Quantum-Safe Public Key Blinding from MPC-in-the-Head Signature Schemes

Sathvika Balumuri[1], Edward Eaton[1] and Philippe Lamontagne[1,2]✉

[1] National Research Council Canada
[2] Université de Montréal, Canada
Philippe.Lamontagne2@cnrc-nrc.gc.ca

Abstract. Key blinding produces pseudonymous digital identities by rerandomizing public keys of a digital signature scheme. It provides privacy in decentralized networks. Current key blinding schemes are based on the discrete log assumption. Eaton, Stebila and Stracovsky (LATINCRYPT 2021) proposed the first post-quantum key blinding schemes from lattice assumptions. However, the large public keys and lack of QROM security means they are not ready to replace existing solutions.
We present a general framework to build post-quantum signature schemes with key blinding based on the MPC-in-the-Head paradigm. This results in schemes that rely on well-studied symmetric cryptographic primitives and admit short public keys. We prove generic security results in the quantum random oracle model (QROM).
We instantiate our framework with the recent AES-based Helium signature scheme (Kales and Zaverucha, 2022) to obtain an efficient post-quantum key blinding scheme with small keys. Both Helium and the aforementioned lattice-based key blinding schemes were only proven secure in the ROM. This makes our results the first QROM proof of Helium and the first fully quantum-safe public key blinding scheme.

Keywords: Key Blinding · Anonymity · MPCitH · Quantum-Safe · QROM

1 Introduction

Decentralized services such as Tor's hidden services [24] or the GNU Name System [25] use a concept known as *public key blinding*[3] to provide privacy against the intermediate nodes that take part in the name resolution when a client attempts to connect to a service. This mechanism protects the anonymity of the server and circumvents censorship. In a digital signature scheme with public key blinding, the public key pk can be rerandomized using a seed τ into a blinded key bpk, such that knowledge of τ and bpk does not allow one to compute pk; and such that only the secret key holder can produce valid signature for bpk.

[3] Not to be confused with blind signatures, which allows signing a message while being oblivious to its content.

Existing key blinding schemes are based on elliptic curve cryptography. Post-quantum schemes have been proposed [17] from lattice-based assumptions. The challenge in introducing new key blinding schemes is that their security does not directly reduce to the unforgeability of the underlying signature scheme and needs to be proven from scratch. In the case of post-quantum security, schemes that employ the random oracle methodology should be proven secure in the quantum random oracle model (QROM). Another challenge is that lattice-based post-quantum signatures schemes typically have large public keys, on the order of kilobytes for the Dilithium scheme selected for standardization by NIST. In the context of Tor's rendezvous spec, this is problematic since public keys represent identities that have to be handled manually by users.

Our work addresses all of these issues and more. We propose a general framework for constructing post-quantum key blinding schemes based on symmetric-key assumptions, with provably secure in the QROM and extremely short public keys.

1.1 Our Results

Our starting point is the idea of adding key blinding to the Picnic signature scheme which was sketched in [17]. We generalize it to any scheme where signatures consist of a message-dependent non-interactive zero-knowledge proof of knowledge (NIZKPoK) of the preimage of a one-way function constructed via Fiat-Shamir heuristic and the MPC-in-the-Head (MPCitH) paradigm[4]. We describe which properties those schemes and the underlying pseudorandom function must satisfy in order to yield an unforgeable and unlinkable key blinding scheme. The advantage of our modular approach is that signature with key blinding schemes can benefit from future improvements for signature schemes based on NIZKPoK[5]. Our proofs are in the quantum-accessible random oracle model (QROM) where the adversary may evaluate the random oracle on an arbitrary quantum superposition of inputs. Since previous works only proved security against classical random oracle queries, this makes our construction the first fully quantum-safe signature scheme with key blinding.

We use the Helium scheme of [21] to demonstrate our techniques. Helium offers several efficiency improvements over previous MPCitH-based schemes and it is an appealing candidate since it is solely based on well-studied symmetric primitives such as the AES block cipher and the SHAKE extendable output function and boasts small proof sizes for the AES circuit. We apply our framework to the Helium signature scheme to get a signature scheme with key blinding we call blHelium. To prove that our scheme is secure in the QROM, we show that the proof system underlying Helium is a post-quantum proof of knowledge in the

[4] Although the MPCitH paradigm can be used to prove any NP statement, throughout this paper, we use the term to refer to proofs of statements about symmetric key primitives.

[5] For example, our techniques could apply to the recent VOLE-in-the-Head paradigm [1] which offers significant improvement in proof size over MPCitH.

QROM. A direct corollary is that Helium is secure in the QROM, a question that was left open in [21].

The blinded public keys for our blHelium scheme are only 64 bytes for the version based on AES256, which makes it an ideal candidate for a post-quantum transition from ECDSA for the Tor and GNU networks and for other key blinding scheme that benefit from small keys.

We have implemented the AES128 version of our blHelium scheme as a fork of the Helium code. Even though this parameter choice is not post-quantum secure, it allows for a direct comparison with Helium to observe the overhead induced by key blinding. We observe that the blinded version is 2× to 3× slower and produces 2× to 3× larger signatures, which is to be expected based on the larger circuit size of the underlying one-way function.

1.2 Related Work

Public key blinding was first introduced as an anonymity protection feature of Tor's Rendezvous protocol [24]. Tor's key blinding scheme is based on discrete logarithm assumptions. Besides Tor, key blinding also has applications to the GNU Name System [25], private airdrop and rate-limited privacy pass [16]. Key blinding is just one of many public key rerandomization techniques. For an overview and comparison of signature schemes with key rerandomization, we refer the reader to [6].

The first post-quantum key blinding schemes were proposed in [17]. Their constructions are built from lattice-based post-quantum signature schemes and are only proven secure in the (classical) random oracle model. Their schemes also suffer from fairly long public keys (on the order of kilobytes). The general construction of key blinding for MPCitH schemes that we present was first sketched in the appendix of [17] for the Picnic signature scheme [7]. They briefly sketch a proof of unlinkability in the classical ROM, but provide no argument towards unforgeability.

Efforts have been made towards standardizing key blinding in an IETF technical specification draft [10]. The companion paper [16] provides security proofs for the scheme from the draft.

Multiparty computation in the head (MPCitH) is a technique for proving NP statements about arbitrary Boolean circuits introduced by [20]. This technique was refined by ZKBoo [18] and further improved in the paper that introduced Picnic [7], the first signature scheme built from the MPCitH framework to prove knowledge of a preimage of a one-way function. There have since been several improvements. Katz, Kolesnikov, and Wang [22] added a preprocessing phase to the MPC computation used in [7]. BBQ [8] is the first MPCitH signature scheme that instantiates the one-way function with the well studied AES block cipher. It mitigates the larger circuit size by avoiding private keys that lead to circuits which have the 0 byte as input to an s-box, allowing for an efficient computation of the nonlinear operation. Baum and Nof [3] introduces sacrificial multiplication triples (or beaver triples) to replace cut-and-choose checks, which leads to better soundness and less repetitions of the MPC protocol. The Banquet [2] signature

scheme achieves 50% smaller signatures by running an MPC protocol that verifies the correctness of the circuit computation, instead of computing the result itself. Dobraunig, Kales, Rechberger, Schofnegger, and Zaverucha [11] offer additional improvements to AES-based MPCitH signature schemes and proposes a scheme based on the *Rain* cipher optimized for MPCitH. Helium [21] lifts multiple small fields $\mathbb{F}$ elements into a larger field $\mathbb{K}$, such that multiplying the $\mathbb{F}$ elements component-wise can be realised by a single operation on $\mathbb{K}$, a technique that was also used in Limbo [9].

2 Preliminaries

We let $\lambda \in \mathbb{N}$ be our security parameter throughout the paper. For two functions f, g let $f \circ g$ denote the function $x \mapsto f(g(x))$. Basic security notions of sets of functions are defined in Appendix A.1.

2.1 Digital Signature Schemes with Public Key Blinding

The definitions of this section are reproduced from previous work on signature schemes with key blinding [17,16].

Definition 1 (Digital Signature with Key Blinding). *A digital signature scheme with key blinding scheme is a tuple of algorithms:*

- $\mathsf{KGen}(1^\lambda)$*: returns a private key* sk *and an* identity *public key* pk
- $\mathsf{BlindPK}(\mathsf{pk}, \tau)$*: takes as input the identity public key* pk *and a blinding parameter* τ *and produces a* blinded public key bpk_τ.
- $\mathsf{Sign}(\mathsf{sk}, \tau, m)$*: produces a signature* σ *for* m *that is valid for the blinded key* bpk_τ.
- $\mathsf{Verify}(\mathsf{bpk}, m, \sigma)$*: returns* 1 *if* σ *is a valid signature of* m *for blinded key* bpk *and* 0 *otherwise.*

The scheme is (perfectly) correct if for $(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{KGen}(1^\lambda)$*, then for all* m *and* τ*:*

$$\mathsf{Verify}(\mathsf{BlindPK}(\mathsf{pk}, \tau), m, \mathsf{Sign}(\mathsf{sk}, \tau, m)) = 1 \ .$$

The unforgeability of signature schemes with key blinding is similar to that of regular unforgeability with the difference that we give the adversary control over which blinded key it targets for its forgery. The adversary is also allowed access to a signature oracle[6] for an arbitrary (polynomial) number of blinded keys.

Definition 2 (Unforgeability – Chosen Message and Blinding Attack). *Let* $(\mathsf{KGen}, \mathsf{BlindPK}, \mathsf{Sign}, \mathsf{Verify})$ *be a key blinding signature scheme. The chosen message and blinding attack experiment* EUF-CMBA *is defined as the following game between a challenger and an adversary* $\mathcal{A}$*:*

[6] In [16], the adversary can request signatures with respect to the original (non-blinded) signature scheme. In the context of our framework, this is not possible since signatures for blinded keys are incompatible with the original scheme.

– *The challenger samples* $(\mathsf{pk}, \mathsf{sk}) \leftarrow\!\!\$\ \mathsf{KGen}(1^\lambda)$ *and sends* pk *to* $\mathcal{A}$.
– $\mathcal{A}$ *can query a signing oracle* Sig *on message* m *and blinding parameter* τ *to receive* $\sigma = \mathsf{Sign}(m, \mathsf{sk}, \tau)$.
– $\mathcal{A}$ *sends its output* (m^*, σ^*, τ^*) *to the challenger who computes* $\mathsf{bpk}^* = \mathsf{BlindPK}(\mathsf{pk}, \tau^*)$ *and outputs 1 if* $\mathsf{Verify}(\mathsf{bpk}^*, m^*, \sigma^*) = 1$ *and if* (m^*, τ^*) *was not previously queried to the signing oracle. Otherwise it outputs 0.*

The advantage of an adversary $\mathcal{A}$ *for the* EUF-CMBA *game is defined as the probability* $\mathsf{Adv}^{\text{EUF-CMBA}}_{\mathcal{A}}$ *that the challenger outputs* 1.

The notion of privacy provided by key blinding is that of unlinkability. A scheme is unlinkable if an adversary cannot tell if two blinded keys originate from the same identity public key or from different keys. In the unlinkability experiment, we also allow the adversary to request new blinded keys at will and to request signatures of arbitrary messages with respect to the blinded keys.

Definition 3 (Unlinkability – Chosen Message and Blinding Attack). *Let* $(\mathsf{KGen}, \mathsf{BlindPK}, \mathsf{Sign}, \mathsf{Verify})$ *be a key blinding signature scheme. The* unlinkability under chosen message and blinding attack UL-CMBA *experiment is defined as the following game:*

– *The challenger samples* $(\mathsf{pk}_0, \mathsf{sk}_0) \leftarrow\!\!\$\ \mathsf{KGen}(1^\lambda)$.
– $\mathcal{A}$ *can query a blinding oracle* bl*, which on input* τ *returns* $\mathsf{bpk} \leftarrow \mathsf{BlindPK}(\mathsf{pk}_0, \tau)$.
– $\mathcal{A}$ *can query a signing oracle* Sig*, which on a message* m *and a blinding parameter* τ *returns* $\sigma = \mathsf{Sign}(m, \mathsf{sk}_0, \tau)$.
– $\mathcal{A}$ *sends a blinding parameter* τ^* *to the challenger. The challenger aborts the experiment if* τ^* *was previously queried to the blinding oracle.*
– *The challenger picks a new key pair* $(\mathsf{pk}_1, \mathsf{sk}_1) \leftarrow \mathsf{KGen}(1^\lambda)$*, samples a bit* $b \leftarrow\!\!\$\ \{0, 1\}$ *and sends* $\mathsf{bpk}^*_b \leftarrow \mathsf{BlindPK}(\mathsf{pk}_b, \tau^*)$ *to* $\mathcal{A}$.
– $\mathcal{A}$ *again has access to the blinding and signing oracle, but now the oracles use the key pair* $(\mathsf{sk}_b, \mathsf{pk}_b)$ *if* $\tau = \tau^*$ *and use the pair* $(\mathsf{sk}_0, \mathsf{pk}_0)$ *if* $\tau \neq \tau^*$.
– $\mathcal{A}$ *outputs a guess* b' *and wins if* $b' = b$.

The advantage of an adversary $\mathcal{A}$ *for the experiment is defined as the probability* $\mathsf{Adv}^{\text{UL-CMBA}}_{\mathcal{A}} = \left|\Pr[b = b'] - \frac{1}{2}\right|$.

