Improved All-but-One Vector Commitment with Applications to Post-Quantum Signatures

Dung Bui[1], Kelong Cong[2] and Cyprien Delpech de Saint Guilhem[3]

[1] IRIF & Université Paris Cité, Paris, France
[2] Zama SAS, Paris, France
[3] KU Leuven, COSIC, Leuven, Belgium

Abstract. Post-quantum digital signature schemes have recently received increased attention due to the NIST standardization project for additional signatures. MPC-in-the-Head and VOLE-in-the-Head are general techniques for constructing such signatures from zero-knowledge proof systems. A common theme between the two is an *all-but-one* vector commitment scheme which internally uses GGM trees. This primitive is responsible for a significant part of the computational time during signing and verification.

A more efficient technique for constructing GGM trees is the half-tree technique, introduced by Guo et al. (Eurocrypt 2023). Our work builds an all-but-one vector commitment scheme from the half-tree technique, and further generalizes it to an all-but-τ vector commitment scheme. Crucially, our work avoids the use of the random oracle assumption in an important step, which means our binding proof is non-trivial and instead relies on the random permutation oracle. Since this oracle can be instantiated using fixed-key AES which has hardware support, we achieve faster signing and verification times.

We integrate our vector commitment scheme into FAEST (`faest.info`), a round one candidate in the NIST standardization process, and demonstrates its performance with a prototype implementation. For $\lambda = 128$, our experimental results show a nearly 3.5-fold improvement in signing and verification times.

Keywords: Vector commitment · Correlation robustness · Post-quantum signature · Zero-knowledge proof

1 Introduction

Many signature schemes are built from zero-knowledge (ZK) proofs, which allows a prover, who holds a secret witness, to convince a verifier that this witness satisfies some public statement without revealing any other information about the witness. From digital signature schemes to privacy-preserving electronic voting, ZK proofs have many applications and are widely used in practice.

Due to the threat of quantum computing, there is a strong demand for post-quantum ZK proofs and signature schemes. A powerful, generic technique called MPC-in-the-Head (MPCitH), introduced by Ishai et al. [IKOS07], has been applied to build many efficient proof systems [KKW18, BN20, DOT21, FJR22, AGH+23, CCJ23]. Many of these can be used to construct signature schemes using the Fiat–Shamir transform [FS87] since MPCitH protocols are always public-coin. In this line of work, the proof size and the computational complexity are inherently linear in the circuit size, where the circuit represents the statement that the prover wishes to prove. As such, the research community

E-mail: `bui@irif.fr` (Dung Bui), `kelong.cong@zama.ai` (Kelong Cong), `cyprien.delpechdesaintguilhem@kuleuven.be` (Cyprien Delpech de Saint Guilhem)

Date of this document: 2024-01-22.

has given a lot of attention to reducing the concrete costs incurred by MPCitH proof system.

Another line of work which also has linear computation and communication complexity is called VOLE-based ZK proofs [WYKW21, BMRS21, YSWW21], which are typically more efficient than most MPCitH protocols. However, these protocols assume the existence of vector oblivious linear function evaluation (VOLE) correlations between the prover and the verifier, which can be generated efficiently using two-party protocols [BCG$^+$19]. Due of this requirement, VOLE-based protocols could not operate in the public-coin model, until the introduction of the VOLE-in-the-Head (VOLEitH) approach by Baum et al. [BBD$^+$23]. In addition to a generic ZK proof system, Baum et al. used this approach to build the FAEST post-quantum signature scheme, only based on AES and hash functions, which outperforms prior MPCitH constructions when AES is used as the circuit/statement.

A common requirement between the two techniques, MPCitH and VOLEitH, is the need to commit to a set of randomness (e.g., transcript of the parties) and then to reveal all but one element in this set. This can be accomplished using GGM trees [GGM84] where a single seed is expanded to many leaves using a length-doubling PRG. Revealing all but one leaf then requires a communication cost that is only logarithmic in the total number of leaves since the prover can simply send the co-path to the verifier, and the verifier can expand the sub-trees defined by this co-path to obtain all the leaves except one.

In 2023, Guo et al. introduced the notion of *correlated* GGM trees [GYW$^+$23] where the authors replaced the length-doubling PRG using the more efficient Davies–Meyer construction [Win84]. This change reduces the number of PRG calls by a half for the intermediate nodes of the GGM tree, hence the name "half-tree". The authors constructed a puncturable pseudorandom function (PPRF) from such correlated GGM trees and showed how it can be applied to two-party protocols such as correlated oblivious transfer (COT) and distributed point function (DPF) evaluation.

However, the construction by Guo et al. [GYW$^+$23] is not directly compatible with MPCitH and VOLEitH protocols to improve their efficiency since these protocols instead require an *all-but-one* vector commitment scheme, which is a stronger primitive than a PPRF. This work constructs an all-but-one vector commitment scheme using correlated GGM trees and demonstrates how it can be applied to post-quantum ZK proof systems and signature schemes.

1.1 Contributions

Our contributions are summarized below.

1. We propose an all-but-one vector commitment (VC) scheme $\mathsf{VC}_{\mathsf{HT}}$ in Section 4 and prove its binding and hiding properties. Our VC construction is based on the half-tree technique, which uses a circular correlation robust (CCR) hash function, and leads to better concrete efficiency than the PPRFs used in existing MPCitH schemes.

   In particular, we reduce the number of AES calls by a half in the intermediate nodes and replace the leaf expansion procedure with fixed-key AES, which is about 50× faster than SHAKE.

2. Our security proof of the binding property is not trivial and relies on modelling our chosen instantiation of the (CCR) hash function using a random permutation oracle. This is an alternative model to the random oracle and it can be instantiated using more efficient primitives such as fixed-key AES.

3. We generalize the notion of our all-but-one VC scheme $\mathsf{VC}_{\mathsf{HT}}$ to the all-but-τ case, making use of multiple correlated trees, denoted by $\mathsf{VC}_{\mathsf{MT}}(N, \tau)$ (Section 5). This captures the setting where τ repetitions of $\mathsf{VC}_{\mathsf{HT}}$ of size N are used. Using $\mathsf{VC}_{\mathsf{MT}}(N, \tau)$,

we estimate 2-fold improvement in both proving and verification times in a state-of-the-art MPCitH system [AGH+23] by running their implementation. In the all-but-τ setting, we additionally show the security can be improved using different instantiations of the CCR hash function (Section 6).

4. Finally, we give a prototype implementation (Section 7) of the FAEST signature scheme that uses our all-but-τ VC scheme and perform experimental evaluation. The improved reference implementation is between $1.11\times$ and $1.57\times$ faster across all parameter sets. The optimized implementation is nearly $3.5\times$ faster for $\lambda = 128$.

1.2 Related Work

1.2.1 MPC-in-the-Head

The state of the art MPCitH based signatures use a variety of problems as the statement in the ZK proof such as the syndrome decoding problem [FJR22, AGH+23, CCJ23], the min rank problem [ABC+23] and so on. In this work, we are not concerned with the concrete hard problem, but the core idea from the seminal work of [IKOS07] which is a compiler that can convert an MPC protocol into an honest-verifier zero-knowledge (HVZK) proof. Suppose we have an MPC protocol with the following properties:

- N parties $(P_1, \cdots, P_N)$, each party hold an additive share $[\![\vec{x}]\!]_i$ of input $\vec{x}$ and their goal is to securely evaluate a function $f : \{0,1\}^* \to \{0,1\}$ on $\vec{x}$.
- Secure against passive corruption of $N-1$ parties i.e the view of $(N-1)$ parties does not reveal any information about the secret $\vec{x}$.

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

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

For efficiency, the GGM tree is used to stretch the witness and open the view of $N-1$ parties with a communication cost of only $O(\lambda \log_2 N)$. The GGM tree needs to have a specific structure where the last leaves are obtained from a random oracle and these leaves are then used as the shares of witness and input to a *commitment scheme*, to commit the view of the parities. To ensure the soundness and the ZK properties, this commitment scheme needs to have (extractable) binding and hiding respectively. From this point of view, an *all-but-one* vector commitment (detailed in Section 3.3) can be directly plugged into MPCitH protocols.

1.2.2 VOLE-in-the-Head and All-but-One VC

The VOLEitH technique, introduced by Baum et al. [BBD+23], turns a designated verifier proof system based on VOLE such as Wolverine [WYKW21] and QuickSilver [YSWW21] into a public-coin proof system. In more detail, if the prover holds $(\vec{u}, \vec{v})$ and the verifier holds $(\vec{w}, \Delta)$ such that the correlation $\vec{w} = \vec{u}\Delta - \vec{v}$ holds, it is possible to create a very efficient ZK proof for circuits. Another way to view the correlation above is in the form of information-theoretic message authentication codes (IT-MAC) on the vector $\vec{u}$, where $\vec{v}$ is called the MAC and $\vec{w}$ is called the MAC key. However, a setup phases is needed to produce the correlation above which makes such proof systems only work in the designated verifier setting.

The central observation in the VOLEitH technique is that the verifier does not need $(\vec{w}, \Delta)$ at all in the VOLE-based proof system until the final step where it performs the MAC check. To remove the setup step, the prover commits to $(\vec{u}, \vec{v})$ and then runs the VOLE-based proof system with the verifier. At the end, the verifier sends the challenge Δ which then allows it to compute $\vec{w}$.

The procedure above makes use of an *all-but-one* vector commitment (VC) scheme, defined by Baum et al. [BBD+23], where the prover commits to the leaves of a GGM tree and hides one of the leaves. The opening essentially reveals the hidden leaf and binds it to the initial commitment. The definition of the all-but-one VC is given in Section 3.3 and how VC is turned into VOLE correlations is detailed in Section 7.3.

1.2.3 Correlated GGM Tree

The correlated GGM tree technique, by Guo et al. [GYW+23], is a key building block in our construction, which we explain in more detail in Section 2.

1.3 Concurrent Work

Recently, Cui et al. [CLY+24] also proposed to use the half tree optimization on the FAEST signature schemes, the result is a new scheme called ReSolveD. However, since ReSolveD only applied the half-tree construction to compute intermediate nodes of the tree instead of using it for the entire tree i.e., to construct the leaves that are later used to define seeds and commitments. Therefore, ReSolveD still needs to call SHAKE to expand a leaf node into a seed and a commitment of length 3λ, which is the primary bottleneck, whereas our construction uses AES for this step. As a result, our scheme outperforms both constructions in terms of signing and verification time.

2 Technical Overview

We begin with an overview of the components and techniques used in this work.

2.1 Correlated GGM Tree

Consider the GGM-based punturable pseudorandom function where one party has a master sd and then expandss it into a binary tree of depth $\log_2 N$ with N leaves $\{\mathsf{sd}_0, \ldots, \mathsf{sd}_{N-1}\}$ by using a length-doubling PRG. To reveal all the seeds except sd_i, for a given i, the party only needs to send $\log_2 N$ nodes on the co-path from the root sd to the i-th leave. An example is given in Figure 1.

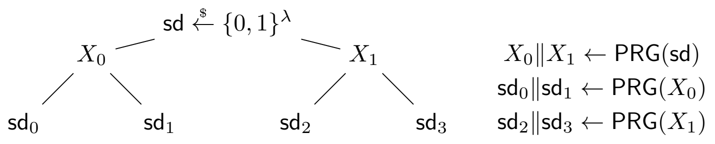
$\mathsf{sd} \xleftarrow{\$} \{0,1\}^\lambda$
X_0
X_1
sd_0
sd_1
sd_2
sd_3
$X_0 \| X_1 \leftarrow \mathsf{PRG}(\mathsf{sd})$
$\mathsf{sd}_0 \| \mathsf{sd}_1 \leftarrow \mathsf{PRG}(X_0)$
$\mathsf{sd}_2 \| \mathsf{sd}_3 \leftarrow \mathsf{PRG}(X_1)$

Figure 1: Example of a GGM tree.

Recently, a variant of GGM trees called *correlated* GGM trees was introduced to optimize both the communication and computation cost of GGM tree using the half-tree technique [GYW+23]. Following the Davies–Meyer construction [Win84], for a parent x, the left child is constructed using $\mathsf{H}(x)$ and the right child is constructed using $\mathsf{H}(x) \oplus x$. We give an example in Figure 2.

Consequently, the sum of all nodes on every layer of the tree is always equal to an certain offset Δ. The pseudorandomness of the hidden leaf is guaranteed when H has the circular correlation robustness (CCR) property which we detail in Definition 2. Informally, this means the adversary cannot distinguish between the real oracle defined as $\mathcal{O}^{\mathsf{ccr}}_{\mathsf{H},\Delta}(x,b) := \mathsf{H}(x \oplus \Delta) \oplus b \cdot \Delta$ and an ideal oracle that outputs the result of a random function. Existing instantiations of CCR-secure functions H are all based on a random permutation $\pi : \{0,1\}^\lambda \to \{0,1\}^\lambda$ and a linear orthomorphism $\sigma : \{0,1\}^\lambda \to \{0,1\}^\lambda$, where λ is the security parameter [GKWY20].

