Pairing-Based Non-Interactive Zero-Knowledge Arguments and Applications

Zaira Pindado

December 2020

TESI DOCTORAL UPF / 2020
Supervisor: Carla Ràfols
Department of Information and Communication Technologies

To Máximo and Lourdes.

Thanks

I want to thank Carla Ràfols for all the hard work and effort into helping me in these four years. I am very happy to have had her as a supervisor. It was hard for me to understand at the beginning, but when I got it, I enjoyed a lot our discussions and work. She always had a different perspective and intuitions that I hadn't imagined. I want to thank her for everything she taught me, and for being a very caring, dedicated and patient supervisor.

Big thanks to Vanesa Daza who also had a very important role during these years, not just because all her priceless advice, but also for her personal support.

Thanks to administrative people in Tanger building who always solve queries efficiently, especially to Lydia García who makes bureaucracy less annoying.

I want to thank the people that give me unconditional support in any challenge that I put myself into: my parents, Máximo and Lourdes, and my close friends, Laura and Paula. Thanks also to all the members of my family who blindly support me, even though they sometimes cannot fully understand what I am doing since I started mathematics: Teresa, Carmen, Tati, Mar, Mónica, Lluna and Mar. I also want to dedicate a warm thank to my grandma, who cannot see this project finished but, since she believed more in me than myself, she already knew I was able to do it.

Thanks to my second family between Barcelona, Valencia, Galicia, Madrid and Mallorca. I want to thank especially Ana and Laura, my friends and flatmates at No drama llama, one of the places where I lived, enjoyed very nice moments, passed a confinement and learnt a lot from them. I will never be able to forget all your support. To Alex, Miguel and Marcos for all the great moments we shared, specially Alex for all his advice. To Charlie, who, finally, can change my name in his phone from predoctor to doctor. To my loved mathematical poplities, Lucia and Mary, who were very important since the first years of the degree, as well as Julia, Roser and Xesca. To all people related to Esplai, particularly, Alba, Ali, Anna, Berta, Cris, Laura (again) and Marta.

I want to thank all the people of my past who made me possible to start this project and supported me as a part of their family during some years.

I want to thank Helger Lipmaa for hosting me at the University of Tartu, and the people of the crypto group who made me feel very welcomed, especially Janno Siim, Karim Baghery and Shahla Atapoor.

Finally, thanks to all 52.210 people and neighbours to make the office, the club and Sopena such nice places to meet, discuss and hang out. Special thanks to Rasoul, Fede, Javi, Pablo, Geordie, Miquel, Javi V., Fede M., Xavi, Arantxa, Alexandros, Amelia, Adri, Juan, Cecilia, Conor, Sergi, Marta.
I want to especially thank Javi, who started being the unknown guy who stole me the

scholarship and ended being one of my best friends. In this process, he was a very important figure to me because he helped me a lot, not just for crypto and the classes, but also for personal stuff, he always has the most suitable sentence to make you feel good. I learned a lot from him and I hope, as he isn't a typical Galician guy, he will stay in Barcelona for a long time.
Another especial dedication for Fede, who began being my friend very soon since his natural sociability and his interest in learning Catalan. When I got to know him I also appreciated his effort in trying to change the world into a better place starting with the rubbish in the department and ending with giving peace and serenity always in his speaking. After some years, he ended being one of the most important people in my life, as my love, friend and colleague of life.

Abstract

Elliptic curves with a bilinear map, or pairing, have a rich algebraic structure that has been fundamental to develop practical Non-Interactive Zero-Knowledge (NIZK) proofs.

On the theoretical side, we explore how efficient can NIZK proofs be under weak complexity assumptions. Specifically, we reduce the cost of proofs of satisfiability of quadratic equations, we define a new commitment scheme that is compatible with other pairing-based NIZK arguments, and we construct a simulation-sound argument that results in a new a signature of knowledge with communication sublinear in the circuit size under standard assumptions.

Additionally, we study how to reduce the cost of verification in one of the most widely deployed NIZK arguments in practice.

Resum

Les corbes el·líptiques amb una aplicació bilineal, o pairing, tenen una estructura algebraica molt rica que ha sigut fonamental per desenvolupar les proves no interactives de zero coneixement (NIZK).

En la banda teòrica, explorem quant eficients poden ser les proves NIZK sota hipòtesis de complexitat dèbils. Més concretament, reduïm el cost de les proves de satisfacció per equacions quadràtiques, definim un nou esquema de compromís que és compatible amb altres proves NIZK basades en pairings i construïm una prova que resulta en una nova signatura de coneixement amb una comunicació sublineal en la mida del circuit sota hipòtesis estàndards.

A més, estudiem com es redueix el cost de verificació en una de les proves NIZK més desenvolupades a la pràctica.

Summary

List of Figures

List of Tables

Chapter 1

Introduction

Etymologically, *cryptography* means hidden or invisible writing. Traditionally, it was concerned with the design of cyphers, but modern cryptography comprises a huge amount of different primitives, systems and protocols.

In the Ancient Age, we find some examples of encryptions like the Spartan Scytale, which is based on transposing the letters of the message, or the Julius Caesar, a substitution cypher where we change each letter of the message for the letter in the alphabet at some fixed distance. At that time, it was used mostly by governants and military people. However, the concept has been changing over the time with the emergence of new technologies, especially during the last century with the development of computing machines and, even more, since the launch of the internet.

In the middle of the last century, cryptography became the science between Mathematics and Computer Science that studies the mathematical techniques for protecting information and systems against adversarial attacks. Nowadays, this discipline represents an essential set of complex tools and protocols designed for protecting information in the communications. We will begin giving a general view of the principles and basic notions of this science, and then we present Zero-Knowledge proofs, a cryptographic primitive that allows to validate the correctness of some statement without leaking secret information about it, that is the main object of study of this thesis.

1.1 Modern cryptography and basic notions

> *Cryptography is a puzzling kaleidoscope where we have to face and use malice at the same time, to rationalize irrational behaviors, to prove un-*

provability, and to twist mathematics to make it fit applications.

Serge Vaudenay

Designing cryptographic schemes is not an easy task. As cryptographers, when creating a new scheme for some specific functionality we have to take into account the presence of malicious parties, that try to extract information or modify data for some particular interest. The difficulty resides on that we cannot prevent all the possible actions for these attacks. As every other science, *cryptography* has some basic principles that must be followed in the process of constructing new cryptographic schemes, Katz and Lindell [89] enumerate them as formal definitions, conditions where the definitions hold and rigorous proofs that guarantee that a scheme satisfies the definition under the specified conditions.

The *formal definitions* establish the security goals that we desire for our scheme and set what we consider as a successful attack. This determines the power of the adversary, like its computational limitations and the threat model, for example, in a chosen-plaintext attack, the adversary learns some plaintexts-ciphertexts pairs of its choice and it has to deduce information about the plaintext of a new ciphertext (all generated with the same key).

Some cryptographic constructions can be proven to be unconditionally secure. However, most of them are proven conditionally secure, that is, security holds if some computational complexity problems are hard, or "infeasible". We use the term *assumptions* to express mathematically that solving some computational problem is conjectured to be hard.

Some assumptions provide more confidence than others. It is preferable to base security on those that have been studied and tested for more time. For example, the Factorization assumption that, given a very big natural number N product of two prime numbers, $N = p \cdot q$, it is infeasible to find p, q; or the Discrete Logarithm assumption, that given two elements $g, a \in \mathbb{Z}_p^*$, with p prime, it is infeasible to find the Discrete logarithm of a respect to g, i.e. α such that $g^\alpha = a$. Both assumptions are examples of extensively researched assumptions.

Moreover, it is preferable to work with a weak assumption than a strong one. Because the stronger can be refuted, while the weaker can remain true.

In 2003, Naor classified assumptions [105] into *falsifiable*, those for which we can design an experiment that allows us to efficiently check if it is false or true that someone has broken the assumption, and *non-falsifiable*, where we cannot design such an experiment. In the latter type, these assumptions usually not only assume that a certain problem is hard, but also restrict the type of strategies the adversary can follow to come up with the final message.

The assumptions that we mentioned above and most of the assumptions that we

consider in this thesis are falsifiable. However, many interesting constructions of NIZK proofs are secure under non-falsifiable assumptions. For example, the Knowledge of Exponent assumption (KEA) [42] assumes that an adversary who is given g, g^α and comes up with elements Y, C, such that $Y = C^\alpha$, must know c such that $C = g^c$. Indirectly, we are assuming that the way of computing Y is $Y = (g^\alpha)^c$. Another example is the assumption that the adversary is generic or algebraic. In both, we restrict the strategies of the adversary to group operations and we can extract the coefficients that express each new element output by the adversary as a linear combination of the elements that it has access to.

If we sort the assumptions that we have mentioned already by their strength, we have the weakest of the falsifiable assumptions first, progressively stronger falsifiable assumptions, and then the non-falsifiable from weakest to strongest: first, knowledge of exponent type, and then, those where we assume restricted type of adversaries, first the algebraic group model (AGM, [57]) and, the last, the generic group model (GGM, [117]).

Once the formal definition and the assumption in which security relies on are determined, the *proof of security* provides a mathematical guarantee that no adversary will succeed in attacking the scheme. Typically, this is done by a reduction argument where we assume an adversary success in the attack to the scheme and, by a sequence of rigorous steps, we deduce the assumption is broken. Then, the advantage of the adversary, i.e. the probability of breaking the scheme, is upper bound by the probability of breaking the assumption with some multiplication factor. Ideally, we want the factor to be similar to 1, which means the time and effort to break the assumption and the time and effort to break the scheme are similar, which is captured by the notion of *tightness*.

Our scope

In this thesis, we work on Zero-Knowledge (ZK) proof schemes, which is a primitive that allows one party, called the *prover*, to provide a convincing proof about the validity of some statement to another party, the *verifier*, without revealing any additional information. Intuitively, having such a proof that some claim holds is equivalent to having received from a trusted party that this claim is true.

Zero-Knowledge proofs are considered one of the fundamental primitives in cryptography. Although, for many years it was a mostly theoretical tool, in the last decade it has been improved with new techniques from pairing-based cryptography, becoming order of magnitude more efficient. Nowadays, it is used in practice as a proof of correct computations in blockchain applications.

Our results propose new trade-offs between better security, with weaker assumptions and stronger definitions, and better efficiency, in terms of generation of the public

parameters, the complexity of the parties or proof size.

The rest of this chapter is organized as follows. We present in Section 1.2 the basic definitions and the building blocks used in this thesis, but formal definitions are given in Chapter 2. In Section 1.3, we present a summary of our results and finally, a list of the publications.

1.2 Zero-Knowledge Proofs

In mathematics, a proof is a sequence of consistent statements derived logically from the axioms and premises to the conclusion by using some rules. These proofs are considered as fixed objects and they are fundamental for trusting the validity of a result, for example, a theorem. In real-life, proofs have a dynamic interpretation, they are considered as a process by which the validity of a claim is proven. In both cases, there is an entity who provides the proof, the prover, and a verification procedure, that is a simpler procedure to validate and it is executed by the verifier.

In 1989, Goldwaser, Micali and Rackoff introduced Zero-Knowledge (ZK) Proofs in [66]. Given a language $\mathcal{L}$, that defines a type of problems, the prover goal's is to convince the verifier that some public statement x is in this language without revealing its secret information, the witness w. They interchange a series of messages until the verifier is convinced about the validity of the statement, $x \in \mathcal{L}$ or the contrary, so it decides to accept or reject the proof. There is a polynomial-time algorithm $\mathbf{R}_{\mathcal{L}}$ associated to the language $\mathcal{L}$, to recognize if a possible solution w satisfies the relation with x, i.e. if $(x, w) \in \mathbf{R}_{\mathcal{L}}$. Ideally, if both parties are honest and the relation is satisfied, the verifier should accept.

In the following, we briefly present the fundamental properties of these proofs, then other additional properties that we use in our results, and the concrete objectives of the thesis.

Fundamental properties

We require that all the proofs have two fundamental properties, *Completeness* and *Soundness*. The proof should be convincing for the verifier as long as the statement is true, which is captured by the definition of completeness. This is the desired behaviour when both parties are honest. On the other hand, the soundness property ensures that malicious provers cannot convince the verifier of false statements.

Apart from these properties, in ZK proofs we also require the *Zero-Knowledge* property, which guarantees that no extra information is leaked from the messages involved in the proof, beyond the validity of the statement. Here, by extra information

we mean that the verifier cannot gain more knowledge about the witness from the exchange of the messages. In other words, what the verifier was able to do or what it knew about the witness is the same before and after the exchange of messages, i.e. it did not gain any additional knowledge. To prove this property, we define an algorithm called *simulator*, that has access to some trapdoor, a secret parameter, that gives the power to make a proof that passes the verification without knowing the witness. Then, we prove Zero-Knowledge by showing indistinguishability between honest proofs, those created with the witness, and simulated proofs, those created with the trapdoor. The idea behind this is that if such a simulator exists, in the verifier's view, the same output could have been generated without the actual witness.

There are different variations in the definitions of these fundamental properties. For example, soundness can be guaranteed by a statistical argument (soundness is proven unconditionally). In this case, we have *perfect soundness*. On the other hand, in scenarios where the prover has limited resources of computation, a relaxed notion is enough. When a proof has *computational soundness* is called *Argument*.

Arguments are interesting because computationally-sound proofs are much more efficient than those perfectly sound. Moreover, usually, they can be constructed with perfect zero-knowledge property. Many NIZK arguments for general statements (i.e. for NP-complete languages), prioritize efficiency over security and prove soundness under very strong (often non-falsifiable) assumptions. Several results in this thesis solve this trade-off in the opposite way and study how efficient can we go in efficiency under mild assumptions.

Additional properties

Beyond the fundamental properties of ZK proofs, there exist other notions that guarantee stronger security.

The *simulation soundness* property is a stronger notion of soundness that guarantees that no adversary can come up with a valid proof for a false statement even when it has seen some simulated proofs previously. This notion is interesting because it ensures that the proof is sound and non-malleable, that is the adversary cannot re-use previous proofs, obtained from querying the simulator with some statements of its choice, to compute a fresh one for a new statement.

Additionally, a proof is *of Knowledge*, or *knowledge sound*, if the prover claims knowledge of some object and uses it to compute the proof. Technically, we consider that a machine knows some object if there exists another efficient algorithm that can extract this object from it. We distinguish between two types of extraction, *black-box* when the extractor can extract the witness from the outputs of the prover, and *non-black-box* when we assume the extractor has access to the code of the prover.

Black-box extraction is more realistic since it does not need to access the code of the adversary.

Moreover, if the proof is knowledge and simulation sound, we say it is *simulation-extractable* (SE), [70]. This is the strongest definition, and it guarantees that the adversary cannot come up with a fresh valid proof unless it knows a witness, even if it has seen an arbitrary number of simulated proofs.

Finally, in all our constructions we work in the non-interactive setting, where the proof consists of just one single message. Non-interactive ZK (NIZK) proofs are very interesting in practice because its verification can be done offline.

Pairing-based Non-Interactive Zero-Knowledge Proofs

In this thesis, we deal with Non-Interactive Zero-Knowledge Proofs. The first ZK proofs were introduced as an interactive protocol between the prover and the verifier, where the verifier randomly samples elements to create challenges for the prover and expects convincing answers. In contrast, in the non-interactive proofs, this exchange of messages is substituted by a single one from the prover to the verifier that constitutes the proof and can be checked off-line by the verifier. Then, intuitively, some parameters that substitute the verifier's challenge are needed. The Common Reference String (crs) model introduced by Blum et al. [20] assumes that the prover and the verifier share some public parameters generated by a trusted third party. In this model, the prover should combine the witness with the elements in the crs to create the proof, instead of the challenge of the verifier.

The usability of NIZK proofs depends on the class of *languages* that they apply to and the *efficiency* associated with the proof system. Ideally, one would like to define proof systems that allow to prove very general statements, like CircuitSat that is a very powerful language because it is NP-complete. This means that any NP problem can be converted in an efficient way to a CircuitSat instantiation. Further, circuits encode in a natural way many types of computation. However, historically it was difficult to design efficient proofs for these powerful languages. For many years, the only known efficient constructions were for very specific languages, like identification schemes ([55]) and shuffle arguments for electronic voting [116].

In the last decades, this field suffered a big change with the development of cryptography in bilinear groups. A bilinear group consists of two elliptic curves along with a bilinear operation called pairing or bilinear map. The bilinear structure is very suitable to develop efficient constructions of NIZK proofs with efficient public verification. Pairing-based NIZK proofs were introduced by Groth, Ostrovsky and Sahai [75] in 2006 where the authors constructed the first efficient NIZK argument for NP languages in the crs model. Although this work was much more efficient concretely than

any other NIZK proof in the crs model, a proof for CircuitSat requires communication linear in the circuit size, which is completely impractical for most interesting circuits. The techniques introduced in Groth, Ostrovsky and Sahai were important to inspire the framework of Groth-Sahai proofs [78] in 2008, which defines proofs for specific languages, concretely proofs of satisfiability of several types of quadratic equations in bilinear maps. For many equations types they remain the best alternative based on falsifiable assumptions in bilinear maps. In this line of work all proofs are secure under weak falsifiable assumptions.

After that, in 2010, Groth [71] presented the first constant size NIZK argument for CircuitSat combining ideas from the interactive setting and techniques from previous pairing-based NIZK proofs. Intuitively, since the proof is very small (constant, independent of the circuit size), it is not possible under standard assumptions to extract a witness that allows to decide if the adversary has cheated. Then, a non-black-box assumption is needed to extract a witness linear in the circuit size in the security proof, as formalized by the result of Gentry-Wichs [63]. Therefore, in Groth's work [71] the security is proven under a knowledge of exponent assumption, which is non-falsifiable. This technique started a line of research followed by Gennaro et al. [61] and other works that progressively decrease the size of the proof, all of them are based on non-falsifiable assumptions. These constructions are called *zk-Succinct Non-Interactive Arguments of Knowledge* (zk-SNARKs).

Another line of research that makes the proof very efficient in terms of communication under very weak falsifiable assumptions was introduced by Jutla and Roy in [85] with the *Quasi-Adaptive NIZK* (QA-NIZK) proofs. For some very specific pairing languages, they consist in just one group element, for example, for membership in a linear space defined in a group.

Our work combines techniques from the three lines of research. In the following, we explain these building blocks with more detail and we introduce commitment schemes that, are a fundamental tool for building NIZK proofs.

Building blocks

"You may seek it with thimbles — and seek it with care;
You may hunt it with forks and hope;
You may threaten its life with a railway-share;
You may charm it with smiles and soap — "
("That's exactly the method", the Bellman bold in a hasty parenthesis cried,
"That's exactly the way I have always been told.
That the capture of Snarks should be tried!")

Zk-SNARKs represent an important breakthrough in the zero-knowledge field. The combination of the succinctness property with the fact that they are defined for very general statements, makes them very useful to work in different scenarios. In verifiable delegation of computation, some party delegates to another party with more resources a computation, and receives the computation result along with a zk-SNARK proving its correctness (Pinocchio [108]). On the other hand, they have been implemented in the field of cryptocurrencies (Zcash[2, 82], Ethereum[29], Monero[1]) where zk-SNARKs guarantee the correctness of the transactions, in the sense of preventing double-spending and offering anonymity. They are also implemented in smart contracts (Hawk [93]) and anonymous identification systems (iden3[3]).

More in detail, the key of their efficiency is that zk-SNARKs are not only succinct, but also they are concretely efficient in terms of communication and verification. The shortest proof consists of just 3 group elements and its verification is dominated by 3 pairings.

The main idea to achieve constant size proof is based on characterizing CircuitSat in terms of some polynomials identities as we will explain in the following. We are interested in the study of this technique because in our results we explore how it can be adapted to work with other NIZK proofs to improve their efficiency and we propose a codification for boolean CircuitSat.

In 2013 Gennaro et al [61] presented the first zk-SNARK with linear crs. The authors abstracted ideas from Groth [71] and defined two models of computation, *Quadratic Span Programs* (QSP) for boolean circuits and *Quadratic Arithmetic Programs* (QAP) for arithmetic circuits. A QAP, or a QSP, express a CircuitSat relation in terms of some polynomial identity. The name "Quadratic" comes because gate operations of a circuit can be expressed as quadratic equations. All these equations are compressed to a single divisibility relation of polynomials that is equivalent to the satisfiability of all the gates of the circuit.

The idea to make the proof succinct is that this divisibility relation is checked at one single point chosen by the setup algorithm. This is enough because the prover only knows this secret point in the source group of some bilinear group. If the divisibility relation of polynomials would not hold, the only way a cheating prover can still prove that the relation holds at the secret point is to know it in the field, which is hard because of the DLOG assumption.

The choice of codification for CircuitSat has a direct impact in the performance of the zk-SNARK. For this reason, some alternatives have been proposed. For example,

Square Span Programs (SSP) were presented by Danezis et al. in [45], which are a simplified version of QSP and in 2017 Groth and Maller [74] defined the analogous codification for arithmetic circuits, *Square Arithmetic Programs* (SAP). Both "Square" versions offer better efficiency in proof terms but roughly double the number of polynomials, which increases the crs size by approximately a factor of 2.

Quasi-Adaptive NIZK Arguments were introduced in 2013 by Jutla and Roy [85] for membership in linear spaces in $\mathbb{G}_1^m$ with $m \in \mathbb{N}$. *Quasi-adaptive* means the crs of the argument depends on the specific language we are proving membership to, which allows us to construct very efficient proofs under very weak assumptions. More technically, these NIZK proofs are constructed for a relation $\mathbf{R}_\rho$ chosen from a collection of relations $\{\mathbf{R}_\rho\}_{\rho\in\mathcal{D}}$ where the ρ parameter is chosen according to some distribution $\mathcal{D}$ and the crs is defined as a function of ρ. In the most important QA-NIZK constructions, ρ is a matrix $\mathbf{A}$ of group elements chosen according to a matrix distribution $\mathcal{M}$ that parametrizes the language of membership in the column-space of $\mathbf{A}$ in $\mathbb{G}_\iota$, $\mathcal{L}_\mathbf{A} = \{\boldsymbol{x} \in \mathbb{G}_\iota^m \text{ such that } \exists \boldsymbol{w}, \mathbf{A}\boldsymbol{w} = \boldsymbol{x}\}$. In addition, the relation $\mathbf{R}_\rho$ is *witness-sampleable*, if the parameter ρ can be sampled with a witness that is a valid parameter. In the previous example, the witness is a matrix with the DLOGs of the entries of $\mathbf{A}$.

In the most efficient instances, the proof consists of just one group element, like constructions in Jutla and Roy [86], and Kiltz and Wee [92] for linear spaces in $\mathbb{G}_1^m$, $m \in \mathbb{N}$ for witness-sampleable distributions. There is also a simulation sound construction for an unbounded number of queries in Kiltz and Wee [92] for the same language, that we use in our constructions. Other arguments have been developed in the QA-NIZK setting for different languages, like bilateral spaces (linear spaces in $\mathbb{G}_1^m \times \mathbb{G}_2^n$), the same opening language, or the set of integer commitments that open to bits, in González et al. [67].

Common applications of these constructions are shuffles and range proofs, like in González et al. [68]. Shuffle arguments are used to check that two collections of ciphertexts encrypt the same set of permuted plaintexts. It is very useful in electronic voting to check correct mixing of the votes. On the other hand, range arguments are proofs that guarantee some encrypted element belongs to a certain interval.

In our constructions we will use some of these QA-NIZK arguments as sub-proofs of our schemes, like the same opening argument, and we also prove new security properties of some of them.

Commitment schemes and NIZK arguments. A commitment scheme is a primitive where a party commits itself to a secret value, that can be revealed after some time. We can think of it as the cryptographic version of an envelope because this value is not

revealed until the party decides to open it. The *hiding* property ensures that no one can gain any knowledge of the value from the envelope. It also should be guaranteed that the opening value is the one chosen at the committing phase, the so-called *binding* property.

Commitment schemes are naturally used in NIZK proofs, where often the prover commits to the witness and gives the commitment along with a proof that the opening satisfies some relation. Moreover, if a party owns the extraction trapdoor of the commitment key, it can extract the witness, which is very suitable for the reduction in a knowledge sound proof. We refer to the *commit-and-prove* technique to designate those proofs that follow this strategy.

In this thesis, commitment schemes play a fundamental role. They are used as a building block in most of our results in NIZK arguments and they are the main subject of the results in Chapter 6.

1.3 Our results

We construct several NIZK arguments in bilinear groups building on the recent results on QA-NIZK arguments and zk-SNARKs. We can classify the contributions in three main directions.

1. We construct QA-NIZK arguments for a specific type of quadratic equations in a finite field under falsifiable assumptions. We improve on the state-oh-the-art in terms of communication complexity and crs size, in exchange for stronger assumptions.
    - In Chapter 3, we give three QA-NIZK arguments for different equations sub-types extending the techniques of the zk-SNARK for boolean circuit of [45] to work under new falsifiable assumptions.
    - In Chapter 6, we present an analogous result for general arithmetic circuits.
2. We strengthen the soundness of some previous notable NIZK arguments to make them simulation-extractable (SE) sound with minimal overhead:
    - In Chapter 4, we add SE to the González and Ràfols's QA-NIZK argument [69] for boolean CircuitSat, which is sub-linear in the circuit size under falsifiable assumptions. One of our main contributions is a new analysis of the Unbounded Simulation Sound arguments for membership in linear spaces of Kiltz and Wee [92] and the tight variant of Abe et al. [5], which might be of independent interest . We also give two Signatures of Knowledge constructed from them.
    - In Chapter 5, we add SE to the most efficient zk-SNARK in the literature, Groth16 [72]. We build on the work of Bowe and Gabizon [26], that has

the same crs size and prover complexity as Groth16, but it is proven secure in the RO (a part from the GGM inherited from Groth16). We give two constructions, where we reduce the verification to just one extra pairing respect to Groth16. In our second zk-SNARK, we avoid the use of a RO by making minimal changes to the crs and the verifier.

3. In Chapter 6 we define a new primitive, called *Somewhere Statistically Binding* commitments, that is a generalization of the Extended Multi-Pedersen commitments defined in [67]. We give the formal definitions and some applications in CircuitSat and Oblivious Database Queries. We also define *Functional Somewhere Statistically Binding commitment schemes*, which formalize the commitment scheme already used in our first result (Chapter 3) as an implicit technique to extract linear functions of the witness in the security proof.

In the following Sections 1.3.1 to 1.3.3 we give more detail of our contributions and finally, provide a list with the publications of this work.

1.3.1 Shorter QA-NIZK proofs for quadratic equations

Quadratic equations in a finite field appear naturally in cryptographic schemes, like shuffle and range arguments, and also they can be proven efficiently in bilinear groups. Often these equations are used to prove that certain committed value is a bit or, more generally a set of committed values opens to a bit-vector. We can prove membership in that language using the Groth-Sahai proof system [78] with constant size crs, commitment and proof sizes linear in the number of bits under very weak assumptions[1]. To the best of our knowledge, the only improvement is the argument in González and Ràfols[67], with a crs quadratic in the size of the bit-vector, constant proof and commitment linear in the size of the number of bits.

In Chapter 3, our main result is a QA-NIZK argument for l quadratic equations of the form $X_i(X_i - 2) = 0$, where X_i is a linear combination of n variables in $\mathbb{Z}_p$. In particular, with this argument, we can prove that a set of commitments opens to a bit-string. We use techniques inspired in the zk-SNARK for Square Span Programs (SSP, [45]) to express these equations to a single equation of polynomials and the argument results in the most efficient in proof and crs size based on falsifiable assumptions. We give two other arguments based on the main construction. The first one is a unit vector argument, that uses a weaker version of our assumption, where the bit-vector is a unit vector in a multi-dimensional space. The second argument is a generalization of the main argument, that proves a commitment opens to values that are in a set of size m in $\mathbb{Z}_p$, $\{z_1, \ldots, z_m\} \subset \mathbb{Z}_p$, instead of membership in the set $\{0, 2\}$ as the main argument.

[1] We think in commit-and-prove as was formalized by Escala and Groth [50]

We have some new applications: two shuffle arguments and one range argument.

Technically, the motivation of this work is to explore how efficient NIZK proofs based on falsifiable assumptions can be when one exploits the polynomial aggregation techniques of zk-SNARKs. By *polynomial aggregation* we refer to the technique of proving many equations together, by expressing them in a single divisibility relation as a SSP. If we think of the construction as a commit-and-prove argument, the "prove" part, that proves the divisibility relation holds, is constant-size. Since our assumption is falsifiable, to extract the whole witness in the security proof, we need a commitment linear in the witness size. This is because quadratic equations are NP-complete and Gentry-Wichs [63] states that any proof for an NP statement cannot be sub-linear in the witness size under falsifiable assumptions, unless some surprising results in complexity theory hold. We use a QA-NIZK argument for same opening [67] to prove the opening of the commitment is the witness of the SSP relation.

A common strategy to all soundness security proofs of commit-and-prove arguments, the commitment is extracted and the witness is used to break a hard problem with the "prove" part. In particular, Danezis et al.[45] use a non-falsifiable assumption to extract from a succinct commitment the whole witness in the field and use the "prove" part to break the q-TSDH assumption. We can extract from the linear commitment the whole witness but in the source group, so we need to use a different assumption. Still, we want a new assumption that is no too far from the q-TSDH, then we design a new one that is a generalization of the q-TSDH. For that, the reduction needs to extract some linear functions of the witness in the group to break it. Instead of doing itself, we use a strategy of González et al. [67] to make the adversary compute these linear functions and commit to them in a way the reduction can use it to break the new assumption.

Overall, we give some commit-and-prove arguments that allow us to have succinct proofs while avoiding the Gentry-Wichs impossibility result because the commitment is long. In some scenarios, like electronic voting, the commitment can be used in different proofs, so it is preferable to have the linear part of the proof in the "commit" and not in the "prove".

1.3.2 Simulation Extractability

As we explained above, simulation extractability (SE) is the strongest soundness notion (knowledge and simulation), where the adversary cannot come up with a fresh valid proof unless it knows a witness, even if it has seen an arbitrary number of simulated proofs. In the following two contributions, we adapt some arguments to satisfy this property.

Signatures of Knowledge for Boolean Circuits Under Standard Assumptions. In Chapter 4, we build a SE-NIZK for boolean CircuitSat based on the scheme of González and Ràfols [69], which is the first NIZK argument sub-linear in the circuit size under standard assumptions in bilinear groups. Their proof is linear in the depth and the secret input size of the circuit, so it is independent of the number of gates.

Several SE proofs are constructed using the traditional OR approach described in Groth [70]. Briefly, given some circuit and public input, a proof shows that either the circuit is satisfiable or a signature is known for some signature scheme specified in the crs. The honest prover gives a proof using its witness and the simulator uses the secret key of the signature as a trapdoor to sign. This approach changes the relation and the crs. On the contrary, we use a new strategy for simulation to achieve SE with a minimum overhead based on the structure of the scheme in [69].

The main idea of [69] is to prove the satisfiability of the circuit by giving a proof of knowledge of the secret input along with a proof of satisfiability of all gates in one level of the circuit. At each level, they prove separately quadratic (gate operations) and linear equations (correct wiring).

Our construction is based on the observation of that it is enough to have a simulation sound QA-NIZK for the linear equations to provide simulation-soundness to the whole proof.

González and Ràfols need to prove a stronger notion of soundness for linear equations. Intuitively, they want to prove that a QA-NIZK satisfies a "linear knowledge transfer". Technically, they use a QA-NIZK argument for membership in linear spaces such that the adversary cannot create a valid proof for statements $(\boldsymbol{x}^\top, \boldsymbol{y}^\top)^\top \in \mathrm{Im}(\mathbf{M}^\top, \mathbf{N}^\top)^\top$ where $\boldsymbol{x} = \mathbf{M}\boldsymbol{w}$ for some known $\boldsymbol{w}$, but $\boldsymbol{y} \neq \mathbf{N}\boldsymbol{w}$. The condition that $\boldsymbol{x} = \mathbf{M}\boldsymbol{w}$ for public value of $\boldsymbol{w}$ is called the *promise*, because it is never verified. When we use this argument as a sub-argument in our CircuitSat proof, the promise holds at some level because it is already proven in the previous levels, starting from the proof of knowledge of the input.

Concretely, we adapt the security proof of the unbounded simulation sound[2] argument of membership in linear spaces of Kiltz and Wee [92], which is the most efficient in the literature, to satisfy the stronger soundness notion related to promise problems, which is non-trivial.

As we mentioned, the simulation soundness property of our construction is derived from the USS property of our sub-argument for linear constraints. We also prove that when the USS QA-NIZK is tight, this property is also extended to the whole proof. We adapt the security proof of the tightest USS QA-NIZK in the literature [5] to work with promise problems resulting in a tight Signature of Knowledge for boolean circuits

[2]Unbounded refers to the number of queries of the adversary to the simulator.

under falsifiable assumptions. A Signature of Knowledge is a generalization of digital signatures where one can sign the message just if it has a valid witness for membership in the language.

As we explained in Section 1.2, QSP and SSP were presented as models of computation for boolean CircuitSat. As Groth observed in [72], QAP, originally defined for arithmetic CircuitSat, can be also used to express boolean CircuitSat. Although, this transformation is not hard to do, there are several possible ways. We propose a simple, canonical transformation.

Simulation Extractable Versions of Groth's zk-SNARK Reviseted. Since zk-SNARKs are widely developed and used in practice, and efficiency is extremely important, in our next result we accept to work with non-falsifiable assumptions. In Chapter 5, we improve the soundness notion of Groth16 [72], the most efficient zk-SNARK in terms of proof size and verification cost. However, Groth and Maller [74] showed the proof is malleable, so an adversary who has access to a valid proof can modify it and obtain very easily a fresh proof that will pass the verification.

There have been many efforts to make Groth16 SE to avoid malleability. The Bowe and Gabizon's approach [26] has the most efficient prover and crs size, while it adds Random Oracle (RO). Atapoor and Baghery's approach [11] reduces in 1 pairing the verification cost in comparison to other approaches, but it increases the crs and the prover's work considerably. Another construction given by Lipmaa [102] adds just one element in the proof respect to Groth16, but it increases the other complexities.

We construct two zk-SNARKs with the same verification as Atapoor and Baghery [11]. The first construction has the same crs and proof size as [26], and requires 4 pairings in the verification as [11], it just increases 1 pairing compared to Groth16. In the second construction, we get rid of the RO by using a collision-resistant hash function and adding one element in the crs and one exponentiation in the target group in the verification. Then, the second approach is proven directly in the GGM as the original Groth16.

Both constructions follow Bowe and Gabizon's approach, that randomizes one parameter of Groth16's zk-SNARK and the prover gives the modified proof along with a proof of knowledge (PoK) of the randomization factor. We observe that in the GGM it is enough to replace the PoK with a variation of Boneh-Boyen's signature. This is possible because the proof of knowledge of the randomization factor never needs to be simulated for proving ZK of the zk-SNARK.

1.3.3 Somewhere Statistically Binding Commitment Schemes

In Chapter 6, we define a new primitive called *Somewhere Statistically Binding* (SSB) commitment scheme, which is a generalization of dual-mode commitments of vectors. In this primitive, the commitment key sets some indices where the coordinates are statistically binding and computationally hiding, and the remaining coordinates are statistically hiding and computationally binding. The set of indices where statistically binding holds is predetermined but only known by the commitment key generator.

The interest of this primitive is that it appears naturally in commit-and-prove arguments for NP languages. Indeed, in the falsifiable setting, the best we can hope for is that the "prove part" is constant and the "commit" part is linear in the witness size. In this sense, we can think of the "prove" part as a second commitment to the witness that is succinct. Therefore, the information that we can extract of the witness from the "prove" part (under falsifiable assumptions) can only be constant for information-theoretic reasons. The challenge for constructing the security proof is to combine this constant size information of the witness with the whole witness extracted from the "commit" part to break a hard problem. In González et al. [67] construction it is enough to extract just one coordinate of the shrinking commitment. Our generalization allows extraction[3] of a set of coordinates of arbitrary size. This primitive provides additional flexibility that can be useful to apply this strategy to other contexts.

We formally define SSB commitment schemes and prove that they cover as a special case the Extended Multi-Pedersen (EMP) commitments introduced in González et al. [67, 68].

We also give a characterization of Algebraic Commitment Schemes, where commitment keys are matrices, proving that the key matrices' distributions define the SSB properties at each coordinate. We also show that the SSB properties can be expressed in terms of membership in linear spaces.

Moreover, we introduce another primitive called *Functional SSB commitment* scheme, which is a generalization of an SSB commitment scheme where the extraction key instead of recovering certain coordinates of the committed vector, returns some functions of the committed vector. The set of functions is defined when the commitment key is set up. This primitive for a family of linear functions was already used in our first result Chapter 3 as a technique in the security proof to extract linear functions of the witness computed by the adversary.

We have some new applications of Functional SSB commitments:

- A QA-NIZK Argument for SAP relations, which is a generalization of our result in Chapter 3,
- Two efficient applications in Oblivious transfer.

[3] F-extraction for a chosen function F.

1.3.4 List of Publications

[46] *Shorter quadratic QA-NIZK proofs*, V. Daza, A. González, Z. Pindado, C. Ràfols, J. Silva. IACR *International Workshop on Public Key Cryptography 2019*, 314-343.

[14] *Signatures of Knowledge for Boolean Circuits Under Standard Assumptions*, K. Baghery, A. González, Z. Pindado and C. Ràfols. *International Conference on Cryptology in Africacrypt 2020*, 24-44.

[16] *Simulation Extractable Versions of Groth's zk-SNARK Revisited*, K. Baghery, Z. Pindado and C. Ràfols. *International Conference on Cryptology and Network Security 2020*.

Somewhere Statistically Binding Commitments, P. Fauzi, H. Lipmaa, Z. Pindado and J. Siim. Accepted for publication in the *Financial Cryptography 2021*.

Manuscript

Signatures of Knowledge for Boolean Circuits Under Standard Assumptions, K. Baghery, A. González, Z. Pindado and C. Ràfols.

Chapter 2

Preliminaries

In this chapter we establish the notation used in the thesis, define the circuit satisfiability problem, bilinear groups and non-interactive zero-knowledge proofs. We briefly explain the idealized models of computation that are used to analyse security of the schemes, as well as the assumptions. Finally, we present some concrete schemes that we use in our constructions as building blocks.

2.1 Notation and Preliminaries

2.1.1 Algorithms, functions and probabilities notions

Let λ be the security parameter, an integer value that parametrizes cryptographic schemes and the involved parties. We consider the running-time of any adversary and the probability of its success as functions of this parameter.

Algorithms and functions

We write $y \leftarrow S$ for sampling y uniformly at random from the set S.

For an algorithm $\mathcal{A}$, let $\mathbf{Im}(\mathcal{A})$ be the image of $\mathcal{A}$, i.e., the set of valid outputs of $\mathcal{A}$. By $y \leftarrow \mathcal{A}(x; r)$ we denote the fact that $\mathcal{A}$, given an input x and a randomizer r, outputs y. For algorithms $\mathcal{A}$ and $\mathcal{E}_{\mathcal{A}}$, we write $(y \parallel y') \leftarrow (\mathcal{A} \parallel \mathcal{E}_{\mathcal{A}})(x; r)$ as a shorthand for "$y \leftarrow \mathcal{A}(x; r)$, $y' \leftarrow \mathcal{E}_{\mathcal{A}}(x; r)$".

All algorithms by default are considered probabilistic, or randomized. Let $\mathsf{RND}_\lambda(\mathcal{A})$ denote the random tape of an algorithm $\mathcal{A}$, it is given in addition to its input and contains the uniformly distributed bits that $\mathcal{A}$ can use in its execution.

All adversaries will be stateful, i.e. the algorithms they use are capable of storing information seen previously.

Let PPT denote probabilistic polynomial-time, and NUPPT denote non-uniform PPT. NUPPT is a stronger model where the algorithm is given some advice depending on the length of the input. Then, we have more confidence if a NUPPT algorithm cannot do some computation.

We denote by $poly(\lambda)$ an arbitrary polynomial function in λ and by $negl(\lambda)$ an arbitrary negligible function in λ, that is a function asymptotically smaller than any inverse polynomial function in λ.

Oracle access. An oracle is an ideal black-box machine that receives inputs and returns some output. If an algorithm is given access to an oracle, it means this algorithm gets access to the answers of some queries without specifying how the answers are computed. We denote by $\mathcal{A}^{\mathcal{O}}(x)$ the fact that the algorithm $\mathcal{A}$ on input x is given oracle access to $\mathcal{O}$.

Asymptotic notation. We write $f(x) = \mathcal{O}(g(x))$ to express that when x goes to infinity, the function g bounds the function f.

If a function $f(x)$ verifies $|f(x)| = \mathcal{O}\left(\frac{1}{p(x)}\right)$ for all polynomial $p(x)$, this function is called *negligible*. A function f is called *overwhelming* if $1 - f$ is negligible.

Given a family of computational problems parametrized by λ, informally we say that It is hard to solve a problem of size λ, when the probability to find a solution to this problem is negligible in terms of the security parameter λ.

Functions f, g are negligibly close, denoted $f \approx_\lambda g$, if $|f - g| = negl(\lambda)$. In case of negligible functions, we write $f \approx_\lambda 0$, or just $f \approx 0$.

Computational indistinguishability. Given two probability distributions X and Y, we say they are computationally indistinguishable, which is noted by $X \approx Y$, if no adversary is able to decide whether a given string s is sampled from distribution X or Y. Formally, for all PPT adversary $\mathcal{A}$, $|\Pr[s \leftarrow X : \mathcal{A}(s) = 1] - \Pr[s \leftarrow Y : \mathcal{A}(s) = 1]| \leq negl(\lambda)$.

Advantages and winning experiments. We denote that an adversary $\mathcal{A}$ wins the experiment E by $\mathcal{A}(E) = 1$. The probability of the adversary wins an experiment E is the *advantage* and we denote it by $\mathsf{Adv}_E(\mathcal{A})$, i.e. $\mathsf{Adv}_E(\mathcal{A}) = \Pr[\mathcal{A}(E) = 1]$.

2.1.2 Arithmetic

Sets. We denote by $\mathbb{N}$ the set of the natural numbers and $\mathbb{Z}$ the set of integers. We denote by $\{0,1\}^n$ the set of bit-strings of length n, for a positive integer n, and $\{0,1\}^*$ the set of all bit-strings. The *order* of a set is the number of elements that belong to this set.

Groups

Let $\mathbb{F}_p$ be a finite field of order p, the set $\mathbb{F}_p \setminus \{0\}$ is an *abelian group* with the associated binary operation $\cdot$, i.e. a commutative group. Mostly, we use $\mathbb{Z}_p$ as the finite field with p prime and denote by $\mathbb{G} = \mathbb{Z}_p \setminus \{0\}$ the group.
For a group $\mathbb{G}$, we denote by $\mathbb{G}^*$ the set of all the elements in $\mathbb{G}$ but the identity element.

Cyclic groups. A *cyclic group* is a group $\mathbb{G}$ that can be generated by a single element, called *generator*. If $\mathbb{G}$ is a group of order N and $\mathcal{P}$ a generator of $\mathbb{G}$, then $\{\mathcal{P}, 2\mathcal{P}, \ldots, N\mathcal{P}\}$ is all $\mathbb{G}$. Equivalently, given a generator $\mathcal{P}$, for all $[\mathcal{Q}] \in \mathbb{G}$ there exists a unique $\alpha \in \mathbb{Z}_N$ such that $\alpha\mathcal{P} = [\mathcal{Q}]$.

Note that we are using additive notation because it is more convenient when we work with vectors and matrices of elements in the group. In the cryptographic literature, often multiplicative notation is used to express elements in a group where g is the generator we write g^α, and the group is generated by exponentiating the generator: $\{g, g^2, \ldots, g^N\} = \mathbb{G}$.

Matrices and vectors notation

$\mathbf{I}_n$ refers to the identity matrix in $\mathbb{Z}_p^{n\times n}$, $\mathbf{0}_{m\times n}$ refers to the all-zero matrix in $\mathbb{Z}_p^{m\times n}$, and $\boldsymbol{e}_i^n$ the ith element of the canonical basis of $\mathbb{Z}_p^n$ (simply $\mathbf{I}$, $\mathbf{0}$, and $\boldsymbol{e}_i$, respectively, if n, m are clear from the context).

2.1.3 Lagrange Interpolation

Given a set $\mathcal{R} = \{r_1, \ldots, r_n\} \subset \mathbb{Z}_p$, $r_i \neq r_j$ for all $i \neq j$, we denote by $\lambda_i(X) = \prod_{j\neq i} \frac{(X - r_i)}{(r_j - r_i)}$ the ith Lagrange basis polynomial associated to $\mathcal{R}$.

The linear combination of these polynomials with coefficients $\{y_1, \ldots, y_n\} \subset \mathbb{Z}_p$ is the Lagrange Interpolation polynomial $\sum_{i=1}^n y_i\lambda_i(X)$, a degree $n-1$ polynomial that passes through all the points $(r_1, y_1), \ldots, (r_n, y_n)$.

2.2 Circuit Satisfiability problem

In computational complexity theory, a decision problem is a question whose answer is yes or no. We consider those problems where possible inputs are a set of binary strings or strings over a finite field. The subset of strings for which the problem returns yes is a formal language.

We are interested in the *Circuit Satisfiability* (CircuitSat) problem, a decision problem that determines whether a given circuit has an assignment of its inputs that makes the output true. We briefly define circuits and the CircuitSat problem in the following.

2.2.1 Circuits

A circuit is a directed acyclic graph that consists in a set of wires and gates with a set of equations relating the inputs and outputs of the gates.

There are two types of circuits, *boolean* and *arithmetic*, depending on the data type of the inputs, bits or elements in a field $\mathbb{F}$, and the type of operations in the gates, binary operations or addition and multiplication in $\mathbb{F}$, respectively.

Formally, a *boolean circuit* is a function $C : \{0,1\}^{n_0} \to \{0,1\}^{\ell}$, where n_0 is the number of inputs and ℓ the number of outputs. An *arithmetic circuit* over a field $\mathbb{F}$ is a function $C : \mathbb{F}^{n_0} \to \mathbb{F}^{\ell}$, where n_0 is the number of inputs and ℓ the number of outputs.

2.2.2 CircuitSat

We define in the following the Circuit Satisfiability (CircuitSat) problem for boolean and arithmetic circuits. A boolean circuit C is satisfiable if and only if there exists an assignment of inputs that makes the output of the circuit true. An arithmetic circuit is satisfiable if and only if for an assignment of inputs and outputs the circuit is correctly computed.

We formalize the satisfiability definition by a circuit checker function associated to the circuit that outputs $0/1$ if the circuit is satisfiable. Let G_i be the operation of the ith gate of the circuit. We denote the assignment of the input by $\boldsymbol{x} \in \{0,1\}^{n_0+n}$, including the input values, intermediate values that are the output of the gates and the output values.

Definition 1. *Let $C : \{0,1\}^{n_0} \to \{0,1\}^{\ell}$ be a boolean circuit with n gates. Given an input $(x_1, \ldots, x_{n_0})$, a* circuit checker *is a function $G : \{0,1\}^{n_0+n} \to \{0,1\}$ associated to C such that $G(x_1, \ldots, x_{n_0+n}) = 1$ if and only if $C(x_1, \ldots, x_{n_0}) = 1$ and x_{n_0+i} is the output of the ith gate when C is evaluated on input $(x_1, \ldots, x_{n_0})$ for all $i = 1, \ldots, n$.*

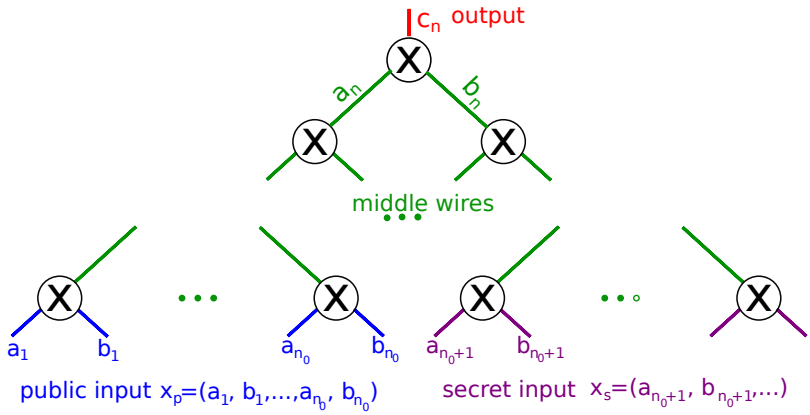

Figure 2.1: Circuit sketch

Analogously, we define the satisfiability of an arithmetic circuit.

Definition 2. *Let $C : \mathbb{F}^{n_0} \to \mathbb{F}^{\ell}$ be an arithmetic circuit with n gates. Given an input $(x_1, \ldots, x_{n_0})$ and output $(x_{n-\ell+1}, \ldots, x_n)$, a* circuit checker *is a function $G : \mathbb{F}^{n_0+n} \to \{0,1\}$ associated to C such that $G(x_1, \ldots, x_{n_0+n}) = 1$ if and only if $C(x_1, \ldots, x_{n_0}) = (x_{n-\ell+1}, \ldots, x_n)$ and x_{n_0+i} is the output of the ith gate when C is evaluated on input $(x_1, \ldots, x_{n_0})$ for all $i = 1, \ldots, n$.*

In Figure 2.1 we give a sketch of a circuit where we show the graph with the input wires, middle wires and output wires. We split the vector into public and secret values, $\boldsymbol{x} = (\boldsymbol{x}_p, \boldsymbol{x}_s)$, because in most cases the public values are given in the statement. Given some $\boldsymbol{x}_p$, the prover shows there exists some $\boldsymbol{x}_s$ such that the gate checker is satisfied. We write CircuitSat as a system of equations with variables $\boldsymbol{a}$, $\boldsymbol{b}$, $\boldsymbol{c}$ for left, right and output, wires respectively. The operations in the gates depend on the circuit, we use here the operator symbol $\otimes$ to indicate some boolean gate when the circuit is boolean and a multiplication gate when the circuit is arithmetic

Note that for any circuit C with more than one output, we can always construct an equivalent circuit C' with enough additional gates that has just one output gate indicating the satisfiability of the circuit with $0/1$. This output value should be the same as the circuit checker output. Then, we assume w.l.o.g. that circuits have one single output which is a bit.

Now, we give the formal definition of an NP language to define CircuitSat formally.

Definition 3. *A language $\mathcal{L}$ is in $\mathcal{NP}$ if there exists a Boolean relation $\mathbf{R}_{\mathcal{L}} \subset \{0,1\}^* \times \{0,1\}^*$ and a polynomial p such that $\mathbf{R}_{\mathcal{L}}$ can be recognized in deterministic polynomial time and $x \in \mathcal{L}$ if and only if there exists a w such that $|w| \leq p(|x|)$ and $(x,w) \in \mathcal{R}_{\mathcal{L}}$. We call w the* witness for membership *of $x \in \mathcal{L}$.*

A CircuitSat problem is an NP language that we write formally:

$$\mathcal{L}_C = \{\boldsymbol{x}_p : \exists \boldsymbol{w} = (\boldsymbol{x}_s, x_{n_0+1}, \ldots, x_{n_0+n}), \text{ such that } G(\boldsymbol{x}_p, \boldsymbol{w}) = 1\},$$

where G is the circuit checker of the circuit C, so it outputs 1 if and only if the circuit is satisfied in the terms of the above definitions respect to $\boldsymbol{w}$.

The boolean satisfiability problem is NP-complete by the Cook-Levin Theorem ([39], [95]). That is, all NP boolean problems can be reduced to a circuit satisfiability problem, however this does not imply the reduction is concretely efficient. Given an instantiation x of a problem $\mathcal{L}'$, it could be translated to a CircuitSat instantiation, $f(x)$, for some efficient function f, such that $x \in \mathcal{L}'$ if and only if $f(x) \in \mathcal{L}_C$.

2.3 Bilinear Groups and Implicit Notation

Bilinear Groups were introduced in 1993 for breaking elliptic curve schemes, concretely using the Weil and Tate pairings [118]. Although they were initially used for cryptanalysis, in 2000 they started being used for designing new primitives and improve schemes like Identity-Based Encryption [84], Signatures and much more. They have a very rich structure that allows to design very efficient primitives.

In this section we define bilinear groups and establish the notation that we use to designate the elements in those groups.

2.3.1 Definition

Definition 4. *A* bilinear group *is a tuple $gk := (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, \mathcal{P}_1, \mathcal{P}_2)$, where $\mathbb{G}_1, \mathbb{G}_2$ and $\mathbb{G}_T$ are additive groups of prime order p, the elements $\mathcal{P}_1, \mathcal{P}_2$ are generators of $\mathbb{G}_1, \mathbb{G}_2$ respectively, $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ is an efficiently computable, non-degenerate bilinear map, and there is no efficiently computable isomorphism between $\mathbb{G}_1$ and $\mathbb{G}_2$.*

We may refer to the e above as a *bilinear map* or *pairing* indistinguishably. By definition, $e(\mathcal{P}_1, \mathcal{P}_2)$ is a generator of $\mathbb{G}_T$.

Galbraith et al. in [59] give a classification of bilinear groups of these three types:

- Type I: Symmetric bilinear groups. When $\mathbb{G}_1 = \mathbb{G}_2$, we denote $\mathbb{G}_1$ and $\mathbb{G}_2$ as $\mathbb{G}$ and the bilinear group as $(N, \mathbb{G}, g, \mathbb{G}_T, e)$.

- Type II: Asymmetric bilinear groups for which there exists an efficiently-computable homomorphism $\psi : \mathbb{G}_2 \to \mathbb{G}_1$, but its inverse is hard to compute.

- Type III: The asymmetric bilinear groups for such homomorphism is not known.

We are more interested in type III because asymmetric bilinear groups are more efficient in practice than the other types. Elliptic curves are the only instantiations of these groups, and they are commonly used in cryptography. All the constructions in this thesis are defined over Bilinear groups. We give some of the assumptions used to prove the constructions in section 2.4.

Note that N can be a prime or composite number hard to factor. In this thesis, we consider prime order bilinear groups because the computations of pairings are more efficient.

We use implicit (or additive) notation explained in the following that is taken from elliptic curves where the elements of the group are points and the operation is the addition of points.

2.3.2 Implicit notation

Elements in $\mathbb{G}_\iota$ are denoted implicitly as $[a]_\iota := a\mathcal{P}_\iota$, where $\iota \in \{1, 2, T\}$ and $\mathcal{P}_T := e(\mathcal{P}_1, \mathcal{P}_2)$. For simplicity, we often write $[a]_{1,2}$ for the pair $[a]_1, [a]_2$. The pairing operation will be written as a product $\cdot$ or any of the forms: $[a]_1 \cdot [b]_2 = [a]_1[b]_2 = e([a]_1, [b]_2) = [ab]_T$.

Bilinearity. By *bilinearity*, for all $\alpha, \beta \in \mathbb{Z}_N$, $[a]_1 \in \mathbb{G}_1, [b]_2 \in \mathbb{G}_2$ it holds that $e(\alpha[a]_1, \beta[b]_2) = e(\alpha\beta[a]_1, [b]_2) = e([a]_1, \alpha\beta[b]_2) = \alpha\beta \cdot e([a]_1, [b]_2)$.

Vectors and matrices notation. Vectors and matrices are denoted in boldface. Given a matrix $\mathbf{T} = (t_{i,j})$, $[\mathbf{T}]_\iota$ is the natural embedding of $\mathbf{T}$ in $\mathbb{G}_\iota$, that is, the matrix whose (i, j)th entry is $t_{i,j}\mathcal{P}_\iota$. We denote by $|\mathbb{G}_\iota|$ the bit-size of the elements of $\mathbb{G}_\iota$ and by $(\cdot, \cdot)$ the bit-size of elements in $\mathbb{G}_1$ and $\mathbb{G}_2$ in each component.

We write the pairing operation for matrices and vectors in groups writing $[\mathbf{M}_1]_1 \cdot [\mathbf{M}_2]_2 = [\mathbf{M}_1\mathbf{M}_2]_T$ for any compatible matrices $\mathbf{M}_1$ and $\mathbf{M}_2$.

2.4 Computational Assumptions

In this section we introduce the standard assumptions DLOG and DDH, some matrix assumptions that we use in our work including some q-assumptions and knowledge assumptions.

2.4.1 Standard Assumptions

We give the definitions of two standard assumptions, the Discrete Logarithm (DLOG) Assumption and the Decisional Diffie-Hellman (DDH) assumption, that is stronger and relies on it. They are the basis of the assumptions presented in the following.

Discrete Logarithm Assumption

DLOG. Let $\mathbb{G}$ a cyclic group, remember that for all $[\mathcal{Q}] \in \mathbb{G}$ there exists a unique $\alpha \in \mathbb{Z}_N$ such that $\alpha\mathcal{P} = [\mathcal{Q}]$. This α is called the *Discrete Logarithm of* $[\mathcal{Q}]$ respect to $\mathcal{P}$, we denote it as $\text{DLOG}_{\mathcal{P}}[\mathcal{Q}]$.

The *Discrete Logarithm Problem* in a cyclic group $\mathbb{G}$ with generator $\mathcal{P}$ is to compute $\text{DLOG}_{\mathcal{P}}[\mathcal{Q}]$ for a uniform element $[\mathcal{Q}] \in \mathbb{G}$.

Definition 5 (Discrete Logarithm Assumption)**.** *The Discrete Logarithm (DLOG) assumption holds relative to $\mathcal{G}$, if for all PPT adversary $\mathcal{A}$,*

$$\Pr\left[\ (\mathbb{G}, N, \mathcal{P}) \leftarrow \mathcal{G}(1^\lambda); [\mathcal{Q}] \leftarrow \mathbb{G}; \alpha \leftarrow \mathcal{A}(\mathbb{G}, N, \mathcal{P}, [\mathcal{Q}]) : [\mathcal{Q}] = \alpha\mathcal{P}\ \right] \approx 0.$$

Decisional and Computational Diffie-Hellman Assumption. The *Decisional Diffie-Hellman* (DDH) problem relative to $\mathcal{G}$, given a tuple $(\mathcal{P}, a\mathcal{P}, b\mathcal{P}, [\mathcal{Q}])$ where $\mathcal{P}$ is a generator of a cyclic group $\mathbb{G} \leftarrow \mathcal{G}(1^\lambda)$ of order N, $a, b \in \mathbb{Z}_N$ and $[\mathcal{Q}] \in \mathbb{G}$, is to distinguish if $[\mathcal{Q}]$ has been generated computing $[\mathcal{Q}] = ab\mathcal{P}$ or if $[\mathcal{Q}]$ is generated uniformly at random in the group ($[\mathcal{Q}] \leftarrow \mathbb{G}$).

We give the definition of DDH assumption in the following. To denote the adversary receives the tuple $(\mathcal{P}, a\mathcal{P}, b\mathcal{P}, \mathcal{Q})$ and wins in the sense it guesses correctly the last element is a uniformly sampled element $\mathcal{Q}$ in $\mathbb{G}$, or it is generated by computing $ab\mathcal{P}$, we write $\mathcal{A}(\mathcal{P}, a\mathcal{P}, b\mathcal{P}, [\mathcal{Q}]_T) = 1$.

Definition 6 (Decisional Diffie-Hellman Assumption)**.** *The* Decisional Diffie-Hellman *(DDH) assumption holds relative to $\mathcal{G}$ if, for all probabilistic polynomial time adversary $\mathcal{A}$,*

$$|\Pr[\mathcal{A}(gk, a\mathcal{P}, b\mathcal{P}, ab\mathcal{P}) = 1)] - \Pr[\mathcal{A}(gk, a\mathcal{P}, b\mathcal{P}, \mathcal{Q}) = 1]| \approx 0,$$

where the probability is taken over $gk = (N, \mathcal{P}, \mathbb{G}, \mathbb{G}_T, e) \leftarrow \mathcal{G}(1^\lambda)$, $a, b \leftarrow \mathbb{Z}_N$, $[\mathcal{Q}] \leftarrow \mathbb{G}$ *and the coin tosses of adversary* $\mathcal{A}$.

The computational version of the problem is the *Computational Diffie-Hellman* (CDH) that, relative to $\mathcal{G}$, given a tuple $(\mathcal{P}, a\mathcal{P}, b\mathcal{P})$ where $\mathcal{P}$ is a generator of a cyclic group $\mathbb{G} \leftarrow \mathcal{G}(1^\lambda)$ of order N, $a, b \in \mathbb{Z}_N$, consists of computing $ab\mathcal{P}$. If a, b are sampled uniformly, by the DLOG assumption it is hard to compute such an element $ab\mathcal{P}$ without knowing a or b.

Note that if we work with symmetric bilinear pairings, the decisional problem is easy because if we know $a\mathcal{P}$ and $b\mathcal{P}$ with $\mathcal{P} \in \mathbb{G}$ a generator of the group and a symmetric bilinear pairing $e : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$, we can compute $e(a\mathcal{P}, b\mathcal{P}) = e(\mathcal{P}, ab\mathcal{P})$ and compare it to $e(\mathcal{P}, \mathcal{Q})$. In bilinear case, we use analogous problems based on the DH problem for bilinear groups that are hard even with symmetric bilinear groups, for example the 2-LIN assumption introduced by Boneh et al. [23]. This assumptions states that the following tuples are computational indistinguishable $([a_1], [a_2], [a_1 r_1], [a_2 r_2], [r_1 + r_2]) \approx ([a_1], [a_2], [a_1 r_1], [a_2 r_2], [z])$, where $a_1, a_2, r_1, r_2, z \leftarrow \mathbb{Z}_p$, and it is generalized to k-LIN assumptions where 1-LIN=DDH ([81]).

2.4.2 Matrix Assumptions

We present the generalization of DDH to group matrix distributions defined in [52]. Let $\ell, k \in \mathbb{N}$ be the parameters that define the dimensions of the matrices.

Definition 7. *We call* $\mathcal{D}_{\ell,k}$ *a* matrix distribution *if it outputs (in PPT time, with overwhelming probability) matrices in* $\mathbb{Z}_p^{\ell \times k}$. *We define* $\mathcal{D}_k := \mathcal{D}_{k+1,k}$.

The following applies for $\mathbb{G}_\iota$, where $\iota \in \{1, 2\}$.

Assumption 1 (Matrix Decisional Diffie-Hellman Assumption in $\mathbb{G}_\iota$ [52])**.** *For all non-uniform PPT adversaries* $\mathcal{A}$,

$$|\Pr[\mathcal{A}(gk, [\mathbf{A}]_\iota, [\mathbf{A}\boldsymbol{w}]_\iota) = 1] - \Pr[\mathcal{A}(gk, [\mathbf{A}]_\iota, [\boldsymbol{z}]_\iota) = 1]| \approx 0,$$

where the probability is taken over $gk \leftarrow \mathcal{G}(1^\lambda)$, $\mathbf{A} \leftarrow \mathcal{D}_{\ell,k}, \boldsymbol{w} \leftarrow \mathbb{Z}_p^k, [\boldsymbol{z}]_\iota \leftarrow \mathbb{G}_\iota^\ell$ *and the coin tosses of adversary* $\mathcal{A}$.

Intuitively, the $\mathcal{D}_{\ell,k}$-MDDH assumption means that it is hard to decide whether a vector is in the image space of a matrix or it is a random vector, where the matrix is drawn from $\mathcal{D}_{\ell,k}$. We use the following matrix distributions:

$$\mathcal{L}_k : \mathbf{A} = \begin{pmatrix} a_1 & 0 & \dots & 0 \\ 0 & a_2 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & a_k \\ 1 & 1 & \dots & 1 \end{pmatrix}, \qquad \mathcal{U}_{\ell,k} : \mathbf{A} = \begin{pmatrix} a_{1,1} & \dots & a_{1,k} \\ \vdots & \ddots & \vdots \\ a_{\ell,1} & \dots & a_{\ell,k} \end{pmatrix},$$

where $a_i, a_{i,j} \leftarrow \mathbb{Z}_p$. The $\mathcal{L}_k$-MDDH Assumption is the k-linear family of Decisional Assumptions and corresponds to the Decisional Diffie-Hellman (DDH) Assumption in $\mathbb{G}_\iota$ when $k = 1$. The SXDH Assumption states that DDH holds in $\mathbb{G}_\iota$ for all $\iota \in \{1, 2\}$. The $\mathcal{U}_{\ell,k}$-MDDH assumption is the *Uniform* Assumption and is the weakest of all matrix assumptions of size $\ell \times k$.

Additionally, we use the family of computational assumptions called *Kernel Diffie-Hellman* (KerDH) Assumptions in $\mathbb{G}_\iota$ and its analogue for asymmetric bilinear groups *Split Kernel Diffie-Hellman* Assumption. The KerMDH was presented in [104] as a natural computational analogue of the MDDH assumption, which generalizes different assumptions already used for particular cases. The problem behind this assumption is given $[\mathbf{A}]_\iota$ to find a non-zero vector which image by the matrix $\mathbf{A}$ is the vector zero. Actually, it is the right computational analogue of the MDDH assumption in the sense that, given a distribution $\mathcal{D}_{\ell,k}$, the $\mathcal{D}_{\ell,k}$-KerMDH is implied by the $\mathcal{D}_{\ell,k}$-MDDH, since a solution to the kernel allows to decide membership in $\mathbf{Im}([\mathbf{A}]_\iota)$.

Assumption 2 (Kernel Diffie-Hellman Assumption in $\mathbb{G}_\iota$ [104])**.** *For all non-uniform PPT adversaries $\mathcal{A}$:*

$$\Pr\left[[\boldsymbol{x}]_{3-\iota} \leftarrow \mathcal{A}(gk, [\mathbf{A}]_\iota) : \boldsymbol{x} \neq 0 \wedge \boldsymbol{x}^\top \mathbf{A} = \mathbf{0}\right] \approx 0,$$

where the probability is taken over $gk \leftarrow \mathcal{G}(1^\lambda)$, $\mathbf{A} \leftarrow \mathcal{D}_{\ell,k}$ and the coin tosses of adversary $\mathcal{A}$.

In asymmetric bilinear groups, there is a natural variant of this assumption, that is the *Split Kernel Diffie-Hellman* Assumption.

Assumption 3 (Split Kernel Diffie-Hellman Assumption [67])**.** *For all non-uniform PPT adversaries $\mathcal{A}$:*

$$\Pr\left[[\boldsymbol{r}]_1, [\boldsymbol{s}]_2 \leftarrow \mathcal{A}(gk, [\mathbf{A}]_{1,2}) : \boldsymbol{r} \neq \boldsymbol{s} \wedge \boldsymbol{r}^\top \mathbf{A} = \boldsymbol{s}^\top \mathbf{A}\right] \approx 0,$$

where the probability is taken over $gk \leftarrow \mathcal{G}(1^\lambda)$, $\mathbf{A} \leftarrow \mathcal{D}_{\ell,k}$ and the coin tosses of adversary $\mathcal{A}$.

While the KerMDH Assumption says one cannot find a non-zero vector in one of the groups which is in the kernel of $\mathbf{A}$, the split assumption says one cannot find different vectors in $\mathbb{G}_1^\ell \times \mathbb{G}_2^\ell$ such that the difference of the vector of their discrete logarithms is in the co-kernel of $\mathbf{A}$.

2.4.3 q-type Assumptions

A q-type assumption is a family of assumptions that depend on a parameter q, that commonly is the number of queries to an oracle. For example, in signatures schemes

it is the number of queries of a signature scheme, in zk-SNARKs it is the number of powers of a secret point. The dependency on this parameter is a drawback in the design of schemes because the security of the scheme depends on the number of queries the adversary does, which is a weakness. For example, Cheon in [37] shows that when q holds some particular relation with the order of the group, the complexity of the scheme that relies on the q-assumption is reduced by $O(\sqrt{q})$. However, they are very useful in the works that we take as a reference and some times it is preferable for efficiency to use them in our constructions. Moreover, the constant case of these assumptions, i.e. when $q = 1$, usually is weaker than those with $q > 1$, because they do not have this dependency on a parameter of the scheme.

We recall the q-Target Strong Diffie-Hellman assumption, that we refer to in our results in Chapters 3 and 6. This assumption essentially says that the inversion operation is hard in the exponent, even given q powers of the element to invert.

Assumption 4 (***q***-Target Strong Diffie-Hellman Assumption, ***q***-TSDH [21])**.** *For all non-uniform PPT adversaries $\mathcal{A}$:*

$$\Pr\left[(r, [\nu]_T) \leftarrow \mathcal{A}(gk, \{[s^i]_{1,2}\}_{i=1}^{q}) : \nu = \frac{1}{s-r}\right] \approx 0,$$

where the probability is taken over $gk \leftarrow \mathcal{G}(1^\lambda)$, $s \leftarrow \mathbb{Z}_p$ and the coin tosses of adversary $\mathcal{A}$.

2.4.4 Knowledge Assumptions

Knowledge assumptions are a family of non-falsifiable assumptions because they assume if the adversary comes out with some elements that satisfy a desired condition, it has to know something. For example, the coefficients used to output a linear combination of the input elements. In other words, they assume which strategy the adversary used to create its output. This is formalized for each such adversary assuming the existence of an extractor that gives this knowledge to the adversary.

The Knowledge of Exponent Assumption (KEA) is a basic example presented by Damgård in [42]. KEA establishes that given the tuple $(\mathcal{P}, \alpha\mathcal{P})$ as input, it is infeasible to output $([c], \alpha[c])$ without knowing the $\text{DLOG}_{\mathcal{P}}[c] = \text{DLOG}_{\alpha\mathcal{P}}(\alpha[c])$, i.e. an element a such that $[c] = a\mathcal{P}$ and $\alpha[c] = \alpha a\mathcal{P}$.

There are several generalizations of this assumption, we define the bilinear version for the Diffie-Hellman assumption of KEA, the BDH-KEA, that we use in Chapters 3 and 6 to reduce our new q-assumptions to the q-TSDH assumption, basically we show that they are equivalent under BDH-KE assumption.

Assumption 5 (Bilinear Diffie-Hellman Knowledge of Exponent Assumption, BDH-KE [4])**.** *For all non-uniform PPT adversaries $\mathcal{A}$:*

$$\Pr\left[\left([\alpha_1]_1, [\alpha_2]_2 \,\|a\right) \leftarrow (\mathcal{A}\|\mathcal{X}_{\mathcal{A}})(gk) : e\left([\alpha_1]_1, [1]_2\right) = e\left([1]_1, [\alpha_2]_2\right) \wedge a\mathcal{P}_1 \neq \alpha_1\right] \approx 0,$$

where the probability is taken over $gk = (N, \mathcal{P}_1, \mathcal{P}_2, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e) \leftarrow \mathcal{G}(1^\lambda)$ and the coin tosses of adversary $\mathcal{A}$.

2.5 Idealized Models of Computation

In this section we briefly describe two idealized models of computation widely used in the literature to prove some security proofs that we also use in our results.

2.5.1 Generic Group Model

The Generic Group Model (GGM) is an idealized model where groups do not have any specific property, [117]. It is a tool commonly used in the analysis of cryptographic problems.

In this model, the algorithms are restricted just to ask an oracle for basic group operations, such as computing the group law, checking for equality of elements, and possibly additional operations without being able to exploit any specific property of a given group representation.

In any proof, a generic adversary constructs the elements as linear combinations of the elements that it has access. Being secure in the GGM is the minimum required for any cryptographic protocol. For new assumptions, for example, holding in GGM is the first that we prove and then, we would try to find a reduction to another assumption.

2.5.2 Random Oracle Model

The Random Oracle (RO) Model is another idealized model where it is assumed that a Random Oracle machine that behaves as a truly random function exists and some parties have access to it, [89]. These parties can send queries x to the RO, who answers with a random value.

This model is widely used to prove cryptographic schemes. We use it in Chapter 5 for the first construction where we assume the hash function is evaluated only by querying the RO. The parties send x and the RO answers with $H(x)$.

In practice, when the schemes are implemented in real-world, the RO is instantiated by a hash function and the parties evaluate it by its own. It does not exist a perfect hash function that generates truly random elements, because it is a deterministic function,

but there exist hash functions that are considered to be close enough to a RO in practise. In section 2.6.4 we define hash functions.

2.6 Cryptographic Primitives

In this section we define the basic cryptographic primitives commitment schemes, encryption schemes and signature schemes briefly. We also give examples of them that are used in our results.

2.6.1 Commitment schemes

A commitment scheme is a cryptographic primitive where a party P commits itself to a value, that can be a vote or a solution of a problem, that does not want to reveal until it decides and it cannot change it. Briefly, in the first phase, the sender P sends the commitment of the value to a receiver V, and the requirement is *secrecy*, i.e. V cannot gain any knowledge of the value from the commitment before the next phase. In the revealing phase, P opens the commitment revealing the value used to commit and V checks that this value corresponds to the commitment that it has received in the previous phase, the requirement is *unambiguity*, P is bound to the value, it cannot alter the content of the commitment. We define it formally in the following.

Definition 8. *A* commitment scheme *is a tuple of probabilistic polynomial-time algorithms* $(\mathsf{Gen}, \mathsf{Com})$ *such that:*

- $\mathsf{Gen}(1^\lambda)$*: The generator algorithm takes the security parameter as input and outputs the public parameters* pp*, which includes the commitment key* $\mathtt{ck}$*, the commitment, message and randomizer spaces,* $\mathtt{CSP}$*,* $\mathtt{MSP}$*,* $\mathtt{RSP}$*, respectively, and also the commitment algorithm* Com*.*

- $\mathsf{Com}(\mathsf{pp}, \mathsf{m}, r)$*: The commitment algorithm takes the public parameters* pp*, a message* $\mathsf{m} \in \mathtt{MSP}$ *and a randomness* $r \in \mathtt{RSP}$*, and outputs a commitment* $\mathsf{c} \in \mathtt{CSP}$*. This algorithm is used by the sender* P *to commit the value* m *by sampling* $r \leftarrow \mathtt{RSP}$ *itself.*

The sender can later decommit c *and reveal* m *by sending* (m, r) *to the receiver. The receiver verifies this by checking that* $\mathsf{Com}(\mathsf{pp}, \mathsf{m}, r) = \mathsf{c}$ *holds.*

The requirements of privacy and unambiguity are captured by the hiding and binding properties, respectively, defined in the following.

Definition 9 (Hiding)**.** *The commitment* $(\mathsf{Gen}, \mathsf{Com})$ *is* hiding *if for all PPT adversary* $\mathcal{A}$*,*

$$\Pr\left[\begin{array}{c}\mathsf{pp} \leftarrow \mathsf{Gen}(1^\lambda), (\mathsf{m}_0, \mathsf{m}_1) \leftarrow \mathcal{A}(\mathsf{pp}), \mathsf{m}_0, \mathsf{m}_1 \in \mathtt{MSP}, \\ b \leftarrow \{0,1\}, r \leftarrow \mathtt{RSP}, \mathsf{c} = \mathsf{com}(\mathsf{pp}, \mathsf{m}_b, r) : \mathcal{A}(\mathsf{c}) = b\end{array}\right] \leq \tfrac{1}{2} + negl(\lambda).$$

Definition 10 (Binding)**.** *The commitment* $(\mathsf{Gen}, \mathsf{Com})$ *is* binding *if for all PPT adversary* $\mathcal{A}$*,*

$$\Pr\left[\begin{array}{c}\mathsf{pp} \leftarrow \mathsf{Gen}(1^\lambda), (\mathsf{m}_0, \mathsf{m}_1, r_0, r_1, \mathsf{c}) \leftarrow \mathcal{A}(\mathsf{pp}) : \\ \mathsf{m}_0 \neq \mathsf{m}_1, \mathsf{com}(\mathsf{pp}, \mathsf{m}_0, r_0) = \mathsf{c} = \mathsf{com}(\mathsf{pp}, \mathsf{m}_1, r_1)\end{array}\right] \leq negl(\lambda).$$

2.6.2 Public-key encryption schemes

We give in the following the definition of public-key encryption scheme. Basically, the difference between private-key and public-key encryption schemes is that the public-key contains two keys, the secret key sk and the public key pk, and the encryption and decryption processes are different, while in private-key there is just one secret key and the encryption and decryption processes are interchangeable. We also call symmetric and asymmetric for private and public key encryption schemes, respectively.

Definition 11. *A* public-key encryption scheme *is a triple of probabilistic polynomial-time algorithms* $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ *such that*

- $\mathsf{Gen}(1^\lambda)$*: The generator algorithm takes the security parameter as input and outputs* $(\mathsf{pk}, \mathsf{sk})$ *where* pk *is the* public key *and* sk *the* secret key*. It also fixes the message and ciphertext spaces* $\mathtt{MSP}, \mathtt{CSP}$*.*
- $\mathsf{Enc}_{\mathsf{pk}}(\mathsf{m})$*: The probabilistic encryption algorithm takes a public key* pk *and a message* $\mathsf{m} \in \mathtt{MSP}$ *as input, and outputs a ciphertext* $\mathsf{c} \leftarrow \mathsf{Enc}_{\mathsf{pk}}(\mathsf{m})$*.*
- $\mathsf{Dec}_{\mathsf{sk}}(\mathsf{c})$*: The deterministic decryption algorithm takes a secret key* sk *and a ciphertext* $\mathsf{c} \in \mathtt{CSP}$*, and outputs a message* m *or the symbol* $\perp$ *denoting failure.*

For any $\mathsf{m} \in \mathtt{MSP}$*, it holds that* $\mathsf{Dec}_{\mathsf{sk}}(\mathsf{Enc}_{\mathsf{pk}}(\mathsf{m})) = \mathsf{m}$ *except with negligible probability over* $(\mathsf{pk}, \mathsf{sk})$*.*

Example: Lifted ElGamal encryption. As an example of encryption scheme we give the definition of the Lifted ElGamal encryption, that is a version of the ElGamal encryption scheme where the message is encrypted and recovered in the group. Both are IND-CPA secure under the DDH assumption, which means that no PPT adversary is able to, given the public key and for any two messages of its choice, distinguish with probability higher than $\frac{1}{2}$ which one of them was encrypted.

Definition 12. *The* Lifted ElGamal encryption scheme *is defined by the following algorithms:*

- $\mathsf{Gen}(1^\lambda)$*: The generator algorithm takes the security parameter as input and runs* $(\mathbb{G}, q, \mathcal{P}) \leftarrow gk(1^\lambda)$ *that gives a group, the order of the group and a generator. Then, chooses a uniform* $x \leftarrow \mathbb{Z}_q$ *and fixes* $\mathsf{sk} = x$*,* $\mathsf{pk} = \mathsf{sk}\mathcal{P} = [\mathsf{sk}] \in \mathbb{G}$*. It outputs* $(\mathsf{pk}, \mathsf{sk})$*.*

- $\mathsf{Enc}_{\mathsf{pk}}(\mathsf{m})$*: The probabilistic encryption algorithm takes a public key* pk *and a message* $\mathsf{m} \in \mathtt{MSP}$ *as input, chooses a random element* $r \in \mathbb{Z}_q$*, and outputs the ciphertext* $\begin{bmatrix} \mathsf{c}_1 \\ \mathsf{c}_2 \end{bmatrix} = \mathsf{Enc}_{[\mathsf{sk}]}(\mathsf{m}, r) = \mathsf{m}[\boldsymbol{e}_2] + r \begin{bmatrix} 1 \\ \mathsf{sk} \end{bmatrix}$.

- $\mathsf{Dec}_{\mathsf{sk}}(\mathsf{c})$*: The deterministic decryption algorithm takes a secret key* sk *and a ciphertext* $\mathsf{c} \in \mathtt{CSP}$*, and recovers the message* m *in* $\mathbb{G}$ *by computing* $[c_2] - \mathsf{sk}[c_1] = [\mathsf{m}]$.

The Lifted ElGamal encryption scheme can be seen as a commitment scheme, in which case it is perfectly binding and computationally hiding under the DDH assumption, and in fact this is how we will use it in our schemes.

2.6.3 Signature schemes

In this section we briefly define signature schemes and give as an example the Boneh-Boyen signatures used in our results (directly in Chapter 3 and some variations in Chapter 5).

A signature scheme is a cryptographic primitive that allows a signer S to sign a message using a secret key sk in such a way that anyone who knows the associated public key pk can verify that S was who signed the message and it was not modified. We give the formal definition in the follwing.

Definition 13. *A* signature scheme *is a triple of probabilistic polinomial-time algorithms* $(\mathsf{Gen}, \mathsf{Sign}, \mathsf{Ver})$ *such that*

- $\mathsf{Gen}(1^\lambda)$*: The generator algorithm takes the security parameter as input and outputs* $(\mathsf{pk}, \mathsf{sk})$ *where* pk *is the* public key *and* sk *the* private key*. It also fixes the message and signature spaces* $\mathtt{MSP}, \mathtt{SSP}$*.*

- $\mathsf{Sign}_{\mathsf{sk}}(\mathsf{m})$*: The probabilistic encryption algorithm takes a private key* pk *and a message* $\mathsf{m} \in \mathtt{MSP}$ *as input, and outputs a signature* $\sigma \leftarrow \mathsf{Sign}_{\mathsf{sk}}(\mathsf{m})$*.*

- $\mathsf{Ver}_{\mathsf{pk}}(\sigma)$*: The deterministic verification algorithm takes as input a public key* sk *and a signature* $\sigma \in \mathtt{SSP}$*, and outputs a bit* $b = \mathsf{Ver}_{\mathsf{pk}}(\mathsf{m}, \sigma)$ *which is* 1 *if the signature is valid for the message* m*, and* 0 *otherwise.*

For any $\mathsf{m} \in \mathtt{MSP}$*, it holds that* $\mathsf{Ver}_{\mathsf{pk}}(\mathsf{Sign}_{\mathsf{sk}}(\mathsf{m})) = 1$ *except with negligible probability over* $(\mathsf{pk}, \mathsf{sk})$.

The sender S runs the generator algorithm $\mathsf{Gen}(1^\lambda)$ to obtain keys $(\mathsf{pk}, \mathsf{sk})$. Then, pk is publicized as the public key of S. When S wants to authenticate a message m, it computes the signature $\sigma \leftarrow \mathsf{Sign}_{\mathsf{sk}}(\mathsf{m})$ using its private key sk and sends (m, σ). The receiver verifies the authenticity of m by checking if $\mathsf{Ver}_{\mathsf{pk}}(\mathsf{m}, \sigma) = 1$.

The requirement is that no adversary can forge a signature. For a fixed public key pk of S, a forgery is a message m along a valid signature σ that was not previously signed by S.

As an example of signature schemes, we recall the Boneh-Boyen signatures presented in [22] in the following. In Section 2.7.3 we give the definition and security definitions of another kind of signatures schemes called signatures of knowledge that can be seen as a generalization of signatures schemes, where one can sign if, instead of who possesses the secret key, who has a valid witness for membership in a language.

Example: Boneh-Boyen signatures. The generator algorithm returns a bilinear group $(p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, \mathcal{P}_1, \mathcal{P}_2)$, fixes the message space to be $\mathbb{Z}_p$, and the signature space $\mathbb{G}_2$. It chooses the secret key $\mathsf{sk} \leftarrow \mathbb{Z}_p$, and defines the public key as $[\mathsf{sk}]_1 \in \mathbb{G}_1$. To sign a message $\mathsf{m} \in \mathbb{Z}_p$, the signer computes

$$[\sigma]_2 = \mathsf{Sign}_{\mathsf{sk}}(\mathsf{m}) = \left[\frac{1}{\mathsf{sk} - \mathsf{m}}\right]_2.$$

The receiver validates the signature if the equation $e([\mathsf{sk}]_1 - [\mathsf{m}]_1, [\sigma]_2) = [1]_T$ holds, it requires one pairing. Boneh-Boyen signatures are existentially unforgeable under the q-SDH assumption, which means no adversary can forge a signature even when it knowns many signatures of messages chosen by itself.

2.6.4 Hash functions

In this section we briefly define the hash functions, which are functions that take a string of some length and compress it into a shorter string of a fixed-length. A common requirement for a hash function H is to have few collisions, where a collision is a pair of different inputs x, x' such that the output of the hash for both strings is the same, i.e. $H(x) = H(x')$.

