DISSERTATIONES
INFORMATICAE
UNIVERSITATIS
TARTUENSIS
17

BEHZAD ABDOLMALEKI

On Succinct Non-Interactive Zero-Knowledge Protocols Under Weaker Trust Assumptions

DISSERTATIONES INFORMATICAE UNIVERSITATIS TARTUENSIS

17

DISSERTATIONES INFORMATICAE UNIVERSITATIS TARTUENSIS
17

BEHZAD ABDOLMALEKI

On Succinct Non-Interactive Zero-Knowledge Protocols Under Weaker Trust Assumptions

UNIVERSITY OF TARTU
Press

Institute of Computer Science, Faculty of Science and Technology, University of Tartu, Estonia.

Dissertation has been accepted for the commencement of the degree of Doctor of Philosophy (PhD) in Computer Science on 16th of June, 2020 by the Council of the Institute of Computer Science, University of Tartu.

Supervisor

Helger Lipmaa
Simula UiB,
Bergen, Norway.
University of Tartu,
Tartu, Estonia.

Opponents

Carla Ràfols Salvador
Universitat Pompeu Fabra,
Barcelona, Spain.

Olivier Blazy
Université de Limoges,
Limoges, France.

The public defense will take place on 26th of August, 2020 at 14:15 in Zoom.

The publication of this dissertation was financed by the Institute of Computer Science, University of Tartu.

ISSN 2613-5906
ISBN 978-9949-03-375-1 (print)
ISBN 978-9949-03-376-8 (PDF)

University of Tartu Press
http://www.tyk.ee/

To my wife

ABSTRACT

Very fundamental primitives in cryptographic protocols are commitment schemes and zero-knowledge proofs that are frequently employed in real-world applications. A commitment scheme enables a committer to compute a commitment to a secret message, and later open it (i.e, expose the secret message) in a verifiable manner. Universally composable (UC) commitment schemes are of particular interest, as one can seamlessly combine them with other UC protocols and primitives while the entire protocol remains secure. Moreover, efficient UC-secure commitments significantly enable more efficient constructions of protocols for some applications like as two-party computation in garbled circuits, zero-knowledge proofs, and multiparty computation. However, uniting universal composability and commitment schemes often needs to scarify efficiency.

A zero-knowledge proof is a protocol between a prover and a verifier that allows the prover to convince the verifier of the validity of a statement without disclosing any more additional information. Zero-knowledge proofs and in particular non-interactive zero-knowledge proofs (NIZKs) are a critical primitive for building cryptographic protocols. They allow ensuring the correctness of some computations while valuing preserving the privacy of the user. NIZKs are getting increasingly used in real-world applications, with cryptocurrencies or more generally distributed ledger technologies being the prime examples. Thus it is desirable and crucial that NIZKs have fast verification and small proof sizes. Due to the specifics in this setting, succinct non-interactive zero-knowledge proofs (so-called zk-SNARKs) are of particular interest and research in this direction within academia and industry is currently erupting. Besides, recently there has been a research line of another type of NIZKs for a linear language called quasi adaptive NIZK (QA-NIZK). These constructions of NIZKs also provide succinctness (having fast verification and small proof sizes) and together with SNARKs perceiving are getting a lot of consideration in the real world cryptographic protocols.

A serious issue and the main drawback of the practical usage of NIZKs is the demand for a trusted setup for generating the common reference string (CRS). In this line there are two significant practical outlines to diminish the trust in the setup procedure: (i) using the new notion of security called *subversion zero-knowledge* initiated by Bellare et al. in [22], where the zero-knowledge property even remains when the CRS is generated maliciously, i.e., the CRS generator is subverted, and (ii) utilizing multi-party computation (MPC) to generate the CRS. In the latter case, the UC-security property is crucial, to securely compose the CRS generation protocol with the NIZK (zk-SNARK or QA-NIZK) in a black-box way (with warranting the security of the NIZK).

In this thesis, we investigate the aforementioned outlines and propose concrete constructions for them. We first investigate subversion zk-SNARKs (Sub-zk-SNARKs), in particular, constructing subversion of the most efficient SNARKs of [94]. Then we initiate the study of subversion QA-NIZK (Sub-QA-NIZK) and

construct subversion resistant version of the most efficient QA-NIZKs of [106]. For the second outline, first using hash proof systems or smooth projective hash functions (SPHFs), we introduce a new cryptographic primitive called publicly computable SPHFs (PC-SPHFs) and construct the currently most efficient non-interactive UC-secure commitment. Finally, we develop a new technique for constructing UC-secure commitments schemes that enables one to generate CRS of NIZKs by using MPC in a UC-secure manner.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

