SoK: Zero-Knowledge Range Proofs

Miranda Christ
Columbia University
mchrist@cs.columbia.edu

Foteini Baldimtsi
George Mason University
Mysten Labs
foteini@mystenlabs.com

Konstantinos Kryptos Chalkias
Mysten Labs
kostas@mystenlabs.com

Deepak Maram
Mysten Labs
deepak@mystenlabs.com

Arnab Roy
Mysten Labs
arnab@mystenlabs.com

Joy Wang
Mysten Labs
joy@mystenlabs.com

ABSTRACT

Zero-knowledge range proofs (ZKRPs) allow a prover to convince a verifier that a secret value lies in a given interval. ZKRPs have numerous applications: from anonymous credentials and auctions, to confidential transactions in cryptocurrencies. At the same time, a plethora of ZKRP constructions exist in the literature, each with its own trade-offs. In this work, we systematize the knowledge around ZKRPs. We create a classification of existing constructions based on the underlying building techniques, and we summarize their properties. We provide comparisons between schemes both in terms of properties as well as efficiency levels, and construct a guideline to assist in the selection of an appropriate ZKRP for different application requirements. Finally, we discuss a number of interesting open research problems.

1 INTRODUCTION

Zero-knowledge (ZK) proofs have received much attention in recent years, with an abundance of generic protocols being developed using various assumptions and techniques. Although these generic protocols are becoming very efficient and easier to implement, there are still cases for specific types of statements, where customized ZK protocols are preferable.

Zero-knowledge range proofs (ZKRPs) are a subclass of zero-knowledge proofs that proves a structured kind of set membership. A ZKRP allows a prover to convince a verifier that a secret, committed value lies in a given (integer) interval. Brickell et al. [BCDvdG88] introduced the first type of zero-knowledge range proof as a building block in a protocol for revealing a secret discrete logarithm bit-by-bit. Since their introduction, ZKRPs have been used in various applications such as private e-cash protocols [CFT98] (to verify non-negative transaction amounts), anonymous credentials systems [CCS08, BCC+09, CCL+21] (to prove that a secret credential attribute, i.e. user age, falls in a specific range) as well as private voting [Gro05], auctions [AW13] and privacy preserving federated learning [BGL+23] and so on. Additionally, ZKRPs are often used as building blocks for more complex cryptographic schemes. For instance, they have been used to construct ZK proofs of non-membership [LLNW18] and ZK proofs of certain polynomial relations over the integers [CM99, CBM99], and they have also been used to prove well-formedness of RLWE ciphertexts [DPLS19, Lib23] and well-formedness of shares in secret-sharing schemes [Gro21, GHL22].

At the same time, with the rise of decentralized systems and cryptocurrencies, range proofs have received increased attention due to their use in mechanisms that preserve the privacy of transactions posted on the blockchain. For instance, ZKRPs are a key ingredient in confidential transactions [Max16, BAZB20, Poe16] — which hide the amount of each transaction posted on the blockchain. The transaction amounts are stored in a committed fashion, and to ensure validity of the transaction the sender must prove that the sum of the output amounts does not exceed the sum of the input amounts. For this check to be sound, the sender must also prove that all output amounts are positive (else an adversarial sender could commit to negative output amounts and create coins out of thin air). For commitments in a group, such as Pedersen commitments, this positivity check also involves showing that the committed value is much less than the order of the group. This check essentially amounts to showing that the committed value is in some integer range $[0, 2^k - 1]$ and is done via a ZKRP. Additionally, ZKRPs are heavily used in protocols for blockchain auditing and solvency solutions [DBB+15, Cam14, JC21, CB21] to show that transactions or reserves of an organization satisfy certain policies.

This increased interest in ZKRPs has also resulted in a growing number of proposed constructions with different characteristics and properties. With numerous ZKRP constructions available, selecting the suitable scheme for a specific application can be challenging. The goals of this SoK are to organize the space on the various techniques used to construct range proofs, compare their properties in a systematic way, identify open research questions, and provide a guideline to select the appropriate protocol for each type of application.

Our contributions and organization. We start by defining the necessary background on cryptographic schemes and computational assumptions in Section 2. In Section 3, we provide a taxonomy of general approaches used in the construction of zero-knowledge range proofs. Concretely, we identify three underlying methods used in the constructions of known ZKRP schemes: (a) square decomposition, (b) binary/n-ary decomposition and (c) hash-chain approach. We describe each method in detail, and for n-ary decomposition we present an abstraction that allows us to synthesize the several techniques used. Our abstraction is of independent interest, and could potentially lead to new insights. Then, in Section 4, we collect the set of properties beyond the standard soundness and zero-knowledge that are desirable in certain application scenarios of ZKRPs, such as aggregation, transparent setup and efficiency considerations. In Sections 5-7 we classify all known (to the best of our knowledge) ZKRP constructions under the three methods we identified in Section 3. For each method, we provide an analytical

list of known protocols and we compare all protocols based on the desirable properties listed in Section 4. In Section 8, we provide a guideline for how to select the best type of ZKRP construction based on the desired properties and then in Section 9, we report storage and computation (verifier/prover time) costs of the most popular ZKRP constructions using existing and new benchmarks. In Section 10, we identify a series of research gaps relevant to ZKRP which we believe can serve as a starting point for future research works. Finally, we provide a more detailed list of known ZKRP applications in Appendix 11.

Comparison with prior work. We compare our paper with the previous survey of range proofs by Morais, Koens, van Wijk, and Koren [MKvWK19]. The technical portion of [MKvWK19] focuses largely on Boudot's four-square decomposition construction [Bou00], the signature-based construction of CCs [CCS08], and Bulletproofs [BBB+18]. It omits or does not go into detail on many other works, such as the line of lattice- and code-based constructions, the newer and more efficient square-decomposition constructions, the polynomial commitment-based constructions, and hash chain constructions. In particular, many of the most efficient schemes such as Sharp [CGKR22] and BFGW [BFGW20] are not covered in their survey. Their work also provides a comparison only of the three schemes that it focuses on. Our SoK is significantly more comprehensive, and here is a summary of how our work goes beyond [MKvWK19]. First, to the best of our knowledge, we provide a complete description of techniques and schemes in the ZKRP category and we extensively compare all such schemes based on their techniques, assumptions, and other properties. Additionally, we observe a useful abstraction for binary decomposition-based range proofs, breaking such proofs into two components, and presenting the techniques used for each of these components. An important aspect for our work, especially for practitioners who will use our SoK to determine the most suitable ZKRP for their application, is that we provide new benchmarks and assemble existing benchmarks for easier comparison. We plan to open-source the code used for our benchmarks. Finally, we include open questions and research gaps, and a flowchart to help identify the most appropriate range proof construction family for various applications.

2 PRELIMINARIES

We use boldface, like $\mathbf{a} = (a_1, \ldots, a_n)$, to denote a vector. We use $\circ$ to denote the Hadamard product, i.e., $\mathbf{a} \circ \mathbf{b} = (a_1 b_1, \ldots, a_n b_n)$. For a nonzero value a, we use $\mathbf{a^n}$ to denote the vector $(1, a, a^2, \ldots, a^{n-1})$. We let $\mathbf{0^n}$ denote the length-n vector $(0, \ldots, 0)$. We use λ to denote the security parameter, $\mathcal{A}$ to denote an adversary, $\mathbb{Z}$ to denote the integers, and $negl(\cdot)$ to denote a negligible function. We use the word *efficient* to mean probabilistic polynomial time.

Definition 2.1 (Commitment scheme [KL07]). A commitment scheme is a pair of efficiently computable algorithms (Gen, Com) where:

- $\mathsf{Gen}(1^\lambda)$ is an efficient randomized algorithm that outputs public parameters pp.
- $\mathsf{Com}(\mathsf{pp}, m, r)$ is an efficient deterministic function that takes as input the public parameters, a message m, and randomness r. It outputs a commitment to m.

A commitment scheme must be *binding* and *hiding*, defined as follows:
A commitment scheme is *binding* if for all p.p.t. adversaries $\mathcal{A}$, it is infeasible to come up with two different messages corresponding to a given commitment.

$$\Pr_{\mathsf{pp}\leftarrow\mathsf{Gen}(1^\lambda)}\left[\begin{array}{c}(m_0, r_0), (m_1, r_1) \leftarrow \mathcal{A}(1^\lambda, \mathsf{pp}) \wedge \\ (m_0 \neq m_1) \wedge \\ \mathsf{Com}(\mathsf{pp}, m_0, r_0) = \mathsf{Com}(\mathsf{pp}, m_1, r_1)\end{array}\right] = negl(\lambda)$$

A commitment scheme is computationally (resp., statistically) *hiding* if for all p.p.t. (resp., unbounded) adversaries $\mathcal{A}$, it is infeasible to distinguish whether a commitment corresponds to any m_0 or m_1 known to $\mathcal{A}$. That is, for all m_0, m_1:

$$\Pr_{r\leftarrow\$}\left[\begin{array}{c}c \leftarrow \mathsf{Com}(\mathsf{pp}, m_0, r) \\ \mathcal{A}(1^\lambda, \mathsf{pp}, c, m_0, m_1) = 1\end{array}\right] \approx \Pr_{r\leftarrow\$}\left[\begin{array}{c}c \leftarrow \mathsf{Com}(\mathsf{pp}, m_1, r) \\ \mathcal{A}(1^\lambda, \mathsf{pp}, c, m_0, m_1) = 1\end{array}\right]$$

A commitment scheme is *homomorphic* if $\mathsf{Com}(\mathsf{pp}, m_0, r_0) + \mathsf{Com}(\mathsf{pp}, m_1, r_1) = \mathsf{Com}(\mathsf{pp}, m_0 + m_1, r_0 + r_1)$.

Next we define Zero-knowledge proof and non-interactive zero-knowledge proof (NIZK). Most of the ZKRPs in this SoK are in fact non-interactive. In the following sections, we will skip mention of the non-interactive aspect, unless not clear from context. We provide informal definitions next, while deferring the formal definition of NIZK and its properties to appendix A.

Definition 2.2 (Zero-knowledge proof). Let L be a language in NP and R be a polynomially verifiable relation, such that $x \in L \iff \exists w : R(x, w)$. A Zero-knowledge proof system for L is a tuple of efficient interactive algorithms (Prover, Verifier, Simulator), such that the following properties hold:

- *Completeness.* Given $(x, w) \in R$, the honest execution of the Prover (given x, w) and the Verifier (given only x) result in the Verifier outputting 1.
- *Soundness.* Given $x \notin L$, a malicious Prover interacting with the Verifier can only make it output 1 with negligible probability.
- *Zero-Knowledge.* Given $x \in L$, the Simulator can produce an interaction transcript of an honest Prover with a (possibly) malicious Verifier, that is computationally indistinguishable from an actual execution transcript of the Prover with the Verifier. Note that the Simulator doesn't get w, while the Prover gets w.

A *non-interactive zero-knowledge (NIZK)* proof system is a zero-knowledge proof system, where the Prover, given (x, w) just sends one message π to the Verifier and the Verifier outputs 0/1 based on (x, π). A NIZK has an additional setup algorithm *CRSGen*, which outputs a common reference string (CRS) used by all the proofs and verifications. Instead of a CRS, some NIZKs can also specify a random oracle. The Simulator algorithm is allowed to keep trapdoors about the CRS, or be able to simulate the Random Oracle.

A zero-knowledge *proof of knowledge* requires that an adversary which produces a valid proof for a statement also knows a valid witness. This is formally captured by requiring the existence of an extractor, which can run the adversary's code and produce the witness.

Definition 2.3 (Zero-knowledge range proof (ZKRP)). A *zero-knowledge range proof (ZKRP)* is a zero-knowledge proof of knowledge for the

following relation:

$$R_{\text{pp}} = \{((y, u, v), (m, r)) : y = \text{Com}(\text{pp}, m, r) \wedge u \leq m \leq v\}$$

where pp, y, u, and v are known to the verifier, and Com is some particular commitment scheme.

A question may arise since pp is hard-coded in the language definition: what if a malicious prover samples pp badly and thus renders the NIZK-soundness property vacuous? We note that most applications require both commitment security and NIZK-soundness. This enforces that the attacker of the application's security cannot badly sample pp.

Pedersen commitments. Most range proofs use *Pedersen commitments* [Ped91] as the underlying commitment scheme. Let $\mathbb{G}$ be a cyclic group of prime order and g and h be generators of that group, where the relationship between g and h is not known. The Pedersen commitment $\text{Com}(x, r)$ for a value $x \in \mathbb{G}$ with randomness r is $g^x h^r$.

Pedersen commitments are statistically hiding, and their binding property is based on the hardness of the discrete logarithm assumption.

Definition 2.4 (Discrete Logarithm Assumption). Let $\mathbb{G}$ be a group of order p and let g be a generator of $\mathbb{G}$. A challenger samples a random $x \leftarrow \mathbb{Z}_p$ and sends g^x to an adversary. The Discrete Logarithm Assumption states that it is infeasible for the adversary to output x, given $(\mathbb{G}, g, g^x)$.