2.2 Quantum Random Oracle Model

In the quantum random oracle model (QROM), the adversary has quantum oracle access to a unitary $\mathcal{O}^H : |c\rangle|x\rangle|y\rangle \mapsto |c\rangle|x\rangle|y \oplus c \cdot H(x)\rangle$ that computes a random function H in superposition. While the QROM does not permit observing and reprogramming random oracle queries as easily as in the classical ROM, there are now powerful tools for proving security in the QROM, which we present in the appendix (Section A.2).

2.3 Non-Interactive Proof Systems in the QROM

Let $R \subseteq \mathcal{X} \times \mathcal{W}$ be a relation. A non-interactive proof system for R in the quantum random oracle model is a pair of oracle-aided algorithms $\Sigma = (\mathsf{P}^H, \mathsf{V}^H)$. The proof system is correct (or complete) if there is a negligible function $\kappa(\cdot)$ such that for every $(x, w) \in R$,

$$\Pr_H[\mathsf{V}^H(x, \pi) \mid \pi \leftarrow \mathsf{P}^H(x, w)] \geq 1 - \kappa(\lambda) \ . \tag{1}$$

Definition 4 (Post-Quantum Zero-Knowledge). *A non-interactive proof $\Sigma = (\mathsf{P}^H, \mathsf{V}^H)$ is post-quantum zero-knowledge (pqNIZK) if there is a simulator Sim and a function $\varepsilon_{\mathrm{zk}}$ such that for every QPT adversary $\tilde{\mathsf{V}}^H$ that makes at most q_H queries to H,*

$$\left|\Pr_H[\tilde{\mathsf{V}}^H(x, \pi) = 1 \mid \pi \leftarrow \mathsf{P}^H(x, w)] - \Pr_H[\tilde{\mathsf{V}}^{H_\Theta}(x, \pi) = 1 \mid (\pi, \Theta) \leftarrow \mathcal{S}^H(x)]\right| \leq \varepsilon_{\mathrm{zk}}(\lambda, q_H) \tag{2}$$

where $\Theta = \{(x_1, y_1), \ldots, (x_t, y_t)\}$ is a list of assignments and H_Θ satisfies $H_\Theta(x_i) = y_i$ for every $(x_i, y_i) \in \Theta$ and is equal to H otherwise, and where $\varepsilon_{\mathrm{zk}}(\lambda, q_H)$ is negligible in λ if q_H is polynomial in λ.

The soundness notion we will use is the definition of *online extractability* from [14]. We first present some notation. Let $\tilde{\mathsf{P}}$ be a dishonest prover that outputs an instance x, a proof π and some auxiliary (potentially quantum) information Z. Let $\mathsf{V}^H \leftrightarrow \tilde{\mathsf{P}}^H(1^\lambda)$ denote an execution of the proof, which we define as $(x, \pi, Z) \leftarrow \tilde{\mathsf{P}}^H(1^\lambda)$ followed by $v \leftarrow \mathsf{V}^H(x, \pi)$. The malicious prover may receive an additional input (e.g. a public key in the unforgeability game). For an interactive algorithm $\mathcal{E}$ (that we call the *online extractor*) which controls the interface to the random oracle, we let $\mathsf{V}^{\mathcal{E}} \leftrightarrow \tilde{\mathsf{P}}^{\mathcal{E}}(1^\lambda)$ denote the execution with the calls to H simulated by $\mathcal{E}$ and where $\mathcal{E}$ additionally outputs $w \in \mathcal{W}$. Let $[(x, \pi, v, Z)]_{\mathsf{V}^{\mathcal{O}} \leftrightarrow \tilde{\mathsf{P}}^{\mathcal{O}}(1^\lambda)}$ denote the distribution of the outputs of the execution with oracle $\mathcal{O} \in \{H, \mathcal{E}\}$ and let δ be the statistical distance.

Definition 5. *A non-interactive proof in the quantum random oracle model for a relation R is a* online extractable *against adaptive adversaries if there exists an online extractor $\mathcal{E}$, and functions $\varepsilon_{\mathrm{sim}}$ (the* simulation error*) and $\varepsilon_{\mathrm{ex}}$ (the* extraction error*), with the following properties. For any $\lambda \in \mathbb{N}$ and for any q–query dishonest prover $\tilde{\mathsf{P}}$,*

$$\delta\big([(x, \pi, v, Z)]_{\mathsf{V}^H \leftrightarrow \tilde{\mathsf{P}}^H(1^\lambda)}, [(x, \pi, v, Z)]_{\mathsf{V}^{\mathcal{E}} \leftrightarrow \tilde{\mathsf{P}}^{\mathcal{E}}(1^\lambda)}\big) \leq \varepsilon_{\mathrm{sim}}(\lambda, q)$$

and

$$\Pr\big[v = 1 \wedge (x, w) \notin R : (x, \pi, v, Z, w) \leftarrow \mathsf{V}^{\mathcal{E}} \leftrightarrow \tilde{\mathsf{P}}^{\mathcal{E}}(1^\lambda)\big] \leq \varepsilon_{\mathrm{ex}}(\lambda, q) \,.$$

Furthermore, the runtime of $\mathcal{E}$ is polynomial in $\lambda + q$, and $\varepsilon_{\mathrm{sim}}(\lambda, q)$ and $\varepsilon_{\mathrm{ex}}(\lambda, q)$ are negligible in λ whenever q is polynomial in λ.

Consider an execution of the proof system with honest verifier and a prover that has oracle access to the zero-knowledge simulator Sim. In this execution, verification is done with respect to the final state of the random oracle $\hat{H}$ has been reprogrammed at certain points from H by Sim. We let $(x, \pi, v, Z) \leftarrow V^{\hat{H}} \leftrightarrow \tilde{\mathsf{P}}^{H,\mathsf{Sim}}(1^\lambda)$ denote the outcome of this execution where $(x, \pi, Z) \leftarrow \tilde{\mathsf{P}}^{H,\mathsf{Sim}}$ and $v = \mathsf{V}^{\hat{H}}(x, \pi) \wedge x \notin Q_{\mathsf{Sim}}$ where Q_{Sim} is the list of queries made to the simulator.

Definition 6. *A non-interactive proof system in the QROM $\Pi = (\mathsf{P}^H, \mathsf{V}^H)$ is* conditionally simulation-sound with respect to a simulator Sim *if for all QPT provers $\tilde{\mathsf{P}}^{H,\mathsf{Sim}}$ with oracle access to H and Sim, there is a function $\varepsilon_{\mathrm{ss}}$ and an oracle algorithm $\mathcal{B}^H$ such that*

$$\delta\left([x, \pi, v, Z]_{V^{\hat{H}} \leftrightarrow \tilde{\mathsf{P}}^{H,\mathsf{Sim}}(1^\lambda)}, [x, \pi, v, Z]_{V^H \leftrightarrow \mathcal{B}^H(1^\lambda)}\right) \leq \varepsilon_{\mathrm{ss}}(\lambda, q_H, q_S, n) \quad (3)$$

where $\varepsilon_{\mathrm{ss}}(\lambda, q_H, q_S, n)$ is negligible in λ whenever q_H, q_S and n are polynomial in λ.

Definition 6 does not by itself imply any soundness notion. However, if a proof system is sound and satisfies Definition 6, then it is simulation-sound; similarly if it is (online) extractable, then it is simulation-sound (online) extractable.

On the Satisfiability of this Section's Definitions. The definitions from this section are somewhat strong, i.e. that extractability is online and simulation-sound. We have chosen to go with these stronger requirements because 1- they give tight bounds for reductions and 2- they are achievable using standard constructions. More precisely, the Fiat-Shamir transform applied to special-sound commit-and-open interactive proofs is online extractable [15,14]. Simulation-soundness and simulation-sound (extractability) of the Fiat-Shamir transform was shown by Unruh [27]; and it was observed in [12] that the argument also applies FS of multi-round proof systems.

3 Key Blinding for MPC-in-the-Head Signature Schemes

3.1 Blinding MPCitH Public Keys

Let $\mathcal{F} = \{f_k : \{0,1\}^\lambda \to \{0,1\}^\lambda\}_{k \in \{0,1\}^\lambda}$ be a family of pseudorandom functions. For $x \in \{0,1\}^\lambda$, we let $F_x : k \mapsto f_k(x)$. We consider signature schemes that are constructed as follows. Let $\Pi = (\mathsf{P}^H, \mathsf{V}^H)$ be a NIZKPoK for the relation $R = \{(y, k)) \mid y = F_x(k)\}$. It can be turned into a signature scheme as follows.

- Key generation: sample $x \in \{0,1\}^\lambda$ and $k \in \{0,1\}^\lambda$. Output $\mathsf{sk} := k$ and $\mathsf{pk} := (x, f_k(x))$.
- Signature: to sign a message m, use P on input $(\mathsf{pk}, \mathsf{sk})$ to produce a non-interactive zero-knowledge proof of knowledge of k such that $\mathsf{pk} = (x, f_k(x))$ that depends[7] on m.

[7] For example, if the proof system uses the Fiat-Shamir heuristic, m can be included in the random oracle queries that compute the verifier's challenge.

– Verification: run the verification protocol V on input pk.

For such protocols, we consider a generic blinding procedure which was first informally proposed in [17]. To blind a public key $\mathsf{pk} = (x, f_k(x))$ using a seed τ, one encrypts x a second time using a new key derived from pk and τ, for example using a cryptographic hash function G modelled as a random oracle. The new blinded public key is $\mathsf{bpk}_\tau = (x, f_{G(\tau,\mathsf{pk})}(\mathsf{pk}))$. Observe however that we want the same verification procedure for every blinded key (verification should depend only on the blinded public key and should not require knowledge of τ), so the value x cannot be itself encrypted. Moreover, unlinkability requires that we use the same input x for every public key, otherwise it becomes trivial to link blinded keys to the original key. Based on these observations, we conclude that each public key must use the same input x. It was shown in the full version of [7] that $k \mapsto f_k(x)$ is one-way for any input x if f_k is pseudorandom.

We now present our general framework for instantiating post-quantum digital signatures with key blinding from MPCitH. We assume for simplicity that the messages are of a fixed size $n(\lambda)$ determined by the security parameter.

Construction 1. Assume the following prerequisites:

– A security parameter $\lambda \in \mathbb{N}$
– A family of pseudorandom permutations $\mathcal{F} = \{f_k : \{0,1\}^{2\lambda} \to \{0,1\}^{2\lambda}\}_{k\in\{0,1\}^\lambda}$,
– A fixed input $\mathsf{inp} \in \{0,1\}^{2\lambda}$.
– $F : \{0,1\}^{2\lambda} \to \{0,1\}^{2\lambda}$ defined as $F(k,k') := f_{k'} \circ f_k(\mathsf{inp})$.
– Cryptographic hash functions $H, G : \{0,1\}^* \to \{0,1\}^\lambda$ (modelled as random oracles).
– A NIZKPoK $\Pi = (\mathsf{P}^H, \mathsf{V}^H)$ for the relation $R_F : \{0,1\}^{2\lambda\cdot n(\lambda)} \times \{0,1\}^{2\lambda}$ defined as $R_F = \{(y\|m, (k,k')) \mid y = F(k,k'), |m| = n(\lambda)\}$.

We define $\mathsf{blSig} = (\mathsf{KGen}, \mathsf{BlindPK}, \mathsf{Sign}, \mathsf{Verify})$ as the following signature scheme with key blinding:

– $\mathsf{KGen}(1^\lambda)$ returns $\mathsf{sk} \leftarrow\!\$\ \{0,1\}^\lambda$ and $\mathsf{pk} = f_{\mathsf{sk}}(\mathsf{inp})$
– $\mathsf{BlindPK}(\mathsf{pk}, \tau)$ returns $\mathsf{bpk} = f_{G(\tau,\mathsf{pk})}(\mathsf{pk})$
– $\mathsf{Sign}(m, \mathsf{sk}, \tau)$ computes $\mathsf{pk} = f_{\mathsf{sk}}(\mathsf{inp})$ and $\mathsf{bpk} = \mathsf{BlindPK}(\mathsf{pk}, \tau)$ and returns $\mathsf{P}^H(\mathsf{bpk}\|m, \mathsf{sk}, G(\tau, \mathsf{pk}))$
– $\mathsf{Verify}(\mathsf{bpk}, m, \sigma)$ returns $\mathsf{V}^H(\mathsf{bpk}\|m, \sigma)$

In the remainder of this section, we show the security of our construction assuming certain properties of $\mathcal{F}$ and Π.

3.2 Security of (Blinded) Key Generation

Forging signatures for a scheme built from Construction 1 is at most as hard as finding preimages of F. The hardness of inverting F relies on the fact that if $\{f_k\}_k$ is a family of pseudorandom permutations, then for any x, $f_k(x)$ is one-way with respect to k, i.e. $\{F_x : k \mapsto f_k(x)\}_x$ is a one-way function family.

Lemma 1 ([7]). *Let* $\{f_k : \{0,1\}^\lambda \to \{0,1\}^\lambda\}_{k\in\{0,1\}^\lambda}$ *be a family of pseudorandom permutations, then* $F_x(k) := f_k(x)$ *is a one-way function for any* $x \in \{0,1\}^\lambda$.

The proof of Lemma 1 is found in the full version of [7]. It relies on the fact that the message space (or block size) of f is equal to its key space (or key size). Since the function F applies f twice with two different keys, the block size should be twice the key size.

