Digital Signatures from Symmetric-Key Primitives

David Derler[1,‡], Claudio Orlandi[2,‖], Sebastian Ramacher[1,‡], Christian Rechberger[1,3,‡,§], and Daniel Slamanig[1,‡]

[1] IAIK, Graz University of Technology, Austria
[2] Aarhus University, Denmark
[3] DTU Compute, DTU, Denmark
`{firstname.lastname}@iaik.tugraz.at, orlandi@cs.au.dk`

Abstract. We propose practically efficient signature schemes which feature several attractive properties: (a) they only rely on the security of symmetric-key primitives (block ciphers, hash functions), and are therefore a viable candidate for *post-quantum* security, (b) they have extremely small signing keys, essentially the smallest possible, and, (c) they are highly parametrizable.

For this result we take advantage of advances in two very distinct areas of cryptography. The first is the area of primitives in symmetric cryptography, where recent developments led to designs which exhibit an especially low number of multiplications. The second is the area of zero-knowledge proof systems, where significant progress for efficiently proving statements over general circuits was recently made.

We follow two different directions, one of them yielding the first practical instantiation of a design paradigm due to Bellare and Goldwasser without relying on structured hardness assumptions. For both our schemes we explore the whole design spectrum to obtain optimal parameter choices for different settings. Within limits, in all cases our schemes allow to trade-off computational effort with signature sizes. We also demonstrate that our schemes are parallelizable to the extent that they can practically take advantage of several cores on a CPU.

Keywords: post-quantum cryptography ⋄ signatures ⋄ block cipher ⋄ Fiat-Shamir ⋄ Bellare-Goldwasser ⋄ implementation

1 Introduction

More than two decades ago Shor published his polynomial-time quantum algorithm for factoring and computing discrete logarithms [Sho94]. Since then, we know that a sufficiently powerful quantum computer is able to break nearly all public key cryptography used in practice today and essentially destroys confidentiality and integrity of all our digital communication. Clearly, this necessitates

‡ Supported by EU H2020 project PRISMACLOUD, grant agreement n°644962.
‖ Supported by COST Action IC1306.
§ Supported by EU H2020 project PQCRYPTO, grant agreement n°645622.

the invention of cryptographic schemes that remain secure in the advent of such an up to now hypothetical quantum computer—especially when considering the steady progress in constructing real-world quantum computers within the last two decades.[4]

Today experts conjecture the advent of sufficiently powerful quantum computers within the next few decades. Although the statements seem rather speculative and far from certain, it is important to start the transition to post-quantum cryptography early enough to eventually not end up in a rush. Therefore, NIST recently announced a post-quantum crypto project[5] seeking for public key encryption, key exchange and digital signatures basing their security on problems being assumed to be secure in a post-quantum era. A call for submissions to this competition is scheduled for late 2017.

In this paper we are concerned with constructing signature schemes. The building blocks of our schemes are interactive honest-verifier zero-knowledge proof systems (Σ-protocols) and symmetric key primitives, which are conjectured to be secure also in a post-quantum world. We believe that a diversity in the used assumptions within the zoo of (post-quantum) signature schemes is important and increases the chance to obtain secure schemes despite cryptanalytic progress. As our focus is on provably secure schemes (i.e., ones that come with a reductionist security proof), we only compare our results to other provably secure candidates. Furthermore, we stress that although our approach is also of interest to the classical setting, we focus on a comparison with post-quantum signature candidates.

Post-Quantum Signatures. One-time signatures for a post-quantum world can be built when instantiating the approach due to Lamport [Lam79] using hash functions (aka hash-based signatures). As Grover's quantum algorithm to invert any black-box function [Gro96] gives a quadratic speed-up, this requires to double the bitsize of the hash function's domain, but requires no additional assumptions to provably achieve post-quantum security. Combined with Merkle-trees, this approach yields stateful signatures for any polynomial number of messages [Mer89] and combined with several other tricks and optimizations (cf. [Gol86]) this yields practical stateless hash-based signatures secure in the standard model [BHH$^+$15]. Code-based signature schemes can be obtained from identification schemes based on the syndrome decoding (SD) problem [Ste93, Vér96, MGS11] by applying a variant of the well known Fiat-Shamir (FS_Σ) transform [FS86]. Lattice-based signature schemes secure under the short integer solution (SIS) problem on lattices following the Full-Domain-Hash paradigm [BR93] have been introduced in [GPV08].[6] More efficient approaches [Lyu09, Lyu12, BG14, ABB$^+$15] rely on the FS_Σ transform instead of FDH. The entirely practical (and indeed highly efficient) scheme BLISS [DDLL13] also relies on the FS_Σ transform, but buys efficiency at the cost

[4] `https://www.technologyreview.com/s/600715/nsa-says`

[5] `http://csrc.nist.gov/groups/ST/post-quantum-crypto/index.html`

[6] Lattice-based signatures have actually been proposed far earlier [GGH97], but this approach turned out to be insecure.

of more pragmatic security assumptions, i.e., a ring version of the SIS problem. For signatures based on problems related to multivariate systems of quadratic equations only recently provably secure variants relying on the FS_{Σ} transform have been proposed in [HRSS16].

When it comes to trust, among all these existing approaches, arguably, hash-based signatures are preferable. All practical signatures supporting arbitrary message lengths need to rely on collision-resistant hash functions anyway, but in addition to that need to rely on some structured assumption. In the post-quantum setting these assumption are related to lattices, codes or multivariate systems of quadratic equations (MQ). Our approach, like state-of-the-art hash-based signatures, also only requires symmetric primitives like hash functions and pseudorandom functions (PRFs) and we also require *no* additional structured assumptions.

On the Tightness of Security Proofs. When investigating (post-quantum secure) signature schemes, it is important and increasingly popular to assess the security of a scheme in a concrete [BR96] instead of an asymptotic setting. Thereby, the goal is to tightly relate the security of the signature scheme to that of the underlying problem. Tightness is important because it allows the implemented scheme to use small parameters (i.e., being as efficient as possible) while retaining provable security. The common way to assess the security of a cryptographic primitive is to perform a reductionist security proof. Thereby, one constructs a reduction algorithm $\mathcal{R}$ showing that any hypothetical algorithm $\mathcal{A}$ (the adversary) achieving a well defined adversarial goal (given by the security notion) yields a solver for a presumably hard problem, thus contradicting the hardness of this problem. Concrete security analysis now relates the runtime t of the hypothetical adversary $\mathcal{A}$ as well as its success probability ϵ to the runtime t' and success probability ϵ' of the reduction $\mathcal{R}$. A reductionist security proof is called *tight*, if $t' \approx t$ and $\epsilon' \approx \epsilon$ and *non-tight* if $t' \gg t$ or $\epsilon' \ll \epsilon$. Obviously, the desirable goal is to come up with a tight proof, as such a proof says that breaking the security of the primitive (within the respective model) is at least as hard as breaking the problem. A non-tight reduction, however, gives only much weaker guarantees and the primitive needs to be instantiated with larger security parameters to account for the non-tightness of the proof. This, in turn, often yields a noticeable performance penalty. Tightness has been studied for standard model signatures [HJ12, BHH$^+$15, BKKP15, BL16], various schemes in the ROM [BR96, GJ03, KW03] and generic compilers from identification to signature schemes (where FS is one of them) [AFLT12, ABP13, BPS16, KMP16].

When looking at existing post-quantum candidates, the state-of-the-art hash-based signature scheme SPHINCS [BHH$^+$15] comes with a tight-reduction. All other practical code-based, lattice-based and MQ-based schemes discussed above use the FS_{Σ} transform and require a rewinding extractor (essentially the application of the forking lemma [PS96]) and thus come with a non-tight reduction losing at least a factor $O(Q_H^n)$ (cf. [CK16]), where Q_H is the number of queries to the random oracle (RO) H and n the number of rewinds, in the overall security. We illustrate the issue with non-tight security using the recent MQ-based post-quantum candidate from ASIACRYPT'16 [HRSS16]. Their security reduction requires three rewinds of the adversary, yielding a multiplicative reduction loss of $\approx Q_H^3$. Considering an 128-bit security level and assuming $Q_H = 2^{60}$ (a very reasonable assumption already made 20 years ago in [BR96]), no provable security margin remains. In fact, one would have to increase the security level by 180 bits to 308 bit to obtain a scheme being provably secure at the 128-bit level.

Contribution. For post-quantum signature schemes it is most desirable to have security proofs in the quantum-accessible random oracle model (QROM; cf. [BDF+11]), or even in the standard model. However, these strong models often impact the practicality of the schemes, which is why all existing practical candidates except for hash-based signatures come with a security proof in the random oracle model (ROM; cf. Section 7 for a more detailed discussion).

- We contribute a novel class of a post-quantum signature schemes to the existing zoo of post-quantum signatures. As with hash-based signatures, our approach only requires symmetric key primitives like hash functions and PRFs and *does not* require additional structured hardness assumptions as all the remaining candidates do. Our proofs of security are in the ROM.
- We present two concrete approaches to construct signature schemes from Σ-protocols for statements expressed as arithmetic circuits. In particular, we present a first construction via the FS_Σ transform and a second novel approach, which constitutes the first practically efficient instantiation of a design paradigm for signatures due to Bellare and Goldwasser [BG89] proposed more than 25 years ago.[7] Together with our contribution, latter opens up a new direction for the design of practical signature schemes. While the approach relying on FS_Σ yields non-tight security reductions, we show how to obtain a tight security reduction for the second approach by carefully crafting the security proof of the scheme. This allows us to tightly relate the security of the signature scheme to the security parameter of the used symmetric primitive.
- We review existing symmetric primitives with respect to their suitability to be used in our application and conclude that the LowMC family of block ciphers [ARS+15, ARS+16] is well suited. We explore the entire design space of the block cipher family LowMC and show that we can obtain various trade-offs between signature sizes and computation times. Thereby, our approach turns out to be very flexible as besides the aforementioned trade-offs we are also able to adjust the security parameter of our construction in a very fine-grained way.
- We provide an implementation of both schemes for 128 bit security in the pre- as well as post-quantum setting, demonstrating the practical relevance of our approach. Moreover, we rigorously compare our schemes with other practical provably secure post-quantum schemes.

[7] We note that [BKP14] used this paradigm in the algebraic setting, whereas, to the best of our knowledge, we are the first to instantiate this paradigm without structured hardness assumptions.

1.1 Outline

Our goal in this paper is to design signature schemes that solely rely on the security of symmetric key primitives. In doing so we construct non-interactive zero-knowledge (NIZK) proof based signatures building upon the approach—termed ZKBoo—taken by Giacomelli et al. in [GMO16]. Their approach builds upon the MPC-in-the-head paradigm by Ishai et al. [IKOS07] and yields practically efficient Σ-protocols for statements expressible via arithmetic circuits, and their results show that proving pre-images of hash functions from the SHA-family results in reasonable performance. Subsequently, we briefly sketch the ideas behind the two directions we pursue to obtain secure signatures, i.e., signatures that are provably existentially unforgeable under adaptively chosen message attacks (EUF-CMA).

Fish. Our first scheme builds upon the well-known FS_{Σ} transform to turn Σ-protocols into EUF-CMA-secure signature schemes. Loosely speaking, the public key is the evaluation of a one-way function (OWF) on a random key. The signature is the transcript of the FS_{Σ}-transformed non-interactive version of the ZKBoo protocol demonstrating knowledge of the respective key. Unfortunately, security proofs for such schemes require rewinding/application of the forking lemma and are thus non-tight (they lose at least a multiplicative factor Q_H, being the number of RO queries, in the success probability). Consequently, one cannot tightly relate the security of the scheme to the security parameter of the underlying OWF and when it comes to a concrete choice of parameters one has to compensate the loss by choosing larger parameters.

Begol. The second scheme builds upon a paradigm to construct signatures that has been proposed more than 25 years ago as a theoretical approach by Bellare and Goldwasser [BG89].[8] Loosely speaking, the public key is an encryption of a PRF key and the signature is an evaluation of a PRF on the message under the key together with a NIZK proof confirming that the key used for the PRF is the one in the ciphertext in the public key. Although the resulting signatures are larger than the ones from the first approach, we can tightly relate the security of the scheme to the security of the underlying PRF and encryption scheme, respectively. When adjusting the parameters to compensate the non-tightness of the first approach, we then obtain two comparable approaches regarding computational efficiency.

Big Picture. Figure 1 sketches the involved building blocks in our constructions. In a nutshell, in our first approach, we use the FS_{Σ} transform to directly turn the ZKBoo protocol for the languages of pre-images of a OWF into the EUF-CMA secure signature scheme Fish. In contrast, when following the Bellare-Goldwasser paradigm, we turn the ZKBoo protocol into a NIZK proof system (henceforth dubbed NIZKBoo) using another variant of the Fiat-Shamir transform ($\mathrm{FS}_{\mathsf{NIZK}}$). Together with a fixed-value-key-binding PRF we then obtain the EUF-CMA secure signature scheme Begol.

[8] To the best of our knowledge this paradigm has only once been used implicitly in [BKP14] but never explicitly to construct practical signature schemes.

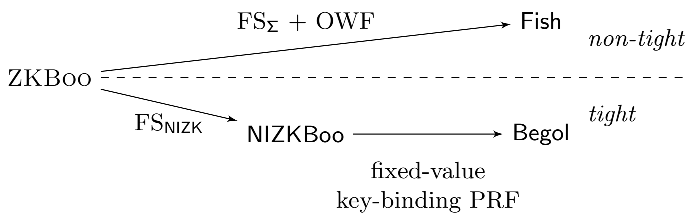


Fig. 1. Overview of our constructions.

Technical Perspective. It is well known that (without assuming any specific properties of the underlying Σ-protocol), the FS_Σ yields a non-tight reduction. We revisit this approach in Section 3. Since we currently do not see how to bypass this loss (i.e., avoid rewinding), our main focus from a technical perspective lies on a tight security reduction for Begol. Our result in this context is summarized by the following theorem.

Theorem 1. *Let* NIZKBoo *denote the* ZKBoo *protocol [GMO16] turned into a NIZK proof system via* FS_{NIZK} *relative to RO* $H : \mathsf{A} \times \mathsf{X} \to \{0,1,2\}^\gamma$*, where the commitments are instantiated via a RO* $H' : \{0,1\}^\alpha \times \{0,1\}^\beta \times \{0,1\}^\nu \to \{0,1\}^\rho$ *so that* $\{0,1\}^\nu$ *is sampled uniformly at random for every commitment. Furthermore, let* Q_S *be the number of signing queries, and let* Q_H *and* $Q_{H'}$ *be the number of queries to the ROs. If the Bellare-Goldwasser paradigm [BG89] is instantiated with a* $(t_{\mathsf{p}}, \epsilon_{\mathsf{p}})$*-secure PRF which is fixed-value-key-binding [Fis99] with respect to some value* β*, and* NIZKBoo*, then Begol is* (t, ϵ)-EUF-CMA *secure in the ROM, where*

$$\epsilon(\kappa) \leq \overbrace{\epsilon_{\mathsf{p}}(\kappa) + \underbrace{\frac{Q_{H'}^2}{2^\rho} + (2/3)^\gamma}_{\textit{Lemma 6}} + \underbrace{\frac{\gamma}{2^\nu} + \frac{Q_S \cdot Q_H}{2^{3\cdot\rho}}}_{\textit{Lemma 7}}}^{\textit{Lemma 4}}, \textit{ and } \underbrace{t \approx t_{\mathsf{p}}}_{\textit{Lemma 4}}.$$

Throughout the next sections, we will prove the theorem above by proving Lemma 4-7. In Section 4 we first recall the Bellare-Goldwasser [BG89] approach and then prove Lemma 4, which resembles the results of Bellare and Goldwasser but details the required security guarantees in a setting without structured hardness assumptions. In Section 4.1 we present the instantiation of NIZKBoo and prove Lemma 6 and Lemma 7. In Lemma 6 we prove soundness of NIZKBoo without requiring to rewind the adversary. This yields a tighter bound than when using s-special soundness of the underlying ZKBoo protocol as a starting point. Finally, in Lemma 7, we prove zero-knowledge of NIZKBoo. Our proof for this lemma is similar to the general result for Σ-protocols in [FKMV12], but we require an additional technicality to account for the fact that the outputs of the simulator in ZKBoo are only statistically close to an original transcript (as opposed to the identical distribution in [FKMV12]). In addition we also provide explicit bounds.

Finally, we note that our implementation of Begol makes use of computational fixed-value-key-binding PRF families for practical reasons. For this setting we currently only have a non-tight proof (cf. Lemma 5).

Implementation Perspective. From an implementation point of view our findings are twofold. Firstly, we review existing symmetric primitives with respect to their suitability to be employed in our application. We conclude that the LowMC family of block ciphers [ARS+15, ARS+16] provides favorable properties in this context (cf. Section 5). Secondly, we present a highly optimized implementation which uses SIMD instruction sets of modern CPU architectures and also exploits parallelization using standard APIs for parallel programming. Furthermore, we explore the whole design space of LowMC to find the most suitable parametrizations. Our results confirm that instantiations of both proposed signature schemes are entirely practical. A detailed overview of our implementation results is provided in Section 6.

