IACR Communications in Cryptology
ISSN 3006-5496, Vol. 2, No. 1, 27 pages.
https://doi.org/10.62056/an-4c3c2h

Beyond the Circuit

How to minimize foreign arithmetic in ZKP circuits

Michele Orrù[1], George Kadianakis[2], Mary Maller[2,3] and Greg Zaverucha[4]

[1] CNRS, FR
[2] Ethereum Foundation, CH
[3] PQ Shield, UK
[4] Microsoft Research, US

Abstract. A fundamental challenge in zero-knowledge proof systems is implementing operations that are "foreign" to the underlying constraint system, in that they are arithmetic operations with a different modulus than the one used by the proof system. The modulus of the constraint system is a large prime, and common examples of foreign operations are Boolean operations, field arithmetic, or public-key cryptography operations. We present novel techniques for efficiently embedding such foreign arithmetic in zero-knowledge, including (i) equality of discrete logarithms across different groups; (ii) scalar multiplication without requiring elliptic curve operations; (iii) proving knowledge of an AES encryption. Our approach combines rejection sampling, sigma protocols, and lookup protocols. We implement and provide concrete benchmarks for our protocols.

1 Introduction

Zero-knowledge proofs [GMR89] allow a prover to convince a verifier about the truth of a statement without revealing more information than its validity. They are a core tool in complexity theory, cryptography, and security. Over the decades, the cryptographic community has witnessed a transformative evolution of zero-knowledge proofs, from theoretical tools to practical systems, with a focus on proving statements about NP relations. The surge of interest in real-world applications, such as blockchain scalability [BCL+21], private payments [BCG+14], and more recently authenticity of images and documents [KHSS22; BCG+22], has catalyzed the development of efficient zero-knowledge proofs across a variety of systems.

The engineering effort involved in expressing zero-knowledge proof statements is significant. Not only is efficiency a challenge, but it requires a non-trivial amount of complex code, and is prone to errors.[1] Mistakes in the instantiation of the statement, particularly when constraints are missing, can easily void the security guarantees of the proof system [ZKB].

The case of foreign arithmetic is particularly challenging. To write in a SNARK circuit operating over $\mathbb{F}_p$ proving knoweledge of some secret x such that $X = g^x \bmod q$, where g is the generator of some finite group, a translator is needed. This translator must map the modular arithmetic performed modulo q into circuit arithmetic compatible with the modulus p of the field in which the statement is defined. Additionally, in the case of elliptic curve groups, the group law must be implemented directly within the circuit. Consequently, both the efficiency overhead and the attack surface grow dramatically.

E-mail: m@orru.net (Michele Orrù), george.kadianakis@ethereum.org (George Kadianakis), mary.maller@ethereum.org (Mary Maller), gregz@microsoft.com (Greg Zaverucha)

[1] https://penumbra.zone/pdfs/zksecurity_penumbra_2023.pdf

Received: 2025-01-13 Accepted: 2025-03-11

In practice, circuit programming generally relies on domain-specific languages[2] and a large engineering effort has been put into securely programming zero-knowledge gadgets for computations over (i) symmetric-key operations, (ii) public-key operations, and (iii) arithmetic over a foreign finite field. In this paper, we provide techniques that avoid the need of writing the circuit gadget altogether.

1.1 Our contributions

We provide three protocols:

Protocol Π_{dleq}. Let $\mathbb{G}_p$ and $\mathbb{G}_q$ to denote groups of different prime order p and q, with generators (G_p, H_p) and (G_q, H_q). We give a Σ-protocol to prove that commitments in $\mathbb{G}_p$, and $\mathbb{G}_q$ have the same secret opening x. Our protocol (Figure 1) requires 3–6 scalar multiplications in each of $\mathbb{G}_p$ and $\mathbb{G}_q$ (more details in Table 2) and as little as 81 bytes. To achieve our result, we leverage rejection sampling and the Fiat–Shamir with aborts paradigm. While this proof can always be done with zero-knowledge proofs for NP statements, generic approaches proposed so far fail to deliver efficient proofs.

Protocol Π_{dlhash}. We offload elliptic curve scalar multiplications from a circuit (Figure 2). Consider a circuit tasked with proving that $X = xG$ where (X, G) are public elliptic curve points and x is a secret integer. Instead of performing scalar multiplication within the constraint system, we prove knowledge of x outside the circuit (in a Schnorr proof (R, c, z)), and then bind this proof to the circuit. To bind the proofs, we include a hash $h = H(x, r)$ of the witness x and the commitment randomness r in the first flow of the Σ-protocol. Within the circuit, we prove knowledge of the preimage of H and verify that the public response z is computed correctly as $z = r + cx$. This method is most efficient when H is a ZK-friendly hash function, such as Poseidon, reducing the cost from tens of thousands of circuit constraints to approximately 300 additional constraints and just two scalar multiplications. Even with standard hash functions, this approach outperforms the computation of a scalar multiplication within the circuit. We evaluate an implementation of our protocol [arn] in Section 4.2.

Protocol Π_{aes}. We provide a gadget that reduces in zero-knowledge the Rijndael (AES) cipher to a lookup protocol (Figure 3). Given a committed AES key a committed plaintext, we show that a public ciphertext is correctly computed from them. While any lookup argument can be used here, to showcase the simplicity of our protocol, in [OKMZ24, Appendix C] we present a lookup argument that relies solely on (compressed) Σ-protocols and is of independent interest. For AES-128 our zero-knowledge proof runs in about 30ms. Using compressed Σ-protocols proof size is 2 848 bytes, and the verifier time is below 20ms. An evaluation of our open-source implementation can be seen in Table 4. The code is open-source and released under the BSD license.[3]

1.2 Applications

We list below some applications that can benefit from our techniques.

Linking credentials and assets inside proofs. One overarching theme of our contributions is the ability to easily and efficiently *link* commitments and credentials inside zero-knowledge proofs.

[2] Some examples: Cairo, Circom, Halo2, Leo, Noir.
[3] `https://github.com/mmaker/tinybear`

Linking Assets. Cross-chain asset transfers between Monero and Ethereum currently require two separate range proofs[4]. Using our protocol Π_{dleq}, we can implement an asset transfer protocol with half the proof size and computational complexity compared to the existing solution. This difference in efficiency is particularly significant in blockchain systems, where space optimization is crucial. Moreover, the existing solution lacks rigorous security analysis.

Linking anonymous credentials. Anonymous credentials can include multiple user attributes, such as account identifiers, email addresses, and social security numbers [CL04; BBS04; ASM06; SAB+19; CMZ14; BBDT16; CDDH19; CPZ20]. Using our Π_{dleq} protocol, credentials can be securely linked through a common attribute, such as a user ID (say, a small 128-bit scalar), while maintaining anonymity. This linking capability works seamlessly regardless of whether the credentials are issued by the same authority or across different issuers. For cross-issuer scenarios, the second issuer can independently incorporate attributes during blind issuance, though this requires mutual trust in the respective credential security.

In-circuit proof and signature verification. To implement blind swaps on Bitcoin, one must construct concurrently-secure blind Schnorr signatures compatible with `ed25519`. Fuchsbauer and Wolf [FW24] achieve this by modifying the blind Schnorr protocol to prove correct computation of the blinded challenge via a SNARK. Π_{dlhash} eliminates the need for non-native scalar multiplication in their circuit, which currently accounts for 96% of the code.[5] For context, implementing non-native field multiplication for BN254 curves incurs a 600x overhead in R1CS constraints.[6] This optimization extends to blockchain bridge protocols where signature verification dominates circuit complexity [XZC+22; DD23; JBK+24].

AES Middleboxes and verifiable encryption. To implement zk-middleboxes [GAZ+22], AES proofs are essential. While the approach taken in the literature uses xJsnark's circuit [KPS18] requires 14K constraints, our solution achieves the same goal with just 2K constraints. This translates to approximately 21x fewer group operations. Furthermore, our Π_{aes} is naturally extensible to multi-blocks and common block cipher modes of operation. It also fits within the Commit-and-Prove Zero-Knowledge Proof (CP-ZKP) framework—originally introduced by Kilian [Kil90], extended by Canetti et al. [CLOS02], and recently proposed for standardization [BCF+21b]—enabling direct links between AES-encrypted messages, keys, and Pedersen commitments.

1.3 Related work

Discrete Logarithm EQuality (DLEQ) proofs. Without RSA groups, R_{dleq} is not easy. A previous attempt at proving discrete logarithm equality across two generic DL groups was addressed by Agrawal, Ganesh, and Mohassel [AGM18, Appendix D Fig. 12], who also underline the applications for extending SNARKs. However, the protocol is more involved than ours: it has soundness error 1/2 and requires soundness amplification (the prover performs a binary decomposition of the secret, commits to each bit in both groups, and then cut-and-choose over the bit openings in $\mathbb{G}_p$ and $\mathbb{G}_q$). Their proof size is $O(\lambda(\log q + \log p))$, whereas ours is $O(\min(\log(p), \log(q)))$ when a range proof is needed

[4] See `https://www.getmonero.org/resources/research-lab/pubs/MRL-0010.pdf`. Implementation: `https://github.com/AthanorLabs/atomic-swap`.

[5] Specifically, this eliminates code from `https://github.com/mottla/Blind-Schnorr-Signatures/tree/main/secp256k1_non_native_modP`

[6] Compared to arkwork's non-native library: `https://github.com/arkworks-rs/nonnative`.

in the outside protocol and $O(1)$ for the cases without (this is the case of all examples in the applications section).

The special case where $p = q$ has been studied by Chaum and Pedersen [CP93]. Benarroch et al. [BCF+21a] provide a protocol for proving equality of commitments over $\mathbb{Z}_N^*$ and elliptic-curve groups. The problem of efficiently proving discrete logarithm equality across different groups can be found in Camenisch and Lysyanskaya [CL02], who describe an efficient zero-knowledge proof of knowledge that a committed value is in an accumulator. Values are committed in a group where the discrete logarithm (DL) is hard, while the accumulator is constructed in an RSA group. The problem considered in this work is slightly different, because we consider two groups where DL is hard. In the cryptocurrency area, the problem was already highlighted in Zerocoin [MGGR13], where they use the same techniques of Camenisch and Lysyanskaya [CL02] to provide an anonymous cryptocurrency. Dagher et al. [DBB+15] provide proofs of assets, solvency and non-collusion for Bitcoin, evoking the need of zkSNARKs for efficiency but the associated cost in expressing a large circuit. Sun et al. [SSS+22] formulate the problem of proving discrete logarithm equality across pairing-friendly and non-pairing-friendly groups.

The aborting technique we use to avoid leaking information about the secret when the prover sends a response computed over the integers originates in [Lyu08; Lyu09], where it was used in the context of lattice-based signatures. It then was adapted to signatures based on the short discrete log problem in Abdalla et al. [AFLT12]. The setting of this latter work is closer to ours, and we use the main lemma from it in our analysis.

Public-key operations. Chase et al. [CGM16] introduced a technique that combines algebraic-based proof protocols, such as Σ-protocols, with proofs based on garbled circuits. This integration efficiently handles algebraic operations in the former and non-algebraic operations (e.g. hash functions) in the latter. The linkage between these proof systems relies on using a *private* garbling scheme to compute a one-time MAC of the witness and then proving the correctness of the MAC using a Σ-protocol. However, this technique requires garbled circuits with privacy properties, as the verifier learning the MAC value directly reveals the witness. As a result, the approach is not immediately applicable to proof systems that do not employ private garbled circuits.

GoblinPlonk[7] introduces a mechanism for deferring expensive operations in SNARK circuits by batching them into a single expensive step: when encountering an expensive operation $X = xG$, the prover defers the actual computation and directly provides the final result for X. Once multiple such operations have been deferred, a specialized circuit verifies the correctness of all deferred operations in a single step. This strategy is orthogonal and compatible with our techniques.

Ben-Sasson, Chiesa, and Tromer [BCTV14] introduced the notion of *curve cycles*, elliptic curves where the scalar field of one curve is equal to the coordinate field of another. Note that the case where the scalar field is the coordinate field is called *anomalous curve*, and they are susceptible to attacks [Sma99; Yas12]. This approach has been shown itself extremely powerful, but also dangerous: little is known about, and deferred computations in recursive settings [Val08; CT10; BGH19] have already had a history of subtle vulnerabilities arising only when the elliptic curve is instantiated with cycles [NBS23].

LegoSNARK [CFQ19] introduced a generic framework for linking different proof systems. Using the commit-and-prove paradigm [Kil90; CLOS02], it provides a framework and generic compiler to facilitate the generic integration of proof systems and demonstrates its applicability across various use cases. Π_{dlhash} can be seen as providing an efficient specialized LegoSNARK link between Σ-protocols and generic SNARK protocols.

[7] https://hackmd.io/@aztec-network/B19AA8812

Symmetric-key operations. Designing zero-knowledge (zk) circuits for traditional symmetric primitives has long been challenging due to the non-algebraic nature of these primitives. This often introduces substantial computational overhead in zk settings.