BPK	Bare Public Key Model
BDH-KE	Bilinear Diffie-Hellman Knowledge of Exponents
CS	Cramer-Shoup
CP	Commit-and-Prove
CRS	Common Reference String
DAA	Direct Anonymous Attestation
DDH	Decisional Diffie-Hellman
EPID	Enhanced Privacy ID
GBGM	Generic Bilinear Group Model
HAK	Hash-Algebraic Knowledge
HVZK	Honest-Verifier Zero-Knowledge
KERMDH	Kernel Matrix Diffie-Hellman
KoE	Knowledge of Exponent
KWKE	Kiltz-Wee Knowledge of Exponent
MDDH	Matrix Decisional Diffie-Hellman
MPC	Multi-Party Computation
NIZKs	Non-Interactive Zero-Knowledge proofs
PAKE	Password-Authenticated Key Exchange
PHFs	Projective Hash Functions
PC-SPHFs	Publicly Computable Smooth Projective Hash Functions
PPT	Probabilistic Polynomial Time
QAP	Quadratic Arithmetic Program
QA-NIZK	Quasi-Adaptive Non-Interactive Zero-Knowledge
RO	Random Oracle
RPK	Registered Public Key Model
Sub-SND	Subversion Soundness
Sub-PAR	Subversion Parameter
Sub-ZK	Subversion Zero-Knowledge
Sub-GBGM	Subversion Generic Bilinear Group Model
SCS	Short Cramer-Shoup
SPHFs	Smooth Projective Hash Functions
TPM	Trusted Platform Module
UC	Universal Composability
zk-SNARK	zero-knowledge Succinct Non-interactive ARguments of Knowledge

LIST OF ORIGINAL PUBLICATIONS

Publications included in the thesis

I Behzad Abdolmaleki, Karim Baghery, Helger Lipmaa, and Michal Zajac. *A subversion-resistant SNARK*. In ASIACRYPT 2017, volume 10626 of LNCS, pages 3-33, Springer, Heidelberg.

II Behzad Abdolmaleki, Karim Baghery, Helger Lipmaa, Janno Siim, and Michal Zajac. *DL-extractable UC-commitment schemes*. In ACNS 2019, volume 11464 of LNCS, pages 385-405. Springer, Heidelberg.

III Behzad Abdolmaleki, Hamidreza Khoshakhlagh, and Daniel Slamanig. *A Framework for UC-Secure Commitments from Publicly Computable Smooth Projective Hashing*. In IMACC 2019, volume 11929, pages 1-21, Springer, Cham.

IV Behzad Abdolmaleki, Helger Lipmaa, Janno Siim, and Michal Zajac. *On QA-NIZK in the BPK Model*. In PKC 2020, volume 12110-12111 of LNCS, Springer, Heidelberg.

Publications not included in the thesis

I Behzad Abdolmaleki, Karim Baghery, Helger Lipmaa, Janno Siim, and Michal Zajac. *UC-Secure CRS Generation for SNARKs*. In AFRICACRYPT 2019,volume 11627 of LNCS, pages 99-117, Springer, Cham.

II Behzad Abdolmaleki, Sebastian Ramacher, and Daniel Slamanig. *Lift-and-Shift: Obtaining Simulation Extractable Subversion and Updatable SNARKs Generically*. In ACM CCS 2020 (to appear).

III Behzad Abdolmaleki and Daniel Slamanig. *Subversion Simulation-Sound Quasi-Adaptive NIZK Proofs and Applications to Modular zk-SNARKs*. In IACR Cryptology ePrint Archive 2020.

IV Behzad Abdolmaleki, Hamidreza Khoshakhlagh, and Helger Lipmaa. *Smooth Zero-Knowledge Hash Functions*. In IACR Cryptology ePrint Archive 2020.

Other published work of the author

I Behzad Abdolmaleki, Karim Baghery, Shahram Khazaei, and Mohammad Reza Aref. *Game-Based Privacy Analysis of RFID Security Schemes for Confident Authentication in IoT*. Journal of Wireless Personal Communications 2017, Volume 95, Issue 4, pp 5057-5080, Springer US.

II Karim Baghery and Behzad Abdolmaleki. *An AKARI-based Secure Communication Scheme for EPC Tags*. Advances in Wireless and Optical Communications (RTUWO), 2017, PP 208-213, IEEE.

III Karim Baghery, Behzad Abdolmaleki, Shahram Khazaei, and Mohammad Reza Aref. *Breaking anonymity of some recent lightweight RFID authentication protocols*. Journal of Wireless Networks 2019, Volume 25, Issue 3, pp 1235–1252, Springer US.

1. INTRODUCTION

The main goal of cryptography for a long time was to allow two parties who agreed on a secret value (i.e., secret key) to communicate confidentially by using encryption and commitment schemes. Since the introduction of key exchange [69], public-key encryption and digital signatures [128], the extent of cryptography has vastly been broadened. The public key encryption and key exchange enable a sender to make a confidential communication, without knowing the receiver. Digital signatures guarantee that a message belongs to a valid entity or a valid user. While commitment schemes ensure that the committed message will not change during the protocol.

