbf{S}^b\mathbf{R}')^{\otimes n}\mathbf{y})$, which is a linear combination of $B_i^{(a,b)}(\mathbf{x}, \mathbf{y})$ for $a, b \in \{0, \ell-1\}$. It follows that with $\mathbf{u}_i = \mathbf{R}_i^{\otimes n}\mathbf{u}_0 + \mathbf{v}_i$ for $i \in \{1, \ldots, \ell-1\}$ the quadratic part of $\mathcal{P}(\mathbf{U})$ is of the form

$$\mathcal{P}(\mathbf{U}) = \mathbf{E}_{\mathbf{R}} \cdot \mathcal{B}(\mathbf{u}_0, \mathbf{u}_0)\,,$$

where $\mathbf{E}_{\mathbf{R}} = \tilde{\mathbf{E}}_{\mathbf{R}}^{\otimes n}$ is a block diagonal matrix whose entries are quadratic functions of the coefficients of $\mathbf{R}_1, \ldots, \mathbf{R}_{\ell-1}$. The advantage of taking this approach is that now we are solving a generalized MinRank problem with $\ell(\ell-1)$ variables (the coefficients of $\mathbf{R}_1, \ldots, \mathbf{R}_{\ell-1}$), instead of just $\ell - 1$ variables (the $\alpha_1, \ldots, \alpha_{\ell-1}$). This extra freedom allows us to find matrices with lower rank in step 1, which means that solving the system in step 2 becomes more efficient.

5.1 The minimal rank r.

The attack is parameterized by r, the rank of the matrix $\tilde{\mathbf{E}}_{\mathbf{R}}$ used in the attack. This attack gives a trade-off between the cost of steps 1 and 2 of the attack. For smaller values of r, the cost of step 1 is higher, since finding linear combinations with smaller ranks is computationally more difficult, or even impossible. However, the system of quadratic equations that needs to be solved in step 2 has mr equations, so decreasing the value of r can decrease the cost of step 2 significantly.

Recall that $\tilde{\mathbf{E}}_{\mathbf{R}}$ is a ℓ^2-by-ℓ^2 matrix whose entries are quadratic polynomials in $\ell(\ell-1)$ variables. Therefore we are looking for a matrix with rank r in a family of $q^{\ell(\ell-1)}$ matrices. A random ℓ^2-by-ℓ^2 matrix has rank $\leq \ell^2 - d$ with probability roughly q^{-d^2}, so if we model the matrices in the family as independent random matrices, then heuristically we expect the average number of matrices of rank d to be roughly $q^{\ell(\ell-1)-d^2}$, and we expect that for most public keys the minimal achievable rank of $\tilde{\mathbf{E}}_{\mathbf{R}}$ is $\ell^2 - d$, where d is the largest integer such that $\ell(\ell-1)-d^2$ is non-negative, i.e. we expect the minimal achievable rank to typically be

$$\ell^2 - \lfloor \sqrt{\ell(\ell-1)} \rfloor = \ell^2 - \ell + 1 \,.$$

We expect that lower ranks are achievable for a small fraction of public keys. According to the heuristic, we expect the probability that a matrix of rank $\ell^2 - d$ or lower exists to be roughly

$$1 - \left(1 - q^{-d^2}\right)^{q^{\ell(\ell-1)}} \approx 1 - \exp(-q^{\ell(\ell-1)-d^2}) \leq q^{\ell(\ell-1)-d^2} \,.$$

However, the matrices $\tilde{\mathbf{E}}_{\mathbf{R}}$ are neither uniformly random nor independent, so the heuristic might not be very accurate. Our limited experiments of table 2 show that for a typical public key, the lowest rank is indeed $\ell^2 - \ell + 1$. However, for $\ell = 2$ the heuristic significantly underestimates the probability that minimal rank is lower. For $\ell = 3$ the probability of having ranks lower than $\ell^2 - \ell + 1 = 7$ is in agreement with our heuristic.