Definition 14. *A* hash function *with output length ℓ is a pair of probabilistic polynomial-time algorithms (Gen, H) such that*

- $\mathsf{Gen}(1^\lambda)$*: the generator of the key takes as input a security parameter and outputs a key s.*
- $H(s, x)$*: the function H takes as input a key s and a string $x \in \{0,1\}^*$ and outputs a string $H^s(x) \in \{0,1\}^\ell$.*

Note that sometimes the input length is fixed, say ℓ', then we require that $\ell' > \ell$.

The collision resistance property states that no adversary is able to find a collision except with negligible probability. Collision-resistant hash functions were formally defined by Damgärd in [41].

Definition 15. *A hash function (Gen, H) is* collision resistant *if for all probabilistic polynomial-time adversaries $\mathcal{A}$*

$$\Pr\left[\begin{array}{c} s \leftarrow \mathsf{Gen}(1^\lambda), (x, x') \leftarrow \mathcal{A}(s, \ell') : \\ x, x' \in \{0,1\}^{\ell'}, x \neq x', H^s(x) = H^s(x') \end{array}\right] \leq negl(\lambda).$$

For simplicity we will refer to H or H^s as a collision-resistant hash function, instead of (Gen, H).

2.7 Non-Interactive Zero-Knowledge

Zero-Knowledge (ZK) proofs allow to prove a statement without yielding anything beyond its validity. In these proofs the prover is the party who provides the proof and the verifier, the one who validates it. Formally, the prover claims some statement of the type $x \in \mathcal{L}$, where $\mathcal{L}$ is a language, and gives a proof for that. For example, if the prover claims that some equations hold, let $\mathcal{L}$ be the language defined by solvable equations of a fixed number of variables. Then, $x \in \mathcal{L}$ if and only if there exists a valid assignment of the variables such that the equations hold.

To set the formal definitions of ZK proofs properties, we use the notation of languages and relations already stablished in Section 2.2.2. We recall briefly that a language is defined by

$$\mathcal{L} = \{x : \text{there exists some } w \text{ such that } (x, w) \in \mathbf{R}_\mathcal{L}\}$$

where $\mathbf{R}_\mathcal{L}$ is the relation checker that validates if the tuple (x, w) satisfies the relation. In ZK proofs the prover claims $x \in \mathcal{L}$, where we call x the *statement*, and gives a proof of the existence of a w, that we call the *witness*, such that $(x, w) \in \mathbf{R}_\mathcal{L}$.

ZK proofs should satisfy the basic properties of completeness, soundness and zero-knowledge. We briefly recall them here and give the formal definitions for NIZK Arguments in the following.

- Completeness: The prover can always convince the verifier that a given statement is true if the prover knows a witness testifying to the truth of the statement.
- Soundness: A malicious prover cannot convince the verifier. The statement might be true and still if the malicious prover does not have the witness, it should not be able to convince the verifier.
- Zero-knowledge: The proof does not reveal any information beyond the validity of the statement. It means that the message exchange between prover and verifier does not leak any knowledge about the witness. This is formalized by the existence of a simulator who generates the proofs with indistinguishable distribution just using the statement and public parameters.

2.7.1 NIZK Arguments

Non-interactive ZK proofs are those which consists in just one message from the prover to the verifier. Blum, Feldman and Micali [20] introduced NIZK proofs using a common reference string (crs) as public parameters shared between the prover and the verifier. The authors showed the crs is enough to achieve zero-knowledge without requiring any interaction. The crs consists of some group elements created honestly by a trust party and shared by the other parties. The prover uses them to produce the proof, the verifier to check the proof and the simulator uses them along with some secret information to produce a simulated proof. It should be created by a trusted third party, the same who produces this secret information to the simulator that we call *simulation trapdoor* (tr), or just *trapdoor* when it's clear from the context.

There exist different notions for the soundness property in ZK proofs: *computational soundness* guarantees that no polynomial-time adversary chan cheat and *statiscal or perfect soundness* guarantees even an adversary with unbounded prover cannot convince the verifier of a false statement. An *Argument* is a computationally sound Zero-Knowledge proof that we define in the following.

Definition 16. *Given a language* $\mathcal{L} \in \mathcal{NP}$*, a* non-interactive argument *for* $\mathcal{R}_{\mathcal{L}}$ *is a quadruple of efficient algorithms* $(\mathsf{Setup}, \mathsf{Prove}, \mathsf{Verify}, \mathsf{Simulate})$ *such that*

$(\mathsf{crs}, \mathsf{tr}) \leftarrow \mathsf{Setup}(1^\lambda, \mathbf{R})$**:** Setup *is a PPT algorithm that takes as input a security parameter and a relation* $\mathbf{R} \in \mathcal{R}_{\mathcal{L}}$ *and returns a common reference string* crs *and a simulation trapdoor* tr *for the relation* $\mathbf{R}$*.*

$\pi \leftarrow \mathsf{Prove}(\mathsf{crs}, x, w)$**:** Prove *is a PPT algorithm executed by the prover, given a common reference string* crs *for a relation* $\mathbf{R}$*, a statement* x *and a witness* w *such that* $(x, w) \in \mathbf{R}$*, and returns an argument* π*.*

$\{0,1\} \leftarrow \mathsf{Verify}(\mathsf{crs}, x, \pi)$**:** Verify *is a PPT algorithm executed by the verifier, that takes as input a* crs*, a statement* x *and a proof* π*, and returns either* 0 *(reject) or* 1 *(accept).*

$\pi \leftarrow \mathsf{Simulate}(\mathsf{crs}, x, \mathsf{tr})$**:** Simulate *is a PPT algorithm that, given* $(\mathsf{crs}, \boldsymbol{x}, \mathsf{tr})$*, outputs a simulated argument* π*.*

We say that $(\mathsf{Setup}, \mathsf{Prove}, \mathsf{Verify}, \mathsf{Simulate})$ *is a non-interactive zero-knowledge argument for* $\mathcal{R}_{\mathcal{L}}$ *if it has perfect completeness, soundness and perfect zero-knowledge defined as follows.*

Perfect Completeness. *For all* $\mathbf{R} \in \mathcal{R}_{\mathcal{L}}$*,* $(x, w) \in \mathbf{R}$*,*

$$\Pr\left[\begin{array}{c}(\mathsf{crs}, \mathsf{tr}) \leftarrow \mathsf{Setup}(1^\lambda, \mathbf{R}), \pi \leftarrow \mathsf{Prove}(\mathbf{R}, \mathsf{crs}, x, w) : \\ \mathsf{Verify}(\mathbf{R}, \mathsf{crs}, x, w) = 1\end{array}\right] = 1.$$

Computational Soundness. *For all non-uniform polynomial time adversaries* $\mathcal{A}$*,*

$$\Pr\left[\begin{array}{c}(\mathbf{R}, z) \leftarrow \mathcal{R}_{\mathcal{L}}(1^\lambda), (\mathsf{crs}, \mathsf{tr}) \leftarrow \mathsf{Setup}(1^\lambda, \mathbf{R}), \\ (x, \pi) \leftarrow \mathcal{A}(\mathbf{R}, z, \mathsf{crs}) : x \notin \mathcal{L}, \mathsf{Verify}(\mathsf{crs}, x, \pi) = 1\end{array}\right] \approx 0.$$

Perfect Zero-Knowledge. *For all* $(\mathbf{R}, z) \leftarrow \mathcal{R}_{\mathcal{L}}(1^\lambda)$*,* $(x, w) \leftarrow \mathbf{R}$ *and all adversaries* $\mathcal{A}$*,*

$$\Pr\left[\begin{array}{c}(\mathsf{crs}, \mathsf{tr}) \leftarrow \mathsf{Setup}(1^\lambda, \mathbf{R}), \\ \pi \leftarrow \mathsf{Prove}(\mathbf{R}, \mathsf{crs}, x, w) : \\ \mathcal{A}(\mathbf{R}, z, \mathsf{crs}, \pi) = 1\end{array}\right] = \Pr\left[\begin{array}{c}(\mathsf{crs}, \mathsf{tr}) \leftarrow \mathsf{Setup}(1^\lambda, \mathbf{R}), \\ \pi \leftarrow \mathsf{Simulate}(\mathbf{R}, \mathsf{crs}, x, \mathsf{tr}) : \\ \mathcal{A}(\mathbf{R}, z, \mathsf{crs}, \pi) = 1\end{array}\right].$$

Extraction

Arguments of knowledge have the additional requirement that to pass the verification the prover should use the witness w to create valid proofs such that $(x, w) \in \mathbf{R}$. As we have explained previously, this property is formulated with the existence of an extractor able to compute a witness whenever a statement x and a proof π are valid. There exist two types of extractors, black box (BB) and non-black box (nBB). In BB extraction the extractor $\mathcal{E}$ does not need access to the source code to extract the witness, the extraction procedure works for all the adversaries; while nBB that is a stronger notion, the

extractor $\mathcal{E}_{\mathcal{A}}$ depends on the adversary $\mathcal{A}$. Then, it is preferable to have BB extraction, like some of the Groth-Sahai proofs [78] and our construction in Chapter 4, where we define formally *BB Knowledge Soundness*. The *nBB Knowledge Soundness* property is the one used in zk-SNARKs and in our construction in Chapter 5, we define it formally in the following.

Definition 17. *Let $\mathcal{L}$ be a language in $\mathcal{NP}$, a NIZK argument π for a relation $\mathbf{R} \in \mathcal{R}_{\mathcal{L}}$ has* Knowledge Soundness *if for all non-uniform polynomial-time adversaries $\mathcal{A}$, there exists a non-uniform polynomial-time extractor $\mathcal{E}_{\mathcal{A}}$ such that*

$$\Pr\left[\begin{array}{c}(\mathsf{crs},\mathsf{tr}) \leftarrow \mathsf{Setup}(1^\lambda,\mathbf{R}); ((x,\pi)||w) \leftarrow (\mathcal{A}||\mathcal{E}_{\mathcal{A}})(\mathbf{R},\mathsf{crs}) : \\ (x,w) \notin \mathbf{R}, \mathsf{Verify}(\mathsf{crs},x,\pi)=1\end{array}\right] \approx 0.$$

We recall the definitions of an extractable proof and the notion of f-extractability defined by Belenkiy et al. [17].

We also consider a generalization of extractability where instead of extracting the witness, the extractor gives a function of the witness, which is a weaker notion but enough in some cases. For example, in some of our results we use the exponentiation in the group, $f(x) := [x]_\iota$, and in the case the witness a string of bits, this is equivalent to have extraction in the field.

Definition 18. *In a* f-extractable *proof system for $\mathbf{R}$, the extractor extracts a value z such that $z = f(w)$ and $(x,w) \in \mathbf{R}$ for some witness w. If $f(\cdot)$ is the identity function, we get the usual notion of extractability.*

2.7.2 Groth-Sahai proofs and QA-NIZK arguments

In this section we present the Groth-Sahai proof system, the definition of Quasi-Adaptive NIZK argument and some concrete languages of membership in linear spaces for which there exists constant-size QA-NIZK constructions that we use as building blocks in our results.

Dual-mode commitments and Groth-Sahai proofs [78]

Groth-Sahai proofs allow to prove satisfiability of quadratic equations in bilinear groups in the non-interactive setting. More precisely, Groth-Sahai proofs deal with equations of the form

$$\sum_{j=1}^{m_y} a_j \mathsf{y}_j + \sum_{i=1}^{m_x} b_i \mathsf{x}_i + \sum_{i,j=1}^{m_x,m_y} \gamma_{i,j} \mathsf{x}_i \mathsf{y}_j = t,$$

in which the set of variables is divided into two disjoint subsets $\mathsf{X} = \{\mathsf{x}_1, \ldots, \mathsf{x}_{m_x}\}$ and $\mathsf{Y} = \{\mathsf{y}_1, \ldots, \mathsf{y}_{m_y}\}$, and depending on the type of equation $\mathsf{X}, \mathsf{Y} \subset \mathbb{Z}_p$ (quadratic equations in $\mathbb{Z}_p$), $\mathsf{X} \subset \mathbb{Z}_p, \mathsf{Y} \subset \mathbb{G}_\iota$ (multi-exponentiation equations in $\mathbb{G}_\iota$) for $\iota \in \{1, 2\}$ or $\mathsf{X} \subset \mathbb{G}_1$ and $\mathsf{Y} \subset \mathbb{G}_2$ (pairing product equations). Here the product means a bilinear operation which is multiplication in $\mathbb{Z}_p$, exponentiation or the pairing operation.

The scheme can be seen as a commit-and-prove scheme [50], where in the first step the prover gives commitments to the solutions, and in the second provides a proof that these commitments verify the corresponding equation. In particular, the commitments used are *dual-mode commitments*, that is, commitments that can be either perfectly binding or perfectly hiding, and we can switch from one to the other with an indistinguishable change of security game. More precisely, Groth-Sahai commitments to field elements $z \in \mathbb{Z}_p$ and group elements $[z] \in \mathbb{G}_\iota$ are, respectively:

$$\mathsf{Com}(z; w) = z\,[\boldsymbol{u}]_\iota + w[\boldsymbol{u}_1]_\iota, \qquad \mathsf{Com}([z]_\iota; w_1, w_2) = \begin{bmatrix} 0 \\ z \end{bmatrix}_\iota + w_1[\boldsymbol{u}_1]_\iota + w_2[\boldsymbol{u}_2]_\iota,$$

where $[\boldsymbol{u}], [\boldsymbol{u}_1], [\boldsymbol{u}_2]$ are vectors in $\mathbb{G}^2$ given in the commitment key, and their definitions depend on whether we want the commitments to be perfectly binding or perfectly hiding.

Groth-Sahai proofs are sound, witness-indistinguishable and, in many cases, zero-knowledge. More precisely, the proof is always zero-knowledge for quadratic equations in $\mathbb{Z}_p$ and multi-exponentiation equations, and also for pairing product equations provided that $t = 1$. Some generalizations are possible, as discussed in [51].

Quasi-Adaptive Non-Interactive Zero-Knowledge Arguments

In this section we recall the formal definition of Quasi-Adaptive non-interactive zero-knowledge proofs. A Quasi-Adaptive NIZK proof system [85] enables to prove membership in a language defined by a relation $\mathbf{R}_\rho$, which in turn is completely determined by some parameter ρ sampled from a distribution $\mathcal{D}_{gk}$. We say that $\mathcal{D}_{gk}$ is *witness sampleable* if there exists an efficient algorithm that samples (ρ, ω) from a distribution $\mathcal{D}_{gk}^{\mathsf{par}}$ such that ρ is distributed according to $\mathcal{D}_{gk}$, and membership of ρ in the *parameter language* $\mathcal{L}_{\mathsf{par}}$ can be efficiently verified with ω. While the Common Reference String (crs) can be set based on ρ, the zero-knowledge simulator is required to be a single probabilistic polynomial-time algorithm that works for the whole collection of relations $\mathbf{R}_{gk}$.

Definition 19. *A tuple of algorithms* $(\mathsf{K}_0, \mathsf{K}_1, \mathsf{P}, \mathsf{V})$ *is called a* QA-NIZK proof system *for witness-relations* $\mathbf{R}_{gk} = \{\mathbf{R}_\rho\}_{\rho \in \sup(\mathcal{D}_{gk})}$ *with parameters sampled from a distribution* $\mathcal{D}_{gk}$ *over associated parameter language* $\mathcal{L}_{\mathsf{par}}$*, if there exists a probabilistic*

polynomial time simulator $(\mathsf{S}_1, \mathsf{S}_2)$*, such that for all non-uniform PPT adversaries* $\mathcal{A}_1$*,* $\mathcal{A}_2$*,* $\mathcal{A}_3$ *we have:*

Perfect Quasi-Adaptive Completeness:

$$\Pr\left[\begin{array}{l|c} gk \leftarrow \mathsf{K}_0(1^\lambda); \rho \leftarrow \mathcal{D}_{gk}; \mathsf{crs} \leftarrow \mathsf{K}_1(gk, \rho); & \mathsf{V}(\mathsf{crs}, x, \pi) = 1 \\ (x, w) \leftarrow \mathcal{A}_1(gk, \mathsf{crs}); \pi \leftarrow \mathsf{P}(\mathsf{crs}, x, w) & \textit{if } \mathbf{R}_\rho(x, w) \end{array}\right] = 1.$$

Computational Quasi-Adaptive Soundness:

$$\Pr\left[\begin{array}{l|c} gk \leftarrow \mathsf{K}_0(1^\lambda); \rho \leftarrow \mathcal{D}_{gk}; & \mathsf{V}(\mathsf{crs}, x, \pi) = 1 \textit{ and} \\ \mathsf{crs} \leftarrow \mathsf{K}_1(gk, \rho); (x, \pi) \leftarrow \mathcal{A}_2(gk, \mathsf{crs}) & \neg(\exists w : \mathbf{R}_\rho(x, w)) \end{array}\right] \approx 0.$$

Perfect Quasi-Adaptive Zero-Knowledge:

$$\Pr[gk \leftarrow \mathsf{K}_0(1^\lambda); \rho \leftarrow \mathcal{D}_{gk}; \mathsf{crs} \leftarrow \mathsf{K}_1(gk, \rho) : \mathcal{A}_3^{\mathsf{P}(\mathsf{crs}, \cdot, \cdot)}(gk, \mathsf{crs}) = 1] =$$
$$\Pr[gk \leftarrow \mathsf{K}_0(1^\lambda); \rho \leftarrow \mathcal{D}_{gk}; (\mathsf{crs}, \mathsf{tr}) \leftarrow \mathsf{S}_1(gk, \rho) : \mathcal{A}_3^{\mathsf{S}(\mathsf{crs}, \mathsf{tr}, \cdot, \cdot)}(gk, \mathsf{crs}) = 1]$$

where

- $\mathsf{P}(\mathsf{crs}, \cdot, \cdot)$ *emulates the actual prover. It takes input* (x, w) *and outputs a proof* π *if* $(x, w) \in \mathbf{R}_\rho$*. Otherwise, it outputs* $\perp$*.*
- $\mathsf{S}(\mathsf{crs}, \mathsf{tr}, \cdot, \cdot)$ *is an oracle that takes input* (x, w) *and outputs a simulated proof* $\mathsf{S}_2(\mathsf{crs}, \mathsf{tr}, x)$ *if* $(x, w) \in \mathbf{R}_\rho$ *and* $\perp$ *if* $(x, w) \notin \mathbf{R}_\rho$*.*

We assume that crs *contains an encoding of* ρ*, which is thus available to* V*.*

We have defined the basic properties of NIZK that we require in QA-NIZK proof systems. However, for witness sampleable distributions, there is a stronger notion of soundness, where the adversary has also access to a witness of the parameter ρ. This notion is defined in the full version of [67] and we recall it in the following. We use this notion in Chapters 3 and 6.

Definition 20. *A QA-NIZK argument* $(\mathsf{K}_0, \mathsf{K}_1, \mathsf{P}, \mathsf{V})$ *is* Computational Quasi-Adaptive Strong Soundness *if for all non-uniform PPT adversary* $\mathcal{A}_2$*,*

$$\Pr\left[\begin{array}{l|c} gk \leftarrow \mathsf{K}_0(1^\lambda); (\rho, \omega) \leftarrow \mathcal{D}_{gk}^{\mathsf{par}}; & \mathsf{V}(\mathsf{crs}, x, \pi) = 1 \textit{ and} \\ \mathsf{crs} \leftarrow \mathsf{K}_1(gk, \rho); (x, \pi) \leftarrow \mathcal{A}_2(gk, \omega, \mathsf{crs}) & \neg(\exists w : \mathbf{R}_\rho(x, w)) \end{array}\right] \approx 0.$$

QA-NIZK Arguments of Membership in Linear Spaces [85]. We describe some languages for which there exist constant-size QA-NIZK arguments of membership which will be used as building blocks in our constructions. These languages are (i) linear subspaces of $\mathbb{G}_\iota^m$, $\iota \in \{1,2\}$, in [86, 92], and (ii) bilateral linear subspaces, that is, linear subspaces of $\mathbb{G}_1^m \times \mathbb{G}_2^n$, in [67]. For $\iota \in \{1,2\}$,

$$\mathcal{L}_{[\mathbf{M}]_\iota} := \{[\boldsymbol{x}]_\iota \in \mathbb{G}_\iota^n : \exists \boldsymbol{w} \in \mathbb{Z}_q^t,\ \boldsymbol{x} = \mathbf{M}\boldsymbol{w}\}, \tag{i}$$

$$\mathcal{L}_{[\mathbf{M}]_1,[\mathbf{N}]_2} := \{([\boldsymbol{x}]_1, [\boldsymbol{y}]_2) \in \mathbb{G}_1^m \times \mathbb{G}_2^n : \exists \boldsymbol{w} \in \mathbb{Z}_q^t,\ \boldsymbol{x} = \mathbf{M}\boldsymbol{w},\ \boldsymbol{y} = \mathbf{N}\boldsymbol{w}\}, \tag{ii}$$

We use LS (BLS) to designate (bilateral) linear subspace proof systems for the languages $\mathcal{L}_{[\mathbf{M}]_\iota}$, $\mathcal{L}_{[\mathbf{M}]_1,[\mathbf{N}]_2}$. These proof systems verify strong soundness, which essentially means that they are sound even when the discrete logarithm of the matrices is given. This property is formally defined in González et al. [67]. Case (i) can be instantiated based on the Kernel Diffie-Hellman Assumption 2, and the proof has size $|\mathbb{G}_\iota|$, whereas (ii) can be based on the Split Kernel Diffie-Hellman Assumption 3, and the proof has size $2|\mathbb{G}_1| + 2|\mathbb{G}_2|$.

2.7.3 Signatures of Knowledge

A Signature of Knowledge (SoK) [36] generalizes the concept of digital signature. One can sign the message just if it has a valid witness for membership in a language, for example, in Chapter 4 we give one SoK for boolean CircuitSat.

A SoK requires the three basic properties: *Correctness* ensures that all signers with a valid witness can always produce a signature that convinces the verifier, *Simulation-Extractability* that any adversary able to issue a new signature, even after seeing arbitrary signatures for different instances, should know a witness and *Perfect Simulatability* ensures that the verifier learns nothing new about the witness from a signature. We give the formal definitions of [74] in the following.

Definition 21 (Signature of Knowledge)**.** *Let $\mathcal{L}$ be a language and $\mathcal{R}_\mathcal{L}$ a set of relations parametrized by the language $\mathcal{L}$. Then, a tuple* $(\mathsf{SSetup}, \mathsf{SSign}, \mathsf{SVer}, \mathsf{SSimulate})$ *is a* Signature of Knowledge *scheme for $\mathbf{R} \in \mathcal{R}_\mathcal{L}$ if it is correct, simulatable, simulation-extractable (defined in the following) and it is composed by the following algorithms:*

$\mathsf{tr}_s, \mathsf{tr}_e, \mathsf{pp} \leftarrow \mathsf{SSetup}(1^\lambda, \mathbf{R})$*: the setup algorithm is a PPT algorithm that takes as input the public parameter 1^λ and a relation $\mathbf{R} \in \mathcal{R}_\mathcal{L}$ and returns public parameters* pp*, together with a simulation trapdoor* tr_s *and an extraction trapdoor* tr_e*. It also fixes the message and the signature spaces,* $\mathtt{MSP}$*,* $\mathtt{SSP}$*, respectively.*

$\sigma \leftarrow \mathsf{SSign}(\mathsf{pp}, x, w, \mathsf{m})$*: the signing algorithm is a PPT algorithm that takes as input the public parameters* pp*, a pair* $(\boldsymbol{x}, \boldsymbol{w}) \in \mathbf{R}$ *and a message* $\mathsf{m} \in \mathtt{MSP}$ *and returns a signature* σ*.*

$0/1 \leftarrow \mathsf{SVer}(\mathsf{pp}, x, \mathsf{m}, \sigma)$*: the verification algorithm is a deterministic polynomial time algorithm that takes as input some public parameters* pp*, an instance* x*, a message* $\mathsf{m} \in \mathcal{M}_\lambda$*, and a signature* σ *and outputs either* 0 *or* 1 *if it rejects or accepts, respectively.*

$\sigma \leftarrow \mathsf{SSimulate}(\mathsf{pp}, \mathsf{tr}_s, x, \mathsf{m})$*: the simulated signing algorithm is a PPT algorithm that takes as input some public parameters* pp*, a simulation trapdoor* tr_s*, and an instance* x *and returns a signature* σ*.*

A trusted party runs the SSetup algorithm to obtain pp, tr_s, tr_e and publicizes pp. When the sender wants to authenticate a message m, it computes the signature σ using its witness w as secret key. The verifier checks m and σ using pp. The simulator uses the simulation trapdoor to produce a valid signature without knowing the witness.

Definition 22. *A Signature of Knowledge is* correct *if for all* $\mathbf{R} \in \mathcal{R}_\mathcal{L}$*, for all* $(x, w) \in \mathbf{R}$ *and for all* $\mathsf{m} \in \mathtt{MSP}$*,*

$$\Pr\left[\mathsf{pp} \leftarrow \mathsf{SSetup}(1^\lambda, \mathbf{R}); \sigma \leftarrow \mathsf{SSign}(\mathsf{pp}, x, w, \mathsf{m}) : \mathsf{SVer}(\mathsf{pp}, x, \mathsf{m}, \sigma) = 1\right] = 1,$$

Definition 23. *A Signature of Knowledge for* $\mathcal{R}_\mathcal{L}$ *is* simulatable *if for all* $\mathbf{R} \in \mathcal{R}_\mathcal{L}$*, for any non-uniform PPT adversary* $\mathcal{A}$*,*

$$\Pr\left[\begin{array}{c} \mathsf{pp} \leftarrow \mathsf{SSetup}(1^\lambda, \mathbf{R}); b \leftarrow \{0,1\}, \\ b' \leftarrow \mathcal{A}^{\mathsf{OSign}^b_{\mathsf{pp},\mathsf{tr}_s}(\cdot,\cdot,\cdot)}(\mathsf{pp}) \end{array} : b = b' \right] = \frac{1}{2} + negl(\lambda),$$

where $\mathsf{OSign}^b_{\mathsf{pp},\mathsf{tr}_s}(x_i, w_i, \mathsf{m}_i)$ *checks* $((x_i, w_i) \in \mathbf{R}, \mathsf{m}_i \in \mathtt{MSP})$ *and returns* $\sigma_i \leftarrow \mathsf{SSign}(\mathsf{pp}, x_i, w_i, \mathsf{m}_i)$ *if* $b = 0$ *and* $\sigma_i \leftarrow \mathsf{SSimulate}(\mathsf{pp}, \mathsf{tr}_s, x_i, \mathsf{m}_i)$ *if* $b = 1$*.*

Definition 24. *A Signature of Knowledge for* $\mathcal{R}_\mathcal{L}$ *is* (black-box) simulation-extractable *if for all* $\mathbf{R} \in \mathcal{R}_\mathcal{L}$*, for any non-uniform PPT adversary* $\mathcal{A}$*, there exists a PPT extractor* $\mathcal{E}$ *such that*

$$\Pr\left[\begin{array}{c} (\mathsf{pp}, \mathsf{tr}_s, \mathsf{tr}_e) \leftarrow \mathsf{SSetup}(1^\lambda, \mathbf{R})(x, \mathsf{m}, \sigma) \leftarrow \mathcal{A}^{\mathsf{OSim}_{\mathsf{pp},\mathsf{tr}_s}(\cdot,\cdot)}, \\ w \leftarrow \mathcal{E}(\mathsf{pp}, \mathsf{tr}_e, (x, \mathsf{m}, \sigma)) : \\ (x, w) \notin \mathbf{R}, (x, \mathsf{m}, \sigma) \notin \mathcal{Q}, 1 \leftarrow \mathsf{SVer}(\mathsf{pp}, \vec{x}, \mathsf{m}, \sigma) \end{array}\right] \approx 0,$$

where $\mathsf{OSim}_{\mathsf{pp},\mathsf{tr}_s}(x_i, \mathsf{m}_i)$ *returns* $\sigma_i \leftarrow \mathsf{SSimulate}(\mathsf{pp}, \mathsf{tr}_s, x_i, \mathsf{m}_i)$ *and adds* $\{(x_i, \mathsf{m}_i, \sigma_i)\}$ *to the set* $\mathcal{Q}$*, which is initialized to* $\emptyset$*.*

In this definition the extractor $\mathcal{E}$ is a PPT algorithm that only accesses $\mathcal{A}$'s output, as opposed to the (white-box) simulation-extractable definition where the extractor has nBB access to the adversary.

Chapter 3

Shorter QA-NIZK for Quadratic Equations under falsifiable Assumptions

This chapter is based in our result *Shorter quadratic QA-NIZK proofs* [46] published in the Public Key Conference 2019.

3.1 Introduction

NIZK in Bilinear Groups. As we have explained in previous chapters, NIZK proofs are a very useful building block in the construction of cryptographic protocols. Since the first pairing-friendly NIZK proof system of Groth, Ostrovsky and Sahai [75] many different constructions have emerged in different models and under different assumptions, for various types of statements. Compared to a plain discrete logarithm setting, bilinear groups have a rich structure which is specially amenable to construct NIZK proofs.

Among this variety of results, there are three particularly interesting families with different advantages in terms of generality, efficiency or strength of the assumptions. On the one hand, there is a line of research initiated by Groth, Ostrovsky and Sahai [75, 76] and which culminated in the Groth-Sahai proof system [78, 79] that we explained in in Section 2.7.2. The latter result provides relatively efficient proofs for proving satisfiability of several types of quadratic equations in bilinear groups based

on standard assumptions. Although several works have tried to improve the efficiency of Groth-Sahai proofs like Escala and Groth [50] and Ràfols [112], for many equation types they still remain the best alternative based on falsifiable assumptions.

Another family of results are the constructions of quasi-adaptive NIZK (QA-NIZK) arguments initiated by Jutla and Roy [85] and leading to very efficient proofs of very concrete statements that we explained in Section 2.7.2. Most notably, given a bilinear group $gk := (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, \mathcal{P}_1, \mathcal{P}_2)$, proving membership in linear spaces in $\mathbb{G}_1^m$ or $\mathbb{G}_2^m$, for some $m \in \mathbb{N}$, requires only one group element [92, 86]. The power of the quasi-adaptive notion of zero-knowledge allows to specialize the common reference string to the language one is proving membership in, trading generality for efficiency under very weak computational assumptions. Other works have constructed proofs for different languages in the QA-NIZK setting, like the proof for bilateral spaces (linear spaces in $\mathbb{G}_1^m \times \mathbb{G}_2^n$) [67], or, beyond linear spaces, the language of vector commitments to integers opening to a boolean vector [67] or shuffles and range proofs [68].

Finally, as we explained in Section 1.2 in the last few years, an extremely successful line of research has constructed succinct non-interactive arguments of knowledge (zk-SNARKs) [71, 101, 61, 45, 72] for NP complete languages, which are not only constant-size (independent of the witness size) but which are also very efficient in a concrete sense. One of the main downsides of zk-SNARKs is that their security relies on knowledge of exponent assumptions, a very strong type of assumptions classified as non-falsifiable [105]. However, one cannot achieve succinctness (proofs essentially independent of the size of the statement being proved and its witness) and security based on falsifiable assumptions at the same time, as per the impossibility result by Gentry and Wichs [64].

Commit-and-Prove. In a broad sense, we can think of many of the results in these three families as commit-and-prove schemes [32]. This is very clear for the Groth-Sahai proof system, which has even been recasted in the commit-and-prove formalism by Escala and Groth [50]. This is probably less obvious for some results in the QA-NIZK setting. However, as noted already in the first QA-NIZK construction of membership in linear spaces [85], in these cases one can often think of the statement as a commitment to the witness. For instance, in the case of proving that a vector $\boldsymbol{y}$ in the exponent is in the linear span of the columns of some matrix $\mathbf{A}$, this means that $\boldsymbol{y} = \mathbf{A}\boldsymbol{w}$ and we can think of $\boldsymbol{y}$ as a commitment to $\boldsymbol{w}$. Finally, in the case of many zk-SNARK constructions, e.g. [45] the commitment is usually a "knowledge commitment" – from which the witness is extracted in the soundness proof using knowledge assumptions – while the rest can be considered the "proof".

With this idea in mind, it is interesting to compare these three approaches for con-

structing proofs of satisfiability of l equations in n variables in bilinear groups in terms of proof size. We observe that for linear equations, while the original Groth-Sahai proof system required $O(n)$ group elements for the commit step and $O(l)$ for the "prove" one, recent works have shown how to aggregate the proof in the quasi-adaptive setting [86, 67], reducing the "prove" step to $O(1)$ in many cases. For quadratic equations in the other hand, we summarize the three different approaches in Table 3.1. For concreteness, assume that one wants to prove that a set of values $x_1, \ldots, x_n$ form a bitstring, that is, satisfiability of $x_i(x_i - 1) = 0$. In this table we added two constructions that come after our contribution was published, González and Ràfols [67] for CicuitSat that is linear in the depth of the circuit, i.e. the number of multiplicative layers in the circuit, and the compack NIZK constructions of Katsumata et al. [88] under standard assumptions for all NP languages. We consider both as proofs for CircuitSat where the size of the circuit is the number of wires and multiplicative gates $(n + l)$.

Construction	Assumption	Commitment	Proof	crs
Groth-Sahai [77]	Standard	$O(n)$	$O(l)$	$O(1)$
Danezis et al. [45]	Non-falsifiable	$\lvert\mathbb{G}_1\rvert + \lvert\mathbb{G}_2\rvert$	$2\lvert\mathbb{G}_1\rvert$	$O(n + l)$
GonHevRaf15 [67]	Falsifiable	$O(n)$	$10\lvert\mathbb{G}_1\rvert + 10\lvert\mathbb{G}_2\rvert$	$O(l^2)$
GonRaf19 [67]	Falsifiable	$O(d + n_s)\lvert\mathbb{G}_1\rvert$ $+O(d)\lvert\mathbb{G}_2\rvert$	$O(d)\lvert\mathbb{G}_1\rvert$ $+O(d)\lvert\mathbb{G}_2\rvert$	$O(n + l)$
Katsumata et al. [88]	Standard	$n + \mathsf{poly}(\lambda)$	$l + \mathsf{poly}(\lambda)$	$O(n + l)$

Table 3.1: Different approaches for proving l quadratic equations with n variables in bilinear groups. Note that d denotes the depth of the circuit, n_s the secret input size. Consider $\lvert\mathbb{G}_1\rvert$ and $\lvert\mathbb{G}_2\rvert$ are multiplicative in λ.

Motivation. Quadratic equations are much more powerful than linear ones. In particular, they allow to prove boolean CircuitSat, but they are also important to prove other statements like range, shuffle proofs or validity of an encrypted vote. While for proving statements about large circuits non-falsifiable assumptions are necessary to get around impossibility results, it would be desirable to eliminate them in less demanding settings, to understand better what the security claims mean in a concrete sense. As in the QA-NIZK arguments for linear spaces, there are even natural situations in which the statement is already "an encrypted witness", and it seems unnatural to use the full power of knowledge of exponent assumptions in these cases (for instance, in the case of vote validity).

In summary, it is worth investigating efficiency improvements for quadratic equations under falsifiable assumptions. In particular, aggregating the "prove" step would

be an important step towards this goal. The techniques for the linear case do not apply to the quadratic one, and we are only aware of one result in aggregating the proof of quadratic equations, namely the bitstring argument of González et al. [67] for proving that a set of commitments to integers opens to boolean values. There is a large concrete gap between this result and the others in the non-falsifiable setting both in terms of the size of the proof and the common reference string. Thus, it is natural to ask if it is possible to reduce the gap and improve on this result importing techniques from zk-SNARKs in the falsifiable setting.

3.1.1 Our results

We introduce new techniques to aggregate proofs of quadratic equations. We summarize our constructions in Table 3.2.

First, in Section 3.3.1, we construct a proof system for proving that l equations of the type $X_i(X_i - 2) = 0$ are satisfied, where X_i is an affine combination of some $a_1, \ldots, a_n$. The size of the proof is constant and the set of commitments to the variables is of size linear in n, and the size of the crs is linear in l. The prover computes a number of exponentiations linear in $n + l$, while the verifier computes a number of pairings linear in l. Our proof system is perfect zero-knowledge and computationally sound under a variant of the so-called target strong Diffie-Hellman assumption, which is equivalent to it under KEA (proven in Section 3.2.1). These assumptions belong to the broader class of q-assumptions, where each instance of the problem is of size proportional to some integer q, which in our case is the number of equations. In particular, the bitstring language of [67] can be formulated as such a system of equations.

In Section 3.4 we discusss as a particular case an argument for unit vector, and argue how to modify our general proof system so that it can be proven sound under static assumptions. A typical application of membership in these languages is for computing disjunctions of statements such as "the committed verification key vk is equal to $\mathcal{V}_1$, or $\mathcal{V}_2$, ..., or $\mathcal{V}_m$", which might be expressed as $vk = \sum_{i=1}^{m} b_i \mathcal{V}_i, b_i \in \{0, 1\}$ and $(b_1, \ldots, b_m)$ is a unit vector.

Next, in Section 3.5, we generalize the previous argument to prove that d equations of the type $(X_i - z_1)(X_i - z_2) \ldots (X_i - z_m) = 0$ are satisfied, where X_i is an affine combination of the variables $a_1, \ldots, a_n$. For this we combine techniques from the interactive setting of [30] for proving set membership in a set of size $m \in \mathbb{Z}_p$ with ideas from Section 3.3.1 and from quasi-adaptive aggregation [86]. In Section 3.6.2, we illustrate how to use this for improve range proofs in bilinear groups under falsifiable assumptions.

Section	Language	Proof size	crs size	Assumption
3.3.1	Quadratic equations	$4\|\mathbb{G}_1\| + 6\|\mathbb{G}_2\|$	$(l + O(1))\|\mathbb{G}_1\|$ $+(l + 3n + O(1))\|\mathbb{G}_2\|$	q-STSDH (7)
3.4	Unit vector	$6\|\mathbb{G}_1\| + 6\|\mathbb{G}_2\|$	$(4(n + 1) + O(1))\|\mathbb{G}_1\|$ $+(5(n + 1) + O(1))\|\mathbb{G}_2\|$	1-STSDH (7)
3.5.2	Set Membership	$6\|\mathbb{G}_1\| + 6\|\mathbb{G}_2\|$	$(mn + 2n + 3m + O(1))\|\mathbb{G}_1\|$ $+(5mn + O(1))\|\mathbb{G}_2\|$	$\mathcal{Z}$-GSDH (6), q-QTSDH (8)

Table 3.2: The table shows the proof sizes (not including commitments) and crs sizes of our constructions. We consider l variables and n equations, and m is the size of the set from the set membership proof. The assumptions 6, 7 and 8 are new.

Finally, in Section 3.6.1 we discuss two approaches to construct shuffle arguments. They are the most efficient in terms of proof size in the common reference string model under falsifiable assumptions in bilinear groups (comparing favorably even to the best constructions in the algebraic group model [7]), but they have large public parameters (quadratic in the shuffle size). We give a comparison of our shuffle arguments with state-of-the-art arguments in Table 3.3.

Work	Proof size	crs size	Assumption
[73]	$15n + 246$	$2n + 8$	PPA, SPA, DLIN
[53]	$(4n - 1)\|\mathbb{G}_1\| + (3n + 1)\|\mathbb{G}_2\|$	$O(n)(\|\mathbb{G}_1\| + \|\mathbb{G}_2\|)$	GGM
[68]	$(4n + 17)\|\mathbb{G}_1\| + 14\|\mathbb{G}_2\|$	$O(n^2)\|\mathbb{G}_1\| + O(n)\|\mathbb{G}_2\|$	SXDH, SSDP [68]
Sect. 3.6.1	$(4n + 11)\|\mathbb{G}_1\| + 8\|\mathbb{G}_2\|$	$O(n^2)\|\mathbb{G}_1\| + O(n)\|\mathbb{G}_2\|$	SXDH, 1-STSDH (7)
Sect. 3.6.1	$(2n + 11)\|\mathbb{G}_1\| + 8\|\mathbb{G}_2\|$	$O(n^2)(\|\mathbb{G}_1\| + \|\mathbb{G}_2\|)$	SXDH, n-QTSDH (7)
[7]	$4n\|\mathbb{G}_1\| + 3n\|\mathbb{G}_2\|$	$O(n)(\|\mathbb{G}_1\| + \|\mathbb{G}_2\|)$	AGM

Table 3.3: Comparison of our shuffle arguments with state-of-the-art arguments. PPA stands for the Pairing Permutation Assumption and SPA for the Simultaneous Pairing Assumption.

3.1.2 Our techniques

Let $\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T$ be groups of prime order p and let $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ be a bilinear map. Both zk-SNARKs and our schemes can be seen as "commit-and-prove" schemes [50]: in the first step we commit to the solution of the equations. In the case of zk-SNARKs, the knowledge assumption allows to extract the solutions from a constant-size commitment during the soundness proof, but we are trying to avoid using these assumptions, so we require perfectly binding commitments for each element of the solution. The second step is a proof of the opening of the commitments verifying the equations.

Let $r_1, \ldots, r_l \in \mathbb{Z}_p$. The "prove" part is handled with a polynomial aggregation technique in which satisfiability of a set of l equations is encoded into a polynomial $p(X)$ such that $p(r_j) = 0$ if and only if the jth equation is satisfied. To prove that l equations are satisfied, one needs to prove that $p(X)$ is divisible by $\prod_{j=1}^{l}(X - r_j)$. The key to succinctness is that the divisibility condition is only checked at a secret point s chosen by the trusted party who generates the crs. This preserves soundness as long as the prover only knows s (or powers thereof) in $\mathbb{G}_1$ or $\mathbb{G}_2$, but not its discrete logarithm.

In the soundness proof, the witness is extracted from the knowledge commitment, and then used to find some r_j such that $p(r_j) \neq 0$ and compute auxiliary information which, together with the proof, allows to break a hard problem, e.g. the q-Target Strong Diffie-Hellman Assumption in [45]. Under non-falsifiable assumptions the commitments, even if perfectly binding, can be only opened in the source groups, instead of in $\mathbb{Z}_p$. This has an impact on the soundness proof, as it is not possible to eliminate some terms in the proof to find a solution to the q-TSDH assumption, so we need to consider a more flexible assumption. Furthermore, since the solutions define the coefficients of polynomial $p(X)$, our access to this polynomial is much more limited.

For our set-membership proof we start from the following insight: the satisfiability of equation $b(b-1) = 0$ can be proven showing knowledge of a signature for b if only signatures for 0 or 1 are known. This approach can be easily extended for larger sets of solutions as done by Camenisch et al. [30]. To express the validity of many signature and message pairs, we again encode the signature verification equations as a problem of divisibility of polynomials.

This requires the signature verification to be expressible as a set of quadratic equations. While structure preserving signatures clearly solve this problem, it is overkill, since we only need unforgeability against static queries. Further, even the generic group construction of [72] requires at least 3 group elements. We choose basic Boneh-Boyen signatures [23], defined in Section 2.6.3, since each signature consists of only one group element. Our argument needs to solve other technical difficulties which are explained in more detail in Section 3.5.

3.1.3 Related Works

The recent line of research in zk-SNARKs started with [71], in which the first sub-linear arguments without random oracles were presented, but with crs of quadratic size. Subsequent works have defined alternative models for the encoding of the circuit [100, 61, 45, 72], reducing the crs size to linear and obtaining smaller proofs, going as small as 3 group elements in the case of [72]. In particular, our encodings are based on those of [61, 45].

When considering falsifiable assumptions, one classic way to prove quadratic equations in the non-interactive setting makes use of Groth-Sahai proofs [78], which are quite efficient and can be aggregated to obtain a constant-size proof of many equations.

In this work, we also use techniques from QA-NIZK proofs. This model was introduced in [85] to build proofs of membership in linear subspaces over $\mathbb{G}_1$ or $\mathbb{G}_2$. It was later improved to make proofs constant-size (independent of the size of the witness) [86, 92, 97] and adapted to the asymmetric setting [67]. Although introduced initially to build proofs of linear equations, the QA-NIZK setting has also been used to build the first constant-size aggregated proofs of some quadratic equations under standard assumptions [67], in particular the proof that a set of commitments open to bits.

The usage of signatures for proving membership in a set dates back to the work of Camenisch et al. [30] in the interactive setting, and in the non-interactive setting by Rial et al. [114]. Both works achieve constant-size proofs but without aggregation (i.e. proving n instances requires $O(n)$ communication). Set membership proofs were also recently investigated by Bootle and Groth [25] in the interactive setting. They construct proofs logarithmic in the size of the set and aggregate n instances with a multiplicative overhead of $O(\sqrt{n})$. In the non-interactive setting, González et al. [68] constructed set membership proofs of size linear in the size of the set and aggregated many instances without any overhead.

Organization

This chapter is organized in the following sections. In Section 3.2 we present our new assumptions, prove they are falsifiable and secure in the GGM. As a novelty, compared to the published version [46], we prove in Section 3.2.1 the new assumptions are equivalent to the q-Target Strong Diffie Hellman assumption under the BDH-KE Assumption 5. In Section 3.3, we present the main construction for satisfiability of quadratic equations, where as a novelty compared to the published version we added the knowledge soundness proof for some specific cases. In Section 3.4, we give an argument to prove that a commitment opens to a unit vector which can be proven secure based on

a static assumption. In Section 3.5 we present an aggregated argument to prove membership in a set of $\mathbb{Z}_p$. In Section 3.6.1 we discuss new approaches to construct shuffle arguments and in Section 3.6.2, to construct range proofs.

3.2 New Falsifiable q-Assumptions

The soundness proofs of our schemes will rely on the following variations of the two assumptions q-SDH and q-TSDH recalled in Section 2.4.3.

Assumption 6 ($\mathcal{Z}$-Group Strong DH Assumption in $\mathbb{G}_\gamma$, $\mathcal{Z}$-GSDH)**.** *Let $\mathcal{Z} \subset \mathbb{Z}_p$ such that $\#\mathcal{Z} = q$. For all non-uniform PPT adversaries $\mathcal{A}$:*

$$\Pr\left[([z_1]_1, [z_2]_\iota, [\nu]_2) \leftarrow \mathcal{A}\left(gk, \mathcal{Z}, [\varepsilon]_{1,2}, \{[s^i]_{1,2}\}_{i=1}^q\right) : \begin{array}{c} z_1 \notin \mathcal{Z} \wedge z_2 = \varepsilon z_1 \\ \nu = \frac{\prod_{z\in\mathcal{Z}}(s-z)}{s-z_1} \end{array}\right] \approx 0,$$

where the probability is taken over $gk \leftarrow \mathcal{G}(1^\lambda)$, $s, \varepsilon \leftarrow \mathbb{Z}_p$ and the coin tosses of adversary $\mathcal{A}$.

The name is motivated by the fact that it is a variant of the q-SDH Assumption in which the adversary must only give $[z_1]_1$ in the group $\mathbb{G}_1$, instead of giving it in $\mathbb{Z}_p$ as in the q-SDH Assumption.

Assumption 7 ($\boldsymbol{q}$-Square TSDH Assumption, $\boldsymbol{q}$-STSDH)**.** *For all non-uniform PPT adversaries $\mathcal{A}$:*

$$\Pr\left[(r, [\beta_1]_1, [\beta_2]_2, [\nu]_T) \leftarrow \mathcal{A}\left(gk, [\varepsilon]_2, \{[s^i]_{1,2}\}_{i=1}^q\right) : \begin{array}{c} \beta_1 \neq \pm 1 \\ \beta_2 = \varepsilon\beta_1 \wedge \nu = \frac{\beta_1^2-1}{s-r} \end{array}\right] \approx 0,$$

where the probability is taken over $gk \leftarrow \mathcal{G}(1^\lambda)$, $s, \varepsilon \leftarrow \mathbb{Z}_p$ and the coin tosses of adversary $\mathcal{A}$.

Note that the challenger knows ε, s, so this assumption is falsifiable. Indeed, upon receiving $(r, [\beta_1]_1, [\beta_2]_2, [\nu]_T)$, the challenger verifies that $[\beta_1]_1 \neq [\pm 1]_1, e([1]_1, [\beta_2]_2) = e(\varepsilon[\beta_1]_1, [1]_2)$, and $\varepsilon(s-r)[\nu]_T = e([\beta_1]_1, [\beta_2]_2) - e([\varepsilon]_1, [1]_2)$. A similar argument can be made for the other assumptions in this section.

Assumption 8 ($\boldsymbol{q}$-Quadratic TSDH Assumption, $\boldsymbol{q}$-QTSDH)**.** *For all non-uniform PPT adversaries $\mathcal{A}$:*

$$\Pr\left[\begin{array}{c} \left(r, [\beta_1]_1, [\beta_2]_1, [\tilde{\beta}_1]_2, [\tilde{\beta}_2]_2, [\nu]_T\right) \leftarrow \mathcal{A}\left(gk, [\varepsilon]_{1,2}, \{[s^i]_{1,2}\}_{i=1}^q\right) : \\ \beta_1\tilde{\beta}_1 \neq 1 \\ \beta_2 = \varepsilon\beta_1 \wedge \tilde{\beta}_2 = \varepsilon\tilde{\beta}_1 \wedge \nu = \frac{\beta_1\tilde{\beta}_1-1}{s-r} \end{array}\right] \approx 0,$$

where the probability is taken over $gk \leftarrow \mathcal{G}(1^\lambda)$, $s, \varepsilon \leftarrow \mathbb{Z}_p$ *and the coin tosses of adversary* $\mathcal{A}$.

3.2.1 Hardness of Assumptions

In this section we prove our new assumptions are secure in the GGM and we prove our q-STSDH and q-QTSDH assumptions are equivalent to the q-TSDH assumptions under KEA.

Generic Hardness

Proposition 1. *The* $\mathcal{Z}$-GSDH *Assumption (6) in* $\mathbb{G}_\gamma$ *holds in the generic group model.*

Proof. A generic adversary receives $\mathcal{Z}$ in $\mathbb{Z}_p$, ε and the powers $1, s, \ldots, s^q$ in $\mathbb{G}_1$, and ε and $1, s, \ldots, s^q$ in $\mathbb{G}_2$. Then any z_1 output by the adversary must be of the form

$$z_1 = \sum_{i=0}^{q} b_i s^i,$$

for some coefficients $b_i, i \in \{0, \ldots, q\}$, and since $z_2 = \varepsilon z_1$, we have that necessarily

$$z_2 = \sum_{i=0}^{q} b_i \varepsilon s^i.$$

This forces $b_i = 0$ for $i \in \{1, \ldots, q\}$, since a generic adversary cannot compute εs^i in $\mathbb{G}_\gamma$. Thus $z_1 = b_0$.

Then the adversary cannot compute

$$\frac{\prod_{r \in \mathcal{Z}}(s - r)}{s - r_1}$$

in $\mathbb{G}_T$, since r_1 is not a root of $p(s) = \prod_{r \in \mathcal{Z}}(s - r)$, so the above is a rational function, which cannot be computed with group operations. □

Proposition 2. *The* q-STSDH *Assumption(7) holds in the generic group model.*

Proof. A generic adversary receives the powers $1, s, \ldots, s^q$ in $\mathbb{G}_1$, and ε and $1, s, \ldots, s^q$ in $\mathbb{G}_2$. Then any β_1 output by the adversary must be of the form

$$\beta_1 = \sum_{i=0}^{q} b_i s^i,$$

for some coefficients $b_i, i \in \{0, \ldots, q\}$, and since $\beta_2 = \varepsilon\beta_1$, we have that necessarily

$$\beta_2 = \sum_{i=0}^{q} b_i \varepsilon s^i.$$

This forces $b_i = 0$ for $i \in \{1, \ldots, q\}$, since a generic adversary cannot compute εs^i in $\mathbb{G}_1$. Thus $\beta_1 = b_0$. Now, if a generic adversary is able to compute $\frac{\beta_1^2 - 1}{s - r}$ in $\mathbb{G}_T$, necessarily there exist polynomials p_1, p_2 such that

$$\frac{b_0^2 - 1}{s - r} = p_1(s, \varepsilon) \cdot p_2(s),$$

where $\deg p_1, \deg p_2 \leq q$ and p_1 does not have terms in εs^i for any i. However, since β_1 is a constant with respect to s, ε, and $\beta_1^2 - 1 \neq 0$, the above is a rational function, which cannot be computed with group operations. □

Proposition 3. *The q-*QTSDH *Assumption (8) holds in the generic group model.*

Proof. A generic adversary receives ε and the powers $1, s, \ldots, s^q$ in $\mathbb{G}_1$, and $1, s, \ldots, s^q$ in $\mathbb{G}_2$. Then any β_1 output by the verifier must be of the form

$$\beta_1 = \sum_{i=0}^{q} b_i s^i + b_{q+1}\varepsilon,$$

for some coefficients $b_i, i \in \{0, \ldots, q+1\}$, and since $\beta_2 = \varepsilon\beta_1$, we have that necessarily

$$\beta_2 = \sum_{i=0}^{q} b_i \varepsilon s^i + b_{q+1}\varepsilon^2.$$

This forces $b_i = 0$ for $i \in \{1, \ldots, q\}$, since a generic adversary cannot compute εs^i in $\mathbb{G}_1$, and $b_{q+1} = 0$, since it cannot compute ε^2 either. Thus $\beta_1 = b_0$. Analogously, $\tilde{\beta}_1 = \tilde{b}_0$ for some constant $\tilde{b}_0$. Now, if a generic adversary is able to compute $\frac{\beta_1\tilde{\beta}_1 - 1}{s - r}$ in $\mathbb{G}_T$, necessarily there exist polynomials p_1, p_2 such that

$$\frac{b_0\tilde{b}_0 - 1}{s - r} = p_1(s, \varepsilon) \cdot p_2(s, \varepsilon),$$

where $\deg p_1, \deg p_2 \leq q$ and p_1 does not have terms in εs^i for any i. However, since $\beta_1\tilde{\beta}_1$ is a constant with respect to s, ε, and $\beta_1\tilde{\beta}_1 - 1 \neq 0$, the above is a rational function, which cannot be computed with group operations. □

Reduction to Knowledge Assumptions

We prove that if the Knowledge of Exponent Assumption in bilinear groups holds, then both q-Target Strong Diffie-Hellman (q-TSDH) and q-Square Target Strong Diffie-Hellman assumptions are equivalent, similarly it can be proven that both q-TSDH and q-QTSDH are equivalent as well.

Lemma 4. *Given a bilinear group $gk = (q, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T)$, if the q-STSDH assumption holds then the q-TSDH assumption holds.*

Proof. Assume that $\mathcal{A}$ is an adversary against the q-TSDH assumption, we construct another adversary $\mathcal{B}$ against q-STSDH assumption. This adversary $\mathcal{B}$ receives a challenge tuple $(gk, \{[s^i]_{1,2}\}_{i=1}^q, [\varepsilon]_2)$ and sends the elements $(gk, \{[s^i]_{1,2}\}_{i=1}^q)$ to $\mathcal{A}$. Then, $\mathcal{A}$ returns $(r, [\nu]_T)$ such that breaks q-TSDH, the adversary $\mathcal{B}$ chooses $\beta_1 \leftarrow \mathbb{Z}_p$ such that $\beta_1 \neq 1$ and sends to the Challenger $\left(r, [\beta_1]_1, \beta_1[\varepsilon]_2, (\beta_1^2 - 1)[\nu]_T\right)$ that breaks the q-STSDH assumption. □

Lemma 5. *Given a bilinear group $gk = (q, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T)$ where BDH-KE assumption holds, if the q-TSDH assumption holds then the q-STSDH assumption holds.*

Proof. Assume that $\mathcal{A}$ is an adversary against the q-STSDH assumption, we construct an another adversary $\mathcal{B}$ against q-TSDH assumption. This adversary $\mathcal{B}$ receives a challenge tuple $(gk, \{[s^i]_{1,2}\}_{i=1}^q)$, chooses $\varepsilon \leftarrow \mathbb{Z}_p$ and sends the elements $(gk, \{[s^i]_{1,2}\}_{i=1}^q, [\varepsilon]_2)$ to $\mathcal{A}$. Then, the adversary $\mathcal{A}$ returns $(r, [\beta_1]_1, [\beta_2]_2, [\nu]_T)$ that breaks q-STSDH. Now $\mathcal{B}$ computes $\frac{1}{\varepsilon}[\beta_2]_2$ which satisfies $e([\beta_1]_1, [1]_2) = e([1]_1, \frac{1}{\varepsilon}[\beta_2]_2)$. By the BDH-KE assumption there exists an extractor of β_1 that solves the q-TSDH assumption with $\left(r, \frac{1}{\beta_1^2 - 1}[\nu]_T\right)$. □

3.3 Proving Satisfiability of Quadratic Equations

In this section we present a scheme in which soundness is based on the q-STSDH Assumption.

3.3.1 Arguments for Quadratic Equations from q-Assumptions

Intuition

Given $n, l \in \mathbb{N}$, the number of variables and equations, respectively, we build a proof system for the family of languages

$$\mathcal{L}_{\text{quad},ck} = \left\{ ([\boldsymbol{c}]_1, \mathbf{V}, \boldsymbol{b}) \in \mathbb{G}_1^{2n} \times \mathbb{Z}_p^{n \times l} \times \mathbb{Z}_p^l \;\middle|\; \begin{array}{l} \exists \boldsymbol{a}, \boldsymbol{w} \in \mathbb{Z}_p^n \text{ s.t} \\ [\boldsymbol{c}]_1 = \mathsf{Com}_{ck}(\boldsymbol{a}, \boldsymbol{w}) \text{ and} \\ \left\{\boldsymbol{a}^\top \boldsymbol{v}_j + b_j\right\}_{j=1}^{l} \in \{0, 2\} \end{array} \right\}$$

where $[\boldsymbol{c}]_1 = \mathsf{Com}_{ck}(\boldsymbol{a}, \boldsymbol{w})$ is a vector of ElGamal encryption. This generalizes to any other perfectly binding commitment of the form $[\boldsymbol{c}]_1 = \mathsf{Com}_{ck}(\boldsymbol{a}; \boldsymbol{w}) = [\mathbf{U}_1 \boldsymbol{a} + \mathbf{U}_2 \boldsymbol{w}]_1$ for $ck = ([\mathbf{U}_1]_1, [\mathbf{U}_2]_1)$, and $[\mathbf{U}_1]_1, [\mathbf{U}_2]_1$ are from a witness sampleable distribution.

We follow the approach of Danezis et al. [45] and encode the equations

$$\boldsymbol{a}^\top \boldsymbol{v}_j + b_j \in \{0, 2\}$$

into a *Square Span Program (SSP)*: we construct $n+1$ polynomials $v_0(X), \ldots, v_n(X)$ and a target polynomial $t(X)$, where $\deg(v_i) < \deg(t) = l$ for all $i \in \{0, \ldots, n\}$. This codification asserts that a witness $\boldsymbol{a}$ satisfies the set of equations if and only if $t(X)$ divides $p(X)$, where

$$p(X) = \left(v_0(X) + \sum_{i=1}^{n} a_i v_i(X) \right)^2 - 1.$$

The polynomials $v_i(X)$, $i \in \{1, \ldots, n\}$, are defined as the interpolation polynomials of the coefficients v_{ij} of $\mathbf{V}$ at $r_1, \ldots, r_l$, which are fixed, arbitrary, pairwise different points of $\mathbb{Z}_p$. Similarly, $v_0(X)$ is the interpolation polynomial of $b_j - 1$ at the same points. That is, if $\boldsymbol{v}_j$ is the jth column of $\mathbf{V}$,

$$\boldsymbol{a}^\top \boldsymbol{v}_j + b_j - 1 = \sum_{i=1}^{n} a_i v_{ij} + b_j - 1 = \sum_{i=1}^{n} a_i v_i(r_j) + v_0(r_j).$$

Note that the statement $Z \in \{0, 2\}$ is equivalent to $(Z-1)^2 - 1 = 0$ and hence, the polynomial $p(X)$ interpolates the left side of this equation in $r_1, \ldots, r_l$ when Z is replaced by $\boldsymbol{a}^\top \boldsymbol{v}_j + b_j - 1$ for each $j \in \{1, \ldots, l\}$. The target polynomial $t(X) = \prod_{i=1}^{l} (X - r_i)$ is 0 at $r_1, \ldots, r_l$ and therefore encodes the right sides. This codification gives us the equivalence: the equations hold if and only if $t(X)$ divides $p(X)$.

Danezis et al. constructed a zk-SNARK for this statement, "$t(X)$ divides $p(X)$", which is very efficient because it just checks that the divisibility relation holds at a single secret point $s \in \mathbb{Z}_p$ whose powers $[s]_1, [s]_2, \ldots, [s^l]_1, [s^l]_2$ are published in the crs. That is, the proof essentially shows "in the exponent" that

$$p(s) = h(s)t(s),$$

where $h(X) = p(X)/t(X)$. When all the equations hold, $h(X)$ is a polynomial and the evaluation at s can be constructed as a linear combination of the powers of s in the crs. When some equation does not hold, $h(X)$ is a rational function, and its evaluation at s is no longer efficiently computable from the crs. The actual proof system has some additional randomization elements to achieve Zero-Knowledge, but its soundness follows from this argument.

In the scheme of Danezis et al., the prover outputs a perfectly hiding commitment to the witness. In the soundness proof, one uses a knowledge of exponent assumption to extract the witness in $\mathbb{Z}_p^n$ from the commitment. The witness is used to derive a reduction from breaking soundness to the l-TSDH Assumption. More precisely, it follows from the SSP characterization that if the equation with index j^* does not hold, then $p(X) = q(X)(X - r_{j^*}) + b$, for some $b \neq 0$. From the extracted value of the witness $\boldsymbol{a}$ one can identify at least one such j^* and also recover the coefficients of $q(X)$ and the value b in $\mathbb{Z}_p$. From the verification equation, the reduction can obtain

$$\left[\frac{p(s)}{s - r_{j^*}}\right]_T = \left[q(s) + \frac{b}{s - r_{j^*}}\right]_T \tag{3.1}$$

and using $b, q(s)$ derive $\left[\frac{1}{s - r_{j^*}}\right]_T$.

In other words, there are two ways in which the Danezis et al.'s scheme (as well as most other zk-SNARKs) use knowledge assumptions: (a) extracting vectors of committed values from one single group element (beyond what is information-theoretically possible), and (b) extract in the base field, so computing discrete logarithms. Our goal is to avoid knowledge of exponent assumptions, so to circumvent (a) we change the scheme to include perfectly binding commitments to the witness. However, we still have to deal with (b), as our commitments to $\boldsymbol{a}$ can only be opened to $[\boldsymbol{a}]_\gamma \in \mathbb{G}_\gamma$. Therefore, we are no longer able to compute $[q(s)]_T$ since it requires to compute terms of the form $[a_i a_j s^k]_T$ from $[a_i]_1, [a_j]_2$ and powers of s in one of the groups, in any case it would be a multiplication of three group elements.

At this point, we would like to be able to include in the proof a commitment that allows the reduction to extract $q(s)$, but the fact that $q(s)$ is "quadratic" in the witness makes this difficult. For this reason, we factor $q(X)$ into two polynomials $q_1(X)$ and

$q_2(X)$. In the soundness game we will program the crs[1] to depend on an index j^* and let the prover compute binding commitment to $[q_2(s)]_2$, while $[q_1(s)]_1$ can be directly computed from the proof. From these factors we are able to compute $[q(s)]_T$. However, extracting b in $\mathbb{Z}_p$ to obtain a reduction to the l-TSDH problem seems difficult, so we will rely on a more flexible security assumption where we do not need to remove b. The idea of the new assumption is to give the adversary powers of s in the source groups and ask the adversary to output

$$\left(r_{j^*}, [\beta]_1, \left[\frac{b}{s - r_{j^*}}\right]_T\right), \text{ where } \beta^2 - 1 = b.$$

However, this is not a hard problem, as the adversary can set b as a combination of $s - r_{j^*}$ to achieve elimination of the denominator in $\frac{b}{s-r_{j^*}}$. For example, if an adversary sets $\beta = s - r_{j^*} + 1$, it can compute a valid solution as $(r_{j^*}, [\beta]_1, [s - r_{j^*} + 2]_T)$, since $\beta^2 - 1 = (s - r_{j^*} + 2)(s - r_{j^*})$. To prevent this type of attacks from happening, we add an element $[\varepsilon]_2 \in \mathbb{G}_2$ to the challenge, and ask the adversary to output $[\varepsilon\beta]_2$ too, so that β cannot be set as a function of s (since the adversary will not be able to compute εs in $\mathbb{G}_2$). We call the modified assumption the l-STSDH, which is proven to be generically secure and equivalent to TSDH under KEA in Section 3.2.1). Further, it can be easily checked that the assumption is falsifiable as we note in Section 3.2. To make sure that we can extract $[\varepsilon\beta]_2$ from the prover's output and also that the rest of the elements of the proof are of the right form, we will require the prover to show that its output is in a given linear space.

Scheme description

Given $n, l \in \mathbb{N}$ we construct a QA-NIZK argument for the language $\mathcal{L}_{\text{quad},\mathtt{ck}}$.

K_0: The algorithm $\mathsf{K}_0(gk, n, l)$ samples $\mathtt{ck} = [\boldsymbol{u}]_1 \leftarrow \mathcal{L}_1$. A commitment $\mathsf{Com}_{\mathtt{ck}}(\boldsymbol{a}; \boldsymbol{w})$ is the concatenation of $\mathsf{Enc}_{\mathtt{ck}}(a_i; w_i) = [a_i \boldsymbol{e}_2 + w_i \boldsymbol{u}]_1$. That is, $\mathsf{Com}_{\mathtt{ck}}(\boldsymbol{a}; \boldsymbol{w}) = [\mathbf{U}_1 \boldsymbol{a} + \mathbf{U}_2 \boldsymbol{w}]_1$, where $\mathbf{U}_1, \mathbf{U}_2$ are $2n \times n$ matrices such that $\mathbf{U}_1$ has $\boldsymbol{e}_2$ in the diagonal and $\mathbf{U}_2$ has $\boldsymbol{u}$ in the diagonal.

K_1: The algorithm $\mathsf{K}_1(gk, \mathtt{ck}, n, l)$ picks $s \leftarrow \mathbb{Z}_p$, $\left\{\hat{\boldsymbol{\phi}}_i\right\}_{i \in \{1,\ldots,n+1\}} \leftarrow \mathbb{Z}_p^3$, $\mathbf{Q}_2 \leftarrow \mathcal{U}_{3,3}$ and generates also the crs for proving membership in bilateral linear spaces

[1] This is why we lose a factor $1/l$ in the soundness reduction.

of Section 3.2, BLS.CRS, for the linear spaces generated by the matrices:

$$[\mathbf{M}]_1 = \left[\begin{array}{ccc|c|cc} \boldsymbol{e}_2 & & & \boldsymbol{u} & & \\ & \ddots & & & \ddots & \mathbf{0} \\ & & \boldsymbol{e}_2 & & & \boldsymbol{u} \\ \hline v_1(s) & \dots & v_n(s) & \mathbf{0} & t(s) & \mathbf{0} \end{array}\right]_1,$$

$$[\mathbf{N}]_2 = \left[\begin{array}{ccc|c|cc} v_1(s) & \dots & v_n(s) & \mathbf{0} & t(s) & \mathbf{0} \\ \hat{\boldsymbol{\phi}}_1 & \dots & \hat{\boldsymbol{\phi}}_n & & \hat{\boldsymbol{\phi}}_{n+1} & \mathbf{Q}_2 \end{array}\right]_2$$

$[\mathbf{M}]_1 \in \mathbb{G}_1^{(2n+1)\times(2n+4)}$, $[\mathbf{N}]_2 \in \mathbb{G}_2^{4\times(2n+4)}$.
The crs includes the elements

$$\left(gk, \mathtt{ck}, \left\{\left[s^i\right]_{1,2}\right\}_{i\in\{1,\dots,l\}}, \left\{\left[\hat{\boldsymbol{\phi}}_i\right]_2\right\}_{i\in\{1,\dots,n+1\}}, [\mathbf{Q}_2]_2, \mathsf{BLS.CRS}\right).$$

P: The prover P with input $(\mathsf{crs}, [\boldsymbol{c}]_1, \mathbf{V}, \boldsymbol{b}, \boldsymbol{a})$ picks $\delta \leftarrow \mathbb{Z}_p, \boldsymbol{r}_{q.2} \leftarrow \mathbb{Z}_p^3$ and defines the polynomial

$$p(X) = \left(v_0(X) + \sum_{i=1}^{n} a_i v_i(X) + \delta t(X)\right)^2 - 1 \in \mathbb{Z}_p[X],$$

where each $v_i(X)$, for $i \in \{1,\dots,n\}$, is the interpolation polynomial of the components v_{ij} of $\mathbf{V}$ at points r_j, for $j \in \{1,\dots,l\}$, and $v_0(X)$ is the interpolation polynomial of $b_j - 1$ at the same points. It then computes $h(X) = \frac{p(X)}{t(X)}$, which is a polynomial in $\mathbb{Z}_p[X]$ because $\boldsymbol{a}$ satisfies the equations, and the following elements:

$$[V]_1 = [\textstyle\sum_{i=1}^n a_i v_i(s) + \delta t(s)]_1 \qquad [V]_2 = [\textstyle\sum_{i=1}^n a_i v_i(s) + \delta t(s)]_2$$
$$[H]_1 = [h(s)]_1 \qquad [\boldsymbol{q}_2]_2 = \left[\textstyle\sum_{i=1}^n a_i \hat{\boldsymbol{\phi}}_i + \delta\hat{\boldsymbol{\phi}}_{n+1} + \mathbf{Q}_2\boldsymbol{r}_{q.2}\right]_2.$$

The prover can compute all these elements as linear combinations of the powers of s in the crs. The prover also computes a BLS proof ψ of

$$([\boldsymbol{c}]_1, [V]_1, [V]_2, [\boldsymbol{q}_2]_2)^\top \in \mathbf{Im}\begin{pmatrix} [\mathbf{M}]_1 \\ [\mathbf{N}]_2 \end{pmatrix}$$

with witness $(\boldsymbol{a}, \boldsymbol{w}, \delta, \boldsymbol{r}_{q.2})^\top \in \mathbb{Z}_p^{2n+4}$.
Finally, it sends the proof π to the verifier, where $\pi := \left([H]_1, [V]_{1,2}, [\boldsymbol{q}_2]_2, \psi\right)$.

V**:** The verifier V with input $(\mathsf{crs}, [\boldsymbol{c}]_1, \mathbf{V}, \boldsymbol{b}, \pi)$ checks whether the equation

$$e([v_0(s)+V]_1, [v_0(s)+V]_2) - [1]_T = e([H]_1, [t(s)]_2) \tag{3.2}$$

holds and $\mathsf{BLS.verify}(\psi) = 1$. If both conditions hold, it returns 1, else it returns 0.

Completeness

This property is based on the perfect completeness of membership in bilateral spaces, and the observation that the left hand side of the verification equation is

$$e\left([v_0(s)+V]_1, [v_0(s)+V]_2\right) - [1]_T = \left[(v_0(s)+V)^2 - 1\right]_T = [p(s)]_T,$$

and the right hand side is $e\left([H]_1, [t(s)]_2\right) = e\left([h(s)]_1, [t(s)]_2\right) = [p(s)]_T$.

Soundness

We introduce two technical lemmas that we will use in the following to prove the soundness of the scheme. We define $\mathcal{U}_{k,k,r}$ to be the uniform distribution on $k \times k$ matrices over $\mathbb{Z}_p$ with rank r.

Lemma 6. *For any $k, r \in \mathbb{N}, r < k$, there exists an $\mathcal{L}_1$-$\mathsf{MDDH}_{\mathbb{G}_1}$ PPT adversary $\mathcal{B}_0$ such that for any PPT adversary $\mathcal{A}$*

$$\Pr[\mathbf{M} \leftarrow \mathcal{U}_{k,k,r+1} : \mathcal{A}([\mathbf{M}]_1) = 1] - \Pr[\mathbf{M} \leftarrow \mathcal{U}_{k,k,r} : \mathcal{A}([\mathbf{M}]_1) = 1]| \\ \leq \mathrm{Adv}_{\mathcal{L}_1\text{-}\mathsf{MDDH},\mathbb{G}_1}(\mathcal{B}_0).$$

Proof. Direct application of Theorem 1 of [119]. □

Lemma 7. *Let $v(X)$ be a polynomial in $\mathbb{Z}_p[X]$. For any $r \in \mathbb{Z}_p$, we define $q_2(X)$ and β as the quotient and remainder, respectively, of the polynomial division of $v(X)$ by $X - r$, i.e. $v(X) = q_2(X)(X-r) + \beta$. If $p(X) = v(X)^2 - 1$, then*

$$p(X) = (v(X) + \beta)\, q_2(X)(X-r) + \beta^2 - 1.$$

Proof. By definition, $p(X) = v(X)^2 - 1$, if we expand this expression using the definition of $q_2(X)$ we have:

$$\begin{aligned} p(X) =& v(X)\left(q_2(X)(X-r) + \beta\right) - 1 = v(X)q_2(X)(X-r) + v(X)\beta - 1 \\ =& v(X)q_2(X)(X-r) + q_2(X)(X-r)\beta + \beta^2 - 1 \\ =& (v(X)+\beta)q_2(X)(X-r) + \beta^2 - 1. \quad \square \end{aligned}$$

□

Theorem 8. *Let* $\mathsf{Adv}_{\text{Sound}}(\mathcal{A})$ *be the advantage of any PPT adversary* $\mathcal{A}$ *against the soundness of the scheme. There exist PPT adversaries* $\mathcal{B}_1, \mathcal{B}_3$ *against the* $\mathcal{L}_1$-$\mathsf{MDDH}_{\mathbb{G}_2}$ *and* l-STSDH *Assumptions, respectively, and an adversary* $\mathcal{B}_2$ *against strong soundness (Definition 20) of the* BLS *argument such that*

$$\mathsf{Adv}_{\text{Sound}}(\mathcal{A}) \leq l\Big(2\mathsf{Adv}_{\mathcal{L}_1\text{-}\mathsf{MDDH},\mathbb{G}_2}(\mathcal{B}_1) + \mathsf{Adv}_{\mathsf{BLS}}(\mathcal{B}_2) + \mathsf{Adv}_{l\text{-}\mathsf{STSDH}}(\mathcal{B}_3)\Big).$$

Proof. In order to prove soundness we will prove indistinguishability of the following games.

- Real: This is the real soundness game. The output is 1 if the adversary produces a false accepting proof, i.e. if there is some equation $\boldsymbol{a}^\top \boldsymbol{v}_i + b_i \notin \{0, 2\}$ and the verifier accepts the proof.
- Game_0: This game is identical to the previous one, except that the commitment key $\boldsymbol{u}$ is chosen by the game.
- Game_1: This game is identical to the previous one, except that some $j^* \leftarrow \{1, \ldots, l\}$ is chosen and the game aborts if $\boldsymbol{a}$ satisfies the j^*-th equation, i.e. $[\boldsymbol{a}]_1^\top \boldsymbol{v}_{j^*} + [b_{j^*}]_1 \in \{[0]_1, [2]_1\}$.
- Game_2: For $r = r_{j^*}$ and $i \in \{1, \ldots, n+1\}$ let $\alpha_i(X)$ and β_i be the quotient and the reminder of the polynomial division of $v_i(X)$ by $X - r_{j^*}$ if $i \in \{1, \ldots, n\}$, and of $t(X)$ by $X - r_{j^*}$ if $i = n + 1$. This game is identical to the previous one, except that $\mathbf{Q}_2$ is now a uniformly random matrix conditioned on having rank 1, and each $\left[\hat{\boldsymbol{\phi}}_i\right]_2$ is changed to

$$\left[\hat{\boldsymbol{\phi}}_i\right]_2 = [\alpha_i(s)]_2 \boldsymbol{e}_2 + \beta_i [\varepsilon]_2 \boldsymbol{e}_3 + [\mathbf{Q}_2]_2 \boldsymbol{r}_i,$$

  where $\varepsilon \leftarrow \mathbb{Z}_p$, $\boldsymbol{r}_i \leftarrow \mathbb{Z}_p^3$ and $\boldsymbol{e}_i$ is the ith vector of the canonical basis of $\mathbb{Z}_p^3$.

Obviously, the games Real and Game_0 are indistinguishable.

Lemma 9. $\Pr[\mathsf{Game}_0(\mathcal{A}) = 1] \leq l \cdot \Pr[\mathsf{Game}_1(\mathcal{A}) = 1]$.

Proof. If $\mathcal{A}$ breaks soundness, at least one equation does not hold. Thus the challenger has at least a probability of $1/l$ of guessing this equation. □

Lemma 10. *There exists a* $\mathcal{L}_1$-$\mathsf{MDDH}_{\mathbb{G}_2}$ *adversary* $\mathcal{B}_1$ *such that*

$$|\Pr[\mathsf{Game}_1(\mathcal{A}) = 1] - \Pr[\mathsf{Game}_2(\mathcal{A}) = 1]| \leq 2\mathsf{Adv}_{\mathcal{L}_1\text{-}\mathsf{MDDH},\mathbb{G}_2}(\mathcal{B}_1).$$

Proof. We construct an adversary $\mathcal{B}_1$ against the $\mathcal{U}_{3,3}$-rank problem in which it receives $[\mathbf{Q}_2]_2 \in \mathbb{G}_2^{3\times 3}$ as input and must decide if the matrix has rank 1 or rank 3. $\mathcal{B}_1$ constructs the elements of crs as in the previous game, but it uses $[\mathbf{Q}_2]_2$ as commitment key and defines $[\hat{\boldsymbol{\phi}}_i]_2$ as:

$$\left[\hat{\boldsymbol{\phi}}_i\right]_2 = [\alpha_i(s)]_2 \boldsymbol{e}_2 + [\mathbf{Q}_2]_2 \boldsymbol{r}_i, \qquad \text{where } \boldsymbol{r}_i \leftarrow \mathbb{Z}_p^3.$$

If $[\mathbf{Q}_2]_2$ has full rank, then $[\mathbf{Q}_2]_2 \boldsymbol{r}_i$ is a uniformly distributed element of $\mathbb{G}_2^3$, so adversary perfectly simulates Game_1, else it perfectly simulates Game_2.

We conclude by using the reduction between the rank problem and the $\mathcal{L}_1$-$\mathsf{MDDH}_{\mathbb{G}_2}$ problem, as established in Lemma 6. □

Lemma 11. *There exists an adversary $\mathcal{B}_2$ against the strong soundness of the* BLS *proof and a l-*STSDH *adversary $\mathcal{B}_3$ such that*

$$\Pr[\mathsf{Game}_3(\mathcal{A}) = 1] \leq \mathsf{Adv}_{\mathsf{BLS}}(\mathcal{B}_2) + \mathsf{Adv}_{l\text{-}\mathsf{STSDH}}(\mathcal{B}_3).$$

Proof. For any adversary which breaks soundness $\mathcal{A}$, let E be the event that

$$([\boldsymbol{c}]_1, [V]_1, [V]_2, [\boldsymbol{q}_2]_2)^\top \in \mathbf{Im} \begin{pmatrix} [\mathbf{M}]_1 \\ [\mathbf{N}]_2 \end{pmatrix}$$

of Section 2.7.1 and $\overline{E}$ be the complementary event. Obviously,

$$\Pr[\mathsf{Game}_3(\mathcal{A}) = 1] \leq \Pr[\mathsf{Game}_3(\mathcal{A}) = 1 | E] + \Pr[\mathsf{Game}_3(\mathcal{A}) = 1 | \overline{E}]. \tag{3.3}$$

We can bound the second summand by the advantage of an adversary $\mathcal{B}_2$ against the strong soundness of BLS. Such an adversary receives $[\mathbf{M}]_1, [\mathbf{N}]_2$ sampled according to the distribution specified by Game_3 and the witness that proves that $\mathbf{M}, \mathbf{N}$ are sampled according to this distribution, which is s (see strong soundness, defined in Section 2.7.1). It also generates the BLS.CRS, and the rest of the crs is chosen in the usual way. Adversary $\mathcal{B}_2$ can use the output of $\mathcal{A}$ to break the soundness of BLS in a straightforward way.

In the following, we bound the first term of the sum in equation (3.3) by constructing an adversary $\mathcal{B}_3$ which breaks the l-STSDH Assumption in the case that E happens. Note that in this case there exists a witness $(\boldsymbol{a}, \boldsymbol{w}, \delta, \boldsymbol{r}_{q.2})^\top$ of membership in $\mathbf{Im} \begin{pmatrix} [\mathbf{M}]_1 \\ [\mathbf{N}]_2 \end{pmatrix}$. Further, this witness is partially unique, because $[\boldsymbol{c}]_1$ is a perfectly binding commitment, so $\boldsymbol{a}, \boldsymbol{w}, \delta$ are uniquely determined, and in particular this uniquely determines the polynomial $p(X)$.

We now describe the full reduction. Adversary $\mathcal{B}_3$ receives a challenge of the l-STSDH Assumption and plugs it in the crs. The rest of the elements are chosen by adversary $\mathcal{B}_3$ with the distribution specified by the game. The crs is then sent to the soundness adversary $\mathcal{A}$, who eventually outputs π for the corresponding $[\boldsymbol{c}]_1$.

Adversary $\mathcal{B}_3$ extracts $[\boldsymbol{a}]_1 \in \mathbb{G}_1$ from the knowledge of $\boldsymbol{u} \in \mathbb{Z}_p^2$ and aborts if the j^*-th equation is satisfied. By definition $e([v_0(s)+V]_1, [v_0(s)+V]_2) - [1]_T = [p(s)]_T$. If we divide both sides of the verification equation (3.2) by $s - r_{j^*}$,

$$\left[\frac{p(s)}{s-r_{j^*}}\right]_T = e\left([H]_1, \left[\frac{t(s)}{s-r_{j^*}}\right]_2\right) = e\left([H]_1, \left[\prod_{i\neq j^*}(s-r_i)\right]_2\right), \tag{3.4}$$

so the adversary $\mathcal{B}_3$ can compute $\left[\frac{p(s)}{s-r_{j^*}}\right]_T$ from $[H]_1$ and the powers of $[s]_{1,2}$ in the crs. On the other hand, if we apply Lemma 7 to $p(X)$, we have

$$\left[\frac{p(s)}{s-r_{j^*}}\right]_T = \left[(v(s)+\beta)q_2(s) + \frac{\beta^2-1}{s-r_{j^*}}\right]_T, \tag{3.5}$$

and we have $\beta^2 - 1 \neq 0$ (otherwise the j^*-th equation is satisfied, in which case the game aborts). We describe in the following how $\mathcal{B}_3$ can compute right side of (3.5) and the elements to break the l-STSDH Assumption.

$\mathcal{B}_3$ can compute $[\beta]_1 = \sum_{i=0}^{n}[a_i]_1\beta_i$ and also $[v(s)+\beta]_1 = [V]_1 + [\beta]_1$, because it knows $[V]_1$ from the proof π and the extracted values $[a_i]_1$, and β_i are the reminders of dividing $v_i(X)$ by $X - r_{j^*}$.

