Shorter VOLEitH Signature from Multivariate Quadratic

Dung Bui

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

Abstract. The VOLE-in-the-Head paradigm, recently introduced by Baum *et al.* (Crypto 2023), is a compiler that uses SoftspokenOT (Crypto 2022) to transfer any VOLE-based designated-verifier zero-knowledge protocol into a publicly verifiable zero-knowledge protocol. Together with the Fiat-Shamir transformation, a new digital signature scheme FAEST (faest.info) is proposed, and it outperforms all MPC-in-the-Head signatures.
We propose a new candidate post-quantum signature scheme from the Multivariate Quadratic (MQ) problem in the VOLE-in-the-Head framework, which significantly reduces the signature size compared to previous works. We achieve a signature size ranging from 3.5KB to 6KB for the 128-bit security level. Compared to the state-of-the-art MQ-based signature schemes and existing VOLE-in-the-Head signatures, our scheme achieves the smallest signature size (1.5 to 2 times compared to MQ-based schemes) while keeping the computational efficiency competitive.

1 Introduction

Zero-Knowledge, Code-based signature schemes, and Multivariate Quadratic Assumption. Zero-knowledge proof allows a prover to convince a verifier their knowledge of a witness for an NP statement without revealing any additional information. Zero-knowledge proof has numerous applications in cryptography, especially for designing efficient signature schemes by applying the Fiat-Shamir transform [FS87] to convert any public-coin zero-knowledge proof system into a signature scheme.

Digital signatures are fundamental to the digital world. 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, many of which are based on error-correcting codes. Code-based signature schemes are built from various assumptions (syndrome decoding, multivariate quadratic, MinRank, subset sum assumptions) and using two main paradigms that are MPC-in-the-head (MPCitH) [FJR22, AGH$^+$23] and later VOLE-in-the-head (VOLEitH) [BBD$^+$23, CLY$^+$24].

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], stands out as one of the earliest and most studied signature schemes in multivariate cryptography. Nevertheless, its security has been reduced by recent attacks [Beu21,Beu22]. This leads to the question of whether we can construct an efficient signature scheme based on the multivariate quadratic problem.

Code-based signature schemes from MPCitH. Numerous recent studies on Fiat-Shamir code-based digital signatures have applied the MPC-in-the-head paradigm, initially introduced in [IKOS07]. In essence, this paradigm allows the prover to "simulate" 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. The zero-knowledge property is established through the security of the MPC protocol against an honest-but-curious adversary, who only gains access to the views 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 paradigm having signature sizes from 3 KB-7KB. While signatures based on multivariate quadratic assumption [Beu20, Wan22, Fen22] have a larger signature size 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 offers 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. Later, [CLY+24] introduced a signature scheme based on the Regular Syndrome Decoding assumption with signature sizes of 4KB for the security parameter of 128 bits 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

Publicly verifiable ZK. We first propose a new zero-knowledge protocol for the multivariate quadratic problem over a finite field. Our zero-knowledge proof is publicly verifiable and is constructed from a designated-verifier ZK protocol by combining 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 $\mathsf{MQ}_{p,m,n}$ it is equivalent to proving the knowledge of the solutions in a set of degree-2 polynomials over $\mathbb{F}_p$, therefore, we adapt the QuickSilver ZK proof [YSWW21] for nullity check of a polynomial with the cost of $O(n \log p)$ bits.
- From VOLE-based designated-verifier ZK protocol to publicly verifiable ZK protocol, we design a new Vector Commitment based on multi-instances PPRF, it has a better performance compared to the GGM-tree-based vector commitment from [BBD+23] while its security satisfies multi-hiding and extractable binding properties. Then from Vector Commitment, SoftspokenOT is used to allow us to obtain VOLE which is the IT-MAC in QuickSilver. Since all frameworks of VOLE-in-the-Head, SoftspokenOT, or QuickSilver are only over the binary field $\mathbb{F}_2$, we carefully adapt all of them to get a publicly verifiable ZK protocol over any finite field $\mathbb{F}_p$.

VOLEitH-based signature. We introduce a new signature scheme based on the multivariate quadratic problem using the Fiat-Shamir transform, we compile our publicly verifiable zero-knowledge protocol into an MQ-based signature scheme.

Table 1. The parameter choice and estimated security of the $\mathsf{MQ}_{p,m,n}$ problem in three NIST security levels (the notation of parameters for attack algorithms follows Crossbred [CKPS00], and FXL [JV17]).

Category	p	m	n	Estimated security (bits)	Attack Algorithm
I	2	150	150	142	Crossbred (D: 13, d: 1, k: 30)
	4	88	88	149	Crossbred (D: 17, d: 1, k: 24)
III	2	224	224	206	Crossbred (D: 19, d: 1, k: 41)
	4	128	128	218	Crossbred (D: 20, d: 3, k: 43)
V	2	320	320	298	Crossbred (D: 20, d: 2, k: 60)
	4	160	160	266	BooleanSolveFXL (k: 73, variant: las_vegas)

We choose the number of variables n to be being equal to the number of polynomials m to maximize the hardness of the $\mathsf{MQ}_{p,m,n}$ instance. We compare our scheme with the state-of-the-art

using two fields $\mathbb{F}_2$ and $\mathbb{F}_4$, our VOLEitH-based signature works for any field that is a power of 2 but the larger the field the more computation the signer and the verifier need to pay so for efficient computation, we pick the size of the field is of $p = 2$ and $p = 4$. The value n varies depending on the security level and it is chosen by executing the estimator[1] of [BMSV22, EVZB23]. We present in Table 1 the choice of n and the estimated security for three NIST security levels. The details of parameter choice are shown in Section 5.3.

We present our signature size for different settings on Table 3 and the comparison with the state-of-the-art of MQ-based signatures on Table 2. Compared to the state-of-the-art MQ-based signature schemes, our signature reduces the size of the signature by a factor between 1.5 to 2 in various settings. Specifically, for the set of parameters $(p, m, n) = (4, 88, 88)$ and $(2, 150, 150)$, our shortest signature has the size of around 3.5KB and 4KB respectively. For each set of parameters, we show the signature size for two variants (fast and short). We argue that our scheme outperforms other MQ-based signatures by having a smaller signature size while keeping a computational competitive efficiency (see Section 5.4 for estimated computation cost).

Table 2. Comparison of our new signature scheme with other MQ-based signatures (restricting to the schemes using the FS heuristics) in terms of parameter choice, estimated security, sizes of public key, and signature for the security level I of NIST's security standard. Each signature has two variants (fast and short), N is the width of GGM-tree expansion (the number of parties in MPCitH-based signature and the length of vector commitment in VOLEitH-based signature), τ is the number of repetitions.

Scheme	MQ parameters			MPC/VOLE parameters		Size	
	p	$m = n$	Security	N	τ	Public key	Signature
MQ-DSS [CHR+16]	31	48	141	-	-	46 B	28400 B
MudFish [Beu20]	4	88	149	4	68	38 B	14640 B
Mesquite$_{\mathsf{fast}}$ [Wan22]	4	88	149	8	49	38 B	9492 B
Mesquite$_{\mathsf{short}}$ [Wan22]	4	88	149	32	28	38 B	8844 B
Fen22$_{\mathsf{fast}}$ [Fen22]	251	40	135	32	40	56 B	8488 B
Fen22$_{\mathsf{short}}$ [Fen22]	251	40	135	256	25	56 B	7114 B
MQOM-L1$_{\mathsf{fast}}$ [BFR23]	251	43	144	32	34	59 B	7809 B
MQOM-L1$_{\mathsf{short}}$ [BFR23]	251	43	144	256	22	59 B	6575 B
MQOM-L1$_{\mathsf{fast}}$ [BFR23]	31	49	143	32	35	47 B	7621 B
MQOM-L1$_{\mathsf{short}}$ [BFR23]	31	49	143	256	20	47 B	6348 B
Our-L1$_{\mathsf{fast}}$	4	88	149	16	33	38 B	5103 B
Our-L1$_{\mathsf{short}}$	4	88	149	256	17	38 B	3792 B
Our-L1$_{\mathsf{fast}}$	2	150	142	16	43	35 B	6697 B
Our-L1$_{\mathsf{short}}$	2	150	142	256	19	35 B	4209 B

1.2 Concurrent work

Following the footprint of VOLEitH-based signature, recently in the second round of NIST call[2], FAEST team (`faest.info`) proposed improvements for VOLEitH-based signature [BBM+24] that includes 1) new construction of batch vector commitment and 2) new VOLEitH-based signatures from various hardness assumptions, one of them (called KuMQuat) is also based on MQ problem. KuMQuat has the smallest signature size compared to other VOLEitH signatures (ranging from 2555B to 3588B for the security level I), and [BBM+24] provided a benchmark for showing the performance of KuMQuat is still competitive.

1.3 Organization

We provide a detailed technical overview of our results in Section 2. We introduce necessary preliminaries in Section 3, and describe our designated verifier ZK, our publicly-verifiable ZK using our new vector commitment in Section 4. In Section 5, we describe the description of our signature, parameter choices, and signature size for various settings, we then evaluate the performance of our scheme.

[1] https://github.com/Crypto-TII/CryptographicEstimators

[2] https://csrc.nist.gov/Projects/post-quantum-cryptography/post-quantum-cryptography-standardization/Call-for-Proposals

2 Technical Overview

2.1 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 $\{f_i(\mathbf{x}) = \mathbf{x}^T\mathbf{A}_i\mathbf{x} + \mathbf{b}_i^T\mathbf{x} - y_i\}_{i\leq m}$ 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, a DVZK protocol for the MQ problem is equivalent to a DVZK protocol for batch m nullity check of a polynomial set. Its instantiation is based on the 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 easily adapted as a constant coefficient). For the efficiency of the signature based on the DVZK protocol, we need to choose a small set of parameters of DVZK protocol therefore to make sure the soundness is negligible the DVZK protocol is needed to repeat 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 defines a polynomial $\mathbf{g_i}(X)$ and computes locally its coefficients $\mathbf{A}_{i,0}, \mathbf{A}_{i,1} \in \mathbb{F}_{p^r}^{\tau}$ such that

$$\begin{aligned}\mathbf{g_i}(X) &= f_{i,2}(\mathbf{M}[\mathbf{x}] + \mathbf{x}\cdot[1\cdots1]\cdot\mathsf{diag}(X)) + f_{i,1}(\mathbf{M}[\mathbf{x}] + \mathbf{x}\cdot[1\cdots1]\cdot\mathsf{diag}(X))\cdot X - y_i\cdot[1\cdots1]\cdot X^2\\ &= \mathbf{A}_{i,0} + \mathbf{A}_{i,1}\cdot X.\end{aligned}$$

  The degree of $\mathbf{g_i}(X)$ is only 1 since the coefficient of degree 2 is 0 if $\mathbf{x}$ is a 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

$$\mathbf{B}_i := f_{i,2}(\mathbf{K}[\mathbf{x}]) + f_{i,1}(\mathbf{K}[\mathbf{x}]) \circ \boldsymbol{\Delta} - y_i\cdot\boldsymbol{\Delta}^2 \in \mathbb{F}_{p^r}^{\tau}.$$

  Observe 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 set of sample challenges $(\chi_i)_{i\leq m} \in \mathbb{F}_{p^r}^{\tau}$ and later a linear combination of $(\sum_{i=1}^m \chi_i\cdot\mathbf{A}_{i,0}, \sum_{i=1}^m \chi_i\mathbf{A}_{i,1})$ constructing from χ_i respectively.
  To make sure the zero-knowledge property, in the preprocessing phase, P and V need to generate τ extra OLEs over $\mathbb{F}_{p^r}$, while P holds $(\mathbf{A}_0^*, \mathbf{A}_1^*)$ and V holds $(\mathbf{B}^*, \boldsymbol{\Delta})$ such that

$$\mathbf{B}^* = \mathbf{A}_0^* + \mathbf{A}_1^* \circ \boldsymbol{\Delta} \in \mathbb{F}_{p^r}^{\tau}.$$

  Then P uses $(\mathbf{A}_0^*, \mathbf{A}_1^*)$ to mask $(\sum_{i=1}^m \chi_i\cdot\mathbf{A}_{i,0}, \sum_{i=1}^m \chi_i\mathbf{A}_{i,1})$ in the linear combination sent from P and later V can still locally verifies using $(\mathbf{B}^*, \boldsymbol{\Delta})$. Generating τ extra OLEs only costs $(r\cdot\tau)$-sVOLE correlations over $\mathbb{F}_p$ and it is negligible since $r\cdot\tau$ 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 $\mathbb{F}_{p^r}$.