A significant line of research addresses this issue by leveraging secure multi-party computation (MPC) techniques to construct zk proofs for symmetric-key operations. Among these, MPC-in-the-Head (MPCitH) is a prominent approach, offering highly efficient performance for small Boolean circuits. This has led to practical applications such as AES pre-image proofs, which gained renewed attention during the NIST post-quantum cryptography standardization process.[8] However, while MPCitH provides post-quantum security guarantees, it comes with limitations:

Incompatibility with IOP-Based Proof Systems. Unlike our proposed zero-knowledge proof system, MPCitH is inherently incompatible with the larger ecosystem of Interactive Oracle Proofs (IOPs) like Plonk [GWC19], Halo2 [zca], and Marlin [CHM+20]. These proof systems rely on specific commitment schemes and arithmetic structures that are difficult to reconcile with MPCitH approaches. In contrast, our approach is designed to integrate seamlessly with these systems, ensuring compatibility with widely adopted zk ecosystems.

Proof Size and Efficiency Trade-Offs. MPCitH-based techniques tend to produce large proofs, which can be problematic in contexts like blockchain applications where minimizing proof size is crucial. In most blockchain zero-knowledge applications, relying on DL-based assumptions is the only available choice to reduce size and optimize Ethereum gas costs. Our approach, which relies on Pedersen commitments, avoids this issue by leveraging commitments well-suited to large-field arithmetic, though it remains inherently non-post-quantum sound.

2 Preliminaries

We denote by $(\mathbb{G}, p, G, H)$ the description of a group $\mathbb{G}$ of prime order p, with two "nothing-up-my-sleeve" generators G, H (that is, two generators in $\mathbb{G}$ such that the discrete logarithm of H to the base G is not known to anyone). Groups are additive, and given a scalar $x \in \mathbb{Z}_p$, xG indicates scalar multiplication. When needing multiple "nothing-up-my-sleeve" (NUMS) generators (that is, generators whose respective DL is not known), we will consider $G_1, G_2, \ldots, G_n, H$. We denote probabilistic algorithms in sans-serif, and by writing $y \leftarrow \mathsf{M}(x)$ we denote the act of sampling the value y from the probabilistic algorithm M on input x. We assume that probabilistic algorithms run in time polynomial in the security parameter λ (abbrev p.p.t.) and have the security parameter implicitly as input. We use standard vector notation: by $\mathbf{x} \in \mathbb{Z}_p^n$ we refer to elements $(x_1, x_2, \ldots, x_n)$, with $\langle \mathbf{x}, \mathbf{y} \rangle$ we denote the inner-product $\sum_i x_i y_i$ and by $\mathbf{x} \otimes \mathbf{y}$ the "vectorized" tensor product $[x_i y_j]_{i \cdot n + j}$.

DL assumption. The Discrete Logarithm problem asks, given a group generator GrGen, a group description $(\mathbb{G}, p, G) \leftarrow \mathsf{GrGen}(1^\lambda)$ and a uniformly-random group element $X \leftarrow\$ \mathbb{G}$, to find $x \in \mathbb{Z}_p$ such that $X = xG$. The *discrete logarithm (DL) is hard* for GrGen if no p.p.t. algorithm solves the discrete logarithm problem with more than $\mathsf{negl}(\lambda)$ advantage.

Pedersen commitments. Pedersen's commitment scheme [Ped92] lets us *commit* to a value $x \in \mathbb{Z}_p$. To do so, sample $r \leftarrow\$ \mathbb{Z}_p$ and set

$$C := xG + rH\,.$$

[8] https://csrc.nist.gov/Projects/post-quantum-cryptography

We say that C is a Pedersen commitment. A pair $(x, r) \in \mathbb{Z}_p^2$ is a *valid opening* if $C = xG + rH$. Pedersen commitments are *perfectly hiding* and *computationally binding* under the discrete logarithm assumption. Informally, perfectly hiding means that no information about the pair (x, r) is revealed by C. Computationally binding means that no efficient adversary can produce two different valid openings (x, r) and (x', r') for a commitment C. Any adversary that given as input a group description is able to output a commitment C along with two distinct valid openings immediately gives a solution to an instance of DL. In fact, if (x, r) and (x', r') are a pair of valid openings, then $\log_G H = (r - r')^{-1}(x - x')$.

We will also use the well-known fact that Pedersen commitments are *additively homomorphic*: given commitments C, C', the sum of the openings $(x + x',\ r + r')$ is valid for the sum of the commitments $C + C'$. In addition, when committing to multiple elements $x_1, x_2, x_3, \cdots, x_n$ we will use the notation $C = \sum_i x_i G_i + rH$ as the commitment to the vector $x = (x_1, x_2, x_3, \ldots, x_n)$.

Σ-protocols. We recap the standard of Σ-protocols from Cramer [Cra97] (as described in Boneh–Shoup [BS20, §19.4]), with a few minor changes to model the prover's ability to abort the protocol. Let R be a binary relation of instances denoted by ϕ and witnesses denotes by w. By $\mathsf{R}(\phi)$ we denote the set of possible witnesses for the instance ϕ in R. A Σ-protocol for the relation R is a three-move protocol between a prover (with inputs ϕ and w) and a verifier (with input ϕ) consisting of a triple of efficient algorithms $(\mathsf{Com}, \mathsf{Ch}, \mathsf{Resp})$ run as follows:

- the prover executes $(a, \rho) \leftarrow \mathsf{Com}(\phi, w)$, sends a and internally stores the state ρ. Com is a randomized algorithm and may have additional inputs such as the group description and security parameter
- the verifier sends $c \leftarrow \mathsf{Ch}()$ to the prover; c is distributed uniformly at random from a fixed set of possible challenges
- the prover calls $\mathsf{Resp}(\phi, w, \rho, c)$ which may return some value z or abort (in which case we consider $z = \perp$)
- finally, the verifier calls $\mathsf{Verify}(\phi, (a, c, z))$ which returns a bit $b \in \{0, 1\}$. If $b = 1$ the verifier accepts the proof, otherwise rejects.

The tuple of exchanged messages (a, c, z) is called *transcript*; a is called commitment, c is called challenge, and z response. An *accepting transcript* (a, c, z) for ϕ is a transcript for which $\mathsf{Verify}(\phi, (a, c, z)) = 1$. Σ-protocols must satisfy:

- **Completeness:** A Σ-protocol is δ-complete if honestly-generated transcripts always verify, except when the prover aborts (with probability δ). More formally, for all honestly generated transcripts (a, c, z) and $(\phi, w) \in \mathsf{R}$ we have that

  $$\Pr[\mathsf{Verify}(\phi, a, c, z) = 1 \mid z \neq \perp] = 1\,, \text{ and } \Pr[z = \perp] = \delta$$

  over the choice of prover randomness.

- **Special soundness:** A Σ-protocol is (computationally) special sound if there exists an efficient extractor Ext such that for any p.p.t. adversary outputting an instance ϕ and two (non-aborting) accepting transcripts $(a, c, z), (a, c', z')$ for ϕ such that $c \neq c'$, $\mathsf{Ext}(\phi, (a, c, z), (a, c', z'))$ returns a valid witness $w \in \mathsf{R}(\phi)$ except with probability ϵ. The probability ϵ is called the *knowledge error* of the protocol.

- **Honest verifier zero-knowledge:** A Σ-protocol is honest verifier zero-knowledge (HVZK) if there exists an efficient simulator algorithm Sim such that for all $(\phi, w) \in \mathsf{R}$ the distributions

  $$\{(a, c, z) \mid c \leftarrow \mathsf{Ch}(); (a, z) \leftarrow \mathsf{Sim}(\phi, c)\}, \text{ and}$$
  $$\{(a, c, z) \mid c \leftarrow \mathsf{Ch}(); (a, \rho) \leftarrow \mathsf{Com}(\phi, w);\ z \leftarrow \mathsf{Resp}(\phi, w, \rho, c)\}$$

are indistinguishable. Our definition is sometimes referred to as *special HVZK* – *special* since the challenge is input to the simulator, as opposed to being chosen by the simulator. If the two distributions are perfectly indistinguishable (which can be the case for all our protocols), we will say the protocol enjoys perfect special HVZK since the simulated distribution is identical to the real one.

Two example Σ-protocols relevant to our protocol are Schnorr's protocol [Sch91], which proves knowledge of a discrete logarithm and Okamoto's protocol [Oka93], which proves knowledge of the opening of a Pedersen commitment. The protocols and their knowledge extractors are well-known in the literature, see for example the description in the textbook of Boneh and Shoup [BS20, §19.1, 19.5.1].

Other soundness notions and composition. We often simplify a complex relation by reducing it to a more manageable sub-claim, which is then addressed in a separate proof. Instantiations of these sub-protocols as Σ-protocols are also provided, along with separate security proofs for each component. This approach ensures modularity, as we anticipate that sub-claims will often be integrated into larger proofs conducted within external protocols.

The protocol Π_{aes} relies on a (common) relaxation of special soundness, called 3-special soundness. In a k-special-sound protocol the extractor receives k transcripts, with the same commitment, but with all different challenges.

If we consider sequential composition of a Σ-protocol, the resulting proof is $(2, \dots, 2)$-special sound: it is possible to build a set of accepting transcripts, arranged in a (binary) tree structure, where every branching node at layer i and index j splits into the two transcripts demanded for the i-th layer at the j-th round, for which we provide an extractor. In section Section 5 we relax the above notion slightly, proving $(3, \dots, 3, 2)$-special soundness. We note that we compose at most a constant number of special-sound protocols. Bootle's et al. [BCC+16, Lemma 1] show that a $(n_1, \dots, n_k)$-special sound protocol satisfies witness-extended emulation [Lin03, Def. 10] if $\prod_i^k n_i = \mathsf{poly}(\lambda)$. (In our case $\prod_i n_i < 3^{\log p}$ and the logarithmic factor is involved only when calling the sumcheck protocol.) A similar approach to ours has been taken by Attema and Cramer [AC20]. For honest-verifier zero-knowledge, it is possible to consider the transcripts generated by each HVZK simulator.

If we consider non-interactive proofs, knowledge soundness refers to the existence of a p.p.t. extractor that can extract a witness from a proof using a trapdoor, and zero-knowledge referring to the existence of a simulator that can generate a proof without knowledge of the witness [GOS06; GS08]. In these cases, when embedding the proof within a Σ-protocol, the special-sound extractor will internally run the extractor of the non-interactive proof, and the honest-verifier zero-knowledge its simulator to produce the simulated proof transcript for the sub-proof.

The Fiat–Shamir transformation. As is common in the literature on Σ-protocols and identification schemes, we present and analyze the interactive version of our protocol with the understanding that can be easily made non-interactive using the Fiat–Shamir (FS) transformation [FS87]. In the FS transformation, the prover computes $(a, \rho) \leftarrow \mathsf{Com}(\phi, w)$ as usual, then computes the challenge as $c \leftarrow \mathsf{H}(\phi \| a)$ where H is a cryptographic hash function whose image is in the codomain of Ch. The response is computed as before, and the output is (a, c, z), which can usually be compressed to (c, z) (as in our protocol). The resulting protocol is secure in the random oracle model, via the forking lemma [PS00]. Again, since the FS transform and the related analysis are well-known, we refer to Boneh and Shoup [BS20, Chapter 19] for additional details.

2.1 Σ-reduction protocols

Here we define reductions of knowledge [KP23] for Σ-protocols. A reduction of knowledge reduces proving knowledge of a witness in a relation R to checking knowledge of a witness in a (simpler) relation $\bar{\mathsf{R}}$. For example, in our analysis of Π_{dleq} R is discrete logarithm equality across groups, and $\bar{\mathsf{R}}$ is the range proof on the discrete logarithm.

A Σ-reduction protocol from R to $\bar{\mathsf{R}}$ is a three-move protocol between a prover (with inputs ϕ and w) and a verifier (with input ϕ) consisting of a tuple of efficient algorithms $\Pi := (\mathsf{Com}, \mathsf{Ch}, \mathsf{Resp}, \mathsf{Verify})$ run as follows:

- the prover executes $(a, \rho) \leftarrow \mathsf{Com}(\phi, w)$, sends a and internally stores the state ρ. Com is a randomized algorithm and may have additional inputs such as the group description and security parameter;
- the verifier sends $c \leftarrow \mathsf{Ch}()$ to the prover; c is distributed uniformly at random from a fixed set of possible challenges
- the prover calls $\mathsf{Resp}(\phi, w, \rho, c)$ which may abort (in which case we consider the output to be $\perp$) or return a value z which is sent to the verifier and a reduced witness $\bar{w} \in \bar{\mathsf{R}}$.
- finally, the verifier calls $\mathsf{Verify}(\phi, (a, c, z))$ which returns either false or a reduced instance $\bar{\phi}$ for the reduced relation $\bar{\mathsf{R}}$.

The tuple $(a, c, z, \bar{\phi}, \bar{w})$ is called *extended transcript*; a is called commitment, c is called challenge, and z response. An *accepting transcript* $(a, c, z, \bar{\phi}, \bar{w})$ for ϕ is a transcript for which $\mathsf{Verify}(\phi, (a, c, z))$ does not output false and $(\bar{\phi}, \bar{w})$ is in the relation $\bar{\mathsf{R}}$.

