Shorter VOLEitH Signature from Multivariate Quadratic

Dung Bui

IRIF, Université Paris Cité, Paris, France
bui@irif.fr

Abstract. VOLE-in-the-head paradigm recently introduced by Baum *et al.* (Crypto 2023) allows transforming zero-knowledge protocols in the designated verifier setting into public-coin protocols, which can be made non-interactive and publicly verifiable. Our transformation applies to a large class of ZK protocols based on vector oblivious linear evaluation (VOLE) and leads to resulting ZK protocols that have linear proof size and are simpler, smaller, and faster than related approaches based on MPC-in-the-head.
We propose a new candidate post-quantum signature scheme from the Multivariate Quadratic (MQ) problem based on a new protocol for the VOLE-in-the-head paradigm, which significantly reduces the signature size compared to previous works. We achieve a signature size of 2.5KB for a 128-bit security level. Compared to the state-of-the-art MQ-based signature schemes, our signature scheme achieves a factor from 3 to 4 improvement in terms of the signature size while keeping the computational efficiency competitive.

1 Introduction

Zero-Knowledge, Code-based signature schemes, and Multivariate Quadratic Assumption. Zero-knowledge proofs allow a prover to demonstrate to a verifier their knowledge of a witness for an NP statement without disclosing any additional information. These proofs have numerous applications in cryptography. Notably, the Fiat-Shamir transform [FS87] enables the conversion of any public-coin zero-knowledge proof system into a signature scheme, making it one of the primary methods for developing efficient signature schemes.

Digital signatures are fundamental to Internet authentication. However, the majority of existing constructions are susceptible to attacks by quantum computers [Sho94]. This drives the exploration of alternative digital signature schemes based on assumptions that are believed to resist quantum computer attacks. The recent call by NIST to standardize post-quantum cryptographic primitives has catalyzed research into efficient post-quantum signature schemes, with particular attention on code-based signatures. Recent code-based signature schemes built based on various assumptions (syndrome decoding, multivariate quadratic, MinRank, subset sum assumptions) and using two main paradigms that are MPC-in-the-head and later VOLE-in-the-head.

A multivariate quadratic map $\mathcal{F} : \mathbb{F}_p^n \to \mathbb{F}_p^m$ is a system of m quadratic polynomials in n variables defined over some finite field $\mathbb{F}_p$. The $\mathsf{MQ}_{p,m,n}$ problem is, for uniformly random $\mathcal{F} : \mathbb{F}_p^n \to \mathbb{F}_p^m$ and $\mathbf{x} \in \mathbb{F}_p^n$, to find $\mathbf{x}$ given $\mathcal{F}$ and $\mathcal{F}(\mathbf{x})$. The average-case hardness of the multivariate quadratic problem is one of the leading candidate post-quantum cryptographic assumptions. The Rainbow signature scheme [DS05] is one of the oldest and most studied signature schemes in multivariate cryptography. However, recent attacks [Beu21, Beu22] have reduced the security of this well-known construction.

Code-based signature schemes from MPCitH. Numerous recent studies on Fiat-Shamir code-based digital signatures have embraced the MPC in the head paradigm, initially introduced in the seminal work of [IKOS07]. In essence, this paradigm allows the prover to mentally execute an MPC protocol while virtual parties receive shares of the witness, and a target function validates the correctness of the witness. Subsequently, the prover commits to the view of all parties, and upon request from the verifier, opens a random subset of these views. The verifier checks the consistency of these views and verifies that the output corresponds to the correct witness. Soundness is guaranteed by the inability of a cheating prover to generate consistent views for all parties. Moreover, zero-knowledge property is established through the security of the MPC protocol against an honest-but-curious adversary, who only gains access to the perspectives of a subset of corrupted parties. Recently, there have been efficient code-based signatures [AGH$^+$23, FJR22, CCJ23, BCC$^+$24] from syndrome decoding assumption and MPCitH with signature sizes from 3 KB-7KB. Turing attention to multivariate quadratic assumption, [Beu20,Wan22,Fen22] proposed signatures with sizes ranging from 7KB-14KB.

Code-based signature schemes from VOLEitH. Another paradigm utilized in constructing signature schemes is VOLE-based zero-knowledge (ZK) proofs [WYKW21, BMRS21, YSWW21], which typically offer higher efficiency compared to most MPC in the head (MPCitH) protocols. However, these protocols rely on the existence of vector oblivious linear function evaluation (VOLE) correlations between the prover and the verifier, which can be efficiently generated using two-party protocols [BCG$^+$19]. Due to this prerequisite, VOLE-based protocols were unable to operate effectively in the public-coin model until the introduction of the VOLE-in-the-Head (VOLEitH) approach by Baum et al. [BBD$^+$23]. In addition to a generic zero-knowledge proof system, Baum et al. employed this approach to develop the FAEST post-quantum signature scheme, solely based on AES and hash functions. [CLY$^+$24] recently introduced a signature scheme based on the Regular Syndrome Decoding assumption with signature sizes of 4KB and it combines the VOLE-in-the-Head technique from [BBD$^+$23] with a sketching method of [BGI16] to reduce the check of the noise structure to a system of degree-2 equations, which are then proven using the Quicksilver VOLE-based zero-knowledge proof [YSWW21].

1.1 Our contribution

We first propose a new zero-knowledge protocol for the multivariate quadratic problem over a finite field. Our zero-knowledge proof is a publicly verifier and is constructed from a designated-verifier ZK protocol by using the VOLE-in-the-Head technique [BBD$^+$23], SoftspokenOT [Roy22] and multi-PPRF (puncturable pseudorandom function) [BCC$^+$24]. To construct the designated-verifier ZK protocol for the multivariate quadratic problem, firstly we observe that to prove the knowledge of the witness in MQ problem $\mathsf{MQ}_{p,m,n}$ is equivalent to proving the knowledge of the solutions in a set of polynomials of degree 2 over $\mathbb{F}_p$, then, therefore, we adapt the Quicksilver ZK proof [YSWW21] for nullity check of a polynomial with the cost of $O(n \log p)$ bits. Since all frameworks of VOLE-in-the-Head, SoftspokenOT, or Quicksilver ZK proof when applied to signature is only over the binary field $\mathbb{F}_2$, we carefully adapt all of them to get a public verifier ZK protocol over any finite field $\mathbb{F}_p$.

Table 1. Comparison of the new signature scheme with other signatures relying on the $\mathcal{MQ}$ problem (restricting to the schemes using the FS heuristics).

Instance	Protocol Name	Variant	Signature Size
$q = 4$ $n = 88$ $m = 88$	[SSH11] (3 rounds)	-	28 502 B
	MQ-DSS [CHR$^+$16]	-	41 444 B
	MudFish [Beu20]	-	14 640 B
	Mesquite [Wan22]	Fast	9 578 B
		Short	8 609 B
	[Fen22] (MQ version)	Fast	10 764 B
		Short	9 064 B
	Our scheme	Fast	5725 B
		Short	**2523 B**
$q = 4$ $n = 88$ $m = 88$	[SSH11] (3 rounds)	-	40 328 B
	MQ-DSS [CHR$^+$16]	-	28 768 B
	MudFish [Beu20]	Fast	15 958 B
		Short	13 910 B
	Mesquite [Wan22]	Fast	11 339 B
		Short	9 615 B
	[Fen22] (MQ version)	Fast	8 488 B
		Short	7 114 B
	Our scheme	Fast	9236 B
		Short	**2535 B**

Secondly, we achieve a new signature scheme based on the multivariate quadratic problem using the Fiat-Shamir transform, we compile our publicly verifier zero-knowledge protocol into an MQ-based signature scheme. We compare our scheme with the state of the art in two MQ instances:

- Multivariate Quadratic equations over a small field: $(q, m, n) = (4, 88, 88)$.
- Multivariate Quadratic equations over a larger field: $(q, m, n) = (256, 40, 40)$.

Both of these instances are believed to correspond to the security of 128 bits [BMPS20]. We present the asymptotic of our signature size for different settings on Table 2 and the comparison with the state-of-the-art of MQ-based signatures on Table 1. Compared to the state-of-the-art MQ-based signature schemes, our signature reduces the size of the signature by a factor from 3 to 4 improvement in various settings. Specifically, for the set of parameter $(q, m, n) = (4, 88, 88)$, our shortest signature has the size of 2523B while the current state of the art [Wan22] has the size of 8609B, and for the set of parameter $(q, m, n) = (256, 40, 40)$, we obtain the shortest of signature size of 2535B while the state of art [Fen22] is 7114B.

2 Preliminaries

2.1 Notation

Given a set S, we write $s \leftarrow_r S$ to indicate that s is uniformly sampled from S. Given a probabilistic Turing machine $\mathcal{A}$ and an input x, we write $y \leftarrow_r \mathcal{A}(x)$ to indicate that y is sampled by running $\mathcal{A}$ on x with a uniform random tape, or $y \leftarrow \mathcal{A}(x; r)$ when we want to make the random coins explicit. Given an integer $n \in \mathbb{N}$, we denote by $[n]$ the set $\{1, \cdots, n\}$. We use $\lambda = 128$ for the computational security parameter. We let $\mathsf{negl}(\lambda)$ denote any function that is negligible in the security parameter.

Field and Operations. We use $\mathbb{F}_p$ to denote a field and $\mathbb{F}_{p^r} = \mathbb{F}_p[X]/f(X)$ $(r \in \mathbb{N})$ as an etension field of $\mathbb{F}_p$. Wlog, we assume $\mathbb{F}_p$ is an extension field of $\mathbb{F}_2$. Given a vector $\mathbf{x} \in \mathbb{F}_p^t$ (or $\mathbb{F}_{p^r}^n$), we say $y = \mathsf{lift}_r(\mathbf{x})$ mean that we lift $\mathbf{x}$ to $\mathbb{F}_{p^r}$ as $y = \sum_{i=1}^{t} x_i \cdot X^i \in \mathbb{F}_{p^r}$ for $t \leq r$.

Vectors and Matrix. For a matrix $\mathbf{M}$, we denote $\mathbf{M}_{i,j}$ for entry in i^{th} row and j^{th} column, also $\mathbf{M^j}$, $\mathbf{M_i}$ as the j^{th} column and the i^{th} row respectively. The symbol $\odot$ is the point-wise product between a vector and a matrix i.e given a vector $\mathbf{u} = (u_1, \cdots, u_m) \in \mathbb{F}^m$ and a matrix $\mathbf{M} \in \mathbb{F}^{n \times m}$, $\mathbf{u} \odot \mathbf{M} = (u_1 \cdot \mathbf{M^1}, \cdots, u_m \cdot \mathbf{M^m}) \in \mathbb{F}^{n \times m}$. For the simplicity, given a polynomial f over n variables, a matrix $\mathbf{M} \in \mathbb{F}_p^{n \times \tau}$ we denote $f(\mathbf{M}) = (f(\mathbf{M}^0), \cdots, f(\mathbf{M}^{\tau-1}))$ as a vector in $\mathbb{F}_p^\tau$. We denote $\mathsf{diag}(\mathbf{\Delta})$ where $\mathbf{\Delta} = (\Delta_1, \cdots, \Delta_\tau) \in \mathbb{F}_p^\tau$ as a matrix over $\mathbb{F}_p^{\tau \times \tau}$ of the form $\begin{bmatrix} \Delta_1 & & \\ & \cdots & \\ & & \Delta_\tau \end{bmatrix}$. We use $[1 \cdots 1]$ for all-one row vector and $\mathbf{U} = [1 \cdots 1] \cdot \mathbf{u}$ is a matrix where each row is a repetition codeword $\mathbf{u}$.

Binary tree. Given a tree of size 2^D, for each leaf $i \in [2^D]$, we define $\mathsf{CoPath}(i)$ as co-path to i in the tree,i.e., the set of intermediate nodes that can be used to recover all leaves except the $i-$th one. Denote bit-decompose i as $\sum_{j=1}^{D} 2^{j-1} \cdot i_j$ for $i_j \in \{0, 1\}$, the associated value of i-th leaf is defined as $X_i := X_{i_1, \ldots, i_D}$.

2.2 Basic Cryptographic Definitions

Definition 1 (Indistinguishability). *Two distributions X, Y are (t, ϵ)-indistinguishable if for an algorithm $D : \{0,1\}^m \to \{0,1\}$ running in time t, we have $|\Pr[D(X) = 1] - \Pr[D(Y) = 1]| \leq \epsilon$.*

Definition 2 ((t, ϵ)-secure PRG). *Let $G : \{0,1\}^* \to \{0,1\}^*$ and let $l(.)$ be a polynomial such that for any input $s \in \{0,1\}^\lambda$ we have $G(s) \in \{0,1\}^{l(\lambda)}$. Then, G is a (t, ϵ)-secure pseudorandom generator if*

- *Expansion: $l(\lambda) > \lambda$;*
- *The distributions $\{G(s) | s \leftarrow \{0,1\}^\lambda\}$ and $\{r | r \leftarrow \{0,1\}^{l(\lambda)}\}$ are (t, ϵ)-indistinguishable.*

Definition 3 (Collision-Resistant Hash Functions). *A family of functions $\mathsf{Hash}_k : \{0,1\}^* \to \{0,1\}^{l(\lambda)}$; $k \in \{0,1\}^{\kappa(\lambda)}$ indexed by a security parameter λ is collision-resistant if there exists a negligible function v such that, for any PPT algorithm $\mathcal{A}$, we have:*

$$\Pr\left[\begin{array}{c} x \neq x' \\ \cap \mathsf{Hash}_k(x) = \mathsf{Hash}_k(x') \end{array} \middle| \begin{array}{c} k \in \{0,1\}^{\kappa(\lambda)} \\ (x, x') \leftarrow \mathcal{A}(k) \end{array}\right] \leq v(\lambda)$$

2.3 Multivariate Quadratic Problem

Given a tuple of parameters p, m, n, the Multivariate Quadratic Problem asks to find a vector solution in $\mathbb{F}_p^n$ (under the promise that it exists) to a random system of m linear equations over $\mathbb{F}_p$.