Since F is actually built from the function family $\{f_{k'} \circ f_k\}_{(k,k')}$. It remains to show that this is a family of pseudorandom functions if f_k are pseudorandom.

Lemma 2. *If* $\{f_k\}_k$ *is a family of pseudorandom permutations, then* $\{f_{k'} \circ f_k\}_{(k,k')}$ *is also a family of pseudorandom permutations.*

Proof. Suppose there is a distinguisher $\mathcal{A}$ against the pseudorandomness game: it is given oracle access to either $f_{k'} \circ f_k$ for random k, k' or to a random function R, and tries to distinguish between both cases. We construct an adversary $\mathcal{B}$ against the pseudorandomness of f_k as follows: given an oracle $\mathcal{O}$ which is either f_k or a random function, sample a key k' and run $\mathcal{A}$ with oracle $f_{k'} \circ \mathcal{O}$.

Since $f_{k'}$ is a permutation, for a random R the function $f_{k'} \circ R$ is also random. Therefore, if $\mathcal{O} = f_k$, $\mathcal{A}$ has an oracle for $f_{k'} \circ f_k$ and if $\mathcal{O} = R$, $\mathcal{A}$ has an oracle for the random function $R' = f_{k'} \circ R$. We conclude that $\mathcal{B}$ distinguishes f_k from a random function with the same advantage as $\mathcal{A}$ distinguishes $f_{k'} \circ f_k$ from random. □

3.3 Unforgeability of Construction 1

We now want to show that forging valid signatures for blinded keys is at least as hard (asymptotically) as inverting the one-way function F. Recall that for breaking the unforgeability game, the adversary on input pk must produce τ^*, m^* and a valid proof of knowledge σ^* of (k, k') such that $\mathsf{BlindPK}(\mathsf{pk}, \tau^*) = F(k, k')$. The typical strategy of using the knowledge extractor of the NIZKPoK on σ^* does not directly work here. In particular the adversary could produce a proof for $(k, k') \neq (\mathsf{sk}, G(\tau^*, \mathsf{pk}))$, such that the knowledge extractor would not necessarily allow us recover the secret sk. Furthermore, the adversary has some control over the target blinded public key bpk for its forgery, however this control is limited to the choice τ which rerandomizes pk through the random oracle.

To precisely capture the problem the adversary needs to solve for forging signatures, we define the following NP relation.

Definition 7. *Let* $G : \{0,1\}^* \to \{0,1\}^\lambda$ *be a quantumly accessible random oracle, let* $\{f_k : \{0,1\}^{2\lambda} \to \{0,1\}^{2\lambda}\}_{k\in\{0,1\}^\lambda}$ *be a family of pseudorandom permutations and let* $\mathsf{inp} \in \{0,1\}^{2\lambda}$. *We define* $R_{G,f}$ *as the NP relation where the instances are of the form* $y \in \{0,1\}^{2\lambda}$ *and witnesses are tuples* (k, k', τ) *such that*

$$(y, (k, k', \tau)) \in R_{G,f} \iff f_{G(\tau,y)}(y) = f_{k'}(f_k(\mathsf{inp})) \ . \tag{4}$$

Using the fact that G is a random oracle, we can show that this relation is as hard as inverting F by reprogramming the oracle at a certain point. Let $\mathcal{A}^G$ be an adversary that on input y outputs a witness for $R_{G,f}$ after q queries to G with some probability p. We can invert F on a specific point z by sampling a key s, setting $y = f_s^{-1}(z)$, running $\mathcal{A}^G$ on input y, and reprogramming one of its oracle queries to output s. With probability p, $\mathcal{A}^G$ outputs (τ, k, k') such that $f_{G(\tau,y)}(y) = f_{k'}(f_k(\mathsf{inp}))$. With probability $\frac{1}{q}$, (τ, y) is the point that was programmed as $G(\tau, y) = s$, such that $z = f_s(y) = f_{k'}(f_k(\mathsf{inp}))$. Thus (k, k') is a preimage of z for F as required. We formalize this argument in the QROM using the measure-and-reprogram approach [12,13] with Lemma 3 whose proof is in Appendix B.

Lemma 3. *Let $\{f_k\}_{k\in\{0,1\}^\lambda}$ be a family of pseudorandom permutations. Let G be a quantum-accessible random oracle. If $F : (k, k') \mapsto f_{k'}(f_k(\mathsf{inp}))$ is a one-way function, then the relation $R_{G,f}$ is hard with advantage*

$$\mathsf{Adv}^{R_{G,f}}(\lambda) \leq (2q_G + 1)^2 \cdot \mathsf{Adv}_F^{\mathrm{OWF}}(\lambda) \tag{5}$$

where q_G is the number of queries to G.

Before going into the proof of unforgeability, we observe that unlike the unforgeability for regular digital signatures, in the EUF-CMBA game, the adversary can output a message m^* that was queried to the signature oracle, as long as τ^* differs. The proof thus requires bounding the probability of reusing a signature that was issued for m^* with a different τ. Only then can we move to standard techniques to bound the probability of producing a forgery. To this end, we make some assumptions about the functions f_k. We require that it is hard to find $\tau \neq \tau'$ such that $f_{G(\mathsf{pk},\tau)}(\mathsf{pk}) = f_{G(\mathsf{pk},\tau')}(\mathsf{pk})$. This is the case if we assume that G is collision resistant and that the f_k are key-collision-resistant[8] (Definition 11).

Theorem 1. *Let* blSig *be the signature scheme with key blinding form Construction 1. Assume that $\mathcal{F} = \{f_k\}_k$ is a family of key-collision-resistant pseudorandom permutations. Then* blSig *is* EUF-CMBA *in the QROM with advantage at most*

$$\begin{aligned}\mathsf{Adv}^{\text{EUF-CMBA}}(\lambda) \leq \mathsf{Adv}^{R_{G,f}}(\lambda) + O(q_G^3 \cdot 2^{-\lambda}) + \mathsf{Adv}^{\mathrm{KCR}}(\lambda) \\ + q_s \cdot \varepsilon_{\mathrm{zk}}(\lambda, q_H) + \varepsilon_{\mathrm{ss}}(\lambda, q_H, q_S) \\ + \varepsilon_{\mathrm{sim}}(\lambda, q_H) + \varepsilon_{\mathrm{ex}}(\lambda, q_H) \ .\end{aligned}$$

where q_H, q_G and q_S are respectively the number of queries to H, G and Sig *made by the adversary; where $\varepsilon_{\mathrm{zk}}, \varepsilon_{\mathrm{sim}}, \varepsilon_{\mathrm{ss}}$ and $\varepsilon_{\mathrm{ex}}$ are defined in Section 2.3; and where $\mathsf{Adv}^{\mathrm{KCR}}(\lambda)$ is the advantage in the key-collision-resistance game (Definition 11).*

[8] Since keys are derived from the random function G, key-collision-resistance could potentially be replaced with a weaker assumption.

Proof. Let $\mathcal{A}^{G,H,\mathsf{Sig}}$ be an adversary against the EUF-CMBA experiment with Construction 1 that has access to a signature oracle Sig, to the QRO H for the proof system Π and to the QRO G used to blind public keys.

There are two scenarios for a forgery produced by $\mathcal{A}^{G,H,\mathsf{Sig}}$: either m^* was not queried to the signing oracle, or m^* was queried to Sig, but with a $\tau \neq \tau^*$. The first case requires the adversary to produce a new valid proof of knowledge and will be handled using standard techniques. The second case occurs if τ and τ^* lead to the same blinded public key. We handle the second and simpler case first.

The proof proceeds in a sequence of hybrid games, bounding the difference in the probability of winning between each hybrid. We start with the original EUF-CMBA game (Definition 2). Let WIN_i denote the event that the challenger outputs 1 in game i and let Q_{Sig} be the list of pairs (m, τ) queried to the signing oracle.

Game 0 (EUF-CMBA). $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KGen}(1^\lambda)$ and $(m^*, \sigma^*, \tau^*) \leftarrow \mathcal{A}^{G,H,\mathsf{Sig}}(\mathsf{pk})$

$$\text{WIN}_0 = \mathsf{Verify}(\mathsf{BlindPK}(\mathsf{pk}, \tau^*), m^*, \sigma^*) \wedge (m^*, \tau^*) \notin Q_{\mathsf{Sig}} \tag{6}$$

We modify this game by adding the condition that the challenger outputs 0 if there exist $\tau \neq \tau^*$ such that $(m^*, \tau) \in Q_{\mathsf{Sig}}$ and $G(\mathsf{pk}, \tau) = G(\mathsf{pk}, \tau^*)$

Game 1. $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KGen}(1^\lambda)$ and $(m^*, \sigma^*, \tau^*) \leftarrow \mathcal{A}^{G,H,\mathsf{Sig}}(\mathsf{pk})$

$$\text{WIN}_1 = \text{WIN}_0 \wedge \forall \tau \neq \tau^*, (m^*, \tau) \in Q_{\mathsf{Sig}} \implies G(\mathsf{pk}, \tau) \neq G(\mathsf{pk}, \tau^*) \tag{7}$$

Let COLL be the event $\exists \tau \neq \tau^* : (m^*, \tau) \in Q_{\mathsf{Sig}} \wedge G(\mathsf{pk}, \tau) = G(\mathsf{pk}, \tau^*)$. We have $\text{WIN}_1 = \text{WIN}_0 \wedge \neg\text{COLL}$. The difference between games 0 and 1 is thus

$$\begin{aligned} |\Pr[\text{WIN}_0] - \Pr[\text{WIN}_1]| &= |\Pr[\text{WIN}_0 \wedge \text{COLL}] + \Pr[\text{WIN}_0 \wedge \neg\text{COLL}] - \Pr[\text{WIN}_1]| \\ &\leq \Pr[\text{COLL}] \\ &\leq O(q_G^3 2^{-\lambda}) \end{aligned}$$

where the above bound comes from the bound on quantum collision finding (Lemma 6) for the quantum random oracle G.

Next, we get the challenger to abort whenever there is a τ in the list of queries that lead to the same blinded key as τ^*.

Game 2. $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KGen}(1^\lambda)$ and $(m^*, \sigma^*, \tau^*) \leftarrow \mathcal{A}^{G,H,\mathsf{Sig}}(\mathsf{pk})$

$$\text{WIN}_2 = \text{WIN}_1 \wedge \forall \tau \neq \tau^* : (m^*, \tau) \in Q_{\mathsf{Sig}} \implies f_{G(\mathsf{pk},\tau)}(\mathsf{pk}) \neq f_{G(\mathsf{pk},\tau^*)}(\mathsf{pk}) \tag{8}$$

We introduce the event KEYCOLL defined as $\exists \tau \neq \tau^* : (m^*, \tau) \in Q_{\mathsf{Sig}} \wedge f_{G(\mathsf{pk},\tau)}(\mathsf{pk}) = f_{G(\mathsf{pk},\tau^*)}(\mathsf{pk})$. Then, WIN_2 is the event $\text{WIN}_0 \wedge \neg\text{COLL} \wedge \neg\text{KEYCOLL}$ and

$$|\Pr[\text{WIN}_1] - \Pr[\text{WIN}_2]| \leq \Pr[\neg\text{COLL} \wedge \text{KEYCOLL}] \leq \mathsf{Adv}^{\text{KCR}}_{\mathcal{A}}(\lambda)$$

since, conditioned on $\neg$COLL, if event KEYCOLL occurs we can find a key collision by looking through the signature queries to find τ and τ^* such that $G(\mathsf{pk}, \tau) \neq G(\mathsf{pk}, \tau^*)$ is a key collision for f.

In the next game, the oracle $\mathsf{Sig}(m,\tau) = \mathsf{Sign}(m,\tau,\mathsf{sk})$ is replaced with the oracle $\mathsf{Sig}'(m,\tau) := \mathsf{Sim}(\mathsf{BlindPK}(\mathsf{pk},\tau)\|m)$ where Sim is the zero-knowledge simulator of the NIZKPoK Π.

Game 3. $(\mathsf{pk},\mathsf{sk}) \leftarrow \mathsf{KGen}(1^\lambda)$, $(m^*,\sigma^*,\tau^*) \leftarrow \mathcal{A}^{G,H,\mathsf{Sig}'}(\mathsf{pk})$, $\mathsf{Sig}'(m,\tau) := \mathsf{Sim}(\mathsf{BlindPK}(\mathsf{pk},\tau)\|m)$

$$\text{WIN}_3 = \text{WIN}_2$$

Since the scheme is zero-knowledge (Definition 4) and by a union bound over the number of signature queries q_S, we have

$$|\Pr[\text{WIN}_3] - \Pr[\text{WIN}_2]| \leq q_S \cdot \varepsilon_{\text{zk}}(\lambda, q_H) \tag{9}$$

Recall that in Game 3, the adversary can only win if

$$\forall \tau \neq \tau^*, (m^*,\tau) \in Q_{\mathsf{Sig}} \implies \mathsf{BlindPK}(\mathsf{pk},\tau) \neq \mathsf{BlindPK}(\mathsf{pk},\tau^*) \tag{10}$$

Furthermore, since the signature oracle Sig' now calls the zero-knowledge simulator, if we let $\mathsf{bpk}^* = \mathsf{BlindPK}(\mathsf{pk},\tau^*)$, condition (10) implies that $\mathsf{bpk}^*\|m^* \notin Q_{\mathsf{Sim}}$ where Q_{Sim} is the list of queries to the zero-knowledge simulator. Thus, we can rewrite event WIN_3 as

$$\begin{aligned}\text{WIN}_3 &= \mathsf{Verify}(\mathsf{bpk}^*, m^*, \sigma^*) \wedge \mathsf{bpk}^*\|m^* \notin Q_{\mathsf{Sim}} \\ &= \mathsf{V}^{H_\theta}(\mathsf{bpk}^*\|m^*, \sigma^*) \wedge \mathsf{bpk}^*\|m^* \notin Q_{\mathsf{Sim}}\end{aligned}$$

where verification of the NIZKPoK is performed with respect to the reprogrammed oracle H_θ.