1.2 Related Work

In this section we give a brief overview of other candidate schemes and defer a detailed comparison of parameters as well as performance figures to Section 6. We start with the only existing instantiation that only relies on standard assumptions, i.e., comes with a security proof in the standard model (SM) and does not require the ROM idealization. The remaining existing schemes rely on structured assumptions related to codes, lattices and multivariate systems of quadratic equations that are assumed to be quantum-immune and have a security proof in the ROM. By the end of the section, we review the state of the art in zero-knowledge proofs for non-algebraic statements.

Hash-Based Signatures (SM). Hash-based signatures are attractive in the sense that they can be proven secure in the standard model (i.e., without requiring ROs) under well-known properties of hash functions such as second preimage resistance. Unfortunately, efficient schemes like XMSS [BDH11] are stateful, a property which seems to be problematic for practical applications. Proper state management is not yet well understood [MKF+16] and seems to be a property that is desirable to omit. Stateless schemes like the most recent proposal SPHINCS [BHH+15] are thus more desirable, but are more inefficient in terms of lower speed and increased signature size. SPHINCS has a security reduction that tightly relates its EUF-CMA security to the security of the used building blocks, i.e., hash functions, PRGs and PRFs. On a 128 bit post-quantum security level (SPHINCS-256), one obtains a signature size of about 41 kB, and a public as well as a private key size of about 1 kB each.

Code-Based Signatures (ROM). In the code-based setting the most well-known and provably secure approach is to convert identification schemes due to Stern [Ste93] and Véron [Vér96] to signatures using FS_Σ. Consequently, the reductions require rewinding and are not tight. For the 128 bit security level (not accounting for the loss induced by the reduction) and accounting for Grover one obtains signature sizes of around $\approx$ 75 kB (in the best case) and public key size of ≈ 80 bytes.[9] We note that there are also other code-based signatures [CFS01] based on the Niederreither [Nie86] dual of the McEliece cryptosystem [McE78], which do not come with a security reduction, have shown to be insecure [FGO+13] and also do not seem practical even for very low security levels [LS12]. There is a more recent approach with provable security [ELL+15], however, it is not immediate if this leads to efficient signatures.

Lattice-Based Signatures (ROM). In context of lattice based signatures there are two major directions. The first are schemes that rely on the hardness of worst-to-average-case problems in standard lattices [GPV08, Lyu12, BG14, DBG+14, ABB+15]. Although they are desirable from a security point of view, they suffers from huge public keys, i.e., sizes in the orders of a few to some 10 MBs. The GPV scheme [GPV08] comes with a tight security reduction. Also TESLA[10] [ABB+15] (based upon [BG14, Lyu12]), using the proof strategy of Katz and Wang[11] [KW03], has a tight reduction to the learning with error (LWE) problem on lattices. TESLA improves all aspects in the performance of GPV, but still has keys in the order of 1 MB. More efficient lattice-based schemes are based on ring analogues of classical lattice problems [GLP12, DDLL13, BB13, ABB+16, BLN+16] whose security is related to hardness assumptions in ideal lattices. These constructions allow to drop key sizes to the order of a few kBs. Most notable is BLISS [DDLL13, Duc14], which achieves very good performance nearly comparable to RSA. However, it has a non-tight security reduction as it uses the classical FS_Σ transform. It must also be noted, that ideal lattices have not been investigated nearly as deeply as standard lattices and thus there is less confidence in the related hardness conjectures (cf. [Pei16]).

MQ-Based Signatures (ROM). Recently, Hülsing et al. in [HRSS16] proposed a post-quantum signature scheme (MQDSS) whose security is based on the problem of solving a multivariate system of quadratic equations. Their scheme is obtained by building upon the 5-pass (or 3-pass) identification scheme in [SSH11] and applying the FS_Σ transform. Their security proof requires a rewinding extractor and thus yields a non-tight security reduction in the ROM. For 128 bit post-quantum security (which does not account for the reduction loss) signature sizes are about 40 kB, public key sizes are 72 bytes and secret key sizes are 64 bytes. We note that there are other MQ-based approaches like Unbalanced Oil-and-Vinegar (UOV) variants [PCG01] or FHEv$^-$ variants (cf. [PCY+15]), having

[9] The given estimations are taken from a recent talk of Nicolas Sendrier (available at `https://pqcrypto.eu.org/mini.html`), as, unfortunately, there are no free implementations available.

[10] The authors of TESLA also show how to adapt their security reduction to the QROM, but do not implement their benchmarks accordingly and say: *"We believe that the need for a chameleon hash function is merely an artifact of the proof and we would be surprised if our decision to use SHA-256 could be exploited in an attack."*

[11] This strategy allows to obtain tight reductions when relying on decisional assumptions on the public key: if the instance is valid then we have a real key and in case of a fake key an adversary can only guess a valid signature with negligible probability. This idea has later been generalized by Abdalla et al. [AFLT12] via the notion of lossy identification schemes.

somewhat larger keys (order of kBs) but much shorter signatures. However, they have no provable security guarantees, the parameter choice seems very aggressive, there are no parameters for conservative (post-quantum) security levels, and no implementations are available.

Zero-Knowledge for Arithmetic Circuits. Zero-knowledge proofs [GMR85] are a very powerful tool and exist for any language in **NP** [GMW86]. Nevertheless, practically efficient proofs, were until recently only known for restricted languages and in particular for proving algebraic statements in certain algebraic structures, e.g., discrete logarithms [Sch91, CDS94] or equations over bilinear groups [GS08]. Expressing any **NP** language as a combination of algebraic circuits could be done for example by expressing the relation as a circuit, i.e., express each gate as an algebraic relation between input and output wires. However, for circuits interesting for practical applications (such as hash functions or block ciphers), this gets prohibitive. Even SNARKS, where proof size can be made small (and constant) and verification is highly efficient, have very costly proofs (cf. [GGPR13, BCG+13, CFH+15] and the references therein). Using SNARKS is reasonable in scenarios where provers are extremely powerful (such as verifiable computing [GGPR13]) or the runtime of the prover is not critical (such as Zerocash [BCG+14]). Unfortunately, signatures require small proof computation times (efficient signing procedures), and this direction is not suitable.

Quite recently, dedicated zero-knowledge proof systems for statements expressed as Boolean circuits by Jawurek et al. [JKO13] and statements expressed as RAM programs by Hu et al. [HMR15] have been proposed. As we exclusively focus on circuits below lets take a look at Jawurek et al. [JKO13]. They proposed to use garbled circuits to obtain zero-knowledge proofs, which allow to efficiently prove statements like knowledge of a pre-image under a hash function like SHA-256. Unfortunately, this approach is inherently interactive and thus not suitable for the design of practical signature schemes. The very recent ZKBoo protocol due to Giacomelli et al. [GMO16], for the first time, allows to construct Σ-protocols with performance being of interest for many practical applications. We use their approach as a starting point for our work.

2 Building Blocks

Before we start, we recall the required building blocks. Henceforth, we denote algorithms by sans-serif letters, e.g., A, B. If not stated otherwise, all algorithms are required to run in polynomial time and return a special symbol $\perp$ on error. By $y \leftarrow \mathsf{A}(x)$, we denote that y is assigned the output of the potentially probabilistic algorithm A on input x and fresh random coins. Similarly, $y \xleftarrow{R} S$ means that y is sampled uniformly at random from a set S. We use $\langle \mathsf{A}, \mathsf{B} \rangle$ to denote a pair of interactive algorithms and use $\langle \mathsf{A}, \mathsf{B} \rangle = 1$ to denote an interaction where both algorithms accept. We let $[n] := \{1, \ldots, n\}$ and use $\|_{i \in [n]} x_i$ as a shorthand notation for $x_1 \| \ldots \| x_n$. We write $\Pr[\Omega : \mathcal{E}]$ to denote the probability of an event $\mathcal{E}$ over the probability space Ω. We use $\mathcal{C}$ to denote challengers of security experiments, and $\mathcal{C}_\kappa$ to make the security parameter explicit. A function $\varepsilon(\cdot) : \mathbb{N} \to \mathbb{R}_{\geq 0}$ is called negligible, iff it vanishes faster than every inverse polynomial, i.e., $\forall\ k : \exists\ n_k : \forall\ n > n_k : \varepsilon(n) < n^{-k}$. Finally, we use $\mathsf{poly}(\cdot)$ to denote a polynomial function.

We postpone the presentation of standard primitives such as OWFs $f : D \to \mathsf{R}$, PRFs $F : \mathcal{S} \times D \to \mathsf{R}$ and signature schemes $\Sigma = (\mathsf{Gen}, \mathsf{Sign}, \mathsf{Verify})$ to Appendix A.

Σ-Protocols. Let $L \subseteq \mathsf{X}$ be an **NP**-language with associated witness relation R so that $L = \{x \mid \exists w : R(x, w) = 1\}$. A Σ-protocol for language L is defined as follows.

Definition 1. *A Σ-protocol for language L is an interactive three-move protocol between a PPT prover $\mathsf{P} = (\mathsf{Commit}, \mathsf{Prove})$ and a PPT verifier $\mathsf{V} = (\mathsf{Challenge}, \mathsf{Verify})$, where P makes the first move and transcripts are of the form $(\mathsf{a}, \mathsf{e}, \mathsf{z}) \in \mathsf{A} \times \mathsf{E} \times \mathsf{Z}$. Additionally they satisfy the following properties*

Completeness. *A Σ-protocol for language L is complete, if for all security parameters κ, and for all $(x, w) \in R$, it holds that*

$$\Pr[\langle \mathsf{P}(1^\kappa, x, w), \mathsf{V}(1^\kappa, x)\rangle = 1] = 1.$$

s-Special Soundness. *A Σ-protocol for language L is s-special sound, if there exists a PPT extractor E so that for all x, and for all sets of accepting transcripts $\{(\mathsf{a}, \mathsf{e}_i, \mathsf{z}_i)\}_{i\in[s]}$ with respect to x, generated by any algorithm with polynomial runtime in κ, it holds that*

$$\Pr\left[w \leftarrow \mathsf{E}(1^\kappa, x, \{(\mathsf{a}, \mathsf{e}_i, \mathsf{z}_i)\}_{i\in[s]}) \ \ : \ \ \begin{array}{r} (x, w) \in R\ \wedge \\ \forall i, j \in [s], i \neq j : \mathsf{e}_i \neq \mathsf{e}_j \end{array} \right] \geq 1 - \epsilon(\kappa).$$

Special Honest-Verifier Zero-Knowledge. *A Σ-protocol is special honest-verifier zero-knowledge, if there exists a PPT simulator S so that for every $x \in L$ and every challenge e from the challenge space, it holds that a transcript $(\mathsf{a}, \mathsf{e}, \mathsf{z})$, where $(\mathsf{a}, \mathsf{z}) \leftarrow \mathsf{S}(1^\kappa, x, \mathsf{e})$ is computationally indistinguishable from a transcript resulting from an honest execution of the protocol.*

The s-special soundness property gives an immediate bound for the soundness of the protocol: if no witness exists then (ignoring a negligible error) the prover can successfully answer at most to ${}^{s-1}/_{t}$ challenges, where $t = |\mathsf{E}|$ is the size of the challenge space. In case this value is too large, it is possible to reduce the soundness error using well known properties of Σ-protocols which we restate here for completeness.

Lemma 1. *The properties of Σ-protocols are invariant under parallel repetition. In particular, the ℓ fold parallel repetition of a Σ-protocol for relation R with challenge length t yields a new Σ-protocol with challenge length ℓt.*

Lemma 2. *If there exists a Σ-protocol for R with challenge length t, then there exists a Σ-protocol for R with challenge length t' for any t'.*

Below, we recall another well known fact about the AND composition of Σ-protocols.

Lemma 3. *Let L_1 and L_2 be two languages with associated witness relations R_1 and R_2, respectively. Further, let Σ_1 and Σ_2 be two Σ-protocols with identical challenge space so that Σ_1 is for L_1 and Σ_2 is for L_2. Then a Σ-protocol for the conjunction of L_1 and L_2, i.e., $L_1 \wedge L_2 \coloneqq \{(x_1, x_2) \mid \exists\ w_1, w_2 : (x_1, w_1) \in L_1\ \wedge\ (x_2, w_2) \in L_2\}$ is obtained by running Σ_1 and Σ_2 in parallel using a single common challenge* e.

Finally, we note that an equality (EQ) composition of Σ-protocols resulting in a language $L = \{(x_1, x_2) \mid \exists\ w : (x_1, w) \in R_1\ \wedge\ (x_2, w) \in R_2\}$ can be achieved as a special case of an AND composition, where besides an identical challenge e also the same random tape is used for the prover in both instances.

The Fiat-Shamir Transform. The Fiat-Shamir ($\mathrm{FS}_{\mathsf{NIZK}}$) transform [FS86] is a frequently used tool to convert Σ-protocols $\langle \mathsf{P}, \mathsf{V} \rangle$ to their non-interactive counterparts. Essentially, the transform removes the interaction between P and V by using a RO $H : \mathsf{A} \times \mathsf{X} \rightarrow \mathsf{E}$ to obtain the challenge e.[12] That is, one uses a PPT algorithm $\mathsf{Challenge}'(1^\kappa, \mathsf{a}, x)$ which obtains $\mathsf{e} \leftarrow H(\mathsf{a}, x)$ and returns e. Then, the prover can locally obtain the challenge e *after* computing the initial message a. Starting a verifier $\mathsf{V}' = (\mathsf{Challenge}', \mathsf{Verify})$ on the same initial message a will then yield the same challenge e. More formally, we obtain the non–interactive PPT algorithms $(\mathsf{P}_H, \mathsf{V}_H)$ indexed by the used RO:

$\mathsf{P}_H(1^\kappa, x, w)$: Start P on $(1^\kappa, x, w)$, obtain the first message a, answer with $\mathsf{e} \leftarrow H(\mathsf{a}, x)$, and finally obtain z. Returns $\pi \leftarrow (\mathsf{a}, \mathsf{z})$.

$\mathsf{V}_H(1^\kappa, x, \pi)$: Parse π as (a, z). Start V' on $(1^\kappa, x)$, send a as first message to the verifier. When V' outputs e, reply with z and output 1 if V accepts and 0 otherwise.

Non-Interactive ZK Proof Systems. Now, we recall a standard definition of non-interactive zero-knowledge proof systems. Therefore, let L be an **NP**-language with witness relation R, i.e., so that $L = \{x \mid \exists\ w : R(x, w) = 1\}$.

Definition 2 (Non-Interactive Zero-Knowledge Proof System). *A non-interactive proof system* Π *is a tuple of algorithms* $(\mathsf{Setup}, \mathsf{Proof}, \mathsf{Verify})$*, which are defined as follows:*

$\mathsf{Setup}(1^\kappa)$: *This algorithm takes a security parameter κ as input, and outputs a common reference string* crs.

$\mathsf{Proof}(\mathsf{crs}, x, w)$: *This algorithm takes a common reference string* crs*, a statement x, and a witness w as input, and outputs a proof π.*

$\mathsf{Verify}(\mathsf{crs}, x, \pi)$: *This algorithm takes a common reference string* crs*, a statement x, and a proof π as input, and outputs a bit $b \in \{0, 1\}$.*

[12] This is a stronger variant of FS (cf. [FKMV12, BPW12]). The original weaker variant of the FS transform does not include the statement x in the challenge computation.

We require the properties *completeness*, *adaptive zero-knowledge*, *soundness*, and *simulation sound extractability*. In Appendix A.1 we recall formal definition of those properties.

Converting Σ-protocols to Non-Interactive Proof Systems. One can obtain a non-interactive proof system satisfying the properties above by applying the $\mathrm{FS}_{\mathsf{NIZK}}$ transform to any Σ-protocol where the min-entropy α of the commitment a sent in the first phase is so that $2^{-\alpha}$ is negligible in the security parameter κ and the challenge space E is exponentially large in the security parameter. Formally, $\mathsf{Setup}(1^\kappa)$ fixes a RO $H : \mathsf{A} \times \mathsf{X} \to \mathsf{E}$, sets $\mathsf{crs} \leftarrow (1^\kappa, H)$ and returns crs. The algorithms Prove and Verify are defined as follows:

$$\mathsf{Prove}(\mathsf{crs}, x, w) := \mathsf{P}_H(1^\kappa, x, w), \quad \mathsf{Verify}(\mathsf{crs}, x, \pi) := \mathsf{V}_H(1^\kappa, x, \pi).$$

Combining [FKMV12, Thm. 1, Thm. 2, Thm. 3, Prop. 1] (among others) shows that a so-obtained proof system is complete, sound, adaptively zero-knowledge, and simulation sound extractable, if the underlying Σ-protocol is (2-)special sound and the commitments sent in the first move are unconditionally binding. Those properties also generalize to ZKBoo. For the tight-security reduction of Begol, we will later prove concrete bounds for certain properties of ZKBoo.