2.2 MQ-based signature from VOLE-in-the-Head paradigm

Publicly-verifiable Zero-Knowledge proof. We first design a public-coin ZK proof for the 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 using the VOLE in-the-head paradigm. We describe our publicly-verifiable ZK protocol as below:

- Firstly, 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 Softspoken OT to construct sVOLE, this manner allows P and V can simulate the sVOLE correlation before sampling $\boldsymbol{\Delta}$ as a challenge in publicly-verifiable ZK protocol. Along with Softspoken OT, we propose a new all-but-one Vector Commitment based on Multi-instance PPRF which is used to generate efficiently SoftspokenOT-based sVOLE. Our vector commitment satisfies multi-hiding (where $\mathcal{A}$ can query polynomial time to hiding game to break the randomness) and extractable binding proofs.
- Plugging sVOLE obtained from vector commitment to our DVZK protocol while now the global key $\boldsymbol{\Delta}$ is sampled after P committed the witness.
- V outputs accept if all checks for SoftspokenOT, QuickSilver, and DVZK are passed.

VOLEitH-based signature. Finally, applying the Fiat-Shamir heuristic to the publicly verifiable ZK protocol, we achieve a new signature scheme from the MQ problem, and the security is maintained from the publicly verifiable ZK protocol and random oracles under $\mathsf{EUF\text{-}KO}$, $\mathsf{EUF\text{-}CMA}$ security.

3 Preliminaries

3.1 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 $\{1, \cdots, n\}$ and $[0, n)$ for the set $\{0, \cdots, n-1\}$. 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 $\mathbb{F}_{p^r} = \mathbb{F}_p[X]/f(X)$ ($r \in \mathbb{N}$) where $f(X)$ is a monic, irreducible polynomial of degree r as an extension 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 $t \leq r$. Given a polynomial f over $\mathbb{F}_p$ taking n variables, for a vector $\mathbf{x} = (x_1, \cdots, x_n) \in \mathbb{F}_p^n$ we denote $f(\mathbf{x}) = f(x_1, \cdots, x_n)$ and from f over $\mathbb{F}_p$ we naturally define it as a polynomial over $\mathbb{F}_{p^r}$ by operating additions and multiplications over $\mathbb{F}_{p^r}$. We denote $\mathbf{x}^k$ as $(x_1^k, \cdots, x_n^k)$ for $k \in N$.

Vectors and Matrix. For a matrix $\mathbf{M}$, we denote $\mathbf{M}_{i,j}$ for the 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 corresponding. 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 and 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 $\mathbb{F}_p^{\tau-1}$. We denote $\mathsf{diag}(\boldsymbol{\Delta})$ where $\boldsymbol{\Delta} = (\Delta_0, \cdots, \Delta_{\tau-1}) \in \mathbb{F}_p^{\tau}$ as a matrix over $\mathbb{F}_p^{\tau \times \tau}$ of the form $\begin{bmatrix} \Delta_0 & & \\ & \cdots & \\ & & \Delta_{\tau-1} \end{bmatrix}$. 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 $\mathbf{u}$.

3.2 Cryptographic Definitions

Basic definitions. We define the definitions of secure PRG and collision-resistant hash function as below.

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

- *Expansion: $m > \lambda$;*
- *For any PPT $\mathcal{A}$ running in time t, the distributions $\{G(s) | s \leftarrow \{0,1\}^\lambda\}$ and $\{r | r \leftarrow \{0,1\}^m\}$ are $(t, \epsilon_{\mathsf{PRG}^m})$-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:*

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

UC model. Given $\mathcal{F}$ be a two-party functionality and Π be a secure protocol for instantiating $\mathcal{F}$. The protocol Π is said to be secure against malicious adversary if for all non-uniform PPT adversary $\mathcal{A}$ in the real model, there exists a non-uniform PPT adversary $\mathcal{S}$ in the deal model satisfying:

$$\text{IDEAL}_{(\mathcal{F},\mathcal{S},i)}(X,Y) \overset{c}{\approx} \text{REAL}_{(\Pi,\mathcal{A},i)}(X,Y)$$

where $i \in \{1,2\}$ index of corrupted party. $\text{IDEAL}_{(\mathcal{F},\mathcal{S},i)}(X,Y)$ is the output pair of the honest party and the adversary $\mathcal{S}$ in the ideal model, $\text{REAL}_{(\Pi,\mathcal{A},i)}(X,Y)$ is defined as the output pair of the honest party and the adversary $\mathcal{A}$ from the real execution of Π. Informally, proving in the UC model means constructing a simulator that can simulate the view of adversary with the help of $\mathcal{F}$.

Zero-knowledge definitions. An argument $\Pi = (\mathsf{Setup}, \mathsf{P}, \mathsf{V})$ is a public coin if the verifier's messages are chosen uniformly at random independently of the messages sent by the prover, i.e., the challenges correspond to the verifier's randomness ρ.

Definition 4 (Perfect completeness). *A proof system $\Pi = (\mathsf{Setup}, \mathsf{P}, \mathsf{V})$ for $\mathcal{R}$ is perfectly complete, if*

$$\Pr\left[\langle \mathsf{P}(\mathsf{crs}, x, w), \mathsf{V}(\mathsf{crs}, x)\rangle = 1 \;\middle|\; \begin{array}{c} \mathsf{crs} \leftarrow \mathsf{Setup}(1^\lambda) \\ (x,w) \in \mathcal{R} \end{array}\right] = 1$$

Definition 5 (Computationally soundness). *A proof system $\Pi = (\mathsf{Setup}, \mathsf{P}, \mathsf{V})$ is computational sound if for every efficient adversary $\mathcal{A}$*

$$\Pr\left[\begin{array}{c} \langle \mathcal{A}, \mathsf{V}(\mathsf{crs}, x)\rangle = 1 \\ x \notin \mathscr{L} \end{array} \;\middle|\; \begin{array}{c} \mathsf{crs} \leftarrow \mathsf{Setup}(1^\lambda) \\ x \leftarrow \mathcal{A}(1^\lambda, \mathsf{crs}) \end{array}\right] = \mathsf{negl}(\lambda)$$

Definition 6 (Special honest-verifier zero-knowledge (SHVZK)). *A public coin argument Π is a SHVZK if there exists a probabilistic polynomial time simulator Sim such that for all non-uniform polynomial time adversaries $\mathcal{A}$we have*

$$\begin{aligned} &\Pr\left[\begin{array}{c} \mathcal{A}(tr) = 1 \\ (x,w) \in \mathcal{R} \end{array} \;\middle|\; \begin{array}{c} \mathsf{crs} \leftarrow \mathsf{Setup}(1^\lambda) \\ (x,w,\rho) \leftarrow \mathcal{A}(\mathsf{crs}); tr \leftarrow \langle \mathsf{P}(\mathsf{crs}, x, w), \mathsf{V}(\mathsf{crs}, x)\rangle \end{array}\right] \\ \approx &\Pr\left[\begin{array}{c} \mathcal{A}(tr) = 1 \\ (x,w) \in \mathcal{R} \end{array} \;\middle|\; \begin{array}{c} \mathsf{crs} \leftarrow \mathsf{Setup}(1^\lambda) \\ (x,w,\rho) \leftarrow \mathcal{A}(\mathsf{crs}); tr \leftarrow \mathsf{Sim}(\mathsf{crs}, x, \rho) \end{array}\right] \end{aligned}$$

where ρ is the public coin randomness used by the verifier.

Lemma 7 (Collisions in random oracle). *Given a collision-resistant hash function $\mathsf{H} : \{0,1\}^* \to \{0,1\}^{2\lambda}$ and H is modeled as an random oracle. For any PPT algorithm making q queries to H ha probability of at most $q^2/2^{2\lambda}$ encountering a collision.*

3.3 Multivariate Quadratic Problem

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

Definition 8 (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 $(\mathbf{A}_i)_{i\le m} \leftarrow_r \mathbb{F}_p^{n\times n}$, $(\mathbf{b}_i)_{i\le m} \leftarrow_r \mathbb{F}_p^n$. Set $y_i \leftarrow \mathbf{x}^T\mathbf{A}_i\mathbf{x} + \mathbf{b}_i^T\mathbf{x}$. Output $(\mathbf{A}_i, \mathbf{b}_i, y_i)_{i\le m}$.*
- *(Goal) Given $(\mathbf{A}_i, \mathbf{b}_i, y_i)_{i\le m}$, 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 $i \in [m]$.*

While recent attacks [Beu21, Beu22] have reduced the security of well-known signatures based on the MQ problem as Rainbow, in our work we set the parameter set by executing the estimator of [BMSV22,EVZB23] to estimate the hardness of given an instance of MQ against all existing attack algorithms, this estimator is also used to compute the secure parameters for MPCitH signatures based on MQ problem [Fen22, BFR23].

3.4 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.

3.5 Information-Theoretic Message Authentication Codes

We recall 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 $\mathbb{F}_{p^r}$ and $y = \sum_{i=1}^{l} c_i \cdot x_i + c$, the parties can locally compute $[\![y]\!] = (\mathsf{K}[y], \mathsf{M}[y], y)$ from $[\![x_i]\!]$ as $\mathsf{K}[y] = \sum_{i=1}^{l} c_i \cdot \mathsf{K}[x_i] + c \cdot \Delta$ and $\mathsf{M}[y] = \sum_{i=1}^{l} c_i \cdot \mathsf{M}[x_i]$. To authenticated x, P reveals 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 produce an valid IT-MACs $[\![x']\!] = (\mathsf{K}[x'], \mathsf{M}[x'], x')$ from $[\![x]\!]$, P needs to guess $\Delta \in \mathbb{F}_{p^r}$ such that $\Delta = (\mathsf{M}(x) - \mathsf{M}(x')) \cdot (x - x')^{-1}$.

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 authenticate 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 $1/p^r$ by following the SZ lemma (Section 3.5).

Lemma 9 (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* $\Pr[P(r_1, r_2, \ldots, r_n) = 0] \le d/|S|$.

In the 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 $\boldsymbol{\Delta} = (\Delta_1, \cdots, \Delta_\tau)$ corresponding, where $\mathbf{K}[\mathbf{x}], \mathbf{M}[\mathbf{x}] \in \mathbb{F}_{p^r}^{n \times \tau}$ and $\boldsymbol{\Delta} \in \mathbb{F}_p^\tau$ such that $\mathbf{M_i}[\mathbf{x}] = \mathbf{K_i}[\mathbf{x}] - \mathbf{x} \cdot \Delta_i$ for all $i \in [0, \tau)$. The batch IT-MACs of a vector are generated by the sVOLE protocol that securely realizes the ideal functionality $\mathcal{F}_{\mathsf{sVOLE}}^{n,\tau}$ (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 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 $\boldsymbol{\Delta} \leftarrow_r \mathbb{F}_{p^r}^{\tau}$ and computes $\mathbf{W} := \mathbf{V} + \mathbf{u} \cdot [1 \cdots 1] \cdot \mathsf{diag}(\boldsymbol{\Delta})$.
  - If the V is corrupted then wait for $\mathcal{A}$ to send $\boldsymbol{\Delta} \in \mathbb{F}_{p^r}^{\tau}$, $\mathbf{W} \in \mathbb{F}_{p^r}^{n\times\tau}$; samples $\mathbf{u} \leftarrow_r \mathbb{F}_p^n$ and computes $\mathbf{V} := \mathbf{W} - \mathbf{u} \cdot [1 \cdots 1] \cdot \mathsf{diag}(\boldsymbol{\Delta})$.
  - Otherwise, samples $\mathbf{u} \leftarrow_r \mathbb{F}_p^n, \mathbf{W} \leftarrow_r \mathbb{F}_{p^r}^{n\times\tau}, \boldsymbol{\Delta} \leftarrow_r \mathbb{F}_{p^r}^{\tau}$ and computes $\mathbf{W} := \mathbf{V} + \mathbf{u} \cdot [1 \cdots 1] \cdot \mathsf{diag}(\boldsymbol{\Delta})$.
- If P and V send (init) to functionality, sends $(\mathbf{u},\ \mathbf{V})$ to P.
- If P and V send (get) to functionality, sends $(\boldsymbol{\Delta},\ \mathbf{W})$ 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 $\mathbb{F}_p$

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 $\mathcal{F}_{\mathsf{polyZK}}^{p,t}$

3.6 Designated-Verifier ZK for nullity check of Polynomial sets

Nullity check of polynomial sets is a batch check of whether a witness is a root of a set of polynomials, we define the ideal functionality of nullity check for a set of t degree-2 polynomials having n variables over $\mathbb{F}_p$ in Figure 2. From the footprint of VOLE-based IT-MACs, the instantiation of $\mathcal{F}_{\mathsf{polyZK}}^{p,t}$ is followed by Quicksilver technique [YSWW21].

In particular, given t polynomials $\{f_i\}_{i\leq t}$ 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 locally compute all the coefficients.

Batch nullity check. P and V generates randomly an IT-MACs $[\![A]\!] = (\mathsf{M}[A], \mathsf{K}[A], A)$, given a set of challenge $(\chi_i)_{i\in[t]}$ sampled from $(\mathbb{F}_{p^r})^t$, 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 $\sum_{i=1}^{t} \chi_i \cdot B_i + \mathsf{K}[A] = A_1 \cdot \Delta + A_0$. By lifting the IT-MAC $[\![A]\!]$ to the extension field $\mathbb{F}_{p^r}$ instead of $\mathbb{F}_p$ (see Section 2 for details), we obtain a negligible soundness error of being $3/p^r$ (note that in [YSWW21] the soundness error is $(t+2)/p^r$ since the challenge set is defined as $(\chi^i)_{i\in[t]}$ for $\chi \in \mathbb{F}_{p^r}$). As a consequence, the batch nullity check results in a total communication of $(n+2r)\log p$ bits in the sVOLE-hybrid model.