Definition 1. A Σ-reduction protocol Π from R to $\bar{\mathsf{R}}$ has completeness error δ if

$$\Pr\left[\begin{array}{l|l} (z, \bar{w}) := out_p \ \wedge \ \bar{\phi} := out_v \ \wedge & (a, \rho) \leftarrow \mathsf{Com}(\phi, w) \\ (\bar{\phi}, \bar{w}) \in \bar{\mathsf{R}} & c \leftarrow \mathsf{Ch}(); \\ & (z, \bar{w}) \leftarrow \mathsf{Resp}(\phi, w, \rho, c) \\ & out_v \leftarrow \mathsf{Verify}(\phi, (a, c, z)) \end{array}\right] \geqslant 1 - \delta$$

Definition 2. A Σ-reduction Π for R to $\bar{\mathsf{R}}$ is 2-special sound if there exists an extractor Ext such that for any p.p.t. adversary that outputs an instance and two (non-aborting) accepting transcripts $(a, c, z, \bar{\phi}, \bar{w}), (a, c', z', \bar{\phi}', \bar{w}')$ for ϕ such that $c \neq c'$, and $\mathsf{Ext}(\phi, (a, c, z, \bar{\phi}, \bar{w}), (a, c', z', \bar{\phi}', \bar{w}'))$ returns a valid witness $w \in \mathsf{R}(\phi)$ except with probability ϵ. The probability ϵ is called the *knowledge error* of the protocol.

Definition 3. A Σ-reduction Π for R to $\bar{\mathsf{R}}$ is honest-verifier zero-knowledge if there exists an efficient simulator Sim such that for all $(\phi, w) \in \mathsf{R}$ the distributions of non-aborting transcripts:

$$\left\{ (a, c, z, \bar{\phi}) \left| \begin{array}{l} a, \rho \leftarrow \mathsf{Com}(\phi, w); \\ c \leftarrow \mathsf{Ch}(); \\ (z, \bar{w}) \leftarrow \mathsf{Resp}(\phi, w, \rho, c) \\ \bar{\phi} \leftarrow \mathsf{Verify}(\phi, (a, c, z)) \end{array} \right. \right\} \qquad \left\{ (a, c, z, \bar{\phi}) \left| \begin{array}{l} c \leftarrow \mathsf{Ch}(); \\ (a, z, \bar{\phi}) \leftarrow \mathsf{Sim}(\phi, c) \end{array} \right. \right\}$$

is indistinguishable. Our definition is similar to *special HVZK* – *special* since the challenge is input to the simulator, as opposed to being chosen by the simulator. If the two distributions are perfectly indistinguishable (which can be the case for all our protocols), we will say the protocol enjoys perfect special HVZK since the simulated distribution is identical to the real one.

3 General discrete logarithm equality

In this section we describe Π_{dleq}, our protocol for equality of discrete logarithms. We prove that two Pedersen commitments in different groups commit to the same value, and in [OKMZ24, Appendix A, B] we discuss a variant of Π_{dleq} for simple discrete logarithms.

Table 1: Summary of notation and variables names used throughout Section 3.

p, q	Order of the groups $\mathbb{G}_p$ and $\mathbb{G}_q$
G_p, G_q	Generators of $\mathbb{G}_p$ and $\mathbb{G}_q$
H_p, H_q	Additional generators of $\mathbb{G}_p$ and $\mathbb{G}_q$, independent of G_p, G_q
x, x_p, x_q	The witness as an integer x, or a value mod p or q
b_g	bit-length of the smaller group, i.e., $b_g = \lceil \log_2(\min(p, q)) \rceil$
b_c	bit-length of the challenge c
b_x	bit-length of the witness x
b_f	Parameter controlling the probability of aborts

Notation. Since we will have two groups in our protocol, we use the subscripts p and q to indicate that an element or scalar belongs to $\mathbb{G}_p$ or $\mathbb{G}_q$. That is, we denote by $(\mathbb{G}_p, p, G_p, H_p)$ the description of a group $\mathbb{G}_p$ of prime order p. We will often lift scalars from $\mathbb{Z}_p$ to $\mathbb{Z}$ in the canonical way, and when we say that values $x_p \in \mathbb{Z}_p$ and $x_q \in \mathbb{Z}_q$ are equal we mean they are the same as integers. In Table 1 we summarize the variable names and notation used in this work.

We prove the following theorem.

Theorem 1. *Let $\mathbb{G}_p$ and $\mathbb{G}_q$ be additive groups of prime order p, q where discrete logarithm is hard. Let $b_x, b_c, b_f \in \mathbb{N}$ such that $b_x + b_c + b_f < \lceil \log_2(\min(p, q)) \rceil$. Then Π_{dleq} of Figure 1 is a Σ-reduction from the relation*

$$\mathsf{R}_{dleq} := \left\{ \begin{array}{c} ((X_p, X_q)\,,\ (x, r_p, r_q)) \in (\mathbb{G}_p \times \mathbb{G}_q) \times (\{0, \ldots, 2^{b_x} - 1\} \times \mathbb{Z}_p \times \mathbb{Z}_q) \colon \\ X_p = xG_p + r_p H_p \ \wedge \ X_q = xG_q + r_q H_q \end{array} \right\} \tag{1}$$

to the relation

$$\mathsf{R}_{rp} := \left\{ (X_p, (x, r)) \in \mathbb{G}_p \times \mathbb{Z}_p^2 \colon X_p = xG_p + rH_p \ \wedge \ 0 \leqslant x < 2^{b_x} \right\} \ ,$$

with:

- *completeness error 2^{-b_f},*
- *special soundness error 2^{-b_c},*
- *perfect honest-verifier zero-knowledge,*
- *proof size $b_c + b_f + \lceil \log q \rceil + \lceil \log p \rceil$.*

Our protocol has a similar structure to Okamoto's identification protocol [Oka93] and Chaum–Pedersen's representation proof [CP93]. The main differences are: the response value is computed over the integers (so that a single value is used in both groups during verification) and a range proof π_{rp}, parametrized by the group description $(\mathbb{G}_p, p, G_p, H_p)$ and the bound b_x, for the relation R_{rp}. The range proofs ensures that the discrete log "fits" in both groups. Our analysis will require that the range proof be knowledge-sound, since in our analysis we need to extract the opening of the Pedersen commitment X_p from both π_{rp} and from our new protocol, to ensure that both proofs are about the same opening of X_p (which holds since Pedersen commitments are binding). In practice, π_{rp} can be realized in constant size with Sharp [CGKR22] when G_p is a prime-order group (we discuss some options in [OKMZ24, Appendix B]).

We study the protocol as an interactive Σ-protocol (with aborts) with the understanding that it can be directly made non-interactive with the Fiat–Shamir transformation (cf. Section 2). In the Fiat–Shamir with aborts paradigm, provers in this class will abort the protocol with a bounded probability: intuitively, the prover will abort when providing a response would leak information about the witness. When this occurs, the prover and verifier restart the protocol from the beginning. In the non-interactive version, the prover repeats locally, and only outputs a non-aborting transcript.

Prover$((X_p, X_q), (x, r_p, r_q))$ **Verifier**(X_p, X_q)

$k \leftarrow\!\!\$ \{0, \ldots, 2^{b_x+b_c+b_f}-1\}$

$t_p \leftarrow\!\!\$ \{0, \ldots, p-1\}$

$t_q \leftarrow\!\!\$ \{0, \ldots, q-1\}$

$K_p := kG_p + t_pH_p$

$K_q := kG_q + t_qH_q$

$\xrightarrow{K_p, K_q}$

$c \leftarrow\!\!\$ \{0, \ldots, 2^{b_c}-1\}$

$\xleftarrow{c}$

if $c \notin \{0, \ldots, 2^{b_c}-1\}$
then abort

$z := k + cx$ (in $\mathbb{Z}$)

$s_p = t_p + cr_p \pmod p$

$s_q = t_q + cr_q \pmod q$

if $z \notin \{2^{b_x+b_c}, \ldots, 2^{b_x+b_c+b_f}-1\}$
then abort

$\xrightarrow{z, s_p, s_q}$

Ensure:

(i) $zG_p + s_pH_p \stackrel{?}{=} K_p + cX_p$

(ii) $zG_q + s_qH_q \stackrel{?}{=} K_q + cX_q$

(iii) $z \notin \{2^{b_x+b_c}, \ldots, 2^{b_x+b_c+b_f}-1\}$

Prover and Verifier:

$\Pi_{rp} : ((x, r_p), X_p) \in \mathsf{R}_{rp}$

Output: π_{rp} # show $0 \leqslant x_p < 2^{b_x}$

$\longleftrightarrow$

Figure 1: Protocol Π_{dleq} for equality of committed values across groups. The input commitments are $X_p = xG_p + r_pH_p \in \mathbb{G}_p$ and $X_q = xG_q + r_qH_q \in \mathbb{G}_q$ for $0 \leqslant x < 2^{b_x}$, and $r_p, r_q \in \mathbb{Z}_q$.

Parameter selection. We must choose parameters so that $b_x + b_c + b_f < b_g$ so that the response is an integer and no modular reduction occurs in either group. We must also choose the number of parallel repetitions τ so that $\tau \cdot b_c \approx 128$, for non-interactive security. In Table 2 we give some possible parameters for a popular selection of groups, Ristretto[9] (which is not pairing-friendly) and the BLS12-381 group [BLS03; Bow17] (which is pairing-friendly).

3.1 Proof of Theorem 1

We show that Π_{dleq} satisfies statistical completeness with a small error (Lemma 2), special soundness (Theorem 2), and honest-verifier zero-knowledge (Theorem 3). To prove them, we first provide a variation of [AFLT12, Lemma 1], which will be useful for arguing completeness and zero-knowledge.

Lemma 1. *In an honest execution of* Π_{dleq} *the probability that the prover aborts is* $1/2^{b_f}$. *If the prover does not abort, the value* z *in the transcript is uniformly distributed in* $\{2^{b_x+b_c}, \ldots, 2^{b_x+b_c+b_f}-1\}$.

[9] https://ristretto.group

Table 2: Possible parameter choices for 128-bit security when $\mathbb{G}_p$ is Ristretto and $\mathbb{G}_q$ is BLS12-381. Column τ is the number of repetitions; $|\pi|$ is the proof size in bytes excluding the size of the range proof; all other columns are in bits.

b_c	b_x	b_f	τ	$\|\pi\|$	Notes
192	52	8	1	89B	
128	112	12	1	81B	Ideal for the credential linking application
64	128	60	2	158B	Increase b_f since $\tau = 2$ means we can reduce b_c
64	180	8	2	145B	
32	212	8	4	274B	See alternative approach for large x in [OKMZ24, Appendix B].
16	228	8	8	532B	See alternative approach for large x in [OKMZ24, Appendix B].

Proof. In the response value $z = k + cx_p$, since k and c are independent and k is distributed uniformly at random, the value z is distributed uniformly at random in the set

$$Z_0 = \{cx, cx + 1, \ldots, cx + 2^{b_x+b_c+b_f} - 1\} .$$

Let $Z = \{2^{b_x+b_c}, \ldots, 2^{b_x+b_c+b_f} - 1\}$ be the set of responses for which the prover does not abort, and note that Z is properly contained in Z_0. The probability that $z \in Z$ is

$$|Z|/|Z_0| = \frac{2^{b_x+b_c+b_f} - 2^{b_x+b_c}}{2^{b_x+b_c+b_f}} = 1 - 1/2^{b_f}$$

and hence the probability that the prover aborts is $1/2^{b_f}$. Consider a fixed response $z_0 \in Z$, we have

$$\Pr[z = z_0 | z \in Z] = \frac{\Pr[z = z_0]}{\Pr[z \in Z]} = \frac{1/2^{b_x+b_c+b_f}}{|Z|/2^{b_x+b_c+b_f}} = \frac{1}{|Z|}$$

and so the response is uniformly distributed in the set of responses that do not cause the prover to abort. □

Given the above lemma, completeness is straightforward.

Lemma 2. Π_{dleq} *has completeness error* 2^{-b_f}.

Proof. By Lemma 1, we have that the prover aborts with probability 2^{-b_f}. When the prover does not abort, the verification equation is always satisfied, since $0 \leqslant c < 2^{b_c}$ and

$$zG_p + s_pH_p = (k + cx)G_p + (t_p + cr_p)H_p = K_p + cX_p .$$

Similarly, one proves that also (ii) is satisfied. □

3.1.1 Soundness

Our soundness analysis reduces to the binding property of Pedersen commitments, and establishes the constraints on the protocol parameters b_x, b_c, and b_f.

Theorem 2. *If* $b_x + b_c + b_f < \lceil \log_2(\min(p, q)) \rceil$, Π_{dleq} *is a 2-special sound* Σ*-reduction from* R_{dleq} *to* R_{rp} *with knowledge error* $\epsilon = 2^{-b_c+1} + \epsilon_{dl}$*, where* $\epsilon_{dl} = \max(\epsilon_{dl_p}, \epsilon_{dl_q})$ *is the advantage in solving the discrete logarithm problem in* $\mathbb{G}_p$ *or* $\mathbb{G}_q$.

