Proofs of discrete logarithm equality across groups*

Melissa Chase
Microsoft Research
melissac@microsoft.com

Michele Orrù
UC Berkeley
michele.orru@berkeley.edu

Trevor Perrin
Signal Foundation
trevp@signal.org

Greg Zaverucha
Microsoft Research
gregz@microsoft.com

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 past decades, remarkable progress has been made to make zero-knowledge proof schemes practical but unfortunately no zero-zero-knowledge proof system is fit for all uses, and different schemes rely on different algebraic structures with different security guarantees. The problem of bridging proof systems is well-known in the literature, and often boils down to proving equality of two committed values across the same or different groups.

Our contribution. We provide a simple protocol to prove that two secrets, committed across different groups, are equal. We will use $\mathbb{G}_p$ and $\mathbb{G}_q$ to denote groups of prime order p and q, with generators (G_p, H_p) and (G_q, H_q) and constant $b_x > 0$ such that $b_x < \lceil \log_2(\min(p,q)) \rceil - 2$. We prove the following theorem.

Theorem 1 (informal)**.** *If discrete logarithm is hard in $\mathbb{G}_p$ and $\mathbb{G}_q$, and $0 \leq x < 2^{b_x}$, then Π_{dleq} of Figure 1 is a Σ-protocol for relation*

$$\mathsf{R}_{dleq} \coloneqq \{((x, r_p, r_q), X_p, X_q) : X_p = xG_p + r_pH_q \ \wedge \ X_q = xG_q + r_qH_q\}\,.$$

The protocol itself is parametrized by two constants $b_f > 0$ (determining the prover's runtime) and $b_c > 0$ (determining the so-called *knowledge error*) such that $b_x + b_c + b_f < \lceil \log_2(\min(p,q)) \rceil$. The restriction $x < 2^{b_x}$ can be overcome with a small extension (cf. Section 5) and the restriction on the knowledge error can be reduced via generic amplification techniques (repeating the proof τ times). We illustrate some valid choices for the above parameters in Table 2.

Intuitively, we require that x is bounded to simplify the reasoning about the relation above, and work with x over $\mathbb{Z}$. We formalize this precondition by requiring a zero-knowledge proof be provided as input to the protocol. This can be guaranteed via an explicit range proof either in $\mathbb{G}_p$ or $\mathbb{G}_q$, or one of the input commitments may already be known to satisfy the length constraint (e.g., because of an earlier range proof, or because the commitments are authenticated and their opening was previously validated). We list some range proofs in Section 5, but consider the problem of building an efficient range proof outside the scope of this paper.

Our protocol adopts the Fiat-Shamir with abort paradigm of Lyubashevsky [Lyu08, Lyu09]. We also investigated a version without aborts, however we found that the constraints on parameters were limiting and the resulting scheme was less efficient (see the discussion of Girault's scheme in [Lyu09, §1.3] for intuition).

Applications. While the above relation can always be proven with zero-knowledge proofs for arbitrary NP statements, generic approaches must embed foreign group arithmetic in the proof statement and fail to deliver efficient protocols, whereas proofs with our protocol Π_{dleq} require on average 3–6 scalar multiplications

*Preliminary draft, November 16, 2022.

in each of $\mathbb{G}_p$ and $\mathbb{G}_q$ (more details in Table 2) and as little as 118 bytes. We believe this protocol to be suitable for the following applications:

- **Credential Linking.** Our main application is to show that two credentials from different cryptographic groups, are linked. By *credential*, we refer to anonymous credentials defined in groups of prime order such as bilinear CL [CL04], BBS+ [LKWL22, BBS04, ASM06], U-Prove [PZ13] and Brands [Bra94], PS signatures [PS16], Coconut [SAB+19] and keyed-verification credentials such as those used in the Signal private groups system [CMZ14, BBDT16, CDDH19, CPZ20]. For instance, if the first credential contains a user's account ID, e-mail address, phone number, and social security number – the second credential can be linked to the first by including the user ID attribute in both credentials. This is possible both for credentials to be provided by the same issuer, as well as across multiple issuers.

  Linking credentials allows to essentially join authorization attributes via a unique linking attribute, and giving the relying party assurance that both credentials were issued to the same user. In the case of sharing credentials across issuers, the issuers must rely on each other's credential security for the attributes being issued. The second issuer may use a blind issuance protocol to use a unique attribute from the first credential, as a linking attribute in the second credential. Since the cryptographic groups defining the credential systems may be different, the issuers do not need to coordinate.

- **Extending SNARKs.** Expressing arithmetic circuits as Plonk relations [GWC19] or rank-1 constraint systems [BCG+13] is non-trivial. Failure in properly engineering a zero-knowledge circuit might result in either efficiency losses, or void the security guarantees of the proof system itself. As a result, it is difficult to update and maintain the statements being proven, or change the underlying algebraic constructions. By using our protocol to efficiently export data from its home group to the native group of a proof system, proofs become much simpler, and more efficient.

  Linking secrets across groups generically provides not only more flexibility, but also better security: we can decouple the security level of the of the proof system from the security level of the larger protocol. For instance, a larger protocol could work on a large security parameter, say ≈ 500 bit group order, while some predicate proofs could be run with a smaller security parameter, perhaps even using a 160-bit curve elliptic curve group for performance.