Since $\mathcal{B}_3$ sampled $\mathbf{Q}_2$ itself, it can recover $[q_2(s)]_2$ and $[\varepsilon\beta]_2$ from $[\boldsymbol{q}_2]_2$ because it can compute two vectors $\boldsymbol{v}_2, \boldsymbol{v}_3 \in \mathbb{Z}_p^3$ such that $\boldsymbol{v}_i^\top[\mathbf{Q}_2]_2 = \mathbf{0}$, $\boldsymbol{v}_i^\top \boldsymbol{e}_j = 0$ if $i \neq j$ and $\boldsymbol{v}_i^\top \boldsymbol{e}_j = 1$ if $i = j$. $\mathcal{B}_3$ multiplies these vectors by $\boldsymbol{q}_2$ (which is correctly computed, because E holds), resulting in:

$$\boldsymbol{v}_2^\top[\boldsymbol{q}_2]_2 = \left[\boldsymbol{v}_2^\top \sum_{i=1}^{n+1} a_i\left(\alpha_i(s)\boldsymbol{e}_2 + \beta_i\varepsilon\boldsymbol{e}_3 + \mathbf{Q}_2\boldsymbol{r}_i\right) + \boldsymbol{v}_2^\top\mathbf{Q}_2\boldsymbol{r}_{q.2}\right]_2 = \left[\sum_{i=1}^{n+1} a_i\alpha_i(s)\right]_2,$$

$$\boldsymbol{v}_3^\top[\boldsymbol{q}_2]_2 = \left[\sum_{i=1}^{n+1} a_i\beta_i\varepsilon\right]_2.$$

From these values, $\mathcal{B}_3$ can compute $[q_2(s)]_2$ and $[\varepsilon\beta]_2$ by adding $[\alpha_0(s)]_2$ and $\beta_0[\varepsilon]_2$ to the above extracted elements, respectively:

$$\left[\alpha_0(s) + \sum_{i=1}^{n+1} a_i\alpha_i(s)\right]_2 = [q_2(s)]_2, \qquad \beta_0[\varepsilon]_2 + \left[\varepsilon\sum_{i=1}^{n+1} a_i\beta_i\right]_2 = [\varepsilon\beta]_2.$$

From these values and $[v(s)+\beta]_2$, computed above, $\mathcal{B}$ can derive $[(v(s)+\beta)q_2(s)]_T$ as $e([v(s)+\beta]_1, [q_2(s)]_2)$, and from equation (3.5) recover $\left[\frac{\beta^2-1}{s-r_{j^*}}\right]_T$.

Finally, $\mathcal{B}_3$ returns $\left(r_{j^*}, [\beta]_1, [\varepsilon\beta]_2, \left[\frac{\beta^2-1}{s-r_{j^*}}\right]_T\right)$, breaking the l-STSDH Assumption. □

With the last lemma we finalize the proof of strong soundness security of the Theorem 8. □

Knowledge Soundness

Now, we analyse the extractability of the scheme. The definitions used in this section are recalled in Section 2.7.1. First we show the scheme above is f-extractable where f is the exponentiation function in the group. Then, we prove plain extraction of the witness when the matrix $\mathbf{V}$ of the language has rank at least n. Intuitively, this condition sets the system of equations has a unique solution, then it is efficient to compute it solving the system. Then, when this condition holds our proof in Section 3.3.1 is knowledge sound.

Theorem 12. *For any adversary $\mathcal{A}$ able to produce a valid proof for the language $\mathcal{L}_{quad,\mathtt{ck}}$ there is a universal PPT $[\cdot]$-extractor $\mathcal{E}$ that can extract efficiently a $[\cdot]$-witness when the proof is accepted.*

Proof. Let ExtractSetup be an algorithm that samples the trapdoor $\boldsymbol{u}_{\mathcal{E}} \leftarrow \mathcal{L}_1$ and the crs identically distributed to the ones generated by the Setup algorithm of the argument. Let $\mathcal{E}$ be an algorithm that receives $(\mathsf{crs}, \boldsymbol{u}_{\mathcal{E}}, [\boldsymbol{c}]_1, \mathbf{V}, \boldsymbol{b}, \pi)$. It computes the orthogonal vector to the trapdoor $\boldsymbol{u}_{\mathcal{E}}^{\perp}$ such that $\boldsymbol{u}_{\mathcal{E}}^{\perp}\boldsymbol{e}_2^{\top} = 1$ and then extracts the witness in the group by computing: $\boldsymbol{u}_{\mathcal{E}}^{\perp}[c_i^{\top}]_1 = \boldsymbol{u}_{\mathcal{E}}^{\perp}[a_i\boldsymbol{e}_2 + w_i\boldsymbol{u}_{\mathcal{E}}] = \left[a_i\boldsymbol{u}_{\mathcal{E}}^{\perp}\boldsymbol{e}_2\right]_1 = [a_i]_1 \in \mathbb{G}_1$ for all $i \in [n]$. □

Theorem 13. *For any adversary $\mathcal{A}$ able to produce a valid proof for $([\boldsymbol{c}]_1, \mathbf{V}, \boldsymbol{b}) \in \mathcal{L}_{quad,ck}$ for a matrix $\mathbf{V}$ of rank at least n, there is a PPT extractor $\mathcal{E}$ that can extract efficiently a witness from such a proof with overwhelming probability when the proof is accepted.*

Proof. Let ExtractSetup and $\mathcal{E}$ be the Setup and the extraction algorithms constructed in the same way as in the previous proof for $f(\cdot)$-extractability. We have proven that $\mathcal{E}$ can extract the solution vector in the group $[\boldsymbol{a}]_1 \in \mathbb{G}_1^n$.

Since this argument is computational sound (Section 3.3.1), we know the equations $\{\boldsymbol{a}^\top \boldsymbol{v}_j + b_j\}_{j=1}^l \in \{0, 2\}$ hold, which are equivalent to $\{\boldsymbol{a}^\top \boldsymbol{v}\}_{j=1}^l \in \{-b_j, 2 - b_j\}$. The algorithm $\mathcal{E}$ can check efficiently if $\boldsymbol{a}^\top \boldsymbol{v}_j = -b_j$ or $\boldsymbol{a}^\top \boldsymbol{v}_j = 2 - b_j$ for each j because $\boldsymbol{v}_j, b_j$ are given in the statement. In l steps it can define c_j as the correct option in each case, for $j = 1, \ldots, l$ the extractor $\mathcal{E}$ knows the equations:

$$\boldsymbol{a}^\top \mathbf{V} = \boldsymbol{c} \in \mathbb{G}_1^l$$

where the entrances of $\mathbf{V}$ and $\boldsymbol{c}$ are known in $\mathbb{Z}_p$. So, since the matrix $\mathbf{V}$ has rank at least n, it is enough by the extractor $\mathcal{E}$ to compute the solution $\boldsymbol{a} \in \mathbb{Z}_p^n$ efficiently just solving the system (by gaussian elimination or LU decomposition methods). Moreover, it just needs n independent equations of the total l equations. □

Zero-Knowledge

We describe the simulation algorithms $(\mathsf{S}_1, \mathsf{S}_2)$ in the following.

S_1: The crs simulator $\mathsf{S}_1(gk)$ outputs $(\mathsf{crs}, \mathsf{tr} = \{s\}, \mathsf{tr}_{\mathsf{BLS}})$, the common reference string computed in the usual way plus the simulation trapdoor $s \in \mathbb{Z}_p$ and the simulation trapdoor of the bilateral spaces membership proof.

S_1: The simulator S_2 with input $(\mathsf{crs}, [\boldsymbol{c}]_1, \mathsf{tr}, \mathsf{tr}_{\mathsf{BLS}})$ samples $V^S \in \mathbb{Z}_p$, $\left[\boldsymbol{q}_2^S\right]_2 \leftarrow \mathbb{G}_2^3$, and defines:

$$\left[H^S\right]_1 = \left[\frac{(V^S)^2 - 1}{t(s)}\right]_1.$$

S_2 also constructs $\psi^S \leftarrow \mathsf{BLS.simulator}(\mathsf{crs}, [\boldsymbol{c}]_1, \left[V^S\right]_1, \left[V^S\right]_2, \left[\boldsymbol{q}_2^S\right]_2, \mathsf{tr}_{\mathsf{BLS}})$. The algorithm outputs $\pi := ([\boldsymbol{c}]_1, \left[V^S\right]_1, \left[V^S\right]_2, \left[\boldsymbol{q}_2^S\right]_2, \psi^S)$.

Theorem 14. *The scheme above is Perfect Zero-Knowledge.*

Proof. The key idea behind the proof is that all its the elements can be seen as perfectly hiding commitments to $\boldsymbol{a}$, where $\boldsymbol{a}$ is the opening of $[\boldsymbol{c}]_1$. For any V^S and any $\boldsymbol{a}$, there always exists a compatible δ. Further, since $\mathbf{Q}_2$ has full rank, $\left[\boldsymbol{q}_2^S\right]_2$ is compatible with any values $\boldsymbol{a}$, δ. $\left[H^S\right]_1$ is uniquely determined by V^S and the rest of the elements of the crs. Finally, perfect zero-knowledge follows from the perfect zero-knowledge property of the bilateral space membership proof. □

3.4 Unit Vector from Static Assumptions

Given n, we build a proof system for the family of languages

$$\mathcal{L}_{\text{uv,ck}} = \left\{ [\boldsymbol{c}]_1 \in \mathbb{G}_1^{2n} \;\middle|\; \begin{array}{l} \exists \boldsymbol{a}, \boldsymbol{w} \in \mathbb{Z}_p^n \text{ s.t. } [\boldsymbol{c}]_1 = \mathsf{Com}_{ck}(\boldsymbol{a}, \boldsymbol{w}), \\ \boldsymbol{a} \in \{0,1\}^n \text{ and } \sum_{j=1}^{n} a_j = 1 \end{array} \right\},$$

where Com_{ck} is a perfectly binding commitment scheme, with $\mathtt{ck}$ chosen from a witness samplable distribution $\mathcal{D}_\rho$. For simplicity, we assume that $[\boldsymbol{c}]_1$ is a vector of ElGamal encryptions as in the previous schemes.

Alternatively, to better match the description of the language $\mathcal{L}_{\text{quad},ck}$ given in Section 3.3.1, we can also define this language as:

$$\mathcal{L}_{\text{uv,ck}} = \left\{ [\boldsymbol{c}]_1 \in \mathbb{G}_1^{2n} \;\middle|\; \begin{array}{l} \exists \boldsymbol{a}, \boldsymbol{w} \in \mathbb{Z}_p^n \text{ s.t. } [\boldsymbol{c}]_1 = \mathsf{Com}_{ck}(\boldsymbol{a}, \boldsymbol{w}), \\ \boldsymbol{a}^\top \mathbf{V} + \boldsymbol{b}^\top \in \{0,2\}^n \end{array} \right\},$$

where

$$\mathbf{V} = \begin{pmatrix} 2 & & & 1 \\ & \ddots & & \vdots \\ & & 2 & 1 \end{pmatrix}, \qquad \boldsymbol{b} = \begin{pmatrix} 0 \\ \vdots \\ 0 \\ 1 \end{pmatrix}.$$

That is, $\mathbf{V} = (2\mathbf{I}_n | \mathbf{1})$ and $\boldsymbol{b} = (\mathbf{0}_n | 1)^\top$. In particular, this is a special case of the language $\mathcal{L}_{\text{quad},ck}$, with $\mathbf{V} = (v_{ij})$ and $\boldsymbol{b}$ fixed.

Our argument for this language is almost identical to the argument in Section 3.3.1, except that we use a dual point of view and now the points $\mathcal{R} = \{r_1, \ldots, r_{n+1}\}$ are published only in the exponent, while $s \in \mathbb{Z}_p$ can be public[2]. We remark that this change affects crucially the information that must be included in the crs to allow the prover to compute $[H]_1$, the quotient of dividing $p(s)$ by $t(s)$. In the general case (for any $\mathbf{V}, \boldsymbol{b}$), this information would be quadratic in n after this change. On the other hand, the advantage of this approach is that soundness is based on a static assumption. The intuition behind it is that if the points $r_1, \ldots, r_{n+1}$ are random and unrelated, one can reduce satisfiability of the jth equation to a computational problem which is only related to r_j and independent of the rest.

The fact that the crs is quadratic makes the scheme less interesting in the general case. For this reason, we restrict this dual approach to the unit vector argument. A

[2]Actually it is not necessary for completeness, but it can be published without compromising security.

similar situation is found in the paper of González *et al.* [67], in which they provided a constant proof that a set of perfectly binding commitments to integers open to bits. In the general case, the common reference string was quadratic, while in the unit vector case it was linear.

Intuition

Apart from the change of basis for computing $[H]_1$, another important but very technical difference with respect to the previous scheme is that we use a special form of interpolation. Given some points $\{r_1, \ldots, r_{n+1}\}$ we can define the polynomial $v_i(x) = \sum_{j=1}^{n+1} v_{ij}\tilde{\lambda}_j(X)$ where $\tilde{\lambda}_j(X) = \prod_{k\neq j} \frac{X - r_k}{r_j - r_k}$ is the (normalized) Lagrange interpolation polynomial for which $v_i(X)$ is the polynomial that at the point r_j goes through v_{ij}, that is the ijth matrix entry of $\mathbf{V}$ of Section 3.3.1. We want to prove security under static assumptions. So, we just want one point challenge instead of l as in the previous constructions, where the assumptions were not static. In our construction we need to compute the interpolation polynomials knowing all the interpolation points in $\mathbb{Z}_p$ but one, say r_{j^*}, that we know in the group $\mathbb{G}_\gamma$. The polynomials $\tilde{\lambda}_j(x)$ are rational functions in terms of r_{j^*} and they are infeasible to compute in this situation. Our approach allows us to compute the interpolation polynomials as degree 1 polynomials in terms of r_{j^*}. We achieve that using the non-normalized Lagrange interpolation with polynomials $\lambda_j(X) = \prod_{k\neq j}(X - r_k)$ for which $v_i(x)$ in point r_j goes through $\mu_j v_{ij}$, where $\mu_j = \prod_{k\neq j}(r_j - r_k)$.

As in Section 3.3.1 if we consider $\boldsymbol{Z} = \boldsymbol{a}^\top\mathbf{V} + \boldsymbol{b}^\top$, $\boldsymbol{Z}$ satisfies equations $\boldsymbol{Z} \in \{0, 2\}^{n+1}$ if and only if $(\boldsymbol{Z} - \mathbf{1})^2 = \mathbf{1}$. Given a set of points $\mathcal{R} = \{r_1, \ldots, r_{n+1}\}$, the non-normalized interpolation polynomials, $v_i(X)$ such that $v_i(r_j) = \mu_j v_{ij}$ for $i \in \{1, \ldots, n\}$ and $v_0(r_j) = \mu_j(b_j - 1)$ have a very specific form, namely

$$v_0(X) = -\sum_{i=1}^{n} \lambda_i(X), \qquad v_i(X) = 2\lambda_i(X) + \lambda_{n+1}(X), \qquad \text{for } i \in \{1, \ldots, n\}.$$

With the definition of $\lambda_i(X)$ that we are using, by a similar argument as in previous sections now the polynomial $p(X)$ is of the form:

$$p(X) = \left(-\sum_{i=1}^{n} \lambda_i(X) + \sum_{i=1}^{n} a_i \left(2\lambda_i(X) + \lambda_{n+1}(X)\right)\right)^2 - \left(\sum_{i=1}^{n+1} \lambda_i(X)\right)^2, \quad (3.6)$$

where $\sum_{i=1}^{n+1} \lambda_i(X)$ is the interpolation polynomial that has value μ_i in each point r_i. The equation (3.6) is 0 in $\{r_1, \ldots, r_{n+1}\}$ if and only if all the equations are satisfied.

If $[\boldsymbol{c}]_1$ is in the language and $\boldsymbol{a}$ is its opening, there exists an index i^* such that $a_{i^*} = 1$ and $a_j = 0$ if $j \neq i^*$. Thus, substituting these values in the equation (3.6),

$$p(X) = \left(- \sum_{i=1, i\neq i^*}^{n} \lambda_i(X) + \lambda_{i^*}(X) + \lambda_{n+1}(X) \right)^2 - \left(\sum_{i=1}^{n+1} \lambda_i(X) \right)^2 .$$

Consequently, in order to compute the polynomial $h(X) = \frac{p(X)}{t(X)}$, the prover would need products like $\lambda_{j,i} := \frac{\lambda_j(X)\lambda_i(X)}{t(X)} = \prod_{k\neq i,j}(X - r_k)$. The trivial solution is to provide $\{\lambda_{j,i}\}_{i,j=1}^{n+1}$ in the crs, but this implies a quadratic crs. Our approach allows to give just $n + 1$ combinations of λ_i as we can see in the following, which results in a linear crs.

Again, the key difference with the scheme of Section 3.3.1 is that here we want the prover to know s in $\mathbb{Z}_p$ but not the interpolation points, so the way we compute H changes. We decompose $p(X)$ in a product of polynomials as follows. Let $\overline{v}(X) = v_0(X) + \sum_{i=1}^{n} a_i v_i(X) = -\sum_{i=1, i\neq i^*}^{n} \lambda_i(X) + \lambda_{i^*}(X) + \lambda_{n+1}(X)$ and $k(X) = \sum_{i=1}^{n+1} \lambda_i(X)$. Then,

$$p(X) = (\overline{v}(X) + k(X))(\overline{v}(X) - k(X)). \tag{3.7}$$

Note that

$$\begin{aligned}
\overline{v}(X) + k(X) =& - \sum_{i=1, i\neq i^*}^{n} \lambda_i(X) + \lambda_{i^*}(X) + \lambda_{n+1}(X) + \sum_{i=1}^{n+1} \lambda_i(X) \\
=& 2\left(\lambda_{i^*}(X) + \lambda_{n+1}(X)\right), \\
\overline{v}(X) - k(X) =& - \sum_{i=1, i\neq i^*}^{n} \lambda_i(X) + \lambda_{i^*}(X) + \lambda_{n+1}(X) - \sum_{i=1}^{n+1} \lambda_i(X) \\
=& -2 \sum_{i=1, i\neq i^*}^{n} \lambda_i(X).
\end{aligned} \tag{3.8}$$

Now we can use this decomposition to compute $h(X)$:

$$\begin{aligned}
h(X) = \frac{(\overline{v}(X) + k(X))(\overline{v}(X) - k(X))}{t(X)} &= \frac{-4\left(\sum_{i=1, i\neq i^*}^{n} \lambda_i(X)\right)\left(\lambda_{i^*}(X) + \lambda_{n+1}(X)\right)}{t(X)} \\
&= -4 \sum_{i=1, i\neq i^*}^{n} \lambda_{i,i^*}(X) + \lambda_{i,n+1}(X).
\end{aligned} \tag{3.9}$$

This $h(X)$ can be computed evaluated in s for any i^* using equation (3.9) by an honest prover who is given

$$\left\{ \sum_{j=1, j\neq i}^{n} \lambda_{j,i}(s) + \lambda_{j,n+1}(s) \right\}_{i\in\{1,\ldots,n+1\}}$$

in the crs.

Note that in the scheme, $h(X)$ is randomized with an additional term $\delta t(X)$ in $v(X)$, where $\delta \leftarrow \mathbb{Z}_p$, in order to get zero-knowledge.

Scheme description

Given $n \in \mathbb{Z}_p$ we construct a QA-NIZK argument for the language $\mathcal{L}_{\mathrm{uv},\mathtt{ck}}$.

K_0**:** The algorithm $\mathsf{K}_0(gk, n)$ samples $ck = ([\boldsymbol{u}]_1)$ from the 1-Lin distribution $\mathcal{L}_1$. A commitment $\mathsf{Com}_{\mathtt{ck}}(\boldsymbol{a}; \boldsymbol{w})$ is the concatenation of $\mathsf{Enc}_{\mathtt{ck}}(a_i; w_i) = [a_i \boldsymbol{e}_2 + w_i \boldsymbol{u}]_1$ of ElGamal encryptions.

K_1**:** The algorithm $\mathsf{K}_1(gk, \mathtt{ck}, n)$ picks $s, \{r_j\}_{j\in\{1,\ldots,n+1\}} \leftarrow \mathbb{Z}_p$, computes the non-normalized Lagrange interpolation polynomials $\lambda_i(X) = \prod_{k\neq i}(X - r_k)$ using the points r_j as interpolation points, and evaluates $\lambda_i(s)$, for $i \in \{1, \ldots, n+1\}$. It also defines $t(X) := \prod_{i=1}^{n+1}(X - r_i)$ and

$$\left\{ L_i(s) := \sum_{j=1, j\neq i}^{n} \lambda_{j,i}(s) + \lambda_{j,n+1}(s) \right\}_{i\in\{1,\ldots,n+1\}}.$$

It picks $\left\{\boldsymbol{\phi}_i, \hat{\boldsymbol{\phi}}_i\right\}_{i\in\{1,\ldots,n+1\}} \leftarrow \mathbb{Z}_p^2 \times \mathbb{Z}_p^3$, $\mathbf{Q}_1 \leftarrow \mathcal{U}_{2,2}$, $\mathbf{Q}_2 \leftarrow \mathcal{U}_{3,3}$ and generates also the crs for proving membership in bilateral linear spaces $\mathsf{BLS.CRS}$, for the linear space generated by the matrices:

$$\left[\begin{array}{ccc|c|ccc} \boldsymbol{e}_2 & & & \boldsymbol{u} & & & \\ & \ddots & & & \ddots & \mathbf{0} & \\ & & \boldsymbol{e}_2 & & & \boldsymbol{u} & \\ \hline 2\lambda_1(s)+\lambda_{n+1}(s) & \ldots & 2\lambda_n(s)+\lambda_{n+1}(s) & \mathbf{0} & t(s) & \mathbf{0} & \mathbf{0} \\ \boldsymbol{\phi}_1 & \ldots & \boldsymbol{\phi}_n & & \boldsymbol{\phi}_{n+1} & \mathbf{Q}_1 & \mathbf{0} \end{array}\right]_1,$$

$$\left[\begin{array}{ccc|c|ccc} 2\lambda_1(s)+\lambda_{n+1}(s) & \ldots & 2\lambda_n(s)+\lambda_{n+1}(s) & \mathbf{0} & t(s) & \mathbf{0} & \mathbf{0} \\ \hat{\boldsymbol{\phi}}_1 & \ldots & \hat{\boldsymbol{\phi}}_n & & \hat{\boldsymbol{\phi}}_{n+1} & \mathbf{0} & \mathbf{Q}_2 \end{array}\right]_2,$$

$[\mathbf{M}]_1 \in \mathbb{G}_2^{4\times(2n+6)}$, $[\mathbf{N}]_2 \in \mathbb{G}_1^{(2n+3)\times(2n+6)}$, respectively.
The crs includes the elements

$$\left(gk, \mathtt{ck}, \left\{ [\lambda_i(s)]_{1,2}, [L_i(s)]_{1,2}, [\boldsymbol{\phi}_i]_1, \left[\hat{\boldsymbol{\phi}}_i\right]_2 \right\}_{i\in\{1,\ldots,n+1\}}, [t(s)]_{1,2}, [\mathbf{Q}_1]_1, [\mathbf{Q}_2]_2, \mathsf{BLS.CRS} \right).$$

P**:** The prover $\mathsf{P}(\mathsf{crs}, [\boldsymbol{c}]_1, \mathbf{V}, \boldsymbol{b}, \boldsymbol{a})$ picks $\delta \leftarrow \mathbb{Z}_p, \boldsymbol{r}_{q.1}, \boldsymbol{r}_{q.2} \leftarrow \mathbb{Z}_p^2 \times \mathbb{Z}_p^3$ and computes

$$[V]_{1,2} = [2\lambda_{i^*}(s) + \lambda_{n+1}(s) + \delta t(s)]_{1,2},$$

defines $p(s) = (v_0(s) + V + \sum_{i=1}^{n+1} \lambda_i(s))(v_0(s) + V - \sum_{i=1}^{n+1} \lambda_i(s))$ and $H = h(s) = \frac{p(s)}{t(s)}$. The prover can compute

$$[H]_1 = \left[-4L_{i^*}(s) + 2\delta(v_0(s) + V) - \delta^2 t(s)\right]_1 \text{ (see the intuition above)}$$

and the following elements:

$$\begin{array}{rcl} [\boldsymbol{q}_1]_1 & = & [\sum_{i=1}^{n} a_i\boldsymbol{\phi}_i + \delta\boldsymbol{\phi}_{n+1} + \mathbf{Q}_1\boldsymbol{r}_{q.1}]_1, \\ [\boldsymbol{q}_2]_2 & = & \left[\sum_{i=1}^{n} a_i\hat{\boldsymbol{\phi}}_i + \delta\hat{\boldsymbol{\phi}}_{n+1} + \mathbf{Q}_2\boldsymbol{r}_{q.2}\right]_2. \end{array}$$

The prover also computes a BLS proof ψ that

$$([\boldsymbol{c}]_1, [V]_1, [\boldsymbol{q}_1]_1, [V]_2, [\boldsymbol{q}_2]_2)^\top \in \mathbf{Im}\begin{pmatrix} [\mathbf{M}]_1 \\ [\mathbf{N}]_2 \end{pmatrix},$$

with witness $(\boldsymbol{a}, \boldsymbol{w}, \delta, \boldsymbol{r}_{q.1}, \boldsymbol{r}_{q.2})^\top \in \mathbb{Z}_p^{2n+6}$. Finally, it sends the proof π to the verifier, where

$$\pi := ([H]_1, [V]_1, [V]_2, [\boldsymbol{q}_1]_1, [\boldsymbol{q}_2]_2, \psi).$$

V**:** The verifier $\mathsf{V}(\mathsf{crs}, [\boldsymbol{c}]_1, \mathbf{V}, \boldsymbol{b}, \pi)$ checks whether the equation

$$e([v_0(s) + V]_1, [v_0(s) + V]_2) - e\left(\left[\sum_{i=1}^{n+1}\lambda_i(s)\right]_1, \left[\sum_{i=1}^{n+1}\lambda_i(s)\right]_2\right) = e([H]_1, [t(s)]_2) \tag{3.10}$$

holds, where $[v_0(s)]_1 = -\sum_{i=1}^{n}[\lambda_i(s)]_1$, and $\mathsf{BLS.verify}(\psi) = 1$. If both conditions hold, it returns 1, else it returns 0.

Completeness

The reason why the prover can compute H is explained in the intuition. On the other hand, membership in bilateral spaces is perfectly complete. Further, the right hand side of the verification equation is $e\left([H]_1, [t(s)]_2\right) = e\left(\left[\frac{p(s)}{t(s)}\right]_1, [t(s)]_2\right) = [p(s)]_T$, while the left hand is $\left[(v_0(s)+V)^2 - \left(\sum_{i=1}^{n+1}\lambda_i(s)\right)^2\right]_T = [p(s)]_T$.

Soundness

Theorem 15. *Let $\mathsf{Adv}_{\mathsf{PS}}(\mathcal{A})$ be the advantage of any PPT adversary $\mathcal{A}$ against the knowledge soundness of the scheme. There exist PPT adversaries $\mathcal{B}_1, \mathcal{B}_2, \mathcal{B}_3, \mathcal{B}_4$ such that*

$$\mathsf{Adv}_{\mathsf{KS}}(\mathcal{A}) \leq (n+1)\Big(\mathsf{Adv}_{\mathcal{L}_1\text{-}\mathsf{MDDH},\mathbb{G}_1}(\mathcal{B}_1) + 2\mathsf{Adv}_{\mathcal{L}_1\text{-}\mathsf{MDDH},\mathbb{G}_2}(\mathcal{B}_2) + \mathsf{Adv}_{\mathsf{BLS}}(\mathcal{B}_3) + \mathsf{Adv}_{1\text{-}\mathsf{STSDH}}(\mathcal{B}_4)\Big).$$

Proof. In order to prove soundness we will prove indistinguishability of the following games.

- Real: This is the real soundness game. The output is 1 if the adversary produces a false accepting proof, i.e. if there is some equation $\boldsymbol{a}^\top \boldsymbol{v}_i + b_i \notin \{0, 2\}$ and the verifier accepts the proof.

- Game_0: This game is identical to the previous one, except that the commitment key $\boldsymbol{u}$ is chosen by the game.

- Game_1: This game is identical to the previous one, except that some $j^* \leftarrow \{1, \ldots, n+1\}$ is chosen and the game aborts if $\boldsymbol{a}$ satisfies the j^*-th equation. This can be checked by opening $\boldsymbol{c}$ thanks to knowledge of $\boldsymbol{u}$ and checking whether $[\boldsymbol{a}]_1^\top \boldsymbol{v}_{j^*} + [b_{j^*}]_1 \in \{[0]_1, [2]_1\}$.

- Game_2: This game is identical to the previous one, except that $\mathbf{Q}_1$ is now a uniformly random matrix conditioned on having rank 1. If $j^* \neq n+1$, the elements $[\boldsymbol{\phi}_i]_1$ are changed to

$$[\boldsymbol{\phi}_i]_1 = \begin{cases} [\mathbf{Q}_1]_1 \boldsymbol{r}_i, & i = 1, \ldots, n+1, i \neq j^* \\ [2\lambda_{j^*}]_1 \boldsymbol{e}_1^2 + [\mathbf{Q}_1]_1 \boldsymbol{r}_{j^*}, & i = j^* \end{cases}$$

where $\boldsymbol{r}_i, \boldsymbol{r}_{j^*}, \leftarrow \mathbb{Z}_p^2$ and $\boldsymbol{e}_1^2$ is the first element in the canonical basis of $\mathbb{Z}_p^2$, while if $j^* = n+1$, each $[\boldsymbol{\phi}_i]_1$ is changed to

$$[\boldsymbol{\phi}_i]_1 = \begin{cases} [\lambda_{n+1}(s)]_1 \, \boldsymbol{e}_1^2 + [\mathbf{Q}_1]_1 \boldsymbol{r}_i, & i = 1, \ldots, n \\ [\mathbf{Q}_1]_1 \, \boldsymbol{r}_{n+1}, & i = n+1 \end{cases}$$

where $\boldsymbol{r}_i, \boldsymbol{r}_{n+1} \leftarrow \mathbb{Z}_p^2$.

- Game_3: This game is identical to the previous one, except that $\mathbf{Q}_2$ is now a uniformly random matrix conditioned on having rank 1. If $j^* \neq n+1$, the elements $\left[\hat{\boldsymbol{\phi}}_i\right]_2$ are changed to

$$\left[\hat{\boldsymbol{\phi}}_i\right]_2 = \begin{cases} [2\lambda_{i,j^*}(s) + \lambda_{n+1,j^*}(s)]_2 \, \boldsymbol{e}_1^3 + [\mathbf{Q}_2]_2 \widetilde{\boldsymbol{r}}_i, & i = 1, \ldots, n, i \neq j^* \\ [\varepsilon]_2 \, \boldsymbol{e}_3^3 + [\mathbf{Q}_2]_2 \widetilde{\boldsymbol{r}}_{j^*} & i = j^* \\ [\lambda_{j^*}(s)]_2 \, \boldsymbol{e}_1^3 + [\mathbf{Q}_2]_2 \widetilde{\boldsymbol{r}}_{n+1} & i = n+1 \end{cases}$$

where $\widetilde{\boldsymbol{r}}_i, \widetilde{\boldsymbol{r}}_{j^*}, \widetilde{\boldsymbol{r}}_{n+1} \leftarrow \mathbb{Z}_p^3$, $\varepsilon \leftarrow \mathbb{Z}_p$ and $\boldsymbol{e}_i^3$ is the ith element in the canonical basis of $\mathbb{Z}_p^3$, while if $j^* = n+1$, the elements $\left[\hat{\boldsymbol{\phi}}_i\right]_2$ are changed to

$$\left[\hat{\boldsymbol{\phi}}_i\right]_2 = \begin{cases} [2\lambda_{i,n+1}(s)]_2 \, \boldsymbol{e}_1^3 + [\varepsilon]_2 \, \boldsymbol{e}_3^3 + [\mathbf{Q}_2]_2 \widetilde{\boldsymbol{r}}_i, & i = 1, \ldots, n, i \neq j^* \\ [\lambda_{n+1}(s)]_2 \, \boldsymbol{e}_1^3 + [\mathbf{Q}_2]_2 \widetilde{\boldsymbol{r}}_{n+1}, & i = n+1 \end{cases}$$

where $\widetilde{\boldsymbol{r}}_i, \widetilde{\boldsymbol{r}}_{n+1} \leftarrow \mathbb{Z}_p^3$, $\varepsilon \leftarrow \mathbb{Z}_p$.

Obviously, the games Real and Game_0 are indistinguishable. The indistinguishability of the other games is based on the rank problem as in the soundness proof in Section 3.3.1.

Lemma 16. *There exists an adversary $\mathcal{B}_3$ against the strong soundness of the* BLS *argument and an adversary $\mathcal{B}_4$ against the* 1-STSDH *assumption such that*

$$\Pr[\mathsf{Game}_3(\mathcal{A}) = 1] \leq \mathsf{Adv}_{\mathsf{BLS}}(\mathcal{B}_3) + \mathsf{Adv}_{1\text{-}\mathsf{STSDH}}(\mathcal{B}_4).$$

Proof. As in the proof of Lemma 11, we distinguish two events, the event that the adversary succeeds in giving a false proof of membership in bilinear spaces (event $\overline{E}$), and the complementary event E, which is the interesting part of the proof. It can be proved easily following an analogous argument that such an adversary $\mathcal{B}_3$ could break soundness of BLS. We describe the reduction for adversary $\mathcal{B}_4$ that receives a challenge $(gk, [\varepsilon]_2, [\alpha]_{1,2})$ of the 1-STSDH Assumption and plugs $(gk, [\varepsilon]_2)$ in the crs.

Adversary $\mathcal{B}_4$ chooses $s \leftarrow \mathbb{Z}_p$, and all the points r_i, $i \neq j^*$ as random elements in $\mathbb{Z}_p$. Then it implicitly sets r_{j^*} to be $s - r_{j^*} = \alpha$, so that r_{j^*} can be computed in $\mathbb{G}_1$ and $\mathbb{G}_2$ but not in $\mathbb{Z}_p$. Then it sets:

$$\begin{array}{lcl} [\lambda_j(s)]_{1,2} & = & \prod_{i \neq j, j^*} (s - r_i) \, [\alpha]_{1,2} \text{ for } j \neq j^*, \\ [\lambda_{j^*}(s)]_{1,2} & = & \left[\prod_{i \neq j^*} (s - r_i)\right]_{1,2}, \\ [L_j(s)]_{1,2} & = & \sum_{i=1, i \neq j}^{n} \left([\lambda_{j,i}(s)]_{1,2} + [\lambda_{n+1,i}(s)]_{1,2}\right), \\ [t(s)]_{1,2} & = & \prod_{i \neq j^*} (s - r_i) \, [\alpha]_{1,2} \end{array}$$

for $j \in \{1, \ldots, n+1\}$. The rest of the elements of the crs are computed as sampled as specified by the game. All these elements are added to the crs and sent to the soundness adversary $\mathcal{A}$, who eventually outputs π for the corresponding $[\boldsymbol{c}]_1$.

Adversary $\mathcal{B}_4$ extracts $[\boldsymbol{a}]_1 \in \mathbb{G}_1$ from $[\boldsymbol{c}]_1$ and the knowledge of $\boldsymbol{u} \in \mathbb{Z}_p^2$ and aborts if the j^*th equation is satisfied.

By definition $e([v_0(s) + V]_1, [v_0(s) + V]_2) - e\left(\left[\sum_{i=1}^{n+1} \lambda_i(s)\right]_1, \left[\sum_{i=1}^{n+1} \lambda_i(s)\right]_2\right) = [p(s)]_T$. We note that it can compute $\left[\frac{p(s)}{\alpha}\right]_T$ from $[H]_1$:

$$\left[\frac{p(s)}{\alpha}\right]_T = e\left([H]_1, \left[\frac{t(s)}{\alpha}\right]_2\right) = e([H]_1, [\lambda_{j^*}(s)]_2) \tag{3.11}$$

by the verification equation (3.10) and $[\lambda_{j^*}(s)]_2$ is efficiently computable by the adversary.

Moreover, $p(s)$ can be factored as $p(s) = (\overline{v}(s) + k(s))(\overline{v}(s) - k(s))$ for equation (3.9). We can write $\overline{v}(s) = (V_0 + \alpha V_1)$ and $k(s) = (K_0 + \alpha K_1)$, where V_0, K_0 are the terms which the adversary does not know how to divide by α. More specifically,

$$p(s) = (V_0 + \alpha V_1)^2 - (K_0 + \alpha K_1)^2, \tag{3.12}$$

and therefore

$$\begin{aligned} \frac{p(s)}{\alpha} &= \left(\frac{V_0}{\alpha} + V_1\right)(V_0 + \alpha V_1) - \left(\frac{K_0}{\alpha} + K_1\right)(K_0 + \alpha K_1) \\ &= \frac{V_0^2 - K_0^2}{\alpha} + 2V_0V_1 + \alpha V_1^2 - 2K_0K_1 - \alpha K_1^2 \\ &= \frac{V_0^2 - K_0^2}{\alpha} + (2V_0 + \alpha V_1)\, V_1 - 2K_0K_1 - \alpha K_1^2 \\ &= \frac{V_0^2 - K_0^2}{\alpha} + (V_0 + V)\, V_1 - 2K_0K_1 - \alpha K_1^2 \end{aligned} \tag{3.13}$$

The BLS proof guarantees the existence of values $\boldsymbol{a}, \delta, \boldsymbol{r}_{q.1}, \boldsymbol{r}_{q.2}$, binding property of commitments c_i assure a_i are unique. So, the elements V_0, V_1, K_0, K_1 are uniquely determined. We remember here the polynomials $\overline{v}(X), k(X)$ that we have defined above in (3.8) but adding the randomization term in $\overline{v}(X)$:

$$\begin{aligned}\overline{v}(X) &= \sum_{i=1}^{n}(2a_i - 1)\lambda_i(X) + \sum_{i=1}^{n} a_i\lambda_{n+1}(X) + \delta t(X),\\ k(X) &= \sum_{i=1}^{n+1}\lambda_i(X),\end{aligned}$$

for which the equation (3.6) holds. Assuming $j^* \neq n+1$, if we divide by $X - r_{j^*}$, we obtain

$$\begin{aligned}\frac{\overline{v}(X)}{X - r_{j^*}} =& \frac{(2a_{j^*} - 1)\lambda_{j^*}(X)}{X - r_{j^*}} + \sum_{i=1, i\neq j^*}^{n}(2a_i - 1)\lambda_{i,j^*}(X)\\ &+ \sum_{i=1}^{n} a_i\lambda_{n+1,j^*}(X) + \delta\lambda_{j^*}(X),\\ \frac{k(X)}{X - r_{j^*}} =& \frac{\lambda_{i^*}(X)}{X - r_{j^*}} + \sum_{i=1, i\neq j^*}^{n+1}\lambda_{i,j^*}(X),\end{aligned} \tag{3.14}$$

where the first term that is not divisible by $X - r_{j^*}$ corresponds to V_0, K_0 in each equation, respectively when the polynomials are evaluated on s. The other terms of the equations correspond to V_1, K_1 respectively. So, if $j^* \neq n+1$:

$$\begin{aligned}V_0 &= (2a_{j^*} - 1)\lambda_{j^*}(s)\\ V_1 &= \sum_{i=1, i\neq j^*}^{n}(2a_i - 1)\lambda_{i,j^*}(s) + \left(\sum_{i=1}^{n} a_i\right)\lambda_{n+1,j^*}(s) + \delta\lambda_{j^*}(s)\\ K_0 &= \lambda_{j^*}(s)\\ K_1 &= \sum_{i=1, i\neq j^*}^{n+1}\lambda_{i,j^*}(s),\end{aligned}$$

otherwise, if $j^* = n+1$:

$$\begin{aligned}
V_0 &= \sum_{i=1}^{n} a_i \lambda_{n+1}(s) \\
V_1 &= \sum_{i=1}^{n} (2a_i - 1)\lambda_{i,n+1}(s) + \delta\lambda_{n+1}(s) \\
K_0 &= \lambda_{n+1}(s) \\
K_1 &= \sum_{i=1}^{n} \lambda_{i,n+1}(s).
\end{aligned}$$

In either case, $\mathcal{B}_4$ knows $[V]_{1,2}$ from the proof, $K_0, K_1 \in \mathbb{Z}_p$, we will now argue that V_0 can be computed in $\mathbb{G}_1$ from one of the extracted values of $[\boldsymbol{q}_1]_1$ and V_1 can be computed in $\mathbb{G}_2$ from the extracted values of $[\boldsymbol{q}_2]_2$. More specifically, remember that in this game if $j^* \neq n+1$,

$$\begin{aligned}
[\boldsymbol{q}_1]_1 &= \left[\sum_{i=1, i\neq j^*}^{n} a_i \mathbf{Q}_1 \boldsymbol{r}_i + a_{j^*} \left(2\lambda_{j^*}(s)\boldsymbol{e}_1^2 + \mathbf{Q}_1 \boldsymbol{r}_{j^*}\right) + \delta \mathbf{Q}_1 \boldsymbol{r}_{n+1} + \mathbf{Q}_1 \boldsymbol{r}_{q.1} \right]_1 \\
[\boldsymbol{q}_2]_2 &= \left[\sum_{i=1, i\neq j^*}^{n} a_i \left((2\lambda_{i,j^*}(s) + \lambda_{n+1,j^*}(s))\boldsymbol{e}_1^3 + \mathbf{Q}_2 \widetilde{\boldsymbol{r}}_i\right) + a_{j^*} \left(\varepsilon \boldsymbol{e}_3^3 + \mathbf{Q}_2 \widetilde{\boldsymbol{r}}_{j^*}\right) \right]_2 \\
&\quad + \left[\delta \left(\lambda_{j^*}(s)\boldsymbol{e}_1^3 + \mathbf{Q}_2 \widetilde{\boldsymbol{r}}_{n+1}\right) + \mathbf{Q}_2 \boldsymbol{r}_{q.2}\right]_2,
\end{aligned}$$

and if $j^* = n+1$,

$$\begin{aligned}
[\boldsymbol{q}_1]_1 &= \left[\sum_{i=1}^{n} a_i \left(\lambda_{n+1}(s)\boldsymbol{e}_1^2 + \mathbf{Q}_1 \boldsymbol{r}_i\right) + \delta \mathbf{Q}_1 \boldsymbol{r}_{n+1} + \mathbf{Q}_1 \boldsymbol{r}_{q.1} \right]_1 \\
[\boldsymbol{q}_2]_2 &= \left[\sum_{i=1}^{n} a_i \left(2\lambda_{i,n+1}(s)\boldsymbol{e}_1^3 + \varepsilon \boldsymbol{e}_2^3 + \mathbf{Q}_2 \widetilde{\boldsymbol{r}}_i\right) + \delta \left(\lambda_{n+1}(s)\boldsymbol{e}_1^3 + \mathbf{Q}_2 \widetilde{\boldsymbol{r}}_{n+1}\right) + \mathbf{Q}_2 \boldsymbol{r}_{q.2} \right]_2.
\end{aligned}$$

Since $\mathcal{B}_4$ sampled $\mathbf{Q}_1, \mathbf{Q}_2$ itself, it can extract the following values from $[\boldsymbol{q}_1]_1$ and $[\boldsymbol{q}_2]_2$ defining appropriate orthogonal vectors to theses matrices, similarly to the extraction explained in Lemma 11:

- if $j^* \neq n+1$, it extracts $\left[\sum_{i=1, i\neq j^*}^{n} a_i(2\lambda_{i,j^*}(s) + \lambda_{n+1,j^*}(s)) + \delta\lambda_{j^*}(s)\right]_2$, $[a_{j^*} 2\lambda_{j^*}(s)]_1$ and $[\varepsilon a_{j^*}]_2$.

- if $j^* = n+1$, it extracts $[\sum_{i=1}^n a_i \lambda_{n+1}(s)]_1$, $[\sum_{i=1}^n 2a_i\lambda_{i,n+1} + \delta\lambda_{n+1}(s)]_2$ and $[\varepsilon \sum_{i=1}^n a_i]_2$.

From these values it can compute $[V_0]_1$, $[V_1]_2$ in both cases, and also defining $\beta := 2a_{j^*} - 1$ for $j^* \neq n+1$ and $\beta := \sum_{i=1}^n a_i$ for $j^* = n+1$, it can compute $[\varepsilon\beta]_2$ in both cases (if $j^* \neq n+1$, computes $2\,[\varepsilon a_{j^*}]_2 - [\varepsilon]_2$, otherwise it has extracted $[\varepsilon \sum_{i=1}^n a_i]_2$).

Combining $[V_0]_1$, $[V_1]_2$, $[\alpha]_{1,2}$ with K_0, K_1 it can substract from equation (3.11) the terms $[(V+V_0)V_1 + 2K_0K_1 + \alpha K_1^2]_T$ in equation (3.13), so the adversary can compute in $\mathbb{G}_T$:

$$\left[\frac{V_0^2 - K_0^2}{\alpha}\right]_T = \begin{cases} \left[\dfrac{(2a_{j^*}-1)^2 - 1}{\alpha}\lambda_{j^*}^2(s)\right]_T, & j^* \neq n+1 \\ \left[\dfrac{(\sum_{i=1}^n a_i)^2 - 1}{\alpha}\lambda_{n+1}^2(s)\right]_T, & j^* = n+1. \end{cases}$$

Since the adversary knows $\lambda_{j^*}^2(s) \in \mathbb{Z}_p$ in both cases, it can compute:

$$\left[\frac{V_0^2 - K_0^2}{\alpha\lambda_{j^*}^2(s)}\right]_T = \begin{cases} \left[\dfrac{(2a_{j^*}-1)^2 - 1}{\alpha}\right]_T, & j^* \neq n+1 \\ \left[\dfrac{(\sum_{i=1}^n a_i)^2 - 1}{\alpha}\right]_T, & j^* = n+1 \end{cases}$$

which is $\left[\dfrac{\beta^2-1}{\alpha}\right]_T$ in both cases.

Finally, the adversary returns $\left(r_{j^*}, [\beta]_1, [\varepsilon\beta]_2, \left[\dfrac{\beta^2-1}{\alpha}\right]_T\right)$, which breaks the 1-$\mathsf{STSDH}$ Assumption. □

Since the matrix $\mathbf{V}$ has rank n, by the Theorem 13, we know there exists an extractor of the witness in the field. Then, we have proven knowledge soundness. □

Theorem 17. *The scheme above is Perfect Zero-Knowledge, i.e. there exists a simulator algorithm* S *who has access to the trapdoor* $\mathsf{tr} = \{s, r_1, \ldots, r_{n+1}\}$*, that constructs a simulated proof* π^S *such that it is statistically indistinguishable from the real proof* π*.*

The proof is analogous to the one of Theorem 14 of Section 3.3.1.

3.4.1 Detailed Efficiency Comparison

In Table 3.4 we give more a detailed comparison of our arguments for bit-strings and unit vector in Sections 3.3.1 and 3.4 with the analogous results in [67].

	Language	Proof size	crs size	Assumption
Sect. 3.3.1	Bitstring	$4\|\mathbb{G}_1\| + 6\|\mathbb{G}_2\|$	$(n + O(1))\|\mathbb{G}_1\|+$ $+(4n + O(1))\|\mathbb{G}_2\|$	q-STSDH [7]
Sect. 5 of [67]		$10\|\mathbb{G}_1\| + 10\|\mathbb{G}_2\|$	$O(n^2)\|\mathbb{G}_1\|+$ $+O(n^2)\|\mathbb{G}_2\|$	SSDP
Sect. 3.4	Unit vector	$6\|\mathbb{G}_1\| + 6\|\mathbb{G}_2\|$	$(4(n + 1) + O(1))\|\mathbb{G}_1\|+$ $+(5(n + 1) + O(1))\|\mathbb{G}_2\|$	1-STSDH [7]
Sect. 5 of [67]		$10\|\mathbb{G}_1\| + 10\|\mathbb{G}_2\|$	$(20n + O(1))\|\mathbb{G}_1\|+$ $+(18n + O(1))\|\mathbb{G}_2\|$	SSDP

Table 3.4: The table shows the proof sizes (not including commitments) for bitstrings and unit vectors of size n.

3.5 Aggregated Set Membership Arguments

In the construction of Section 3.3.1, if $\mathbf{V}$ is the identity matrix and $\boldsymbol{b} = \mathbf{0}$, the equations $\boldsymbol{a}\mathbf{V} + \boldsymbol{b} \in \{0, 2\}^l$ just prove that each $a_i \in \{0, 2\}$. In this section we consider a generalization and build a proof system which proves that some perfectly binding commitments open to $a_i \in \mathcal{Z} = \{z_1, \ldots, z_m\} \subset \mathbb{Z}_p$. The proof is constant-size and uses the Boneh-Boyen signature scheme (the basic scheme from [22, Sect. 4.3]) together with a technique to aggregate quadratic equations similar to the one of Section 3.3.1 and inspired by the quadratic span programs of Gennaro et al. [61].

First, in Section 3.5.1, we describe how to construct an argument of membership for a single $a \in \mathcal{Z}$ and then in Section 3.5.2 we show how to aggregate the argument. In Section 3.6.2 we show how to apply these ideas to construct a range proof.

3.5.1 Non-Aggregated Set Membership Argument

Intuition

We build a constant-size proof of membership for polynomially-large sets in $\mathbb{Z}_p$ with linear crs. The idea is to give in the common reference string Boneh-Boyen signatures

to each element of the set. The proof of membership is just a proof of knowledge of a valid signature. Recall that $[\sigma]_2$ is a valid signature for x if and only if

$$e([\mathsf{sk} - x]_1, [\sigma]_2) - [1]_T = [0]_T.$$

The statement $x \in \mathcal{Z}$ is proven committing to x and to $[\sigma]_2 = \left[\frac{1}{\mathsf{sk}-x}\right]_2$, and giving a Groth-Sahai proof for the satisfiability of the verification equation.

The problem with this approach is that it is not possible to extract $x \in \mathbb{Z}_p$ from its Groth-Sahai commitment, but only $[x]_1 \in \mathbb{G}_1$. Therefore, it is not clear how to reduce soundness to the EUF-CMA security of Boneh-Boyen, as the reduction can only output a "relaxed form" of forgery $([x]_1, [\sigma]_2)$, for some $x \notin \mathcal{Z}$, instead of $(x, [\sigma]_2)$.[3].

It turns out that Boneh-Boyen signatures are not unforgeable when purported forgeries are pairs of the form $([x]_1, [\sigma]_2)$. The problem is that $[x]_1$ may be dependent of sk, whereas this is impossible when $x \in \mathbb{Z}_p$ must be given. Indeed, for any message of the form $[\mathsf{sk} - x]_1$ one might compute a forgery as $[1/x]_2$.

To solve this issue, we force the prover to commit to $[\varepsilon x]_1$, where the discrete logarithm of $[\varepsilon]_1$ remains hidden. Since $[\mathsf{sk} \cdot \varepsilon]_1$ is not given, the adversary cannot choose x to be a function of sk.

Scheme description

We give a proof of membership in $\mathcal{Z} = \{z_1, \ldots, z_m\} \subset \mathbb{Z}_p$. More precisely, we build a proof for the family of languages:

$$\mathcal{L}_{\mathsf{memb},\mathcal{Z},\mathsf{ck}} := \left\{ [\boldsymbol{c}]_1 \in \mathbb{G}_1^2 \;\middle|\; \exists \boldsymbol{w} \in \mathbb{Z}_p \text{ s.t. } [\boldsymbol{c}]_1 = \mathsf{Com}_{ck}(x; w) \text{ and } x \in \mathcal{Z} \right\}.$$

Setup**:** The setup algorithm generates the parameters for the Boneh-Boyen signatures, chooses $\varepsilon \leftarrow \mathbb{Z}_p$ and computes the crs that contains $[\varepsilon]_2$, signatures $[\sigma_j]_2 = \left[\frac{1}{\mathsf{sk}-z_j}\right]_2$ of each $z_j \in \mathcal{Z}$, and the Groth-Sahai crs. The simulation trapdoor is ε and the GS simulation trapdoor for equations which are right-simulatable [4].

P**:** The prover P does the following. If $x \in \mathcal{Z}$, then there is some pair $([y]_2, [\sigma]_2)$, where $[\sigma]_2$ is in the crs, such that

$$e([\mathsf{sk}]_1 - [x]_1, [\sigma]_2) = [1]_T \quad \text{and} \quad [y]_2 = x[\varepsilon]_2.$$

[3] An alternative is of course to commit to x bit-by-bit to make it extractable, but it is completely impractical.

[4] See Ràfols [112]. These are statements for which only the commitments in $\mathbb{G}_2$ need to be perfectly hiding and where it is sufficient to get the simulation trapdoor to equivocate commitments in $\mathbb{G}_2$.

The prover produces a Groth-Sahai proof of the equations:

$$e\left([\mathsf{sk}]_1 - [X]_1, [\Sigma]_2\right) = [1]_T \quad \text{and} \quad [Y]_2 = X[\varepsilon]_2$$

where X, Y, Σ are the variables.

V: The verifier V accepts if and only if both proofs are valid.

Theorem 18. *The argument above is computationally quasi-adaptively sound under the $\mathcal{Z}$-GSDH Assumption in $\mathbb{G}_2$ and the soundness of Groth-Sahai proofs.*

Proof. We construct an adversary $\mathcal{B}$ against the $\mathcal{Z}$-GSDH assumption, which receives $gk := (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, \mathcal{P}_1, \mathcal{P}_2)$ together with $[\varepsilon]_{1,2}$ and $\{[s^i]_{1,2}\}_{i=1}^m$ from the challenger. The adversary defines a new generator for $\mathbb{G}_2$, $\overline{\mathcal{P}}_2 = [\prod_{i=1}^m (s - z_i)]_2$, defines a new group key $\overline{gk} := (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, \mathcal{P}_1, \overline{\mathcal{P}}_2)$, and defines $[\mathsf{sk}]_1 = [s]_1$. Note that we use implicit notation with respect to $\mathcal{P}_1, \mathcal{P}_2$ and not with respect to the new generators.

The adversary can now build the signatures

$$\left(z_j[\varepsilon]_2, \left[\prod_{\substack{i=1 \\ i \neq j}}^{m} (s - z_i)\right]_2\right) = \left(z_j[\varepsilon]_2, \frac{1}{\mathsf{sk} - z_j}\overline{\mathcal{P}}_2\right)$$

which are valid with respect to the group key $\overline{gk}$.

Let $\mathcal{A}$ be an adversary against our set membership proof. Adversary $\mathcal{B}$ runs $\mathcal{A}$ with the new group key $\overline{gk}$, Groth-Sahai commitment keys for which it knows the discrete logarithm (in order to open commitments), and signatures $([\sigma_1]_2, \ldots, [\sigma_m]_2)$. Suppose that $\mathcal{A}$ wins by producing an accepting proof for some $x \notin \mathcal{Z}$. From the adversary's proof and committed values one can extract $[x]_1$ and $([y^*]_2, [\sigma^*]_2)$ and, from perfect soundness of Groth-Sahai proofs, it follows that

$$e([\mathsf{sk}]_1 - [x]_1, [\sigma^*]_2) = e(\mathcal{P}_1, \overline{\mathcal{P}}_2) \quad \text{and} \quad [y^*]_2 = x[\varepsilon]_2.$$

This implies that $[\sigma^*]_2 = \left[\frac{\prod_{j=1}^m (\mathsf{sk} - z_j)}{\mathsf{sk} - x}\right]_2$, and hence $([x]_1, [y^*]_2, [\sigma^*]_2)$ is a solution to the $\mathcal{Z}$-GSDH problem. □

Theorem 19. *The argument above is composable zero-knowledge under the composable zero-knowledge property of Groth-Sahai proofs.*

Proof. The proof simulator uses the Groth-Sahai trapdoor and ε to simulate the Groth-Sahai proof of both equations (note that even though the commitment $[\boldsymbol{c}]_1$ is part of the statement, both equations are right-simulatable when ε is known). □

3.5.2 Aggregated Set Membership Argument

Let $\mathcal{Z} \subset \mathbb{Z}_p$, $m = |\mathcal{Z}|$, and $n \in \mathbb{N}$. We construct a QA-NIZK argument for the following language

$$\mathcal{L}_{\mathsf{memb},\mathcal{Z},\mathtt{ck}} := \left\{ [\boldsymbol{c}]_1 \in \mathbb{G}_1^{2n} \;\middle|\; \begin{array}{l} \exists \boldsymbol{w} \in \mathbb{Z}_p^n \text{ s.t. } [\boldsymbol{c}]_1 = \mathsf{Com}_{ck}(\boldsymbol{x}; \boldsymbol{w}) \\ \text{and } x_1, \ldots, x_n \in \mathcal{Z} \end{array} \right\},$$

where $[\boldsymbol{c}]_1 = \mathsf{Com}_{ck}(\boldsymbol{x}; \boldsymbol{w})$ is a vector of ElGamal encryptions. The generalization to other perfectly binding commitments is straightforward.

Intuition. To express the validity of n signatures and message pairs, we construct polynomials $v(X), y(X)$, which encode the set of n verification equations for the Boneh-Boyen signatures. Given the set $\mathcal{R} = \{r_1, \ldots, r_n\} \subset \mathbb{Z}_p$, recall that we denote as $\lambda_i(X)$ the ith Lagrange interpolation polynomial associated to $\mathcal{R}$.

We define $v_0(X)$ as the constant polynomial $v_0(X) = \mathsf{sk}$, and $t(X) = \prod_{r_j \in \mathcal{R}}(X - r_j)$. The set of polynomials $v_0(X), \{\lambda_i(X)\}_{i=0}^n, t(X)$ accepts $x_1, \ldots, x_n$ if and only if $t(X)$ divides $(v_0(X) - v(X))y(X) - 1$, where

$$v(X) = \sum_{j=1}^{n} x_i \lambda_i(X), \qquad y(X) = \sum_{i=1}^{m} \sigma_{k(i)} \lambda_i(X),$$

and $\sigma_{k(i)}$ is the signature of some $z_{k(i)}$ such that $x_i = z_{k(i)}$.

That is, at any point $r_j \in \mathcal{R}$, if $x_j = v(r_j)$, then $y(r_j)$ is a a valid signature of x_j. This follows from

$$\begin{array}{r} (v_0(X) - v(X))y(X) - 1 = h(X)t(X) \text{ for some polynomial } h(X) \\ \implies (v_0(r_j) - v(r_j))y(r_j) - 1 = 0 \iff (\mathsf{sk} - x_j)y(r_j) - 1 = 0. \end{array}$$

In particular, if $j \in [n]$ is such that $x_j \notin \mathcal{Z}$, then $y(r_j)$ is a forgery for x_j. For simplicity, in this exposition we ignore the issue mentioned in previous section about commitment extractability, but this is taken into account in the argument.

Note that to compute $y(X)$ given $\lambda_i(X)$ in some source group, the prover would need to know the discrete logarithm of the signatures. To render the interpolation polynomials efficiently computable, we include in the crs the terms $[\sigma_i s^j]_2$, where $\sigma_i = \frac{1}{\mathsf{sk} - z_i}$, for all $i \in \{1, \ldots, m\}, j \in \{1, \ldots, n\}$, and all other values which require the signature's discrete logarithm. Consequently, our crs is of size $O(nm)$.

A direct instantiation of techniques from Section 3.3.1 requires perfectly binding commitments to each of the signatures and hence, a proof of size linear in the number

of statements. But it turns out that perfectly binding commitments to signatures are not necessary for proving membership in $\mathcal{Z}$. To achieve this, we use a trick similar to Section 3.3.1. We program the crs in order to extract a valid signature for x_{j^*}, for a random $j^* \in \{1, \ldots, n\}$, in such a way that the adversary might only detect the change in the crs with negligible probability.

Scheme description

Given $m, n \in \mathbb{N}$ and a set $\mathcal{Z} \subset \mathbb{Z}_p$, $|\mathcal{Z}| = m$, we construct a QA-NIZK argument for the language $\mathcal{L}_{\mathsf{memb},\mathcal{Z},\mathsf{ck}}$.

K_0**:** Algorithm $\mathsf{K}_0(gk)$ sets $\mathsf{ck} = [\boldsymbol{u}]_1 \leftarrow \mathcal{L}_1$.

K_1**:** Algorithm $\mathsf{K}_1(gk, \mathsf{ck})$ picks $s \leftarrow \mathbb{Z}_p$, $\left\{\boldsymbol{\phi}_i, \hat{\boldsymbol{\phi}}_i\right\}_{i\in\{1,\ldots,n+1\}} \leftarrow \mathbb{Z}_p^3 \times \mathbb{Z}_p^4$, $\mathbf{Q}_1 \leftarrow \mathcal{U}_{3,3}, \mathbf{Q}_2 \leftarrow \mathcal{U}_{4,4}$, picks a Boneh-Boyen secret key $\mathsf{sk} \leftarrow \mathbb{Z}_p$, generates signatures $[\sigma_1]_2, \ldots, [\sigma_m]_2$ for each element in $\mathcal{Z}$ and generates also crs_{Π_1} and crs_{Π_2} for proving membership in the linear spaces generated, respectively, by the matrices $\mathbf{M}, \mathbf{N}$, where:

$$[\mathbf{M}]_1 = \left[\begin{array}{ccc|ccc|cc} \boldsymbol{e}_2 & & & \boldsymbol{u} & & & & \\ & \ddots & & & \ddots & & \multicolumn{2}{c}{\mathbf{0}} \\ & & \boldsymbol{e}_2 & & & \boldsymbol{u} & & \\ \hline \lambda_1(s) & \ldots & \lambda_n(s) & \multicolumn{3}{c|}{\multirow{2}{*}{$\mathbf{0}$}} & t(s) & \mathbf{0} \\ \boldsymbol{\phi}_1 & \ldots & \boldsymbol{\phi}_n & & & & \boldsymbol{\phi}_{n+1} & \mathbf{Q}_1 \end{array}\right]_1,$$

$$[\mathbf{N}]_2 = \left[\begin{array}{ccccc|cc} \sigma_1\lambda_1(s) & \sigma_1\lambda_2(s) & \ldots & \sigma_m\lambda_n(s) & & t(s) & \mathbf{0} \\ \sigma_1\hat{\boldsymbol{\phi}}_1 & \sigma_1\hat{\boldsymbol{\phi}}_2 & \ldots & \sigma_m\hat{\boldsymbol{\phi}}_n & & \hat{\boldsymbol{\phi}}_{n+1} & \mathbf{Q}_2 \end{array}\right]_2,$$

$[\mathbf{M}]_1 \in \mathbb{G}_1^{(2n+4)\times(2n+4)}$, $[\mathbf{N}]_2 \in \mathbb{G}_2^{5\times(nm+5)}$.

The crs includes the elements

$$\left(gk, \mathsf{ck}, \left\{[s^j]_1, [\mathsf{sk}s^j]_1, [\sigma_i s^j]_{1,2}, [\boldsymbol{\phi}_i]_1, [\sigma_i\hat{\boldsymbol{\phi}}_j]_2\right\}_{i\in\{1,\ldots,m\},j\in\{1,\ldots,n\}}, [\boldsymbol{\phi}_{n+1}]_1, [\hat{\boldsymbol{\phi}}_{n+1}]_2, [\mathbf{Q}_1]_1, [\mathbf{Q}_2]_2, \mathsf{crs}_{\Pi_1}, \mathsf{crs}_{\Pi_2}\right).$$

P**:** The prover $\mathsf{P}(\mathsf{crs}, [\boldsymbol{c}]_1, \boldsymbol{x}, \boldsymbol{w})$ picks $\delta_v, \delta_y \leftarrow \mathbb{Z}_p, \boldsymbol{r}_{q.1} \leftarrow \mathbb{Z}_p^3, \boldsymbol{r}_{q.2} \leftarrow \mathbb{Z}_p^4$ and de-

fines the polynomials

$$v(X) = \sum_{i=1}^{n} x_i \lambda_i(X) + \delta_v t(X), \qquad y(X) = \sum_{i=1}^{n} \sigma_{k(i)} \lambda_i(X) + \delta_y t(X)$$
$$h(X) = \frac{(v_0(X) - v(X))y(X) - 1}{t(X)}$$

where $v_0(r_j) = \mathsf{sk}$, for all $j \in \{1, \ldots, n\}$, $t(X) = \prod_{r \in \mathcal{R}}(X - r)$ and $\lambda_i(X)$ is the ith Lagrangian interpolation polynomial associated to $\mathcal{R}$. By definition of the language, each x_i is equal to $z_{k(i)}$, for some $k(i) \in \{1, \ldots, m\}$.
The prover computes the following elements:

$$\begin{array}{lcllcl} [H]_1 &=& [h(s)]_1\,, & [\boldsymbol{q}_1]_1 &=& \left[\sum_{i=1}^{n} x_i \boldsymbol{\phi}_i + \delta_v \boldsymbol{\phi}_{n+1} + \mathbf{Q}_1 \boldsymbol{r}_{q.1}\right]_1, \\ [V]_1 &=& [v(s)]_1\,, & & & \\ [Y]_2 &=& [y(s)]_2\,, & [\boldsymbol{q}_2]_2 &=& \left[\sum_{i=1}^{n} \sigma_{k(i)} \hat{\boldsymbol{\phi}}_i + \delta_y \hat{\boldsymbol{\phi}}_{n+1} + \mathbf{Q}_2 \boldsymbol{r}_{q.2}\right]_2. \end{array}$$

The prover also computes two LS proofs

$$\psi_1 \leftarrow \Pi_1.\mathsf{LS.prove}\left(\mathsf{crs}_{\Pi_1}, \begin{bmatrix} \boldsymbol{c} \\ V \\ \boldsymbol{q}_1 \end{bmatrix}_1, \begin{pmatrix} \boldsymbol{x} \\ \boldsymbol{w} \\ \delta_v \\ \boldsymbol{r}_{q.1} \end{pmatrix}\right),$$
$$\psi_2 \leftarrow \Pi_2.\mathsf{LS.prove}\left(\mathsf{crs}_{\Pi_2}, \begin{bmatrix} Y \\ \boldsymbol{q}_2 \end{bmatrix}_2, \begin{pmatrix} \boldsymbol{y} \\ \delta_y \\ \boldsymbol{r}_{q.2} \end{pmatrix}\right),$$

where $\boldsymbol{y} = (y_{1,1}, y_{1,2}, \ldots, y_{n,m})$ and $y_{i,j}$ is equal to 1 if $i = k(j)$ and 0 otherwise. Finally, it sends the proof π to the verifier, where

$$\pi := ([H]_1, [V]_1, [Y]_2, [\boldsymbol{q}_1]_1, [\boldsymbol{q}_2]_2, \psi_1, \psi_2)\,.$$

V**:** The verifier $\mathsf{V}(\mathsf{crs}, \pi)$ checks whether the equation

$e([H]_1, [t(s)]_2) = e([v_0(s)]_1 - [V]_1, [Y]_2) - [1]_T$ holds, and

$$\Pi_1.\mathsf{LS.verify}\left(\mathsf{crs}_{\Pi_1}, \begin{bmatrix} \boldsymbol{c} \\ V \\ \boldsymbol{q}_1 \end{bmatrix}_1, \psi_1\right) = 1, \quad \Pi_2.\mathsf{LS.verify}\left(\mathsf{crs}_{\Pi_2}, \begin{bmatrix} Y \\ \boldsymbol{q}_2 \end{bmatrix}_2, \psi_2\right) = 1.$$

If all of these conditions hold, it returns 1, else 0.

Completeness

If $x_1, \ldots, x_n \in \mathcal{Z}$ then $(v_0(r_j) - v(r_j))y(r_j) - 1 = (x_{k(j)} + \mathsf{sk})\sigma_{k(j)} - 1 = 0$ for all j, and thus $(v_0(X) - v(X))y(X) = 1 \mod t(X)$. This implies that $h(X)$ is a well defined polynomial in $\mathbb{Z}_p[X]$ such that $e\left([h(s)]_1, [t(s)]_2\right) = e\left([v_0(s) - v(s)]_1, [y(s)]_2\right) - [1]_T$. It is easy to check that

$$\begin{pmatrix} \boldsymbol{c} \\ V \\ \boldsymbol{q}_1 \end{pmatrix} = \mathbf{M} \begin{pmatrix} \boldsymbol{x} \\ \boldsymbol{w} \\ \delta_v \\ \boldsymbol{r}_{q.1} \end{pmatrix} \text{ and } \begin{pmatrix} Y \\ \boldsymbol{q}_2 \end{pmatrix} = \mathbf{N} \begin{pmatrix} \boldsymbol{y} \\ \delta_y \\ \boldsymbol{r}_{q.2} \end{pmatrix},$$

where $\boldsymbol{y} = (y_{1,1}, \ldots, y_{m,n})$, and therefore ψ_1, ψ_2 are valid proofs.

Soundness

Theorem 20. *Let $\mathsf{Adv}_{\mathsf{PS}}(\mathcal{A})$ be the advantage of a PPT adversary $\mathcal{A}$ against the soundness of the scheme. There exist PPT adversaries $\mathcal{B}_1, \mathcal{B}_2, \mathcal{B}_{3,1}, \mathcal{B}_{3,2}, \mathcal{B}_4, \mathcal{B}_5$ such that*

$$\begin{aligned}\mathsf{Adv}_{\mathsf{PS}}(\mathcal{A}) \leq n \big(& 2\mathsf{Adv}_{\mathcal{L}_1\text{-}\mathsf{MDDH},\mathbb{G}_1}(\mathcal{B}_1) + 3\mathsf{Adv}_{\mathcal{L}_1\text{-}\mathsf{MDDH},\mathbb{G}_2}(\mathcal{B}_2) + \mathsf{Adv}_{\mathsf{LS},\Pi_1}(\mathcal{B}_{3,1}) \\ & + \mathsf{Adv}_{\mathsf{LS},\Pi_2}(\mathcal{B}_{3,2}) + \mathsf{Adv}_{\mathcal{Z}\text{-}\mathsf{GSDH},\mathbb{G}_1}(\mathcal{B}_4) + \mathsf{Adv}_{n\text{-}\mathsf{QTSDH}}(\mathcal{B}_5)\big).\end{aligned}$$

Proof. In order to prove soundness we will prove indistinguishability of the following games.

- Real: This is the real soundness game. The output is 1 if the adversary produces a false accepting proof, i.e. if there is some $x_i \notin \mathcal{Z}$ and the verifier accepts the proof.
- Game_0: This game is identical to the previous one, except that the commitment key $\boldsymbol{u}$ is chosen by the game in order to extract $[\boldsymbol{x}]_1$ from $[\boldsymbol{c}]_1$.
- Game_1: This game is identical to the previous one, except that some $j^* \leftarrow \{1, \ldots, n\}$ is chosen and the game aborts if the extracted value $[\boldsymbol{x}]_1$ is such that $[x_{j^*}]_1 \in [\mathcal{Z}]_1$.
- Game_2: For $i = 1, \ldots, n$, let $\alpha_i(X)$ and β_i be the quotient and the reminder, respectively, of dividing $\lambda_i(X)$ by $X - r_{j^*}$. Let $\alpha_{n+1}(X)$ and β_{n+1} be the quotient and the reminder of dividing $t(X)$ by $X - r_{j^*}$. This game is identical to the previous one, except that $\mathbf{Q}_1$ is now a uniformly random matrix conditioned on having rank 1, and for $i = 1, \ldots, n+1$, $[\boldsymbol{\phi}_i]_1$ is changed to

$$[\boldsymbol{\phi}_i]_1 = [\alpha_i(s)]_1 \boldsymbol{e}_2^3 + \beta_i [\varepsilon]_1 \boldsymbol{e}_3^3 + [\mathbf{Q}_1]_1 \boldsymbol{r}_i,$$

where $\boldsymbol{e}_j^3$ is the jth vector of the canonical basis of $\mathbb{Z}_p^3$, $\boldsymbol{r}_i \leftarrow \mathbb{Z}_p^3$, $\varepsilon \leftarrow \mathbb{Z}_p$.

- Game_3: Let $\alpha_i(X)$ and β_i be defined as above. This game is identical to the previous one, except that $\mathbf{Q}_2$ is now a uniformly random matrix conditioned on having rank 1, and each $\left[\hat{\boldsymbol{\phi}}_i\right]_2$ is now defined as

$$\left[\hat{\boldsymbol{\phi}}_i\right]_2 = [\alpha_i(s)]_2\boldsymbol{e}_2^4 + [\beta_i]_2\boldsymbol{e}_3^4 + \beta_i[\varepsilon]_2\boldsymbol{e}_4^4 + [\mathbf{Q}_2]_2\widetilde{\boldsymbol{r}}_i,$$

where $\boldsymbol{e}_j^4$ is the jth vector of the canonical basis of $\mathbb{Z}_p^4$, $\widetilde{\boldsymbol{r}}_i \leftarrow \mathbb{Z}_p^4$ and $\varepsilon \leftarrow \mathbb{Z}_p$ is the same value used in the definition of $[\boldsymbol{\phi}_i]_1$.

Obviously, the games Real and Game_0 are indistinguishable. The proofs of indistinguishablility of $\mathsf{Game}_1, \mathsf{Game}_2$ and $\mathsf{Game}_2, \mathsf{Game}_3$ are the same as their analogues in Section 3.3.1. We proceed to prove that in Game_3 the adversary wins only with negligible probability.

Lemma 21. *There exists adversaries $\mathcal{B}_{3,i}$ against the soundness of $\Pi_i.\mathsf{LS}$, an adversary $\mathcal{B}_4$ against $\mathcal{Z}$-GSDH in $\mathbb{G}_1$, and an adversary $\mathcal{B}_5$ against n-QTSDH such that*

$$\Pr[\mathsf{Game}_3(\mathcal{A}) = 1] \leq \mathsf{Adv}_{\mathsf{LS}}(\mathcal{B}_{3,1}) + \mathsf{Adv}_{\mathsf{LS}}(\mathcal{B}_{3,2}) + \mathsf{Adv}_{n\text{-}\mathsf{QTSDH}}(\mathcal{B}_4) + \mathsf{Adv}_{\mathcal{Z}\text{-}\mathsf{GSDH},\mathbb{G}_1}(\mathcal{B}_5).$$

Proof. Let E_1 be the event where $(\boldsymbol{c}, V, \boldsymbol{q}_1)$ is not in the image of $\mathbf{M}$, E_2 the event that $(Y, \boldsymbol{q}_2)$ is not in the image of $\mathbf{N}$, and $E_3 = \overline{E_1 \cup E_2}$. Then

$$\begin{aligned}\Pr[\mathsf{Game}_3(\mathcal{A}) = 1] \leq{}& \Pr[\mathsf{Game}_3(\mathcal{A}) = 1|E_1] + \Pr[\mathsf{Game}_3(\mathcal{A}) = 1|E_2] + \\ &+ \Pr[\mathsf{Game}_3(\mathcal{A}) = 1|E_3],\end{aligned} \tag{3.15}$$

and, clearly,

$$\Pr[\mathsf{Game}_3(\mathcal{A}) = 1|E_1] + \Pr[\mathsf{Game}_3(\mathcal{A}) = 1|E_2] \leq \mathsf{Adv}_{\Pi_1.\mathsf{LS}}(\mathcal{B}_{3,1}) + \mathsf{Adv}_{\Pi_2.\mathsf{LS}}(\mathcal{B}_{3,2}).$$

We now proceed to bound $\Pr[\mathsf{Game}_3(\mathcal{A}) = 1|E_3]$. Conditioned on E_3, there exist some $\boldsymbol{x}^\dagger, \boldsymbol{w}, \delta_v, \boldsymbol{r}_{q.1}$ and $\boldsymbol{y}^\dagger, \delta_y, \boldsymbol{r}_{q.2}$ such that $(\boldsymbol{c}, V, \boldsymbol{q}_1)^\top = \mathbf{M}(\boldsymbol{x}^\dagger, \boldsymbol{w}, \delta_v, \boldsymbol{r}_{q.1})^\top$ and $(Y, \boldsymbol{q}_2)^\top = \mathbf{N}(\boldsymbol{y}^\dagger, \delta_y, \boldsymbol{r}_{q.2})^\top$. Given that $\boldsymbol{c}$ is perfectly binding, it must be that $\boldsymbol{x} = \boldsymbol{x}^\dagger$. It follows that $V = \sum_{i=1}^n x_i\lambda_i(s) + \delta_v t(s) = v(s)$ and $Y = y^\dagger(s)$ for some polynomial $y^\dagger(X) = \sum_{i=1}^n \sum_{j=1}^m y_{i,j}^\dagger \sigma_i \lambda_i(X) + \delta_y t(X)$. Further, except with probability $1/q$, each $\boldsymbol{e}_j^i$ is linearly independent of the columns of $[\mathbf{Q}_1]_1, [\mathbf{Q}_2]_2$, so one can extract from $[\boldsymbol{q}_1]_1$ (resp. $[\boldsymbol{q}_2]_2$) the coefficients of these vectors in its expression in terms of

$[\mathbf{Q}_1]_1, \boldsymbol{e}_2^3, \boldsymbol{e}_3^3$ (resp. $[\mathbf{Q}_2]_2, \boldsymbol{e}_2^4, \boldsymbol{e}_3^4, \boldsymbol{e}_4^4$), which are:

$$\begin{bmatrix} \sum_{i=1}^{n+1} x_i \alpha_i(s) \\ \sum_{i=1}^{n+1} x_i \beta_i \varepsilon \end{bmatrix}_1 = \begin{bmatrix} \alpha(s) \\ \beta\varepsilon \end{bmatrix}_1,$$

$$\begin{bmatrix} \sum_{i,j=1}^{m,n} y_{i,j}^\dagger \sigma_i \tilde{\alpha}_j(s) + \delta_y \tilde{\alpha}_{n+1}(s) \\ \sum_{i,j=1}^{m,n} y_{i,j}^\dagger \sigma_i \beta_j + \delta_y \tilde{\beta}_{n+1} \\ \sum_{i,j=1}^{m,n} y_{i,j}^\dagger \sigma_i \beta_j \varepsilon + \delta_y \tilde{\beta}_{n+1} \varepsilon \end{bmatrix}_2 = \begin{bmatrix} \tilde{\alpha}(s) \\ \tilde{\beta} \\ \tilde{\beta}\varepsilon \end{bmatrix}_2$$

where $x_{n+1} = \delta_v$ and $\alpha(X), \tilde{\alpha}(X)$ are the quotients and $\beta, \tilde{\beta}$ are the reminders of dividing, respectively, $v(X)$ and $y(X)$ by $X - r_{j^*}$.

If we divide both sides of the verification equation by $(s - r_{j^*})$, and we denote by $\alpha_0(s), \beta_0$ we get that

$$\begin{aligned} e\left([H]_1, \left[\frac{t(s)}{s - r_{j^*}}\right]_2\right) =& \frac{1}{s - r_{j^*}} (e([v_0(s)]_1 - [v(s)]_1, [y(s)]_2) - [1]_T) \\ =& \frac{1}{s - r_{j^*}} \left[(v_0(s) - v(s))(\tilde{\alpha}(s)(s - r_{j^*}) + \tilde{\beta}) - 1\right]_T \\ =& [(v_0(s) - v(s))\tilde{\alpha}(s) + \alpha(s)\tilde{\beta}]_T + \left[\frac{(v_0(s) - \beta)\tilde{\beta} - 1}{s - r_{j^*}}\right]_T. \end{aligned}$$

Note that $\beta = v(r_{j^*}) = x_{j^*}, v_0(s) = \mathsf{sk}$ and thus if $(v_0(s) - \beta)\tilde{\beta} - 1 = 0$, then $\tilde{\beta}$ is a valid signature for x_{j^*}.

Let E_4 the event $(v_0(s) - \beta)\tilde{\beta} - 1 = 0$ and thus $\Pr[\mathsf{Game}_4(\mathcal{A}) = 1 | E_3] \leq \Pr[\mathsf{Game}_4(\mathcal{A}) = 1 | E_4 \cap E_3] + \Pr[\mathsf{Game}_4(\mathcal{A}) = 1 | \overline{E}_4 \cap E_3]$.

We build an adversary $\mathcal{B}_4$ against Assumption 6 which receives $gk, \{[\mathsf{sk}^i]_1, [\mathsf{sk}^i]_2\}_{i \in [m]}$, $[\varepsilon]_{1,2}$. Essentially, the adversary works as the one described in Section 3.5.1 for the (non-aggregated) set membership argument. It simulates $\mathsf{Game}_4(\mathcal{A})$ computing all the discrete logarithms of the crs itself, except for the Boneh-Boyen secret key, $[\varepsilon]_{1,2}$, and the signatures in the crs are computed as in Section 3.5.1. When $\mathcal{A}$ outputs $[\boldsymbol{q}_1]_1, [\boldsymbol{q}_2]_2$, $\mathcal{B}_4$ extracts $[\beta\varepsilon]_1, [\tilde{\beta}]_2$ and returns $([x_{j^*}]_1, [\beta\varepsilon]_1, [\tilde{\beta}]_2)$. In the case E_4, we have already argued that $\tilde{\beta}$ is a valid signature for x_{j^*}, and in this game $x_{j^*} \notin S$. We conclude that $\Pr[\mathsf{Game}_4(\mathcal{A}) = 1 | E_4 \cap E_3] \leq \mathsf{Adv}_{\mathcal{Z}\text{-}\mathbf{GSDH}, \mathbb{G}_1}(\mathcal{B}_4)$.

We also construct $\mathcal{B}_5$ an adversary against Assumption 8. It receives as input $[\varepsilon]_1, [\varepsilon]_2, [s]_1, [s]_2, \ldots, [s^d]_1 [s^d]_2$ and it starts a simulation of $\mathsf{Game}_4(\mathcal{A})$, by sampling honestly the rest of the elements of the crs. Finally, $\mathcal{A}$ outputs $[V]_1, [Y]_2, [\boldsymbol{q}_1]_1, [\boldsymbol{q}_2]_2$ as part of the purported proof for $[\boldsymbol{c}]_1$. We will see in the following how $\mathcal{B}_4$ computes $[\nu]_T := \left[\frac{(v_0(s) - \beta)\tilde{\beta} - 1}{s - r_{j^*}}\right]_T$ and returns $([v_0(s) - \beta]_1, [(v_0(s) - \beta)\varepsilon]_1, [\tilde{\beta}]_2, [\tilde{\beta}\varepsilon]_2,$

$[\nu]_T$), with $(v_0(s) - \beta)\tilde{\beta} - 1 \neq 0$, breaking Assumption 8.

The values $[\tilde{\alpha}(s)]_2, [\tilde{\beta}]_2$ and $[\tilde{\beta}\varepsilon]_2$ are extracted from $[\boldsymbol{q}_2]_2$, while $[\alpha(s)]_1, [\beta\varepsilon]_1$ are extracted from $[\boldsymbol{q}_1]_1$, $[\beta]_1 = [x_{j^*}]_1$ is extracted from $[\boldsymbol{c}]_1$, $\beta_0 = \mathsf{sk}$, and $[v_0(s)\varepsilon]_1 = \mathsf{sk}[\varepsilon]_1$ can be computed by $\mathcal{B}_5$ because it sampled sk. The value $[\nu]_T$ is computed as

$$[\nu]_T := e\left([H]_1, \left[\frac{t(s)}{s - r_{j^*}}\right]_2\right) - e([v_0(s)]_1 - [V]_1, [\tilde{\alpha}(s)]_2) - e\left([\alpha(s)]_1, [\tilde{\beta}]_2\right).$$

□

With this lemma we finish the security proof of the Theorem 20. □

Zero-Knowledge.

The proof of perfect zero-knowledge is essentially the same as for Theorem 14. Note that $[V]_1, [Y]_2, [\boldsymbol{q}_1]_1, [\boldsymbol{q}_2]_2$ are independent of $\boldsymbol{x}$, while $[H]_1$ is the unique solution to the verification equation. Perfect zero-knowledge of the argument of membership in linear spaces implies that the proofs ψ_1, ψ_2 can be simulated with the same distribution as honest proofs.

3.6 Applications

3.6.1 Shuffle Arguments

From our results, we can construct two different shuffle arguments in the crs model under falsifiable assumptions. They both follow the basic template of the shuffle argument of [68]. Let $[\boldsymbol{c}_1]_2, [\boldsymbol{c}_2]_2$ be two vectors of n ciphertexts which open to vectors of plaintexts $[\boldsymbol{m}_1]_2, [\boldsymbol{m}_2]_2$, respectively, and we want to prove that $\boldsymbol{m}_2$ is a permutation of $\boldsymbol{m}_1$. The shuffle argument of [68] consists of the following steps. The crs includes a vector of group elements $[\boldsymbol{z}]_1 = ([z_1]_1, \ldots, [z_n]_1)$ sampled uniformly and independently. The prover chooses a permutation $[\boldsymbol{x}]_1 = ([x_1]_1, \ldots, [x_n]_1)$ of $[\boldsymbol{z}]_1$ and proves: (1) $x_i \in \mathcal{Z} = \{z_1, \ldots, z_n\}$ for all $i \in \{1, \ldots, n\}$, (2) $\sum x_i = \sum z_i$ and (3) $\sum z_i m_{1,i} = \sum x_i m_{2,i}$.