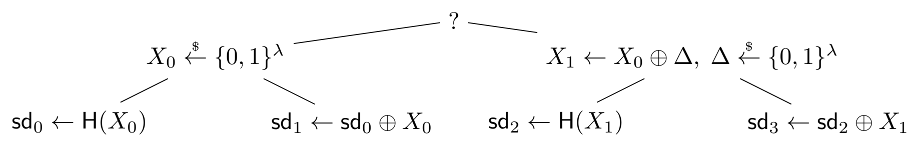
?
$X_0 \xleftarrow{\$} \{0,1\}^\lambda$
$X_1 \leftarrow X_0 \oplus \Delta,\ \Delta \xleftarrow{\$} \{0,1\}^\lambda$
$\mathsf{sd}_0 \leftarrow \mathsf{H}(X_0)$
$\mathsf{sd}_1 \leftarrow \mathsf{sd}_0 \oplus X_0$
$\mathsf{sd}_2 \leftarrow \mathsf{H}(X_1)$
$\mathsf{sd}_3 \leftarrow \mathsf{sd}_2 \oplus X_1$

Figure 2: Example of a correlated GGM tree.

2.2 Vector Commitment from Half-Tree

Taking advantage of *correlated* GGM trees, we propose an *all-but-one* vector commitment construction $\mathsf{VC}_{\mathsf{HT}}$. In our construction, the root sd is first expanded into two children X_0, X_1 using a length doubling PRG. Then, the intermediate nodes are computed the same way as in a correlated GGM tree using a CCR hash function $\mathsf{H}_{\mathsf{ccr}}$ where the correlation $\Delta = X_0 \oplus X_1$ is defined by the first level nodes. Finally, once the tree is expanded, we use the technique from Guo et al. [GYW+23] to derive both a message sd_i and a commitment com_i using the same $\mathsf{H}_{\mathsf{ccr}}$ (and not a random oracle) while taking care to break the correlation with Δ so that revealing of all the commitments does not leak additional information. To maintain security, we modify the way to compute the commitment com_i, instead of calling $\mathsf{H}_{\mathsf{ccr}}$ once, we use the concatenation of two $\mathsf{H}_{\mathsf{ccr}}$ calls to extend the length of com_i to be 2λ, this helps us to avoid a collision attack in the commitment scheme against the binding property.

We then formally prove that our $\mathsf{VC}_{\mathsf{HT}}$ construction satisfies hiding and straight-line extractable binding. Specifically, the original all-but-one vector commitment construction [BBD+23] relies on the random oracle model for its security proof. This is not possible in our construction since one of the random oracles is replaced by the CCR oracle. Our insight is to rely on the specific instantiation of CCR-secure function $\mathsf{H}_{\mathsf{ccr}}$ that is built on top of a public permutation. This observation allows us to "partially" model $\mathsf{H}_{\mathsf{ccr}}$ using a random permutation oracle (RPO), which is similar to a random oracle except input and output have the same length and must be a permutation.

In term of efficiency, we instantiate $\mathsf{H}_{\mathsf{ccr}}(x) : \{0,1\}^\lambda \to \{0,1\}^\lambda$ as $\mathsf{H}_{\mathsf{ccr}}(x) := \pi(\sigma(x)) \oplus \sigma(x)$. For $\lambda = 128$, the permutation π is implemented by using fixed-key AES which can be up to 50× faster than using a cryptographic hash function due to hardware support for AES provided by modern processors [BHKR13]. For the intermediate nodes, our construction reduce the number of calls to AES by a half compared to other MPCitH schemes based on GGM tree [BDK+21, FJR22, AGH+23, BBD+23]. Additionally, many tree expansion algorithms are implemented using AES counter mode [FJR22] which is less efficient than fixed-key AES used in our protocol, due to the key schedule. Regarding the leaf nodes, which may be used as shares of the witness and/or shares of the preprocessing material of the virtual parties in MPCitH protocols, our construction only needs AES calls instead of SHAKE calls [BBD+23]. For $\lambda \in \{192, 256\}$, we instantiate the permutation using Rijndael since AES does not support block sizes other than 128.

2.3 Generalized to Multi-instance Vector Commitment

We expect our $\mathsf{VC}_{\mathsf{HT}}$ scheme can be used to optimize all the existing MPCitH and VOLEitH signatures based on the GGM tree or Punturable PRF [BDK+21, FJR22, AGH+23, CCJ23, BBD+23] in both signing and verification times. For all of these schemes, to maintain the soundness of the underlying special honest verifier zero-knowledge protocol [FJR22, AGH+23, CCJ23] or to have a better performance [BBD+23], parallel repetition is needed with different input and randomness τ times (τ is usually between 10 and 30). Therefore, we generalize our $\mathsf{VC}_{\mathsf{HT}}$ to multi-instance vector commitment denoted by $\mathsf{VC}_{\mathsf{MT}}$, and introduce multi-hiding and multi-binding definitions that are based on single hiding and binding.

Specifically, our $\mathsf{VC}_{\mathsf{MT}}$ run parallel τ different single $\mathsf{VC}_{\mathsf{HT}}$ in the same security setting i.e., depth of the tree, hash function, permutation for CCR in each tree (same iv), except for each tree the offset Δ_i are generated independently, using the same method as in the single tree setting. The randomness of PRG and sd^i allow us to reduce the security of our multi-instance $\mathsf{VC}_{\mathsf{MT}}$ in the multi-hiding game to that of $\mathsf{VC}_{\mathsf{HT}}$ in the single-hiding game. Additionally, our $\mathsf{VC}_{\mathsf{MT}}$ satisfies multi-extractable binding where the proof directly follows from the single tree case as well.

2.4 Better Concrete Security

Since the goal of our $\mathsf{VC}_{\mathsf{HT}}$ is to improve the efficiency of digital signature schemes while preserving their security, we need to make sure the advantage $\mathsf{Adv}^{\text{EUF-CMA}}_{\mathcal{A}}(\lambda)$ of $\mathcal{A}$ in $\mathsf{EUF\text{-}CMA}$ (existential unforgeability under adaptive chosen-message attacks) game is negligible i.e., the verifier cannot produce a valid signature for an unsigned message after making polynomially-many queries to a signing oracle.

When using our VC scheme within an MPCitH signature scheme Sig, where the signer produces the tree expansion, the advantage $\mathsf{Adv}^{\text{AdpHiding}}_{\mathcal{A},\mathsf{VC}_{\mathsf{HT}}}(\lambda)$ in the hiding game counts towards $\mathsf{Adv}^{\text{EUF-CMA}}_{\mathcal{A},\mathsf{Sig}}(\lambda)$. Therefore, we investigate using two different variants of the CCR hash function: tweakable CCR and multi-tweakable CCR. These result in two variants of our $\mathsf{VC}_{\mathsf{HT}}$ scheme with tighter concrete security at the expense of slightly increased computation.

In our concrete instantiation, the AES-based $\mathsf{H}_{\mathsf{ccr}}$ is (t, q, ρ, ϵ)-circular correlation robust where $\epsilon = O((qt + q^2)/2^\lambda)$, $\mathcal{A}$ makes q queries in running time t (t can be considered as the number of AES calls made by $\mathcal{A}$). The first variant uses tweakable CCR to get a smaller ϵ while the second variant of VC from multi-instance tweakable TCC in the ideal cipher model (using different permutations for every single tree) that allows $\epsilon = O(q/2^\lambda)$. The details are shown in Section 6.

3 Preliminaries

We recall various security definitions of hash functions that are securely implemented using a block-cipher (circular correlation robustness, tweakable and multi-tweakable circular correlation robustness). Since all of these hash functions are instantiated and proved secure in random permutation model (RPM) or ideal cipher model (ICM), we also define these oracles. The definition of all-but-one vector commitment [BBD+23] is shown in Section 3.3 followed by the real-or-random hiding and extractable-binding games.

Notation. Let λ denote a computational security parameter. If b is a bit, let $\bar{b}$ denote its negation. For $N \in \mathbb{N}$, let $[N]$ denote the set $\{0, \dots, N-1\}$. We let $\mathsf{negl}(\lambda)$ denote any function that is negligible in the security parameter.

3.1 Oracles

The random permutation model (RPM) assumes that all parties have oracle access to a public permutation $\pi : \{0,1\}^n \to \{0,1\}^n$ (selected at random from the set of permutations) and its inverse π^{-1}. An ideal cipher is an ideal primitive that models a random block cipher $E : \{0,1\}^\lambda \times \{0,1\}^n \to \{0,1\}^n$. Each key $k \in \{0,1\}^\lambda$ defines a random permutation $E_k = E(k, .)$ on $\{0,1\}^n$. The ideal primitive provides oracle access to E and E^{-1}; that is, on query $(0, k, m)$, the primitive answers $c = E_k(m)$, and on query $(1, k, c)$, the primitive answers m such that $c = E_k(m)$. Modelling a fixed-key block cipher as a random permutation is weaker than modelling the block cipher as an ideal cipher; in particular, related-key attacks are not relevant in the fixed-key setting [CPS08, GKWY20].

Definition 1 (Random Permutation Oracle (RPO))**.** Given $k \in \{0,1\}^\lambda$, let $\pi_k : \{0,1\}^\lambda \to \{0,1\}^\lambda$ be a permutation selected uniformly at random. The *random permutation oracle* $\mathcal{O}_\pi$ is defined as follows:

- On input $x \in \{0,1\}^\lambda$, outputs $\pi_k(x)$.
- On input (inv, x), outputs $\pi_k^{-1}(x)$.

In this work, we denote the query and response of $\mathcal{O}_\pi$ as $(x, \mathcal{O}_\pi(x))$, since the queries for the inverse of permutation can be considered as $(\mathcal{O}_\pi^{-1}(x), x)$.

3.2 Hash Function

Definition 2 (Circular Correlation Robustness (CCR) [GKWY20])**.** Let $\mathsf{H} : \{0,1\}^\lambda \to \{0,1\}^\lambda$, χ be a distribution on $\{0,1\}^\lambda$, and $\mathcal{O}^{\mathsf{ccr}}_{\mathsf{H},\Delta}(x, b) := \mathsf{H}(x \oplus \Delta) \oplus b \cdot \Delta$ be an oracle for $x, \Delta \in \{0,1\}^\lambda$ and $b \in \{0,1\}$.

H is (t, q, ρ, ϵ)-*circular correlation robust* if, for any distinguisher $\mathcal{D}$ running in time at most t and making at most q queries to $\mathcal{O}^{\mathsf{ccr}}_{\mathsf{H},\Delta}(\cdot,\cdot)$, and any χ with min-entropy at least ρ, it holds that

$$\left| \Pr_{\Delta \leftarrow \chi}\left[\mathcal{D}^{\mathcal{O}^{\mathsf{ccr}}_{\mathsf{H},\Delta}(\cdot,\cdot)}(1^\lambda) = 1\right] - \Pr_{f \leftarrow \mathcal{F}_{\lambda+1,\lambda}}\left[\mathcal{D}^{f(\cdot,\cdot)}(1^\lambda) = 1\right] \right| \leq \epsilon,$$

where $\mathcal{D}$ cannot query both $(x, 0)$ and $(x, 1)$ for any $x \in \{0,1\}^\lambda$.

A CCR hash function can be constructed from a fixed-key block cipher (e.g., AES) modelled as a random permutation and a linear orthomorphism [GKWY20]. Specifically, the construction $\mathsf{H}_{\mathsf{ccr}}(x) = \mathsf{AES}(\sigma(x)) + \sigma(x)$ is proven to be secure, where σ is the linear orthomorphism. The authors give two efficient constructions for the linear orthomorphism. (1) If x is in a finite field, then we can define $\sigma(x) = c \cdot x$, where $c \neq \{0,1\}$. (2) In the bit-vector representation, another orthomorphism can be defined as $\sigma(x_l \| x_r) = (x_l \oplus x_r) \| x_l$. The overhead introduced by the orthomorphism is negligible compared to running AES [GKWY20].