Given an efficient adversary $\mathcal{A}^{G,H,\mathsf{Sig}'}$ against Game 3, we construct an adversarial prover[9] $\mathcal{P}^{H,\mathsf{Sim}}$ against Π that takes input pk and runs $\mathcal{A}^{G,H,\mathsf{Sig}'}$:

- Whenever $\mathcal{A}^{G,H,\mathsf{Sig}'}$ makes a query to Sig' on input (m,τ), $\mathcal{P}^{H,\mathsf{Sim}}$ queries Sim on input $\mathsf{BlindPK}(\mathsf{pk},\tau)\|m$.
- When $\mathcal{A}^{G,H,\mathsf{Sig}'}$ outputs (m^*,τ^*,σ^*), $\mathcal{P}^{H,\mathsf{Sim}}$ outputs the instance/proof pair $(x,\pi) = (\mathsf{BlindPK}(\mathsf{pk},\tau^*)\|m^*,\sigma^*)$ along with auxiliary output $Z = \tau^*$.

To ensure that x has the correct form throughout the following games, we introduce the parameterized set $\mathcal{X}_{\mathsf{pk},\tau^*} := \{\mathsf{BlindPK}(\mathsf{pk},\tau^*)\|m\}_{m\in\{0,1\}^{n(\lambda)}}$.

Game 4. $(\mathsf{pk},\mathsf{sk}) \leftarrow \mathsf{KGen}(1^\lambda)$, $(x,\pi,\tau^*) \leftarrow \mathcal{P}^{H,\mathsf{Sim}}(\mathsf{pk})$,

$$\text{WIN}_4 = x \in \mathcal{X}_{\mathsf{pk},\tau^*} \wedge \mathsf{V}^{H_\theta}(x,\pi) \wedge x \notin Q_{\mathsf{Sim}}$$

We have that by definition of $\mathcal{P}^{H,\mathsf{Sim}}$,

$$\Pr[\text{WIN}_4] = \Pr[\text{WIN}_3] \ .$$

Let $\mathcal{B}^H$ be the malicious prover without access to Sim that emulates $\mathcal{P}^{H,\mathsf{Sim}}$ from Definition 6 (simulation-soundness). In the next game, we let (x,π,τ^*) be

[9] $\mathcal{P}$ also has oracle access to G, but from this point on, we drop the superscript to lighten notation.

generated by $\mathcal{B}^H$ instead of $\mathcal{P}^{H,\mathsf{Sim}}$ and make verification with respect to the unprogrammed oracle H.

Game 5. $(\mathsf{pk},\mathsf{sk}) \leftarrow \mathsf{KGen}(1^\lambda)$, $(x,\pi,\tau^*) \leftarrow \mathcal{B}^H(\mathsf{pk})$

$$\text{WIN}_5 = x \in \mathcal{X}_{\mathsf{pk},\tau^*} \wedge \mathsf{V}^H(x,\pi)$$

By Definition 6, we have

$$\Pr[\text{WIN}_5] = \Pr[\text{WIN}_4] \leq \varepsilon_{\text{ss}}(\lambda, q_H, q_S)$$

for ε_{ss} negligible in λ when q_H and q_S are polynomial in λ.

We now invoke the online knowledge extractor for $\mathcal{B}^H$ from Definition 5. In a first step, the random oracle is emulated by the extractor.

Game 6. $(\mathsf{pk},\mathsf{sk}) \leftarrow \mathsf{KGen}(1^\lambda)$, $(x,\pi,(k,k'),\tau^*) \leftarrow \mathcal{B}^{\mathcal{E}}(\mathsf{pk})$

$$\text{WIN}_6 = x \in \mathcal{X}_{\mathsf{pk},\tau^*} \wedge \mathsf{V}^{\mathcal{E}}(x,\pi)$$

By Definition 5, we have

$$|\Pr[\text{WIN}_6] - \Pr[\text{WIN}_5]| \leq \varepsilon_{\text{sim}}(\lambda, q_H)$$

Next, we change the winning condition to also check that the witness produced by the knowledge extractor is a valid preimage for F.

Game 7. $(\mathsf{pk},\mathsf{sk}) \leftarrow \mathsf{KGen}(1^\lambda)$, $(x,\pi,(k,k'),\tau^*) \leftarrow \mathcal{B}^{\mathcal{E}}(\mathsf{pk})$

$$\text{WIN}_7 = x \in \mathcal{X}_{\mathsf{pk},\tau^*} \wedge \mathsf{V}^{\mathcal{E}}(x,\pi) \wedge F(k,k') = \mathsf{BlindPK}(\mathsf{pk},\tau^*)$$

Recall the relation $R_F = \{(x,(k,k')) \mid \exists m : x = y \| m \wedge y = F(k,k')\}$ which is proven by the NIZKPoK Π. The condition $x \in \mathcal{X}_{\mathsf{pk},\tau^*} \wedge F(k,k') = \mathsf{BlindPK}(\mathsf{pk},\tau^*)$ implies that $(x,(k,k')) \in R_F$. By Definition 5, we thus have

$$|\Pr[\text{WIN}_7] - \Pr[\text{WIN}_6]| \leq \Pr[\mathsf{V}^{\mathcal{E}}(x,\pi) \wedge (x,(k,k')) \notin R_F] \leq \varepsilon_{\text{ex}}(\lambda, q_H)$$

We now define $\mathcal{M}$, an algorithm for the relation $R_{G,f}$ that, on input $y = \mathsf{pk}$, runs the online knowledge extractor on input pk to get k, k' and τ^*.

Game 7. $(\mathsf{pk},\mathsf{sk}) \leftarrow \mathsf{KGen}(1^\lambda)$, $(k,k',\tau^*) \leftarrow \mathcal{M}(\mathsf{pk})$

$$\text{WIN}_8 = (\mathsf{pk},(k,k',\tau^*)) \in R_{G,f}$$

We have

$$\Pr[\text{WIN}_8] \geq \Pr[\text{WIN}_7]$$

Since Game 0 is the EUF-CMBA game and Game 8 is finding a witness for the relation $R_{G,f}$, the statement follows by accounting for the errors across all games:

$$\begin{aligned}\Pr[\text{WIN}_0] \leq \Pr[\text{WIN}_8] &+ O(q_G^3 \cdot 2^{-\lambda}) + \mathsf{Adv}^{\text{KCR}}(\lambda) \\ &+ q_s \cdot \varepsilon_{\text{zk}}(\lambda, q_H) + \varepsilon_{\text{ss}}(\lambda, q_H, q_S) \\ &+ \varepsilon_{\text{sim}}(\lambda, q_H) + \varepsilon_{\text{ex}}(\lambda, q_H) \ .\end{aligned}$$

□

3.4 Unlinkability of Construction 1

We show unlinkability of Construction 1 in the QROM using a similar strategy as [17]. That is, we first use the zero-knowledge simulator to remove the signature oracle. Next, we remove the dependence on the identity public key pk of the blinded keys by first replacing $G(\tau, \mathsf{pk})$ with random values and then changing the blinding oracle so that it returns encryptions of independently generated public keys. This reduces the unlinkability game to the task of distinguishing many encryptions of the same plaintext from many encryptions of independent plaintexts. This corresponds to the notion of chosen plaintext indistinguishability security in a multi-user setting, defined in Section A.1.

Theorem 2. *Let* $\mathsf{blSig} = (\mathsf{KGen}, \mathsf{BlindPK}, \mathsf{Sign}, \mathsf{Verify})$ *be the blinded signature scheme of Construction 1 built from proof system* $\Pi = (\mathsf{P}^H, \mathsf{V}^H)$ *and he family of PRP* $\mathcal{F} = \{f_k\}_k$*. If* Π *is zero-knowledge (Definition 4) in the QROM and if* $\mathcal{F}$ *satisfies multi-user ciphertext indistinguishability (Definition 12) with advantage* $\mathsf{Adv}^{\text{MU-IND}}(n)$ *then* blSig *is unlinkable under chosen message and blinding attack (Definition 3) in the QROM with advantage*

$$\mathsf{Adv}^{\text{UL-CMBA}}(\lambda) \leq \frac{1}{2} + \mathsf{Adv}^{\text{MU-IND}}(\lambda) + 2q_H\sqrt{\mathsf{Adv}^{\text{MU-IND}}(\lambda)} + q_S \cdot \varepsilon_{\text{zk}}(\lambda, q_H) \quad (11)$$

where q_G*,* q_H*,* q_S *and* q_B *are respectively the number of queries to the random oracles* G *and* H*, to the signing oracle and to the blinding oracle.*

Proof. We proceed by a game-hopping argument to reduce the unlinkability game to the multi-user indistinguishability of f_k. In each new game, we remove some part of the game that depends on the identity public key pk. We assume w.l.o.g. that the adversary never sends τ^* that was previously queried to the signature oracle since this would make the challenger abort. Let $\mathcal{A}_1^{G,H,\mathsf{bl},\mathsf{Sig}}$ and $\mathcal{A}_2^{G,H,\mathsf{bl},\mathsf{Sig}}$ denotes the two phases of the adversary (before and after the challenge blinded key).

Game 0 (UL-CMBA).

$$\begin{aligned}(\mathsf{sk}_i, \mathsf{pk}_i) &\leftarrow \mathsf{KGen}(1^\lambda), i \in \{0,1\}\\ \tau^* &\leftarrow \mathcal{A}_1^{G,H,\mathsf{bl},\mathsf{Sig}}(1^\lambda)\\ b &\leftarrow\!\!\$\ \{0,1\}\\ \mathsf{bpk}^* &\leftarrow \mathsf{BlindPK}(\mathsf{pk}_b, \tau^*)\\ b' &\leftarrow \mathcal{A}_2^{G,H,\mathsf{bl},\mathsf{Sig}}(\mathsf{bpk}^*)\end{aligned}$$

$$\mathsf{bl}(\tau) := \begin{cases}\mathsf{BlindPK}(\mathsf{pk}_b, \tau) & \text{if } \tau = \tau^*\\ \mathsf{BlindPK}(\mathsf{pk}_0, \tau) & \text{if } \tau \neq \tau^*\end{cases}$$

$$\mathsf{Sig}(m, \tau) := \begin{cases}\mathsf{Sign}(m, \mathsf{sk}_b, \tau) & \text{if } \tau = \tau^*\\ \mathsf{Sign}(m, \mathsf{sk}_0, \tau) & \text{if } \tau \neq \tau^*\end{cases}$$

Game 0 is the UL-CMBA experiment.

$$\Pr[b' = b \mid \text{Game } 0] = \mathsf{Adv}_{\mathcal{A}}^{\text{UL-CMBA}}(\lambda) \quad (12)$$

By construction, the signature algorithm is

$$\mathsf{Sign}(m, \mathsf{sk}, \tau) = \mathsf{P}^H(\mathsf{BlindPK}(\mathsf{pk}, \tau) \| m, \mathsf{sk}, G(\tau, \mathsf{pk})).$$

In the next hybrid, we remove dependence of the signature algorithm on sk and $G(\tau, \mathsf{pk})$ by using the HVZK simulator for Π which depends only on the first argument.

Game 1.

$$
\begin{aligned}
(\mathsf{sk}_i, \mathsf{pk}_i) &\leftarrow \mathsf{KGen}(1^\lambda), i \in \{0,1\} \\
\tau^* &\leftarrow \mathcal{A}_1^{G, H_\Theta, \mathsf{bl}, \mathsf{Sig}}(1^\lambda) \\
b &\leftarrow\!\!\$\ \{0,1\} \\
\mathsf{bpk}^* &\leftarrow \mathsf{BlindPK}(\mathsf{pk}_b, \tau^*) \\
b' &\leftarrow \mathcal{A}_2^{G, H_\Theta, \mathsf{bl}, \mathsf{Sig}}(\mathsf{bpk}^*)
\end{aligned}
\qquad
\mathsf{bl}(\tau) := \begin{cases} \mathsf{BlindPK}(\mathsf{pk}_b, \tau) & \text{if } \tau = \tau^* \\ \mathsf{BlindPK}(\mathsf{pk}_0, \tau) & \text{if } \tau \neq \tau^* \end{cases}
$$

$$
\mathsf{Sig}(m, \tau) := \begin{cases} \mathsf{Sim}(\mathsf{BlindPK}(\mathsf{pk}_b, \tau) \| m) & \text{if } \tau = \tau^* \\ \mathsf{Sim}(\mathsf{BlindPK}(\mathsf{pk}_0, \tau) \| m) & \text{if } \tau \neq \tau^* \end{cases}
$$

Two changes are introduced in the above game: all calls to Sign are changed to corresponding calls to Sim, and the adversary's oracle H is replaced with a reprogrammed oracle H_Θ. By invoking the zero-knowledge property of Π (Definition 4) for the q_S simulated proofs, we have

$$
|\Pr[b' = b \mid \text{Game 1}] - \Pr[b' = b \mid \text{Game 1}]| \leq q_S \cdot \varepsilon_{\mathrm{zk}}(\lambda, q_H) \tag{13}
$$

Next, we show how to remove dependence on pk from the symmetric key used to blind the public key. In game 2, whenever $\mathsf{BlindPK}$ is called, it returns $f_{k'}(\mathsf{pk})$ for a random $k' \in \{0,1\}^\lambda$ instead of $f_{G(\tau, \mathsf{pk})}(\mathsf{pk})$. We denote this as $f_\$(\mathsf{pk})$ below.