Proof. We prove 2-special soundness. Let $\mathcal{A}$ be an adversary that outputs two accepting transcripts and reduced witnesses:

$$(K_p, c, z, s_p, s_q), (\bar{x}_p, \bar{r}_p), \quad \text{and} \quad (K_p, c', z', s'_p, s'_q), (\bar{x}'_p, \bar{r}'_p) .$$

Those are given as input to Ext, which internally runs the Okamoto extractor for the proof transcripts (K_p, c, z, s_p, s_q) and $(K_p, c', z', s'_p, s'_q)$. The Okamoto extractor succeeds with probability 2^{-b_c} (since $c \neq c'$) in producing witnesses (x_p, r_p) and (x_q, r_q), such that $X_p = x_pG + r_pH_p$ and $X_q = x_qG + r_qH_q$. Then, the extractor aborts if

$$\begin{aligned} \bar{x}_p &\neq \bar{x}'_p \ \ \text{, or} \\ (x_p, r_p) &\neq (\bar{x}_p, \bar{r}_p) \ \ \text{, or} \\ (z - cx_p, s_p - cr_p) &\neq (z' - c'x_p, s'_p - c'r_p) \ \ \text{, or} \\ (z - cx_q, s_q - cr_q) &\neq (z' - c'x_q, s'_q - c'r_q) \end{aligned} \tag{2}$$

If none of the above holds, the extractor returns (x_p, r_p, r_q).

From special soundness, we have two pairs of accepting transcripts proving knowledge of the opening of a Pedersen commitment in $\mathbb{G}_p$ and $\mathbb{G}_q$, namely $((K_p, c, z, s_p), (K_p, c', z', s'_p))$ and $((K_q, c, z, s_q), (K_q, c', z', s'_q))$. If any of the checks in Equation (2) holds, by the binding property of Pedersen commitments X_p, K_p and K_q, a solution for DL in $\mathbb{G}_p$ or $\mathbb{G}_q$ can be found.

We must argue that $x_p = x_q$, when seen as integers. From the verification checks (i) and (ii) we have that $\exists\ k, k', a, a', b, b' \in \mathbb{Z}$ such that

$$\begin{aligned} z &= k + cx_p + ap & \qquad z &= k' + cx_q + bq \\ z' &= k + c'x_p + a'p & \qquad z' &= k' + c'x_q + b'q \end{aligned}$$

Note that k and k' are well-defined, since the check above establishes a single commitment opening for K_p, K_q in each pair of transcripts. The integers (a, a', b, b') are non-negative because verification checks that $2^{b_x+b_c} \leqslant z < 2^{b_x+b_c+b_f}$ and parameters are chosen such that $b_x + b_c + b_f < \lceil \log_2(\min(p, q)) \rceil$. By subtracting the responses corresponding to the mod p and mod q equations, we have

$$(z - z') = (c - c')x_p + (a - a')p \qquad\qquad (z - z') = (c - c')x_q + (b - b')q,$$

Without loss of generality, assume that $z - z'$ is positive. Since π_{rp} ensures that x_p is “small” and $|c - c'|$ is also “small”, then $(a - a') = 0$. More precisely, $z - z'$ has bit-length less than $b_g \leqslant \lceil \log_2(p) \rceil$ by our choice of parameters (namely the constraint $b_x + b_c + b_f < b_g$), and check (iii) during verification, which ensures that $z < 2^{b_x+b_c+b_f}$.

Equating the two representations of $z - z'$, and noting that $(a - a') = 0$ we have (still over $\mathbb{Z}$)

$$\begin{aligned} (c - c')x_p &= (c - c')x_q + (b - b')q \\ (c - c')(x_p - x_q) &= (b - b')q \end{aligned}$$

Since q is prime, it must divide $(c - c')$ or $(x_p - x_q)$. But since the bit-length of q is at least b_g, and $b_g > b_c$, then q is too large to divide $|c - c'|$. Therefore $q \mid (x_p - x_q)$ which means that $x_p = x_q \pmod q$. Since x_p and x_q are equal mod q, and the bit-length of x_p is strictly less than $\lceil \log_2(q) \rceil$, it must be that $x_p = x_q$ over $\mathbb{Z}$ as well. To conclude, Ext extracts a valid witness with error $\epsilon = 2^{-b_c+1} + \max(\epsilon_{\mathsf{dl}_p}, \epsilon_{\mathsf{dl}_q})$. $\square$

Parallel repetitions. The knowledge error might not be negligible depending on the choice of b_c. For τ repetitions, the reduction for the range proofs needs to be done only once for all repetitions, and the reductions to commitment binding can be done all at once. This means that τ repetitions of Π_{dleq} lead to a knowledge error $2^{(-b_c+1)\tau} + \max(\epsilon_{\mathsf{dl}_p}, \epsilon_{\mathsf{dl}_q})$.

3.1.2 Zero-knowledge

Zero-knowledge with aborts. Schemes where the prover may abort [Lyu09; AFLT12] are generally not honest-verifier zero-knowledge (HVZK). The challenge in proving HVZK is in simulating the prover's commitment message in aborting transcripts. However, it is often possible to prove the schemes satisfies a relaxed notion of HVZK, sometimes called no-abort honest-verifier zero-knowledge (naHVZK) [KLS18]. In naHVZK, the simulator either returns a valid transcript, or returns $\perp$ and the verifier forgets about the incomplete session made only of commitment and challenge. Since naHVZK is sufficient to simulate non-interactive proofs (or signatures) when the Fiat–Shamir transformation is applied, naHVZK is still a useful notion. Our protocol in Figure 1 is not affected by this limitation: intuitively, the responses s_p, s_q, which are distributed uniformly at random in $\mathbb{Z}_p$ and $\mathbb{Z}_q$, guarantee that the commitment message is always uniformly random, both in aborting and succeeding transcripts. Thus, we prove standard honest-verifier zero-knowledge, and our protocol may also be used interactively.

Theorem 3. Π_{dleq} *is perfectly honest-verifier zero-knowledge.*

Proof. Upon receiving as input c, the simulator samples z uniformly at random from $\{2^{b_x+b_c}, \dots, 2^{b_x+b_c+b_f}-1\}$ and s_p and s_q uniformly from $\mathbb{Z}_p$ and $\mathbb{Z}_q$. Then the simulator solves for K_p, as $K_p := (zG_p + s_pH_p) - cX_p$ (similarly for K_q). With probability $1/2^{b_f}$ the simulator outputs $(K_p, K_q, c, \perp)$ (the abort case) and otherwise outputs $(K_p, K_q, c, (z, s_p, s_q))$.

We now argue that the real and simulated transcripts are identically distributed. For the prover's first message, since s_p was chosen uniformly by the simulator, then $K_p = kG_p + t_pH_p = kG_p + (s_p - zc)H_p$ is distributed uniformly at random in $\mathbb{G}_p$, regardless of whether the response is $\perp$ or (z, s_p, s_q). We note that in the abort case k will be distributed differently in real and simulated transcripts, but because K_p and K_q are perfectly hiding commitments they are identically distributed. In non-aborted transcripts, both real and simulated transcripts have uniform z value (in the given range), by Lemma 1 and (s_p, s_q) are sampled uniformly at random in both cases. The abort probability of the simulator is the same as the honest prover, by Lemma 1 honest transcripts are aborted with probability $1/2^{b_f}$ exactly as in the simulated case. □

4 Trading group operations for hash evaluations

Our protocol Π_{dlhash} for trading elliptic curve group operations for hash evaluations is described in Figure 2. It is parametrized by a linear morphism $M \in \mathbb{G}^{m\times n}$ denoting the linear relation to be proven. Valid choices include $M = [G]$ for discrete logarithm relations $xG = X$, or $M = [G, H]$ for Pedersen commitments $[G, H] \cdot [x_0, x_1]^t = x_0G + x_1H$, but at the core it should be hard to find $\mathbf{x}, \mathbf{x}'$ such that $M\mathbf{x} = M\mathbf{x}'$. The verifier's inputs are $(\mathbf{X}, x_h)$, respectively commitment and hash of the same value $\mathbf{x}$. We require that the matri M has a kernel hard to solve, that is:

Definition 4 (Kernel-Matrix Diffie-Hellman [MRV16])**.** KMDH is hard for a group generator GrGen and a matrix distribution D if it is infeasible, given a group description $\Gamma := (\mathbb{G}, p, G) \leftarrow \mathsf{GrGen}(1^\lambda)$ and a matrix $M \leftarrow \mathsf{D}(\Gamma)$ in $\mathbb{G}^{n\times m}$ to find non-trivial elements of the null space, that is, to exhibit an $\mathbf{x} \in \mathbb{Z}_p^n$ such that $M\mathbf{x} = 0$ and $\mathbf{x} \neq 0$.

The simplest examples of the above is the group mapping $M = [G]$, (where G is the group generator) which is into and thus perfectly collision resistant. Another valid example are Pedersen commitments, for which $M = [G, H]$ ($m = 1$, $n = 2$, and $H \leftarrow\$\ \mathbb{G}$) and the binding property follows straightforwardly from hardness of DL in $\mathbb{G}$. We will say that KMDH is hard in $\mathbb{G}$ for a matrix M if we consider the distribution D to be the distribution of matrices M parametrized solely by the group description output of GrGen.

Prover$(\mathbf{X}, x_h, (\mathbf{x}))$		**Verifier**$(\mathbf{X}, x_h)$
$\mathbf{k} \leftarrow\!\!\$\ \mathbb{Z}_p^m$		
$k_h := \mathsf{H}(\mathbf{k})$		
$\mathbf{K} := M\mathbf{k}$	$\xrightarrow{\quad k_h, \mathbf{K} \quad}$	$c \leftarrow\!\!\$\ \mathbb{Z}_p$
	$\xleftarrow{\quad c \quad}$	
$\mathbf{z} := \mathbf{k} + c\mathbf{x}$	$\xrightarrow{\quad \mathbf{z} \quad}$	
		$M\mathbf{z} \stackrel{?}{=} \mathbf{K} + c\mathbf{X}$

Prover and Verifier:

$\Pi_{crh} : ((\mathbf{x}, \mathbf{k}), x_h, k_h, c, \mathbf{z}) \in \mathsf{R}_{crh}$

Output: π_{crh} # show $x_h = \mathsf{H}(\mathbf{x})$,

#$k_h = \mathsf{H}(\mathbf{k})$, and $\mathbf{z} = \mathbf{k} + c\mathbf{x}$

$\longleftrightarrow$

Figure 2: Protocol Π_{dlhash}, a Σ-protocol for proving knowledge of $\mathbf{x}$ such that $\mathbf{X} = M\mathbf{x}$ and $x_h = \mathsf{H}(\mathbf{x})$.

In addition, in order to provide HVZK, the hash function H must be compatible with the group GrGen: it must be computationally hard to distinguish the pair $(M\mathbf{x}, \mathsf{H}(\mathbf{x}))$ from the pair $(M\mathbf{x}', \mathsf{H}(\mathbf{x}'))$ for $\mathbf{x} \neq \mathbf{x}'$. We call this notion *hiding-compatibility.*

Definition 5. Let $f = \{f_\lambda\}_\lambda, h = \{h_\lambda\}_\lambda$ be two function families indexed in $\lambda \in \mathbb{N}$ with domain $\mathcal{X}_\lambda$. (f, h) are *hiding-compatible* with error ϵ_{hc} if the distributions

$$\{x \leftarrow\!\!\$\ \mathcal{X}_\lambda : (f_\lambda(x), h_\lambda(x))\} \qquad \text{and} \qquad \{x, s \leftarrow\!\!\$\ \mathcal{X}_\lambda : (f_\lambda(x), h_\lambda(s))\}$$

have statistical distance ϵ_{hc}.

We prove the following theorem:

Theorem 4. *Let $m, n \in \mathsf{poly}(\lambda)$. Let $M \in \mathbb{G}^{m \times n}$ be a matrix over some group $\mathbb{G}$, and H be a collision-resistant hash function. Then, Π_{dlhash} of Figure 2 is a Σ-reduction from the relation*

$$\mathsf{R}_{dlhash} := \{((x_h, \mathbf{X}), \mathbf{x})) : \mathbf{X} = M\mathbf{x} \ \wedge \ x_h = \mathsf{H}(\mathbf{x})\} \ , \tag{3}$$

to the relation

$$\mathsf{R}_{crh} := \{((\mathbf{x}, \mathbf{k}), x_h, k_h, c, \mathbf{z}) : x_h = \mathsf{H}(\mathbf{x}) \ \wedge \ k_h = \mathsf{H}(\mathbf{k}) \ \wedge \ \mathbf{z} = \mathbf{k} + c\mathbf{x}\} \ .$$

with:

- *perfect completeness,*
- *knowledge soundness error ϵ_{kmdh},*
- *honest-verifier zero-knowledge with error ϵ_{hc},*
- *proof size $|\mathsf{H}| + m \cdot |\mathbb{G}| + n \cdot |\mathbb{Z}_p|$.*

Π_{dlhash} requires at the end a proof for the pre-image of a collision-resistant hash function H, parametrized by the field $\mathbb{Z}_p$, that is, a proof for the relation R_{crh}.

This protocol can be instantiated (for instance) using our Π_{aes} from Section 5, taking particular care in tweaking the block size to be large enough in order to provide sufficient collision resistance,[10] but this proof can of course be provided with any other general-purpose proof system for collision-resistant hash functions, algebraic or boolean.

4.1 Proof of Theorem 4

Completeness is straightforward: the response $\mathbf{z} = \mathbf{k} + c\mathbf{x}$ satisfies $M\mathbf{z} = M\mathbf{k} + c(M\mathbf{x}) = \mathbf{K} + c\mathbf{X}$ by linearity of M, and validity of the proof π_{crh} relies on completeness of the underlying proof for the relation R_{crh}. In the remainder of this section, we focus on proving soundness (Lemma 3) and zero-knowledge (Lemma 4) We then discuss the concrete efficiency of our protocol when instantiated for simple discrete logarithm relations.