- **Proofs for assets.** It is challenging to prove in zero-knowledge complex statements about data on-chain, like to prove that a payment happened or that some assets are held by a public key. On the one hand we have that most cryptocurrencies work over a group without a pairing, and on the other that many efficient proof systems require a pairing. The problem is even harder when attempting to bridge assets across shielded currencies, where values being transmitted are committed across different groups and not publicly available.

  For instance, signatures in Ethereum [But14] or Bitcoin [Nak08] are done over non-pairing-friendly curves, which excludes a large class of efficient proof systems that could be used for proving in zero-knowledge that a payment happened. The protocol Π_{dleq} works on Pedersen commitments which are used (across different elliptic-curve groups) in confidential transactions for Bitcoin [Max15], Monero, and Mimblewimble [FOS19]. Previous attempts at linking assets in these currencies [SSS+22] had to resort to full-fledged proof systems for NP, which entail a larger set of assumptions, a larger engineering burden and higher cost.

Related work. The (simpler) case where $p = q$ has been studied by Chaum and Pedersen [CP93]. The problem of efficiently proving discrete logarithm equality across different groups can be found in Camenisch and Lysyanskaya [CL02], who describe 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. The problem of proving discrete logarithm equality across two generic DL groups was not addressed until Agrawal, Ganesh, and Mohassel [AGM18], who also underline the applications for extending SNARKs. A protocol directly comparable to ours is given

Table 1: Summary of notation and variables names used throughout this work.

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	bitlength of the smaller group, i.e., $b_g = \lceil \log_2(\min(p, q)) \rceil$
b_c	bitlength of the challenge c
b_x	bitlength of the witness x
b_f	Parameter controlling the probability of aborts

in [AGM18], however the protocol is more involved than ours (e.g., it requires commitments to the bits of x in both $\mathbb{G}_p$ and $\mathbb{G}_q$) and thus considerably more expensive.

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 in our analysis.

2 Preliminaries

We denote by $(\mathbb{G}_p, p, G_p, H_p)$ the description of a group $\mathbb{G}_p$ of prime order p, with with two "nothing-up-my-sleeve" generators G_p, H_p (that is, two generators in $\mathbb{G}_p$ such that the discrete logarithm of H_p to the base G_p is not known to anyone). We denote group operations additively, and given a scalar $x \in \mathbb{Z}_p$ we denote with xG_p scalar multiplication. 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 $\mathbb{G}_q$. 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 and notation used in this work. 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.

DL assumption. The discrete logarithm problem asks, given a group description $(\mathbb{G}_p, p, G_p, H_p)$ and an element $X \leftarrow\!\!\$\, \mathbb{G}_p$, to find $x \in \mathbb{Z}_p$ such that $X = xG_p$. The *discrete logarithm (DL) is hard in* $\mathbb{G}_p$ if no p.p.t. algorithm solves the discrete logarithm problem with more than negligible advantage ϵ_{DL_p}.

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

$$C_p := xG_p + rH_p\,.$$

We say that C_p is a Pedersen commitment. A pair $(x, r) \in \mathbb{Z}_p^2$ is a *valid opening* if $C_p = xG_p + rH_p$. 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_p. Computationally binding means that no efficient adversary can produce two different valid openings (x, r) and (x', r') for a commitment C_p. Any adversary that given as input a group description is able to output a

commitment C_p 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 $\log_{G_p} H_p = (r - r')^{-1}(x - x')$.

In Section 5 we will use the well-known fact that Pedersen commitments are *additively homomorphic*: given commitments C_p, C'_p, the sum of the openings $(x_p + x'_p,\ r_p + r'_p)$ is valid for the sum of the commitments $C_p + C'_p$.

Σ-protocols. We briefly recap Σ-protocols. Our definition is a slight variation of the standard Σ-protocol definition from Cramer [Cra97] (as described in Boneh–Shoup [BS20, §19.4]), except we make a few minor changes to model the prover's ability to abort the protocol. Let R be a binary relation between statements denoted by ϕ and witnesses denotes by w. By $\mathsf{R}(\phi)$ we denote the set of possible witnesses for the statement ϕ in R. A Σ-protocol for 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 a statement 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

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

  are identical. Our definition is sometimes referred to as *perfect special HVZK*: perfect since the simulated distribution is identical to the real one, and special since the challenge is input to the simulator, as opposed to being chosen by the simulator.

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].

Non-interactive proofs. As is common in the 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) transform [FS87]. In the FS transform, 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] for additional details.

Range proofs. In Figure 1 require an input π_{RP}, a range proof (as discussed in Section 1). A range proof is parametrized by the group description $(\mathbb{G}_p, p, G_p, H_p)$ and the bound b_x. It proves the relation

$$\mathsf{R}_{rp} \coloneqq \left\{ ((x, r), X_p) : X_p = xG_p + rH_p \ \wedge \ 0 \leq x < 2^{b_x} \right\}.$$

We ask the range proof to satisfy *honest verifier zero-knowledge* and *knowledge soundness*. To simplify the presentation, we assume the range proof is non-interactive and straight-line extractable (also named online extractable in the literature), meaning that there exists an efficient algorithm that can extract the witness from an accepting proof without interacting with the prover.