3.3 Vector Commitment Schemes

A (non-interactive) vector commitment (VC) scheme (with message space $\mathcal{M}$ and commitment space $\mathcal{C}$) in the RO model is defined by four PPT algorithms $(\mathsf{Setup}^{\mathsf{H}}, \mathsf{Commit}^{\mathsf{H}}_{\mathsf{crs}}, \mathsf{Open}^{\mathsf{H}}_{\mathsf{crs}}, \mathsf{Verify}^{\mathsf{H}}_{\mathsf{crs}})$ (see [BBD+23] for details) which satisfy *correctness, binding and hiding.* We also say a VC scheme is iv-based if some of its internal components require an initialisation vector. Although three out of the four algorithms take crs as an implicit input, we do not work in the CRS model that is common in other proof systems since we do not require a trusted setup. The CRS in our work can be considered as a public parameter.

When constructing a VC scheme using the half-tree technique, we need to ensure the hiding property still holds; that is, seeds at unopened indices remain hidden, even after opening a subset of indices.

Definition 3 (VC Real-or-Random Hiding [BBD+23])**.** Let VC be an iv-based vector commitment scheme in the RO model with random oracle H. The *adaptive hiding experiment for* VC*, AdpHiding*, with $N = \mathsf{poly}(\lambda)$ and stateful $\mathcal{A}$ is defined as follows.

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

$$\mathsf{Adv}^{\mathrm{AdpHiding}}_{\mathcal{A},\mathsf{VC}}(\lambda) = \Pr[\mathcal{A} \text{ wins AdpHiding}] - \frac{1}{2},$$

and similarly for the SelHiding game.

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

We must also show that a VC scheme is extractable-binding, so 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 4 (VC Extractable-Binding)**.** Let VC be a vector commitment constructed from two hash functions H_1 and $\mathsf{H}_{\mathsf{ccr}}$ and let Ext be a PPT algorithm defined as:

- $\mathsf{Ext}(\mathsf{pp}, (Q_1, Q_\pi), \mathsf{com}) \to (\mathsf{sd}_i)_{i\in[N]}$: given an instance of VC public parameters pp, query-response lists Q_1 and Q_π, 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 $N \leq \mathsf{poly}(\lambda)$ and stateful adversary $\mathcal{A}$, we define the straight-line extractable-binding game, ExtBind, against VC as follows:

1. $\mathsf{pp} \leftarrow \mathsf{Setup}^{\mathsf{H}_1,\mathsf{H}_{\mathsf{ccr}}}(1^\lambda, N)$
2. $\mathsf{com} \leftarrow \mathcal{A}^{\mathsf{H}_1,\mathsf{H}_{\mathsf{ccr}}}(1^\lambda, \mathsf{pp})$
3. $\{\mathsf{sd}_i^*\}_{i\in[N]} \leftarrow \mathsf{Ext}(\mathsf{pp}, (Q_1, Q_\pi), \mathsf{com})$.
4. $(I, \mathsf{pdecom}_I) \leftarrow \mathcal{A}^{\mathsf{H}_1,\mathsf{H}_{\mathsf{ccr}}}(\mathtt{open})$

5. Ouput 0 (failure) if:
   - $\mathsf{Verify}^{\mathsf{H}_1,\mathsf{H}_{\mathsf{ccr}}}(\mathsf{com}, j^*, \mathsf{pdecom}_{j^*}) = \bot$, i.e., the opening is not valid; or
   - $\mathsf{Verify}^{\mathsf{H}_1,\mathsf{H}_{\mathsf{ccr}}}(\mathsf{com}, j^*, \mathsf{pdecom}_{j^*}) = \{\mathsf{sd}_i\}_{i\in I}$ 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 $\mathsf{Adv}^{\mathrm{ExtBind}}_{\mathcal{A},\mathsf{VC}}(\lambda) = \Pr[1 \leftarrow \mathrm{ExtBind}]$.

Let $\mathcal{A}$ be any PPT adversary with oracle access to H_1 and $\mathsf{H}_{\mathsf{ccr}}$, resulting in query-response lists Q_1 and Q_π respectively. Note that Q_π stores the permutation queries which is used internally by $\mathsf{H}_{\mathsf{ccr}}$. We say VC is *straight-line extractable-binding* w.r.t. Ext if

$$\mathsf{Adv}^{\mathrm{ExtBind}}_{\mathcal{A},\mathsf{VC}}(\lambda) \leq \mathsf{negl}(\lambda).$$

To prove VC extractable-binding, we use a lemma about collisions in random oracle below.

Lemma 1 (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 *has probability of at most* $q^2/2^{2\lambda}$ *encountering a collision.*

4 The Half-Tree Construction

In this section, we use the half-tree technique of Guo et al. [GYW+23] to construct a vector commitment scheme. While their work improved the state-of-the-art schemes such as PPRF, DPF and COT by reducing the number of random permutation calls and the communication by a half; we show that the half-tree technique can also be used to construct a vector commitment scheme with strong security, i.e., adaptive hiding and straight-line extractable binding. Our construction is provably secure in the random permutation model and the random oracle model.

Our vector commitment construction closely follows the half-tree construction of Guo et al. [GYW+23] except that we break the correlation on the leaf nodes to produce $(\mathsf{sd}_i, \mathsf{com}_i)$ using $\mathsf{H}_{\mathsf{ccr}}$, the same function used to expand the tree. This is because in the proof we need to create the secret un-opened leaf using the CCR oracle without knowing the nodes leading to the secret leaf. Specifically, we define $(\mathsf{sd}_i = \mathsf{H}_{\mathsf{ccr}}(X_i), \mathsf{com}_i := \mathsf{H}_{\mathsf{ccr}}(X_i \oplus 1)\|\mathsf{H}_{\mathsf{ccr}}(X_i \oplus 2)$ where X_i is corresponding parent in previous layer. The final commitment com to the tree is still computed using the random oracle H_1 as in the FAEST specification. Our construction, which we name $\mathsf{VC}_{\mathsf{HT}}$, is given in Figure 3.

4.1 All-but-one Hiding Security

We first prove that our half-tree based construction $\mathsf{VC}_{\mathsf{HT}}$ is all-but-one-hiding.

Remark 1. Guo et al. proved that their $\mathsf{H}_{\mathsf{ccr}}$ construction in the random permutation model is $(t, |Q_\pi|, \rho, \epsilon_{\mathsf{ccr}})$-circular collision resistant [GKWY20, Theorem 5] with

$$\epsilon_{\mathsf{ccr}} = \frac{2t|Q_\pi|}{2^\rho} + \frac{|Q_\pi|^2}{2^{\lambda+1}}.$$

Other constructions of $\mathsf{H}_{\mathsf{ccr}}$ exist, which we explore in Section 6, that offer better security but less efficiency.

Theorem 1. $\mathsf{VC}_{\mathsf{HT}}$ *is all-but-one adaptive hiding Let* $\mathsf{H}_{\mathsf{ccr}}$ *be* (t, q, ρ, ϵ)*-circular correlation robust and let* H_1 *be a random oracle Then, for any adversary* $\mathcal{A}$ *in the adaptive hiding vector commitment game against the* $\mathsf{VC}_{\mathsf{HT}}$ *scheme making* $|Q_1|$ *queries to* H_1*, we have*

$$\mathsf{Adv}^{AdpHiding}_{\mathcal{A},\mathsf{VC}_{\mathsf{HT}}}(\lambda) \leq \frac{|Q_1|}{2^{2\lambda}} + \mathsf{Adv}^{PRG}_{\mathcal{B}_{\mathsf{PRG}},\mathsf{PRG}}(\lambda) + \epsilon.$$

Figure 3: $\mathsf{VC}_{\mathsf{HT}}(N)$ the all-but-one vector commitment.

Parameters:
- Tree size $N = 2^d \in \mathbb{N}$, computational security parameter λ.
- $\mathsf{H}_{\mathsf{ccr}} : \{0,1\}^\lambda \to \{0,1\}^\lambda$ circular correlation-robust hash function.
- $\mathsf{H}_1 : \{0,1\}^* \to \{0,1\}^{2\lambda}$ collision-resistant hash function.

$\mathsf{Setup}(1^\lambda, N)$:
1. Compute $\mathsf{crs}_0 \leftarrow \mathsf{H}_{\mathsf{ccr}}.\mathsf{Setup}(1^\lambda)$ resp. $\mathsf{crs}_1 \leftarrow \mathsf{H}_1.\mathsf{Setup}(1^\lambda)$.
2. Define $\mathsf{crs} = (\lambda, d, \mathsf{crs}_0, \mathsf{crs}_1)$, which is implicitly input to all other algorithms.

$\mathsf{Commit}(\mathsf{sd}, \mathsf{iv})$:
1. Compute $X_0 \| X_1 \leftarrow \mathsf{PRG}(\mathsf{sd}, \mathsf{iv}; 2\lambda)$.
2. Compute $\Delta := X_0 \oplus X_1$.
3. For $i \in [2, d]$, define
$$X_{b_1,\ldots,b_{i-1},0} = \mathsf{H}_{\mathsf{ccr}}(X_{b_1,\ldots,b_{i-1}}),$$
$$X_{b_1,\ldots,b_{i-1},1} = \mathsf{H}_{\mathsf{ccr}}(X_{b_1,\ldots,b_{i-1}}) \oplus X_{b_1,\ldots,b_{i-1}}$$
where each $b_j \in \{0,1\}$ for all $j \in [1, i-1]$.
4. For $j \in [0, N-1]$, bit-decompose j as $\sum_{i=1}^{d} 2^{d+1-i} \cdot b_i$ for $b_i \in \{0,1\}$, let
$$\mathsf{sd}_j := \mathsf{H}_{\mathsf{ccr}}(X_{b_1,\ldots,b_d}) \text{ and } \mathsf{com}_j := \mathsf{H}_{\mathsf{ccr}}(X_{b_1,\ldots,b_d} \oplus 1) \| \mathsf{H}_{\mathsf{ccr}}(X_{b_1,\ldots,b_d} \oplus 2).$$
5. Compute $\mathsf{com} := \mathsf{H}_1(\mathsf{iv} \| \mathsf{com}_0 \| \ldots \| \mathsf{com}_{N-1})$.
6. Output the commitment com, the opening $\mathsf{decom} := (\Delta, X)$ and the seeds $(\mathsf{sd}_0, \ldots, \mathsf{sd}_{N-1})$.

$\mathsf{Open}(\mathsf{decom} = (\Delta, X), j^* \in [N])$:
1. Bit-decompose j^* as $\sum_{i=1}^{d} 2^{d+1-i} \cdot b_i$ for $b_i \in \{0,1\}$, compute the path $K_{j^*} = \{X_{\bar{b}_1}, X_{b_1,\bar{b}_2}, \ldots, X_{b_1,b_2,\ldots\bar{b}_d}\}$ as in Commit.
2. Output the all-but-one opening $\mathsf{pdecom}_{j^*} := (\mathsf{com}_{j^*}, K_{j^*})$.

$\mathsf{Verify}(\mathsf{com}, I = [0, N-1] \setminus j^*, \mathsf{decom}_I = (\mathsf{com}_{j^*}, K_{j^*}))$:
1. Recompute $(\mathsf{sd}'_i, \mathsf{com}'_i)$ from decom_I for $i \neq j^*$.
2. Let $\mathsf{com}'_{j^*} := \mathsf{com}_{j^*}$.
3. If $\mathsf{com} \neq \mathsf{com}'$ where $\mathsf{com}' := \mathsf{H}_1(\mathsf{com}'_0, \ldots, \mathsf{com}'_{N-1})$ output $\perp$.
   Otherwise output $(\mathsf{sd}'_i)_{i \in I}$.

In the proof, we rely on the following observation of Guo et al. [GYW+23]. Suppose we start with two nodes $(X, X \oplus \Delta)$ in the correlated GGM tree, we show an example of how to construct inputs for adversary $\mathcal{A}$ using the CCR oracle $\mathcal{O}_{\mathsf{ccr}}$, i.e., without knowing Δ. Suppose the secret path is $j^* = (1, 0)$, we sample the first off-path node $X \xleftarrow{\$} \{0,1\}^\lambda$. Then the second off-path node $X_{(1,1)} \leftarrow \mathsf{H}_{\mathsf{ccr}}(X \oplus \Delta) \oplus (X \oplus \Delta)$ can be computed using $\mathcal{O}_{\mathsf{ccr}}(X, 1) \oplus X$. Next we need to compute the secret seed $\mathsf{sd}_{(1,0)} \leftarrow \mathsf{H}_{\mathsf{ccr}}(\mathsf{H}_{\mathsf{ccr}}(X \oplus \Delta))$ and commitment $\mathsf{com}_{(1,0)} \leftarrow \mathsf{H}_{\mathsf{ccr}}(\mathsf{H}_{\mathsf{ccr}}(X \oplus \Delta) \oplus 1) \| \mathsf{H}_{\mathsf{ccr}}(\mathsf{H}_{\mathsf{ccr}}(X \oplus \Delta) \oplus 2)$. This can be done using calls to $\mathcal{O}_{\mathsf{ccr}}(X_{(1,1)} \oplus X, 0)$ and $\mathcal{O}_{\mathsf{ccr}}(X_{(1,1)} \oplus X \oplus 1, 0)$. Observe that in the real world, $\mathcal{O}_{\mathsf{ccr}}(X_{(1,1)} \oplus X, 0) = \mathsf{H}_{\mathsf{ccr}}(\mathsf{H}_{\mathsf{ccr}}(X \oplus \Delta))$, as desired. Using $\mathcal{O}_{\mathsf{ccr}}$ with a fixed second input 0 for the left and the right child is a trick that bypasses the need to compute the parent of $(\mathsf{sd}_{(1,0)}, \mathsf{com}_{(1,0)})$. This example is illustrated in Figure 4 and these ideas are generalized in the full proof below. We prove the adaptive hiding scenario where the adversary is stronger than the selective hiding scenario.