3 Fish: Signatures from Plain Fiat-Shamir

The FS_{Σ} transform can elegantly be used to convert (canonical) identification schemes into adaptively (EUF-CMA) secure signature schemes. The basic idea is similar to $\mathrm{FS}_{\mathsf{NIZK}}$, but the challenge generation is defined via the RO $H : \mathsf{A} \times \mathcal{M} \to \mathsf{E}$. More precisely, one uses the provers first message a and the message $m \in \mathcal{M}$ to be signed to obtain the challenge. That is, the algorithms P_H and V_H additionally take the message m as input and compute the challenge as $\mathsf{e} \leftarrow H(\mathsf{a}, m)$. There are numerous works on the security of signatures schemes obtained from three move identification schemes [OO98, PS96, AABN02, KMP16, BPS16]. All these approaches to prove security of such constructions either yield non-tight reductions or the techniques to obtain tight reductions are not applicable to our setting (as they require some trapdoor). Essentially, all these techniques rely on rewinding/forking and lose (depending on the number of rewinds) at least a multiplicative factor of Q_H—the number of queries to the RO—in the overall success probability. We note that all above results are for three round schemes and there are also generalizations to schemes of $2k+1$ rounds for $k > 1$ [ADV+12, DGV+16], which also yield non-tight reductions.

The only class of schemes for which the FS_{Σ} conversion yields tight security by means of a black-box transformation are so called lossy identification schemes [AFLT12]. These are identification schemes, which beside normal keys have lossy keys being computationally indistinguishable from real ones and given a lossy key, for an adversary it is statistically impossible to provide a valid response to a random challenge after making a commitment. Such identification schemes are known for various settings (cf. [KW03, AFLT12, ABP13]). While this approach applies when working with lattices [AFLT12] in the post-quantum setting, unfortunately, it does not help us with our approach to obtain tight security.

Now, consider a language L_R with associated witness relation R of pre-images of a OWF $f : D \to \mathsf{R}$:

$$(y, s) \in R \iff y = f(s).$$

When using ZKBoo to prove knowledge of such a pre-image, we know [GMO16] that this Σ-protocol provides 3-special soundness. We apply the FS_Σ transform to this Σ-protocol to obtain an $\mathsf{EUF\text{-}CMA}$ secure signature scheme.

In the so-obtained signature scheme the public verification key pk contains the image y (and a description of f) and the secret signing key sk contains a sufficiently large random element s from R. The corresponding signature scheme, dubbed Fish, is illustrated in Scheme 1.

$\mathsf{Gen}(1^\kappa)$: Choose $\mathsf{s} \xleftarrow{R} D$, compute $y \leftarrow f(\mathsf{s})$, set $\mathsf{pk} \leftarrow y$ and $\mathsf{sk} \leftarrow (\mathsf{pk}, \mathsf{s})$ and return $(\mathsf{sk}, \mathsf{pk})$.

$\mathsf{Sign}(\mathsf{sk}, m)$: Parse sk as $(\mathsf{pk}, \mathsf{s})$, compute $\pi = (\mathsf{a}, \mathsf{z}) \leftarrow \mathsf{P}_H(1^\kappa, (y, m), \mathsf{s})$ and return $\sigma \leftarrow \pi$, where internally the challenge is computed as $\mathsf{e} \leftarrow H(\mathsf{a}, m)$.

$\mathsf{Verify}(\mathsf{pk}, m, \sigma)$: Parse pk as y, and σ as $\pi = (\mathsf{a}, \mathsf{z})$. Return 1 if the following holds, and 0 otherwise:

$$\mathsf{V}_H(1^\kappa, (y, m), \pi) = 1,$$

where internally the challenge is computed as $\mathsf{e} \leftarrow H(\mathsf{a}, m)$.

Scheme 1: Fish: building upon Fiat-Shamir.

If we view ZKBoo as a canonical identification scheme that is secure against passive adversaries one just needs to keep in mind that most definitions are tailored to (2-)special soundness, and the 3-special soundness of ZKBoo requires an additional rewind. In particular, an adapted version of the proof of [Kat10, Theorem 8.2] which considers this additional rewind attests the security of Scheme 1. Secondly, using the results of [FKMV12] together with [CL06] (again considering the additional rewind), also immediately yields $\mathsf{EUF\text{-}CMA}$ secure signatures, but additionally requires to include the proven statement x upon computing the challenge. Latter can be viewed as a direct construction from an simulation sound extractable NIZK proof system and therefore we also obtain key-versatile signatures as defined in [BMT14]. Let us stick with the first approach, then we obtain the following:

Corollary 1. *Scheme 1 instantiated with* ZKBoo *and a secure OWF yields an* $\mathsf{EUF\text{-}CMA}$ *secure signature scheme.*

Note that the corollary above holds in the asymptotic sense, i.e., the reductions loses a multiplicative factor of $\approx Q_H^2$, with Q_H being the number of queries to the RO. Thus one has to accordingly adjust the security parameter if concrete security is required.

4 Begol: Instantiating the Bellare-Goldwasser Paradigm

Bellare and Goldwasser in [BG89] introduced an approach to construct adaptively ($\mathsf{EUF\text{-}CMA}$) secure signatures from a NIZK proof system $\Pi = (\mathsf{Setup}, \mathsf{Proof}, \mathsf{Verify})$, a public key encryption scheme and a PRF. In this work however we want to use symmetric primitives only, and therefore we replace the public key encryption scheme with a PRF. This leads to the following issue: in asymmetric encryption schemes, the public key somehow acts as a "commitment" of the secret key, while the standard notion of PRF security does not immediately imply any binding property on the used key i.e., there could be two keys s, s' such that $F_{\mathsf{s}}(x) = F_{\mathsf{s}'}(x)$. We therefore need to assume that our PRF is *fixed-value-key-binding* as defined in [Fis99]. That is, there exists some β such that given $y = F_{\mathsf{s}}(\beta)$ it is hard to come up with s' such that $F_{\mathsf{s}'}(\beta) = y$ (see Definition 5).

Henceforth, let $\mathcal{F}$ be a family of fixed-value-key-binding PRFs $F : \mathcal{S} \times \mathcal{M} \to \{0,1\}^\kappa$ which are key-binding with respect to $\beta \in \mathcal{M}$. The private signing key of the scheme is $\mathsf{sk} := \mathsf{s}$ where $\mathsf{s} \xleftarrow{R} \mathcal{S}$ is a PRF key. The public verification key $\mathsf{pk} := (\mathsf{crs}, c, \beta)$ of the scheme consists of the CRS $\mathsf{crs} \leftarrow \Pi.\mathsf{Setup}(1^\kappa)$ of the NIZK proof system and a key-binding evaluation $c \leftarrow \mathsf{F_s}(\beta)$ of the PRF. Let $L_R := \{x \mid \exists\, w : R(x, w) = 1\}$ with witness relation R be defined as:

$$((c, y, \beta, m), \mathsf{s}) \in R \iff c = F_{\mathsf{s}}(\beta) \quad \wedge \quad y = F_{\mathsf{s}}(m). \tag{1}$$

A signature for a message $m \in \mathcal{M} \setminus \{\beta\}$ in the Bellare-Goldwasser approach is the pair $\sigma := (y, \pi)$. Thereby, $y \leftarrow F_{\mathsf{s}}(m)$ is the evaluation of the PRF using key s, and $\pi \leftarrow \Pi.\mathsf{Proof}(\mathsf{crs}, (c, y, \beta, m), \mathsf{s})$ is a NIZK proof, proving the correct evaluation of the PRF on the message m with respect to unknown PRF key s committed to using the fixed-value-key-binding property of the PRF. In Scheme 2 we formally state the construction, which we term Begol.

$\mathsf{Gen}(1^\kappa)$: Run $\mathsf{crs} \leftarrow \Pi.\mathsf{Setup}(1^\kappa)$, choose $\mathsf{s} \xleftarrow{R} \mathcal{S}$, compute $c \leftarrow F_{\mathsf{s}}(\beta)$, set $\mathsf{pk} \leftarrow (\mathsf{crs}, c, \beta)$ and $\mathsf{sk} \leftarrow (\mathsf{pk}, \mathsf{s})$ and return $(\mathsf{sk}, \mathsf{pk})$.

$\mathsf{Sign}(\mathsf{sk}, m)$: Parse sk as $((\mathsf{crs}, c, \beta), \mathsf{s})$, compute $y \leftarrow F_{\mathsf{s}}(m)$, $\pi \leftarrow \Pi.\mathsf{Proof}(\mathsf{crs}, (c, y, \beta, m), \mathsf{s})$, and return $\sigma \leftarrow (y, \pi)$.

$\mathsf{Verify}(\mathsf{pk}, m, \sigma)$: Parse pk as (crs, c, β), and σ as (y, π). Return 1 if the following holds, and 0 otherwise:

$$\Pi.\mathsf{Verify}(\mathsf{crs}, (c, y, m, \beta), \pi) = 1 \ \wedge \ m \neq \beta.$$

Scheme 2: Begol: building upon Bellare-Goldwasser.

Tight Security. As already mentioned, we are interested to tightly relate the security of our instantiation to the security of the underlying symmetric primitives. The subsequent lemma is tailored to our setting and makes the requirements for the underlying primitives explicit by making use of state-of-the-art security notions. Furthermore, we provide explicit bounds.

Lemma 4. *If $\mathcal{F}$ is a $(t_\mathsf{p}, \epsilon_\mathsf{p})$-secure PRF family which is fixed-value-key-binding with respect to β, Π is ϵ_s-sound, and ϵ_z-adaptively zero-knowledge, then the BG signature scheme is (t, ϵ)-*EUF-CMA *secure, where*

$$\epsilon(\kappa) \leq {}^1/_{2^\kappa} + \epsilon_\mathsf{p}(\kappa) + \epsilon_\mathsf{s}(\kappa) + \epsilon_\mathsf{z}(\kappa), \text{ and } t \approx t_\mathsf{p}.$$

We prove the theorem above in Appendix B. In the next section, we show how to instantiate the required non-interactive zero-knowledge proof-system NIZKBoo from ZKBoo and prove the required bounds for soundness and adaptive zero-knowledge which then completes the proof of Theorem 1.

Security of our Instantiation. For our instantiation of Begol in Section 6, we plug the following lemma into Theorem 1, which allows us to obtain practical instantiations. We, however, note that we cannot circumvent the use of the extractor in the proof for this lemma, which—due to the fact that ZKBoo requires rewinding to extract—yields a concrete bound which is non-tight, i.e., the reductions loses a multiplicative factor of $\approx Q_H^2$, with Q_H being the number of queries to the RO. Thus one has to accordingly adjust the security parameter if concrete security is required.

Lemma 5. *If $\mathcal{F}$ is a $(t_\mathsf{p}, \epsilon_\mathsf{p})$-secure PRF family which is $(t_\mathsf{b}, \epsilon_\mathsf{b})$-computationally fixed-value-key-binding with respect to β, Π is $(t_\mathsf{e}, \epsilon_\mathsf{e})$-simulation sound extractable, and ϵ_z-adaptively zero-knowledge, then the BG signature scheme is (t, ϵ)-*EUF-CMA *secure, where*

$$\epsilon(\kappa) \leq {}^1/_{2^\kappa} + \epsilon_\mathsf{p}(\kappa) + \epsilon_\mathsf{b}(\kappa) + \epsilon_\mathsf{e}(\kappa) + \epsilon_\mathsf{z}(\kappa), \text{ and } t \approx \max\{t_\mathsf{p}, t_\mathsf{b}, t_\mathsf{e}\}.$$

We prove the lemma above in Appendix C.

4.1 Using ZKBoo as Non-Interactive ZK Proof System

In this section, we describe how to turn ZKBoo into a non-interactive zero-knowledge (NIZK) proof system so that it can be used to efficiently instantiate Begol. While ZKBoo is generally applicable to the class of functions which can be decomposed into three branches, so that the values computed in two branches reveal no information about the input to the respective function, we will henceforth focus on Boolean circuits being a subclass of these functions (cf. Section 5 for more details on this choice). Such a decomposition is called $(2,3)$-function decomposition and works as follows.

Linear (2,3)-Function Decomposition for Boolean Circuits [GMO16]. Let $\phi : W \to X$ be a function which can be expressed by a Boolean circuit with a total number N of gates. The $(2,3)$-function decomposition is defined by the following building blocks. First, one requires three random tapes (k_1, k_2, k_3). Second, the Share function computes a linear secret sharing (s_1, s_2, s_3) of the input value $w \in W$ with respect to random tapes (k_1, k_2, k_3). Third, the family $\mathcal{F} = \bigcup_{c=1}^{N} \{\phi_1^{(c)}, \phi_2^{(c)}, \phi_3^{(c)}\}$ is defined as follows. Let $w_i[c], i \in [3]$ denote the output of gate c, let $(w_1[0], w_2[0], w_3[0]) = (s_1, s_2, s_3)$, and let $(x_1, x_2, x_3) =$

$(w_1[N], w_2[N], w_3[N])$. Then, depending on the type of the gate, the functions $\phi_i^{(c)}, i \in [3]$ are defined as follows, where the wires are coming from the gate numbers a and b with $a, b < c$ (only from a for unary gates):

Unary XOR α gate: $w_i[c] = \phi_i^{(c)}(w_i[a]) = \begin{cases} w_i[a] \ \oplus \ \alpha & \text{if } i = 1, \text{ and} \\ w_i[a] & \text{else.} \end{cases}$

Unary AND α gate: $w_i[c] = \phi_i^{(c)}(w_i[a]) = \alpha \ \wedge \ w_i[a]$.

Binary XOR gate: $w_i[c] = \phi_i^{(c)}(w_i[a], w_i[b]) = w_i[a] \ \oplus \ w_i[b]$.

Binary AND gate:

$$\begin{aligned} w_i[c] &= \phi_i^{(c)}(w_i[a,b], w_{i+1}[a,b], k_i, k_{i+1}) \\ &= w_i[a] \ \wedge \ w_i[b] \ \oplus \ w_{i+1}[a] \ \wedge \ w_i[b] \\ &\quad \oplus w_i[a] \ \wedge \ w_{i+1}[b] \ \oplus \ F_{k_i}(c) \ \oplus \ F_{k_{i+1}}(c), \end{aligned}$$

where F_{k_i} and $F_{k_{i+1}}$, respectively, denote PRFs, being sampled from an appropriate family using k_i and k_{i+1}.

NIZK from ZKBoo Σ-Protocol (NIZKBoo). The NIZKBoo protocol is presented in Scheme 3, where we slightly abuse the notation introduced above and assume that the selection of the input wires is hard-coded into the functions $\phi_i^{(c)}$. Therefore, all functions $\phi_i^{(c)}$ take input $(w_i[0, \ldots, c-1], w_{i+1}[0, \ldots, c-1], k_i, k_{i+1})$, i.e., the values of all wires up to gate number $c - 1$, as well as the random tapes k_i and k_{i+1}. Since ZKBoo builds up on commitments in the ROM, we index the algorithms $\mathsf{P}_H, \mathsf{V}_H$ with an additional RO $H' : \{0,1\}^\alpha \times \{0,1\}^\beta \times \{0,1\}^\nu \to \{0,1\}^\rho$. Furthermore, for our illustrations we assume that $H : \mathsf{A} \times \mathsf{X} \to \{0,1,2\}^\gamma$ with $\gamma = 1$, and stress that we can adjust the soundness error arbitrarily by parallel composition, i.e., increasing γ (cf. Lemma 1 and Lemma 2).

Modeling the BG Language. The witness relation in Equation (1) can be seen as the composition of witness relations using an AND and an EQ composition. Technically, we represent the entire statement as a single circuit which represents the parallel evaluation of the PRF circuit where the inputs corresponding to the PRF seed s are re-used. In Section 5 we will discuss how we instantiate both building blocks from a block cipher (LowMC).

Given the completeness of the underlying ZKBoo protocol, the subsequent corollary follows from inspection.

Corollary 2. *Scheme 3 is complete.*

Now, we directly prove the remaining required security properties of NIZKBoo when used to instantiate the Bellare-Goldwasser paradigm. An important observation is that we are dealing with an **NP**-language where soundness actually gives sufficient guarantees for the security of the overall signature scheme and we do not need to be able to extract a witness. This allows us to bypass the requirement for rewinding the adversary, and to obtain a tight security bound.