Game 2.

$$
\begin{aligned}
(\mathsf{sk}_i, \mathsf{pk}_i) &\leftarrow \mathsf{KGen}(1^\lambda), i \in \{0,1\} \\
\tau^* &\leftarrow \mathcal{A}_1^{G, H_\Theta, \mathsf{bl}, \mathsf{Sig}}(1^\lambda) \\
b &\leftarrow\!\!\$\ \{0,1\} \\
\mathsf{bpk}^* &\leftarrow f_\$(\mathsf{pk}_b) \\
b' &\leftarrow \mathcal{A}_2^{G, H_\Theta, \mathsf{bl}, \mathsf{Sig}}(\mathsf{bpk}^*)
\end{aligned}
\qquad
\mathsf{bl}(\tau) := \begin{cases} f_\$(\mathsf{pk}_b) & \text{if } \tau = \tau^* \\ f_\$(\mathsf{pk}_0) & \text{if } \tau \neq \tau^* \end{cases}
$$

$$
\mathsf{Sig}(m, \tau) := \begin{cases} \mathsf{Sim}(f_\$(\mathsf{pk}_b) \| m) & \text{if } \tau = \tau^* \\ \mathsf{Sim}(f_\$(\mathsf{pk}_0) \| m) & \text{if } \tau \neq \tau^* \end{cases}
$$

Since $\mathsf{BlindPK}$ is deterministic, we may assume without loss of generality that the adversary queries the bl oracle at most once per input τ to avoid the need for the oracles to record the keys.

Intuitively, the only way the adversary can detect this change in the behavior of the blinding oracle is if it has queried G on an input containing pk. We use the one-way to hiding technique of [26] (Lemma 5) to make this formal in the context of quantum access to G.

The one-way to hiding lemma allows us to relate the difference in winning probability between games 1 and 2 with the probability of extracting pk from the adversary's random oracle queries. To this end, define $\mathcal{E}^G$ to be an algorithm that simulates the interaction of $\mathcal{A}^{G, H_\Theta, \mathsf{bl}, \mathsf{Sig}}$ with the challenger, and that picks one of the quantum queries of $\mathcal{A}$ to G at random, measures this query and outputs the result. Then, invoking Lemma 5 q_B times (once for each blinding query in which $G(\tau, \mathsf{pk})$ is replaced with a random value) implies that

$$
|\Pr[b = b' \mid \text{Game 2}] - \Pr[b = b' \mid \text{Game 1}]| \leq 2 q_B q_G \sqrt{\Pr[\mathsf{pk} \leftarrow \mathcal{E}^G]} \tag{14}
$$

The probability that $\mathcal{E}$ outputs pk is at most the probability that an adversary recovers the plaintext pk from many encryptions of pk under f using different keys. It is trivially upper-bounded by

$$\Pr[\mathsf{pk} \leftarrow \mathcal{E}^G] \leq \mathsf{Adv}_{\mathcal{E}}^{\text{MU-IND}}(\lambda)$$

since one could turn its advantage into one for the MU-IND (Definition 12) game by sampling $\mathsf{pk}_0, \mathsf{pk}_1$, requesting ciphertexts $c_i = \mathsf{Enc}_{k_i}(\mathsf{pk}_b)$, sending $c_1, \ldots, c_n$ to the adversary which returns pk', and testing for $\mathsf{pk}' = \mathsf{pk}_0$.

We now change the blinding oracle again by replacing the real public key pk with a freshly sampled independent public key pk'. In game 3, when the adversary requests the blinding of the key with a seed τ, the challenger samples $k' \leftarrow\!\!\$\ \{0,1\}^\lambda$ and $\mathsf{pk}' \leftarrow \mathsf{KGen}(1^\lambda)$ and returns $f_{k'}(\mathsf{pk}')$. We denote this as $f_{\$}(\$)$ in the game below (though pk is not necessarily uniformly distributed).

Game 3.

$$\begin{aligned}
(\mathsf{sk}_i, \mathsf{pk}_i) &\leftarrow \mathsf{KGen}(1^\lambda), i \in \{0,1\} \\
\tau^* &\leftarrow \mathcal{A}_1^{G, H_\Theta, \mathsf{bl}, \mathsf{Sig}}(1^\lambda) \\
b &\leftarrow\!\!\$\ \{0,1\} \\
\mathsf{bpk}^* &\leftarrow f_{\$}(\$) \\
b' &\leftarrow \mathcal{A}_2^{G, H_\Theta, \mathsf{bl}, \mathsf{Sig}}(\mathsf{bpk}^*)
\end{aligned}
\qquad
\begin{aligned}
\mathsf{bl}(\tau) &:= f_{\$}(\$) \\
\mathsf{Sig}(m, \tau) &:= \mathsf{Sim}(f_{\$}(\$) \| m)
\end{aligned}$$

We can relate the probability of success in this game with the multi-user indistinguishability of f_k. In game 3, the blinded keys that the adversary receives are encryptions of pk' under a secret key k' where both pk' and k' are unrelated to pk (and sk). If the adversary's behaviour changes in an observable way between games 2 and 3, then we can turn this into a distinguisher for the MU-IND property of f_k in the following way.

Let $\mathcal{D}$ be the following adversary against the MU-IND game:

- Let q_B be an upper-bound on the number of blinding queries made by $\mathcal{A}$. Sample $q_B + 1$ public keys $\mathsf{pk}_0, \ldots, \mathsf{pk}_n$ and query the MU-IND challenger on $(i, \mathsf{pk}_0, \mathsf{pk}_i)_{i \in [q_B]}$ to get the resulting ciphertexts $c_1, \ldots, c_{q_B}$ (which are either an encryption of pk_0 or pk_i with a key k_i).
- $\mathcal{D}$ now runs $\mathcal{A}$ by acting as the challenger in game 3 and by simulating its blinding oracle as follows: on $\mathcal{A}$'s ith query to bl, reply with c_i.
- Output whatever $\mathcal{A}$ outputs.

If $b = 0$ in the MU-IND game, then every c_i is an encryption of pk_0 and thus $\mathcal{A}$ is playing game 2. If $b = 1$, then each c_i is the encryption of a new public key pk_i, so $\mathcal{A}$ is playing game 3. By the MU-IND property of f_k, we have that

$$|\Pr[b = b' \mid \text{Game 3}] - \Pr[b = b' \mid \text{Game 2}]| \leq \mathsf{Adv}_{\mathcal{D}}^{\text{MU-IND}}(\lambda)$$

In game 3, bpk^* and the two oracles bl and Sig are independent of b. Therefore

$$\Pr[b = b' \mid \text{Game 3}] = \frac{1}{2} \ .$$

Summing up all the errors, we have

$$\mathsf{Adv}^{\text{UL-CMBA}}_{\mathcal{A}}(\lambda) \leq \frac{1}{2} + \mathsf{Adv}^{\text{MU-IND}}_{\mathcal{D}}(\lambda) + 2q_B q_H \sqrt{\mathsf{Adv}^{\text{MU-IND}}_{\mathcal{E}}(\lambda)} + q_S \cdot \varepsilon_{\text{zk}}(\lambda, q_H) \ . \quad (15)$$

□

4 The blHelium Signature Scheme with Key Blinding

We now present our blHelium protocol, which follows Construction 1 instantiated with the AES block cipher and the Helium proof system [21]. We begin by giving a brief overview of the Helium signature scheme before describing the changes we make to equip it with key blinding, along with parameter choices. The detailed Helium proof system can be found in [21]. We prove its QROM security in Appendix C.

4.1 Overview of Helium

The scheme takes place over seven *phases*, representing the different message flows in the identification scheme prior to being converted to a signature scheme via the Fiat-Shamir transform (i.e., Phase 1 represents the first prover message, Phase 2 the first challenge, etc.). A high-level description of the proof system is as follows:

- **Committing to MPC Party Seeds.** Each MPC party's randomness is derived from a single seed which is committed to. The protocol runs through a distributed computation of AES, with each party holding a share of the secret key.
- **Checking of the MPC Computation.** Proving that the AES circuit was evaluated correctly means verifying correctness of the shares at each step of the MPC protocol. Every linear operation in the AES circuit can be evaluated locally. The non-linear S-box (a field inverse operation) is done efficiently by injecting shares of s and t such that $s \cdot t = 1$.
- **Challenging the Checking Protocol.** The injected shares of (s_i, t_i) must be checked for consistency. This is done efficiently by using polynomials S and T interpolated such that $S(i) = s_i$ and $T(i) = t_i$. The prover distributes shares of $P = S \cdot T$ to the parties. To verify correctness of the polynomials, a test $P(r) = S(r) \cdot T(r)$ is performed for a random r.

Verification consists of reconstructing the view of each MPC party whose seed was opened, testing for consistency. For a more thorough description of the protocol, we refer the reader to [21].

4.2 The blHelium Signature Scheme

At a high level, the blHelium signature scheme with key blinding follows Construction 1. We add a procedure to blind public keys and modify the signature

and verification protocols of Helium to accommodate signing and verification according to blinded keys.

We describe a 128-bit version of blHelium to provide a direct comparison with Helium (which is implemented for 128-bit AES). However, this parameter set is not secure enough for most scenarios. We elaborate on this below before describing the 256-bit version, which is similar, but uses 256-bit AES and 4 blocks of 16 bytes instead of 2. The 128-bit blHelium consists of the following algorithms:

- KGen: The secret key sk is selected at random from the set of keys such that the circuits for $\text{AES}_{\mathsf{sk}}(\mathbf{0})$ and $\text{AES}_{\mathsf{sk}}(\mathbf{1})$ have no s-boxes which receives the 0 byte input[10]. The public key consists of $\mathsf{pk} = (\text{AES}_{\mathsf{sk}}(\mathbf{0}), \text{AES}_{\mathsf{sk}}(\mathbf{1}))$.
- $\mathsf{BlindPK}(\mathsf{pk}, \tau)$: computes $\mathsf{bk} \leftarrow G(\tau, \mathsf{pk}, t)$ and increments t until the circuits for $\text{AES}_{\mathsf{bk}}(\mathsf{pk}_0)$ and $\text{AES}_{\mathsf{bk}}(\mathsf{pk}_1)$ have no s-boxes which receives the 0 byte input. The blinded public key is $\mathsf{bpk} = (\text{AES}_{\mathsf{bk}}(\mathsf{pk}_0), \text{AES}_{\mathsf{bk}}(\mathsf{pk}_1))$.
- $\mathsf{Sign}(\mathsf{sk}, \tau, m)$: computes the blinding keys $(\mathsf{bk}, \mathsf{bpk})$ as in BlindPK and runs the Helium NIZKPoK to prove knowledge of sk and bk such that $\mathsf{bpk} = (\text{AES}_{\mathsf{bk}}(\text{AES}_{\mathsf{sk}}(\mathbf{0})), \text{AES}_{\mathsf{bk}}(\text{AES}_{\mathsf{sk}}(\mathbf{1})))$.
- $\mathsf{Verify}(\mathsf{bpk}, m, \sigma)$: runs the verification protocol for the NIZKPoK.

4.3 blHelium Parameters & Performance

We consider two parameter sets for the blHelium scheme. The first uses the 128 bit AES block cipher as in Helium, but it suffers from some vulnerabilities. The 256 bits variants of AES offers a secure instantiation of blHelium.

128-bit blHelium. In the context of quantum adversaries 128-bit AES is not enough given that the Grover's unstructured search algorithm halves the security parameter asymptotically. More concretely, recent estimates of the resources required to break AES128 using Grover's algorithm have come up with a circuit depth in the order of 2^{80} [19].

However, even in the classical setting there are issues with instantiating Construction 1 with an 128-bit key cipher. An attacker only needs to find $\tau \neq \tau'$ that lead to the same blinded key to cheat the EUF-CMBA game. For Construction 1, this is at most as hard as finding a collision to a 128-bit hash function, which by the birthday bound is of complexity 2^{64} against classical attacks and even lower against quantum attacks. Note that strictly speaking this is a break of Definition 2, but the resulting forgery is valid for a message and blinded public key pair that was already produced by the signing oracle. All that the adversary has achieved is to find another blinding parameter which leads to the same blinded public key. Depending on the context, this might not be considered a security break, or it might be infeasible to carry out such an attack if, for example, the space of admissible τ is small.

We describe here our scheme based on 128-bit AES. We have implemented this version to provide a direct comparison with Helium (which is only available

[10] This step is necessary for the optimisation that computes the s-box as a field inverse.

for 128-bit AES). We encrypt two different plaintexts, first with sk, and then with bk. The choice of plaintext is arbitrary but must be fixed for all users – for simplicity we have chosen the all 0 and all 1 plaintext. This requires a total of two AES128 key schedules (which takes $2 \times 40 = 80$ s-boxes) and four AES128 encryptions (which takes $4 \times 160 = 640$ s-boxes) for a total of 720 s-boxes. Recall that s-box computation in the MPC protocol are checked using polynomial interpolation. Since the degree of each polynomial is limited by the field of size 128, we split these into 6 sets of polynomials $(S_i, T_i, P_i)_{i\in[6]}$, each being used to prove correctness of $720/6 = 120$ s-box computations. In contrast, Helium requires two sets of polynomials.