Parameter set	minimal rank	fraction of keys	heuristic
$q = 16, \ell = 2$	$r \leq 1$	0.000699%	$3.7 \cdot 10^{-9}$
	$r \leq 2$	2.093200%	0.389863%
	$r \leq 3$	99.999999%	99.999988%
	$r \leq 4$	100%	100%
$q = 16, \ell = 3$	$r \leq 5$	0%	$9 \cdot 10^{-13}$
	$r \leq 6$	0.00024%	0.0002441%
	$r \leq 7$	100%	100%
	$r \leq 8$	100%	100%

Table 2. Experiments of minimal rank of $\tilde{\mathbf{E}}_{\mathbf{R}}$ over all $\mathbf{R}_i \in \mathbb{F}_q[\mathbf{S}]$ for $i \in \{1, \ell-1\}$ for randomly generated public keys. We generated 100 million public keys for $q = 16, \ell = 2$ and 100 thousand public keys for $q = 16, \ell = 3$ and exhaustively compute the minimum of the rank of $\tilde{\mathbf{E}}_{\mathbf{R}}$ over all $\mathbf{R}_i \in \mathbb{F}_q[\mathbf{S}]$. We see that for $\ell = 2$ the heuristic underestimates the fraction of weak keys, for $\ell = 3$ the heuristic is much better. For $\ell = 4$ there are $q^{12} = 2^{48}$ possible $\mathbf{R}_i \in \mathbb{F}_q[\mathbf{S}]$, so it is not efficient to exhaustively compute the minimum rank. (In comparison, for the $\ell = 3$ experiment we computed the rank of $2^{40.6}$ matrices and this took 17 hours on a single machine.)

5.2 Solving the quadratic system.

In the second step of the attack, the attacker can forge a signature $\mathsf{sig} = (\mathbf{U}, \mathsf{salt})$ for an arbitrary message $M \in \{0,1\}^*$. The attacker picks a random salt and computes the target $\mathbf{t} = \mathcal{H}(M||\mathsf{salt})$. Finding a signature for the message is then equivalent to finding $\mathbf{U} \in \mathbb{F}_q^{n\ell \times \ell}$ such that $\mathcal{P}(\mathbf{U}) = \mathbf{t}$. The attacker picks $\mathbf{v}_1, \ldots, \mathbf{v}_{\ell-1} \in \mathbb{F}_q^{n\ell}$ uniformly at random and substitutes $\mathbf{u}_i = \mathbf{R}_i^{\otimes n}\mathbf{u}_0 + \mathbf{v}_i$ in $\mathcal{P}(\mathbf{U}) = \mathbf{t}$ for $i \in \{1, \ldots, \ell - 1\}$ to get a new system of quadratic equations in $\mathbf{u}_0$:

$$\mathcal{P}^*(\mathbf{u}_0) = \mathbf{E}_\mathbf{R} \cdot \mathcal{B}(\mathbf{u}_0, \mathbf{u}_0) + \text{Affine function of } \mathbf{u}_0 = \mathbf{t}^* \,.$$

We ensured in step 1 that the rank of $\mathbf{E}_\mathbf{R}$ is at most mr, so the span of the equations in $\mathcal{P}^*(\mathbf{u}_0) = \mathbf{t}^*$ contains at least $m(\ell^2 - r)$ linearly independent linear equations. The linear equations can be separated efficiently from the quadratic equations, e.g., by doing Gaussian elimination on the Macaulay matrix representing $\mathcal{P}^*(\mathbf{u}_0) = \mathbf{t}^*$ with respect to a graded monomial order. The linear equations can then be used to eliminate at least $m(\ell^2-r)$ variables to end up with a system