Since our protocol is using the range proof in a limited way, only on the input commitment, we see no reason why more common notions would not be sufficient: in the interactive case, range proofs that satisfy witness-extended emulation as defined by Lindell [Lin03, Def. 10], and computational knowledge soundness [Gro16, p. 8] in the non-interactive case.

3 Candidate protocol

Our protocol is described in Figure 1, and is parametrized on values $b_x, b_c, b_f > 0$ such that $b_x + b_c + b_f < b_g$ with $b_g \coloneqq \lceil \log_2(\min(p, q)) \rceil$. It has a similar structure to Okamoto's identification protocol [Oka93] and Chaum–Pedersen's representation proof [CP93]. The main differences are: we require a range proof to ensure that the discrete log "fits" in both groups, and the response value is computed over the integers, so that a single value is used in both groups during verification.

The verifier's input to the protocol are two Pedersen commitments $(X_p, X_q) \in \mathbb{G}_p \times \mathbb{G}_q$ committing to values $(x_p, x_q) \in \mathbb{Z}_p \times \mathbb{Z}_q$. The verifier is also given a range proof π_{RP} that the committed value x_p is in $\{0, \ldots, 2^{b_x} - 1\}$. The relation being proven can be expressed as

$$\mathsf{R}_{dleq} \coloneqq \{ ((X_p, X_q), \ (x, r_p, r_q)) : X_p = xG_p + r_pH_p \ \wedge \ X_q = xG_q + r_qH_q \} \tag{1}$$

(where (x, r_p, r_q) is the witness) and holds under the precondition that π_{RP} is valid. 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 for example with Bulletproofs [BBB+18] when G_p is a prime-order group (we discuss some options in Section 5).

We describe the protocol as an interactive Σ-protocol (with aborts) with the understanding that it can be directly made non-interactive with the Fiat-Shamir transform [FS87] (with aborts [Lyu09]). 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.

Parameter selection. In Table 2 we give some possible parameters when $b_g = \min(253, 255) = 253$, where the bitlengths 253 and 255 correspond to the group orders of the Ristretto [HdVLA22] group and the BLS12-381 group [BLS03, Bow17]. 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 $\tau \cdot b_c \geq 128$, for non-interactive security.