Helium and blHelium only use a subset of possible keys, since we restrict to AES circuits which have no substitution box that receives the 0 byte as input. Using the methodology of [8] which assumes the input to each s-box to be a uniformly random element of $\mathbb{F}_{2^8}$, we can estimate the fraction of keys for which the AES circuit has no 0-input s-box as $0.457 \approx (1 - \frac{1}{256})^{200}$. Therefore, the number of admissible keys is approximately 2^{127} (i.e. the key space is reduced roughly by half). Note also that for this reason, the BlindPK of blHelium presented in Section 4.2 differs slightly from Construction 1. It includes an additional parameter t that is incremented until $G(\tau, \mathsf{pk}, t)$ yields an admissible key.

We have implemented our 128-bit blHelium protocol as a fork of the Helium source code and have compared its per performance with Helium. Our focus in benchmarking is on signing and verification: key generation and blinding consist of only a handful AES operations and thus do not represent a significant burden. The sizes of signatures and the CPU time for signing and verification are reported in Figure 1 and compared to Helium. We observe that signature and verification are 2× to 3× slower than Helium and signature size are 2× to 3× larger. This is to be expected from the fact that the circuit form our blHelium scheme evaluates 720 s-boxes instead of 200 for Helium (a factor of 3.6×).

	Helium+AES128			blHelium+AES128		
(N, M)	Sign	Verify	Size	Sign	Verify	Size
(17, 31)	8.169	7.605	17 580	16.816	14.086	52 424
(19, 30)	8.088	7.507	17 016	17.704	15.690	50 736
(31, 26)	8.342	7.810	14 760	20.095	17.569	43 984
(57, 22)	9.918	9.370	12 856	26.391	24.118	37 584
(107, 19)	12.513	12.448	11 420	38.459	35.607	32 776
(139, 18)	14.267	14.196	11 112	44.906	43.037	31 344
(185, 17)	17.881	17.900	10 500	55.109	52.827	29 608
(255, 16)	21.636	21.593	9 888	67.647	67.490	27 872
(371, 15)	28.132	28.698	9 516	90.303	88.953	26 376

Fig. 1. Benchmarking (signature sizes in bytes and signing and verification times in milliseconds) information for our implementation of blinded Helium and comparison with Helium for different parameters N and M representing the number of MPC parties and the number of repetitions, respectively. Timing is averaged over 100 iterations.

256-bit blHelium. The 256-bit version of blHelium is very similar to the 128-bit one, so we focus on the distinctions. It uses AES256, in which keys are composed of 32 bytes and the cipher operates on blocks of 16 bytes. Since we are encrypting twice for blinded public keys, the total key space is composed of 64 bytes. As explained in Section 3.2, we want the plaintext size to match the key size, therefore the AES256 version of blHelium uses 4 blocks of 16 bytes. This means the signature algorithm runs a total of 2 key schedules and 8 encryptions as part of the MPCitH computation. Encryption in AES256 consists of 14 rounds where each round performs an s-box on each of the 16 byte of the state for a total of 224 s-boxes per encryption circuit. Each key schedule consists of 56 s-boxes. Hence, a grand total of $2 \times 56 + 8 \times 224 = 1904$ s-boxes are involved in the circuit we need to run as part of the MPCitH proof. To prove knowledge of preimages for such a circuit using the Helium proof system, one would require $\lceil 1904/127 \rceil = 15$ polynomials to check the multiplication triples. To prevent division by 0, we can again use rejection sampling on the keys and estimate using the methods of [8] that roughly $\frac{1}{3}$ of keys are valid for the encryption circuit (that is, the circuit contains no 0-input s-box) for a loss of roughly 1.4 bits of security.

5 Conclusion & Open Questions

Our generic construction provides key blinding to any MPCitH signature schemes based on symmetric cryptography. They only rely on well-studied primitives and produce short public and blinded keys, which can be used as identifiers in anonymity networks. We have provided an implementation based on the Helium signature scheme that is ready to be experimented with.

A way to avoid introducing the relation $R_{G,f}$ in the proof of unforgeability would be to set the parameters such that key generation has a *lossy mode*, for example by having the block size larger than the key size. Would this lossy generator preserve the key-one-wayness of the encryption scheme? Is the tighter security reduction worth the increased overhead of running more encryption circuits?

We have not considered strong unforgeability for our construction, but we conjecture that it is implied by the concept of *computationally unique responses* for interactive proofs, which is known to imply strong unforgeability for Fiat-Shamir-based signatures [23,12,27].

Acknowledgments. We thank Thomas Bellebaum for spotting the attack against the EUF-CMBA game by finding collisions over τ which was not handled in a previous version of this paper. We also thank an anonymous reviewer for noticing an issue with the 128-bit parameters.

References

1. Baum, C., Braun, L., de Saint Guilhem, C.D., Klooß, M., Orsini, E., Roy, L., Scholl, P.: Publicly Verifiable Zero-Knowledge and Post-Quantum Signatures from VOLE-in-the-Head. In: Handschuh, H., Lysyanskaya, A. (eds.) Advances in Cryptology –

CRYPTO 2023. pp. 581–615. Springer Nature Switzerland, Cham (2023). https://doi.org/10.1007/978-3-031-38554-4_19
2. Baum, C., de Saint Guilhem, C.D., Kales, D., Orsini, E., Scholl, P., Zaverucha, G.: Banquet: Short and Fast Signatures from AES. In: Garay, J.A. (ed.) Public-Key Cryptography – PKC 2021. pp. 266–297. Lecture Notes in Computer Science, Springer International Publishing, Cham (2021). https://doi.org/10.1007/978-3-030-75245-3_11
3. Baum, C., Nof, A.: Concretely-Efficient Zero-Knowledge Arguments for Arithmetic Circuits and Their Application to Lattice-Based Cryptography. In: Kiayias, A., Kohlweiss, M., Wallden, P., Zikas, V. (eds.) Public-Key Cryptography – PKC 2020. pp. 495–526. Lecture Notes in Computer Science, Springer International Publishing, Cham (2020). https://doi.org/10.1007/978-3-030-45374-9_17
4. Bellare, M., Boldyreva, A., Micali, S.: Public-key encryption in a multi-user setting: Security proofs and improvements. In: Preneel, B. (ed.) Advances in Cryptology — EUROCRYPT 2000. pp. 259–274. Lecture Notes in Computer Science, Springer, Berlin, Heidelberg (2000). https://doi.org/10.1007/3-540-45539-6_18
5. Buchmann, J., Dahmen, E., Ereth, S., Hülsing, A., Rückert, M.: On the security of the Winternitz one-time signature scheme. International Journal of Applied Cryptography **3**(1), 84–96 (Jan 2013). https://doi.org/10.1504/IJACT.2013.053435
6. Celi, S., Griffy, S., Hanzlik, L., Perez-Kempner, O., Slamanig, D.: SoK: Signatures with Randomizable Keys. In: Clark, J., Shi, E. (eds.) Financial Cryptography and Data Security. pp. 160–187. Springer Nature Switzerland, Cham (2025). https://doi.org/10.1007/978-3-031-78679-2_9
7. Chase, M., Derler, D., Goldfeder, S., Orlandi, C., Ramacher, S., Rechberger, C., Slamanig, D., Zaverucha, G.: Post-quantum zero-knowledge and signatures from symmetric-key primitives. In: Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security. pp. 1825–1842. CCS '17, Association for Computing Machinery, New York, NY, USA (Oct 2017). https://doi.org/10.1145/3133956.3133997
8. de Saint Guilhem, C.D., De Meyer, L., Orsini, E., Smart, N.P.: BBQ: Using AES in Picnic Signatures. In: Paterson, K.G., Stebila, D. (eds.) Selected Areas in Cryptography – SAC 2019. pp. 669–692. Lecture Notes in Computer Science, Springer International Publishing, Cham (2020). https://doi.org/10.1007/978-3-030-38471-5_27
9. de Saint Guilhem, C.D., Orsini, E., Tanguy, T.: Limbo: Efficient Zero-knowledge MPCitH-based Arguments. pp. 3022–3036. CCS '21, Association for Computing Machinery, New York, NY, USA (Nov 2021). https://doi.org/10.1145/3460120.3484595
10. Denis, F., Eaton, E., Lepoint, T., Wood, C.A.: Key blinding for signature schemes. Internet-Draft, Internet Engineering Task Force / Internet Engineering Task Force (Jul 2023), https://datatracker.ietf.org/doc/draft-irtf-cfrg-signature-key-blinding/04/
11. Dobraunig, C., Kales, D., Rechberger, C., Schofnegger, M., Zaverucha, G.: Shorter Signatures Based on Tailor-Made Minimalist Symmetric-Key Crypto. In: Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security. pp. 843–857. CCS '22, Association for Computing Machinery, New York, NY, USA (Nov 2022). https://doi.org/10.1145/3548606.3559353
12. Don, J., Fehr, S., Majenz, C.: The measure-and-reprogram technique 2.0: Multi-round fiat-shamir and more. In: Micciancio, D., Ristenpart, T. (eds.) Advances

in Cryptology – CRYPTO 2020. pp. 602–631. Lecture Notes in Computer Science, Springer International Publishing, Cham (2020). https://doi.org/10.1007/978-3-030-56877-1_21
13. Don, J., Fehr, S., Majenz, C., Schaffner, C.: Security of the Fiat-Shamir Transformation in the Quantum Random-Oracle Model. In: Boldyreva, A., Micciancio, D. (eds.) Advances in Cryptology – CRYPTO 2019. pp. 356–383. Lecture Notes in Computer Science, Springer International Publishing, Cham (2019). https://doi.org/10.1007/978-3-030-26951-7_13
14. Don, J., Fehr, S., Majenz, C., Schaffner, C.: Efficient NIZKs and Signatures from Commit-and-Open Protocols in the QROM. In: Dodis, Y., Shrimpton, T. (eds.) Advances in Cryptology – CRYPTO 2022. pp. 729–757. Springer Nature Switzerland, Cham (2022). https://doi.org/10.1007/978-3-031-15979-4_25
15. Don, J., Fehr, S., Majenz, C., Schaffner, C.: Online-extractability in the quantum random-oracle model. In: Dunkelman, O., Dziembowski, S. (eds.) Advances in Cryptology – EUROCRYPT 2022. pp. 677–706. Lecture Notes in Computer Science, Springer International Publishing, Cham (2022). https://doi.org/10.1007/978-3-031-07082-2_24
16. Eaton, E., Lepoint, T., Wood, C.A.: Security analysis of signature schemes with key blinding (2023), https://eprint.iacr.org/2023/380
17. Eaton, E., Stebila, D., Stracovsky, R.: Post-quantum key-blinding for authentication in anonymity networks. In: Longa, P., Ràfols, C. (eds.) Progress in Cryptology – LATINCRYPT 2021. pp. 67–87. Lecture Notes in Computer Science, Springer International Publishing, Cham (2021). https://doi.org/10.1007/978-3-030-88238-9_4
18. Giacomelli, I., Madsen, J., Orlandi, C.: ZKBoo: Faster zero-knowledge for boolean circuits. pp. 1069–1083. SEC'16, USENIX Association, USA (Aug 2016)
19. Grassl, M., Langenberg, B., Roetteler, M., Steinwandt, R.: Applying Grover's Algorithm to AES: Quantum Resource Estimates. In: Takagi, T. (ed.) Post-Quantum Cryptography. pp. 29–43. Springer International Publishing, Cham (2016). https://doi.org/10.1007/978-3-319-29360-8_3
20. Ishai, Y., Kushilevitz, E., Ostrovsky, R., Sahai, A.: Zero-knowledge from secure multiparty computation. In: Proceedings of the Thirty-Ninth Annual ACM Symposium on Theory of Computing. pp. 21–30. STOC '07, Association for Computing Machinery, New York, NY, USA (Jun 2007). https://doi.org/10.1145/1250790.1250794
21. Kales, D., Zaverucha, G.: Efficient lifting for shorter zero-knowledge proofs and post-quantum signatures. Cryptology ePrint Archive, Paper 2022/588 (2022), https://eprint.iacr.org/2022/588
22. Katz, J., Kolesnikov, V., Wang, X.: Improved Non-Interactive Zero Knowledge with Applications to Post-Quantum Signatures. In: Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security. pp. 525–537. CCS '18, Association for Computing Machinery, New York, NY, USA (Oct 2018). https://doi.org/10.1145/3243734.3243805
23. Kiltz, E., Lyubashevsky, V., Schaffner, C.: A concrete treatment of fiat-shamir signatures in the quantum random-oracle model. In: Nielsen, J.B., Rijmen, V. (eds.) Advances in Cryptology: EUROCRYPT 2018. pp. 552–586. Lecture Notes in Computer Science, Springer International Publishing, Cham (2018). https://doi.org/10.1007/978-3-319-78372-7_18
24. Project Tor: Tor rendezvous specification version 3.0 (2020)
25. Schanzenbach, M., Grothoff, C., Fix, B.: The GNU name system. RFC 9498 (Nov 2023). https://doi.org/10.17487/RFC9498

26. Unruh, D.: Revocable Quantum Timed-Release Encryption. Journal of the ACM **62**(6), 49:1–49:76 (Dec 2015). https://doi.org/10.1145/2817206
27. Unruh, D.: Post-quantum Security of Fiat-Shamir. In: Takagi, T., Peyrin, T. (eds.) Advances in Cryptology – ASIACRYPT 2017. pp. 65–95. Lecture Notes in Computer Science, Springer International Publishing, Cham (2017). https://doi.org/10.1007/978-3-319-70694-8_3
28. Zhandry, M.: How to Record Quantum Queries, and Applications to Quantum Indifferentiability. In: Boldyreva, A., Micciancio, D. (eds.) Advances in Cryptology – CRYPTO 2019. pp. 239–268. Springer International Publishing, Cham (2019). https://doi.org/10.1007/978-3-030-26951-7_9

A Additional Preliminaries

A.1 Security Definitions for Symmetric Primitives

Definition 8 (One-Way Function). *A function $F : \mathcal{X} \to \mathcal{Y}$ is one-way (OWF) if for all QPT adversary $\mathcal{A}$,*

$$\Pr[F(x) = y \mid y \in_R \mathcal{Y}, x \leftarrow \mathcal{A}(y)] \leq \mathsf{negl}(\lambda) \ . \tag{16}$$

Definition 9 (Pseudorandom Function). *A family of function $\{f_k : \mathcal{X} \to \mathcal{Y}\}_{k\in\{0,1\}^\lambda}$ is a pseudorandom function family (PRF) if for all QPT oracle adversary $\mathcal{A}^{\mathcal{O}(\cdot)}$,*

$$\left|\Pr[\mathcal{A}^{f_k(\cdot)}(1^\lambda) = 1 \mid k \in_R \{0,1\}^\lambda] - \Pr[\mathcal{A}^{F(\cdot)}(1^\lambda) = 1]\right| \leq \mathsf{negl}(\lambda) \tag{17}$$

where F is a uniformly random function.