Apart from the Discrete Logarithm setting, we will also describe schemes based on the hardness of the RSA problem, as well as lattices.

Definition 2.5 (RSA Assumption). A challenger samples primes p and q and sets $N = pq$. It picks a quantity e co-prime to $\phi(N)$. Then it randomly samples $z \leftarrow [1, N]$ and sends (N, e, z) to the adversary. The adversary outputs y. The RSA Assumption states that the probability of $y^e = z \pmod N$ is negligible.

Definition 2.6 (Strong RSA Assumption). The Strong RSA Assumption states that the RSA problem is intractable even when the adversary is allowed to choose the public exponent e (for $e \geq 3$).

Definition 2.7 (SIS Assumption). Let $q, n, m \in \mathbb{Z}$, $\beta \in R$ be given. A challenger samples a random matrix $A \leftarrow \mathbb{Z}^{n \times m}$. The SIS Assumption states that it is infeasible for an adversary to find a nonzero m-vector e, such that $Ae = 0 \mod q$ and $||e||_2 \leq \beta$.

3 GENERAL APPROACHES

Efficient zero-knowledge range proofs typically use three classes of approaches: square decomposition, n-ary decomposition, and hash chains. We present these approaches below, then explore specific instantiations of these approaches in more detail in their respective sections. We also mention the approach of using generic zero-knowledge proofs.

We describe these approaches for proving that a committed value lies in a range of the form $[0, n^k - 1]$, or that a committed value is positive in the case of square decomposition. Most works consider ranges of this form, which may seem at a first glance to be a relaxed version of the problem. However, when the commitments used are homomorphic, it turns out to be sufficient for constructing more general range proofs with only a small amount of work to translate.

Assume that we the ability to prove that any committed value is in the interval $[0, n^k - 1]$. To prove that z is in some interval $[u, v]$, one can show first that $(z - u) \in [0, n^k - 1]$ and then that $(v - z) \in [0, n^k - 1]$. Thus, $z \geq u$ and $z \leq v$. Certain constructions from integer commitments (e.g., CKLR [CKLR21]) can combine these checks into proving a single equation: $(z - u)(v - z) \geq 0$. It is easy to obtain commitments for $(z-u)$ and $(v-z)$ homomorphically, given a commitment to z. For non-homomorphic commitments, one can do this translation by creating a commitment c to $z - u$, proving in zero knowledge that c indeed commits to $z - u$, and performing this range proof with respect to c.

3.1 Square decomposition

The square decomposition method involves writing the committed integer as a sum of squares. A common version of this method, the four-square decomposition method, uses Lagrange's four-square theorem. This theorem states that for any integer $z \in \mathbb{Z}_{\geq 0}$, there exist $x_1, x_2, x_3, x_4 \in \mathbb{Z}$ such that

$$z = x_1^2 + x_2^2 + x_3^2 + x_4^2 \tag{1}$$

Thus, to prove that a committed value z is non-negative, it suffices to prove knowledge of $x_1, \ldots, x_4$ such that Equation 1 holds. This approach requires a special type of commitment called an integer commitment, which ensures that equations that hold over committed values also hold *over the integers*. The issue that this property avoids is that Equation 1 may hold for a negative z if we are working in some group rather than over $\mathbb{Z}$. For example, in $\mathbb{Z}_5$ it is possible that $z = -1$ and $0^2 + 1^2 + 2^2 + 2^2 = 9 = z \pmod 5$.

3.1.1 Integer commitments. An *integer commitment scheme* is a commitment scheme where binding holds *over* $\mathbb{Z}$. That is, for all p.p.t. adversaries $\mathcal{A}$,

$$\Pr_{\text{pp} \leftarrow \text{Gen}(1^\lambda)} \left[\begin{array}{c} (m_0, r_0), (m_1, r_1) \leftarrow \mathcal{A}(1^\lambda, \text{pp}) \\ \wedge (m_0 \neq_{\mathbb{Z}} m_1) \\ \wedge \text{Com}(\text{pp}, m_0, r_0) = \text{Com}(\text{pp}, m_1, r_1) \end{array} \right] = negl(\lambda)$$

where $m_0 \neq_{\mathbb{Z}} m_1$ denotes that m_0 and m_1 are not equal *over the integers*. Bounded integer commitments (used in [CKLR21]) satisfy the same binding property, but are weaker in that the message space is restricted to some bounded interval, e.g., $\{x \in \mathbb{Z} : |x| \leq B\}$. For constructing range proofs, this boundedness is not an issue as long as the ranges in question are well within the bounds.

Pedersen commitments, for example, are not integer commitments as their message space is $\mathbb{Z}_p$, and any messages that are equivalent $\pmod p$ result in the same commitment given the same randomness: $g^m h^r = g^{m+p} h^r$ over a cyclic group of order p. This attack against binding fails if the order of the group is unknown, and indeed many integer commitment schemes (e.g., Fujisaki-Okamoto commitments, and constructions of [CKLR21, CGKR22]) operate in groups of unknown order.

Fujisaki-Okamoto commitments [FO97]. We recall an overview of FO commitments but refer the reader to [FO97] for details. FO commitments operate over a group of unknown order $\mathbb{Z}_N$. g and h are generators of large subgroups of $\mathbb{Z}_N$, whose relation is unknown. The commitment to $x \in \mathbb{Z}$ is

$$\mathsf{Com}_{FO}(\mathsf{pp}, x, r) := g^x h^r$$

This commitment is computationally hiding when r is chosen uniformly in the interval $[2^{-\lambda} \cdot N + 1, \ldots, 2^{\lambda} \cdot N - 1]$. Fujisaki-Okamoto commitments are binding under the factoring assumption.

3.2 n-ary decomposition

The n-ary decomposition method involves committing to the digits of the committed value z in some base n. For simplicity, assume for this explanation that we use base 2, although certain approaches can be generalized to other bases. Thus, if the prover wishes to show that $z \in [0, 2^k - 1]$, the prover writes $z = z_0 \cdot 2^0 + z_1 \cdot s^1 + \ldots + z_{k-1} \cdot 2^{k-1}$ and generates commitments to $z_0, \ldots, z_{k-1}$. The prover then shows that both of the following properties hold, which we present as predicates:

Digit validity ($\mathsf{DV}(\mathbf{z})$): $\mathsf{DV}(\mathbf{z}) = 1$ if and only if $z_i \in \{0, 1\}$ for all $i \in [0, k-1]$.

Representativeness ($\mathsf{Rep}(\mathbf{z}, z)$): $\mathsf{Rep}(\mathbf{z}, z) = 1$ if and only if $z = \sum_{i=0}^{k-1} z_i \cdot 2^i$.

In terms of these predicates, the n-ary decomposition method proves membership in the following relation:

$$\mathcal{R}_{\mathsf{decomp}} = \{(\mathsf{pp}, (c_1, c_2, n, k), (z, \mathbf{z}, r, \mathbf{r})) : c_1 = \mathsf{Com}(\mathsf{pp}, z, r) \wedge c_2 = \mathsf{Com}(\mathsf{pp}, \mathbf{z}, \mathbf{r}) \wedge \mathsf{DV}(\mathbf{z}) \wedge \mathsf{Rep}(\mathbf{z}, z)\}$$

There are (at least) four common tools used to show that the digits are valid for the desired base; i.e., for binary decomposition they all lie in $\{0, 1\}$. These tools include zero-knowledge *set membership arguments*, *product arguments*, *inner product arguments*, and *polynomial commitments*. These strategies are primarily applicable for base 2, with the exception of set membership, which easily extends to any arbitrary base.

3.2.1 Set membership. A set membership proof shows that a committed value lies in some publicly known set; that is, it is a proof of knowledge for the following relation:

$$\{(\mathsf{pp}, (\Phi, y), (m, r)) : y = \mathsf{Com}(\mathsf{pp}, m, r) \wedge m \in \Phi\}$$

Although one could define a set membership proof with respect to a *private* committed set, in our application the set is determined by the publicly known base.

Digit validity. Set membership arguments are useful for instances of $\mathcal{R}_{\mathsf{decomp}}$ where the commitment scheme used for $\mathbf{z}$ commits to its components individually; that is,

$$c_2 = (\mathsf{Com}(\mathsf{pp}, z_0, r_0), \ldots, \mathsf{Com}(\mathsf{pp}, z_{k-1}, r_{k-1}))$$

for some scheme Com. Then, one can show digit validity by providing a set membership proof for each element of c_2, with respect to the set $\Phi = \{0, 1, \ldots, n-1\}$. However, such protocols require commitments and range proofs of length at least linear in k.

Representativeness. There is no general way to show representativeness using set membership proofs; schemes using this construction (e.g., [CCS08]) rely on properties of the specific commitment scheme used.

3.2.2 Product arguments. A product argument is a proof system for showing that the product of two committed values a and b is some value c. Typically, this equality holds in the group underlying the commitment scheme. For example, for Pedersen commitments in a group of prime order p, this argument shows that $ab \equiv c \pmod{p}$. For integer commitments, we have the stronger property that this equality holds over the integers: $ab = c$.

Digit validity. Product arguments are useful for proving digit validity base 2, if as with set membership c_2 consists of individual bit commitments. To show that a committed bit b is in $\{0, 1\}$, the prover can commit to a value a and prove that $ab = 0$ and $a + b = 1$. Observe that if $b \neq 0$, a must be 0 to satisfy the first equation. Then the second equation implies that $b = 1$. Thus, b must be 0 or 1. Furthermore, the prover can always find a satisfying a; if $b = 0$, $a = 1$, and if $b = 1$, $a = 0$. Inner product arguments, which we present next, allow the prover to simultaneously show many product relations more efficiently.

Representativeness. As is the case with set membership proofs, product arguments primarily useful for showing digital validity rather than representativeness.

3.2.3 Inner product arguments. An inner product argument (IPA) is a proof system for showing that the inner product of two committed vectors is some value. The inner product used in Bulletproofs [BBB+18] shows the following relation, using Pedersen commitments, where $\mathbb{G}$ denotes a group of prime order:

$$\left\{\left(\mathbf{g}, \mathbf{h} \in \mathbb{G}^k, P \in \mathbb{G}, z \in \mathbb{Z}_p \; ; \; \mathbf{a}, \mathbf{b} \in \mathbb{Z}_p^n\right) \; : \; P = \mathbf{g}^{\mathbf{a}} \mathbf{h}^{\mathbf{b}} \wedge z = \langle \mathbf{a}, \mathbf{b} \rangle\right\}$$

Bulletproofs also constructs an argument for the Hadamard product relation (i.e., $\mathbf{c} = \mathbf{a} \circ \mathbf{b}$) from their inner product argument, though we do not present the details here.

Digit validity. A useful fact used when constructing zero-knowledge range proofs from inner product arguments is that with overwhelming probability, the inner product of a nonzero vector $\mathbf{a}$ and a random vector $\mathbf{b}$ is nonzero. Thus, the prover can convince the verifier that $\mathbf{a}$ is $\mathbf{0}^k$ by showing that its inner product with a random challenge vector is 0. Using the same idea as for product arguments, the prover can commit to the binary representation of the given value as a vector $\mathbf{z}$, then use an inner product argument to show simultaneously that *all* components of this vector are indeed bits. That is, the prover commits to a vector $\mathbf{z}' = \mathbf{1}^k - \mathbf{z}$, and shows:

$$\langle \mathbf{z}', \mathbf{1}^k - \mathbf{z} \rangle = 0 \text{ and } \mathbf{z}' \circ \mathbf{z} = \mathbf{0}^k$$

The lattice-based scheme [ALS20] uses this approach as well.

Representativeness. Although we presented an inner product relation where the value z is a public input, many inner product arguments, such as that of Bulletproofs, work also when z is secret and the public input includes only a commitment to z. One shows representativeness by a single application of this inner product argument, showing $\langle \mathbf{z}, \mathbf{2}^k \rangle = z$.

Bulletproofs combines some of these checks for greater efficiency and uses blinding factors to make their argument zero-knowledge.

3.2.4 Polynomial commitments. A polynomial commitment scheme allows a prover to commit to a polynomial $p(\cdot)$ over a finite field

$\mathbb{F}_p$, such that a verifier can query a point x to the prover, which can respond with $p(x)$ and a proof π that this evaluation is correct. The scheme should be hiding in that the commitment reveals nothing about the polynomial, and the evaluation proofs reveal no extra information beyond the evaluations themselves. Polynomial commitments are binding in that it is computationally infeasible to produce a verifying proof for an incorrect evaluation of the committed polynomial. A useful property of polynomial commitments is that it is easy for a prover to show that a committed polynomial is identically zero, by providing a proof that its evaluation at a random point is zero. By binding and the Schwartz-Zippel lemma, this occurs with only negligible probability if the polynomial is nonzero.