3.7 Universal Hashing

We recall the definitions of n-hiding and ϵ-universal hash function [Roy22] in the following definitions.

Definition 10 (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 $\mathbf{x} \in \mathbb{F}_p^n$*

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

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

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

Definition 11 (Hiding). *A matrix $\mathbf{H} \in \mathbb{F}_p^{r\times(n+h)}$ is $\mathbb{F}_p^n$-hiding if the distribution of $\mathbf{Hv}$ is independent from $\mathbf{v}[0,n)$ when $\mathbf{v}[n,n+h) \leftarrow \mathbb{F}_p^h$. 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 can be done by using the proposition below and the instantiation of $\mathcal{H}^{\mathsf{UHF}}$ is shown in [Roy22].

Proposition 12. *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. *$\mathcal{H}^{\mathsf{UHF}}$ is ϵ-almost universal.*
2. *$\mathcal{H}^{\mathsf{UHF}}$ is $\mathbb{F}_p^n$-hiding.*

3.8 Multi-Instance Puncturable PRF

Informally, a puncturable pseudorandom function (PPRF) [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 satisfies:

- There is an algorithm $F.\mathsf{Eval}$ such that $F.\mathsf{Eval}(k\{x\}, x') = F_K(x')$ for all $x' \neq x$ (evaluating F at every point except for x).
- $F_K(x)$ is indistinguishable from random given $k\{x\}$.

The motivation for using τ-multi-instance PRF is to construct a vector commitment that includes τ-parallel instances of PPRF using the same key K, while distinct keys K are used across distinct signature queries. Using the multi-instance PPRF allows us to generalize and simplify the security of vector commitment. The security of vector commitment is also tighter compared to when using directly the concept of single PPRF (in this case the advantage of adversary is always proportional to τ). We then recall the formal definition of τ-multi-instance PRF that is proposed in [BCC+24].

Definition 13 ((N,τ)-instance (t,ϵ)-secure PPRF [BCC+24]). *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.

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

3.9 Vector Commitment

A (non-interactive) vector commitment (VC) scheme (with message space $\mathcal{M}$ and commitment space $\mathcal{C}$) is defined by four PPT algorithms ($\mathsf{Setup},\mathsf{Commit},\mathsf{Open},\mathsf{Verify}$). While in formal definition the Setup algorithm gives the crs that is one of the inputs of three other algorithms, in our construction, we consider crs as a public parameter then for simplicity we omit crs in these algorithms. We define the hiding and extractable-binding properties of vector commitment in the following definitions. While the hiding is used to prove that the adversary learns nothing from the $\mathsf{VC.Open}$ even if the adversary is allowed to choose the opening challenge, the extractable-binding game shows that after committing, an adversary is bound to the messages contained in the commitments, except for the one(s) that it does not open.

Definition 14 (VC Real-or-Random Hiding [BBD+23]). *Let* VC *be a vector commitment scheme in the RO model with random oracles* H *and* H_1*. The* adaptive hiding experiment for VC, AdpHiding*, with* $N = \mathsf{poly}$ *and stateful* $\mathcal{A}$ *is defined as follows.*

1. $\mathsf{crs} \leftarrow \mathsf{Setup}(1^\lambda, N)$
2. $b \leftarrow\$ \{0,1\}$
3. $\mathsf{seed} \leftarrow\$ \{0,1\}^\lambda$
4. $(\mathsf{com},\mathsf{decom},(\mathsf{sd}_1^*,\ldots,\mathsf{sd}_N^*)) \leftarrow \mathsf{Commit}(\mathsf{seed})$
5. $I \leftarrow \mathcal{A}(1^\lambda,\mathsf{crs},\mathsf{com})$
6. $\mathsf{pdecom}_I \leftarrow \mathsf{Open}(\mathsf{decom}, I)$
7. $\mathsf{sd}_i \leftarrow \mathsf{sd}_i^*$ *for* $i \in I$
8. *For* $i \notin I$, $\mathsf{sd}_i \leftarrow \begin{cases} \mathsf{sd}_i^*, & \text{if } b = 0 \\ \leftarrow\$\, \mathcal{M}, & \text{otherwise} \end{cases}$
9. $b' \leftarrow \mathcal{A}(\{\mathsf{sd}_i\}_{i\in[N]},\mathsf{decom}_I)$
10. *Output 1 (win) if* $b' = b$ *else 0 (lose).*

The selective hiding experiment for VC*, SelHiding, is defined similarly but* $\mathcal{A}$ *must choose* I *prior to receiving* com.

We define $\mathcal{A}$*'s advantage as:*

$$\mathsf{Adv}^{\mathsf{AdpHiding}}_{\mathcal{A},\mathsf{VC}} = \Pr[\mathcal{A} \textit{ wins } \textit{AdpHiding}] - \frac{1}{2},$$

and similarly for the SelHiding game.

We say VC *is* adaptively (resp. selectively) hiding *if every PPT adversary* $\mathcal{A}$ *has a negligible advantage in the respective game.*

Definition 15 (VC Extractable-Binding [BBD+23]). *Let* VC *be a vector commitment constructed from two random oracles* H *and* H_1*,* $\mathcal{A}$ *be any PPT adversary with oracle access to* H *and* H_1*, resulting in query-response lists* Q *and* Q_1 *respectively.*
An extractor Ext *be a PPT algorithm defined as:*

– $\mathsf{Ext}\,(\mathsf{pp},(Q,Q_1),\mathsf{com}) \to (\mathsf{sd}_i)_{i\in[N]}$*: given an instance of* VC *public parameters* pp*, query-response lists* Q *and* Q_1*, and a commitment* $\mathsf{com} \in \mathcal{C}$*, return the messages committed to by* com*. (*Ext *may output* $\mathsf{sd}_i = \perp$ *to signify an invalid commitment.)*

For any $N \le \mathsf{poly}(\lambda)$ *and stateful adversary* $\mathcal{A}$*, we define the straight-line extractable/binding game,* ExtBind*, against* VC *as follows:*

1. $\mathsf{pp} \leftarrow \mathsf{Setup}^{\mathsf{H},\mathsf{H}_1}(1^\lambda, N)$
2. $\mathsf{com} \leftarrow \mathcal{A}^{\mathsf{H},\mathsf{H}_1}(1^\lambda, \mathsf{pp})$
3. $\{\mathsf{sd}_i^*\}_{i\in[N]} \leftarrow \mathsf{Ext}(\mathsf{pp}, (Q, Q_1), \mathsf{com})$
4. $(I, \mathsf{pdecom}_I) \leftarrow \mathcal{A}^{\mathsf{H},\mathsf{H}_1}(\mathtt{open})$
5. *Output 0 (failure) if:*
   - $\mathsf{Verify}^{\mathsf{H},\mathsf{H}_1}(\mathsf{com}, j^*, \mathsf{pdecom}_{j^*}) = \perp$*, i.e., the opening is not valid; or*
   - $\mathsf{Verify}^{\mathsf{H},\mathsf{H}_1}(\mathsf{com}, j^*, \mathsf{pdecom}_{j^*}) = \{\mathsf{sd}_i\}_{i\in I}$ *and* $\mathsf{sd}_i = \mathsf{sd}_i^*$ *for all* $i \neq j^*$*, i.e., the opening is valid but extraction by* Ext *was successful*
6. *Otherwise, output 1 (success).*

We define $\mathcal{A}$*'s advantage as* $\mathsf{Adv}^{\mathsf{ExtBinding}}_{\mathcal{A},\mathsf{VC}} = \Pr[1 \leftarrow \mathrm{ExtBind}]$*, and say* VC *is* straight-line extractable-binding *w.r.t.* Ext *if*

$$\mathsf{Adv}^{\mathsf{ExtBinding}}_{\mathcal{A},\mathsf{VC}} \leq \mathsf{negl}.$$

4 Publicly-Verifiable ZK for Multivariate Quadratic

Observe that for $\mathsf{MQ}_{p,m,n}$, given $(\mathbf{A}_i, \mathbf{b}_i, y_i)_{i\leq m}$, 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 3.6) 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 verifiable, we apply the VOLEitH framework [BBD+23] and adapt all constructions for multi-times τ repetitions. We present our publicly-verifiable ZK protocol in Section 4.2 where we introduce our new GGM-tree style vector commitment based on multi-instance PPRF.

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 $\mathcal{F}^{n,\tau}_{\mathsf{sVOLE}}$ hybrid model, and its security (soundness and zero-knowledge) is shown in Theorem 16. The technical overview is presented in Section 2, to be more specific, we argue about two main concerns that 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.
  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, applying the Quicksilver technique for the polynomial set (Section 3.6) 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:

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

Theorem 16. *The protocol $\Pi_{\mathsf{MQ-DVZK}}$ is a designated-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 malicious verifier with the soundness error $\epsilon_{\Pi_{\mathsf{MQ-DVZK}}}$ bounded by $(3/p^r)^\tau$ and information-theoretic security.*

Sketch proof. The soundness error comes from whether the malicious prover can 1) choose an invalid witness such that $\mathbf{A}_{i,2} = 0$ and this happens with a probability of $(1/p^r)^\tau$, or 2) cheat in Equation (1) to keep this equation hold, observe that this equation has degree 2, $\boldsymbol{\Delta} \leftarrow_r \mathbb{F}_{p^r}^\tau$ is uniformly random and kept secret from the adversary's view then from Section 3.5, the probability that the above equation holds is bounded by $(2/p^r)^\tau$ in this case. In total $\epsilon_{\Pi_{\mathsf{MQ-DVZK}}} \leq (3/p^r)^\tau$.
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 $\chi_i \in \mathbb{F}_{p^r}^\tau$ 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 simulator Sim that simulates the view of a malicious prover and an honest verifier to argue soundness and zero-knowledge properties respectively.

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 $\boldsymbol{\Delta} \in \mathbb{F}_{p^r}^\tau$, 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 $\mathbf{B}^* = \mathbf{A}_0^* + \mathbf{A}_1^* \circ \boldsymbol{\Delta} \in \mathbb{F}_{p^r}^\tau$.
- When $\mathcal{A}$ sends $\gamma \in \mathbb{F}_p^n$ in step 1 of online phase, Sim extracts the witness as $\mathbf{x} := \gamma_i + \mathbf{u}[0, n)$ for $i \in [n]$.
- Sim executes the remaining part of protocol $\Pi_{\mathsf{MQ-DVZK}}$ as an honest verifier, using $\boldsymbol{\Delta}$ 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 $\mathcal{F}_{\mathsf{polyZK}}$ does not satisfy that $f_i(\mathbf{x}) = 0$ for all $i \in [m]$. We denote this probability as $\epsilon_{\Pi_{\mathsf{MQ-DVZK}}}$ and it is bounded as below.

Assume $(f_i(\mathbf{x}) = y_i)_{i \in [m]}$ for some $y_i \in \mathbb{F}_{p^r}$ and $\mathbf{x}$ is the vector extracted by Sim, then we have

$$\mathbf{g}_i(\boldsymbol{\Delta}) = \mathbf{A}_{i,0} + \mathbf{A}_{i,1} \circ \boldsymbol{\Delta} - y_i \cdot \boldsymbol{\Delta}^2$$

At round 4 of Figure 3, Sim receives $\mathsf{QS}'_0 = \mathsf{QS}_0 + \mathbf{t}_0$, $\mathsf{QS}'_1 = \mathsf{QS}_1 + \mathbf{t}_1$ where $\mathsf{QS}_0, \mathsf{QS}_1$ are the actual values that are sent from a prover following the protocol. The Equation (1) becomes

$$\mathbf{B} = \left(\sum_{i=1}^{m} -y_i \cdot \chi_i\right) \circ \boldsymbol{\Delta}^2 + \left(\mathsf{QS}'_1 - \mathbf{t}_1\right) \circ \boldsymbol{\Delta} + \left(\mathsf{QS}'_0 - \mathbf{t}_0\right) \in \mathbb{F}_{p^r}^{\tau}$$

Since we are computing $\epsilon_{\Pi_{\mathsf{MQ-DVZK}}}$, Verifier accepts the proof, i.e., $\mathbf{B} = \mathsf{QS}'_1 \circ \boldsymbol{\Delta} + \mathsf{QS}'_0$, it means

$$\left(\sum_{i=1}^{m} y_i \cdot \chi_i\right) \circ \boldsymbol{\Delta}^2 + \mathbf{t}_1 \circ \boldsymbol{\Delta} + \mathbf{t}_0 = 0 \tag{2}$$

The Equation (2) holds if

1. $\sum_{i=1}^{m} y_i \cdot \chi_i = 0$ and malicious prover sets $\mathbf{t}_0 = \mathbf{t}_1 = 0$. However, $\mathbf{x}$ is not a valid witness so there exists at least one y_i such that $y_i \neq 0$ this leads to $\sum_{i=1}^{m} y_i \cdot \chi_i = 0$ with a probability of $1/p^{r\tau}$ since $\chi_i \in \mathbb{F}_{p^r}^{\tau}$ and is sampled before $\mathbf{x}$ is committed using $\mathcal{F}_{\mathsf{sVOLE}}$.
2. $\sum_{i=1}^{m} y_i \cdot \chi_i \neq 0$, from lemma 3.5,the Equation (2) holds with probability of $(2/p^r)^{\tau}$ since $\boldsymbol{\Delta} \leftarrow_r \mathbb{F}_{p^r}^{\tau}$ is kept secret from prover's view and each Δ_i is sampled independent for $i \in [\tau]$.