These days, cryptography is prevalent in our daily life, even sometimes in hidden forms such as access badges emails, credit cards and smartphones which all use cryptographic protocols. Also, the field of cryptography has been broadened by containing more practical primitives and cryptographic protocols such as zero-knowledge proofs, introduced by Goldwasser et al. [86]. A zero-knowledge proof is a protocol between a prover and a verifier that allows the prover to convince the verifier of the validity of a statement without revealing any more information. Zero-knowledge proofs and in particular non-interactive zero-knowledge proofs (NIZKs) are an important primitive for cryptographic protocols, by which you can ensure both the correctness of computation and the user privacy at the same time. Practically they were first employed in electronic voting systems[1]. Later they played a central role in both the practice and theory of cryptography and increasingly found their way into real-world applications.[2,3,4] Some important applications are anonymous credentials [21,46–49,55,78], electronic voting [63,93,129], and group signatures [18, 36, 37, 56, 66–68], including widely deployed schemes such as direct anonymous attestation (DAA) [40,45] employed in Intel's Enhanced Privacy ID (EPID) [41] or the Trusted Platform Module (TPM), as well as many other applications that need integrity and balancing privacy (cf. [75]). In addition they are a core of verifiable computation [38, 81, 82, 126] and increasingly more popular in privacy-preserving cryptocurrencies [25, 57], self-sovereign identity systems [123], and smart contracts [108]. Latter arguably, nowadays represent the most popular real-world applications of zero-knowledge, where it uses in systems such as Zcash and Ethereum. Due to aforementioned applications, a long line of research [82, 92, 94, 97, 98, 100, 102, 105, 106, 113] has led to efficient zero-knowledge Succinct Non-interactive ARguments of Knowledge (zk-SNARKs)

[1] Helios Voting cf. `https://vote.heliosvoting.org`.

[2] ZKProof (`https://zkproof.org/`) being the most notable industry and academic initiative towards a common framework and standards in the zero-knowledge area has been founded in 2018.

[3] Zero-knowledge proofs are *on the rise* in Gartners' Hype Cycle for Privacy 2019, cf. `https://www.gartner.com/en/documents/3947373/hype-cycle-for-privacy-2019`.

[4] MIT technology review mentioned zk-SNARKS as one of the "10 Breakthrough Technologies of 2018" cf. `https://www.technologyreview.com/lists/technologies/2018/`.

and Quasi-Adaptive Non-Interactive Zero-Knowledge arguments (QA-NIZKs) in the common reference string (CRS) model. QA-NIZK constructions are a relaxation of NIZK proofs while the CRS depends on the specific language parameter for which proofs have to be generated [17, 100, 102, 106, 110, 111].

So far there has been a tight relation between zero-knowledge proofs and commitment schemes. In most of the zero-knowledge protocols, the use of commitment schemes is crucial in a way that by using commitment schemes one can make a claim (statement) and later use the zero-knowledge proof techniques to prove the claim. In real-world applications of the cryptographic protocols i.e, commitment schemes and zero-knowledge proofs, they have to satisfy a strong property called Universal Composability (UC) which were introduced by [52]. Informally, UC-secure protocols ensure the security even if executed with other instances of the same protocol, or when composed with other protocols. Due that there has been several efforts [7, 26, 109] to construct UC-secure NIZKs (UC-secure SNARKs) and [4–6, 27, 74, 100] to achieve UC-secure commitment schemes where zero-knowledge proofs are at the core of some of them. However, adding universal composability to such schemes often needs to scarify the efficiency, therefore having efficient UC-secure protocols is still an important need in practice and for lots of applications.

Another important issue for the practical use of NIZKs is the requirement of a trusted setup for generating CRS. Recently there have some efforts to minimize trust in such systems. To this aim, Bellare *et al.* in [22] initiated different notions of resistance to CRS subversion and their achievability. They introduced subversion soundness (Sub-SND), meaning that no adversary can create a CRS (possibly malicious) along with valid proof for a false statement. They also introduced a subversion-resistant analogue for zero-knowledge, meaning that the zero-knowledge property even remains when the CRS is generated maliciously, i.e., the CRS generator is subverted. In a different line of research [7, 26] has employed multi-party computation (MPC) techniques to decrease the trust of the CRS generation's procedure.

1.1. Scope and Claim of This Thesis

Non-interactive zero-knowledge proofs in the CRS model are increasingly getting attention in real-world applications and distributed ledger technologies such as cryptocurrencies. Thus such NIZK should have fast verification and small proof sizes. These desired properties are provided by QA-NIZKs and zk-SNARKs.

As we noted before the important issues for the practical use of NIZKs are (i) the need for a trusted setup for generating CRS, and (ii) adding universal composability to guarantee that security remains if the protocol is concurrently run with other protocols or even copies of itself. In this thesis, we mainly focus on these subjects and present some concrete constructions that provide the aforementioned properties. More precisely, in the line of diminishing the trust, we

show how one can construct a subversion zero-knowledge version of the most efficient[5] zk SNARK [94] and the most efficient QA-NIZK [106] constructions where the zero-knowledge property of the constructions remains when the CRS is subverted. In terms of universal composability issue, by using smooth projective hash functions (SPHFs), we first introduce a new cryptographic primitive called publicly computable SPHFs (PC-SPHFs) and construct the currently most efficient non-interactive UC-secure commitment. Then we develop a new technique for constructing UC-secure commitments and propose a new scheme called DL-extractable commitment that allows one to generate CRS of zk-SNARKs in a UC-secure manner.