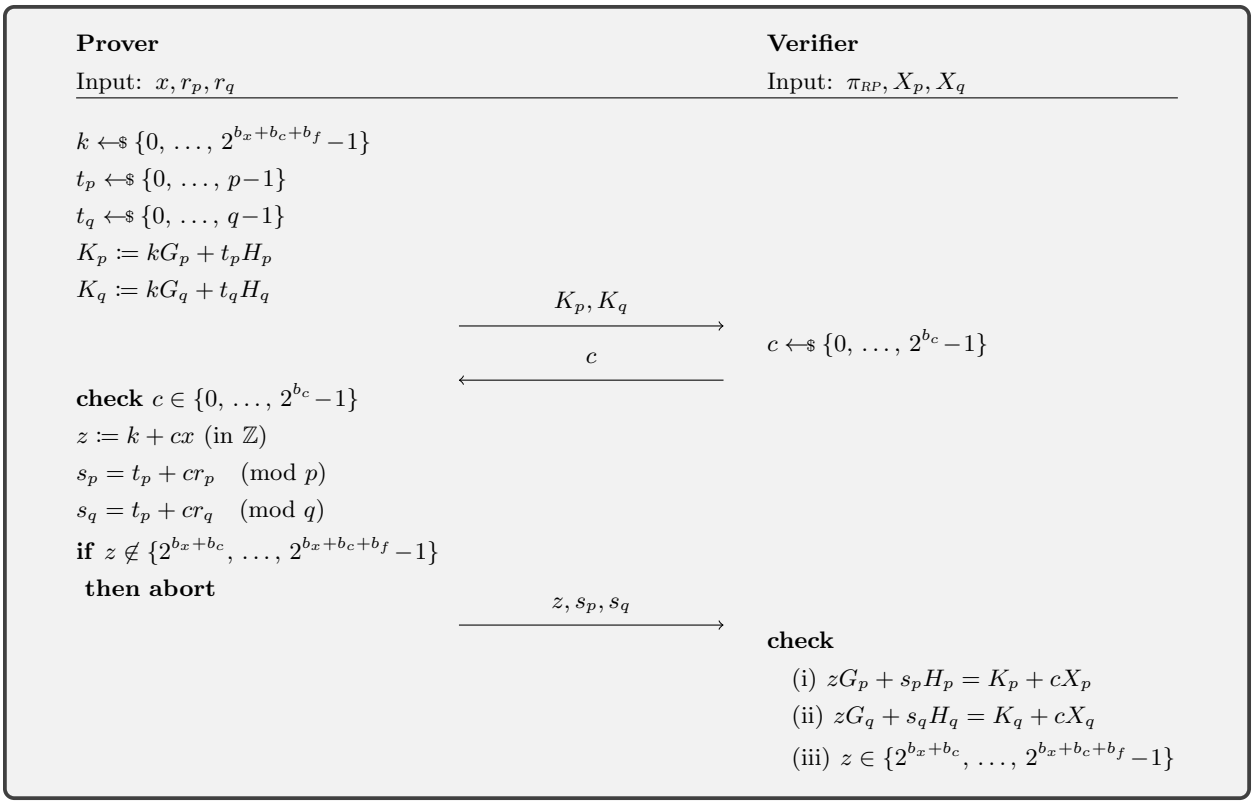

Figure 1: Protocol Π_{dleq}, a Σ-protocol for equality of committed values across groups. The input commitments are $X_p = xG_p + r_pH_p \in \mathbb{G}_p$ and $X_q = xG_q + r_qH_q \in \mathbb{G}_q$ for $x \in \{0, \ldots, 2^{b_x}-1\}$, $r_p \in \mathbb{Z}_p$ and $r_q \in \mathbb{Z}_q$. The verifier's input includes π_{RP}, a range proof that x is in the specified range, which we assume the verifier has checked is valid before the protocol begins.

Performance. For τ repetitions, the size of the proof after applying the Fiat-Shamir transform and compressing the transcript into $\pi = (c, z, s_p, s_q)$ is $\tau(b_x + b_c + b_f + b_c + \lceil \log_2 p \rceil + \lceil \log_2 q \rceil)$ bits. The prover and verifier computational costs are 2τ multi-scalar multiplications (τ in each of $\mathbb{G}_p$ and $\mathbb{G}_q$, each with three terms) when there is no abort (in general the expected cost depends on b_f). Some proof size estimates are given in Table 2.

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 parallel repetitions; $|\pi| = \tau(b_c + b_f + \lceil \log_2 p \rceil + \lceil \log_2 q \rceil)$ is the proof size in bits after applying the Fiat-Shamir transform and excluding the size of the range proof (not required in all applications).

b_c	b_x	b_f	τ	$\lvert\pi\rvert$	Notes
192	52	8	1	951	
128	112	12	1	887	Ideal for the credential linking application
64	128	60	2	1646	Increase b_f since $\tau = 2$ means we can reduce b_c
64	180	8	2	1646	
32	212	8	4	3164	See alternative approach for large x in Section 5
16	228	8	8	6200	

4 Analysis

In this section, we prove our main theorem (Theorem 1) by showing that Π_{dleq} satisfies completeness, special soundness, and honest-verifier zero-knowledge. We introduce a lemma, which is essentially the same as [AFLT12, Lemma 1], propaedeutic for the proofs of completeness and zero-knowledge. The protocols are different, but this lemma applies almost exactly because of the way the response is computed over $\mathbb{Z}$ and the aborting condition.

Lemma 2 ([AFLT12])**.** *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 $\{2^{b_x+b_c}, \ldots, 2^{b_x+b_c+b_f}-1\}$.*

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.

Theorem 3. *The protocol Π_{dleq} for relation R_{dleq} is 2^{-b_f}-complete.*

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

$$zG_p + s_pH_p = (k+cx)G_p + (t_p + cr_p)H_p = (kG_p + t_pH_p) + c\,(xG_P + r_pH_p) = K_p + cX_p\,.$$

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

4.1 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 4. *If $b_x + b_c + b_f < \lceil \log_2(\min(p,q)) \rceil$, the protocol Π_{dleq} is computational special sound for relation R_{dleq} with knowledge error $\epsilon = 2^{-b_c+1} + \epsilon_{RP} + \epsilon_{DL}$, where ϵ_{RP} is the knowledge error of π_{RP} and $\epsilon_{DL} = \epsilon_{DL_p} + \epsilon_{DL_q}$ is the advantage in solving the discrete logarithm problem in $\mathbb{G}_p$ or $\mathbb{G}_q$.*

Proof. We describe an extractor algorithm Ext, that on input π_{RP}, $(X_p, X_q) \in \mathbb{G}_p \times \mathbb{G}_q$, and accepting transcripts $((K_p, K_q), c, z, s)$ and $((K_p, K_q), c', z', s'_p, s'_q)$ with must recover x, r_p, r_q, such that $X_p = xG_p + r_pH_p$ and $X_q = xG_q + r_qH_q$.

Since π_{RP} is assumed to be valid, using the knowledge extractor for that proof we can extract (x^*_p, r^*_p) such that $X_p = x^*_pG_p + r^*_pH_p$ and $x^*_p < 2^{b_x}$. We denote by ϵ_{RP} the probability that the range proof extractor fails in providing a valid witness (x^*_p, r^*_p). From the parts of verification that are done mod p and mod q, we have two pairs of accepting transcripts proving knowledge of the opening of a Pedersen commitment (or,

transcript of Okamoto's identification protocol [Oka93]) in $\mathbb{G}_p$ and $\mathbb{G}_q$, namely $((K_p, c, z, s_p), (K_p, c', z', s'_p))$ and $((K_q, c, z, s_q), (K_q, c', z', s'_q))$. Ext internally runs the Okamoto extractor for both pairs of transcripts, which succeeds with probability 2^{-b_c} (since $c \neq c'$) in producing witnesses (x_p, r_p) and (x_q, r_q), such that $X_p = x_p G + r_p H_p$ and $X_q = x_q G + r_q H_q$.