To conclude, the soundness $\epsilon_{\Pi_{\mathsf{MQ-DVZK}}}$ is bounded by $(3/p^r)^{\tau}$.

Malicious Verifier. Sim emulates $\mathcal{F}_{\mathsf{polyZK}}$. If Sim receives false from $\mathcal{F}_{\mathsf{polyZK}}$, 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 $\boldsymbol{\Delta}$ and the keys $\mathbf{K}[\mathbf{x}]$ for all the authenticated values, which are received from $\mathcal{A}$. Additionally, S also receives $\mathbf{B}^* = \mathbf{A}_0^* + \mathbf{A}_1^* \circ \boldsymbol{\Delta} \in \mathbb{F}_{p^r}^{\tau}$.
- Sim executes the step 1 of online phase in $\Pi_{\mathsf{MQ-DVZK}}$ by sending uniform $\gamma \in \mathbb{F}_p^n$ to $\mathcal{A}$.
- Sim receives $\{\chi_i\}_{i \leq m}$ from $\mathcal{A}$.
- For steps 5–6 of $\Pi_{\mathsf{MQ-DVZK}}$, Sim computes $\mathbf{W}, \mathbf{B}_i, \mathbf{B}$, by using $\boldsymbol{\Delta}$, the keys $\mathbf{K}[\mathbf{x}]$, the challenge $\{\chi_i\}_{i \leq m}$, 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 $\mathsf{QS}_0 := \mathbf{B} - \mathsf{QS}_1 \circ \boldsymbol{\Delta}$. 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. □

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 $\chi_i \in \mathbb{F}_{p^r}^{\tau}$ 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.

4.2 Publicly-Verifiable Zero Knowledge for Multivariate Quadratic problem

In this section, we show how to transform our designated-verifier ZK protocol $\Pi_{\mathsf{MQ-DVZK}}$ (Figure 5) to publicly-verifiable ZK protocol using SoftspokenOT and Vector Commitment by VOLE in-the-Head paradigm.

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:

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

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 $f_i(x_1, \cdots, x_n) = \mathbf{x}^T \mathbf{A}_i \mathbf{x} + \mathbf{b}_i^T \mathbf{x}$ 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
     $$\mathbf{g}_i(X) = f_{i,1}(\mathbf{M}[\mathbf{x}] + \mathbf{x}\cdot[1\cdots1]\cdot\mathsf{diag}(X))\cdot X + f_{i,2}(\mathbf{M}[\mathbf{x}] + \mathbf{x}\cdot[1\cdots1]\cdot\mathsf{diag}(X)) - y_i\cdot[1\cdots1]\cdot X^2,$$
     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 $\chi_i \leftarrow_r \mathbb{F}_{p^r}^\tau$ for $i \in [m]$ and sends them 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 $\mathbf{A}_0^*$, $\mathbf{A}_1^*$ are considered 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:
     $$\mathbf{W} = \begin{bmatrix}\mathbf{W}_1 \\ \mathbf{W}_2\end{bmatrix},\ \mathbf{W}_1 := \mathbf{V}_1 + \gamma \cdot [1\cdots1] \cdot \mathsf{diag}(\boldsymbol{\Delta}),$$
     $$\mathbf{W}_2 \ \longrightarrow \mathbf{B}^* = \mathbf{A}_0^* + \mathbf{A}_1^* \circ \boldsymbol{\Delta} \in \mathbb{F}_{p^r}^\tau.$$
     V defines $\mathbf{W}_1 \to \mathbf{K}[\mathbf{x}]$.
  6. For $i \in [1, m]$:
     - V computes $\mathbf{B}_i := f_{i,1}(\mathbf{K}[\mathbf{x}]) \circ \boldsymbol{\Delta} + f_{i,2}(\mathbf{K}[\mathbf{x}]) - y_i \cdot \boldsymbol{\Delta}^2$.
  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

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 multi-instance sVOLE is defined by concatenating each instance.
P defines:

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

While V defines:

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

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

$$\mathbf{W} = \mathbf{V} + \mathbf{u}\cdot[1\ldots1]\cdot\mathsf{diag}(\boldsymbol{\Delta}).$$

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 $\mathsf{H}^{\mathsf{UHF}}$, 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 $\tilde{\mathbf{V}} + \tilde{\mathbf{u}}\cdot[1\cdots1]\cdot\mathsf{diag}(\boldsymbol{\Delta}) \stackrel{?}{=} \mathsf{H}^{\mathsf{UHF}}\cdot(\mathbf{W} + [0\|\mathbf{C}]\cdot\mathsf{diag}(\boldsymbol{\Delta}))$. The probability for P to cheat is negligible bounded by $\binom{\tau}{2}$ [Roy22, BBD+23] and we denote it as ϵ_{Hash}.

Vector Commitment The initial construction of Vector Commitment is based on GGM-tree PPRF [KKW18]. Recently, to construct efficient signatures, [CLY+24, BCdSG24] optimized GGM tree-based PPRF in terms of computation by using the half-tree technique and circular correlation robust hash function. To instantiate SoftspokenOT-based sVOLE, we propose a new construction of vector commitment from multi-instance PPRF (Figure 4), its security has relied on the random oracle and the randomness property of multi-instance PPRF. We prove our construction satisfies multi-hiding and extractable-binding properties in Theorem 17 and Theorem 18 respectively.

Multi-instance PPRF. The construction is constructed by using τ-PPRF trees using the same key K. The usage of multi-instance PPRF has three advantages 1) the security shown is tighter compared to using τ separately PPRF trees under the constrain using the same key [BCC+24] 2) the concept can be directly adapted to $\mathsf{EUF\text{-}CMA}$ proof security of signature (where $\mathcal{A}$ can make polynomial times queries to signing oracle) and 3) the existing instantiation of multi-instance PPRF based on fix-key AES is more efficient compared to others GGM-tree style PPRF when plugging to the signature [BCC+24].

Multi-instance Vector Commitment Construction. We present our vector commitment $\mathsf{VC}(N,\tau)$ in Figure 4, informally we have τ-trees having N leaves later V knows all-but-τ leaves (all-but-one for each tree) and P needs to convince V about the knowledge of τ-trees with a negligible advantage of cheating after committing.
Our trees are expanded from random seeds using PPRF and for the last leaves which are used to define $(\mathsf{sd}_j^i, \mathsf{com}_j^i)_{i\in[0,\tau),j\in[0,N)}\}$, we use a random oracle to 1) make sure the adversary in hiding game learns nothing about the secret leaves even this adversary can query more than one query and 2) allow us to define the bit-length of commitment for each sd of being 2λ this leads to the extractor in the binding game can not find the collision with high probability.

To fit into signature syntax where the adversary can query Q-times to signing oracle, we prove our $\mathsf{VC}(N,\tau)$ is secure in Q-multi-hiding game that is defined as Definition 14 except adversary making Q queries and each query giving adversary information of all-but-τ vector commitment.

Theorem 17 (Multi-hiding). *Assume that* PPRF *is a* (Q,τ)*-instance* $(t,\epsilon_{\mathsf{PPRF}})$*-secure PPRF,* PRG,H *are pseudorandom generator with output domains of* $\tau\lambda, 3\lambda$ *bits respectively. Then, for any PPT adversary* $\mathcal{A}$ *in the adaptive* Q*-multi-hiding vector commitment game against the* $\mathsf{VC}^{\mathsf{H},\mathsf{H}_1}(N,\tau)$ *scheme, we have*

$$\mathcal{A}_{\mathcal{A},\mathsf{VC}}^{\mathsf{MulHiding}}[Q] \leq Q\cdot\epsilon_{\mathsf{PRG}^{\tau\lambda}} + \epsilon_{\mathsf{PPRF}} + Q\cdot\tau\cdot\epsilon_{\mathsf{PRG}^{3\lambda}}$$

PARAMETERS:

- Tree depth $N = 2^d \in \mathbb{N}$, computational security parameter λ.
- A repetition parameter $\tau \in \mathbb{N}$.
- $\mathsf{PRG} : \{0,1\}^* \to \{0,1\}^{\tau\lambda}, \mathsf{H} : \{0,1\}^\lambda \to \{0,1\}^\lambda \times \{0,1\}^{2\lambda}$ pseudorandom generators and $\mathsf{H}_1 : \{0,1\}^* \to \{0,1\}^{2\lambda}$ collision-resistant hash functions.
- H, H_1 are modeled as random oracles.
- An instantiation of multi-instance of PPRF.

CONSTRUCTION:

- $\mathsf{Commit}(\mathsf{seed}, \mathsf{K}, 1^\lambda)$:
  1. Compute $\mathsf{seed}^0 \| \ldots \| \mathsf{seed}^{\tau-1} \leftarrow \mathsf{PRG}(\mathsf{seed}; \tau\lambda)$.
  2. For $i \in [0, \tau)$:
     - For $j \in [0, N)$, compute:
       (a) $\mathsf{seed}^i_j \leftarrow \mathsf{PPRF}_\mathsf{K}(\mathsf{seed}^i, j)$.
       (b) $(\mathsf{sd}^i_j, \mathsf{com}^i_j) \leftarrow \mathsf{H}(\mathsf{seed}^i_j)$.
     - Compute $\mathsf{com}^i \leftarrow \mathsf{H}_1(\mathsf{com}^i_0, \ldots, \mathsf{com}^i_{N-1})$.
  3. Compute $h_\mathsf{com} \leftarrow \mathsf{H}_1(\mathsf{com}^0 \| \ldots \| \mathsf{com}^{\tau-1})$. Let $\mathsf{decom} := (\mathsf{seed}, \mathsf{K})$.
  4. Output $(h_\mathsf{com}, \mathsf{decom}, (\mathsf{sd}^i_j)_{i\in[0,\tau), j\in[0,N)})$.
- $\mathsf{Open}\left(\mathsf{decom}, I = \cup_{i=0}^{\tau-1}[0,N) \setminus \{\Delta_i\}\right)$:
  1. For $i \in [0, \tau)$, compute:
     (a) Write $I^i = [0, N) \setminus \{\Delta_i\}$.
     (b) Compute $\mathsf{pdecom}_i \leftarrow (\mathsf{K}, \mathsf{CoPath}_\mathsf{K}(\mathsf{seed}^i, \Delta_i), \mathsf{com}^i_{\Delta_i})$.
  2. Output the all-but-τ opening $\mathsf{pdecom}_I := (\mathsf{pdecom}_i)_{i\in[0,\tau)}$.
- $\mathsf{Verify}\,(h_\mathsf{com}, I, \mathsf{pdecom}_I)$:
  1. For $i \in [0, \tau)$, compute:
     (a) Write $I^i = [0, N) \setminus \{\Delta_i\}$.
     (b) Recompute $(\mathsf{sd}^i_j, \mathsf{com}^i_j)_{j\in I^i}$ from pdecom_i using PPRF_K.
     (c) Compute $\mathsf{com}^i \leftarrow \mathsf{H}_1(\mathsf{com}^i_0 \| \ldots \| \mathsf{com}^i_{N-1})$.
  2. Compute $h'_\mathsf{com} \leftarrow \mathsf{H}_1(\mathsf{com}^0 \| \ldots \| \mathsf{com}^{\tau-1})$.
  3. If $h'_\mathsf{com} \neq h_\mathsf{com}$, output $\perp$; otherwise output $(\mathsf{sd}^i_j)_{i\in[0,\tau), j\in[I^i]}$.

Fig. 4. Construction $\mathsf{VC}^{\mathsf{H},\mathsf{H}_1}(N, \tau)$ of Multi-instance of Vector Commitment

Sketch proof. From the randomness of PRG, we step by step replace all values of intermediate nodes of the hidden path (from the root of the hidden leaves) with random values. In particular, first of all, we replace the set of seeds that are used to define the root of τ-trees with random values, i.e., $(\mathsf{seed}^0 \| \ldots \| \mathsf{seed}^{\tau-1}) \leftarrow \{0,1\}^{\tau\lambda}$, the advantage of $\mathcal{A}$ in this step is $Q \cdot \epsilon_{\mathsf{PRG}^{\tau\lambda}}$, then later for all intermediate levels of i-th PPRF tree ($i \in [0, \tau)$) we replace the PRG calls along the path of seed^i to Δ_i by truly random values, since the PRG used to construct PPRF is a (Q, τ)-instance $(t, \epsilon_\mathsf{PRG})$-secure PRG then the advantage of $\mathcal{A}$ after replacing all nodes in the hidden path is ϵ_PPRF (we note the term Q is omitted since PPRF is Q-multi-instance secure PPRF). After this step all $(\mathsf{seed}^i_j)_{i\in[0,\tau), j\in[0,N)}$ are random values and they are indistinguishable from the actual values. Since H we continue to replace $(\mathsf{sd}^i_j)_{i\in[0,\tau), j\in[0,N)}$ by truly random values and the advantage of $\mathcal{A}$ to distinguish this step and previous one is $Q \cdot \tau \cdot \epsilon_{\mathsf{PRG}^{3\lambda}}$.