Zero-Knowledge proofs. Zero-knowledge proofs were introduced by Goldwasser *et al.* [86] which is a cryptographic protocol between the two parties called the verifier and the prover. Roughly speaking, in such system the prover wants to convince the verifier of his claim (more formally the membership of a statement in any language $\mathscr{L}$) without revealing any more information of her witness w warranting language membership of word x (called zero-knowledge property). Besides that, such a system needs to guarantee that the prover can not convince the verifier with provided proofs for words outside of the language, this property is called soundness. While zero-knowledge proofs, generally, may need some rounds of interaction, a variant highly proper with practical applications is non-interactive zero-knowledge (NIZK) proofs [33]. They need only a single round in a way that the prover provides proofs which then can be verified by only one round.

There are two models used to obtain NIZK from interactive zero-knowledge proofs: (i) the random oracle (RO) model, and (ii) the common reference string (CRS) model. In this thesis, we are mainly interested in the CRS model, in which both the prover and the verifier have access to a structured random string generated by some trusted setup. In practice, such a CRS can be generated by multi-party computation.

Trust in the CRS. An important aspect of practical applications of NIZKs, especially for zk-SNARKs is the challenge of the generation of CRS or structured random string [33]. While the CRS model is widely accepted, one has to guarantee that the CRS has been generated honestly, meaning that no one knows the associated trapdoor to the CRS which allows breaking soundness or zero-knowledge property. In theory, it is easily solved by assuming some trusted party that provides the CRS, but implementing such a party is hard in the real-world.

For NIZK proofs, Bellare *et al.* [22] tried to tackle this problem by studying the security one still can achieve when the CRS is subverted. They proved several

[5]The efficiency respect to the size of the proof and the verification compelexity.

positive and negative results. More precisely, they proved that it is impossible to achieve simultaneously subversion soundness and zero-knowledge (even non-subversion). While subversion zero-knowledge (Sub-ZK) can be achieved. We consider this notion for SNARKs and QA-NIZKs in this thesis and published the results in [9, 11] [6] which can be used in practical applications in cryptocurrencies [51, 77] [7]. In the line of diminishing the trust, some systems such as Zcash and Ethereum employed an alternative route to implementing the CRS generation and used a secure multi-party computation (MPC) [26, 39, 126] for zk-SNARKs instead of building them on top of subversion-resistant zk-SNARKs.

Very recently, Groth *et al.* [96] introduced the novel notion of *updatable zk-SNARKs* in which every party can update the CRS of SNARK to diminish the trust in the generation of CRS. While there is a way to check the correctness of an update. Here, zero-knowledge holds in the presence of a malicious CRS generator and the verifier can trust the CRS (soundness holds) as long as one operation, either the creation of the CRS or one update, has been performed honestly. Updatable zk-SNARKs also are crucially Sub-zk-SNARKs. This technique is promising, but it is still inefficient for practical applications and currently, there are some efforts [58, 89, 107, 117] to make it usable in real-world applications.

SPHF. Smooth projective hash functions (SPHFs, [60]) for a language $\mathscr{L}_{\mathtt{lpar}}$, parameterized by a language parameter $\mathtt{lpar}$, are cryptographic primitives with the following properties. Given a word x, one can compute a hash of it in two different ways: either (i) using a projection key hp (an analogue of a public key), a word x, and a witness w, as $\mathsf{pH} \leftarrow \mathsf{projhash}(\mathtt{lpar}; \mathsf{hp}, x, w)$, or (ii) using a hashing key hk (an analogue of a secret key) and a word x, as $\mathsf{H} \leftarrow \mathsf{hash}(\mathtt{lpar}; \mathsf{hk}, x)$. where the algorithms $\mathsf{projhash}$ and hash are respectively the projection hash and the hash generators. If $x \in \mathscr{L}_{\mathtt{lpar}}$ and w is a correct witness of this fact, then the *correctness* property guarantees that the two ways of computing the hash result in the same value, $\mathsf{pH} = \mathsf{H}$. If $x \notin \mathscr{L}$, then the *smoothness* property guarantees that, knowing hp but not hk, one cannot distinguish H from random. SPHFs are useful in many different applications, starting from constructing IND-CCA2 secure cryptosystems [60] and password-authenticated key exchange (PAKE) [83], and ending with honest-verifier zero knowledge (HVZK, [30]) and NIZK [2]. Later in Section 4.1 of this thesis we introduce a new primitive based on SPHFs, called publicly computable SPHFs (PC-SPHFs) and use it to construct a generic framework for non-interactive UC-secure commitment schemes (this result is published in [10]).

UC-secure commitment. Another important aspect for practical applications of

[6] Later we improved the security of both the constructions by adding a strong security property is called Simulation Extractability (SE) in [13–15] and they are not included in this thesis.

[7] Also Fuchsbauer [76] implemented the notation with a different technique for zk-SNARKS.

cryptographic protocols such as NIZKs (i.e, for implementing its CRS) and commitment schemes is achieving universal composability. This is a strong property and was introduced by [52]. Informally, UC-secure protocols ensure the security even if executed with other instances of this same protocol, or when composed with other protocols. A commitment scheme allows a committer C to transmit an analogue of a sealed envelope of her message m to a receiver R. Later the committer C can open her committed message with m and some additional information. The receiver R verifies whether m is correctly enveloped. Such systems should guarantee that C cannot change later the enveloped message m to some $m' \neq m$ (binding property); while the receiver R must not learn any information of the enveloped message m (hiding property).