Proof of Theorem 1. Let $\eth_0$ be the real game where $\mathcal{A}$ is given the real opening and seeds. Let $\eth_1$ be the same as $\eth_0$ except the seed on the secret path is random.

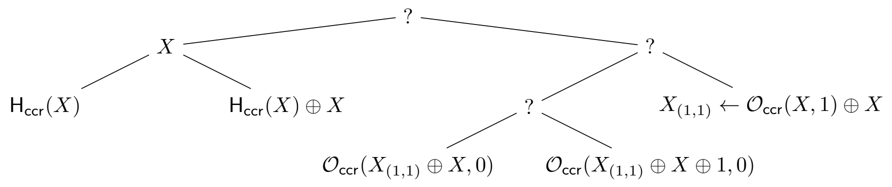
?
X
?
$\mathsf{H}_{\mathsf{ccr}}(X)$
$\mathsf{H}_{\mathsf{ccr}}(X) \oplus X$
?
$X_{(1,1)} \leftarrow \mathcal{O}_{\mathsf{ccr}}(X,1) \oplus X$
$\mathcal{O}_{\mathsf{ccr}}(X_{(1,1)} \oplus X, 0)$
$\mathcal{O}_{\mathsf{ccr}}(X_{(1,1)} \oplus X \oplus 1, 0)$

Figure 4: Example of how to use $\mathcal{O}_{\mathsf{ccr}}$ to construct a VC opening where the secret path is $(1, 0)$, i.e., traverse right and then traverse left.

The reduction. Let $\mathcal{B}$ be a distinguisher which has access to the CCR oracle $\mathcal{O}_{\mathsf{ccr}}(x, b)$. Recall that the oracle takes input $x \in \lambda$ and a bit b, and then outputs either $\mathsf{H}_{\mathsf{ccr}}(x \oplus \Delta) \oplus b \cdot \Delta$ or the output of some truly random function, depending on some internal bit b^*. Further, $\mathcal{B}$ can program the random oracle $\mathcal{O}_{\mathsf{H}_1}$. The goal of $\mathcal{B}$ is to output a bit that distinguishes between the two cases. Distinguisher $\mathcal{B}$, which can be seen as an adversary against the CCR game, works as follows in the adaptive setting.

1. Sample $b \xleftarrow{\$} \{0,1\}$, $\mathsf{sd} \xleftarrow{\$} \{0,1\}^\lambda$ and $\mathsf{iv} \in \{0,1\}^{2\lambda}$.
2. Sample a commitment $\mathsf{com} \xleftarrow{\$} \{0,1\}^\lambda$ instead of executing $\mathsf{Commit}^{\mathsf{H}}_{\mathsf{crs}}(\mathsf{sd})$.
3. Receive the challenge $I \leftarrow \mathcal{A}(1^\lambda, \mathsf{crs}, \mathsf{com})$, where $I = [N] \setminus j^*$.
4. View j^* as the secret path in the tree in binary, i.e., $j^* = \sum_{i=1}^{d} 2^{d+1-i} b_i$. If $b_i = 0$, it means traverse to the left child, otherwise traverse to the right child.
5. Assume w.l.o.g. $b_1 = 1$, i.e., the secret path begins on the right sub-tree. Then we simulate the execution of $\mathsf{Commit}(\mathsf{sd})$ as follows.
   (a) Sample $X_{\bar{b}_1} \leftarrow \{0,1\}^\lambda$ as the left node.
   (b) For $i \in [2, d]$, compute
$$X_{b_1,\ldots,\bar{b}_i} \leftarrow \mathcal{O}_{\mathsf{ccr}}\left(\bigoplus_{j\in[i-1]} X_{b_1,\ldots,\bar{b}_j}, \bar{b}_i\right) \oplus \left(\bar{b}_i \cdot \bigoplus_{j\in[i-1]} X_{b_1,\ldots,\bar{b}_j}\right)$$
$$X'_{b_1,\ldots,\bar{b}_i} \leftarrow X_{b_1,\ldots,\bar{b}_i} \oplus \left(\bar{b}_i \cdot \bigoplus_{j\in[i-1]} X_{b_1,\ldots,\bar{b}_j}\right).$$
6. Given the sibling nodes $\{X_{\bar{b}_1}, \ldots, X_{b_1,\ldots,\bar{b}_d}\}$, all nodes before the final level in the tree can be obtained using $\mathsf{H}_{\mathsf{ccr}}$ except $X_{b_1,\ldots,b_d}$ (i.e., X_{j^*}).
7. Next, the leaves can be computed as $\mathsf{sd}_j \leftarrow \mathsf{H}_{\mathsf{ccr}}(X_j)$, $\mathsf{com}_j \leftarrow \mathsf{H}_{\mathsf{ccr}}(X_j \oplus 1) \| \mathsf{H}_{\mathsf{ccr}}(X_j \oplus 2)$, for $j \in [0, N-1] \setminus \{j^*\}$.
8. Additionally, $\mathsf{sd}_{j^*} \leftarrow \begin{cases} \mathcal{O}_{\mathsf{ccr}}(X'_{b_1,\ldots\bar{b}_d}, 0), & \text{if } b = 0 \\ \text{random from } \{0,1\}^\lambda, & \text{if } b = 1 \end{cases}$,
   and $\mathsf{com}_{j^*} \leftarrow \mathcal{O}_{\mathsf{ccr}}(X'_{b_1,\ldots\bar{b}_d} \oplus 1, 0) \| \mathcal{O}_{\mathsf{ccr}}(X'_{b_1,\ldots\bar{b}_d} \oplus 2, 0)$.
9. Program $\mathcal{O}_{\mathsf{H}_1}$ to output com on input $\mathsf{iv} \| \mathsf{com}_0 \| \ldots \| \mathsf{com}_{N-1}$ or abort if it has already been queried.
10. Assign $\mathsf{decom}_I \leftarrow \{\mathsf{com}_{j^*}, \{X_{\bar{b}_1}, \ldots, X_{b_1,\ldots,\bar{b}_d}\}\}$ and then call
$$b' \leftarrow \mathcal{A}(\{\mathsf{sd}_i\}_{i\in[0,N-1]}, \mathsf{decom}_I, \mathsf{iv}).$$

11. Output 1 if $b = b'$, otherwise output 0.

The case for $b_1 = 0$ is symmetrical since we can start with two nodes $(X, X \oplus \Delta)$ for an unknown Δ. This is the same as generating $Y \xleftarrow{\$} \{0,1\}^\lambda$ and setting the two initial nodes to be $Y \oplus \Delta = X, Y = X \oplus \Delta$.

$\mathcal{B}$'s advantage. Recall that $b^* \in \{0,1\}$ is the CCR oracle's hidden bit; if $b^* = 1$ then $\mathcal{B}$ is in the real world ($\mathcal{O}_{\mathsf{ccr}}$ runs $\mathsf{H}_{\mathsf{ccr}}$ internally) and $b^* = 0$ is the ideal world. The advantage of $\mathcal{B}$ in the CCR distinguishing game is

$$\mathsf{Adv}^{\text{CCR}}_{\mathcal{B},\mathsf{H}_{\mathsf{ccr}}}(\lambda) = \left|\Pr[1 \leftarrow \mathcal{B}^{\mathcal{O}_{\mathsf{ccr}}}(1^\lambda) \mid b^* = 1] - \Pr[1 \leftarrow \mathcal{B}^{\mathcal{O}_{\mathsf{ccr}}}(1^\lambda) \mid b^* = 0]\right|.$$

There are two differences between the adaptative hiding game of Definition 3 and the game that $\mathcal{B}$ simulates for $\mathcal{A}$:

(i) $\mathcal{B}$ aborts if the programming of $\mathcal{O}_{\mathsf{H}_1}$ fails; and
(ii) X_0 or X_1 is uniformly random, instead of being computed from $\mathsf{PRG}(\mathsf{sd}, \mathsf{iv}; 2\lambda)$.

The first difference is noticeable if $\mathcal{A}$ queries $\mathcal{O}_{\mathsf{H}_1}$ with the same iv as in the programming attempt; since $\mathsf{iv} \in \{0,1\}^{2\lambda}$ is sampled at random and hidden from $\mathcal{A}$ before the programming, we have

$$\Pr[\mathcal{A} \text{ notices (i)}] \leq \frac{|Q_1|}{2^{2\lambda}} \tag{1}$$

The second difference reduces to distinguishing the output of PRG from a uniformly sampled output, so the probability that $\mathcal{A}$ behaves differently is exactly $\mathsf{Adv}^{\text{PRG}}_{\mathcal{B}_{\mathsf{PRG}},\mathsf{PRG}}(\lambda)$, for a PPT reduction $\mathcal{B}_{\mathsf{PRG}}$ which plays the PRG game against PRG using $\mathcal{A}$ as a subroutine. Therefore we have

$$\Pr[\mathcal{A} \text{ notices (ii)}] \leq \mathsf{Adv}^{\text{PRG}}_{\mathcal{B}_{\mathsf{PRG}},\mathsf{PRG}}(\lambda) \tag{2}$$

Observe that if $b^* = 1$, then, after changes (i) and (ii), $\mathcal{A}$'s view is distributed identically to $\eth_0$ if $b = 1$ (i.e., $\mathcal{B}$ uses the real seed sd_{j^*}) and $\mathcal{A}$'s view is distributed identically to $\eth_1$ if $b = 0$. So we can write

$$\Pr[1 \leftarrow \mathcal{B}^{\mathcal{O}_{\mathsf{ccr}}}(1^\lambda) \mid b^* = 1] = \Pr[\mathcal{A} \text{ wins changed VC game}].$$

On the other hand, if $b^* = 0$, then the view of $\mathcal{A}$ is always sampled uniformly at random. So $\mathcal{A}$'s output, i.e., b', is independent of the bit b selected by $\mathcal{B}$. Thus we have

$$\Pr[1 \leftarrow \mathcal{B}^{\mathcal{O}_{\mathsf{ccr}}}(1^\lambda) \mid b^* = 0] = \frac{1}{2}.$$

Putting Equations (1) and (2) and the probabilities together, we arrive at

$$\mathsf{Adv}^{\text{AdpHiding}}_{\mathcal{A},\mathsf{VC}_{\mathsf{HT}}}(\lambda) \leq \frac{|Q_1|}{2^{2\lambda}} + \mathsf{Adv}^{\text{PRG}}_{\mathcal{B}_{\mathsf{PRG}},\mathsf{PRG}}(\lambda) + \mathsf{Adv}^{\text{CCR}}_{\mathcal{B},\mathsf{H}_{\mathsf{ccr}}}(\lambda).$$

Assuming that $\mathsf{H}_{\mathsf{ccr}}$ is (t, q, ρ, ϵ)-CCR yields the theorem statement. □

4.2 Extractable-Binding Security

We now prove our VC satisfies extractable binding property in random oracle and random permutation model. Thanks to the extension of the length of commitment com_i to 2λ, our VC is computationally binding with the security level λ since the probability of collision for com_i will be bounded by $q^2/2^{2\lambda}$, where q is the number of queries to permutation oracle (see lemma 2 for details).