Proof. We proceed in a sequence of hybrids where each hybrid relies on the randomness of PRGs. We recall that PPRF is a (Q, τ)-instance $(t, \epsilon_\mathsf{PPRF})$-secure PPRF, PRG, H are pseudorandom generators with the advantages of $\epsilon_{\mathsf{PRG}^{\tau\lambda}}, \epsilon_{\mathsf{PRG}^{3\lambda}}$ respectively.

- **Experiment 0** ($\mathsf{Exp}^\mathbf{0}$). All τ-trees of each k-th instance ($k \leq Q$) are obtained through the actual scheme described in Figure 4, first of all, the root of trees $(\mathsf{seed}^i)_{i\in[0,\tau)}$ is generated from a PRG later the expansion is built by applying $\mathsf{PPRF}(\mathsf{seed}^i)$ to generate the N leaves for each tree and to construct the $(\mathsf{sd}^i_j, \mathsf{com}^i_j)_{i\in[0,\tau), j\in[0,N)}$ a PRG^* used. In the end, $\mathcal{A}$ knows all sd^i_j except $(\mathsf{sd}^i_{\Delta_i})_{i\in[0,\tau)}$.
  Experiment 1 ($\mathsf{Exp}^\mathbf{1}$). Same as the previous experiment, the only difference is at the root of each tree. For all $i \in [0, \tau)$, the root of each tree is not generated with PRG, but is instead randomly

sampled $(\mathsf{seed}^i)_{i\in[0,\tau)} \to \{0,1\}^{\tau\lambda}$. Since PRG is an $(t, \epsilon_{\mathsf{PRG}^{\tau\lambda}})$-secure PRG, then

$$|\Pr[\mathsf{Exp}^0(\lambda)=1] - \Pr[\mathsf{Exp}^1(\lambda)=1]| \le Q \cdot \epsilon_{\mathsf{PRG}^{\tau\lambda}}$$

Experiment 2 (Exp^2). The difference with the previous experiment is in the hidden path of Δ_i to the root of each tree: all the intermediate nodes are now randomly chosen from $\{0,1\}^\lambda$. Using the secure property that PPRF is a (Q,τ)-instance $(t, \epsilon_{\mathsf{PPRF}})$-secure PPRF, we obtain

$$|\Pr[\mathsf{Exp}^1(\lambda)=1] - \Pr[\mathsf{Exp}^2(\lambda)=1]| \le \epsilon_{\mathsf{PPRF}}$$

Experiment 3 (Exp^3). In the last experiment, all nodes on the co-path to $(\Delta_i)_{i\in[0,\tau)}$ as well as the leaf $(\mathsf{seed}^i_{\Delta_i})_{i\in[0,\tau)}$ are picked uniformly at random. This experiment is the same as the previous one except the value $(\mathsf{sd}^i_{\Delta_i})_{i\in[0,\tau)}$ is chosen randomly instead of generating from H. In total, the number of $(\mathsf{sd}^i_{\Delta_i})_{i\in[0,\tau)}$ needed to be replaced is $Q\cdot\tau$ (Q instances of $\mathsf{VC}(N,\tau)$). Since PRG is an $(t, \epsilon_{\mathsf{PRG}^{3\lambda}})$-secure PRG, we have

$$|\Pr[\mathsf{Exp}^0(\lambda)=1] - \Pr[\mathsf{Exp}^3(\lambda)=1]| \le Q \cdot \tau \cdot \epsilon_{\mathsf{PRG}^{3\lambda}}$$

which concludes the proof. □

Theorem 18 (Extractable-binding). *Let H, H_1 be random oracles, for any PPT adversary $\mathcal{A}^{\mathsf{H},\mathsf{H}_1}$ making $|Q|$ queries to H and $|Q_1|$ queries to H_1, there exists an extractor Ext such that*

$$\mathsf{Adv}^{\mathsf{ExtBinding}}_{\mathcal{A},\mathsf{VC}} \le \frac{(|Q_1|+\tau+1)^2 + (|Q|+\tau N)^2}{2^{2\lambda}}$$

Sketch proof. We build a straight line extractor from observing the list of queries to $\mathcal{A}$ to two random oracles H, H_1. To be clear, the maximum total numbers of queries that $\mathcal{A}$ can make to oracles H and H_1 are $|Q|+\tau N$ and $|Q_1|+\tau+1$.

Proof. From the list of queries to H and H_1, we construct an extractor Ext in the following way:

1. Ext finds the pre-image $\mathsf{com}^0\|\ldots\|\mathsf{com}^{\tau-1}$ of h_{com} from the list of queries Q_1 if it does not exist or is multiply-defined, then Ext outputs $\perp$ for all messages. $\mathcal{A}$ wins if there exists at least one pre-image or a collision for H in Q. And this probability is bounded by $|Q_1|^2/2^{2\lambda}$.
2. Ext finds the pre-image $\mathsf{com}^i_0\|\ldots\|\mathsf{com}^i_{N-1}$ of com^i for $i\in[0,\tau)$; if it does not exist, or is multiply-defined, then Ext outputs $\perp$ for all messages from the i-th tree. The probability of $\mathcal{A}$ to win for each com^i is bound by $\tau|Q_1|^2/2^{2\lambda}$.
3. Now for each com^i_j for $i\in[0,\tau), j\in[0,N)$, using the list of queries in Q, Ext extracts the leaf seed^i_j, Ext defines sd^i_j as $(\mathsf{sd}^i_j, \mathsf{com}^i_j) = \mathsf{H}(\mathsf{seed}^i_j)$, otherwise it sets $\mathsf{sd}^i_j = \perp$. The probability of $\mathcal{A}$ to win for each sd^i_j is bound by $(|Q|+\tau N)^2/2^{2\lambda}$.

We explain the final step in detail as follows: From each com^i_j for $i\in[0,\tau), j\in[0,N)$, Ext uses Q to extract the tree leaf seed^i_j. We note that the bit length of com^i_j is 2λ bits.

- If there is an unique seed^i_j, Ext defines sd^i_j as $(\mathsf{sd}^i_j, \mathsf{com}^i_j) = \mathsf{H}(\mathsf{seed}^i_j)$.
- If seed^i_j does not exist in the list of Q, or it is found multiple times, then Ext sets $\mathsf{sd}^i_j = \perp$. If any com^i_j has more than one preimage, then this contradicts the collision-resistance of H; therefore the probability that this happens is bound by $(|Q|+\tau N)^2/2^{2\lambda}$.

□

VOLE in-the-Head. Putting all techniques together and using the compiler from [BBD+23], we obtain a publicly verifiable ZK protocol $\Pi_{\mathsf{MQ-PVZK}}$ from MQ problem in Figure 5 based on SoftspokenOT, multi-instance PPRF, and nullity check for a polynomial set.

Theorem 19. *Let PRG be pseudorandom generator, $\mathcal{H}^{\mathsf{UHF}} \subseteq \mathbb{F}_p^{t\times(k+t)}$ be a family of k-hiding, p^{-t}-universal hash function, $\mathsf{VC}(N,\tau)$ be multi-instances vector commitment with $\mathsf{Adv}^{\mathsf{mulHiding}}_{\mathcal{A},\mathsf{VC}}, \mathsf{Adv}^{\mathsf{ExtBinding}}_{\mathcal{A},\mathsf{VC}}$ being advantages of $\mathcal{A}$ in multi-hiding and extractable binding games respectively, and $\Pi_{\mathsf{MQ-DVZK}}$ (Figure 3) be a designated-verifier ZK protocol with soundness error $\epsilon_{\Pi_{\mathsf{MQ-DVZK}}}$. The protocol $\Pi_{\mathsf{MQ-PVZK}}$ (Figure 5) is a publicly-verifiable ZK protocol for multivariate quadratic problem $\mathsf{MQ}_{p,m,n}$ satisfying the following properties.*

- *Knowledge soundness with soundness error $\epsilon_{\Pi_{\mathsf{MQ-PVZK}}}$ bounded by $\epsilon_{\mathsf{Hash}} + \epsilon_{\Pi_{\mathsf{MQ-DVZK}}} + \mathsf{Adv}^{\mathsf{ExtBinding}}_{\mathcal{A},\mathsf{VC}}$*
- *Special honest-verifier ZK.*

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.
- An instantiation of vector commitment $\mathsf{VC}(N, \tau)$.

PROTOCOL:

1. P samples $\mathsf{seed} \leftarrow_r \{0,1\}^\lambda$ and establishes $\mathsf{VC}(N,\tau)$:
   - $\mathsf{VC.Commit}(\mathsf{seed}, 1^\lambda) \to (h_{\mathsf{com}}, \mathsf{decom}, (\mathsf{sd}_j^i)_{i\in[0,\tau), j\in[0,N)})$.
2. From $\{\mathsf{sd}_j^i\} \leftarrow_r \{0,1\}^\lambda$ for $i \in [0,\tau), j \in [0,N)$, P defines $\mathbf{U}, \mathbf{V} \in \mathbb{F}_p^{(k+h)\times\tau}$ as follow
$$\mathbf{U} = \left[\sum_{j=0}^{N-1}\mathsf{PRG}(\mathsf{sd}_j^0)\cdots\sum_{j=0}^{N-1}\mathsf{PRG}(\mathsf{sd}_j^{\tau-1})\right],\ \mathbf{V} = -\left[\sum_{j=0}^{N-1} j\cdot\mathsf{PRG}(\mathsf{sd}_j^0)\cdots\sum_{j=0}^{N-1} j\cdot\mathsf{PRG}(\mathsf{sd}_j^{\tau-1})\right]$$
   P defines $\mathbf{u} = \mathbf{U}_0$ and $\mathbf{C} := [\mathbf{U}_1 - \mathbf{u}\|\cdots\|\mathbf{U}_{\tau-1} - \mathbf{u}] \in \mathbb{F}_p^{(k+t)\times(\tau-1)}$, then sends $(\mathbf{C}, h_{\mathsf{com}})$ to V.
3. V samples $\mathsf{H}^{\mathsf{UHF}} \leftarrow_r \mathcal{H}^{\mathsf{UHF}}$ and sends it to P.
4. 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 V.
5. 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 $\mathbf{V}$.
6. V samples randomly $\boldsymbol{\Delta} = (\Delta_0, \cdots, \Delta_{\tau-1})$, sends $(\mathsf{get}, \boldsymbol{\Delta})$ to P and receives back $\mathsf{pdecom}_I \leftarrow \mathsf{VC.Open}\left(\mathsf{decom}, I = \cup_{i=0}^{\tau-1}[0,N)\setminus\{\Delta_i\}\right)$.
7. V verifies vector commitment VC if
   - $\mathsf{VC.Verify}\left(h_{\mathsf{com}}, I, \mathsf{pdecom}_I\right) = \perp$, then **abort**.
   - $\mathsf{VC.Verify}\left(h_{\mathsf{com}}, I, \mathsf{pdecom}_I\right) = (\mathsf{sd}_j^i)_{i\in[0,\tau), j\in[I^i]}$ then V computes
$$\mathbf{W} = \left[\sum_{j=0}^{N-1}(j-\Delta_0)\cdot\mathsf{PRG}(\mathsf{sd}_j^0)\cdots\sum_{j=0}^{N-1}(j-\Delta_{\tau-1})\cdot\mathsf{PRG}(\mathsf{sd}_j^{\tau-1})\right]$$
8. P and V run step $5-6$ of $\Pi_{\mathsf{MQ-DVZK}}$ in Figure 3.
9. V outputs accept if two following checks are passed:
   - **SoftSpokenOT.** $\tilde{\mathbf{V}} + \tilde{\mathbf{u}}\cdot[1\cdots1]\cdot\mathsf{diag}(\boldsymbol{\Delta}) = \mathsf{H}^{\mathsf{UHF}}\cdot(\mathbf{W} + [0\|\mathbf{C}]\cdot\mathsf{diag}(\boldsymbol{\Delta}))$.
   - **QuickSilver.** P and V run step 7 of $\Pi_{\mathsf{MQ-DVZK}}$ in Figure 3.

Fig. 5. The publicly verifiable zero-knowledge protocol $\Pi_{\mathsf{MQ-PVZK}}$ for Multivariate Quadratic

Sketch proof. The correctness of the proof follows the correctness of vector commitment VC, the underlying designated-verifier ZK protocol $\Pi_{\mathsf{MQ-DVZK}}$ and SoftspokenOT. For security, we construct a knowledge extractor Ext that extracts the witness of a malicious prover and a simulator Sim that simulates the view of a semi-honest verifier to argue soundness and zero-knowledge properties respectively.