Definition 10 (Key-One-Way Function). *A family of function $\{f_k : \mathcal{X} \to \mathcal{Y}\}_{k\in\{0,1\}^\lambda}$ is key-one-way (KOW) if for all QPT adversary $\mathcal{A}$,*

$$\Pr[f_{k'}(x) = f_k(x) \mid k \in_R \{0,1\}^\lambda, x \in_R \mathcal{X}, k' \leftarrow \mathcal{A}(x, f_k(x))] \leq \mathsf{negl}(\lambda) \tag{18}$$

Definition 11 (Key Collision Resistant Function). *A family of function $\{f_k : \mathcal{X} \to \mathcal{Y}\}_{k\in\{0,1\}^\lambda}$ is key-collision-resistant (KCR) if for all QPT adversary $\mathcal{A}$,*

$$\Pr[f_{k'}(x) = f_k(x) \wedge k' \neq k \mid x \in_R \mathcal{X}, (k, k') \leftarrow \mathcal{A}(x)] \leq \mathsf{negl}(\lambda) \tag{19}$$

The following lemma establishes a relation between key-collision-resistance and key-one-wayness.

Lemma 4 ([5]). *Let $\mathcal{F} = \{f_k : \{0,1\}^n \to \{0,1\}^n\}_{k\in\{0,1\}^n}$ be a family of functions. Assuming there exist key collisions for $\mathcal{F}$ and that $\mathcal{F}$ is key-collision-resistant, then $\mathcal{F}$ is key-one-way.*

Definition 12. *Let $\{f_k : \mathcal{X} \to \mathcal{Y}\}_k$ be a family of permutations. The* multi-user indistinguishability *(MU-IND) experiment of parameter n is defined as follows:*

- *The challenger samples n keys $r_1, \ldots, r_n \leftarrow\!\!\$\ \{0,1\}^\lambda$ and a bit $b \leftarrow\!\!\$\ \{0,1\}$.*

– *The adversary may send queries of the form* $(i, m_0, m_1) \in [n] \times \mathcal{X} \times \mathcal{X}$ *to which the challenger responds to with* $f_{r_i}(m_b)$.
– *The adversary outputs a bit* b'.

The advantage of an adversary $\mathcal{A}$ *in the MU-IND game is* $\mathsf{Adv}^{\text{MU-IND}}_{\mathcal{A}}(\lambda) = \left|\Pr[b = b'] - \frac{1}{2}\right|$.

Multi-user security was studied in the context of public-key encryption in [4] where the authors found that it is implied by IND-CPA up to some loss in security. Their results also apply to the private-key setting, thus if we assume that AES, our chosen cipher for blHelium, satisfies chosen plaintext indistinguishability, then it is secure according to Definition 12.

A.2 Quantum Random Oracle Model

Theorem 3 (Measure-and-reprogram [12,13]). *Let* $\mathcal{X}$ *and* $\mathcal{Y}$ *be finite non-empty sets. There exists a black-box two-stage quantum algorithm* $\mathcal{S}$ *with the following property. Let* $\mathcal{A}$ *be an arbitrary oracle quantum algorithm that makes* q *queries to a uniformly random* $H : \mathcal{X} \to \mathcal{Y}$ *and that outputs some* $x \in \mathcal{X}$ *and a (possibly quantum) output* z. *Then, the two-stage algorithm* $\mathcal{S}^{\mathcal{A}}$ *outputs some* $x \in \mathcal{X}$ *in the first stage and, upon a random* $\Theta \in \mathcal{Y}$ *as input to the second stage, a (possibly quantum) output* z, *so that for any* $x_\circ \in \mathcal{X}$ *and any (possibly quantum) predicate* V:

$$\Pr_{\Theta}\left[x = x_\circ \wedge V(x, \Theta, z) : (x, z) \leftarrow \langle \mathcal{S}^{\mathcal{A}}, \Theta \rangle\right] \geq \frac{1}{(2q+1)^2} \Pr_{H}\left[x = x_\circ \wedge V(x, H(x), z) : (x, z) \leftarrow \mathcal{A}^{H}\right].$$

Furthermore, $\mathcal{S}$ *runs in time polynomial in* q, $\log|\mathcal{X}|$ *and* $\log|\mathcal{Y}|$.

Lemma 5 (One-Way to Hiding [26]). *Let* $H : \mathcal{X} \to \mathcal{Y}$ *be a quantumly accessible random oracle and let* $\mathcal{A}^H$ *be an adversary that makes at most* q *queries to* H. *Let* $\mathcal{E}^H$ *be an algorithm that picks* $i \in [q]$ *and* $y \in \mathcal{Y}$ *at random, runs* $\mathcal{A}^H(x, y)$ *until it's* i*th query, measures the input of the query in the computational basis and outputs the measurement outcome.*

$$\left|\Pr[1 \leftarrow \mathcal{A}^H(x, H(x))] - \Pr[1 \leftarrow \mathcal{A}^H(x, y)]\right| \leq 2q \cdot \sqrt{\Pr[x \leftarrow \mathcal{E}^H(x)]}\ . \tag{20}$$

Lemma 6 (Quantum Collision Finding [28]). *Let* $H : \{0,1\}^* \to \{0,1\}^\lambda$ *be a random oracle and let* $\mathcal{A}^H$ *be a* QPT *adversary that makes at most* q *queries to* H. *Then*

$$\Pr[H(x) = H(x') \wedge x \neq x' \mid (x, x') \leftarrow \mathcal{A}^H(1^\lambda)] \leq O(q^3 2^{-\lambda}) \tag{21}$$

B Additional Proofs

B.1 Proof of Lemma 3

We proceed as with the argument presented in Section 3.3, but use the measure-and-reprogram technique (Theorem 3) to insert s into a random oracle query. Let $\mathcal{A}^G$ be an adversary against $R_{G,f}$. Since Theorem 3 asks the adversary to output the point x which is reprogrammed, we define $\hat{\mathcal{A}}^G$ that on input y, runs $(\tau, k, k') \leftarrow \mathcal{A}^G(y)$ and outputs $x = (\tau, y)$ along with (τ, k, k'). Let $\mathcal{S}^{\hat{\mathcal{A}}}$ be the quantum algorithm from Theorem 3 for $\hat{\mathcal{A}}^G$. The reduction $\mathcal{R}$ for inverting F proceeds as follows:

1. On input $z \in \mathrm{Im}F$, sample s uniformly at random and set $y = f_s^{-1}(z)$.
2. Run the first stage of $\mathcal{S}^{\hat{\mathcal{A}}}$ on input y to get a point x.
3. Run the second stage of $\mathcal{S}^{\hat{\mathcal{A}}}$ with input s.
4. When $\mathcal{S}^{\hat{\mathcal{A}}}$ produces an output (x, τ, k, k'), output (k, k').

We now show that this reduction inverts F with probability polynomially related to the probability that $\mathcal{A}^G$ breaks relation $R_{G,f}$. Let $V_y(s, k, k')$ be the predicate that returns 1 if and only if $f_s(y) = f_{k'}(f_k(\mathsf{inp}))$ such that $(y, (\tau, k, k')) \in R_{G,f} \iff V_y(G(\tau, y), k, k') = 1$. Then by Theorem 3,

$$
\begin{aligned}
&\Pr[z = f_{k'}(f_k(\mathsf{inp})) \mid (k, k') \leftarrow \mathcal{R}(z)] \\
&= \Pr[x = (\tau, y) \wedge f_s(y) = f_{k'}(f_k(\mathsf{inp})) \mid (x, \tau, k, k') \leftarrow \langle \mathcal{S}^{\hat{\mathcal{A}}}(y), s \rangle] \\
&= \Pr[x = (\tau, y) \wedge V_y(s, k, k') = 1 \mid (x, \tau, k, k') \leftarrow \langle \mathcal{S}^{\hat{\mathcal{A}}}(y), s \rangle] \\
&\geq \frac{1}{(2q_G + 1)^2} \Pr[x = (\tau, y) \wedge V_y(G(x), k, k') = 1 \mid (x, \tau, k, k') \leftarrow \hat{\mathcal{A}}^G(y)]
\end{aligned}
$$

where q_G is the number of queries to G made by $\hat{\mathcal{A}}^G$. Ignoring the inverse polynomial factor, the above probability corresponds to

$$
\begin{aligned}
&\Pr[V_y(G(\tau, y), k, k') = 1 \mid (\tau, k, k') \leftarrow \mathcal{A}^G(y)] \\
&\quad = \Pr[f_{G(\tau,y)}(y) = f_{k'}(f_k(\mathsf{inp})) \mid (\tau, k, k') \leftarrow \mathcal{A}^G(y)] \\
&\quad = \Pr[(y, (\tau, k, k')) \in R_{G,f} \mid (\tau, k, k') \leftarrow \mathcal{A}^G(y)]
\end{aligned}
$$

which is the advantage of $\mathcal{A}^G$ for finding a witness for relation $R_{G,f}$. □

C QROM Security of blHelium

In the following, we refer to the "Helium proof system" as the 7–message interactive proof from which the Helium signature scheme [21] is obtained through the Fiat-Shamir transform. The verifier checks the validity of the commitments, recreates the views of the MPC parties using the seeds, and verifies that they are consistent with the MPC protocol. We refer to [21] for the full details of the verifier.

C.1 Online Extractability

Our proof relies on online extractability in the QROM. The following Theorem is the online extractability with early extraction result (Corollary 4 in [15] with the simplified bound from Theorem 3).

Theorem 4 (Online Extractability with Early Extraction [15]). *Let $H : \{0,1\}^* \to \{0,1\}^n$ be a random function. There exists an extractable RO-simulator $\mathcal{S}$, with interfaces $\mathcal{S}.RO$ and $\mathcal{S}.E$, that satisfies the following properties. Let $\mathcal{A}$ be a two-round polynomial-time oracle adversary that outputs $t_1, \ldots, t_\ell$ in the first round and $x_1, \ldots, x_\ell \in \mathcal{X}$ and W after the second round, resulting in a transcript $[\boldsymbol{t}, \boldsymbol{x}, H(\boldsymbol{x}), W]_{\mathcal{A}^H}$. Let $[\boldsymbol{t}, \boldsymbol{x}, \boldsymbol{h}, W]_{G^{\mathcal{A}}_{\mathcal{S}}}$ be the transcript where, when $\mathcal{A}$ outputs t_i, $\mathcal{S}.E$ is queried on t_i to obtain $\hat{x}_i \in \mathcal{X} \cup \{\perp\}$ and when $\mathcal{A}$ halts, $\mathcal{S}.RO$ is queried on $\mathcal{A}$'s outputs x_i to generate h_i. There are negligible functions δ_1 and δ_2 such that*

$$\Pr_{G^{\mathcal{A}}_{\mathcal{S}}}[\exists i : x_i \neq \hat{x}_i \wedge h_i = t_i] \leq \delta_1 \tag{22}$$

and

$$\delta([\boldsymbol{t}, \boldsymbol{x}, H(\boldsymbol{x}), W]_{\mathcal{A}^H}, [\boldsymbol{t}, \boldsymbol{x}, \boldsymbol{h}, W]_{G^{\mathcal{A}}_{\mathcal{S}}}) \leq \delta_2 \tag{23}$$

for $\delta_1 + \delta_2 \leq 34\ell q/\sqrt{2^n} + 2365q^3/2^n$ where q is the number of oracle queries.

Note that [14] contains generic results about the online extractability of non-interactive proofs in the QROM. These results are framed in the context of three-message protocols. It is directly clear if they generalize to more rounds, so we present here a direct proof that Helium is online-extractable.