Lemma 2 (Collision probability of concatenated CCR)**.** *Given Q_π random permutation queries, and $\mathsf{H}_{\mathsf{ccr}}$ is defined in Remark 1, then the collision probability*

$$\begin{aligned}&\Pr_{x,x'\in Q_\pi}\left(x\neq x' \mid \mathsf{H}_{\mathsf{ccr}}(x\oplus 1)=\mathsf{H}_{\mathsf{ccr}}(x'\oplus 1)\wedge \mathsf{H}_{\mathsf{ccr}}(x\oplus 2)=\mathsf{H}_{\mathsf{ccr}}(x'\oplus 2)\right)\\ \leq\ & \frac{|Q_\pi|(|Q_\pi|-1)}{2^{2\lambda+1}}.\end{aligned}$$

Proof of Lemma 2. Using Bayer's formula we rewrite the probability of collision as below:

$$\Pr_{x,x'\in Q_\pi}\left[x\neq x' \mid \mathsf{H}_{\mathsf{ccr}}(x\oplus 1)=\mathsf{H}_{\mathsf{ccr}}(x'\oplus 1)\wedge \mathsf{H}_{\mathsf{ccr}}(x\oplus 2)=\mathsf{H}_{\mathsf{ccr}}(x'\oplus 2)\right]=A\cdot B$$

where $A:=\Pr_{x,x'\in Q_\pi}\left(x\neq x' \mid \mathsf{H}_{\mathsf{ccr}}(x\oplus 1)=\mathsf{H}_{\mathsf{ccr}}(x'\oplus 1)\right)$
and $B:=\Pr\left(\mathsf{H}_{\mathsf{ccr}}(x\oplus 2)=\mathsf{H}_{\mathsf{ccr}}(x'\oplus 2) \mid \mathsf{H}_{\mathsf{ccr}}(x\oplus 1)=\mathsf{H}_{\mathsf{ccr}}(x'\oplus 1)\right)$.

To compute A, fix an arbitrary $x\in Q_\pi$, we have

$$\begin{aligned}&\Pr_{x'\in Q_\pi}\left(\exists x\neq x' \mid \mathsf{H}_{\mathsf{ccr}}(x\oplus 1)=\mathsf{H}_{\mathsf{ccr}}(x'\oplus 1)\right)\\ =\ &\Pr_{x'\in Q_\pi}\left(\exists x\neq x' \mid \pi(\sigma(x\oplus 1))\oplus\sigma(x\oplus 1)=\pi(\sigma(x'\oplus 1))\oplus\sigma(x'\oplus 1)\right)\\ \leq\ &(|Q_\pi|-1)\cdot\Pr\left(x\neq x' \mid \pi(\sigma(x'\oplus 1))=\pi(\sigma(x\oplus 1))\oplus\sigma(x\oplus 1)\oplus\sigma(x'\oplus 1)\right)\\ \leq\ &(|Q_\pi|-1)\cdot\frac{1}{2^\lambda}\ \text{(since π is a random permutation)}.\end{aligned}$$

Then consider $x\in\ Q_\pi$, each pair (x,x') is computed twice. we have

$$A=\Pr_{x,x'\in Q_\pi}\left(x\neq x' \mid \mathsf{H}_{\mathsf{ccr}}(x\oplus 1)=\mathsf{H}_{\mathsf{ccr}}(x'\oplus 1)\right)\leq\frac{1}{2}\cdot\frac{|Q_\pi|(|Q_\pi|-1)}{2^\lambda},$$

To compute B we observe about the instantiation of $\mathsf{H}_{\mathsf{ccr}}$ i.e $\mathsf{H}_{\mathsf{ccr}}(x)=\pi(\sigma(x))+\sigma(x)$ where $\sigma(x):=x_L\oplus x_R\|x_L$. For $x\neq x'$ such that $\mathsf{H}_{\mathsf{ccr}}(x\oplus 1)=\mathsf{H}_{\mathsf{ccr}}(x'\oplus 1)$ we then compute the probability of $\mathsf{H}_{\mathsf{ccr}}(x\oplus 2)=\mathsf{H}_{\mathsf{ccr}}(x'\oplus 2)$. From the instantiation of $\mathsf{H}_{\mathsf{ccr}}$, we have if $\mathsf{H}_{\mathsf{ccr}}(x\oplus 2)=\mathsf{H}_{\mathsf{ccr}}(x'\oplus 2)$ then

$$\begin{aligned}&\mathsf{H}_{\mathsf{ccr}}(x\oplus 1)+\mathsf{H}_{\mathsf{ccr}}(x\oplus 2)=\mathsf{H}_{\mathsf{ccr}}(x'\oplus 1)+\mathsf{H}_{\mathsf{ccr}}(x'\oplus 2)\\ &\implies \pi(\sigma(x\oplus 1))+\pi(\sigma(x\oplus 2))=\pi(\sigma(x'\oplus 1))+\pi(\sigma(x'\oplus 2))\\ &\implies \pi(\sigma(x\oplus 2))+\pi(\sigma(x'\oplus 2))=\pi(\sigma(x\oplus 1))+\pi(\sigma(x'\oplus 1))\leq\frac{1}{2^\lambda}\end{aligned}$$

since π is random permutation and $\pi(\sigma(x\oplus 1))+\pi(\sigma(x'\oplus 1))$ is fixed. Therefore $B\leq\frac{1}{2^\lambda}$. □

Theorem 2 ($\mathsf{VC}_{\mathsf{HT}}$ is extractable-binding)**.** *Let H_1 be a random oracle and let $\mathsf{H}_{\mathsf{ccr}}$ be defined in the random permutation model by $\mathsf{H}_{\mathsf{ccr}}: x\mapsto\mathcal{O}_\pi(\sigma(x))\oplus\sigma(x)$, where σ is a linear orthomorphism. Then, for any PPT adversary $\mathcal{A}^{\mathsf{H}_1,\mathcal{O}_\pi}$ making $|Q_1|$ queries to H_1, and assuming this construction of $\mathsf{H}_{\mathsf{ccr}}$ is $(t,|Q_\pi|,\rho,\epsilon_{\mathsf{ccr}})$-CCR, there exist an extractor algorithm Ext such that*

$$\mathsf{Adv}^{ExtBinding}_{\mathcal{A},\mathsf{VC}_{\mathsf{HT}}}(\lambda)\leq\frac{|Q_1|^2}{2^{2\lambda}}+\frac{|Q_\pi|(|Q_\pi|-1)}{2^{2\lambda+1}}.$$

Proof of Theorem 2. We define the extractor algorithm Ext which takes as input the tuple $(\mathsf{pp},(Q_1,Q_\pi),\mathsf{com})$, where Q_1 and Q_π result from $\mathcal{A}$'s queries to H_1 and $\mathcal{O}_\pi$ respectively, and com is output by $\mathcal{A}$. First we argue that Ext is capable of inverting $\mathsf{H}_{\mathsf{ccr}}$.

Extractability of $\mathsf{H}_{\mathsf{ccr}}$. To recover sd_i from any com_i, Ext must first invert $\mathsf{H}_{\mathsf{ccr}}$. Since the latter is defined in terms of $\mathcal{O}_\pi$, the extractor only has access to $\mathcal{A}$'s queries on Q_π. However, note that if $Y = \mathsf{H}_{\mathsf{ccr}}(X)$, then it holds that $Y \oplus \sigma(X) = \mathcal{O}_\pi(\sigma(X))$ and therefore the pair $(\sigma(X), Y \oplus \sigma(X))$ must appear on Q_π.

Thus, when given com_i, Ext checks whether two entries $(x_1, \mathcal{O}_\pi(x_1)), (x_2, \mathcal{O}_\pi(x_2))$ exists on Q_π such that $x_1 \oplus \mathcal{O}_\pi(x_1) \| x_2 \oplus \mathcal{O}_\pi(x_2) = \mathsf{com}_i$ and $\sigma^{-1}(x_1) \oplus 1 = \sigma^{-1}(x_2) \oplus 2$; if it does, then Ext concludes that $\mathsf{com}_i = \mathsf{H}_{\mathsf{ccr}}(\sigma^{-1}(x_1)) \| \mathsf{H}_{\mathsf{ccr}}(\sigma^{-1}(x_2))$, and therefore that the committed leaf is $X_i = \sigma^{-1}(x_1) \oplus 1$; if Ext finds no such pair of queries on Q_π, it outputs $\perp$ as the committed leaf.

Extraction. Then, Ext proceeds as follows:

1. Ext parses Q_1 to find the pre-image $\mathsf{iv}\|\mathsf{com}_0\| \dots \|\mathsf{com}_{N-1}$ of com under H_1. If this either does not exist or is multiply-defined, then Ext outputs $\perp$ for all $\{\mathsf{sd}_i^*\}_{i \in [N]}$.

   We show that the probability Ext outputs $\perp$ is at most $|Q_1|^2/2^{2\lambda}$. By assumption that $\mathcal{A}$ wins the ExtBind game, com is valid and verified using the H_1 oracle, therefore Q_1 must have at least one pre-image for com. Multiple preimages would consist in a collision for H_1, since it is modelled as a random oracle then Lemma 1 implies that the probability Ext outputs $\perp$ for this reason is bound by $|Q_1|^2/2^{2\lambda}$.

2. If Ext did not output $\perp$ for every seed, then for each com_i, for $i \in [N]$, Ext uses Q_π to extract the tree leaf X_i as described above.

   - If there is an unique X_i, Ext sets $\mathsf{sd}_i^* = \mathsf{H}_{\mathsf{ccr}}(X_i)$.
   - If Ext finds no such X_i, or it finds multiple suitable leaves, then it sets $\mathsf{sd}_i^* = \perp$.

   By assumption that $\mathcal{A}$ wins the ExtBind game, every com_i for $i \in I$, must have at least one valid $\mathsf{H}_{\mathsf{ccr}}$ preimage that is extractable using Q_π, and there can be at most one value of $i \in [N]$ for which $\mathsf{sd}_i^* = \perp$. If *any* com_i has more than one preimage, then this contradicts the collision-resistance of $\mathsf{H}_{\mathsf{ccr}}$; therefore the probability that this happens is bound by the probability in Lemma 2.

3. If Ext did not output $\perp$ for more than one seed, then the adversary can only win if (1) $\mathsf{sd}_i^* \neq \perp$ for $i \in I$ and (2) there exists $i^* \in I$ such that $\mathsf{sd}_{i^*} \neq \mathsf{sd}_{i^*}^*$ for $\{\mathsf{sd}_i\}_{i \in I} \leftarrow \mathsf{Verify}^{\mathsf{H}_1, \mathcal{O}_\pi}(\mathsf{com}, j^*, \mathsf{pdecom}_{j^*})$ (i.e., verification passes but extraction failed). However, case (2) is impossible by the existence and uniqueness of the pre-image X_i of com_i for $i \in I$ and the deterministic derivation of sd_i from X_i.

Thus the only way for $\mathcal{A}$ to win is for Ext to fail in the extraction, and by the arguments above, we have

$$\mathsf{Adv}^{\mathrm{ExtBind}}_{\mathcal{A}, \mathsf{VC}_{\mathsf{HT}}}(\lambda) \leq \frac{|Q_1|^2}{2^{2\lambda}} + \frac{|Q_\pi|(|Q_\pi| - 1)}{2^{2\lambda+1}}$$

which concludes the proof. □

5 Multi-Tree Vector Commitment

In this section, we show how to generalize our $\mathsf{VC}_{\mathsf{HT}}$ to a multi-tree vector commitment scheme $\mathsf{VC}_{\mathsf{MT}}(N, \tau)$ in Figure 5. Intuitively, our multi-tree is a repetition of τ single tree $\mathsf{VC}_{\mathsf{HT}}$ by carefully using the randomness of PRG, CCR hash function, and random oracle to ensure the security is maintained. In particular, the $\{\mathsf{sd}^i\}_{i \in [0, \tau-1]}$ for each tree is generated from PRG with a random input sd.