Definition 4 (Multivariate Quadratic Problem - Matrix form). *Let $\mathbb{F}_p$ be the finite field. Let (m, n) be positive integers. The multivariate quadratic problem $\mathsf{MQ}_{p,m,n}$ with parameters (p, m, n) is the following problem:*

- *(Problem generation) Sample $\mathbf{x} \leftarrow_r \mathbb{F}_p^n$ and $(\mathbf{A}_i)_{i\leq m} \leftarrow_r \mathbb{F}_p^{n\times n}$, $(\mathbf{b}_i)_{i\leq m} \leftarrow_r \mathbb{F}_p^n$. Set $y_i \leftarrow \mathbf{x}^T\mathbf{A}_i\mathbf{x} + \mathbf{b}_i^T\mathbf{x}$. Output $(\mathbf{A}_i, \mathbf{b}_i, y_i)_{i\leq m}$.*
- *(Goal) Given $(\mathbf{A}_i, \mathbf{b}_i, y_i)_{i\leq m}$, find $\mathbf{x} \in \mathbb{F}_p^n$ such that $\mathbf{x}^T\mathbf{A}_i\mathbf{x} + \mathbf{b}_i^T\mathbf{x} = y_i$ for all $i \in [m]$.*

Recent attacks have reduced the security of well-known constructions [Beu21, Beu22] as rainbow, in our work we use the parameter set in [BMPS20] that are considered as secure parameters for the security level of 128 bits.

2.4 The MPC-in-the-Head Paradigm

The MPC-in-the-head paradigm was initiated by the work of Ishai et al [IKOS07] and provided a compiler that can build honest-verifier zero-knowledge (HVZK) proofs for arbitrary circuits from secure MPC protocols. Assume we have an MPC protocol with the following properties:

- N parties $(P_1, \cdots, P_N)$ securely and jointly evaluate a function $f : \{0,1\}^* \to \{0,1\}$ on $\mathbf{x}$ while each party possess an additive share $[\![\mathbf{x}]\!]_i$ of input $\mathbf{x}$,
- Secure against passive corruption of $N-1$ parties i.e any $(N-1)$ parties can not recover any information about the secret $\mathbf{x}$.

Then the HVZK proof of knowledge of $\mathbf{x}$ such that $f(\mathbf{x}) = 1$ is constructed as:

- Prover generates the additively shares of the witness $\mathbf{x}$ into $([\![\mathbf{x}_1]\!], \cdots, [\![\mathbf{x}_N)]\!])$ among N virtual parties $(P_1, \cdots, P_N)$ and emulate the MPC protocol "in-the-head".
- Prover commits to the view of each party and sends commitments to the verifier.
- Verifier chooses randomly $(N-1)$ parties and asks the prover to reveal the view of these parties except one. The verifier later accepts if all the views are consistent with an honest execution of MPC protocol with output 1 and agrees with the commitments.

Security of MPC protocol implies that the verifier learns nothing about the input $\mathbf{x}$ from the $N-1$ shares, and MPC correctness guarantees that the Prover can only cheat with probability $1/N$. Security can then be amplified with parallel repetitions.

2.5 Information-Theoretic Message Authentication Codes

We use information-theoretic message authentication codes (IT-MACs) based on subfield Vector Obvious Linear (sVOLE) [YSWW21] to authenticate values over $\mathbb{F}_p$ or $\mathbb{F}_{p^r}$. Specifically, $[\![x]\!] = (\mathsf{K}[x], \mathsf{M}[x], x)$ is IT-MACs authenticated value $[\![x]\!]$ where $x \in \mathbb{F}_p$ is known by the P can be authenticated by the V who holds a global key $\Delta \in \mathbb{F}_{p^r}$ and a local key $\mathsf{K}[x] \in \mathbb{F}_{p^r}$, then P is given a MAC defined as $\mathsf{M}[x] = \mathsf{K}[x] - x \cdot \Delta$. IT-MACs is additively homomorphic, in particular, given the public coefficients $c_1, \cdots, c_l, c \in \mathbb{F}_p$ or $\mathbb{F}_{p^r}$, given $y = \sum_{i=1}^{l} c_i \cdot x_i + c$, the parties can locally compute $[\![y]\!] = (\mathsf{K}[y], \mathsf{M}[y], y)$ from $[\![x_i]\!]$ as $\mathsf{K}[y] = \sum_{i=1}^{l} c_i \cdot \mathsf{K}[x_i] + c$ and $\mathsf{M}[y] = \sum_{i=1}^{l} c_i \cdot \mathsf{M}[x_i] + c \cdot \Delta$. To authenticated x, P reveal x, $\mathsf{M}[x]$ to V to valid the correctness of sVOLE correlation. Note that the P can only cheat with a probability of $1/p^r$ since to find an IT-MACs $[\![x']\!] = (\mathsf{K}[x'], \mathsf{M}[x'], x')$, P needs to guess $\Delta \in \mathbb{F}_{p^r}$ such that $\Delta = (\mathsf{M}(x) - \mathsf{M}(x')) \cdot (x - x')^{-1}$.

Lemma 5 (Schwartz–Zippel lemma). *Let $P \in \mathbb{F}[x_1, x_2, \ldots, x_n]$ be a non-zero polynomial of total degree d over an field $\mathbb{F}$. Let S be a finite subset of $\mathbb{F}$ and let $r_1, \cdots, r_n$ be selected at random independently and uniformly from S. Then* $\Pr[P(r_1, r_2, \ldots, r_n) = 0] \leq d/|S|$.

Batch IT-MACs. We extend the above notation to vectors of authenticated values as well. In this case, $[\![\mathbf{x}]\!]$ means that $\mathsf{M}[\mathbf{x}] = \mathsf{K}[\mathbf{x}] - \mathbf{x} \cdot \Delta$ where $\Delta \in \mathbb{F}_{p^r}$, $\mathbf{x} \in \mathbb{F}_p^n$ and $\mathsf{M}[\mathbf{x}],\ \mathsf{K}[\mathbf{x}] \in \mathbb{F}_{p^r}^n$. To authenticated a vector $\mathbf{x} = (x_1, \cdots, x_n) \in \mathbb{F}_p^n$, instead of opening 2 vectors $\mathbf{x} \in \mathbb{F}_p^n, \mathsf{M}[\mathbf{x}] \in \mathbb{F}_{p^r}^n$, P only need to reveal a combinations $\sum_{i=1}^n \chi_i \cdot x_i$, $\sum_{i=1}^n \chi_i \cdot \mathsf{M}[x_i]$ where $\{\chi_i\}_{i \le n}$ are sampled randomly over $\mathbb{F}_{p^r}$, V then checks $\sum_{i=1}^n \chi_i \cdot \mathsf{M}[x_i] + \Delta \cdot \sum_{i=1}^n \chi_i \cdot x_i = \sum_{i=1}^n \chi_i \cdot \mathsf{K}[x_i]$. The security holds with the soundness error of $(n+1)/p^r$ by following the SZ lemma (Section 2.5).
In our concept of signature to maintain both soundness and efficiency, we extend the authenticated vectors multiple times (denoted as τ times). It means for an authenticated vector $\mathbf{x} \in \mathbb{F}_p^n$, $[\![\mathbf{x}, \tau]\!] = (\mathbf{K}[\mathbf{x}], \mathbf{M}[\mathbf{x}], \mathbf{x})$ and has a global key $\mathbf{\Delta} = (\Delta_1, \cdots, \Delta_\tau)$ corresponding, where $\mathbf{K}[\mathbf{x}], \mathbf{M}[\mathbf{x}] \in \mathbb{F}_{p^r}^{n \times \tau}$ and $\mathbf{\Delta} \in \mathbb{F}_p^\tau$ such that $\mathbf{M_i}[\mathbf{x}] = \mathbf{K_i}[\mathbf{x}] - \mathbf{x} \cdot \Delta_i$ for all $i \in [0, \ldots, \tau)$. The batch IT-MACs of a vector are generated by the sVOLE protocol that securely realizes the ideal functionality Figure 1 and the efficient way to authenticate is followed by technique in SoftSpokenOT (see Section 4.2).

PARAMETERS:

- Given a field extension $\mathbb{F}_p \subseteq \mathbb{F}_{p^r}$. Denote n as the length of each vector that is produced in each sVOLE instance and τ as the repetition parameter.

FUNCTIONALITY:

- Depending on P and V:
  - If the P is corrupted then wait for $\mathcal{A}$ to send $\mathbf{u} \in \mathbb{F}_p^n, \mathbf{V} \in \mathbb{F}_{p^r}^{n \times \tau}$; samples $\mathbf{\Delta} \leftarrow_r \mathbb{F}_{p^r}^\tau$ and computes $\mathbf{W} := \mathbf{V} + \mathbf{u} \cdot [1 \cdots 1] \cdot \mathsf{diag}(\mathbf{\Delta})$.
  - If the V is corrupted then wait for $\mathcal{A}$ to send $\mathbf{\Delta} \in \mathbb{F}_{p^r}^\tau$, $\mathbf{W} \in \mathbb{F}_{p^r}^{n \times \tau}$; samples $\mathbf{u} \leftarrow_r \mathbb{F}_p^n$ and computes $\mathbf{V} := \mathbf{W} - \mathbf{u} \cdot [1 \cdots 1] \cdot \mathsf{diag}(\mathbf{\Delta})$.
  - Otherwise, samples $\mathbf{u} \leftarrow_r \mathbb{F}_p^n, \mathbf{W} \leftarrow_r \mathbb{F}_{p^r}^{n \times \tau}, \mathbf{\Delta} \leftarrow_r \mathbb{F}_{p^r}^\tau$ and computes $\mathbf{W} := \mathbf{V} + \mathbf{u} \cdot [1 \cdots 1] \cdot \mathsf{diag}(\mathbf{\Delta})$.
- If P and V send (init) to functionality, sends $(\mathbf{u},\ \mathbf{V})$ to P
- If P and V send (get) to functionality and $(\mathbf{\Delta},\ \mathbf{W})$ to V.
- P defines $\mathbf{M}[\mathbf{x}] = \mathbf{V}$ and V defines $\mathbf{K}[\mathbf{x}] = \mathbf{W}$.

Fig. 1. Ideal functionality $\mathcal{F}_{\mathsf{sVOLE}}^{n,\tau}$ of multi-subVOLE over $\mathbb{F}_p$

2.6 Designated-Verifier ZK for nullity check of Polynomial sets

We recall the ideal functionality of nullity check for a set of t polynomials of degree 2 having n variables over $\mathbb{F}_p$ in Figure 2. From the footprint of IT-MACs, the instantiation of $\mathcal{F}_{\mathsf{polyZK}}^{p,t}$ is followed by Quicksilver technique [YSWW21].

PARAMETERS:

- Given an arbitrary field $\mathbb{F}_p$.
- Prover P and verifier V hold t polynomials $f_1, \cdots, f_t$ of degree 2 all over n variables.
- Prover P holds a witness $\mathbf{w} \in \mathbb{F}_p^n$ such that $f_i(\mathbf{w}) = 0$ for all $i \in [t]$.

FUNCTIONALITY:

- Upon receiving $(\mathsf{prove}, \{f_i\}_{i \in [t]}, \mathbf{w})$ from P and $(\mathsf{verify}, \{f_i\}_{i \in [t]})$ from V then: Send true to V if $f_i(\mathbf{w}) = 0$ for all $i \in [t]$ and false to V otherwise.

Fig. 2. Ideal functionality $\mathcal{F}_{\mathsf{polyZK}}^{p,t}$

In particular, given t polynomials $\{f_i\}_{i \le t}$ of degree 2 over n variables, P holds a witness $\mathbf{w} \in \mathbb{F}_p^n$ and wants to prove that $f_i(\mathbf{w}) = 0$. For every polynomial f_i, we present it as $f_i = f_{i,2} + f_{i,1} + f_{i,0}$ where $f_{i,h}$ is a degree-h polynomial such that all terms in $f_{i,h}$ have exactly degree h.

Given an IT-MACs of $[\![\mathbf{w}]\!] = (\mathsf{M}[\mathbf{w}], \mathsf{K}[\mathbf{w}], \mathbf{w})$, P and V hold $(\mathbf{w}, \mathsf{M}[\mathbf{w}])$ and $(\Delta, \mathsf{K}[\mathbf{w}])$ respectively. Now V computes:

$$\begin{aligned} B_i &= f_{i,2}(\mathsf{K}[\mathbf{w}]) + f_{i,1}(\mathsf{K}[\mathbf{w}]) \cdot \Delta + f_{i,0} \cdot \Delta^2 \\ &= f_{i,2}(\mathsf{M}[\mathbf{w}] + \Delta \cdot \mathbf{w}) + f_{i,1}(\mathsf{M}[\mathbf{w}] + \Delta \cdot \mathbf{w}) \cdot \Delta + f_{i,0} \cdot \Delta^2 \\ &= f_i(\mathbf{w}) \cdot \Delta^2 + A_{i,1} \cdot \Delta + A_{i,0} = A_{i,1} \cdot \Delta + A_{i,0}. \end{aligned}$$

where $A_{i,0}, A_{i,1}$ are the aggregated coefficient for all terms with Δ and constant coefficients. Note that the prover with witnesses $\mathbf{w}$ and MACs $\mathsf{M}[\mathbf{w}]$ can compute all the coefficients locally.

Batch nullity check. P and V generates randomly an IT-MACs $[\![A]\!] = (\mathsf{M}[A], \mathsf{K}[A], A)$, P then defines (A_1, A_0) and sends to V:

$$A_1 := \sum_{i=1}^{t} \chi^i \cdot A_{i,1} + A,\ A_0 = \sum_{i=1}^{t} \chi^i \cdot A_{i,0} + \mathsf{M}[A]$$

While V checks if $\sum_{i=1}^{t} \chi^i \cdot B_i + \mathsf{K}[A] = A_1 \cdot \Delta + A_0$. Note that to obtain a negiliable soundness error ($(t+2)/p^r$ [YSWW21]), we lift the IT-MAC $[\![A]\!]$ to the extension field $\mathbb{F}_{p^r}$ instead of $\mathbb{F}_p$ (see Section 3 for details). As a consequence, the batch nullity check results in a total communication of $(n+2r)\log p$ bits in the sVOLE-hybrid model. When using the interpolation approach to compute the coefficients A_0, A_1, we have that the computational cost of the prover and verifier is $O(4tz + 2n)$ and $O(2tz)$ respectively, where z is the maximum number of terms in all t polynomials.