4.1.1 Soundness

Our soundness analysis assumes that M is "binding" (i.e., KMDH is hard for M).

Lemma 3. *Π_{dlhash} a 2-special sound Σ-reduction from R_{dlhash} to R_{crh} with knowledge error ϵ_{kmdh}.*

Proof. We prove 2-special soundness, extracting $\mathbf{x}$ such that $\mathbf{X} = M\mathbf{x}$ and $h_x = \mathsf{H}(\mathbf{x})$. Consider a p.p.t. adversary that outputs two accepting transcripts and reduced witnesses:

$$(\mathbf{K}, k_h, c_0, \mathbf{z}_0), (\overline{\mathbf{x}}_0, \overline{\mathbf{k}}_0) \quad \text{and} \quad (\mathbf{K}, k_h, c_1, \mathbf{z}_1), (\overline{\mathbf{x}}_1, \overline{\mathbf{k}}_1) \ , \tag{4}$$

with $c_0 \neq c_1$. We claim that $\overline{\mathbf{x}}_0$ is the witness, and now argue that it is indeed valid. Since both transcripts are accepting, the extracted $\overline{\mathbf{x}}_0, \overline{\mathbf{x}}_1, \overline{\mathbf{k}}_0, \overline{\mathbf{k}}_1$ satisfy

$$\begin{aligned} \mathsf{H}(\overline{\mathbf{x}}_0) &= \mathsf{H}(\overline{\mathbf{x}}_1) = x_h \ , & \overline{\mathbf{k}}_0 &= \mathbf{z}_0 - c_0\overline{\mathbf{x}}_1 \pmod p \ , \\ \mathsf{H}(\overline{\mathbf{k}}_0) &= \mathsf{H}(\overline{\mathbf{k}}_1) = k_h \ , & \overline{\mathbf{k}}_1 &= \mathbf{z}_1 - c_1\overline{\mathbf{x}}_1 \pmod p \ . \end{aligned} \tag{5}$$

If $\overline{\mathbf{k}}_0 \neq \overline{\mathbf{k}}_1$ or $\overline{\mathbf{x}}_0 \neq \overline{\mathbf{x}}_1$, we found a break for collision-resistance of H. Since the two transcripts of Equation (4) are valid, define $\mathbf{x} := (c_0 - c_1)^{-1}(\mathbf{z}_0 - \mathbf{z}_1)$ satisfying

$$\mathbf{K} = M\mathbf{z}_0 - c_0 M\mathbf{x} = M\mathbf{z}_1 - c_1 M\mathbf{x} \ .$$

(Note $c_0 \neq c_1$, so the inverse always exists.) Since $\overline{\mathbf{k}}_0 = \overline{\mathbf{k}}_1$ and $\overline{\mathbf{x}}_0 = \overline{\mathbf{x}}_1$, we have

$$\begin{aligned} \mathbf{z}_0 - c_0\overline{\mathbf{x}}_0 &= \mathbf{z}_1 - c_1\overline{\mathbf{x}}_0 \pmod p \\ M\mathbf{z}_0 - c_0 M\mathbf{x} &= M\mathbf{z}_1 - c_1 M\mathbf{x} \end{aligned}$$

If $\overline{\mathbf{x}}_0 \neq \mathbf{x}$, we have a non-trivial element of the kernel of M since $(c_0(\mathbf{x} - \overline{\mathbf{x}}_0), c_1(\mathbf{x} - \overline{\mathbf{x}}_0))$ are different ($c_0 \neq c_1$) and have the same image under M. Therefore, $x_0 = x = x_1$ and hence $M\overline{\mathbf{x}}_0 = X$. In addition, from Equation (5), $\mathsf{H}(\overline{\mathbf{x}}_0) = x_h$. To conclude, $(\overline{\mathbf{x}}_0, \mathbf{X}, x_h) \in \mathsf{R}_{dlhash}$ with error ϵ_{kmdh}. □

4.1.2 Zero-knowledge

Similarly to the case of soundness, in the statement below we assume that the proof π_{crh} is zero-knowledge. More information about the zero-knowledge property of π_{crh} can be found in Section 2.

Lemma 4. *Π_{dlhash} is honest-verifier zero-knowledge with error ϵ_{hc}.*

[10] A secure hash mode for AES, derivative of the Davies-Meyer construction, has been proposed in https://csrc.nist.rip/groups/ST/toolkit/BCM/documents/proposedmodes/aes-hash/aeshash.pdf.

Table 3: The protocol Π_{dlhash} vs non-native scalar multiplication inside a Groth16 [Gro16] circuit ("Naive"). The hash function used is Poseidon [GKR+21] and the proof size is in bytes after applying the Fiat–Shamir transformation, for two popular choices of elliptic curves. Benchmarks on a laptop equipped with an Intel i7-1370P CPU and 32GB of RAM running Debian Linux.

	$\lvert\pi\rvert$	Prover time	R1CS constraints
Π_{dlhash} (BN254)	256	20ms	325
Naive (BN254)	128	10.1s	1.7 million
Π_{dlhash} (BLS12-381)	336	21ms	325
Naive (BLS12-381)	192	17.6s	2.5 million

Proof. The zero-knowledge simulator samples $c, \mathbf{z}, \mathbf{k}^* \leftarrow\!\!\$\ \mathbb{Z}_p \times \mathbb{Z}_p^n \times \mathbb{Z}_p^n$ and computes $\mathbf{R} := M\mathbf{z} - c\mathbf{X}$, and $r_h := \mathsf{H}(\mathbf{k}^*)$. Then, simulates π_{crh} for the statement $(\tau, (\mathbf{x}, \mathbf{k}))$ and returns the transcript $(\mathbf{R}, r_h, c, \mathbf{z}, \pi_{crh})$ We show that it is difficult for an adversary to distinguish simulated transcripts from genuine transcripts generated by an honest prover via a hybrid argument on the distribution of prover transcripts:

- $\mathbf{H}_1$ An honestly-generated prover transcript is a tuple $(\mathbf{K}, k_h, c, \mathbf{z}, \pi)$ where $\mathbf{K} = M\mathbf{k}$ for some $\mathbf{k}$ uniformly distributed and $k_h := \mathsf{H}(\mathbf{k})$, $\mathbf{z} = c\mathbf{x} + \mathbf{k}$. The proof π is honestly generated for $((\mathbf{x}, \mathbf{k}), (x_h, k_h)) \in \mathsf{R}_{crh}$.
- $\mathbf{H}_2$ This game behaves identically to the previous except that π_{crh} is now computed using the simulator for the statement $(x_h, k_h, c, \mathbf{z})$. The two distributions are indistinguishable by zero-knowledge of π_{crh}.
- $\mathbf{H}_3$ Replace the computation of k_h: instead of honestly computing it via $k_h := \mathsf{H}(\mathbf{k})$, sample $\mathbf{k}^* \leftarrow\!\!\$\ \mathbb{Z}_p^n$ and compute $k_h := \mathsf{H}(\mathbf{k}^*)$. This follows directly from hiding-compatibility of (M, H).
- $\mathbf{H}_4$ Compute the elements $(\mathbf{K}, c, \mathbf{z})$ differently: instead of computing $\mathbf{K} = M\mathbf{k}$ and $\mathbf{z} := \mathbf{k} + c\mathbf{x}$ for some uniformly distributed $c \in \mathbb{Z}_p$ and $\mathbf{k}$, we sample $c, \mathbf{z} \leftarrow\!\!\$\ \mathbb{Z}_p \times \mathbb{Z}_p^n$ and compute $\mathbf{K} = M\mathbf{z} + c\mathbf{x}$. The two distributions are both uniformly distributed satisfying the relation $\mathbf{K} = M\mathbf{z} + c\mathbf{x}$ and perfectly indistinguishable. (The adversary cannot see the order in which values are sampled).

The simulated transcript is exactly the distribution output of the simulator. Therefore, the protocol Π_{dlhash} is honest-verifier zero-knowledge. □

4.2 Efficiency

Roughly speaking, Π_{dlhash} saves $O(\lambda \log p)$ constraints from the zero-knowledge SNARK circuit and trading them off with twice a hash circuit evaluation. Let $|\mathsf{H}|$ denote the size of the output of the hash function H and $|\pi_{crh}|$ the size of the proof π_{crh}. For a linear relation $M \in \mathbb{G}^{n\times m}$, the prover time and proof size of Π_{dlhash} the proof size will be $|\pi| = (n+1)|\mathbb{F}| + |\mathsf{H}| + |\pi_{crh}|$ after applying the Fiat–Shamir transformation and the prover time will be dominated by m multi-scalar multiplications of size n, plus the cost for computing the sub-proof π_{crh}.

We benchmark our proof system for simple DL relations, using $M = [G]$, Poseidon [GKR+21] as the CRH function H, and π_{crh} using R1CS and Groth16 [Gro16].[11] In Table 3 we benchmark the performance of Π_{dlhash} against non-native scalar multiplication xG using the double-and-add algorithm inside a SNARK circuit, for an $x \in \mathbb{Z}_p$ of 255 bits. In this case, Π_{dlhash} reduces the number of constraints of about 4 orders of magnitude. We expect more recent proofs based on custom gates such as Plonk to have a lower (but still signigicant) gap. In Table 3 we benchmark the performance of Π_{dlhash} against the naive

[11] https://github.com/arnaucube/sigmabus-poc

approach of performing a non-native scalar multiplication xG using the double-and-add algorithm inside a SNARK circuit, for an $x \in \mathbb{Z}_p$ of 255 bits.

5 Rijndael via one lookup

The Rijndael cryptosystem [DR91] is a symmetric encryption algorithm, established as Advanced Encryption Standard (AES) by NIST in 2001 [AES01], which has become a widely-used standard for securing electronic data globally. We prove the following theorem:

Theorem 5. *Π_{aes} of Figure 3 is a Σ-reduction from the relation*

$$\mathsf{R}_{aes} := \left\{ \begin{aligned} ((ctx, M, K), (\mathbf{m}, \mu, \mathbf{k}, \kappa)) : \quad & ctx = \mathit{AES}(\mathbf{k}, \mathbf{m}) \\ & \wedge \; M = \sum_i m_i G_i + \mu H_i \\ & \wedge \; K = \sum_i k_i G_i + \kappa H_i \end{aligned} \right\}, \tag{6}$$

to the relation

$$\mathsf{R}_{lup} := \left\{ ((\mathbf{f}, \phi), F, \mathbf{t}) : \; F = \textstyle\sum_{i=1}^{n} f_i G_i + \phi H \;\; \wedge \;\; \forall i \in [1, n], f_i \in \{t_j\}_{j=1}^{|\mathbf{t}|} \right\} .$$

where AES is the Rijndael block cipher, and $\mathbf{m}, \mathbf{k}$ are the bit-strings of (respectively) message and key. The proof system enjoys:

- *perfect completeness,*
- *knowledge soundness error $\epsilon_{dl} + 5/p$,*
- *perfect honest-verifier zero-knowledge,*
- *proof size $|\mathbb{G}|$.*

Protocol. The protocol is illustrated in Figure 3. Prover and verifier see the AES encryption as a circuit of three low level functions:

(1) the XOR operation, denoted with infix notation as the map $\oplus \colon \mathbb{F}_{2^8} \times \mathbb{F}_{2^8} \to \mathbb{F}_{2^8} : (a, b) \mapsto a + b$;
(2) multiplication by $\{2\}$ in Rijndael's Galois field $\mathsf{rj2} \colon \mathbb{F}_{2^8} \to \mathbb{F}_{2^8} \colon a \mapsto \alpha \cdot a$, where $\alpha^2 = 1$ is a non-trivial 2-root of unity the field $\mathbb{F}_{2^8}$;
(3) the S-Box operation, denoted $\mathsf{sbox} \colon \mathbb{F}_{2^8} \to \mathbb{F}_{2^8}$ that maps a to its modular inverse a^{-1} if $a \neq 0$ otherwise 0 and combines the result with an affine transformation.

In fact,SubBytes consists solely of one application of sbox, ShiftRows is a permutation, MixColumns is a linear transformation over $\mathbb{F}_{2^8}$ and as such can be written as composition of $\oplus$ and rj2, and AddRoundKey is a simple XOR operation. With a slight abuse of notation, we consider component-wise applications of the above functions: $st' := \mathsf{sbox}(st)$ denotes the application of the S-Box operation to each element of the state $st \in (\mathbb{F}_{2^8})^{16}$.

Denote with $\mathbf{w} := (tr, ksch) := \mathsf{AesTrace}(\mathbf{m}, \mathbf{k})$ the witness vector, containing the computation trace of the AES state across the different rounds, grouped in bit-segments, denoted tr, a vector of small integers in $\{0, \ldots, 2^8 - 1\}$. This algorithm computes intermediate state values $st_{i,j}$ for each round i, and returns their concatenation as the execution trace is detailed in [OKMZ24, Appendix E]. After committing to $\mathbf{w}$, consider the following matrices and equations that hold for a valid cipher trace tr:

- $S_{xor,L}, S_{xor,R}, S_{xor,O}$: matrices of the left inputs, right inputs, and outputs of XOR. We have that

$$S_{xor,L} \cdot tr \;\oplus\; S_{xor,R} \cdot tr = S_{xor,O} \cdot tr \tag{7}$$

  if and only if all $\oplus$ operations over the AES trace are computed correctly.