The following approach, which we describe at a high level, was introduced in BFGW [BFGW20] and is detailed nicely in [Tom20]. Suppose that we are given a commitment to z in the form of a polynomial commitment to f such that $f(1) = z$. In constructing a range proof for $z \in [0, 2^k - 1]$, it is useful to work over a subgroup $H = \{1, \omega, \omega^2, \ldots, \omega^{k-1}\}$ and use polynomials whose evaluations over H encode the binary representation of z. That is, the prover computes a polynomial g such that:

$$g(\omega^{k-1}) = z_{k-1}$$
$$g(\omega^i) = 2g(\omega^{i+1}) + z_i \; \forall i \in \{0, \ldots, k-2\}$$

Another useful property of polynomial commitments is that one can show that a polynomial $g(X)$ is zero on all of H by committing to a related polynomial $g'(X)$ and proving that $g'(X)$ is identically zero over $\mathbb{F}_p$.

Digit validity. The prover shows that the following two polynomials are zero over all of H:

$$w_2 = g \cdot (1-g)(X-1)(X-\omega)\cdots(X-\omega^{k-2})$$
$$w_3 = [g(X) - 2g(X\omega)] \cdot [1 - g(X) + 2g(X\omega)] \cdot (X - \omega^{k-1})$$

w_2 has zeros at $1, \omega, \ldots, \omega^{k-2}$ by construction. It is zero at 1 if and only if $g(\omega^{k-1}) \in \{0, 1\}$. For w_3, observe that $g(X) - 2g(X\omega)$ is exactly z_i when evaluated at ω^i. Therefore, w_3 is zero at $\{1, \ldots, \omega^{k-2}\}$ if and only if $z_i \in \{0, 1\}$.

Representativeness. The prover shows that the following polynomial is zero over all of H:

$$w_1 = (g - f)(X-\omega)(X-\omega^2)\cdots(X-\omega^{k-1})$$

As [BFGW20] notes, this approach can be instantiated with any polynomial commitment scheme that is hiding, binding, and additively homomorphic.

3.3 Hash chains

Hash chains can be used to prove that a committed value at least some threshold. In the hash chain approach, a commitment to a value z is $C_z = H^z(r)$, the output of a hash function applied z times to a random r. The proof that z exceeds some threshold t is $\pi = H^{z-t}(r)$. A verifier can check that $H^t(\pi) = C_z$; if $z < t$, then $z - t$ is negative and it is infeasible for a cheating prover to compute a preimage of r under H.

This simple hash chain requires prover and verifier time that is exponential in k for ranges $[0, 2^k - 1]$. However, using decomposition techniques, HashWires constructs a hash chain-based range proof requiring only $O(k)$ work.

3.4 Generic zero-knowledge

There are many efficient generic zk-SNARKs, such as [Gro16, GWC19, BSBHR18, BFS20a]. These proof systems can be used to construct range proofs. However, because they are generic and do not leverage the structure of the range proof relation, they are less efficient than the tailored range proofs we explore. In Section 9, we include efficiency benchmarks for Groth16 [Gro16], one of the most popular generic zk-SNARKs used in practice.

It is worth noting that practical benefits may outweigh these efficiency losses. In particular, because of their wide-ranging applications, generic zk-SNARKs offer convenient, well-engineered, and optimized libraries. For example, we used Circom [BMIMT+22] and rapidsnark [ide23] for our Groth16 benchmarks. Even so, the prover and verifier times for Groth16 are roughly an order of magnitude larger than the more tailored range proofs. Furthermore, if range proofs are required in a larger system that already uses a generic zk-SNARK elsewhere, using this zk-SNARK for the range proof as well may be practically convenient.

4 DESIRABLE PROPERTIES

All zero-knowledge range proofs must satisfy the standard notions of soundness, completeness, and zero knowledge. All ZKRPs that we cover in this SoK are non-interactive. In this section, we discuss some additional nice features that might be desirable in some settings.

Efficiency. Unsurprisingly, it is desirable for ZKRPs to be efficient. In blockchain applications, where a transactor must pay for the storage cost and the amount of computation done by validators, it is especially important to minimize proof size and verifier time. The proof size should be at most linear in k for intervals $[0, 2^k - 1]$, and several schemes offer even constant-sized proofs. Though proof size and verifier time are often priorities, prover time also should not be prohibitively large. In Section 9, we discuss efficiency further.

Transparent setup. Some range proofs require public parameters that are generated using secret randomness. It is crucial for the security of the proofs that this randomness is not known to the prover. For example, several square decomposition range proofs use RSA-based integer commitments, which require an RSA modulus. Importantly, this modulus N must be generated in such a way that no party know the factorization of $N = pq$. Similarly, BFGW [BFGW20] instantiated with KZG commitments requires a powers-of-tau common reference string, which consists of a series of values g^{τ^i}, where no party knows τ. Protocols that require secrecy of the randomness used in parameter generation are said to require *trusted setup*.

Ideally, protocols should have a *transparent* setup procedure that does not require secret randomness. Note that trusted setup is different from having a trusted issuer responsible for distributing the proper commitments to users, e.g., a Pedersen commitment corresponding to that user's account balance. Any protocol needs

to assume that the prover and verifier agree on the commitment at hand.

Aggregation. Aggregation allows multiple range proofs to be compressed into a single succinct proof. That is, a single prover holding m commitments to values in the same range $[0, 2^k - 1]$ can efficiently generate a short aggregate proof π proving all of these range statements simultaneously. For this aggregation property to be nontrivial, π should be shorter than the concatenation of $\pi_1, \ldots, \pi_m$. For example, for Bulletproofs, Bulletproofs+, and Bulletproofs++, the aggregate proof for m values in $[0, 2^k - 1]$ consists of only $O(\log(m \cdot k))$ group elements. As the concatenation of m proofs would require $O(m \cdot \log(k))$ group elements, aggregation results in considerable space savings.

In the notion of aggregation considered so far, a single prover knows the openings of all commitments that are being aggregated. A stronger notion of *multi-prover* aggregation allows one to combine range proofs generated by multiple provers, who wish to hide their openings from one another. Bulletproofs enables such aggregation via an MPC protocol run by the parties holding the commitments. Multi-prover aggregation is harder to achieve, and is less well studied than single-prover aggregation.

Aggregation is especially useful for confidential transactions, where minimizing the amount of space used on-chain decreases gas costs. Since range proofs are used to show non-negativity, all range proofs typically prove membership in the same interval.

Batch verification. A related property is batch verification, where there exists a process for verifying many proofs together that is more efficient than verifying each proof individually. Batch verification is especially useful in blockchain applications, where a block proposer can aggregate the range proofs for its block and other validators can batch verify this proof more efficiently. Bulletproofs provides batch verification, using an observation that verifying many statements of the form $g^x = 1$ can be done by combining them into a single equation requiring fewer exponentiations.

Aggregated range proofs often naturally enable batch verification, as some of the work is effectively done by the aggregator. However, neither aggregation nor batch verification in general implies the other.

Compatibility with homomorphic commitments. A commitment scheme Com is homomorphic if $\mathsf{Com}(m_0, r_0) + \mathsf{Com}(m_1, r_1) = \mathsf{Com}(m_0 + m_1, r_0 + r_1)$. It is convenient for applications such as confidential transactions for the underlying commitments to be homomorphic; in particular, homomorphism makes it easier to prove that the sum of transaction output amounts is at least the sum of input amounts.

Most ZKRPs use Pedersen commitments, which are homomorphic. Some exceptions are HashWires [CCL+21] and various lattice-based constructions, which often achieve weaker homomorphism.

5 SQUARE DECOMPOSITION CONSTRUCTIONS

Recall that the square decomposition method involves writing the committed value as the sum of four squares and proving that this equality holds over the integers. Integer commitments, which were discussed in greater detail in Section 3, are a useful tool here:

Integer commitments. An integer commitment scheme is a commitment scheme for which binding holds over the integers: it is computationally infeasible for an adversary to find messages m_0, m_1 and randomness r_0, r_1 such that $\mathsf{Com}(m_0, r_0) = \mathsf{Com}(m_1, r_1)$, where $m_0 \neq m_1$ *over* $\mathbb{Z}$.

Approaches in this class combine integer commitment schemes with a way to prove in zero knowledge that, given a commitments Com_x and Com_y, the committed values satisfy $x^2 = y$. This implies that y is non-negative. One can generalize this argument to work not just for squares y, but for all non-negative integers.

Boudot [Bou00] introduced the approach of proving that a committed value is positive by representing an arbitrary integer as a sum of squares (although not four squares). It uses Fujisaki-Okamoto commitments [FO97], which require a group of unknown order such as an RSA group. Damgård-Fujisaki commitments [DF01] are slightly more efficient integer commitments used in subsequent work [Lip03] which refined Boudot's idea and used Lagrange's four square theorem (which states that every integer can be written as the sum of the squares of four integers). In order to do so, it also introduced an efficient algorithm for finding this four-square decomposition. [Gro05] similarly followed this approach and improved its efficiency by observing that x's of a certain form can be written as the sum of only three squares rather than four. [CPP17] further improved the efficiency and showed that the RSA assumption (rather than the strong RSA assumption, as previously shown) is sufficient to show the security of Damgård-Fujisaki commitments.

The integer commitments used by all of [Bou00, Lip03, Gro05] require groups of unknown order and therefore trusted setup procedures. A newer line of work [CKLR21, CGKR22] develops new integer commitment schemes, some of which do not require a trusted setup. These schemes also yield much better efficiency, though Bulletproofs and subsequent binary-decomposition-based proofs are still more efficient in practice due to compatibility with available optimized libraries.

CKLR [CKLR21] build a *bounded integer commitment* by modifying Pedersen commitments; their scheme essentially enforces that the Pedersen commitment can only be opened to values within some bounded range. They then use this bounded integer commitment to construct their ZKRP following the square decomposition approach. However, their commitment scheme operates over *rationals* rather than integers; while honest openers round these rationals to integers, malicious openers may open to rationals instead which can be problematic for some applications and results in a relaxed notion of soundness. Sharp [CGKR22] improves upon CKLR in several ways. In addition to improving over the efficiency of CKLR, Sharp is compatible with standard Pedersen commitments. This is because Sharp effectively moves CKLR's modifications of Pedersen commitments to the *proof* rather than modifying the commitment itself. Two variants of Sharp ($\mathsf{Sharp}_{\mathsf{GS}}$, $\mathsf{Sharp}^{\mathsf{PO}}_{\mathsf{SO}}$), like CKLR, achieve a relaxed notion of soundness. However, they show how to boost the soundness by adding an additional commitment using a hidden-order group such as an RSA group or class group; the resulting variants $\mathsf{Sharp}_{\mathsf{HO}}$ achieve standard soundness but require longer proofs. The RSA version also requires a trusted setup. Finally, Sharp improves over CKLR by offering some batching capabilities.

Square Decomposition-Based Range Proofs			
Scheme	Commitment Scheme	Assumptions	Transparent Setup
Boudot [Bou00]	F-O [FO97]	Strong RSA	N
Lipmaa [Lip03]	RDF integer comm.*	Strong RSA	N
Groth [Gro05]**	RDF integer comm.*	Strong RSA	N
CKLR [CKLR21]	$\overline{\text{Ped}}$***	DLOG (optionally DSLE)	N
CKLR [CKLR21]	ElGamal variant [CKLR21]	DXDH, ORD	Y (class groups)
Sharp_{GS}, $\text{Sharp}_{\text{SO}}^{\text{PO}}$ [CGKR22]†	Pedersen	DLOG, SEI	Y
Sharp_{HO} [CGKR22]†	Pedersen	1/2-fROOT	N (RSA), Y (class groups)

Figure 1: Properties of square decomposition-based range proofs

*An extension of the Dåmgard-Fujisaki commitment [DF01] that [Lip03] constructs.
**[Gro05] is not exactly a new scheme; its contribution is observing a trick that can be applied to make [Lip03] more efficient. Integers of a certain form can be written as a sum of *three* squares, and one can quickly find this decomposition.
*** A bounded integer commitment scheme based on Pedersen commitments.
†Sharp is only a *relaxed* range proof and not sufficient for all applications. [CGKR22] has a thorough discussion; it is sufficient for anonymous credentials and can be used for some but not all proofs in anonymous transactions, with some modifications. Sharp_{HO} refers to a scheme where Sharp_{GS} or $\text{Sharp}_{\text{SO}}^{\text{PO}}$ is modified using an additional commitment requiring an RSA group or class group in order to achieve improved soundness.

6 BINARY DECOMPOSITION CONSTRUCTIONS

CCs [CCS08] introduced the n-ary decomposition paradigm to zero-knowledge range proofs. CCs [CCS08] operates over Pedersen commitments and constructs a zero-knowledge set membership protocol by having the verifier publish a signature of each element in the set. The prover then shows in zero knowledge that it knows a signature of its committed value x under the verifier's secret key; by unforgeability this is only possible if the value is in this set. By choosing this set to be $\{0, \ldots, n-1\}$ for base n, the prover can commit to the digits of x and prove that they are valid digits under that base. CCs then uses properties of Pedersen commitments to show that the committed digits indeed represent x. The size of the proof is linear in $\log_n 2^k$, where n is the base used and the range is of size 2^k. By optimizing the choice of the base n, this results in a slightly sublinear (in k) proof size for a range $[0, 2^k - 1]$. This scheme requires a trusted setup for the signature generation, and it does not offer aggregation.