In order to attach the UC property to commitment schemes, one needs to provide the following additional properties:(i) extractability, states that given a trapdoor, one (i.e, the simulator Sim in UC proof) can recover the committed value m, and (ii) equivocability means that given a trapdoor, one (i.e, the simulator Sim in UC proof) can open a commitment to any message $m' \neq m$. In Chapter 4 of this thesis we investigate constructing efficient UC-secure commitments; one with using PC-SPHF which is currently the most efficient non-interactive UC-secure commitment [10], and another one [6] with a new technique that allows one to generate CRS of zk-SNARKs in a UC secure manner. Later Abdolmaleki *et al.* [7] used the latter construction to generate the CRS of SNARK [94] with multi-party computation in UC-secure manner.

1.2. Thesis Outline and Contributions

In this part, we outline the contents of chapters in the thesis, and briefly explain the main contributions of the author towards the co-authored papers.

Chapter 2 provides a quick overview of the preliminaries, basic assumptions and the primitives used throughout this thesis. In particular, in this chapter, we introduce zk-SNARKs and QA-NIZK constructions and their security requirements which we use throughout this thesis to construct subversion zk-SNARKs and QA-NIZKs. Also, we mention the various assumptions, encryption schemes and SPHFs primitives used in constructing UC-secure commitment schemes in the subsequent chapters.

Chapter 3 introduces the notation of NIZKs (focusing on zk-SNARKs and QA-NIZK) with a subverted setup. Mainly it describes the challenges and the ideas of building such constructions and shows how one can construct subversion zk-SNARKS and QA-NIZK. This chapter refers to the following two papers included in this thesis.

I Behzad Abdolmaleki, Karim Baghery, Helger Lipmaa, and Michal Zajac. *A*

subversion-resistant SNARK. In ASIACRYPT 2017, volume 10626 of LNCS, pages 3-33, Springer, Heidelberg.
This paper investigates the zk-SNARKs when the CRS is subverted. In ASIACRYPT 2016, Bellare *et al.* [22] studied the security, one still can achieve when the CRS is subverted (the CRS is generated by a malicious party) and proved several positive and negative results. On the positive side, they proved that it is possible to achieve simultaneously subversion zero-knowledge and soundness in NIZK constructions. This paper studies the subversion zero-knowledge security for Groth's zk-SNARK for CIRCUIT-SAT from EUROCRYPT 2016 and implements the subversion security definitions of [22]. Then it shows how one can achieve Groth's zk-SNARK [94] in subverted setup (when the setup is maliciously created) without effecting the complexity of the verification and the proof's size of the construction. The author's contributions consist of a complexity analysis of our Sub-zk-SANRK construction, a comparison between the proposed construction with existing works, and finally providing an improvement on the efficiency of the CRS verification algorithm.

II Behzad Abdolmaleki, Helger Lipmaa, Janno Siim, and Michal Zajac. *On QA-NIZK in the BPK Model*. In PKC 2020, volume 12110-12111 of LNCS, Springer, Heidelberg..
This paper addresses a conceptually important observation that subversion zero-knowledge in the CRS model is equal to no-auxiliary-string non-black-box zero-knowledge (called nonuniform zero-knowledge) in the bare public key (BPK) model. This important nexus was missed in the previous work on subversion zero-knowledge and makes it more accessible to construct and analyze those systems including both SNARKs and QA-NIZKs. Also, it defines a new notion of subversion security for QA-NIZKs. Finally, this paper shows how one can construct a subversion version of the most efficient known QA-NIZK for linear subspaces presented by Kiltz and Wee. In the end, it proves that the scheme is nonuniform zero knowledge in the BPK model under some novel non-falsifiable assumptions. The author's contributions include improving the public key verification algorithm, parts of security definitions of QA-NIZKs in BPK model, investigating the applications of the new construction and parts of the security proofs.

Chapter 4 mainly addresses how to construct UC-secure commitment schemes with the focus on (i) the efficiency of the scheme and (ii) compatibility with the UC-secure MPC. The latter property plays a crucial role to guarantee the UC security of the CRS generation of zk-SNARKs. This chapter first introduces a new cryptographic primitive called PC-SPHFs and constructs a generic framework for non-interactive UC-secure commitment. Then it develops a new technique for constructing UC-secure commitments schemes that allow one to generate CRS

of zk-SNARKs in a UC secure manner. This chapter refers to the following two papers included in this thesis.

I Behzad Abdolmaleki, Hamidreza Khoshakhlagh, and Daniel Slamanig. *A Framework for UC-Secure Commitments from Publicly Computable Smooth Projective Hashing*. In IMACC 2019, volume 11929, pages 1-21, Springer, Cham. This paper first revisits the notion of SPHFs and introduces a new property (a third mode of constructing the hashing) that enables to compute the hash value of an SPHF without knowing neither the hashing key nor the witness, but some auxiliary information. This new type of SPHFs is called PC-SPHFs. Later it shows how this new primitive leads generically to UC-secure commitment schemes (secure against adaptive adversaries, assuming erasures). Finally, it shows that instantiating the PC-SPHF with labeled Cramer-Shoup encryption [60] returns the currently most efficient non-interactive UC-secure commitment. The author constructed the new primitive, PC-SPHF and proposed the new UC-secure framework for commitment schemes together with their security proofs. The author also constructed the efficient non-interactive UC-secure commitment scheme.