Then, the extractor checks that all commitment openings are consistent between each other, and that the adversary did not manage to change the committed values in the second transcript. The extractor aborts if $(x_p, r_p) \neq (x_p^*, r_p^*)$ or $(z - cx_p, s_p - cr_p) \neq (z' - c'x_p, s'_p - c'r_p)$. Then, it makes a similar check for $\mathbb{G}_q$ too: if $(z - cx_q, s_q - cr_q) \neq (z' - c'x_q, s'_q - c'r_q)$, abort. This happens with negligible probability, by the binding property of Pedersen commitments X_p, K_p and K_q (that is, hardness of DL in $\mathbb{G}_p$ and $\mathbb{G}_q$).

Finally, the extractor returns (x_p, r_p, r_q). We must now 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

$$z = k + cx_p + ap \qquad z = k' + cx_q + bq$$
$$z' = k + c'x_p + a'p \qquad z' = k' + c'x_q + b'q$$

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 $2^{b_x+b_c} \leq z < 2^{b_x+b_c+b_f}$ 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

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

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 bitlength less than $b_g \leq \lceil \log_2(p) \rceil$ 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 $\mathbb{Z}$):

$$(c - c')x_p = (c - c')x_q + (b - b')q$$
$$(c - c')(x_p - x_q) = (b - b')q$$

Since q is prime, it must divide $(c - c')$ or $(x_p - x_q)$. But since the bitlength 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 bitlength 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 $\epsilon = 2^{-b_c+1} + \epsilon_{RP} + \epsilon_{DL_p} + \epsilon_{DL_q}$. □

Parallel repetitions. The knowledge error might not be negligible depending on the choice of b_c. Generically, τ repetitions result in a knowledge knowledge error ϵ^τ, but in this case the extractor for the range proofs needs to be run 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 $2^{(-b_c+1)\tau} + \epsilon_{RP} + \epsilon_{DL_p} + \epsilon_{DL_q}$.

4.2 Zero-knowledge

Zero-knowledge with aborts. Identification 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 transform 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 $\mathbb{Z}_p$, guarantee that the commitment message is always uniformly random, both in aborting as well as succeeding transcripts. Thus, we prove standard honest-verifier zero-knowledge, and our protocol may also be used interactively.

Theorem 5. *The protocol Π_{dleq} for relation R_{dleq} is perfectly honest-verifier zero-knowledge.*

Proof. On input c, the simulator samples z uniformly at random from $\{2^{b_x+b_c}, \ldots, 2^{b_x+b_c+b_f}-1\}$ 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 2 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 2 honest transcripts are aborted with probability $1/2^{b_f}$ exactly as in the simulated case. □

5 Instantiation and Extensions

In this section we discuss some of the considerations for concretely realizing and implementing Π_{dleq}.

Handling larger values. One limitation of the protocol presented above is that it requires that x be b_x bits or fewer, and b_x cannot be as large as the group order (of the smaller group). Here we describe how to address this, by breaking x into chunks and proving the relation on each chunk using Π_{dleq}.Let C_p and C_q be commitments to the same value $x \in \{0, \ldots, \min(p,q)-1\}$, and suppose b_x has be chosen subject to the constraints given above. Define $\ell := \lceil \log_2 x/b_x \rceil$. Denote by $(x^{(0)}, \ldots, x^{(\ell-1)})$ the representation of x in base 2^{b_x} that is, $x = \sum_{i=0}^{\ell-1} 2^{i \cdot b_x} x^{(i)}$. Sample random $r_p^{(i)}$ such that $r_p = \sum_i 2^{i \cdot b_x} r_p^{(i)} \pmod p$. Construct the commitments $C_p^{(0)}, \ldots, C_p^{(\ell-1)}$ as $C^{(i)} := x^{(i)}G_p + r_p^{(i)}H_p$. Proceed in the same way for C_q. They satisfy

$$C_p = \sum_{i=0}^{\ell-1} 2^{i \cdot b_x} C_p^{(i)} \qquad C_q = \sum_{i=0}^{\ell-1} 2^{i \cdot b_x} C_q^{(i)} \tag{2}$$

The prover sends $C_p^{(0)}, \ldots C_p^{(\ell-1)}$ and $C_q^{(0)}, \ldots, C_q^{(\ell-1)}$, along with ℓ range proofs[1], to prove that each $x^{(i)} \in \{0, \ldots, 2^{b_x}-1\}$. Then the prover and verifier invoke the protocol in Figure 1 for each $i \in \{0, \ldots, \ell-1\}$ to prove that $C_p^{(i)}$ and $C_q^{(i)}$ commit to the same short value. The verifier additionally checks Equation (2) holds.

Range proofs. Range proofs may not be necessary if the application provides assurance that x is in the correct range. For example, in the credential linking application, we can trust that the issuer only issues credentials with a valid x. In systems using keyed-verification anonymous credentials [CMZ14], this is especially reasonable since the issuer and verifier are the same party. When the credential is presented in order to produce X_p, our soundness analysis of Theorem 4 can be modified to extract x from the presentation proof, rather than the range proof.