Subsequent constructions (which we call "Bulletproofs-style") improve on the efficiency of CCs to avoid this near-linear dependence on k. They use inner product arguments or polynomial commitment schemes in clever ways to avoid showing individually that each bit is in $\{0, 1\}$; instead, they are able to roll all of these checks into a shorter proof.

There are also several newer lattice- and code-based constructions that use binary decomposition. While these schemes are less efficient and have very large proofs, their main merit is that they are plausibly post-quantum secure. Additionally, they do offer transparent setup. Developing more practical lattice-based ZKRPs is an interesting research direction.

When surveying binary decomposition constructions, we separate them into two categories: Bulletproofs-style constructions, which are very practical; and lattice-based constructions, which are primarily of theoretical interest.

6.1 Bulletproofs-Style Constructions

Bulletproofs [BBB+18], arguably considered the state-of-the-art range proof scheme, uses the binary decomposition technique.

Bulletproofs combines the binary decomposition technique with an inner product argument to enable the prover to send only $O(\log k)$ elements. Bulletproofs improves and uses their improvement of an inner product argument (IPA) of [BCC+16] where the prover sends only $O(\log k)$ group elements for an IPA over length-k vectors. The key idea in Bulletproofs is that the prover can use this IPA to execute the binary decomposition approach more efficiently; we give intuition for this idea here.

We write $x = a_0 \cdot 2^0 + a_1 \cdot 2^1 + \ldots + a_{k-1} \cdot 2^{k-1}$ and let $\mathbf{a_L} = [a_0, a_1, \ldots, a_{k-1}]$. We let $\mathbf{2^k} := [2^0, 2^1, \ldots, 2^{k-1}]$. The prover shows that it knows a vector $\mathbf{a_R}$ such that the following hold:

(1) $\mathbf{a_L} \cdot \mathbf{a_R} = \mathbf{0^k}$
(2) $\mathbf{a_L} - \mathbf{a_R} = \mathbf{1^k}$
(3) $\mathbf{a_L} \cdot \mathbf{2^k} = x$

Conditions (1) and (2) show that each component of $\mathbf{a_L}$ is in $\{0, 1\}$, using the standard inner product strategy described in Section 3. Condition (3) shows that indeed $\mathbf{a_L}$ contains the binary decomposition of x.

These three checks can be combined into a single invocation of the IPA. The IPA used employs a technique that reduces each IPA of length-n vectors to an equivalent IPA over length-$\frac{n}{2}$ vectors. Using this IPA results in a proofs size of $O(\log_2 k)$.

Subsequent works [CHJ+22, Eag22, WC22, WCL23] slightly optimize Bulletproofs but keep the scheme and its properties (in particular, its transparent setup and aggregation properties) largely the same. Bulletproofs+ [CHJ+22] slightly optimizes the Bulletproofs argument to reduce the number of group elements sent by the prover. Bulletproofs++ [Eag22] further improves efficiency by reducing both prover and verifier time. All of these Bulletproofs derivatives maintain the same aggregation properties.

BFGW [BFGW20] takes a different approach to the binary decomposition idea, using a polynomial commitment scheme. We

detail this approach in Section 3. This scheme assumes that the commitment to a value x is formed as commitment to a polynomial f such that $f(1) = x$. For some polynomial commitment schemes, such a commitment is nonstandard; conveniently, there is a version of KZG commitments for which this is a Pedersen commitment.

BFGW works with any hiding and binding polynomial commitment scheme, yielding different properties based on the scheme used. Notably, when instantiated with KZG commitments [KZG10], BFGW has constant-sized proofs and is competitive efficiency-wise with Bulletproofs. Though KZG commitments require a trusted setup, this setup ceremony is perhaps one of the most commonly run, and some blockchains such as Ethereum have run a KZG ceremony.[1] In Section 9, we provide the first efficiency (prover and verifier time) benchmarks that we know of for BFGW + KZG. If the Pedersen variant of KZG commitments is used, BFGW + KZG is compatible with Pedersen commitments. BFGW can also be instantiated with DARKs [BFS20b], which do not require a trusted setup. Both BFGW + KZG and BFGW + DARKs are aggregatable.

6.2 Lattice- and code-based constructions

There are several lattice- and code- based zero knowledge range proof schemes. These schemes have the advantages that they are plausibly post-quantum secure and have a transparent setup. However, they are concretely much less efficient than the discrete logarithm-based schemes such as Bulletproofs. In particular, they have very long proofs. Thus, one worthwhile research direction is to improve the efficiency of these lattice-based protocols. One area for improvement is in the repetition required to achieve negligible soundness error. Most of these schemes build on protocols with constant soundness and must repeat the protocol $\Omega(\lambda)$ times to achieve λ bits of security. When made non-interactive, this amplification results in large proofs.

Lattice- and code-based schemes typically use the binary decomposition approach, where the prover already holds a commitment to the bits $b_0, \ldots, b_{k-1}$ of the value in question. The prover wants to show that $\sum_{i=0}^{k-1} 2^i \cdot b_i \leq \beta$ for some β. This condition can be written equivalently as a system of equations over the bits modulo 2. Such systems of equations can be proven in zero-knowledge using *Stern-like* protocols.

In this section, we present several ideas involved in lattice-based schemes. We first present a lattice-based commitment scheme, KTX [KTX08], that is used in some of these ZKRPs. In doing so, we emphasize several challenges common to many lattice-based schemes. We then give a high-level description of Stern-like protocols, a standard technique for lattice-based zero-knowledge proofs.

KTX commitment scheme ([KTX08]). The KTX commitment scheme is based on the hardness of the Short Integer Solution (SIS) problem. Let λ be the security parameter, L be the number of bits to be committed to, and q be a prime modulus of size $O(\lambda\sqrt{L})$. Let $m = 2\lambda\lceil \log q \rceil$. The scheme uses public parameters $(\mathbf{A}, \mathbf{B})$ chosen uniformly from $\mathbb{Z}_q^{\lambda \times L} \times \mathbb{Z}_q^{\lambda \times m}$. The commitment to a bit vector $\mathbf{x} \in \{0,1\}^L$ is the vector

$$\mathbf{c} = \mathbf{A} \cdot \mathbf{x} + \mathbf{B} \cdot \mathbf{r} \pmod{q}$$

where $\mathbf{r}$ is sampled uniformly from $\{0,1\}^m$. This scheme is statistically hiding and computationally binding assuming that the public parameters are sampled uniformly.

Note that KTX commitments are only approximately homomorphic. While it is the case that

$$\mathbf{A} \cdot \mathbf{x_1} + \mathbf{B} \cdot \mathbf{r_1} + \mathbf{A} \cdot \mathbf{x_2} + \mathbf{B} \cdot \mathbf{r_2} = \mathbf{A}(\mathbf{x_1} + \mathbf{x_2}) + \mathbf{B}(\mathbf{r_1} + \mathbf{r_2}) \pmod{q},$$

note that $(\mathbf{x_1} + \mathbf{x_2})$ and $(\mathbf{r_1} + \mathbf{r_2})$ may not be 0/1 vectors. Therefore, $A(x_1 + x_2) + B(r_1 + r_2)$ is not necessarily a valid commitment; in particular, it is not clear how to provide an opening proof. Many commitment schemes used by schemes in this section have similar limited homomorphism.

Note also that KTX commitments do not require a trusted setup to generate the public parameters $\mathbf{A}$, $\mathbf{B}$, and q, as these matrices are uniformly random and q can be publicly known. Many lattice-based commitment schemes similarly use random matrices as the public parameters. All of the range proofs in this section offer transparent setup.

Stern-like protocols. Stern's original protocol [Ste96] proves in zero knowledge that a committed bit vector has a certain Hamming weight; that is, it is a zero-knowledge argument of knowledge for the following relation:

$$\{((\mathbf{H}, \mathbf{y}), \mathbf{s}) \in \mathbb{Z}_2^{n \times m} \times \mathbb{Z}_2^n \times \mathbb{Z}_2^m : (\mathrm{wt}(\mathbf{s}) = w) \wedge (\mathbf{H} \cdot \mathbf{x} = \mathbf{y})\}$$

The key idea behind Stern's protocol is that the prover permutes the bits of $\mathbf{s}$ to obtain $\mathbf{s}'$ which it reveals to the verifier. It also convinces the verifier that $\mathbf{s}'$ is indeed a permutation of $\mathbf{s}$ under some π. $\mathbf{s}'$ has the same Hamming weight as $\mathbf{s}$, and the distribution of $\mathbf{s}'$ is identical for any $\mathbf{s}$ satisfying the relation—therefore, $\mathbf{s}'$ reveals no information about $\mathbf{s}$. At a high level, the prover samples a random blinding factor $\mathbf{r}$ and constructs three commitments, which it sends to the verifier, as follows:

$$c_1 = \mathrm{Com}(\pi, \mathbf{H} \cdot \mathbf{r})$$
$$c_2 = \mathrm{Com}(\pi(\mathbf{r}))$$
$$c_3 = \mathrm{Com}(\pi(\mathbf{r} \oplus \mathbf{s}))$$

Here, $\pi(\mathbf{v})$ denotes the vector obtained by permuting the components of $\mathbf{v}$ under π. We now run one of three randomized checks: the verifier sends the prover $b \in \{0,1,2\}$. In each of these tests, the prover opens a different combination of the commitments and sends some additional information, e.g., $\pi(\mathbf{s})$ for $b = 2$. The cheating prover cannot pass all of these tests simultaneously and therefore fails with probability at least 1/3. Note that running all of these tests at once would reveal information about $\mathbf{s}$.

This permute-then-reveal strategy can be used for other relations with similar properties. [NTWZ19] provides an abstraction of such relations, in terms of some set VALID, which in Stern's original protocol was $\mathrm{VALID} = \{\mathbf{s} : \mathrm{wt}(\mathbf{s}) = \mathrm{w}\}$:

$$R = \{((\mathbf{H}, \mathbf{y}), \mathbf{s}) \in \mathbb{Z}_2^{n \times m} \times \mathbb{Z}_2^n \times \mathrm{VALID} : \mathbf{H} \cdot \mathbf{x} = \mathbf{y}\}$$

Correctness under permutation: For all $((\mathbf{H}, \mathbf{y}), \mathbf{s}) \in \mathbb{Z}_2^{n \times m} \times \mathbb{Z}_2^n \times \mathbb{Z}_2^m$ and all permutations π over $[m]$,

$$\mathbf{s} \in \mathrm{VALID} \iff \pi(\mathbf{s}) \in \mathrm{VALID}$$

Hiding under permutation: For all $\mathbf{s} \in \mathrm{VALID}$, the distribution of $\pi(\mathbf{s})$ where π is a random permutation over $[m]$ is uniform over the set VALID

[1] https://blog.ethereum.org/2023/01/16/announcing-kzg-ceremony

Bulletproofs-Style Range Proofs (all DLOG-based, all aggregatable)		
Scheme	Commitment Scheme	Transparent Setup
Bulletproofs [BBB+18]	Pedersen	Y
Bulletproofs+ [CHJ+22]	Pedersen	Y
Bulletproofs++ [Eag22]	Pedersen	Y
Flashproofs [WC22]	Pedersen	Y
SwiftRange [WCL23]	Pedersen	Y
DRZ [DRZ20]	Pedersen	N
ZZT+ [ZZT+23]	Pedersen	N
Libert [Lib23]	Pedersen	N
BFGW [BFGW20] + KZG_{Ped}	Pedersen	N
BFGW [BFGW20] + DARKs [BFS20b]	DARK [BFS20b]	Y with class groups; N with RSA

Figure 2: Properties of Bulletproofs-style range proofs

Even given a relation that does not fit the above requirements, one can sometimes construct an associated relation (e.g., using a common technique called *extension*) that does fall into this paradigm and allows one to construct the desired argument.

Other relations that can be proven under Stern's paradigm include proving knowledge of one secret bit that may appear in multiple equations [LLNW16], or proving the knowledge of the product of two secret bits [LLM+16]. Stern-like techniques underlie the majority of lattice- and code-based zero knowledge protocols. However, recall that due to the randomized tests, Stern's original protocol has soundness error 2/3. In general, Stern-like protocols have constant soundness error and thus require roughly λ repetitions for λ bits of security. Thus, once made non-interactive via Fiat-Shamir, these protocols result in long proofs.

Only recently have techniques emerged for avoiding Stern-like protocols in constructing lattice-based ZKRPs, whose state-of-the-art is thus not reflected in the previous ZKRP survey [MKvWK19]. These new techniques have resulted in a surge of lattice-based constructions with greatly improved efficiency, with proofs on the order of 10,000 KB rather than 100,000 KB. However, this efficiency still lags behind many non-lattice-based constructions with $\tilde{5}00$-byte proofs, as seen in Figure 5. Improving lattice-based schemes remains a fruitful research direction.

[ESLL19] proposes techniques for avoiding the repetition that Stern-like protocols require for soundness. Their *one-shot* protocol saves a factor of λ computation time over repeated Stern-like protocols, though the proofs are still quite long as shown in Figure 5. One-shot approaches are a fruitful direction for developing a more practical (in terms of both communication and computation) lattice-based ZKRP.