II Behzad Abdolmaleki, Karim Baghery, Helger Lipmaa, Janno Siim, and Michal Zajac. *DL-extractable UC-commitment schemes*. In ACNS 2019, volume 11464 of LNCS, pages 385-405. Springer, Heidelberg.
This paper proposes a new UC functionality and constructs a corresponding UC-secure commitment (DL-extractable UC-secure commitment scheme) that fits the following setting: when the secrecy of a committed value x is important and revealing x enables to break the privacy of the scheme; while in the UC setting the simulator needs to know x to simulate the corrupted committer C. More precisely DL-extractable UC-secure commitment schemes allow the committer C to open a commitment to a group element g^x while in the UC security proof the simulator can extract the integer x. The author's contributions include parts of designing the new UC functionality for such commitment schemes and the UC-security proofs of the DL-extractable UC-secure commitment scheme.

Chapter 5 summarizes the author's works on the important issues for the practical use of NIZKs that (i) need a trusted setup for generating CRS, and (ii) adds a universal composability to guarantee that security remains if the protocol is concurrently run with other protocols or even copies of itself. This chapter also points out the open problems left in the research line of the thesis.

2. PRELIMINARIES

In this chapter, we provide the preliminaries, basic assumptions and the primitives used throughout this thesis. First, we recall some standard computational and mathematical concepts. Then we recall some well-known computational assumptions and also provide most of the cryptographic tools and primitives used in this work, including commitment schemes, encryption schemes, NIZKs, SNARKs, QA-NIZKs, and SPHFs.

2.1. Notation

We define $\mathbb{N}$ to be the set of positive integers and $\mathbb{Z}$ be the set of integers. We denote the security parameter by λ. A function $f : \mathbb{N} \to [0,1]$ is called negligible if $f(\lambda) = O(\lambda^{-c})$ for every constant $c \in \mathbb{N}$. We denote this by $f \approx_\lambda 0$. We define by $\mathsf{negl}(\lambda)$ a negligible function of λ. We write $a \approx_\lambda b$ if $|a-b| \leq \mathsf{negl}(\lambda)$.

Integers modulo p are denoted by $\mathbb{Z}_p$ and the ring of polynomials with variables $X_1, \ldots, X_n$ over $\mathbb{Z}_p$ by $\mathbb{Z}_p[X_1, \ldots, X_n]$. Vectors $a := (a_i)_{i=1}^n \in \mathbb{Z}_p^n$ are column vectors by default and are usually denoted by bold lower-case letters. Length n vector of ones is denoted by 1_n and vector of zeros by 0_n. Matrices $A \in \mathbb{Z}_p^{n \times m}$ are denoted bold upper-case letters and i-th row vector of A is denoted by A_i. Identity matrix is denoted by $I_n \in \mathbb{Z}_p^{n \times n}$. Set $\{1, \ldots, n\}$ is denoted by $[n]$.

Sampling x from a distribution $\mathscr{D}$ is written as $x \leftarrow_\$ \mathscr{D}$. If $\mathscr{D}$ is a set, then $x \leftarrow_\$ \mathscr{D}$ means uniform sampling. We often express probabilities in the form $\Pr[x \leftarrow_\$ \mathscr{D} : \mathsf{Pred}(x)]$ where $\mathscr{D}$ is a probability distribution and Pred some predicate. Probabilistic polynomial time is abbreviated by PPT. Random tape RND is a Turing machine tape which stores infinitely many bits, each bit is selected independently and uniformly at random from $\{0,1\}$ before the execution of the algorithm. Let $\mathsf{RND}_\lambda(\mathsf{A})$ denote the random tape of an algorithm A (assuming the given value of λ)[1]. We denote sampling sufficiently long randomness by $r \leftarrow_\$ \mathsf{RND}_\lambda(\mathsf{A})$ and A outputting y on input x and random coins r is denoted by $y \leftarrow \mathsf{A}(x;r)$. For the sake of simplicity, we may often leave out the random coins and just write $y \leftarrow \mathsf{A}(x)$. We assume that algorithms are stateful, e.g., we may first execute A on input x and then again on another input y without A losing its state from the first execution.