2.7 Universal Hashing

We recall the definitions of n-hiding and ϵ-universal [Roy22] in the following definitons:

Definition 6 (Universal). *A family of linear hash functions is a family of matrices $\mathcal{H} \subseteq \mathbb{F}_p^{r \times n}$. The family is ϵ-almost universal if for any non-zero $\mathbf{x} \in \mathbb{F}_p^n$*

$$\Pr_{\mathbf{H} \leftarrow \mathcal{H}}[\mathbf{H}\mathbf{x} = 0] \leq \epsilon$$

The family is ϵ-almost uniform, if for any non-zero $\mathbf{x} \in \mathbb{F}_p^n$ and for any non-zero $\mathbf{x} \in \mathbb{F}_p^r$.

$$\Pr_{\mathbf{H} \leftarrow \mathcal{H}}[\mathbf{H}\mathbf{x} = \mathbf{v}] \leq \epsilon$$

Definition 7 (Hiding). *A matrix $\mathbf{H} \in \mathbb{F}_p^{r \times (n+h)}$ is $\mathbb{F}_p^n$-hiding if the distribution of $\mathbf{H}\mathbf{v}$ is independent from $\mathbf{v}[0 \dots n)$ when $\mathbf{v}[n \dots n+h) \leftarrow \mathbb{F}_p^h$. A hash family $\mathcal{H} \subseteq \mathbb{F}_p^{r \times (n+h)}$ is $\mathbb{F}_p^n$-hiding if every $\mathbf{H} \in \mathcal{H}$ is $\mathbb{F}_p^n$-hiding.*

Transforming a uniform hash family into a universal family that is hiding is followed by the below proposition.

Proposition 8. *Let $\mathcal{H} \subseteq \mathbb{F}_p^{r \times n}$ be an ϵ-almost uniform hash family. Let $\mathcal{H}^{\mathsf{UHF}} \subseteq \mathbb{F}_p^{r \times (n+r)}$ be the family $\{[\mathbf{H}\mathbf{I}_r] : \mathbf{H} \in \mathcal{H}\}$, where $\mathbf{I}_r$ is the $r \times r$ identity matrix. Then, it holds that (1) $\mathcal{H}'$ is ϵ-almost universal, and (2) $\mathcal{H}^{\mathsf{UHF}}$ is $\mathbb{F}_p^n$-hiding.*

In this paper, we assume the family $\mathcal{H}^{\mathsf{UHF}}$ is a simple matrix hash family [BJKS94], where $\mathcal{H} = \mathbb{F}_p^{r \times n}$ which is p^{-r}-uniform.

2.8 Multi-Instance Puncturable PRF

Pseudorandom functions [GGM86], are families of keyed functions F_k such that no adversary can distinguish between a black-box access to F_k for a random key k and access to a truly random function. A puncturable pseudorandom function (PPRF) [KPTZ13, BW13, BGI14] is a PRF F such that given an input x, and a PRF key k, one can generate a *punctured* key, denoted $k\{x\} = F.\mathsf{Punc}(K, x)$, which allows evaluating F at every point except for x (i.e., there is an algorithm $F.\mathsf{Eval}$ such that

$F.\mathsf{Eval}(k\{x\}, x') = F_K(x')$ for all $x' \neq x$), and such that $F_k(x)$ is indistinguishable from random given $k\{x\}$. Then we recall the definition of τ-multi-instance PRF [BCC+24]. The motivation for using τ-multi-instance PRF from our use of PPRFs in signatures: our signature construction uses τ parallel instances of the PPRF using the same K, while distinct salts are used across distinct signature queries. Then,

Definition 9 ((N,τ)-instance (t,ϵ)-secure PPRF). *A function family $F = \{F_K\}$ with input domain $[2^D]$, salt domain $\{0,1\}^s$, and output domain $\{0,1\}^\lambda$, is an (N,τ)-instance (t,ϵ)-secure PPRF if it is a PPRF which additionally takes as input a salt K, and for every non-uniform PPT distinguisher $\mathcal{D}$ running in time at most t, it holds that for all sufficiently large λ,*

$$\mathsf{Adv}^{\mathsf{PPRF}}(\mathcal{D}) = |\Pr[\mathsf{Exp}^{\mathsf{rw\text{-}pprf}}_{\mathcal{D}}(\lambda) = 1] - \Pr[\mathsf{Exp}^{\mathsf{iw\text{-}pprf}}_{\mathcal{D}}(\lambda) = 1]| \leq \epsilon(\lambda)$$

where the experiments $\mathsf{Exp}^{\mathsf{rw\text{-}pprf}}_{\mathcal{D}}(\lambda)$ and $\mathsf{Exp}^{\mathsf{iw\text{-}pprf}}_{\mathcal{D}}(\lambda)$ are defined below.

$\underline{\mathsf{Exp}^{\mathsf{rw\text{-}pprf}}_{\mathcal{D}}(\lambda)}$:	$\underline{\mathsf{Exp}^{\mathsf{iw\text{-}pprf}}_{\mathcal{D}}(\lambda)}$:
– $((K_{j,e})_{j\leq N, e\leq\tau} \leftarrow_r (\{0,1\}^\lambda)^{N\cdot\tau}$ – $\mathsf{K} := (\mathsf{K}_1,\ldots,\mathsf{K}_N) \leftarrow_r \{0,1\}^s$ – $\mathbf{i} := ((i_{1,e})_{e\leq\tau},\ldots,(i_{1,e})_{e\leq\tau}) \leftarrow_r [2^D]^{N\cdot\tau}$ – $\forall j \leq N, e \leq \tau : K^{i_{j,e}}_{j,e} \leftarrow F.\mathsf{Punc}(K_{j,e}, i_{j,e})$ – $(y_{j,e})_{j\leq N,e\leq\tau} \leftarrow (F_{K_{j,e}}(i_{j,e},\mathsf{K}_i))_{j\leq N,e\leq\tau}$ **Output** $b \leftarrow \mathcal{D}(\mathsf{K}, \mathbf{i}, (K^{i_{j,e}}_{j,e}, y_{j,e})_{j\leq N,e\leq\tau})$	– $((K_{j,e})_{j\leq N,e\leq\tau} \leftarrow_r (\{0,1\}^\lambda)^{N\cdot\tau}$ – $\mathsf{K} := (\mathsf{K}_1,\ldots,\mathsf{K}_N) \leftarrow_r \{0,1\}^s$ – $\mathbf{i} := ((i_{1,e})_{e\leq\tau},\ldots,(i_{1,e})_{e\leq\tau}) \leftarrow_r [2^D]^{N\cdot\tau}$ – $\forall j \leq N, e \leq \tau : K^{i_{j,e}}_{j,e} \leftarrow F.\mathsf{Punc}(K_{j,e}, i_{j,e})$ – $(y_{j,e})_{j\leq N,e\leq\tau} \leftarrow_r (\{0,1\}^\lambda)^{N\cdot\tau}$ **Output** $b \leftarrow \mathcal{D}(\mathsf{K}, \mathbf{i}, (K^{i_{j,e}}_{j,e}, y_{j,e})_{j\leq N,e\leq\tau})$

3 Technical Overview

Designated-Verifier Zero-Knowledge Proofs from batch nullity checks. Since $\mathbf{x} \in \mathbb{F}_p^n$ in $\mathsf{MQ}_{p,m,n}$ problem is the solution of a system of equations $\{f_i(x_1,\cdots,x_n) = \mathbf{x}^T\mathbf{A}_i\mathbf{x} + \mathbf{b}_i^T\mathbf{x} - y_i\}_{i\leq m}$ of degree 2 over n variables (consider $\mathbf{x}^T\mathbf{A}_i\mathbf{x}+\mathbf{b}_i^T\mathbf{x}$ as a multivariate polynomial). Therefore, we transfer constructing a DVZK protocol for the MQ problem into a DVZK protocol for batch m nullity check of a polynomial set. Its instantiation is based on Quicksilver technique [WWCY22] in the sVOLE hybrid model. Each polynomial f_i is presented as $f_i = f_{i,1} + f_{i,2}$ where all terms in $f_{i,1}$ and $f_{i,2}$ have degree of 1 and 2 respectively (specifically, $f_{i,1}$ and $f_{i,2}$ have n and n^2 terms, for simplicity we omitted y_i since y_i can be easy adapted as a constant coefficient). For the efficiency of the signature constructed from this DVZK protocol, we need to repeat the DVZK protocol τ-times. To build a nullity check for the set of polynomials, Quicksilver technique [WWCY22] for polynomials check is applied and upgraded to the version of τ- repetitions. In particular,

- Given an IT-MACs of $\mathbf{x}$ in the version of τ- repetitions i.e., $(\mathbf{M}[\mathbf{x}], \mathsf{K}[\mathbf{x}], \mathbf{x}) \in \mathbb{F}_{p^r}^{n\times\tau} \times \mathbb{F}_{p^r}^{n\times\tau} \times \mathbb{F}_p^n$, P locally computes $f_i(\mathbf{M}[\mathbf{x}] + \mathbf{x}\cdot[1\cdots1]\cdot\mathsf{diag}(X)) = \mathbf{A}_{i,0} + \mathbf{A}_{i,1}X$ (degree only 1 since the coefficient of degree 2 is 0 if $\mathbf{x}$ is solution of f_i).
- From $(\boldsymbol{\Delta}, \mathbf{K}[\mathbf{x}]) \in \mathbb{F}_{p^r}^{\tau} \times \mathbb{F}_{p^r}^{n\times\tau}$, V can locally compute $\mathbf{B}_i := f_{i,1}(\mathbf{K}[\mathbf{x}]) \circ \boldsymbol{\Delta} + f_{i,2}(\mathbf{K}[\mathbf{x}])$. Note that $\mathbf{B}_i = \mathbf{A}_{i,0} + \mathbf{A}_{i,1} \circ \boldsymbol{\Delta}$ since $\mathbf{K}[\mathbf{x}] := \mathbf{M}[\mathbf{x}] + \mathbf{x}\cdot[1\ldots1]\cdot\mathsf{diag}(\boldsymbol{\Delta})$.
- V and P check relation $\mathbf{B}_i = \mathbf{A}_{i,0} + \mathbf{A}_{i,1} \circ \boldsymbol{\Delta}$ for $i \in [m]$ by sending to each other a sample challenge $\chi \in \mathbb{F}_{p^r}^{\tau}$ and a linear combination of $(\mathbf{A}_{i,0}, \mathbf{A}_{i,1})$ constructing from χ respectively. Note that we need to generate τ extra OLEs over $\mathbb{F}_{p^r}$ to mask $(\mathbf{A}_{i,0}, \mathbf{A}_{i,1})$ in the linear combination sent from P, this only costs $(r\cdot\tau)$ sVOLE correlations over $\mathbb{F}_p$ and it is negligible when $r\cdot\tau$ is small. See Section 4 for more details about lifting $(r\cdot\tau)$ sVOLEs from the subfield $\mathbb{F}_p$ to τ OLEs over the extension field $\mathbb{F}_{p^r}$.

MQ-based signature from VOLE-in-the-Head paradigm. We first design a public-coin ZK proof for MQ problem and then apply the Fiat-Shamir transform to make it into a signature scheme. Our public-coin ZK protocol is constructed from our DVZK protocol by using the VOLE in-the-head paradigm. In particular, since in our DVZK protocol only V who holds the global key $\boldsymbol{\Delta}$ obtained from sVOLE correlation can valid the proof, therefore we use Spoftspoken OT to construct sVOLE, this manner allows every Pr defines $\boldsymbol{\Delta}$ as a challenge in publicly verifier ZK protocol. Then we use multi-instance PPRF to efficiently generate and open the set of sd that is used in constructing softspokenOT-based sVOLE. Note that compared to the original VOLE-in-the-head mamner [BBD+23], we use

multi-instance PPRF instead of vector commitment and construct commitment that is associated with each sd by a pseudorandom generator PRG later it is modeled as a random oracle. Finally, applying the Fiat-Shamir heuristic to publicly verifier ZK protocol, we achieve a new signature scheme from the MQ problem, and the security is maintained from multi-instance PPRF, publicly verifier ZK protocol, and random oracles.

4 Publicly-Verifier ZK for Multivariate Quadratic problem

Observe that for $\mathsf{MQ}_{p,m,n}$, given $(\mathbf{A}_i, \mathbf{b}_i, y_i)_{i\le m}$, to prove that the prover P holds $\mathbf{x} \in \mathbb{F}_p^n$ such that $\mathbf{x}^T\mathbf{A}_i\mathbf{x} + \mathbf{b}_i^T\mathbf{x} = y_i$ for all $i \in [m]$, P needs to convince that P knows the solution of a system of m questions of form $\mathbf{x}^T\mathbf{A}_i\mathbf{x} + \mathbf{b}_i^T\mathbf{x} = y_i$. If we consider $\mathbf{x}^T\mathbf{A}_i\mathbf{x} + \mathbf{b}_i^T\mathbf{x}$ as a multivariate polynomial of degree 2 over n variables $\mathbf{x} = (x_i, \cdots, x_n) \in \mathbb{F}_p^n$ then it is equivalent to prove that P holds a root of a set of m polynomials degree 2. It means that to prove the knowledge of $\mathsf{MQ}_{p,m,n}$, it is sufficient to prove the knowledge of a root of a polynomial set of degree 2.

Firstly, we present in the Section 4.1 a zero-knowledge protocol for MQ based on Quicksilver (see Section 2.6) in the sVOLE hybrid model. Since the limit of sVOLE, this construction Figure 3 is only a designated verifier ZK protocol. To turn it into a publicly verifier, we apply techniques from [BBD+23] to combine VOLEitH, multi-instance PRF [BCC+24] and SoftspokenOT [Roy22]. Note that to achieve a signature scheme based on VOLEitH signature FAEST [BBD+23] we adapt all constructions for multi-times τ repetitions.