$$\mathcal{P}^{**}(\mathbf{y}) = \mathbf{t}^{**} \,,$$

with at most mr quadratic equations in at most $n\ell - m(\ell^2 - r)$ remaining variables $\mathbf{y}$. The attacker now computes a solution $\mathbf{y}$ to this system of equations using standard techniques such as FXL or F4/F5. Then, he extends the solution to an assignment to $\mathbf{u}_0$ such that $\mathcal{P}^*(\mathbf{u}_0) = \mathbf{t}^*$ using the $m(\ell^2 - r)$ linear equations. Finally the attacker computes $\mathbf{u}_i = \mathbf{R}_i^{\otimes n}\mathbf{u}_0 + \mathbf{v}_i$ for $i \in \{1, \ldots, \ell - 1\}$ to obtain the forgery $(\mathbf{U}, \mathsf{salt})$ for the message M. The computational bottleneck is finding the solution to $\mathcal{P}^{**}(\mathbf{y}) = \mathbf{t}^{**}$, the other steps can be done in polynomial time and are very efficient in practice.

Remark 1. For the proposed parameter sets in the SNOVA submission document, the system $\mathcal{P}^{**}(\mathbf{y}) = \mathbf{t}^{**}$ can be very underdetermined e.g. 17 equations in 57 variables for the first SNOVA parameter set with $r = 1$. To find a solution to this system the attacker should use one of the solving algorithms that exploit the underdeterminedness of the system [12,5,6]. These algorithms reduce the problem of finding a solution to an underdetermined system to that of finding a solution to some smaller overdetermined systems. E.g., finding a solution to 17 quadratic equations in 57 variables can be reduced to solving on average $q = 16$ systems of 14 equations in 13 variables, which is more efficient than trying to solve the larger system directly.

5.3 Cost Estimates and experimental results.

Table 3 shows the estimated cost of our attack applied to the various SNOVA parameter sets and with various values for the target rank r. The cost of the attack is always dominated by the cost of finding a solution to the system $\mathcal{P}^{**}(\mathbf{y}) = \mathbf{t}^{**}$ in step 2 of the attack. We estimate the cost of this attack with the standard methodology, applied to Hashimoto's method for solving underdetermined systems. For details, we refer to Appendix A.

We see in Table 3 that if we choose the parameter $r = \ell^2 - \ell + 1$, such that the attack applies to all public keys, then the estimated cost of our attack is lower that the estimated cost of the previously known attacks by a factor between 2^8 and 2^{39}. We also see that our attack works better for lower values of ℓ. This is because these parameter sets have higher values of m, so the rank defects of $\tilde{\mathbf{E}}_{\mathbf{R}}$ get amplified more strongly, leading to a larger reduction in the number of quadratic equations in the system $\mathcal{P}^{**}(\mathbf{y}) = \mathbf{t}^{**}$ in step 2 of the attack. For some of the parameter sets the number of bit operations required for our attack is lower than $2^{143}, 2^{207}$, or 2^{271}, the number of bit operations required for a key search on AES-128, AES-196 or AES-256 which correspond to NIST security levels I, III, and V respectively. However, counting the number of bit operations is a very crude cost model, and in more realistic cost models the attacks might be more costly than the cost of an AES key search.

If we pick smaller values of r, then we get more efficient attacks but they only apply to a subset of weak public keys. Note that in all cases the cost of finding and recognizing a weak key is much lower than the cost of running the weak-key attack. So, our weak key attacks further reduce security in a scenario where an attacker wants to forge a signature for one out of a sufficiently large set of public keys. A particularly dramatic reduction in security for the $\ell = 2$ parameter sets comes from $r = 1$. In these cases roughly one out of every 143000 keys is vulnerable to the attack and the estimated cost of the attack is only $2^{45}, 2^{68}$, and 2^{88} bit operations for the security level I, III, and V parameters respectively.

Experiments. To produce Table 2, we implemented a simple C program that repeatedly calls the key generation functionality of the reference implementation of SNOVA and computes the rank of $\tilde{\mathbf{E}}_{\mathbf{R}}$ for all $\mathbf{R} = (\mathbf{R}_1, \ldots, \mathbf{R}_{\ell-1}) \in \mathbb{F}_q[\mathbf{S}]^{\ell-1}$ and report the results. With $\ell = 2$ checking how weak a key is comes down to computing the rank of only $q^{\ell(\ell-1)} = 2^8$ matrices of size 4 by 4, which is much faster than running the key generation algorithm itself. For $\ell = 3$, we compute the rank of 2^{24} matrices of size 9 by 9, which takes roughly 4 seconds. We did not implement the search for $\ell = 4$, since an exhaustive search would require computing the rank of 2^{48} matrices of size 16 by 16.