The first two steps force $\boldsymbol{x}$ to be a permutation of $\boldsymbol{z}$: if all $x_i \in \mathcal{Z}$ and their sum equals the sum of all the elements in $\mathcal{Z}$ and $\boldsymbol{x}$ is not a permutation, the prover has found a non-trivial combination of elements of $\mathcal{Z}$ which is 0, which is a type of kernel problem. The last step links this fact with $\boldsymbol{m}_2$ being a permutation of $\boldsymbol{m}_1$.

In both our constructions and in the original argument of [68], Steps (2) and (3) are handled with the following Groth-Sahai equations, in which uppercase letters are

variables for which the prover has provided commitments: (2) $\sum[X_i]_1 = \sum[z_i]_1$ and (3) $\sum e([z_i]_1, [M_{1,i}]_2) = \sum e([X_i]_1, [M_{2,i}]_2)$.

We next specify two different ways of proving Step 1, which results in two different constructions with different performance.

Unit Vector Argument

The first approach is the closest to the work of González et al. [68]. There, Step 1 is rewritten as proving that $\boldsymbol{x} = \boldsymbol{z}^\top \boldsymbol{B}$, for a matrix $\boldsymbol{B} = (\boldsymbol{b}_1 | \ldots | \boldsymbol{b}_n) \in \{0,1\}^{n^2}$, where the $\boldsymbol{b}_i$ are unitary vectors (not necessarily different, as this is handled by step 2). The approach of [68] is to adopt a commit-and-prove strategy using arguments for linear spaces and the bitstring argument of [67]. The 'prove' part is constant-size, but the 'commit' part is a priori quadratic, as we would need to commit to each entry of the matrix $\boldsymbol{B}$.

To overcome this and obtain linear complexity, they switch to shrinking commitments to each row $\boldsymbol{b}_i^*$ of $\boldsymbol{B}$, which take only two elements each. Obviously these commitments cannot be perfectly binding, and this fact interferes with the extraction step in soundness proof. However, a key step in their argument is that they set these commitments in a way that one single coordinate j^* (which remains unknown to the adversary) is perfectly binding. Thus the corresponding column is uniquely determined and can be extracted in the proof. From here, it is concluded that an adversary cannot cheat in the j^*-th ciphertext, and since j^* is unknown to the adversary, general soundness is reduced to this case with a tightness loss of $1/n$. Note that this is on top of the factor $1/n$ from the bitstring argument, resulting in a soundness loss of $1/n^2$.

We observe that we can plug our unit vector argument instead of the one from [67], modified to accept shrinking commitments to each of the rows of $\mathbf{B}$ as those in [68]. We include an additional game at the beginning of the soundness proof of the unit vector argument, in which we choose a random coordinate and abort if the corresponding commitment is not in the language. From here on the proof works as in Section 3.4. This proof inherits the disadvantages of [68], namely the quadratic crs and the tightness loss in the security reduction, but we improve the proof size from $(4n+17)|\mathbb{G}_1| + 14|\mathbb{G}_2|$ to $(4n+11)|\mathbb{G}_1| + 8|\mathbb{G}_2|$ and our proof still uses falsifiable and static assumptions.

Argument of Membership in a Set of Group Elements

Another approach to Step 1, instead of the aggregated unit vector proofs, is to prove directly membership in a subset $\mathcal{Z} = \{[z_1]_1, \ldots, [z_n]_1\} \subset \mathbb{G}_1$. Note that the set is witness sampleable and in particular, the discrete logarithms might be known when

generating the crs. More precisely, we want to construct an argument for the language

$$\mathcal{L}_{\mathsf{memb},\mathcal{Z},\mathsf{ck}} := \left\{ [\boldsymbol{c}]_1 \in \mathbb{G}_1^2 \,\middle|\, \exists \boldsymbol{w} \in \mathbb{Z}_p \text{ s.t. } [\boldsymbol{c}]_1 = \mathsf{Com}_{ck}([x]_1; w) \text{ and } [x]_1 \in \mathcal{Z} \right\},$$

and for efficiency, the proof should be aggregated. This can be achieved by modifying the aggregated membership proof in a subset of $\mathbb{Z}_p$ from Section 3.5.2. Note that there we had $x \in \mathbb{Z}_p$, and this was necessary to produce the proof, so to ensure completeness when the prover knows only $[x]_1 \in \mathcal{Z} \subset \mathbb{G}_1$, we provide additional elements in the crs. This is possible because the set is witness sampleable. More precisely, x was involved in the definition of the terms

$$[V]_1 = [v(s)]_1, \qquad \text{where } v(X) = \sum_{i=1}^{n} x_i \lambda_i(X) + \delta_v t(X),$$
$$[\boldsymbol{q}_1]_1 = \left[\sum_{i=1}^{n} x_i \boldsymbol{\phi}_i + \delta_v \boldsymbol{\phi}_{n+1} + \mathbf{Q}_1 \boldsymbol{r}_{q.1} \right]_1,$$

so we include the elements $\{[z_i \lambda_j(s)]_1, [z_i \phi_j]_1\}_{i,j \in \{1,\ldots,n\}}$ in the crs. The proof works exactly the same, as the reduction could only open the commitments in the group.

We can use this to prove Step 1 of the shuffle argument above. In this case, the crs size is still quadratic in the number of ciphertexts, but we avoid losing the second factor $1/n$ in the reduction, and the proof consists only of the commitments to $[x_i]_1$ and a constant number of elements. More precisely, the proof size is $(2n+11)|\mathbb{G}_1|+8|\mathbb{G}_2|$.

3.6.2 Range Argument in the Interval $[0, 2^n - 1]$

We want to prove that a Groth-Sahai commitment $[\boldsymbol{c}]_1$ opens to some integer y in the range $[0, 2^n - 1]$. That is, we want to construct a NIZK proof system for the language

$$\mathcal{L}_{\mathsf{range},ck} := \left\{ [\boldsymbol{c}]_1 \in \mathbb{G}_1^2 : \begin{array}{l} \exists y, r \in \mathbb{Z}_p \text{ s.t. } [\boldsymbol{c}]_1 = \mathsf{Com}_{ck}(y; r) \\ \text{and } y \in [0, 2^n - 1] \end{array} \right\},$$

where $ck := ([\boldsymbol{u}_1]_1, [\boldsymbol{u}_2]_1) \leftarrow \mathsf{K}_0(1^\lambda)$.

We follow a widely used approach (for example [114, 30] to name a few), which divides the statement $y \in [0, 2^n - 1]$ into ℓ range proofs in smaller intervals. That is,

1. commit to $y_1, \ldots, y_\ell$,
2. show that $y_i \in [0, d-1]$, for each $i \in [\ell]$,
3. show that $y = \sum_{i \in [\ell]} y_i d^{i-1}$.

We commit to $y_1, \ldots, y_\ell$ using only $\ell + 1$ group elements using a simple adaptation of ElGamal to vectors of size n. To prove point 3 we could use membership in linear spaces, as done in [67, Sect. 5.5], requiring only one element from $\mathbb{G}_1$. For point 2 we use our aggregated set-membership proof which requires 6 elements of $\mathbb{G}_1$ and 6 of $\mathbb{G}_2$. The total size of the proof is thus $\left(\frac{n}{\log d} + 7\right) \mathbb{G}_1 + 6\mathbb{G}_2$. Choosing $d = n^k$ we get that $\ell = \frac{n}{\log n^k} = \frac{n}{k \log n}$, and thus the size of our Range Proof is $\left(\frac{n}{k \log n} + 7\right) \mathbb{G}_1 + 6\mathbb{G}_2$, for an arbitrarily chosen $k \in \mathbb{N}$. The size of the crs is dominated by $5\ell \cdot d = 5\frac{n^{k+1}}{k \log n}$ (the size of matrix $\mathbf{N}$ in our set membership proof).

In practice, the size of the proof is bounded by the security parameter, i.e. $n < 128$ (one can't commit to a number bigger than the field size). Although for such a big n the size of the crs is huge, ≈ 12000 and ≈ 730000 group elements for $k = 1, 2$ respectively, the size of the proof is just 26 and 18 group elements for $k = 1, 2$ respectively. For $n = 64$ and $k = 2$, the size of a proof is 13 group elements, it requires roughly 70000 group elements in the crs. For $n = 64$ and $k = 1$, the size of a proof is bounded by ≈ 18 group elements and the crs contains roughly 2000 group elements. For more conservative ranges, say $n \approx 10$, one gets proofs of size 10 group elements while the crs contains roughly 500 group elements, for $k = 2$, or of size 12 with a crs of size 50 for $k = 1$.

	Language	Proof size	crs size	Assumption
Sect. 3.6.2	Range Proof	$\left(\frac{n}{k \log n} + 7\right) \lvert\mathbb{G}_1\rvert$ $+6\lvert\mathbb{G}_2\rvert$	$\left(\frac{n^{k+1}}{k \log n} + O(1)\right) \lvert\mathbb{G}_1\rvert +$ $+ \left(5\frac{n^{k+1}}{k \log n} + O(1)\right) \lvert\mathbb{G}_2\rvert$	$\mathcal{Z}$-GSDH [6], q-QTSDH [8]
Sect. 4 [114]		$\approx 15\frac{n}{\log n}(\lvert\mathbb{G}_1\rvert + \lvert\mathbb{G}_2\rvert)$	$O\left(\frac{n}{\log n}\right)$	q-HSDH

Table 3.5: The table shows the proof sizes (not including commitments for bitstring and unit vector) and crs sizes of our results in range proofs. The range considered is $[0, 2^n - 1]$ and $k > 0$ is a free parameter (e.g. $k = 1/4, 1/2, 1, 2, \ldots$), and the constant of [114] is at least 4, for committing to signatures, plus $3 \cdot 4$ elements for Groth-Sahai proofs of the signature verification.

Chapter 4

Signatures of Knowledge for Boolean Circuits under Standard Assumptions

In this chapter we present the full version of our result *Signatures of Knowledge for Boolean Circuits Under Standard Assumptions* published in *Africacrypt* 2020.

4.1 Introduction

Due to their impressive advantages and functionalities, as we have already mentioned, NIZK proof systems are used ubiquitously to build larger cryptographic protocols and systems [18, 80]. Among the various constructions of NIZK arguments, there is usually a trade-off between several performance measures, in particular, between efficiency, generality and the strength of the assumptions used in the security proof.

Zero-knowledge Succinct Arguments of Knowledge (zk-SNARKs) [61, 72] are among the most practically interesting NIZK proofs. They allow to generate succinct proofs for NP-complete languages (3 group elements for CircuitSat [72]), but they are constructed based on non-falsifiable assumptions (e.g. knowledge assumptions [42], Section 2.4). A well-known impossibility result of Gentry and Wichs [64] shows that this is unavoidable if one wants to have succinctness for general languages. Thus, non-falsifiable assumptions are an essential ingredient to have very efficient constructions, while falsifiable assumptions give stronger security guarantees and more explicit and

meaningful security reductions [105].

Groth-Sahai proofs [78] also allow to prove general languages[1] under standard assumptions non-succinctly, trading security for succinctness. On the other hand, some QA-NIZK constructions generate very efficient proofs based on falsifiable assumptions for very specific statements (e.g. membership in linear spaces). Somewhere in between, recent work by González and Ràfols [69] constructs a NIZK argument for boolean CircuitSat under falsifiable assumptions by combining techniques of QA-NIZK arguments and zk-SNARKs. The proof size of their construction is $O(n_s + d)$ group elements, where n_s is the length of the secret input and d is the depth of the circuit.

The primary requirements in a NIZK argument are *Completeness*, *Zero-Knowledge* (ZK), and *Soundness*. However, in practice usually bare soundness is not sufficient and one might need stronger variations of it, known as *Knowledge Soundness*, *Simulation Soundness* or *Simulation Knowledge Soundness* (a.k.a. Simulation Extractability) [115, 70]. As we explain in Section 2.7.1, knowledge soundness ensures that if an adversary manages to come up with an acceptable proof, he must *know* the witness. Simulation soundness (a.k.a. unbounded simulation soundness) ensures that an adversary cannot come up with valid proof for a false statement, even if he has seen an arbitrary number of simulated proofs. This notion basically guarantees that the proofs are sound and non-malleable. The strongest case, Simulation Extractability (SE) implies that an adversary cannot come up with a *fresh* valid proof unless he knows a witness, even if he has seen an arbitrary number of simulated proofs. In both notions knowledge soundness and simulation extractability the concept of *knowing* is formalized by showing that there exists an extraction algorithm, either non-Black-Box (nBB) or Black-Box (BB), that can extract the witness from the proof.

Zk-SNARKs (either knowledge sound ones [61, 72], or SE ones [74, 13]) are probably the best-known family of NIZK arguments. They achieve knowledge soundness with nBB extraction under non-falsifiable assumptions. As we mentioned in Section 1.2 about the two types of extraction, although SE with nBB extraction is a stronger notion in comparison with (knowledge) soundness, an ideal-world simulator should be able to extract witnesses without getting access to the source code of environment's algorithm, which is only guaranteed by BB SE [32, 70].

SE NIZK arguments have great potential to be deployed in practice [94, 90], or construct other primitives such as Signature-of-Knowledge (SoK) [36]. In a SoK, a valid signature of a message m for some statement $\boldsymbol{x}$ and a relation $\mathbf{R}$ can only be produced if the signer knows a valid witness $\boldsymbol{w}$ such that $(\boldsymbol{x}, \boldsymbol{w}) \in \mathbf{R}$. Groth and

[1] GS proofs allow to prove satisfiability of any quadratic equation over $\mathbb{Z}_p$, where p is the order of a bilinear group. In particular, this can encode CircuitSat. The size of the resulting proof is linear in the total number of wires.

Maller [74] constructed a SE zk-SNARK and a generic construction of a SoK from any SE NIZK argument, resulting in an SoK for CircuitSat. While their construction is for general NP relations and it is also succinct, it also relies on non-falsifiable assumptions and the extraction is nBB.

In this chapter, we construct a SE NIZK argument with BB extraction for Boolean CircuitSat which is secure under falsifiable assumptions. The proposed construction is based on the result of [69]. We show that the proposed construction adds minimal overhead to the original construction, resulting in a SE NIZK argument with BB extraction and proof size $O(n + d)$, where d is the depth and n is the input size of the circuit. Moreover, the proposed construction also allows one to construct a tight SoK of the same size.

The restriction to Boolean CircuitSat (and not arithmetic) for our SE-NIZK argument is inherited from the NIZK argument of [69] on which our argument is based. This restriction is due to the fact that we need the DLOG-based commitments to the input of the circuit to be extractable, and this is only possible (for a BB extractor) if the message space is of polynomial size. Thus, we restrict ourselves to the important special case of Boolean CircuitSat. As an independent result, in this paper we also give a simple formula to encode Boolean CircuitSat as a Quadratic Arithmetic Program [61], which we later use for our construction.

4.1.1 Our Contribution

Trivial Approach for Boolean CircuitSat

Let ϕ be some boolean circuit, and let a_i, b_i, c_i be the left, right and output wires of gate i. A zero-knowledge argument for Boolean CircuitSat, where the prover shows knowledge of some secret input satisfying the circuit, can be divided into three sub-arguments:

1) an argument of knowledge of some boolean input: to prove that the secret input is boolean, the prover must show that each input value satisfies some quadratic equation,

2) a set of linear constraints, which proves "correct wiring", namely that a_i, b_i are consistent with $\boldsymbol{c}$ and the specification of the circuit,

3) a set of quadratic constraints, which proves that for all i, a_i, b_i and c_i are in some quadratic relation which expresses correct evaluation of gate i.

It is straightforward to prove CircuitSat by computing perfectly binding commitments to all the wires a_i, b_i, c_i and use, for example, Groth-Sahai NIZK proofs for each of

the three sub-arguments. However, the proof size is obviously linear in the number of wires.

New Techniques

In a recent result, González and Ràfols [69] give a proof for Boolean CircuitSat of size $O(n_s + d)$ group elements under falsifiable assumptions in bilinear groups. We now give an overview of their techniques, which is the main building block of our paper. The key to their result is to prove 2) and 3) succinctly for each level of the circuit. More specifically (ignoring zero-knowledge, momentarily), if L_j (resp. R_j, O_j) is a shrinking (non-hiding, deterministic) commitment to all left (resp. right, output) wires at depth j, they construct:

2') an argument that shows that the opening of L_j (resp. R_j) is in the correct linear relation (given by the wiring constraints in the circuit specification) with the input and the openings of $O_1, \ldots, O_{j-1}$,

3') an argument that shows that the opening of O_j is in the correct quadratic relation (which depends on the type of gates at level j) with the opening of L_j and R_j.

The abstraction given above of the results of [69] hides an important subtlety: "the opening of L_j" (and similarly for the other shrinking commitments O_j, R_j) is not well defined, as many openings are possible, so it is unclear what it means for these sub-arguments to be sound. However, as the authors of [69] observe, when we are using these as part of a global proof of CircuitSat, "the opening of L_j" to which we intuitively refer is well defined in terms of the openings in previous levels. In other words, in the soundness proof, 2') can be used to prove that if the reduction can extract an opening of $O_1, \ldots, O_{j-1}$ consistent with the input and the circuit, it can also extract a consistent opening of L_j (and similarly R_j). On the other hand, 3') shows that if the reduction can extract an opening of L_j and R_j consistent with the input and the circuit, it can also extract an opening of O_j. For this reason, González and Ràfols informally called 2') and 3') "arguments of knowledge transfer" (linear and quadratic, respectively): given knowledge of the input, arguments 2') and 3') can be used alternatively to transfer this knowledge to lower levels of the circuit.

Promise Problems

To formalize this intuitive notion, the authors of [69] define their sub-arguments 2') and 3') as arguments (with completeness and soundness) for certain promise problems:

2') Given the input $\boldsymbol{c}_0$ and openings $(\boldsymbol{c}_1, \ldots, \boldsymbol{c}_{j-1})$ of $O_1, \ldots, O_{j-1}$, the argument shows that L_j can be opened to some $\boldsymbol{a}_j$ with the correct linear relation to $(\boldsymbol{c}_0, \boldsymbol{c}_1, \ldots, \boldsymbol{c}_{j-1})$ (similarly for R_j).

3') Given $\boldsymbol{a}_j$ and $\boldsymbol{b}_j$, openings of L_j and R_j, the argument shows that there is an opening $\boldsymbol{c}_j$ of O_j that is in the correct quadratic relation (which depends on the type of gates at level j) with $\boldsymbol{a}_j$ and $\boldsymbol{b}_j$.

From an efficiency point of view, the interesting thing is that the arguments are of constant size. This explains the proof size $O(n+d)$: $O(n)$ is for committing to the input (with extractable commitments, which exist under falsifiable assumptions because the input is boolean), and d is the cost of doing 2') and 3') repeatedly for each level. At a conceptual level, the key issue is that the verifier never checks that the openings are correct (i.e. in 2') it never checks that $\boldsymbol{c}_i$ is a valid opening of O_i, and in 3') that $\boldsymbol{a}_j$, $\boldsymbol{b}_j$ are valid openings of L_j, R_j), which is *the promise*. Soundness is only guaranteed if the promise holds, and nothing is said when it does not hold (when the given openings are invalid). In fact, the verifier does not need these openings, they are just part of the statement to define soundness in a meaningful way, reflecting the fact that in the global argument for boolean CircuitSat, the openings at level j are uniquely determined by transferring the knowledge of the circuit to lower levels. So excluding the need to read the statement, the verifier works in constant time (it would work in linear time if it verified the statement). In particular, when using the sub-arguments in a global proof, verification of each of the sub-arguments is constant size, and the global verifier runs in time $O(n_s + d)$.

Security Proof

The sub-arguments 2') and 3') of [69] are not new. More specifically, for 2') the authors just use the QA-NIZK argument of linear spaces for non-witness samplable distributions of Kiltz and Wee [92], a generalization of [85, 96] and for 3') they use techniques appeared in the context of zk-SNARKs (as e.g. [61]) to write many quadratic equations as a single relation of polynomial divisibility that can be proven succinctly. The challenge they solve is to give a proof that 2') and 3') are sound for the aforementioned promise problems under falsifiable assumptions, which is not implied by the soundness of the NIZK arguments they use for 2') and 3'). More specifically, for the linear constraints the soundness of the argument of membership in a linear space does not protect from "witness switching attacks" as explained in [69]. Indeed, to prove that two shrinking commitments $\boldsymbol{c}_1$, $\boldsymbol{c}_2$ open to vectors of values with a certain linear relation, it is natural to write this as a membership proof in a linear space defined by matrices $\mathbf{M}$,

$\mathbf{N}$, i.e. to prove that $\begin{pmatrix} \boldsymbol{c}_1 \\ \boldsymbol{c}_2 \end{pmatrix} \in \begin{pmatrix} \mathbf{M} \\ \mathbf{N} \end{pmatrix}$, which ensures that there exists some $\boldsymbol{w}$ such that $\boldsymbol{c}_1 = \mathbf{M}\boldsymbol{w}$ and $\boldsymbol{c}_2 = \mathbf{N}\boldsymbol{w}$. However, given some opening $\boldsymbol{w}$ of $\boldsymbol{c}_1$ (which in our analysis is known because of knowledge of the input and the transfer to lower levels of the circuit), the argument does not prove that $\boldsymbol{c}_2 = \mathbf{N}\boldsymbol{w}$, as it only proves that there is *some* common opening. Therefore, standard soundness does not prevent the adversary from "switching the witness": if the adversary is able to find another witness $\boldsymbol{w}'$ such that $\boldsymbol{c}_1 = \mathbf{M}\boldsymbol{w} = \mathbf{M}\boldsymbol{w}'$ it can use $\boldsymbol{w}'$ for $\boldsymbol{c}_2$, for some $\boldsymbol{w}'$ that does not satisfy the linear constraints.

This attack is easy to reduce to the binding property of commitment schemes if the reduction can extract $\boldsymbol{w}'$ from the adversary, but since the commitments are shrinking, this would require some non black-box extraction, deviating from the goal of using standard assumptions. The authors of [69] get around this by showing how to prove soundness for the promise problems associated to linear constraints using a decisional assumption related to the matrix $\mathbf{M}$. For 3') they prove that the soundness of their argument for the promise problem is a straightforward consequence of a q-type assumption in bilinear groups.

Our Techniques: General Approach

This paper builds a SE NIZK for CircuitSat under falsifiable assumptions building on the work of [69]. There are several generic techniques to solve this problem. To the best of our knowledge, existing generic solutions are variations of the following approach, described for example in [70]: build an OR proof that given some circuit ϕ and a public input $\boldsymbol{x}_p$, either the circuit is satisfiable with public input $\boldsymbol{x}_p$ or a signature of $M = (\phi, \boldsymbol{x}_p)$ is known. The simulator uses as a trapdoor the signature secret key. We note that this approach results in a considerable (although also constant) overhead (around 20 group elements).[2] Our approach is based on the following observation: to compute "fake proofs" of satisfiability, a simulator just needs to lie either about the satisfiability of quadratic equations or linear equations, but not both. Further, it is sufficient to lie in the last gate. In particular, we choose the following strategy to simulate a proof for a circuit ϕ and a public input $\boldsymbol{x}_p$: complete the input arbitrarily, compute consistent assignments to all gates but choose the last left and right wire arbitrarily so that the last gate outputs one. Thus the simulator outputs only honest proofs except for the last linear relation, which is a simulated proof for a false statement, i.e. the simulator does not need the simulation trapdoor for sub-arguments 1) and 3') and standard soundness is sufficient. To be consistent with this strategy, our SE NIZK for boolean CircuitSat

[2]Using OR proofs (the less efficient construction for PPE given in [112] or adding a bit as an auxiliary variable) plus the Boneh-Boyen signature for adaptive soundness.

uses the construction of [69] but replaces 2'), the proof that the linear relation holds, with 2") an unbounded simulation-sound proof for the same promise problem.

Recall that the argument 2') of [69] is just the QA-NIZK argument for membership in linear spaces of Kiltz and Wee for non-witness samplable distributions with a security proof adapted for promise problems (non-trivially). We take the most efficient Unbounded Simulation Sound (USS) QA-NIZK argument of membership in linear spaces in the literature, also due to Kiltz and Wee [92] and we adapt the USS argument to work for bilateral linear spaces (linear spaces split among the two source groups in a bilinear group) as in [67] and for promise problems as in [69]. The overhead of the construction with respect to the original CircuitSat proof is minimal ($3|\mathbb{G}_1|$). BB extractability is achieved because of the soundness of the argument which proves that the input is boolean and the fact that ElGamal ciphertexts of 0 or 1 are BB extractable (the extraction trapdoor is the secret key).

Our approach modularly combines a USS argument of membership in linear spaces with other arguments. The USS NIZK argument of Kiltz and Wee is not tight. However, to get tight security we only need to construct a tight USS for promise problems for linear spaces (or for bilateral spaces if we want to improve efficiency). In Section 4.7 we give such a construction, we take the most tight QA-NIZK argument in the literature, Abe et al. [5], and we adapt the security proof to build an argument for the promise problem related to satisfiability of linear constraints. The result is a signature of knowledge for circuits with a loss of d (the circuit depth) in the reduction (inherited from [69]), but independent of the number of queries to the simulation oracle.

As Groth and Maller [74] pointed out, USS arguments for CircuitSat are very close to Signatures of Knowledge (SoK). We use the fact that our CircuitSat argument is tag-based to obtain a very simple transformation to SoK. In particular, our second construction results in a tight SoK.

Adapting the USS Argument to Promise Problems

Technically, the main challenge that we solve is to prove that the tag-based USS arguments for membership in linear spaces of Kiltz and Wee [92] (in Section 4.6) and of Abe et al. [5] (in Section 4.7) are sound for the promise problem defined in [69] for linear constraints. More precisely, what we prove is that the adversary cannot create a valid proof for the statement

$$\begin{pmatrix}\boldsymbol{x}\\ \boldsymbol{y}\end{pmatrix} \in \mathbf{Im}\begin{pmatrix}\mathbf{M}\\ \mathbf{N}\end{pmatrix}$$

such that $\boldsymbol{x} = \mathbf{M}\boldsymbol{w}$ for some known $\boldsymbol{w}$ but $\boldsymbol{y} \neq \mathbf{N}\boldsymbol{w}$ even after seeing many simulated proofs. The idea is that if the linear constraints are satisfied until a certain level, they must be satisfied also at lower levels of the circuit.

In the following, we give an overview on how we adapt Kiltz and Wee USS argument for this promise problem. The tight construction based on Abe et al. in Section 4.7 follows the same lines. The main idea of the USS argument of Kiltz and Wee, $\Pi_{\mathsf{LIN\text{-}USS}}$ is to add a pseudorandom MAC to their QA-NIZK argument of membership in linear spaces Π_{LIN}. The soundness of the argument Π_{LIN} that proves membership in the space spanned by the columns of some matrix $\mathbf{U}$ is guaranteed by the fact that $\boldsymbol{y}^\top \mathbf{K}$ is uniformly random in the adversary's view given $\mathbf{UK}$ if $\boldsymbol{y} \notin \text{Span}(\mathbf{U})$. The proof of simulation soundness of $\Pi_{\mathsf{LIN\text{-}USS}}$ shows, in the first place, that under some decisional assumption, the queries made by the adversary do not give additional information to the adversary, in particular, they do not leak additional information about the secret key other than the one in the common reference string. We can adapt this part of their argument in a straightforward way. Then their proof concludes by arguing that in the final game the common reference string information theoretically hides part of the secret key, more concretely, $\boldsymbol{y}^\top \mathbf{K}$ remains information theoretically hidden.

We need to add one extra game in the proof of $\Pi_{\mathsf{LIN\text{-}USS}}$ to account for the fact that in our case $\mathbf{U} = \begin{pmatrix} \mathbf{M} \\ \mathbf{N} \end{pmatrix}$ spans all of the space. In particular, on the one hand, our soundness condition is different, as explained (the adversary breaks soundness for $(\boldsymbol{x}^\top, \boldsymbol{y}^\top)$ if $\boldsymbol{x} = \mathbf{M}\boldsymbol{w}$ for some known $\boldsymbol{w}$ but $\boldsymbol{y} \neq \mathbf{N}\boldsymbol{w}$). On the other hand, the common reference string reveals all information about the secret key (since $\mathbf{U}^\top \mathbf{K}$ reveals everything about $\mathbf{K}$), so the information theoretic argument used by Kiltz and Wee to conclude the proof of $\Pi_{\mathsf{LIN\text{-}USS}}$ does not apply. We solve this in the same way as González and Ràfols [69], who show that if the Matrix Decisional DDH Assumption [52] associated to the distribution of the first block $\mathbf{M}$ holds, then we can switch to a game where $(\mathbf{0}^\top, \mathbf{N}^\top)\mathbf{K}$ is information theoretically hidden. Intuitively, this means the adversary cannot compute valid proofs such that if $\boldsymbol{x} = \mathbf{M}\boldsymbol{w}$ for some known $\boldsymbol{w}$ but $\boldsymbol{y} \neq \mathbf{N}\boldsymbol{w}$), because it does not know the projection of the secret key on the second block without involving the first block.

Generalization of Our Techniques

The observation that to add unbounded simulation soundness to NIZK arguments which prove both quadratic and linear equations it suffices to have USS in the linear part can have other applications. For example, a direct application is to give USS to our main construction in Chapter 3, which gives a compact proof that a set of perfectly binding commitments open to 0 or 1.

A Canonical Transformation of Boolean Circuits to QAPs

To prove quadratic equations compactly, González and Ràfols adopt the idea of [61] to encode many quadratic equations as a problem of divisibility among polynomials. More in detail, in a breakthrough result building on [71], Gennaro et al. [61] introduced in 2013 two caracterizations of circuit satisfiability, as we have already mentioned, (Quadratic Span Programs or QSPs for boolean circuits and Quadratic Arithmetic Programs or QAPs for arithmetic circuits over $\mathbb{Z}_p$ where p should be the order of the bilinear group of the zk-SNARK, inspired by the notion of Span Programs [87]) and proposed an efficient zk-SNARK for it. As we explained in Section 1.2, the basic idea is that the correctness of all the computations of the circuit is expressed as a divisibility relation among certain polynomials which define the program. This leads to a succinct proof in the crs model by checking the divisibility relation only in a secret point given in the crs "in the exponent". In 2014, Danezis et al. [45] introduced Square Span Programs (SSP) for boolean circuits to simplify QSPs. The reason why special encodings for Boolean circuits exist is because these are an important special case, and they have special characteristics (a part from checking gate satisfiability, one must check that the wires are boolean). In 2016, Groth [72] introduced the most efficient zk-SNARK for QAPs, and also mentioned that QAPs can encode boolean CircuitSat but did not give an explicit transformation. González and Ràfols [69] gave an explicit encoding of Boolean CircuitSat, separating linear and quadratic constraints and dividing the encoding by layers of same depth as needed by their construction. That is, essentially they were spelling out a QAP for satisfiability of all boolean gates of the same depth.

We spell out a canonical QAP to describe boolean CircuitSat as a problem of satisfiability of polynomials. We call the transformation canonical because it is essentially the direct and simplest way to do this transformation. Although encoding Boolean CircuitSat as a QAP is not difficult and can be easily done with a computer, we give an exact formula that describes a simple QAP from the description of the gates. This is a contribution of independent interest, and when combined with Groth16's zk-SNARK it results in an argument with the polynomials that define the QAP are very simple, lagrangian polynomials or sums of them. Then, we use this transformation from boolean CircuitSat to QAP to derive a simpler transformation from Boolean CircuitSat (separated in linear and quadratic constraints for each depth) compared to González and Ràfols [69] (they needed to check a more complex quadratic equation at each depth).

Organization

In Section 4.2 we give the concrete security definitions of the simulation QA-NIZK arguments that we use in this contribution. In Section 4.3, we define a canonical QAP codification for Boolean Circuits. In Section 4.4 we recall the sub-schemes of Ag-

Construction	Lang	Signature Size	Assumption	Tightness
BFG [19]	PE	$(n_{\mathsf{PPE}} n_X, n_{\mathsf{PPE}} n_Y) + \ell_K$	Falsifiable	-
GM [74]	SAP	$(2,1) + \ell_K$	Non-falsifiable	$O(Q)$
Sec. 4.5.1. 4.6	QE	$(2n_s + 10d - 4, 6d + 4)$	Falsifiable	$O(Q)$
Sec. 4.5.1. 4.7	QE	$(2n_s + 10d + 8, 6d + 4)$	Falsifiable	$O(\log Q)$

Table 4.1: A comparison of our proposed SoK schemes in Sec. 4.5.1 with the USS argument for membership in linear spaces for in Section 4.6 and Section 4.7 respectively, with prior schemes. Lang means language. In the last column we show the tightness respect to the number of the queries Q for those constructions that are simulation sound. n_s denotes the secret input size in a boolean circuit, d the depth of the circuit, n_{PPE} is the number of pairing product equations (each multiplication gate in an arithmetic circuit can be encoded as a pairing product equation, in such case $n_{\mathsf{PPE}} = n$), n_X, n_Y are the number of variables in all the pairing product equations in $\mathbb{G}_1, \mathbb{G}_2$, respectively, ℓ_K is the size of the output of a hash function. PE: Pairing Equations, SAP: Square Arithmetic Equations, QE: Quadratic Equations.

gregated Proofs of Quadratic Equations and Aggregated Proofs of Linear Equations applied to our codification. In Section 4.5 we give our main construction, we present a framework of SE NIZK Argument for Boolean CircuitSat that uses three building blocks, two concrete instantiations of the framework in 4.5.1 and the SoK based on the SE NIZK framework in 4.5.2. In Section 4.6 we prove the USS argument of Kiltz and Wee is still secure with the promise problem. Same for Abe et al. USS argument in Section 4.7. Finally, in Section 4.8 we show how to improve the efficiency of the main construction with respect to a naive use of Groth-Sahai proofs.

4.2 Preliminaries

We gave the formal definition of QA-NIZK arguments inSection 2.7.2, now we define a QA-NIZK argument system that works with a tag space and give additional definitions of simulation. For witness-relations $\mathbf{R}_{gk} = \{\mathbf{R}_\rho\}_{\rho \in \mathsf{sup}(\mathcal{D}_{gk})}$ with parameters sampled from a distribution $\mathcal{D}_{gk}$ over associated parameter language $\mathcal{L}_{\mathsf{par}}$, a QA-NIZK argument system Π consists of tuple of PPT algorithms $\Pi = (\mathsf{K}_0, \mathsf{K}_1, \mathsf{P}, \mathsf{V}, \mathsf{S}_0, \mathsf{S}_1, \mathcal{E})$, defined as follows,

Parameter generator, $gk \leftarrow \mathsf{K}_0(1^\lambda)$**:** K_0 is a PPT algorithm that given 1^λ generates

group description gk.

CRS generator, $\mathsf{crs} \leftarrow \mathsf{K}_1(gk, \rho)$**:** K_1 is a PPT algorithm that given gk, samples string $\rho \leftarrow \mathcal{D}_{gk}$, and then uses gk, ρ and generates $(\mathsf{crs}, \mathsf{tr}_s, \mathsf{tr}_e)$, it also defines the tag space $\mathcal{T}$; finally output crs (that also contains parameter ρ) and stores the *simulation* trapdoor tr_s and *extraction* trapdoor tr_e as trapdoors.

Prover, $\pi \leftarrow \mathsf{P}(\mathsf{crs}, \boldsymbol{x}, \boldsymbol{w}, \tau)$**:** P is a PPT algorithm that, given $(\mathsf{crs}, \boldsymbol{x}, \boldsymbol{w}, \tau)$, where $(\boldsymbol{x}, \boldsymbol{w}) \in \mathbf{R}$ outputs an argument π with respect to a tag $\tau \in \mathcal{T}$. Otherwise, it outputs $\perp$.

Verifier, $\{0,1\} \leftarrow \mathsf{V}(\mathsf{crs}, \boldsymbol{x}, \pi, \tau)$**:** V is a PPT algorithm that, given $(\mathsf{crs}, \boldsymbol{x}, \pi, \tau)$, returns either 0 (reject) or 1 (accept).

Prover Simulator, $\pi \leftarrow \mathsf{S}(\mathsf{crs}, \boldsymbol{x}, \mathsf{tr}_s, \tau)$**:** S is a PPT algorithm that, given $(\mathsf{crs}, \boldsymbol{x}, \mathsf{tr}_s)$, outputs a simulated argument π with respect to a tag $\tau \in \mathcal{T}$.

Extractor, $\boldsymbol{w} \leftarrow \mathcal{E}(gk, \mathsf{crs}, \boldsymbol{x}, \pi, \tau, \mathsf{tr}_e)$**:** $\mathcal{E}$ is a PPT algorithm that, given $(\mathsf{crs}, \boldsymbol{x}, \pi, \tau, \mathsf{tr}_e)$ extracts the witness $\boldsymbol{w}$; where tr_e is the extraction trapdoor.

We require an argument QA-NIZK system Π to be *quasi-adaptive complete*, *computational quasi-adaptive sound* and *computational quasi-adaptive zero-knowledge*, as defined below.

Definition 25 (Quasi-Adaptive Completeness)**.** *A quasi-adaptive argument Π is perfectly complete for $\mathbf{R}_\rho$, if for all λ, all $(\boldsymbol{x}, \boldsymbol{w}) \in \mathbf{R}_\rho$, and all $\tau \in \mathcal{T}$,*

$$\Pr\left[\begin{array}{l} gk \leftarrow \mathsf{K}_0(1^\lambda),\ \rho \leftarrow \mathcal{D}_{gk}, \\ \mathsf{crs} \leftarrow \mathsf{K}_1(gk, \rho), \pi \leftarrow \mathsf{P}(\mathsf{crs}, \boldsymbol{x}, \boldsymbol{w}, \tau) \end{array} : \mathsf{V}(\mathsf{crs}, \boldsymbol{x}, \pi, \tau) = 1\right] = 1\ .$$

Definition 26 (Computational Quasi-Adaptive Soundness)**.** *A quasi-adaptive argument Π is computationally quasi-adaptive sound for $\mathbf{R}_\rho$, if for all λ, and for all non-uniform PPT $\mathcal{A}$,*

$$\Pr\left[\begin{array}{l} gk \leftarrow \mathsf{K}_0(1^\lambda),\ \rho \leftarrow \mathcal{D}_{gk}, \\ \mathsf{crs} \leftarrow \mathsf{K}_1(gk, \rho),\ (\boldsymbol{x}, \pi, \tau) \leftarrow \mathcal{A}(gk, \mathsf{crs}) \end{array} : \begin{array}{c} \mathsf{V}(\mathsf{crs}, \boldsymbol{x}, \pi, \tau) = 1 \wedge \\ (\boldsymbol{x}, \boldsymbol{w}) \notin \mathbf{R}_\rho \end{array}\right] \approx 0.$$

Definition 27 (Computational Quasi-Adaptive Zero-Knowledge)**.** *A quasi-adaptive argument Π is computationally quasi-adaptive zero-knowledge for $\mathbf{R}_\rho$, if for all λ, all*

$\tau \in \mathcal{T}$, and for all non-uniform PPT $\mathcal{A}$,

$$\Pr\begin{bmatrix} gk \leftarrow \mathsf{K}_0(1^\lambda), \rho \leftarrow \mathcal{D}_{gk}, \\ \mathsf{crs} \leftarrow \mathsf{K}_1(gk, \rho) : \\ \mathcal{A}^{\mathcal{O}_{real}(\boldsymbol{x},\boldsymbol{w})}(gk, \mathsf{crs}) = 1 \\ (\boldsymbol{x}, \boldsymbol{w}) \in \mathbf{R}_\rho \end{bmatrix} \approx \Pr\begin{bmatrix} gk \leftarrow \mathsf{K}_0(1^\lambda), \rho \leftarrow \mathcal{D}_{gk}, \\ (\mathsf{crs}, \mathsf{tr}_s, \mathsf{tr}_e) \leftarrow \mathsf{K}_1(gk, \rho) : \\ \mathcal{A}^{\mathcal{O}_{sim}(\boldsymbol{x},\boldsymbol{w})}(gk, \mathsf{crs}) = 1 \\ (\boldsymbol{x}, \boldsymbol{w}) \in \mathbf{R}_\rho \end{bmatrix}$$

where $\mathcal{O}_{real}(\boldsymbol{x}, \boldsymbol{w}, \tau)$ returns $\mathsf{P}(\mathsf{crs}, \boldsymbol{x}, \boldsymbol{w}, \tau)$ which emulates the actual prover for $(\boldsymbol{x}, \boldsymbol{w}) \in \mathbf{R}_\rho$, otherwise it outputs $\perp$; and $\mathcal{O}_{sim}(\boldsymbol{x}, \boldsymbol{w}, \tau)$ that returns $\mathsf{S}(\mathsf{crs}, \mathsf{tr}_s, \boldsymbol{x}, \tau)$ on input $(\boldsymbol{x}, \boldsymbol{w}) \in \mathbf{R}_\rho$ and $\perp$ if $(\boldsymbol{x}, \boldsymbol{w}) \notin \mathbf{R}_\rho$.

We also consider simulation soundness for our proofs, we take the next definition from Kiltz and Wee [92].

Definition 28 (Unbounded Simulation Adaptive Soundness)**.** *A quasi-adaptive argument Π is unbounded simulation adaptive sound for $\mathbf{R}_\rho$, if for all λ, and for all non-uniform PPT $\mathcal{A}$,*

$$\Pr\left[\begin{array}{l} gk \leftarrow \mathsf{K}_0(1^\lambda), \rho \leftarrow \mathcal{D}_{gk}, \\ (\mathsf{crs}, \mathsf{tr}) \leftarrow \mathsf{K}_1(gk, \rho); \\ (\boldsymbol{x}^*, \pi^*, \tau^*) \leftarrow \mathcal{A}^{O(\cdot)}(gk, \mathsf{crs}, \rho) \end{array} : \begin{array}{c} \tau^* \notin \mathcal{Q}_{tags} \ \wedge \ (\boldsymbol{x}^*, \boldsymbol{w}^*) \notin \mathbf{R}_\rho \\ \wedge \ \mathsf{V}(\mathsf{crs}, \boldsymbol{x}^*, \pi^*, \tau^*) = 1 \end{array}\right] \approx 0,$$

where $O(\boldsymbol{x})$ returns $\mathsf{S}(\mathsf{crs}, \mathsf{tr}, \boldsymbol{x}, \tau)$ and adds τ to the set $\mathcal{Q}_{tags}$.

Now, we define a variation of definition *BB simulation extractability* for QA-NIZKs that is satisfied by our schemes.

Definition 29 (Quasi-Adaptive BB Simulation Extractability)**.** *A non-interactive argument scheme Π is* quasi-adaptive black-box simulation-extractable for $\mathbf{R}_\rho$*, if for all λ, and for all non-uniform PPT $\mathcal{A}$, there exists a black-box extractor $\mathcal{E}$ such that,*

$$\Pr\left[\begin{array}{l} gk \leftarrow \mathsf{K}_0(1^\lambda), \ \rho \leftarrow \mathcal{D}_{gk}, \\ (\mathsf{crs}, \mathsf{tr}_s, \mathsf{tr}_e) \leftarrow \mathsf{K}_1(gk, \rho); \\ (\boldsymbol{x}^*, \pi^*, \tau^*) \leftarrow \mathcal{A}^{O(\cdot)}(gk, \mathsf{crs}, \rho), \\ \boldsymbol{w}^* \leftarrow \mathcal{E}(gk, \mathsf{crs}, \boldsymbol{x}^*, \pi^*, \tau^*, \mathsf{tr}_e) \end{array} : \begin{array}{c} \mathsf{V}(\mathsf{crs}, \boldsymbol{x}^*, \pi^*, \tau^*) = 1 \\ \wedge (\boldsymbol{x}^*, \boldsymbol{w}^*) \notin \mathbf{R}_\rho \wedge \ (\boldsymbol{x}^*, \pi^*) \notin \mathcal{Q} \\ \tau^* \notin \mathcal{Q}_{tags} \end{array}\right] \approx 0,$$

where $O(\boldsymbol{x}, \tau)$ returns $\mathsf{S}(\mathsf{crs}, \mathsf{tr}_s, \boldsymbol{x}, \tau)$ and adds $(\boldsymbol{x}, \pi)$ to the set of simulated proofs $\mathcal{Q}$ and τ to the set $\mathcal{Q}_{tags}$.

A key point about Def. 29 is that the extraction procedure is black-box and the extractor $\mathcal{E}$ works for all adversaries.

4.3 Canonical QAP for Boolean Circuits

Boolean circuits are acyclic directed graphs where the edges are called wires and the vertices are called gates. In this work, we consider boolean circuits $\phi : \{0,1\}^{n_0} \longrightarrow \{0,1\}^{\ell}$, with possibly some set of public inputs n_p and some set of private inputs n_s, $n_s + n_p = n_0$. Gates are arbitrary gates of fan-in two, (excluding non-interesting or trivial gate types). We denote m the total number of wires, n the number of boolean gates of the circuit. Usually, it would be the case that $m = n_p + n_s + n + 1$.

It is a well-known fact that, if $a, b \in \{0,1\}$, correct gate evaluation can be expressed as a quadratic equation over $\mathbb{Z}$. That, is for each gate type there exist values $\rho, \omega, \gamma, \varepsilon \in \mathbb{Z}$, such that if $a, b \in \{0,1\}$, and $c = \rho ab + \omega a + \gamma b + \varepsilon$, then $c \in \{0,1\}$ and c is the correct value of the gate evaluated at a, b. The constants satisfy that $\varepsilon \in \{0,1\}$, $\omega, \gamma \in \{0, \pm 1\}$, $\rho \in \{\pm 1\}$ for all gate types except for XOR and XNOR, where $\rho \in \{\pm 2\}$. More specifically, the important gate types are the following[3]

$\mathsf{AND}(a,b,c)$: $c = ab$.

$\mathsf{NAND}(a,b,c)$: $c = -ab + 1$

$\mathsf{OR}(a,b,c)$: $c = -ab + a + b$.

$\mathsf{NOR}(a,b,c)$: $c = ab - a - b + 1$

$\mathsf{XOR}(a,b,c)$: $c = -2ab + a + b$.

$\mathsf{XNOR}(x,y,x)$: $c = 2ab - a - b + 1$.

$\mathsf{G}_1(a,b,c) = (c = \overline{a} \wedge b) : c = -ab + b$.

$\mathsf{G}_2(a,b,c) = (c = \overline{\overline{a} \wedge \overline{b}}) : c = ab - b + 1$.

$\mathsf{G}_3(a,b,c) = (c = a \wedge \overline{b}) : c = -ab + a$.

$\mathsf{G}_4(a,b,c) = (c = \overline{a \wedge \overline{b}}) : c = ab - a + 1$.

Therefore, we can express a boolean circuit of m wires and n gates as a tuple $(\mathbf{F}, \mathbf{G}, \boldsymbol{\rho}, \boldsymbol{\omega}, \boldsymbol{\gamma}, \boldsymbol{\varepsilon})$, where $\mathbf{F} = (f_{ij}), \mathbf{G} = (g_{ij}) \in \{0,1\}^{m \times n}$ are the matrices which express the constraints for the left and right inputs for every gate and $\boldsymbol{\rho}, \boldsymbol{\omega}, \boldsymbol{\gamma}, \boldsymbol{\varepsilon} \in \mathbb{Z}^n$ are the vectors of constants associated to every gate. That is, if a_{j_L} (resp. a_{j_R}) is the left (resp. right) wire of gate j, then $a_{j_L} = \sum_{i=1}^{m} f_{ij} a_i$ (resp. $a_{j_R} = \sum_{i=1}^{m} g_{ij} a_i$), i.e. $\boldsymbol{f}_j = (f_{1j}, \ldots, f_{mj})$ is a unit vector which selects the left wire.

We show how to encode correct boolean circuit computation to prove that some pair $(\boldsymbol{x}, \boldsymbol{y})$ satisfies that $\phi(\boldsymbol{x}) = \boldsymbol{y}$ as a simple QAP. The vector $\boldsymbol{a}$ will denote the assignment of the circuit, so $(a_1, \ldots, a_{n_0}) = \boldsymbol{x}$ and $(a_{m-\ell+1}, \ldots, a_m) = \boldsymbol{y}$.

Theorem 22. *Let p be some prime number, $p > 2$. Let $\phi : \{0,1\}^{n_0} \to \{0,1\}^{\ell}$, be a circuit with n boolean gates, m wires, n_s secret inputs and n_p public inputs, defined by $(\mathbf{F}, \mathbf{G}, \boldsymbol{\rho}, \boldsymbol{\omega}, \boldsymbol{\gamma}, \boldsymbol{\varepsilon}) \in (\{0,1\}^{m \times n})^2 \times (\mathbb{Z}_p^n)^4$ as described above. Define the matrices $\mathbf{A}, \mathbf{B}, \mathbf{C} \in \mathbb{Z}_p^{m \times (n_s + n)}$ as*

$$\mathbf{A} = \left(\begin{array}{c|c} \mathbf{0}_{(n_p+1)\times n_s} & \boldsymbol{\gamma} \\ \mathbf{I}_{n_s} & \mathbf{F}' \\ \mathbf{0}_{n \times n_s} & \end{array}\right), \quad \mathbf{B} = \left(\begin{array}{c|c} \mathbf{0}_{(n_p+1)\times n_s} & \boldsymbol{\omega}' \\ \mathbf{I}_{n_s} & \mathbf{G} \\ \mathbf{0}_{n \times n_s} & \end{array}\right),$$

[3] As observed in [45], the last remaining 6 gate types depend mostly on one input and are not used.

$$\mathbf{C} = \left(\begin{array}{c|c} \mathbf{0}_{(n_p+1)\times n_s} & \boldsymbol{\varepsilon} - \boldsymbol{\gamma} \circ \boldsymbol{\omega}' \\ \mathbf{I}_{n_s} & \mathbf{0}_{n_s \times n} \\ \mathbf{0}_{n_s \times n} & \mathbf{I}_n \end{array}\right),$$

where $\mathbf{F}' = \mathbf{F} \begin{pmatrix} \rho_1 & & \\ & \ddots & \\ & & \rho_n \end{pmatrix}$, $\boldsymbol{\omega}' = \boldsymbol{\omega} \begin{pmatrix} \rho_1^{-1} & & \\ & \ddots & \\ & & \rho_n^{-1} \end{pmatrix}$.

Then, $\boldsymbol{a} = (1, a_1, \ldots, a_m) \in \mathbb{Z}_p^{m+1}$ *is a valid assignment of the circuit wires if and only if*

$$(\boldsymbol{a}^\top \mathbf{A}) \circ (\boldsymbol{a}^\top \mathbf{B}) - \boldsymbol{a}^\top \mathbf{C} = \mathbf{0}_{n_s+n}^\top, \tag{4.1}$$

which is equivalent to

$$(\boldsymbol{a}'^\top \underline{\mathbf{A}} + \hat{\boldsymbol{\gamma}}) \circ (\boldsymbol{a}'^\top \underline{\mathbf{B}} + \hat{\boldsymbol{\omega}}) - \boldsymbol{a}'^\top \underline{\mathbf{C}} + \hat{\boldsymbol{\varepsilon}} - \hat{\boldsymbol{\gamma}} \circ \hat{\boldsymbol{\omega}} = \mathbf{0}_{n_s+n}^\top, \tag{4.2}$$

where $\boldsymbol{a}' = (a_1, \ldots, a_m)$, $\underline{\mathbf{A}}(\underline{\mathbf{B}}, \underline{\mathbf{C}}) \in \mathbb{Z}_p^m$ *is the matrix* $\mathbf{A}$ *(resp.* $\mathbf{B}$*,* $\mathbf{C}$*) without the first row,* $\hat{\boldsymbol{\gamma}} = \begin{pmatrix} \mathbf{0}_{n_s} & \boldsymbol{\gamma} \end{pmatrix}$, $\hat{\boldsymbol{\omega}} = \begin{pmatrix} \mathbf{0}_{n_s} & \boldsymbol{\omega}' \end{pmatrix}$, $\hat{\boldsymbol{\varepsilon}} = \begin{pmatrix} \mathbf{0}_{n_s} & \boldsymbol{\varepsilon} \end{pmatrix} \in \mathbb{Z}_p^{n_s+n}$.

As we will see, the first n_s equations (corresponding to the first n_s columns) prove that the secret inputs of the circuit $a_{n_p+1}, \ldots, a_{n_p+n_s}$ are boolean and the last n columns correspond with correct gate evaluation equations for the wiring corresponding with matrices $\mathbf{F}, \mathbf{G}$.

Proof. We first observe that the matrices are well defined since $\rho_j^{-1} \mod p$ is always defined because $\rho_j \neq 0$ and its absolute value is at most 2 for the type of gates considered.

We then note that when restricted to $i = n_p + 2, \ldots, n_p + n_s + 1$, $j = 1, \ldots, n_s$, all three matrices A_{ij}, B_{ij}, C_{ij} are the identity matrix $\mathbf{I}_{n_s}$. Therefore, for any assignment $\boldsymbol{a}$ the first n_s columns of equation (4.2) expresses the fact that the secret input is boolean. If $\boldsymbol{A}_j$, $\boldsymbol{B}_j$, $\boldsymbol{C}_j$ are the jth column of the matrices $\mathbf{A}, \mathbf{B}, \mathbf{C}$, for $j = 1, \ldots, n_s$ we have $(\boldsymbol{a}^\top \boldsymbol{A}_j) \circ (\boldsymbol{a}^\top \boldsymbol{B}_j) - \boldsymbol{a}^\top \boldsymbol{C}_j = a_j a_j - a_j = a_j^2 - a_j = 0$ is satisfied if and only if $a_i \in \{0, 1\}$, for $i = n_p + 2, \ldots, n_p + n_s + 1 = n_0$.

We now look at the equations determined by the last n columns of equation (4.2). If $\boldsymbol{F}'_j$, $\boldsymbol{G}_j$ are the jth columns of $\mathbf{F}'$, $\mathbf{G}$, then, the $(n_s + j)$th equation in expression (4.2) can be rewritten as:

$$(\boldsymbol{a}'^\top \boldsymbol{F}'_j + \gamma_j) \circ (\boldsymbol{a}'^\top \boldsymbol{G}_j + \omega'_j) - \boldsymbol{a}'^\top_{n|} \boldsymbol{I}_{n,j} + \varepsilon_j - \gamma_j \omega'_j = 0 \tag{4.3}$$

where the vector $\boldsymbol{a}'_{n|}$ contains the last n components of $\boldsymbol{a}'$, i.e. $(a'_{n_{0+1}}, \ldots, a'_m)$.

The circuit ϕ specifies, for the jth circuit gate, a pair of indexes j_L, j_R which indicate the left and right wires. By definition of $\mathbf{F}' = (f'_{i,j}), \mathbf{G} = (g_{i,j})$, for $i =$

$1, \ldots, m$, $j = 1, \ldots, n$, the constants $f'_{i,j}$ and $g_{i,j}$ are 0 everywhere except for $f'_{j_L,j} = \rho_j$ and $g_{j_R,j} = 1$. Then, $\boldsymbol{a}'^{\top} \boldsymbol{F}'_j = \rho_j a_{j_L}$, $\boldsymbol{a}'^{\top} \boldsymbol{G}_j = a_{j_R}$ and $\boldsymbol{a}'^{\top} \boldsymbol{I}_{n_s+j} = a'_{n_0+j} + \varepsilon_j$.

Replacing these values in equation (4.3), we obtain:

$$(\rho_j a_{j_L} + \gamma_j)(a_{j_R} + \omega'_j) - a_{n_0+j} - \gamma_j \omega'_j + \varepsilon_j = 0. \tag{4.4}$$

Using the fact that, by definition, $\omega'_j = (\rho_j^{-1} \omega_j) \mod p$, we can rewrite this equation as:

$$a_{n_0+j} = \rho_j a_{j_L} a_{j_R} + a_{j_L} \omega_j + a_{j_R} \gamma_j + \varepsilon_j, \tag{4.5}$$

which by definition of the constants encodes the satisfiability of gate j. □ □

The reason why the encoding is very simple is because the matrices $\underline{\mathbf{B}}$ and $\underline{\mathbf{C}}$ are mostly independent of the gate type, and have only $0, 1$ entries, whereas the entries of matrix $\underline{\mathbf{A}}$ are $\{0, \pm 1, \pm 2\}$. Further, matrices $\underline{\mathbf{A}}, \underline{\mathbf{B}}, \underline{\mathbf{C}}$ are as sparse as possible (with $n + n_s$ non-zero entries) and all columns have exactly one non-zero value. This is optimal, since $n + n_s$ equations are required to prove that the secret input (of size n_s) is boolean and n gates are satisfied, this is why we call it *canonical*. For completeness, in the next Theorem, we express all the quadratic equations (boolean input and correct gate evaluation) as a divisibility relation following the usual "polynomial aggregation technique" of [61].

Theorem 23. *Let $\mathcal{R} \subset \mathbb{Z}_p$ be some fixed set of cardinal $n_s + n$ and let $\lambda_i(X)$ be the associated Lagrangian polynomials and $t(X)$ the polynomial whose roots are the elements of $\mathcal{R}$. Let $\phi : \{0,1\}^{n_0} \to \{0,1\}$, be any circuit with n boolean gates, m wires, n_s secret inputs. There exist some polynomials $\{u_i(X); v_i(X); w_i(X)\}_{i=0}^m\}$ such that $\boldsymbol{a} = (a_0, a_1, \ldots, a_m)$, with $a_0 = 1$, is a valid assignment to the circuit wires if and only if*

$$(\textstyle\sum_{i=0}^m a_i u_i(X)) \cdot (\sum_{i=0}^m a_i v_i(X)) - (\sum_{i=0}^m a_i w_i(X)) \equiv 0 \mod t(X). \tag{4.6}$$

Proof. Numerate the rows of matrices $\mathbf{A}, \mathbf{B}, \mathbf{C}$ from $0, \ldots, m$. For $i \in [1, m]$ set

$$u_i(X) = \sum_{j=1}^{n_s+n} A_{ij} \lambda_j(X), \quad v_i(X) = \sum_{j=1}^{n_s+n} B_{ij} \lambda_j(X),$$

$$w_i(X) = \sum_{j=1}^{n_s+n} C_{ij} \lambda_j(X).$$

Further, define

$$u_0(X) = \sum_{j=n_s+1}^{n_s+n} \gamma_j \lambda_j(X),\ v_0(X) = \sum_{j=n_s+1}^{n_s+n} \omega'_j \lambda_j(X),$$

$$w_0(X) = \sum_{j=n_s+1}^{n_s+n} (\varepsilon_j - \gamma_j \omega'_j) \lambda_j(X).$$

Finally, if we let

$$u(X) = \sum_{i=1}^{m} a_i u_i(X) + u_0(X),\ v(X) = \sum_{i=1}^{m} a_i v_i(X) + v_0(X),$$

$$w(X) = \sum_{i=1}^{m} a_i w_i(X) + w_0(X)$$

it holds that $\boldsymbol{a}$ satisfies equation (4.6) if and only if $t(X)$ divides $p(X) = u(X)v(X) - w(X)$. This is a direct consequence of the definition of the polynomials and Theorem 22. □

The simple form of matrices $\underline{\mathbf{A}}$, $\underline{\mathbf{B}}$ and $\underline{\mathbf{C}}$ translates into very simple expressions for $\{u_i(X), v_i(X), w_i(X)\}_{i=1}^m$. For instance, the $v_i(X)$'s can be computed as a sum of Lagrangian polynomials, without any exponentiation. Similarly, $u_0(X)$ has a very simple expression as $\gamma_j \in \{\pm 1\}$, $v_0(X)$ is slightly more complicated (the coefficients take values in $\{\pm 1, \pm 2^{-1} \mod p\}$) and so is $w_0(X)$.

4.3.1 Circuit Slicing

As we explain in Section 4.4 following González and Ràfols [69], the prover aggregates the proofs that all the gates are satisfied at level i (a set of quadratic equations), on the one hand, and all the linear equations that show "correct wiring", i.e. that the outputs at level at most $i-1$ are correctly transferred to inputs at level i, on the other hand.

For this, as in [69], we *slice* a boolean circuit in layers according to the depth of each gate. That is, we index the gates of ϕ by a pair (i, j), where i denotes the gate depth and j is some index in the range $1, \ldots, n_i$, where n_i is the number of gates at level i, and we write down, for each level, the set of quadratic and affine constraints that need to be satisfied. In the following, $\phi : \{0,1\}^n \to \{0,1\}$ and we call d the depth of the circuit.

We define a witness for Boolean CircuitSat as a tuple $(\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{c})$ which is, respectively, a valid assignment to the left, right and output wires of ϕ when each boolean gate is written as a multiplicative constraint, as explained below. To "slice" the circuit, each of these vectors is written as a concatenation of vectors, one for each multiplicative depth. That is, $\boldsymbol{a} = (\boldsymbol{a}_1, \ldots, \boldsymbol{a}_d)$, $\boldsymbol{b} = (\boldsymbol{b}_1, \ldots, \boldsymbol{b}_d)$ and $\boldsymbol{c} = (\boldsymbol{c}_0, \boldsymbol{c}_1, \ldots, \boldsymbol{c}_d)$ and $\boldsymbol{y}_i = (y_{i,1}, \ldots, y_{i,n_i})$ for all $\boldsymbol{y} \in \{\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{c}\}$. Gate (i, j) is described by constants $\rho_{i,j}, \omega_{i,j}, \gamma_{i,j}, \varepsilon_{i,j}$, and $\boldsymbol{\rho}_i, \boldsymbol{\omega}_i, \boldsymbol{\gamma}_i, \boldsymbol{\varepsilon}_i \in \mathbb{Z}^{n_i}$ are the vectors of constants associated to the n_i gates at level i.

A valid assignment should give $a_{i,j}$, $b_{i,j}$ and $c_{i,j}$ the values that prove correct evaluation of gate (i, j), namely, $c_{i,j} = (a_{i,j} + \gamma_{i,j})(b_{i,j} + \omega'_{i,j}) - (\gamma_{i,j}\omega'_{i,j} + \varepsilon_{i,j})$ that are consistent with some boolean input $c_{0,1}, \ldots, c_{0,n}$ are some boolean values that represent a satisfying input.

We differ from [69] in that we take advantage of our work in the previous section characterizing Boolean CircuitSat as a QAP, therefore, the set of equations that need to be satisfied is simpler.

Lemma 24 breaks down CircuitSat in different items which reflect the different building blocks used by [69] and also our work. The input vector $\boldsymbol{x}$ (which corresponds to $\boldsymbol{c}_0$) is divided in two parts, the first n_p components being the public input $\boldsymbol{x}_p$ and the rest is the secret input $\boldsymbol{x}_s$ of length n_s. The main achievement of [69] is to do two aggregated proofs of all the constraints at the same depth with just two constant size proofs, one for the multiplicative and the other for the linear constraints. Therefore, items c) (resp. d)) require that for each $i = 1, \ldots, d$, a set of quadratic (resp. linear) equations holds. In the next two subsections (Section 4.4.1,4.4.2) we sketch the aggregated proofs of the sets of equations described in c) and d).

Lemma 24. *Let $\phi : \{0,1\}^{n_0} \to \{0,1\}$, be a circuit with m boolean gates. Then, for any public input $\boldsymbol{x}_p \in \{0,1\}^{n_p}$, $(\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{c})$ is a valid input for satisfiability of $\phi(\boldsymbol{x}_p, \cdot)$ if and only if:*

a) $(c_{0,1}, \ldots, c_{0,n_p}) = (\boldsymbol{x}_p)$.

b) Boolean secret input: $(c_{0,n_p+1}, \ldots, c_{0,n}) = (\boldsymbol{x}_s) \in \{0,1\}^{n_s}$.

c) Correct gate evaluation at level i, for $i = 1, \ldots, d$ there exists a vector of constants $\boldsymbol{k}_i$ such that:

$$\boldsymbol{c}_i = \boldsymbol{k}_i + \boldsymbol{a}_i \circ \boldsymbol{b}_i, \qquad j = 1, \ldots, n_i.$$

d) Correct "wiring" (linear constraints) at level i: there exist some matrices $\widetilde{\mathbf{F}}_i, \widetilde{\mathbf{G}}_i$ such that $\boldsymbol{a}_i = \widetilde{\mathbf{F}}_i \boldsymbol{c}_{|i-1}$ and $\boldsymbol{b}_i = \widetilde{\mathbf{G}}_i \boldsymbol{c}_{|i-1}$, where $\boldsymbol{c}_{|i-1}^\top = (1, \boldsymbol{c}_0^\top, \ldots, \boldsymbol{c}_{i-1}^\top)$.

e) Correct output: $c_{d,1} = 1$.

The matrices $\widetilde{\mathbf{F}}_i, \widetilde{\mathbf{G}}_i$ and the constants $k_{i,j}$ are defined naturally from the description in Theorem 22, namely:

- $\widetilde{\mathbf{F}}_i = \begin{pmatrix} \boldsymbol{\gamma}_i & \mathbf{F}'_i \end{pmatrix}$ where $\mathbf{F}'_i = \begin{pmatrix} \rho_{i,1} & & \\ & \ddots & \\ & & \rho_{i,n_i} \end{pmatrix} \mathbf{F}_i^\top$, where if $\mathbf{F}$ is the matrix given in the circuit description, $\mathbf{F}_i \in \mathbb{Z}_p^{(\sum_{j=0}^{i-1} n_j) \times n_i}$ is the matrix that describes the left wires of gates at level i.

- $\widetilde{\mathbf{G}}_i = \begin{pmatrix} \boldsymbol{\omega}'_i & \mathbf{G}_i^\top \end{pmatrix}$, where $\boldsymbol{\omega}'_i = \boldsymbol{\omega}_i \begin{pmatrix} \rho_{i1}^{-1} & & \\ & \ddots & \\ & & \rho_{in_i}^{-1} \end{pmatrix}$, and if $\mathbf{G}$ is the matrix given in the circuit description, $\mathbf{G}_i \in \mathbb{Z}_p^{(\sum_{j=0}^{i-1} n_j) \times n_i}$ is the matrix that describes the right wires of gates at level i.

- $k_{i,j} = \varepsilon_{i,j} - \gamma_{i,j}\omega_{i,j}\rho_{i,j}^{-1}$.

4.4 GR19 Argument for Boolean CircuitSat

In Section 4.3 we have described Boolean CircuitSat as d sets of linear and quadratic constraints, where d is the depth of the circuit. In this section, we revisit the results of González and Ràfols [69] but using the simpler characterization of Boolean CircuitSat given in 4.3.1. Recall that [69] shows how to give a constant size proof for each of these sets of constraints while basing security on falsifiable assumptions provided a witness of satisfiability is known for the "previous" sets of equations (ordering the sets of equations in the natural order from the input).

4.4.1 Aggregated Proofs of Quadratic Equations

We now describe the construction proposed in González and Ràfols [69] to prove correct gate evaluation at level i, for $i = 1, \ldots, d-1$, i.e. a proof that $c_{i,j} = k_{i,j} - a_{i,j}b_{i,j}$, for all $j = 1, \ldots, n_i$. It consists, for $k = 1, 2$, of a Groth-Sahai NIZK Proof that some

secret values $[L_{i,k}]_1, [R_{i,k}]_2, [O_{i,k}]_1, [O^*_{i,k}]_2, [H_{i,k}]_1$ satisfy the following relation[4]:

$$e([K_{i,k}]_1, [1]_2) + e([L_{i,k}]_1, [R_{i,k}]_2) - e([O_{i,k}]_1, [1]_2) = e([H_{i,k}]_1, [t_k]_2), \quad (4.7)$$
$$e([O_{i,k}]_1, [1]_2) = e([1]_1, [O^*_{i,k}]_2). \quad (4.8)$$

where if $t(X) = \prod_{r\in\mathcal{R}}(X - r)$, $t_k = t(s_k)$ and $\lambda_i(X) = \prod_{j\in\mathcal{R}\setminus\{r_i\}} \frac{(X - r_j)}{(r_i - r_j)}$ is the ith Lagrangian polynomial associated to $\mathcal{R}$, a set of $W = \max_{i=1,\ldots,d} n_i$ points used for interpolation, then

$$L_{i,k} = \sum a_j \lambda_j(s_k), \quad R_{i,k} = \sum b_j \lambda_j(s_k), \quad C_{i,k} = \sum c_j \lambda_j(s_k), \quad H_{i,k} = h_i(s_k),$$

where s_1, s_2 are random secret points specified in the crs,

$$h_i(X) = (1 - (\sum a_j \lambda_j(X))(\sum b_j \lambda_j(X)) - \sum c_j \lambda_j(X))/t(X)$$

and $[K_{i,k}]_1 = \sum k_{i,j} \lambda_j(s_k)$. Alternatively, for each n_i we define

$$\boldsymbol{\Lambda}_{n_i} = \begin{pmatrix} \lambda_1(s_1) & \ldots & \lambda_{n_i}(s_1) \\ \lambda_1(s_2) & \ldots & \lambda_{n_i}(s_2) \end{pmatrix},$$

$$[\boldsymbol{L}_i]_1 = [\boldsymbol{\Lambda}_{n_i} \boldsymbol{a}_i]_1, [\boldsymbol{R}_i]_2 = [\boldsymbol{\Lambda}_{n_i} \boldsymbol{b}_i]_2, [\boldsymbol{O}_i]_1 = [\boldsymbol{\Lambda}_{n_i} \boldsymbol{c}_i]_1,$$

and $\boldsymbol{\Lambda}$ is called Lagrangian Pedersen commitment in [69].

To the reader familiar with the literature, it is obvious that equation (4.7) uses zk-SNARK techniques originally appeared in [61] (what we could call "polynomial aggregation") for proving many quadratic equations simultaneously. What is new in [69], is the security analysis, which avoids non-falsifiable assumptions.

GS proofs are necessary for zero-knowledge because $\boldsymbol{L}_i, \boldsymbol{R}_i, \boldsymbol{O}_i$ need to be deterministic for the proof to work. The authors of [69] use this proof as a building block in a larger proof, and for this we prove the following:
"if $(\boldsymbol{a}_i, \boldsymbol{b}_i)$ are valid openings of $[L_{i,k}]_1, [R_{i,k}]_2$ for $k = 1, 2$ then $\boldsymbol{k}_i + \boldsymbol{a}_i \circ \boldsymbol{b}_i$ is a valid opening of $O_{i,k}$."

Formally, we define the languages

$$\mathcal{L}^{\mathsf{quad}}_{\mathsf{YES}} = \left\{ \begin{array}{c} (\boldsymbol{a}, \boldsymbol{b}, [\boldsymbol{L}]_1, [\boldsymbol{R}]_2, [\boldsymbol{O}]_1) : \boldsymbol{k} + \boldsymbol{a} \circ \boldsymbol{b} = \boldsymbol{c}, \\ [\boldsymbol{L}]_1 = [\boldsymbol{\Lambda}]_1 \boldsymbol{a}, [\boldsymbol{R}]_2 = [\boldsymbol{\Lambda}]_2 \boldsymbol{b}, [\boldsymbol{O}]_1 = [\boldsymbol{\Lambda}]_1 \boldsymbol{c} \end{array} \right\}$$

[4]The second equation is added to have the element $O_{i,k}$ in both groups $\mathbb{G}_1, \mathbb{G}_2$. This will allow us to use simple QA-NIZK proofs of membership in linear spaces in $\mathbb{G}_1$ and $\mathbb{G}_2$ for the linear constraints, instead of using proofs of membership in bilateral spaces (spaces with parts in $\mathbb{G}_1$ and in $\mathbb{G}_2$.)

$$\mathcal{L}_{\mathsf{NO}}^{\mathsf{quad}} = \left\{ \begin{array}{c} (\boldsymbol{a}, \boldsymbol{b}, [\boldsymbol{L}]_1, [\boldsymbol{R}]_2, [\boldsymbol{O}]_1) : \boldsymbol{k} + \boldsymbol{a} \circ \boldsymbol{b} = \boldsymbol{c}, \\ [\boldsymbol{L}]_1 = [\boldsymbol{\Lambda}]_1 \boldsymbol{a}, [\boldsymbol{R}]_2 = [\boldsymbol{\Lambda}]_2 \boldsymbol{b}, [\boldsymbol{O}]_1 \neq [\boldsymbol{\Lambda}]_1 \boldsymbol{c} \end{array} \right\}.$$

The argument consists of giving some values $\boldsymbol{H}, \boldsymbol{O}^*$ chosen by the prover which satisfies equations (4.7) for $\boldsymbol{L}, \boldsymbol{R}, \boldsymbol{O}$. *Completeness* holds for $\mathcal{L}_{\mathsf{YES}}^{\mathsf{quad}}$ and *soundness* for $\mathcal{L}_{\mathsf{NO}}^{\mathsf{quad}}$ under the $(\mathcal{R}, m)$-Rational Strong Diffie-Hellman assumption ([69]). When (4.7) are proven with GS proofs, the authors argue that *zero-knowledge* also holds.

Note that the fact $[\boldsymbol{L}]_1 = [\boldsymbol{\Lambda}]_1 \boldsymbol{a}$, or $[\boldsymbol{R}]_2 = [\boldsymbol{\Lambda}]_2 \boldsymbol{b}$ is never checked by the verifier, this is the promise. The argument does not give any guarantee when this does not hold.

4.4.2 Aggregated Proofs of Linear Equations

In this section we explain the technique used in González and Ràfols [69] to prove correct "wiring" at level i, for $i = 1, \ldots, d-1$, i.e. an aggregated proof for linear constraints applied to the equations defined in 4.3.1. As we have seen in Lemma 24, we can express linear constraints at level i as:

$$\boldsymbol{a}_i = \widetilde{\mathbf{F}}_i \boldsymbol{c}_{|i-1}, \quad \boldsymbol{b}_i = \widetilde{\mathbf{G}}_i \boldsymbol{c}_{|i-1} \text{ for all } i = 1, \ldots, d. \tag{4.9}$$

Then at level i left and right constraints can be expressed, respectively as:

$$\begin{pmatrix} \boldsymbol{O}_{|i-1} \\ \boldsymbol{L}_i \end{pmatrix} = \begin{pmatrix} \mathbf{C}_i \\ \mathbf{N}_i^L \end{pmatrix} \boldsymbol{c}_{|i-1}, \quad \begin{pmatrix} \boldsymbol{O}_{|i-1} \\ \boldsymbol{R}_i \end{pmatrix} = \begin{pmatrix} \mathbf{C}_i \\ \mathbf{N}_i^R \end{pmatrix} \boldsymbol{c}_{|i-1} \tag{4.10}$$

where $\mathbf{C}_i = \begin{pmatrix} \mathbf{I} & \mathbf{0} & \ldots & \mathbf{0} \\ \mathbf{0} & \boldsymbol{\Lambda}_{n_1} & \ldots & \mathbf{0} \\ & & \ddots & \\ \mathbf{0} & \mathbf{0} & \ldots & \boldsymbol{\Lambda}_{n_{i-1}} \end{pmatrix}$, $\mathbf{N}_i^L = \boldsymbol{\Lambda}_{n_i} \widetilde{\mathbf{F}}_i$, $\mathbf{N}_i^R = \boldsymbol{\Lambda}_{n_i} \widetilde{\mathbf{G}}_i$ and $\boldsymbol{\Lambda}_{n_i}$ is the matrix of the Lagrangian Pedersen commitment key defined in the last section, and $\boldsymbol{O}_0$ is just the input of the circuit.

To make the argument zero-knowledge, the prover does never give $\boldsymbol{O}_i, \boldsymbol{L}_i$ or $\boldsymbol{R}_i$ in the clear, but rather, for $k = 1, 2$ and any $i \in [d]$, it gives GS commitments $[\boldsymbol{z}]_1$ to the input (i.e. to all components of $\boldsymbol{O}_0 = \boldsymbol{c}_0$), to the vector $\boldsymbol{O}_i$ as $[\boldsymbol{z}_{O,i}]_1$, to the vector $\boldsymbol{L}_i$ as $[\boldsymbol{z}_{L,i}]_1$ and to the vector $\boldsymbol{R}_i$ as $[\boldsymbol{z}_{R,i}]_2$ (a part from other GS commitments necessary for the quadratic proof). The matrices which define the linear relation between committed values are defined from $\mathbf{C}_i$, $\mathbf{N}_i^L = \boldsymbol{\Lambda}_{n_i} \widetilde{\mathbf{F}}_i$, $\mathbf{N}_i^R = \boldsymbol{\Lambda}_{n_i} \widetilde{\mathbf{G}}_i$ adding columns and rows to accommodate for the GS commitment keys in the relevant groups (see full details in [69]). We denote the matrix that define the left (resp. right) constraints until level $i-1$

as $\mathbf{M}_i^L$ (resp. $\mathbf{M}_i^R$), that is:

$$\mathbf{M}_i^L = \begin{pmatrix} \mathbf{C}_i \\ \mathbf{N}_1^L \\ \vdots \\ \mathbf{N}_{i-1}^L \end{pmatrix}, \qquad \mathbf{M}_i^R = \begin{pmatrix} \mathbf{C}_i \\ \mathbf{N}_1^R \\ \vdots \\ \mathbf{N}_{i-1}^R \end{pmatrix}.$$

González and Ràfols prove that the QA-NIZK argument of Kiltz and Wee [92] (with standard soundness) for membership in linear spaces for non-witness samplable distributions is an argument for the following promise problem parametrized by matrices $\mathbf{M}, \mathbf{N}$:

$$\mathcal{L}_{\text{YES}}^{\mathsf{Lin}} = \left\{ (\boldsymbol{w}, [\boldsymbol{x}]_1, [\boldsymbol{y}]_1) : \begin{array}{c} [\boldsymbol{x}]_1 = [\mathbf{M}]_1 \boldsymbol{w} \text{ and} \\ [\boldsymbol{y}]_1 = [\mathbf{N}]_1 \boldsymbol{w} \end{array} \right\}$$

$$\mathcal{L}_{\text{NO}}^{\mathsf{Lin}} = \left\{ (\boldsymbol{w}, [\boldsymbol{x}]_1, [\boldsymbol{y}]_1) : \begin{array}{c} [\boldsymbol{x}]_1 = [\mathbf{M}]_1 \boldsymbol{w} \text{ and} \\ [\boldsymbol{y}]_1 \neq [\mathbf{N}]_1 \boldsymbol{w} \end{array} \right\}.$$

If we use this construction for matrices $\mathbf{M}_i^L$ and $\mathbf{N}_i^L$ (similarly for right side), this argument can be used to prove that, if we can extract $\boldsymbol{c}_{|i-1}$, then we can extract an opening $\boldsymbol{a}_i$ of $\boldsymbol{L}_i$ which is in the correct linear relation with $\boldsymbol{c}_{|i-1}$. In other words, this proves that if all the linear constraints are satisfied until level $i-1$, they must be satisfied until level i.

The authors prove completeness of the argument for statements in $\mathcal{L}_{\text{YES}}^{\mathsf{Lin}}$ and soundness for $\mathcal{L}_{\text{NO}}^{\mathsf{Lin}}$ under $\mathcal{M}_L^\top$-MDDH, $\mathcal{M}_R^\top$-MDDH and KerMDH assumption, where $\mathcal{M}_L$ (resp. $\mathcal{M}_R$) is the distribution of matrices $\mathbf{M}_i^L$ (resp. $\mathbf{M}_i^R$) described above[5].

Efficiency Improvements

We note that for simplicity, we have explained the result of [69] as proving a linear system of constraints for each level and each side (left or right), but in fact a single QA-NIZK argument for bilateral spaces for non-witness samplable distributions [67] is used in [69] to gain efficiency (the proof requires then only 2 elements in $\mathbb{G}_1$ and $\mathbb{G}_2$ instead of $O(d)$ elements).

[5] An important point is that these MDDH assumptions can be reduced to a decisional assumption in bilinear groups which does not depend on the circuit. In fact, $\mathbf{M}_i^L$ only depends on $n, n_1, \ldots, n_s$, and the assumption can be reduced to a decisional assumption which only depends on $\boldsymbol{\Lambda}$ and the GS commitment key.

4.5 SE NIZK Argument for Boolean CircuitSat

We present our Quasi-Adaptive argument for Boolean CircuitSat for the language defined as

$$\mathcal{L}_\phi = \left\{ \ (\boldsymbol{x}_p) \ \middle| \ \exists \boldsymbol{x}_s \in \{0,1\}^{n_s} \text{ s.t. } \phi(\boldsymbol{x}_p, \boldsymbol{x}_s) = 1 \ \right\}.$$

As consequence of Lemma 24 the language $\mathcal{L}_{\phi,ck}$ can be equivalently defined as

$$\mathcal{L}_\phi = \left\{ \ (\boldsymbol{x}_p) \ \middle| \ \begin{array}{l} \exists \boldsymbol{x}_s \text{ s.t. } \boldsymbol{x}_s \circ (\boldsymbol{x}_s - \mathbf{1}) = \mathbf{0}; \\ \boldsymbol{c}_0 := (\boldsymbol{x}_p, \boldsymbol{x}_s); \\ \forall i \in [d], \exists \boldsymbol{a}_i, \boldsymbol{b}_i, \boldsymbol{c}_i \in \mathbb{Z}_p^{n_i} \text{ s.t. } ; \\ \boldsymbol{a}_i = \widetilde{\mathbf{F}}_i \boldsymbol{c}_{|i-1}, \boldsymbol{b}_i = \widetilde{\mathbf{G}}_i \boldsymbol{c}_{|i-1} \in \mathbb{Z}_p^{n_i}, \\ \boldsymbol{k}_i + \boldsymbol{a}_i \circ \boldsymbol{b}_i = \boldsymbol{c}_i. \end{array} \right\}.$$

In the following Π_Q denotes the argument for Quadratic Equations described in Section 4.4.1, Π_L a tag-based USS membership argument for linear spaces that can be either the one presented in Section 4.6 or the one presented in Section 4.7 and Input an argument to prove that some BB extractable commitments to integers open to binary values.

$\underline{\mathsf{K}_0(\lambda, W, \mathcal{R})}$: On input some set $\mathcal{R} \subset \mathbb{Z}_p$ of cardinal W, choose a bilinear group gk and output (gk, W).

$\underline{\mathcal{D}_{gk,W,\mathcal{R}}}$: Pick commitment keys $(\mathsf{ck}_1, \mathsf{ck}_2) = ([\boldsymbol{\Lambda}]_1, [\boldsymbol{\Lambda}]_2)$ that are the Lagrangian Pedersen commitment keys associated to $\mathcal{R}$. Output $(ck_1, ck_2, \mathsf{crs}_{\mathsf{GS}})$.

$\underline{\mathsf{K}_1\,(gk, \phi)}$: Given $(ck_1, ck_2, \mathsf{crs}_{\mathsf{GS}}) \leftarrow \mathcal{D}_{gk,W}$ and $\phi : \{0,1\}^n \to \{0,1\}$ of maximum width W. For each $i \in [d]$ define matrices $[\mathbf{M}_i^L]_1$, $[\mathbf{M}_i^R]_2$, $[\mathbf{N}_i^L]_1$, $[\mathbf{M}_i^R]_2$ as explained in Section 4.4.2. Let $\mathsf{crs}_{\mathsf{Input}}$ the crs of the argument Input for a vector of size n_s is binary. Let crs_Q the crs of Π_Q for proving correct evaluation of (at most) W gates. For each $i \in [d]$, let $\mathsf{crs}_{L,i}^L$ ($\mathsf{crs}_{L,i}^R$) the crs for the USS argument of linear knowledge transfer Π_L of left (right) wires at depth i. Let $\mathsf{crs}_L = \left\{\mathsf{crs}_{L,i}^L, \mathsf{crs}_{L,i}^R\right\}_{i\in[d]}$ and $\mathsf{tr}_L = \left\{\mathsf{tr}_{L,i}^L, \mathsf{tr}_{L,i}^R\right\}_{i\in[d]}$ be the crs and the trapdoors of the Π_L arguments of left (right) wires at depth i, where crs_L includes the tag space $\mathcal{T}$.