Pairings. Let Pgen be a PPT algorithm (a bilinear group generator) that on input 1^λ outputs a tuple $\mathsf{p} = (p, \hat{e}, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, g_1, g_2)$ where p is a prime of length $\Theta(\lambda)$, $\mathbb{G}_1$, $\mathbb{G}_2$, $\mathbb{G}_T$ are additive groups of order p with an efficient operation, g_1 and g_2 are respectively the generators of $\mathbb{G}_1$ and $\mathbb{G}_2$. The function $\hat{e}$ is a map $\mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ with the following properties: (i) Bilinearity: $\hat{e}(g_1^x, g_2^y) = g_T^{xy}$ where $g_T := \hat{e}(g_1, g_2)$ for any $x, y \in \mathbb{Z}_p$, (ii) Non-degeneracy: g_T as defined above is a non-zero element and thus also a generator of the target group $\mathbb{G}_T$, and (iii) Ef-

[1] A PPT A is able to read only polynomially many (in security parameter λ) of the random tape.

ficiency: $\hat{e}$ can be efficiently computed. Pairings are usually divided into three types: Type 1 pairings have $\mathbb{G}_1 = \mathbb{G}_2$, Type 2 pairings have $\mathbb{G}_1 \neq \mathbb{G}_2$, and Type 3 pairings have $\mathbb{G}_1 \neq \mathbb{G}_2$ even such that there is no efficiently computable isomorphism between $\mathbb{G}_1$ and $\mathbb{G}_2$. Type 3 pairings are currently the most widely used due to their efficiency and also in this thesis, we will use Type 3 pairings. In this thesis we use them as a black box and for state-of-the-art pairings, we refer the reader to [20].

We use the bracket notation of [71], that is, we write $[a]_\iota$ to denote g_ι^a where g_ι is a fixed generator of $\mathbb{G}_\iota$. We denote $\hat{e}(g_1^a, g_2^b) = \hat{e}([a]_1, [b]_2)$ as $[a]_1[b]_2$. Thus, $[a]_1[b]_2 = [ab]_T$. We denote $s[a]_\iota = [sa]_\iota$ for $s \in \mathbb{Z}_p$. We freely use the bracket notation together with matrix notation, for example, if $XY = Z$ then $[X]_1[Y]_2 = [Z]_T$ where X, Y, and Z are matrices.

Lagrange polynomials. Let $\omega_1, \ldots, \omega_{n+1}$ be unique elements in $\mathbb{Z}_p$. Lagrange polynomial $\ell_i(X)$ is the unique n-th degree polynomial such that $\ell_i(\omega_i) = 1$ and $\ell_i(\omega_j) = 0$ for $j \neq i$. More explicitly we may express $\ell_i(X) = \prod_{j \neq i}^{n+1} \frac{X - \omega_i}{\omega_j - \omega_i}$. Lagrange polynomials $\ell_1(X), \ldots, \ell_{n+1}(X)$ form a basis for all polynomials in $\mathbb{Z}_p[X]$ which are at most degree n. Lagrange polynomials are often used for interpolation. Namely, if $f \in \mathbb{Z}_p[X]$ has a degree at most n and $f(\omega_i) = a_i$ for $i \in [1..n+1]$, then $f(X) = \sum_{i=1}^{n+1} a_i \ell_i(X)$.

2.2. Computational Assumptions

In cryptography to prove security of a primitive we usually apply reduction technique which allows reducing the problem to a previously known hard problem. Then these two problems are called polynomial-time equivalent if the reduction is provided in polynomial time. Besides, we define security for some particular fixed size input. This shows non-uniformity, as an adversary $\mathscr{A}$ maybe given the input as an auxiliary string to solve some computational problems of the same size. In contrast with complexity theory [2], in cryptography, we usually use the hardness of computational assumptions in the average case. In the cryptographic protocols, for the security against non-uniform PPT adversary, we use the fact that if one breaks the protocol, then she can solve a hard problem in the average case. Note that we assume a non-uniform PPT adversary $\mathscr{A}$ is successful if, with non-negligible probability in the security parameter λ, she returns a solution for the computational problem.

Let GPgen be a PPT group generator that on input 1^λ outputs $\mathsf{gp} = (p, \mathbb{G}, g)$ where $\mathbb{G}$ is an additive group of prime order $p = \Theta(\lambda)$ and g is a random generator of the group. Let Pgen be a PPT algorithm (A bilinear group generator) that on input 1^λ outputs a tuple $\mathsf{p} = (p, \hat{e}, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, g_1, g_2)$ where $\mathbb{G}_1$, $\mathbb{G}_2$, $\mathbb{G}_T$ are

[2] In complexity theory, most problems are based on the hardness in the worst-case.

additive groups of order p with an efficient operation, g_1 and g_2 are respectively the generators of $\mathbb{G}_1$ and $\mathbb{G}_2$. The function $\hat{e}$ is a map $\mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$. We still use the bracket notation to represent group elements. We recall the computational assumptions that are used in this thesis.