Lemma 6. *Let κ be the security parameter and $Q_{H'}$ be the number of queries to H'. Then the probability $\epsilon(\kappa)$ for all PPT adversaries $\mathcal{A}$ to break soundness of γ parallel executions of Scheme 3 is bounded by $\epsilon(\kappa) \leq Q_{H'}^2/2^\rho + (2/3)^\gamma$.*

$\mathsf{P}_{H,H'}(1^\kappa, x, w)$: Sample random tapes $(k_1, k_2, k_3) \xleftarrow{R} (\{0,1\}^\alpha, \{0,1\}^\alpha, \{0,1\}^\alpha)$, run $(s_1, s_2, s_3) = (w_1[0], w_2[0], w_3[0]) \leftarrow \mathsf{Share}(w; k_1, k_2, k_3)$. Obtain $(\mathbf{w}_1, \mathbf{w}_2, \mathbf{w}_3) = ((w_1[i], w_2[i], w_3[i]))_{1<i\leq N}$, where

$$(w_j[i] \leftarrow \phi_j^{(i)}(w_j[0,\ldots,i-1], w_{j+1}[0,\ldots,i-1], k_j, k_{j+1})_{j\in[3],1<i\leq N},$$

and let $(x_1, x_2, x_3) = (w_1[N], w_2[N], w_3[N])$. For $i \in [3]$, compute $c_i \leftarrow H'(k_i, \mathbf{w}_i, r_i)$, where $r_i \xleftarrow{R} \{0,1\}^\nu$. Then, set $\mathsf{a} \leftarrow (x_1, x_2, x_3, c_1, c_2, c_3)$, compute $\mathsf{e} \leftarrow H(\mathsf{a}, x)$, set $\mathsf{z} \leftarrow (k_\mathsf{e}, k_{\mathsf{e}+1}, \mathbf{w}_\mathsf{e}, \mathbf{w}_{\mathsf{e}+1}, r_\mathsf{e}, r_{\mathsf{e}+1})$ and return $\pi \leftarrow (\mathsf{a}, \mathsf{z})$.

$\mathsf{V}_{H,H'}(1^\kappa, x, \pi)$: Parse π as (a, z), compute $\mathsf{e} \leftarrow H(\mathsf{a}, x)$, and check whether $x = x_1 \oplus x_2 \oplus x_3$, if $x_i = w_i[N]$ for $i \in \{\mathsf{e}, \mathsf{e}+1\}$ and if the following holds for all $1 < i \leq N$:

$$w_\mathsf{e}[i] = \phi_\mathsf{e}^{(i)}(w_\mathsf{e}[0,\ldots,i-1], w_{\mathsf{e}+1}[0,\ldots,i-1], k_\mathsf{e}, k_{\mathsf{e}+1}) \wedge$$
$$c_\mathsf{e} = H'(k_\mathsf{e}, \mathbf{w}_\mathsf{e}, r_\mathsf{e}) \ \wedge\ c_{\mathsf{e}+1} = H'(k_{\mathsf{e}+1}, \mathbf{w}_{\mathsf{e}+1}, r_{\mathsf{e}+1}).$$

If all checks hold, return 1 and 0 otherwise.

Scheme 3: $\mathsf{NIZKBoo}$ for L_ϕ with decomposition $\mathsf{Dec}_\phi = (\mathsf{Share}, \mathcal{F})$ and $\gamma = 1$.

Proof. We prove the theorem above using a sequence of games.

Game 0: The original soundness game. The environment maintains the lists $\mathtt{H}$ and $\mathtt{H}'$, which are initially empty. The ROs are honestly simulated:
$H(\mathsf{a}, x)$: If $\mathtt{H}[(\mathsf{a}, x)] = \perp$, set $\mathtt{H}[(\mathsf{a}, x)] \xleftarrow{R} \{0,1\}^\gamma$. Return $\mathtt{H}[(\mathsf{a}, x)]$.
$H'(||_{j\in[\rho]}(k_j, \mathbf{w}_j))$: If $\mathtt{H}'[||_{j\in[\rho]}(k_j, \mathbf{w}_j)] = \perp$, set $\mathtt{H}'[(k_j, \mathbf{w}_j)_{j\in[\rho]}] \xleftarrow{R} \{0,1\}^\rho$. Return $\mathtt{H}'[||_{j\in[\rho]}(k_j, \mathbf{w}_j)]$.

Game 1: As Game 0, but we abort as soon as there $\exists\ x, y : \mathtt{H}'[x] = \mathtt{H}'[y] \neq \perp\ \wedge\ x \neq y$. We term this abort event E.

Transition Game 0 → Game 1: Game 0 and Game 1 proceed identically, unless event E occurs. Thus, we have that for appropriately chosen ρ both games are statistically close, i.e., $|\Pr[S_0] - \Pr[S_1]| \leq \Pr[E] \leq Q_{H'}^2/2^\rho$

In Game 1, it is impossible to find two different openings for a single RO commitment. Now, we observe that if $x \notin L$ and the adversary does not correctly guess the challenge, at least one challenge yields two inconsistent views. Since the challenge is uniformly random and independent of the adversary's view, this yields $\Pr[S_1] \leq (2/3)^\gamma$ and completes the proof. □

Lemma 7. *Let κ be the security parameter, Q_H be the number of queries to H, Q_S be the overall queries to the simulator, and let the commitments in Scheme 3 be instantiated via a RO $H' : \{0,1\}^\alpha \times \{0,1\}^\beta \times \{0,1\}^\nu \to \{0,1\}^\rho$ so that $\{0,1\}^\nu$ is sampled uniformly at random for every commitment. Then the probability $\epsilon(\kappa)$ for all PPT adversaries $\mathcal{A}$ to break adaptive zero-knowledge of γ parallel executions of Scheme 3 is bounded by $\epsilon(\kappa) \leq \gamma/2^\nu + (Q_S \cdot Q_H)/2^{3\cdot\rho}$.*

The subsequent proof is similar to the general results for Σ-protocols from [FKMV12], yet we have to account for the additional challenge that the simulator only outputs transcripts which are statistically close to original transcripts (which is in contrast to the identically distributed transcripts in [FKMV12]). Furthermore, we also provide concrete bounds.

Proof. We bound the probability of any PPT adversary $\mathcal{A}$ to win the zero-knowledge game by showing that the simulation of the proof oracle is statistically close to the real proof oracle.

Game 0: The zero-knowledge game where the proofs are honestly computed, and the ROs are simulated exactly as in the proof of Lemma 6.

Game 1: As Game 0, but whenever the adversary requests a proof for some tuple (x, w) we choose $\mathsf{e} \xleftarrow{R} \{0, 1, 2\}^\gamma$ before computing a and z. If $\mathtt{H}[(\mathsf{a}, x)] \neq \perp$ we abort. Otherwise, we set $\mathtt{H}[(\mathsf{a}, x)] \leftarrow \mathsf{e}$.

Transition - Game 0 → Game 1: Game 0 and Game 1 proceed identically unless E happens. The message a includes 3 RO commitments with respect to H', i.e., a lower bound for the min-entropy is $3 \cdot \rho$. We have that $|\Pr[S_0] - \Pr[S_1]| \leq {(Q_S \cdot Q_H)}/{2^{3 \cdot \rho}}$.

Game 2: As Game 1, but we compute $((c_{i1}, c_{i2}, c_{i3}))_{i \in [\gamma]}$ in a so that the commitments which will never be opened according to e contain random values.

Transition - Game 1 → Game 2: The statistical difference between Game 1 and Game 2 can be upper bounded by $|\Pr[S_1] - \Pr[S_2]| \leq \gamma \cdot {1}/{2^\nu}$ (for compactness we collapsed the γ game changes into a single game).

Game 3: As Game 2, but we use the HVZK simulator to obtain $(\mathsf{a}, \mathsf{e}, \mathsf{z})$.

Transition - Game 2 → Game 3: This change is conceptual, i.e., $\Pr[S_2] = \Pr[S_3]$.

In Game 0, we sample from the first distribution of the zero-knowledge game, whereas we sample from the second one in Game 3; the distinguishing bounds shown above conclude the proof. □

5 Selecting an Underlying Symmetric Primitive

Our schemes require one or more symmetric primitives suitable to instantiate an OWF and a PRF. In this section we first investigate how choosing a primitive with certain properties impacts an overall instantiation of our schemes. From this, we derive concrete requirements, and, finally, present our chosen primitive.

Signature Size. The size of the signature is mostly influenced by two factors. Firstly by the number γ of parallel repetitions and, secondly, by the size β of the views generated during the ZKBoo protocol. The latter is fully determined by the choice of the OWF f for Fish, and the PRF F for Begol. We denote by a_f and a_F the number of binary multiplication gates contained in the circuit representation of f and F, and by λ_f and λ_F the respective sizes of rings in which the multiplications take place. Furthermore we denote by $(D_f,\, D_F)$ and $(I_f,\, I_F)$ the size of the domain and the images of the functions. More explicitly, the signature sizes are as follows. For instantiations of the Fish scheme we obtain signatures taking up

$$\underbrace{I_f}_{\text{image of } f} + \underbrace{2 \cdot \gamma}_{\text{challenge}} + \gamma \cdot \Big(\underbrace{3 \cdot \rho}_{\text{commitments}} + \underbrace{2 \cdot (\overbrace{D_f + \lambda_f \cdot a_f + I_f}^{\text{views}} + \overbrace{\nu + \alpha}^{\text{randomness}})}_{\text{openings}} \Big)$$

bits of storage. When using Begol, twice as many views and commitments need to be stored as well as another $3 \cdot \gamma$ images. Hence we get signatures of bitlength

$$\underbrace{I_F}_{\text{image of } F} + \underbrace{2 \cdot \gamma}_{\text{challenge}} + \gamma \cdot \Big(\underbrace{6 \cdot \rho}_{\text{commitments}} + \underbrace{4 \cdot (\overbrace{D_F + \lambda_F \cdot a_F + I_F}^{\text{views}} + \overbrace{(\nu + \alpha)}^{\text{randomness}})}_{\text{openings}} \Big).$$

Runtime for Signing and Verification. In the context of multi-party computation (MPC) it is often argued that the runtime is mainly determined by the number of AND gates. Intuitively, the runtime should thus be directly proportional to the signature size. While this is definitely true in an asymptotic sense, the runtime of a concrete instantiation is also influenced by the number of XOR gates (especially for larger numbers of XOR gates). This is also confirmed by our implementation results in Section 6.

5.1 Survey of Suitable Primitives

The size of the signature depends on constants that are close to the security expectation (say, 128 or 256; cf. Section 6 for our concrete choices). The only exceptions are the number of binary multiplication gates, and the size of the rings, which all depend on the choice of the primitive. Hence we survey existing designs that can serve as a OWF or fixed-value-key-binding PRF, respectively. We focus on the following categories.

- Standardized and widely used primitives such as AES, or SHA-256 or SHA-512 or SHA-3.
- Low complexity, “lightweight” ciphers for use in constrained environments.
- Custom block and stream ciphers designed for evaluation by a fully/somewhat homomorphic encryption scheme, a SNARK, or in an MPC protocol.

Standardized General-Purpose Primitives. The smallest known Boolean circuit of AES-128 needs 5440 AND gates, AES-192 needs 6528 AND gates, and AES-256 needs 7616 AND gates [BMP13]. An AES circuit in $\mathbb{F}_{2^4}$ might be more efficient in our setting, as in this case the number of multiplications is lower than 1000 [CGP+12]. This results in an impact on the signature size that is equivalent to 4000 AND gates. Even though collision resistance is often not required, hash functions like SHA-256 are a popular choice for proof-of-concept implementations. The number of AND gates of a single call to the SHA-256 compression function is about 25000 and a single call to the permutation underlying SHA-3 is 38400.

Lightweight Ciphers. Most early designs in this domain focused on small area when implemented as a circuit in hardware where the size of an XOR gate is by a small factor larger than the size of an AND or NAND gate. Notable designs with a low number of AND gates at the 128-bit security level are the block ciphers Noekeon [DPVAR00] (2048) and Fantomas [GLSV14] (2112). Furthermore, one should mention Prince [BCG+12] (1920), or the stream cipher Trivium [DP08] (1536 AND gates to compute 128 output bits) with 80-bit security.

Custom Ciphers with a Low Number of Multiplications. Motivated by various applications in SHE/FHE schemes, MPC protocols and SNARKs, recently a trend to design symmetric encryption primitives with a low number of multiplications or a low multiplicative depth started to evolve. As we shall see, this is a trend we can take advantage of.

We start with the block cipher family called LowMC [ARS+15]. In the most recent version of the proposal [ARS+16], the number of AND gates can be below 500 for 80-bit security, below 800 for 128-bit security, and below 1400 for 256-bit security. The stream cipher Kreyvium [CCF+16] needs similarly to Trivium 1536 AND gates to compute 128 output bits, but offers a higher security level of 128 bit. Even though FLIP [MJSC16] was designed to have especially low depth, it needs hundreds of AND gates per bit and is hence not competitive in our setting.

Last but not least there are the block ciphers and hash functions around MiMC [AGR+16] which need less than $2 \cdot s$ multiplications for s-bit security in a field of size close to 2^s. Note that MiMC is the only design in this category which aims at minimizing multiplications in a field larger than $\mathbb{F}_2$. However, since the size of the signature depends on both the number of multiplications and the size of the field, this leads to a factor $2s^2$ which, for all arguably secure instantiations of MiMC, is already larger than the number of AND gates in the AES circuit.

LowMC has two important advantages over other designs: It has the lowest number of AND gates for every security level: The closest competitor Kreyvium needs about twice as many AND gates and only exists for the 128-bit security level. The fact that it allows for an easy parameterization of the security level is another advantage. We hence use LowMC for our concrete proposal and discuss it in more detail in the following.

5.2 LowMC

LowMC is a flexible block cipher family based on a substitution-permutation network where the block size n, the key size k, the number of 3-bit S-boxes m in the substitution layer and the allowed data complexity d of attacks can independently be chosen. To reduce the multiplicative complexity, the number of S-boxes applied in parallel can be reduced, leaving part of the substitution layer as the identity mapping. The number of rounds r needed to achieve the goals is then determined as a function of all these parameters. For the sake of completeness we include a brief description of LowMC in Appendix D.

To minimize the number of AND gates for a given k and d, we want to minimize $r \cdot m$. A natural strategy would be to set m to 1, and then look for an n that minimizes r. Examples of such an approach are already given in the document describing version 2 of the design [ARS+16]. In our setting, this approach may not lead to the best results in practice, as it ignores the impact of the large amount of XOR operations it requires. What we hence do to find the most suitable parameterization is to explore a larger range of values from m also.

Whenever we want to use LowMC to instantiate one of the required symmetric primitives with s-bit security, we set $k = d = s$. For post-quantum security we double those parameters to take generic quantum-speedups into account. This takes into account current knowledge of quantum-cryptanalysis for models that are very generous to the attacker [KLLN16, KLLN15].

An exception to this conservative choice can be found in our optimization discussion in Section 6.5 where we describe why a less conservative choice for the parameter d is possible.

6 Implementation and Design Space Exploration

We provide a library implementing both the Fish and the Begol signature scheme. The library provides an API exposing an interface to generate LowMC instances for a given parameter set, as well as an easy to use interface for key generation, signature generation/verification in both schemes. To be able to explore various directions in the LowMC parameter space, we also implemented a benchmarking framework. The library is implemented in C using the OpenSSL[13] and m4ri[14] libraries and is available online at https://github.com/IAIK/fish-begol.

6.1 Implementation of Building Blocks

The building blocks in the protocol are instantiated similar as in the implementation of ZKBoo [GMO16]:

PRNG. Random tapes are generated pseudorandomly using AES in counter mode, where the AES keys are generated using OpenSSL's secure random number generator. In the linear decomposition of the AND gates we use a random function that picks the bits from the bit stream generated using AES. Since the number of AND gates is known a-priori, we can pre-compute all random bits at the beginning of the protocol.

Commitments. The RO H' used to commit to the views is implemented using SHA-256, i.e. $\mathsf{Com}(k, w) \coloneqq \text{SHA-256}(k, w, r)$ where $r \xleftarrow{R} \{0,1\}^{\nu}$.

Challenge Generation. For both approaches we instantiate RO $H : \{0,1\}^* \rightarrow \{0,1,2\}^{\gamma}$ using SHA-256 and rejection sampling: we split the output bits of SHA-256 in pairs of two bits and reject all pairs with both bits set.

LowMC-Based Primitives. It is straight forward to use LowMC as PRF by setting $f_{\mathsf{s}}(x) = \mathsf{Enc}_{\mathsf{s}}(x) \oplus x$. This PRF based on LowMC can also be considered to yield a computational fixed-value-key-binding PRF: It is reasonable to assume that finding a new key which maps one particular input to one particular output is no easier than generic key search. The OWF is instantiated by fixing a plaintext and using the input to the OWF as key for LowMC. One-wayness is thereby mapped to the difficulty of finding a secret block-cipher key with a known plaintext. This is a standard requirement any modern block cipher is expected to meet.

[13] https://openssl.org

[14] https://bitbucket.org/malb/m4ri

6.2 Circuit for LowMC

For the linear (2,3)-decomposition we view LowMC as circuit over $\mathbb{F}_2$. From the description of LowMC it is clear, that the circuit consists only of AND and XOR gates. Hence we have $\lambda_f = \lambda_F = 1$. When optimizing the circuit so that modified AND gates as described in [GMO16] are used, the number of bits we have to store for each view is further reduced to $a_f = a_F = 3 \cdot r \cdot m$, where r is the number of rounds and m is the number of S-boxes in LowMC.