Figure 5: $\mathsf{VC}_{\mathsf{MT}}(N, \tau)$ the all-but-τ vector commitment.

```
Parameters:
  • Tree depth $N = 2^d \in \mathbb{N}$, computational security parameter $\lambda$.
  • A repetition parameter $\tau \in \mathbb{N}$.
  • $\mathsf{H}_{\mathsf{ccr}} : \{0,1\}^\lambda \to \{0,1\}^\lambda$ circular correlation-robust hash function.
  • $\mathsf{H}_1 : \{0,1\}^* \to \{0,1\}^{2\lambda}$ collision-resistant hash function and $\mathsf{H}$ is instantiated as a
    random oracle.

$\mathsf{Setup}(1^\lambda, N, \tau)$:
  1. Compute $\mathsf{crs}_0 \leftarrow \mathsf{H}_{\mathsf{ccr}}.\mathsf{Setup}(1^\lambda)$ resp. $\mathsf{crs}_1 \leftarrow \mathsf{H}_1.\mathsf{Setup}(1^\lambda)$.
  2. Define $\mathsf{crs} = (\lambda, d, \tau, \mathsf{crs}_0, \mathsf{crs}_1)$ which is implicitly input to all other algorithms.

$\mathsf{Commit}(\mathsf{sd}, \mathsf{iv})$:
  1. Compute $\mathsf{sd}^0 \| \dots \| \mathsf{sd}^{\tau-1} \leftarrow \mathsf{PRG}(\mathsf{sd}, \mathsf{iv}; \tau\lambda)$.
  2. For $i \in [\tau]$, compute $(\mathsf{com}^i, \mathsf{decom}^i, (\mathsf{sd}^i_j)_{j \in [N]}) \leftarrow \mathsf{VC}_{\mathsf{HT}}.\mathsf{Commit}(\mathsf{sd}^i, \mathsf{iv})$.
  3. Compute $h_{\mathsf{com}} \leftarrow \mathsf{H}_1(\mathsf{iv} \| \mathsf{com}^0 \| \dots \| \mathsf{com}^{\tau-1})$.
  4. Let $\mathsf{decom} := (\mathsf{decom}^0, \dots, \mathsf{decom}^{\tau-1})$.
  5. Output $(h_{\mathsf{com}}, \mathsf{decom}, (\mathsf{sd}^i_j)_{i \in [\tau], j \in [N]})$.

$\mathsf{Open}\left(\mathsf{decom}, I = \cup_{i=0}^{\tau-1} [iN, (i+1)N - 1] \setminus \tilde{j}^i\right)$:
  1. For $i \in [\tau]$, compute
       1. Write $I^i = [N] \setminus (\tilde{j}^i - iN)$.
       2. Compute $\mathsf{pdecom}_{I^i} \leftarrow \mathsf{VC}_{\mathsf{HT}}.\mathsf{Open}(\mathsf{decom}^i, I^i)$.
  2. Output the all-but-$\tau$ opening $\mathsf{pdecom}_I := (\mathsf{pdecom}_{I^i})_{i \in [\tau]}$.

$\mathsf{Verify}\ (h_{\mathsf{com}}, I, \mathsf{pdecom}_I)$:
  1. For $i \in [\tau]$, compute
       1. Write $I^i = [N] \setminus (\tilde{j}^i - iN)$.
       2. Recompute $(\overline{\mathsf{sd}}^i_j, \overline{\mathsf{com}}^i_j)_{j \in I^i}$ from $\mathsf{pdecom}_{I^i}$.
       3. Let $\overline{\mathsf{com}}^i_{\tilde{j}^i} := \mathsf{com}^i_{\tilde{j}^i}$ from $\mathsf{pdecom}_{I^i}$.
       4. Compute $\overline{\mathsf{com}}^i \leftarrow \mathsf{H}_1(\mathsf{iv} \| \overline{\mathsf{com}}^i_0 \| \dots \| \overline{\mathsf{com}}^i_{N-1})$.
  2. Compute $h'_{\mathsf{com}} \leftarrow \mathsf{H}_1(\mathsf{iv} \| \overline{\mathsf{com}}^0 \| \dots \| \overline{\mathsf{com}}^{\tau-1})$.
  3. If $h'_{\mathsf{com}} \neq h_{\mathsf{com}}$, output $\perp$; otherwise output $(\overline{\mathsf{sd}}^i_j)_{i \in [\tau], j \in [I^i]}$.
```

5.1 All-but-*tau* Hiding Security

The multi-tree vector commitment scheme of Figure 5 keeps τ messages out of its opening, instead of just one. Because of this difference, we first sketch the following definition.

Definition 5 (Multi-Tree VC all-but-τ Hiding)**.** Let $\mathsf{VC}_{\mathsf{MT}}(N, \tau)$ be an iv-based vector commitment scheme in the RO model with random oracle H_1 the *adaptive multi-hiding experiment* for $\mathsf{VC}_{\mathsf{MT}}$, *AdpMulHiding*, with $N = \mathsf{poly}(\lambda)$ and stateful $\mathcal{A}$ is defined similarly to the AdpHiding experiment, with the difference that the set I of opened messages must omit τ indices (rather than one) and that these τ messages are either all real or all random when presented to $\mathcal{A}$.

Theorem 3. *Let $\mathsf{H}_{\mathsf{ccr}}$ be (t, q, ρ, ϵ)-circular correlation robust and let H_1 be a random oracle. Then, for any PPT adversary $\mathcal{A}$ in the adaptive multi-hiding vector commitment game against the $\mathsf{VC}_{\mathsf{MT}}$ scheme, we have*

$$\mathsf{Adv}^{AdpMulHiding}_{\mathcal{A}, \mathsf{VC}_{\mathsf{MT}}}(\lambda) \leq \frac{|Q_1|}{2^{2\lambda}} + (1 + \tau)\mathsf{Adv}^{PRG}_{\mathcal{B}_{\mathsf{PRG}}, \mathsf{PRG}}(\lambda) + \tau \cdot \epsilon.$$

Theorem 3, sketch. Similarly to the proof of Theorem 1, we first perform two game changes:

(i) We sample $h_{\mathsf{com}} \xleftarrow{\$} \{0,1\}^{2\lambda}$ to give to $\mathcal{A}$ before later programming $\mathcal{O}_{\mathsf{H}_1}$ appropriately and aborting if that fails.

(ii) We sample the root seeds $\mathsf{sd}^0, \ldots, \mathsf{sd}^{\tau-1}$ from $\{0,1\}^\lambda$ instead of computing them with $\mathsf{PRG}(\mathsf{sd}, \mathsf{iv}; \tau\lambda)$.

Letting $\eth$ denote the AdpMulHiding game originally played by $\mathcal{A}$, and $\eth'$ denote the game after these two changes, we have

$$\Pr[\mathcal{A} \text{ wins } \eth] \le \frac{|Q_1|}{2^{2\lambda}} + \mathsf{Adv}^{\mathrm{PRG}}_{\mathcal{B}_{\mathsf{PRG}}, \mathsf{PRG}}(\lambda) + \Pr[\mathcal{A} \text{ wins } \eth'].$$

Now that the random oracle is programmed, and the seeds are uniformly random, we run a hybrid argument where the output of each $\mathsf{VC}_{\mathsf{HT}}.\mathsf{Commit}$ call has its hidden message selected by I replaced by a random value. Applying Theorem 1 (with the omission of the $\frac{|Q_1|}{2^{2\lambda}}$ term since the programming happens outside of $\mathsf{VC}_{\mathsf{HT}}$) yields the final advantage statement. □

5.2 Extractable-Binding Security

The multi-tree construction $\mathsf{VC}_{\mathsf{MT}}$ is extractable-binding in a similar way to the single-tree construction; however, since there are additional calls to H_1 modelled as a random oracle, one of the terms is looser by a factor of τ. We formalise this in the following theorem.

Theorem 4. *Let H_1 be a random oracle and let $\mathsf{H}_{\mathsf{ccr}}$ be defined in the random permutation model by $\mathsf{H}_{\mathsf{ccr}} : x \mapsto \mathcal{O}_\pi(\sigma(x)) \oplus \sigma(x)$, where σ is a linear orthomorphism. Then, for any PPT adversary $\mathcal{A}^{\mathsf{H}_1, \mathcal{O}_\pi}$ making $|Q_1|$ queries to H_1 and $|Q_\pi|$ queries to $\mathcal{O}_\pi$, there exists an extractor Ext such that*

$$\mathsf{Adv}^{\mathit{ExtBinding}}_{\mathcal{A}, \mathsf{VC}_{\mathsf{MT}}}(\lambda) \le \frac{(1+\tau)|Q_1|^2}{2^{2\lambda}} + \frac{|Q_\pi|(|Q_\pi|-1)}{2^{2\lambda+1}}$$

Theorem 4. We first note that Ext can invert $\mathsf{H}_{\mathsf{ccr}}$ using the list Q_π of queries to $\mathcal{O}_\pi$ in the same way as in the proof of Theorem 2.

Extraction. The extractor Ext must extract in three steps:

1. Using Q_1, Ext finds the pre-image $\mathsf{iv} \| \mathsf{com}^0 \| \ldots \| \mathsf{com}^{\tau-1}$ of h_{com}; if it does not exist or is multiply-defined, then Ext outputs $\perp$ for all messages.

   As for Theorem 2, the assumption that $\mathcal{A}$ wins implies at least one pre-image exists and the probability that a collision for H_1 is found is bound by $|Q_1|^2/2^{2\lambda}$ by Lemma 1.

2. For each $i \in [\tau]$, Ext finds the pre-image $\mathsf{iv} \| \mathsf{com}^i_0 \| \ldots \| \mathsf{com}^i_{N-1}$ of com^i; if it does not exist, or is multiply-defined, then Ext outputs $\perp$ for all messages from the i-th instance of $\mathsf{VC}_{\mathsf{HT}}$.

   As for Theorem 2, the probability that this happens for each i is bound by $|Q_1|^2/2^{2\lambda}$ by Lemma 1 and by the assumption that $\mathcal{A}$ wins. Taking the union bound implies that the probability of Ext not finding pre-images for each com^i is bound by $\tau|Q_1|^2/2^{2\lambda}$.

3. Finally, for each com^i_j for $i \in [\tau], j \in [N]$, Ext uses Q_π to extract the leaf X^i_j; if it exists and is unique, it sets $\mathsf{sd}^i_j = \mathsf{H}_{\mathsf{ccr}}(X^i_j)$, otherwise it sets $\mathsf{sd}^i_j = \perp$.

   By the same argument as for Theorem 2, the probability that a collision is found for $\mathsf{H}_{\mathsf{ccr}}$ in the random permutation model is bound by Lemma 2.

Since, as before, the only way for $\mathcal{A}$ to win is for Ext to fail in one of the three types of extraction outlined above, we obtain the final advantage statement. □

6 Improving Security

Our schemes, $\mathsf{VC_{HT}}$ and $\mathsf{VC_{MT}}$, are constructed using a CCR hash function $\mathsf{H_{ccr}}$ in random permutation model. In this section, we investigate the efficiency and security level of our construction for the multi-hiding advantage of $\mathsf{Adv}^{\text{AdpMulHiding}}_{\mathcal{A},\mathsf{VC_{MT}}}(\lambda)$ by using different kinds of CCR hash functions and their instantiations. We show that our $\mathsf{VC_{MT}}$ can achieve a tighter security while the efficiency is still comparable with PPRF protocols used in current MPCitH and VOLEitH schemes.

6.1 VC from CCR in Random Permutation Model

In our construction of $\mathsf{VC_{HT}}$ (Figure 3), recall that the CCR hash function $\mathsf{H_{ccr}} : \{0,1\}^\lambda \to \{0,1\}^\lambda$ is instantiated from a random permutation π and a linear orthomorphism σ as $\mathsf{H_{ccr}}(x) := \pi(\sigma(x)) + \sigma(x)$. Guo et al. [GKWY20] used the H-coefficient technique [PS04, CS14] to show that $\mathsf{H_{ccr}}$ is a $(t, q, \rho, \epsilon_{\mathsf{ccr}})$-circular correlation robust where

$$\epsilon_{\mathsf{ccr}} = \frac{2tq}{2^\rho} + \frac{q^2}{2^{\lambda+1}}.$$

Because we use block-cipher (AES) to implement the permutation π, we can assume $\rho = \lambda$, the values t, q are the number of queries which $\mathcal{A}$ makes to the random permutation oracle $\mathcal{O}_\pi$ and $\mathcal{O}_{\mathsf{ccr}}$, respectively. Recall that the advantage in multi-hiding game is

$$\mathsf{Adv}^{\text{AdpHiding}}_{\mathcal{A},\mathsf{VC_{HT}}}(\lambda) \le \frac{|Q_1|}{2^{2\lambda}} + \mathsf{Adv}^{\text{PRG}}_{\mathcal{B}_{\mathsf{PRG}},\mathsf{PRG}}(\lambda) + \epsilon_{\mathsf{ccr}},$$

for $N = 2^{16}$ (this is parameters for hybercube MPCitH protocols [AGH⁺23]) and the security level λ (meaning the random permutation domain is $\{0,1\}^\lambda$), adversary $\mathcal{A}$ needs to make $2^{\lambda-16}$ queries to the permutation π to completely break our $\mathsf{VC_{HT}}$.

Using techniques in the CCR hash function literature [GKW⁺20, GKWY20, GYW⁺23, CT21], we show that our security can be strengthened by using *tweakable*-CCR (TCCR) and *multi-instance* tweakable-CCR (MiTCCR) in our all-but-τ construction $\mathsf{VC_{MT}}$.