We implemented our attack for the $(q, \ell, m, n) = (16, 2, 17, 54)$ parameter set with $r = 1$ using SageMath [11] and the implementation of the M4GB Gröbner basis algorithm by Makarim and Stevens [9]. First, the script reads a public key from a text file and computes $\mathbf{R}_1 \in \mathbb{F}_q[\mathbf{S}]$ such that $\tilde{\mathbf{E}}_{\mathbf{R}}$ has rank 1. Then it computes the system $\mathcal{P}^{**}(\mathbf{y}) = \mathbf{t}^{**}$, which is an underdetermined system of

Security Level	parameters (q, ℓ, m, n)	Attack	bit cost	fraction of weak keys	bit cost of finding a weak key
I	(16,2,17,54)	previous best	151	100%	-
		ours, $r = 3$	137	100%	14
		ours, $r = 2$	97	$2^{-8.9}$	26
		ours, $r = 1$	45	$2^{-17.1}$	35
I	(16,3,8,33)	previous best	159	100%	-
		ours, $r = 7$	150	100%	29
		ours, $r = 6$	130	$2^{-12.0}$	49
		ours, $r = 5$	112	$2^{-40.0\ *}$	77^*
I	(16,4,5,29)	previous best	175	100%	-
		ours, $r = 13$	167	100% *	52^*
		ours, $r = 12$	156	$2^{-16\ *}$	80^*
		ours, $r = 11$	145	$2^{-52\ *}$	116^*
III	(16,2,25,81)	previous best	215	100%	-
		ours, $r = 3$	189	100%	14
		ours, $r = 2$	132	$2^{-8.9}$	26
		ours, $r = 1$	68	$2^{-17.1}$	35
III	(16,3,11,60)	previous best	213	100%	-
		ours, $r = 7$	194	100%	29
		ours, $r = 6$	169	$2^{-12.0}$	49
		ours, $r = 5$	143	$2^{-40.0\ *}$	77^*
III	(16,4,8,45)	previous best	271	100%	-
		ours, $r = 13$	253	100% *	52^*
		ours, $r = 12$	235	$2^{-16\ *}$	80^*
		ours, $r = 11$	218	$2^{-52\ *}$	116^*
V	(16,2,33,108)	previous best	279	100%	-
		ours, $r = 3$	240	100%	14
		ours, $r = 2$	167	$2^{-8.9}$	26
		ours, $r = 1$	88	$2^{-17.1}$	35
V	(16,3,15,81)	previous best	285	100%	-
		ours, $r = 7$	253	100%	29
		ours, $r = 6$	221	$2^{-12.0}$	49
		ours, $r = 5$	187	$2^{-40.0\ *}$	77^*
V	(16,4,10,70)	previous best	335	100%	-
		ours, $r = 13$	307	100% *	52^*
		ours, $r = 12$	285	$2^{-16\ *}$	80^*
		ours, $r = 11$	264	$2^{-52\ *}$	116^*

Table 3. The estimated bit cost of our attack against the various SNOVA parameter sets. The entries marked by '*' depend on the heuristic from section 5.1 for the fraction of public keys that have a minimal rank r. The weak-key fractions without '*' were determined empirically.

17 quadratic equations in 57 variables. This process takes roughly 20 seconds. Then, the script runs our implementation of Hashimoto's algorithm for finding a solution to an underdetermined system of equations. We use the parameters $a = 3$ and $k = 1$, which means that the bottleneck is repeatedly trying to solve systems of $17 - a = 14$ quadratic equations in $17 - a - k = 13$ variables until one of the systems has a solution, which happens on average after roughly $q = 16$ attempts. However, there is a lot of variance in the number of attempts, since it follows a geometric distribution. Each attempt takes approximately 12 seconds for M4GB to solve it, in addition to a few seconds to generate the system and write it to a file in the M4GB format. Finally, when a solution is found for one of the systems, it takes less than a second to extend it to a valid forgery. The overall process takes usually between one and ten minutes, depending on how lucky we are with the number of systems that need to be solved.