4.1 Designated-Verifier ZK for Multivariate Quadratic problem

In this section, we present an efficient zero-knowledge proof for the MQ problem Figure 3 in the $\mathcal{F}_{\mathsf{sVOLE}}^{n,\tau}$ hybrid model, and its security (soundness and zero-knowledge) is shown in Theorem 10. Specifically, two main concerns need to be addressed:

- Firstly, packing subfield VOLE correlations between $\mathbb{F}_p$ and $\mathbb{F}_{p^r}$ into OLEs correlations over $\mathbb{F}_{p^r}$ to get mask for QS check with a soundness of $O(2^{-\lambda})$. Given $(r\cdot\tau)$ instances of sVOLE correlation over $\mathbb{F}_p$ i.e.,

$$\mathbf{W} = \mathbf{V} + \mathbf{u}\cdot[1\cdots 1]\cdot\mathsf{diag}(\boldsymbol{\Delta}) \text{ where } \mathbf{u} \in \mathbb{F}_p^{r\cdot\tau},\ \mathbf{V}, \mathbf{W} \in \mathbb{F}_{p^r}^{(r\cdot\tau)\times\tau}$$

  we define $(\mathbf{A}_0^*, \mathbf{A}_1^*, \mathbf{B}^*) \in \mathbb{F}_{p^r}^{\tau}$ as for all $i \in [0, \tau)$:

$$\begin{aligned}
\mathbf{A}_{1,i}^* &= \mathsf{lift}_r(\mathbf{u}_{[i\cdot r\ldots(i+1)\cdot r)}) \in \mathbb{F}_{p^r},\\
\mathbf{A}_{0,i}^* &= \mathsf{lift}_r(\mathbf{V}^i_{[i\cdot r\ldots(i+1)\cdot r)}) \in \mathbb{F}_{p^r},\\
\mathbf{B}_i^* &= \mathsf{lift}_r(\mathbf{W}^i_{[i\cdot r\ldots(i+1)\cdot r)}) \in \mathbb{F}_{p^r}.
\end{aligned}$$

  From the additive homomorphic property of sVOLE correlation, it is easy to check that $\mathbf{B}^* = \mathbf{A}_0^* + \mathbf{A}_1^* \circ \boldsymbol{\Delta} \in \mathbb{F}_{p^r}^{\tau}$.
- Secondly, how to apply Quicksilver techniques for the polynomial set (Section 2.6) to τ-repetitions of nullity check. Recall notation, given a polynomial f over n variables, a matrix $\mathbf{M} \in \mathbb{F}_p^{n\times\tau}$ we denote $f(\mathbf{M}) = (f(\mathbf{M}^0), \cdots, f(\mathbf{M}^{\tau-1}))$ as a vector in $\mathbb{F}_p^{\tau}$. We can see that the correctness is shown similarly as in Quicksilver except working on the vector of length τ instead of a single instance. We consider $\mathbf{A}_0^*, \mathbf{A}_1^*$ as vectors over $\mathbb{F}_{p^r}^{\tau}$ and we have:

$$\begin{aligned}
\mathbf{B} &= \sum_{i=1}^{m} \mathbf{B}_i \circ \chi^i + \mathbf{B}^* = \sum_{i=1}^{m} \left(f_{i,1}(\mathbf{K}[\mathbf{x}]) \circ \boldsymbol{\Delta} + f_{i,2}(\mathbf{K}[\mathbf{x}])\right) \circ \chi^i + \mathbf{B}^*\\
&= \sum_{i=1}^{m} \mathbf{g}_i(\boldsymbol{\Delta}) \circ \chi^i + \mathbf{B}^* \qquad (\text{since } \mathbf{K}[\mathbf{x}] := \mathbf{M}[\mathbf{x}] + \mathbf{x}\cdot[1\ldots 1]\cdot\mathsf{diag}(\boldsymbol{\Delta}))\\
&= \sum_{i=1}^{m} \left(\mathbf{A}_{i,0} + \mathbf{A}_{i,1} \circ \boldsymbol{\Delta} + \mathbf{A}_{i,2} \circ \boldsymbol{\Delta}^2\right) \circ \chi^i + \mathbf{A}_0^* + \mathbf{A}_1^* \circ \boldsymbol{\Delta}\\
&= \mathsf{QS}_0 + \mathsf{QS}_1 \circ \boldsymbol{\Delta} \qquad (\text{since } \mathbf{A}_{i,2} = 0 \text{ if } \mathbf{x} \text{ is witness })
\end{aligned} \tag{1}$$

Theorem 10. *The protocol $\Pi_{\mathsf{MQ-DVZK}}$ is an honest verifier zero-knowledge protocol for multivariate quadratic problem $\mathsf{MQ}_{p,m,n}$ in the $\mathcal{F}_{\mathsf{sVOLE}}$-hybrid model. The security holds against a malicious prover or a semi-honest verifier with the soundness error in the former case is bounded by $((m+2)/p^r)^\tau$ and information-theoretic security.*

The proof is followed by [BBD+23, CLY+24]. The soundness error comes from whether the malicious prover can 1) guess correctly the value of $\mathbf{\Delta} \leftarrow_r \mathbb{F}_{p^r}^\tau$ this happens with a probability of $(1/p^r)^\tau$, or 2) cheat in Equation (1) to keep this equation hold since this equation has degree 2, $\mathbf{\Delta} \leftarrow_r \mathbb{F}_{p^r}^\tau$ is uniformly random and kept secret from the adversary's view then from Section 2.5, the probability that the above equation holds is bounded by $((m+2)/p^r)^\tau$.
We can use the Fiat-Shamir heuristic to make the online phase non-interactive at the cost of the information-theoretic security is degraded to computation security. Specifically, both parties can compute $\chi \in \mathbb{F}_{p^r}^\tau$ as $\mathsf{H}(\gamma_0, \cdots, \gamma_{n-1})$, where $\mathsf{H} : \{0,1\}^* \to \mathbb{F}_{p^r}^\tau$ is a cryptographic hash function modeled as a random oracle and $(p^r)^\tau \geq 2^\lambda$. The communication then consists of sending $\gamma \in \mathbb{F}_p^n, \mathsf{QS}_0 \in \mathbb{F}_{p^r}^\tau, \mathsf{QS}_1 \in \mathbb{F}_{p^r}^\tau$. In total, the asymptotic communication cost is around $(n+2\cdot\tau\cdot r)\cdot\log p$ bits.

Proof. The correctness of the proof follows the explanation above. For security, we prove in the UC model where we construct simulators that are secure against a malicious prover and an honest verifier to argue soundness and zero-knowledge properties respectively.

Malicious Prover. Malicious prover. Sim emulates functionality $\mathcal{F}_{\mathsf{sVOLE}}$ and interacts with adversary $\mathcal{A}$ as follows:

- Sim emulates $\mathcal{F}_{\mathsf{sVOLE}}$ for $\mathcal{A}$ by choosing uniform $\mathbf{\Delta} \in \mathbb{F}_{p^r}^\tau$, and recording all the vector $\mathbf{u}$ and their corresponding MAC tags $\mathbf{V}$ that are received by $\mathcal{F}_{\mathsf{sVOLE}}$ from adversary $\mathcal{A}$. These values define the corresponding keys naturally. When emulating $\mathcal{F}_{\mathsf{sVOLE}}$, Sim also receives $\{\mathbf{A}_0^*, \mathbf{A}_1^*\} \in \mathbb{F}_{p^r}^\tau \times \mathbb{F}_{p^r}^\tau$ and can locally construct $\mathbf{B}^* = \mathbf{A}_0^* + \mathbf{A}_1^* \circ \mathbf{\Delta} \in \mathbb{F}_{p^r}^\tau$.
- When $\mathcal{A}$ sends $\gamma \in \mathbb{F}_p^n$ in step 1 of online phase, Sim extracts the witness as $x_i := \gamma_i + u_i for i \in [n]$.
- Sim executes the remaining part of protocol $\Pi_{\mathsf{MQ-DVZK}}$ as an honest verifier, using $\mathbf{\Delta}$ and the keys defined in the first step. If the honest verifier outputs false, then Sim sends $\mathbf{x} = \perp$ and $(f_i)_{i\in[m]}$ to $\mathcal{F}_{\mathsf{polyZK}}$ and aborts. If the honest verifier outputs true, Sim sends $\mathbf{x}$ and $(f_i)_{i\in[m]}$ to $\mathcal{F}_{\mathsf{polyZK}}$ where $\mathbf{x} = (x_1, \cdots, x_n)$ is extracted by Sim as above.

It is easy to see that the view of $\mathcal{A}$ simulated by Sim has an identical distribution as its view in the real-world execution. Whenever the honest verifier in the real-world execution outputs false, the honest verifier in the ideal-world execution outputs false as well (since Sim sends $\perp$ to $\mathcal{F}_{\mathsf{polyZK}}$ in this case). Therefore, we only need to bound the probability that the verifier in the real-world execution outputs true but the witness $\mathbf{x}$ sent by Sim to $\mathcal{F}_{\mathsf{polyZK}}$ satisfies that $f_i(\mathbf{x}) = 0$ for some $i \in [m]$. And this happens with probability at most $(m+2)/p^r$ [YSWW21] since we repeat τ times so the soundness is bounded by $((m+2)/p^r)^\tau$ using union bound.

Semi-honest Verifier. Sim emulates $\mathcal{F}_{\mathsf{polyZK}}$. If Sim receives false from $\mathbb{F}$, then it simply aborts. Otherwise, Sim interacts with $\mathcal{A}$ as follows:

- In the preprocessing phase, Sim emulates $\mathcal{F}_{\mathsf{sVOLE}}$, gets the global key $\mathbf{\Delta}$ and the keys $\mathbf{K}[\mathbf{x}]$ for all the authenticated values, which are received from $\mathcal{A}$. Additionally, S also receives $\mathbf{B}^* = \mathbf{A}_0^* + \mathbf{A}_1^* \circ \mathbf{\Delta} \in \mathbb{F}_{p^r}^\tau$.
- Sim executes the step 1 of online phase in $\Pi_{\mathsf{MQ-DVZK}}$ by sending uniform $\gamma \in \mathbb{F}_p^n$ to $\mathcal{A}$.
- For steps 5–6 of $\Pi_{\mathsf{MQ-DVZK}}$, Sim computes $\mathbf{W}, \mathbf{B}_i$ by using $\mathbf{\Delta}$, the keys $\mathbf{K}[\mathbf{x}]$ and $\mathbf{B}^*$ received from $\mathcal{A}$ following the protocol description, and then samples $\mathsf{QS}_1 \leftarrow_r \mathbb{F}_{p^r}^\tau$ and computing $\mathsf{QS}_0 := \mathbf{B} - \mathsf{QS}_1 \circ \mathbf{\Delta}$. Then, Sim sends $(\mathsf{QS}_0, \mathsf{QS}_1)$ to $\mathcal{A}$.
  Note that γ and $(\mathbf{A}_0^*, \mathbf{A}_1^*)$ are uniform and kept secret from the view of adversary $\mathcal{A}$. Therefore, we easily obtain that the view of $\mathcal{A}$ simulated by Sim is distributed identically to its view in the real-world execution, which concludes the proof.

4.2 Publicly-Verifier ZK for MQP

In this section, we show how to transfer our designated-verifier ZK protocol $\Pi_{\mathsf{MQ-DVZK}}$ (Figure 6) to publicly-verifier ZK protocol using SoftspokenOT and vector commitment in VOLE in-the-Head paradigm.

PARAMETERS:

- Given a field $\mathbb{F}_p$ and $\tau, r \in \mathbb{N}$, $\tau \in \mathbb{N}$ number of repetitions.
- Prover P and verifier V hold $(\mathbf{A}_i, \mathbf{b}_i, y_i)_{i \le m} \in \mathbb{F}_p^{n \times n} \times \mathbb{F}_p^n \times \mathbb{F}_p$.
- P holds $\mathbf{x} = (x_1, \cdots, x_n) \in \mathbb{F}_p^n$ such that $\mathbf{x}^T \mathbf{A}_i \mathbf{x} + \mathbf{b}_i^T \mathbf{x} = y_i$ for all $i \in [m]$.
- P and V define a set of polynomials $\{f_i\}_{i \le m}$ of degree 2 as $f_i(x_1, \cdots, x_n) = \mathbf{x}^T \mathbf{A}_i \mathbf{x} + \mathbf{b}_i^T \mathbf{x} - y_i$ over $\mathbb{F}_p$. Each polynomial f_i is presented as $f_i = f_{i,1} + f_{i,2}$ where all terms in $f_{i,1}$ and $f_{i,2}$ have degree of 1 and 2 respectively.
- An instantiation of $\mathcal{F}_{\mathsf{sVOLE}}$ for multi-subVOLE over $\mathbb{F}_p$.