Prover$((M, K, ctx), (\mathbf{m}, \mu, \mathbf{k}, \kappa))$ **Verifier**(M, K, ctx)

$(tr, ksch) := \mathsf{AesTrace}(\mathbf{m}, \mathbf{k})$

$\omega \leftarrow\!\!\$\ \mathbb{Z}_p$

$W := \sum_{w_i \in tr \| ksch} w_i G_i + \omega H$

$\xrightarrow{\quad W \quad}$

$c \leftarrow\!\!\$\ \mathbb{Z}_p$

$\xleftarrow{\quad c \quad}$

$(c_m, c_k) := (c, c^2)$

$\mathbf{f} := (\mathbf{m} \,\|\, \mathbf{k} \,\|\, tr)$

$(c_{\text{sbox}}, c_{\text{rj2}}, c_{\text{xor}}) := (c^3, c^4, c^5)$

$\phi := c_m \mu + c_k \kappa + \omega$

$\mathbf{t}_{sbox} := [i + c_{\text{sbox}} \cdot \mathsf{sbox}(i)]_i^{255}$

$\mathbf{G}' := S \cdot (c_m \mathbf{G}_{0..|\mathbf{m}|} \,\|\, c_k \mathbf{G}_{0..|\mathbf{k}|} \,\|\, \mathbf{G})$

$\mathbf{t}_{rj2} := [i + c_{\text{rj2}} \cdot \mathsf{rj2}(i)]_i^{255}$

$F := W + c_m M + c_k K$

$\mathbf{t}_{xor} := \left[i + c_{\text{xor}} j + c_{\text{xor}}^2 (i \oplus j)\right]_{i,j}^{127}$

$\mathbf{t} := (\mathbf{t}_{sbox} \| \mathbf{t}_{rj2} \| \mathbf{t}_{xor})$

Prover and Verifier:

$\Pi_{lup} : ((\mathbf{f}, \phi), |f|, \mathbf{t}, F) \in \mathsf{R}_{lup}$

Output: π_{lup} # show $\mathbf{f} \subset \mathbf{t}$

$\longleftrightarrow$

Figure 3: Π_{aes} for proving that ctx is the correct AES-encryption of message $\mathbf{m}$ with key $\mathbf{k}$ committed as M, K; the matrix S is defined in Equation (13).

- $S_{rj2,I}, S_{rj2,O}$: select the inputs and outputs of multiplication by $\{2\}$ in Rijndael's field. We have that
$$\mathsf{rj2}\left(S_{rj2,I} \cdot tr\right) = S_{rj2,O} \cdot tr \tag{8}$$
if and only if all $\mathsf{rj2}$ operations over the AES trace are computed correctly.
- $S_{sbox,I}, S_{sbox,O}$: select the inputs and outputs of the S-Box. We have that
$$\mathsf{sbox}\left(S_{sbox,I} \cdot tr\right) = S_{sbox,O} \cdot tr \tag{9}$$
if and only if all sbox operations over the AES trace are computed correctly.

Instead of checking Equations (7) to (9) directly, the verifier sends challenges $c_{\text{xor}}, c_{\text{rj2}}, c_{\text{sbox}}$ and we check that:

$$S_{xor,L} \cdot tr + c_{\text{xor}} S_{xor,R} \cdot tr + c_{\text{xor}}^2 S_{xor,O} \cdot tr \subset \mathbf{t}_{xor} := [i + c_{\text{xor}} j + c_{\text{xor}}^2 \cdot (i \oplus j)]_{i,j=0}^{255} \tag{10}$$
$$S_{rj2,I} \cdot tr + c_{\text{rj2}} S_{rj2,O} \cdot tr \subset \mathbf{t}_{rj2} := [i + c_{\text{rj2}} \cdot \mathsf{rj2}(i)]_{i=0}^{255} \tag{11}$$
$$S_{sbox,I} \cdot tr + c_{\text{sbox}} S_{sbox,O} \subset \mathbf{t}_{sbox} := [i + c_{\text{sbox}} \cdot \mathsf{sbox}(i)]_{i=0}^{255} \tag{12}$$

where (abusing notation) the subset symbol indicates that all elements in the left-hand side vector appear in the right-hand side vector. In other words, upon receiving a challenge c_{sbox}, the prover computes $x + c_{\text{sbox}} \cdot y$ and proves that it is contained in $i + c_{\text{sbox}} \cdot \mathsf{sbox}(i)$ (for $i \in \{0, \ldots, 2^8 - 1\}$). We proceed similarly for the other operations. Range-checks and shuffles need not be performed, as extraction of a valid witness can be already guaranteed from the lookup protocol itself.

Equations (10) to (12) can be then proven with a generic lookup protocol Π_{lup}, but some improvements can be made on top. First, lookups can be easily batched concatenating the respective instances and proving the concatenation is contained in the concatenation of the respective tables.

Second, the relative Boolean functions $B\colon \mathbb{F}_2^n \to \mathbb{F}_2^n$ where each component is evaluated independently, i.e. $B(\mathbf{v}) = (b(v_0), \ldots, b(v_n))$ for $b\colon \mathbb{F}_2 \to \mathbb{F}_2$ can (naïvely) be seen as a lookup table of size $N := 2^n$. To optimize the concrete efficiency of our protocol, for such functions we instead perform c lookups over tables of size $\sqrt[c]{N}$, for some $c \in \{1, 2, 4, 8\}$. This comes at the cost of committing to a larger vector $\mathbf{w}$ of size $c \cdot N$. For instance, in the case of 8-bit XOR, we consider two lookups of 4-bit segments instead of a single lookup of size 2^8 instead of a single one of size 2^{16}). In our implementation, we selected $c \in 1, 2, 4, 8$ in the case of AES, we have a single table of size $3 \cdot 2^8 = 768$ elements. In our implementation, for AES-128 and AES-256 we represent the key and the message split into 4-bit segments, i.e. $\mathbf{m} = (m_0, \ldots, m_{31})$ with $0 \leqslant m_i < 16$ for all i's. If values are too large then decomposition will fail. Overall, the protocol simply boils down to generating a witness vector of all intermediate computation results and looking up the elements of the computation trace in a table of 768 elements. In AES-128 the computation trace consists of 1232 elements and looks up 1808 elements in a table of 768; in AES-256 the computation trace consists of 1744 elements and 2576 elements to look up in a table of 768.

While the commitment F to $\mathbf{f}$ is part of the statement, it is never sent throughtout the protocol. One may obtain it via a linear transformation of the generators used: consider the matrix S parametrized by the challenges $c_{\mathrm{rj2}}, c_{\mathrm{sbox}}, c_{\mathrm{xor}}$

$$S := \begin{bmatrix} S_{sbox,I} + c_{\mathrm{sbox}}\, S_{sbox,O} \\ S_{rj2,I} + c_{\mathrm{rj2}}\, S_{rj2,O} \\ S_{xor,L} + c_{\mathrm{xor}}\, S_{xor,R} + c_{\mathrm{xor}}^2\, S_{xor,O} \end{bmatrix} \tag{13}$$

and note that $\mathbf{f} = S\mathbf{w}$, and thus $F = \langle \mathbf{w}, S \cdot \mathbf{G} \rangle + \omega H$.

5.1 Proof of Theorem 5

In this section, we prove Theorem 5 showing that it satisfies completeness, special soundness, and honest-verifier zero-knowledge. The analysis is fairly simple as the core of the protocol boils down to one batch lookup invocation.

Completeness is straightforward: the verifier of Π_{aes} internally computes the vector $\mathbf{t} := (\mathbf{t}_{sbox} \| \mathbf{t}_{rj2} \| \mathbf{t}_{xor})$, sees F as a Pedersen commitment under generators $S \cdot \mathbf{G}$, and internally invokes the lookup protocol verifier. Completeness of the whole protocol immediately follows from completeness of the lookup protocol.

Zero-knowledge is guaranteed by the hiding property of the commitment scheme. Specifically, the zero-knowledge simulator just samples a random element $W \leftarrow\!\!\$\ \mathbb{G}$. The result follows by a union bound.

Lemma 5. *The protocol Π_{aes} for the relation $\mathsf{R}_{\mathrm{aes}}$ is 3-special sound with knowledge error $\epsilon_{dl} + 5/p$.*

Proof Sketch. Consider an adversary that outputs valid transcripts and reduced witnesses $(W, c_0), \mathbf{f}_0$, $(W, c_1), \mathbf{f}_1$, and $(W, c_2), \mathbf{f}_2$, with $c_0 \neq c_1$, $c_1 \neq c_2$, and $c_0 \neq c_2$. Let J_1 and J_2 denote the indices of $\mathbf{f}$ encoding the XOR constraints of the initial state and the state at the end of the 0-th round. (Recall that the first round consists solely of $\mathsf{AddRoundKey}$ and the 0-th key is the AES key itself.) Consider the linear system in unknowns $x, y, z \in \mathbb{Z}_p$:

$$\begin{bmatrix} b_0 \\ b_1 \\ b_2 \end{bmatrix} = \begin{bmatrix} 1 & c_0^5 & c_0^{10} \\ 1 & c_1^5 & c_1^{10} \\ 1 & c_2^5 & c_2^{10} \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} \tag{14}$$

where $b_i \in \{f_{i,j}\}_{j \in J_1} \cup \{f_{i,j}\}_{j \in J_2} \cup \{\phi_j\}$ which admits one solution since $c_0 \neq c_1 \neq c_2$, and thus the matrix is Vandermonde. (We assume p much larger than 5.) The extractor checks $x, y, z \in \{0, \ldots, 2^4{-}1\}$ and $z = x \oplus y$. If found, the extractor outputs the recovered values as the witness for the relation $\mathsf{R}_{\mathrm{aes}}$. The extractor outputs $\perp$ if no such values exist.

Table 4: Comparison between zero-knowledge AES-128 schemes and our protocol Π_{aes}. Proof size $|\pi|$ is in bytes. "Proof Type" indicates the techniques used among MPC-in-the-Head [IKOS07], FRI-based [BBHR18], or DL-based, and between parenthesis we indicate if they are plausibly post-quantum *for the zero-knowledge property.* Benchmarks on a laptop equipped with an Intel i7-1370P CPU and 32GB of RAM running Debian Linux.

	Proof type	$\lvert\pi\rvert$	Prover time	Verifier time	PQ (ZK)
PICNIC1-L3 [CDG+17]	MPCitH	74134	3.2ms	2.5ms	✓ (✓)
PICNIC2-L3 [CDG+17]	MPCitH	27173	123ms	41ms	✓ (✓)
FAEST [BBdS+23]	MPCitH	6336	14ms	13ms	✓ (✓)
Preon128A [CCC+23]	FRI	139000	64s	414ms	✓ (✓)
Preon128B [CCC+23]	FRI	372000	65s	576ms	✓ (✓)
Lambdaclass [lam]	DL	855	34s	?	✗ (✓)
Ours (Σ-protocols)	DL	80864	37ms	13ms	✗ (✓)
Ours (compressed-Σ)	DL	2848	180ms	16ms	✗ (✓)

If the extractor outputs $\perp$, it follows that exists (different) $x_j, y_j, z_j \in \{0, \ldots, 2^4-1\}$ such that $x_j \oplus y_j = z_j$ and $x_j + y_j c_j^5 + z_j c_j^{10} = f_{j,0}$. (This is always the case since the proofs are valid.) However, this means that the adversary has found two different openings for the commitment $F = W + c_m M + c_k K$, which is a contradiction to the binding property of Pedersen commitment, which itself happens only with probability ϵ_{dl}. We are left with arguing that the values extracted are indeed from the commitments M and K, which follows from the Schwartz-Zippel lemma. Thus, the knowledge error of the extractor is at most $\epsilon_{dl} + 5/p$. □

5.2 Efficiency

The overall prover's time cost consists of the cost of running the lookup protocol over $|\mathbf{f}|$ needles into a haystack vector $|\mathbf{t}|$ of $2^8 + 2^8 + 2^{16/c}$ elements (in our implementation, 768), plus a multi-scalar multiplication of size $c \cdot n$ for small elements (of size $2^{8/c}$) and one scalar multiplication (for zero-knowledge). When instantiated with Π_{lup} from [OKMZ24, Appendix C] the prover and verifier time complexity are dominated by a linear number of group and field operations.

For the AES-128 cipher with $c = 2$, the prover performs MSMs of 1808 and 3616 $\mathbb{Z}_p$ elements, and the verifier one MSM of 3616 elements. For the AES-256 cipher with $c = 2$, the prover handles MSMs of 2576 and 3488 elements. The prover's larger MSM can be precomputed during an offline phase, reducing the final cost.

We have implemented and made Π_{aes} available as an open-source library in Rust using the arkworks library[12], released under the BSD license.[13] Being based on Σ-protocols, the proof size is linear in the size of the witness. Additionally, we highlight that in the Σ-protocol Π_{lin} ([OKMZ24, Figure 7]), the commitment operation (which is the most expensive of the whole protocol) is independent of the witness and can be precomputed in an offline phase. As part of our implementation, we revisited arkworks' implementation of Pippenger's algorithm and optimized it for small scalars: in order to perform an MSM of the form $\sum_i x_i G_i$, we consider buckets $B_1, \ldots, B_8$ and add G_i to the bucket B_i if the i-th bit of x_i is set. Finally, we return $\sum_i 2^i B_i$. We also employ a batch version of the sumcheck protocol which is described in [OKMZ24, Appendix D]. A summary of our benchmarks is shown in Table 4, comparing Π_{aes} with `curve25519` [Ber06] and alternative approaches. Note that, in the table, the cryptographic assumptions used are different: FRI- and MPCitH-based proofs are for digital signatures and are thus proving equality of

[12] https://arkworks.rs
[13] https://github.com/mmaker/tinybear

secret keys; Lambdaclass's implementation also considers the keyschedule (whereas we do consider only the cipher). Our AES-128 zero-knowledge proof runs in about 30 ms on a MacBook M1 Pro.