Our implementation builds upon the m4ri library [ABH10] and to better fit the memory layout used by m4ri, all operations are performed using transposed matrices and column vectors. For improved performance of the S-box layer, we use a bit-sliced version of the S-box, i.e., instead of looping over m bit triples and applying the S-box separately, we evaluate the full S-box layer using a sequence of AND, XOR and shift instructions independent of m.

Since the affine layer of LowMC only consists of AND and XOR operations, it benefits from using block sizes such that all computations of this layer can be performed using SIMD instruction sets that provide bitwise AND and XOR instruction for 128 bit respectively 256 bit operands like AVX2 and SSE2 or NEON. Similarly the bit-sliced version of the S-box layer can be improved by using SIMD AND and XOR instructions. Only inter-lane bit-shifts, which are necessary for the bit-sliced S-box, do not fit the model of SIMD instruction sets. Thus shifts require up to five instructions and only benefit slightly. Since our implementation uses (arrays of) native words to store the bit vectors, the implementation benefits from a choice of parameters such that $3 \cdot m$ is close to the word size. This choice allows us to maximize the number of parallel S-box evaluations in the bitsliced implementation. Note that also the storage overhead of individual views benefits from $3 \cdot m$ close to byte boundaries as we always use bytes instead of individual bits.

While we optimized the AND and XOR gates in the LowMC circuit to use SIMD instructions where possible, we have kept the implementation very generic for arbitrary choices of the LowMC parameters. Hence we are able to explore the parameter space in multiple directions while still profiting from the speed up of vectorized operations.

6.3 Experimental Setup and Results

Our experiments were performed on an Intel Core i7-4790 CPU clocked at 3.60 GHz with 4 cores[15] and 16 GB RAM running Ubuntu 16.04. We omit any benchmarks of Gen, since the key generation only needs to request k respectively $2 \cdot k$ bits from the random number generator and to compute one LowMC encryption and has a runtime of some microseconds. Henceforth, we target the 128 bit pre- as well as post-quantum setting and we globally set our repetition count to $\gamma := 219$ in the pre-quantum setting and to $\gamma := 438$ in the post-quantum setting. Based on this, we globally fix the output size of the RO to $\rho := 256$, and

[15] HyperThreading was disabled for the experiments to reduce noise in the benchmarks.

the size of the randomness in the RO commitment to $\nu := 136$ (then all terms in the bound involving these factors are negligible in the security parameter).

Similar to the observations in [HRSS16], we need to double the repetition count in the post-quantum setting due to Grover's algorithm [Gro96]. To see the effects of the search algorithm, an adversary at first computes γ views such that it can answer two of the three possible challenges honestly for each view. Considering the possible permutations of the individual views, the adversary is thus able to answer 2^γ out of the 3^γ challenges. Grover's algorithm is then tasked to find a permutation of the views such that they correspond to one of the 2^γ challenges. Out of the 2^γ permutations, the expected number of solutions is $(\frac{4}{3})^\gamma$, hence Grover's algorithm reduces the time to find a solution to $(\frac{3}{2})^{\frac{\gamma}{2}}$. So for the 128 bit security level, we require γ large enough to satisfy $(\frac{3}{2})^{\frac{\gamma}{2}} \geq 2^{128}$, where $\gamma = 438$ is the smallest possible repetition count.

Selection of the Most Suitable LowMC Instances. We now explore the design space of LowMC to select the most suitable instances among the 128 bit pre- as well as post-quantum instances. Our results are presented in Figure 2 (note that the plot axes are in logarithmic scale). Choosing a concrete LowMC instance results in a trade-off between computational efficiency and signature size, parametrized by the number of rounds and by the number of S-boxes.

We select the instances yielding the best balance between signature size and computational efficiency. In particular, using the notation [blocksize]-[keysize]-[#sboxes]-[#rounds], we choose the 128-128-12-26 instance from the 128 bit pre-quantum instances and the 256-256-20-31 instance from the 128 post-quantum instances. To underpin our argumentation regarding the choice of LowMC, we also include the runtime and proof size of the SHA-256 implementation (with one call to the compression function) from the authors of ZKBoo in [GMO16]. Informally speaking, this can be seen as a rough estimation of a lower bound for an instantiation of our scheme Fish with SHA-256 instead of LowMC.

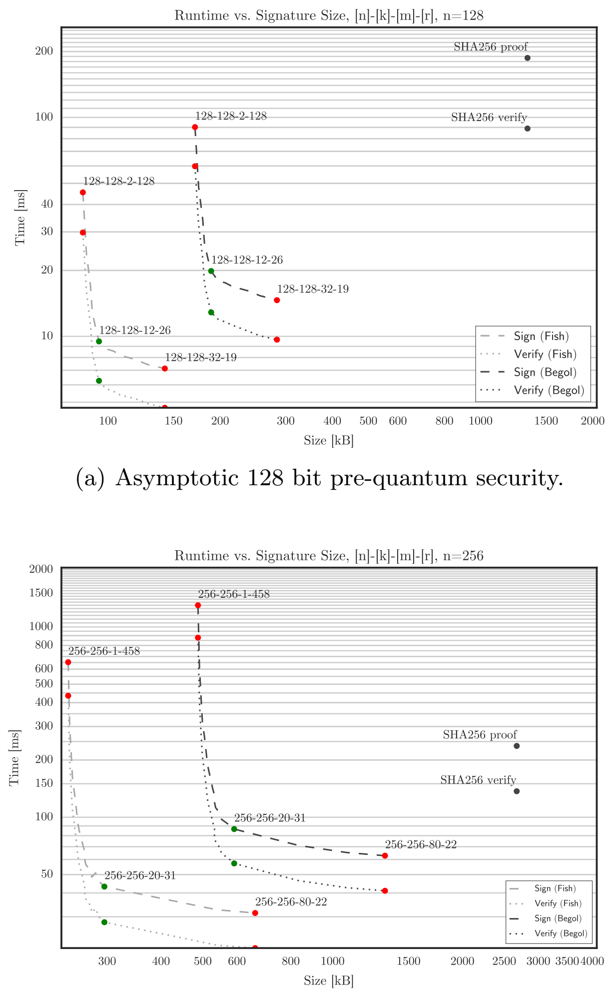
An immediate conclusion we can draw from these results is that both signature schemes can be instantiated such that they are entirely practical.

Parallelization. One positive aspect regarding the γ parallel repetitions is that they are independent of each other.[16] In particular, this holds for all steps in the signing and verification algorithm up to the initial requests to OpenSSL's random number generator and the computation of the challenge. This allows us to take advantage of the multi-core architecture of modern processors using OpenMP.[17] As exemplified for Begol in Figure 3, we can observe a significant performance increase until the number of threads matches the actual number of CPU cores. We note that exactly the same effects also occur for instantiations of Fish. Furthermore, they also occur regardless of the LowMC parameters.

Signing Arbitrary Length Messages. We have not discussed signing of messages of arbitrary length with our approaches so far. For Fish, this is already covered by the FS_Σ transform as we are not limited in the message length in

[16] This observation was also made for ZKBoo in [GMO16].

[17] http://openmp.org



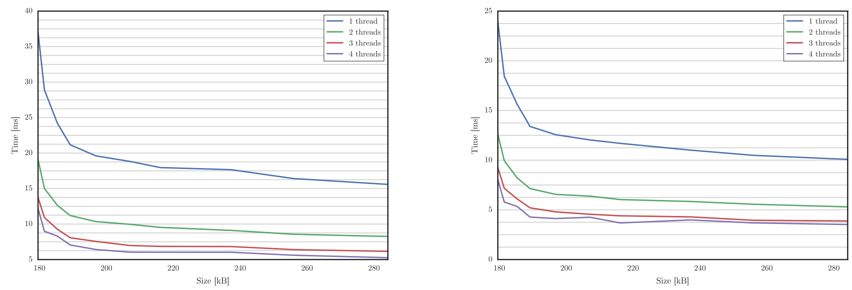
(a) Asymptotic 128 bit pre-quantum security.

(b) Asymptotic 128 bit post-quantum security.

Fig. 2. Measurements for instance selection (average over 100 runs).

the challenge computation. For Begol, we observe that it needs a bit more care as we need use a collision-resistant (CR) hash function prior to computing the signature and thus have to adjust the output size of the CR hash with the input size of the PRF. If we follow the argumentation of Bernstein [Ber09], then like with Fish our instantiations in the benchmarks already support arbitrary length messages when using SHA-256 as a CR hash. If we follow the more conservative approach of assuming cost $2^{b/3}$ for b bit collisions, then for the 128-bit post-quantum setting we would have to slightly increase the input size of the

PRF (i.e., change the blocksize of the second LowMC instance to 384 bit). This would slightly increase computation times as well as signature sizes.



(a) Sign (b) Verify

Fig. 3. Runtime of the parallelized version of Sign and Verify of Begol at 128-bit pre-quantum security level using an increasing number of threads.

6.4 Comparison with Related Work

To compare our signature scheme to other recent proposals, we selected schemes which have reference implementations freely available. We benchmarked these implementations on the machine we used to perform our benchmarks. Table 1 gives an overview of the results, including MQDSS [HRSS16], TESLA [ABB+15], ring-TESLA [ABB+16], BLISS [DDLL13], and SPHINCS-256 [BHH+15].[18]

Our implementation can be considered as a general-purpose implementation, which is flexible enough to cover the entire design spectrum of both our approaches. In contrast, the implementations of other candidate schemes used for comparison come with a highly optimized implementation targeting a specific security level (and often also specific instances). Thus, our timings can be considered to be more conservative than the ones of the other schemes used for comparison. Yet, both signing and verification times are comparable to the ones of the MQ 5pass scheme, and the signing times are comparable to SPHINCS-256.

It seems that avoiding additional structured hardness assumptions comes at the cost of larger signatures. An interesting observation when compared to SPHINCS-256 (which is the only other scheme without additional structured hardness assumptions) is that our approach can be interpreted as using the RO heuristic to trade smaller keys for larger signatures. Furthermore, it should also be noted that our work establishes a new direction to design signature schemes and we believe that there is much room for improvement, e.g., by the design of

[18] Key sizes and signature sizes from BLISS were taken from [DDLL13], as they were not readily available in the implementation.

Scheme	Gen [ms]	Sign [ms]	Verify [ms]	$\lvert\mathsf{sk}\rvert$ [bytes]	$\lvert\mathsf{pk}\rvert$ [bytes]	$\lvert\sigma\rvert$ [bytes]	Tight	M	PQ
Fish-128-128-12-26	0.01	9.46	6.25	16	16	86747	$\times$	ROM	$\times$
Fish-256-256-20-31	0.01	43.19	28.07	32	32	302704	$\times$	ROM	✓
Begol-256-256-20-31	0.01	86.66	57.18	32	32	605362	?§	ROM	✓
MQ 5pass	0.96	7.21	5.17	32	74	40952	$\times$	ROM	✓
SPHINCS-256	0.82	13.44	0.58	1088	1056	41000	✓	SM	✓
BLISS-I	44.16	0.12	0.02	2048	7168	5732	✓	ROM	✓
Ring-TESLA	16528.50	0.06	0.03	12288	8192	1568	$\times$‡,‖	ROM	✓
TESLA-768	48570.01	0.65	0.36	3293216	4227072	2336	$\times$‡	(Q)ROM	✓

Table 1. Timings and sizes of private keys (sk), public keys (pk) and signatures (σ). §... Our tight security reduction does not apply to our current instantiation. It remains a major open question to find an instantiation where the tight reduction applies. ‡... There seems to be a flaw in the tight reduction, but there still exists a valid non-tight reduction (cf. https://eprint.iacr.org/2016/030).‖... A recent work [Cho16] indicates that the choice of the parameters for Ring-TESLA yields correctness issues and does not provide the claimed security levels. [Cho16] proposes a new set of parameters which negatively impact the efficiency.

new symmetric primitives especially focusing on optimizing the metrics required by our approach.

6.5 Optimizing Signature Sizes

It is a natural question to ask whether it is possible to further decrease the signatures sizes. We observe that for our implementation of Fish there seems to be some room for a more aggressive choice of parameters: the way we use the OWF in Fish would allow to use less conservative values for the data complexity. In particular, the most progressive choice would be to use a data complexity $d = 1$ (where the adversary gets to see at most 2 plaintext-ciphertext pairs under the target key).

To give a lower bound on the signature sizes obtained via more progressive parameters, we use instances with a minimal number of AND gates. That is, we use instances with only one S-box ($m = 1$) as our results above show that these instances yield the smallest possible signature sizes. For $m = 1$ and $d = 1$ we derive a suitable number of rounds r. This yields the instance 128-128-1-156 for the 128-bit pre-quantum setting and the instance 256-256-1-243 for the 128-bit post-quantum setting. The resulting signature sizes are 61124 bytes for Fish in the 128-bit pre-quantum setting, and 178860 bytes in the 128-bit post-quantum setting.

We note here however that currently our main proposal is with the more conservative $d = n$ rather than $d = 1$ possibility discussed above. This may be seen as a way to introduce a security margin, taking into account the fact that LowMC is a rather novel design and progress in cryptanalysis is ongoing [DEM15, DLMW15].

7 Conclusion and Open Questions

In this paper we contribute two new practically efficient post-quantum signature candidates called Fish and Begol, which open up a new direction for the design of EUF-CMA secure signature schemes. Likewise to all other practical post-quantum signature candidates besides hash-based signatures, our scheme comes with a security proof in the ROM.

A major open question in all those works is whether it is possible to lift the corresponding security proofs to the QROM while preserving practicality. One major obstacle in the QROM is to handle the rewinding of adversaries within security reductions [DFG13]. Possibilities to circumvent this issue are via history-free reductions [BDF+11] or the use of oblivious commitments within FS_{Σ} (as, e.g., used by the TESLA authors). Unfortunately, these directions do not apply to both our approaches. Nevertheless, a very interesting and indeed promising approach is the adoption of straight-line extractable NIZK secure in the QROM as proposed by Unruh in [Unr15]. Unfortunately, his results cannot be directly applied, among others, due to the 3-special soundness of ZKBoo. Another important observation is that the security proof for Begol does not require to rewind the adversary which yields a viable direction for future work. In particular, such an approach might allow to bypass the overhead imposed by the requirement for straight-line extractability. Yet, we have to note that an efficient instantiation in a setting without structured hardness assumptions would also require to come up with an efficient fixed-value-key-binding PRF in this setting.

Our work in this paper also informs future developments in the design of symmetric primitives with few multiplications. In order to be competitive with other post-quantum signature candidates we could not rely on already standardized approaches like AES or SHA-3, but need to use new ones. Even though LowMC turned out to be the most suitable option, none of the available designs directly considers the metric we would be most interested in for minimizing our signature size: the product of the number of binary multiplication gates contained in the circuit representation and the ring sizes in which the multiplications take place.

Our schemes have the feature of very short private and public keys but comparatively large signatures. A potential use-case to be explored further are devices built in small technology nodes (e.g., 14nm) where non-volatile storage is scarce. Creation or verification and transfer of signatures may well be doable on such devices especially when only occasionally needed and the device is connected to a suitable interface. The cost-driver for such devices could end up being the device-dependent private keys, despite the relatively large but device-independent circuit. Having only 100s of key-bits, e.g., encoded with fuses or stored in encrypted external flash memory could be much preferred over the orders of magnitude more key-bits that would be needed with some other signatures schemes.

Relatedly, another interesting open issue is a rigorous analysis of resistance of our approaches to power- or timing-based side-channel attacks or fault attacks. A first observation in this direction is that the structure of ZKBoo seems to inherently bring some resistance against side-channel attacks due to the use of secret sharing techniques.

Acknowledgements. We are grateful to Mihir Bellare for pointing us to a problem with the initial version of Begol, valuable suggestions how to fix this issue, and additional comments. We also want to thank Eike Kiltz for making us aware of a previous practical use of the Bellare-Goldwasser paradigm in [BKP14] and for independently pointing us to the aforementioned problem with Begol.

References

[AABN02] M. Abdalla, J. H. An, M. Bellare, and C. Namprempre. From identification to signatures via the fiat-shamir transform: Minimizing assumptions for security and forward-security. In *EUROCRYPT*, 2002.

[ABB+15] E. Alkim, N. Bindel, J. Buchmann, Ö. Dagdelen, and P. Schwabe. Tesla: Tightly-secure efficient signatures from standard lattices. Cryptology ePrint Archive, Report 2015/755, 2015.

[ABB+16] S. Akleylek, N. Bindel, J. A. Buchmann, J. Krämer, and G. A. Marson. An efficient lattice-based signature scheme with provably secure instantiation. In *AFRICACRYPT*, 2016.