Output $\mathsf{crs} = (\mathsf{ck}_1, \mathsf{ck}_2, \mathsf{crs}_{\mathsf{GS}}, \mathsf{crs}_{\mathsf{Input}}, \mathsf{crs}_Q, \mathsf{crs}_L)$, $\mathsf{tr} = \mathsf{tr}_L$.

$\underline{\mathsf{P}\,(\mathsf{crs}, \boldsymbol{x}_p, \boldsymbol{x}_s, \boldsymbol{r}, \boldsymbol{a}, \boldsymbol{b}, \boldsymbol{c}, \tau)}$: Computes the commitment of the secret input $[\boldsymbol{z}]_1 = \mathsf{com}_{\mathsf{ck}_1,\mathsf{ck}_2}(\boldsymbol{x}_s, \boldsymbol{r})$ and constructs the proof Input for $[\boldsymbol{z}]_1$. For each $i \in [d]$ compute Lagrangian Pedersen commitments to the output, left and right wires $[\boldsymbol{O}_i]_{1,2}, [\boldsymbol{L}_i]_1, [\boldsymbol{R}_i]_{1,2}$,

give a GS proof $\Pi_{Q,i}$ that they satisfy the equations (4.7) and let $[\boldsymbol{z}_{O,i,k}]_1$, $[\boldsymbol{z}^*_{O,i,k}]_2$, $[\boldsymbol{z}_{L,i,k}]_1$, $[\boldsymbol{z}_{R,i,k}]_2$, $[\boldsymbol{z}^*_{R,i,k}]_1$ the correspondent GS commitments to $\boldsymbol{O}, \boldsymbol{L}, \boldsymbol{R}$, for $k = 1, 2$. Compute proofs $\Pi_{L,i}$ of correct wiring, $\Pi_{L,0}$ that the opening of $[\boldsymbol{z}]_1$ is correctly assigned to $[\boldsymbol{z}_{O,0}]_1$ and that the openings of $[\boldsymbol{z}_R]_2, [\boldsymbol{z}^*_R]_1$ and $[\boldsymbol{z}_O]_1, [\boldsymbol{z}^*_O]_2$ are equal respectively.

The proof is

$$\pi = ([\boldsymbol{z}]_1, \mathsf{Input}, [\boldsymbol{z}_O]_1, [\boldsymbol{z}_L]_1, [\boldsymbol{z}^*_O]_2, [\boldsymbol{z}_R]_2, [\boldsymbol{z}^*_R]_1 \Pi_L, \Pi_{L,0}, \Pi_Q).$$

$\mathsf{V}(\mathsf{crs}, \boldsymbol{x}_p, \pi, \tau)$: Verify all the proofs in π with the corresponding verification algorithms $\mathsf{V}_{\mathsf{Input}}$, V_{Π_L} (which uses τ) and check the GS proofs of equations (4.7).

$\mathsf{S}(\mathsf{crs}, \mathsf{tr}, \boldsymbol{x}_p, \tau)$: Extend the input with zeros, $\boldsymbol{x} = (\boldsymbol{x}_p, 0, \ldots, 0)$ and evaluate the circuit honestly with this input to obtain the corresponding $\boldsymbol{a}_i, \boldsymbol{b}_i, \boldsymbol{c}_i$ for each $i = 1, \ldots, d$. Change the last gate values, i.e. the right and left values of the last gate at level d, $a_{d,1}$, $b_{d,1}$, and $c_{d,1}$ consequently, to have an assignment that satisfies the equation of this gate. Compute the commitment $[\boldsymbol{z}]_1 = \mathsf{com}_{\mathsf{ck}_1,\mathsf{ck}_2}(\boldsymbol{0}, \boldsymbol{r})$, honest proofs Input and $\Pi_{Q,i}$, and commitments $[\boldsymbol{z}_{O,i,k}]_1$, $[\boldsymbol{z}_{L,i,k}]_1$, $[\boldsymbol{z}^*_{O,i,k}]_2$, $[\boldsymbol{z}_{R,i,k}]_2$, $\left[\boldsymbol{z}^*_{R,i,k}\right]_1$ for each $i = 1, \ldots, d$. Run the simulator S_{Π_L} to obtain d simulated $\Pi^S_{L,i}, \Pi^S_{R,i}$ together with $\Pi^S_{L,0}$. Output $\pi^S = ([\boldsymbol{z}]_1, \mathsf{Input}, [\boldsymbol{z}_O]_1, [\boldsymbol{z}_L]_1, [\boldsymbol{z}_R]_2, [\boldsymbol{z}^*_O]_2, \Pi^S_L, \Pi^S_{L,0}, \Pi_Q)$.

Completeness is direct from the completeness of the respective subarguments.

Computational Zero-Knowledge follows from witness sampleability of the GS commitment keys and the fact that in GS proofs, commitments are dual mode commitments. This means that the common reference string can be generated in an indistinguishable way so that all commitments are perfectly hiding. In particular, in this setting, the distributions of real and simulated proofs are indistinguishable.

Unbounded Simulation Extractable Adaptive Soundness is proved in the following theorem.

Theorem 25. *If $\mathcal{A}$ is an adaptive adversary against the Unbounded Simulation BB Extractability Soundness of the Boolean CircuitSat argument described in Section 4.5 that makes at most Q queries to S, then there exist PPT adversaries $\mathcal{B}_1$, $\mathcal{B}_2$, $\mathcal{B}_3$ against the BB Extractable Soundness of Input, the unbounded simulation soundness of Π_L argument and the soundness of Π_Q argument, respectively, such that*

$$\mathsf{Adv}_{\mathit{USS}}(\mathcal{A}) \leq \mathsf{Adv}_{\mathit{ES}\text{-}\mathsf{Input}}(\mathcal{B}_1) + d\mathsf{Adv}_{\mathit{USS}\text{-}\Pi_L}(\mathcal{B}_2) + 2d\mathsf{Adv}_{\mathit{Sound}\text{-}\Pi_Q}(\mathcal{B}_3).$$

Proof. (sketch) The simulator algorithm generates honestly the Input and Π_Q arguments and an adversary sees only simulated proofs of the linear argument Π_L. Therefore, an adversary that creates a new proof for an invalid statement breaks either the knowledge soundness of the Input, the soundness of the Π_Q arguments, or the USS of the linear arguments Π_L. □

4.5.1 Concrete SE QA-NIZK for Boolean CircuitSat

For the scheme described above, one can take as Input, and Π_Q the same building blocks as [69], namely the bitstring argument in Chapter 3 and Π_Q the argument described in Section 4.4.1. An USS argument for promise problems either the one given in Section 4.6 or the one given in Section 4.7.

To simplify the exposition we have omitted many details that actually make the proof more efficient. In particular, instead of using two linear arguments for each depth of the circuit, we can use the linear argument for all the linear constraints of the circuit at once (as it is also done in the original work [69]). First, it is easy to see one can prove all the left (and right) constraints together, by considering a larger matrix. Second, left and right constraints can be merged in a single matrix which consists of elements in both groups, and using an argument for some promise problem in *bilateral* linear spaces. This also makes the auxiliary variable O^* (and related equations) unnecessary.

Efficiency. Then, the building blocks Input, Π_Q of our instantiation are exactly the same as in González and Ràfols [69]. The cost of committing to the input plus proving it is boolean with the argument of Chapter 3 is $(2n_s + 4, 6)$. We take the same idea for quadratic constraints proof from [69] with Zero-Knowledge applied to our equations (4.7, 4.8), that is $(6d-3, 2d-1)$ for the commitments and $(4d-4, 8d-8)$ for the GS proofs. This is the same cost as in [69], using an approach where we add more elements in the crs, but we gain in the commitment size. This approach is explained in detail in Section 4.8, in our case the direct approach gives us $(12d-12, 4d-4)$ elements in the commitment, while using the approach in Section 4.8 we add $(4d-2, 2d-2)$ elements in the crs and the commitment size is reduced to about 25% in group $\mathbb{G}_1$. Finally, the overhead of using an USS argument for promise problems in bilateral spaces as opposed to the argument for bilateral spaces with standard soundness used in González and Ràfols [69] is only 3 elements in $\mathbb{G}_1$ in case of USS argument in 4.6, and 15 elements in 4.7.

$\underline{\mathsf{SSetup}(1^\lambda, \mathbf{R})}$
Run $(\mathsf{crs}, \mathsf{tr}_s, \mathsf{tr}_e) \leftarrow \mathsf{K}_1(gk, \phi)$ where crs fixes a tag space $\mathcal{T}$, define a collision resistant hash function H and return $\mathsf{pp} = (\mathsf{crs}, H)$.

$\underline{\mathsf{SVer}(\mathsf{pp}, \boldsymbol{x}_p, \boldsymbol{\sigma}, \mathsf{m})}$
Compute $\tau = H(\boldsymbol{x}_p, \mathsf{m})$,
return $\mathsf{V}(\mathsf{crs}, \boldsymbol{x}_p, \boldsymbol{\sigma}, \tau)$.

$\underline{\mathsf{SSign}(\mathsf{pp}, \boldsymbol{x}_p, \boldsymbol{w}, \mathsf{m})}$
Compute $\tau = H(\boldsymbol{x}_p, \mathsf{m})$,
return $\boldsymbol{\sigma} \leftarrow \mathsf{P}(\mathsf{crs}, \boldsymbol{x}_p, \boldsymbol{w}, \tau)$.

$\underline{\mathsf{SSimulate}(\mathsf{pp}, \mathsf{tr}_s, \boldsymbol{x}_p)}$
Compute $\tau = H(\boldsymbol{x}_p, \mathsf{m})$,
return $\boldsymbol{\sigma} \leftarrow \mathsf{S}(\mathsf{crs}, \mathsf{tr}_s, \boldsymbol{x}_p, \tau)$.

Figure 4.1: SoK based on the tag-based SE-NIZK of Section 4.5, with algorithms $(\mathsf{P}, \mathsf{V}, \mathsf{S})$ and $\mathsf{m} \in \mathcal{M}$.

4.5.2 Signature of Knowledge

Next, we construct a Signature of Knowledge (SoK) for boolean CircuitSat. Similarly, Groth and Maller [74](See Section 2.7.3) build a SoK using a Simulation Extractable NIZK with non-black-box extraction along with a universal one-way hash function. We use a different approach and take advantage of having a tag-based argument, and we set the tag to be the output of a hash function of the message to be signed together with the public input. The efficiency of the SoK is essentially the same as the SE-NIZK on which it relies, because we just need to add a collision resistant hash function in the public parameters and compute a hash for proving/verifying the relation.

The construction of Groth and Maller is based on knowledge assumptions and non-black box extraction, while our NIZK is based on falsifiable assumptions and the extractor is used as a black box.

Signature of Knowledge for circuit satisfiability under standard assumptions

We present a Signature scheme of Knowledge based in the tag-based SE-NIZK argument of Section 4.5 for boolean CircuitSat. To sign a message m, we use a collision resistant hash function of the message and the public statement, the result is used as the tag of the argument behind. If an adversary tries to reuse the same proof to forge a signature, it should be for a different message, otherwise we have the same tag.

Given a message space $\mathcal{M}$ and a relation $\mathbf{R} \in \mathcal{R}$, we give a signature scheme in Figure 4.1 that is the natural transformation of the tag-based SE-NIZK argument of Section 4.5 to a Signature of Knowledge for $\mathbf{R}$.

4.6 USS QA-NIZK Arguments of Knowledge Transfer for Linear Spaces

In this section we prove that the USS argument for membership in linear spaces of Kiltz and Wee also satisfies the "knowledge transfer" property, or more technically, that it has soundness for the same promise problem described in Section 4.4.2. We give the argument for membership in linear spaces in one group in detail in Section 4.6.1 and we present the scheme for the bilateral version in Section 4.6.2.

4.6.1 USS $\mathsf{Lin}_{\mathcal{D}_k}$ argument

In this section we present $\mathsf{Lin}_{\mathcal{D}_k}$, a quasi-adaptive USS argument of membership in linear spaces in the group $\mathbb{G}_1$ for the promise problem defined by languages

$$\mathcal{L}_{\text{YES}}^{\mathsf{Lin}} = \left\{ (\boldsymbol{w}, [\boldsymbol{x}]_1, [\boldsymbol{y}]_1) : \begin{array}{c} [\boldsymbol{x}]_1 = [\mathbf{M}]_1 \boldsymbol{w} \text{ and} \\ [\boldsymbol{y}]_1 = [\mathbf{N}]_1 \boldsymbol{w} \end{array} \right\}$$

$$\mathcal{L}_{\text{NO}}^{\mathsf{Lin}} = \left\{ (\boldsymbol{w}, [\boldsymbol{x}]_1, [\boldsymbol{y}]_1) : \begin{array}{c} [\boldsymbol{x}]_1 = [\mathbf{M}]_1 \boldsymbol{w} \text{ and} \\ [\boldsymbol{y}]_1 \neq [\mathbf{N}]_1 \boldsymbol{w} \end{array} \right\}$$

parameterized by matrices $\mathbf{M} \in \mathbb{Z}_p^{\ell_1 \times n}, \mathbf{N} \in \mathbb{Z}_p^{\ell_2 \times n}$ sampled from some distributions $\mathcal{M}, \mathcal{N}$. Completeness holds for YES instances, and soundness guarantees that NO instances will not be accepted. That is, as in [69], we assume $[\boldsymbol{x}]_1 = [\mathbf{M}]_1 \boldsymbol{w}$ holds when proving soundness. In the CircuitSat context, this can be assumed because the idea is that this is proven by first proving knowledge of the input and then by "transferring" this knowledge to the lower layers via the quadratic or the linear argument we have presented. We consider the general language $\mathcal{L}$ that includes all tuples $(\boldsymbol{w}, \boldsymbol{x}, \boldsymbol{y})$ of the right dimension, some of them which are outside of $\mathcal{L}_{\text{YES}}^{\mathsf{Lin}} \cup \mathcal{L}_{\text{NO}}^{\mathsf{Lin}}$. We allow simulation queries for any tuple in $\mathcal{L}$. Note that it would be enough to allow the adversary just to ask for queries in $\mathcal{L}_{\text{NO}}^{\mathsf{Lin}}$ in some contexts, as in Section 4.5 for CircuitSat, but we define this more generally.

Scheme Definition

The argument is presented in Figure 4.2 and note that it is just the USS QA-NIZK argument of [92] written in two blocks, which adds a pseudorandom MAC to the basic (not simulation sound, just sound) QA-NIZK argument of membership in linear spaces for general distributions also given in [92]. If in the basic arguments the proofs are of the form $[\boldsymbol{x}^\top, \boldsymbol{y}^\top]_1(\mathbf{K}_1, \mathbf{K}_2)$, in the USS variant they are given by

$\underline{\mathsf{K}\left(gk, [\mathbf{M}]_1, [\mathbf{N}]_1\right):}$
$\mathbf{K}_1 \leftarrow \mathbb{Z}_p^{\ell_1 \times (k+1)}, \mathbf{K}_2 \leftarrow \mathbb{Z}_p^{\ell_2 \times (k+1)},$
$\mathbf{K}^\top = \left(\begin{array}{c} \mathbf{K}_1^\top, \mathbf{K}_2^\top \end{array}\right)$
$\mathbf{A}, \boldsymbol{\Omega} \leftarrow \mathcal{D}_k,$
$\boldsymbol{\Omega}_0, \boldsymbol{\Omega}_1 \leftarrow \mathbb{Z}_p^{(k+1)\times(k+1)}$
$\mathbf{C}_1 = \mathbf{K}_1\mathbf{A}, \mathbf{C}_2 = \mathbf{K}_2\mathbf{A},$
$[\mathbf{B}]_1 = \left[\mathbf{M}^\top\mathbf{K}_1 + \mathbf{N}^\top\mathbf{K}_2\right]_1,$
$(\mathbf{P}_0, \mathbf{P}_1) = (\boldsymbol{\Omega}^\top\boldsymbol{\Omega}_0,$
$\boldsymbol{\Omega}^\top\boldsymbol{\Omega}_1)$
$(\mathbf{Q}_0, \mathbf{Q}_1) = (\boldsymbol{\Omega}_0\mathbf{A}, \boldsymbol{\Omega}_1\mathbf{A})$
Return $\mathsf{crs} = \left(gk, [\mathbf{B}]_1, [\mathbf{A}]_2, [\mathbf{P}_0]_1, [\mathbf{P}_1]_1, [\mathbf{Q}_0]_2, [\mathbf{Q}_1]_2, [\mathbf{C}_1]_2, [\mathbf{C}_2]_2, [\boldsymbol{\Omega}]_1\right)$
$\mathsf{tr} = (\mathbf{K}_1, \mathbf{K}_2)$

$\underline{\mathsf{P}\left(\mathsf{crs}, \tau, [\boldsymbol{x}]_1, [\boldsymbol{y}]_1, \boldsymbol{w}\right):}$
Pick $\boldsymbol{r} \leftarrow \mathbb{Z}_p^k$ and return
$\boldsymbol{\pi} = \left(\boldsymbol{w}^\top[\mathbf{B}]_1 + \boldsymbol{r}^\top[\mathbf{P}_0 + \tau\mathbf{P}_1]_1, [\boldsymbol{r}^\top\boldsymbol{\Omega}^\top]_1\right).$
$\underline{\mathsf{V}\left(\mathsf{crs}, \tau, [\boldsymbol{x}]_1, [\boldsymbol{y}]_1, \boldsymbol{\pi}\right):}$
Check if:
$e\left(\boldsymbol{\pi}_1, [\mathbf{A}]_2\right) - e\left(\left[\boldsymbol{x}^\top, \boldsymbol{y}^\top\right]_1, [\mathbf{C}]_2\right)$
$= e\left(\boldsymbol{\pi}_2, [\mathbf{Q}_0 + \tau\mathbf{Q}_1]_2\right)$
$\underline{\mathsf{S}\left(\mathsf{crs}, \tau, [\boldsymbol{x}]_1, [\boldsymbol{y}]_1, \mathsf{tr}\right):}$
Sample $\boldsymbol{r} \leftarrow \mathbb{Z}_p^k$ and return
$\boldsymbol{\pi} = \left([\boldsymbol{x}^\top, \boldsymbol{y}^\top]_1\mathbf{K} + \boldsymbol{r}^\top[\mathbf{P}_0 + \tau\mathbf{P}_1]_1, [\boldsymbol{r}^\top\boldsymbol{\Omega}^\top]_1\right).$

Figure 4.2: The $\mathsf{Lin}_{\mathcal{D}_k}$ argument for proving membership in linear spaces in blocks $[\boldsymbol{x}, \boldsymbol{y}]_1 \in \mathrm{Im}[\mathbf{M}, \mathbf{N}]_1$ where $\mathbf{M} \in \mathbb{Z}_p^{\ell_1 \times n}, \mathbf{N} \in \mathbb{Z}_p^{\ell_2 \times n}$.

$$\left(\left[(\boldsymbol{x}^\top, \boldsymbol{y}^\top)(\mathbf{K}_1, \mathbf{K}_2) + \boldsymbol{r}^\top\boldsymbol{\Omega}(\boldsymbol{\Omega}_0 + \tau\boldsymbol{\Omega}_1)\right]_1, \left[\boldsymbol{r}^\top\boldsymbol{\Omega}^\top\right]_1\right).$$

Our contribution is not in the scheme but in the security analysis. Our proof follows [69], that proved that the basic argument in [92] is complete and sound for the same promise problem under some MDDH and KerMDH assumptions related to the matrix distribution $\mathcal{M}$. Our contribution is to modify their analysis to adapt it to be simulation sound for the scheme of Figure 4.2.

Perfect Completeness, Perfect Zero-Knowledge. Our language $\mathcal{L}_{\mathrm{YES}}^{\mathsf{Lin}}$ is the same language for membership proofs in a linear space $[\mathbf{M}, \mathbf{N}]_1^\top$ used in [92]: $\left\{(\boldsymbol{w}, [\boldsymbol{x}, \boldsymbol{y}]_1) : [\boldsymbol{x}^\top, \boldsymbol{y}^\top]_1^\top = [\mathbf{M}, \mathbf{N}]_1^\top\, \boldsymbol{w}\right\}$, so perfect completeness and perfect zero-knowledge are immediate.

Unbounded Simulation Soundness. We use Definition 28, for any adversary $\mathcal{A}$ that sends any number Q of queries $(\boldsymbol{w}^i, [\boldsymbol{x}^i, \boldsymbol{y}^i]_1) \in \mathcal{L}$ to the query simulator oracle S, receives simulated proofs $\{\boldsymbol{\pi}^i\}_{i=1}^Q$ as described in Figure 4.2, the probability that the adversary $\mathcal{A}$ comes up with $(\boldsymbol{w}^*, [\boldsymbol{x}^*, \boldsymbol{y}^*]_1, \tau^*, \boldsymbol{\pi}^*)$ such that $(\boldsymbol{w}^*, [\boldsymbol{x}^*, \boldsymbol{y}^*]_1) \in \mathcal{L}_{\mathrm{NO}}^{\mathsf{Lin}}$ different of the queried ones, different tag τ^* and $\mathsf{V}(\mathsf{crs}, \tau^*, [\boldsymbol{x}^*, \boldsymbol{y}^*]_1, \boldsymbol{\pi}^*) = 1$ is negligible.

Our proof is analogous to the USS proof of [92], where the authors argue that partial information about matrix $\mathbf{K}$ is computationally hidden across all the simulated proofs. Essentially, what the authors are doing is to reduce the proof of USS to a standard soundness proof. More concretely, they switch to a game where the simulated proofs hide information theoretically the projection of $\mathbf{K}$ for vectors outside of the span of the columns of a matrix $\tilde{\mathbf{M}}$ that defines the language. Therefore, one can argue, as in the standard soundness proof, that the probability of providing a valid proof for a false statement is negligible.

Our proof combines the work of [92] to show that the queries do not provide additional information, with the work of [69] to show standard soundness to the language associated to the promise problem. Indeed, in the case we are interested in the matrix $\tilde{\mathbf{M}}$ spans the whole space so the standard soundness proof used by [92] cannot be used and we need an extra change of games to use a technique proposed by [69] that proves that the block $\mathbf{K}_{2,2}$ is hidden from the adversary. This block is part of the matrix $\mathbf{K}_2$ and corresponds to the part of the statement that is not in the correct linear space. That is, for breaking soundness the adversary has to create a valid proof for $(\boldsymbol{w}, [\boldsymbol{x}]_1, [\boldsymbol{y}]_2)$ such that $\boldsymbol{y} \neq \mathbf{N}\boldsymbol{w}$ and $\boldsymbol{x} = \mathbf{M}\boldsymbol{w}$), and the coordinates of this block correspond to the projection by matrix $\mathbf{N}$. Concretely, at some point in their proof, Kiltz and Wee change the key matrix uniformly sampled for another of the form $\mathbf{K}' + \boldsymbol{b}\boldsymbol{a}^{\perp}$, where $\mathbf{K}'$ is uniformly sampled and $\boldsymbol{a}^{\perp}$ is in the co-kernel of $\mathbf{A}$. We apply the same change but in blocks, $\boldsymbol{b} = (\boldsymbol{b}_1, \boldsymbol{b}_2)$, so our extra game consists in changing the projection of $\boldsymbol{b}_1$ by $\mathbf{M}^{\top}$ to some random vector $\boldsymbol{z}$, i.e. we change $\mathbf{M}^{\top}\boldsymbol{b}_1 + \mathbf{N}^{\top}\boldsymbol{b}_2$ to $\boldsymbol{z} + \mathbf{N}^{\top}\boldsymbol{b}_2$ by assuming the $\mathcal{M}^{\top}$-$\mathsf{MDDH}_{\mathbb{G}_1}$ assumption, where $\mathcal{M}^{\top}$ is the matrix that defines the distribution of $\mathbf{M}^{\top}$ (as in [69]). So, what the adversary can see about $\boldsymbol{b}$ is just $\mathbf{N}^{\top}\boldsymbol{b}_2$ but it is hidden by $\boldsymbol{z}$.

For the following theorem, we use the Computational Core Lemma of Kiltz and Wee in Section 4.1. of [92], which is independent of $\mathcal{M}, \mathcal{N}$, it just assumes the $\mathcal{D}_k$-$\mathsf{MDDH}_{\mathbb{G}_1}$, so we can use it directly in our proof.

Theorem 26. *The $\mathsf{Lin}_{\mathcal{D}_k}$ scheme in Figure 4.2 is a Quasi-adaptive Non-Interactive Zero-Knowledge Argument with Unbounded Simulation Soundness such that for any adversary $\mathcal{A}$ that makes at most Q queries to S there exist adversaries $\mathcal{B}_1$, $\mathcal{B}_2$, $\mathcal{B}_3$ against the $\mathcal{D}_k$-KerMDH, $\mathcal{M}^{\top}$-MDDH assumptions in $\mathbb{G}_1$ for which the advantage of $\mathcal{A}$ is bounded by*

$$\begin{aligned}\mathsf{Adv}_{\mathsf{USS}\text{-}\mathsf{Lin}_{\mathcal{D}_k}}(\mathcal{A}) \leq & \mathsf{Adv}_{\mathcal{D}_k-\mathsf{KerMDH}_{\mathbb{G}_1}}(\mathcal{B}_1) + 2Q\mathsf{Adv}_{\mathcal{D}_k\text{-}\mathsf{MDDH}_{\mathbb{G}_1}}(\mathcal{B}_2) \\ & + \mathsf{Adv}_{\mathcal{M}^{\top}\text{-}\mathsf{MDDH}_{\mathbb{G}_1}}(\mathcal{B}_3) + \frac{Q+1}{p}.\end{aligned}$$

Proof. Let $\mathcal{A}$ be an adversary that plays the game described in USS definition 28. We

will proceed by changing to indistinguishable games in order to bound the advantage of $\mathcal{A}$. Let Game_0 be the real game and Adv_i the advantage of winning Game_i.

Game_1 is the same as Game_0 except the verification algorithm V is changed to

$$\frac{\mathsf{V}^*(\mathsf{crs}, \tau, [\boldsymbol{x}, \boldsymbol{y}]_1, \boldsymbol{\pi}) :}{\text{Check: } \boldsymbol{\pi}_1 = [\boldsymbol{x}^\top, \boldsymbol{y}^\top]_1 \mathbf{K} + \boldsymbol{\pi}_2(\boldsymbol{\Omega}_0 + \tau \boldsymbol{\Omega}_1).}$$

If a tuple $([\boldsymbol{x}, \boldsymbol{y}]_1, \boldsymbol{\pi})$ passes verification of V but does not pass verification of V^*, it means that the value $\boldsymbol{\pi} - [\boldsymbol{x}^\top, \boldsymbol{y}^\top]_1 \mathbf{K} - \boldsymbol{\pi}_2(\boldsymbol{\Omega}_0 + \tau \boldsymbol{\Omega}_1) \in \mathbb{G}_1^{k+1}$ is a non-zero vector in the co-kernel of $\mathbf{A}$. Thus, there exists an adversary $\mathcal{B}_1$ against $\mathsf{KerMDH}_{\mathbb{G}_1}$ such that

$$|\mathsf{Adv}_0 - \mathsf{Adv}_1| \leq \mathsf{Adv}_{\mathcal{D}_k - \mathsf{KerMDH}_{\mathbb{G}_1}}(\mathcal{B}_1).$$

Game_2 is the same as Game_1 except the simulation algorithm S is changed to

$$\frac{\mathsf{S}^*(\mathsf{crs}, \tau, [\boldsymbol{x}, \boldsymbol{y}]_1, \mathsf{tr}) :}{\boldsymbol{r} \leftarrow \mathbb{Z}_p^k, \mu \leftarrow \mathbb{Z}_p}$$
$$\text{Return: } \boldsymbol{\pi} = ([(\boldsymbol{x}^\top, \boldsymbol{y}^\top)\mathbf{K} + \mu \boldsymbol{a}^\perp + \boldsymbol{r}^\top(\mathbf{P}_0 + \tau \mathbf{P}_1)]_1, [\boldsymbol{r}^\top \boldsymbol{\Omega}]_1),$$

where $\boldsymbol{a}^\perp$ is an element from the Kernel of $\mathbf{A}$. Let $\mathcal{B}_2$ be an adversary against $\mathcal{D}_k$-$\mathsf{MDDH}_{\mathbb{G}_1}$. $\mathcal{B}_2$ picks $\mathbf{K}$ itself and answers queries $(\tau_i, \boldsymbol{w}_i, [\boldsymbol{x}_i, \boldsymbol{y}_i]_1)$ from $\mathcal{A}$:

- if $\tau_i \neq \tau^*$: $\mathcal{B}_2$ queries the oracle $\mathcal{O}_b$, defined in the core lemma [92], who simulates S if $b = 0$, or S^* if $b = 1$.
- if $\tau_i = \tau^*$: $\mathcal{B}_2$ samples $\boldsymbol{r} \leftarrow \mathbb{Z}_p$ and computes $([(\boldsymbol{x}_i^\top, \boldsymbol{y}_i^\top)\mathbf{K} + \boldsymbol{r}^\top(\mathbf{P}_0 + \tau_i \mathbf{P}_1)]_1, [\boldsymbol{r}^\top \boldsymbol{\Omega}_0^\top]_1)$.

Then, $\mathcal{B}_2$ queries V^* to simulate verification of the final message of $\mathcal{A}$, $(\tau^*, \boldsymbol{w}^*, [\boldsymbol{x}^*, \boldsymbol{y}^*]_1)$. Now, it is easy to check if $(\boldsymbol{w}^*, [\boldsymbol{x}^*, \boldsymbol{y}^*]_1) \in \mathcal{L}_{\mathsf{NO}}^{\mathsf{Lin}}$ by computing $[\mathbf{N}]_1 \boldsymbol{w}^*$. The difference between respective advantages is bounded using the core lemma of [92] as

$$|\mathsf{Adv}_1 - \mathsf{Adv}_2| \leq 2Q \mathsf{Adv}_{\mathcal{D}_k - \mathsf{MDDH}_{\mathbb{G}_1}}(\mathcal{B}_2) + \frac{Q}{p}.$$

Game_3 is the same as Game_2 except the matrix $\mathbf{K} \leftarrow \mathbb{Z}_p^{(\ell_1+\ell_2)\times(k+1)}$ is changed in K to $\mathbf{K} = \mathbf{K}' + \boldsymbol{b}\boldsymbol{a}^\perp$ where $\mathbf{K}' \leftarrow \mathbb{Z}_p^{(\ell_1+\ell_2)\times(k+1)}$, $\boldsymbol{b}_1 \leftarrow \mathbb{Z}_p^{\ell_1}$, $\boldsymbol{b}_2 \leftarrow \mathbb{Z}_p^{\ell_2}$, $\boldsymbol{b}^\top = (\boldsymbol{b}_1^\top, \boldsymbol{b}_2^\top)$ and $\mathbf{B} = (\mathbf{M}^\top, \mathbf{N}^\top)\mathbf{K} + (\boldsymbol{z} + \mathbf{N}^\top \boldsymbol{b}_2)\boldsymbol{a}^\perp$, where $\boldsymbol{z} = \mathbf{M}^\top \boldsymbol{b}_1$. It is direct to see that both $\mathbf{K}, \mathbf{K}'$ are uniformly distributed in $\mathbb{Z}_p^{(\ell_1+\ell_2)\times(k+1)}$, so the advantages in both games are equivalent.

Game_4 is the same as Game_3 except that now $\boldsymbol{z} \leftarrow \mathbb{Z}_p^{\ell_1}$. Let $\mathcal{B}_3$ be an adversary against $\mathcal{D}_k$-$\mathsf{MDDH}_{\mathbb{G}_1}$ that receives $([\mathbf{M}^\top]_1, [\boldsymbol{z}]_1)$ as a challenge and computes the crs as in the previous game with this $[\boldsymbol{z}]_1$ in $\mathbf{B}$ and runs $\mathcal{A}$ as in Game_3. Finally, the advantage of $\mathcal{B}_3$ to distinguish between Game_3 and Game_4 is bounded by the probability of distinguishing between a random vector from the image of the matrix $\mathbf{M}^\top$, so

$$|\mathsf{Adv}_3 - \mathsf{Adv}_4| \leq \mathsf{Adv}_{\mathcal{M}^\top - \mathsf{MDDH}_{\mathbb{G}_1}}(\mathcal{B}_3).$$

Now we bound the advantage of adversary $\mathcal{A}$ in winning Game_4. Firstly, we show what is leaked about vector $\boldsymbol{b}$ in the adversary's view:

- the matrix $\mathbf{C} = (\mathbf{K}' + \boldsymbol{b}\boldsymbol{a}^\perp)\mathbf{A}$ completely hides the vector $\boldsymbol{b}$,
- the output of S^*, $(\boldsymbol{x}, \boldsymbol{y})^\top(\mathbf{K}' + \boldsymbol{b}\boldsymbol{a}^\perp) + \mu\boldsymbol{a}^\perp$ completely hides $\boldsymbol{b}$ because μ masks $(\boldsymbol{x}^\top, \boldsymbol{y}^\top)\boldsymbol{b}$,
- the matrix $\mathbf{B}$ contains information about $\boldsymbol{z} + \mathbf{N}^\top\boldsymbol{b}_2$, but $\boldsymbol{z}$ is uniformly random and independent of $\boldsymbol{b}_2$, so $\boldsymbol{z}$ masks $\boldsymbol{b}_2$.

Note that if the adversary $\mathcal{A}$ passes the verification V^* with some $\boldsymbol{\pi}^*$ for a statement $(\boldsymbol{w}^*, \boldsymbol{x}^*, \boldsymbol{y}^*) \in \mathcal{L}_{\text{NO}}^{\mathsf{Lin}}$, it can compute $\boldsymbol{y} = \mathbf{N}\boldsymbol{w}^*$ and construct a valid proof $\pi = (\boldsymbol{\pi}_1^* - \boldsymbol{w}^*\mathbf{B}, \boldsymbol{\pi}_2^*)$ that the vector $(\mathbf{0}, \boldsymbol{y} - \boldsymbol{y}^*)$ is in the span of the columns $(\mathbf{M}^\top, \mathbf{N}^\top)$. It must hold that

$$\pi = (0, \boldsymbol{y} - \boldsymbol{y}^*)(\mathbf{K}' + \boldsymbol{b}\boldsymbol{a}^\perp) = (\boldsymbol{y} - \boldsymbol{y}^*)\mathbf{K}_2' + (\boldsymbol{y} - \boldsymbol{y}^*)\boldsymbol{b}_2\boldsymbol{a}^\perp. \tag{4.11}$$

Note $\boldsymbol{y} - \boldsymbol{y}^*$ is not zero because $\boldsymbol{y} \neq \boldsymbol{y}^*$. Since $\boldsymbol{b}_2$ remains completely hidden to the adversary and $\mathbf{K}_2'$ is independent of $\boldsymbol{b}_2$, the probability than Equation (4.11) holds is less that $1/p$. □

4.6.2 USS $\mathsf{BLin}_{\mathcal{D}_k}$ argument

In this section we present the USS argument for membership in linear spaces in groups $\mathbb{G}_1$, $\mathbb{G}_2$, which is just an extension to bilateral spaces of the USS $\mathsf{Lin}_{\mathcal{D}_k}$ argument presented in Section 4.6.1 for the promise problem defined by languages

$$\mathcal{L}_{\text{YES}}^{\mathsf{Blin}} = \left\{ (\boldsymbol{w}, [\boldsymbol{x}_1]_1, [\boldsymbol{x}_2]_1, [\boldsymbol{y}]_2) : \begin{array}{c} [\boldsymbol{x}_1]_1 = [\mathbf{M}]_1\boldsymbol{w} \text{ and} \\ [\boldsymbol{x}_2]_1 = [\mathbf{N}]_1\boldsymbol{w}, [\boldsymbol{y}]_2 = [\mathbf{P}]_2\boldsymbol{w} \end{array} \right\}$$

$$\mathcal{L}_{\text{NO}}^{\mathsf{Blin}} = \left\{ (\boldsymbol{w}, [\boldsymbol{x}_1]_1, [\boldsymbol{x}_2]_1, [\boldsymbol{y}]_2) : \begin{array}{c} [\boldsymbol{x}_1]_1 = [\mathbf{M}]_1\boldsymbol{w} \text{ and} \\ [\boldsymbol{x}_2]_1 \neq [\mathbf{N}]_1\boldsymbol{w} \text{ or } [\boldsymbol{y}]_2 \neq [\mathbf{P}]_2\boldsymbol{w} \end{array} \right\}$$

$\underline{\mathsf{K}(gk, [\mathbf{M}]_1, [\mathbf{N}]_1, [\mathbf{P}]_2)}$:
$\mathbf{K}_1 \leftarrow \mathbb{Z}_p^{\ell_1 \times (k+1)}, \mathbf{K}_2 \leftarrow \mathbb{Z}_p^{\ell_2 \times (k+1)}$,
$\mathbf{K}_3 \leftarrow \mathbb{Z}_p^{\ell_3 \times (k+1)}$,
$\mathbf{A}, \boldsymbol{\Omega} \leftarrow \mathcal{D}_k, \boldsymbol{\Gamma} \leftarrow \mathbb{Z}_p^{n \times (k+1)}$,
$\boldsymbol{\Omega}_0, \boldsymbol{\Omega}_1 \leftarrow \mathbb{Z}_p^{(k+1) \times (k+1)}$
$\mathbf{C}_1 = \mathbf{K}_1\mathbf{A}, \mathbf{C}_2 = \mathbf{K}_2\mathbf{A}, \mathbf{C}_3 = \mathbf{K}_3\mathbf{A}$,
$[\mathbf{B}]_1 = \left[\mathbf{M}^\top\mathbf{K}_1 + \mathbf{N}^\top\mathbf{K}_2 + \boldsymbol{\Gamma}\right]_1$
$[\mathbf{D}]_2 = \left[\mathbf{P}^\top\mathbf{K}_3 - \boldsymbol{\Gamma}\right]_2$
$(\mathbf{P}_0, \mathbf{P}_1) = (\boldsymbol{\Omega}^\top\boldsymbol{\Omega}_0, \boldsymbol{\Omega}^\top\boldsymbol{\Omega}_1)$
$(\mathbf{Q}_0, \mathbf{Q}_1) = (\boldsymbol{\Omega}_0\mathbf{A}, \boldsymbol{\Omega}_1\mathbf{A})$
Return $\mathsf{crs} = (gk, [\mathbf{B}]_1, [\mathbf{A}]_{1,2}, [\mathbf{P}_0]_2,$
$[\mathbf{P}_1]_2, [\mathbf{Q}_0]_1, [\mathbf{Q}_1]_1, [\mathbf{C}_1]_2, [\mathbf{C}_2]_2,$
$[\mathbf{C}_3]_1, [\boldsymbol{\Omega}]_1)$
$\mathsf{tr} = (\mathbf{K}_1, \mathbf{K}_2, \mathbf{K}_3)$

$\underline{\mathsf{P}(\mathsf{crs}, \tau, [\boldsymbol{x}_1]_1, [\boldsymbol{x}_2]_1, [\boldsymbol{y}]_2, \boldsymbol{w})}$:
Pick $\boldsymbol{r} \leftarrow \mathbb{Z}_p^k$ and return
$\boldsymbol{\pi} = \left(\boldsymbol{w}^\top[\mathbf{B}]_1 + \boldsymbol{r}^\top[\mathbf{P}_0 + \tau\mathbf{P}_1]_1,\right.$
$\left.[\boldsymbol{r}^\top\boldsymbol{\Omega}^\top]_1\right)$,
$\boldsymbol{\theta} = \boldsymbol{w}^\top[\mathbf{D}]_2$.

$\underline{\mathsf{V}(\mathsf{crs}, \tau, [\boldsymbol{x}_1]_1, [\boldsymbol{x}_2]_1, [\boldsymbol{y}]_2, \boldsymbol{\pi}, \boldsymbol{\theta})}$:
Check if: $e(\boldsymbol{\pi}_1, [\mathbf{A}]_2) - e([\mathbf{A}]_1, \boldsymbol{\theta})$
$-e\left(\left[\boldsymbol{x}_1^\top\right]_1, [\mathbf{C}_1]_2\right) - e\left(\left[\boldsymbol{x}_2^\top\right]_1, [\mathbf{C}_2]_2\right)$
$+e\left([\mathbf{C}_3]_1, \left[\boldsymbol{y}^\top\right]_2\right) = e(\boldsymbol{\pi}_2, [\mathbf{Q}_0 + \tau\mathbf{Q}_1]_2)$

$\underline{\mathsf{S}(\mathsf{crs}, \tau, [\boldsymbol{x}_1]_1, [\boldsymbol{x}_2]_1, [\boldsymbol{y}]_2, \mathsf{tr})}$:
Sample $\boldsymbol{r} \leftarrow \mathbb{Z}_p^k$ and return
$\boldsymbol{\pi} = \left([\boldsymbol{x}_1, \boldsymbol{x}_2]_1(\mathbf{K}_1^\top, \mathbf{K}_2^\top)\right.$
$\left.+\boldsymbol{r}^\top(\mathbf{P}_0 + \tau\mathbf{P}_1), [\boldsymbol{r}^\top\boldsymbol{\Omega}^\top]_1\right)$,
$\boldsymbol{\theta} = [\boldsymbol{y}]_2\mathbf{K}_3^\top$.

Figure 4.3: The $\mathsf{BLin}_{\mathcal{D}_k}$ argument for proving membership in linear spaces in blocks $([\boldsymbol{x}_1, \boldsymbol{x}_2]_1, [\boldsymbol{y}]_2) \in \mathrm{Im}\left([\mathbf{M}, \mathbf{N}]_1, [\mathbf{P}]_2, \right)$ where $\mathbf{M} \in \mathbb{Z}_p^{\ell_1 \times n}, \mathbf{N} \in \mathbb{Z}_p^{\ell_2 \times n}, \mathbf{P} \in \mathbb{Z}_p^{\ell_3 \times n}$.

parameterized by matrices $\mathbf{M} \in \mathbb{Z}_p^{\ell_1 \times n}, \mathbf{N} \in \mathbb{Z}_p^{\ell_2 \times n}, \mathbf{P} \in \mathbb{Z}_p^{\ell_3 \times n}$ sampled from some distributions $\mathcal{M}, \mathcal{N}, \mathcal{P}$. This argument is presented in Figure 4.3. QA-NIZK arguments of membership in linear spaces were extended to the bilateral case in [67] for both samplable and non-witness samplable distributions. In [69], the authors proved that the argument for non-witness samplable distributions of [67] is also sound and complete for this promise problem. Adding the pseudorandom MAC given in [92] we get USS. The proof is essentially the same as in 4.6.1, but now the linear spaces are split in two groups $\mathbb{G}_1$ and $\mathbb{G}_2$. The core lemma would be the same and the reduction of the proof of USS is bounded by $\mathsf{SKerMDH}$ and $\mathcal{D}_k$-$\mathsf{MDDH}_{\mathbb{G}_1}$ Assumptions.

4.7 Tight USS QA-NIZK Arguments of Knowledge Transfer for Linear Spaces

In this section we prove that the Tight USS argument of Abe et al. [5] for membership in linear spaces satisfies the knowledge transfer property explained in Section 4.4.2. The authors present a Designated Verifier (DV) QA-NIZK argument and then use a

well-known conversion from DV to public verifier QA-NIZK with pairings. We follow the same approach and we further modify it to be a tag-based argument and adapt the sub-argument for disjunction spaces to the one of Couteau and Hartmann [40] for efficiency.

In Section 4.7.1 we prove the DV QA-NIZK of [5] is perfectly complete, perfectly zero-knowledge and USS for the language associated to promise problems for linear spaces, already defined in Section 4.6, namely:

$$\mathcal{L}^{\mathsf{Lin}}_{\text{YES}} = \left\{ (\boldsymbol{w}, [\boldsymbol{x}]_1, [\boldsymbol{y}]_1) : \begin{array}{c} [\boldsymbol{x}]_1 = [\mathbf{M}]_1 \boldsymbol{w} \text{ and} \\ [\boldsymbol{y}]_1 = [\mathbf{N}]_1 \boldsymbol{w} \end{array} \right\}$$

$$\mathcal{L}^{\mathsf{Lin}}_{\text{NO}} = \left\{ (\boldsymbol{w}, [\boldsymbol{x}]_1, [\boldsymbol{y}]_1) : \begin{array}{c} [\boldsymbol{x}]_1 = [\mathbf{M}]_1 \boldsymbol{w} \text{ and} \\ [\boldsymbol{y}]_1 \neq [\mathbf{N}]_1 \boldsymbol{w} \end{array} \right\}$$

parametrized by matrices $\mathbf{M} \in \mathbb{Z}_p^{\ell_1 \times n}, \mathbf{N} \in \mathbb{Z}_p^{\ell_2 \times n}$ sampled from some distributions $\mathcal{M}, \mathcal{N}$. In Section 4.7.2 we present its natural conversion to a publicly verifiable QA-NIZK argument. We only give the argument for membership in linear spaces in one group, the bilateral version is straightforward following the work of [67], where the authors transform QA-NIZK arguments for membership in linear spaces in one group to membership in linear spaces to both groups, namely bilateral spaces.

Security Proof: Intuition

Our construction revisits the proof of Abe et al.'s DV argument for promise problems. In this approach the secret keys are vectors $\boldsymbol{k}_0, \boldsymbol{k}_1$ and the proofs, $(\boldsymbol{x}_i^\top, \boldsymbol{y}_i^\top)(\boldsymbol{k}_0 + \tau_i \boldsymbol{k}_1)$ where τ is a different value in $\mathbb{Z}_p$ for each proof. We split the secret keys $\boldsymbol{k}_0 = (\boldsymbol{k}_{1,0}, \boldsymbol{k}_{2,0})$, $\boldsymbol{k}_1 = (\boldsymbol{k}_{1,1}, \boldsymbol{k}_{2,1})$ to indicate the components that come with $\mathbf{M}^\top$, $\boldsymbol{k}_{1,0}, \boldsymbol{k}_{1,1}$, and the others with $\mathbf{N}^\top$, $\boldsymbol{k}_{2,0}, \boldsymbol{k}_{2,1}$.

We use a similar solution as in Section 4.6 and argue that partial information of the secret keys necessary to produce a proof in the NO language is hidden across all the proofs. In this construction, the crs contains projections of the secret keys $\boldsymbol{k}_0, \boldsymbol{k}_1$ by matrices $\mathbf{M}^\top, \mathbf{N}^\top$. Assuming the $\mathcal{M}^\top$-$\mathsf{MDDH}_{\mathbb{G}_1}$ assumption holds, where $\mathcal{M}^\top$ is the distribution of $\mathbf{M}^\top$, as in Section 4.6, we change the projection by $\mathbf{M}^\top$ by a random vector $\boldsymbol{z}$, which masks completely the projection by $\mathbf{N}^\top$.

Note that in the construction of Abe et al.'s we use in this section, there are also more projections of the secret keys leaked from simulated proofs, concretely: $\boldsymbol{x}_i^\top(\boldsymbol{k}_{1,0} + \tau_i \boldsymbol{k}_{1,1}) + \boldsymbol{y}_i^\top(\boldsymbol{k}_{2,0} + \tau_i \boldsymbol{k}_{2,1})$. But we can use the same information-theoretic argument as in [5], namely, since τ_i is different each time, $\boldsymbol{k}_{1,0} + \tau_i \boldsymbol{k}_{1,1}$, $\boldsymbol{k}_{2,0} + \tau_i \boldsymbol{k}_{2,1}$ are pairwise independent, then they do not add any clue to the adversary.

4.7.1 Tight DV QA-NIZK Argument of Knowledge Transfer for Linear Spaces.

The DV QA-NIZK argument presented in Figure 4.4 is the argument for linear spaces of Abe et al. [5] written in blocks, and (trivially) modified to admit tags. Also, We use the disjunction argument of Couteau and Hartmann [40], which is 3 group elements more efficient than the one presented in [6] (used in the first construction of Abe et al. [5]), and we denote it by or.

Security

We prove it has completeness for $\mathcal{L}^{\mathsf{Lin}}_{\text{YES}}$ and USS for $\mathcal{L}^{\mathsf{Lin}}_{\text{NO}}$. USS relies in the same core lemma as in Abe et al. (Lemma 3 in [5]), the security of the MAC presented in Gay et al. [60], the soundness of an argument for membership in a disjunction space of [40]. Our contribution is to combine this with the same techniques as in Section 4.6 to adapt the proof for promise problems.

The scheme in Figure 4.4 is perfectly complete and perfect zero knowledge for YES instances, and soundness guarantees that NO instances will not be accepted as we show in the following. As in Section 4.6 we consider the general language $\mathcal{L}$ that includes all tuples $(\boldsymbol{w}, \boldsymbol{x}, \boldsymbol{y})$ of the right dimension, some of them are outside of $\mathcal{L}^{\mathsf{Lin}}_{\text{YES}} \cup \mathcal{L}^{\mathsf{Lin}}_{\text{NO}}$. We allow simulation queries for any tuple in $\mathcal{L}$.

Perfect Completeness, Perfect Zero-Knowledge. Our language $\mathcal{L}^{\mathsf{Lin}}_{\text{YES}}$ is the same language for membership proofs in a linear space $[\mathbf{M}, \mathbf{N}]_1^\top$ used in [5]:

$$\left\{ (\boldsymbol{w}, [\boldsymbol{x}, \boldsymbol{y}]_1) : [\boldsymbol{x}, \boldsymbol{y}]_1^\top = [\mathbf{M}, \mathbf{N}]_1^\top \boldsymbol{w} \right\}.$$

Thus, we directly obtain perfect completeness and perfect zero-knowledge.

Unbounded Simulation Soundness. We use the definition 28 where for any adversary $\mathcal{A}$ that sends any number Q of queries $(\boldsymbol{w}_i, [\boldsymbol{x}_i, \boldsymbol{y}_i]_1, \tilde{\tau}_i)$ to the query simulator oracle S, receives simulated proofs $\{[\pi_i]_1\}_{i=1}^Q$ as described in Figure 4.4. The probability of the adversary $\mathcal{A}$ comes up with a proof $[\pi^*]_1$ for a statement $(\boldsymbol{w}^*, [\boldsymbol{x}^*, \boldsymbol{y}^*]_1) \in \mathcal{L}^{\mathsf{Lin}}_{\text{NO}}$ different of the queried ones and different tag $\tilde{\tau}^*$, such that $\mathsf{V}(\mathsf{crs}, \tilde{\tau}^*, [\boldsymbol{x}^*, \boldsymbol{y}^*]_1, [\pi^*]_1) = 1$, is negligible.

Abe et al.'s construction is based in the USS Kiltz and Wee argument [92], where the security relies in three security features that we use as black-boxes: their core lemma (Lemma 3 in [5]), the security of a MAC scheme presented in Gay et al. [60], and the soundness of the or argument, all proven secure under standard assumptions.

$\underline{\mathsf{K}(gk, [\mathbf{M}]_1, [\mathbf{N}]_1)}$:
$\mathbf{A}_0, \mathbf{A}_1 \leftarrow \mathcal{D}_{2k,k}, H \leftarrow \mathcal{H}$,
$\mathsf{crs}_{\mathsf{or}} \leftarrow \mathsf{K}(gk, \mathbf{A}_0, \mathbf{A}_1)$
$\boldsymbol{k} \leftarrow \mathbb{Z}_p^{2k}, \boldsymbol{k}_0 = (\boldsymbol{k}_{1,0}, \boldsymbol{k}_{2,0})$,
$\boldsymbol{k}_1 = (\boldsymbol{k}_{1,1}, \boldsymbol{k}_{2,1}) \leftarrow \mathbb{Z}_p^n$,
$\boldsymbol{k}_{1,0}, \boldsymbol{k}_{1,1} \in \mathbb{Z}_p^{\ell_1}, \boldsymbol{k}_{2,0}, \boldsymbol{k}_{2,1} \in \mathbb{Z}_p^{\ell_2}$
$[\boldsymbol{p}]_1 = [\mathbf{A}_0^\top \boldsymbol{k}]_1 \in \mathbb{G}_1^k$,
$[\boldsymbol{p}_0]_1 = [\mathbf{M}^\top \boldsymbol{k}_{1,0} + \mathbf{N}^\top \boldsymbol{k}_{2,0}]_1 \in \mathbb{G}_1^n$,
$[\boldsymbol{p}_1]_1 = [\mathbf{M}^\top \boldsymbol{k}_{1,1} + \mathbf{N}^\top \boldsymbol{k}_{2,1}]_1 \in \mathbb{G}_1^n$,
$\mathsf{crs} = (\mathsf{crs}_{\mathsf{or}}, [\mathbf{A}_0]_1, [\boldsymbol{p}]_1, [\boldsymbol{p}_0]_1, [\boldsymbol{p}_1]_1, H)$
$\mathsf{tr} = (\boldsymbol{k}_0, \boldsymbol{k}_1), \mathsf{vk} = (\boldsymbol{k}, \boldsymbol{k}_0, \boldsymbol{k}_1)$.

$\underline{\mathsf{S}(\mathsf{crs}, [\boldsymbol{x}]_1, [\boldsymbol{y}]_1, \tilde{\tau}, \mathsf{tr})}$:
$\boldsymbol{s} \leftarrow \mathbb{Z}_p^k, [\boldsymbol{t}]_1 = [\mathbf{A}_0]_1 \boldsymbol{s}$,
$[\pi_{\mathsf{or}}]_{1,2} \leftarrow \mathsf{P}_{\mathsf{or}}(\mathsf{crs}_{\mathsf{or}}, [\boldsymbol{t}]_1, \boldsymbol{s})$
$\tau = H([\boldsymbol{x}]_1, [\boldsymbol{y}]_1, [\boldsymbol{t}]_1, [\pi_{\mathsf{or}}]_{1,2}, \tilde{\tau}) \in \mathbb{Z}_p$,
$[u]_1 = [\boldsymbol{x}^\top]_1 (\boldsymbol{k}_{1,0} + \tau \boldsymbol{k}_{1,1})$
$+ [\boldsymbol{y}^\top]_1 (\boldsymbol{k}_{2,0} + \tau \boldsymbol{k}_{2,1}) + \boldsymbol{s}^\top [\boldsymbol{p}]_1$
Return $[\pi]_1 = ([\boldsymbol{t}]_1, [u]_1, [\pi_{\mathsf{or}}]_{1,2})$

$\underline{\mathsf{P}(\mathsf{crs}, [\boldsymbol{x}]_1, [\boldsymbol{y}]_1, \tilde{\tau}, \boldsymbol{w})}$:
$\boldsymbol{s} \leftarrow \mathbb{Z}_p^k, [\boldsymbol{t}]_1 = [\mathbf{A}_0]_1 \boldsymbol{s}$
$[\pi_{\mathsf{or}}]_{1,2} \leftarrow \mathsf{P}_{\mathsf{or}}(\mathsf{crs}_{\mathsf{or}}, [\boldsymbol{t}]_1, \boldsymbol{s})$
$\tau = H([\boldsymbol{x}]_1, [\boldsymbol{y}]_1, [\boldsymbol{t}]_1, [\pi_{\mathsf{or}}]_{1,2}, \tilde{\tau}) \in \mathbb{Z}_p$
$[u]_1 = [\boldsymbol{w}^\top (\boldsymbol{p}_0 + \tau \boldsymbol{p}_1) + \boldsymbol{s}^\top \boldsymbol{p}]_1$
Return $[\pi]_1 = ([\boldsymbol{t}]_1, [u]_1, [\pi_{\mathsf{or}}]_{1,2})$.

$\underline{\mathsf{V}(\mathsf{crs}, [\boldsymbol{x}]_1, [\boldsymbol{y}]_1, \mathsf{vk}, [\pi]_1, \tilde{\tau})}$:
Parse $[\pi] = ([\boldsymbol{t}]_1, [u]_1, [\pi_{\mathsf{or}}]_{1,2})$,
$\tau = H([\boldsymbol{x}]_1, [\boldsymbol{y}]_1, [\boldsymbol{t}]_1, [\pi_{\mathsf{or}}]_{1,2}, \tilde{\tau}) \in \mathbb{Z}_p$,
Check $[\pi_{\mathsf{or}}]_{1,2}$ and
$[u]_1 = [\boldsymbol{x}^\top]_1 (\boldsymbol{k}_{1,0} + \tau \boldsymbol{k}_{1,1})$
$+ [\boldsymbol{y}^\top]_1 (\boldsymbol{k}_{2,0} + \tau \boldsymbol{k}_{2,1}) + [\boldsymbol{t}^\top]_1 \boldsymbol{k}$
Return $0/1$.

Figure 4.4: Tight DV QA-NIZK Argument for membership in linear spaces of Abe et al. [5] in blocks, $[\boldsymbol{x}, \boldsymbol{y}]_1 \in \mathrm{Im}[\mathbf{M}, \mathbf{N}]_1$, where $\mathbf{M} \in \mathbb{Z}_p^{\ell_1 \times n}, \mathbf{N} \in \mathbb{Z}_p^{\ell_2 \times n}$ and $\mathcal{H}$ a family of collision-resistant hash functions. The scheme is modified to be tag-based and is written in blocks. We use the disjunction argument or of [40] with $|\mathsf{crs}_{\mathsf{or}}| = (4n+8)|\mathbb{G}_1| + (2\ell_1 + 3)|\mathbb{G}_2|$, $|\pi_{\mathsf{or}}| = 8|\mathbb{G}_1| + 3|\mathbb{G}_2|$.

Both [92] and [5] use a MAC scheme to add randomness to the proof. Concretely, by the Gay et al. MAC, the term $\boldsymbol{t}^\top \boldsymbol{k}$ is added to the proof, where $\boldsymbol{k}$ is uniformly random and $\boldsymbol{t} \in \text{Span}(\mathbf{A}_0) \cup \text{Span}(\mathbf{A}_1)$ for some fixed matrices $\mathbf{A}_0, \mathbf{A}_1 \in \mathbb{Z}_p^{2k\times k}$ in the crs. The basic idea is the prover computes $\boldsymbol{t}$ directly in the image of $[\mathbf{A}_0]_1$, uses the argument or to prove membership of $\boldsymbol{t}$ in $\text{Span}(\mathbf{A}_0) \cup \text{Span}(\mathbf{A}_1)$ and uses the image space of $[\mathbf{A}_1]_1$ to add randomness in the security proof. The last is done by changing to a game where $\boldsymbol{k} \in \mathbb{Z}_p^{2k}$ is switched to $\boldsymbol{k} + \mathsf{RF}(\cdot)$, with $\mathsf{RF} : \mathbb{Z}_p \to \mathbb{Z}_p^{2k}$ a random function. Indistinguishability of both games is proven in [60], concretely, the lemma gives the following tight bound for any adversary $\mathcal{A}$ that is able to distinguish between both MAC schemes:

$$\mathsf{Adv}_{\mathsf{CL}}(\mathcal{A}) \leq (4k\lceil \log Q\rceil + 2)\mathsf{Adv}_{\mathcal{D}_{2k,k}\text{-}\mathsf{MDDH}_{\mathbb{G}_1},\mathcal{B}}(\lambda) + (2\lceil \log Q\rceil + 2)\mathsf{Adv}_{\text{zk-or},\mathcal{B}'}(\lambda)$$

$$+\lceil \log Q\rceil \Delta_{\mathcal{D}_{2k,k}} + \frac{4\lceil \log Q\rceil + 2}{p-1} + \frac{\lceil \log Q\rceil Q}{p},$$

where $\Delta_{\mathcal{D}_{2k,k}}$ is statistically small term for $\mathcal{D}_{2k,k}$, $\mathcal{B}$ and $\mathcal{B}'$ are adversaries against the $\mathcal{D}_{2k,k}$-$\mathsf{MDDH}_{\mathbb{G}_1}$ assumption and zero-knowledge of argument or (zk-or) respectively.

Theorem 27. *The argument of Figure 4.4 is a Designated Verifier Quasi-Adaptive Non-Interactive Zero-Knowledge argument that guarantees USS such that for any adversary $\mathcal{A}$ that makes at most Q queries to S, there exist adversaries $\mathcal{B}_1$, $\mathcal{B}_2$, $\mathcal{B}_3$ against collision resistance of $\mathcal{H}$, core lemma of [60] and $\mathcal{M}^\top$-$\mathsf{MDDH}_{\mathbb{G}_1}$ assumption such that*

$$\mathsf{Adv}_{USS}(\mathcal{A}) \leq \mathsf{Adv}_{\mathsf{CR}}(\mathcal{B}_1) + \mathsf{Adv}_{\mathsf{CL}}(\mathcal{B}_2) + 2\mathsf{Adv}_{\mathcal{M}^\top\text{-}\mathsf{MDDH}_{\mathbb{G}_1}}(\mathcal{B}_3) + \frac{Q}{p}.$$

Proof. We proceed via changes of games starting with Game_0 that is the real USS game of definition 28. Let Adv_i be the advantage of adversary $\mathcal{A}$ winning Game_i.

- Game_1 is the same as Game_0 except the simulator computes the element $[u]_1$ as $[\boldsymbol{x}]_1(\boldsymbol{k}_{1,0} + \tau\boldsymbol{k}_{1,1}) + [\boldsymbol{y}]_1(\boldsymbol{k}_{2,0} + \tau\boldsymbol{k}_{2,1}) + [\boldsymbol{t}^\top]_1\boldsymbol{k}$ and verification of final adversary's message $(\boldsymbol{w}^*, [\boldsymbol{x}^*]_1, [\boldsymbol{y}^*]_1, [\pi^*]_1, \tilde{\tau}^*)$ checks:
  - $(\boldsymbol{w}^*, [\boldsymbol{x}^*]_1, [\boldsymbol{y}^*]_1) \in \mathcal{L}^{\mathsf{Lin}}_{\text{NO}}$,
  - $([\boldsymbol{x}^*]_1, [\boldsymbol{y}^*]_1) \notin \mathcal{Q}_{sim}$,
  - receives $\tilde{\tau}^*$, and checks that $\tilde{\tau}^* \notin \mathcal{Q}_{tag}$. With overwhelming probability, by the collision resistance of H, this implies that $\tau^* = H([\boldsymbol{x}^*]_1, [\boldsymbol{y}^*]_1, [\boldsymbol{t}^*]_1, [\pi^*_{\mathsf{or}}]_{1,2}, \tilde{\tau}^*)$ is also different from all the tags used in the simulated proofs.

The new element $[u]_1$ just differs on the element $[\boldsymbol{t}^\top]_1\boldsymbol{k}$, which in Game_0 is $\boldsymbol{s}^\top[\boldsymbol{p}]_1$, they pass verification with same probability because they are equivalent by definition. Thus,

$$|\mathsf{Adv}_0 - \mathsf{Adv}_1| \leq \mathsf{Adv}_{\mathsf{CR}}(\mathcal{B}_1).$$

- Game_2 is the same as Game_1 except that the key $\boldsymbol{k}$ is changed to $\boldsymbol{k}+\mathsf{RF}(\cdot)$ where $\mathsf{RF} : \mathbb{Z}_p \to \mathbb{Z}_p^{2k}$ is a random function. Concretely, the element $[\boldsymbol{p}]_1 = [\mathbf{A}_0^\top \boldsymbol{k}]_1$ is switched to $[\boldsymbol{p}]_1 = [\mathbf{A}_0^\top(\boldsymbol{k} + \mathsf{RF}(0))]$ in K and the element $[u]_1$ in S is computed as $[u]_1 = [(\boldsymbol{x}_i, \boldsymbol{y}_i)(\boldsymbol{k}_0+\tau_i\boldsymbol{k}_1)+\boldsymbol{t}_i^\top(\boldsymbol{k}+\mathsf{RF}(i))]_1$ for the i-th query. Moreover, the verifier V defines the set $\mathcal{S} = \{[(\boldsymbol{x}^*, \boldsymbol{y}^*)(\boldsymbol{k}_0 + \tau^*\boldsymbol{k}_1) + \boldsymbol{t}^{*\top}(\boldsymbol{k} + \mathsf{RF}(j^*))]_1\}_{j^*=0}^{Q}$ and checks $[u^*]_1 \in \mathcal{S}$. The indistinguishability between Game_1 and Game_2 is direct from the core lemma [5] because it is equivalent of indistinguishability between both MACs defined in the core lemma, thus

$$|\mathsf{Adv}_1 - \mathsf{Adv}_2| \leq \mathsf{Adv}_{\mathsf{CL}}(\mathcal{B}_2).$$

- Game_3 is the same as Game_2 except that the elements $[\boldsymbol{p}_0]_1 = [\mathbf{M}^\top\boldsymbol{k}_{1,0} + \mathbf{N}^\top\boldsymbol{k}_{2,0}]_1$ and $[\boldsymbol{p}_1]_1 = [\mathbf{M}^\top\boldsymbol{k}_{1,1} + \mathbf{N}^\top\boldsymbol{k}_{2,1}]_1$ are switched to $[\boldsymbol{p}_0]_1 = [\boldsymbol{z}_0 + \mathbf{N}^\top\boldsymbol{k}_{2,0}]_1$ and $[\boldsymbol{p}_1]_1 = [\boldsymbol{z}_1 + \mathbf{N}^\top\boldsymbol{k}_{2,1}]_1$ in K, where $\boldsymbol{z}_0, \boldsymbol{z}_1 \leftarrow \mathbb{Z}_p^n$. We can think in an intermediate game where we just switch $[\boldsymbol{p}_0]_1$, then for any adversary $\mathcal{B}_3$ able to distinguish between these intermediate games and Game_2 is breaking $\mathcal{M}^\top$-$\mathsf{MDDH}_{\mathbb{G}_1}$. By the same argument, $\mathcal{B}_3$ distinguishing between the intermediate game and Game_3 is breaking $\mathcal{M}^\top$-$\mathsf{MDDH}_{\mathbb{G}_1}$. Finally,

$$|\mathsf{Adv}_2 - \mathsf{Adv}_3| \leq 2\mathsf{Adv}_{\mathcal{M}^\top\text{-}\mathsf{MDDH}_{\mathbb{G}_1}}(\mathcal{B}_3).$$

Before studying the probability of the adversary $\mathcal{A}$ wins the Game_3, note that by linearity, we observe that the proof π^* is a valid proof to prove membership in the linear space of the vector $([\mathbf{0}]_1, [\boldsymbol{y}^*]_1)$. For any adversary that makes a proof $[\pi^*]_1$ for $(\boldsymbol{w}^*, [\boldsymbol{x}^*]_1, [\boldsymbol{y}^*]_1) \in \mathcal{L}_{\mathsf{NO}}^{\mathsf{Lin}}$, the element $[\overline{u}^*]_1 = [u^*]_1 - \boldsymbol{w}^*[\boldsymbol{p}_0]_1 - \boldsymbol{w}^*[\boldsymbol{p}_1]_1\tau^*$ is a valid proof for $([\mathbf{0}^*]_1, [\boldsymbol{y}^* - \overline{\boldsymbol{y}}]_1)$ where $\overline{\boldsymbol{y}} = \mathbf{N}\boldsymbol{w}^*$ (with same $[\boldsymbol{t}^*]_1$ and $[\pi_{\mathsf{or}}]_{1,2}$).

Now, we use an information-theoretic argument to bound the probability of success of the adversary $\mathcal{A}$. In the first place, we study what is leaked about the secret keys. The elements $[\boldsymbol{p}_0]_1 = [\boldsymbol{z}_0 + \mathbf{N}^\top\boldsymbol{k}_{2,0}]_1$, $[\boldsymbol{p}_1]_1 = [\boldsymbol{z}_1 + \mathbf{N}^\top\boldsymbol{k}_{2,1}]_1$ in the crs do not leak information about $\mathbf{N}^\top\boldsymbol{k}_{2,0}$ and $\mathbf{N}^\top\boldsymbol{k}_{2,1}$ because the vectors $[\boldsymbol{z}_0]_1$, $[\boldsymbol{z}_1]_1$ hide completely the projections by $\mathbf{N}$. Then, the element $\boldsymbol{y}^{*\top}(\boldsymbol{k}_{2,0} + \tau^*\boldsymbol{k}_{2,1})$ in the proof, where $[\boldsymbol{y}^*]_1 \notin \mathrm{Span}[\mathbf{N}]_1$, is uniformly random in adversary's view.

The adversary $\mathcal{A}$ also learns the following projections of the secret keys from each query i: $\boldsymbol{x}_i^\top(\boldsymbol{k}_{1,0} + \tau_i\boldsymbol{k}_{1,1}) + \boldsymbol{y}_i^\top(\boldsymbol{k}_{2,0} + \tau_i\boldsymbol{k}_{2,1})$, but they are pairwise independent

and $\boldsymbol{y}_i \neq \boldsymbol{y}^*$ for all $i = 1, \ldots, Q$. So, given $\boldsymbol{x}_i^\top(\boldsymbol{k}_{1,0} + \tau_i \boldsymbol{k}_{1,1}) + \boldsymbol{y}_i^\top(\boldsymbol{k}_{2,0} + \tau_i \boldsymbol{k}_{2,1})$ from the i-th query, the term $\boldsymbol{y}^{*\top}(\boldsymbol{k}_{2,0} + \tau^* \boldsymbol{k}_{2,1})$ in the proof is distributed uniformly at random. Thus, the probability of $\mathcal{A}$ computes this term and passes verification is $1/p$. Finally, taking into account there are Q simulated proofs, we have

$$|\mathsf{Adv}_3(\mathcal{A})| = \frac{Q}{p}.$$

□

4.7.2 Tight USS $\mathsf{Lin}_{\mathcal{D}_k}$ QA-NIZK

The QA-NIZK argument in Figure 4.5 is the Tight USS QA-NIZK argument for membership in linear spaces of Abe et al. [5] written in blocks for promise problem languages $\mathcal{L}^{\mathsf{Lin}}_{\text{YES}}$ and $\mathcal{L}^{\mathsf{Lin}}_{\text{NO}}$ defined in Section 4.7. It is the straightforward construction from the tight DV QA-NIZK of the previous Section 4.7.1 to public verifier QA-NIZK with pairings.

The security proof is analogous to the security proof of the tight QA-NIZK of Abe et al. [5]. In that construction, the authors give a tight reduction where the advantage of breaking the USS of the QA-NIZK is bounded by the advantage of breaking USS of the DV QA-NIZK and a kernel assumption. As we have seen in Section 4.7.1 the USS of our DV QA-NIZK is proven by a tight reduction that is linear in $\log Q$, where Q is the number of simulated queries. So, the USS of the QA-NIZK argument presented here inherits the same tightness loss linear in $\log Q$.

The bilinear QA-NIZK argument of Section 4.5 is a membership proof in linear spaces in two groups $\mathbb{G}_1$, $\mathbb{G}_2$, for the same languages as defined in 4.6.2. It is easily constructed from the bilinear version of the DV QA-NIZK argument 4.7.1. The reduction is analogous to the unilateral QA-NIZK reduction. We bound the advantage of breaking USS of the QA-NIZK for bilateral spaces by the advantage of breaking the USS of DV QA-NIZK for bilateral spaces and the $\mathsf{SKerMDH}$ assumption, with same tightness loss linear in $\log Q$.

4.8 Adapting GS Proofs for Improved Efficiency

In this section we show how to add zero-knowledge to the circuit satisfiability proof. A naive use of GS proofs results in a considerable overhead.

More concretely, we need to prove many quadratic Pairing Product Equations (PPEs), i.e. equations with variables in $\mathbb{G}_1$ and $\mathbb{G}_2$. Recall that GS proofs have a commit-and-prove structure: first, given an equation, the prover commits to the witness (a solution

$\underline{\mathsf{K}\,(gk, [\mathbf{M}]_1, [\mathbf{N}]_1)}$:
$\mathbf{A}_0, \mathbf{A}_1 \leftarrow \mathcal{D}_{2k,k}$,
$\mathsf{crs}_{\mathsf{or}} \leftarrow \mathsf{K}(gk, \mathbf{A}_0, \mathbf{A}_1)$
$H \leftarrow \mathcal{H}, \mathbf{A} \leftarrow \mathcal{D}_k$
$\mathbf{K} \leftarrow \mathbb{Z}_p^{2k\times k}, m = \ell_1 + \ell_2$,
for $i = 0, 1$:
$\mathbf{K}_i = \left(\overline{\mathbf{K}_i}, \underline{\mathbf{K}_i}\right)^\top \leftarrow \mathbb{Z}_p^{m\times(k+1)}$,
$\overline{\mathbf{K}_i} \in \mathbb{Z}_p^{\ell_1\times(k+1)}, \underline{\mathbf{K}_i} \in \mathbb{Z}_p^{\ell_2\times(k+1)}$.
$[\mathbf{P}]_1 = [\mathbf{A}_0^\top \mathbf{K}]_1 \in \mathbb{G}_1^{k\times(k+1)}$
$[\mathbf{P}_0]_1 = [\mathbf{M}^\top \overline{\mathbf{K}_0} + \mathbf{N}^\top \underline{\mathbf{K}_0}]_1 \in \mathbb{G}_1^{n\times(k+1)}$
$[\mathbf{P}_1]_1 = [\mathbf{M}^\top \overline{\mathbf{K}_1} + \mathbf{N}^\top \underline{\mathbf{K}_1}]_1 \in \mathbb{G}_1^{n\times(k+1)}$,
$\mathbf{C} = \mathbf{K}\mathbf{A} \in \mathbb{Z}_p^{2k\times k}$,
$\mathbf{C}_0 = \mathbf{K}_0\mathbf{A}, \mathbf{C}_1 = \mathbf{K}_1\mathbf{A} \in \mathbb{Z}_p^{m\times k}$
$\mathsf{crs} = (\mathsf{crs}_{\mathsf{or}}, [\mathbf{A}_0]_1, [\mathbf{P}]_1, [\mathbf{P}_0]_1, [\mathbf{P}_1]_1, [\mathbf{A}]_2,$
$[\mathbf{C}]_2, [\mathbf{C}_0]_2, [\mathbf{C}_1]_2, H)$,
$\mathsf{tr} = (\mathbf{K}_0, \mathbf{K}_1)$.

$\underline{\mathsf{P}\,(\mathsf{crs}, [\boldsymbol{x}]_1, [\boldsymbol{y}]_1, \boldsymbol{w}, \tilde{\tau})}$:
$\boldsymbol{s} \leftarrow \mathbb{Z}_p^k, [\boldsymbol{t}]_1 = [\mathbf{A}_0]_1\boldsymbol{s}$,
$[\pi_{\mathsf{or}}]_{1,2} \leftarrow \mathsf{P}_{\mathsf{or}}(\mathsf{crs}_{\mathsf{or}}, [\boldsymbol{t}]_1, \boldsymbol{s})$
$\tau = H([\boldsymbol{x}]_1, [\boldsymbol{y}]_1, [\boldsymbol{t}]_1, [\pi_{\mathsf{or}}]_{1,2}, \tilde{\tau}) \in \mathbb{Z}_p$
$[u]_1 = [\boldsymbol{w}^\top(\mathbf{P}_0 + \tau\mathbf{P}_1) + \boldsymbol{s}^\top\mathbf{P}]_1 \in \mathbb{G}_1^{k+1}$
Return $[\pi]_1 = ([\boldsymbol{t}]_1, [u]_1, [\pi_{\mathsf{or}}]_{1,2})$.

$\underline{\mathsf{V}\,(\mathsf{crs}, [\boldsymbol{x}]_1, [\boldsymbol{y}]_1, [\pi]_1, \tilde{\tau})}$:
Parse $[\pi]_1 = ([\boldsymbol{t}]_1, [u]_1, [\pi_{\mathsf{or}}]_{1,2})$,
$\tau = H([\boldsymbol{x}]_1, [\boldsymbol{y}]_1, [\boldsymbol{t}]_1, [\pi_{\mathsf{or}}]_{1,2}, \tilde{\tau}) \in \mathbb{Z}_p$,
Check $[\pi_{\mathsf{or}}]_{1,2}$ and
$\left[u^\top\right]_1 [\mathbf{A}]_2 = [\boldsymbol{x}^\top, \boldsymbol{y}^\top]_1 [\mathbf{C}_0 + \tau\mathbf{C}_1]$
$+[\boldsymbol{t}^\top]_1\mathbf{C}$
Return $0/1$.

$\underline{\mathsf{S}\,(\mathsf{crs}, [\boldsymbol{x}]_1, [\boldsymbol{y}]_1, \mathsf{tr}, \tilde{\tau})}$:
$\boldsymbol{s} \leftarrow \mathbb{Z}_p^k, [\boldsymbol{t}]_1 = [\mathbf{A}_0]_1\boldsymbol{s}$,
$[\pi_{\mathsf{or}}]_{1,2} \leftarrow \mathsf{P}_{\mathsf{or}}(\mathsf{crs}_{\mathsf{or}}, [\boldsymbol{t}]_1, \boldsymbol{s})$
$\tau = H([\boldsymbol{x}]_1, [\boldsymbol{y}]_1, [\boldsymbol{t}]_1, [\pi_{\mathsf{or}}]_{1,2}, \tilde{\tau}) \in \mathbb{Z}_p$,
$[u]_1 = [\boldsymbol{x}^\top, \boldsymbol{y}^\top]_1(\mathbf{K}_0 + \tau\mathbf{K}_1) + \boldsymbol{s}^\top[\mathbf{P}]_1$.

Figure 4.5: Tight QA-NIZK Argument for membership in linear spaces of Abe et al. [5] in blocks, $[\boldsymbol{x}, \boldsymbol{y}]_1 \in \mathrm{Im}[\mathbf{M}, \mathbf{N}]_1$, where $\mathbf{M} \in \mathbb{Z}_p^{\ell_1\times n}, \mathbf{N} \in \mathbb{Z}_p^{\ell_2\times n}$ and $\mathcal{H}$ a family of hash functions that are collision resistant. The scheme is modified to be tag-based. We use the disjunction argument or of [40] with $|\mathsf{crs}_{\mathsf{or}}| = (4n + 8)|\mathbb{G}_1| + (2\ell_1 + 3)|\mathbb{G}_2|$, $|\pi_{\mathsf{or}}| = 8|\mathbb{G}_1| + 3|\mathbb{G}_2|$.

to the equation, which is a vector $[\boldsymbol{x}]_1$ of elements in $\mathbb{G}_1$ and a vector $[\boldsymbol{y}]_2$ of elements in $\mathbb{G}_2$) and then it gives a proof that the committed values satisfy the equation. When trying to save group elements of the proof, we will save on the number of group elements necessary to commit to the witness.

We note that although there are several techniques to save on the "proof part" of GS proofs [86, 67] by aggregating proofs, they work for linear equations and not for quadratic.

In order to commit to the witness of satisfiability (a pair $[\boldsymbol{x}]_1, [\boldsymbol{y}]_2$) of an equation, individual commitments to each coordinate of these vectors are computed. We focus on the Symmetric EXternal Diffie-Hellman assumption instantiation of GS proofs for efficiency. Under this assumption, each individual commitment is either a dual-mode commitment based on DDH or an ElGamal ciphertext.

A natural idea to explore to reduce the commitment cost is to compute a single commitment to the whole vector $[\boldsymbol{x}]_1$ (and similarly for $[\boldsymbol{y}]_2$). This approach fails in general because GS proofs use some homomorphic properties of the commitments to combine them in a proof, and these are lost when using a single commitment to all of $[\boldsymbol{x}]_1$. This explains why, to the best of our knowledge, there is no technique to save on the commitment part of GS proofs which works *in general*, that is, for every set of equations of any form[6].

However, for the specific form of the equations we use in this paper, it is possible to exploit the specific form of the PPEs that we need to prove. More precisely, we can exploit that the equations, which depend on some group variables $\{L_i, R_i, O_i\}_{i=1}^d$ do not have cross terms, i.e. terms which multiply L_i with R_j, $i \neq j$.

More specifically, we show how to reduce the size of GS proofs for equations which can be written in this form:

$$e([k_j]_1, [1]_2) + e([x_j]_1, [y_j]_2) - e([w_j]_1, [1]_2) = e([h_j]_1, [b_j]_2), \qquad j = 1, \ldots, m \tag{4.12}$$

for some constants $[k_j]_1, [b_j]_2$, and variables x_j, y_j, w_j, h_j (in fact in our case b_j is the same for all equations, namely $t(s)$).

GS proofs use dual mode commitments to commit to the witness, meaning that commitments are either used in perfectly hiding or perfectly binding mode. To simulate proofs, the trapdoor is the equivocation trapdoor of the commitment scheme in *both* $\mathbb{G}_1$ and $\mathbb{G}_2$. However, for this particular type of equation it is enough to use standard ElGamal encryption for $\mathbb{G}_2$ (see [51]), the reason being that the equation admits

[6]What is important in the equation form for using simultaneous commitments is the structure of the quadratic part. On the other hand, this is independent of the equation type, i.e. this remark applies to multiscalar multiplication or quadratic equations in the field as well.

the trivial solution in $\mathbb{G}_1$. That is, it is enough for commitments in $\mathbb{G}_2$ to be computationally hiding, it is not necessary that there is a setup mode in which they are perfectly hiding. This allows us to save on the proof size ($(2,4)$ elements per equation).

The idea to save on the number of commitments is to reuse the randomness and encrypt all the variables $\boldsymbol{x}$, (resp. $\boldsymbol{y}, \boldsymbol{z}, \boldsymbol{w}$) with a single vector of commitments. This reduces the size of the commitments from $2m$ to $m+1$ for committing to each of the variable vectors. We define the commitment key in $\mathbb{G}_1^{m+1}$ as:

$$U = (\boldsymbol{u}_1, \boldsymbol{u}_2), \text{ where } \boldsymbol{u}_1 \leftarrow \mathcal{U}_{m+1,1}, \boldsymbol{u}_2 = \tau \boldsymbol{u}_1, \tau \leftarrow \mathbb{Z}_p.$$

and the commitment as:

$$\mathsf{Com}_U([\boldsymbol{x}]_1, \boldsymbol{r}) = \left[\begin{pmatrix}\boldsymbol{x}\\ 0\end{pmatrix}\right]_1 + r_1[\boldsymbol{u}_1]_1 + r_2[\boldsymbol{u}_2]_1,$$

where $\boldsymbol{r} \in \mathbb{Z}_p^2$ and $\mathcal{U}_{m+1,1}$ is the uniform distribution of vectors of $\mathbb{Z}_p^{m+1}$.

On the other hand, in $\mathbb{G}_2$ the commitment key is defined as:

$$V = (\boldsymbol{v}_1), \text{ where } \mathbf{v} \leftarrow \mathcal{U}_{m+1,1},$$

and the commitment as

$$\mathsf{Com}_V([\boldsymbol{y}]_2, s) = \left[\begin{pmatrix}\boldsymbol{y}\\ 0\end{pmatrix}\right]_2 + s[\boldsymbol{v}]_2.$$

The idea is that a commitment $[\boldsymbol{z}_{\boldsymbol{y}}]_\iota$ to a vector $[\boldsymbol{y}]_\iota$ can be divided into small parts $[\boldsymbol{z}_{y_i}]_\iota$, such that each part is a commitment to y_i. More precisely, components $(i, m+1)$ are a commitment to $\boldsymbol{y}_i$ with the commitment key corresponding to the components of $(i, m+1)$ of $\boldsymbol{u}_1, \boldsymbol{u}_2$ (for commitments in $\mathbb{G}_1$) and of $\boldsymbol{v}$ (for commitments in $\mathbb{G}_2$). That is, commitment keys are: $\boldsymbol{u}_1^i = \begin{pmatrix} u_{1,i} \\ u_{1,m+1}\end{pmatrix}$ and $\boldsymbol{u}_2^i = \begin{pmatrix} u_{2,i} \\ u_{2,m+1}\end{pmatrix}$, and $\mathsf{Com}_U([x_i]_1, \boldsymbol{r}) = r_1[\boldsymbol{u}_1^i]_1 + r_2[\boldsymbol{u}_2^i]_1 + \left[\begin{pmatrix} x_i \\ 0\end{pmatrix}\right]_1$. Similarly, we can get a commitment to $[y_i]_2$ by getting the components $(i, m+1)$ of a commitment in $\mathbb{G}_2$ with respect to the key $\boldsymbol{v}^i = \begin{pmatrix} v_i \\ v_{m+1}\end{pmatrix}$.