6 Countermeasures and Conclusions

We have shown that the SNOVA signature scheme uses a weak variant of the "whipping" technique used by the MAYO signature scheme [1]. This leads to new attacks that significantly reduce the security margin of SNOVA. Moreover, there are weak-key attacks that are even more efficient but only apply to a subset of public keys. For one of the proposed parameter sets, we can even perform a forgery attack in practice within a few minutes for roughly a 1/143000 fraction of all public keys. The problem that enabled the attack is that SNOVA has the "whipping" structure as used by MAYO, but where linear combinations of the matrices $\mathbf{E}_{j,k}$ can have a low rank. Therefore, to block the attack it would be good to fully adopt the MAYO construction, i.e., one can use the formula (3) to define the public map, where instead of letting the $\mathbf{E}_{j,k}$ matrices be implicitly defined by the ad-hoc construction of (2), one fixes their values as in the MAYO signature scheme to make sure that non-trivial linear combinations of the matrices have full rank. This would be a significant change, but it would not have an impact on the signature size, and even reduce the public key size, since the seed to generate the $\mathbf{A}_\alpha, \mathbf{B}_\alpha, \mathbf{Q}_{1,\alpha}$, and $\mathbf{Q}_{2,\alpha}$ matrices does no longer need to be included in the public key. Note that SNOVA uses a single parameter ℓ to determine both the size of the matrices used in the base map and the whipping factor, even though there is no reason why these should be the same. Decoupling these parameters would give more flexibility to choose parameters.

MAYO uses the whipping technique on the plain Oil and Vinegar trapdoor, while SNOVA uses the whipping technique on a new and structured variant of the Oil and Vinegar trapdoor. Therefore, any attack on MAYO will likely directly apply to SNOVA as well. As an alternative countermeasure, one could drop the whipping technique altogether, and directly use the public map $\mathcal{P}(\mathbf{x}) := \mathcal{B}(\mathbf{x}, \mathbf{x})$ as trapdoor. This would require using larger parameters, but it would reduce the attack surface. We believe "SNOVA minus whipping" could be an interesting signature scheme, similar to, but not the same as QR-UOV [4]. We leave the specification, performance analysis, and security analysis of this signature scheme as an open problem.

NIST Round 2 version of SNOVA. For the second round of the NIST PQC process for additional signatures, the SNOVA authors have taken countermeasures against the attack in this paper. Contrary to the countermeasures we suggested above, the SNOVA authors persist in their ad-hoc version of the whipping technique. They propose a more complicated version of the SNOVA public map to break the block-diagonal structure of the $\mathbf{E}_{j,k}$ matrices. This makes the rank deficiencies of (linear combinations of) $\mathbf{E}_{j,k}$ less severe, which makes the attack less effective. Nonetheless, linear combinations of the $\mathbf{E}_{j,k}$ matrices with rank deficiencies still occur. For $\ell = 2$ and $\ell = 3$, to avoid weak keys, an additional change is that the A, B, Q matrices are now fixed, rather than chosen randomly during key generation, which means the $\mathbf{E}_{j,k}$ matrices are the same for all public keys. Hence all the public keys are equally strong against our attack. It is feasible to do an exhaustive search to find low-rank linear combinations of the $\mathbf{E}_{j,k}$ matrices, which allows to determine the cost of our attack. However, for larger ℓ this exhaustive search is not possible, so the A, B, Q matrices are sampled at random during key generation, and the SNOVA designers rely on the heuristic from Section 5.1 to lower-bound the cost of our attack, and to lower-bound the probability that weak keys are generated. Since the exhaustive search is too expensive to perform it does not seem possible to assess how accurate the heuristic is for the proposed parameters.