6.2 VC from TCCR in Random Permutation Model

To make the use of $\mathsf{H_{tccr}}$ function which is a (t, q, ρ, ϵ)-TCCR where the oracle is defined as $\mathcal{O}^{\mathsf{tccr}}_{\mathsf{H},\Delta}(x, i, b) := \mathsf{H}(x \oplus \Delta, i) \oplus b \cdot \Delta$ and has a similar security definition as CCR. We refer to [GKWY20] for the exact definition. Our $\mathsf{VC_{MT}}$ construction (Figure 5) can be modified to use one global Δ for all instances. The tweak i in TCCR is defined to be the index $i \in [\tau]$ of the instance. That is, the tweak i is the same within each tree but varies between trees in our construction. We discuss two instances of TCCR.

1. Guo et al. [GKWY20] defined a TCCR hash function $\mathsf{H_{tccr}}(x, i) := \pi(\pi(x) \oplus i) \oplus \pi(x)$ where $\pi : \{0,1\}^\lambda \to \{0,1\}^\lambda$ is modelled as a random permutation and the adversary's advantage in the TCCR game is given by

$$\epsilon = \frac{4q(t+q)}{2^\lambda} + \frac{5q^2}{2^{\lambda+1}} + \frac{tq}{2^\rho} + \frac{q}{2^\lambda}.$$

   Note that one evaluation of H can be computed using only 2 calls to π and the running time of TCCR instantiation using fixed-key AES is still efficient (28x faster than SHA-256) [GKWY20].

2. Chen and Tessaro [CT21] introduced two new TCCR (one-call and two-call to permutation). While the tweakable one-call construction matches the security of the most secure two-call construction of [GKWY20]; the new two-call construction, defined as $\mathsf{H_{tccr}}(m, i) := \pi_2(\pi_1(\sigma(m)) \oplus i) \oplus \sigma(m)$, has better security of $\epsilon = O((\sqrt{q} \cdot t + q^2)/2^\lambda)$.

6.3 VC from MiTCCR in Ideal Cipher Model

MiTCCR hash function $\mathsf{H}^E_{\mathsf{mtccr}}$ is defined using the oracle $\mathcal{O}^{\mathsf{tccr}}_{\mathsf{H},\Delta}(x,i,b) := \mathsf{H}(x \oplus \Delta, i) \oplus b \cdot \Delta$, we refer to [GKWY20] for the definition details. It is more secure than TCCR or CCR because the adversary $\mathcal{A}$ is given access to multiple independently keyed functions rather than just one, and the concrete security bound depends on the maximum number of times μ that the adversary $\mathcal{A}$ repeats any particular tweak.

The function $\mathsf{H}^E_{\mathsf{mtccr}}$ ($(p,q,u,\mu,\rho,\epsilon)$-MiTCCR) is defined by $\mathsf{H}^E_{\mathsf{mtccr}}(x,i) := E(i,\sigma(x)) \oplus \sigma(x)$ under an ideal cipher $E : \{0,1\}^\lambda \times \{0,1\}^\lambda \to \{0,1\}^\lambda$. The distinguishing advantage

$$\epsilon = \frac{2\mu p}{2^\lambda} + \frac{(\mu-1)q}{2^\lambda}$$

is significantly smaller than the advantages in TCCR and CCR, and it is not proportional to $q \cdot p$. When using $\mathsf{H}^E_{\mathsf{mtccr}}$, our $\mathsf{VC}_{\mathsf{MT}}(N,\tau)$ construction is the same except that the i-th tree is built from a different random permutation $\pi_i \in E$ with parameters $q = N \cdot \tau$, μ equal to the number of queries to the forgery game, and tweak domain $\mathcal{T} = [\tau]$.

7 Application and Evaluation

7.1 Identification Schemes Against Malicious Verifier

All signature schemes following the MPCitH paradigm have an underlying 5-round honest-verifier zero-knowledge proof based on a PPRF where the security of the GGM tree for the PPRF is maintained only when the prover knows the random challenge beforehand (non-adaptive security). Therefore, when using our VC in the 5-round ZK protocol we directly obtain a 5-round ZK protocol against *malicious* verifier, which can be used for an identification scheme.

7.2 Faster MPCitH-based Signatures

Our VC works in the random permutation model which can be efficiently instantiated by fixed-key AES. For simplicity, we compare with existing MPCitH signature schemes and estimate our performance at the 128-bit security level.

For one instantiation of our VC of size $N = 2^d$, we require N AES calls for the final layer and $N/2$ AES calls for the tree expansion while other works [BDK+21, AGH+23, FJR22] uses N AES calls for tree expansion and N SHAKE calls for the last layer to expand each leaf into a seed and a commitment. Because of hardware support, using AES instead of a hash function like SHAKE is up to 50× faster. Although it is possible to use a faster hash function such as BLAKE3, submissions to the NIST standardization projects prefer to use a standardized primitive such as SHA3 and its variants. Moreover, to ensure a low soundness error and maintain the security of the signature scheme, all schemes need to perform τ parallel repetitions. Although prior work does not specifically use VC in their description, we claim VC is the correct abstraction since the commitment property is still needed as discussed in Section 1.2.

We benchmarked the hypercube variant of the optimized SDitH implementation called "SDitH-gf256-L1-hyp" [MFG+23], which is in the first round of the additional signatures standardization effort by NIST.[1] The offline, input independent, phase took 2/3 of the time in our experiments. It primarily consists of expanding the the GGM tree, hashing the leaves to create commitments and share aggregation. In other words, the GGM tree expansion and hashing/commitment takes approximately 2/3 of the total signing time. Using our optimization, we estimate the signing time can be reduced by a third since our construction halves the number of calls to SHA3, which is used to create commitments.

[1] https://csrc.nist.gov/Projects/pqc-dig-sig/

7.2.1 Security

Since MPCitH protocols such as [AGH+23] do not use all-but-one vector commitment in a blackbox way, we need to argue the resulting security bound when using our VC. Specifically, we show that if we replace the PRG tree and the commitment defined by H_0 in the hybercube protocol by our VC (the commitment is defined by concatenating of two $\mathsf{H}_{\mathsf{ccr}}$ calls) , the security remains the same except that

- $q_s \cdot \tau \cdot \epsilon_{\mathsf{PRG}}$ is replaced by $\mathsf{Adv}^{\mathrm{AdpHiding}}_{\mathcal{A},\mathsf{VC}_{\mathsf{HT}}}(\lambda)$, the advantage of the hiding game. In particular, this term comes from the HVZK simulator (replacing one leaf in each of τ trees by random values and programming it in the random oracle) when analysing the ZK property of the underlying protocol.
- We separately compute all terms related to H_0 since the probability of collision of commitment defined by H_0 is computed using $\mathsf{Adv}^{\mathrm{ExtBinding}}_{\mathcal{A},\mathsf{VC}_{\mathsf{HT}}}(\lambda)$ of VC.

Note that since our VC is computationally binding, the soundness error is the same so this lead to the $\mathcal{A}^{\mathsf{EUF-KO}}_{\mathcal{A}}$ is the same as [AGH+23]. The security proof for using our VC is stated below.

Theorem 5 (Security of Hypercube based on VC)**.** *Consider the* VC *in Section 4, let it be* $(t, \mathsf{Adv}^{AdpHiding}_{\mathcal{A},\mathsf{VC}_{\mathsf{HT}}}(\lambda), \mathsf{Adv}^{ExtBinding}_{\mathcal{A},\mathsf{VC}_{\mathsf{HT}}}(\lambda))$*-secure and let any adversary running in time* t *have advantage at most* ϵ_{SD} *against the underlying syndrome decoding problem. Let* $\mathsf{H}_{\mathsf{ccr}}$ *be* $(t, q_S \cdot \tau, \rho, \epsilon_{\mathsf{ccr}})$*-circular collision resistant and defined in the random permutation model;* $\mathsf{H}_1, \mathsf{H}_2, \mathsf{H}_3, \mathsf{H}_4$ *be random oracles with output length* 2λ *bits. Then chosen-message adversary against the signature scheme depicted in [AGH+23] after replacing PRG tree and the commitment by* VC *running in time* t*, making* q_0 *queries to the random permutation and* q_1, q_2, q_3, q_4, q_S *queries to the random oracles, and signature scheme respectively, succeeds in outputting a valid forgery with probability*

$$\Pr(forge) = \mathcal{A}^{\mathsf{EUF-CMA}}_{\mathcal{A}} \leq \mathcal{A}^{\mathsf{EUF-KO}}_{\mathcal{A}} + (q_0 + \tau \cdot N^D) \cdot \mathsf{Adv}^{ExtBinding}_{\mathcal{A},\mathsf{VC}_{\mathsf{HT}}}(\lambda) + \frac{2 \cdot (q + \tau \cdot D \cdot q_s)^2}{2 \cdot 2^\lambda} + \frac{q_s \cdot (q_s + q_0 + 4q)}{2^\lambda} + \mathsf{Adv}^{AdpHiding}_{\mathcal{A},\mathsf{VC}_{\mathsf{HT}}}(\lambda)$$

where $q = \max\{q_1, q_2, q_3, q_4\}$*,* D *is the dimension of the hypercube,* N^D *is the number of secret shares,* $\mathcal{A}^{\mathsf{EUF-CMA}}_{\mathcal{A}}$ *and* $\mathcal{A}^{\mathsf{EUF-KO}}_{\mathcal{A}}$ *are existential unforgeability against key-only attack and against chosen-message attacks respectively,* $\mathcal{A}^{\mathsf{EUF-KO}}_{\mathcal{A}}$ *is computed as in [AGH+23] from the soundness.*

7.3 Faster VOLE-in-the-Head Signatures

FAEST is the first VOLE-in-the-Head signature scheme[2], and is also a first round candidate for the additional signatures standardization project by NIST. In FAEST, the output of an all-but-τ VC scheme is converted to VOLE correlations and used by the QuickSilver zero-knowledge proof system [YSWW21] to prove an execution of the AES algorithm.

Here, the multi-tree version of the VC scheme is crucial. For the soundness of the proof, the VOLE correlation must have a statistical security of $O(2^{-\lambda})$, however this would require a single-tree construction to produce $N = O(2^\lambda)$ leaves, which is computationally unfeasable. Instead, FAEST is designed such that the signer produces τ trees, each with $O(2^{\lambda/\tau})$ leaves, producing τ independent VOLE instances each with $O(2^{-\lambda/\tau})$ security. By de-randomising the last $\tau - 1$ VOLE instances to align with the first, they can then be combined into a single instance offering the required $O(2^{-\lambda})$ security.

Below we describe two experiments; both were performed on Rocky Linux 8.8 with kernel version 4.18.0, running on AMD Ryzen 7 PRO 2700 @ 3.2 GHz. The implementations were built using GCC 11.2. The results for both experiments are averaged over 100 executions.

[2]https://faest.info/

Table 1: Performance comparison between our half-tree variant and the standard FAEST signature scheme using the reference implementation.

Variant	Sign (μs)	Speedup	Verify (μs)	Speedup	Sig size (B)
FAEST-128S	30,822	1.48	30,743	1.47	5,006
FAEST-128F	5,678	1.27	5,403	1.28	6,336
FAEST-192S	92,458	1.26	87,156	1.32	12,752
FAEST-192F	16,720	1.14	15,704	1.15	16,800
FAEST-256S	130,480	1.22	124,629	1.25	22,116
FAEST-256F	29,046	1.11	26,968	1.13	28,416
FAEST-EM-128S	27,776	1.57	27,932	1.56	4,566
FAEST-EM-128F	4,757	1.35	4,729	1.34	5,696
FAEST-EM-192S	77,684	1.31	74,913	1.34	10,832
FAEST-EM-192F	12,522	1.19	12,427	1.19	13,920
FAEST-EM-256S	120,245	1.24	118,054	1.25	20,972
FAEST-EM-256F	25,168	1.13	24,722	1.14	26,752

7.3.1 Reference Implementation

We modified the reference implementation of FAEST[3] to use our half-tree construction; our open-source code can be found on GitHub.[4] Specifically, for $\lambda = 128$, the π permutation used in $\mathsf{H}_{\mathsf{ccr}}$ is instantiated using fixed-key AES. For $\lambda \in \{192, 256\}$, however, we resort to using Rijndael-$\{192, 256\}$ to avoid the 128-bit AES block size. Just like FAEST, our implementation uses eXtended Keccak Code Package (XKCP) for SHA3 and OpenSSL libraries for AES. Note that XKCP and OpenSSL are already optimized for the x86 architecture with the AVX2 and AES-NI instruction set extensions. Unfortunately, OpenSSL does not have a Rijndael implementation so we extended the FAEST reference implementation to support it. The experimental results are shown in Table 1. Our best results are for $\lambda = 128$ since this version uses the AES implementation from OpenSSL, which is highly optimized.