ALS [ALS20] uses an inner product argument in the n-ary decomposition approach, which results in significantly shorter proofs compared to other lattice-based constructions; see Figure 5. Its proofs are roughly an order of magnitude larger than those of the most efficient non-lattice schemes, such as Bulletproofs. Another barrier to practical efficiency is that the proofs of ALS cannot be aggregated.

7 HASH CHAIN CONSTRUCTIONS

Payword [RS96] was the first to use hash chains to construct a range proof for electronic payments, and HashWires [CCL+21] more recently revisited this idea with great efficiency improvements. In this approach, the core idea is that a commitment C_x to a value x is the output of a hash function evaluated x times on a random value. That is, $C_x = H^x(r)$ for a random r. The proof that x is at least some threshold t is a value $\pi = H^{x-t}(r)$ such that applying the hash function t more times to π yields C_x; that is, $H^t(\pi) = C_x$. Since the hash function is hard to invert, if $x - t$ is negative it should be hard for the prover to find an accepting π. Importantly, though, C_x must be well-formed to ensure soundness. Thus, the setting where hash chain constructions can be used is slightly more restricted.

HashWires [CCL+21] defines a relaxation of zero-knowledge range proofs called *credential-based range proofs*. This notion is weaker than general ZKRPs in that the commitment is assumed to be well-formed. Soundness is shown only under this assumption, which is motivated by a setting where a trusted authority distributes commitments to parties that later prove that their committed values exceed some threshold. For example, the trusted authority may be a government, and the commitments might be used for credentials including citizens' ages. If a commitment is signed by this trusted authority, a verifier can be confident that the commitment is properly formed.

As described, the time to generate π and C_x is linear in x, and the verifier time is linear in t. This is very expensive if we wish to prove that x is in some large range $[0, 2^k - 1]$; ideally, these costs should grow at most linearly with k. HashWires achieves this by observing that x can be written in some base u, and the proof can be broken into several sub-chains to greatly improve this efficiency (they called this a *minimum dominating partition*). This base can be chosen to trade off between proof size and prover/verifier efficiency. In our later discussion of efficiency, we include benchmarks for a variety of bases. We will see in Section 9 that HashWires is extremely concretely efficient, in terms of both verifier time and prover time. Its proof sizes are also competitive with other constructions.

8 CHOOSING THE CONSTRUCTION FAMILY FOR YOUR APPLICATION

As there are dozens of ZKRP constructions, choosing the appropriate scheme for a particular application can be challenging. In Figure 4, we give a flowchart for narrowing down the class of range proofs depending on constraints. The next section gives an efficiency comparison to help choose a scheme within this class.

Lattice- and Code-Based Range Proofs			
Scheme	Commitment Scheme	Assumptions	Transparent Setup
LLNW [LLNW18]	KTX	SIVP	Y
ESLL [ESLL19]	UMC, HMC [BKLP15, BDL+18, ESS+19]	Module-SIS, Module-LWE	Y
YAZ+ [YAZ+19]	KTX	LWE, SIS	Y
ALS [ALS20]	[BDL+18]	Module-SIS, Module-LWE	Y
CKLR [CKLR21]†	[BDL+18], as modified by [YAZ+19]	LWE, SIS	Y
LNS [LNS20]*	[BDL+18]	Module-SIS, Module-LWE	Y
LNP [LNP22]**	ABDLOP [Ajt96, BDL+18]	Module-SIS, Module-LWE	Y
Code-based [NTWZ19]	[NTWZ19]	2-RNSD	Y

Figure 3: Properties of lattice- and code-based range proofs

†CKLR [CKLR21] uses the square decomposition approach, but one of their constructions is lattice-based.
*In addition to their standard range proof, LNS [LNS20] also constructs an *approximate* range proof, showing that $z \in [0, n \cdot 2^k - 1]$ for some small n. While relaxed, this kind of approximate range proof is sufficient for showing smallness of vectors, which is an application they target. Its efficiency does not depend on k.
**LNP [LNP22] is an *approximate* range proof. For a vector $\mathbf{s}$ such that $||\mathbf{s}||$ is much smaller than some bound B, this proof can be used to show that $||\mathbf{s}|| \leq B$.

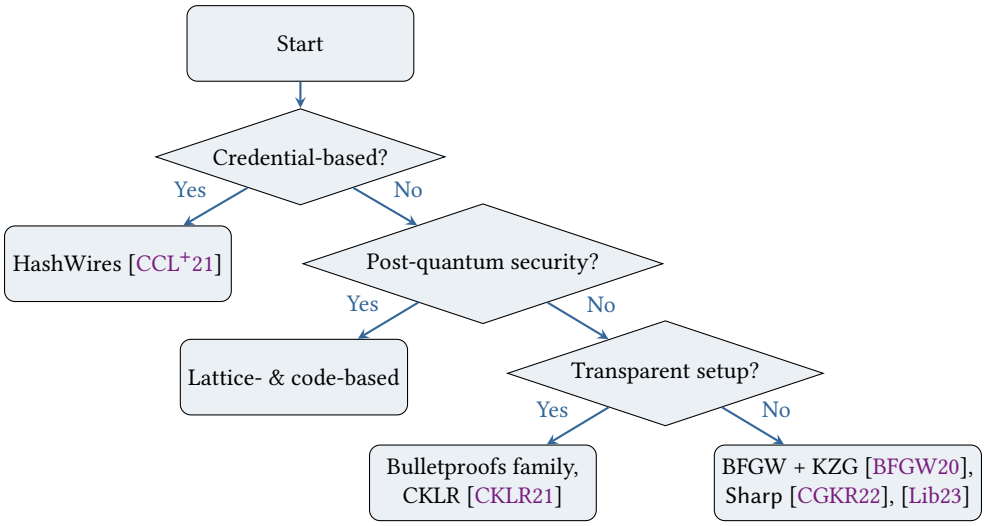

Figure 4: Flowchart for choosing a range proof based on desired properties.

HashWires [CCL+21] are concretely quite efficient and use only hash functions; thus, they are plausibly post-quantum secure and do not require a trusted setup. However, they're in a more stringent trust model (they are *credential-based range proofs*), where there is a trusted issuer distributing commitments; that is, soundness holds only if the commitment is well-formed. If the desired use case does have this trusted issuer, HashWires is likely the most efficient scheme.

Among the remaining constructions, only the lattice-and code-based constructions are plausibly post-quantum secure, and thus if this is a requirement this class is the only option. These schemes have relatively large proof sizes (on the order of 10KB).

If trusted setup is allowed, there are several schemes with very short proofs and efficient verifier and prover. BFGW + KZG [BFGW20], Sharp [CGKR22], and Libert's DLOG-based scheme [Lib23] all have constant-sized proofs.

If trusted setup is undesired, the Bulletproofs family is recommended. Although many lattice- and code-based constructions do not require a trusted setup, all Bulletproofs-style constructions have much shorter proofs. Even if a trusted setup is allowed, Bulletproofs-style constructions may still be worth considering depending on how much one values short proofs. Though their proof sizes are not constant, they seem to be the most commonly used in practice. We list CKLR [CKLR21] as well because it has comparable efficiency to Bulletproofs on paper and also does not require trusted setup. However, it has several drawbacks: it does not allow batching, it is less efficient in practice due to its incompatibility with optimized libraries for common elliptic curves, and it offers a more relaxed notion of security. For certain applications where these drawbacks are less important, CKLR may be worth considering.

9 EFFICIENCY COMPARISON

This section includes an efficiency comparison of various ZKRPs. In Figure 5, we compile both concrete and asymptotic proof sizes for schemes of particular interest. The concrete proof sizes have been extracted directly from the schemes' respective papers, as the proof sizes are largely the same across machine configurations. Groth16 has the shortest range proofs for a 64-bit range at 192 bytes whereas HashWires has the shortest range proofs at 177 bytes for a 32-bit range.

In Figure 6, we record prover and verifier times for various schemes. We add many of our own benchmarks to ensure that

configurations are standardized. In particular, we add a benchmark for Groth16 [Gro16] that was absent in prior work. The configurations for benchmarks that we pull from other papers are noted below.

Other benchmarks. The Sharp paper's [CGKR22] benchmark of their scheme, Sharp, was run on a MacBook Pro with a 2.3 GHz Intel core i7 processor and uses the library libsecp256k1 [Wui18]. The HashWires paper [CCL+21] includes a benchmark for Bulletproofs which is significantly faster than ours. They used an AVX2 backend was used which significantly speeds up curve arithmetic. We include this benchmark in addition to ours, to reflect the speedup possible with their configuration.

Our benchmarks. We add our own benchmarks for Hashwires (base 16 and base 256), Bulletproofs, BFGW + KZG, and Groth16. In all cases, we record the median running time over 100 runs. We plan to open source all of our benchmarks for reproducibility.

For Groth16, we implement range proofs with two versions of the commitment scheme: the well-established Pedersen commitments and the new zk-friendly Poseidon commitments. We've used Circom [BMIMT+22] for writing circuits and rapidsnark [ide23] for generating and verifying the Groth16 proofs.

The implementations for Hashwires, Bulletproofs and BFGW + KZG are in Rust. All the benchmarks were run on a AMD EPYC 7443P 24-Core with 512GB of RAM (a c3.large.x86 machine hosted by latitude.sh). All implementations are in Rust and open sourced at https://github.com/joyqvq/range-proof-benches with various attributions to original implementations. We explicitly chose a non-Mac machine because rapidsnark leverages Intel Assembly to speed up Groth16 proof generation.

Hashwires has the fastest proof generation and verification times. Both BFGW + KZG and Groth16 have constant-sized proofs but they are less computationally efficient than others. Groth16 has the longest proof generation times. This is expected because we are instantiating range proofs within a general-purpose zk proof system.

It is worth noting that in practice the availability of a reliable library may outweigh mild efficiency gains. Bulletproofs is the most widely used range proof in practice and is likely a good choice. Groth16, though not tailored to range proofs, is one of the most popular general-purpose zero-knowledge proof systems and offers several well supported libraries; we use Circom [BMIMT+22] and rapidsnark [ide23]. From our benchmarks, one can see the efficiency gains offered by tailored range proof solutions over generic solutions, which can be seen especially in the long prover times required for Groth16 relative to the other range proofs.

10 RESEARCH GAPS

Research Gap 1. *Practical transparent constant-sized range proofs.*

No zero-knowledge range proofs are practical, transparent, and have constant-sized proofs. Bulletproofs and its close relatives have transparent setup but have proofs of size $O(\log k)$ for a k-bit range. BFGW + KZG has constant-sized proofs but requires a trusted setup; BFGW + DARKs has a transparent setup but requires $O(\log k)$-sized proofs. CKLR has a transparent setup and has constant-sized proofs but achieves only a relaxed notion of soundness. Furthermore, its proofs are not as practically efficient as the above schemes because they use less common curves that optimized libraries do not support.

Research Gap 2. *Shorter (even amortized) lattice- or code-based ZKRPs.*

The proofs of lattice-based and code-based ZKRPs are concretely quite long, as shown in Figure 5. For blockchain applications where one must pay for the space used on-chain, this length is problematic, especially as these constructions do not support aggregation. In order to be competitive with constructions using other techniques shown in Figure 5, the proof size must be under 1 KB.

Research Gap 3. *Lattice- or code-based ZKRPs with multi-prover aggregation.*

Lattice-based ZKRPs with short proofs are desirable for confidential transactions, as blockchains transition to post-quantum security. In such settings, this size issue may be mitigated by multi-prover aggregation. Each block would then contain only an aggregate range proof for all included transactions. However, this aggregation must be multi-prover as these transactions may be made by many different parties, each holding commitments to private values. Lattice- and code-based ZKRPs with multi-prover aggregation have not yet been constructed, leading us to the this related research gap.

Research Gap 4. *Un-replayable credential-based range proofs.*

For credential applications, one might want an interactive range proof that cannot be replayed. Suppose that Alice has a commitment of her age signed by a trusted credential issuer. Alice should be able to visit the DMV and prove in zero knowledge that her committed age is above 16. An observer Bob should not be able to copy Alice's commitment and re-use the transcript of the protocol to prove (possibly falsely) that his age is above 16. If this range proof is non-interactive, Bob can simply copy the proof and re-use it. This re-use might be avoided if the protocol is public-coin interactive, and the DMV issues a random challenge that requires knowledge of the committed value to respond to.

Can we make hash-chain-based range proofs that are un-replayable in this way? As credentials are a primary motivation for HashWires, un-replayability would be a nice property to add.

Research Gap 5. *Integer commitments with full soundness with transparent setup.*

CKLR [CKLR21] and Sharp [CGKR22] construct integer commitments with a relaxed notion of soundness. In order to be used for confidential transactions, they must be augmented with additional proof elements from an RSA group or class group. The RSA version requires a trusted setup, and the class group solution is not compatible with existing optimized libraries. Rather than patching soundness issues by adding these extra elements, it would be preferred to construct practically efficient integer commitments with full soundness and transparent setup.