Therefore, we can now prove the equation i with different commitments keys, that is, it is as if we were using a different GS common reference string for each equation, namely, the keys $\boldsymbol{u}_1^i, \boldsymbol{u}_2^i, \boldsymbol{v}^i$.

The form of the jth verification equation is:

$$e(\left[\boldsymbol{z}^j_{\boldsymbol{x}}\right]_1, \left[\boldsymbol{z}^j_{\boldsymbol{y}}\right]_2) - e(\left[\boldsymbol{z}^j_{\boldsymbol{w}}\right]_1, [\boldsymbol{z}_1]_2)$$
$$= e(\left[\boldsymbol{z}^j_{\boldsymbol{h}}\right]_1, \left[\boldsymbol{z}_{b_j}\right]_2) + \sum_{i=1}^{2} e([\boldsymbol{u}^j_i]_1, [\boldsymbol{\pi}_{i,j}]_2) + e([\boldsymbol{\theta}_j]_1, [\boldsymbol{v}^j]_2),$$

where $\boldsymbol{z}^j_{\boldsymbol{\alpha}}$ is the result of keeping the jth and the $(m+1)$th coordinate of the commitment to vector $\boldsymbol{\alpha}$ and $\boldsymbol{z}_1 = \begin{pmatrix}1\\0\end{pmatrix}$, $\boldsymbol{z}_{\boldsymbol{b}_j} = \begin{pmatrix}b_j\\0\end{pmatrix}$, for $j = 1, \ldots, m$.

Soundness obviously holds because the partial commitment keys define perfectly binding commitments, so the same argument as in GS proofs applies.

On the other hand, one can claim computational witness indistinguishability under the DDH Assumption in $\mathbb{G}_1$. Indeed, in the security proof of witness indistinguishability, after the setup of the common reference string, the adversary can choose two witnesses $W_0 = ([\boldsymbol{x}_0]_1, [\boldsymbol{y}_0]_2, [\boldsymbol{w}_0]_1, [\boldsymbol{h}_0]_1)$, and $W_1 = ([\boldsymbol{x}_1]_1, [\boldsymbol{y}_1]_2, [\boldsymbol{w}_1]_1, [\boldsymbol{h}_1]_1)$, and receive a proof for W_b, $b \leftarrow \{0,1\}$.

We define a sequence of games, $\{\mathsf{Game}_{i,0}, \mathsf{Game}_{i,1}, \mathsf{Game}_{i,2}\}_{i=1}^{m}$.

1. In $\mathsf{Game}_{i,0}$ the commitment key is changed to define a perfectly hiding commitment to the ith coordinate of $\mathbb{G}_1$, as $\boldsymbol{u}_2 = \tau \boldsymbol{u}_1 + \boldsymbol{e}_i$, where $\boldsymbol{e}_i$ is the ith vector in the canonical basis of $\mathbb{Z}_p^{m+1}$.

2. In $\mathsf{Game}_{i,1}$ the challenger samples a bit b but uses the witness $W^*_{i,b}$ to create the proof, where $W^*_{i,b} = ([\boldsymbol{x}_{i,b}]_1, [\boldsymbol{y}_b]_2, [\boldsymbol{w}_{i,b}]_1, [\boldsymbol{h}_{i,b}]_2)$ and $[\boldsymbol{x}_{i,b}]_1, [\boldsymbol{w}_{i,b}]_1, [\boldsymbol{h}_{i,b}]_1$ are the same as $[\boldsymbol{x}_b]_1, [\boldsymbol{w}_b]_1, [\boldsymbol{h}_b]_1$ replacing the first i coordinates with 0.

3. In $\mathsf{Game}_{i,2}$ the coordinate i is changed to define a perfectly binding commitment in $\mathbb{G}_1$, as $\boldsymbol{u}_2 = \tau \boldsymbol{u}_1$.

At the end of the sequence of Games, the part in $\mathbb{G}_1$ of the witness is changed to the all zero vector, and is independent of b.

To complete the proof, we observe that the equation is left simulatable. This means that, in particular, using the properties of GS proofs it is possible to compute a valid proof of the equation given a commitment to the part of the witness of $\mathbb{G}_2$, without knowing an opening. For this reason, in the last m games we can switch to the all-zero witness for the elements in $\mathbb{G}_2$ based on the IND-CPA security of ElGamal, namely based on the DDH Assumption in $\mathbb{G}_2$.

This argues Witness Indistinguishability, which is all we need for our Signature of Knowledge, although ZK follows immediately from the fact that the equations are trivially satisfiable.

This strategy adds to the crs $2(m-1)$ elements in $\mathbb{G}_1$ and $m-1$ in $\mathbb{G}_2$, and, as explained, this reduces the cost of committing to the witness from $3 \cdot 2m$ elements in $\mathbb{G}_1$ and $2m$ in $\mathbb{G}_2$ to $3(m+1)$ in $\mathbb{G}_1$ and $m+1$ in $\mathbb{G}_2$.

Chapter 5

Simulation Extractable zk-SNARK for Circuit SAT

This chapter is based on the full version of our result *Simulation Extractable Versions of Groth's zk-SNARK Revisited* published in CANS 2020.

5.1 Introduction

As we already mentioned, Zero-knowledge Non-Interactive Succinct Arguments of Knowledge (zk-SNARKs) are among the most interesting NIZK proof systems in practice, as they allow to generate very short proofs for NP complete languages and, consequently, they are also very efficient to verify ([61, 72]). zk-SNARKs have had a tremendous impact in cryptographic practice and they have found numerous applications, including verifiable computation systems [109], privacy-preserving (PP) cryptocurrencies [18], PP smart contract systems [94], PP proof-of-stake protocols [90], and efficient ledger verification protocols [24], are some of the best known applications that use zk-SNARKs to prove different statements very efficiently while guaranteeing the privacy of the prover. Because of their practical importance, particularly in large-scale applications like blockchains, even minimal savings (especially in proof size or verification cost) are considered to be relevant.

In 2016, Groth [72] introduced the most efficient zk-SNARK for Quadratic Arithmetic Programs or QAPs, which is still the state-of-the-art, Groth16. Its proof is 3 group elements and the cost of verification is dominated by 3 pairing computations. In the original paper, it is proven to achieve knowledge soundness in the generic

group model (GGM). As we mentioned in Section 1.3.2, the proof of Groth16 is malleable, as it is shown in [74]. Generating non-malleable proofs is a necessary requirement in building various cryptographic schemes, including *universally composable* protocols [94, 90], cryptocurrencies (e.g. Zcash) [18], signature-of-knowledge schemes [74], etc. Therefore, in practice, it is important to have a stronger notion of knowledge soundness, known as (strong) simulation extractability (SE). This notion guarantees that a valid witness can be extracted from any adversary producing a proof accepted by the verifier, even after seeing an arbitrary number of simulated proofs.

There have been considerable efforts to refine Groth's zk-SNARK to achieve SE and guarantee the non-malleability of proofs. Firstly, in 2017 Groth and Maller [74] proposed a SE zk-SNARK, which is very efficient in terms of proof size but very inefficient in terms of Common Reference String (crs) size and prover time. They also showed how one can use SE zk-SNARKs to build Signature of Knowledge (SoK) schemes [36] with *succinct* signatures. In 2018, Bowe and Gabizon [26] proposed a less efficient construction in terms of proof size (5 group elements vs 3 in the original version) based on Groth16 which needs a Random Oracle (RO) (apart from GGM), but with almost no overhead in the crs size or additional cost for the prover. Last year, Lipmaa [102] proposed several constructions, including the most efficient QAP-based SE zk-SNARK in terms of proof size and with the same verification complexity as [74, 26], but less efficient in terms of crs size and prover time compared to [26]. In [11], Atapoor and Baghery used the traditional OR technique to achieve SE in Groth16. Their variant requires 1 paring less for verification in comparison with previous SE constructions, however it comes with an overhead in proof generation, crs, and even larger overhead in proof size. For a particular instantiation they add ≈ 52.000 constrains to the underlying QAP instance, which adds fixed overhead to the prover and crs, that can be considerable for mid-size circuits. They show that for a circuit with 10×10^6 Multiplication (Mul) gates, their prover is about 10% slower, but it can be slower for circuits with less than 10×10^6 gates.

Recently, Baghery, Kohlweiss, Siim, and Volkhov [15] explore another direction. Instead of modifying Groth16 to achieve strong SE, they first show that the original construction of Groth16 achieves weak SE with white-box extraction. Weak SE allows proof randomization, while it guarantees that a proof cannot be changed to prove a new statement. Then, considering the first result, they propose two efficient constructions of Groth16 that achieve weak SE with *black-box* extraction. Both *weak* and *strong* SE zk-SNARKs can be lifted to achieve black-box simulation extractability with a simple compiler [12, 15]. However, to realize the standard ideal functionality defined for NIZK arguments, one would need to use a strong SE NIZK with black-box extraction [70].

SNARK	Model	crs	Prover	Proof	Verifier
Groth [72]	GGM	$m+2n-l\mathbb{G}_1$ $n\mathbb{G}_2$	$m+3n-l\ E_1$ $n\ E_2$	$2\ \mathbb{G}_1$ $1\ \mathbb{G}_2$	$l\ E_1$ 3 P
GM [74]	GGM	$2m+4n\ \mathbb{G}_1$ $2n\ \mathbb{G}_2$	$2m+4n-l\ E_1$ $2n\ E_2$	$2\ \mathbb{G}_1$ $1\ \mathbb{G}_2$	$l\ E_1$ 5 P
BG [26]	GGM, ROM	$m+2n-l\ \mathbb{G}_1$ $n\ \mathbb{G}_2$	$m+3n-l\ E_1$ $n\ E_2$	$3\ \mathbb{G}_1$ $2\ \mathbb{G}_2$	$l\ E_1$ 5 P
AB [11]	GGM	$m'+2n'-l\ \mathbb{G}_1$ $n'\ \mathbb{G}_2$	$m'+3n'-l\ E_1$ $n'\ E_2$	$4\ \mathbb{G}_1$ $2\mathbb{G}_2+2\lambda$	$l'+2\ E_1$ 4 P
Lipmaa [103]	AGM, Tag-based	$m+3n-l\ \mathbb{G}_1$ $n\ \mathbb{G}_2$	$m+4n-l\ E_1$ $n\ E_2$	$3\ \mathbb{G}_1$ $1\ \mathbb{G}_2$	$l+1\ E_1$ 5 P
Section 5.3	GGM, ROM	$m+2n-l\ \mathbb{G}_1$ $n\ \mathbb{G}_2$	$m+3n-l\ E_1$ $n\ E_2$	$3\ \mathbb{G}_1$ $2\ \mathbb{G}_2$	$l\ E_1$,1 E_2 4 P
Section 5.4	GGM, CRH	$m+2n-l\ \mathbb{G}_1$ $n\ \mathbb{G}_2$	$m+3n-l\ E_1$ $n\ E_2$	$3\ \mathbb{G}_1$ $2\ \mathbb{G}_2$	$l\ E_1$, 1 E_2 1 E_T, 4 P

Table 5.1: In the first row we have the Groth's zk-SNARK, Groth16. In the following rows we show a omparison of our proposed variations of Groth16 along with the other SE zk-SNARKs for arithmetic circuit satisfiability with n Mul gates (constraints) and m wires (variables), of which l are public input wires (variables). A typical set of values is $n = m = 10^6$ and $l = 10$. In the case of crs size and prover's computation we omit constants. In [74], n Mul gates and m wires translate to $2n$ squaring gates and $2m$ wires. In [11], SE is achieved with an OR approach which requires to add constraints and variables, resulting in $n' \approx n + 52.000$, $m' \approx m + 52.000$, and $l' = l + 4$. $\mathbb{G}_1, \mathbb{G}_2$ and $\mathbb{G}_T$: group elements, E_i: exponentiation in group $\mathbb{G}_i$, M_i: multiplication in group $\mathbb{G}_i$, P: pairings. GGM: Generic Group Model, ROM: Random Oracle Model, AGM: Algebraic Group Model, CRH: Collision Resistant Hash.

5.1.1 Our Contributions

In this work, we revise the simulation extractable variants of Groth16, presented in [26] and [11], to get the best of both constructions.

Our focus is mainly on Bowe and Gabizon's variation [26] which has the most efficient prover and the shortest crs among other SE zk-SNARKs [74, 26, 103, 11], while requires a RO. To achieve (strong) simulation extractability, their prover replaces all the original computations which depend on some parameter δ given in the crs by some δ' and the prover must give $[\delta']_2$ and a proof of knowledge (PoK) of the DLOG of $[\delta']_2$ w.r.t $[\delta]_2$.

Using the same approach [26], we construct two *strong* SE zk-SNARKs that are the

most efficient simulation extractable variants of Groth16 in terms of crs size, prover complexity, and verification. Both zk-SNARKs use some sophisticated modification of Boneh-Boyen signatures [22] to prove knowledge of the DLOG of δ' which require 1 pairing less in the verification in comparison with the argument in Bowe and Gabizon's construction. The first construction uses non-programmable RO, while in the second construction, in the cost of a single new element in the crs and a collision-resistant hash function, we get rid of the RO and similar to Groth16, prove the security of construction in the GGM model.

Tab. 5.1 presents a comparison of our proposed variants of Groth16 with several other constructions for a particular instance of arithmetic circuit satisfiability. As it can be seen, in comparison with [26], both our constructions require 1 paring less in the verification, while retaining all the other properties of their construction.

The second construction avoids using ROs, in the cost of a single new element in the crs which is negligible in practice. In comparison with [11], both of our variants have a negligible overhead in the proof generation and crs size, and they both also come with smaller overhead in proof size. [1] Among two proposed variants, both constructions require 4 parings in the verification, however considering the number of exponentiations, we expect to have a slightly faster verification in the first construction, presented in Section 5.3.

Finally, we highlight that using the technique proposed in [74], both the proposed SE zk-SNARKs can be used to build *succinct* SoK schemes, which would be more efficient than previous constructions. In general, due to relying on non-falsifiable assumptions, succinct SoK schemes have better efficiency in comparison with the constructions that are built under standard assumptions [36, 19, 14]. We also note that to achieve strong (white-box) SE, our proposed zk-SNARKs require minimal changes in comparison with the original Groth16, particularly the proof generation and proof verification of Groth16 is a part of the proof generation and verification in our protocols. Therefore, one can use the same compiler or ad-hoc approach proposed in [12] and [15], respectively, to construct a more efficient strong *black-box* SE zk-SNARK.

5.1.2 Organization

In Section 5.2, we introduce the relevant security definitions. In Section 5.3, we give our first SE zk-SNARK from non-programmable RO in the GGM, and in Section 5.4 our second SE zk-SNARK in GGM without RO.

[1] In the worst case, our changes add only one element to the crs of Groth16 and since Groth16 is already proven to achieve subversion ZK (ZK without trusting a third party) [4, 56], our variants also can be proven to achieve Sub-ZK using the technique proposed in [13].

5.2 Preliminaries

We use the definitions of NIZK arguments from [72]. Let $\mathcal{G}_{\mathbf{R}}$ be a relation generator, such that $\mathcal{G}_{\mathbf{R}}(\lambda)$ returns a polynomial-time decidable binary relation $\mathbf{R} = \{(\boldsymbol{x}, \boldsymbol{w})\}$. Here, $\boldsymbol{x}$ is the statement and $\boldsymbol{w}$ is the witness. Security parameter λ can be deduced from the description of $\mathbf{R}$. The relation generator also outputs auxiliary information $\mathsf{z}_{\mathbf{R}}$ that will be given to the honest parties and the adversary. As in [72], $\mathsf{z}_{\mathbf{R}}$ is the value returned by $\mathcal{G}(1^{\lambda})$, and is given as an input to the parties.

Let $\mathcal{L}_{\mathbf{R}} = \{\boldsymbol{x} : \exists \boldsymbol{w}, (\boldsymbol{x}, \boldsymbol{w}) \in \mathbf{R}\}$ be an NP-language. A *NIZK argument system* for $\mathcal{G}_{\mathbf{R}}$ consists of tuple of PPT algorithms $(\mathsf{K}, \mathsf{P}, \mathsf{V}, \mathsf{S})$, such that:

CRS Generator: K is a PPT algorithm that, given $(\mathbf{R}, \mathsf{z}_{\mathbf{R}})$ where $(\mathbf{R}, \mathsf{z}_{\mathbf{R}}) \in \mathbf{Im}(\mathcal{G}_{\mathbf{R}}(\lambda))$, outputs $\mathsf{crs} := (\mathsf{crs}_{\mathsf{P}}, \mathsf{crs}_{\mathsf{V}})$ and stores trapdoors of crs as ts. We distinguish $\mathsf{crs}_{\mathsf{P}}$ (needed by the prover) from $\mathsf{crs}_{\mathsf{V}}$ (needed by the verifier).

Prover: P is a PPT algorithm that, given $(\mathbf{R}, \mathsf{z}_{\mathbf{R}}, \mathsf{crs}_{\mathsf{P}}, \boldsymbol{x}, \boldsymbol{w})$, if $(\boldsymbol{x}, \boldsymbol{w}) \in \mathbf{R}$, outputs an argument π; otherwise, it outputs $\perp$.

Verifier: V is a PPT algorithm that, given $(\mathbf{R}, \mathsf{z}_{\mathbf{R}}, \mathsf{crs}_{\mathsf{V}}, \boldsymbol{x}, \pi)$, returns either 0 (reject) or 1 (accept).

Simulator: S is a PPT algorithm that, given $(\mathbf{R}, \mathsf{z}_{\mathbf{R}}, \mathsf{crs}, \mathsf{ts}, \boldsymbol{x})$, outputs a simulated argument π.

Besides *succinct* proofs, i.e. polynomial in λ, an SE zk-SNARK is required to satisfy *completeness*, *simulation extractability*, and *zero-knowledge*.

Definition 30 (Perfect Completeness)**.** *A non-interactive argument is* perfectly complete for $\mathcal{G}_{\mathbf{R}}$*, if for all* λ*, all* $(\mathbf{R}, \mathsf{z}_{\mathbf{R}}) \in \mathbf{Im}(\mathcal{G}_{\mathbf{R}}(1^{\lambda}))$*, and* $(\boldsymbol{x}, \boldsymbol{w}) \in \mathbf{R}$*,*
$\Pr\left[\mathsf{crs} \leftarrow \mathsf{K}(\mathbf{R}, \mathsf{z}_{\mathbf{R}}),\ \pi \leftarrow \mathsf{P}(\mathbf{R}, \mathsf{z}_{\mathbf{R}}, \mathsf{crs}_{\mathsf{P}}, \boldsymbol{x}, \boldsymbol{w}) : \mathsf{V}(\mathbf{R}, \mathsf{z}_{\mathbf{R}}, \mathsf{crs}_{\mathsf{V}}, \boldsymbol{x}, \pi) = 1 \quad \right] = 1.$

Here, $\mathsf{z}_{\mathbf{R}}$ can be seen as a common auxiliary input to $\mathcal{A}$ that is generated by using a benign relation generator.

Definition 31 (Simulation Extractability [74])**.** *Let* $\mathsf{RND}_{\lambda}(\mathcal{A})$ *denote the random tape of* $\mathcal{A}$*. A non-interactive argument is* (strong) simulation-extractable for $\mathcal{G}_{\mathbf{R}}$*, if for any NUPPT* $\mathcal{A}$*, there exists a NUPPT extractor* $\mathsf{Ext}_{\mathcal{A}}$ *s.t. for all* λ*,*

$$\Pr\begin{bmatrix}(\mathbf{R}, \mathsf{z}_{\mathbf{R}}) \leftarrow \mathcal{G}_{\mathbf{R}}(1^{\lambda}), (\mathsf{crs} \,\|\, \mathsf{ts}) \leftarrow \mathsf{K}(\mathbf{R}, \mathsf{z}_{\mathbf{R}}), r \leftarrow \mathsf{RND}_{\lambda}(\mathcal{A}), \\ ((\boldsymbol{x}, \pi) \,\|\, \boldsymbol{w}) \leftarrow (\mathcal{A}^{\mathsf{O}(\mathsf{ts}, .)} \,\|\, \mathsf{Ext}_{\mathcal{A}})(\mathbf{R}, \mathsf{z}_{\mathbf{R}}, \mathsf{crs}; r) : \\ (\boldsymbol{x}, \pi) \notin Q \wedge (\boldsymbol{x}, \boldsymbol{w}) \notin \mathbf{R} \wedge \mathsf{V}(\mathbf{R}, \mathsf{z}_{\mathbf{R}}, \mathsf{crs}_{\mathsf{V}}, \boldsymbol{x}, \pi) = 1\end{bmatrix} = negl(\lambda).$$

Here, Q is the set of simulated statement-proof pairs. Note that *simulation extractability* implies *knowledge soundness*.

Definition 32 (Zero-Knowledge (ZK) [72])**.** *A non-interactive argument is* computationally ZK for $\mathcal{G}_{\mathbf{R}}$*, if for all* λ*, all* $(\mathbf{R}, \mathsf{z}_{\mathbf{R}}) \in \mathbf{Im}(\mathcal{G}_{\mathbf{R}}(1^{\lambda}))$*, and for all NUPPT* $\mathcal{A}$*,* $\varepsilon_0 \approx_c \varepsilon_1$*, where*

$$\varepsilon_b = \Pr[(\mathsf{crs} \,\|\, \mathsf{ts}) \leftarrow \mathsf{K}(\mathbf{R}, \mathsf{z}_{\mathbf{R}}) : \mathcal{A}^{\mathsf{O}_b(\cdot,\cdot)}(\mathbf{R}, \mathsf{z}_{\mathbf{R}}, \mathsf{crs}) = 1].$$

Here, the oracle $\mathsf{O}_0(\boldsymbol{x}, \boldsymbol{w})$ *returns* $\perp$ *(reject) if* $(\boldsymbol{x}, \boldsymbol{w}) \notin \mathbf{R}$*, and otherwise it returns* $\mathsf{P}(\mathbf{R}, \mathsf{z}_{\mathbf{R}}, \mathsf{crs}_{\mathsf{P}}, \boldsymbol{x}, \boldsymbol{w})$*. Similarly,* $\mathsf{O}_1(\boldsymbol{x}, \boldsymbol{w})$ *returns* $\perp$ *(reject) if* $(\boldsymbol{x}, \boldsymbol{w}) \notin \mathbf{R}$*, otherwise it returns* $\mathsf{S}(\mathbf{R}, \mathsf{z}_{\mathbf{R}}, \mathsf{crs}, \mathsf{ts}, \boldsymbol{x})$*. The argument is* perfect ZK for $\mathcal{G}_{\mathbf{R}}$ *if one requires that* $\varepsilon_0 = \varepsilon_1$*.*

5.3 A Simulation Extractable zk-SNARK in the ROM

As we discussed, the main idea in Bowe and Gabizon's [26] work to achieve simulation extractability is to replace all the computations which depend on some parameter δ given in the crs by some randomization of it, say δ', and the prover must give $[\delta']_2$ and a Proof of Knowledge (PoK) in the ROM of the Discrete Logarithm (DLOG) of $[\delta']_2$ w.r.t $[\delta]_2$. This makes it harder for the adversary to re-use elements from the simulated proofs that are created with the original parameter δ.

Our idea is to replace the PoK with a Boneh-Boyen signature. A nice feature of this construction inherited from [26] is that SE is achieved essentially without modifications in the crs or the prover complexity, or changes in the security model (which is still Generic Group Model and Random Oracle Model).

5.3.1 Scheme definition

In Fig. 5.1, we describe the proposed variation of Groth16 that can achieve SE. We highlight the changes in the new construction with gray background.

Our modification follows closely the one of Bowe and Gabizon [26], except that in their scheme $[d]_1 = [y]_1\zeta$ where $[y]_1 = H(A \parallel B \parallel C \parallel \delta')$ and their verification checks that $[\delta']_1[y]_2 = [d]_1[\delta]_2$, which requires 2 pairings. The security proof shows that this is a simulation extractable PoK of the DLOG of $[\delta']_2$ with respect to $[\delta]_2$. We follow the same idea but our approach embeds a Boneh-Boyen signature in the proof as argument of knowledge for this DLOG, which requires 1 pairing, instead of 2.

Setup, $\mathsf{crs} \leftarrow \mathsf{K}(\mathbf{R}, \mathsf{z}_{\mathbf{R}})$**:** Similar to the original scheme it picks $x, \alpha, \beta, \delta \leftarrow \mathbb{Z}_p^*$, $H \leftarrow \mathcal{H}$, and returns crs defined as the following (by considering the observation in [27] that γ in the original scheme can be set 1),

$$(\mathsf{crs}_{\mathsf{P}}, \mathsf{crs}_{\mathsf{V}}) := \mathsf{crs} \leftarrow \begin{pmatrix} [\alpha, \beta, \delta, \{x^i\}_{i=0}^{n-1}, \{u_j(x)\beta + v_j(x)\alpha + w_j(x)\}_{j=0}^{l}, \\ \{(u_j(x)\beta + v_j(x)\alpha + w_j(x))/\delta\}_{j=l+1}^{m}, \{x^i t(x)/\delta\}_{i=0}^{n-2}]_1, \\ [\beta, \delta, \{x^i\}_{i=0}^{n-1}]_2,\ [\alpha\beta, t(x)]_T, H \end{pmatrix}.$$

Prover, $\pi \leftarrow \mathsf{P}(\mathbf{R}, \mathsf{z}_{\mathbf{R}}, \mathsf{crs}_{\mathsf{P}}, \boldsymbol{x} = (a_1, \ldots, a_l), \boldsymbol{w} = (a_{l+1}, \ldots, a_m))$**:** assuming $a_0 = 1$, it acts as follows,

1. Selects a random element $\zeta \leftarrow \mathbb{Z}_p^*$, and sets $[\delta']_2 := \zeta[\delta]_2$
2. Let $A^\dagger(X) \leftarrow \sum_{j=0}^{m} a_j u_j(X)$, $B^\dagger(X) \leftarrow \sum_{j=0}^{m} a_j v_j(X)$, $C^\dagger(X) \leftarrow \sum_{j=0}^{m} a_j w_j(X)$,
3. Set $h(X) = \sum_{i=0}^{n-2} h_i X^i \leftarrow (A^\dagger(X)B^\dagger(X) - C^\dagger(X))/t(X)$,
4. Set $[h(x)t(x)/\delta]_1 \leftarrow \sum_{i=0}^{n-2} h_i [x^i t(x)/\delta']_1$,
5. Set $r_a \leftarrow \mathbb{Z}_p$; Set $[A]_1 \leftarrow \sum_{j=0}^{m} a_j [u_j(x)]_1 + [\alpha]_1 + r_a [\delta']_1$,
6. Set $r_b \leftarrow \mathbb{Z}_p$; Set $[B]_2 \leftarrow \sum_{j=0}^{m} a_j [v_j(x)]_2 + [\beta]_2 + r_b [\delta']_2$,
7. Set $[C]_1 \leftarrow r_b[A]_1 + r_a \left(\sum_{j=0}^{m} a_j [v_j(x)]_1 + [\beta]_1\right) + \sum_{j=l+1}^{m} a_j [(u_j(x)\beta + v_j(x)\alpha + w_j(x))/\delta']_1 + [h(x)t(x)/\delta']_1$,
8. Sets $m = H([A]_1 \parallel [B]_2 \parallel [C]_1 \parallel [\delta']_2)$, where $H : \{0,1\}^* \rightarrow \mathbb{Z}_p^*$ is a secure hash function,
9. Computes $[D]_1 = \frac{1}{\zeta + m}[t(x)/\delta]_1 = [\frac{t(x)}{\delta' + m\delta}]_1$
10. Return $\pi := ([A, C, D]_1, [B, \delta']_2)$.

Verifier, $\{1, 0\} \leftarrow \mathsf{V}(\mathbf{R}, \mathsf{z}_{\mathbf{R}}, \mathsf{crs}_{\mathsf{V}}, \boldsymbol{x} = (a_1, \ldots, a_l), \pi = ([A, C, D]_1, [B, \delta']_2))$**:** assuming $a_0 = 1$, and setting $m = H([A]_1 \parallel [B]_2 \parallel [C]_1 \parallel [\delta']_2)$ checks if

1. $[A]_1[B]_2 = [C]_1[\delta']_2 + \left(\sum_{j=0}^{l} a_j \left[u_j(x)\beta + v_j(x)\alpha + w_j(x)\right]_1\right)[1]_2 + [\alpha\beta]_T$
2. $[D]_1[\delta' + \delta m]_2 = [t(x)]_T$ (Note that: $[t(x)/\delta]_1[\delta]_2 = [t(x)]_T$)

and return 1 if both checks pass, otherwise return 0.

Simulator, $\pi \leftarrow \mathsf{S}(\mathbf{R}, \mathsf{z}_{\mathbf{R}}, \mathsf{crs}_{\mathsf{V}}, \boldsymbol{x} = (a_1, \ldots, a_l), \mathsf{ts})$**:** Given the simulation trapdoors $\mathsf{ts} := (\beta, \delta)$ acts as follows,

1. Choose random $\zeta \leftarrow \mathbb{Z}_p^*$ and set $\delta' := \zeta\delta$
2. Choose random $\delta' \leftarrow \mathbb{Z}_p^*$
3. Choose $A, B \leftarrow \mathbb{Z}_p$
4. Let $[C]_1 = \left[(A \cdot B - \sum_{j=0}^{l} a_j(u_j(x)\beta + v_j(x)\alpha + w_j(x)) - \alpha\beta)/\delta'\right]_1$
5. Let $m = H([A]_1 \parallel [B]_2 \parallel [C]_1 \parallel [\delta']_2)$
6. Set $[D]_1 = \dfrac{t(x)}{\delta' + m\delta}[1]_1$
7. Return $\pi := ([A]_1, [B]_2, [C]_1, [D]_1, [\delta']_2)$.

Figure 5.1: The proposed simulation-extractable variation of Groth16 for $\mathbf{R}$ along with a Boneh-Boyen signature. $\mathcal{H}$ is a family of collision resistant hash functions that maps to $\mathbb{Z}_p^*$. The element $[t(x)]_T$ is redundant and can be computed from the rest of the elements in the crs. Alternatively, one can describe Groth16 as corresponding to $\zeta = 1$ and where the proof consists only of $[A, C]_1, [B]_2$.

5.3.2 Security

A part from saving one pairing on verification with respect to [26], our scheme also has the nice property that the RO maps to elements in $\mathbb{Z}_p$ and it does not need the property that H can sample elements of $\mathbb{G}$ obliviously (i.e. soundness does not use that the DLOG of image elements is hard).

In a nutshell, we show that we can embed a Boneh-Boyen signature in the proof and this results in a SE argument of knowledge in the GGM and the ROM. Namely, the element $[1/(\delta' + \delta m)]_1$, which is a Boneh-Boyen signature of δm for public key $[\delta']_2$ can be constructed from $[1/\delta]_1$ for all $m \in \mathbb{Z}_p$, if and only if, the DLOG of δ' w.r.t δ is known. The adversary might be able to cheat for a specific m (i.e. if it sets $\delta' = k\delta - m^*\delta$ it can cheat for m^*) but the RO ensures that δ' cannot be set as a function of m. Given knowledge of the DLOG of δ', following the same blueprint as the proof of Bowe and Gabizon, we prove that the simulated queries are useless to the adversary. Then, we can easily conclude that the scheme is SE if Groth16 is knowledge sound.

Theorem 28 (Completeness, ZK, SE)**.** *The variation of Groth16 described in Fig. 5.1, guarantees 1) perfect completeness, 2) perfect zero-knowledge and 3) simulation-extractability in the asymmetric Generic Group Model and the Random Oracle Model.*

Proof. Perfect completeness and perfect zero-knowledge are obvious and the proof is omitted. Knowledge extractability is proven by reduction (in the GGM) to the knowledge soundness of Groth16. The reduction works in two steps (similarly to [26], although the proof of each of these steps is different):

Step 1 Extraction of the DLOG of δ'.

Step 2 Reduction to the Knowledge Soundnesss of Groth16.

Proof of Step 1) Suppose $\mathcal{A}$ has made a sequence of queries $\boldsymbol{x}_1, \cdots, \boldsymbol{x}_v$ to $\mathsf{S}(\mathsf{ts}, \cdot)$, and received answers $\{\pi_j = (A_j, B_j, C_j, D_j, \delta_j)\}_{j=1}^{v}$. Let Q' be the union of elements in the crs together with those from the replies of $\mathsf{S}(\mathsf{ts}, \cdot)$; namely,

$$Q' := \left(\begin{array}{l} \big[\alpha, \beta, \delta, \{x^i\}_{i=0}^{n-1}, \\ \{u_j(x)\beta + v_j(x)\alpha + w_j(x)\}_{j=0}^{l}, \\ \left\{ \dfrac{u_j(x)\beta + v_j(x)\alpha + w_j(x)}{\delta} \right\}_{j=l+1}^{m}, \\ \{x^i t(x)/\delta\}_{i=0}^{n-2}\big]_1, [\beta, \delta, \{x^i\}_{i=0}^{n-1}]_2 \end{array} \right) \cup \left(\begin{array}{c} \Big\{ \Big[A_j, C_j := \frac{A_j B_j - \mathsf{ic}_j - \alpha\beta}{\delta_j}, \\ D_j := \dfrac{t(x)}{\delta_j + m_j\delta} \Big]_1 \\ [B_j, \delta_j]_2, m_j \Big\}_{j=1}^{v} \end{array} \right)$$

where $\mathsf{ic}_j = \sum_{i=0}^{l} a_i^j (u_i(x)\beta + v_i(x)\alpha + w_i(x))$, $\boldsymbol{x}_j = (a_1^j, \ldots, a_l^j)$, and $m_j \in \mathbb{Z}_p$ the message that simulator receives from the RO for each A_j, B_j, C_j, δ_j.

Now, assume $\mathcal{A}$ has produced elements (A, B, C, D, δ') such that

$$A \cdot B \equiv C \cdot \delta' + \left(\sum_{j=0}^{l} a_j(u_j(x)\beta + v_j(x)\alpha + w_j(x))\right) + \alpha\beta$$

and, for $m := H(A \parallel B \parallel C \parallel \delta')$, $D(\delta' + \delta m) = t(x)$. Let Q'_1 be the set with the elements of Q' in $\mathbb{G}_1$ and Q'_2 the elements in $\mathbb{G}_2$. Since the adversary is generic it has constructed these elements as a linear combination of the elements in Q' which are in the relevant group (i.e. element of Q'_1 in $\mathbb{G}_1$ for A, C, D and of Q'_2 for B, δ') and we can extract the coefficients of this linear combination.

We will use the notation of $k_{x,q}$ in the following to denote the coefficient of the term q that appears in the expression of the element x.

First, we prove that the adversary has knowledge of the discrete logarithm of δ' w.r.t. δ. From the second verification equation, $D = \frac{t(x)}{\delta' + \delta m}$. On the other hand, from adversary $\mathcal{A}$ we can recover a vector $\boldsymbol{k}_D$ with the coefficients that it has used to construct D, that is, $D = \sum_{q \in Q'_1} k_{D,q}\ q$, and a vector $\boldsymbol{k}_{\delta'}$ with the coefficients that it has used to construct $\delta' = \sum_{q \in Q'_2} k_{\delta',q}\ q$.

We argue that $\delta' + \delta m$ cannot be a polynomial in x, i.e. $\delta' + \delta m$ is a linear combination of terms in Q'_1 without x. Indeed, if $\delta' + \delta m$ is a polynomial in x, then this polynomial must divide $t(x)$ because the adversary does not see any rational functions with x. Then, there exists a polynomial ν such that $\delta' + \delta m = (x - r)\nu$, for some r root of $t(x)$. However, since $\delta' + \delta m$ cannot have any terms $x\delta$ ($x\delta \notin Q'_2$), the only possibility is that ν does not have any term with δ, and neither $\delta' + \delta m$. This means that $\delta' = \delta'' - \delta m$, for some δ'' independent of δ. But since H is a RO, the probability that given δ' and δ'', m satisfies this relation, is $1/p$.

Therefore, x only appears in the numerator of the expression $D = \frac{t(x)}{\delta' + \delta m}$, and thus, we have

$$\frac{t(x)}{\delta' + \delta m} = k_{D,0}\frac{t(x)}{\delta} + \sum_{j=1}^{v} k_{D,j}\frac{t(x)}{\delta_j + m_j} \tag{5.1}$$

where, to simplify the notation, we define $k_{D,0} = k_{D,\frac{t(x)}{\delta}}$. $k_{D,j} = k_{D,\frac{t(x)}{\delta_j + m_j\delta}}$.

Defining $\delta_0 = \delta$, $m_0 = m$, then

$$\frac{t(x)}{\delta' + \delta m} = \sum_{j=0}^{v} k_{D,j} \frac{t(x)}{\delta_j + m_j \delta} \iff \frac{1}{\delta' + \delta m} = \sum_{j=0}^{v} k_{D,j} \frac{\prod_{i=0, i\neq j}^{v} (\delta_i + m_i \delta)}{\prod_{i=0}^{v} (\delta_i + m_i \delta)} \tag{5.2}$$

$$\iff \prod_{i=0}^{v} (\delta_i + m_i \delta) = (\delta' + \delta m) \Big(\sum_{j=0}^{v} k_{D,j} \prod_{i=0, i\neq j}^{v} (\delta_i + m_i \delta) \Big). \tag{5.3}$$

It follows that the term $\delta' + m\delta$ must divide the left side of the equation (5.2). Therefore, there exists some index j^* and $k \in \mathbb{Z}_p$ such that $\delta' + m\delta = k(\delta_{j^*} + m_{j^*}\delta)$. Now, dividing Eq. (5.2) by $(\delta_{j^*} + m_{j^*}\delta)$, we come to the following expression

$$\textstyle\prod_{i=0, i\neq j^*}^{v} (\delta_i + m_i \delta) = k \cdot \Big(k_{D,j^*} \prod_{i=0, i\neq j^*}^{v} (\delta_i + m_i \delta) + \sum_{j=0, j\neq j^*}^{v} k_{D,j} \prod_{i=0, i\neq j}^{v} (\delta_i + m_i \delta) \Big),$$

which is equivalent to

$$0 = (1 - k \cdot k_{D,j^*}) \textstyle\prod_{i=0, i\neq j^*}^{v} (\delta_i + m_i \delta) - \sum_{j=0, j\neq j^*}^{v} k \cdot k_{D,j} \prod_{i=0, i\neq j}^{v} (\delta_i + m_i \delta).$$

Since all summands are linearly independent polynomials, $k = k_{D,j^*}^{-1}$, and $k_{D,j} = 0$ if $j \neq j^*$. We distinguish two cases: (1) $\delta' + m\delta = k\delta$ ($j^* = 0$) or (2) $\delta' + m\delta = k(\delta_{j^*} + m_{j^*}\delta)$ ($j^* \neq 0$).

In case (1), we are done, as we can extract the DLOG of δ' as $k - m$.

In case (2), there exists some $k' \in \mathbb{Z}_p$ such that $\delta' = k\delta_{j^*} + k'\delta$ and $m = km_{j^*} - k'$. Since H is a RO, m is a uniform random element given δ', (and thus, given k, k',) and therefore the probability of this event is $1/p$.

Thus, the adversary cannot compute the elements of such a proof belonging to the Span(Q') unless it knows ζ.

Proof of Step 2) We show that the elements A, B, C do not use the elements of the simulated proofs, i.e. $\{[A_j]_1, [B_j]_2, [C_j]_1, [D_j]_1, [\delta_j]_2\}_{j=1}^{v}$, and then, with the knowledge of ζ such that $\delta' = \zeta\delta$, we can reduce our proof to the knowledge soundness proof of Groth16 [72], since $[A]_1, [B]_2, [C\zeta]_1$ is a valid proof of it.

To prove that A, B, C are not constructed from the elements $[A_j]_1, [B_j]_2, [C_j]_1$, $[\delta_j]_2$, we follow the exact same reasoning as Bowe and Gabizon [26] in the asymmetric generic group model. First of all, we argue that the term $\alpha\beta$ is in the expression of AB, which means the coefficient $k_{AB,\alpha\beta} \neq 0$. Since $AB = C\delta' + \mathsf{ic} + \alpha\beta$ from the verification equation, the term $\alpha\beta$ is not in ic by definition; if it was in $C\delta'$, then the term $\frac{\alpha\beta}{\delta}$ would appear in the expression of C but this cannot be possible because it is not in Q'_1; then $\alpha\beta$ is in the expression of AB, i.e. $k_{AB,\alpha\beta} \neq 0$.

In the following, we show that α is in A and β is in B by ruling out all other possibilities.

Looking at Q' we have that $k_{AB,\alpha\beta} = k_{A,\alpha}k_{B,\beta} + k_{A,C_j}k_{B,\delta_j} \neq 0$ because the elements in Q'_1 capable to generate A and produce $\alpha\beta$ when the product AB is computed are $[\alpha]_1[\beta]_2$ and $\left[\frac{\alpha\beta}{\delta_j}\right]_1[\delta_j]_2$, all the other combinations contain the variable x. Now, we show that $k_{A,\alpha}k_{B,\beta} \neq 0$.

We assume $k_{A,\alpha}k_{B,\beta} = 0$. Then, $k_{A,C_j}k_{B,\delta_j} \neq 0$. We distinguish two cases, and show that no-one is possible:

1. If $k_{A,A_j}k_{B,B_j} = 0$, i.e. A_j does not appear in A and B_j does not appear in B, but C_j appears in A and δ_j in B. We have $k_{C\delta',A_jB_j} = k_{A,C_j}k_{B,\delta_j} = k_{AB,\alpha\beta} \neq 0$, then either the term A_jB_j appears in $\mathsf{ic} + \alpha\beta$, which cannot be possible by definition, or $\frac{A_jB_j}{\delta}$ appears in C, but this term cannot be computed from therms in Q'_1.
2. Otherwise, $k_{A,A_j}k_{B,B_j} \neq 0$, i.e. A_j and B_j appear in A and B, respectively.
   - if C_j appears in A, then $k_{AB,C_jB_j} = k_{C\delta',C_jB_j} = k_{A,C_j}k_{B,B_j} \neq 0$, and $\frac{C_j}{\delta}$ appears in C, but it cannot be produced from elements in Q'_1.
   - if δ_j appears in B, then $k_{AB,A_j\delta_j} = k_{C\delta',A_j\delta_j} = k_{A,A_j}k_{B,\delta_j} \neq 0$, and $\frac{A_j\delta_j}{\delta}$ appears in C, but it cannot be produced from elements in Q'_1.

Now, that we have α, β appear in the expressions of A, B, respectively, we use it to show that A, B, C are not produced by $A_j, B_j, C_j, D_j, \delta_j$.

1. A_j, C_j cannot appear in A, and B_j, δ_j cannot appear in B.
   - If A_j appears in A, then $k_{AB,A_j\beta} = k_{A,A_j}k_{B,\beta} \neq 0$, and $\frac{A_j\beta}{\delta}$ appears in C.
   - If C_j appears in A, then $k_{AB,A_jB_j\frac{\beta}{\delta_j}} = k_{A,C_j}k_{B,\beta} \neq 0$, and $\frac{A_jB_j\beta}{\delta_j\delta}$ appears in C.
   - If B_j appears in B, then $k_{AB,\alpha B_j} = k_{A,\alpha}k_{B,B_j} \neq 0$, and $\frac{\alpha B_j}{\delta}$ appears in C.
   - If δ_j appears in B, then $k_{AB,\alpha\delta_j} = k_{A,\alpha}k_{B,\delta_j} \neq 0$, and $\frac{\alpha\delta_j}{\delta}$ appears in C.

   Any of previous cases cannot occur, because the elements $\frac{A_j\beta}{\delta}$, $\frac{A_jB_j\beta}{\delta_j\delta}$, $\frac{\alpha B_j}{\delta}$, $\frac{\alpha\delta_j}{\delta}$ cannot be produced from the elements in Q'_1.
2. A_j, C_j cannot appear in C. If A_j, C_j appears in C, then $A_j\delta, C_j\delta$ appear in AB, respectively, which implies A_j, C_j appear in A, that is already ruled out.
3. D_j cannot appear in A. Assume A has been generated from some $D_j = \frac{t(x)}{\delta_j + m_j\delta}$, so $k_{A,D_j} \neq 0$. Observe that from the verification equation and the fact that $k_{B,\beta} \neq 0$, implies $k_{AB,D_j\beta} \neq 0$. But this cannot be cancelled out by any of the other terms in the equation, so $k_{C\delta',D_j\beta} \neq 0$. Since β is independent of δ',

$k_{C,D_j} \neq 0$, but $D_j$cannot be computed from elements in Q'_1.

4. D_j cannot appear in C. Otherwise, $k_{C,D_j} \neq 0$, so $k_{C\delta,D_j\delta'} \neq 0$. However, this would imply that $k_{AB,D_j\delta'} \neq 0$ and since $\delta' \in Q'_2$, $k_{A,D_j} \neq 0$, which we ruled out previously.

□

Note that we make the proof in the asymmetric GGM for simplicity, but the analogous proof in the symmetric model gives very similar impossible terms. The difference is that we have to consider the whole Q' in the argumentations and in the second part of the proof we have to analyse more possible cases (considering α in any of the groups).

5.4 A Simulation Extractable zk-SNARK without RO

In this section, we present another variation of the Groth16 which is very similar to the construction in Section 5.3. It also offers simulation extractability in the Generic Group Model, but without involving the Random Oracle. This is done in exchange for adding one element in the crs.

5.4.1 Scheme definition

In Fig. 5.2, we propose our second variation of Groth16 for QAP. It is inspired by the simulation extractable version of Bowe and Gabizon [26] and is very similar to the first construction of this work (see Section 5.3 for intuition).

In this approach, we change the Proof of Knowledge (PoK) of the DLOG of $[\delta']_2$ w.r.t. $[\delta]_2$ to another PoK in the GGM without using random oracles with a variation of Boneh-Boyen signatures, where we just use the collision resistance property of the hash function. We briefly give an intuition in the following.

Avoiding Random Oracle

Our proof uses the collision resistance property of the hash function and the generic group model. Very roughly, the new variable γ gives some additional guarantees because to compute $t(x)\frac{(\gamma+m)}{(\delta'+\delta m)}$ from D_j such that $m_j \neq m$, it is necessary to know both $\frac{1}{(\delta'+\delta m)}$ and $\frac{\gamma}{(\delta'+\delta m)}$, but this is only possible when $\delta' + \delta m = k\delta$. Then, either we have the knowledge of the DLOG of δ' respect to δ $(k - m)$, which is straightforward, or either we have re-used δ'_j and m_j from some jth query. The last case is discarded when we reach that same message had to be re-used, $m = m_j$, which breaks collision resistance of the hash.

5.4.2 Security

We prove security of our construction (in Fig. 5.2) in the following theorem.

Theorem 29 (Completeness, ZK, SE)**.** *The variation of* Groth16 *described in Fig. 5.2, guarantees perfect completeness, perfect zero-knowledge and simulation-extractability in the asymmetric Generic Group Model.*

Proof. Perfect completeness and perfect zero-knowledge are obvious and the proof is omitted. Knowledge extractability is proven in the same way as the proof of Section 5.3 by reduction (in the GGM) to the knowledge soundness of Groth16, the reduction works in these two steps:

Step 1 Extraction of the DLOG of δ'.

Step 2 Reduction to the Knowledge Soundnesss of Groth16.

Proof of Step 1) Suppose $\mathcal{A}$ has made a sequence of queries $x_1, \cdots, x_v$ to $\mathsf{S}(\mathsf{ts}, \cdot)$, and received answers $\{\pi_j = ([A_j]_1, [B_j]_2, [C_j]_1, [D_j]_1, [\delta_j]_2)\}_{j=1}^{v}$. Let Q' be the union of elements in the crs together with those from the replies of $\mathsf{S}(\mathsf{ts}, \cdot)$; namely,

$$Q' := \begin{pmatrix} [\alpha, \beta, \delta, \{x^i\}_{i=0}^{n-1}, \gamma t(x)/\delta \\ \{u_j(x)\beta + v_j(x)\alpha + w_j(x)\}_{j=0}^{l}, \\ \{\frac{u_j(x)\beta + v_j(x)\alpha + w_j(x)}{\delta}\}_{j=l+1}^{m}, \\ \{x^i t(x)/\delta\}_{i=0}^{n-2}]_1, [\beta, \delta, \{x^i\}_{i=0}^{n-1}]_2 \end{pmatrix} \cup \begin{pmatrix} \Big\{\Big[A_j, C_j := \frac{A_j B_j - \mathsf{ic}_j - \alpha\beta}{\delta_j}, \\ D_j := \dfrac{t(x)(\gamma + m_j)}{\delta_j + m_j\delta}\Big]_1 \\ [B_j, \delta_j]_2, m_j\Big\}_{j=1}^{v} \end{pmatrix}$$

where $\mathsf{ic}_j = \sum_{i=0}^{l} a_i^j(u_i(x)\beta + v_i(x)\alpha + w_i(x))$, $\boldsymbol{x}_j = (a_1^j, \ldots, a_l^j)$, and $m_j \in \mathbb{Z}_p$ the message that simulator receives from the hash function for each A_j, B_j, C_j, δ_j.

We assume the adversary $\mathcal{A}$ has produced elements (A, B, C, D, δ') such that

$$A \cdot B \equiv C \cdot \delta' + \Big(\textstyle\sum_{j=0}^{l} a_j(u_j(x)\beta + v_j(x)\alpha + w_j(x))\Big) + \alpha\beta$$

Setup, $\mathsf{crs} \leftarrow \mathsf{K}(\mathbf{R}, \mathsf{z}_{\mathbf{R}})$**:** Similar to the original scheme pick $x, \alpha, \beta, \delta \leftarrow \mathbb{Z}_p^*$, $H \leftarrow \mathcal{H}$, and returns crs defined as the following,

$$(\mathsf{crs}_{\mathsf{P}}, \mathsf{crs}_{\mathsf{V}}) := \mathsf{crs} \leftarrow \begin{pmatrix} [\alpha, \beta, \delta, \{x^i\}_{i=0}^{n-1}, \{u_j(x)\beta + v_j(x)\alpha + w_j(x)\}_{j=0}^{l}, \frac{\gamma t(x)}{\delta}] \\ \left\{ \frac{u_j(x)\beta + v_j(x)\alpha + w_j(x)}{\delta} \right\}_{j=l+1}^{m}, \left\{ \frac{x^i t(x)}{\delta} \right\}_{i=0}^{n-2}]_1, \\ [\beta, \delta, \{x^i\}_{i=0}^{n-1}]_2, \ [\alpha\beta, t(x), \gamma t(x)]_T, H \end{pmatrix}.$$

Prover, $\pi \leftarrow \mathsf{P}(\mathbf{R}, \mathsf{z}_{\mathbf{R}}, \mathsf{crs}_{\mathsf{P}}, \boldsymbol{x} = (a_1, \ldots, a_l), \boldsymbol{w} = (a_{l+1}, \ldots, a_m))$**:** assuming $a_0 = 1$, it acts as follows,

1. Selects a random element $\zeta \leftarrow \mathbb{Z}_p^*$, and sets $[\delta']_2 := \zeta[\delta]_2$
2. Let $A^\dagger(X) \leftarrow \sum_{j=0}^{m} a_j u_j(X)$, $B^\dagger(X) \leftarrow \sum_{j=0}^{m} a_j v_j(X)$,
3. Let $C^\dagger(X) \leftarrow \sum_{j=0}^{m} a_j w_j(X)$,
4. Set $h(X) = \sum_{i=0}^{n-2} h_i X^i \leftarrow (A^\dagger(X)B^\dagger(X) - C^\dagger(X))/t(X)$,
5. Set $[h(x)t(x)/\delta']_1 \leftarrow (1/\zeta)(\sum_{i=0}^{n-2} h_i [x^i t(x)/\delta]_1)$,
6. Set $r_a \leftarrow \mathbb{Z}_p$; Set $[A]_1 \leftarrow \sum_{j=0}^{m} a_j [u_j(x)]_1 + [\alpha]_1 + r_a [\delta']_1$,
7. Set $r_b \leftarrow \mathbb{Z}_p$; Set $[B]_2 \leftarrow \sum_{j=0}^{m} a_j [v_j(x)]_2 + [\beta]_2 + r_b [\delta']_2$,
8. Set $[C]_1 \leftarrow r_b[A]_1 + r_a \left(\sum_{j=0}^{m} a_j [v_j(x)]_1 + [\beta]_1 \right) + (1/\zeta) \sum_{j=l+1}^{m} a_j ([(u_j(x)\beta + v_j(x)\alpha + w_j(x))/\delta]_1) + [h(x)t(x)/\delta']_1$,
9. Set $m = H([A]_1 \parallel [B]_2 \parallel [C]_1 \parallel [\delta']_2)$, where $H : \{0,1\}^* \rightarrow \mathbb{Z}_p^*$ is a secure hash function,
10. Compute $[D]_1 = \frac{m}{\zeta + m}[\frac{t(x)}{\delta}]_1 + \frac{1}{\zeta + m}[\frac{\gamma t(x)}{\delta}]_1 = [\frac{(m+\gamma)t(x)}{\delta' + m\delta}]_1$
11. Return $\pi := ([A, C, D]_1, [B, \delta']_2)$.

Verifier, $\{1, 0\} \leftarrow \mathsf{V}(\mathbf{R}, \mathsf{z}_{\mathbf{R}}, \mathsf{crs}_{\mathsf{V}}, \boldsymbol{x} = (a_1, \ldots, a_l), \pi = ([A, C, D]_1, [B, \delta']_2))$**:** assuming $a_0 = 1$, and setting $m = H([A]_1 \parallel [B]_2 \parallel [C]_1 \parallel [\delta']_2)$ checks if

1. $[A]_1[B]_2 = [C]_1[\delta']_2 + \left(\sum_{j=0}^{l} a_j [u_j(x)\beta + v_j(x)\alpha + w_j(x)]_1 \right) [1]_2 + [\alpha\beta]_T$
2. $[D]_1[\delta' + \delta m]_2 = m[t(x)]_T + [\gamma t(x)]_T$

and returns 1 if both checks pass, otherwise return 0.

Simulator, $\pi \leftarrow \mathsf{S}(\mathbf{R}, \mathsf{z}_{\mathbf{R}}, \mathsf{crs}_{\mathsf{V}}, \boldsymbol{x} = (a_1, \ldots, a_l), \mathsf{ts})$**:** Given the simulation trapdoors $\mathsf{ts} := (\beta, \delta)$ acts as follows,

1. Choose random $\zeta \leftarrow \mathbb{Z}_p^*$ and set $\delta' := \zeta\delta$
2. Choose $A, B \leftarrow \mathbb{Z}_p$
3. Let $[C]_1 = [(A \cdot B - \sum_{j=0}^{l} a_j (u_j(x)\beta + v_j(x)\alpha + w_j(x)) - \alpha\beta)/\delta']_1$
4. Let $m = H([A]_1 \parallel [B]_2 \parallel [C]_1 \parallel [\delta']_2)$
5. $[D]_1 = \frac{m}{\zeta + m}[\frac{t(x)}{\delta}]_1 + \frac{1}{\zeta + m}[\frac{\gamma t(x)}{\delta}]_1 = [\frac{(m+\gamma)t(x)}{\delta' + m\delta}]_1$
6. Return $\pi := ([A, C, D]_1, [B, \delta']_2)$.

Figure 5.2: The proposed simulation-extractable variation of Groth16 for $\mathbf{R}$ along with a modification of the Boneh Boyen signature. $\mathcal{H}$ is a family of collision resistant hash functions that maps to $\mathbb{Z}_p^*$. The elements $[\alpha\beta, t(x), \gamma t(x)]_T$ are redundant and can in fact be computed from the rest of the elements in the crs. Alternatively, one can describe Groth16 as corresponding to $\zeta = 1, \gamma = 0$ and where the proof consists only of $[A, C]_1, [B]_2$. Differences with Groth16 are highlighted.

and, for $m := H([A]_1 \parallel [B]_2 \parallel [C]_1 \parallel [\delta']_2)$, $D(\delta' + \delta m) = t(x)(m + \gamma)$. Let Q'_1 the elements in Q' in $\mathbb{G}_1$ and Q'_2 the elements in $\mathbb{G}_2$. Since the adversary is generic it has constructed these elements as a linear combination of the elements in Q' which are in the relevant group (i.e. element of Q'_1 in $\mathbb{G}_1$ for A, C, D and of Q'_2 for B, δ') and we can extract the coefficients of this linear combination.

First, we prove that the adversary has knowledge of the discrete logarithm of δ' w.r.t. δ. From the second verification equation we know that $D = t(x)\frac{\gamma+m}{\delta'+m\delta}$. On the other hand, from adversary $\mathcal{A}$ we can recover a vector $\boldsymbol{k}_D$ with the coefficients that it has used to construct D, that is, $D = \sum_{q \in Q'_1} k_{D,q}\, q$. Equating these two expressions,

$$t(x)(m + \gamma) = (\textstyle\sum_{q \in Q'_1} k_{D,q}\, q)(\delta' + m\delta), \tag{5.4}$$

where $\delta' = \sum_{q \in Q'_2} k_{\delta',q}\, q$ for another vector of coefficients $\boldsymbol{k}_{\delta'}$. The terms which include γ in both sides of the equation must be the same.

On the other hand, by assumption, in the asymmetric GGM, the term δ' is constructed as a linear combination of elements in Q'_2 and therefore $\delta' + \delta m$ is independent of γ. Then, keeping only the terms with γ in equation (5.4), we obtain the following relation:

$$t(x)\gamma = k_{D,0}\frac{\gamma t(x)}{\delta}(\delta' + m\delta) + \textstyle\sum_{j=1}^{v} k_{D,j}\frac{\gamma t(x)}{\delta_j + m_j\delta}(\delta' + m\delta), \tag{5.5}$$

where we have set $k_{D,0} = k_{D,\frac{\gamma t(x)}{\delta}}$ and $k_{D,j} = k_{D,\frac{\gamma t(x)}{\delta_j + m_j\delta}}$ to simplify the notation.

Dividing both sides of the equation by $t(x)\gamma$ and defining $\delta_0 = \delta'$, $m_0 = 0$, we obtain the following equivalent equation:

$$1 = \left(\textstyle\sum_{j=0}^{v} k_{D,j}\frac{1}{\delta_j + m_j\delta}\right)(\delta' + m\delta) = \textstyle\sum_{j=0}^{v} k_{D,j}\frac{\prod_{i=0, i\neq j}^{v}(\delta_i + m_i\delta)}{\prod_{i=0}^{v}(\delta_i + m_i\delta)}(\delta' + m\delta)$$

$$\Leftrightarrow \textstyle\prod_{i=0}^{v}(\delta_i + m_i\delta) = (\delta' + m\delta)\left(\sum_{j=0}^{v} k_{D,j}\prod_{i=0, i\neq j}(\delta_i + m_i\delta)\right). \tag{5.6}$$

From the last equation it follows that the term $\delta' + m\delta$ must divide the left side of the equation (5.6). Therefore, there exists some index j^* and $k \in \mathbb{Z}_p$ such that $\delta' + m\delta = k(\delta_{j^*} + m_{j^*}\delta)$. Now, dividing Eq. (5.6) by $(\delta_{j^*} + m_{j^*}\delta)$, we come to the following expression

$$0 = (1 - k \cdot k_{D,j^*})\textstyle\prod_{i=0, i\neq j^*}^{v}(\delta_i + m_i\delta) - \sum_{j=0, j\neq j^*}^{v} k_{D,j}\prod_{i=0, i\neq j}^{v}(\delta_i + m_i\delta).$$

Since all summands are linearly independent polynomials, $k = k_{D,j^*}^{-1}$, and $k_{D,j} = 0$ if $j \neq j^*$. We distinguish two cases: (1) $\delta' + \delta m = k\delta$ ($j^* = 0$) or (2) $\delta' + \delta m = k(\delta_{j^*} + m_{j^*}\delta)$ ($j^* \neq 0$).

In case (1), we are done, as we can extract the DLOG of δ' as $k - m$.

In case (2), from equation (5.4) and putting everything together, we have that:

$$t(x)(m+\gamma) = k_{D,j^*}\frac{(\gamma + m_{j^*})}{(\delta_{j^*} + m_{j^*}\delta)}(\delta'+m\delta) = k_{D,j^*}k^{-1}(\gamma+m_{j^*})t(x) = (\gamma+m_{j^*})t(x).$$

This implies that $m_{j^*} = m$ is a collision of H.

Proof of Step 2) We show that the elements A, B, C do not use the elements of the simulated proofs, say $V := \{[A_j]_1, [B_j]_2, [C_j]_1, [D_j]_1, [\delta_j]_2\}_{j=1}^{v}$, and then, with the knowledge of ζ such that $\delta' = \zeta\delta$, we can reduce our proof to the knowledge soundness proof of Groth16 [72], since $[A]_1, [B]_2, [C\zeta]_1$ is a valid proof of Groth16.

For this, we need to argue that A, B, C cannot have been constructed from any of the elements of the queries. To prove that A, B, C are not constructed from the elements $[A_j]_1, [B_j]_2, [C_j]_1, [\delta_j]_2$, we follow the exact same reasoning as Bowe and Gabizon [26] in the GGM and we omit the details. Next, we prove that to construct A, C the prover cannot have used any of the D_j terms, which are the new elements in our proof.

Analogously to proof in Section 5.3, assume A has been generated from some $D_j = \frac{t(x)(m_j+\gamma)}{\delta_j+m_j\delta}$. Observe that the verification equation contains the term $\alpha\beta$ which is produced by AB by a similar argument to Section 5.3.Thus, if $k_{A,D_j} \neq 0$, then $k_{AB,D_j\beta} \neq 0$. However, this term in AB cannot be cancelled out by any of the other terms in the equation. Because if $k_{C\delta',D_j\beta} \neq 0$, then $k_{C,D_j} \neq 0$ because δ' is independent of β, but D_j cannot be computed from elements in Q_1'.

Now, assume D_j appears in C, then $k_{C\delta',D_j\beta} \neq 0$. However, neither the term $\alpha\beta$ nor the sum of public values can include it, so the only possibility is that it appears in AB. Since $\delta' \in Q_2'$, then A would contain D_j, which we ruled out previously.

□

Chapter 6

Somewhere Statistically Binding Commitments

This chaper is based on the result *Somewhere Statistically Binding Commitments* accepted to be published in Financial Cryptography 2021.

6.1 Introduction

As we already mentioned in Chapter 1, commitment schemes are one of the most useful primitives in cryptography. In essence, a commitment to a value binds this value to the commitment but hides it from other parties. Commitment schemes are naturally used in zero-knowledge proofs, where one often proves statements about a committed value while keeping it hidden. For instance, to complete a digital transaction, a party may need to prove he has available funds in his account without actually revealing his exact balance. Such proofs on committed values are very efficient due to works like Bulletproofs [28], and are used in many privacy-preserving cryptocurrency designs such as Mimblewimble [111, 58] and Quisquis [54].

Dual-mode commitment schemes [44, 34, 43] are an interesting variant where the commitment key can be set up in one of two modes: binding or hiding. In the binding mode, the commitment can only be opened to one valid value. Meanwhile, in the hiding mode, a commitment hides the committed value even to unbounded adversaries. For this definition to make sense, one should not be able to guess which mode is being used based on the commitment key, i.e., the commitment key hides the mode. Dual-mode

commitments are an essential tool in Groth-Sahai proofs [78] which is a framework for constructing non-interactive zero-knowledge (NIZK) proofs for algebraic relations.

In the case of committing to a vector, the two modes of a dual-mode commitment can be seen to be two extremes: the commitment is either binding in all positions in the vector or none of them. A natural way to generalize the notion would be to have multiple modes of commitment, specifying that the commitment is binding in some positions in the vector of values. A similar generalization for hash functions is known as somewhere statistically binding hash [83, 107], in which one can compute a hash of a vector v such that the computed hash is statistically binding in one coordinate of v.

A generalization of dual-mode commitments would lead to interesting applications in NIZK arguments. In a typical zero-knowledge succinct argument of knowledge (zk-SNARK) for Circuit-SAT [71, 100, 61, 45], the prover commits to the witness (i.e., all the inputs to a circuit), and the proof of (knowledge) soundness involves using a non-falsifiable assumption to extract the whole committed vector, which is then used to check each gate to establish where exactly the prover cheated; based on the knowledge of the witness one then breaks a computational assumption. One can get a more efficient extraction under falsifiable assumptions if the commitment was binding only on the values corresponding to the inputs and outputs of a specific gate: one then only needs to check the extracted values against a randomly chosen gate. As a caveat, the technique will lead to a security loss linear in the number of gates.

In fact, the above extraction technique has been done before in Chapter 3 and [69] using a generalization of the Pedersen commitment scheme called *Extended Multi-Pedersen* [67, 68] and resulting in efficient NIZK arguments under falsifiable assumptions. However, the above results are not zk-SNARKs: they are *Quasi-Adaptive* NIZK (QA-NIZK) arguments which means the crs may depend on the relation, and while the argument is succinct, the commitment is not.[1] Moreover, Chapter 3 neither [69] did not formalize which properties of a commitment scheme would be required to enable efficient NIZK arguments.

In the above construction, we need a succinct *somewhere statistically binding* property that guarantees that the chosen coordinate is statistically binding while the remaining coordinates can be computationally binding. On the other hand, to get zero-knowledge, the commitment needs to be *almost-everywhere statistically hiding*, that is, computationally hiding at the chosen coordinate, and statistically hiding at any other coordinates. We also need *index-set hiding*, which means an adversary that is given the commitment key does not know which particular coordinate is statistically binding.

[1] One cannot construct NIZK arguments in a black-box way from falsifiable assumptions [64], hence any black-box construction from falsifiable assumptions will not be fully succinct.

Our Contributions

Formalizing the properties of the *Extended Multi-Pedersen* (EMP) commitment scheme [67, 68], we define a *somewhere statistically binding (SSB) commitment scheme* to n-dimensional vectors. In the commitment key generation phase of an SSB commitment scheme, one chooses an index-set $\mathcal{S} \subseteq [1..n]$ of size at most $q \leq n$ and defines a commitment key $\mathtt{ck}$ that depends on n, q and $\mathcal{S}$. A commitment to an n-dimensional vector $\boldsymbol{x}$ will be statistically binding and extractable at coordinates indexed by $\mathcal{S}$ and perfectly hiding at all other coordinates. Moreover, commitment keys corresponding to any two index-sets $\mathcal{S}_1$ and $\mathcal{S}_2$ of size at most q must be computationally indistinguishable. Thus, an *SSB commitment scheme* is required to be SSB, *somewhere statistically extractable* (SSE), *almost everywhere statistically hiding* (AESH), and *index-set hiding* (ISH). An SSB commitment scheme generalizes dual-mode commitment schemes (where $n = 1$ and $q \in \{0, 1\}$ determines the mode) and the EMP commitment scheme (where $q = 1$ and n is arbitrary).

In Section 6.4, we define algebraic commitment schemes (ACS), where the commitments keys are matrices of general matrix distributions. We prove that the distribution of key matrices defines which properties of SSB commitments hold in each coordinate and show that these commitments are suitable for working with QA-NIZK arguments. This is because they behave like linear maps and the properties of SSB commitments can be expressed in terms of membership to linear subspaces. Next, we generalize the EMP commitment scheme to work with arbitrary values of q. Importantly, a single EMP commitment consists of $q+1$ group elements and it is thus succinct given small q. We prove that EMP satisfies the mentioned security requirements under a standard Matrix DDH assumption [52].

In Section 6.5, we define *functional SSB* commitments, which are statistically binding on some components that are outputs of some functions $\mathcal{S} = \{f_i\}_i$ where $|\mathcal{S}| \leq q$. It is a generalization of SSB commitments, where the extracted values are the result of some linear functions of the committed values, instead of the values themselves. We show that results which hold for SSB commitments, also naturally hold for functional SSB commitments. The notion of functional SSB commitments for families of linear functions was already used indirectly in our prior work in Chapter 3; however, they were not formally defined and their security properties were not analyzed. We also see that a minor modification of EMP works as a functional SSB commitment if we consider only linear functions.

We provide some applications of functional SSB commitments. In Section 6.6.1 we propose a novel (but natural) application that we call oblivious database queries (ODQ). In an ODQ protocol, a sender has a private database $\boldsymbol{x}$ and a receiver wants to query the database to learn $f_1(\boldsymbol{x}), \ldots, f_q(\boldsymbol{x})$ without revealing the functions f_i.

This can be directly realized with linear EMP if we restrict f_i to be linear functions. The receiver sends a commitment key (which encodes $\mathcal{S} = \{f_i\}_i$) to the sender who responds with a commitment to the database $\boldsymbol{x}$. The receiver can then extract the query results with an extraction key (SSE property). Unfortunately, linear EMP only has F-extractability [17] (more precisely, one can only extract the message as a vector of group elements, not a vector of integers), and thus we are only able to extract $\{g^{f_i(\boldsymbol{x})}\}_i$ where g is a generator of some cyclic group. The protocol is secure in the semi-honest model[2]. In particular, the receiver's privacy follows from the function-set hiding property (analogue to ISH in functional SSB commitments), which holds under the DDH assumption. Sender's privacy holds information-theoretically since using AESH property, we can perfectly simulate the commitment. We also achieve near-optimal download rate (the ratio between output size and sender's message size) which is $q/(q+1) \approx 1$ but sub-optimal total rate (ratio between output size and total transcript size) of approximately $1/(n+q)$.

A similar approach also gives us oblivious linear function evaluation (OLE) [49, 65, 48] where the sender has a private linear function f and the receiver wants to learn $f(\boldsymbol{x})$ of his private input $\boldsymbol{x}$. However, in this case, both download rate and total rate are sub-optimal.

Recently, Döttling et al. [47] proposed an oblivious matrix-vector product protocol in the semi-honest model using trapdoor hash functions. In their case, the receiver has $\boldsymbol{x}$, the sender has a matrix $\mathbf{M}$, and the receiver wants to learn $\mathbf{M}\boldsymbol{x}$. If we interpret linear functions $\{f_i\}_i$ as a matrix $\mathbf{M}$, then our ODQ can be seen as an OMV protocol where the roles of sender and receiver are switched. They gave a construction under the Learning with Errors (LWE) and the Quadratic Residuosity (QR) problems, which work over fields with small characteristic or rings modulo a smooth integer. Interestingly, they also achieve a download rate of 1 but sub-optimal total rate. Thus our work can be viewed as complementary to their result.

In Section 6.6.2 we present a QA-NIZK for Square Arithmetic Programs (SAP, [74]) that follows a similar strategy to our main construction in Chapter 3, but can be used for arithmetic circuit satisfiability instead of Boolean circuit satisfiability. This QA-NIZK has comparable efficiency to the boolean one in Chapter 3 and it is also proven under falsifiable assumptions.

Application: Shorter QA-NIZK for arithmetic circuits

In Chapter 3, we constructed an efficient commit-and-prove QA-NIZK argument for Square Span Programs (SSP, [45]) under falsifiable assumptions, which can be used

[2]A semi-honest party follows the protocol, but tries to learn extra information from their view in the protocol [47].

to prove boolean circuit satisfiability. We present a QA-NIZK for Square Arithmetic Programs (SAP, [74]) in Section 6.6.2 that follows a similar strategy but can be used for arithmetic circuit satisfiability with comparable efficiency and also proven under falsifiable assumptions. Both constructions use a linear-length perfectly binding commitment of the witness, but they are otherwise succinct arguments; the arguments also contain perfectly hiding commitments that come from zk-SNARK techniques for proving satisfiability of quadratic equations and a functional SSB commitment to extract certain linear functions of the witness in the security reduction.

We note that the construction in Chapter 3 uses linear EMP commitment schemes indirectly. We formalize and generalize them in our framework as functional SSB commitments and then use them as a black box in our QA-NIZK application. This significantly simplifies the understanding of the scheme in two ways. Firstly, the techniques used in the security proof are natural functionalities of *algebraic commitment schemes* that we present in this work, e.g., using a commitment key consisting of two orthogonal matrices to enable extraction. Secondly, the notation of our commitments is more compact, which helps to see that soundness is guaranteed by the SSB, $[\cdot]$-SSE, and FSH properties of functional SSB and zero-knowledge is guaranteed by AESH.

We give an intuition of the proof and soundness strategy in the following. The proof consists of two subarguments: one based on zk-SNARK techniques where many quadratic equations are proved to be satisfied using a single polynomial divisibility relation with polynomials evaluated at a secret point s, and a proof of membership in subspace showing that all the commitments in the argument open to the same witness. We have one linear perfectly binding commitment $[\boldsymbol{c}]_1$, which is an ElGamal encryption of the witness in the source group $\mathbb{G}_1$. Similarly to zk-SNARKs, the witness is extracted in the security proof and used to detect which quadratic equation of the language does not hold. However, our commitment is only F-extractable, which is not enough to break the underlying falsifiable assumption. Note that zk-SNARKs typically use a non-falsifiable assumption at this point to avoid this issue. We instead use a linear EMP commitment $[\boldsymbol{d}]_2$ in the source group $\mathbb{G}_2$ that perfectly hides the witness in the honest proof (setting $\mathcal{S} = \emptyset$).

In the security proof, we change to an indistinguishable game (by the FSH property) where the commitment key now encodes some linear functions that depend on the secret point s. This will allow us to F-extract linear combinations of the form $\sum_i w_i \alpha_i(s)$ where $\{w_i\}_i$ is the witness and $\alpha_i(s)$ are coefficients of the function we choose. Essentially it allows us to trick the prover into computing some secret linear function of the witness. We see that the extra knowledge from the commitment $[\boldsymbol{d}]_2$ allows us to break a variant of the q-target strong Diffie-Hellman (TSDH) assumption [21]. We also prove that the new assumption is falsifiable and equivalent to the q-TSDH assumption under a knowledge assumption in Section 6.6.2.

Relation to other primitives

The SSB requirement makes the EMP commitment scheme look similar to SSB hash functions [83, 107], but there are obvious differences. SSB hash has the local opening property, where the committer can efficiently open just one coordinate of the committed vector, but SSB commitments do not[3]. Meanwhile, we need hiding while SSB hash does not. This is, intuitively, a natural distinction and corresponds to the difference between collision-resistant hash families and statistically hiding commitment schemes. Also, we allow `ck` to be long, but require commitments to be succinct.

SSB commitments are directly related to two-message oblivious transfer (OT) protocols as defined in [8]. Essentially, SSB commitments are non-interactive analogs of such protocols: the commitment key corresponds to the first OT message ot_1 and the commitment corresponds to the second OT message ot_2. Importantly, while in OT, the ot_1 generator is always untrusted, in our applications, it is sufficient to consider a trusted `ck` generator. This allows for more efficient constructions.

We discuss the relation to existing primitives in more detail in Section 6.7.

6.2 Preliminaries

For a set S, let $\mathbb{P}(S)$ denote the power set (i.e., the set of subsets) of S, and let $\mathbb{P}(S, q)$ denote the set of q-size subsets of S. For an n-dimensional vector $\boldsymbol{\alpha}$ and $i \in [1..n]$, let α_i be its ith coefficient. For a tuple $\mathcal{S} = (\sigma_1, \ldots, \sigma_q)$ with $\sigma_i < \sigma_{i+1}$, let $\boldsymbol{\alpha}_{\mathcal{S}} = (\alpha_{\sigma_1}, \ldots, \alpha_{\sigma_q})$. Let $\boldsymbol{\alpha}_{\emptyset}$ be the empty string.

6.3 SSB Commitment Schemes

In an SSB commitment scheme, the commitment key depends on n, q, and an index-set $\mathcal{S} \subseteq [1..n]$ of cardinality $\leq q$ (in the case of Groth-Sahai commitments [78], $n = q = 1$ while in the current result $n = poly(\lambda)$ and $q \geq 1$ is a small constant). At coordinates described by $\mathcal{S}$, an SSB commitment scheme must be *statistically binding* and F-*extractable* [17] for a well-chosen function F, while at all other coordinates it must be *statistically hiding* and *trapdoor*. Moreover, it must be index-set hiding, i.e., commitment keys corresponding to any two index-sets $\mathcal{S}_1$ and $\mathcal{S}_2$ of size $\leq q$ must be computationally indistinguishable.

[3]The properties of SSB and local opening are orthogonal: it is possible to construct efficient SSB hashes without local opening [107] and efficient vector commitments [98, 33] (which have a local opening) without the SSB property

The Groth-Sahai commitments correspond to a *bimodal* setting where either all coefficients are statistically hiding or statistically binding, and these two extremes are indistinguishable. SSB commitments correspond to a more fine-grained *multimodal* setting where some $\leq q$ coefficients are statistically binding and other coefficients are statistically hiding, and all possible selections of statistically binding coefficients are mutually indistinguishable. Our terminology is inspired by [83, 107] who defined SSB hashing; however, the consideration of the hiding property makes the case of SSB commitments sufficiently different.

6.3.1 Formalization and Definitions

Definition 33. *An F-extractable SSB commitment scheme $\mathsf{COM} = (\mathcal{G}, \mathsf{KC}, \mathsf{com}, \mathsf{tdOpen}, \mathsf{Ext}_F)$ consists of the following polynomial-time algorithms:*

Parameter generation: $\mathcal{G}(1^\lambda)$ returns parameters $\mathtt{pp}$ (e.g., description of a bilinear group).

Commitment key generation: for parameters $\mathtt{pp}$, $n = poly(\lambda)$, $q \in [1..n]$, and a tuple $\mathcal{S} \subseteq [1..n]$ with $|\mathcal{S}| \leq q$, $\mathsf{KC}(\mathtt{pp}, n, q, \mathcal{S})$ outputs a commitment key $\mathtt{ck}$ and a trapdoor $\mathtt{td} = (\mathtt{ek}, \mathtt{tk})$ consisting of an *extraction key* $\mathtt{ek}$, and a *trapdoor key* $\mathtt{tk}$. Also, $\mathtt{ck}$ implicitly specifies $\mathtt{pp}$, n, q, the message space $\mathtt{MSP}$, the randomizer space $\mathtt{RSP}$, the extraction space $\mathtt{ESP}$, and the commitment space $\mathtt{CSP}$, such that $F(\mathtt{MSP}) \subseteq \mathtt{ESP}$. For invalid input, KC outputs $(\mathtt{ck}, \mathtt{td}) = (\bot, \bot)$.

Commitment: for $\mathtt{pp} \leftarrow \mathcal{G}(1^\lambda)$, $\mathtt{ck} \neq \bot$, a message $\boldsymbol{x} \in \mathtt{MSP}^n$, and a randomizer $r \in \mathtt{RSP}$, $\mathsf{com}(\mathtt{ck}; \boldsymbol{x}; r)$ outputs a commitment $\boldsymbol{c} \in \mathtt{CSP}$.

Trapdoor opening: for $\mathtt{pp} \leftarrow \mathcal{G}(1^\lambda)$, $\mathcal{S} \subseteq [1..n]$ with $|\mathcal{S}| \leq q$, $(\mathtt{ck}, (\mathtt{ek}, \mathtt{tk})) \leftarrow \mathsf{KC}(\mathtt{pp}, n, q, \mathcal{S})$, two messages $\boldsymbol{x}_0, \boldsymbol{x}_1 \in \mathtt{MSP}^n$, and a randomizer $r_0 \in \mathtt{RSP}$, $\mathsf{tdOpen}(\mathtt{pp}, \mathtt{tk}; \boldsymbol{x}_0, r_0, \boldsymbol{x}_1)$ returns a randomizer $r_1 \in \mathtt{RSP}$.

Extraction: for $\mathtt{pp} \leftarrow \mathcal{G}(1^\lambda)$, $\mathcal{S} = (\sigma_1, \ldots, \sigma_{|\mathcal{S}|}) \subseteq [1..n]$ with $1 \leq |\mathcal{S}| \leq q$, $(\mathtt{ck}, (\mathtt{ek}, \mathtt{tk})) \leftarrow \mathsf{KC}(\mathtt{pp}, n, q, \mathcal{S})$, $F : \mathtt{MSP} \to \mathtt{ESP}$ and $c \in \mathtt{CSP}$, $\mathsf{Ext}_F(\mathtt{pp}, \mathtt{ek}; c)$ returns a tuple $(y_{\sigma_1}, \ldots, y_{\sigma_{|\mathcal{S}|}}) \in \mathtt{ESP}^{|\mathcal{S}|}$. We allow F to depend on $\mathtt{pp}$.

Note that SSB commitment schemes are non-interactive and work in the crs model; the latter is needed to achieve trapdoor opening and extractability.

With the current definition, *perfect completeness* is straightforward: to verify that $\boldsymbol{c}$ is a commitment of $\boldsymbol{x}$ with randomizer r, one just recomputes $\boldsymbol{c}' \leftarrow \mathsf{com}(\mathtt{ck}; \boldsymbol{x}; r)$ and checks whether $\boldsymbol{c} = \boldsymbol{c}'$.

An F-extractable SSB commitment scheme COM is *secure* if it satisfies the following security requirements. (See Table 6.1 for a brief summary.)

Abbreviation	Property	Definition
ISH	Index-set hiding	The commitment key reveals nothing about the index-set $\mathcal{S}$
SSB	Somewhere statistically binding	A commitment to $\boldsymbol{x}$ statistically binds the values $\boldsymbol{x}_{\mathcal{S}}$
AESH	Almost everywhere statistically hiding	The commitment is statistically hiding in the indices outside the set $\mathcal{S}$
F-SSE	Somewhere statistical F-extractability	Given a commitment to $\boldsymbol{x}$ and the extraction key, one can extract the values $F(\boldsymbol{x}_S)$

Table 6.1: Properties of an SSB commitment scheme

Index-Set Hiding (ISH): $\forall\lambda$, PPT $\mathcal{A}$, $n = poly(\lambda)$, $q \in [1..n]$, $\mathsf{Adv}^{ish}_{\mathcal{A},\mathsf{COM},n,q} := 2 \cdot |\varepsilon^{\mathsf{ish}}_{\mathcal{A},\mathsf{COM},n,q}(\lambda) - 1/2| \approx_\lambda 0$, where $\varepsilon^{\mathsf{ish}}_{\mathcal{A},\mathsf{COM},n,q}(\lambda) :=$

$$\Pr\left[\begin{array}{l}\mathsf{pp} \leftarrow \mathcal{G}(1^\lambda); (\mathcal{S}_0, \mathcal{S}_1) \leftarrow \mathcal{A}(\mathsf{pp}, n, q) \text{ s.t. } \forall i \in \{0,1\}, \mathcal{S}_i \subseteq [1..n] \wedge |\mathcal{S}_i| \leq q; \\ \beta \leftarrow \{0,1\}; (\mathtt{ck}_\beta, \mathtt{td}_\beta) \leftarrow \mathsf{KC}(\mathsf{pp}, n, q, \mathcal{S}_\beta) : \mathcal{A}(\mathtt{ck}_\beta) = \beta\end{array}\right] .$$

Somewhere Statistically Binding (SSB): $\forall\lambda$, unbounded adversary $\mathcal{A}$, $n = poly(\lambda)$, $q \in [1..n]$, $\mathsf{Adv}^{ssb}_{\mathcal{A},\mathsf{COM},n,q} \approx_\lambda 0$, where $\mathsf{Adv}^{ssb}_{\mathcal{A},\mathsf{COM},n,q} :=$

$$\Pr\left[\begin{array}{l}\mathsf{pp} \leftarrow \mathcal{G}(1^\lambda); \mathcal{S} \leftarrow \mathcal{A}(\mathsf{pp}, n, q) \text{ s.t. } \mathcal{S} \subseteq [1..n] \wedge |\mathcal{S}| \leq q; \\ (\mathtt{ck}, \mathtt{td}) \leftarrow \mathsf{KC}(\mathsf{pp}, n, q, \mathcal{S}); (\boldsymbol{x}_0, \boldsymbol{x}_1, r_0, r_1) \leftarrow \mathcal{A}(\mathtt{ck}) : \\ \boldsymbol{x}_{0,\mathcal{S}} \neq \boldsymbol{x}_{1,\mathcal{S}}; \mathsf{com}(\mathtt{ck}; \boldsymbol{x}_0; r_0) = \mathsf{com}(\mathtt{ck}; \boldsymbol{x}_1; r_1)\end{array}\right] .$$

COM is *somewhere perfectly binding* (SPB) if $\mathsf{Adv}^{ssb}_{\mathcal{A},\mathsf{COM},n,q} = 0$.