[ABH10] M. R. Albrecht, G. V. Bard, and W. Hart. Algorithm 898: Efficient multiplication of dense matrices over GF(2). *ACM Transactions on Mathematical Software*, 37(1), 2010.

[ABP13] M. Abdalla, F. Ben Hamouda, and D. Pointcheval. Tighter reductions for forward-secure signature schemes. In *PKC*, 2013.

[ADV+12] S. M. E. Y. Alaoui, Ö. Dagdelen, P. Véron, D. Galindo, and P. Cayrel. Extended security arguments for signature schemes. In *AFRICACRYPT*, 2012.

[AFLT12] M. Abdalla, P. Fouque, V. Lyubashevsky, and M. Tibouchi. Tightly-secure signatures from lossy identification schemes. In *EUROCRYPT*, 2012.

[AGR+16] M. R. Albrecht, L. Grassi, C. Rechberger, A. Roy, and T. Tiessen. MiMC: Efficient encryption and cryptographic hashing with minimal multiplicative complexity. In *ASIACRYPT*, pages 191–219, 2016.

[ARS+15] M. R. Albrecht, C. Rechberger, T. Schneider, T. Tiessen, and M. Zohner. Ciphers for MPC and FHE. In *EUROCRYPT*, 2015.

[ARS+16] M. Albrecht, C. Rechberger, T. Schneider, T. Tiessen, and M. Zohner. Ciphers for MPC and FHE. Cryptology ePrint Archive, Report 2016/687, 2016.

[BB13] R. E. Bansarkhani and J. A. Buchmann. Improvement and efficient implementation of a lattice-based signature scheme. In *SAC*, 2013.

[BCG+12] J. Borghoff, A. Canteaut, T. Güneysu, E. B. Kavun, M. Knezevic, L. R. Knudsen, G. Leander, V. Nikov, C. Paar, C. Rechberger, P. Rombouts, S. S. Thomsen, and T. Yalçin. PRINCE - a low-latency block cipher for pervasive computing applications - extended abstract. In *ASIACRYPT*, 2012.

[BCG+13] E. Ben-Sasson, A. Chiesa, D. Genkin, E. Tromer, and M. Virza. Snarks for C: verifying program executions succinctly and in zero knowledge. In *CRYPTO*, 2013.

[BCG$^+$14] E. Ben-Sasson, A. Chiesa, C. Garman, M. Green, I. Miers, E. Tromer, and M. Virza. Zerocash: Decentralized anonymous payments from bitcoin. In *IEEE SP*, 2014.

[BDF$^+$11] D. Boneh, Ö. Dagdelen, M. Fischlin, A. Lehmann, C. Schaffner, and M. Zhandry. Random oracles in a quantum world. In *ASIACRYPT*, 2011.

[BDH11] J. A. Buchmann, E. Dahmen, and A. Hülsing. XMSS - A practical forward secure signature scheme based on minimal security assumptions. In *PQCrypto*, 2011.

[Ber09] D. J. Bernstein. Cost analysis of hash collisions: Will quantum computers make sharcs obsolete? In *SHARCS*, 2009.

[BG89] M. Bellare and S. Goldwasser. New paradigms for digital signatures and message authentication based on non-interative zero knowledge proofs. In *CRYPTO*, 1989.

[BG14] S. Bai and S. D. Galbraith. An improved compression technique for signatures based on learning with errors. In *CT-RSA*, 2014.

[BHH$^+$15] D. J. Bernstein, D. Hopwood, A. Hülsing, T. Lange, R. Niederhagen, L. Papachristodoulou, M. Schneider, P. Schwabe, and Z. Wilcox-O'Hearn. SPHINCS: practical stateless hash-based signatures. In *EUROCRYPT*, 2015.

[BKKP15] O. Blazy, S. A. Kakvi, E. Kiltz, and J. Pan. Tightly-secure signatures from chameleon hash functions. In *PKC*, 2015.

[BKP14] O. Blazy, E. Kiltz, and J. Pan. (hierarchical) identity-based encryption from affine message authentication. In *Advances in Cryptology - CRYPTO*, pages 408–425, 2014.

[BL16] X. Boyen and Q. Li. Towards tightly secure short signature and ibe. Cryptology ePrint Archive, Report 2016/498, 2016.

[BLN$^+$16] P. S. L. M. Barreto, P. Longa, M. Naehrig, J. E. Ricardini, and G. Zanon. Sharper ring-lwe signatures. *IACR Cryptology ePrint Archive*, 2016:1026, 2016.

[BMP13] J. Boyar, P. Matthews, and R. Peralta. Logic minimization techniques with applications to cryptology. *Journal of Cryptology*, 26(2):280–312, 2013.

[BMT14] M. Bellare, S. Meiklejohn, and S. Thomson. Key-versatile signatures and applications: Rka, KDM and joint enc/sig. In *Advances in Cryptology - EUROCRYPT 2014*, pages 496–513, 2014.

[BPS16] M. Bellare, B. Poettering, and D. Stebila. From identification to signatures, tightly: A framework and generic transforms. In *Cryptology ePrint Archive, Report 2015/1157, to appear in Asiacrypt 2016*, 2016.

[BPW12] D. Bernhard, O. Pereira, and B. Warinschi. How not to prove yourself: Pitfalls of the fiat-shamir heuristic and applications to helios. In *ASIACRYPT*, 2012.

[BR93] M. Bellare and P. Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In *ACM CCS*, 1993.

[BR96] M. Bellare and P. Rogaway. The exact security of digital signatures - how to sign with RSA and rabin. In *EUROCRYPT*, 1996.

[CCF$^+$16] A. Canteaut, S. Carpov, C. Fontaine, T. Lepoint, M. Naya-Plasencia, P. Paillier, and R. Sirdey. Stream ciphers: A practical solution for efficient homomorphic-ciphertext compression. In *FSE*, 2016.

[CDS94] R. Cramer, I. Damgård, and B. Schoenmakers. Proofs of partial knowledge and simplified design of witness hiding protocols. In *CRYPTO*, 1994.

[CFH+15] C. Costello, C. Fournet, J. Howell, M. Kohlweiss, B. Kreuter, M. Naehrig, B. Parno, and S. Zahur. Geppetto: Versatile verifiable computation. In *IEEE SP*, 2015.

[CFS01] N. Courtois, M. Finiasz, and N. Sendrier. How to achieve a mceliece-based digital signature scheme. In *ASIACRYPT*, 2001.

[CGP+12] C. Carlet, L. Goubin, E. Prouff, M. Quisquater, and M. Rivain. Higher-order masking schemes for s-boxes. In *FSE*, 2012.

[Cho16] A. Chopra. Improved parameters for the ring-tesla digital signature scheme. Cryptology ePrint Archive, Report 2016/1099, 2016. http://eprint.iacr.org/2016/1099.

[CK16] S. Chatterjee and C. Kamath. A closer look at multiple forking: Leveraging (in)dependence for a tighter bound. *Algorithmica*, 74(4), 2016.

[CL06] M. Chase and A. Lysyanskaya. On signatures of knowledge. In *CRYPTO*, 2006.

[CMR98] R. Canetti, D. Micciancio, and O. Reingold. Perfectly one-way probabilistic hash functions (preliminary version). In *Proceedings of the Thirtieth Annual ACM Symposium on the Theory of Computing, Dallas, Texas, USA, May 23-26, 1998*, pages 131–140, 1998.

[DBG+14] Ö. Dagdelen, R. E. Bansarkhani, F. Göpfert, T. Güneysu, T. Oder, T. Pöppelmann, A. H. Sánchez, and P. Schwabe. High-speed signatures from standard lattices. In *LATINCRYPT*, 2014.

[DDLL13] L. Ducas, A. Durmus, T. Lepoint, and V. Lyubashevsky. Lattice signatures and bimodal gaussians. In *CRYPTO*, 2013.

[DEM15] C. Dobraunig, M. Eichlseder, and F. Mendel. Higher-order cryptanalysis of lowmc. In *Information Security and Cryptology - ICISC*, pages 87–101, 2015.

[DFG13] Ö. Dagdelen, M. Fischlin, and T. Gagliardoni. The fiat-shamir transformation in a quantum world. In *ASIACRYPT*, 2013.

[DGV+16] Ö. Dagdelen, D. Galindo, P. Véron, S. M. E. Y. Alaoui, and P. Cayrel. Extended security arguments for signature schemes. *Des. Codes Cryptography*, 78(2):441–461, 2016.

[DLMW15] I. Dinur, Y. Liu, W. Meier, and Q. Wang. Optimized interpolation attacks on lowmc. In *Advances in Cryptology - ASIACRYPT*, pages 535–560, 2015.

[DP08] C. De Cannière and B. Preneel. Trivium. In *New Stream Cipher Designs - The eSTREAM Finalists*. 2008.

[DPVAR00] J. Daemen, M. Peeters, G. Van Assche, and V. Rijmen. Nessie proposal: Noekeon. In *First Open NESSIE Workshop*, 2000.

[Duc14] L. Ducas. Accelerating bliss: the geometry of ternary polynomials. *IACR Cryptology ePrint Archive*, 2014, 2014.

[ELL+15] M. F. Ezerman, H. T. Lee, S. Ling, K. Nguyen, and H. Wang. A provably secure group signature scheme from code-based assumptions. In *Advances in Cryptology - ASIACRYPT*, pages 260–285, 2015.

[FGO+13] J. Faugère, V. Gauthier-Umaña, A. Otmani, L. Perret, and J. Tillich. A distinguisher for high-rate mceliece cryptosystems. *IEEE Trans. Information Theory*, 59(10):6830–6844, 2013.

[Fis99] M. Fischlin. Pseudorandom function tribe ensembles based on one-way permutations: Improvements and applications. In *Advances in Cryptology - EUROCRYPT '99, International Conference on the Theory and Application of Cryptographic Techniques, Prague, Czech Republic, May 2-6, 1999, Proceeding*, pages 432–445, 1999.

[FKMV12] S. Faust, M. Kohlweiss, G. A. Marson, and D. Venturi. On the non-malleability of the fiat-shamir transform. In *INDOCRYPT*, 2012.

[FS86] A. Fiat and A. Shamir. How to prove yourself: Practical solutions to identification and signature problems. In *CRYPTO '86*, 1986.

[GGH97] O. Goldreich, S. Goldwasser, and S. Halevi. Public-key cryptosystems from lattice reduction problems. In *CRYPTO*, 1997.

[GGPR13] R. Gennaro, C. Gentry, B. Parno, and M. Raykova. Quadratic span programs and succinct nizks without pcps. In *EUROCRYPT*, 2013.

[GJ03] E. Goh and S. Jarecki. A signature scheme as secure as the diffie-hellman problem. In *EUROCRYPT*, 2003.

[GLP12] T. Güneysu, V. Lyubashevsky, and T. Pöppelmann. Practical lattice-based cryptography: A signature scheme for embedded systems. In *CHES*, 2012.

[GLSV14] V. Grosso, G. Leurent, F. Standaert, and K. Varici. Ls-designs: Bitslice encryption for efficient masked software implementations. In *FSE*, 2014.

[GMO16] I. Giacomelli, J. Madsen, and C. Orlandi. Zkboo: Faster zero-knowledge for boolean circuits. In *USENIX Security*, 2016.

[GMR85] S. Goldwasser, S. Micali, and C. Rackoff. The knowledge complexity of interactive proof-systems (extended abstract). In *STOC*, 1985.

[GMW86] O. Goldreich, S. Micali, and A. Wigderson. How to prove all np-statements in zero-knowledge, and a methodology of cryptographic protocol design. In *CRYPTO*, 1986.

[Gol86] O. Goldreich. Two remarks concerning the goldwasser-micali-rivest signature scheme. In *CRYPTO*, 1986.

[GPV08] C. Gentry, C. Peikert, and V. Vaikuntanathan. Trapdoors for hard lattices and new cryptographic constructions. In *STOC*, 2008.

[Gro96] L. K. Grover. A fast quantum mechanical algorithm for database search. In *STOC*, 1996.

[GS08] J. Groth and A. Sahai. Efficient Non-interactive Proof Systems for Bilinear Groups. In *EUROCRYPT*, 2008.

[HJ12] D. Hofheinz and T. Jager. Tightly secure signatures and public-key encryption. In *CRYPTO*, 2012.

[HMR15] Z. Hu, P. Mohassel, and M. Rosulek. Efficient zero-knowledge proofs of non-algebraic statements with sublinear amortized cost. In *CRYPTO*, 2015.

[HRSS16] A. Hülsing, J. Rijneveld, S. Samardjiska, and P. Schwabe. From 5-pass mq-based identification to mq-based signatures. In *Cryptology ePrint Archive, Report 2016/708, to appear in Asiacrypt 2016*, 2016.

[IKOS07] Y. Ishai, E. Kushilevitz, R. Ostrovsky, and A. Sahai. Zero-knowledge from secure multiparty computation. In *Proceedings of the 39th Annual ACM Symposium on Theory of Computing, San Diego, California, USA, June 11-13, 2007*, pages 21–30, 2007.

[JKO13] M. Jawurek, F. Kerschbaum, and C. Orlandi. Zero-knowledge using garbled circuits: how to prove non-algebraic statements efficiently. In *ACM CCS*, 2013.

[Kat10] J. Katz. *Digital Signatures*. Springer, 2010.

[KLLN15] M. Kaplan, G. Leurent, A. Leverrier, and M. Naya-Plasencia. Quantum Differential and Linear Cryptanalysis. *ArXiv e-prints*, October 2015.

[KLLN16] M. Kaplan, G. Leurent, A. Leverrier, and M. Naya-Plasencia. Breaking symmetric cryptosystems using quantum period finding. In *CRYPTO*, 2016.

[KMP16] E. Kiltz, D. Masny, and J. Pan. Optimal security proofs for signatures from identification schemes. In *CRYPTO*, 2016.

[KW03] J. Katz and N. Wang. Efficiency improvements for signature schemes with tight security reductions. In *ACM CCS*, 2003.

[Lam79] L. Lamport. Constructing digital signatures from one-way functions. Technical Report SRI-CSL-98, SRI Intl. Computer Science Laboratory, 1979.

[LS12] G. Landais and N. Sendrier. Cfs software implementation. Cryptology ePrint Archive, Report 2012/132, 2012.

[Lyu09] V. Lyubashevsky. Fiat-shamir with aborts: Applications to lattice and factoring-based signatures. In *ASIACRYPT*, 2009.

[Lyu12] V. Lyubashevsky. Lattice signatures without trapdoors. In *EUROCRYPT*, 2012.

[McE78] R. J. McEliece. A public-key cryptosystem based on algebraic coding theory. Technical Report DSN PR 42-44, 1978.

[Mer89] R. C. Merkle. A certified digital signature. In *CRYPTO*, 1989.

[MGS11] C. A. Melchor, P. Gaborit, and J. Schrek. A new zero-knowledge code based identification scheme with reduced communication. In *ITW*, 2011.

[MJSC16] P. Méaux, A. Journault, F. Standaert, and C. Carlet. Towards stream ciphers for efficient FHE with low-noise ciphertexts. In *EUROCRYPT*, 2016.

[MKF$^+$16] D. McGrew, P. Kampanakis, S. Fluhrer, S.-L. Gazdag, D. Butin, and J. Buchmann. State management for hash based signatures. Cryptology ePrint Archive, Report 2016/357, 2016.

[Nie86] H. Niederreiter. Knapsack-type cryptosystems and algebraic coding theory. *Problems of Control and Information Theory*, 1986.

[OO98] K. Ohta and T. Okamoto. On concrete security treatment of signatures derived from identification. In *CRYPTO*, 1998.

[PCG01] J. Patarin, N. Courtois, and L. Goubin. Quartz, 128-bit long digital signatures. In *CT-RSA*, 2001.

[PCY$^+$15] A. Petzoldt, M. Chen, B. Yang, C. Tao, and J. Ding. Design principles for hfev- based multivariate signature schemes. In *ASIACRYPT*, 2015.

[Pei16] C. Peikert. A decade of lattice cryptography. *Foundations and Trends in Theoretical Computer Science*, 10(4), 2016.

[PS96] D. Pointcheval and J. Stern. Security proofs for signature schemes. In *EUROCRYPT*, 1996.

[Sch91] C. Schnorr. Efficient signature generation by smart cards. *J. Cryptology*, 4(3), 1991.

[Sho94] P. W. Shor. Polynominal time algorithms for discrete logarithms and factoring on a quantum computer. In *ANTS-I*, 1994.

[SSH11] K. Sakumoto, T. Shirai, and H. Hiwatari. Public-key identification schemes based on multivariate quadratic polynomials. In *CRYPTO*, 2011.

[Ste93] J. Stern. A new identification scheme based on syndrome decoding. In *CRYPTO*, 1993.

[Unr15] D. Unruh. Non-interactive zero-knowledge proofs in the quantum random oracle model. In *EUROCRYPT*, 2015.