7.3.2 Optimized Implementation

An optimized implementation of FAEST[5] exists which also uses the the AVX2 and AES-NI instruction set extensions. Although the two implementations are functionally identical, the optimized implementation does not follow the specification and thus achieves better efficiency. We do not have a full half-tree implementation based on the optimized FAEST implementation but to estimate our performance improvements, we tweaked it to use fixed-key Rijndael-$\{128, 192, 256\}$ for constructing the internal nodes and the leaves. The performance results, shown in Table 2, are what we believe to be very close to the results of a complete optimized implementation since the only missing ingredient is the orthomorphism which has very little impact to the performance since each only consumes 1.66 cycle on the CPU using SSE2 [GKWY20]. The performance gain is lower when $\lambda \in \{192, 256\}$ since producing one block of Rijndael-$\{192, 256\}$ is slower than producing three or four blocks of AES-$\{192, 256\}$ (which is used for tree expansion in the original FAEST algorithm) as Rijndael-$\{192, 256\}$ cannot fully utilize the AES-NI instruction set.

[3] https://github.com/faest-sign/faest-ref
[4] https://github.com/KULeuven-COSIC/faest-ref-vc
[5] https://github.com/faest-sign/faest-avx

Table 2: Performance comparison between our half-tree variant and the standard FAEST signature scheme using the optimized implementation.

Variant	Sign (μs)	Speedup	Verify (μs)	Speedup
FAEST-128S	4,587	3.33	4,620	3.21
FAEST-128F	483	3.35	486	3.38
FAEST-192S	18,652	1.76	18,873	1.76
FAEST-192F	2,010	1.67	1,964	1.71
FAEST-256S	26,195	1.63	26,253	1.62
FAEST-256F	3,161	1.56	3,082	1.60
FAEST-EM-128S	4,424	3.33	4,456	3.31
FAEST-EM-128F	468	3.46	459	3.46
FAEST-EM-192S	17,861	1.80	17,997	1.80
FAEST-EM-192F	1,848	1.73	1,825	1.75
FAEST-EM-256S	26,273	1.60	26,300	1.59
FAEST-EM-256F	3,048	1.60	3,031	1.60

Acknowledgement

We would like to thank Geoffroy Couteau (IRIF) and Tim Beyne (COSIC, KU Leuven) for their advice as well as the anonymous reviewers for their insightful comments.

This work was supported by CyberSecurity Research Flanders with reference number VR20192203. Dung Bui 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. Part of this work was done while Dung Bui was visiting COSIC, KU Leuven. Kelong Cong is supported by the Horizon 2020 research and innovation programme under grant agreement No 101018342 SOTERIA. Part of this work was done while Kelong Cong was at COSIC, KU Leuven. Cyprien Delpech de Saint Guilhem is a Junior FWO Postdoctoral Fellow under project 1266123N.

References

[ABC+23] Nicolas Aragon, Loïc Bidoux, Jesús-Javier Chi-Domínguez, Thibauld Feneuil, Philippe Gaborit, Romaric Neveu, and Matthieu Rivain. MIRA: a digital signature scheme based on the minrank problem and the mpc-in-the-head paradigm. *CoRR*, abs/2307.08575, 2023. URL: https://doi.org/10.48550/arXiv.2307.08575, arXiv:2307.08575, doi:10.48550/ARXIV.2307.08575.

[AGH+23] Carlos Aguilar Melchor, Nicolas Gama, James Howe, Andreas Hülsing, David Joseph, and Dongze Yue. The return of the SDitH. In Hazay and Stam [HS23], pages 564–596. doi:10.1007/978-3-031-30589-4_20.

[BBD+23] Carsten Baum, Lennart Braun, Cyprien Delpech de Saint Guilhem, Michael Klooß, Emmanuela Orsini, Lawrence Roy, and Peter Scholl. Publicly Verifiable Zero-Knowledge and Post-Quantum Signatures from VOLE-in-the-Head. In Helena Handschuh and Anna Lysyanskaya, editors, *Advances in Cryptology - CRYPTO 2023 - 43rd Annual International Cryptology Conference, CRYPTO 2023, Santa Barbara, CA, USA, August 20-24, 2023, Proceedings, Part V*, volume 14085 of *Lecture Notes in Computer Science*, pages 581–615. Springer, 2023. doi:10.1007/978-3-031-38554-4_19.

[BCG+19] Elette Boyle, Geoffroy Couteau, Niv Gilboa, Yuval Ishai, Lisa Kohl, and Peter Scholl. Efficient pseudorandom correlation generators: Silent OT extension and more. In Alexandra Boldyreva and Daniele Micciancio, editors, *CRYPTO 2019, Part III*, volume 11694 of *LNCS*, pages 489–518. Springer, Heidelberg, August 2019. doi:10.1007/978-3-030-26954-8_16.

[BDK+21] Carsten Baum, Cyprien Delpech de Saint Guilhem, Daniel Kales, Emmanuela Orsini, Peter Scholl, and Greg Zaverucha. Banquet: Short and fast signatures from AES. In Juan Garay, editor, *PKC 2021, Part I*, volume 12710 of *LNCS*, pages 266–297. Springer, Heidelberg, May 2021. doi:10.1007/978-3-030-75245-3_11.

[BHKR13] Mihir Bellare, Viet Tung Hoang, Sriram Keelveedhi, and Phillip Rogaway. Efficient garbling from a fixed-key blockcipher. In *2013 IEEE Symposium on Security and Privacy*, pages 478–492. IEEE Computer Society Press, May 2013. doi:10.1109/SP.2013.39.

[BMRS21] Carsten Baum, Alex J. Malozemoff, Marc B. Rosen, and Peter Scholl. Mac'n'cheese: Zero-knowledge proofs for boolean and arithmetic circuits with nested disjunctions. In Tal Malkin and Chris Peikert, editors, *CRYPTO 2021, Part IV*, volume 12828 of *LNCS*, pages 92–122, Virtual Event, August 2021. Springer, Heidelberg. doi:10.1007/978-3-030-84259-8_4.

[BN20] Carsten Baum and Ariel Nof. Concretely-efficient zero-knowledge arguments for arithmetic circuits and their application to lattice-based cryptography. In Aggelos Kiayias, Markulf Kohlweiss, Petros Wallden, and Vassilis Zikas, editors, *PKC 2020, Part I*, volume 12110 of *LNCS*, pages 495–526. Springer, Heidelberg, May 2020. doi:10.1007/978-3-030-45374-9_17.

[CCJ23] Eliana Carozza, Geoffroy Couteau, and Antoine Joux. Short signatures from regular syndrome decoding in the head. In Hazay and Stam [HS23], pages 532–563. doi:10.1007/978-3-031-30589-4_19.

[CLY+24] Hongrui Cui, Hanlin Liu, Di Yan, Kang Yang, Yu Yu, and Kaiyi 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. URL: https://eprint.iacr.org/2024/040.

[CPS08] Jean-Sébastien Coron, Jacques Patarin, and Yannick Seurin. The random oracle model and the ideal cipher model are equivalent. In David Wagner, editor, *CRYPTO 2008*, volume 5157 of *LNCS*, pages 1–20. Springer, Heidelberg, August 2008. doi:10.1007/978-3-540-85174-5_1.

[CS14] Shan Chen and John P. Steinberger. Tight security bounds for key-alternating ciphers. In Phong Q. Nguyen and Elisabeth Oswald, editors, *EUROCRYPT 2014*, volume 8441 of *LNCS*, pages 327–350. Springer, Heidelberg, May 2014. doi:10.1007/978-3-642-55220-5_19.

[CT21] Yu Long Chen and Stefano Tessaro. Better security-efficiency trade-offs in permutation-based two-party computation. In Mehdi Tibouchi and Huaxiong Wang, editors, *ASIACRYPT 2021, Part II*, volume 13091 of *LNCS*, pages 275–304. Springer, Heidelberg, December 2021. doi:10.1007/978-3-030-92075-3_10.

[DOT21] Cyprien Delpech de Saint Guilhem, Emmanuela Orsini, and Titouan Tanguy. Limbo: Efficient zero-knowledge MPCitH-based arguments. In Vigna and Shi [VS21], pages 3022–3036. doi:10.1145/3460120.3484595.

[FJR22] Thibauld Feneuil, Antoine Joux, and Matthieu Rivain. Syndrome decoding in the head: Shorter signatures from zero-knowledge proofs. In Yevgeniy Dodis and Thomas Shrimpton, editors, *CRYPTO 2022, Part II*, volume 13508 of *LNCS*, pages 541–572. Springer, Heidelberg, August 2022. doi:10.1007/978-3-031-15979-4_19.

[FS87] Amos Fiat and Adi Shamir. How to prove yourself: Practical solutions to identification and signature problems. In Andrew M. Odlyzko, editor, *CRYPTO'86*, volume 263 of *LNCS*, pages 186–194. Springer, Heidelberg, August 1987. doi:10.1007/3-540-47721-7_12.

[GGM84] Oded Goldreich, Shafi Goldwasser, and Silvio Micali. How to construct random functions (extended abstract). In *25th FOCS*, pages 464–479. IEEE Computer Society Press, October 1984. doi:10.1109/SFCS.1984.715949.

[GKW+20] Chun Guo, Jonathan Katz, Xiao Wang, Chenkai Weng, and Yu Yu. Better concrete security for half-gates garbling (in the multi-instance setting). In Daniele Micciancio and Thomas Ristenpart, editors, *CRYPTO 2020, Part II*, volume 12171 of *LNCS*, pages 793–822. Springer, Heidelberg, August 2020. doi:10.1007/978-3-030-56880-1_28.

[GKWY20] Chun Guo, Jonathan Katz, Xiao Wang, and Yu Yu. Efficient and secure multiparty computation from fixed-key block ciphers. In *2020 IEEE Symposium on Security and Privacy*, pages 825–841. IEEE Computer Society Press, May 2020. doi:10.1109/SP40000.2020.00016.

[GYW+23] Xiaojie Guo, Kang Yang, Xiao Wang, Wenhao Zhang, Xiang Xie, Jiang Zhang, and Zheli Liu. Half-tree: Halving the cost of tree expansion in COT and DPF. In Carmit Hazay and Martijn Stam, editors, *EUROCRYPT 2023, Part I*, volume 14004 of *LNCS*, pages 330–362. Springer, Heidelberg, April 2023. doi:10.1007/978-3-031-30545-0_12.

[HS23] Carmit Hazay and Martijn Stam, editors. *EUROCRYPT 2023, Part V*, volume 14008 of *LNCS*. Springer, Heidelberg, April 2023.

[IKOS07] Yuval Ishai, Eyal Kushilevitz, Rafail Ostrovsky, and Amit Sahai. Zero-knowledge from secure multiparty computation. In David S. Johnson and Uriel Feige, editors, *39th ACM STOC*, pages 21–30. ACM Press, June 2007. doi:10.1145/1250790.1250794.

[KKW18] Jonathan Katz, Vladimir Kolesnikov, and Xiao Wang. Improved non-interactive zero knowledge with applications to post-quantum signatures. In David Lie, Mohammad Mannan, Michael Backes, and XiaoFeng Wang, editors, *ACM CCS 2018*, pages 525–537. ACM Press, October 2018. doi:10.1145/3243734.3243805.

[MFG$^+$23] Carlos Aguilar Melchor, Thibauld Feneuil, Nicolas Gama, Shay Gueron, James Howe, David Joseph, Antoine Joux, Edoardo Persichetti, Tovohery H. Randrianarisoa, Matthieu Rivain, and Dongze Yue. The syndrome decoding in the head (sd-in-the-head) signature scheme. https://www.sdith.org/docs/sdith-v1.1.pdf, 2023. Accessed: 2023-11-03.

[PS04] Dan Page and Martijn Stam. On XTR and side-channel analysis. In Helena Handschuh and Anwar Hasan, editors, *SAC 2004*, volume 3357 of *LNCS*, pages 54–68. Springer, Heidelberg, August 2004. doi:10.1007/978-3-540-30564-4_4.

[VS21] Giovanni Vigna and Elaine Shi, editors. *ACM CCS 2021*. ACM Press, November 2021.

[Win84] Robert S. Winternitz. A Secure One-Way Hash Function Built from DES. In *1984 IEEE Symposium on Security and Privacy*, pages 88–88, Oakland, CA, USA, April 1984. IEEE. URL: http://ieeexplore.ieee.org/document/6234813/, doi:10.1109/SP.1984.10027.

[WYKW21] Chenkai Weng, Kang Yang, Jonathan Katz, and Xiao 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. doi:10.1109/SP40001.2021.00056.

[YSWW21] Kang Yang, Pratik Sarkar, Chenkai Weng, and Xiao Wang. QuickSilver: Efficient and affordable zero-knowledge proofs for circuits and polynomials over any field. In Vigna and Shi [VS21], pages 2986–3001. doi:10.1145/3460120.3484556.