References

1. Ward Beullens. MAYO: Practical post-quantum signatures from oil-and-vinegar maps. In Riham AlTawy and Andreas Hülsing, editors, *SAC 2021*, volume 13203 of *LNCS*, pages 355–376. Springer, Cham, September / October 2022.
2. Ward Beullens, Ming-Shing Chen, Shih-Hao Hung, Matthias J. Kannwischer, Bo-Yuan Peng, Cheng-Jhih Shih, and Bo-Yin Yang. Oil and vinegar: Modern parameters and implementations. *IACR TCHES*, 2023(3):321–365, 2023.
3. Jean-Charles Faugère, Mohab Safey El Din, and Pierre-Jean Spaenlehauer. On the complexity of the generalized MinRank problem. *Journal of Symbolic Computation*, 55:30–58, 2013.
4. Hiroki Furue, Yasuhiko Ikematsu, Yutaro Kiyomura, and Tsuyoshi Takagi. A new variant of unbalanced Oil and Vinegar using quotient ring: QR-UOV. In Mehdi Tibouchi and Huaxiong Wang, editors, *ASIACRYPT 2021, Part IV*, volume 13093 of *LNCS*, pages 187–217. Springer, Cham, December 2021.
5. Hiroki Furue, Shuhei Nakamura, and Tsuyoshi Takagi. Improving thomae-wolf algorithm for solving underdetermined multivariate quadratic polynomial problem. In Jung Hee Cheon and Jean-Pierre Tillich, editors, *Post-Quantum Cryptography - 12th International Workshop, PQCrypto 2021, Daejeon, South Korea, July 20-*

22, 2021, Proceedings, volume 12841 of *Lecture Notes in Computer Science*, pages 65–78. Springer, 2021.
6. Yasufumi Hashimoto. An improvement of algorithms to solve under-defined systems of multivariate quadratic equations. *JSIAM Lett.*, 15:53–56, 2023.
7. Yasuhiko Ikematsu and Rika Akiyama. Revisiting the security analysis of SNOVA. In *Proceedings of the 11th ACM Asia Public-Key Cryptography Workshop*, pages 54–61, 2024.
8. Peigen Li and Jintai Ding. Cryptanalysis of the SNOVA signature scheme. In Markku-Juhani Saarinen and Daniel Smith-Tone, editors, *Post-Quantum Cryptography - 15th International Workshop, PQCrypto 2024, Part II*, pages 79–91. Springer, Cham, June 2024.
9. Rusydi H. Makarim and Marc Stevens. M4GB: an efficient gröbner-basis algorithm. In Michael A. Burr, Chee K. Yap, and Mohab Safey El Din, editors, *Proceedings of the 2017 ACM on International Symposium on Symbolic and Algebraic Computation, ISSAC 2017, Kaiserslautern, Germany, July 25-28, 2017*, pages 293–300. ACM, 2017.
10. Jacques Patarin. The oil and vinegar signature scheme. In *Presented at the Dagstuhl Workshop on Cryptography September 1997*, 1997.
11. The Sage Developers. *SageMath, the Sage Mathematics Software System (Version 9.7)*, 2022. `https://www.sagemath.org`.
12. Enrico Thomae and Christopher Wolf. Solving underdetermined systems of multivariate quadratic equations revisited. In Marc Fischlin, Johannes Buchmann, and Mark Manulis, editors, *PKC 2012*, volume 7293 of *LNCS*, pages 156–171. Springer, Berlin, Heidelberg, May 2012.
13. Lih-Chung Wang, Chun-Yen Chou, Jintai Ding, Yen-Liang Kuan, Ming-Siou Li, Bo-Shu Tseng, Po-En Tseng, and Chia-Chun Wang. SNOVA. Technical report, National Institute of Standards and Technology, 2023. available at `https://csrc.nist.gov/Projects/pqc-dig-sig/round-1-additional-signatures`.
14. Lih-Chung Wang, Po-En Tseng, Yen-Liang Kuan, and Chun-Yen Chou. A simple noncommutative UOV scheme. Cryptology ePrint Archive, Report 2022/1742, 2022. `https://eprint.iacr.org/2022/1742`.
15. Bo-Yin Yang, Owen Chia-Hsin Chen, Daniel J. Bernstein, and Jiun-Ming Chen. Analysis of QUAD. In Alex Biryukov, editor, *FSE 2007*, volume 4593 of *LNCS*, pages 290–308. Springer, Berlin, Heidelberg, March 2007.