- The Ext is constructed from the $\mathsf{VC.Ext}$ of vector commitment VC and the witness is extracted using $\mathsf{Sim}^{\mathsf{P}}_{\Pi_{\mathsf{MQ-DVZK}}}$ in proof Theorem 16. Therefore the soundness error $\epsilon_{\Pi_{\mathsf{MQ-PVZK}}}$ bounded by $\epsilon_{\mathsf{Hash}} + \epsilon_{\Pi_{\mathsf{MQ-DVZK}}} + \mathsf{Adv}^{\mathsf{ExtBinding}}_{\mathcal{A},\mathsf{VC}}$ (ϵ_{Hash} is the probability that $\mathcal{A}$ can cheat in SpoftspokenOT check using universal hash function).

– For special-honest verifier zero-knowledge. The Sim is built from $\mathsf{Sim}^{\mathsf{V}}_{\Pi_{\mathsf{MQ-DVZK}}}$ in proof Theorem 16. And from the multi-hiding property of VC and ZK property of $\Pi_{\mathsf{MQ-DVZK}}$, Sim can simulate the view of a semi-honest verifier in $\Pi_{\mathsf{MQ-PVZK}}$.

Proof. We argue about correctness, knowledge soundness, and special-honest verifier ZK properties as follows.

Correctness. If both the prover and verifier are honest then the protocol is correct i.e., the verifier valid all the checks if the prover has a valid witness since the correctness of vector commitment VC, the underlying designated-verifier ZK protocol $\Pi_{\mathsf{MQ-DVZK}}$ and the SoftspokenOT technique.

Knowledge Extractor. Since VC is extractable biding then there exists an extractor $\mathsf{Ext}_{\mathsf{VC}}(\mathsf{pp}, h_{\mathsf{com}}, Q)$ (Q is the list of queries that $\mathcal{A}$ made to random oracle of VC) that can extract the set of seed $(\mathsf{sd}^i_j)_{i\in[0,\tau),j\in[0,N)}$ if $\mathsf{VC.Verify}$ is valid with the failure probability of $\mathsf{Adv}^{\mathsf{ExtBinding}}_{\mathcal{A},\mathsf{VC}}$. In addition, $\Pi_{\mathsf{MQ-DVZK}}$ is secure against a malicious prover in the UC model then there exists a simulator $\mathsf{Sim}^{\mathsf{P}}_{\Pi_{\mathsf{MQ-DVZK}}}$ that can extract correctly the witness of Adv with a failure probability of $\epsilon_{\Pi_{\mathsf{MQ-PVZK}}}$.

We build a knowledge extractor Ext from $\mathsf{Ext}_{\mathsf{VC}}$ and $\mathsf{Sim}^{\mathsf{P}}_{\Pi_{\mathsf{MQ-DVZK}}}$ that simulates the view of malicious prover as following:

– When $\mathcal{A}$ sends a message $(\mathbf{C}, h_{\mathsf{com}})$ to V, Ext runs $\mathsf{VC.Ext}(\mathsf{pp}, h_{\mathsf{com}}, Q) \to (\mathsf{sd}^j_i)$ for $i \in [0,\tau), j \in [0,N)$.
– From the set of $(\mathsf{sd}^j_i)_{i\in[0,\tau),j\in[0,N)}$, Ext computes $\mathbf{U}, \mathbf{V}$ by the same way as an honest P does. From $\mathbf{U}$, Ext defines $\mathbf{u} := \mathbf{U}_0$ and checks if $\mathbf{C} \stackrel{?}{=} [\mathbf{U}_1 - \mathbf{u} \| \cdots \| \mathbf{U}_{\tau-1} - \mathbf{u}]$. If the check is valid then continue the next step otherwise outputs *fail*.
– Following the protocol, in step 5, Ext run $\mathsf{Sim}^{\mathsf{P}}_{\Pi_{\mathsf{MQ-DVZK}}}$ by using first k-coordinates of $\mathbf{u}$ and k-rows of $\mathbf{V}$ which are constructed from previous step. From the proof Theorem 16, $\mathsf{Sim}^{\mathsf{P}}_{\Pi_{\mathsf{MQ-DVZK}}}$ can extract a witness $\mathbf{x}$ of malicious prover. Ext run Quicksilver check (step 7 of $\Pi_{\mathsf{MQ-DVZK}}$) if $\Pi_{\mathsf{MQ-DVZK}}$ outputs *reject* then outputs *fail* otherwise, Ext continues next step.
– Ext outputs *fail* if $\mathsf{VC.Verify}\,(h_{\mathsf{com}}, I, \mathsf{pdecom}_I) = \perp$, otherwise, Ext outputs $\mathbf{x}$ as a witness of P.

We argue by game hops.

1. Game G_1 is the real game which outputs 1 if the V accepts.
2. Game G_2 is the same as game 1 except we extract all $(\mathsf{sd}^j_i)_{i\in[0,\tau),j\in[0,N)}$ and check whether the correlation between $\mathbf{C}, \mathbf{V}$ is correct. From SoftspokenOT check [Roy22], where $\mathcal{H}^{\mathsf{UHF}} \subseteq \mathbb{F}_p^{t\times(k+t)}$ be a family of k-hiding, p^{-t}-universal hash function then
$$|\Pr[G_1 = 1] - \Pr[G_2 = 1]| \leq \epsilon_{\mathsf{Hash}}$$
and ϵ_{Hash} is negligible.
3. Game G_3, Ext runs $\mathsf{Sim}^{\mathsf{P}}_{\Pi_{\mathsf{MQ-DVZK}}}$ to extract the witness $\mathbf{x}$ of P and output whatever Quicksilver check (step 7 of $\Pi_{\mathsf{MQ-DVZK}}$) outputs. Then
$$|\Pr[G_2 = 1] - \Pr[G_3 = 1]| \leq \epsilon_{\Pi_{\mathsf{MQ-DVZK}}}$$
4. Game G_4, Ext fails if $\mathsf{VC.Verify}$ fails but since VC is extractable biding then
$$|\Pr[G_3 = 1] - \Pr[G_4 = 1]| \leq \mathsf{Adv}^{\mathsf{ExtBinding}}_{\mathcal{A},\mathsf{VC}}$$

Putting all games together we have:

$$\epsilon_{\Pi_{\mathsf{MQ-PVZK}}} \leq \epsilon_{\mathsf{Hash}} + \epsilon_{\Pi_{\mathsf{MQ-DVZK}}} + \mathsf{Adv}^{\mathsf{ExtBinding}}_{\mathcal{A},\mathsf{VC}}$$

Special-honest verifier zero-knowledge. The simulator Sim is build from the simulator $\mathsf{Sim}^{\mathsf{V}}_{\Pi_{\mathsf{MQ-DVZK}}}$ (against malicious verifier) in the proof Theorem 16 as follows:

1. Sim follows the protocol as an honest prover, i.e., computes $\mathsf{VC}(N,\tau)$ from a random seed and then defines $\mathbf{U}, \mathbf{V}, \mathbf{C}$. Sim sends $(\mathbf{C}, h_{\mathsf{com}})$ to $\mathcal{A}$. Since VC is multi-hiding this hybrid is distinguished from the real protocol with a negligible probability $\mathsf{Adv}^{\mathsf{mulHiding}}_{\mathcal{A},\mathsf{VC}}$.
2. Sim receives $\boldsymbol{\Delta}$ of $\mathcal{A}$, then from $\mathbf{u}, \mathbf{V}$; Sim computes
$$\mathbf{W} = \mathbf{V} + \mathbf{u} \cdot [1 \dots 1] \cdot \mathsf{diag}(\boldsymbol{\Delta})$$
Sim then runs $\mathsf{Sim}^{\mathsf{V}}_{\Pi_{\mathsf{MQ-DVZK}}}$ with input of $\boldsymbol{\Delta}, \mathbf{W}$ to produce a simulated view of $\mathcal{A}$ for $\Pi_{\mathsf{MQ-DVZK}}$ protocol. This hybrid is indistinguishable from the previous one by the zero-knowledge property of $\Pi_{\mathsf{MQ-DVZK}}$.

5 A Signature scheme from Multivariate Quadratic

5.1 Description of the Signature Scheme

In this section, we introduce a new signature scheme from the multivariate quadratic decoding assumption. A signature scheme is given by three algorithms $(\mathsf{KeyGen}, \mathsf{Sign}, \mathsf{Verify})$. The KeyGen algorithm returns a key pair $(\mathsf{pk}, \mathsf{sk})$ where pk and sk are the public and private key. The Sign algorithm on an input a message m and the secret key sk, produces a signature σ. The Verify algorithm, 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 21) and against chosen-message attacks ($\mathsf{EUF\text{-}CMA}$, Definition 20).

Definition 20 (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*

$$\mathsf{Adv}^{\mathsf{EUF\text{-}CMA}}_{\mathcal{A}} = \Pr\left[\begin{array}{c}\mathsf{Verify}(\mathsf{pk}, \mu^*, \sigma^*) = 1 \\ \wedge \mu^* \notin Q\end{array}\middle|\begin{array}{c}(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{Setup}(\{0,1\}^\lambda) \\ (\mu^*, \sigma^*) \leftarrow \mathcal{A}^{\mathsf{Sign}(\mathsf{sk}, \cdot)}(\mathsf{pk})\end{array}\right],$$

where $\mathcal{A}^{\mathsf{Sign}(\mathsf{sk}, \cdot)}$ *denotes* $\mathcal{A}$*'s access to a signing oracle* $\mathcal{O}_{\mathsf{Sign}}$ *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 21 (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*

$$\mathsf{Adv}^{\mathsf{EUF\text{-}KO}}_{\mathcal{A}} = \Pr\left[\mathsf{Verify}(\mathsf{pk}, \mu^*, \sigma^*) = 1 \middle| \begin{array}{c}(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{Setup}(\{0,1\}^\lambda) \\ (\mu^*, \sigma^*) \leftarrow \mathcal{A}(\mathsf{pk})\end{array}\right].$$

5.2 Description of the Signature Scheme

The key generation algorithm randomly samples a multivariate quadratic instance $((\mathbf{A}_i, \mathbf{b}_i, y_i)_{i \le m})$ with solution $\mathbf{x} \in \mathbb{F}_p^n$. We describe it on Figure 6. The signing algorithm with secret key $\mathsf{sk} = (\mathsf{seed}, \mathbf{x})$ and message $\mathsf{m} \in \{0,1\}^*$ is described on Figure 7. The verification algorithm with public key $\mathsf{pk} = (\mathbf{A}_i, \mathbf{b}_i, y_i)_{i \le m}$, message $\mathsf{m} \in \{0,1\}^*$, and signature σ, is described in Figure 8.

Inputs:

- A security parameter λ, a finite field $\mathbb{F}_p$.
- An instance of MQ problem with parameters $\mathsf{MQ}_{p,n,n}$.
- Pseudorandom generators $\mathsf{PRG}_1 : \{0,1\}^\lambda \to (\mathbb{F}_p^{n \times n})^n, \mathsf{PRG}_2 : \{0,1\}^\lambda \to (\mathbb{F}_p^n)^n$.

Key Gen:

1. Sample $\mathsf{sd} \leftarrow_r \{0,1\}^\lambda$. Compute $(\mathbf{A}_i)_{i \le n} \leftarrow \mathsf{PRG}_1(\mathsf{sd})$, $(\mathbf{b}_i)_{i \le n} \leftarrow \mathsf{PRG}_2(\mathsf{sd})$.
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 $\mathsf{pk} \leftarrow (\mathsf{sd}, (y_i)_{i \le n})$ and $\mathsf{sk} \leftarrow (\mathsf{sd}, \mathbf{x})$.

Fig. 6. Key generation algorithm of the signature scheme

Theorem 22. *Assume that* $\mathsf{VC}^{\mathsf{H}, \mathsf{H}_0}$ *is a* q_s*-multi-hiding and extractable biding vector commitment and that any adversary running in time* t *has an advantage at most* ϵ_{MQ} *against the multivariate quadratic problem;* $\Pi_{\mathsf{MQ-PVZK}}$ *is a PVZK protocol with a soundness error of* $\epsilon_{\Pi_{\mathsf{MQ-PVZK}}}$*. Model the hash functions* $\mathsf{H}, \mathsf{H}_1, \mathsf{H}_2$ *as random oracles with output of length* 3λ*-bit,* 2λ*-bit respectively and the pseudorandom generator* PRG_2^* *as a random oracle. Then chosen-message adversary against the signature scheme*

Inputs: A secret key $\mathsf{sk} = (\mathsf{sd}, \mathbf{x})$ and a message $\mathsf{m} \in \{0,1\}^*$.
Parameters:

- A finite field $\mathbb{F}_p$ where $\mathsf{MQ}_{p,n,n}$ holds; $r \in \mathbb{N}$, $N = p^r$.
- $k, t \in \mathbb{N}$ such that $\mathcal{H}^{\mathsf{UHF}} \subseteq \mathbb{F}_p^{t\times(k+t)}$ be a family of k-hiding, p^{-t}-universal hash function.
- $\mathsf{PRG} : \{0,1\}^\lambda \to \mathbb{F}_p^{k+t}$, $\mathsf{PRG}_1^* : \{0,1\}^{2\lambda} \to \mathbb{F}_p^{t\times(k+t)}$, $\mathsf{PRG}_2^* : \{0,1\}^{2\lambda} \to \mathbb{F}_{p^r}^{\tau}$.
- $\mathsf{H}_1, \mathsf{H}_2 : \{0,1\}^* \to \{0,1\}^{2\lambda}$, $\mathsf{H}_3 : \{0,1\}^* \to \{0,1\}^{3\lambda}$.