6 Acknowledgements

The authors thank Trevor Perrin (Independent) and Melissa Chase (Microsoft Research), who took part to the initial writeup for Π_{dleq}. The authors would also like to express their gratitude towards Andrija Novakovic (Geometry), who was part of the initial writeup for Π_{dlhash}. Stephan Krenn (AIT) provided initial inputs and suggestions. arnaucube (0xPARC) authored the code used for benchmarking Π_{dlhash}.

References

[AC20] Thomas Attema and Ronald Cramer. Compressed Σ-protocol theory and practical application to plug & play secure algorithmics. In Daniele Micciancio and Thomas Ristenpart, editors, *Advances in Cryptology – CRYPTO 2020, Part III*, volume 12172 of *Lecture Notes in Computer Science*, pages 513–543. Springer, Cham, August 2020. DOI: 10.1007/978-3-030-56877-1_18.

[AES01] Advanced Encryption Standard (AES). National Institute of Standards and Technology, NIST FIPS PUB 197, U.S. Department of Commerce, November 2001.

[AFLT12] Michel Abdalla, Pierre-Alain Fouque, Vadim Lyubashevsky, and Mehdi Tibouchi. Tightly-secure signatures from lossy identification schemes. In David Pointcheval and Thomas Johansson, editors, *Advances in Cryptology – EUROCRYPT 2012*, volume 7237 of *Lecture Notes in Computer Science*, pages 572–590. Springer, Berlin, Heidelberg, April 2012. DOI: 10.1007/978-3-642-29011-4_34.

[AGM18] Shashank Agrawal, Chaya Ganesh, and Payman Mohassel. Non-interactive zero-knowledge proofs for composite statements. In Hovav Shacham and Alexandra Boldyreva, editors, *Advances in Cryptology – CRYPTO 2018, Part III*, volume 10993 of *Lecture Notes in Computer Science*, pages 643–673. Springer, Cham, August 2018. DOI: 10.1007/978-3-319-96878-0_22.

[arn] arnaucube. Proof of concept implementation of sigmabus. URL: https://github.com/arnaucube/sigmabus-poc.

[ASM06] Man Ho Au, Willy Susilo, and Yi Mu. Constant-size dynamic k-TAA. In Roberto De Prisco and Moti Yung, editors, *SCN 06: 5th International Conference on Security in Communication Networks*, volume 4116 of *Lecture Notes in Computer Science*, pages 111–125. Springer, Berlin, Heidelberg, September 2006. DOI: 10.1007/11832072_8.

[BBdS+23] Carsten Baum, Lennart Braun, Cyprien Delpech de Saint Guilhem, Michael Klooß, Emmanuela Orsini, Lawrence Roy, and Peter Scholl. Publicly verifiable zero-knowledge and post-quantum signatures from vole-in-the-head. In *CRYPTO 2023*. Springer, 2023. DOI: 10.1007/978-3-031-38554-4_19.

[BBDT16] Amira Barki, Solenn Brunet, Nicolas Desmoulins, and Jacques Traoré. Improved algebraic MACs and practical keyed-verification anonymous credentials. In Roberto Avanzi and Howard M. Heys, editors, *SAC 2016: 23rd Annual International Workshop on Selected Areas in Cryptography*, volume 10532 of *Lecture Notes in Computer Science*, pages 360–380. Springer, Cham, August 2016. DOI: 10.1007/978-3-319-69453-5_20.

[BBHR18] Eli Ben-Sasson, Iddo Bentov, Yinon Horesh, and Michael Riabzev. Fast reed-solomon interactive oracle proofs of proximity. In Ioannis Chatzigiannakis, Christos Kaklamanis, Dániel Marx, and Donald Sannella, editors, *ICALP 2018: 45th International Colloquium on Automata, Languages and Programming*, volume 107 of *Leibniz International Proceedings in Informatics (LIPIcs)*, 14:1–14:17. Schloss Dagstuhl, July 2018. DOI: 10.4230/LIPIcs.ICALP.2018.14.

[BBS04] Dan Boneh, Xavier Boyen, and Hovav Shacham. Short group signatures. In Matthew Franklin, editor, *Advances in Cryptology – CRYPTO 2004*, volume 3152 of *Lecture Notes in Computer Science*, pages 41–55. Springer, Berlin, Heidelberg, August 2004. DOI: 10.1007/978-3-540-28628-8_3.

[BCC+16] Jonathan Bootle, Andrea Cerulli, Pyrros Chaidos, Jens Groth, and Christophe Petit. Efficient zero-knowledge arguments for arithmetic circuits in the discrete log setting. In Marc Fischlin and Jean-Sébastien Coron, editors, *Advances in Cryptology – EUROCRYPT 2016, Part II*, volume 9666 of *Lecture Notes in Computer Science*, pages 327–357. Springer, Berlin, Heidelberg, May 2016. DOI: 10.1007/978-3-662-49896-5_12.

[BCF+21a] Daniel Benarroch, Matteo Campanelli, Dario Fiore, Kobi Gurkan, and Dimitris Kolonelos. Zero-knowledge proofs for set membership: efficient, succinct, modular. In Nikita Borisov and Claudia Diaz, editors, *Financial Cryptography and Data Security*, pages 393–414, Berlin, Heidelberg. Springer Berlin Heidelberg, 2021. ISBN: 978-3-662-64322-8. DOI: 10.1007/S10623-023-01245-1.

[BCF+21b] Daniel Benarroch, Matteo Campanelli, Dario Fiore, Jihye Kim, Jiwon Lee, Hyunok Oh, and Anaïs Querol. Proposal: commit-and-prove zero-knowledge proof systems and extensions. In *4th ZKProof Workshop*, 2021.

[BCG+14] Eli Ben-Sasson, Alessandro Chiesa, Christina Garman, Matthew Green, Ian Miers, Eran Tromer, and Madars Virza. Zerocash: decentralized anonymous payments from bitcoin. In *2014 IEEE Symposium on Security and Privacy*, pages 459–474. IEEE Computer Society Press, May 2014. DOI: 10.1109/SP.2014.36.

[BCG+22] Kenneth A Bamberger, Ran Canetti, Shafi Goldwasser, Rebecca Wexler, and Evan J Zimmerman. Verification dilemmas in law and the promise of zero-knowledge proofs. *Berkeley Tech. LJ*, 37:1, 2022.

[BCL+21] Benedikt Bünz, Alessandro Chiesa, William Lin, Pratyush Mishra, and Nicholas Spooner. Proof-carrying data without succinct arguments. In Tal Malkin and Chris Peikert, editors, *Advances in Cryptology – CRYPTO 2021, Part I*, volume 12825 of *Lecture Notes in Computer Science*, pages 681–710, Virtual Event. Springer, Cham, August 2021. DOI: 10.1007/978-3-030-84242-0_24.

[BCTV14] Eli Ben-Sasson, Alessandro Chiesa, Eran Tromer, and Madars Virza. Scalable zero knowledge via cycles of elliptic curves. In Juan A. Garay and Rosario Gennaro, editors, *Advances in Cryptology – CRYPTO 2014, Part II*, volume 8617 of *Lecture Notes in Computer Science*, pages 276–294. Springer, Berlin, Heidelberg, August 2014. DOI: 10.1007/978-3-662-44381-1_16.

[Ber06] Daniel J. Bernstein. Curve25519: new Diffie-Hellman speed records. In Moti Yung, Yevgeniy Dodis, Aggelos Kiayias, and Tal Malkin, editors, *PKC 2006: 9th International Conference on Theory and Practice of Public Key Cryptography*, volume 3958 of *Lecture Notes in Computer Science*, pages 207–228. Springer, Berlin, Heidelberg, April 2006. DOI: 10.1007/11745853_14.

[BGH19] Sean Bowe, Jack Grigg, and Daira Hopwood. Halo: recursive proof composition without a trusted setup. Cryptology ePrint Archive, Report 2019/1021, 2019. URL: https://eprint.iacr.org/2019/1021.

[BLS03] Paulo S. L. M. Barreto, Ben Lynn, and Michael Scott. Constructing elliptic curves with prescribed embedding degrees. In Stelvio Cimato, Clemente Galdi, and Giuseppe Persiano, editors, *SCN 02: 3rd International Conference on Security in Communication Networks*, volume 2576 of *Lecture Notes in Computer Science*, pages 257–267. Springer, Berlin, Heidelberg, September 2003. DOI: 10.1007/3-540-36413-7_19.

[Bow17] Sean Bowe. BLS12-381: New zk-SNARK elliptic curve construction, 2017. https://electriccoin.co/blog/new-snark-curve/.

[BS20] Dan Boneh and Victor Shoup. A graduate course in applied cryptography, version 0.5, 2020. Available online https://toc.cryptobook.us/book.pdf.

[CCC+23] Ming-Shing Chen, Yu-Shian Chen, Chen-Mou Cheng, Shiuan Fu, Wei-Chih Hong, Jen-Hsuan Hsiang, Sheng-Te Hu, Po-Chun Kuo, Wei-Bin Lee, Feng-Hao Liu, and Justin Thaler. Preon: zk-snark based signature scheme, 2023. URL: https://csrc.nist.gov/csrc/media/Projects/pqc-dig-sig/documents/round-1/spec-files/Preon-spec-web.pdf. https://csrc.nist.gov/csrc/media/Projects/pqc-dig-sig/documents/round-1/spec-files/Preon-spec-web.pdf.

[CDDH19] Jan Camenisch, Manu Drijvers, Petr Dzurenda, and Jan Hajny. Fast keyed-verification anonymous credentials on standard smart cards. In *ICT Systems Security and Privacy Protection*, pages 286–298, 2019. DOI: 10.1007/978-3-030-22312-0_20.

[CDG+17] Melissa Chase, David Derler, Steven Goldfeder, Claudio Orlandi, Sebastian Ramacher, Christian Rechberger, Daniel Slamanig, and Greg Zaverucha. Post-quantum zero-knowledge and signatures from symmetric-key primitives. In Bhavani M. Thuraisingham, David Evans, Tal Malkin, and Dongyan Xu, editors, *ACM CCS 2017: 24th Conference on Computer and Communications Security*, pages 1825–1842. ACM Press, October 2017. DOI: 10.1145/3133956.3133997.

[CFQ19] Matteo Campanelli, Dario Fiore, and Anaïs Querol. LegoSNARK: modular design and composition of succinct zero-knowledge proofs. In Lorenzo Cavallaro, Johannes Kinder, XiaoFeng Wang, and Jonathan Katz, editors, *ACM CCS 2019: 26th Conference on Computer and Communications Security*, pages 2075–2092. ACM Press, November 2019. DOI: 10.1145/3319535.3339820.

[CGKR22] Geoffroy Couteau, Dahmun Goudarzi, Michael Klooß, and Michael Reichle. Sharp: short relaxed range proofs. In Heng Yin, Angelos Stavrou, Cas Cremers, and Elaine Shi, editors, *ACM CCS 2022: 29th Conference on Computer and Communications Security*, pages 609–622. ACM Press, November 2022. DOI: 10.1145/3548606.3560628.

[CGM16] Melissa Chase, Chaya Ganesh, and Payman Mohassel. Efficient zero-knowledge proof of algebraic and non-algebraic statements with applications to privacy preserving credentials. In Matthew Robshaw and Jonathan Katz, editors, *Advances in Cryptology – CRYPTO 2016, Part III*, volume 9816 of *Lecture Notes in Computer Science*, pages 499–530. Springer, Berlin, Heidelberg, August 2016. DOI: 10.1007/978-3-662-53015-3_18.

[CHM+20] Alessandro Chiesa, Yuncong Hu, Mary Maller, Pratyush Mishra, Psi Vesely, and Nicholas P. Ward. Marlin: preprocessing zkSNARKs with universal and updatable SRS. In Anne Canteaut and Yuval Ishai, editors, *Advances in Cryptology – EUROCRYPT 2020, Part I*, volume 12105 of *Lecture Notes in Computer Science*, pages 738–768. Springer, Cham, May 2020. DOI: 10.1007/978-3-030-45721-1_26.

[CL02] Jan Camenisch and Anna Lysyanskaya. Dynamic accumulators and application to efficient revocation of anonymous credentials. In Moti Yung, editor, *Advances in Cryptology – CRYPTO 2002*, volume 2442 of *Lecture Notes in Computer Science*, pages 61–76. Springer, Berlin, Heidelberg, August 2002. DOI: 10.1007/3-540-45708-9_5.

[CL04] Jan Camenisch and Anna Lysyanskaya. Signature schemes and anonymous credentials from bilinear maps. In Matthew Franklin, editor, *Advances in Cryptology – CRYPTO 2004*, volume 3152 of *Lecture Notes in Computer Science*, pages 56–72. Springer, Berlin, Heidelberg, August 2004. DOI: 10.1007/978-3-540-28628-8_4.