[Vér96] P. Véron. Improved identification schemes based on error-correcting codes. *Appl. Algebra Eng. Commun. Comput.*, 8(1), 1996.

A Required Primitives

One-Way Functions. Below, we recall the notion of one-way functions.

Definition 3. *A function $f : D \to \mathsf{R}$ is called a one-way function, if (1) there exists a PPT algorithm $\mathcal{A}_1$ so that $\forall\, x \in D : \mathcal{A}_1(x) = f(x)$, and if (2) for every PPT algorithm $\mathcal{A}_2$ there is a negligible function $\varepsilon(\cdot)$ such that it holds that*

$$\Pr\left[x \xleftarrow{R} D,\ x^\star \leftarrow \mathcal{A}_2(1^\kappa, f(x)) \ :\ f(x) = f(x^\star)\right] \leq \varepsilon(\kappa).$$

Unless stated otherwise, we assume D to be efficiently sampleable.

Pseudorandom Functions. Let $F : \mathcal{S} \times D \to \mathsf{R}$ be a family of functions and let Γ be the set of all functions $D \to \mathsf{R}$. F is a pseudorandom function (family) if it is efficiently computable and for all PPT distinguishers $\mathcal{D}$ there is a negligible function $\varepsilon(\cdot)$ such that

$$\left|\Pr[\mathsf{s} \xleftarrow{R} \mathcal{S}, \mathcal{D}^{F_\mathsf{s}(\cdot)}(1^\kappa)] - \Pr[f \xleftarrow{R} \Gamma, \mathcal{D}^{f(\cdot)}(1^\kappa)]\right| \leq \varepsilon(\kappa)$$

Below, we provide an explicit definition of a notion introduced in [CMR98, Fis99].

Definition 4 (Fixed-Value-Key-Binding PRF). *A PRF family $F : \mathcal{S} \times D \to \mathsf{R}$ is key-binding if there exists a special value $\beta \in D$ so that it holds for all adversaries $\mathcal{A}$ that:*

$$\Pr[\mathsf{s} \xleftarrow{R} \mathcal{S},\ \mathsf{s}' \leftarrow \mathcal{A}^{F_\mathsf{s}(\cdot)}(F_\mathsf{s}(\beta), \beta) \ :\ F_{\mathsf{s}'}(\beta) = F_\mathsf{s}(\beta)] = 0.$$

Moreover, we introduce a relaxed (computational) version of the above definition.

Definition 5 (Computational Fixed-Value-Key-Binding PRF). *A PRF family $F : \mathcal{S} \times D \to \mathsf{R}$ is key-binding if there exists a special value $\beta \in D$ so that it holds for all PPT adversaries $\mathcal{A}$ that:*

$$\Pr[\mathsf{s} \xleftarrow{R} \mathcal{S},\ \mathsf{s}' \leftarrow \mathcal{A}^{F_\mathsf{s}(\cdot)}(1^\kappa, F_\mathsf{s}(\beta), \beta) \ :\ F_{\mathsf{s}'}(\beta) = F_\mathsf{s}(\beta)] \leq \varepsilon(\kappa).$$

Signature Schemes. Below we recall a standard definition of signature schemes.

Definition 6. *A signature scheme Σ is a triple* $(\mathsf{Gen}, \mathsf{Sign}, \mathsf{Verify})$ *of PPT algorithms, which are defined as follows:*

$\mathsf{Gen}(1^\kappa)$: *This algorithm takes a security parameter κ as input and outputs a secret (signing) key* sk *and a public (verification) key* pk *with associated message space $\mathcal{M}$ (we may omit to make the message space $\mathcal{M}$ explicit).*

$\mathsf{Sign}(\mathsf{sk}, m)$: *This algorithm takes a secret key* sk *and a message $m \in \mathcal{M}$ as input and outputs a signature σ.*

$\mathsf{Verify}(\mathsf{pk}, m, \sigma)$: *This algorithm takes a public key* pk*, a message $m \in \mathcal{M}$ and a signature σ as input and outputs a bit $b \in \{0, 1\}$.*

Besides the usual correctness property, Σ needs to provide some unforgeability notion. In this paper we are only interested in schemes that provide existential unforgeability under adaptively chosen message attacks ($\mathsf{EUF\text{-}CMA}$ security), which we define below.

Definition 7 (EUF-CMA). *A signature scheme Σ is* EUF-CMA *secure, if for all PPT adversaries $\mathcal{A}$ there is a negligible function $\varepsilon(\cdot)$ such that*

$$\Pr\left[\begin{array}{l}(\mathsf{sk},\mathsf{pk}) \leftarrow \mathsf{Gen}(1^\kappa), \\ (m^\star,\sigma^\star) \leftarrow \mathcal{A}^{\mathsf{Sign}(\mathsf{sk},\cdot)}(\mathsf{pk})\end{array} : \begin{array}{r}\mathsf{Verify}(\mathsf{pk},m^\star,\sigma^\star) = 1 \wedge \\ m^\star \notin \mathcal{Q}^{\mathsf{Sign}}\end{array}\right] \leq \varepsilon(\kappa),$$

where the environment keeps track of the queries to the signing oracle via $\mathcal{Q}^{\mathsf{Sign}}$.

A.1 Security Properties of Non-Interactive Proof Systems

Definition 8 (Completeness). *A non-interactive proof system Π is* complete*, if for every adversary $\mathcal{A}$ it holds that*

$$\Pr\left[\begin{array}{l}\mathsf{crs} \leftarrow \mathsf{Setup}(1^\kappa),\ (x,w) \leftarrow \mathcal{A}(\mathsf{crs}), \\ \pi \leftarrow \mathsf{Proof}(\mathsf{crs},x,w)\end{array} : \begin{array}{r}\mathsf{Verify}(\mathsf{crs},x,\pi) = 1 \\ \wedge\ (x,w) \in R\end{array}\right] \approx 1.$$

Definition 9 (Soundness). *A non-interactive proof system Π is* sound*, if for every PPT adversary $\mathcal{A}$ there is a negligible function $\epsilon(\cdot)$ such that*

$$\Pr\left[\mathsf{crs} \leftarrow \mathsf{Setup}(1^\kappa),\ (x,\pi) \leftarrow \mathcal{A}(\mathsf{crs})\ :\ \mathsf{Verify}(\mathsf{crs},x,\pi) = 1\ \wedge\ x \notin L_R\right] \leq \epsilon(\kappa).$$

If we quantify over all adversaries $\mathcal{A}$ and require $\epsilon = 0$, we have perfect soundness, but we present the definition for computationally sound proofs (arguments).

Definition 10 (Adaptive Zero-Knowledge). *A non-interactive proof system Π is* adaptively zero-knowledge*, if there exists a PPT simulator $S = (\mathcal{S}_1,\mathcal{S}_2)$ such that for every PPT adversary $\mathcal{A}$ there is a negligible function $\epsilon(\cdot)$ such that*

$$\left|\begin{array}{l}\Pr\left[\mathsf{crs} \leftarrow \mathsf{Setup}(1^\kappa)\ :\ \mathcal{A}^{\mathcal{P}(\mathsf{crs},\cdot,\cdot)}(\mathsf{crs}) = 1\right] - \\ \Pr\left[(\mathsf{crs},\tau) \leftarrow \mathcal{S}_1(1^\kappa)\ :\ \mathcal{A}^{\mathcal{S}(\mathsf{crs},\tau,\cdot,\cdot)}(\mathsf{crs}) = 1\right]\end{array}\right| \leq \epsilon(\kappa),$$

where, τ denotes a simulation trapdoor. Thereby, $\mathcal{P}$ and $\mathcal{S}$ return $\perp$ if $(x,w) \notin R$ or $\pi \leftarrow \mathsf{Proof}(\mathsf{crs},x,w)$ and $\pi \leftarrow \mathcal{S}_2(\mathsf{crs},\tau,x)$, respectively, otherwise.

Definition 11 (Simulation Sound Extractability). *An adaptively zero-knowledge non-interactive proof system Π is simulation sound extractable, if there exists a PPT extractor $\mathcal{E} = (\mathcal{E}_1,\mathcal{E}_2)$ such that for every adversary $\mathcal{A}$ it holds that*

$$\left|\begin{array}{l}\Pr\left[(\mathsf{crs},\tau) \leftarrow \mathcal{S}_1(1^\kappa)\ :\ \mathcal{A}(\mathsf{crs},\tau) = 1\right] - \\ \Pr\left[(\mathsf{crs},\tau,\xi) \leftarrow \mathcal{E}_1(1^\kappa)\ :\ \mathcal{A}(\mathsf{crs},\tau) = 1\right]\end{array}\right| = 0,$$

and for every PPT adversary $\mathcal{A}$ there is a negligible function $\varepsilon_2(\cdot)$ such that

$$\Pr\left[\begin{array}{l}(\mathsf{crs},\tau,\xi) \leftarrow \mathcal{E}_1(1^\kappa), \\ (x^\star,\pi^\star) \leftarrow \mathcal{A}^{\mathcal{S}(\mathsf{crs},\tau,\cdot)}(\mathsf{crs}), \\ w \leftarrow \mathcal{E}_2(\mathsf{crs},\xi,x^\star,\pi^\star)\end{array} : \begin{array}{r}\mathsf{Verify}(\mathsf{crs},x^\star,\pi^\star) = 1 \wedge \\ (x^\star,\pi^\star) \notin \mathcal{Q}_{\mathsf{S}}\ \wedge\ (x^\star,w) \notin R\end{array}\right] \leq \varepsilon_2(\kappa),$$

where $\mathcal{S}(\mathsf{crs},\tau,x) \coloneqq \mathcal{S}_2(\mathsf{crs},\tau,x)$ and $\mathcal{Q}_{\mathsf{S}}$ keeps track of the queries to and answers of $\mathcal{S}$.

B Proof of Lemma 4

Proof. We prove Lemma 4 using a sequence of games, where S_i denotes the success probability in Game i.

Game 0: The original $\mathsf{EUF\text{-}CMA}$ game.

Game 1: As in the original game, but we change the winning condition by internally using the following modified Verify algorithm Verify':

$\mathsf{Verify}'(\mathsf{pk}, m, \sigma)$: Parse pk as (crs, c, β), and σ as (y, π). Return 1 if the following holds, and 0 otherwise:

$$\Pi.\mathsf{Verify}(\mathsf{crs}, (c, y, \beta, m), \pi) = 1 \;\wedge\; m \neq \beta \;\; \boxed{\wedge \;\; F_{\mathsf{s}}(m) = y}.$$

Transition - Game 0 → Game 1: Game 0 and Game 1 proceed identically, unless $F_{\mathsf{s}}(m) \neq y$. If this happens, we have—by the fixed-value-key-binding property of the PRF (which holds unconditionally)—a valid proof π falsely attesting that $(c, y, \beta, m) \in L_R$ (that is, fixed-value-key-binding ensures that there is no $\mathsf{s}' \neq \mathsf{s}$ such that $c = F_{\mathsf{s}'}(\beta)$ and we either have that $y = F_{\mathsf{s}}(m)$ or $(c, y, \beta, m) \notin L_R$). To obtain a contradiction we construct a reduction where we engage with a soundness challenger to obtain crs upon Gen. Whenever it happens that $F_{\mathsf{s}}(m) \neq y$ we output $((c, y, \beta, m), \pi)$ and break soundness. This gives us the bound $|\Pr[S_0] - \Pr[S_1]| \leq \epsilon_{\mathsf{s}}(\kappa)$.

Game 2: As Game 1, but we use the following modified key generation algorithm Gen':

$\mathsf{Gen}'(1^\kappa)$: Run $\boxed{(\mathsf{crs}, \tau) \leftarrow \mathcal{S}_1(1^\kappa) \text{ and store } \tau}$, run $\mathsf{s} \xleftarrow{R} \mathcal{S}$, compute $c \leftarrow F_{\mathsf{s}}(\beta)$, set $\mathsf{pk} \leftarrow (\mathsf{crs}, c, \beta)$ and $\mathsf{sk} \leftarrow (\mathsf{pk}, \mathsf{s})$ and return $(\mathsf{sk}, \mathsf{pk})$.

Furthermore, we use the following modified signing algorithm Sign' inside the signing oracle:

$\mathsf{Sign}'(\mathsf{sk}, m)$: Parse sk as $((\mathsf{crs}, c, \beta), \mathsf{s})$, compute $y \leftarrow F_{\mathsf{s}}(m)$, $\boxed{\pi \leftarrow \mathcal{S}_2(\mathsf{crs}, \tau,}$ $\boxed{(c, y, \beta, m))}$, and return $\sigma \leftarrow (y, \pi)$.

Transition - Game 1 → Game 2: To show that Game 1 and Game 2 are indistinguishable, we construct a reduction which uses the adaptive zero-knowledge challenger $\mathcal{C}^\kappa_{\mathsf{zk}}$ as an oracle to either simulate Game 1 or Game 2. We use the following key generation algorithm Gen':

$\mathsf{Gen}'(1^\kappa)$: Run $\boxed{\mathsf{crs} \leftarrow \mathcal{C}^\kappa_{\mathsf{zk}}}$, $\mathsf{s} \xleftarrow{R} \mathcal{S}$, compute $c \leftarrow F_{\mathsf{s}}(\beta)$, set $\mathsf{pk} \leftarrow (\mathsf{crs}, c, \beta)$ and $\mathsf{sk} \leftarrow (\mathsf{pk}, \mathsf{s})$ and return $(\mathsf{sk}, \mathsf{pk})$.

Furthermore, we use the following modified signing algorithm Sign' inside the signing oracle:

$\mathsf{Sign}'(\mathsf{sk}, m)$: Parse sk as $((\mathsf{crs}, c, \beta), \mathsf{s})$, compute $y \leftarrow F_{\mathsf{s}}(m)$, $\boxed{\pi \leftarrow \mathcal{C}^\kappa_{\mathsf{zk}}.\mathcal{P}/\mathcal{S}(}$ $\boxed{(c, y, \beta, m), \mathsf{s})}$, and return $\sigma \leftarrow (y, \pi)$.

Then, if the challenger samples from the first distribution we are in Game 1, whereas we are in Game 2 if the challenger samples from the second distribution (with $\mathcal{P}/\mathcal{S}$ we denote the oracle provided by the challenger which either honestly computes the proof or simulates it). Thus, we have that $|\Pr[S_2] - \Pr[S_1]| \leq \epsilon_{\mathsf{z}}(\kappa)$.

Game 3: As Game 2, but we further modify the internally used algorithms:

$\mathsf{Gen}'(1^\kappa)$: Run $(\mathsf{crs}, \tau) \leftarrow \mathcal{S}_1(1^\kappa)$ and store τ, choose $\boxed{c \xleftarrow{R} \{0,1\}^\kappa}$, set $\mathsf{pk} \leftarrow (\mathsf{crs}, c, \beta)$ and $\mathsf{sk} \leftarrow (\mathsf{pk}, \mathsf{s})$ and return $(\mathsf{sk}, \mathsf{pk})$.

$\mathsf{Sign}'(\mathsf{sk}, m)$: Parse sk as $((\mathsf{crs}, c, \beta), \mathsf{s})$, compute $\boxed{y \xleftarrow{R} \{0,1\}^\kappa}$, $\pi \leftarrow \mathcal{S}_2(\mathsf{crs}, \tau, (c, y, \beta, m))$, and return $\sigma \leftarrow (y, \pi)$.

$\mathsf{Verify}'(\mathsf{pk}, m, \sigma)$: Parse pk as (crs, c, β), and σ as (y, π), sample $\boxed{y' \xleftarrow{R} \{0,1\}^\kappa}$. Return 1 if the following holds, and 0 otherwise:

$$\Pi.\mathsf{Verify}(\mathsf{crs}, (c, y, \beta, m), \pi) = 1 \ \wedge \ m \neq \beta \ \boxed{\wedge \ \ y' = y} \ .$$

Transition - Game 2 → Game 3: A distinguisher $\mathcal{D}^{2\to3}$ is an adversary against the PRF. We construct a reduction where we let $\mathcal{C}^\kappa_{\mathsf{p}}$ be a PRF challenger providing the oracle $F_{\mathsf{s}}/f(\cdot)$:

$\mathsf{Gen}'(1^\kappa)$: Run $(\mathsf{crs}, \tau) \leftarrow \mathcal{S}_1(1^\kappa)$ and store τ, choose $\boxed{c \leftarrow \mathcal{C}^\kappa_{\mathsf{p}}.F_{\mathsf{s}}/f(\beta)}$, set $\mathsf{pk} \leftarrow (\mathsf{crs}, c, \beta)$ and $\mathsf{sk} \leftarrow (\mathsf{pk}, \mathsf{s})$ and return $(\mathsf{sk}, \mathsf{pk})$.

$\mathsf{Sign}'(\mathsf{sk}, m)$: Parse sk as $((\mathsf{crs}, c, \beta), \mathsf{s})$, compute $\boxed{y \leftarrow \mathcal{C}^\kappa_{\mathsf{p}}.F_{\mathsf{s}}/f(m)}$, $\pi \leftarrow \mathcal{S}_2(\mathsf{crs}, \tau, (c, y, \beta, m))$, and return $\sigma \leftarrow (y, \pi)$.