Research Gap 6. *Efficient post-quantum ZKRPs compatible with LWE-based ciphertexts.*

Zero-knowledge range proofs can be used to build verifiable LWE-based encryption schemes as discussed in Appendix 11.2.

Scheme	Proof size (bytes)		Proof size (asymptotic)
	64-bit range	32-bit range	k-bit range
Bulletproofs	675	610	$O(\log k)$
BFGW + KZG †	576	576	$O(1)$
Sharp_{GS}	360	318	$O(1)$
$\text{Sharp}^{\text{Po}}_{\text{SO}}$	389	335	$O(1)$
$\text{Sharp}_{\text{RSA}}$	793	751	$O(1)$
HashWires (Base 16)†	263	231	$O(\log k)$
HashWires (Base 256)†	199	167	$O(\log k)$
Groth16 [Gro16]§	192	192	$O(1)$
Lattice-based ALS [ALS20]**	-	5,900	$O(k)$
Lattice-based ESLL [ESLL19]	93,000	58,000	$\Omega(k)$*
Lattice-based LNS [LNS20]**	-	11,800	$o(k)$‡

Figure 5: Proof sizes in bytes for 64- and 32-bit ranges. The benchmark for each of these schemes is from that scheme's original paper, except where otherwise noted.

† Our own benchmark.
§ Benchmark from HashWires [CCL+21], over the BLS12-381 curve.
*See [ESLL19] for the exact expression, which includes several other parameters not described here. It is $\Omega(k)$ and is large relative to the other schemes.
**The proof sizes for 64-bit ranges were not included in [ALS20, LNS20]. Note that [ALS20] has linear growth, so extrapolating from its 5,900-bit proof for 32-bit ranges, its proof for 64-bit ranges would be large.
‡ See [LNS20] for the exact expression, which is complicated; it is sublinear in k.

Scheme	Verifier Time (ms)		Prover Time (ms)	
	64-bit range	32-bit range	64-bit range	32-bit range
Bulletproofs†	2.51	1.37	11.96	6.32
$\text{Sharp}^{\text{Po}}_{\text{SO}}$	0.75	0.74	1.17	0.97
Bulletproofs AVX2 (HashWires benchmark)	0.938	-	6.516	-
HashWires base 16†	0.002	0.002	0.061	0.003
HashWires base 256†	0.01	0.009	0.194	0.083
BFGW + KZG†	5.682	5.653	12.569	9.572
Groth16-Poseidon†	4	4	34.46	34.23
Groth16-Pedersen†	4	4	33.57	31.18

Figure 6: Verifier and prover times. See implementations: https://github.com/joyqvq/range-proof-benches

†Our own benchmark.

However, existing verifiable LWE-based encryption schemes constructed using ZKRPs [DPLS19, Lib23] use discrete logarithm-based ZKRPs. Thus, while they obtain privacy against quantum adversaries due to the LWE-based encryption used, they lack soundness in verification due to the DLOG-based range proofs. If there were efficient post-quantum range proofs compatible with LWE-based ciphertexts, one could obtain verifiable encryption with soundness against quantum adversaries as well. While a lattice-based zk-SNARK (e.g., [ACL+22]) may work in theory, it may not be efficient (yielding long ciphertexts and heavy computation). An efficient lattice-based ZKRP that is compatible with lattice-based encryption would be more satisfactory.

11 APPLICATIONS

11.1 Practical applications

Confidential transactions. Confidential transactions are an application of range proofs that's especially relevant to cryptocurrencies, and have spurred research yielding significant efficiency improvements for ZKRPs in recent years. In most cryptocurrencies, such as Bitcoin and Ethereum, transaction details are visible to everyone; in particular, anyone with access to the underlying blockchain can see the amounts of currency being transferred.

Confidential transactions, initially proposed by Maxwell [Max16], explored further in [BAZB20, Poe16], and deployed in applications

such as Monero[2], aim to hide the amounts involved in each transaction. In Maxwell's approach, these amounts are stored using Pedersen commitments [Ped91], and the sender must prove to the miners that the sum of the output amounts does not exceed the input amount. In other words, if amt_{in} is the input amount and each $(\mathsf{amt}_{out})_i$ is an output amount,

$$\mathsf{amt}_{in} - \sum_i (\mathsf{amt}_{out})_i \geq 0$$

However, this check on its own is not enough to ensure security. A malicious sender Eve could create coins by creating a transaction with herself as the recipient where, for example,

$$\mathsf{amt}_{in} = 0, (\mathsf{amt}_{out})_1 = -1, (\mathsf{amt}_{out})_2 = 1$$

Since $\mathsf{amt}_{in} = 0$, this transaction does not require Eve to spend any coins. However, since Eve receives the output $(\mathsf{amt}_{out})_2 = 1$, she gains a coin. Yet this transaction satisfies the check above. Thus, confidential transactions also require the sender to prove that each output amount is positive: $(\mathsf{amt}_{out})_i \geq 0$ for each i.

Zero knowledge range proofs give us a way to prove that a committed (or encrypted) value x lies in a range $[0, 2^k - 1]$ without revealing any other information about x. Thus, when applied to confidential transactions, they allow the sender to show exactly these checks without compromising confidentiality.

Receiver hiding. In addition to hiding the amounts involved in transactions, confidential transactions (enabled by range proofs) allow the sender to hide the identity of the receiver of a transaction. If party A wants to send x amount to party B, it can create a transaction that sends a confidential amount to n different users $B_1, \ldots, B_n$, one of which is actually B. The amounts sent to $B_i \neq B$ are zero, and the amount sent to B is x. Because the transaction is confidential, an outside observer cannot tell which recipient A actually paid.

However, later transactions may suggest which B_i received nonzero currency from A, since $B_i \neq B$ will likely never spend their amounts of zero, which are useless. Thus, an observer can infer that the party that spends its received amount is likely B. One way to mitigate this issue is for A to send a very small amount to each $B_i \neq B$, to incentivize them to use this amount in a future transaction. This introduces a trade-off between cost and the size of the anonymity set.

Proofs of liabilities, reserves, and solvency. Proofs of solvency allow exchanges or banks to show that they hold enough currency in reserves to pay out all customers. There are two aspects to this: proving that the bank holds at least a certain amount in reserves, and proving that the bank holds at most that amount in liabilities. These proofs should be privacy-preserving in that they don't reveal users' balances. Zero knowledge range proofs have been used in many schemes such as Provisions [DBB+15, Cam14, JC21].

Private voting. Consider the setting where users send encrypted votes to a group of authorities that may jointly decrypt the sums of the votes but should not be able to learn any individual user's vote. A challenge here is that users' votes should be kept private, yet dishonest users should not be able to submit negative votes for candidates that they dislike. [Gro05] constructs several protocols in this setting and mitigates this issue by requiring users to provide zero-knowledge range proofs of their votes' validity.

Federated learning. In federated learning, many distributed clients train machine learning models on locally held data. These models are then aggregated into a global model, often by averaging their weights. To protect the privacy of clients' data, these weights are often submitted under encryption. One concern in this setting is that a malicious client may submit an encrypted model with very large weights, which could hurt the accuracy of the global model by skewing this average. One way to mitigate this problem is to use input validation to ensure that each client's model has bounded weights. Range proofs, used for federated learning by Acorn [BGL+23], enable input validation even over encrypted weights, where clients can show that their inputs satisfy these bounds without revealing their models in the clear.

Auctions. [AW13] uses range proofs for verifiable auctions. Range proofs help an auctioneer to prove to an auditor that the sale price was set correctly without revealing the values of the bids. For example, in a second-price auction with n bids, (where the sale price is equal to the second-highest bid), the auctioneer can provide proofs that $n - 1$ bids were at most the sale price and one bid was greater than the sale price.

Anonymous credentials. In an anonymous credential system, users are issued credentials that they can later use to prove facts about their attributes, such as their age. Ideally, these proofs should reveal no more than is necessary. For example, a user should be able to prove that their age is at least 18 without revealing their exact age. Here, range proofs can be used; this was a motivation for [CCS08, CCL+21]. An additional desirable property motivated by this setting is *unlinkability*, where one cannot tell that two such proofs are for the same commitment.

Verified location. Range proofs can be used to show that a location is in a permitted region, by proving that the latitude and longitude lie in the proper intervals. This and the following application, timestamping, are suggested applications of HashWires [CCL+21].

Timestamping. Suppose one is issued a certificate with a secret expiration date. Range proofs can be used to privately show that the certificate is still valid; i.e., the current date is less than the expiration date.

Certificate transparency. Certificate transparency helps identify when certificate authorities misbehave. This can be done using a public transparency log, allowing all certificates to be publicly viewable. One feature put forth in [EMBB17] is a way for auditors to prove when a certificate has been incorrectly omitted from the log, without revealing which certificate has been excluded for privacy reasons. [EMBB17] uses ZKRPs to construct these zero-knowledge non-inclusion proofs.

Differential privacy. In some applications, encrypted consumer data is aggregated to support statistical studies. For example, encrypted electricity consumption data [MR14]. To enable privacy of individual users, [MR14] added noise for differential privacy. A

[2] https://www.getmonero.org/

ZKRP is used to prove that the added noise is within an accepted range.

11.2 Cryptographic applications

Non-membership proofs. Zero knowledge range proofs can be used to construct zero knowledge proofs of non-membership. In other words, one can prove that a committed value x does not belong to some public set S, without revealing any additional information about the committed value. The approach to doing so is to construct a Merkle tree containing elements of S as leaves in sorted order (if S is not a set over the integers, assume some known mapping from the data universe to integers). A non-membership proof for x is a zero knowledge proof that one knows leaves s_i and s_j such that s_i and s_j are adjacent and $s_i < x < s_j$. It is convenient to use a range proof for this second property.

[EMBB17] constructs non-membership proofs from zero knowledge range proofs in this fashion and uses them for certificate transparency, to allow an auditor to prove in zero knowledge that a valid certificate is not included in the log. [LLNW18] combines range proofs with techniques from [LLNW16] to construct lattice-based non-membership proofs as a generic primitive.

Well-formedness of LWE-based ciphertexts. When computing a function over multiple (potentially untrusted) parties' encrypted inputs, it is important to ensure that these ciphertexts are well formed. FHE schemes based on Ring LWE (RLWE) are often used in such settings [BGV14]. In such schemes, ciphertexts $\mathbf{t}$ take the form $\mathbf{t} = \mathbf{As}$ (over a ring), where $\mathbf{A}$ is a matrix representing the public key and $\mathbf{s}$ is a vector representing the message and randomness used in the encryption. This encryption is correct if and only if all entries of $\mathbf{s}$ are bounded. Zero knowledge range proofs have a natural application here: they can be used to show exactly this boundedness without revealing any information about the underlying plaintext.

[DPLS19] constructs efficient proofs of well-formedness for RLWE ciphertexts by committing to the components of $\mathbf{s}$ using a Pedersen commitment and using Bulletproofs [BBB+18] to prove that these components are in the desired range. [Lib23] follows a similar strategy but replaces the use of Bulletproofs with a more efficient zero knowledge range proof.

Publicly verifiable secret sharing. Publicly verifiable secret sharing allows an untrusted dealer to share a secret in a way that any other party can verify that the secret was properly shared. If the dealer encrypts the shares for the recipients, there must be a way to verify that these recipients can indeed decrypt the ciphertexts to learn well-formed shares.

[Gro21] constructs a PVSS scheme using an encryption scheme which requires solving a discrete logarithm problem for decryption. To make this feasible, it uses chunked encryption, which breaks the plaintext into smaller chunks that allow this discrete logarithm problem to be solved efficiently. It uses range proofs to show that the chunks are indeed small enough. In another PVSS scheme, [GHL22] uses Bulletproofs to construct proofs of correct encryption/decryption of LWE-based PVW ciphertexts [PVW08].

Polynomial relations over the integers. [CM99, CBM99] shows how zero-knowledge range proofs can be used to construct zero-knowledge proofs of polynomial relations among the discrete logs of given elements, even if these elements lie in different groups. This should be reminiscent of integer commitments, which let us prove relations about committed values over the integers, as is useful in the four-square decomposition range proof constructions. Interestingly, range proofs help achieve this functionality as well. More precisely, let $\mathbb{G}_1$ and $\mathbb{G}_2$ be two such groups with generators g_1 and g_2 respectively, and suppose we want to prove equality of discrete logarithms of elements in these two groups. That is, for the simple equality relation, given $y_1 \in \mathbb{G}_1$ and $y_2 \in \mathbb{G}_2$, the prover can convince the verifier that it knows α such that $y_1 = g_1^\alpha$ in $\mathbb{G}_1$ and $y_2 = g_2^\alpha$ in $\mathbb{G}_2$. Showing also that α lies in $-2^\ell < \alpha < 2^\ell$ for appropriately chosen ℓ suffices to show that $\log_{g_1} y_1 = \log_{g_2} y_2$ over $\mathbb{Z}$. The range proof used here serves to show that α does not wrap around either group. The construction used in [CBM99] uses this idea but is much more efficient than combining these proofs naively.