[CLOS02] Ran Canetti, Yehuda Lindell, Rafail Ostrovsky, and Amit Sahai. Universally composable two-party and multi-party secure computation. In *34th Annual ACM Symposium on Theory of Computing*, pages 494–503. ACM Press, May 2002. DOI: 10.1145/509907.509980.

[CMZ14] Melissa Chase, Sarah Meiklejohn, and Greg Zaverucha. Algebraic MACs and keyed-verification anonymous credentials. In Gail-Joon Ahn, Moti Yung, and Ninghui Li, editors, *ACM CCS 2014: 21st Conference on Computer and Communications Security*, pages 1205–1216. ACM Press, November 2014. DOI: 10.1145/2660267.2660328.

[CP93] David Chaum and Torben P. Pedersen. Wallet databases with observers. In Ernest F. Brickell, editor, *Advances in Cryptology – CRYPTO'92*, volume 740 of *Lecture Notes in Computer Science*, pages 89–105. Springer, Berlin, Heidelberg, August 1993. DOI: 10.1007/3-540-48071-4_7.

[CPZ20] Melissa Chase, Trevor Perrin, and Greg Zaverucha. The Signal private group system and anonymous credentials supporting efficient verifiable encryption. In Jay Ligatti, Xinming Ou, Jonathan Katz, and Giovanni Vigna, editors, *ACM CCS 2020: 27th Conference on Computer and Communications Security*, pages 1445–1459. ACM Press, November 2020. DOI: 10.1145/3372297.3417887.

[Cra97] Ronald Cramer. *Modular Design of Secure yet Practical Cryptographic Protocols*. PhD thesis, CWI Amsterdam, The Netherlands, 1997.

[CT10] Alessandro Chiesa and Eran Tromer. Proof-carrying data and hearsay arguments from signature cards. In *ICS*, volume 10, pages 310–331, 2010.

[DBB+15] Gaby G. Dagher, Benedikt Bünz, Joseph Bonneau, Jeremy Clark, and Dan Boneh. Provisions: privacy-preserving proofs of solvency for bitcoin exchanges. In Indrajit Ray, Ninghui Li, and Christopher Kruegel, editors, *ACM CCS 2015: 22nd Conference on Computer and Communications Security*, pages 720–731. ACM Press, October 2015. DOI: 10.1145/2810103.2813674.

[DD23] Sai Deng and Bo Du. zkTree: a zk recursion tree with ZKP membership proofs. Cryptology ePrint Archive, Report 2023/208, 2023. URL: https://eprint.iacr.org/2023/208.

[DR91] Joan Daemen and Vincent Rijmen. The design of rijndael: aes. *Journal of Cryptology*, 4(1):3–72, 1991.

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

[FW24] Georg Fuchsbauer and Mathias Wolf. Concurrently secure blind schnorr signatures. In *EUROCRYPT '24 - 43rd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Zurich, Switzerland, May 26-30, 2024, Proceedings, Part II*, pages 124–160, 2024. DOI: 10.1007/978-3-031-58723-8_5. URL: https://doi.org/10.1007/978-3-031-58723-8%5C_5.

[GAZ+22] Paul Grubbs, Arasu Arun, Ye Zhang, Joseph Bonneau, and Michael Walfish. Zero-knowledge middleboxes. In Kevin R. B. Butler and Kurt Thomas, editors, *31st USENIX Security Symposium, USENIX Security 2022*, 2022.

[GKR+21] Lorenzo Grassi, Dmitry Khovratovich, Christian Rechberger, Arnab Roy, and Markus Schofnegger. Poseidon: A new hash function for zero-knowledge proof systems. In Michael Bailey and Rachel Greenstadt, editors, *USENIX Security 2021: 30th USENIX Security Symposium*, pages 519–535. USENIX Association, August 2021.

[GMR89] Shafi Goldwasser, Silvio Micali, and Charles Rackoff. The knowledge complexity of interactive proof systems. *SIAM Journal on Computing*, 18(1):186–208, 1989.

[GOS06] Jens Groth, Rafail Ostrovsky, and Amit Sahai. Perfect non-interactive zero knowledge for NP. In Serge Vaudenay, editor, *Advances in Cryptology – EUROCRYPT 2006*, volume 4004 of *Lecture Notes in Computer Science*, pages 339–358. Springer, Berlin, Heidelberg, May 2006. DOI: 10.1007/11761679_21.

[Gro16] Jens Groth. On the size of pairing-based non-interactive arguments. In Marc Fischlin and Jean-Sébastien Coron, editors, *Advances in Cryptology – EUROCRYPT 2016, Part II*, volume 9666 of *Lecture Notes in Computer Science*, pages 305–326. Springer, Berlin, Heidelberg, May 2016. DOI: 10.1007/978-3-662-49896-5_11.

[GS08] Jens Groth and Amit Sahai. Efficient non-interactive proof systems for bilinear groups. In Nigel P. Smart, editor, *Advances in Cryptology – EUROCRYPT 2008*, volume 4965 of *Lecture Notes in Computer Science*, pages 415–432. Springer, Berlin, Heidelberg, April 2008. DOI: 10.1007/978-3-540-78967-3_24.

[GWC19] Ariel Gabizon, Zachary J. Williamson, and Oana Ciobotaru. PLONK: permutations over Lagrange-bases for oecumenical noninteractive arguments of knowledge. Cryptology ePrint Archive, Report 2019/953, 2019. URL: https://eprint.iacr.org/2019/953.

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

[JBK+24] Sergio Juárez, Mark Blunden, Joris Koopman, Anish Mohammed, Kapil Shenvi Pause, and Steve Thakur. Cross-chain bridges via backwards-compatible SNARKs. Cryptology ePrint Archive, Paper 2024/995, 2024. URL: https://eprint.iacr.org/2024/995.

[KHSS22] Daniel Kang, Tatsunori Hashimoto, Ion Stoica, and Yi Sun. Zk-img: attested images via zero-knowledge proofs to fight disinformation, 2022. arXiv: 2211.04775 [cs.CR].

[Kil90] Joe Kilian. *Uses of Randomness in Algorithms and Protocols*. MIT Press, Cambridge, MA, USA, 1990. ISBN: 0262111535.

[KLS18] Eike Kiltz, Vadim Lyubashevsky, and Christian Schaffner. A concrete treatment of Fiat-Shamir signatures in the quantum random-oracle model. In Jesper Buus Nielsen and Vincent Rijmen, editors, *Advances in Cryptology – EUROCRYPT 2018, Part III*, volume 10822 of *Lecture Notes in Computer Science*, pages 552–586. Springer, Cham, April 2018. DOI: 10.1007/978-3-319-78372-7_18.

[KP23] Abhiram Kothapalli and Bryan Parno. Algebraic reductions of knowledge. In Helena Handschuh and Anna Lysyanskaya, editors, *Advances in Cryptology – CRYPTO 2023, Part IV*, volume 14084 of *Lecture Notes in Computer Science*, pages 669–701. Springer, Cham, August 2023. DOI: 10.1007/978-3-031-38551-3_21.

[KPS18] Ahmed E. Kosba, Charalampos Papamanthou, and Elaine Shi. xJsnark: A framework for efficient verifiable computation. In *2018 IEEE Symposium on Security and Privacy*, pages 944–961. IEEE Computer Society Press, May 2018. DOI: 10.1109/SP.2018.00018.

[lam] Lambdaclass aes encryption circuit. URL: https://github.com/lambdaclass/AES_zero_knowledge_proof_circuit. https://github.com/lambdaclass/AES_zero_knowledge_proof_circuit.

[Lin03] Yehuda Lindell. Parallel coin-tossing and constant-round secure two-party computation. *Journal of Cryptology*, 16(3):143–184, June 2003. DOI: 10.1007/s00145-002-0143-7.

[Lyu08] Vadim Lyubashevsky. Lattice-based identification schemes secure under active attacks. In Ronald Cramer, editor, *PKC 2008: 11th International Workshop on Theory and Practice in Public Key Cryptography*, volume 4939 of *Lecture Notes in Computer Science*, pages 162–179. Springer, Berlin, Heidelberg, March 2008. DOI: 10.1007/978-3-540-78440-1_10.

[Lyu09] Vadim Lyubashevsky. Fiat-Shamir with aborts: applications to lattice and factoring-based signatures. In Mitsuru Matsui, editor, *Advances in Cryptology – ASIACRYPT 2009*, volume 5912 of *Lecture Notes in Computer Science*, pages 598–616. Springer, Berlin, Heidelberg, December 2009. DOI: 10.1007/978-3-642-10366-7_35.

[MGGR13] Ian Miers, Christina Garman, Matthew Green, and Aviel D. Rubin. Zerocoin: anonymous distributed E-cash from Bitcoin. In *2013 IEEE Symposium on Security and Privacy*, pages 397–411. IEEE Computer Society Press, May 2013. DOI: 10.1109/SP.2013.34.

[MRV16] Paz Morillo, Carla Ràfols, and Jorge Luis Villar. The kernel matrix Diffie-Hellman assumption. In Jung Hee Cheon and Tsuyoshi Takagi, editors, *Advances in Cryptology – ASIACRYPT 2016, Part I*, volume 10031 of *Lecture Notes in Computer Science*, pages 729–758. Springer, Berlin, Heidelberg, December 2016. DOI: 10.1007/978-3-662-53887-6_27.

[NBS23] Wilson Nguyen, Dan Boneh, and Srinath Setty. Revisiting the nova proof system on a cycle of curves. Cryptology ePrint Archive, Paper 2023/969, 2023. URL: https://eprint.iacr.org/2023/969. https://eprint.iacr.org/2023/969.

[Oka93] Tatsuaki Okamoto. Provably secure and practical identification schemes and corresponding signature schemes. In Ernest F. Brickell, editor, *Advances in Cryptology – CRYPTO'92*, volume 740 of *Lecture Notes in Computer Science*, pages 31–53. Springer, Berlin, Heidelberg, August 1993. DOI: 10.1007/3-540-48071-4_3.

[OKMZ24] Michele Orrù, George Kadianakis, Mary Maller, and Greg Zaverucha. Beyond the circuit: how to minimize foreign arithmetic in ZKP circuits. Cryptology ePrint Archive, Paper 2024/265, 2024. URL: https://eprint.iacr.org/2024/265.

[Ped92] Torben P. Pedersen. Non-interactive and information-theoretic secure verifiable secret sharing. In Joan Feigenbaum, editor, *Advances in Cryptology – CRYPTO'91*, volume 576 of *Lecture Notes in Computer Science*, pages 129–140. Springer, Berlin, Heidelberg, August 1992. DOI: 10.1007/3-540-46766-1_9.

[PS00] David Pointcheval and Jacques Stern. Security arguments for digital signatures and blind signatures. *Journal of Cryptology*, 13(3):361–396, June 2000. DOI: 10.1007/s001450010003.

[SAB+19] Alberto Sonnino, Mustafa Al-Bassam, Shehar Bano, Sarah Meiklejohn, and George Danezis. Coconut: threshold issuance selective disclosure credentials with applications to distributed ledgers. In *ISOC Network and Distributed System Security Symposium – NDSS 2019*. The Internet Society, February 2019. DOI: 10.14722/ndss.2019.23272.

[Sch91] Claus-Peter Schnorr. Efficient signature generation by smart cards. *Journal of Cryptology*, 4(3):161–174, January 1991. DOI: 10.1007/BF00196725.

[Sma99] Nigel P. Smart. Elliptic curve cryptosystems over small fields of odd characteristic. *Journal of Cryptology*, 12(2):141–151, March 1999. DOI: 10.1007/PL00003820.

[SSS+22] Huachuang Sun, Haifeng Sun, Kevin Singh, Akhil Sai Peddireddy, Harshad Patil, Jianwei Liu, and Weikeng Chen. The inspection model for zero-knowledge proofs and efficient zerocash with secp256k1 keys. Cryptology ePrint Archive, Report 2022/1079, 2022. URL: https://eprint.iacr.org/2022/1079.

[Val08] Paul Valiant. Incrementally verifiable computation or proofs of knowledge imply time/space efficiency. In Ran Canetti, editor, *TCC 2008: 5th Theory of Cryptography Conference*, volume 4948 of *Lecture Notes in Computer Science*, pages 1–18. Springer, Berlin, Heidelberg, March 2008. DOI: 10.1007/978-3-540-78524-8_1.

[XZC+22] Tiancheng Xie, Jiaheng Zhang, Zerui Cheng, Fan Zhang, Yupeng Zhang, Yongzheng Jia, Dan Boneh, and Dawn Song. zkBridge: trustless cross-chain bridges made practical. In Heng Yin, Angelos Stavrou, Cas Cremers, and Elaine Shi, editors, *ACM CCS 2022: 29th Conference on Computer and Communications Security*, pages 3003–3017. ACM Press, November 2022. DOI: 10.1145/3548606.3560652.

[Yas12] Masaya Yasuda. A generalization of the anomalous attack for the ecdlp over qp. *International Journal of Pure and Applied Mathematics*, 77(1):1–9, 2012.

[zca] zcash. The Halo2 Book. URL: https://zcash.github.io/halo2/.

[ZKB] ZK Bug Tracker. URL: https://github.com/0xPARC/zk-bug-tracker/. https://github.com/0xPARC/zk-bug-tracker/.