When a range proof is necessary, Bulletproofs [BCC+16, BBB+18] give a practical solution. For example, creating range proofs for 64-bit values using the Rust crate `bulletproofs` from the Dalek project [dVYA], the prover time is about 7.3 ms, the verifier time is 1 ms and the proof size is 672 bytes. See [dVYA] for details of the benchmark platform, and benchmarks of other libraries offering range proofs. The library does not support 128-bit ranges, but we expect prover and verifier times to roughly double, and the range proof

[1]It is not secure to send a single range proof for x instead of ℓ proofs for each $x_p^{(i)}$. Consider commitments C_p, C_q to different values $x_p < p$ and $x_q < q$, and π_{RP} a range proof of C_p with witness x_p. By the Chinese remainder theorem there exists a unique integer $x < pq$ such that $x_p = x \pmod p$ and $x_q = x \pmod q$. An attacker is able to freely choose ℓ values $x_p^{(0)}, \ldots x^{(\ell-1)}$ larger than b_x such that $\sum_i 2^{ib_x} x_i = x$, and commit to them both in $\mathbb{G}_p$ and $\mathbb{G}_q$, passing the verification procedure.

size to increase to 704 bytes. We also note that when using the strategy given above that breaks x into ℓ pieces, range proofs for each of the pieces can be grouped together into a single proof, which will be shorter than ℓ individual proofs, for example, a proof of four 64-bit ranges is only 800 bytes (but prover and verifier times are only slightly better than four individual proofs).

When one of the groups is a pairing-based group, one could alternatively do π_{RP} in that group using a zkSNARK with constant size and concretely very short proofs, e.g., [Gro16]. Since our analysis requires a range proof for x in *either one of* $\mathbb{G}_p$ or $\mathbb{G}_q$, applications can choose to implement the range proof in the group that offers better performance.

Constant-time implementation. Depending on the abort probability $1/2^{b_f}$, implementations may leak the number of times the protocol was aborted, since the prover's time is directly proportional to the number of aborts (in a direct implementation). If the number of aborts depends on the secret, this would be sensitive information. However, from Lemma 2 we can see that the abort probability is the same for any secret, and therefore independent of the secret. Therefore, it is not required that implementations attempt to hide the number of aborts that occur when generating a proof.

A prominent example signature scheme that uses rejection sampling is Dilithium [LDK+22]. For for the same reason, implementations of that scheme can safely reveal the number of aborts [LDK+22, §5.5] (which is relatively high when compared to our protocol, it is expected to be between 3–6).

Parallel repetition. In Theorem 4 it is shown that Π_{dleq} has knowledge error 2^{-b_c} and because of our constraints on parameter selection, b_c may not be as large as the security parameter λ, so the soundness error may be non-negligible. In practice, assuming the hash function of the FS transform h sas output bit-length $b_c\tau$, the challenges are obtained by considering each of the τ chunks of b_c bits[2]. The well-known approach to boost soundness of the protocol is to repeat it τ times in parallel, so that the soundness error is $2^{-b_c\tau}$ such that $b_c\tau \geq \lambda$. Note that we exclude π_{RP} from the parallel repetitions, since we consider it to be part of the input statement and have negligible soundness error. We also require that none of the τ repetitions abort, which increases the abort probability from $1/2^{b_c}$ to $\tau/2^{b_c}$, so to hold the abort probability constant b_f should be increased by $\lceil \log_2(\tau) \rceil$. Since Π_{dleq} is zero-knowledge, it is also witness hiding [FS90, Theorem 3] and therefore parallel composition is also witness hiding (at least witness hiding; we expect Theorem 5 can be generalized to handle parallel repetition).

Denote the challenge with parallel repetition as $c = (c_1, \ldots, c_\tau)$. Special soundness provides two transcripts with $c \neq c'$, and when both transcripts are different everywhere, that is $c_i \neq c'_i$ for $i \in [\tau]$, we have τ transcripts where Ext succeeds with probability 2^{-b_c}. Therefore soundness is boosted as expected to $2^{-\tau b_c}$ in this case. More generally, when c is chosen at random (either by an honest verifier or a hash function) there may be a small loss in concrete soundness, since some c_i may be equal. While this loss is in our analysis, we do not know of an attack matching it.

When there are multiple instances of the protocol, like in the variant described above where the protocol is run ℓ times, the witnesses are independent and protocols may be run in parallel. As a minor optimization, each of the ℓ instances may share the same random challenge from the verifier.

Ignoring aborts safely? For some secret lengths and group sizes, it is possible to choose parameters such that the abort probability 2^{-b_f} is statistically negligible. In such cases an implementation that ignores aborts will leak a small amount of information occasionally. For example, suppose we have $b_x = 128$, $b_c = 64$, $b_g = 253$ and $b_f = 60$. Then we expect one in 2^{60} proofs to output a "leaky" response; a response that would have caused an abort, but that we output anyway. In the case of Schnorr and ECDSA signatures slight biases in the nonce appear to be difficult to exploit, see e.g., [ANT+20]. However, since our setting is somewhat different and we do not have a detailed analysis we do not recommend ignoring aborts, but encourage future work on this question. Avoiding the abort path simplifies writing and testing of implementations.

[2]In particular, we ask not to use the same hash function for each each repetition to mitigate grinding attacks, also known in the literature as precomputation attacks.