PROTOCOL:

- **Preprocessing phase:**
  1. P and V invokes on input (init) to $\mathcal{F}_{\mathsf{sVOLE}}^{n+r\cdot\tau,\tau}$, P gets $(\mathbf{u}, \mathbf{V})$ where $\mathbf{V} = \begin{bmatrix}\mathbf{V}^1 \\ \mathbf{V}^2\end{bmatrix}, \mathbf{V}^1 \in \mathbb{F}_{p^r}^{n \times \tau}$, $\mathbf{V}^2 \in \mathbb{F}_{p^r}^{(r\cdot\tau)\times\tau}$.
     Note that the first n-coordinates of $\mathbf{u}$ are used to hide witness $\mathbf{x}$ and the last $(r \cdot \tau)$-coordinates of $\mathbf{u}$ are used to mask polynomials in the QS check i.e., using $\mathbf{u}[n, r \cdot \tau)$ and $\mathbf{V}_2$ to produce τ-OLEs over $\mathbb{F}_{2^\lambda}$ where P gets $\{\mathbf{A}_0^*, \mathbf{A}_1^*\} \in \mathbb{F}_{p^r}^\tau \times \mathbb{F}_{p^r}^\tau$.
- **Online phase:**
  1. P sends $\gamma := \mathbf{x} - \mathbf{u}[0, n) \in \mathbb{F}_p^n$ to V.
     P defines $\mathbf{V}^1 \to \mathbf{M}[\mathbf{x}]$.
  2. For $i \in [1, m]$, P defines a vector that consists of τ univariate 2-degree polynomials over field $\mathbb{F}_{p^r}$ as
     $$\mathbf{g}_i = f_{i,1}(\mathbf{M}[\mathbf{x}] + \mathbf{x} \cdot [1 \cdots 1] \cdot \mathsf{diag}(X)) \cdot X + f_{i,2}(\mathbf{M}[\mathbf{x}] + \mathbf{x} \cdot [1 \cdots 1] \cdot \mathsf{diag}(X)),$$
     and computes the coefficients $\{\mathbf{A}_{i,0}, \mathbf{A}_{i,1}, \mathbf{A}_{i,2}\} \in (\mathbb{F}_{p^r}^\tau)^3$ such that $\mathbf{g}_i = \mathbf{A}_{i,0} + \mathbf{A}_{i,1} \cdot X + \mathbf{A}_{i,2} \cdot X^2$. Note that $\mathbf{A}_{i,2} = 0$.
  3. V samples $\chi \leftarrow_r \mathbb{F}_{p^r}^\tau$ and sends it to P.
  4. P computes
     $$\mathsf{QS}_0 := \sum_{i=1}^{m} \mathbf{A}_{i,0} \circ \chi^i + \mathbf{A}_0^*,$$
     $$\mathsf{QS}_1 := \sum_{i=1}^{m} \mathbf{A}_{i,1} \circ \chi^i + \mathbf{A}_1^*.$$
     and sends them to V. Note $\mathbf{A}_0^*$, $\mathbf{A}_1^*$ is consider as vectors over $\mathbb{F}_{p^r}^\tau$.
  5. P and V invokes on input (get) to $\mathcal{F}_{\mathsf{sVOLE}}^{n+r\cdot\tau,\tau}$, V gets $(\boldsymbol{\Delta},\ \mathbf{W},\ \mathbf{B}^*)$ such that:
     $$\mathbf{W} = \begin{bmatrix}\mathbf{W}_1 \\ \mathbf{W}_2\end{bmatrix},\ \mathbf{W}_1 := \mathbf{V}_1 + \gamma \cdot [1 \cdots 1] \cdot \mathsf{diag}(\boldsymbol{\Delta}),$$
     $$\mathbf{W}_2 \ \longrightarrow \mathbf{B}^* = \mathbf{A}_0^* + \mathbf{A}_1^* \circ \boldsymbol{\Delta} \in \mathbb{F}_{p^r}^\tau.$$
     V defines $\mathbf{W}_1 \to \mathbf{K}[\mathbf{x}]$.
  6. For $i \in [1, m]$:
     - V computes $\mathbf{B}_i := f_{i,1}(\mathbf{K}[\mathbf{x}]) \circ \boldsymbol{\Delta} + f_{i,2}(\mathbf{K}[\mathbf{x}])$.
  7. P and V check that $\mathbf{A}_{i,0} + \mathbf{A}_{i,1} \circ \boldsymbol{\Delta} = \mathbf{B}_i$ in the following way:
     - V computes $\mathbf{B} = \sum_{i=1}^m \mathbf{B}_i \circ \chi^i + \mathbf{B}^*$ and checks that $\mathbf{B} = \mathsf{QS}_0 + \mathsf{QS}_1 \circ \boldsymbol{\Delta}$.
     - If the check fails, V outputs false; otherwise it outputs true.

Fig. 3. The DVZK protocol $\Pi_{\mathsf{MQ-DVZK}}$ for Multivariate Quadratic problem in the $\mathcal{F}_{\mathsf{sVOLE}}$-hybrid model

SoftspokenOT. Given $\mathsf{PRG} : \{0,1\}^\lambda \to \mathbb{F}_p^n$ be a pseudorandom generator. SoftspokenOT [Roy22] shows how to construct a subfield VOLE over the extension field $\mathbb{F}_{p^r}$ that securely realizes the ideal functionality of $\mathcal{F}_{\mathsf{sVOLE}}$ Figure 1.

In particular, assume P has a set of seeds $\{\mathsf{sd}_i\}_{i\in[N]}$ and V has an index $j \in [N]$ and a set of seeds all-but-one $\{\mathsf{sd}_i\}_{i\neq j}$. P and V now construct a VOLE over $\mathbb{F}_{p^r}$ by defining:

$$\mathbf{u} = \sum_{i=1}^{N} \mathsf{PRG}(\mathsf{sd}_i) \in \mathbb{F}_p^n,\ \mathbf{v} = -\sum_{i=1}^{N} i \cdot \mathsf{PRG}(\mathsf{sd}_i) \in \mathbb{F}_{p^r}^n$$
$$\mathbf{w} = \sum_{i\neq j} i \cdot \mathsf{PRG}(\mathsf{sd}_i) = j \cdot \mathbf{u} + \mathbf{v} \in \mathbb{F}_{p^r}^n$$

P and V repeats τ individual times to get τ-sVOLE correlations (the VOLE global key needs to contain enough entropy to ensure soundness), while P has $\{\mathsf{sd}_j^i\}$ for $i \in [0,\tau), j \in [0,N)$ and V has τ-set of all-but-one seeds $(\Delta_i, \{\mathsf{sd}_j^i\}_{j\neq\Delta_i})$ for $i \in [0,\tau)$ then the muli-instance sVOLE is defined by concatenating each instance.
P defines:

$$\mathbf{U} = \left[\sum_{j=1}^{N} \mathsf{PRG}(\mathsf{sd}_j^0) \cdots \sum_{j=1}^{N} \mathsf{PRG}(\mathsf{sd}_j^{\tau-1})\right],\ \mathbf{V} = \left[\sum_{j=1}^{N} j \cdot \mathsf{PRG}(\mathsf{sd}_j^0) \cdots \sum_{j=1}^{N} j \cdot \mathsf{PRG}(\mathsf{sd}_j^{\tau-1})\right]$$

While V defines:

$$\mathbf{W}' = \left[\sum_{j=1}^{N} (j-\Delta_0) \cdot \mathsf{PRG}(\mathsf{sd}_j^0) \cdots \sum_{j=1}^{N} (j-\Delta_{\tau-1}) \cdot \mathsf{PRG}(\mathsf{sd}_j^{\tau-1})\right]$$

To instantiate $\mathcal{F}_{\mathsf{sVOLE}}$, P needs to re-randomize $\mathbf{U}$ by sending $\mathbf{C} := [\mathbf{U}_1 - \mathbf{u} \| \cdots \| \mathbf{U}_{\tau-1} - \mathbf{u}] \in \mathbb{F}_p^{n\times(\tau-1)}$ where $\mathbf{u} := \mathbf{U}_0$. V then defines $\mathbf{W} = \mathbf{W}' + [0\|\mathbf{C}] \cdot \mathsf{diag}(\boldsymbol{\Delta})$. Finally we get

$$\mathbf{W} = \mathbf{V} + \mathbf{u} \cdot [1 \ldots 1] \cdot \mathsf{diag}(\boldsymbol{\Delta}).$$

VOLE consistency check. To make sure that P does not cheat when sending $\mathbf{C}$. The V challenges P to open a random, linear universal hash function applied to $\mathbf{U}_0$ and $\mathbf{V}$. The linear hash function is represented by a compressing matrix $\mathsf{H}^{\mathsf{UHF}}$, and P sends

$$\tilde{\mathbf{u}} = \mathsf{H}^{\mathsf{UHF}} \cdot \mathbf{u},\ \tilde{\mathbf{V}} = \mathsf{H}^{\mathsf{UHF}} \cdot \mathbf{V}$$

and then V checks $\tilde{\mathbf{V}} + \tilde{\mathbf{u}} \cdot [1 \cdots 1] \cdot \mathsf{diag}(\boldsymbol{\Delta}) = \mathsf{H}^{\mathsf{UHF}} \cdot (\mathbf{W} + [0\|\mathbf{C}] \cdot \mathsf{diag}(\boldsymbol{\Delta}))$.

Multi-instance PPRF. The purpose of using PPRF is to allow P and V to efficiently open all but one sd to V following the same technique as in MPCitH signatures [BCC+24]. Currently, there are two approaches to optimizing the GGM tree based PPRF 1) using the half-tree technique and circular correlation robust hash function [CLY+24, BCdSG24] to optimize in terms of computation, 2) constructing new multi-PPRF based on AES in ideal cipher [BCC+24]. In our construction, we use the multi-instance of PPRF to easily plug into our signature. We recall the construction PPRF and muti-instance of PRG in Figure 5 and Figure 4 respectively and followed by their formal theorems of security.

PARAMETERS:

- For each $K \in \{0,1\}^\lambda$, $\pi_K : \{0,1\}^\lambda \to \{0,1\}^\lambda$ is a uniformly random permutation.

CONSTRUCTION:

- Sample $\mathsf{K} \leftarrow_r \{0,1\}^{2\lambda}$. parse $\mathsf{K} := (K_0, K_1)$.
- $\mathsf{F}_b : \{0,1\}^{2\lambda} \to \{0,1\}^\lambda$ is defined as $\mathsf{F}_b(\mathsf{sd}, \mathsf{K}_b) = \pi_{K_b}(\mathsf{sd}) \oplus \mathsf{sd}$ for $b \in \{0,1\}$ and $\mathsf{sd} \in \{0,1\}^\lambda$.

Fig. 4. Multi-instance PRG $\mathsf{F}_0, \mathsf{F}_1$ in the ideal cipher model

Theorem 11. *Let* $\mathsf{F}_0, \mathsf{F}_1$ *be the functions defined in Figure 4. Let* q *be the number of queries to the oracle* $\mathcal{O}_\pi$ *(ideal cipher model). Then* $(\mathsf{F}_0, \mathsf{F}_1)$ *is an* (N, τ)*-instance* (q, ϵ)*-secure* PRG *in the ideal cipher model, where*

$$\varepsilon \leq f_N(\lambda) \cdot q \cdot \left(\frac{1}{2^{\lambda-1}} + \frac{1}{2^\lambda - q}\right) + \frac{4\tau N}{2^{2\lambda}},$$

for some function f_N *such that if* $N \leq 2^{\lambda-1}$, $f_N(\lambda) \leq \frac{3\tau\lambda \cdot \ln 2}{\ln \lambda + \ln \ln 2}$, *and if* $N \leq 2^{\lambda/2}$, $f_N(\lambda) \leq 4\tau$.

PARAMETERS:

- Two functions $\mathsf{F}_0, \mathsf{F}_1 : \{0,1\}^{2\lambda} \to \{0,1\}^\lambda$ is a (N, τ)-instance (t, ϵ)-secure length-doubling PRG.
- Number of leaves $N = 2^D \in \mathbb{N}$, computational security parameter λ.

CONSTRUCTION:

- Sample $(\mathsf{sd}, \mathsf{K}) \leftarrow_r \{0,1\}^{3\lambda}$ where $\mathsf{K} := (\mathsf{K}_0, \mathsf{K}_1)$. We use $\mathsf{K}_0, \mathsf{K}_1$ for $\mathsf{F}_0, \mathsf{F}_1$ respectively. For simplicity, we sometimes write $\mathsf{F}_i(\mathsf{sd}, \mathsf{K}_i)$ as $\mathsf{F}_i(\mathsf{sd}, \mathsf{K})$ for $i \in \{0,1\}$.
- Let $X_0 := \mathsf{F}_0(\mathsf{sd}, \mathsf{K}_0)$, $X_1 := \mathsf{F}_1(\mathsf{sd}, \mathsf{K}_1)$.
- For $i \in [2, D]$, define $X_{b_1,\ldots,b_{i-1},0} = \mathsf{F}_0(\mathsf{F}_{b_{i-1}}(X_{b_1,\ldots,b_{i-1}}), \mathsf{K}_0)$, $X_{b_1,\ldots,b_{i-1},1} = \mathsf{F}_1(\mathsf{F}_{b_{i-1}}(X_{b_1,\ldots,b_{i-1}}), \mathsf{K}_1)$ where $b_j \in \{0,1\}$ for all $j \in [1, i-1]$.
- We generalize the formula to compute the leaf of the tree as follows:
  For each $i \in [0, n-1]$, bit-decompose i as $\sum_{j=1}^D 2^{j-1} \cdot i_j$ for $i_j \in \{0,1\}$ then:

$$\begin{aligned} X_i = X_{i_1,\ldots,i_D} &= \mathsf{F}_{i_D}(\mathsf{F}_{i_{D-1}}(X_{i_1,\ldots,i_{D-1}}), \mathsf{K}_{i_D}) \\ &= \mathsf{F}_{i_D}(\mathsf{F}_{i_{D-1}}(\ldots(\mathsf{F}_{i_1}(\mathsf{sd}_{i_1}, \mathsf{K}_{i_1}), \mathsf{K}_{i_{D-1}}), \mathsf{K}_{i_D}) \end{aligned}$$

  To formalize, the value for each leaf $i \in [0, N-1]$ is denoted as:

$$\begin{aligned} \mathsf{PPRF}_{\mathsf{sd}}(\mathsf{K}, i) &= \mathsf{F}_{i_D}\left(\mathsf{PPRF}_{\mathsf{sd}}\left(\mathsf{K}, i_{1,\ldots,D-1}\right), \mathsf{K}\right) \\ &= \mathsf{F}_{i_D}\left(\mathsf{F}_{i_{D-1}}\left(\ldots\left(\mathsf{F}_{i_1}(\mathsf{sd}, \mathsf{K}), \mathsf{K}\right), \mathsf{K}\right)\right) \end{aligned}$$

  where $i_{1,\ldots,k} = \sum_{j=1}^k 2^{k-j} i_j$ for any $k \in [1, D]$.
- We define the co-path $\mathsf{CoPath}(i)$ for each $i = \sum_{j=1}^D 2^{j-1} \cdot i_j \in [0, N-1]$ as follows:

$$\mathsf{CoPath}(i) = \mathsf{CoPath}(X_{i_1,\ldots,i_D}) = \{X_{\bar{i}_1}, X_{i_1,\bar{i}_2}, \ldots, X_{i_1,\ldots,\bar{i}_D}\}$$

  Formalizing, we have:

$$\mathsf{CoPath}_{\mathsf{sd}}(\mathsf{K}, i) = \mathsf{PPRF}_{\mathsf{sd}}\left(\mathsf{K}, i_{1,\ldots,\bar{j}}\right)_{j=1,\ldots,D}$$

  where $i_{1,\ldots,\bar{k}} = \sum_{j=1}^{k-1} 2^{k-j} . i_j + \bar{i}_k$ for any $k \in [1, D]$.