Initialization.

- Parse sk as $(\mathsf{sd}, \mathbf{x})$;
- Let $(\mathbf{A}_i)_{i\le n} \leftarrow \mathsf{PRG}_1(\mathsf{sd})$, $(\mathbf{b}_i)_{i\le n} \leftarrow \mathsf{PRG}_2(\mathsf{sd})$;
- Set $y_i \leftarrow \mathbf{x}^T\mathbf{A}_i\mathbf{x} + \mathbf{b}_i^T\mathbf{x}$.
- Sample $r \leftarrow_r \{0,1\}^\lambda$, $(\mathsf{K}, \mathsf{seed}) \leftarrow_r \mathsf{H}_3(\mathsf{sk}, r)$. // H_3 is pseudorandom generator
- Set $\mathsf{K} \leftarrow (K_0, K_1)$. // K is the key of multi-instance PPRF, seed is root of PPRF

Phase 1. Establishes $\mathsf{VC}(N, \tau)$:

- Establish $\tau - \mathsf{PPRF}_{\mathsf{K}}$.
- $\mathsf{VC.Commit}(\mathsf{seed}, 1^\lambda) \to (h_{\mathsf{com}}, \mathsf{decom}, (\mathsf{sd}_j^i)_{i\in[0,\tau), j\in[0,N)})$.

Phase 2.

- Define $\mathbf{U}, \mathbf{V} \in \mathbb{F}_p^{(k+t)\times\tau}$ 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. $h_1 \leftarrow \mathsf{H}_1(\mathsf{m}, \mathsf{K}, \mathbf{C}, h_{\mathsf{com}})$;
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 $\gamma \in \mathbb{F}_p^n$ and $(\mathsf{QS}_0, \mathsf{QS}_1) \in \mathbb{F}_{p^r}^{\tau}$;
2. $h_2 \leftarrow \mathsf{H}_2(\mathsf{m}, \mathsf{K}, h_1, \tilde{\mathbf{u}}, \tilde{\mathbf{V}}, \gamma, \mathsf{QS}_0, \mathsf{QS}_1)$;
3. $\boldsymbol{\Delta} \leftarrow \mathsf{PRG}_2^*(h_2)$, define $\boldsymbol{\Delta} = (\Delta_0, \cdots, \Delta_{\tau-1})$.
4. Compute $\mathsf{pdecom}_I \leftarrow \mathsf{VC.Open}\left(\mathsf{decom}, I = \cup_{i=0}^{\tau-1}[0,N)\setminus\{\Delta_i\}\right)$.

Phase 5.
Output $\sigma = \left(\mathsf{K}, h_1, h_2, \mathbf{C}, (\gamma, \mathsf{QS}_0, \mathsf{QS}_1), (\tilde{\mathbf{u}}, \tilde{\mathbf{V}}), \mathsf{pdecom}_I\right)$. // for $\tilde{\mathbf{V}}$, only need to send its collision-resistant hash value.

Fig. 7. Signing algorithm of the signature scheme

depicted in Figure 7, running in time t, making q_s signing queries, and making q_0, q_1, q_2, q_3 queries, respectively, to the random oracles $\mathsf{H}, \mathsf{H}_1, \mathsf{H}_2$ and PRG_2^*, succeeds in outputting a valid forgery with probability

$$\Pr[\text{Forge}] \le \frac{q_s}{2^{3\lambda}} + \frac{q_s\,(q_s+q_1+q_2+q_3)}{2^{2\lambda}} + \frac{(q+q_1+q_2)^2}{2^{2\lambda}} + \epsilon_{\mathsf{MQ}} + \epsilon_{\Pi_{\mathsf{MQ-PVZK}}} + \mathsf{Adv}^{\mathsf{MulHiding}}_{\mathcal{A},\mathsf{VC}}[q_s]$$

Proof. We start by proving the following formula

$$\mathsf{Adv}^{\mathsf{EUF\text{-}CMA}}_{\mathcal{A}} \le \mathsf{Adv}^{\mathsf{EUF\text{-}KO}}_{\mathcal{A}} + \frac{q_s}{2^{3\lambda}} + \frac{q_s\,(q_s+q_1+q_2+q_3)}{2^{2\lambda}} + \mathsf{Adv}^{\mathsf{MulHiding}}_{\mathcal{A},\mathsf{VC}}[q_s]$$

Let us consider an adversary $\mathcal{A}$ against the EUF-CMA property of the signature scheme, where $\mathcal{A}$ plays the game in EUF-CMA security and can access signing oracle q_s times. To prove security we

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

1. Split the signature as follows:
$$\left(\mathsf{K}, h_1, h_2, \mathbf{C}, (\gamma, \mathsf{QS}_0, \mathsf{QS}_1), (\tilde{\mathbf{u}}, \tilde{\mathbf{V}}), \mathsf{pdecom}_I\right);$$
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. Recompute h_{com} from $(\mathsf{K}, \mathsf{pdecom}_I)$ by expanding $\mathsf{PPRF}_{\mathsf{K}}$ tree.
5. Verifies vector commitment VC if
   - $\mathsf{VC.Verify}\,(h_{\mathsf{com}}, I, \mathsf{pdecom}_I) = \perp$, then outputs REJECT.
   - $\mathsf{VC.Verify}\,(h_{\mathsf{com}}, I, \mathsf{pdecom}_I) = (\mathsf{sd}_j^i)_{i\in[0,\tau),j\in[I^i]}$.
6. Compute:
$$\mathbf{W} = \left[\sum_{i=0}^{N-1}(i-\Delta_0)\cdot\mathsf{PRG}(\mathsf{sd}_i^0)\cdots\sum_{i=0}^{N-1}(i-\Delta_{\tau-1})\cdot\mathsf{PRG}(\mathsf{sd}_i^{\tau-1})\right]$$
7. Run step $5-7$ of $\Pi_{\mathsf{MQ-DVZK}}$ in Figure 3. If the output is ACCEPT then check if
$$\tilde{\mathbf{V}} + \tilde{\mathbf{u}}\cdot[1\cdots1]\cdot\mathsf{diag}(\boldsymbol{\Delta}) = \mathsf{H}^{\mathsf{UHF}}\cdot(\mathbf{W} + [0\|\mathbf{C}]\cdot\mathsf{diag}(\boldsymbol{\Delta}))$$
8. Check if $h_1 \leftarrow \mathsf{H}_1(\mathsf{m}, \mathsf{K}, \mathbf{C}, h_{\mathsf{com}})$;
9. Check if $h_2 \leftarrow \mathsf{H}_2(\mathsf{m}, \mathsf{K}, h_1, \tilde{\mathbf{u}}, \tilde{\mathbf{V}}, \gamma, \mathsf{QS}_0, \mathsf{QS}_1)$;
10. Output ACCEPT if three conditions are satisfied.

Fig. 8. Verification algorithm of the signature scheme

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 0** (Gm^0). 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 1** (Gm^1). We abort if $\mathsf{K} \in \{0,1\}^{2\lambda}$ and $\mathsf{seed} \in \{0,1\}^\lambda$ collide with the value sampled in any the previous queries from $\mathcal{O}_{\mathsf{Sign}}$. Since $(\mathsf{K}, \mathsf{seed})$ are sampled randomly from $\{0,1\}^{2\lambda}\times\{0,1\}^\lambda$ in each query to $\mathcal{O}_{\mathsf{Sign}}$. Then
$$|\Pr[\mathsf{Gm}^0(\mathsf{Forge})] - \Pr[\mathsf{Gm}^1(\mathsf{Forge})]| \leq \frac{q_s}{2^{3\lambda}}$$

  Game 2 (Gm^2). For this step, we abort if the values queried to $\mathsf{H}_1, \mathsf{H}_2, \mathsf{PRG}_2^*$ collide with the values queried in any of the previous queries to hash functions $\mathsf{H}_1, \mathsf{H}_2, \mathsf{PRG}_2^*$. We can bound the distinguishable probability of this game and previous game 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 $\boldsymbol{\Delta}^*$. Since all computations 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 $\boldsymbol{\Delta}^*$ then the difference in forgery probability 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 pdecom_i (the related co-path $\mathsf{CoPath}_{\Delta_i^*}$). From $(\mathsf{pdecom}_i)_{i\in[0,\tau)}$ and $\mathsf{sd}_{\Delta_i^*}$, we can compute all $(\mathsf{sd}_j^i)_{i\in[0,\tau),j\in[0,N)}$. Note

that now all $(\mathsf{sd}_j^i)_{i\in[0,\tau),j\in[0,N)}$ are random. Therefore, $(\mathbf{U},\mathbf{V},\mathbf{u},\mathbf{C})$ executed in an actual way is random. Distinguishing between this game and the previous one is perfectly equivalent to breaking the q_s-multi-hiding security of the VC:

$$|\Pr[\mathsf{Gm}^3(\mathsf{Forge})] - \Pr[\mathsf{Gm}^4(\mathsf{Forge})]| \leq \mathsf{Adv}^{\mathsf{MulHiding}}_{\mathcal{A},\mathsf{VC}}[q_s]$$

Game 5 (Gm^5). In this game, we will change Phase 4 by making the signer use the simulator $\mathsf{Sim}^{\mathsf{V}}_{\Pi_{\mathsf{MQ-DVZK}}}$ against the semi-honest verifier described in Theorem 19 to produce the simulated values $(\gamma,\mathsf{QS}_0,\mathsf{QS}_1)$. We have

$$\Pr[\mathsf{Gm}^4(\mathsf{Forge})] = \Pr[\mathsf{Gm}^5(\mathsf{Forge})]$$

We see that at the end of this game, the signature σ produced from $\mathcal{O}_{\mathsf{Sign}}$ is no longer dependent on the sk so we can reduce to the $\mathsf{EUF\text{-}KO}$ games.

Game 6 ($\mathsf{Adv}^{\mathsf{EUF\text{-}KO}}_{\mathcal{A}}$). We say that an execution e^* of a query

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

defines a correct witness if the following criteria are satisfied:

- h_1 was output by a previous query

$$h_1 \leftarrow \mathsf{H}_1(\mathsf{m},\mathsf{K},\mathbf{C},h_{\mathsf{com}})$$

- h_{com} in this query was output by a previous query

$$(h_{\mathsf{com}},\mathsf{decom},(\mathsf{sd}_j^i)_{i\in[0,\tau),j\in[0,N)}) \leftarrow \mathsf{VC}^{\mathsf{H},\mathsf{H}_1}.\mathsf{Commit}(\mathsf{seed},1^\lambda)$$

- The vectors $\tilde{\mathbf{u}},\tilde{\mathbf{V}}$ is defined such that

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

- The vector $\mathbf{x}$ defined by $\gamma,\mathsf{QS}_0,\mathsf{QS}_1$ satisfies $\mathsf{MQ}_{p,m,n}$.

We observe that:

- The probability of finding preimage or collisions of $\mathsf{H},\mathsf{H}_1,\mathsf{H}_2$ is bounded by $(q+q_1+q_2)^2/2^{2\lambda}$.
- If there is an execution e^* that defines a correct witness. Calling this event Solve then $\Pr[\mathsf{Solve}] \leq \epsilon_{\mathsf{MQ}} + \epsilon_{\Pi_{\mathsf{MQ-PVZK}}}$, since if it occurs then $\gamma,\mathsf{QS}_0,\mathsf{QS}_1$ define a solution for the $\mathsf{MQ}_{p,m,n}$.