5.1 Equality of simple discrete logarithms

Our protocol takes Pedersen commitments to (x_p, x_q) as input. A natural question is whether a variant of this protocol also works when the inputs are simple discrete logarithm commitments, namely $X_p = x_p G_p$ and $X_q = x_q G_q$. We investigated this question and believe it has a positive answer, subject to some technicalities and limitations. We did not formalize this section as our main motivation of linking credentials requires Pedersen commitments, so that they can be re-randomized by the credential holder before each presentation proof, in order to make repeated proofs with the same credentials unlinkable. We list some of the issues that must be addressed.

Concrete DL hardness. Our protocol allows x to be short (e.g., 64 or 128 bits), but solving for x given $X = xG$ is easier when x is short. For generic groups, the best-known attack cost (Pollard's lambda algorithm [Pol78]) is $2^{(\log_2 x)/2}$. Therefore, x must be large enough so that the DL instance is hard, and this restricts the choices available for parameter selection (cf. Table 2): in order to keep the response size below the group order, we must use smaller challenges, and this increases the number of parallel repetitions required, or the abort probability and consequently the proving time. With Pedersen commitments, instead, x is unconditionally hidden.

Cross-group DL hardness. Again, when the commitments to x are simple discrete log instances, we have to make a new hardness assumption. Namely, we must assume that given short DL instances $X_p = xG_p$ and $X_q = xG_q$ with the same x, the advantage ϵ_{DL} of finding x is as hard as the short DL in either $\mathbb{G}_p$ or $\mathbb{G}_q$. That is, $\epsilon_{DL} \leq \max(\epsilon_{DL_p}, \epsilon_{DL_q})$. This seems reasonable when x is large enough and the DL problem is hard in both $\mathbb{G}_p$ and $\mathbb{G}_q$, but is not a common cryptographic assumption as secrets are almost universally used only in one primitive, and not across groups.

Simuation of aborting transcripts. Another issue is how to (perfectly) simulate the prover's first message in aborted transcripts. In Section 4 we discuss how our protocol avoids this challenge (in short, the first message consists of Pedersen commitments, which are always uniformly random, independent of whether the prover aborts). For a variant of our protocol that does not use Pedersen commitments, the weaker notion of *no-abort HVZK* [KLS18] (discussed in Section 4.2) should be achievable, and while weaker, this notion is still sufficient for non-interactive proofs, which are suitable for the applications we consider.

6 Acknowledgements

The authors thank Stephan Krenn for initial inputs.

References

[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, *EUROCRYPT 2012*, volume 7237 of *LNCS*, pages 572–590. Springer, Heidelberg, April 2012.

[AGM18] Shashank Agrawal, Chaya Ganesh, and Payman Mohassel. Non-interactive zero-knowledge proofs for composite statements. In Hovav Shacham and Alexandra Boldyreva, editors, *CRYPTO 2018, Part III*, volume 10993 of *LNCS*, pages 643–673. Springer, Heidelberg, August 2018.

[ANT+20] Diego F. Aranha, Felipe Rodrigues Novaes, Akira Takahashi, Mehdi Tibouchi, and Yuval Yarom. LadderLeak: Breaking ECDSA with less than one bit of nonce leakage. In Jay Ligatti, Xinming

Ou, Jonathan Katz, and Giovanni Vigna, editors, *ACM CCS 2020*, pages 225–242. ACM Press, November 2020.

[ASM06] Man Ho Au, Willy Susilo, and Yi Mu. Constant-size dynamic k-TAA. In Roberto De Prisco and Moti Yung, editors, *SCN 06*, volume 4116 of *LNCS*, pages 111–125. Springer, Heidelberg, September 2006.

[BBB+18] Benedikt Bünz, Jonathan Bootle, Dan Boneh, Andrew Poelstra, Pieter Wuille, and Greg Maxwell. Bulletproofs: Short proofs for confidential transactions and more. In *2018 IEEE Symposium on Security and Privacy*, pages 315–334. IEEE Computer Society Press, May 2018.

[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*, volume 10532 of *LNCS*, pages 360–380. Springer, Heidelberg, August 2016.

[BBS04] Dan Boneh, Xavier Boyen, and Hovav Shacham. Short group signatures. In Matthew Franklin, editor, *CRYPTO 2004*, volume 3152 of *LNCS*, pages 41–55. Springer, Heidelberg, August 2004.

[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, *EUROCRYPT 2016, Part II*, volume 9666 of *LNCS*, pages 327–357. Springer, Heidelberg, May 2016.

[BCG+13] Eli Ben-Sasson, Alessandro Chiesa, Daniel Genkin, Eran Tromer, and Madars Virza. SNARKs for C: Verifying program executions succinctly and in zero knowledge. In Ran Canetti and Juan A. Garay, editors, *CRYPTO 2013, Part II*, volume 8043 of *LNCS*, pages 90–108. Springer, Heidelberg, August 2013.

[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*, volume 2576 of *LNCS*, pages 257–267. Springer, Heidelberg, September 2003.

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

[Bra94] Stefan Brands. Untraceable off-line cash in wallets with observers (extended abstract). In Douglas R. Stinson, editor, *CRYPTO'93*, volume 773 of *LNCS*, pages 302–318. Springer, Heidelberg, August 1994.

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

[But14] Vitalik Buterin. Ethereum: A Next-Generation Smart Contract and Decentralized Application Platform, 2014. Available at https://ethereum.org/en/whitepaper/.