Fig. 5. Construction $\mathsf{PPRF}(\mathsf{sd}, \mathsf{K}, 2^D)$ of Puncturable PRF

Theorem 12 (PPRF security [BCC+24]). *Assume that* $\mathsf{PRG} = (\mathsf{F}_0, \mathsf{F}_1)$ *with* $\mathsf{F}_b : \{0,1\}^{2\lambda} \to \{0,1\}^\lambda$ *is an* (N, τ)*-instance* (t, ϵ)*-secure length-doubling PRG. Then the construction* $\mathsf{PPRF}(\mathsf{sd}, \mathsf{K}, 2^D)$ *described in Figure 5 is an* (N, τ)*-instance strongly* $(t, D \cdot \epsilon)$*-secure PPRF with input domain* $[2^D]$ *and punctured key domain* $(\{0,1\}^\lambda)^D$.

VOLE in-the-Head. Putting all techniques together and using the compiler from [BBD+23], we obtain a publicly verifier ZK protocol $\Pi_{\mathsf{MQ-PVZK}}$ from MQ problem in Figure 6 based on SoftspokenOT, multi-instance PPRF, and nullity check for a polynomial set.

5 A Signature scheme from Multivariate Quadratic

In this section, we introduce a new signature scheme from the multivariate quadratic decoding assumption. A signature scheme is given by three algorithms $(\mathsf{KeyGen}, \mathsf{Sign}, \mathsf{Verify})$. KeyGen returns a

PARAMETERS:

- Prover P and verifier V hold $(\mathbf{A}_i, \mathbf{b}_i, y_i)_{i\le m} \in \mathbb{F}_p^{n\times n} \times \mathbb{F}_p^n \times \mathbb{F}_p$.
- P holds $\mathbf{x} = (x_1, \cdots, x_n) \in \mathbb{F}_p^n$ such that $\mathbf{x}^T\mathbf{A}_i\mathbf{x} + \mathbf{b}_i^T\mathbf{x} = y_i$ for all $i \in [m]$.
- P and V define a set of polynomials $\{f_i\}_{i\le m}$ of degree 2 as $f_i(x_1, \cdots, x_n) = \mathbf{x}^T\mathbf{A}_i\mathbf{x} + \mathbf{b}_i^T\mathbf{x}$ over $\mathbb{F}_p$. Each polynomial f_i is presented as $f_i = f_{i,1} + f_{i,2}$ where all terms in $f_{i,1}$ and $f_{,2}$ have degree of 1 and 2 respectively.
- Let $k = n + r, N = p^r$ and $\mathcal{H}^{\mathsf{UHF}} \subseteq \mathbb{F}_p^{t\times(k+t)}$ be a family of k-hiding, p^{-t}-universal hash function.
- Let $\mathsf{PRG} : \{0,1\}^\lambda \to \mathbb{F}_p^{k+t}$ be a pseudorandom generator.

PROTOCOL:

1. P uniformly samples $\{\mathsf{sd}_j^i\} \leftarrow_r \{0,1\}^\lambda$ for $i \in [0,\tau), j \in [0,N)$. The P defines $\mathbf{U}, \mathbf{V} \in \mathbb{F}_p^{(k+h)\times\tau}$ as follow
$$\mathbf{U} = \left[\sum_{j=1}^{N}\mathsf{PRG}(\mathsf{sd}_j^0)\cdots\sum_{j=1}^{N}\mathsf{PRG}(\mathsf{sd}_j^{\tau-1})\right],\ \mathbf{V} = -\left[\sum_{j=1}^{N} j\cdot\mathsf{PRG}(\mathsf{sd}_j^0)\cdots\sum_{j=1}^{N} j\cdot\mathsf{PRG}(\mathsf{sd}_j^{\tau-1})\right]$$
P defines $\mathbf{u} = \mathbf{U}_0$ and $\mathbf{C} := [\mathbf{U}_1 - \mathbf{u}\|\cdots\|\mathbf{U}_{\tau-1} - \mathbf{u}] \in \mathbb{F}_p^{(k+t)\times(\tau-1)}$. P sends $\mathbf{C}$ to V.
2. V samples $\mathsf{H}^{\mathsf{UHF}} \leftarrow_r \mathcal{H}^{\mathsf{UHF}}$ and sends it to P.
3. P defines $\tilde{\mathbf{u}} = \mathsf{H}^{\mathsf{UHF}}\cdot\mathbf{u}, \tilde{\mathbf{V}} = \mathsf{H}^{\mathsf{UHF}}\cdot\mathbf{V}$ and sends them to P.
4. P and V run step $1-4$ of $\Pi_{\mathsf{MQ-DVZK}}$ in Figure 3 by using first k-coordinates of $\mathbf{u}$ and k-rows of $\mathbf{V}$.
5. V samples randomly $\boldsymbol{\Delta} = (\Delta_0, \cdots, \Delta_{\tau-1})$, sends $(\mathsf{get}, \boldsymbol{\Delta})$ to P and receives back $\{\mathsf{sd}_j^i\}$ for $i \in [0,\tau), j \in [0,N] \setminus \{\Delta_i\}$. Then compute
$$\mathbf{W} = \left[\sum_{j=1}^{N}(j-\Delta_0)\cdot\mathsf{PRG}(\mathsf{sd}_j^0)\cdots\sum_{j=1}^{N}(j-\Delta_{\tau-1})\cdot\mathsf{PRG}(\mathsf{sd}_j^{\tau-1})\right]$$
6. P and V run step $5-6$ of $\Pi_{\mathsf{MQ-DVZK}}$ in Figure 3.
7. V outputs accept if two following checks are passed:
   - **SoftSpokenOT.** $\tilde{\mathbf{V}} + \tilde{\mathbf{u}}\cdot[1\cdots1]\cdot\mathsf{diag}(\boldsymbol{\Delta}) = \mathsf{H}^{\mathsf{UHF}}\cdot(\mathbf{W} + [0\|\mathbf{C}]\cdot\mathsf{diag}(\boldsymbol{\Delta}))$.
   - **QuickSliver.** P and V run step 7 of $\Pi_{\mathsf{MQ-DVZK}}$ in Figure 3.

Fig. 6. The publicly verifiable zero-knowledge protocol $\Pi_{\mathsf{MQ-PVZK}}$ for Multivariate Quadratic problem in the $\mathcal{F}_{\mathsf{sVOLE}}$-hybrid model

key pair $(\mathsf{pk}, \mathsf{sk})$ where pk and sk are the public and private key. Sign on an input a message m and the secret key sk, produces a signature σ. Verify, on input a message m, a public key pk and a signature σ, returns 0 or 1. Standard security notions for signature schemes are existential unforgeability against key-only attacks ($\mathsf{EUF\text{-}KO}$, Definition 14) and against chosen-message attacks ($\mathsf{EUF\text{-}CMA}$, Definition 13).

Definition 13 (EUF-CMA security). *Given a signature scheme* $\mathsf{Sig} = (\mathsf{Setup}, \mathsf{Sign}, \mathsf{Verify})$ *and security parameter* λ*, we say that* Sig *is* $\mathsf{EUF\text{-}CMA}$*-secure if any PPT algorithm* $\mathcal{A}$ *has negligible advantage in the* $\mathsf{EUF\text{-}CMA}$ *game, defined as*
$$\mathsf{Adv}_{\mathcal{A}}^{\mathsf{EUF\text{-}CMA}} = \Pr\left[\begin{array}{c}\mathsf{Verify}(\mathsf{pk},\mu^*,\sigma^*) = 1\\ \wedge\mu^* \notin Q\end{array}\middle|\begin{array}{c}(\mathsf{sk},\mathsf{pk}) \leftarrow \mathsf{Setup}(\{0,1\}^\lambda)\\ (\mu^*,\sigma^*) \leftarrow \mathcal{A}^{\mathsf{Sign}(\mathsf{sk},\cdot)}(\mathsf{pk})\end{array}\right],$$
where $\mathcal{A}^{\mathsf{Sign}(\mathsf{sk},\cdot)}$ *denotes* $\mathcal{A}$*'s access to a signing oracle with private key* sk *and* Q *denotes the set of messages* μ *that were queried to* $\mathsf{Sign}(\mathsf{sk},\cdot)$ *by* $\mathcal{A}$*.*

Definition 14 (EUF-KO security). *Given a signature scheme* $\mathsf{Sig} = (\mathsf{Setup}, \mathsf{Sign}, \mathsf{Verify})$ *and security parameter* λ*, we say that* Sig *is* $\mathsf{EUF\text{-}KO}$*-secure if any PPT algorithm* $\mathcal{A}$ *has negligible advantage in the EUF-KO game, defined as*
$$\mathsf{Adv}_{\mathcal{A}}^{\mathsf{EUF\text{-}KO}} = \Pr\left[\mathsf{Verify}(\mathsf{pk},\mu^*,\sigma^*) = 1\middle|\begin{array}{c}(\mathsf{sk},\mathsf{pk}) \leftarrow \mathsf{Setup}(\{0,1\}^\lambda)\\ (\mu^*,\sigma^*) \leftarrow \mathcal{A}(\mathsf{pk})\end{array}\right].$$

5.1 Description of the Signature Scheme

The key generation algorithm randomly samples a multivariate quadratic instance $((\mathbf{A}_i, \mathbf{b}_i, y_i)_{i\leq m})$ with solution $\mathbf{x} \in \mathbb{F}_p^n$. We describe it on Figure 7. The signing algorithm with secret key $\mathsf{sk} = (\mathsf{sd}, \mathbf{x})$ and message $\mathsf{m} \in \{0,1\}^*$ is described on Figure 8. The verification algorithm with public key $\mathsf{pk} = (\mathbf{A}_i, \mathbf{b}_i, y_i)_{i\leq m}$, message $\mathsf{m} \in \{0,1\}^*$, and signature σ, is described in Figure 9.

Inputs:

- A security parameter λ, a finite field $\mathbb{F}_p$.
- Pseudorandom generators $\mathsf{PRG}_1 : \{0,1\}^\lambda \to (\mathbb{F}_p^{n\times n})^m, \mathsf{PRG}_2 : \{0,1\}^\lambda \to (\mathbb{F}_p^n)^m$.

Key Gen:

1. Sample $\mathsf{sd} \leftarrow_r \{0,1\}^\lambda$. Compute $(\mathbf{A}_i)_{i\leq m} \leftarrow \mathsf{PRG}_1(\mathsf{sd})$, $(\mathbf{b}_i)_{i\leq m} \leftarrow \mathsf{PRG}_2(\mathsf{sd})$
2. Sample $\mathbf{x} \leftarrow_r \mathbb{F}_p^n$.
3. Set $y_i \leftarrow \mathbf{x}^T\mathbf{A}_i\mathbf{x} + \mathbf{b}_i^T\mathbf{x}$.
4. Output $\mathsf{pk} \leftarrow (\mathbf{A}_i, \mathbf{b}_i, y_i)_{i\leq m}$ and $\mathsf{sk} \leftarrow (\mathsf{sd}, \mathbf{x})$.

Fig. 7. Key generation algorithm of the signature scheme

Theorem 15. *Assume that* PPRF *is a* (q_s, τ)*-instance* $(t, \epsilon_{\mathsf{PPRF}})$*-secure PPRF, that* PRG *is a* (q_s, τ)*-instance* $(t, \epsilon_{\mathsf{PRG}})$*-secure PRG, and that any adversary running in time* t *has at advantage at most* ϵ_{MQ} *against the multivariate quadratic problem;* $\Pi_{\mathsf{MQ-PVZK}}$ *is a DVZK protocol with a soundness error of* $\epsilon_{\Pi_{\mathsf{MQ-PVZK}}}$*. Model the hash functions* $\mathsf{H}_1, \mathsf{H}_2$ *as random oracles with output of length* 2λ*-bit and the pseudorandom generator* PRG_2^* *as a random oracle. Then chosen-message adversary against the signature scheme depicted in Figure 8, running in time* t*, making* q_s *signing queries, and making* q_1, q_2, q_3 *queries, respectively, to the random oracles* $\mathsf{H}_1, \mathsf{H}_2$ *and* PRG_2^**, succeeds in outputting a valid forgery with probability*

$$\Pr[\text{Forge}] \leq \frac{q_s\,(q_s + q_1 + q_2 + q_3)}{2^{2\lambda}} + \epsilon_{\mathsf{PPRF}} + \varepsilon_{\mathsf{PRG}} + \epsilon_{\mathsf{MQ}} + \epsilon_{\Pi_{\mathsf{MQ-PVZK}}},$$

We start by proving the following lemma:

Lemma 16 ($\mathsf{EUF\text{-}KO} \implies \mathsf{EUF\text{-}CMA}$).

$$\mathsf{Adv}^{\mathsf{EUF\text{-}CMA}}_{\mathcal{A}} \leq \mathsf{Adv}^{\mathsf{EUF\text{-}KO}}_{\mathcal{A}} + \frac{q_s\,(q_s + q_1 + q_2 + q_3)}{2^{2\lambda}} + \epsilon_{\mathsf{PPRF}} + \epsilon_{\mathsf{PRG}}$$

Proof. Let us consider an adversary $\mathcal{A}$ against the $\mathsf{EUF\text{-}CMA}$ property of the signature scheme. To prove security we will define a sequence of experiments involving $\mathcal{A}$, where the first corresponds to the experiment in which $\mathcal{A}$ interacts with the real signature scheme, and the last one is an experiment in which $\mathcal{A}$ is using only a random element independent from the witness.

Game 1 (Gm^1). This corresponds to the actual interaction of $\mathcal{A}$ with the real signature scheme. We need to bound the probability of what we'll call Forge, i.e. the event that $\mathcal{A}$ can generate a valid signature for a message that was not previously queried to the signing oracle.