$\mathsf{Verify}'(\mathsf{pk}, m, \sigma)$: Parse pk as (crs, c, β), and σ as (y, π), sample $y' \xleftarrow{R} \{0,1\}^\kappa$. Return 1 if the following holds, and 0 otherwise:

$$\Pi.\mathsf{Verify}(\mathsf{crs}, (c, y, \beta, m), \pi) = 1 \ \wedge \ m \neq \beta \ \boxed{\wedge \ \ y = \mathcal{C}^\kappa_{\mathsf{p}}.F_{\mathsf{s}}/f(m)} \ .$$

If the challenger's bit is 0, we are in Game 2, whereas we are in Game 3 if the challengers bit is 1. Thus, $|\Pr[S_2] - \Pr[S_3]| \leq \epsilon_{\mathsf{p}}(\kappa)$.

In the last game, the adversary can only win by guessing y correctly, which happens with probability at most $2^{-\kappa}$, thus concluding the proof. □

C Proof of Lemma 5

Proof. We prove Lemma 5 using a sequence of games, where S_i denotes the success probability in Game i.

Game 0: The original $\mathsf{EUF\text{-}CMA}$ game.

Game 1: As Game 0, but we use the following modified key generation algorithm Gen':

$\mathsf{Gen}'(1^\kappa)$: Run $\boxed{(\mathsf{crs}, \tau) \leftarrow \mathcal{S}_1(1^\kappa) \text{ and store } \tau}$, run $\mathsf{s} \xleftarrow{R} \mathcal{S}$, compute $c \leftarrow F_{\mathsf{s}}(\beta)$, set $\mathsf{pk} \leftarrow (\mathsf{crs}, c, \beta)$ and $\mathsf{sk} \leftarrow (\mathsf{pk}, \mathsf{s})$ and return $(\mathsf{sk}, \mathsf{pk})$.

Furthermore, we use the following modified signing algorithm Sign' inside the signing oracle:

$\mathsf{Sign}'(\mathsf{sk}, m)$: Parse sk as $((\mathsf{crs}, c, \beta), \mathsf{s})$, compute $y \leftarrow F_{\mathsf{s}}(m)$, $\boxed{\pi \leftarrow \mathcal{S}_2(\mathsf{crs}, \tau, (c, y, \beta, m))}$, and return $\sigma \leftarrow (y, \pi)$.

Transition - Game 0 → Game 1: To show that Game 0 and Game 1 are indistinguishable, we construct a reduction that uses the adaptive zero-knowledge challenger $\mathcal{C}^\kappa_{\mathsf{zk}}$ as an oracle to either simulate Game 0 or Game 1. We use the following key generation algorithm Gen':

$\mathsf{Gen}'(1^\kappa)$: Run $\boxed{\mathsf{crs} \leftarrow \mathcal{C}^\kappa_{\mathsf{zk}}}$, $\mathsf{s} \xleftarrow{R} \mathcal{S}$, compute $c \leftarrow F_\mathsf{s}(\beta)$, set $\mathsf{pk} \leftarrow (\mathsf{crs}, c, \beta)$ and $\mathsf{sk} \leftarrow (\mathsf{pk}, \mathsf{s})$ and return $(\mathsf{sk}, \mathsf{pk})$.

Furthermore, we use the following modified signing algorithm Sign' inside the signing oracle:

$\mathsf{Sign}'(\mathsf{sk}, m)$: Parse sk as $((\mathsf{crs}, c, \beta), \mathsf{s})$, compute $y \leftarrow F_\mathsf{s}(m)$, $\boxed{\pi \leftarrow \mathcal{C}^\kappa_{\mathsf{zk}}.\mathcal{P}/\mathcal{S}(}$ $\boxed{(c, y, \beta, m), \mathsf{s})}$, and return $\sigma \leftarrow (y, \pi)$.

Then, if the challenger samples from the first distribution we are in Game 0, whereas we are in Game 1 if the challenger samples from the second distribution (with $\mathcal{P}/\mathcal{S}$ we denote the oracle provided by the challenger which either honestly computes the proof or simulates it). Thus, we have that $|\Pr[S_0] - \Pr[S_1]| \leq \epsilon_\mathsf{z}(\kappa)$.

Game 2: As in Game 2, but we further modify the key generation algorithm Gen':

$\mathsf{Gen}'(1^\kappa)$: Run $\boxed{(\mathsf{crs}, \tau, \xi) \leftarrow \mathcal{E}_1(1^\kappa) \text{ and store } \tau \text{ and } \xi}$, run $\mathsf{s} \xleftarrow{R} \mathcal{S}$, compute $c \leftarrow F_\mathsf{s}(\beta)$, set $\mathsf{pk} \leftarrow (\mathsf{crs}, c, \beta)$ and $\mathsf{sk} \leftarrow (\mathsf{pk}, \mathsf{s})$ and return $(\mathsf{sk}, \mathsf{pk})$.

Transition - Game 1 → Game 2: This change is conceptual, i.e., $\Pr[S_1] = \Pr[S_2]$.

Game 3: As Game 2, but for every forgery output by the adversary, we extract $\mathsf{s}' \leftarrow \mathcal{E}_2(\mathsf{crs}, \xi, (c, y, \beta, m), \pi)$. If the extractor fails, we abort.

Transition - Game 2 → Game 3: Both games proceed identically unless the abort event happens. That is, $|\Pr[S_2] - \Pr[S_3]| \leq \epsilon_\mathsf{e}(\kappa)$.

Game 4: As in Game 3, but we abort if we extract an s' so that $\mathsf{s}' \neq \mathsf{s}$.

Transition - Game 3 → Game 4: Both games proceed identically, unless the abort event happens. Now, assume for the sake of contradiction that the abort event happens with non-negligible probability. Then we could simulate the following game, where we use $\mathcal{C}^\kappa_\mathsf{b}$ to denote a fixed-value-key-binding challenger for the PRF:

$\mathsf{Gen}'(1^\kappa)$: Run $(\mathsf{crs}, \tau, \xi) \leftarrow \mathcal{E}_1(1^\kappa)$ and store τ and ξ, run $\boxed{\mathsf{s} \leftarrow \bot}$, compute $(c, \beta) \leftarrow \mathcal{C}^\kappa_\mathsf{b}$, set $\mathsf{pk} \leftarrow (\mathsf{crs}, c, \beta)$ and $\mathsf{sk} \leftarrow (\mathsf{pk}, \mathsf{s})$ and return $(\mathsf{sk}, \mathsf{pk})$.

$\mathsf{Sign}'(\mathsf{sk}, m)$: Parse sk as $((\mathsf{crs}, c, \beta), \bot)$, compute $\boxed{y \leftarrow \mathcal{C}^\kappa_\mathsf{b}.F_\mathsf{s}(m)}$, $\pi \leftarrow \mathcal{S}_2(\mathsf{crs}, \tau, (c, y, \beta, m))$, and return $\sigma \leftarrow (y, \pi)$.

Whenever the adversary outputs a forgery we extract s' and output it to $\mathcal{C}^\kappa_\mathsf{b}$. Now, it is clear that the abort event has the same probability as breaking the computational fixed-value-key-binding property of the PRF and thus $|\Pr[S_3] - \Pr[S_4]| \leq \epsilon_\mathsf{b}(\kappa)$.

Game 5: As in Game 4, but we change the winning condition by internally using the following modified Verify algorithm Verify':

$\mathsf{Verify}'(\mathsf{pk}, m, \sigma)$: Parse pk as (crs, c, β), and σ as (y, π). Return 1 if the following holds, and 0 otherwise:

$$\Pi.\mathsf{Verify}(\mathsf{crs}, (c, y, \beta, m), \pi) = 1 \ \wedge \ m \neq \beta \ \boxed{\wedge \ \ F_\mathsf{s}(m) = y}.$$

Transition - Game 4 → Game 5: In the previous games we already established that we only continue if we successfully extract $\mathsf{s}' = \mathsf{s}$ so that $(c, y, \beta, m) \in L_R$. Thus, the additional check in the winning condition will always be successful and we will accept the same forgeries as in Game 4, i.e., $\Pr[S_4] = \Pr[S_5]$.

Game 6: As Game 5, but we no longer abort if $\mathsf{s} \neq \mathsf{s}'$.

Transition Game 5 → Game 6: In Game 6 accept more forgeries than in Game 5. Thus, the probability in Game 6 is larger than or equal to the winning probability in Game 5, i.e., $\Pr[S_6] \geq \Pr[S_5]$.

Game 7: As Game 6, but we further modify the internally used algorithms:

$\mathsf{Gen}'(1^\kappa)$: Run $(\mathsf{crs}, \tau, \xi) \leftarrow \mathcal{E}_1(1^\kappa)$ and store τ and ξ, run $\mathsf{s} \xleftarrow{R} \mathcal{S}$, choose $\boxed{c \xleftarrow{R} \{0,1\}^\kappa}$, set $\mathsf{pk} \leftarrow (\mathsf{crs}, c, \beta)$ and $\mathsf{sk} \leftarrow (\mathsf{pk}, \mathsf{s})$ and return $(\mathsf{sk}, \mathsf{pk})$.

$\mathsf{Sign}'(\mathsf{sk}, m)$: Parse sk as $((\mathsf{crs}, c, \beta), \mathsf{s})$, compute $\boxed{y \xleftarrow{R} \{0,1\}^\kappa}$, $\pi \leftarrow \mathcal{S}_2(\mathsf{crs}, \tau, (c, y, \beta, m))$, and return $\sigma \leftarrow (y, \pi)$.

$\mathsf{Verify}'(\mathsf{pk}, m, \sigma)$: Parse pk as (crs, c, β), and σ as (y, π), sample $\boxed{y' \xleftarrow{R} \{0,1\}^\kappa}$. Return 1 if the following holds, and 0 otherwise:

$$\Pi.\mathsf{Verify}(\mathsf{crs}, (c, y, \beta, m), \pi) = 1 \ \wedge \ m \neq \beta \ \boxed{\wedge \ \ y' = y}\ .$$

Transition - Game 6 → Game 7: A distinguisher $\mathcal{D}^{6 \to 7}$ is an adversary against the PRF. Consider the following hybrid Game, where we let $\mathcal{C}_{\mathsf{p}}^\kappa$ be a PRF challenger providing the oracle $F_{\mathsf{s}}/f(\cdot)$:

$\mathsf{Gen}'(1^\kappa)$: Run $(\mathsf{crs}, \tau, \xi) \leftarrow \mathcal{E}_1(1^\kappa)$ and store τ and ξ, set $\boxed{c \leftarrow \mathcal{C}_{\mathsf{p}}^\kappa.F_{\mathsf{s}}/f(\beta)}$, set $\mathsf{pk} \leftarrow (\mathsf{crs}, c, \beta)$ and $\mathsf{sk} \leftarrow (\mathsf{pk}, \mathsf{s})$ and return $(\mathsf{sk}, \mathsf{pk})$.

$\mathsf{Sign}'(\mathsf{sk}, m)$: Parse sk as $((\mathsf{crs}, c, \beta), \mathsf{s})$, set $\boxed{y \leftarrow \mathcal{C}_{\mathsf{p}}^\kappa.F_{\mathsf{s}}/f(m)}$, $\pi \leftarrow \mathcal{S}_2(\mathsf{crs}, \tau, (c, y, \beta, m))$, and return $\sigma \leftarrow (y, \pi)$.

$\mathsf{Verify}'(\mathsf{pk}, m, \sigma)$: Parse pk as (crs, c, β), and σ as (y, π), set $\boxed{y' \leftarrow \mathcal{C}_{\mathsf{p}}^\kappa.F_{\mathsf{s}}/f(m)}$. Return 1 if the following holds, and 0 otherwise:

$$\Pi.\mathsf{Verify}(\mathsf{crs}, (c, y, \beta, m), \pi) = 1 \ \wedge \ m \neq \beta \ \boxed{\wedge \ \ y' = y}.$$

If the challenger's bit is 0, we are in Game 6, whereas we are in Game 7 if the challengers bit is 1. Thus, $|\Pr[S_6] - \Pr[S_7]| \leq \epsilon_{\mathsf{p}}(\kappa)$.

In the last game, the adversary can only win by guessing y correctly, which happens with probability at most $2^{-\kappa}$, thus concluding the proof. □

D Description of LowMC

LowMC by Albrecht et al. [ARS+15, ARS+16] is very parametrizable symmetric encryption scheme design enabling instantiation with low AND depth and low multiplicative complexity. Given any blocksize, a choice for the number of S-boxes per round, and security expectations in terms of time and data complexity, instantiations can be created minimizing the AND depth, the number of ANDs, or the number of ANDs per encrypted bit. Table 2 lists the choices for the parameters which are also highlighted in the figures. The files with all parameters used for our experiments are provided online at https://github.com/IAIK/fish-begol.

The description of LowMC is possible independently of the choice of parameters using a partial specification of the S-box and arithmetic in vector spaces

Blocksize n	**S-boxes** m	**Keysize** k	**Data** d	**Rounds** r	**# of ANDs**	**ANDS per bit**
128	2	128	128	128	768	6.00
128	12	128	128	26	936	7.31
128	32	128	128	19	1824	14.25
256	2	256	256	174	1044	4.08
256	2	256	256	232	1392	5.44
256	20	256	256	31	1860	7.27
256	26	256	256	18	1404	7.31
256	64	256	256	14	2688	10.5
256	80	256	256	22	5280	20.62
384	4	384	384	172	2064	5.38
384	30	384	384	33	2970	7.73
384	126	384	384	25	9450	24.61
512	6	512	512	152	2736	5.34
512	34	512	512	37	3774	7.37
512	170	512	512	25	12750	24.9

Table 2. A range of different parameter sets for LowMC. The number of rounds corresponds to the AND depth.

over $\mathbb{F}_2$. In particular, let n be the blocksize, m be the number of S-boxes, k the key size, and r the number of rounds, we choose round constants $C_i \xleftarrow{R} \mathbb{F}_n$ for $i \in [1, r]$, full rank matrices $K_i \xleftarrow{R} \mathbb{F}_2^{n \times k}$ and regular matrices $L_i \xleftarrow{R} \mathbb{F}_2^{n \times n}$ independently during the instance generation and keep them fixed. Keys for LowMC are generated by sampling from $\mathbb{F}_2^k$ uniformly at random.

LowMC encryption starts with key whitening which is followed by several rounds of encryption. A single round of LowMC is composed of an S-box layer, a linear layer, addition with constants and addition of the round key, i.e.

$$\begin{aligned}\textsc{LowMCRound}(i) = \textsc{KeyAddition}(i) \circ \textsc{ConstantAddition}(i) \\ \circ\, \textsc{LinearLayer}(i) \circ \textsc{SboxLayer}.\end{aligned}$$

SboxLayer is an m-fold parallel application of the same 3-bit S-box on the first $3 \cdot m$ bits of the state. The S-box is defined as $S(a, b, c) = (a \oplus bc, a \oplus b \oplus ac, a \oplus b \oplus c \oplus ab)$.

The other layers only consist of $\mathbb{F}_2$-vector space arithmetic. LinearLayer(i) multiplies the state with the linear layer matrix L_i, ConstantAdditon(i) adds the round constant C_i to the state, and KeyAddition(i) adds the round key to the state, where the round key is generated by multiplying the master key with the key matrix K_i.

Algorithm 1 gives a full description of the encryption.

Algorithm 1 LowMC encryption for key matrices $K_i \in \mathbb{F}_2^{n \times k}$ for $i \in [0, r]$, linear layer matrices $L_i \in \mathbb{F}_2^{n \times n}$ and round constants $C_i \in \mathbb{F}_2^n$ for $i \in [1, r]$.

```
Input: plaintext p ∈ F_2^n and key y ∈ F_2^k
  s ← K_0 · y + p
  for i ∈ [1, r] do
      s ← Sbox(s)
      s ← L_i · s
      s ← C_i + s
      s ← K_i · y + s
  end for
  return s
```

E List of Symbols

- t: running time of adversary;
- ϵ: success probability of adversary;
- κ: security parameter;
- Q_H: adversarial queries to oracle H;
- H: random oracle used to generate Fiat-Shamir challenge;
- γ: number of repetition of the Σ-protocol and output size of H;
- H': random oracle used to implement commitments in ZKBoo;
- α: size of random tapes for each thread in ZKBoo;
- β: size of "view" in each thread in ZKBoo;
- ν: size of salt used when committing with H';
- ρ: size of commitments and output size of H';
- n: size of PRF output;
- a_f: number of multiplication gates in circuit implementing f;
- λ_f: size of ring in which the multiplications take place;
- D_f: bit-length of elements from domain of f;
- I_f: bit-length of elements from image of f;
- r: number of rounds in LowMC instance;
- m: number of SBoxes in LowMC instance;