[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.

[CL02] Jan Camenisch and Anna Lysyanskaya. Dynamic accumulators and application to efficient revocation of anonymous credentials. In Moti Yung, editor, *CRYPTO 2002*, volume 2442 of *LNCS*, pages 61–76. Springer, Heidelberg, August 2002.

[CL04] Jan Camenisch and Anna Lysyanskaya. Signature schemes and anonymous credentials from bilinear maps. In Matthew Franklin, editor, *CRYPTO 2004*, volume 3152 of *LNCS*, pages 56–72. Springer, Heidelberg, August 2004.

[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*, pages 1205–1216. ACM Press, November 2014.

[CP93] David Chaum and Torben P. Pedersen. Wallet databases with observers. In Ernest F. Brickell, editor, *CRYPTO'92*, volume 740 of *LNCS*, pages 89–105. Springer, Heidelberg, August 1993.

[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*, pages 1445–1459. ACM Press, November 2020.

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

[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*, pages 720–731. ACM Press, October 2015.

[dVYA] Henry de Valence, Cathie Yun, , and Oleg Andreev. Rust bulletproofs crate. Accessed October 2022 (HEAD at 6fb4135).

[FOS19] Georg Fuchsbauer, Michele Orrù, and Yannick Seurin. Aggregate cash systems: A cryptographic investigation of Mimblewimble. In Yuval Ishai and Vincent Rijmen, editors, *EUROCRYPT 2019, Part I*, volume 11476 of *LNCS*, pages 657–689. Springer, Heidelberg, May 2019.

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

[FS90] Uriel Feige and Adi Shamir. Witness indistinguishable and witness hiding protocols. In *22nd ACM STOC*, pages 416–426. ACM Press, May 1990.

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

[Gro16] Jens Groth. On the size of pairing-based non-interactive arguments. In Marc Fischlin and Jean-Sébastien Coron, editors, *EUROCRYPT 2016, Part II*, volume 9666 of *LNCS*, pages 305–326. Springer, Heidelberg, May 2016.

[GWC19] Ariel Gabizon, Zachary J. Williamson, and Oana Ciobotaru. Plonk: Permutations over lagrange-bases for oecumenical noninteractive arguments of knowledge. Cryptology ePrint Archive, Paper 2019/953, 2019. https://eprint.iacr.org/2019/953.

[HdVLA22] Mike Hamburg, Henry de Valence, Isis Lovecruft, and Tony Arcieri. The Ristretto group, 2022. https://ristretto.group/.

[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, *EUROCRYPT 2018, Part III*, volume 10822 of *LNCS*, pages 552–586. Springer, Heidelberg, April / May 2018.

[LDK+22] Vadim Lyubashevsky, Léo Ducas, Eike Kiltz, Tancrède Lepoint, Peter Schwabe, Gregor Seiler, Damien Stehlé, and Shi Bai. CRYSTALS-DILITHIUM. Technical report, National Institute of Standards and Technology, 2022. available at https://csrc.nist.gov/Projects/post-quantum-cryptography/selected-algorithms-2022.

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

[LKWL22] T. Looker, V. Kalos, A. Whitehead, and M. Lodder. The BBS signature scheme, October 2022. IRTF CFRG working group draft.

[Lyu08] Vadim Lyubashevsky. Lattice-based identification schemes secure under active attacks. In Ronald Cramer, editor, *PKC 2008*, volume 4939 of *LNCS*, pages 162–179. Springer, Heidelberg, March 2008.

[Lyu09] Vadim Lyubashevsky. Fiat-Shamir with aborts: Applications to lattice and factoring-based signatures. In Mitsuru Matsui, editor, *ASIACRYPT 2009*, volume 5912 of *LNCS*, pages 598–616. Springer, Heidelberg, December 2009.

[Max15] Gregory Maxwell. Confidential Transactions, 2015. Available at https://people.xiph.org/~greg/confidential_values.txt.

[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.

[Nak08] Satoshi Nakamoto. Bitcoin: A Peer-to-Peer Electronic Cash System, 2008. Available at http://bitcoin.org/bitcoin.pdf.

[Oka93] Tatsuaki Okamoto. Provably secure and practical identification schemes and corresponding signature schemes. In Ernest F. Brickell, editor, *CRYPTO'92*, volume 740 of *LNCS*, pages 31–53. Springer, Heidelberg, August 1993.

[Ped92] Torben P. Pedersen. Non-interactive and information-theoretic secure verifiable secret sharing. In Joan Feigenbaum, editor, *CRYPTO'91*, volume 576 of *LNCS*, pages 129–140. Springer, Heidelberg, August 1992.

[Pol78] John M. Pollard. Monte carlo methods for index computation (mod p). *Mathematics of computation*, 32(143):918–924, 1978.

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

[PS16] David Pointcheval and Olivier Sanders. Short randomizable signatures. In *CT-RSA 2016*, pages 111–126, 2016.

[PZ13] C. Paquin and G. Zaverucha. U-prove cryptographic specification v1.1 (revision 2), 2013. Available online: www.microsoft.com/uprove.

[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 *NDSS 2019*. The Internet Society, February 2019.

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

[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, Paper 2022/1079, 2022. https://eprint.iacr.org/2022/1079.