Game 2 (Gm^2). For this step, we abort if the sampled salt K collides with the value sampled in any of the previous queries to hash functions H_1 or H_2 or if the input of PRG_2^* collides with the value obtained in any of the previous queries. Therefore we can bound this probability by

$$|\Pr[\mathsf{Gm}^1(\mathsf{Forge})] - \Pr[\mathsf{Gm}^2(\mathsf{Forge})]| \leq \frac{q_s \cdot (q_s + q_1 + q_2 + q_3)}{2^{2\lambda}}$$

Game 3 (Gm^3). The difference with the previous game is that now before signing a message we choose uniformly random values h_1, h_2 and $\mathbf{\Delta}^*$. Since Phase1, Phase2 and Phase5 are computed as before and the only change compared to the previous game is that we set the output of H_1 as h_1, the output of H_2 as h_2 and the output of $\mathsf{PRG}_2^*(h_2)$ as $\mathbf{\Delta}^*$ then the difference in forgery probability

Inputs: A secret key sk and a message $\mathsf{m} \in \{0,1\}^*$.
Parameters:

- $\mathsf{PRG}^* : \{0,1\}^\lambda \to \mathbb{F}_p^{k+t}$, $\mathsf{PRG}_1^* : \{0,1\}^{2\lambda} \to \mathbb{F}_p^{t\times(k+t)}$, $\mathsf{PRG}_2^* : \{0,1\}^{2\lambda} \to \mathbb{F}_{p^r}^{\tau}$.
- $\mathsf{PRG} : \{0,1\}^\lambda \to \{0,1\}^{2\lambda}$, $\mathsf{H}_1, \mathsf{H}_2 : \{0,1\}^* \to \{0,1\}^{2\lambda}$

Initialization.

- Parse sk as $(\mathsf{sd}, \mathbf{x})$;
- Let $(\mathbf{A}_i)_{i\le m} \leftarrow \mathsf{PRG}_1(\mathsf{sd})$, $(\mathbf{b}_i)_{i\le m} \leftarrow \mathsf{PRG}_2(\mathsf{sd})$;
- Set $y_i \leftarrow \mathbf{x}^T\mathbf{A}_i\mathbf{x} + \mathbf{b}_i^T\mathbf{x}$.
- Sample $(K_0, K_1) \leftarrow_r \{0,1\}^\lambda \times \{0,1\}^\lambda$. Set $\mathsf{K} \leftarrow (K_0, K_1)$.

Phase 1. Establish $\tau - \mathsf{PPRF}$:
For each iteration $e \in [\tau]$:

- Sample $\mathsf{sd}^e \leftarrow_r \{0,1\}^\lambda$;
- **For** $i = 0$ **to** $N-1$:
  1. Compute $\mathsf{sd}_i^e \leftarrow \mathsf{PPRF}_\mathsf{K}(\mathsf{sd}^e, i)$; // Can be computed efficiently by always storing the path to the current node: to move from i to $i+1$, start from the closest ancestor of $i+1$ in the path to leave i.
  2. $(\mathsf{sd}_i^e, \mathsf{com}_i^e) \leftarrow \mathsf{PRG}(\mathsf{sd}_i^e)$; // $(\mathsf{sd}_i^e, \mathsf{com}_i^e) \in \{0,1\}^\lambda \times \{0,1\}^\lambda$.

Phase 2.

- Define $\mathbf{U}, \mathbf{V} \in \mathbb{F}_p^{(k+h)\times\tau}$ as follow:

$$\mathbf{U} = \left[\sum_{i=0}^{N-1} \mathsf{PRG}^*(\mathsf{sd}_i^0) \cdots \sum_{i=0}^{N-1} \mathsf{PRG}^*(\mathsf{sd}_i^{\tau-1})\right], \ \mathbf{V} = -\left[\sum_{i=0}^{N-1} j \cdot \mathsf{PRG}^*(\mathsf{sd}_i^0) \cdots \sum_{i=0}^{N-1} i \cdot \mathsf{PRG}^*(\mathsf{sd}_i^{\tau-1})\right]$$

- Define $\mathbf{u} := \mathbf{U}_0$ and $\mathbf{C} := [\mathbf{U}_1 - \mathbf{u} \| \cdots \| \mathbf{U}_{\tau-1} - \mathbf{u}] \in \mathbb{F}_p^{(k+t)\times(\tau-1)}$.

Phase 3.

1. $h_1 \leftarrow \mathsf{H}_1(\mathsf{m}, \mathsf{K}, \mathbf{C}, \mathsf{com}_0^0, \cdots, \mathsf{com}_{N-1}^{\tau-1}, \cdots, \mathsf{com}_0^\tau, \cdots, \mathsf{com}_{N-1}^{\tau-1})$; // Accumulate the commitments inside the hash rather than storing and hashing all at once.
2. $\mathsf{H}^{\mathsf{UHF}} \leftarrow \mathsf{PRG}_1^*(h_1)$;
3. Define $\tilde{\mathbf{u}} = \mathsf{H}^{\mathsf{UHF}} \cdot \mathbf{u}, \tilde{\mathbf{V}} = \mathsf{H}^{\mathsf{UHF}} \cdot \mathbf{V}$.

Phase 4.

1. Run step $1-4$ of $\Pi_{\mathsf{MQ-DVZK}}$ in Figure 3 by using first k-coordinates of $\mathbf{u}$ and k-rows of $\mathbf{V}$, get $(\mathbf{U}_0, \mathbf{U}_1) \in \mathbb{F}_{p^r}^{\tau}$;
2. $h_2 \leftarrow \mathsf{H}_2(\mathsf{m}, \mathsf{K}, h_1, \tilde{\mathbf{u}}, \tilde{\mathbf{V}}, \mathsf{QS}_0, \mathsf{QS}_1)$;
3. $\boldsymbol{\Delta} \leftarrow \mathsf{PRG}_2^*(h_2)$, define $\boldsymbol{\Delta} = (\Delta_0, \cdots, \Delta_{\tau-1})$.

Phase 5.
Output $\sigma = \left(\mathsf{K}, h_1, h_2, (\mathsf{QS}_0, \mathsf{QS}_1), (\tilde{\mathbf{u}}, \tilde{\mathbf{V}}), (\mathsf{CoPath}_\mathsf{K}(\Delta_e, \mathsf{sd}^e), \mathsf{com}_{i^e}^e)_{e<\tau}\right)$. // for $\tilde{\mathbf{V}}$, only need to send its collision-resistant hash value.

Fig. 8. Signing algorithm of the signature scheme

is due to the event that query to H_1, H_2 or PRG_2^* was ever made before but in this scenario Game 2 aborts, so

$$\Pr[\mathsf{Gm}^2(\mathsf{Forge})] = \Pr[\mathsf{Gm}^3(\mathsf{Forge})]$$

Game 4 (Gm^4) In this game we sample at random the Δ_i^*-th seed $\mathsf{sd}_{\Delta_i^*}$ and the related co-path $\mathsf{CoPath}_{\Delta_i^*}$. By using all the seeds $\{\mathsf{sd}_i\}_{i\neq i\Delta_i^*}$ in the $\mathsf{CoPath}_{\Delta_i^*}$ we will proceed by computing all the parties' views as well as the auxiliary material. Therefore, Phase 1 and Phase 3 are executed in the actual way (i.e. by using the real witness) except for Δ_i^*, for which the values are obtained randomly instead of using the PPRF. Distinguishing between this game and the previous one is perfectly equivalent to breaking the multi-instance security of the PPRF:

$$|\Pr[\mathsf{Gm}^4(\mathsf{Forge})] - \Pr[\mathsf{Gm}^6(\mathsf{Forge})]| \le \epsilon_{\mathsf{PPRF}}$$

Inputs: A public key $\mathsf{pk} = (\mathbf{A}_i, \mathbf{b}_i, y_i)_{i\le m}$, a message $\mathsf{m} \in \{0,1\}^*$ and a signature σ.

1. Split the signature as follows:
$$\left(\mathsf{K}, h_1, h_2, (\mathsf{QS}_0, \mathsf{QS}_1), (\tilde{\mathbf{u}}, \tilde{\mathbf{V}}), (\mathsf{CoPath}_{\mathsf{K}}(\Delta_e, \mathsf{sd}^e), \mathsf{com}^e_{i^e})_{e<\tau}\right);$$
2. Recompute $\mathsf{H}^{\mathsf{UHF}} \leftarrow \mathsf{PRG}_1^*(h_1)$ via a pseudorandom generator using h_1.
3. Recompute $\boldsymbol{\Delta} = (\Delta_0, \cdots, \Delta_{\tau-1})$ via a pseudorandom generator using h_2.
4. For each iteration $e \in [\tau]$,
   - For each $i \in [N] \setminus \{\Delta_e\}$:
     - Recompute sd_i^e from the $\mathsf{CoPath}_{\mathsf{K}}(i^e, \mathsf{sd}^e)$;
     - Recompute $(\mathsf{sd}_i^e, \mathsf{com}_i^e) \leftarrow G(\mathsf{sd}_i^e)$;
5. Compute:
$$\mathbf{W} = \left[\sum_{i=0}^{N-1}(i-\Delta_0)\cdot\mathsf{PRG}(\mathsf{sd}_i^0)\cdots\sum_{i=0}^{N-1}(i-\Delta_{\tau-1})\cdot\mathsf{PRG}(\mathsf{sd}_i^{\tau-1})\right]$$
6. Run step $5-7$ of $\Pi_{\mathsf{MQ-DVZK}}$ in Figure 3. If the output is accept then check if
$$\tilde{\mathbf{V}} + \tilde{\mathbf{u}}\cdot[1\cdots1]\cdot\mathsf{diag}(\boldsymbol{\Delta}) = \mathsf{H}^{\mathsf{UHF}}\cdot(\mathbf{W} + [0\|\mathbf{C}]\cdot\mathsf{diag}(\boldsymbol{\Delta}))$$
7. Check if $h_1 \leftarrow \mathsf{H}_1(\mathsf{m}, \mathsf{K}, \mathbf{C}, \mathsf{com}_0^0, \cdots, \mathsf{com}_{N-1}^{\tau-1}, \cdots, \mathsf{com}_0^{\tau}, \cdots, \mathsf{com}_{N-1}^{\tau-1})$;
8. Check if $h_2 \leftarrow \mathsf{H}_2(\mathsf{m}, \mathsf{K}, h_1, \tilde{\mathbf{u}}, \tilde{\mathbf{V}}, \mathbf{U}_0, \mathbf{U}_1)$;
9. Output ACCEPT if three conditions are satisfied.

Fig. 9. Verification algorithm of the signature scheme

Game 5 (Gm^5). Now, before signing a message, we choose a uniformly random value to be used as the Δ_i^*- th party's view, i.e. $\mathsf{sd}_{\Delta_i^*}$, and its commitment $\mathsf{com}_{\Delta_i^*}$. Since in the previous game, these values were computed by using a multi-instance PRG on a random sd, with salt K, we can bound

$$|\Pr[\mathsf{Gm}^4(\mathsf{Forge})] - \Pr[\mathsf{Gm}^6(\mathsf{Forge})]| \le \epsilon_{\mathsf{PRG}}$$

Game 6 (Gm^6) In this game, we will change Phase 4 by making the signer use the simulator against the semi-honest verifier described in Theorem 10. We have

$$\Pr[\mathsf{Gm}^5(\mathsf{Forge})] = \Pr[\mathsf{Gm}^6(\mathsf{Forge})]$$

Game 7 ($\mathsf{Adv}_{\mathcal{A}}^{\mathsf{EUF\text{-}KO}}$) We say that an execution e^* of a query

$$h_2 \leftarrow \mathsf{H}_2(\mathsf{m}, \mathsf{K}, h_1, \tilde{\mathbf{u}}, \tilde{\mathbf{V}}, \mathsf{QS}_0, \mathsf{QS}_1)$$

defines a correct witness if the following criteria are satisfied:

- h_1 was output by a previous query
$$h_1 \leftarrow \mathsf{H}_1(\mathsf{m}, \mathsf{K}, \mathbf{C}, \mathsf{com}_0^0, \cdots, \mathsf{com}_{N-1}^{\tau-1}, \cdots, \mathsf{com}_0^{\tau}, \cdots, \mathsf{com}_{N-1}^{\tau-1})$$
- each $\mathsf{com}_i^{e^*}$ in this query was output by a previous query
$$(\mathsf{sd}_i^{e^*}, \mathsf{com}_i^{e^*}) \leftarrow \mathsf{PRG}(\mathsf{sd}_i^{e^*})$$
for each $i \in [N]$;
- The vectors $\tilde{\mathbf{u}}, \tilde{\mathbf{V}}$ is defined such that
$$\tilde{\mathbf{V}} + \tilde{\mathbf{u}}\cdot[1\cdots1]\cdot\mathsf{diag}(\boldsymbol{\Delta}) = \mathsf{H}^{\mathsf{UHF}}\cdot(\mathbf{W} + [0\|\mathbf{C}]\cdot\mathsf{diag}(\boldsymbol{\Delta}))$$
- The vector $\mathbf{x}$ defined by $\mathsf{QS}_0, \mathsf{QS}_1$ satisfies $\mathsf{MQ}_{p,m,n}$.

In this game, for each query of H_2 made by the adversary, we will check if there is an execution e^* that defines a correct witness. Calling this event Solve then $\Pr[\mathsf{Solve}] \le \epsilon_{\mathsf{SD}} + \epsilon_{\Pi_{\mathsf{MQ-PVZK}}}$, since if it occurs then $\mathsf{QS}_0, \mathsf{QS}_1$ define a solution for the $\mathsf{MQ}_{p,m,n}$. In the end, we obtain

$$\Pr[\text{Forge}] \le \frac{q_s\,(q_s + q_1 + q_2 + q_3)}{2^{2\lambda}} + \epsilon_{\mathsf{PPRF}} + \varepsilon_{\mathsf{PRG}} + \epsilon_{\mathsf{MQ}} + \epsilon_{\Pi_{\mathsf{MQ-PVZK}}}$$

□

5.2 Parameters and Signature Size

Signature size. The signer generates the signature σ which consists of:

- The key $\mathsf{K} \in \{0,1\}^{2\lambda}$ for multi-instances PPRF, 2 hash values $h_1, h_2 \in \{0,1\}^{2\lambda}$.
- The vector $\mathbf{C} \in \mathbb{F}_p^{(k+t)\times(\tau-1)}$ i.e., $\tau - 1$ correction strings $\mathbf{C}_i, \ldots, \mathbf{C}_{\tau-1}$ where $k = n + r$.
- The hashed VOLE secret $\tilde{\mathbf{u}}, \tilde{\mathbf{V}}$. This is used in the VOLE consistency check later. Note that instead of sending $\tilde{\mathbf{V}}$ directly, the signer can send a collision-resistant hash of this. This saves some communication as$\tilde{\mathbf{V}}$ is quite large, and it still allows to verify since the verifier can simply compute $\tilde{\mathbf{V}}$ (from $\boldsymbol{\Delta}$, $\tilde{\mathbf{u}}$, $\mathbf{W}$) and check that its hash matches the collision-resistant hash sent by the signer.
- The QuickSilver proof part $(\mathsf{QS}_0, \mathsf{QS}_1) \in \mathbb{F}_{p^r}^{\tau}$.
- The partial $\{\mathsf{CoPath}_{\mathsf{K}}(\Delta_i, \mathsf{sd}^i), \mathsf{com}^i_{\Delta_i}\}_{i<\tau}$ for each of the τ PPRF instances, opening all positions except $\boldsymbol{\Delta} \in \mathbb{F}_{p^r}^{\tau}$.