REFERENCES

[ACL+22] Martin R Albrecht, Valerio Cini, Russell WF Lai, Giulio Malavolta, and Sri AravindaKrishnan Thyagarajan. Lattice-based snarks: Publicly verifiable, preprocessing, and recursively composable. In *Annual International Cryptology Conference*, pages 102–132. Springer, 2022.

[Ajt96] Miklós Ajtai. Generating hard instances of lattice problems. In *Proceedings of the twenty-eighth annual ACM symposium on Theory of computing*, pages 99–108, 1996.

[ALS20] Thomas Attema, Vadim Lyubashevsky, and Gregor Seiler. Practical product proofs for lattice commitments. In *Annual International Cryptology Conference*, pages 470–499. Springer, 2020.

[AW13] Sebastian Angel and Michael Walfish. Verifiable auctions for online ad exchanges. In *Proceedings of the ACM SIGCOMM 2013 conference on SIGCOMM*, pages 195–206, 2013.

[BAZB20] Benedikt Bünz, Shashank Agrawal, Mahdi Zamani, and Dan Boneh. Zether: Towards privacy in a smart contract world. In *International Conference on Financial Cryptography and Data Security*, pages 423–443. Springer, 2020.

[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 (SP)*, pages 315–334. IEEE, 2018.

[BCC+09] Mira Belenkiy, Jan Camenisch, Melissa Chase, Markulf Kohlweiss, Anna Lysyanskaya, and Hovav Shacham. Randomizable proofs and delegatable anonymous credentials. In Shai Halevi, editor, *Advances in Cryptology - CRYPTO 2009, 29th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 16-20, 2009. Proceedings*, volume 5677 of *Lecture Notes in Computer Science*, pages 108–125. Springer, 2009.

[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 *Advances in Cryptology–EUROCRYPT 2016: 35th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Vienna, Austria, May 8-12, 2016, Proceedings, Part II 35*, pages 327–357. Springer, 2016.

[BCDvdG88] Ernest F Brickell, David Chaum, Ivan B Damgård, and Jeroen van de Graaf. Gradual and verifiable release of a secret. In *Advances in Cryptology—CRYPTO'87: Proceedings 7*, pages 156–166. Springer, 1988.

[BDL+18] Carsten Baum, Ivan Damgård, Vadim Lyubashevsky, Sabine Oechsner, and Chris Peikert. More efficient commitments from structured lattice assumptions. In *International Conference on Security and Cryptography for Networks*, pages 368–385. Springer, 2018.

[BFGW20] D Boneh, B Fisch, A Gabizon, and Z Williamson. A simple range proof from polynomial commitments, 2020. https://hackmd.io/@dabo/B1U4kx8XI.

[BFM88] Manuel Blum, Paul Feldman, and Silvio Micali. Non-interactive zero-knowledge and its applications. In *Proceedings of the twentieth annual ACM symposium on Theory of computing*, pages 103–112, 1988.

[BFS20a] Benedikt Bünz, Ben Fisch, and Alan Szepieniec. Transparent snarks from dark compilers. In *Advances in Cryptology–EUROCRYPT 2020: 39th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Zagreb, Croatia, May 10–14, 2020, Proceedings, Part I 39*, pages 677–706. Springer, 2020.

[BFS20b] Benedikt Bünz, Ben Fisch, and Alan Szepieniec. Transparent snarks from dark compilers. In *Advances in Cryptology–EUROCRYPT 2020:*

39th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Zagreb, Croatia, May 10–14, 2020, Proceedings, Part I 39, pages 677–706. Springer, 2020.

[BGL+23] James Bell, Adrià Gascón, Tancrède Lepoint, Baiyu Li, Sarah Meiklejohn, Mariana Raykova, and Cathie Yun. {ACORN}: Input validation for secure aggregation. In *32nd USENIX Security Symposium (USENIX Security 23)*, pages 4805–4822, 2023.

[BGV14] Zvika Brakerski, Craig Gentry, and Vinod Vaikuntanathan. (leveled) fully homomorphic encryption without bootstrapping. *ACM Transactions on Computation Theory (TOCT)*, 6(3):1–36, 2014.

[BKLP15] Fabrice Benhamouda, Stephan Krenn, Vadim Lyubashevsky, and Krzysztof Pietrzak. Efficient zero-knowledge proofs for commitments from learning with errors over rings. In *European symposium on research in computer security*, pages 305–325. Springer, 2015.

[BMIMT+22] Marta Bellés-Muñoz, Miguel Isabel, Jose Luis Muñoz-Tapia, Albert Rubio, and Jordi Baylina. Circom: A circuit description language for building zero-knowledge applications. *IEEE Transactions on Dependable and Secure Computing*, 2022.

[Bou00] Fabrice Boudot. Efficient proofs that a committed number lies in an interval. In *International Conference on the Theory and Applications of Cryptographic Techniques*, pages 431–444. Springer, 2000.

[BSBHR18] Eli Ben-Sasson, Iddo Bentov, Yinon Horesh, and Michael Riabzev. Scalable, transparent, and post-quantum secure computational integrity. *Cryptology ePrint Archive*, 2018.

[Cam14] Philippe Camacho. Secure protocols for provable security, 2014. https://www.slideshare.net/philippecamacho/protocols-for-provable-solvency-38501620.

[CB21] Panagiotis Chatzigiannis and Foteini Baldimtsi. Miniledger: Compact-sized anonymous and auditable distributed payments. In Elisa Bertino, Haya Shulman, and Michael Waidner, editors, *Computer Security - ESORICS 2021 - 26th European Symposium on Research in Computer Security, Darmstadt, Germany, October 4-8, 2021, Proceedings, Part I*, volume 12972 of *Lecture Notes in Computer Science*, pages 407–429. Springer, 2021.

[CBM99] Jan Camenisch, BRICS, and Markus Michels. Separability and efficiency for generic group signature schemes. In *Annual International Cryptology Conference*, pages 413–430. Springer, 1999.

[CCL+21] Konstantinos Chalkias, Shir Cohen, Kevin Lewi, Fredric Moezinia, and Yolan Romailler. Hashwires: Hyperefficient credential-based range proofs. *Proceedings on Privacy Enhancing Technologies*, 4:76–95, 2021.

[CCS08] Jan Camenisch, Rafik Chaabouni, and Abhi Shelat. Efficient protocols for set membership and range proofs. In *International Conference on the Theory and Application of Cryptology and Information Security*, pages 234–252. Springer, 2008.

[CFT98] Agnes Hui Chan, Yair Frankel, and Yiannis Tsiounis. Easy come - easy go divisible cash. In Kaisa Nyberg, editor, *Advances in Cryptology - EUROCRYPT '98, International Conference on the Theory and Application of Cryptographic Techniques, Espoo, Finland, May 31 - June 4, 1998, Proceeding*, volume 1403 of *Lecture Notes in Computer Science*, pages 561–575. Springer, 1998.

[CGKR22] Geoffroy Couteau, Dahmun Goudarzi, Michael Klooß, and Michael Reichle. Sharp: Short relaxed range proofs. In *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security*, pages 609–622, 2022.

[CHJ+22] Heewon Chung, Kyoohyung Han, Chanyang Ju, Myungsun Kim, and Jae Hong Seo. Bulletproofs+: Shorter proofs for a privacy-enhanced distributed ledger. *IEEE Access*, 10:42081–42096, 2022.

[CKLM12] Melissa Chase, Markulf Kohlweiss, Anna Lysyanskaya, and Sarah Meiklejohn. Malleable proof systems and applications. In *Advances in Cryptology–EUROCRYPT 2012: 31st Annual International Conference on the Theory and Applications of Cryptographic Techniques, Cambridge, UK, April 15-19, 2012. Proceedings 31*, pages 281–300. Springer, 2012.

[CKLR21] Geoffroy Couteau, Michael Klooß, Huang Lin, and Michael Reichle. Efficient range proofs with transparent setup from bounded integer commitments. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 247–277. Springer, 2021.

[CM99] Jan Camenisch and Markus Michels. Proving in zero-knowledge that a number is the product of two safe primes. In *Advances in Cryptology—EUROCRYPT'99: International Conference on the Theory and Application of Cryptographic Techniques Prague, Czech Republic, May 2–6, 1999 Proceedings 18*, pages 107–122. Springer, 1999.

[CPP17] Geoffroy Couteau, Thomas Peters, and David Pointcheval. Removing the strong rsa assumption from arguments over the integers. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 321–350. Springer, 2017.

[DBB+15] Gaby G Dagher, Benedikt Bünz, Joseph Bonneau, Jeremy Clark, and Dan Boneh. Provisions: Privacy-preserving proofs of solvency for bitcoin exchanges. In *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*, pages 720–731, 2015.

[DF01] Ivan Damgård and Eiichiro Fujisaki. An integer commitment scheme based on groups with hidden order. *Cryptology ePrint Archive*, 2001.

[DPLS19] Rafaël Del Pino, Vadim Lyubashevsky, and Gregor Seiler. Short discrete log proofs for fhe and ring-lwe ciphertexts. In *IACR International Workshop on Public Key Cryptography*, pages 344–373. Springer, 2019.

[DRZ20] Vanesa Daza, Carla Ràfols, and Alexandros Zacharakis. Updateable inner product argument with logarithmic verifier and applications. In *Public-Key Cryptography–PKC 2020: 23rd IACR International Conference on Practice and Theory of Public-Key Cryptography, Edinburgh, UK, May 4–7, 2020, Proceedings, Part I 23*, pages 527–557. Springer, 2020.

[Eag22] Liam Eagen. Bulletproofs++. *Cryptology ePrint Archive*, 2022.

[EMBB17] Saba Eskandarian, Eran Messeri, Joseph Bonneau, and Dan Boneh. Certificate transparency with privacy. *Proceedings on Privacy Enhancing Technologies*, 4:232–247, 2017.

[ESLL19] Muhammed F Esgin, Ron Steinfeld, Joseph K Liu, and Dongxi Liu. Lattice-based zero-knowledge proofs: new techniques for shorter and faster constructions and applications. In *Annual International Cryptology Conference*, pages 115–146. Springer, 2019.

[ESS+19] Muhammed F Esgin, Ron Steinfeld, Amin Sakzad, Joseph K Liu, and Dongxi Liu. Short lattice-based one-out-of-many proofs and applications to ring signatures. In *Applied Cryptography and Network Security: 17th International Conference, ACNS 2019, Bogota, Colombia, June 5–7, 2019, Proceedings 17*, pages 67–88. Springer, 2019.

[FO97] Eiichiro Fujisaki and Tatsuaki Okamoto. Statistical zero knowledge protocols to prove modular polynomial relations. In *Advances in Cryptology—CRYPTO'97: 17th Annual International Cryptology Conference Santa Barbara, California, USA August 17–21, 1997 Proceedings 17*, pages 16–30. Springer, 1997.

[GHL22] Craig Gentry, Shai Halevi, and Vadim Lyubashevsky. Practical non-interactive publicly verifiable secret sharing with thousands of parties. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 458–487. Springer, 2022.

[Gro05] Jens Groth. Non-interactive zero-knowledge arguments for voting. In *Applied Cryptography and Network Security: Third International Conference, ACNS 2005, New York, NY, USA, June 7-10, 2005. Proceedings 3*, pages 467–482. Springer, 2005.

[Gro16] Jens Groth. On the size of pairing-based non-interactive arguments. In *Advances in Cryptology–EUROCRYPT 2016: 35th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Vienna, Austria, May 8-12, 2016, Proceedings, Part II 35*, pages 305–326. Springer, 2016.

[Gro21] Jens Groth. Non-interactive distributed key generation and key resharing. *Cryptology ePrint Archive*, 2021.

[GWC19] Ariel Gabizon, Zachary J Williamson, and Oana Ciobotaru. Plonk: Permutations over lagrange-bases for oecumenical noninteractive arguments of knowledge. *Cryptology ePrint Archive*, 2019.

[ide23] iden3. rapidsnark. https://github.com/iden3/rapidsnark, 2023.

[JC21] Yan Ji and Konstantinos Chalkias. Generalized proof of liabilities. In *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*, pages 3465–3486, 2021.

[KL07] Jonathan Katz and Yehuda Lindell. *Introduction to modern cryptography: principles and protocols*. Chapman and hall/CRC, 2007.

[KTX08] Akinori Kawachi, Keisuke Tanaka, and Keita Xagawa. Concurrently secure identification schemes based on the worst-case hardness of lattice problems. In *Advances in Cryptology-ASIACRYPT 2008: 14th International Conference on the Theory and Application of Cryptology and Information Security, Melbourne, Australia, December 7-11, 2008. Proceedings 14*, pages 372–389. Springer, 2008.

[KZG10] Aniket Kate, Gregory M Zaverucha, and Ian Goldberg. Constant-size commitments to polynomials and their applications. In *Advances in Cryptology-ASIACRYPT 2010: 16th International Conference on the Theory and Application of Cryptology and Information Security, Singapore, December 5-9, 2010. Proceedings 16*, pages 177–194. Springer, 2010.

[Lib23] Benoit Libert. Vector commitments with short proofs of smallness. *Cryptology ePrint Archive*, 2023.