Theorem 5. *Assuming* commit *is a random oracle, then the* Helium *proof system instantiated with a post-quantum one-way function F is an online extractable proof system (Definition 5) for the relation $R_F = \{(x, w) : x = F(w)\}$ with extraction error ε_{ex} and simulation error ε_{sim} satisfying*

$$\begin{aligned}\varepsilon_{\text{sim}}(n,q) + \varepsilon_{\text{ex}}(n,q) \leq\ & 34M \cdot N \cdot q/\sqrt{2^n} + 2365q^3/2^n \\ & + \max_{M_1,M_2,M_3} \left(\frac{2L-2}{|\mathbb{K}|}\right)^{M_1} \cdot \left(\frac{1}{2^8}\right)^{M_2} \cdot \left(\frac{1}{N}\right)^{M_3}\end{aligned} \tag{24}$$

against quantum polynomial-time adversaries making q queries to commit *where n is the bit size of commitments and where $M_1 + M_2 + M_3 = M$ is the number of parallel repetitions.*

Proof. A quantum adversary $\mathcal{A}$ against Helium has quantum superposition access to the commitment oracle $H_c = \mathsf{commit}$. We need to construct a knowledge extractor $\mathcal{E}$ whose success probability in producing a witness is related to $\mathcal{A}$'s probability of cheating the protocol. The extractor will simulate the random oracle H_c with the online extractable oracle $\mathcal{S}$ of [15] specified in Theorem 4. The extractor will run $\mathcal{A}$ by replacing H_c with the oracle interface $\mathcal{S}.RO$. After the prover's first message σ_1, $\mathcal{E}$ will use the extraction interface $\mathcal{S}.E$ on every commitment $\mathsf{com}_e^{(i)}$ for $e \in [M]$ and $i \in [N]$ to get either an oracle input

$(\mathsf{salt}, e, i, \mathsf{seed}_e^{(i)})$ or a symbol $\perp$ which means that $\mathcal{A}$ did not query H_c on this input. Then, $\mathcal{E}$ proceeds in a similar fashion as the extractor of [21, Appendix A] in their proof of unforgeability of BN++ to compute the inputs and shares of the MPC parties.

In more details, when $\mathcal{A}$ produces its first message σ_1, $\mathcal{E}$ does the following:

- parse σ_1 as $(\mathsf{salt}, ((\mathsf{com}_e^{(i)}, \mathsf{ct}_e^{(i)})_{i\in[N]}, \dots)$,
- for $i \in [N]$ and $e \in [M]$, use the $\mathcal{S}.E$ interface on input $\mathsf{com}_e^{(i)}$ to get either $\hat{s}_e^{(i)}$ or $\perp$.
- For each $i \in [N]$ and $e \in [M]$ such that $\mathsf{seed}_e^{(i)}$ is successfully extracted, compute the input shares $\mathsf{sk}_e^{(i)}$ of party i for repetition e using the seed contained in $\hat{s}_e^{(i)}$.
- If there is an $e \in [M]$ for which $\mathsf{sk}_e = \sum_i \mathsf{sk}_e^{(i)}$ is a preimage for pk, output sk_e. Otherwise, output $\perp$.

We now show that the probability that $\mathcal{E}$ outputs a preimage sk is non-negligible if $\mathcal{A}$ produces a forgery with non-negligible probability.

We let $V(\boldsymbol{\sigma}, \boldsymbol{h})$ denote the probabilistic event that V accepts the transcript $(\boldsymbol{\sigma}, \boldsymbol{h})$. We denote by $\boldsymbol{s}_{h_3}$ the set of seeds announced when the challenge is $h_3 = (\bar{i}_e)_{e\in[M]}$, i.e. $\boldsymbol{s}_{h_3} = \{(\mathsf{salt}, e, i, \mathsf{seed}_e^{(i)}) \mid e \in [M], i \neq \bar{i}_e\}$. Similarly, let $\boldsymbol{y}_{h_3}$ denote the commitments to the revealed seeds; i.e. $\boldsymbol{y}_{h_3} = H_c(\boldsymbol{s}_{h_3})$.

We first bound the difference in probability between an execution with H_c and an execution with $\mathcal{S}.RO$.

$$\begin{aligned}
\Pr[V(\boldsymbol{\sigma}, \boldsymbol{h})] &= \Pr[V(\boldsymbol{\sigma}, \boldsymbol{h}) \wedge H_c(\boldsymbol{s}_{h_3}) = \boldsymbol{y}_{h_3}] \\
&\leq \Pr[V(\boldsymbol{\sigma}, \boldsymbol{h}) \wedge \mathcal{S}.RO(\boldsymbol{s}_{h_3}) = \boldsymbol{y}_{h_3}] \\
&+ \Pr[H_c(\boldsymbol{s}_{h_3}) = \boldsymbol{y}_{h_3} \wedge H_c(\boldsymbol{s}_{h_3}) \neq \mathcal{S}.RO(\boldsymbol{s}_{h_3})] \\
&\leq \Pr[V(\boldsymbol{\sigma}, \boldsymbol{h}) \wedge \mathcal{S}.RO(\boldsymbol{s}_{h_3}) = \boldsymbol{y}_{h_3}] + \delta_1
\end{aligned}$$

where δ_1 is the negligible error term of (22).

Next, we bound the probability that the values $\hat{s}_e^{(i)}$ obtained through the $\mathcal{S}.E$ interface differ from the committed values.

$$\begin{aligned}
&\Pr[V(\boldsymbol{\sigma}, \boldsymbol{h}) \wedge \mathcal{S}.RO(\boldsymbol{s}_{h_3}) = \boldsymbol{y}_{h_3}] \\
&\leq \Pr[V(\boldsymbol{\sigma}, \boldsymbol{h}) \wedge \mathcal{S}.RO(\boldsymbol{s}_{h_3}) = \boldsymbol{y}_{h_3} \wedge \mathcal{S}.E(\boldsymbol{y}_{h_3}) = \boldsymbol{s}_{h_3}] \\
&\quad + \Pr[\mathcal{S}.RO(\boldsymbol{s}_{h_3}) = \boldsymbol{y}_{h_3} \wedge \mathcal{S}.E(\boldsymbol{y}_{h_3}) \neq \boldsymbol{s}_{h_3}] \\
&\leq \Pr[V(\boldsymbol{\sigma}, \boldsymbol{h}) \wedge \mathcal{S}.RO(\boldsymbol{s}_{h_3}) = \boldsymbol{y}_{h_3} \wedge \mathcal{S}.E(\boldsymbol{y}_{h_3}) = \boldsymbol{s}_{h_3}] + \delta_2 \\
&= \Pr[V(\boldsymbol{\sigma}, \boldsymbol{h}) \wedge \hat{s}_{h_3} = \boldsymbol{s}_{h_3}] + \delta_2
\end{aligned}$$

where $\hat{s}_{h_3} = \mathcal{S}.E(\boldsymbol{y}_{h_3})$. We again take note of δ_2 and add it at the end.

The expression $\Pr[V(\boldsymbol{\sigma}, \boldsymbol{h}) \wedge \hat{s}_{h_3} = \boldsymbol{s}_{h_3}]$ corresponds to the probability that V accepts when the committed seeds are the values $\hat{s}_e^{(i)}$ extracted by $\mathcal{E}$ through $\mathcal{S}.E$. Note that the prover sends $N \cdot M$ commitments $y_e^{(i)}$, so the values $\hat{s}_e^{(i)}$ are well defined for each $(i, e) \in [N] \times [M]$. We let $\mathsf{sk} \leftarrow \mathcal{E}$ denote the event that

the seeds $\hat{s}_e^{(i)}$ allow $\mathcal{E}$ to compute the shares of the witness sk and $\perp \leftarrow \mathcal{E}$ its complement (when $\mathcal{E}$ outputs $\perp$). We have

$$\begin{aligned}&\Pr[V(\boldsymbol{\sigma}, \boldsymbol{h}) \wedge \hat{s}_{h_3} = \boldsymbol{s}_{h_3}] \leq \\ &\quad \Pr[\mathsf{sk} \leftarrow \mathcal{E}] + \Pr[V(\boldsymbol{\sigma}, \boldsymbol{h}) \wedge \hat{s}_{h_3} = \boldsymbol{s}_{h_3} \mid \perp \leftarrow \mathcal{E}] \ .\end{aligned}$$

We now look at this last probability that the verifier accepts when the extractor is unable to reconstruct a witness from the MPC shares.

Recall that the verifier recomputes the view of each party in the MPC protocol to the exception of the excluded party $\bar{i}$. Since $\hat{s}_{h_3} = \boldsymbol{s}_{h_3}$, it computes those views using the extracted seeds $\hat{s}_e^{(i)}$. If $\mathcal{E}$ outputs $\perp$, then for every e it holds that the key $\hat{\mathsf{sk}}_e = \sum_i \hat{\mathsf{sk}}_e^{(i)}$ expanded from the seed $\hat{s}_e^{(i)}$ is not a valid preimage. This means that at least one of the views computed by the verifier is inconsistent (i.e. that party cheated). In this case the verifier accepts if one of three things happen:

1. the prover injects invalid polynomials S, T and P (such that $S \cdot T \neq P$); or
2. the prover injects invalid multiplication triples; or
3. the view of the inconsistent party is not opened.

There are M parallel repetitions which must pass verification and the prover may try to cheat in a different round in each repetition. We analyze the probability of each event below. The probability that the prover cheats in all M repetitions corresponds to the *trivial* cheating probability in the proof of [15]. At this point, the rest of the analysis is entirely classical and is very similar to the proof of [21] and to other proofs of soundness for multi-round interactive proofs. We bound the probability that the verifier accepts when the MPC shares are computed using the extracted seeds $\hat{s}_e^{(i)}$, conditioned on the extraction failing.

Cheating the first challenge. The first challenge h_1 is used to test the checking polynomials. In Helium, there are two checking polynomials P_1 and P_2 since there are not enough field elements in $\mathbb{F}_{2^8}$ to interpolate a single polynomial with the desired degree. For other one-way functions, there might be more polynomials, for example in blHelium, we use a total of 6 checking polynomials since we are effectively applying 4 AES circuits (see Section 4.3 for details). Let n_p denote the number of checking polynomials and C the total count of field inverse in the circuit such that the degree of each polynomial is $L = \lceil C/n_p \rceil$. By the Schwartz–Zippel Lemma, the probability that a random point R_e satisfies $S_e(R_e) \cdot T_e(R_e) - P_e(R_e) = 0$ is at most $\frac{2L-2}{|\mathbb{K}|}$ where $\mathbb{K}$ is the extension field of $\mathbb{F}_{2^8}$. The challenge h_1 is parsed as $(R_e)_{e \in [M]}$ where for each e, R_e is used to check that $S \cdot T = P$ by checking that $S(R_e) \cdot T(R_e) = P(R_e)$. If we let M_1 denote the number of parallel repetitions e for which the prover cheats in round 1, the probability that the adversary isn't caught is at most

$$\left(\frac{2L-2}{|\mathbb{K}|}\right)^{M_1} \ . \tag{25}$$

Cheating the second challenge. The second challenge h_2 is used to challenge the multiplication triples used to check $S_e(R_e) \cdot T_e(R_e) = P_e(R_e)$. The Helium protocol uses a dot-product checking protocol, which has soundness $1/|\mathbb{F}_{2^8}|$. If we let M_2 denote the number of repetitions where the adversary cheats in the second round, its probability of passing verification is at most

$$\left(\frac{1}{2^8}\right)^{M_2} .$$

Cheating the third challenge. The third challenge is used to challenge the views of the MPC protocol. If the prover did not cheat in any of the previous two rounds, then there is at least one party whose view is inconsistent with that of the others. The prover can cheat in this round if the inconsistent view is not challenged by the verifier. This occurs with probability $\frac{1}{N}$. There are $M_3 = M - M_1 - M_2$ repetitions where the adversary attempts to cheat in the last round. So the probability of success in this round is

$$\left(\frac{1}{N}\right)^{M_3} \tag{26}$$

To complete the proof, we add all the error terms and obtain the bound

$$\begin{aligned}\Pr[V(\boldsymbol{\sigma}, \boldsymbol{h}) = 1] &\leq \Pr[\mathsf{sk} \leftarrow \mathcal{E}] + \delta_1 + \delta_2 \\ &+ \max_{M_1, M_2, M_3} \left(\frac{2L-2}{|\mathbb{K}|}\right)^{M_1} \cdot \left(\frac{1}{|\mathbb{F}_{2^8}|}\right)^{M_2} \cdot \left(\frac{1}{N}\right)^{M_3}\end{aligned}$$

where the bound $\delta_1 + \delta_2 \leq 34\ell q/\sqrt{2^n} + 2365q^3/2^n$ is given by Theorem 4. Since our extractor extracts $\ell = M \cdot N$ points, the Theorem statement follows. □

C.2 Simulation Soundness

The Helium signature scheme is obtained by applying the Fiat-Shamir transform on the Helium proof system. Simulation-soundness of the Fiat-Shamir transform in the QROM was first shown by Unruh [27, Theorems 22 and 24 of the full version]. More precisely, Unruh showed that if a 3-message public-coin proof system is sound (respectively extractable), then the non-interactive proof system obtained by applying the Fiat-Shamir transform is simulation-sound (respectively simulation-sound extractable). It was observed in [12] that Unruh's result extends to Fiat-Shamir applied to multi-round interactive proofs.