In the end, we obtain

$$\mathsf{Adv}^{\mathsf{EUF\text{-}KO}}_{\mathcal{A}} \leq \frac{(q+q_1+q_2)^2}{2^{2\lambda}} + \epsilon_{\mathsf{MQ}} + \epsilon_{\Pi_{\mathsf{MQ-PVZK}}}$$

Putting all game hops together, we conclude the proof

$$\mathsf{Adv}^{\mathsf{EUF\text{-}CMA}}_{\mathcal{A}} \leq \frac{q_s}{2^{3\lambda}} + \frac{q_s\,(q_s+q_1+q_2+q_3)}{2^{2\lambda}} + \frac{(q+q_1+q_2)^2}{2^{2\lambda}} + \epsilon_{\mathsf{MQ}} + \epsilon_{\Pi_{\mathsf{MQ-PVZK}}} + \mathsf{Adv}^{\mathsf{MulHiding}}_{\mathcal{A},\mathsf{VC}}[q_s]$$

□

5.3 Parameters and Signature Size

Parameters and Optimizations. 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 bits. For feasible implementation, $\mathbb{F}_p$ is chosen as an extension field of $\mathbb{F}_2$ and subfield of $\mathbb{F}_{2^\lambda}$. We run the estimator from [BMSV22,EVZB23] to estimate the running time and memory usages of the existing attack algorithms for the MQ problem, where we choose the number of unknowns and the number of equations being equal ($m=n$) and other parameters for each algorithm is chosen default by the estimator. For the memory representation, following from [Wan22], observe that when choosing p,n for a given security level if n decreases as q increases — but $n\log p$ will slightly increase. Since the asymptotic signature size is proportional to $n\log p$ therefore the signature size also increases if p increases. However, larger p dramatically reduces n as the multivariate quadratic equations so the memory storage also is reduced. Concretely, [Wan22] found the speed of their signature over $\mathbb{F}_4$ is faster than that over $\mathbb{F}_2$, even though multiplication requires more bitwise operations. We choose the parameter choice for both $p=2$ and $p=4$ in Table 1 according to the best attacks for each choice of parameters.

- 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 extension fields $\mathbb{F}_{p^r}$ instead of a field of size $O(2^\lambda)$. Since QuickSilver is applied and the challenge space is $\mathbb{F}_{p^r}^\tau$, from the Theorem 16, the underlying DVZK (Figure 3) achieves soundness error of $O(2^{-\lambda})$ if
$$\left(\frac{3}{p^r}\right)^\tau = 2^{-\lambda}$$
Since the speed of the singing algorithm is dominant by establishing τ vector commitments of length $N = p^r$ then the choice for τ offers tradeoffs between signature size and running time. 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 p^r and hence more computational work for the signer and verifier and as reverse for a large τ it leads to less computational work for the signer and verifier.
- SoftspokenOT parameters, we use a sVOLE constructed from SoftspokenOT of length $(n+r\cdot\tau+t)$ where first n-sVOLE correlations are used to hide the witness of length n, next $r\cdot\tau$-sVOLE correlations are for nullity check for polynomial set using Quicksilver and last t-correlations is added more to make sure verifier learns nothing about $\mathbf{u}$ in VOLE consistent checks. Specifically, this check reveals a $(n + r\cdot\tau + t)$ linear function of $\mathbf{u}$ to the verifier, which needs to hide the underlying witness. From Proposition 12 and the universal hash function $\mathsf{H}^{\mathsf{UHF}} \in \mathbb{F}_p^{t\times(n+r\cdot\tau+t)}$ is defined as a form of $[\mathbf{H}\|\mathbf{I}_t]$ where $\mathbf{H} \leftarrow_r \mathbb{F}_p^{t\times(n+r\cdot\tau)}$, t needs to be chosen such that $p^{-t} = O(2^{-\lambda})$.[3]

Public key size. Our public key size includes a seed $\mathsf{sd} \in \{0,1\}^\lambda$ that is used to get MQ instance $(\mathbf{A}_i, \mathbf{b}_i)_{i\le n}$ by using PRG and $(y_i)_{i\le n} \in \mathbb{F}_p^n$ as $y_i := \mathbf{x}^T\mathbf{A}_i\mathbf{x} + \mathbf{b}_i^T\mathbf{x}$ where $\mathbf{x}$ is hidden secret in $\mathsf{MQ}_{p,n,n}$. This leads to a public key size of $(\lambda + n\log p)$ bits.

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 $\mathbf{C}_0, \ldots, \mathbf{C}_{\tau-1}$ where $k = n + r\cdot\tau$.
- The hashed VOLE secrets $\tilde{\mathbf{u}}, \tilde{\mathbf{V}}$. 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 $\gamma \in \mathbb{F}_p^n$ (to hide witness) and $(\mathsf{QS}_0, \mathsf{QS}_1) \in \mathbb{F}_{p^r}^\tau$.
- The partial $\mathsf{pdecom}_I = \{\mathsf{CoPath}_\mathsf{K}(\Delta_i, \mathsf{seed}^i), \mathsf{com}^i_{\Delta_i}\}_{i<\tau}$ for each of the τ-PPRF instances, opening all positions except $\boldsymbol{\Delta} = (\Delta_0, \ldots, \Delta_{\tau-1}) \in \mathbb{F}_{p^r}^\tau$.

The signature size is

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

We show in Table 3 the estimated size of our signature for three levels of security (using Python script), we choose the set parameters (r, N, τ) of VOLEitH protocol such that the efficiency still is competitive (keeping $N = 256$ and $N = 16$ for short and fast versions respectively) and the MQ parameter based on Table 1. Compared to other VOLEitH-based signatures, our signature size is slightly smaller, for security level I, our smallest signature size is 3792B, while [CLY+24] (based on regular syndrome decoding assumption) and FAEST (based on AES OWF) have a signature size of 3916B and 5006B respectively [CLY+24].

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

Table 3. The signature size in Bytes of our signature for three NIST security levels where τ is the number of repetitions, r is the degree of extension field and $N = p^r$ is the width of PPRF tree, using two $\mathsf{MQ}_{p,n,n}$ instances for each security level with $p = 2$ and $p = 4$.

Category	Variant	MQ parameters		VOLEitH parameters			Size	
		p	$m = n$	r	N	τ	Public key	Signature
I	Our-L1$_{\mathsf{short}}$	2	150	8	256	19	35 B	4209 B
		4	88	4	256	17	38 B	3792 B
	Our-L1$_{\mathsf{fast}}$	2	150	4	16	43	35 B	6697 B
		4	88	2	16	33	38 B	5103 B
III	Our-L3$_{\mathsf{short}}$	2	224	8	256	28	52 B	9180 B
		4	128	4	256	25	56 B	8230 B
	Our-L3$_{\mathsf{fast}}$	2	224	4	16	65	52 B	15077 B
		4	128	2	16	49	56 B	11205 B
V	Our-L5$_{\mathsf{short}}$	2	320	8	256	37	72 B	16166 B
		4	160	4	256	33	72 B	14298 B
	Our-L5$_{\mathsf{fast}}$	2	320	4	16	86	72 B	26701 B
		4	160	2	16	65	72 B	19545 B

5.4 Efficiency and Instantiation

In this section, we show how we instantiate our signature to have a competitive performance. We note that since our signature works over the extension field of $\mathbb{F}_2$ instead of a prime field ($\mathbb{F}_{31}, \mathbb{F}_{251}$ as in [BFR23]) then operations can be transferred to XOR and AND operations.
The computational cost for the signer is dominant by:

- Establishing the vector commitment $\mathsf{VC}(N, \tau)$ and then computing the vector $\mathbf{U}, \mathbf{V}$ from the output of $\mathsf{VC}(N, \tau)$.
- Executing the role of P in the underlying DVZK protocol (Figure 3) by using inputs $(\mathbf{U}, \mathbf{V})$.

For the verifier, the computational work is dominant by:

- Verifying the vector commitment $\mathsf{VC}(N, \tau)$ by splitting the signature and recomputing the challenge set $\boldsymbol{\Delta} = (\Delta_0, \ldots, \Delta_{\tau-1})$ for VC and then computing $\mathbf{W}$ as a component of sVOLE correlation.
- Executing the role of V in the underlying DVZK protocol (Figure 3) by using the inputs $(\boldsymbol{\Delta}, \mathbf{W})$.

Vector commitment Efficiency. There are two parameters to constructing a vector commitment: the number of leaves in one PPRF-tree and the number of repetitions τ of the PPRF-tree to have described soundness for a given security level. For efficient computation, we always fix the number of leaves of PPRF-tree $N = 256$ to get the shortest signature and $N = 16$ to get a signature with fast speed, and τ varies depending on the security level, i.e., for $\lambda = 128$, τ ranging from 17-19 for the short version and from 33-43 for the fast version.

We use the construction of multi-instance PPRF of [BCC+24] to instantiate our vector commitment. Taking advantage of hardware support for AES fix-key, [BCC+24] has a better $14 - 18\%$ performance compared to other GGM-tree PPRF in other existing MPCitH-based signatures such as [AGH+23], as the methodology under the hood of using PPRF in VOLEitH and MPCitH are similar, multi-instance PPRF has a similar performance when dropping into VOLEitH. In [BCC+24], the expansion of the PPRF-tree is constructed by $\mathsf{PRG} : x \to (\mathsf{AES}_{\mathsf{K}_0}(x), \mathsf{AES}_{\mathsf{K}_1}(x))$, x is the value of parent node and the left child and right child are defined by $(\mathsf{AES}_{\mathsf{K}_0}(x)$ and $\mathsf{AES}_{\mathsf{K}_1}(x)$ respectively, the key $\mathsf{K} = (\mathsf{K}_0, \mathsf{K}_1)$ is fixed for all τ-PPRF instances.

In total, the computation cost to establish a vector commitment ($\mathsf{VC.Commit}$) includes:

- Expanding PPRF-tree: $(1/2 \cdot N \cdot \tau)$ calls to $(\mathsf{AES}_{\mathsf{K}_0}, \mathsf{AES}_{\mathsf{K}_1})$ which AES-NI can instantiate, for $N \cdot \tau$ ranging from 2^{11} to 2^{13} it is speedy since encrypting by AES-NI only takes 1.3 cycles per Byte [MSY21].
- Obtaining commitment: $N \cdot \tau$ calls to a hash function H with output 3λ and τ calls to a hash function H_1 with output 2λ, these hash functions can instantiated by SHAKE as in [BBD+23,CLY+24]. Using SHAKE to get commitments is unavoidable since the adversary in the forgery attack can query polynomial times to vector commitment, leading to the output domain of commitment being at least 2λ for a security level λ.

DVZK Efficiency. The computation work of the prover and the verifier intuitively are the same as in Quicksilver-based polynomial satisfiability [YSWW21] for a batch of degree-2 polynomial over the field $\mathbb{F}_p$ and the extension field $\mathbb{F}_{p^r}$ by repeating τ times. The dominant computational cost of prover and verifier in protocol $\Pi_{\mathsf{MQ-DVZK}}$ are computing $(\mathbf{A}_{i,0}, \mathbf{A}_{i,1})_{i\leq m} \in (\mathbb{F}_{p^r})^\tau$ and $(\mathbf{B}_i)_{i\leq m} \in (\mathbb{F}_{p^r})^\tau$.

When using the Lagrange interpolation approach to compute the coefficients $(\mathbf{A}_{i,0}, \mathbf{A}_{i,1})$, we have that the computational complexity of the prover and verifier is $O(4mz + 2n)$ and $O(2mz)$ respectively [YSWW21], where $z = mn(n+3)/2$ is the maximum number of terms in all $\mathsf{MQ}_{p,m,n}$ problem. In total, the prover and verifier needs to do $O(\tau n^3)$ multiplications over $\mathbb{F}_{p^r}$ $(m = n)$ that is competitive since $N = p^r \in \{16, 256\}$, τ is small and m, n are MQ parameters (Table 1).

Acknowledgement

We would like to thank Geoffroy Couteau (IRIF), Thibauld Feneuil (CryptoExperts), Kelong Cong (Zama), and Lawrence Roy (Aarhus University) as well as the Faest team (`faest.info`) for their advice and insightful comments.

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.

BBM+24. C. Baum, W. Beullens, S. Mukherjee, E. Orsini, S. Ramacher, C. Rechberger, L. Roy, and P. Scholl. One tree to rule them all: Optimizing ggm trees and owfs for post-quantum signatures. Cryptology ePrint Archive, Paper 2024/490, 2024. `https://eprint.iacr.org/2024/490`.

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`.

BFR23. R. Benadjila, T. Feneuil, and M. Rivain. Mq on my mind: Post-quantum signatures from the non-structured multivariate quadratic problem. Cryptology ePrint Archive, Paper 2023/1719, 2023. `https://eprint.iacr.org/2023/1719`.

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.

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.

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.

CKPS00. N. Courtois, A. Klimov, J. Patarin, and A. Shamir. Efficient algorithms for solving overdefined systems of multivariate polynomial equations. In *EUROCRYPT 2000*, *LNCS* 1807, pages 392–407. Springer, Heidelberg, May 2000.

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.

EVZB23. A. Esser, J. Verbel, F. Zweydinger, and E. Bellini. `CryptographicEstimators`: a software library for cryptographic hardness estimation. Cryptology ePrint Archive, Paper 2023/589, 2023. `https://eprint.iacr.org/2023/589`.

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.

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.

JV17. A. Joux and V. Vitse. A crossbred algorithm for solving Boolean polynomial systems. Cryptology ePrint Archive, Report 2017/372, 2017. `https://eprint.iacr.org/2017/372`.

KKW18. J. Katz, V. Kolesnikov, and X. Wang. Improved non-interactive zero knowledge with applications to post-quantum signatures. In *ACM CCS 2018*, pages 525–537. ACM Press, October 2018.

MSY21. J.-P. Münch, T. Schneider, and H. Yalame. Vasa: Vector aes instructions for security applications. Cryptology ePrint Archive, Paper 2021/1493, 2021. `https://eprint.iacr.org/2021/1493`.

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.

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.