The asymptotic signature size is

$$\underbrace{(\tau-1)\cdot(n+r+t)\cdot\log p}_{\mathbf{C}} + \underbrace{\tilde{u}}_{t} + \underbrace{2\cdot\tau\cdot r\cdot\log p}_{\mathbf{U}_0,\mathbf{U}_1} + \underbrace{2\cdot\lambda}_{\tilde{\mathbf{V}}} + \underbrace{2\cdot\lambda}_{\mathsf{K}} + \underbrace{4\cdot\lambda}_{h_1,h_2} + \underbrace{\tau\cdot r\cdot\log p\cdot\lambda}_{\mathsf{CoPath}}$$

Parameters. In this section, we explain how to select parameters for our new signature scheme with a security level of λ.

- The field size of $\mathbb{F}_p$ and $\mathbb{F}_{p^r}$ depends on the security of $\mathsf{MQ}_{p,m,n}$ problem to ensure the security level of λ [BMSV22]. For feasible implementation, $\mathbb{F}_p$ is chosen as an extension field of $\mathbb{F}_2$ and subfield of $\mathbb{F}_{2^\lambda}$.
- The repetitions $\tau \in \mathbb{N}$, instead of running a VOLEitH protocol to achieve a security level of λ which costs $O(2^\lambda)$ computation, we can run several parallel τ-instances of the VOLEitH protocol over smaller fields and concatenating the VOLE tags and keys that they produce. This creates VOLE correlations over the $\mathbb{F}_{2^\lambda}$ with only a polynomial amount of work. Since QuickSilver is applied and the challenge space is $\mathbb{F}_{p^r}^{\tau}$, from the Theorem 10, the underlying DVZK achieves soundness error of $O(2^{-\lambda})$ if
$$\left(\frac{m+2}{p^r}\right)^{\tau} = 2^{-\lambda}$$
.
The choice for τ offers tradeoffs between signature size and speed. A small τ means computing fewer VOLEitH protocols and hence a smaller signature size (because signature size scales in the number of VOLE instances), but at the cost of larger values r and hence more work for the signer and verifier and as reverse for a large τ. work for the signer and verifier. Observe that τ needs to be divided by λ. This constraint leads to a limit choice of $\tau \in \mathbb{N}$ i.e., an inefficient actual implementation then we take into account the existing optimization of FAEST for sVOLE correlations. In particular, given a suitable τ, choosing two length parameters $k_0, k_1 \in \mathbb{N}$ such that
$$k_0 := \lceil \lambda/(\log p \cdot r \cdot \tau) \rceil \text{ and } k_1 := \lfloor \lambda/(\log p \cdot r \cdot \tau) \rfloor$$
while two repetition parameters are defined as $\tau_0 := \lambda/(\log p \cdot r) \mod \tau$ and $\tau_1 := \tau - \tau_0$, such that $k_0 \cdot \tau_0 + k_1 \cdot \tau_1 = \lambda/(\log p \cdot r)$, ensuring that concatenating the outputs of τ_0 instances of VOLEitH for $\mathbb{F}_{p^{k_0}}$ and τ_1 instances of VOLEitH for $\mathbb{F}_{p^{k_1}}$ produces VOLE correlations in $\mathbb{F}_{2^\lambda}$ exactly.
- Softspoken OT parameters, we use a sVOLE constructed from SoftspokenOT of length $(n + r + t$ where nth-sVOLE correlations are used to hide the witness of length n, next rth-sVOLE correlations are for nullity check for polynomial set and last tth-correlations is added more to make sure verifier learns nothing about $\mathbf{u}$ in VOLE consistent checks. Specifically, this check reveals a $(n + r + t)$ linear function of $\mathbf{u}$ to the verifier, which needs to hide the underlying witness. From Proposition 8 and the universal hash function $\mathsf{H}^{\mathsf{UHF}} \in \mathbb{F}_p^{t\times(n+r+t)}$ is defined as a form of $[\mathbf{H}|\mathbf{I}_t]$ where $\mathbf{H} \leftarrow_r \mathbb{F}_p^{t\times(n+r)}$, r needs to be chosen such that $p^{-t} = O(2^{-\lambda})$[1].

[1] More details, $p^{-t} = O(2^{-\lambda-B})$ where $B = 16$ is added The extra few bits of security compensate for the security loss $\binom{\tau}{2}$ in the proof of the SoftSpokenVOLE protocol from [BBD+23]

5.3 Efficiency

We outline below a few satisfied parameter set constraints in Section 5.2 for different values of r. For all values of r, the smallest signature size was achieved by setting $(q, m, n) = (4, 88, 88)$ and $(r, \tau) = (36, 2)$ respectively.

Table 2. Signature size in Bytes for various values of (r, τ) for the security level of 128 bits where r is degree of extension field and τ is the number of repetitions, using two $\mathsf{MQ}_{p,m,n}$ parameter sets $(q, m, n) = (4, 88, 88)$ and $(q, m, n) = (256, 40, 40)$ [Fen22].

Parameters	r	τ	Signature size
$(q, m, n) = (4, 88, 88)$	36	2	**2523B**
	20	4	2865B
	12	8	3543B
	8	16	4896B
	6	24	5725B
$(q, m, n) = (256, 40, 40)$	9	2	**2535B**
	5	4	2913B
	3	8	3663B
	2	13	4206B
	1	49	9236B

Acknowledgement

This work is supported by DIM Math Innovation 2021 (N°IRIS: 21003816) from the Paris Mathematical Sciences Foundation (FSMP) funded by the Paris Ile-de-France Region.

References

AGH+23. C. Aguilar Melchor, N. Gama, J. Howe, A. Hülsing, D. Joseph, and D. Yue. The return of the SDitH. In *EUROCRYPT 2023, Part V*, *LNCS* 14008, pages 564–596. Springer, Heidelberg, April 2023.

BBD+23. C. Baum, L. Braun, C. Delpech de Saint Guilhem, M. Klooß, E. Orsini, L. Roy, and P. Scholl. Publicly Verifiable Zero-Knowledge and Post-Quantum Signatures from VOLE-in-the-Head. In *Advances in Cryptology - CRYPTO 2023 - 43rd Annual International Cryptology Conference, CRYPTO 2023, Santa Barbara, CA, USA, August 20-24, 2023, Proceedings, Part V*, *Lecture Notes in Computer Science* 14085, pages 581–615. Springer, 2023.

BCC+24. D. Bui, E. Carozza, G. Couteau, D. Goudarzi, and A. Joux. Short signatures from regular syndrome decoding, revisited. Cryptology ePrint Archive, Paper 2024/252, 2024. `https://eprint.iacr.org/2024/252`.

BCdSG24. D. Bui, K. Cong, and C. D. de Saint Guilhem. Improved all-but-one vector commitment with applications to post-quantum signatures. Cryptology ePrint Archive, Paper 2024/097, 2024. `https://eprint.iacr.org/2024/097`.

BCG+19. E. Boyle, G. Couteau, N. Gilboa, Y. Ishai, L. Kohl, and P. Scholl. Efficient pseudorandom correlation generators: Silent OT extension and more. In *CRYPTO 2019, Part III*, *LNCS* 11694, pages 489–518. Springer, Heidelberg, August 2019.

Beu20. W. Beullens. Sigma protocols for mq, pkp and sis, and fishy signature schemes. In *39th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Zagreb, Croatia, May 10–14, 2020, Proceedings*, *Lecture Notes in Computer Science* 12105. Springer, 2020.

Beu21. W. Beullens. Improved cryptanalysis of uov and rainbow. page 348–373, Berlin, Heidelberg, 2021. Springer-Verlag.

Beu22. W. Beullens. Breaking rainbow takes a weekend on a laptop. Cryptology ePrint Archive, Paper 2022/214, 2022. `https://eprint.iacr.org/2022/214`.

BGI14. E. Boyle, S. Goldwasser, and I. Ivan. Functional signatures and pseudorandom functions. In *PKC 2014*, *LNCS* 8383, pages 501–519. Springer, Heidelberg, March 2014.

BGI16. E. Boyle, N. Gilboa, and Y. Ishai. Function secret sharing: Improvements and extensions. In *ACM CCS 2016*, pages 1292–1303. ACM Press, October 2016.

BJKS94. J. Bierbrauer, T. Johansson, G. Kabatianskii, and B. Smeets. On families of hash functions via geometric codes and concatenation. In *CRYPTO'93*, *LNCS* 773, pages 331–342. Springer, Heidelberg, August 1994.

BMPS20. J.-F. Biasse, G. Micheli, E. Persichetti, and P. Santini. LESS is more: Code-based signatures without syndromes. In *AFRICACRYPT 20*, *LNCS* 12174, pages 45–65. Springer, Heidelberg, July 2020.

BMRS21. C. Baum, A. J. Malozemoff, M. B. Rosen, and P. Scholl. Mac'n'cheese: Zero-knowledge proofs for boolean and arithmetic circuits with nested disjunctions. In *CRYPTO 2021, Part IV*, *LNCS* 12828, pages 92–122, Virtual Event, August 2021. Springer, Heidelberg.

BMSV22. E. Bellini, R. H. Makarim, C. Sanna, and J. A. Verbel. An estimator for the hardness of the MQ problem. In *AFRICACRYPT 22*, *LNCS* 2022, pages 323–347. Springer Nature, July 2022.

BW13. D. Boneh and B. Waters. Constrained pseudorandom functions and their applications. In *ASIACRYPT 2013, Part II*, *LNCS* 8270, pages 280–300. Springer, Heidelberg, December 2013.

CCJ23. E. Carozza, G. Couteau, and A. Joux. Short signatures from regular syndrome decoding in the head. In *EUROCRYPT 2023, Part V*, *LNCS* 14008, pages 532–563. Springer, Heidelberg, April 2023.

CHR+16. M.-S. Chen, A. Hülsing, J. Rijneveld, S. Samardjiska, and P. Schwabe. From 5-pass MQ-based identification to MQ-based signatures. In *ASIACRYPT 2016, Part II*, *LNCS* 10032, pages 135–165. Springer, Heidelberg, December 2016.

CLY+24. H. Cui, H. Liu, D. Yan, K. Yang, Y. Yu, and K. Zhang. Resolved: Shorter signatures from regular syndrome decoding and vole-in-the-head. Cryptology ePrint Archive, Paper 2024/040, 2024. `https://eprint.iacr.org/2024/040`.

DS05. J. Ding and D. Schmidt. Rainbow, a new multivariable polynomial signature scheme. In *ACNS 05*, *LNCS* 3531, pages 164–175. Springer, Heidelberg, June 2005.

Fen22. T. Feneuil. Building mpcith-based signatures from mq, minrank, rank sd and pkp. Cryptology ePrint Archive, Paper 2022/1512, 2022. `https://eprint.iacr.org/2022/1512`.

FJR22. T. Feneuil, A. Joux, and M. Rivain. Syndrome decoding in the head: Shorter signatures from zero-knowledge proofs. In *CRYPTO 2022, Part II*, *LNCS* 13508, pages 541–572. Springer, Heidelberg, August 2022.

FS87. A. Fiat and A. Shamir. How to prove yourself: Practical solutions to identification and signature problems. In *CRYPTO'86*, *LNCS* 263, pages 186–194. Springer, Heidelberg, August 1987.

GGM86. O. Goldreich, S. Goldwasser, and S. Micali. How to construct random functions. *Journal of the ACM*, 33(4):792–807, October 1986.

IKOS07. Y. Ishai, E. Kushilevitz, R. Ostrovsky, and A. Sahai. Zero-knowledge from secure multiparty computation. In *39th ACM STOC*, pages 21–30. ACM Press, June 2007.

KPTZ13. A. Kiayias, S. Papadopoulos, N. Triandopoulos, and T. Zacharias. Delegatable pseudorandom functions and applications. In *ACM CCS 2013*, pages 669–684. ACM Press, November 2013.

Roy22. L. Roy. SoftSpokenOT: Quieter OT extension from small-field silent VOLE in the minicrypt model. In *CRYPTO 2022, Part I*, *LNCS* 13507, pages 657–687. Springer, Heidelberg, August 2022.

Sho94. P. W. Shor. Algorithms for quantum computation: Discrete logarithms and factoring. In *35th FOCS*, pages 124–134. IEEE Computer Society Press, November 1994.

SSH11. K. Sakumoto, T. Shirai, and H. Hiwatari. Public-key identification schemes based on multivariate quadratic polynomials. In *CRYPTO 2011*, *LNCS* 6841, pages 706–723. Springer, Heidelberg, August 2011.

Wan22. W. Wang. Shorter signatures from mq. Cryptology ePrint Archive, Paper 2022/344, 2022. `https://eprint.iacr.org/2022/344`.

WWCY22. Z. Wang, Y. Wang, Y. Chen, and J. Yang. Poster: Fingerprint-face friction based earable authentication. In *ACM CCS 2022*, pages 3487–3489. ACM Press, November 2022.

WYKW21. C. Weng, K. Yang, J. Katz, and X. Wang. Wolverine: Fast, scalable, and communication-efficient zero-knowledge proofs for boolean and arithmetic circuits. In *2021 IEEE Symposium on Security and Privacy*, pages 1074–1091. IEEE Computer Society Press, May 2021.

YSWW21. K. Yang, P. Sarkar, C. Weng, and X. Wang. QuickSilver: Efficient and affordable zero-knowledge proofs for circuits and polynomials over any field. In *ACM CCS 2021*, pages 2986–3001. ACM Press, November 2021.