[Lip03] Helger Lipmaa. On diophantine complexity and statistical zero-knowledge arguments. In *Advances in Cryptology-ASIACRYPT 2003: 9th International Conference on the Theory and Application of Cryptology and Information Security, Taipei, Taiwan, November 30–December 4, 2003. Proceedings 9*, pages 398–415. Springer, 2003.

[LLM+16] Benoît Libert, San Ling, Fabrice Mouhartem, Khoa Nguyen, and Huaxiong Wang. Zero-knowledge arguments for matrix-vector relations and lattice-based group encryption. In *International Conference on the Theory and Application of Cryptology and Information Security*, pages 101–131. Springer, 2016.

[LLNW16] Benoît Libert, San Ling, Khoa Nguyen, and Huaxiong Wang. Zero-knowledge arguments for lattice-based accumulators: logarithmic-size ring signatures and group signatures without trapdoors. In *Advances in Cryptology–EUROCRYPT 2016: 35th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Vienna, Austria, May 8-12, 2016, Proceedings, Part II 35*, pages 1–31. Springer, 2016.

[LLNW18] Benoît Libert, San Ling, Khoa Nguyen, and Huaxiong Wang. Lattice-based zero-knowledge arguments for integer relations. In *Annual International Cryptology Conference*, pages 700–732. Springer, 2018.

[LNP22] Vadim Lyubashevsky, Ngoc Khanh Nguyen, and Maxime Plançon. Lattice-based zero-knowledge proofs and applications: shorter, simpler, and more general. In *Annual International Cryptology Conference*, pages 71–101. Springer, 2022.

[LNS20] Vadim Lyubashevsky, Ngoc Khanh Nguyen, and Gregor Seiler. Practical lattice-based zero-knowledge proofs for integer relations. In *Proceedings of the 2020 ACM SIGSAC conference on computer and communications security*, pages 1051–1070, 2020.

[Max16] Greg Maxwell. Confidential transactions, 2016. https://people.xiph.org/~greg/confidential_values.txt.

[MKvWK19] Eduardo Morais, Tommy Koens, Cees van Wijk, and Aleksei Koren. A survey on zero knowledge range proofs and applications. *CoRR*, abs/1907.06381, 2019.

[MR14] Daisuke Mashima and Arnab Roy. Privacy preserving disclosure of authenticated energy usage data. In *2014 IEEE international conference on smart grid communications (SmartGridComm)*, pages 866–871. IEEE, 2014.

[NTWZ19] Khoa Nguyen, Hanh Tang, Huaxiong Wang, and Neng Zeng. New code-based privacy-preserving cryptographic constructions. In *Advances in Cryptology–ASIACRYPT 2019: 25th International Conference on the Theory and Application of Cryptology and Information Security, Kobe, Japan, December 8–12, 2019, Proceedings, Part II 25*, pages 25–55. Springer, 2019.

[Ped91] Torben Pryds Pedersen. Non-interactive and information-theoretic secure verifiable secret sharing. In *Annual international cryptology conference*, pages 129–140. Springer, 1991.

[Poe16] Andrew Poelstra. Mimblewimble. 2016.

[PVW08] Chris Peikert, Vinod Vaikuntanathan, and Brent Waters. A framework for efficient and composable oblivious transfer. In *Annual international cryptology conference*, pages 554–571. Springer, 2008.

[RS96] Ronald L Rivest and Adi Shamir. Payword and micromint: Two simple micropayment schemes. In *International workshop on security protocols*, pages 69–87. Springer, 1996.

[Ste96] Jacques Stern. A new paradigm for public key identification. *IEEE Transactions on Information Theory*, 42(6):1757–1768, 1996.

[Tom20] Alin Tomescu. Range proofs from polynomial commitments, re-explained, March 2020. https://decentralizedthoughts.github.io/2020-03-03-range-proofs-from-polynomial-commitments-reexplained/.

[WC22] Nan Wang and Sid Chi-Kin Chau. Flashproofs: Efficient zero-knowledge arguments of range and polynomial evaluation with transparent setup. In *International Conference on the Theory and Application of Cryptology and Information Security*, pages 219–248. Springer, 2022.

[WCL23] Nan Wang, Sid Chi-Kin Chau, and Dongxi Liu. Swiftrange: A short and efficient zero-knowledge range argument for confidential transactions and more. *Cryptology ePrint Archive*, 2023.

[Wui18] Pieter Wuille. libsecp256k1, 2018. https://github.com/bitcoin/secp256k1.

[YAZ+19] Rupeng Yang, Man Ho Au, Zhenfei Zhang, Qiuliang Xu, Zuoxia Yu, and William Whyte. Efficient lattice-based zero-knowledge arguments with standard soundness: construction and applications. In *Advances in Cryptology–CRYPTO 2019: 39th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 18–22, 2019, Proceedings, Part I 39*, pages 147–175. Springer, 2019.

[ZZT+23] Zibo Zhou, Zongyang Zhang, Hongyu Tao, Tianyu Li, and Boyu Zhao. Efficient inner product arguments and their applications in range proofs. *IET Information Security*, 17(3):485–504, 2023.

A FORMAL DEFINITION OF NIZKS

NIZK. Non-interactive zero knowledge (NIZK) [BFM88] proof is a cryptographic primitive enables a prover to convince a (sceptical) verifier about the truth of a statement without disclosing any additional information in one round of communication. A NIZK can be build in two possible settings: either in Random Oracle Model (ROM) or in the Common Reference String (CRS) model. Next we recall the definition of NIZK proofs in the CRS model and list their main security properties.

Definition A.1 (Non-Interactive Zero-Knowledge Proofs). Let $\mathcal{R}$ be an NP-relation, the language $\mathcal{L}_{\mathcal{R}}$ can be defined as $\mathcal{L}_{\mathcal{R}} = \{x \mid \exists\ w \text{ s.t. } (x, w) \in \mathcal{R}\}$, where x and w denote public statement and secret witness, respectively. A NIZK, denoted by Π, for $\mathcal{R}$ consists of three main PPT algorithms $\Pi = (\mathsf{CRSGen}, \mathsf{Prove}, \mathsf{Verify})$ defined as follows:

- $\Pi.\mathsf{CRSGen}(1^\lambda, \mathcal{R}) \to \mathsf{CRS}$: The CRS generation algorithm takes the unary representation of the security parameter λ and relation $\mathcal{R}$ as inputs and returns a set of common reference string CRS as output.
- $\mathsf{Prove}(\mathsf{CRS}, x, w) \to \pi$: The prove algorithm takes CRS, a public statement x and a secret witness w as inputs, and it then returns a proof π as output.
- $\mathsf{Verify}(\mathsf{CRS}, x, \pi) \to 0/1$: The verify algorithm takes CRS, a public statement x and a proof π as input, and it then returns a bit indicating either the acceptance, 1, or rejection, 0, as output.

Informally speaking, a NIZK proof has three main security properties: Completeness, Zero-Knowledge and soundness (extractability), which we formally recall them in below:

Definition A.2 (Completeness). A NIZK proof, Π, is called complete, if for all security parameters, λ, and all pairs of valid $(x, w) \in R$ we have,

$$\Pr\left[\begin{array}{c}\mathsf{CRS} \leftarrow \mathsf{CRSGen}(1^\lambda) : \\ \mathsf{Verify}(\mathsf{CRS}, x, \mathsf{Prove}(\mathsf{CRS}, x, w)) = 1\end{array}\right] \geq 1 - \mathsf{negl}(\lambda)\ .$$

Definition A.3 (Zero-Knowledge). A NIZK proof system, Π, for a given relation $\mathcal{R}$ and its corresponding language $\mathcal{L}_{\mathcal{R}}$, we define a pair of algorithms $\mathsf{Sim} = (\mathsf{Sim}_1, \mathsf{Sim}_2)$ as the simulator. The simulator operates such that $\mathsf{Sim}'(\mathsf{CRS}, \mathsf{tpd}, x, w) = \mathsf{Sim}_2(\mathsf{CRS}, \mathsf{tpd}, x)$ when $(x, w) \in \mathcal{R}$, and $\mathsf{Sim}'(\mathsf{CRS}, \mathsf{tpd}, x, w) = \perp$ when $(x, w) \notin \mathcal{R}$, where tpd is a trapdoor. For $b \in \{0, 1\}$, we define the experiment $\mathsf{ZK}^{\Pi}_{b,\mathsf{Sim}}(1^\lambda, \mathcal{A})$ in fig. 7. The associated advantage of an adversary $\mathcal{A}$ is defined as

$$Adv^{\mathsf{ZK}}_{\Pi,\mathcal{A},\mathsf{Sim}}(\lambda) := \left|\begin{array}{l}\Pr[\mathsf{ZK}^{\Pi}_{0,\mathsf{Sim}}(1^\lambda, \mathcal{A}) = 1] - \\ \Pr[\mathsf{ZK}^{\Pi}_{1,\mathsf{Sim}}(1^\lambda, \mathcal{A}) = 1]\end{array}\right|\ .$$

A NIZK proof system Π achieves perfect and computational zero-knowledge, w.r.t a simulator $\mathsf{Sim} = (\mathsf{Sim}_1, \mathsf{Sim}_2)$, if for all PPT adversaries $\mathcal{A}$ we have $Adv^{\mathsf{ZK}}_{\Pi,\mathcal{A},\mathsf{Sim}}(\lambda) = 0$, and $Adv^{\mathsf{ZK}}_{\Pi,\mathcal{A},\mathsf{Sim}}(\lambda) \leq \mathsf{negl}(\lambda)$, respectively.

$\mathsf{ZK}^{\Pi}_{0,\mathsf{Sim}}(1^\lambda, \mathcal{A})$	$\mathsf{ZK}^{\Pi}_{1,\mathsf{Sim}}(1^\lambda, \mathcal{A})$
$\mathsf{CRS} \leftarrow \mathsf{CRSGen}(1^\lambda)$	$(\mathsf{CRS}, \mathsf{tpd}) \leftarrow \mathsf{Sim}_1(1^\lambda)$
$\alpha \leftarrow \mathcal{A}^{\mathsf{Prove}(\mathsf{CRS},\cdot,\cdot)}(\mathsf{CRS})$	$\alpha \leftarrow \mathcal{A}^{\mathsf{Sim}'(\mathsf{CRS},\mathsf{tpd},\cdot,\cdot)}(\mathsf{CRS})$
return α	return α

Figure 7: Zero-knowledge security property of a NIZK, Π.

Definition A.4 (Extractability [CKLM12]). A NIZK proof system Π for a relation $\mathcal{R}$ and the language L is called extractable if there

exists a pair of algorithms $\mathsf{Ext} := (\mathsf{Ext}_1, \mathsf{Ext}_2)$ called extractors with the following advantage for all PPT adversaries $\mathcal{A}$:

$$Adv^{\mathsf{CRS}}_{\Pi,\mathcal{A}} := |\Pr[\mathsf{CRS} \leftarrow \mathsf{CRSGen}(1^\lambda); 1 \leftarrow \mathcal{A}(\mathsf{CRS})] - \Pr[(\mathsf{CRS}, \mathsf{st}) \leftarrow \mathsf{Ext}_1(1^\lambda); 1 \leftarrow \mathcal{A}(\mathsf{CRS})]|,$$

and

$$Adv^{\mathsf{Ext}}_{\Pi,\mathcal{A}}(\lambda) := \Pr\begin{bmatrix}(\mathsf{CRS}_{\mathsf{Ext}}, \mathsf{st}_{\mathsf{Ext}}) \leftarrow \mathsf{Ext}_1(1^\lambda) \\ (x, \pi) \leftarrow \mathcal{A}(\mathsf{CRS}_{\mathsf{Ext}}) : \\ \mathsf{Verify}(\mathsf{CRS}_{\mathsf{Ext}}, x, \pi) = 1 \wedge \\ (x, \mathsf{Ext}_2(\mathsf{CRS}_{\mathsf{Ext}}, \mathsf{st}_{\mathsf{Ext}}, x, \pi)) \notin \mathcal{R}\end{bmatrix}.$$

A NIZK proof system Π is called extractable, w.r.t an extractor $\mathsf{Ext} = (\mathsf{Ext}_1, \mathsf{Ext}_2)$, if $Adv^{\mathsf{CRS}}_{\Pi,\mathcal{A}} \leq \mathsf{negl}(\lambda)$ and $Adv^{\mathsf{Ext}}_{\Pi,\mathcal{A}}(\lambda) \leq \mathsf{negl}(\lambda)$. Additionally, we refer to an extractable NIZK proof as a non-interactive zero-knowledge proof of knowledge, or NIZKPoK in short.

Succinctness. Zero-Knowledge Succinct Non-Interactive Arguments of Knowledge, zkSNARK in short, are NIZKPoK proofs that adhere to succinctness requirements. These proofs maintain communication complexity (proof size) at sublinear levels, and in some cases, the verifier's computational workload remains sublinear, regardless of the size of the witness. In this paper, we primarily concentrate on zkSNARKs, ensuring that the proofs are short and verification cost is low while the mentioned security definitions for NIZK remain applicable for them.