A Methodology for table 3

Cost of Wiedemann FXL Under a genericity assumption, we can estimate the cost of solving a determined system of m equations in n variables using the Wiedemann FXL method [15] as

$$C_{\mathsf{FXL}}(n,m) := \min_{k'} q^{k'} \cdot 3\binom{n-k'+D}{D}^2 \binom{n-k'+2}{2} \left(\log(q)^2 + \log(q)\right)$$

bit operations, where D is the operating degree of XL, which is chosen to be the smallest integer such that the coefficient of the t^D term in the power series expansion of

$$\frac{(1-t^2)^m}{(1-t)^{n-k'+1}}$$

is non-positive.

Cost of Hashimoto's method Hashimoto's method [6] reduces the problem of solving an underdetermined system of m equations in n variables to finding a solution (if a solution exists) of on average

- $m-1-k+1$ systems of a quadratic equations in a variables,
- q^k systems of $a-1$ quadratic equations in $a-1$ variables, and
- q^k systems of $m-a$ quadratic equations in $m-a-k$ variables.

Where a, k are parameters of the algorithm chosen to minimize the overall cost of the attack, subject to the constraints

$$\begin{aligned} n &\geq (a+1)(m-k-a+1) \quad \text{, and} \\ n &\geq a(m-k)-(a-1)^2+k\,. \end{aligned} \tag{4}$$

Therefore the cost of Hashimoto's method measured as the number of bit operations is estimated as the minimum of

$$(m-a-k+1)\cdot C_{\mathsf{FXL}}(a,a)+q^k\left(C_{\mathsf{FXL}}(a-1,a-1)+C_{\mathsf{FXL}}(m-a-k,m-a)\right)$$

over the values of (a, k) satisfying (4). We computed these values and reported them in Table 3.

Weak key fraction. For some combinations of ℓ and r we determined the probability that $\mathbf{R}_1,\ldots,\mathbf{R}_{\ell-1}$ exist such that $\tilde{\mathbf{E}}_{\mathbf{R}}$ has rank $\leq r$ empirically. For those combinations, we have put the empirically determined probability in Table 3. For the remaining entries of the table, we instead rely on the heuristic from Section 5.1. This might underestimate the fraction of weak keys.

Cost of finding weak keys. Table 3 reports on the estimated cost of finding a weak key. For this estimate, we assume the attacker uses a very naive approach, where he enumerates all the possible $\mathbf{R}_1,\ldots,\mathbf{R}_{\ell-1}\in\mathbb{F}_q[\mathbf{S}]$, and computes the rank of the corresponding $\tilde{\mathbf{E}}_{\mathbf{R}}$ matrices. We believe it is likely that better approaches are possible, but since finding a weak key is already much cheaper than attacking the keys, we do not feel the need to optimize this. For the sake of concreteness, we say the cost of computing the rank of each $\tilde{\mathbf{E}}_{\mathbf{R}}$ matrix is

$$\ell^6/3 \cdot \big(\log(q)^2 + \log(q)\big)$$

bit operations, since $\ell^6/3$ is the number of required field multiplications and additions, and we say $\log(q)^2 + \log(q)$ is the cost of a field multiplication and a field addition. We multiply this by the expected number of ranks that need to be computed, which is either determined empirically, or otherwise we take it to be $q^{(\ell^2-r)^2}$ according to the heuristic from Section 5.1. The cost of computing the $\tilde{\mathbf{E}}_{\mathbf{R}}$ matrices can be amortized away, so we do not take it into account in the estimate.