Almost Everywhere Statistically Hiding (AESH): $\forall\lambda$, unbounded adversary $\mathcal{A}$, $n = poly(\lambda)$, $q \in [1..n]$, $\mathsf{Adv}^{\mathsf{aesh}}_{\mathcal{A},\mathsf{COM},n,q} := 2 \cdot |\varepsilon^{\mathsf{aesh}}_{\mathcal{A},\mathsf{COM},n,q}(\lambda) - 1/2| \approx_\lambda 0$, where $\varepsilon^{\mathsf{aesh}}_{\mathcal{A},\mathsf{COM},n,q}(\lambda) :=$

$$\Pr\left[\begin{array}{l}\mathsf{pp} \leftarrow \mathcal{G}(1^\lambda); \mathcal{S} \leftarrow \mathcal{A}(\mathsf{pp}, n, q) \text{ s.t. } \mathcal{S} \subseteq [1..n] \wedge |\mathcal{S}| \leq q; \\ (\mathtt{ck}, \mathtt{td}) \leftarrow \mathsf{KC}(\mathsf{pp}, n, q, \mathcal{S}); (\boldsymbol{x}_0, \boldsymbol{x}_1) \leftarrow \mathcal{A}(\mathtt{ck}) \text{ s.t. } \boldsymbol{x}_{0,\mathcal{S}} = \boldsymbol{x}_{1,\mathcal{S}}; \\ \beta \leftarrow \{0,1\}; r \leftarrow \mathtt{RSP} : \mathcal{A}(\mathsf{com}(\mathtt{ck}; \boldsymbol{x}_\beta; r)) = \beta\end{array}\right] .$$

COM is *almost everywhere perfectly hiding* (AEPH) if $\mathsf{Adv}^{\mathsf{aesh}}_{\mathcal{A},\mathsf{COM},n,q} = 0$. If $\mathcal{A}$ is PPT, COM is *almost everywhere computationally hiding* (AECH).

Somewhere Statistical F-Extractability (F-SSE): $\forall\lambda,\ n = poly(\lambda),\ q \in [1..n]$, $\mathcal{S} = (\sigma_1, \ldots, \sigma_{|\mathcal{S}|})$ with $|\mathcal{S}| \leq q$, $(\mathtt{ck}, (\mathtt{ek}, \mathtt{tk})) \leftarrow \mathsf{KC}(\mathsf{pp}, n, q, \mathcal{S})$, and PPT $\mathcal{A}$, $\mathsf{Adv}^{sse}_{\mathcal{A},F,\mathsf{COM},n,q} :=$

$$\Pr\left[\boldsymbol{x}, r \leftarrow \mathcal{A}(\mathtt{ck}) : \mathsf{Ext}_F(\mathsf{pp}, \mathtt{ek}; \mathsf{com}(\mathtt{ck}; \boldsymbol{x}; r)) \neq (F(x_{\sigma_1}), \ldots, F(x_{\sigma_{|\mathcal{S}|}}))\right] \approx_\lambda 0 \ .$$

Additionally, an SSB commitment scheme can but does not have to be *trapdoor*.

Almost Everywhere Statistical Trapdoor (AEST): $\forall\lambda,\ n = poly(\lambda),\ q \in [1..n]$, and unbounded $\mathcal{A}$, $\mathsf{Adv}^{aest}_{\mathcal{A},\mathsf{COM},n,q} \approx_\lambda 0$, where $\mathsf{Adv}^{aest}_{\mathcal{A},\mathsf{COM},n,q} =$

$$\Pr\left[\begin{array}{l}\mathsf{pp} \leftarrow \mathcal{G}(1^\lambda); \mathcal{S} \leftarrow \mathcal{A}(\mathsf{pp}, n, q) \text{ s.t. } \mathcal{S} \subseteq [1..n] \wedge |\mathcal{S}| \leq q; \\ (\mathtt{ck}, \mathtt{td}) = (\mathtt{ek}, \mathtt{tk})) \leftarrow \mathsf{KC}(\mathsf{pp}, n, q, \mathcal{S}); (\boldsymbol{x}_0, r_0, \boldsymbol{x}_1) \leftarrow \mathcal{A}(\mathtt{ck}) \text{ s.t. } \boldsymbol{x}_{0,\mathcal{S}} = \boldsymbol{x}_{1,\mathcal{S}}; \\ r_1 \leftarrow \mathsf{tdOpen}(\mathsf{pp}, \mathtt{tk}; \boldsymbol{x}_0, r_0, \boldsymbol{x}_1) : \mathsf{com}(\mathtt{ck}; \boldsymbol{x}_0; r_0) \neq \mathsf{com}(\mathtt{ck}; \boldsymbol{x}_1; r_1)\end{array}\right] \ .$$

It is *almost everywhere perfect trapdoor (AEPT)* if $\mathsf{Adv}^{aest}_{\mathsf{COM},n,q} = 0$.

It is important to consider the case $|\mathcal{S}| \leq q$ instead of only $|\mathcal{S}| = q$. For example, when $q = n$, the perfectly binding (PB) commitment key ($|\mathcal{S}| = n$) has to be indistinguishable from the perfectly hiding (PH) commitment key ($|\mathcal{S}| = 0$). Moreover, in the applications to construct QA-NIZK argument systems, like those in [67, 68]and Chapter 3, one should not be able to distinguish between the cases $|\mathcal{S}| = 0$ and $|\mathcal{S}| = q$.

F-extractability [17] allows one to model the situation where $x_i \in \mathbb{Z}_p$ but we can only extract the corresponding bracketed value $[x_i]_\iota \in \mathbb{G}_\iota$; similar limited extractability is satisfied say by the Groth-Sahai commitment scheme for scalars [78]. Note that in this case, F depends on pp. Interestingly, extractability implies SSB.

Lemma 30 (F-SSE & F is injective $\Rightarrow$ SSB)**.** *Let* COM *be an SSB commitment scheme. Fix* n *and* q*. Assume* F *is injective. For all PPT* $\mathcal{A}$*, there exists a PPT* $\mathcal{B}$ *such that* $\mathsf{Adv}^{ssb}_{\mathcal{A},\mathsf{COM},n,q} \leq 2 \cdot \mathsf{Adv}^{sse}_{\mathcal{B},F,\mathsf{COM},n,q}$.

Proof. Assume that for given n and q, $\mathcal{A}$ breaks SSB with probability $\mathsf{Adv}^{ssb}_{\mathcal{A},\mathsf{COM},n,q}$. This means that for some $\mathcal{S}$ of cardinality $\leq q$ and honestly generated $\mathtt{ck}$ (w.r.t. $\mathcal{S}$), $\mathcal{A}$ outputs $(\boldsymbol{x}_0, \boldsymbol{x}_1, r_0, r_1)$ such that $\boldsymbol{x}_{0\mathcal{S}} \neq \boldsymbol{x}_{1\mathcal{S}}$ and $C := \mathsf{com}(\mathtt{ck}; \boldsymbol{x}_0; r_0) = \mathsf{com}(\mathtt{ck}; \boldsymbol{x}_1; r_1)$.

Since $\boldsymbol{x}_{0,\mathcal{S}} \neq \boldsymbol{x}_{1,\mathcal{S}}$ and F is injective, we get that $\boldsymbol{F}_0 := (F(x_{0\sigma_1}), \ldots, F(x_{0\sigma_{|\mathcal{S}|}})) \neq (F(x_{1\sigma_1}), \ldots, F(x_{1\sigma_{|\mathcal{S}|}})) =: \boldsymbol{F}_1$. Therefore, there exists $\beta \in \{0,1\}$, such that $\mathsf{Ext}_F(\mathsf{pp}, \mathtt{ek}; C) \neq \boldsymbol{F}_\beta$. Thus, if $\mathcal{B}$ outputs $(\boldsymbol{x}_\beta, r_\beta)$ for $\beta \leftarrow \{0,1\}$, $\mathsf{Adv}^{sse}_{\beta,F,\mathsf{COM},n,q} \geq \mathsf{Adv}^{ssb}_{\mathcal{A},\mathsf{COM},n,q}/2$ and hence $\mathsf{Adv}^{ssb}_{\mathcal{A},\mathsf{COM},n,q} \leq 2 \cdot \mathsf{Adv}^{sse}_{\beta,F,\mathsf{COM},n,q}$. □

If $q = 0$ then AESH is equal to the standard statistical hiding (SH) requirement, and AEST is equal to the standard statistical trapdoor requirement. If $q = n$ then SSB is equal to the standard statistical binding (SB) requirement, and F-SSE is equal to the standard statistical F-extractability requirement. We will show that any secure SSB commitment scheme must also be computationally hiding and binding in the following sense.

Computational Binding (CB): $\forall$ PPT $\mathcal{A}$, $n = poly(\lambda)$, $q \in [1..n]$, where $\mathsf{Adv}^{cb}_{\mathcal{A},\mathsf{COM},n,q} :=$

$$\Pr\left[\begin{array}{l}\mathsf{pp} \leftarrow \mathcal{G}(1^\lambda); \mathcal{S} \leftarrow \mathcal{A}(\mathsf{pp}, n, q) : \mathcal{S} \subseteq [1..n] \wedge |\mathcal{S}| \leq q; \\ (\mathtt{ck}, \mathtt{td}) \leftarrow \mathsf{KC}(\mathsf{pp}, n, q, \mathcal{S}); (\boldsymbol{x}_0, \boldsymbol{x}_1, r_0, r_1) \leftarrow \mathcal{A}(\mathtt{ck}) \\ \text{s.t. } \boldsymbol{x}_0 \neq \boldsymbol{x}_1; \mathsf{com}(\mathtt{ck}; \boldsymbol{x}_0; r_0) = \mathsf{com}(\mathtt{ck}; \boldsymbol{x}_1; r_1)\end{array}\right] \approx_\lambda 0 \ .$$

Computational Hiding (CH): $\forall$ PPT $\mathcal{A}$, $n = poly(\lambda)$, $q \in [1..n]$, $\mathsf{Adv}^{ch}_{\mathcal{A},\mathsf{COM},n,q} := 2 \cdot |\varepsilon^{\mathsf{ch}}_{\mathcal{A},\mathsf{COM},n,q}(\lambda) - 1/2| \approx_\lambda 0$, where $\varepsilon^{\mathsf{ch}}_{\mathcal{A},\mathsf{COM},n,q}(\lambda) :=$

$$\Pr\left[\begin{array}{l}\mathsf{pp} \leftarrow \mathcal{G}(1^\lambda); \mathcal{S} \leftarrow \mathcal{A}(\mathsf{pp}, n, q) \text{ s.t. } \mathcal{S} \subseteq [1..n] \wedge |\mathcal{S}| \leq q; \\ (\mathtt{ck}, \mathtt{td}) \leftarrow \mathsf{KC}(\mathsf{pp}, n, q, \mathcal{S}); (\boldsymbol{x}_0, \boldsymbol{x}_1) \leftarrow \mathcal{A}(\mathtt{ck}); \beta \leftarrow \{0,1\}; \\ r \leftarrow \mathtt{RSP} : \mathcal{A}(\mathsf{com}(\mathtt{ck}; \boldsymbol{x}_\beta; r)) = \beta\end{array}\right] \ .$$

Theorem 31. *Let* COM *be an SSB commitment scheme. Fix* n *and* q*.*

(i) *(ISH + SSB ⇒ CB) For all PPT* $\mathcal{A}$*, there exist PPT* $\mathcal{B}_1$ *and unbounded* $\mathcal{B}_2$*, such that* $\mathsf{Adv}^{cb}_{\mathcal{A},\mathsf{COM},n,q} \leq \mathsf{Adv}^{ish}_{\mathcal{B}_1,\mathsf{COM},n,q} + \frac{n}{q - 4 \cdot \mathsf{Adv}^{ish}_{\mathcal{B}_1,\mathsf{COM},n,q}} \cdot \mathsf{Adv}^{ssb}_{\mathcal{B}_2,\mathsf{COM},n,q}$.

(ii) *(ISH + AESH ⇒ CH) For all PPT* $\mathcal{A}$*, there exist PPT* $\mathcal{B}_1$ *and unbounded* $\mathcal{B}_2$*, such that* $\mathsf{Adv}^{ch}_{\mathcal{A},\mathsf{COM},n,q} \leq \mathsf{Adv}^{ish}_{\mathcal{B}_1,\mathsf{COM},n,q} + \mathsf{Adv}^{aesh}_{\mathcal{B}_2,\mathsf{COM},n,q}$.

Proof. Let $\Pr[\mathrm{Game}_i(\mathsf{Adv}) = 1]$ denote the probability $\mathcal{A}$ wins in Game_i.

(i: ISH + SSB ⇒ CB) We prove the theorem using a sequence of hybrid games, defined as follows, where $\varepsilon_i := \Pr[\mathrm{Game}_i(\mathsf{Adv}) = 1]$.

Game_1 The original computational binding game. For given n and q, by definition $\mathcal{A}$ can break CB with probability $\varepsilon_1 = \mathsf{Adv}^{cb}_{\mathcal{A},\mathsf{COM},n,q}$.

Game_2 Game_1, but instead of $\mathtt{ck}$, $\mathcal{A}$ gets $\mathtt{ck}'$ where $(\mathtt{ck}', \mathtt{td}') \leftarrow \mathsf{KC}(\mathsf{pp}, n, q, \mathcal{S}_1)$ for $\mathcal{S}_1 \leftarrow \mathbb{P}([1..n], q)$. Note that a distinguisher $\mathcal{B}_1$ for Game_1 and Game_2 can be used to break the ISH game with advantage $\varepsilon_{\mathsf{ish}} = \mathsf{Adv}^{ish}_{\mathcal{B}_1,\mathsf{COM},n,q}$. Hence $|\varepsilon_1 - \varepsilon_2| \leq \varepsilon_{\mathsf{ish}}$, which implies that $\varepsilon_2 \geq \varepsilon_1 - \varepsilon_{\mathsf{ish}}$.

We now require the following lemma.

Lemma 32. *Assume $\mathcal{A}$ outputs $(\boldsymbol{x}_0, r_0, \boldsymbol{x}_1, r_1)$ with $\boldsymbol{x}_0 \neq \boldsymbol{x}_1$. Then* $\Pr[(\boldsymbol{x}_0)_{\mathcal{S}_1} \neq (\boldsymbol{x}_1)_{\mathcal{S}_1}$ *in* $\mathsf{Game}_2] \geq q/n - 4 \cdot \varepsilon_{\mathsf{ish}}$.

Proof. Assume for any $\mathcal{S}_1$ of size q sampled uniformly at random, $\mathcal{A}$ can output distinct $\boldsymbol{x}_0, \boldsymbol{x}_1$ such that $\Pr[(x_0)_{\mathcal{S}_1} \neq (x_1)_{\mathcal{S}_1}$ in $\mathsf{Game}_2] = \varepsilon$.

We construct an adversary $\mathcal{B}$ that uses $\mathcal{A}$ to break ISH as follows.

1. Given pp, n, q, $\mathcal{B}$ sets $\mathcal{S}_1 \leftarrow \mathbb{P}([1..n], q)$ and receives $S_0 \leftarrow \mathcal{A}(\mathsf{pp}, n, q)$.
2. $\mathcal{B}$ sends $(\mathcal{S}_0, \mathcal{S}_1)$ to the ISH challenger, and receives $\mathtt{ck}$ corresponding to $\mathcal{S}_\beta$.
3. $\mathcal{B}$ gets $(\boldsymbol{x}_0, r_0, \boldsymbol{x}_1, r_1) \leftarrow \mathcal{A}(\mathtt{ck})$.
   - If $\mathcal{A}$ does not win, abort.
   - If $(\boldsymbol{x}_0)_{\mathcal{S}_1} \neq (\boldsymbol{x}_1)_{\mathcal{S}_1}$ return $\beta' \leftarrow \{0, 1\}$.
   - Else, return 1.

Note that $\beta = 0$ corresponds to Game_1, and $\beta = 1$ corresponds to Game_2. Moreover, for $\beta = 0$, $\mathcal{A}$'s output $(\boldsymbol{x}_0, r_0, \boldsymbol{x}_1, r_1)$ is independent of S_1, in which case $\Pr[(\boldsymbol{x}_0)_{\mathcal{S}_1} \neq (\boldsymbol{x}_1)_{\mathcal{S}_1}] \geq |\mathcal{S}_1|/n = q/n$. Hence we get that if $\mathcal{A}$ wins,

$$\begin{aligned}
\Pr[\mathrm{Game}_{\mathsf{ish}}(\mathcal{B}) = 1] &= \frac{1}{2}\Pr[\mathrm{Game}_{\mathsf{ish}}(\mathcal{B}) = 1 | \beta = 0] + \frac{1}{2}\Pr[\mathrm{Game}_{\mathsf{ish}}(\mathcal{B}) = 1 | \beta = 1] \\
&= \frac{1}{2}\Pr[(x_0)_{\mathcal{S}_1} \neq (x_1)_{\mathcal{S}_1} \text{ in } \mathsf{Game}_1 \wedge \beta' = 0] \\
&+ \frac{1}{2}\Pr[(x_0)_{\mathcal{S}_1} = (x_1)_{\mathcal{S}_1} \text{ in } \mathsf{Game}_2] \\
&+ \frac{1}{2}\Pr[(x_0)_{\mathcal{S}_1} \neq (x_1)_{\mathcal{S}_1} \text{ in } \mathsf{Game}_2 \wedge \beta' = 1] \\
&\geq \frac{q}{4n} + \frac{1-\epsilon}{2} + \frac{\epsilon}{4} \\
&= \frac{1}{2} + \frac{q - n\epsilon}{4n} \ .
\end{aligned}$$

Hence $4 \cdot \varepsilon_{\mathsf{ish}} \geq q/n - \epsilon$, as required. □

It is easy to see that an adversary that wins Game_2 with $(\boldsymbol{x}_0)_{\mathcal{S}_1} \neq (\boldsymbol{x}_1)_{\mathcal{S}_1}$ also wins the SSB game. Hence there exists an adversary $\mathcal{B}_2$ such that

$$\begin{aligned}
\mathsf{Adv}^{ssb}_{\mathcal{B}_2,\mathsf{COM},n,q} &\geq \varepsilon_2 \cdot \Pr[(\boldsymbol{x}_0)_{\mathcal{S}_1} \neq (\boldsymbol{x}_1)_{\mathcal{S}_1} \text{ in } \mathsf{Game}_2 | \boldsymbol{x}_0 \neq \boldsymbol{x}_1] \\
&\geq (\varepsilon_1 - \varepsilon_{\mathsf{ish}})(q/n - 4 \cdot \varepsilon_{\mathsf{ish}}) \text{ (due to Lemma 32).}
\end{aligned}$$

This is equivalent to $\varepsilon_1 \leq \varepsilon_{\mathsf{ish}} + \frac{n}{q - 4 \cdot n \cdot \varepsilon_{\mathsf{ish}}} \cdot \mathsf{Adv}^{ssb}_{\mathcal{B}_2,\mathsf{COM},n,q}$.

(ii: ISH + AESH $\Rightarrow$ CH) Assume that for given n and q, $\mathcal{A}$ can break CH with probability $\mathsf{Adv}^{ch}_{\mathcal{A},\mathsf{COM},n,q}$. Consider the following sequence of games with $\varepsilon_i := \Pr[\mathrm{Game}_i(\mathsf{Adv}) = 1]$.

Game_1**:** In this game, $\mathcal{A}$ breaks CH with probability ε_1. That is, given pp, $\mathcal{A}(\mathsf{pp}, n, q)$ outputs $\mathcal{S}_0$ such that $|\mathcal{S}_0| \leq q$, and for $(\mathtt{ck}_0, \mathtt{td}_0) \leftarrow \mathsf{KC}(\mathsf{pp}, n, q, \mathcal{S}_0)$, $\mathcal{A}(\mathtt{ck}_0)$ outputs $(\boldsymbol{x}_0, \boldsymbol{x}_1)$, s.t. $\Pr[\beta \leftarrow \{0,1\} : \mathcal{A}(\mathsf{com}(\mathtt{ck}_0; \boldsymbol{x}_\beta; r)) = \beta] = \varepsilon_1$.

Game_2**:** In this game, instead of $\mathtt{ck}_0$, $\mathcal{A}$ obtains $\mathtt{ck}_1$ where $(\mathtt{ck}_1, \mathtt{td}_1) \leftarrow \mathsf{KC}(\mathsf{pp}, n, q, \mathcal{S}_1)$ for $\mathcal{S}_1 = \emptyset$. Clearly, for any PPT $\mathcal{A}$ that tries to distinguish Game_1 and Game_2, there exists a PPT $\mathcal{B}_1$, such that $|\varepsilon_2 - \varepsilon_1| \leq \mathsf{Adv}^{ish}_{\mathcal{B}_1,\mathsf{COM},n,q}$.

Let us consider the following AESH adversary $\mathcal{B}_2$ in Game_2.

1. Given pp, n, q, $\mathcal{B}_2$ sets $\mathcal{S}_1 \leftarrow \emptyset$ and receives $S_0 \leftarrow \mathcal{A}(\mathsf{pp}, n, q)$.
2. $\mathcal{B}_2$ computes $(\mathtt{ck}_1, \mathtt{td}_1) \leftarrow \mathsf{KC}(\mathsf{pp}, n, q, \mathcal{S}_1)$ and receives $(\boldsymbol{x}_0, \boldsymbol{x}_1) \leftarrow \mathcal{A}(\mathtt{ck})$.
3. $\mathcal{B}_2$ forwards $(\boldsymbol{x}_0, \boldsymbol{x}_1)$ to the AESH challenger, and receives $c \leftarrow \mathsf{com}(\mathtt{ck}_1, \boldsymbol{x}_\beta; r)$ for some $\beta \leftarrow \{0,1\}$, $r \leftarrow \mathtt{RSP}$.
4. $\mathcal{B}$ gets and outputs $\beta' \leftarrow \mathcal{A}(c)$.

If $\mathcal{A}$ returns the correct β' then clearly also $\mathcal{B}_2$ returns the correct β'. For the success of $\mathcal{B}_2$, it is also needed that $\boldsymbol{x}_{0,\mathcal{S}_1} = \boldsymbol{x}_{1,\mathcal{S}_1}$, which clearly holds since $\mathcal{S}_1 = \emptyset$. Thus, $\mathsf{Adv}^{aesh}_{\mathcal{B}_2,\mathsf{COM},n,q} = \varepsilon_2$. Hence, $\mathsf{Adv}^{ch}_{\mathcal{A},\mathsf{COM},n,q} \leq |\varepsilon_2 - \varepsilon_1| + \varepsilon_2 \leq \mathsf{Adv}^{ish}_{\mathcal{B}_1,\mathsf{COM},n,q} + \mathsf{Adv}^{aesh}_{\mathcal{B}_2,\mathsf{COM},n,q}$. □

6.4 Constructing SSB Commitment Schemes

In this section we generalize the notion of algebraic commitment schemes to general matrix distributions, we show that they work nicely with QA-NIZK arguments and that certain matrix distributions give us an SSB commitment scheme in Section 6.4.1. We focus on the particular case of EMP in Section 6.4.2, where we propose a general version of EMP and prove that it is an SSB commitment scheme.

6.4.1 Algebraic Commitment Schemes

Ràfols and Silva [113] defined the notion of *algebraic commitment schemes (ACSs)*, where the commitment keys are matrices, already used implicitly in other works [35, 38]. Since they behave like linear maps, it is very natural to work with them. We give a more general definition in the following where the matrices are sampled from general distributions.

Definition 34. *Let $\iota \in \{1,2\}$, and let n, m, k be small integers. Let $\mathcal{D}_1$ be a distribution of matrices from $\mathbb{G}_\iota^{k\times n}$ and let $\mathcal{D}_2$ be a distribution of matrices from $\mathbb{G}_\iota^{k\times m}$. A commitment scheme* COM *is a* $(\mathcal{D}_1, \mathcal{D}_2)$-algebraic commitment scheme (ACS) *for vectors in $\mathbb{Z}_p^n$, if for commitment key $\mathtt{ck} = [\mathbf{U}_1, \mathbf{U}_2]_\iota \leftarrow \mathcal{D}_1 \times \mathcal{D}_2$, the commitment of a vector $\boldsymbol{x} \in \mathbb{Z}_p^n$ is computed as a linear map of $\boldsymbol{x}$ and randomness $\boldsymbol{r} \leftarrow \mathbb{Z}_p^m$, i.e., $\mathsf{com}_{\mathtt{ck}}(\boldsymbol{x}, \boldsymbol{r}) := [\mathbf{U}_1]_\iota \boldsymbol{x} + [\mathbf{U}_2]_\iota \boldsymbol{r} \in \mathbb{G}_\iota^k$.*

Ràfols and Silva mention that given different commitment key matrices, their distributions are computationally indistinguishable under the MDDH assumption, and each concrete distribution defines which coordinates of the commitments are SB or SH. We prove in the following that it also gives a characterization of the coordinates of the key matrices for the different SSB properties (AECH, ISH, SPB, SPE) based on linear dependency and we also prove that to extract q elements from an ACS we need at least $q+1$ rows.

ACS as SSB commitment schemes. We will show that ACS defined in Section 6.4 are computationally hiding under MDDH. They are also perfectly binding in those components that correspond to the linearly independent columns of $\mathbf{U}_1$. If they are also pair-wise to columns of $\mathbf{U}_2$, the system of equations has maximum rank and unique solution. We give this characterisation in Lemma 33.

Moreover, for extraction assume that $\mathrm{span}\{\mathbf{U}_1\} \cap \mathrm{span}\{\mathbf{U}_2\} = \{\mathbf{0}\}$. Intuitively, $\mathbf{U}_1$ defines the space of the opening $\boldsymbol{x}$, while $\mathbf{U}_2$ defines the randomness space. To extract in q positions, we hence need $\mathtt{ek}$ is such that $\mathtt{ek}[\mathbf{U}_2]_\iota = \mathbf{0}$ and $\mathtt{ek}[\mathbf{U}_1]_\iota = (\boldsymbol{b}_i)_{i=1}^n$, where $\boldsymbol{b}_i$ is $\boldsymbol{e}_i$ in q positions and $\mathbf{0}$ elsewhere. Then by the linearity of ACS, $\mathtt{ek} \cdot \mathsf{com}_{\mathtt{ck}}(\boldsymbol{x}, \boldsymbol{r}) = \mathtt{ek} \cdot [\mathbf{U}_1]_\iota \boldsymbol{x} = [\boldsymbol{x}]_\iota$.

Lemma 33. *Let $n \geq 1$ and $q \leq n$. Let* COM *be an ACS with commitment key $\mathtt{ck} = [\mathbf{U}_1, \mathbf{U}_2]_\iota$ sampled from $\mathcal{D}_1 \times \mathcal{D}_2$ as defined in Definition 34.*

(i) COM *is AECH under $\mathcal{D}_2$-MDDH$_{\mathbb{G}_\iota}$.*

(ii) COM *is ISH under $\mathcal{D}_1, \mathcal{D}_2$-MDDH$_{\mathbb{G}_\iota}$.*

(iii) COM *is SPB if $\mathbf{U}_1$ has rank q and* $\mathrm{span}\{\mathbf{U}_1\} \cap \mathrm{span}\{\mathbf{U}_2\} = \{\mathbf{0}\}$.

(iv) COM *is $[\cdot]_\iota$-SPE if $\mathbf{U}_1$ has rank q and* $\mathrm{span}\{\mathbf{U}_1\} \cap \mathrm{span}\{\mathbf{U}_2\} = \{\mathbf{0}\}$.

Proof. Let $\mathcal{S} \subseteq [1..n]$, $|\mathcal{S}| \leq q$ be the indices of $\boldsymbol{x}$ one can extract during opening.

(i: AECH) Let $\mathcal{A}$ be an adversary that breaks AECH with non-negligible probability, say $\varepsilon_{\mathcal{A}}$. Consider the following $\mathbb{G}_\iota$-MDDH adversary $\mathcal{B}$. $\mathcal{B}$ receives a challenge $[\mathbf{A}, \boldsymbol{y}_\beta]_\iota$ where $\mathbf{A} \leftarrow \mathcal{D}_2$, $\boldsymbol{y}_0 \leftarrow \mathbb{Z}_p^k$, and $\boldsymbol{y}_1 \leftarrow \mathbf{A}\boldsymbol{r}$ for $\boldsymbol{r} \leftarrow \mathbb{Z}_p^m$. $\mathcal{B}$ sets $[\mathbf{U}_2]_\iota \leftarrow [\mathbf{A}]_\iota$,

and generates $\mathbf{U}_1$ from the distribution $\mathcal{D}_1$. $\mathcal{B}$ sends $\mathtt{ck} = [\mathbf{U}_1, \mathbf{U}_2]_\iota$ to $\mathcal{A}$ who replies with two messages $\boldsymbol{x}_0, \boldsymbol{x}_1$, such that $\boldsymbol{x}_{0,\mathcal{S}}, \boldsymbol{x}_{1,\mathcal{S}}$. $\mathcal{B}$ computes $\boldsymbol{c}_0 \leftarrow [\mathbf{U}_1]_\iota \boldsymbol{x}_0 + [\mathbf{U}_2]_\iota \boldsymbol{r}$, for $\boldsymbol{r} \leftarrow \mathbb{Z}_p^m$, and $\boldsymbol{c}_1 \leftarrow [\mathbf{U}_1]_\iota \boldsymbol{x}_1 + [\boldsymbol{y}_\beta]_\iota$. $\mathcal{B}$ picks $\beta' \leftarrow \{0,1\}$ and sends $c_{\beta'}$ to $\mathcal{A}$. $\mathcal{A}$ guesses which message was committed by returning $\beta_\mathcal{A} \in \{0,1\}$ to $\mathcal{B}$. $\mathcal{B}$ sends $\beta_\mathcal{A}$ to the MDDH challenger. Clearly,

$$\begin{aligned}\Pr[\beta_\mathcal{A} = \beta] =& \Pr[\beta_\mathcal{A} = 0 | \beta = 0]/2 + \Pr[\beta_\mathcal{A} = 1 | \beta = 1]/2 \\ =& \varepsilon_\mathcal{A}/2 + (\Pr[\beta_\mathcal{A} = 1 | \beta = 1, \beta' = 0]/2 + \Pr[\beta_\mathcal{A} = 1 | \beta = 1, \beta' = 1]/2)/2 \\ =& \varepsilon_\mathcal{A}/2 + \varepsilon_\mathcal{A}/4 + \varepsilon_\mathcal{A}/8 = 7/8 \cdot \varepsilon_\mathcal{A} \ .\end{aligned}$$

Thus if $\mathcal{A}$ succeeded with non-negligible probability, then so did $\mathcal{B}$.

(ii: ISH) Firstly we prove that for any $\mathcal{S}_0$ with $|\mathcal{S}_0| \leq n$, if $\mathcal{S}_1 = \mathcal{S}_0 \cup \{i^*\}$ for some $i^* \notin \mathcal{S}_0$ and $\mathcal{S}_0, \mathcal{S}_1 \subseteq [1..n]$, then $\mathcal{D}_{1,2}^{0,q} := ([\mathcal{D}_{n,k}^{\mathcal{S}_0}]_\iota, [\mathcal{D}_{m,k}^{\mathcal{S}_0}]_\iota)$ and $\mathcal{D}_{1,2}^{1,q} := ([\mathcal{D}_{n,k}^{\mathcal{S}_1}]_\iota, [\mathcal{D}_{m,k}^{\mathcal{S}_1}]_\iota)$ are computationally indistinguishable under MDDH. Let $\mathcal{A}$ be an adversary that can distinguish $\mathcal{D}_{1,2}^{0,q}$ and $\mathcal{D}_{1,2}^{1,q}$. We construct the following MDDH adversary $\mathcal{B}$ that receives a challenge $[\mathbf{A}, \boldsymbol{y}_\beta]_\iota$ where $\mathbf{A}_1, \mathbf{A}_2 \leftarrow \mathcal{D}_{1,2}^0$, $\boldsymbol{y}_0 \leftarrow \mathbb{Z}_p^k$, and $\boldsymbol{y}_1 \leftarrow (\mathbf{A}_1^\top | \mathbf{A}_2^\top)\boldsymbol{r}$ for $\boldsymbol{r} \leftarrow \mathbb{Z}_p^m$. $\mathcal{B}$ sets $[\mathbf{U}_1]_\iota \leftarrow [\mathbf{A}_1]_\iota$, and $[\mathbf{U}_2]_\iota \leftarrow ([\mathbf{A}_2]_\iota | [\boldsymbol{y}_\beta]_\iota)$. $\mathcal{B}$ computes $\boldsymbol{c}_\beta \leftarrow [\mathbf{U}_1]_\iota \boldsymbol{x} + [\mathbf{U}_2]_\iota \boldsymbol{r}$, for $\boldsymbol{r} \leftarrow \mathbb{Z}_p^m$ and sends $\boldsymbol{c}_\beta$ to $\mathcal{A}$ who replies with $\beta_\mathcal{A}$. Thus, $\mathcal{B}$ has the same advantage in breaking MDDH as $\mathcal{A}$ has in distinguishing $\mathcal{D}_{1,2}^{0,q}$ and $\mathcal{D}_{1,2}^{1,q}$.

Now, for any sets $\mathcal{S}_0$ and $\mathcal{S}_1$ it holds that $\mathsf{Adv}^{indist}_{\mathcal{A},\mathcal{D}_{1,2}^0,\mathcal{D}_{1,2}^1} \leq (|\mathcal{S}_0 \cup \mathcal{S}_1| - |\mathcal{S}_0 \cap \mathcal{S}_1|) \cdot \mathsf{Adv}^{\mathsf{MDDH}}_{\mathcal{B},\mathcal{D}_{1,2}^{n,q},\mathcal{G}}$.

(iii: SPB) Assume that all columns of $\mathbf{U}_1$ and $\mathbf{U}_2$ are pairwise linearly independent. Consider the matrix system of equations defined by $(\mathbf{U}_1, \mathbf{U}_2)\left(\begin{smallmatrix}\boldsymbol{x}\\ \boldsymbol{r}\end{smallmatrix}\right) = \mathsf{com}_{\mathtt{ck}}(\boldsymbol{x}, \boldsymbol{r})$. This system has a unique solution because the matrix has full rank. Hence, each commitment corresponds to a unique vector $\left(\begin{smallmatrix}\boldsymbol{x}\\ \boldsymbol{r}\end{smallmatrix}\right)$. Now, if $\mathbf{U}_1$ has q columns pair-wise linear independent and columns of $\mathbf{U}_2$ pair-wise linear independent to all of them, consider the system that has a matrix with those q columns of $\mathbf{U}_1$ and the whole $\mathbf{U}_2$. Its rank is maximum as well and the result follows.

(iv: $[\cdot]$-SPE) Since $k > m$, for any matrix $\mathbf{U}_2$ of size $k \times m$ there exist matrices $\mathtt{ek} \in \mathbf{U}_2^\perp$ that define orthogonal spaces of $\mathbf{U}_2$ of size $k' \times k$ for $k' \geq k - m$ such that $\mathtt{ek} \cdot \mathbf{U}_2 = \begin{pmatrix} \mathbf{0}_{(k-m)\times m} \\ \boldsymbol{a} \end{pmatrix}$ where $\boldsymbol{a} \in \mathbb{Z}_p^{(k'-k+m)\times m}$. This space has at least dimension 1 because $k > m$. Moreover, there exists an appropriate change of basis of the space such that $\mathtt{ek} \cdot \mathbf{U}_1 = \left(\begin{array}{c|c} \mathbf{I}_q & \boldsymbol{b}_2 \\ \boldsymbol{b}_1 & \end{array}\right)$ where $\boldsymbol{b}_1 \in \mathbb{Z}_p^{(k'-q)\times q}, \boldsymbol{b}_2 \in \mathbb{Z}_p^{k' \times (n-q)}$. This is well-defined since $k - m \geq q$ and if q columns of the matrices are pair-wise linear independent then $k' - q \geq k - m - q \geq 0$. □

Corollary 34. *The minimum size of the $k \times m$ matrix to guarantee $[\cdot]_\iota$-extraction of $n \geq 1$ elements is $k = n + 1$, $m = 1$.*

Proof. Information theoretically the commitment size should be no less than the dimension of the opening in order to extract it completely, so $k \geq n$. The orthogonal space has to be at least of dimension 1 in order to provide extraction, so the minimal difference is $k - m \geq 1$. We have $k \geq n + m$ directly by the linear independence of the columns in matrices $\boldsymbol{U}_1, \boldsymbol{U}_2$. Hence, the minimal constants are $m = 1$, $k = n + 1$. □

ACS and QA-NIZK arguments. Algebraic commitments are suitable to work with QA-NIZK arguments for linear spaces because most of their properties can be expressed in terms of membership or non-membership to certain linear subspaces. For example, the works of González *et al.* [67, 68] and our construction in Chapter 3 implicitly use an SSB commitment scheme COM to construct efficient QA-NIZK argument systems based on falsifiable assumptions. The soundness of their QA-NIZK system depends on the ISH, SSB, and SSE properties, while the zero-knowledge property depends on the AESH and CH properties. On the other hand, honest parties never need to actually open the commitment; the opening (more precisely, extraction) is only done inside the security proof by using the SSE property[4]. Moreover, in our QA-NIZK argument in Section 6.6.2, as well as Chapter 3, we use functional SSB commitments since linear EMP is more straightforward to our use of it in the soundness proof.

6.4.2 The EMP Commitment Scheme

Extended Multi-Pedersen (EMP) commitment [67, 68] is a variant of the standard vector Pedersen commitment scheme [110]. In this section, we will depict a general version of the EMP commitment scheme[5] in group $\mathbb{G}$. We redefine EMP by using a division of the generator matrix $\boldsymbol{g}$ as a product of two matrices $\mathbf{R}$ and $\mathbf{M}$; this representation results in very short security proofs for EMP. To simplify notation, we will write Ext instead of $\mathsf{Ext}_{[\cdot]}$. We use a distribution $\mathcal{D}_{q+1}^{p,n,\mathcal{S}}$ that outputs $n+1$ vectors $\boldsymbol{g}^{(i)}$, such that if $i \in \mathcal{S}' = \mathcal{S} \cup \{n+1\}$ then $\boldsymbol{g}^{(i)}$ is distributed uniformly over $\mathbb{Z}_p^{q+1}$, and otherwise $\boldsymbol{g}^{(i)}$ is a random scalar multiple of $\boldsymbol{g}^{(n+1)}$.[6]

Definition 35. *Let $p = p(\lambda)$, $n = poly(\lambda)$, and let $q \leq n$ be a small positive integer. Let $\mathcal{S} \subseteq [1..n]$ with $|\mathcal{S}| \leq q$. Then the distribution $\mathcal{D}_{q+1}^{p,n,\mathcal{S}}$ is defined as the first part*

[4] In this sense, one could also call them trapdoor hash functions [47] with the SSB and AESH properties

[5] González *et al.* [68] mostly considered the case $q = 1$; they also did not formalize its security by using notions like ISH

[6] We add $+1$ to the dimension (e.g., $q + 1$) to accommodate the randomizer in EMP.

of $\mathcal{D}_{gen}(p, n, \mathcal{S}, q)$ in Figure 6.1 (i.e., just $\boldsymbol{g}$, without the associated extraction key or trapdoor).

Note that [68] uses a distribution $\mathcal{D}_{q+1,k}$ instead of the uniform distribution $\mathcal{U}_{q+1}$ over $\mathbb{Z}_p^{q+1}$, which means that taking a larger k gives a weaker security assumption but with worse efficiency. Our version of EMP also works with a general distribution, but for ease of presentation we only use $\mathcal{U}_{q+1}$.

$\underline{\mathcal{D}_{gen}(p, n, \mathcal{S}, q)}$
$\mathcal{S}' \leftarrow \mathcal{S} \cup \{n+1\}$; // $\mathcal{S}' = \{\sigma_1, \ldots, \sigma_{q+1}\}$
$\mathbf{R} \leftarrow \mathbb{Z}_p^{q+1\times(q+1)}, \mathbf{M} \leftarrow \mathbf{0}_{(q+1)\times(n+1)}; M_{q+1,n+1} \leftarrow 1$;
for $j = 1$ to n, do
  if $j \notin \mathcal{S}'$, then $M_{q+1,j} = \delta_j \leftarrow \mathbb{Z}_p$;
  else let i be such that $j = \sigma_i$; $M_{i,\sigma_i} \leftarrow 1$;
$\boldsymbol{g} \leftarrow \mathbf{R}\mathbf{M}$; $\mathtt{tk} \leftarrow (\delta_j)_{j\in[1..n]\setminus\mathcal{S}}$; // $\boldsymbol{g} \in \mathbb{Z}_p^{(q+1)\times(n+1)}$;
return $(\boldsymbol{g}, \mathbf{R}, \mathtt{tk})$;

Figure 6.1: Generating $\mathcal{D}_{q+1}^{p,n,\mathcal{S}}$, with associated extraction key $\mathbf{R}$ and trapdoor $\mathtt{tk}$.

Example 1. *In the Groth-Sahai commitment scheme, $n = q = 1$, so $\mathcal{D}_{gen}$ first samples $\mathbf{R} = \left(\begin{smallmatrix} r_{11} & r_{12} \\ r_{21} & r_{22} \end{smallmatrix}\right) \leftarrow \mathbb{Z}_p^{2\times 2}$. If $\mathcal{S} = \{1\}$ then $\mathbf{M} = \left(\begin{smallmatrix} 1 & 0 \\ 0 & 1 \end{smallmatrix}\right)$ and $\boldsymbol{g} = \mathbf{R}\mathbf{M} = \left(\begin{smallmatrix} r_{11} & r_{12} \\ r_{21} & r_{22} \end{smallmatrix}\right)$. On the other hand, if $\mathcal{S} = \emptyset$ then $\mathbf{M} = \left(\begin{smallmatrix} 0 & 0 \\ \delta_1 & 1 \end{smallmatrix}\right)$ and $\boldsymbol{g} = \mathbf{R}\mathbf{M} = \left(\begin{smallmatrix} \delta_1 r_{12} & r_{12} \\ \delta_1 r_{22} & r_{22} \end{smallmatrix}\right)$ for $\delta_1 \leftarrow \mathbb{Z}_p$.*

Consider the case $n = 3$, $q = 2$, and $\mathcal{S} = \{3\}$. Then

$$\mathbf{M} = \left(\begin{smallmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ \delta_1 & \delta_2 & 0 & 1 \end{smallmatrix}\right), \quad \boldsymbol{g} = \mathbf{R}\mathbf{M} = \left(\begin{smallmatrix} \delta_1 r_{13} & \delta_2 r_{13} & r_{11} & r_{13} \\ \delta_1 r_{23} & \delta_2 r_{23} & r_{21} & r_{23} \\ \delta_1 r_{33} & \delta_2 r_{33} & r_{31} & r_{33} \end{smallmatrix}\right), \text{ for } \delta_1, \delta_2 \leftarrow \mathbb{Z}_p, \mathbf{R} \leftarrow \mathbb{Z}_p^{3\times 3} \ .$$

The following lemma shows that distributions $[\mathcal{D}_{q+1}^{p,n,\mathcal{S}}]$ for different sets $\mathcal{S}$ are indistinguishable under the MDDH assumption.

Lemma 35. *Let $\iota \in \{1, 2\}$. Let $p = p(\lambda)$ be created by $\mathcal{G}(1^\lambda)$, $n = poly(\lambda)$, and let $q \le n$ be a positive integer. Let $\mathcal{S} \subseteq [1..n]$ with $|\mathcal{S}| \le q$. The distribution families $\mathcal{D}^0 := \{[\mathcal{D}_{q+1}^{p,n,\mathcal{S}}]\}_\lambda$ and $\mathcal{D}^1 := \{[\mathcal{D}_{q+1}^{p,n,\emptyset}]\}_\lambda$ are computationally indistinguishable under the $\mathcal{U}_{q+1}$-$\mathsf{MDDH}_{\mathbb{G}_\iota}$ assumption relative to $\mathcal{G}$: for any PPT $\mathcal{A}$, there exists a PPT $\mathcal{B}$, such that $\mathsf{Adv}^{indist}_{\mathcal{A},\mathcal{D}^0,\mathcal{D}^1} \le |\mathcal{S}| \cdot \mathsf{Adv}^{\mathsf{MDDH}}_{\mathcal{B},\mathcal{U}_{q+1},\mathcal{G}}$.*

Proof. Fix λ. We first prove that for any $\mathcal{S}_0$ with $|\mathcal{S}_0| \le q-1$, if $\mathcal{S}_1 = \mathcal{S}_0 \cup \{i^*\}$ for $i^* > \max_i\{i \in \mathcal{S}_0\}$ and $\mathcal{S}_0, \mathcal{S}_1 \subseteq [1..n]$, then $\mathcal{D}_0 := [\mathcal{D}_{q+1}^{p,n,\mathcal{S}_0}]$ and $\mathcal{D}_1 := [\mathcal{D}_{q+1}^{p,n,\mathcal{S}_1}]$ are computationally indistinguishable.

$\underline{\mathsf{KC}(p, n, \mathcal{S}) \;/\!/\; \mathcal{S} \subseteq \{1, 2, \ldots, n\} \text{ with } |\mathcal{S}| \leq q}$
Sample $(\boldsymbol{g}, \mathbf{R}, \mathtt{tk}_\iota) \leftarrow \mathcal{D}_{gen}(p, n, \mathcal{S}, q)$ s.t. $\mathbf{R}$ has full rank;
$\mathtt{ck} \leftarrow [\boldsymbol{g}]$; $\mathtt{ek} \leftarrow \mathbf{R}$; $/\!/ \boldsymbol{g} \in \mathbb{Z}_p^{(q+1)\times(n+1)}$, $\mathbf{R} \in \mathbb{Z}_p^{(q+1)\times(q+1)}$
$\mathtt{td} \leftarrow (\mathtt{ek}, \mathtt{tk})$; return $(\mathtt{ck}, \mathtt{td})$;

$\underline{\mathsf{com}(\mathtt{ck}; \boldsymbol{x} \in \mathbb{Z}_p^n; r \in \mathbb{Z}_p)}$
return $[\boldsymbol{g}]\binom{\boldsymbol{x}}{r}$; $/\!/ = \sum_{j=1}^{n} x_j[g^{(j)}] + r[g^{(n+1)}] \in \mathbb{G}^{q+1}$

$\underline{\mathsf{tdOpen}(\mathsf{pp}, \mathtt{tk}_\iota; \boldsymbol{x}_0, r_0, \boldsymbol{x}_1)}$
$r_1 \leftarrow \sum_{i\in[1..n]\setminus\mathcal{S}}(x_{0,i} - x_{1,i})\delta_i + r_0$;
return r_1;

$\underline{\mathsf{Ext}(\mathsf{pp}, \mathtt{ek}; [\mathbf{c}])}$
$[\boldsymbol{x}'] \leftarrow \mathbf{R}^{-1}[\mathbf{c}]$;
return $[\boldsymbol{x}_{\mathcal{S}}] \leftarrow [\boldsymbol{x}'_{[1..|\mathcal{S}|]}]$;

Figure 6.2: The EMP commitment scheme COM

Let $\mathcal{A}$ be an adversary that can distinguish $\mathcal{D}_0$ and $\mathcal{D}_1$. We construct the following MDDH adversary $\mathcal{B}$. The challenger $\mathcal{C}$ of the MDDH game samples $\mathbf{A} \leftarrow \mathbb{Z}_p^{q+1}$ and $\boldsymbol{w} \leftarrow \mathbb{Z}_p$. If $\beta = 0$ then $\mathcal{C}$ samples $\boldsymbol{y} \leftarrow \mathbb{Z}_p^{q+1}$, otherwise $\mathcal{C}$ sets $\boldsymbol{y} \leftarrow \mathbf{A}\boldsymbol{w}$. $\mathcal{C}$ sends $(\mathsf{pp}, [\mathbf{A}, \boldsymbol{y}]_\iota)$ to $\mathcal{B}$. $\mathcal{B}$ does the following:

$\underline{\mathcal{B}(\mathsf{pp}, [\mathbf{A}, \boldsymbol{y}])}$
$[\boldsymbol{g}^{(n+1)}] \leftarrow [\mathbf{A}]$;
for i in $[1..n]$, do
  if $i = i^*$, then $[\boldsymbol{g}^{(i)}] \leftarrow [\boldsymbol{y}]$;
  elseif $i \in \mathcal{S}_0$, then $\boldsymbol{g}^{(i)} \leftarrow \mathbb{Z}_p^{q+1}$;
  else, $\delta_i \leftarrow \mathbb{Z}_p$; $[\boldsymbol{g}^{(i)}] \leftarrow [\boldsymbol{g}^{(n+1)}]\delta_i$;
return $\beta \leftarrow \mathcal{A}(\mathsf{pp}, [\boldsymbol{g}])$;

Clearly, $[\boldsymbol{g}]$ is distributed according to $\mathcal{D}_\beta$. Thus, $\mathcal{B}$ has the same advantage in breaking MDDH as $\mathcal{A}$ has in distinguishing $\mathcal{D}_0$ from $\mathcal{D}_1$. By using a standard hybrid argument, $\mathsf{Adv}^{indist}_{\mathcal{A},\mathcal{D}^0,\mathcal{D}^1} \leq |\mathcal{S}| \cdot \mathsf{Adv}^{\mathsf{MDDH}}_{\mathcal{B},\mathcal{U}_{q+1},\mathcal{G}}$.

□

As a simple generalization of Lemma 35, for any $\mathcal{S}_0, \mathcal{S}_1 \subseteq [1..n]$ with $\mathcal{S}_i \leq q$, $\mathsf{Adv}^{indist}_{\mathcal{A},[\mathcal{D}^{p,n,\mathcal{S}_0}_{q+1}],[\mathcal{D}^{p,n,\mathcal{S}_1}_{q+1}]} \leq |\mathcal{S}_0 \triangle \mathcal{S}_1| \cdot \mathsf{Adv}^{mddh}_{\mathcal{B},\mathcal{U}_{q+1},\mathcal{G}}$.

We define EMP in Figure 6.2. We claim that it is indeed an SSB commitment scheme in the following Theorem.

Theorem 36. *Let $\mathcal{G}$ be a bilinear group generator. Fix λ, n, and q. The EMP commitment scheme is*

(i) *ISH under the $\mathcal{U}_{(q+1)}$-$\mathsf{MDDH}_{\mathbb{G}_\iota}$ assumption,*

(ii) *F-SSE for $F = [\cdot]$ (thus, F depends on pp),*

(iii) *AEPT,*

(iv) *SPB,*

(v) *AEPH,*

(vi) *CB and CH under the $\mathcal{U}_{(q+1)}$-MDDH$_{\mathbb{G}_\iota}$ assumption.*

Proof. **(i: ISH)** Due to the properties of $\mathcal{D}_{q+1}^{p,n,\mathcal{S}}$, $\boldsymbol{g}^{(\mathcal{S}\cup\{n+1\})}$ has columns distributed uniformly over $\mathbb{Z}_p^{q+1}$ and hence by the Schwartz-Zippel lemma has full rank with probability $\geq 1-(q+1)/p$. It follows from Lemma 35 that for any PPT $\mathcal{A}$, there exists a PPT $\mathcal{B}$, such that $\mathsf{Adv}^{ish}_{\mathcal{A},\mathsf{COM},n,q} \leq q \cdot \mathsf{Adv}^{\mathsf{MDDH}}_{\mathcal{B},\mathcal{U}_{(q+1)\times(n+1)},\iota,\mathcal{G}} + (q+1)/p$.

(ii: $[\cdot]$-SSE) We have $[\mathbf{c}] = [\boldsymbol{g}](\begin{smallmatrix}\boldsymbol{x}\\ r\end{smallmatrix}) = [\mathbf{RM}](\begin{smallmatrix}\boldsymbol{x}\\ r\end{smallmatrix})$ for some $(\begin{smallmatrix}\boldsymbol{x}\\ r\end{smallmatrix})$, where $\mathbf{R}$ has full rank. But then $[\boldsymbol{x}'] = \mathbf{R}^{-1}[\mathbf{c}] = [\mathbf{M}](\begin{smallmatrix}\boldsymbol{x}\\ r\end{smallmatrix})$. Let $\mathcal{S} = \{\sigma_i\}$. By the definition of $\mathbf{M}$, clearly $x'_i = \mathbf{M}_i(\begin{smallmatrix}\boldsymbol{x}\\ r\end{smallmatrix}) = x_{\sigma_i}$ for $i \leq |\mathcal{S}|$.

(iii: AEPT) Let $\boldsymbol{x}_0 \neq \boldsymbol{x}_1$ but $\boldsymbol{x}_{0,\mathcal{S}} = \boldsymbol{x}_{1,\mathcal{S}}$. Then $\mathsf{com}(\mathsf{ck};\boldsymbol{x}_0;r_0) - \mathsf{com}(\mathsf{ck};\boldsymbol{x}_1;r_1) = \mathbf{RM}(\begin{smallmatrix}\boldsymbol{x}_0-\boldsymbol{x}_1\\ r_0-r_1\end{smallmatrix}) = \mathbf{R}\Big(\begin{smallmatrix}\mathbf{0}_q\\ \sum_{i\in[1..n]\setminus\mathcal{S}}(x_{0,i}-x_{1,i})\delta_i+(r_0-r_1)\end{smallmatrix}\Big) = \mathbf{0}_{q+1}$, since from tdOpen, $r_1 = \sum_{i\in[1..n]\setminus\mathcal{S}}(x_{0,i}-x_{1,i})\delta_i + r_0$.

(iv: SPB) Since $F = [\cdot]$ is injective (because the bilinear group has a prime order), this follows from Theorem 36 and Lemma 30.

(v: AEPH) Let $\boldsymbol{x}_0, \boldsymbol{x}_1$ be such that $\boldsymbol{x}_{0,\mathcal{S}} = \boldsymbol{x}_{1,\mathcal{S}}$. Then $\mathbf{M}(\begin{smallmatrix}\boldsymbol{x}_0\\ r_0\end{smallmatrix}) = (\boldsymbol{x}_{0,\mathcal{S}}^\top, 0, \ldots, 0, r_0 + \sum_{i\in[1..n]\setminus\mathcal{S}} x_{0,i}\sigma_i)^\top$ and similarly $\mathbf{M}(\begin{smallmatrix}\boldsymbol{x}_1\\ r_1\end{smallmatrix}) = ((\boldsymbol{x}_{1,\mathcal{S}})^\top, 0, \ldots, 0, r_1 + \sum_{i\in[1..n]\setminus\mathcal{S}} x_{1,i}\sigma_i)^\top$. Thus, both have first q elements equal and the last element is uniformly random. Clearly then also $\mathsf{com}(\mathsf{ck};\boldsymbol{x}_0;r_0) = \mathbf{RM}(\begin{smallmatrix}\boldsymbol{x}_0\\ r_0\end{smallmatrix})$ and $\mathsf{com}(\mathsf{ck};\boldsymbol{x}_1;r_1) = \mathbf{RM}(\begin{smallmatrix}\boldsymbol{x}_1\\ r_1\end{smallmatrix})$ are indistinguishable.

(vi: CB and CH): Follows from Theorem 31, Theorem 36, SPB and AEPH. □

Alternative constructions

One can also construct a SSB commitment from any IND-CPA secure cryptosystem if both the message space and the randomness space are additively homomorphic, i.e., $\mathsf{Enc}_{\mathsf{pk}}(m_1;r_1) + \mathsf{Enc}_{\mathsf{pk}}(m_2;r_2) = \mathsf{Enc}_{\mathsf{pk}}(m_1+m_2;r_1+r_2)$ for any public key pk, messages m_1, m_2 and randomness r_1, r_2. For simplicity, consider the case when $q = 1$ and

the i-th index is binding. We can set $\mathtt{ck} = (\mathsf{pk}, \boldsymbol{c} := (\mathsf{Enc}_{\mathsf{pk}}(e_{i,1}; r_1), \ldots, \mathsf{Enc}_{\mathsf{pk}}(e_{i,n}; r_n))$, $\mathtt{tk} = \mathsf{sk}$ where $\boldsymbol{e}_i$ is the i-th unit vector. In order to commit to $\boldsymbol{x}$, we compute $\boldsymbol{c} \cdot \boldsymbol{x} + \mathsf{Enc}_{\mathsf{pk}}(0; r) = \mathsf{Enc}_{\mathsf{pk}}(x_i, r + \sum_{i=1}^{n} r_i)$ for $r \leftarrow \mathtt{RSP}$. Now, ISH follows directly from the IND-CPA security, SSB and F-SSE follow from the correctness of the cryptosystem, and AESH follows since $\mathsf{Enc}_{\mathsf{pk}}(x_i, r + \sum_{i=1}^{n} r_i)$ only depends on x_i. However, we obtain a less efficient construction than EMP. E.g., if we instantiate with lifted Elgamal we would have a commitment size of $2q$ group elements, whereas EMP has $q + 1$.

The above is similar to the technique of obtaining 2-message oblivious transfer (OT) from additively homomorphic cryptosystems [8] and this is no coincidence. SSB commitments can indeed be constructed from OT, and we can conversely construct OT from SSB commitments. Hence there are various alternative constructions of SSB, but in this paper we concentrate on EMP due to the applications we are interested in. See Section 6.7.2 for more details.

6.5 Functional SSB Commitments

We generalize the notion of SSB commitments from being statistically binding on an index-set $\mathcal{S} \subseteq [1..n]$ to being statistically binding on outputs of the functions $\{f_i\}_{i=1}^{q}$ from some function family $\mathcal{F}$. We construct a functional SSB commitment scheme for the case when $\mathcal{F}$ is the set of linear functions. In particular, this covers functions $f_j(\boldsymbol{x}) = x_j$ and hence we also have the index-set functionality of EMP commitment.

In our definition, given a family of functions $\mathcal{F}$ we require that the commitment key $\mathtt{ck}$ will hide the functions $\{f_i\}_{i=1}^{q} \subset \mathcal{F}$ and given a commitment $\mathsf{com}(\mathtt{ck}; \boldsymbol{x}; r)$ and an extraction key $\mathtt{ek}$ it is possible to F-extract $f_i(\boldsymbol{x})$ for $i \in [1..q]$, i.e. if F is the exponentiation function in the group, $[f_i(\boldsymbol{x})]_\iota$. The commitment uniquely determines the outputs of the functions (due to the SSB property) and commitments to messages which produce equal function outputs are statistically indistinguishable (due to the AESH property). Our definition is similar to Döttling et al.'s [47] definition for trapdoor hash functions for a family of predicates $\mathcal{F}$.

6.5.1 Definitions

Essentially the only difference between an SSB commitment and a functional SSB commitment is that in the former $\mathcal{S}$ is a subset of $[1..q]$ and in the latter $\mathcal{S}$ is a subset of some function-set $\mathcal{F}$. For the sake of completeness we provide the formal definition below.

Definition 36. *An F-extractable functional SSB commitment scheme* $\mathsf{COM} = (\mathcal{G}, \mathsf{KC}, \mathsf{com}, \mathsf{tdOpen}, \mathsf{Ext}_F)$ *for a function family $\mathcal{F}$ consists of the following polynomial-time algorithms:*

Parameter generation: $\mathcal{G}(1^\lambda)$ *returns parameters* $\mathtt{pp}$ *(for example, group description). We allow F to depend on* $\mathtt{pp}$.

Commitment key generation: *for parameters* $\mathtt{pp}$*, a positive integer $n = poly(\lambda)$, an integer $q \in [1..n]$, and a tuple $\mathcal{S} = (f_1, \ldots, f_{|\mathcal{S}|}) \subseteq \mathcal{F}$ with $|\mathcal{S}| \leq q$,* $\mathsf{KC}(\mathtt{pp}, n, q, \mathcal{S})$ *outputs a commitment key* $\mathtt{ck}$ *and a trapdoor* $\mathtt{td} = (\mathtt{ek}, \mathtt{tk})$*. Here,* $\mathtt{ck}$ *implicitly specifies* $\mathtt{pp}$*, the message space* $\mathtt{MSP}$*, the randomizer space* $\mathtt{RSP}$*, and the commitment space* $\mathtt{CSP}$*, such that* $F(\mathtt{MSP}) \subseteq \mathtt{CSP}$*,* $\mathtt{ek}$ *is the* extraction key*, and* $\mathtt{tk}$ *is the* trapdoor key*. For any other input,* KC *outputs* $(\mathtt{ck}, \mathtt{td}) = (\perp, \perp)$.

Commitment: *for* $\mathtt{pp} \in \mathcal{G}(1^\lambda)$*, a commitment key* $\mathtt{ck} \neq \perp$*, a message* $\boldsymbol{x} \in \mathtt{MSP}^n$*, and a randomizer* $r \in \mathtt{RSP}$*,* $\mathsf{com}(\mathtt{ck}; \boldsymbol{x}; r)$ *outputs a commitment* $c \in \mathtt{CSP}$.

Trapdoor opening: *for* $\mathtt{pp} \in \mathcal{G}(1^\lambda)$*, $\mathcal{S} \subseteq \mathcal{F}$ with $|\mathcal{S}| \leq q$,* $(\mathtt{ck}, (\mathtt{ek}, \mathtt{tk})) \in \mathsf{KC}(\mathtt{pp}, n, q, \mathcal{S})$*, two messages* $\boldsymbol{x}_0, \boldsymbol{x}_1 \in \mathtt{MSP}^n$*, and a randomizer* $r_0 \in \mathtt{RSP}$*,* $\mathsf{tdOpen}(\mathtt{pp}, \mathtt{tk}; \boldsymbol{x}_0, r_0, \boldsymbol{x}_1)$ *returns a randomizer* $r_1 \in \mathtt{RSP}$.

Extraction: *for* $\mathtt{pp} \in \mathcal{G}(1^\lambda)$*, $\mathcal{S} = (f_1, \ldots, f_{|\mathcal{S}|}) \subseteq \mathcal{F}$ with $1 \leq |\mathcal{S}| \leq q$,* $(\mathtt{ck}, (\mathtt{ek}, \mathtt{tk})) \in \mathsf{KC}(\mathtt{pp}, n, q, \mathcal{S})$*, and* $c \in \mathtt{CSP}$*,* $\mathsf{Ext}_F(\mathtt{pp}, \mathtt{ek}; c)$ *returns a tuple* $\big(F(f_1(x)), \ldots, F(f_{|\mathcal{S}|}(x))\big) \in \mathtt{MSP}^{|\mathcal{S}|}$*;*

For $\{f_i\}_{i=1}^q \subseteq \mathcal{F}$ and vector $\boldsymbol{x}$ let us denote $\boldsymbol{x}_\mathcal{S} = (f_1(\boldsymbol{x}), \ldots, f_q(\boldsymbol{x}))$.

Definition 37. *An F-extractable functional SSB commitment scheme* COM *for function family $\mathcal{F}$ is* secure *if it satisfies the following security requirements.*

Function-Set Hiding (FSH): $\forall\lambda$*, PPT $\mathcal{A}$, $n = poly(\lambda)$, $q \in [1..n]$,* $\mathsf{Adv}^{fsh}_{\mathcal{A},\mathsf{COM},n,q} := 2 \cdot |\varepsilon^{\mathsf{fsh}}_{\mathcal{A},\mathsf{COM},n,q}(\lambda) - 1/2| \approx_\lambda 0$*, where* $\varepsilon^{\mathsf{fsh}}_{\mathcal{A},\mathsf{COM},n,q}(\lambda) :=$

$$\Pr\left[\begin{matrix}\mathtt{pp} \leftarrow \mathcal{G}(1^\lambda); (\mathcal{S}_0, \mathcal{S}_1) \leftarrow \mathcal{A}(\mathtt{pp}, n, q) \text{ s.t. } \forall i \in \{0,1\}, \mathcal{S}_i \subseteq \mathcal{F} \wedge |\mathcal{S}_i| \leq q; \\ \beta \leftarrow \{0,1\}; (\mathtt{ck}_\beta, \mathtt{td}_\beta) \leftarrow \mathsf{KC}(\mathtt{pp}, n, q, \mathcal{S}_\beta) : \mathcal{A}(\mathtt{ck}_\beta) = \beta\end{matrix}\right] .$$

Somewhere Statistically Binding (SSB): $\forall\lambda$*, unbounded $\mathcal{A}$, $n = poly(\lambda)$, $q \in [1..n]$,* $\mathsf{Adv}^{ssb}_{\mathcal{A},\mathsf{COM},n,q} \approx_\lambda 0$*, where* $\mathsf{Adv}^{ssb}_{\mathcal{A},\mathsf{COM},n,q} :=$

$$\Pr\left[\begin{matrix}\mathtt{pp} \leftarrow \mathcal{G}(1^\lambda); \mathcal{S} \leftarrow \mathcal{A}(\mathtt{pp}, n, q) \text{ s.t. } \mathcal{S} \subseteq \mathcal{F} \wedge |\mathcal{S}| \leq q; \\ (\mathtt{ck}, \mathtt{td}) \leftarrow \mathsf{KC}(\mathtt{pp}, n, q, \mathcal{S}); (\boldsymbol{x}_0, \boldsymbol{x}_1, r_0, r_1) \leftarrow \mathcal{A}(\mathtt{ck}) \text{ s.t. } \boldsymbol{x}_{0\mathcal{S}} \neq \boldsymbol{x}_{1\mathcal{S}} : \\ \mathsf{com}(\mathtt{ck}; \boldsymbol{x}_0; r_0) = \mathsf{com}(\mathtt{ck}; \boldsymbol{x}_1; r_1)\end{matrix}\right] .$$

We say that COM *is* somewhere perfectly binding *(SPB) if* $\mathsf{Adv}^{ssb}_{\mathcal{A},\mathsf{COM},n,q} = 0$.

Almost Everywhere Statistically Hiding (AESH): $\forall\lambda$, *unbounded* $\mathcal{A}$, $n = poly(\lambda)$, $q \in [1..n]$, $\mathsf{Adv}^{\varepsilon^{\mathsf{aesh}}}_{\mathcal{A},\mathsf{COM},n,q} := 2\cdot|\varepsilon^{\mathsf{aesh}}_{\mathcal{A},\mathsf{COM},n,q}(\lambda) - 1/2| \approx_\lambda 0$, *where* $\varepsilon^{\mathsf{aesh}}_{\mathcal{A},\mathsf{COM},n,q}(\lambda) :=$

$$\Pr\left[\begin{array}{l}\mathsf{pp}\leftarrow\mathcal{G}(1^\lambda);\mathcal{S}\leftarrow\mathcal{A}(\mathsf{pp},n,q)\ \textit{s.t.}\ \mathcal{S}\subseteq\mathcal{F}\wedge|\mathcal{S}|\leq q;\\ (\mathtt{ck},\mathtt{td})\leftarrow\mathsf{KC}(\mathsf{pp},n,q,\mathcal{S});(\boldsymbol{x}_0,\boldsymbol{x}_1)\leftarrow\mathcal{A}(\mathtt{ck})\ \textit{s.t.}\ \boldsymbol{x}_{0\mathcal{S}}=\boldsymbol{x}_{1\mathcal{S}};\\ \beta\leftarrow\{0,1\};r\leftarrow\mathtt{RSP}:\mathcal{A}(\mathsf{com}(\mathtt{ck};\boldsymbol{x}_\beta;r))=\beta\end{array}\right].$$

COM *is* almost everywhere perfectly hiding *(AEPH) if* $\mathsf{Adv}^{aesh}_{\mathcal{A},\mathsf{COM},n,q} = 0$.

Somewhere Statistical F-Extractability (F-SSE): $\forall\lambda$, $\mathsf{pp} \in \mathcal{G}(1^\lambda)$, $n = poly(\lambda)$, $q \in [1..n]$, $\mathcal{S} = (f_1,\ldots,f_{|\mathcal{S}|}) \subseteq \mathcal{F}$ *with* $|\mathcal{S}| \leq q$, $(\mathtt{ck},(\mathtt{ek},\mathtt{tk}))\leftarrow\mathsf{KC}(\mathsf{pp},n,q,\mathcal{S})$, *and PPT* $\mathcal{A}$, $\mathsf{Adv}^{sse}_{\mathcal{A},F,\mathsf{COM},n,q} \approx_\lambda 0$, *where* $\mathsf{Adv}^{sse}_{\mathcal{A},F,\mathsf{COM},n,q} :=$

$$\Pr\left[\boldsymbol{x},r\leftarrow\mathcal{A}(\mathtt{ck}):\mathsf{Ext}_F(\mathsf{pp},\mathtt{ek};\mathsf{com}(\mathtt{ck};\boldsymbol{x};r))\neq\big(F(f_1(\boldsymbol{x})),\ldots,F(f_{|\mathcal{S}|}(\boldsymbol{x}))\big)\right].$$

It is somewhere perfect extractable if $\mathsf{Adv}^{sse}_{\mathcal{A},F,\mathsf{COM},n,q} = 0$.

Almost Everywhere Statistical Trapdoor (AEST): $\forall\lambda$, $n = poly(\lambda)$, $q \in [1..n]$ *and unbounded* $\mathcal{A}$, $\mathsf{Adv}^{aest}_{\mathcal{A},\mathsf{COM},n,q}(\lambda) \approx_\lambda 0$, *where* $\mathsf{Adv}^{aest}_{\mathcal{A},\mathsf{COM},n,q}(\lambda) =$

$$\Pr\left[\begin{array}{l}\mathsf{pp}\in\mathcal{G}(1^\lambda);\mathcal{S}\leftarrow\mathcal{A}(\mathsf{pp},n,q)\ \textit{s.t.}\ \mathcal{S}\subseteq\mathcal{F}\wedge|\mathcal{S}|\leq q;\\ (\mathtt{ck},\mathtt{td})\leftarrow\mathsf{KC}(\mathsf{pp},n,q,\mathcal{S});(\boldsymbol{x}_0,\boldsymbol{x}_1,r_0)\leftarrow\mathcal{A}(\mathtt{ck})\ \textit{s.t.}\ \boldsymbol{x}_{0\mathcal{S}}=\boldsymbol{x}_{1\mathcal{S}}:\\ r_1\leftarrow\mathsf{tdOpen}(\mathsf{pp},\mathtt{tk};\boldsymbol{x}_0,r_0,\boldsymbol{x}_1):\mathsf{com}(\mathtt{ck};\boldsymbol{x}_0;r_0)\neq\mathsf{com}(\mathtt{ck};\boldsymbol{x}_1;r_1)\end{array}\right].$$

It is AEPT (almost everywhere perfect trapdoor) if $\mathsf{Adv}^{aest}_{\mathcal{A},\mathsf{COM},n,q}(\lambda) = 1$.

Computational Binding (CB): $\forall$ *PPT* $\mathcal{A}$, $n = poly(\lambda)$, $q \in [1..n]$, $\mathsf{Adv}^{cb}_{\mathcal{A},\mathsf{COM},n,q} = negl(\lambda)$, *where* $\mathsf{Adv}^{cb}_{\mathcal{A},\mathsf{COM},n,q} :=$

$$\Pr\left[\begin{array}{l}\mathsf{pp}\leftarrow\mathcal{G}(1^\lambda);\mathcal{S}\leftarrow\mathcal{A}(\mathsf{pp},n,q)\ \textit{s.t.}\ \mathcal{S}\subseteq\mathcal{F}\wedge|\mathcal{S}|\leq q;\\ (\mathtt{ck},\mathtt{td})\leftarrow\mathsf{KC}(\mathsf{pp},n,q,\mathcal{S});(\boldsymbol{x}_0,\boldsymbol{x}_1,r_0,r_1)\leftarrow\mathcal{A}(\mathtt{ck})\ \textit{s.t.}\ \boldsymbol{x}_0\neq\boldsymbol{x}_1:\\ \mathsf{com}(\mathtt{ck};\boldsymbol{x}_0;r_0)=\mathsf{com}(\mathtt{ck};\boldsymbol{x}_1;r_1)\end{array}\right].$$

Computational Hiding (CH): $\forall$ *PPT* $\mathcal{A}$, $n = poly(\lambda)$, $q \in [1..n]$, $\mathsf{Adv}^{ch}_{\mathcal{A},\mathsf{COM},n,q} := 2\cdot|\varepsilon^{\mathsf{ch}}_{\mathcal{A},\mathsf{COM},n,q}(\lambda) - 1/2| = negl(\lambda)$, *where* $\varepsilon^{\mathsf{ch}}_{\mathcal{A},\mathsf{COM},n,q}(\lambda) :=$

$$\Pr\left[\begin{array}{l}\mathsf{pp}\leftarrow\mathcal{G}(1^\lambda);\mathcal{S}\leftarrow\mathcal{A}(\mathsf{pp},n,q)\ \textit{s.t.}\ \mathcal{S}\subseteq\mathcal{F}\wedge|\mathcal{S}|\leq q;\\ (\mathtt{ck},\mathtt{td})\leftarrow\mathsf{KC}(\mathsf{pp},n,q,\mathcal{S});(\boldsymbol{x}_0,\boldsymbol{x}_1)\leftarrow\mathcal{A}(\mathtt{ck});\beta\leftarrow\{0,1\};\\ r\leftarrow\mathtt{RSP}:\mathcal{A}(\mathsf{com}(\mathtt{ck};\boldsymbol{x}_\beta;r))=\beta\end{array}\right].$$

$\underline{\mathsf{KC}(p, n, q, [\mathbf{M}]_\iota \in \mathbb{G}_\iota^{q\times n})}$
Set implicitly $\mathtt{MSP} = \mathtt{RSP} = \mathbb{Z}_p^n$ and $\mathtt{CSP} = \mathbb{G}_\iota^{q+1}$;
Sample $\mathbf{R} \leftarrow \mathbb{Z}_p^{(q+1)\times(q+1)}$ so that it has full rank; Sample $\boldsymbol{\varrho} \leftarrow \mathbb{Z}_p^n$;
Set $\mathbf{M}' \leftarrow \left[\begin{smallmatrix} \mathbf{M} & \mathbf{0} \\ \boldsymbol{\varrho}^\intercal & 1 \end{smallmatrix}\right]_\iota \in \mathbb{G}_\iota^{(q+1)\times(n+1)}$;
Set $\mathtt{ck} \leftarrow [\mathbf{R}\mathbf{M}']_\iota \in \mathbb{G}_\iota^{(q+1)\times(n+1)}$, $\mathtt{td} \leftarrow (\mathtt{ek} \leftarrow \mathbf{R}^{-1}, \mathtt{tk} \leftarrow \boldsymbol{\varrho})$;
return $(\mathtt{ck}, \mathtt{td})$;

$\underline{\mathsf{com}(\mathtt{ck}; \boldsymbol{x} \in \mathbb{Z}_p^n; r \in \mathbb{Z}_p)}$
return $\mathtt{ck}\left(\begin{smallmatrix} \boldsymbol{x} \\ r \end{smallmatrix}\right)$;

$\underline{\mathsf{tdOpen}(\mathsf{pp}, \mathtt{tk}; \boldsymbol{x}_0, r_0, \boldsymbol{x}_1) \mathbin{/\!\!/} [\mathbf{M}]_\iota \boldsymbol{x}_0 = [\mathbf{M}]_\iota \boldsymbol{x}_1}$
return $r_1 \leftarrow \sum_{i\in[1..n]}(x_{0,i} - x_{1,i})\mathtt{tk}_i + r_0$;

$\underline{\mathsf{Ext}(\mathsf{pp}, \mathtt{ek}; [\mathbf{c}])}$
return $\mathtt{ek}[\boldsymbol{c}]_\iota$ without the last component;

Figure 6.3: Functional SSB commitment for linear functions

Linear EMP

We construct a functional SSB commitment for a family of linear functions. Our construction follows the ideas of Chapter 3 which only dealt with some concrete functions and never formalized the ideas.

We represent q linear functions by a matrix $\mathbf{M} \in \mathbb{Z}_p^{q\times n}$ where each row contains coefficients of one function. From a commitment to vector $\boldsymbol{x} \in \mathbb{Z}_p^n$, our construction allows to extract $[\mathbf{M}\boldsymbol{x}]_\iota$. In particular, if we take $\mathbf{M} = (\boldsymbol{e}_{i_1}|\dots|\boldsymbol{e}_{i_q})^\top$ where $\boldsymbol{e}_{i_j} \in \mathbb{Z}_p^n$ is the i_jth unit vector, then $[\mathbf{M}\boldsymbol{x}]_\iota = [x_{i_1}, \dots, x_{i_q}]_\iota^\top$. A detailed construction is given in Figure 6.3.

We want to note that the matrix $[\mathbf{M}]_\iota$ is extended into one row to place the randomness vector $\boldsymbol{\varrho}$ and one column to place the randomizator of the commitment, r, to perfectly hide the secret vector $\boldsymbol{x}$ when we extract. Concretely, in the extraction phase we obtain $\left[\begin{smallmatrix} \mathbf{M} & \mathbf{0} \\ \boldsymbol{\varrho}^\intercal & 1 \end{smallmatrix}\right]_\iota \left[\begin{smallmatrix} \boldsymbol{x} \\ r \end{smallmatrix}\right]_\iota = \left[\begin{smallmatrix} \mathbf{M}\boldsymbol{x} \\ \boldsymbol{\varrho}^\intercal\boldsymbol{x}+r \end{smallmatrix}\right]_\iota$ from multiplying the commitment by the inverse matrix of $\mathbf{R}$. The first q rows contain the functions of $\boldsymbol{x}$ in the group that we want and the last component contains a combination of $\boldsymbol{x}$ with $\boldsymbol{\varrho}$ that is completely masked by r.

Moreover, if we take an ACS (Def. 34), the commitment key is $\mathtt{ck} = [\mathbf{U}_1, \mathbf{U}_2]_\iota \in \mathbb{G}_\iota^{(q+1)\times n} \times \mathbb{G}_\iota^{(q+1)\times 1}$, which is optimal size for extraction in q coordinates, as proven in Corollary 34. The main differences with the EMP construction in Section 6.4.2 is that in EMP $\mathbf{M}$ is a matrix in reduced row echelon form (with multiples of the column vector $(0, \dots, 0, 1)^T$ possibly inserted in between). We prove security of linear EMP in the following.

Security proofs. Before proving the security of linear EMP, let us recall some well-known decisional assumptions.

Decisional Diffie-Hellman (DDH) Assumption. Let $\iota \in \{1,2\}$. *DDH*$_{\mathbb{G}_\iota}$ holds relative to $\mathcal{G}$, if $\forall$ PPT $\mathcal{A}$, $\mathsf{Adv}^{ddh}_{\mathcal{A},\iota,\mathcal{G}} := |\varepsilon^0_{\mathsf{Adv}}(\lambda) - \varepsilon^1_{\mathsf{Adv}}(\lambda)| = negl(\lambda)$, where

$$\varepsilon^{\beta}_{\mathsf{Adv}}(\lambda) := \Pr\left[\mathsf{pp} \leftarrow \mathcal{G}(1^\lambda); x,y,z \leftarrow \mathbb{Z}_p : \mathcal{A}(\mathsf{pp}, [x,y,xy+\beta z]_\iota) = 1 \quad \right] .$$

Rank Assumption. Let $\iota \in \{1,2\}$. (ℓ,k,r_0,r_1)-*Rank assumption* for $1 \leq r_0 < r_1 \leq \min(\ell,k)$ holds relative to $\mathcal{G}$, if $\forall$ PPT $\mathcal{A}$, $\mathsf{Adv}^{rank}_{\mathcal{A},\ell,k,r_0,r_1,\iota,\mathcal{G}} := |\varepsilon^0_{\mathsf{Adv}}(\lambda) - \varepsilon^1_{\mathsf{Adv}}(\lambda)| = negl(\lambda)$, if

$$\varepsilon^{\beta}_{\mathsf{Adv}}(\lambda) := \Pr\left[\mathsf{pp} \leftarrow \mathcal{G}(1^\lambda); \mathbf{A} \leftarrow \mathcal{U}^{(r_\beta)}_{\ell,k} : \mathcal{A}(\mathsf{pp}, [\mathbf{A}]_\iota) = 1 \quad \right] ,$$

where $\mathcal{U}^{(r_\beta)}_{\ell,k}$ is the uniform distribution over rank r_β matrices $\mathbb{Z}_p^{\ell\times k}$.

Theorem 37 ([119])**.** *Let $\iota \in \{1,2\}$. For any $\ell,k,r_0,r_1 \in \mathbb{Z}$ such that $1 \leq r_0 < r_1 \leq \min(\ell,k)$, any PPT $\mathcal{A}$, and any $\mathcal{G}$,*

$$\mathsf{Adv}^{rank}_{\mathcal{A},\ell,k,r_0,r_1,\iota,\mathcal{G}} \leq \lceil \log_2(r_1/r_0) \rceil \cdot \mathsf{Adv}^{ddh}_{\mathcal{A},\iota,\mathcal{G}} .$$

Theorem 38. *Let $\mathcal{G}_{bg}$ be a bilinear group generator. Fix n and q. The commitment scheme in Figure 6.3 is*

(i) *FSH relative to $\mathcal{G}_{bg}$ under the DDH$_{\mathbb{G}_\iota}$ assumption: for each PPT $\mathcal{A}$, there exists a PPT $\mathcal{B}$, such that $\mathsf{Adv}^{fsh}_{\mathcal{A},\mathsf{COM},n,q} \leq \lceil \log_2(q+1) \rceil \cdot \mathsf{Adv}^{ddh}_{\mathcal{B},\iota,\mathcal{G}}$.*

(ii) *F-SSE for $F = [\cdot]_\iota$ (thus, F depends on pp),*

(iii) *SPB,*

(iv) *AEPH,*

(v) *AEPT,*

(vi) *CB and CH.*

Proof. **(i: FSH)** Since given a matrix $\boldsymbol{M'}$ of rank $k \in [1..q+1]$, the matrix $\mathbf{R}\boldsymbol{M'}$ is a random matrix of rank k with an overwhelming probability. Then, distinguishing commitment keys $\mathtt{ck}_1 = [\mathbf{R}_1\boldsymbol{M'_1}]_\iota$ and $\mathtt{ck}_2 = [\mathbf{R}_2\boldsymbol{M'_2}]_\iota$ is equivalent to breaking the rank assumption. Now, considering Theorem 37 we get that for each adversary $\mathcal{A}$ against FSH, there exists an adversary $\mathcal{B}$ against the DDH in $\mathbb{G}_\iota$ such that the bound

$\mathsf{Adv}^{fsh}_{\mathcal{A},\mathsf{COM},n,q} \simeq \mathsf{Adv}^{rank}_{\mathcal{B},\iota,\mathcal{G}} \leq \lceil \log_2(r_1/r_0) \rceil \cdot \mathsf{Adv}^{ddh}_{\mathcal{B},\iota,\mathcal{G}}$ holds. In the worst case one matrix has rank $r_0 = 1$ and the other has rank $r_1 = q+1$, so the worst bound is $\lceil \log_2(q+1) \rceil \cdot \mathsf{Adv}^{ddh}_{\mathcal{B},\iota,\mathcal{G}}$.

(ii: F-SSE) For any $\boldsymbol{x} \in \mathbb{Z}_p^n$ and $\boldsymbol{r} \in \mathbb{Z}_p^{q+1}$, we have $\mathsf{com}(\mathtt{ck}; \boldsymbol{x}; r) = [\mathbf{R}\boldsymbol{M'}(\begin{smallmatrix}\boldsymbol{x}\\ r\end{smallmatrix})]_\iota = [\boldsymbol{c}]_\iota$. Then, $\mathsf{Ext}(\mathtt{pp}, \mathtt{ek} = \mathbf{R}^{-1}; [\boldsymbol{c}]_\iota)$ computes $\mathbf{R}^{-1}[\boldsymbol{c}]_\iota = [\boldsymbol{M'}(\begin{smallmatrix}\boldsymbol{x}\\ r\end{smallmatrix})]_\iota = \left[\begin{smallmatrix}\mathbf{M}\boldsymbol{x}\\ \boldsymbol{\varrho}^\top \boldsymbol{x} + r\end{smallmatrix}\right]_\iota$ and outputs $[\mathbf{M}\boldsymbol{x}]_\iota$ which is exatly what we wanted to extract.

(iii: SPB) Clearly, there are no $\boldsymbol{x}_0, \boldsymbol{x}_1 \in \mathbb{Z}_p^n$ such that $\mathbf{M}\boldsymbol{x}_0 \neq \mathbf{M}\boldsymbol{x}_1$ and $[\boldsymbol{c}]_\iota := \mathsf{com}(\mathtt{ck}; \boldsymbol{x}_0; r_0) = \mathsf{com}(\mathtt{ck}; \boldsymbol{x}_1; r_1)$ since by the F-SSE property we have that $\mathsf{Ext}(\mathtt{pp}, \mathtt{ek} = \mathbf{R}^{-1}; [\boldsymbol{c}]_\iota) = [\mathbf{M}\boldsymbol{x}_0]_\iota = [\mathbf{M}\boldsymbol{x}_1]_\iota$.

(iv: AEPH) Suppose that the adversary $\mathcal{A}$ on input $(\mathtt{pp}, n, q)$ outputs $\mathcal{S} = \mathbf{M} \in \mathbb{Z}_p^{q \times n}$, then gets as an input the public key $\boldsymbol{g} = \mathbf{R} \cdot \mathbf{M}'$ where $\mathbf{M}' = \left(\begin{smallmatrix}\mathbf{M} & \mathbf{0}\\ \boldsymbol{\varrho}^\top & 1\end{smallmatrix}\right)$, $\mathbf{R} \in \mathbb{Z}_p^{(q+1)(q+1)}$ is some full rank matrix, and $\boldsymbol{\varrho} \in \mathbb{Z}_p^n$, and finally outputs $(\boldsymbol{x}_0, \boldsymbol{x}_1)$ such that $\mathbf{M}\boldsymbol{x}_0 = \mathbf{M}\boldsymbol{x}_1$.

Let us analyse distributions of $C_0 = \mathsf{com}(\mathtt{ck}; \boldsymbol{x}_0; _0)$ and $C_1 = \mathsf{com}(\mathtt{ck}; \boldsymbol{x}_1; r_1)$ for a uniformly random r_0, r_1. For $\beta \in \{0,1\}$, we can define $[\boldsymbol{u}_\beta] := [\mathbf{M}'(\begin{smallmatrix}\boldsymbol{x}_\beta\\ r_\beta\end{smallmatrix})] = \left[\begin{smallmatrix}\mathbf{M}\boldsymbol{x}_\beta\\ \boldsymbol{\varrho}^\top \boldsymbol{x}_\beta + r_\beta\end{smallmatrix}\right]$. We see that top q elements of $\boldsymbol{u}_0$ and $\boldsymbol{u}_1$ are equal and the last element is uniformly random. Thus, $\boldsymbol{u}_0$ and $\boldsymbol{u}_1$ are indistinguishable. Since $C_\beta = \mathsf{com}(\mathtt{ck}; \boldsymbol{x}_\beta; r_\beta) = \mathbf{R}[\boldsymbol{u}_\beta]$, then also C_1 and C_2 are indistinguishable.

(v: AEPT) Let $r_0 \in \mathbb{Z}_p$ and $\boldsymbol{x}_0, \boldsymbol{x}_1 \in \mathbb{Z}_p^n$ such that $\mathbf{M}\boldsymbol{x}_0 = \mathbf{M}\boldsymbol{x}_1$. In tdOpen, we define $r_1 = \sum_{i \in [1..n]} (x_{0,i} - x_{1,i})\varrho_i + r_0$. Then, $\boldsymbol{\varrho}^\top \boldsymbol{x}_1 + r_1 = \boldsymbol{\varrho}^\top \boldsymbol{x}_0 + r_0$. Using, the definition of $\boldsymbol{u}_b$ from the previous property, we see that $\boldsymbol{u}_0 = \boldsymbol{u}_1$ and then also $\mathsf{com}(\mathtt{ck}; \boldsymbol{x}_0; r_0) = \mathsf{com}(\mathtt{ck}; \boldsymbol{x}_1; r_1)$.

(vi: CB and CH) Follows directly from the analog of Theorem 31. □

6.6 Applications of Functional SSB Commitments

We present three applications of functional SSB commitments. In Section 6.6.1 we have two straightforward applications for linear EMP commitments: Oblivious Database Queries (ODQ) and Oblivious Linear Function Evaluation (OLE) [49, 65, 48]. OLE allows the receiver to learn $f(\boldsymbol{x})$ where $\boldsymbol{x}$ is the receiver's private vector and f is the sender's private linear function. ODQ essentially switches the roles of receiver and sender: the receiver wants to learn $f(\boldsymbol{x})$ where $\boldsymbol{x}$ is the sender's private database and f is the receiver's linear query function. In Section 6.6.2 we present a new QA-NIZK argument for SAP relations that uses linear EMP commitments as a technical tool in the security proof.

6.6.1 ODQ & OLE

A very straight-forward application of linear EMP is oblivious database queries (ODQ). We consider a scenario where the sender knows a private database $\boldsymbol{x}$ and the receiver knows a set of private linear functions $f_i(X_1, \ldots, X_n) = b_i + \sum_{j=1}^{n} a_{i,j} X_j$ for $i \in [1..q]$ that he wants to evaluate on that database.

Our ODQ protocol works as follows:

- Receiver defines matrices $\mathbf{A} = (a_{ij}) \in \mathbb{Z}_p^{q \times n}$, $\mathbf{B} = \text{diag}(b_1, \ldots, b_q) \in \mathbb{Z}_p^{q \times q}$, and $\mathbf{M} = (\mathbf{A} \mid \mathbf{B}) \in \mathbb{Z}_p^{q \times (n+q)}$. Following the KC algorithm it creates the commitment key $\mathtt{ck}$, the extraction key $\mathtt{ek}$, and sends $\mathtt{ck}$ to the sender.

- Sender has $\boldsymbol{x} \in \mathbb{Z}_p^n$ and $\mathtt{ck}$ as input. It sets $\boldsymbol{x}' = \left(\begin{smallmatrix} \boldsymbol{x} \\ \mathbf{1}_q \end{smallmatrix}\right)$, picks random $r \leftarrow \mathbb{Z}_p$ and sends $\mathsf{COM} = \mathtt{ck}\left(\begin{smallmatrix} \boldsymbol{x}' \\ r \end{smallmatrix}\right)$ to the receiver.

- Receiver extracts $[\mathbf{M} \cdot \boldsymbol{x}']$ from COM using the Ext algorithm with $\mathtt{ek}$.

Privacy and Correctness. We follow privacy and correctness definitions proposed by Döttling et al. [47] (see Section 5.1 of their paper for full definitions). From the SSE property we know that the receiver can recover $[\mathbf{M}\left(\begin{smallmatrix} \boldsymbol{x} \\ \mathbf{1}_q \end{smallmatrix}\right)]_\iota = [\mathbf{A}\boldsymbol{x} + \boldsymbol{b}]_\iota$ and thus correctness holds. Receiver's (computational) privacy follows directly from the FSH property, that is, any two function-sets of size at most q are indistinguishable. Sender's privacy is defined through simulatability of the protocol transcript given only receiver's input $\mathbf{M}$ and receiver's output $[\mathbf{M}\boldsymbol{x}']$ to the simulator. Simulatability is slightly stronger than the AEPH property but still holds for linear EMP. As a first message, the simulator can generate $\mathtt{ck}$ with $\mathbf{M}$ and store $\mathbf{R}$. An honestly computed second message has the form $[\mathbf{R}\left(\begin{smallmatrix} \mathbf{M} & \mathbf{0} \\ \boldsymbol{r}^\intercal & 1 \end{smallmatrix}\right)]\left(\begin{smallmatrix} \boldsymbol{x}' \\ r \end{smallmatrix}\right) = \mathbf{R}\left[\begin{smallmatrix} \mathbf{M}\boldsymbol{x}' \\ \boldsymbol{x}'\boldsymbol{r}^\top + r \end{smallmatrix}\right]$ and therefore we can simulate it by sampling $r^* \leftarrow \mathbb{Z}_p$ and computing $\mathbf{R}\left(\begin{smallmatrix} [\mathbf{M}\boldsymbol{x}'] \\ r^* \end{smallmatrix}\right)$. Thus sender's privacy also holds.

Efficiency. We define download rate as the ratio between output size and sender's message and total rate as the ratio between output size and total transcript size. The total rate of our protocol is $|[\mathbf{M}\boldsymbol{x}']|/(|\mathtt{ck}|+|\mathsf{COM}|) = q/((n+q+2)(q+1))$. However, we achieve very good download rate $|[\mathbf{M}\boldsymbol{x}']|/|\mathsf{COM}| = q/(q+1)$ which tends to 1. This is similar to Döttling et al. [47] where they achieve an optimal download rate but sub-optimal total rate.

OLE

We can achieve OLE in a very similar way. Suppose that now the sender has a function $f(X_1, \ldots, X_n) = b + \sum_{i=1}^{n} a_i X_i$ and the receiver has $\boldsymbol{x}$. Then the receiver can send a commitment key with $\mathbf{M} = (x_1, \ldots, x_n, 1)$ and the sender responds with a commitment to $(a_1, \ldots, a_n, b)$. The receiver extracts to obtain $[f(\boldsymbol{x})]_\iota$. The proof is identical to the ODQ case. However, the resulting OLE is less efficient with download rate $1/2$ and total rate $1/(2n+4)$.

6.6.2 QA-NIZK Argument for Quadratic Equations

We present a QA-NIZK argument which uses linear EMP commitments as an important technical tool in the security proof, inspired by our work in Chapter 3 where we presented a commit-and-prove QA-NIZK argument for Square Span Programs (SSP, [45]) which can be used to encode the Boolean circuit satisfiability language. Their construction uses a specific setting of linear EMP commitments without explicitly formalizing it. Our QA-NIZK is for Square Arithmetic Programs (SAP) [74] which can be used to encode the arithmetic circuit satisfiability language, has roughly the same complexity as the argument in Chapter 3 and follows a similar overall strategy. However, we use linear EMP commitments as a black-box and thus have a more compact and clear presentation.

A rough intuition of our commit-and-prove QA-NIZK is as follows. The statement of our language $\mathcal{L}_{\mathtt{SAP},\mathtt{ck}}$ contains a linear-length perfectly binding (and $[\cdot]_1$-extractable) commitment $[\boldsymbol{c}]_1$ of the SAP witness. Note that the commitment is only computed once but can be reused for many different SAP relations. For simplicity, we use ElGamal encryption in this role and the commitment key $\mathtt{ck}$ as a parameter of the language. The argument itself is succinct and contains the following elements:

- a succinct zk-SNARK-type argument $[V, H, W]_1$, $[V]_2$ for the SAP relation,
- a succinct linear EMP commitment $[\tilde{\boldsymbol{c}}]_2$ that commits to the SAP witness and to the randomness of the zk-SNARK,
- a succinct linear subspace argument BLS [67] that shows that commitments open to consistent values (see BLS argument in Chapter 2). I.e., it guarantees that the opening of $[\boldsymbol{c}]_1$ is also used in the zk-SNARK and in $[\tilde{\boldsymbol{c}}]_2$.

Preliminaries

Square Arithmetic Program (SAP). A square arithmetic program is a tuple $\mathtt{SAP} = (\mathsf{pp}, n, l, \mathbf{V} \in \mathbb{Z}_p^{n \times l}, \mathbf{W} \in \mathbb{Z}_p^{n \times l})$. We define a commit-and-prove language for $\mathtt{SAP}$ as

the following language with n variables and l quadratic equations

$$\mathcal{L}_{\mathtt{SAP},\mathtt{ck}} = \left\{ [\boldsymbol{c}]_1 \in \mathbb{G}_1^{2n} \;\middle|\; \begin{array}{c} \exists \boldsymbol{a}, \boldsymbol{r} \in \mathbb{Z}_p^n \colon [\boldsymbol{c}]_1 = \mathsf{com}_{ck}(\boldsymbol{a}, \boldsymbol{r}) \wedge \\ \left\{ \left(\boldsymbol{a}^\top \boldsymbol{v}_j\right)^2 - \boldsymbol{a}^\top \boldsymbol{w}_j = 0 \right\}_{j=1}^{l} \end{array} \right\}$$

where com_{ck} is a perfectly binding commitment scheme, $\boldsymbol{v}_j$ is j-th column of the matrix $\mathbf{V}$ and $\boldsymbol{w}_j$ is the j-th column of the matrix $\mathbf{W}$.

SNARK for SAP. Let $\chi_1, \ldots, \chi_l \in \mathbb{Z}_p$ be unique interpolation points. We define

$$v(X) = \sum_{i=1}^{n} a_i v_i(X), \quad w(X) = \sum_{i=1}^{n} a_i w_i(X) \tag{6.1}$$

where $v_i(X)$, $w_i(X)$ are polynomials of degree less than l such that $v_i(\chi_j) = v_{ij}$ and $w_i(\chi_j) = -w_{ij}$. Moreover, let us define $p(X) = v(X)^2 - w(X)$ and $t(X) = \prod_{j=1}^{l}(X - \chi_j)$. We have that $p(\chi_j) = (\boldsymbol{a}^\top \boldsymbol{v}_j)^2 - \boldsymbol{a}^\top \boldsymbol{w}_j$ and thus the j-th SAP equation is satified exactly when χ_j is a root of $p(X)$. In particular, when all interpolation points are roots of $p(X)$, then $t(X)$ divides $p(X)$ and all the SAP equations are satisfied.

Similarly as we explained in other chapters, we can use these polynomial representations to construct a zk-SNARK. Our crs will contain $\{[s^i]_{1,2}\}_{i=1}^{l}$ where $s \leftarrow \mathbb{Z}_p$ is a secret point. The prover will compute $[V]_{1,2} = [V(s)]_{1,2}$, $[W]_1 = [W(s)]_1$ and $[H]_1 = [H(s)]_1$ where $V(X) = v(X) + \delta_v t(X)$, $W(X) = w(X) + \delta_w t(X)$, and $H(X) = (V(X)^2 - W(X))/t(X)$. Elements δ_v and δ_w are picked randomly to hide the witness. The verifier checks that the equation $[V]_1[V]_2 - [W]_1[1]_2 = [H]_1[t(s)]_2$ is satisfied. Intuitively, we can use this to show that $t(X)$ divides $P(X) := V(X)^2 - W(X)$. It is easy to see that if $t(X) \mid P(X)$ then also $t(X) \mid p(X)$ and thus the SAP relation is satisfied.

New target assumption. The q-target strong Diffie-Hellman assumption [21] says that given $\{[s^i]_{1,2}\}_{i=1}^{q}$ for a random s, it is computationally hard to find $[\nu]_T = [1/(s-r)]_T$ for any $r \in \mathbb{Z}_p$. We generalize this assumption and intuitively say that it is hard to compute $[\nu]_T = [c/(s-r)]_T$ where $r \in \mathbb{Z}_p$ and c is a constant independent of s. In order to satisfy the latter requirement, we include a challenge value $[z]_2$ and let the adversary additionally output $[c]_1$ and $[c']_2$ such that $zc = c'$. Intuitively, then c cannot depend on s^i since otherwise c' should depend on zs^i which is not a part of the challenge. For technical reasons, c in our assumption has a slightly more structured form $\beta_1^2 - \beta_2$.

Definition 38 (q-SATSDH)**.** *The q-Square Arithmetic Target Strong Diffie-Hellman assumption holds relative to $\mathcal{G}$, if $\forall$ PPT adversaries $\mathcal{A}$,*

$$\Pr\left[\begin{array}{l}\mathsf{pp}\leftarrow\mathcal{G}(1^\lambda);s,z\leftarrow\mathbb{Z}_p;\\ \Big(r,[\beta_1,\beta_2]_1,[\tilde{\beta}_1,\tilde{\beta}_2]_2,[\nu]_T\Big)\leftarrow\mathcal{A}\big(\mathsf{pp},\{[s^i]_{1,2}\}_{i=1}^q,[z]_2\big):\\ \tilde{\beta}_1=z\beta_1\wedge\tilde{\beta}_2=z\beta_2\wedge\beta_1^2\neq\beta_2\wedge\nu=\frac{\beta_1^2-\beta_2}{s-r}\end{array}\right]\approx_\lambda 0.$$

We prove in the following that our new assumption is falsifiable and equivalent to TSDH assumption under a knowledge assumption.

Let us first see that q-SATSDH is falsifiable. Observe that the challenger knows $z,s\in\mathbb{Z}_p$. Thus, upon receiving $(r,[\beta_1,\beta_2]_1,[\tilde{\beta}_1,\tilde{\beta}_2]_2,[\nu]_T)$ it verifies that: (a) $[1]_1[\tilde{\beta}_1]_2=[\beta_1]_1[z]_2$, (b) $[1]_1[\tilde{\beta}_2]_2=[\beta_2]_1[z]_2$, (c) $\frac{1}{z}[\beta_1]_1[\tilde{\beta}_1]_2\neq[\beta_2]_1[1]_2$, and (d) $(s-r)[\nu]_T=\frac{1}{z}[\beta_1]_1[\tilde{\beta}_1]_2-[\beta_2]_1[1]_2$.

Lemma 39. *Given a bilinear group $gk=(q,\mathbb{G}_1,\mathbb{G}_2,\mathbb{G}_T)$, if the q-SATSDH assumption holds then the q-TSDH assumption holds.*

Proof. Assume that $\mathcal{A}$ is an adversary against the q-TSDH assumption, we construct another adversary $\mathcal{B}$ against q-SATSDH assumption that receives a challenge tuple $(gk,\{[s^i]_{1,2}\}_{i=1}^q,[z]_2)$ and sends the elements $(gk,\{[s^i]_{1,2}\}_{i=1}^q)$ to $\mathcal{A}$. $\mathcal{A}$ then returns $(r,[\nu]_T)$ that breaks q-TSDH. The adversary $\mathcal{B}$ chooses $\beta_1,\beta_2\leftarrow\mathbb{Z}_p$ such that $\beta_1^2\neq\beta_2$ and returns $\big(r,[\beta_1,\beta_2]_1,\beta_1[z]_2,\beta_2[z]_2,(\beta_1^2-\beta_2)[\nu]_T\big)$ which breaks the q-SATSDH assumption. □

Lemma 40. *Given a bilinear group $gk=(q,\mathbb{G}_1,\mathbb{G}_2,\mathbb{G}_T)$ where* BDH-KE *assumption holds, if the q-TSDH assumption holds then the q-SATSDH assumption holds.*

Proof. Assume that $\mathcal{A}$ is an adversary against the q-SATSDH assumption, we construct an another adversary $\mathcal{B}$ against the q-TSDH assumption that receives a challenge tuple $(gk,\{[s^i]_{1,2}\}_{i=1}^q)$. $\mathcal{B}$ chooses $z\leftarrow\mathbb{Z}_p$ and sends the elements $(gk,\{[s^i]_{1,2}\}_{i=1}^q,[z]_2)$ to $\mathcal{A}$. The adversary $\mathcal{A}$ then returns $(r,[\beta_1,\beta_2]_1,[\beta_3,\beta_4]_2,[\nu]_T)$ that breaks q-SATSDH. Now $\mathcal{B}$ computes $[\hat{\beta}_1]_2=\frac{1}{z}[\beta_3]_2$ and $[\hat{\beta}_2]_2=\frac{1}{z}[\beta_4]_2$ which satisfy $e([\beta_i]_1,[1]_2)=e([1]_1,[\hat{\beta}_i]_2)$ for $i=1,2$. By the BDH-KE assumption there exists and extractor of β_1,β_2 that solves the q-TSDH assumption with $\Big(r,\frac{1}{\beta_1^2-\beta_2}[\nu]_T\Big)$. □

QA-NIZK Argument scheme

Given $n,l\in\mathbb{N}$ we construct a QA-NIZK argument for $\mathcal{L}_{\mathtt{SAP},\mathtt{ck}}$.
$\mathsf{K}_0(\lambda)$**:** The algorithm K_0 returns $\mathsf{pp}\leftarrow\mathcal{G}(1^\lambda)$.

$\mathcal{D}_{\mathsf{pp}}(n,l)$**:** The algorithm $\mathcal{D}_{\mathsf{pp}}(n,l)$ returns a commitment key $\mathtt{ck} = [\boldsymbol{u}]_1 = [1,u]_1^\top$ where $u \leftarrow \mathbb{Z}_p$.

$\mathsf{K}_1(\mathsf{pp},n,l,\mathtt{ck})$**:** The algorithm K_1 picks $s \leftarrow \mathbb{Z}_p$, then sets $q_v = 4$, $n' = n+1$, $\mathbf{M} = \mathbf{0} \in \mathbb{Z}_p^{q_v \times n'}$ (i.e., $S_v = \emptyset$) and generates a linear EMP key $\mathtt{ck}' = [\mathbf{K}]_2 \leftarrow \mathsf{KC}_2(\mathsf{pp},n',q_v,\mathbf{M}) \in \mathbb{G}_2^{5\times(n+2)}$. Finally, it runs $(\mathsf{BLS.CRS},\mathsf{td}_{\mathsf{BLS}}) \leftarrow \mathsf{K}_{\mathsf{BLS}}([\mathbf{N}_1]_1 \in \mathbb{G}_1^{(2n+2)\times(2n+3)}, [\mathbf{N}_2]_2 \in \mathbb{G}_2^{5\times(2n+3)})$ for

$$[\mathbf{N}_1]_1 = \left[\begin{array}{ccc|ccc|ccc} \boldsymbol{e}_2 & & & \boldsymbol{u} & & & & & \\ & \ddots & & & \ddots & & & \mathbf{0} & \\ & & \boldsymbol{e}_2 & & & \boldsymbol{u} & & & \\ \hline v_1(s) & \dots & v_n(s) & & \mathbf{0} & & t(s) & 0 & 0 \\ w_1(s) & \dots & w_n(s) & & & & 0 & t(s) & 0 \end{array}\right]_1,$$

$$[\mathbf{N}_2]_2 = \left[\begin{array}{ccc|ccc|ccc} v_1(s) & \dots & v_n(s) & & & & t(s) & 0 & 0 \\ \mathbf{K}^{(1)} & \dots & \mathbf{K}^{(n)} & & \mathbf{0} & & \mathbf{K}^{(n+1)} & 0 & \mathbf{K}^{(n+2)} \end{array}\right]_2.$$

Return the $\mathsf{crs} = (\mathsf{pp},\mathtt{ck},\mathtt{ck}',\{[s^i]_{1,2}\}_{i=1}^l,\mathsf{BLS.CRS})$ with trapdoor $(s,\mathsf{td}_{\mathsf{BLS}})$.

P**:** The prover receives an input $(\mathsf{crs},([\boldsymbol{c}]_1,\mathbf{V},\mathbf{W}),(\boldsymbol{a},\boldsymbol{r}))$. Let $v_i(X)$ and $w_i(X)$ be the interpolation polynomials at some points $\{\chi_j\}_j$ for the i-th column of $\mathbf{V}$ and $\mathbf{W}$ respectively for $i \in [1..n]$, and set $t(X) = \prod_{i=j}^{l}(X-\chi_j)$. The prover picks $\delta_v,\delta_w,r_v \leftarrow \mathbb{Z}_p$ and defines:

$$\begin{array}{lllllll} V(X) & := & \sum_{i=1}^n a_i v_i(X) + \delta_v t(X), & W(X) & := & \sum_{i=1}^n a_i w_i(X) + \delta_w t(X) \\ P(X) & := & V(X)^2 - W(X) & H(X) & := & P(X)/t(X) \end{array} \tag{6.2}$$

The prover computes group elements $[V]_{1,2} = [V(s)]_{1,2}$, $[W]_1 = [W(s)]_1$, $[H]_1 = [H(s)]_1$ and a linear EMP commitment $[\tilde{\boldsymbol{c}}]_2 = \mathsf{com}(\mathtt{ck}';(\boldsymbol{a},\delta_v),r_v)$. The prover also computes a BLS argument ψ for the statement $\mathsf{Input}_{\mathsf{BLS}} := ([\boldsymbol{c}]_1,[V]_1,[W]_1,[V]_2,[\tilde{\boldsymbol{c}}]_2)^\top \in \mathbf{Im}\begin{pmatrix}[\mathbf{N}_1]_1 \\ [\mathbf{N}_2]_2\end{pmatrix}$ with witness $(\boldsymbol{a},\boldsymbol{r},\delta_v,\delta_w,r_v)^\top \in \mathbb{Z}_p^{2n+3}$. Finally, it outputs the argument $\pi := \left([H]_1,[V]_{1,2},[W]_1,[\tilde{\boldsymbol{c}}]_2,\psi\right)$.

V**:** The verifier with input $(\mathsf{crs},[\boldsymbol{c}]_1,\mathbf{V},\mathbf{W},\pi)$ returns 1 iff $[V]_1[V]_2 - [W]_1[1]_2 = [H]_1[t(s)]_2$ and $\mathsf{V}_{\mathsf{BLS}}(\mathsf{BLS.CRS},\mathsf{Input}_{\mathsf{BLS}},\psi) = 1$.

Security intuition

In the security proof, the soundness game is first changed by randomly picking one of the SAP equations $\left(\boldsymbol{a}^\top\boldsymbol{v}_{j^*}\right)^2 - \boldsymbol{a}^\top\boldsymbol{w}_{j^*} = 0$ for some $j^* \in [1..l]$; with probability

$\geq 1/l$ this equation does not hold, assuming that the adversary is successful. By the characterization of the SAP, if the j^*-th equation does not hold, then $X - \chi_{j^*} \nmid P(X)$. In particular, let $q_v(X), q_w(X)$ be unique polynomials and $\beta_v, \beta_w \in \mathbb{Z}_p$ be unique values such that $V(X) = q_v(X)(X - \chi_{j^*}) + \beta_v$ and $W(X) = q_w(X)(X - \chi_{j^*}) + \beta_w$. Then we can express the division of $P(X) = V(X)^2 - W(X)$ by $X - \chi_{j^*}$ as follows,

$$\begin{aligned} P(X) &= V(X)(q_v(X)(X - \chi_{j^*}) + \beta_v) - q_w(X)(X - \chi_{j^*}) - \beta_w \\ &= (X - \chi_{j^*})\left(V(X)q_v(X) - q_w(X)\right) + V(X)\beta_v - \beta_w \\ &= (X - \chi_{j^*})\left(V(X)q_v(X) - q_w(X)\right) + (q_v(X)(X - \chi_{j^*}) + \beta_v)\beta_v - \beta_w \\ &= (X - \chi_{j^*})\left(q_v(X)\left(V(X) + \beta_v\right) - q_w(X)\right) + (\beta_v^2 - \beta_w) \ . \end{aligned} \tag{6.3}$$

Since, $X - \chi_{j^*} \nmid P(X)$ we get that $(\beta_v^2 - \beta_w) \neq 0$.

We denote by $\alpha_i(X)$ and $\beta_{v,i}$ the quotient and the remainder of the polynomial division of $v_i(X)$ by $X - \chi_{j^*}$, i.e., $v_i(X) = \alpha_i(X)(X - \chi_{j^*}) + \beta_{v,i}$. Similarly, we can also express $w_i(X) = \hat{\alpha}_i(X)(X - \chi_{j^*}) + \beta_{w,i}$. As a special case, we define $t(X) = \alpha_t(X)(X - \chi_{j^*}) + \beta_t$. The definition of $V(X)$ and Equation (6.1) give us $V(X) = \left(\sum_{i=1}^n a_i\alpha_i(X) + \delta_v\alpha_t(X)\right)(X - \chi_{j^*}) + \sum_{i=1}^n a_i\beta_{v,i} + \delta_v\beta_t$, and thus

$$q_v(X) = \sum_{i=1}^n a_i\alpha_i(X) + \delta_v\alpha_t, \quad \beta_v = \sum_{i=1}^n a_i\beta_{v,i} + \delta_v\beta_t. \tag{6.4}$$

Similarly, we get that

$$q_w(X) = \sum_{i=1}^n a_i\hat{\alpha}_i(X) + \delta_w\beta_t, \quad \beta_w = \sum_{i=1}^n a_i\beta_{w,i} + \delta_w\beta_t. \tag{6.5}$$

The security proof extracts the following functions of the witness $\boldsymbol{a}$ and δ_v, δ_w: $[q_v(s)]_2 = [\sum_{i=1}^n a_i\alpha_i(s) + \delta_v\beta_t]_2$, $[\beta_v z]_2 = [\sum_{i=1}^n a_i z\beta_{v,i} + \delta_v z\beta_t]_2$, and $[\beta_w z]_2 = [\sum_{i=1}^n a_i z\beta_{w,i} + \delta_w z\beta_t]_2$, where $z, s \in \mathbb{Z}_p$ are secrets of SATSDH assumption. The idea is that we can break the l-SATSDH assumption by computing $[\beta_v]_1 = \sum_{i=1}^n \beta_{v,i}[a_i]_1 + \beta_t[\delta_v]_1$ (note that $[a_i]_1$ and $[\delta_v]_1$ are extractable from the PB commitment and $[V]_1$), $[\beta_w]_1 = \sum_{i=1}^n \beta_{w,i}[a_i]_1 + \beta_t[\delta_w]_1$ and moreover by Equation (6.3), $\left[\frac{\beta_v^2 - \beta_w}{s - \chi_{j^*}}\right]_T = \left[\frac{P(s)}{s - \chi_{j^*}}\right]_T - ([V]_1 + [\beta_v]_1)[q_v(s)]_2 + [q_w(s)]_T$, where $[\frac{P(s)}{s - \chi_{j^*}}]_T$ can be computed from the verification equation. Together with other extracted elements, this is now enough to break the SATSDH assumption. We refer to Theorem 42 for more details.

SSB functionality in the security proof

The security proof of the argument uses similar techniques as Chapter 3 but simplified because we rely on the properties of SSB commitments. Intuitively, in the security re-

duction we need to compute some elements of the form $[\sum_i a_i y_i]_2$ where $(a_1, \ldots, a_n)$ is the witness and $[y_1, \ldots, y_n]_2$ are elements that can be computed from the challenge of some falsifiable assumption or public elements. The actual reduction requires us to extract multiple such linear combinations.

If an adversary wins the soundness game, its argument passes verification but at least one SAP equation does not hold. In the security proof, the soundness game is first changed by randomly picking one of the SAP equations $\left(\boldsymbol{a}^\top \boldsymbol{v}_{j^*}\right)^2 - \boldsymbol{a}^\top \boldsymbol{w}_{j^*} = 0$ for some $j^* \in [1..l]$. To complete the proof, we have to check the equation and break a computational assumption. For the former, since our perfectly binding commitment is only $[\cdot]_1$-extractable, we can at best extract $[a_i]_1$ which is not enough to check the j^*-th equation, even if $\boldsymbol{v}_{j^*}$ and $\boldsymbol{w}_{j^*}$ are public. We need a square of $\boldsymbol{a}$, so it suffices to extract $\sum [a_i]_2 v_{j^*,i}$ in $\mathbb{G}_2$ and prove the equation in the target group. For the latter, we break the l-SATSDH assumption 38, that is a version of the l-TSDH assumption with some extra elements that are linear combinations of the witness.

Next, we switch the EMP commitment key that is in perfectly hiding mode in the honest proof ($\mathcal{S} = \emptyset$) to the mode that encodes the functions $f(a_1, \ldots, a_n) = \sum_i a_i [y_i]_2$ that we need. Then, from $[\tilde{\boldsymbol{c}}]_2$ we can extract $[\sum_i a_i v_{j^*,i}]_2$, and so check the equation in $\mathbb{G}_T$, and also the linear combinations to break the assumption.

The *FSH* property guarantees that the adversary cannot learn the index j^* and thus the j^*-th SAP equation is not satisfied with probability $\geq 1/l$. The $[\cdot]_2$-*SSE* property allows us to extract some linear combinations of the claimed witness and break the l-SATSDH assumption. Zero-knowledge is straightforwardly guaranteed by the *AEPH* property. The full security proof is given in the following.

Proofs of security

The following two theorems prove the completeness, zero-knowledge, and soundness properties of our QA-NIZK construction.

Theorem 41. *The QA-NIZK argument has perfect completeness and perfect zero-knowledge.*

Proof. Completeness. Since the BLS argument is perfectly complete, we only need to check the last verification equation: the left hand side is $[V]_1[V]_2 - [W]_1[1]_2 = \left[V^2 - W\right]_T = [P(s)]_T$, and the right hand side is $[H]_1[t(s)]_2 = [H(s)]_1[t(s)]_2 = [P(s)]_T$.

Zero-knowledge. We prove it by showing that the proof can be efficiently simulated given the BLS trapdoor $\mathsf{td}_{\mathsf{BLS}}$. Since we set $S_v = \emptyset$, then the SSB commitments are perfectly hiding by the AEPH property. Thus we may simulate $[\tilde{\boldsymbol{c}}]_2$ by committing to $\mathbf{0}$. Next, V and W are uniformly random and independently distributed in the honest

proof. Hence, the simulator can pick $\mu_1, \mu_2 \leftarrow \mathbb{Z}_p$ and define $[V]_{1,2} = \mu_1[t(s)]_{1,2}$, $[W]_1 = \mu_2[t(s)]_1$. Then, $[H]_1 = \mu_1^2[t(s)]_1 - [\mu_2]_1$ and the verification equation will be satisfied. Finally, the BLS proof ψ can be perfectly simulated (see [67]) using the trapdoor $\mathsf{td}_{\mathsf{BLS}}$. □

Theorem 42. *Let* $\mathsf{Adv}_{snd}(\mathsf{Adv})$ *be the advantage of any PPT adversary* $\mathcal{A}$ *against the soundness of the QA-NIZK argument. There exist PPT adversaries* $\mathcal{B}_1$ *against the* DDH *assumption in* $\mathbb{G}_2$*,* $\mathcal{B}_2$ *against strong soundness of the BLS argument, and* $\mathcal{B}_3$ *against the* l-SATSDH *assumption such that*

$$\mathsf{Adv}_{\mathsf{Snd}}(\mathcal{A}) \leq 3\mathsf{Adv}_{\mathsf{DDH},\mathbb{G}_2}(\mathcal{B}_1) + l\big(\mathsf{Adv}_{\mathsf{BLS}}(\mathcal{B}_2) + \mathsf{Adv}_{l\text{-}\mathsf{SATSDH}}(\mathcal{B}_3)\big).$$

Proof. In order to prove soundness we will prove indistinguishability of the following games.

- Real: This is the real soundness game. The output is 1 if the adversary produces a false accepting proof, i.e., if there is some equation $\left(\boldsymbol{a}^\top \boldsymbol{v}_i\right)^2 - \boldsymbol{a}^\top \boldsymbol{w}_i \neq 0$ and the verifier accepts the proof. Note that $\boldsymbol{a}$ is uniquely determined since commitment $[\boldsymbol{c}]_1$ is perfectly binding.

- Game_0: This game is identical to the previous one, except instead of generating the commitment key as $\mathtt{ck} \leftarrow \mathcal{D}_{\mathsf{pp}}(n, l)$, the game samples $u \leftarrow \mathbb{Z}_p$ himself, sets $\mathtt{ck} = [1, u]_1^\top$, and stores u. Clearly, $\mathcal{A}$'s advantage is the same in Real and Game_0.

- Game_1: This game is identical to the previous one except that some $j^* \leftarrow [1..l]$ is chosen randomly and we change the commitment key $\mathtt{ck}'$ by using a different matrix $\mathbf{M} \neq \mathbf{0}$ during its generation. For each $i \in [1..n]$, let us express

$$v_i(X) = \alpha_i(X)(X - \chi_{j^*}) + \beta_{v,i}$$

$$w_i(X) = \hat{\alpha}_i(X)(X - \chi_{j^*}) + \beta_{w,i}$$

and $t(X) = \alpha_t(X)(X - \chi_{j^*}) + \beta_t$. We will pick $[z]_2 \leftarrow \mathbb{G}_2$ that is part of the SATSDH challenge and change the EMP commitment key $\mathtt{ck}'$ by setting

$$\mathbf{M} = \begin{pmatrix} \alpha_1(s) & \dots & \alpha_n(s) & \alpha_t(s) \\ \beta_{v,1} z & \dots & \beta_{v,n} z & 0 \\ \beta_{w,1} z & \dots & \beta_{w,n} z & 0 \\ v_{j^*,1} & \dots & v_{j^*,n} & 0 \end{pmatrix}.$$

It is important to note that from $\{[s^i]_{1,2}\}_{i=1}^l$ and $[z]_2$ we can only compute $[\mathbf{M}]_2$. However, looking at the KC algorithm in Figure 6.3, it is clear that $\mathtt{ck}'$ can be

computed even if only $[\mathbf{M}]_2$ is known. The game aborts if $\boldsymbol{a}$ satisfies the j^*-th equation, i.e. if $\left(\boldsymbol{a}^\top \boldsymbol{v}_{j^*}\right)^2 - \boldsymbol{a}^\top \boldsymbol{w}_{j^*} = 0$[7].

Let us now analyze the games.

Lemma 43. *There exists an adversary $\mathcal{B}_1$ against DDH in $\mathbb{G}_2$ such that* $|\Pr[\mathsf{Game}_0(\mathsf{Adv}) = 1] - \Pr[\mathsf{Game}_1(\mathsf{Adv}) = 1]| \leq 3\mathsf{Adv}_{\mathsf{DDH},\mathbb{G}_2}(\mathcal{B}_1)$.

Proof. Game_0 and Game_1 differ only in the linear EMP commitment key that encode different functions, but these keys are indistinguishable due to the FSH property. In particular, we can bound the advantage of an adversary $\mathcal{B}_1$ against the $\mathsf{DDH}_{\mathbb{G}_2}$ assumption as in Theorem 38: $\mathsf{Adv}^{fsh}_{\mathcal{A},\mathsf{COM},n,q} \leq \lceil \log_2(q+1) \rceil \cdot \mathsf{Adv}^{\mathsf{DDH}}_{\mathcal{B}_1,2,\mathcal{G}}$ where in this case $q = 4$. □

Lemma 44. *There exists an adversary $\mathcal{B}_2$ against the strong soundness of the* BLS *proof and a l-*SATSDH *adversary $\mathcal{B}_3$ such that*

$$\Pr[\mathsf{Game}_1(\mathcal{A}) = 1] \leq l\left(\mathsf{Adv}_{\mathsf{BLS}}(\mathcal{B}_2) + \mathsf{Adv}_{l\text{-}\mathsf{SATSDH}}(\mathcal{B}_3)\right).$$

Proof. First of all, if $\mathcal{A}$ breaks soundness, at least one equation j^* does not hold, and the challenger can guess j^* with probability at least $\frac{1}{l}$.

Let E be the event that $([\boldsymbol{c}]_1, [V]_1, [W]_1, [V]_2, [\tilde{\boldsymbol{c}}]_2)^\top \in \mathbf{Im}\begin{pmatrix} [\mathbf{N}_1]_1 \\ [\mathbf{N}_2]_2 \end{pmatrix}$ and $\overline{E}$ be the complementary event. Obviously,

$$\Pr[\mathsf{Game}_1(\mathcal{A}) = 1] \leq \Pr[\mathsf{Game}_1(\mathcal{A}) = 1|E] + \Pr[\mathsf{Game}_1(\mathcal{A}) = 1|\overline{E}]. \tag{6.6}$$

For the latter event, we can easily construct from $\mathcal{A}$ a PPT adversary $\mathcal{B}_2$ that breaks strong quasi-adaptive soundness of the BLS argument. Such an adversary receives as an input $(\mathsf{crs}_{\mathsf{BLS}}, \varrho = ([\mathbf{N}_1]_1, [\mathbf{N}_2]_2), \omega_\rho = (\mathbf{N}_1, \mathbf{N}_2))$ sampled according to the distribution specified by Game_3. In particular, $\mathbf{N}_2$ contains $t(s)$ and thus $\mathcal{B}_2$ can efficiently recover s by finding roots of the polynomial $t(X) - t(s)$. This is sufficient to construct the rest of the crs chosen in the usual way. Now adversary $\mathcal{B}_2$ can use the output of $\mathcal{A}$ to break the soundness of BLS in a straightforward way. Thus, $\Pr[\mathsf{Game}_1(\mathcal{A}) = 1|\overline{E}] \leq \mathsf{Adv}_{\mathsf{BLS}}(\mathcal{B}_2)$.

In the following, we bound the first term of the sum in Equation (6.6) by constructing an adversary $\mathcal{B}_3$ which breaks the d-SATSDH assumption in the case that E happens. Note that in this case there exists a witness $(\boldsymbol{a}, \boldsymbol{r}, \delta_v, \delta_w, r_v)^\top$ for membership in $\mathbf{Im}\begin{pmatrix} [\mathbf{N}_1]_1 \\ [\mathbf{N}_2]_2 \end{pmatrix}$. Furthermore, this witness is unique since

[7] This statement is well-defined since $\boldsymbol{a}$ is uniquely determined by the commitment $[\boldsymbol{c}]_1$. The check can be done in $\mathbb{G}_T$ from $[a_i]_1$ and $[\sum a_i v_{j^*,i}]_2$.

- $[\boldsymbol{c}]_1$ is perfectly binding and thus uniquely fixes $\boldsymbol{a}$ and $\boldsymbol{r}$,
- $[V]_1$ and $\boldsymbol{a}$ uniquely fix δ_v,
- $[W]_1$ and $\boldsymbol{a}$ uniquely fix δ_w, and
- $[\boldsymbol{a}]_1$ and δ_v uniquely fix r_v.

In particular, this uniquely determines the polynomial $P(X) = (v(X) + \delta_v t(X))^2 - w(X) + \delta_w t(X)$.

We now describe the full reduction. Adversary $\mathcal{B}_3$ receives the l-SATSDH assumption challenge $\left(\mathsf{pp}, \{[s^i]_{1,2}\}_{i=1}^q, [z]_2\right)$ and uses this to construct the crs just as it is specified in Game_1. Note that to create the commitment key $\mathtt{ck}'$, it constructs the matrix $\mathbf{M}$ and the corresponding extraction key $\mathtt{ek}'$. The crs is then sent to the soundness adversary $\mathcal{A}$ that returns $[\boldsymbol{c}]_1$ and π.

The adversary $\mathcal{B}_3$ extracts $[\boldsymbol{a}]_1, [\delta_v]_1, [\delta_w]_1 \in \mathbb{G}_1$ from $[\boldsymbol{c}]_1$ by using the secret key u; and extracts $[q_v(s)]_2 = [\sum_{i=1}^{n+1} a_i \alpha_i(s) + \delta_v \alpha_{n+1}(s)]_2$, $[\beta_v z]_2$, $[\beta_w z]_2$ and $[\sum_i a_i v_{j^*,i}]_2$ from $\mathtt{ek}'$. Then it aborts if the j^*-th equation is satisfied, i.e. if

$$\left(\sum_{i=1}^{n} [a_i]_1 v_{j^*,i}\right) \cdot \left[\sum_{i=1}^{n} a_i v_{j^*,i}\right]_2 - \left(\sum_{i=1}^{n} [a_i]_1 w_{j^*,i}\right) \cdot [1]_2 = [0]_T.$$

Since verification succeeds, $[V]_1[V]_2 - [W]_T = [H(s)]_1[t(s)]_2$. By the definition of $P(X)$, we have that the left hand side is $[V^2 - W]_T = [P(s)]_T$.

If we divide both sides of the verification equation by $s - \chi_{j^*}$, then

$$\left[\frac{P(s)}{s-\chi_{j^*}}\right]_T = [H]_1 \cdot \left[\frac{t(s)}{s-\chi_{j^*}}\right]_2 = [H]_1 \cdot \left[\prod_{i \neq j^*} (s - \chi_i)\right]_2,$$

so the adversary $\mathcal{B}_3$ can compute $\left[\frac{P(s)}{s-\chi_{j^*}}\right]_T$ from $[H]_1$ and the powers of $[s]_2$ in the crs. On the other hand, if we use equation (6.3) on $P(X)$, then

$$\left[\frac{P(s)}{s-\chi_{j^*}}\right]_T = \left[(V(s) + \beta_v) q_v(s) - q_w(s) + \frac{\beta_v^2 - \beta_w}{s - \chi_{j^*}}\right]_T, \tag{6.7}$$

and we have $\beta_v^2 - \beta_w \neq 0$ (otherwise the j^*-th equation is satisfied, in which case the game aborts). We describe in the following how $\mathcal{B}_3$ can compute the right hand side of Equation (6.7) and the elements to break the d-SATSDH Assumption.

According to Equation (6.4) and Equation (6.5), $\mathcal{B}_3$ can compute $[\beta_v]_1 = \sum_{i=0}^{n}[a_i]_1\beta_{v,i} + [\delta_v]_1\beta_t$, $[\beta_w]_1 = \sum_{i=0}^{n}[a_i]_1\beta_{w,i} + [\delta_w]_1\beta_t$ and also $[V(s) + \beta_v]_1 = [V]_1 + [\beta_v]_1$, because it knows $[V]_1$ from the proof π and the extracted values $[a_i]_1$, and β_i are the reminders of dividing $V_i(X)$ by $X - \chi_{j^*}$.

From these values, the extracted values and $[V(s)+\beta_v]_2$, $\mathcal{B}_3$ can derive $[(V(s) + \beta_v)q_v(s)]_T$ as $[V(s) + \beta_v]_1 \cdot [q_v(s)]_2$. Finally, it can directly compute $[q_w(s)]_T$ from extracted elements $[a_i]_1$ for $i \in [1..n]$ and $[\delta_w]_1$, and public $\hat{\alpha}_i(s)$: $[\sum_{i=1}^{n} a_i\hat{\alpha}_i(s) + \delta_w\beta_t]_1$. Thus, from equation (6.7) $\mathcal{B}_3$ recovers $\left[\frac{\beta_v^2 - \beta_w}{s - \chi_{j^*}}\right]_T$ and returns

$$\left(\chi_{j^*}, [\beta_v]_1, [\beta_w]_1, [z\beta_v]_2, [z\beta_w]_2, \left[\frac{\beta_v^2 - \beta_w}{s - \chi_{j^*}}\right]_T\right),$$

breaking the l-SATSDH assumption.

Hence by the triangle inequality we have $\frac{1}{l}\Pr[\mathsf{Game}_1(\mathcal{A}) = 1] \leq \mathsf{Adv}_{\mathsf{BLS}}(\mathcal{B}_2) + \mathsf{Adv}_{l\text{-SATSDH}}(\mathcal{B}_3)$. □

Finally, by Lemmas 43 and 44 we get that

$$\mathsf{Adv}_{\mathrm{Snd}}(\mathcal{A}) \leq 3\mathsf{Adv}_{\mathsf{DDH},\mathbb{G}_2}(\mathcal{B}_1) + l\big(\mathsf{Adv}_{\mathsf{BLS}}(\mathcal{B}_2) + \mathsf{Adv}_{l\text{-SATSDH}}(\mathcal{B}_3)\big).$$

□

Efficiency

The proof size in the original construction in Chapter 3 is 4 elements in $\mathbb{G}_1$ and 6 elements in $\mathbb{G}_2$, while our construction's proof size is 5 elements in $\mathbb{G}_1$ and 8 elements in $\mathbb{G}_2$.

6.7 Relation to Existing Primitives

6.7.1 Relation to SSB Hashes

The SSB requirement makes the EMP commitment scheme look similar to SSB hash functions [83, 107], in which one can compute a hash of a vector v such that the computed hash is statistically binding in one coordinate of v. However, there are also obvious differences. First, to obtain zero-knowledge, we need hiding (AESH) that is not required from hash functions. This is, intuitively, a natural distinction and corresponds to the difference between collision-resistant hash families and statistically hiding commitment schemes.

Second, [83, 107] require that an SSB hash has the local opening property, meaning that the committer can efficiently open just one coordinate of the committed vector. In the QA-NIZK application, we do not need this property: the commitment key $\mathtt{ck}$ is created by a trusted third party, and there is no need for the honest parties to ever open the commitment. Instead, in the soundness proof, we need *somewhere statistical extractability* (SSE), stating that the creator of $\mathtt{ck}$ (e.g., the adversary $\mathcal{B}$) must be able to extract the succinct guilt witness. SSE is not needed in the case of SSB hashes. Although not needed in our concrete applications, it is also desirable to have the *almost everywhere statistical trapdoor* (AEST) property, where the creator of $\mathtt{ck}$ is able to replace non-SB coordinates with anything she wishes. Finally, we allow $\mathtt{ck}$ to be long, but require commitments to be succinct.

The properties of SSB and local opening are orthogonal: it is possible to construct efficient SSB hashes without local opening [107] and efficient vector commitments [98, 33] (which have a local opening) without the SSB property.

6.7.2 Relation to Oblivious Transfer (OT)

SSB commitments are directly related to two-message OT protocols as defined in [8]. In an OT protocol, the sender has an n-element database and the chooser has an index-set $\mathcal{S}$ with $|\mathcal{S}| \leq q$. The chooser wants to obtain $\boldsymbol{x}_{\mathcal{S}}$; no additional information should be leaked either to the chooser or the sender. In a two-message OT protocol (in the plain model), the chooser sends the first message $\mathtt{otq}$ (an encoding of $\mathcal{S}$) to the sender who replies with the second message $\mathtt{otr}$ (an encoding of $\boldsymbol{x}_{\mathcal{S}}$). OT protocols have very wide applications in many areas of cryptography, with two-message OT protocols in the plain model such as [106, 8, 62, 99] being of special interest because of their efficiency.

Essentially, SSB commitments are non-interactive analogues of such protocols, the commitment key corresponding to the first OT message ot_1, and the commitment corresponding to the second OT message ot_2. However, the connection is not completely one-to-one, since there are subtle differences in the security definitions between SSB commitment schemes and OT protocols. Importantly, while in OT, the ot_1 generator is always untrusted, in our applications it is sufficient to consider a trusted $\mathtt{ck}$ generator, which allows to develop more efficient constructions. Additionally, SSB commitment schemes (such as EMP) result in a flavour of OT where the receiver's message ot_1 is long but can be reused multiple times, while the sender's message ot_2 is much shorter.

Thus, all secure two-message OT protocols are also secure SSB commitment schemes. Unfortunately, none of the known efficient two-message OT protocols have the required algebraic structure to construct QA-NIZKs, and thus they are unsuitable for our main application.

6.7.3 Relation to PCP-Based zk-SNARKs

The QA-NIZK application of SSB commitments is based on the observation that the language of bit-strings (resp., CircuitSAT) has a local verifiability property, similar to PCP [10, 9]: one can establish, by checking one random coordinate of the bit-string (resp., all adjacent wires of a random gate), whether an input belongs to the language or not. Typical PCP-based zero-knowledge arguments like [91] use PCPs with small soundness error; as a drawback, such PCPs have a long proof and an inefficient reduction from CircuitSAT. The construction in Chapter 3 and the current contribution use a trivial PCP with a large soundness error but with a trivial reduction from CircuitSAT. The use of SSB commitments means that the efficiency loss is logarithmic in n (one needs to use $\approx 2 \log n$-bit longer group elements) while in the case of earlier PCP-based arguments the efficiency loss is much larger. Nevertheless, the use of SSB commitments is not limited to trivial PCP; one can use them together with any PCP that has a small number of queries and short proof length.

Conclusion

A common theme in this thesis is to reduce the strength of the assumptions used in the security proof, while keeping the efficiency of the constructions. We emphasize two techniques that we believe have potential for future work.

We constructed the first simulation-extractable QA-NIZK argument for boolean CircuitSat, that is sub-linear in the circuit size but has full extraction under falsifiable assumptions. To achieve complete extraction of the witness, all previous simulation-extractable QA-NIZK proofs either use non-falsifiable assumptions or a proof linear in the witness size. Then, the knowledge soundness under falsifiable assumptions for non-interactive proofs, requires to have a linear proof. Our construction follows an approach presented in González and Ràfols [69]. Briefly, the witness is split into smaller pieces that are interlinked in a chain: if we have knowledge of the first piece and the link to the next piece is correctly done, this knowledge is transferred in some way. Like in a chain, if we prove the correctness of the links between each piece to the next one, knowledge is transferred up to the final piece. This idea that naturally appears in CircuitSat approach, we believe that can be exploited in other contexts to reduce the proof size and improve the analysis security.

In some cases, arguments for membership in linear spaces are used as proof of knowledge. For example, Campanelli et al.[31] use QA-NIZK argument to prove that two Pedersen commitments, that are perfectly hiding, open to the same values. If we try to write it as a proof of membership in linear spaces, we would express the space as the image of some full rank matrix with more columns than rows. Since the image of this matrix is the whole space, this only makes sense if we use the QA-NIZK as a proof of knowledge of the witness. Our techniques extends the work of González and Ràfols [69], which gives a way to analyse this use of QA-NIZK schemes under falsfiable assumptions, in the simulation soundness setting.

Finally, we construct a simulation-extractable QA-NIZK for boolean CircuitSat, which is sub-linear in the circuit size. We believe that other signatures can be developed from this construction similarly as our SoK. For example, an Attribute-Based Signature

where the signer has to prove knowledge the secret keys of some attributes that satisfy a certain circuit (signing policy). We think our techniques can be used to design an Attribute-Based Signature scheme for general signing policies.

Acknowledgement.

The research leading to this thesis was partially supported by Project RTI2018-102112-B-I00 (AEI/FEDER, UE). The mobility visit of the author was supported by the Estonian Research Council grant (PRG49) and by Dora Plus Grant funded by the European Regional Development Fund, Republic of Estonia and Archimedes Foundation.

Bibliography

[1] Monero, 2014. https://getmonero.org/.

[2] Zcash, 2016. https://z.cash/.

[3] iden3, 2019. https://www.iden3.io/.

[4] B. Abdolmaleki, K. Baghery, H. Lipmaa, and M. Zajac. A subversion-resistant SNARK. In T. Takagi and T. Peyrin, editors, *ASIACRYPT 2017, Part III*, volume 10626 of *LNCS*, pages 3–33. Springer, Heidelberg, Dec. 2017.

[5] M. Abe, C. S. Jutla, M. Ohkubo, J. Pan, A. Roy, and Y. Wang. Shorter QA-NIZK and SPS with tighter security. In S. D. Galbraith and S. Moriai, editors, *ASIACRYPT 2019, Part III*, volume 11923 of *LNCS*, pages 669–699. Springer, Heidelberg, Dec. 2019.

[6] M. Abe, C. S. Jutla, M. Ohkubo, and A. Roy. Improved (almost) tightly-secure simulation-sound QA-NIZK with applications. In T. Peyrin and S. Galbraith, editors, *ASIACRYPT 2018, Part I*, volume 11272 of *LNCS*, pages 627–656. Springer, Heidelberg, Dec. 2018.

[7] A. Aggelakis, P. Fauzi, G. Korfiatis, P. Louridas, F. Mergoupis-Anagnou, J. Siim, and M. Zajac. A non-interactive shuffle argument with low trust assumptions. In S. Jarecki, editor, *CT-RSA 2020*, volume 12006 of *LNCS*, pages 667–692. Springer, Heidelberg, Feb. 2020.

[8] W. Aiello, Y. Ishai, and O. Reingold. Priced oblivious transfer: How to sell digital goods. In B. Pfitzmann, editor, *EUROCRYPT 2001*, volume 2045 of *LNCS*, pages 119–135. Springer, Heidelberg, May 2001.

[9] S. Arora, C. Lund, R. Motwani, M. Sudan, and M. Szegedy. Proof verification and hardness of approximation problems. In *33rd FOCS*, pages 14–23. IEEE Computer Society Press, Oct. 1992.

[10] S. Arora and S. Safra. Probabilistic checking of proofs; A new characterization of NP. In *33rd FOCS*, pages 2–13. IEEE Computer Society Press, Oct. 1992.

[11] S. Atapoor and K. Baghery. Simulation extractability in Groth's zk-SNARK. In *Data Privacy Management, Cryptocurrencies and Blockchain Technology - ESORICS 2019 International Workshops, DPM 2019 and CBT 2019*, volume 11737 of *LNCS*, pages 336–354. Springer, 2019.

[12] K. Baghery. On the efficiency of privacy-preserving smart contract systems. In J. Buchmann, A. Nitaj, and T. eddine Rachidi, editors, *AFRICACRYPT 19*, volume 11627 of *LNCS*, pages 118–136. Springer, Heidelberg, July 2019.

[13] K. Baghery. Subversion-resistant simulation (knowledge) sound NIZKs. In M. Albrecht, editor, *17th IMA International Conference on Cryptography and Coding*, volume 11929 of *LNCS*, pages 42–63. Springer, Heidelberg, Dec. 2019.

[14] K. Baghery, A. González, Z. Pindado, and C. Ràfols. Signatures of knowledge for boolean circuits under standard assumptions. In A. Nitaj and A. M. Youssef, editors, *AFRICACRYPT 20*, volume 12174 of *LNCS*, pages 24–44. Springer, Heidelberg, July 2020.

[15] K. Baghery, M. Kohlweiss, J. Siim, and M. Volkhov. Another look at extraction and randomization of groth's zk-SNARK. Cryptology ePrint Archive, Report 2020/811, 2020. `https://eprint.iacr.org/2020/811`.

[16] K. Baghery, Z. Pindado, and C. Ràfols. Simulation extractable versions of groth's zk-SNARK revisited. In *CANS 20*, LNCS, pages 453–461. Springer, Heidelberg, 2020.

[17] M. Belenkiy, M. Chase, M. Kohlweiss, and A. Lysyanskaya. P-signatures and noninteractive anonymous credentials. In R. Canetti, editor, *TCC 2008*, volume 4948 of *LNCS*, pages 356–374. Springer, Heidelberg, Mar. 2008.

[18] E. Ben-Sasson, A. Chiesa, C. Garman, M. Green, I. Miers, E. Tromer, and M. Virza. Zerocash: Decentralized anonymous payments from bitcoin. In *2014 IEEE Symposium on Security and Privacy*, pages 459–474. IEEE Computer Society Press, May 2014.

[19] D. Bernhard, G. Fuchsbauer, and E. Ghadafi. Efficient signatures of knowledge and DAA in the standard model. In M. J. Jacobson Jr., M. E. Locasto, P. Mohassel, and R. Safavi-Naini, editors, *ACNS 13*, volume 7954 of *LNCS*, pages 518–533. Springer, Heidelberg, June 2013.

[20] M. Blum, P. Feldman, and S. Micali. Non-Interactive Zero-Knowledge and Its Applications. In *STOC 1988*, pages 103–112, Chicago, Illinois, USA, May 2–4, 1988. ACM Press.

[21] D. Boneh and X. Boyen. Secure identity based encryption without random oracles. In M. Franklin, editor, *CRYPTO 2004*, volume 3152 of *LNCS*, pages 443–459. Springer, Heidelberg, Aug. 2004.

[22] D. Boneh and X. Boyen. Short signatures without random oracles and the SDH assumption in bilinear groups. *Journal of Cryptology*, 21(2):149–177, Apr. 2008.

[23] D. Boneh, X. Boyen, and H. Shacham. Short group signatures. In M. Franklin, editor, *CRYPTO 2004*, volume 3152 of *LNCS*, pages 41–55. Springer, Heidelberg, Aug. 2004.

[24] J. Bonneau, I. Meckler, V. Rao, and E. Shapiro. Coda: Decentralized cryptocurrency at scale. Cryptology ePrint Archive, Report 2020/352, 2020. `https://eprint.iacr.org/2020/352`.

[25] J. Bootle and J. Groth. Efficient batch zero-knowledge arguments for low degree polynomials. In M. Abdalla and R. Dahab, editors, *PKC 2018, Part II*, volume 10770 of *LNCS*, pages 561–588. Springer, Heidelberg, Mar. 2018.

[26] S. Bowe and A. Gabizon. Making groth's zk-SNARK simulation extractable in the random oracle model. Cryptology ePrint Archive, Report 2018/187, 2018. `https://eprint.iacr.org/2018/187`.

[27] S. Bowe, A. Gabizon, and I. Miers. Scalable multi-party computation for zk-SNARK parameters in the random beacon model. Cryptology ePrint Archive, Report 2017/1050, 2017. `http://eprint.iacr.org/2017/1050`.

[28] B. Bünz, J. Bootle, D. Boneh, A. Poelstra, P. Wuille, and G. Maxwell. Bulletproofs: Short proofs for confidential transactions and more. In *2018 IEEE Symposium on Security and Privacy*, pages 315–334. IEEE Computer Society Press, May 2018.

[29] V. Buterin. A next-generation smart contract and decentralized application platform, 2014. white paper, https://cryptorating.eu/whitepapers/Ethereum/Ethereum_white_paper.pdf.

[30] J. Camenisch, R. Chaabouni, and a. shelat. Efficient protocols for set membership and range proofs. In J. Pieprzyk, editor, *ASIACRYPT 2008*, volume 5350 of *LNCS*, pages 234–252. Springer, Heidelberg, Dec. 2008.

[31] M. Campanelli, D. Fiore, and A. Querol. LegoSNARK: Modular design and composition of succinct zero-knowledge proofs. In L. Cavallaro, J. Kinder, X. Wang, and J. Katz, editors, *ACM CCS 2019*, pages 2075–2092. ACM Press, Nov. 2019.

[32] R. Canetti, Y. Lindell, R. Ostrovsky, and A. Sahai. Universally composable two-party and multi-party secure computation. In *34th ACM STOC*, pages 494–503. ACM Press, May 2002.

[33] D. Catalano and D. Fiore. Vector commitments and their applications. In K. Kurosawa and G. Hanaoka, editors, *PKC 2013*, volume 7778 of *LNCS*, pages 55–72. Springer, Heidelberg, Feb. / Mar. 2013.

[34] D. Catalano and I. Visconti. Hybrid trapdoor commitments and their applications. In L. Caires, G. F. Italiano, L. Monteiro, C. Palamidessi, and M. Yung, editors, *ICALP 2005*, volume 3580 of *LNCS*, pages 298–310. Springer, Heidelberg, July 2005.

[35] M. Chase, C. Ganesh, and P. Mohassel. Efficient zero-knowledge proof of algebraic and non-algebraic statements with applications to privacy preserving credentials. In M. Robshaw and J. Katz, editors, *CRYPTO 2016, Part III*, volume 9816 of *LNCS*, pages 499–530. Springer, Heidelberg, Aug. 2016.

[36] M. Chase and A. Lysyanskaya. On signatures of knowledge. In C. Dwork, editor, *CRYPTO 2006*, volume 4117 of *LNCS*, pages 78–96. Springer, Heidelberg, Aug. 2006.

[37] J. H. Cheon. Security analysis of the strong Diffie-Hellman problem. In S. Vaudenay, editor, *EUROCRYPT 2006*, volume 4004 of *LNCS*, pages 1–11. Springer, Heidelberg, May / June 2006.

[38] A. Chiesa, M. A. Forbes, and N. Spooner. A zero knowledge sumcheck and its applications. Cryptology ePrint Archive, Report 2017/305, 2017. http://eprint.iacr.org/2017/305.

[39] S. A. Cook. The complexity of theorem-proving procedures. *ACM Symposium on Theory of Computing*, pages 151–158, May 1971.

[40] G. Couteau and D. Hartmann. Shorter non-interactive zero-knowledge arguments and ZAPs for algebraic languages. In D. Micciancio and T. Ristenpart, editors, *CRYPTO 2020, Part III*, volume 12172 of *LNCS*, pages 768–798. Springer, Heidelberg, Aug. 2020.

[41] I. Damgård. Collision free hash functions and public key signature schemes. In D. Chaum and W. L. Price, editors, *EUROCRYPT'87*, volume 304 of *LNCS*, pages 203–216. Springer, Heidelberg, Apr. 1988.

[42] I. Damgård. Towards practical public key systems secure against chosen ciphertext attacks. In J. Feigenbaum, editor, *CRYPTO'91*, volume 576 of *LNCS*, pages 445–456. Springer, Heidelberg, Aug. 1992.

[43] I. Damgård, S. Fehr, C. Lunemann, L. Salvail, and C. Schaffner. Improving the security of quantum protocols via commit-and-open. In S. Halevi, editor, *CRYPTO 2009*, volume 5677 of *LNCS*, pages 408–427. Springer, Heidelberg, Aug. 2009.

[44] I. Damgård and J. B. Nielsen. Perfect hiding and perfect binding universally composable commitment schemes with constant expansion factor. In M. Yung, editor, *CRYPTO 2002*, volume 2442 of *LNCS*, pages 581–596. Springer, Heidelberg, Aug. 2002.

[45] G. Danezis, C. Fournet, J. Groth, and M. Kohlweiss. Square span programs with applications to succinct NIZK arguments. In P. Sarkar and T. Iwata, editors, *ASIACRYPT 2014, Part I*, volume 8873 of *LNCS*, pages 532–550. Springer, Heidelberg, Dec. 2014.

[46] V. Daza, A. González, Z. Pindado, C. Ràfols, and J. Silva. Shorter quadratic QA-NIZK proofs. In D. Lin and K. Sako, editors, *PKC 2019, Part I*, volume 11442 of *LNCS*, pages 314–343. Springer, Heidelberg, Apr. 2019.

[47] N. Döttling, S. Garg, Y. Ishai, G. Malavolta, T. Mour, and R. Ostrovsky. Trapdoor hash functions and their applications. In A. Boldyreva and D. Micciancio, editors, *CRYPTO 2019, Part III*, volume 11694 of *LNCS*, pages 3–32. Springer, Heidelberg, Aug. 2019.

[48] N. Döttling, S. Ghosh, J. B. Nielsen, T. Nilges, and R. Trifiletti. TinyOLE: Efficient actively secure two-party computation from oblivious linear function

evaluation. In B. M. Thuraisingham, D. Evans, T. Malkin, and D. Xu, editors, *ACM CCS 2017*, pages 2263–2276. ACM Press, Oct. / Nov. 2017.

[49] N. Döttling, D. Kraschewski, and J. Müller-Quade. Statistically secure linear-rate dimension extension for oblivious affine function evaluation. In A. Smith, editor, *ICITS 12*, volume 7412 of *LNCS*, pages 111–128. Springer, Heidelberg, Aug. 2012.

[50] A. Escala and J. Groth. Fine-tuning Groth-Sahai proofs. In H. Krawczyk, editor, *PKC 2014*, volume 8383 of *LNCS*, pages 630–649. Springer, Heidelberg, Mar. 2014.

[51] A. Escala and J. Groth. Fine-Tuning Groth-Sahai Proofs. In H. Krawczyk, editor, *PKC 2014*, volume ? of *LNCS*, pages 630–649, Buenos Aires, Argentina, March 26–28, 2014. Springer, Heidelberg.

[52] A. Escala, G. Herold, E. Kiltz, C. Ràfols, and J. Villar. An algebraic framework for Diffie-Hellman assumptions. In R. Canetti and J. A. Garay, editors, *CRYPTO 2013, Part II*, volume 8043 of *LNCS*, pages 129–147. Springer, Heidelberg, Aug. 2013.

[53] P. Fauzi, H. Lipmaa, J. Siim, and M. Zajac. An efficient pairing-based shuffle argument. In T. Takagi and T. Peyrin, editors, *ASIACRYPT 2017, Part II*, volume 10625 of *LNCS*, pages 97–127. Springer, Heidelberg, Dec. 2017.

[54] P. Fauzi, S. Meiklejohn, R. Mercer, and C. Orlandi. Quisquis: A new design for anonymous cryptocurrencies. In S. D. Galbraith and S. Moriai, editors, *ASIACRYPT 2019, Part I*, volume 11921 of *LNCS*, pages 649–678. Springer, Heidelberg, Dec. 2019.

[55] U. Feige, A. Fiat, and A. Shamir. Zero-knowledge proofs of identity. *Journal of Cryptology*, 1(2):77–94, June 1988.

[56] G. Fuchsbauer. Subversion-zero-knowledge SNARKs. In M. Abdalla and R. Dahab, editors, *PKC 2018, Part I*, volume 10769 of *LNCS*, pages 315–347. Springer, Heidelberg, Mar. 2018.

[57] G. Fuchsbauer, E. Kiltz, and J. Loss. The algebraic group model and its applications. In H. Shacham and A. Boldyreva, editors, *CRYPTO 2018, Part II*, volume 10992 of *LNCS*, pages 33–62. Springer, Heidelberg, Aug. 2018.

[58] G. Fuchsbauer, M. Orrù, and Y. Seurin. Aggregate cash systems: A cryptographic investigation of Mimblewimble. In Y. Ishai and V. Rijmen, editors, *EUROCRYPT 2019, Part I*, volume 11476 of *LNCS*, pages 657–689. Springer, Heidelberg, May 2019.

[59] S. Galbraith, K. Paterson, and N. Smart. Pairings for cryptographers. Cryptology ePrint Archive, Report 2006/165, 2006. `http://eprint.iacr.org/2006/165`.

[60] R. Gay, D. Hofheinz, L. Kohl, and J. Pan. More efficient (almost) tightly secure structure-preserving signatures. In J. B. Nielsen and V. Rijmen, editors, *EUROCRYPT 2018, Part II*, volume 10821 of *LNCS*, pages 230–258. Springer, Heidelberg, Apr. / May 2018.

[61] R. Gennaro, C. Gentry, B. Parno, and M. Raykova. Quadratic span programs and succinct NIZKs without PCPs. In T. Johansson and P. Q. Nguyen, editors, *EUROCRYPT 2013*, volume 7881 of *LNCS*, pages 626–645. Springer, Heidelberg, May 2013.

[62] C. Gentry and Z. Ramzan. Single-database private information retrieval with constant communication rate. In L. Caires, G. F. Italiano, L. Monteiro, C. Palamidessi, and M. Yung, editors, *ICALP 2005*, volume 3580 of *LNCS*, pages 803–815. Springer, Heidelberg, July 2005.

[63] C. Gentry and D. Wichs. Separating Succinct Non-Interactive Arguments from All Falsifiable Assumptions. In S. Vadhan, editor, *STOC 2011*, pages 99–108, San Jose, California, USA, June 6–8, 2011. ACM Press.

[64] C. Gentry and D. Wichs. Separating succinct non-interactive arguments from all falsifiable assumptions. In L. Fortnow and S. P. Vadhan, editors, *43rd ACM STOC*, pages 99–108. ACM Press, June 2011.

[65] S. Ghosh, J. B. Nielsen, and T. Nilges. Maliciously secure oblivious linear function evaluation with constant overhead. In T. Takagi and T. Peyrin, editors, *ASIACRYPT 2017, Part I*, volume 10624 of *LNCS*, pages 629–659. Springer, Heidelberg, Dec. 2017.

[66] S. Goldwasser, S. Micali, and C. Rackoff. The Knowledge Complexity of Interactive Proof Systems. *SIAM J. Comput.*, 18(1):186–208, 1989.

[67] A. González, A. Hevia, and C. Ràfols. QA-NIZK arguments in asymmetric groups: New tools and new constructions. In T. Iwata and J. H. Cheon, editors,

ASIACRYPT 2015, Part I, volume 9452 of *LNCS*, pages 605–629. Springer, Heidelberg, Nov. / Dec. 2015.

[68] A. González and C. Ràfols. New techniques for non-interactive shuffle and range arguments. In M. Manulis, A.-R. Sadeghi, and S. Schneider, editors, *ACNS 16*, volume 9696 of *LNCS*, pages 427–444. Springer, Heidelberg, June 2016.

[69] A. González and C. Ràfols. Shorter pairing-based arguments under standard assumptions. In S. D. Galbraith and S. Moriai, editors, *ASIACRYPT 2019, Part III*, volume 11923 of *LNCS*, pages 728–757. Springer, Heidelberg, Dec. 2019.

[70] J. Groth. Simulation-sound NIZK proofs for a practical language and constant size group signatures. In X. Lai and K. Chen, editors, *ASIACRYPT 2006*, volume 4284 of *LNCS*, pages 444–459. Springer, Heidelberg, Dec. 2006.

[71] J. Groth. Short pairing-based non-interactive zero-knowledge arguments. In M. Abe, editor, *ASIACRYPT 2010*, volume 6477 of *LNCS*, pages 321–340. Springer, Heidelberg, Dec. 2010.

[72] J. Groth. On the size of pairing-based non-interactive arguments. In M. Fischlin and J.-S. Coron, editors, *EUROCRYPT 2016, Part II*, volume 9666 of *LNCS*, pages 305–326. Springer, Heidelberg, May 2016.

[73] J. Groth and S. Lu. A non-interactive shuffle with pairing based verifiability. In K. Kurosawa, editor, *ASIACRYPT 2007*, volume 4833 of *LNCS*, pages 51–67. Springer, Heidelberg, Dec. 2007.

[74] J. Groth and M. Maller. Snarky signatures: Minimal signatures of knowledge from simulation-extractable SNARKs. In J. Katz and H. Shacham, editors, *CRYPTO 2017, Part II*, volume 10402 of *LNCS*, pages 581–612. Springer, Heidelberg, Aug. 2017.

[75] J. Groth, R. Ostrovsky, and A. Sahai. Perfect non-interactive zero knowledge for NP. In S. Vaudenay, editor, *EUROCRYPT 2006*, volume 4004 of *LNCS*, pages 339–358. Springer, Heidelberg, May / June 2006.

[76] J. Groth, R. Ostrovsky, and A. Sahai. New Techniques for Noninteractive Zero-Knowledge. *Journal of the ACM*, 59(3), 2012.

[77] J. Groth, R. Ostrovsky, and A. Sahai. New techniques for noninteractive zero-knowledge. J. ACM, 59(3):11, 2012.

[78] J. Groth and A. Sahai. Efficient non-interactive proof systems for bilinear groups. In N. P. Smart, editor, *EUROCRYPT 2008*, volume 4965 of *LNCS*, pages 415–432. Springer, Heidelberg, Apr. 2008.

[79] J. Groth and A. Sahai. Efficient Noninteractive Proof Systems for Bilinear Groups. *SIAM J. Comput.*, 41(5):1193–1232, 2012.

[80] D. Hofheinz, D. Jia, and J. Pan. Identity-based encryption tightly secure under chosen-ciphertext attacks. In T. Peyrin and S. Galbraith, editors, *ASIACRYPT 2018, Part II*, volume 11273 of *LNCS*, pages 190–220. Springer, Heidelberg, Dec. 2018.

[81] D. Hofheinz and E. Kiltz. Secure hybrid encryption from weakened key encapsulation. In A. Menezes, editor, *CRYPTO 2007*, volume 4622 of *LNCS*, pages 553–571. Springer, Heidelberg, Aug. 2007.

[82] D. Hopwood, S. Bowe, T. Hornby, and N. Wilcox. Zcash protocol specification, 2020. `https://github.com/zcash`.

[83] P. Hubacek and D. Wichs. On the communication complexity of secure function evaluation with long output. In T. Roughgarden, editor, *ITCS 2015*, pages 163–172. ACM, Jan. 2015.

[84] A. Joux. A one round protocol for tripartite Diffie-Hellman. *Journal of Cryptology*, 17(4):263–276, Sept. 2004.

[85] C. S. Jutla and A. Roy. Shorter quasi-adaptive NIZK proofs for linear subspaces. In K. Sako and P. Sarkar, editors, *ASIACRYPT 2013, Part I*, volume 8269 of *LNCS*, pages 1–20. Springer, Heidelberg, Dec. 2013.

[86] C. S. Jutla and A. Roy. Switching lemma for bilinear tests and constant-size NIZK proofs for linear subspaces. In J. A. Garay and R. Gennaro, editors, *CRYPTO 2014, Part II*, volume 8617 of *LNCS*, pages 295–312. Springer, Heidelberg, Aug. 2014.

[87] M. Karchmer and A. Wigderson. On span programs. In *Proceedings of Structures in Complexity Theory*, pages 102–111, 1993.

[88] S. Katsumata, R. Nishimaki, S. Yamada, and T. Yamakawa. Compact NIZKs from standard assumptions on bilinear maps. In A. Canteaut and Y. Ishai, editors, *EUROCRYPT 2020, Part III*, volume 12107 of *LNCS*, pages 379–409. Springer, Heidelberg, May 2020.

[89] J. Katz and Y. Lindell. Introduction to modern cryptography., 2015. CRC Press, ISBN: 9781466570269.

[90] T. Kerber, A. Kiayias, M. Kohlweiss, and V. Zikas. Ouroboros crypsinous: Privacy-preserving proof-of-stake. In *2019 IEEE Symposium on Security and Privacy*, pages 157–174. IEEE Computer Society Press, May 2019.

[91] J. Kilian. On the complexity of bounded-interaction and noninteractive zero-knowledge proofs. In *35th FOCS*, pages 466–477. IEEE Computer Society Press, Nov. 1994.

[92] E. Kiltz and H. Wee. Quasi-adaptive NIZK for linear subspaces revisited. In E. Oswald and M. Fischlin, editors, *EUROCRYPT 2015, Part II*, volume 9057 of *LNCS*, pages 101–128. Springer, Heidelberg, Apr. 2015.

[93] A. E. Kosba, A. Miller, E. Shi, Z. Wen, and C. Papamanthou. Hawk: The Blockchain Model of Cryptography and Privacy-Preserving Smart Contracts. pages 839–858, 2016.

[94] A. E. Kosba, A. Miller, E. Shi, Z. Wen, and C. Papamanthou. Hawk: The blockchain model of cryptography and privacy-preserving smart contracts. In *2016 IEEE Symposium on Security and Privacy*, pages 839–858. IEEE Computer Society Press, May 2016.

[95] L. A. Levin. Universal search problems. *Problems of Information Transmission, Vol. 9, No. 3. (1973)*, 1973.

[96] B. Libert, T. Peters, M. Joye, and M. Yung. Linearly homomorphic structure-preserving signatures and their applications. In R. Canetti and J. A. Garay, editors, *CRYPTO 2013, Part II*, volume 8043 of *LNCS*, pages 289–307. Springer, Heidelberg, Aug. 2013.

[97] B. Libert, T. Peters, M. Joye, and M. Yung. Non-malleability from malleability: Simulation-sound quasi-adaptive NIZK proofs and CCA2-secure encryption from homomorphic signatures. In P. Q. Nguyen and E. Oswald, editors, *EUROCRYPT 2014*, volume 8441 of *LNCS*, pages 514–532. Springer, Heidelberg, May 2014.

[98] B. Libert and M. Yung. Concise mercurial vector commitments and independent zero-knowledge sets with short proofs. In D. Micciancio, editor, *TCC 2010*, volume 5978 of *LNCS*, pages 499–517. Springer, Heidelberg, Feb. 2010.

[99] H. Lipmaa. An oblivious transfer protocol with log-squared communication. In J. Zhou, J. Lopez, R. H. Deng, and F. Bao, editors, *ISC 2005*, volume 3650 of *LNCS*, pages 314–328. Springer, Heidelberg, Sept. 2005.

[100] H. Lipmaa. Progression-free sets and sublinear pairing-based non-interactive zero-knowledge arguments. In R. Cramer, editor, *TCC 2012*, volume 7194 of *LNCS*, pages 169–189. Springer, Heidelberg, Mar. 2012.

[101] H. Lipmaa. Succinct non-interactive zero knowledge arguments from span programs and linear error-correcting codes. In K. Sako and P. Sarkar, editors, *ASIACRYPT 2013, Part I*, volume 8269 of *LNCS*, pages 41–60. Springer, Heidelberg, Dec. 2013.

[102] H. Lipmaa. Simulation-extractable SNARKs revisited. Cryptology ePrint Archive, Report 2019/612, 2019. `https://eprint.iacr.org/2019/612`.

[103] H. Lipmaa. Simulation-extractable SNARKs revisited. Cryptology ePrint Archive, Report 2019/612, 2019. `http://eprint.iacr.org/2019/612`.

[104] P. Morillo, C. Ràfols, and J. L. Villar. The kernel matrix Diffie-Hellman assumption. In J. H. Cheon and T. Takagi, editors, *ASIACRYPT 2016, Part I*, volume 10031 of *LNCS*, pages 729–758. Springer, Heidelberg, Dec. 2016.

[105] M. Naor. On cryptographic assumptions and challenges (invited talk). In D. Boneh, editor, *CRYPTO 2003*, volume 2729 of *LNCS*, pages 96–109. Springer, Heidelberg, Aug. 2003.

[106] M. Naor and B. Pinkas. Efficient oblivious transfer protocols. In S. R. Kosaraju, editor, *12th SODA*, pages 448–457. ACM-SIAM, Jan. 2001.

[107] T. Okamoto, K. Pietrzak, B. Waters, and D. Wichs. New realizations of somewhere statistically binding hashing and positional accumulators. In T. Iwata and J. H. Cheon, editors, *ASIACRYPT 2015, Part I*, volume 9452 of *LNCS*, pages 121–145. Springer, Heidelberg, Nov. / Dec. 2015.

[108] B. Parno, J. Howell, C. Gentry, and M. Raykova. Pinocchio: Nearly Practical Verifiable Computation. In *IEEE SP 2013*, pages 238–252, Berkeley, CA, USA, May 19-22, 2013. IEEE Computer Society.

[109] B. Parno, J. Howell, C. Gentry, and M. Raykova. Pinocchio: Nearly practical verifiable computation. In *2013 IEEE Symposium on Security and Privacy*, pages 238–252. IEEE Computer Society Press, May 2013.

[110] T. P. Pedersen. Non-interactive and information-theoretic secure verifiable secret sharing. In J. Feigenbaum, editor, *CRYPTO'91*, volume 576 of *LNCS*, pages 129–140. Springer, Heidelberg, Aug. 1992.

[111] A. Poelstra. Mimblewimble, 2016. Available at `https://download.wpsoftware.net/bitcoin/wizardry/mimblewimble.pdf`.

[112] C. Ràfols. Stretching groth-sahai: NIZK proofs of partial satisfiability. In Y. Dodis and J. B. Nielsen, editors, *TCC 2015, Part II*, volume 9015 of *LNCS*, pages 247–276. Springer, Heidelberg, Mar. 2015.

[113] C. Ràfols and J. Silva. QA-NIZK arguments of same opening for bilateral commitments. In A. Nitaj and A. M. Youssef, editors, *AFRICACRYPT 20*, volume 12174 of *LNCS*, pages 3–23. Springer, Heidelberg, July 2020.

[114] A. Rial, M. Kohlweiss, and B. Preneel. Universally composable adaptive priced oblivious transfer. In H. Shacham and B. Waters, editors, *PAIRING 2009*, volume 5671 of *LNCS*, pages 231–247. Springer, Heidelberg, Aug. 2009.

[115] A. Sahai. Non-malleable non-interactive zero knowledge and adaptive chosen-ciphertext security. In *40th FOCS*, pages 543–553. IEEE Computer Society Press, Oct. 1999.

[116] K. Sako and J. Kilian. Receipt-free mix-type voting scheme - a practical solution to the implementation of a voting booth. In L. C. Guillou and J.-J. Quisquater, editors, *EUROCRYPT'95*, volume 921 of *LNCS*, pages 393–403. Springer, Heidelberg, May 1995.

[117] V. Shoup. Lower bounds for discrete logarithms and related problems. In W. Fumy, editor, *EUROCRYPT'97*, volume 1233 of *LNCS*, pages 256–266. Springer, Heidelberg, May 1997.

[118] J. H. Silverman. The arithmetic of elliptic curves. Springer, 1955.

[119] J. L. Villar. Optimal reductions of some decisional problems to the rank problem. In X. Wang and K. Sako, editors, *ASIACRYPT 2012*, volume 7658 of *LNCS*, pages 80–97. Springer, Heidelberg, Dec. 2012.