Definition 1 (Decisional Diffie-Hellman (DDH) Assumption [34])**.** . *Let* $\iota \in \{1,2\}$. DDH *holds relative to* GPgen*, for every PPT adversary* $\mathscr{A}$*,* $|\mathsf{Exp}^{\mathsf{DDH}}_{\mathscr{A}}(\mathsf{gp}) - 1/2| \approx_\lambda 0$*, where* $\mathsf{Exp}^{\mathsf{DDH}}_{\mathscr{A}}(\mathsf{gp}) :=$

$$\Pr\left[\begin{array}{l} \mathsf{gp} \leftarrow_\$ \mathsf{GPgen}(1^\lambda); u,v,w \leftarrow_\$ \mathbb{Z}_p; b \leftarrow_\$ \{0,1\}; \\ b^* \leftarrow \mathscr{A}(\mathsf{gp}, [u]_\iota, [v]_\iota, [b\cdot uv + (1-b)w]_\iota) \end{array} : \; b = b^* \right],$$

Definition 2 (Hard-Subset-Membership Languages)**.** *Let* $\mathscr{L}_{\mathtt{lpar}}$ *be a language parameterized with some language parameters* lpar*. Let* $\mathscr{R}_{\mathtt{lpar}}$ *be the corresponding binary relation of all pairs* (x,w) *where x that we call a statement is in* $\mathscr{L}_{\mathtt{lpar}}$ *and w is any of the efficiently verifiable witnesses for x. Let* $\mathscr{X}_{\mathtt{lpar}}$ *be the underlying domain of the language* $\mathscr{L}_{\mathtt{lpar}}$ *(e.g., a group). A hard-subset-membership language family* $(\mathscr{L}_{\mathtt{lpar}} \subseteq \mathscr{X}_{\mathtt{lpar}})_{\mathtt{lpar}}$ *is a language family and satisfying the following properties. Here,* $\mathsf{Pgen.lpar}(\mathsf{p})$ *is an efficient algorithm that — given system parameters* p *— outputs the language parameter* lpar *together with a language trapdoor* td *(a secret key of the language parameter* lpar*).*

$\mathscr{R}$-sampleability. *There exists a PPT algorithm that takes as input a parameter* lpar *and randomly samples words x from* $\mathscr{L}_{\mathtt{lpar}}$ *together with a valid witness w, according to some distribution (which might not be uniform). We write* $(x,w) \leftarrow_\$ \mathscr{R}_{\mathtt{lpar}}$ *to say that x and w are sampled by this algorithm. We write* $x \leftarrow_\$ \mathscr{L}_{\mathtt{lpar}}$ *when w is not important. We assume that for any* lpar *and any* (x,w) *sampled by this algorithm,* $(x,w) \in \mathscr{R}_{\mathtt{lpar}}$*.*

$(\mathscr{X} \setminus \mathscr{L})$-sampleability. *There exists a PPT algorithm which takes as input a parameter* lpar *and randomly samples words x from* $\mathscr{X}_{\mathtt{lpar}} \setminus \mathscr{L}_{\mathtt{lpar}}$ *according to some distribution (which might not be uniform). We write* $x \leftarrow_\$ \mathscr{X}_{\mathtt{lpar}} \setminus \mathscr{L}_{\mathtt{lpar}}$ *to say that x is sampled by this algorithm. We assume that for any* lpar *and any x sampled by this algorithm,* $x \in \mathscr{X}_{\mathtt{lpar}} \setminus \mathscr{L}_{\mathtt{lpar}}$*.*

Hard-subset-membership. *Randomly sampled words x from* $\mathscr{L}_{\mathtt{lpar}}$ *and from* $\mathscr{X}_{\mathtt{lpar}} \setminus \mathscr{L}_{\mathtt{lpar}}$ *(by the two previous algorithms respectively) are computationally indistinguishable, without knowing* td*. Formally, for a language* $\mathscr{L}$ *and adversary* $\mathscr{A}$*, let* $\mathsf{Adv}^{\mathrm{subset}}_{\mathscr{L},\mathscr{A}}(\lambda)$ *be the advantage* $\mathscr{A}$ *has in distinguishing* $x_0 \in \mathscr{L}$ *and* $x_1 \in \mathscr{X} \setminus \mathscr{L}$*:* $\mathsf{Adv}^{\mathrm{subset}}_{\mathscr{L},\mathscr{A}}(\lambda) = 2 \cdot |\varepsilon_{\mathscr{L}} - 1/2|$*, where*

$$\varepsilon_{\mathscr{L}} := \Pr\left[\begin{array}{l} \mathsf{p} \leftarrow_\$ \mathsf{Pgen}(1^\lambda); (\mathtt{lpar}, \mathsf{td}) \leftarrow_\$ \mathsf{Pgen.lpar}(\mathsf{p}); \\ b \leftarrow_\$ \{0,1\}; x_0 \leftarrow_\$ \mathscr{L}; x_1 \leftarrow_\$ \mathscr{X} \setminus \mathscr{L}; b' \leftarrow \mathscr{A}(\mathtt{lpar}; x_b) : b = b' \end{array}\right].$$

The language is hard-subset-membership, if this advantage is negligible, for any polynomial-time adversary $\mathscr{A}$*.*

In what follows, let $\mathscr{D}_k$ be taken as the uniform distribution over a set of all matrices in $\mathbb{Z}_p^{(k+1)\times k}$. So $A \leftarrow_\$ \mathscr{D}_k$ means that each $a_{ij} \leftarrow_\$ \mathbb{Z}_p^*$, which in [71] is denoted as $\mathscr{U}_k$, i.e., $\mathscr{D}_k := \mathscr{U}_k$.

Thus $A \leftarrow_\$ \mathscr{D}_k$:

$$\mathscr{U}_k\colon A = \begin{pmatrix} a_{11} & \dots & a_{1k} \\ \dots & \dots & \dots \\ a_{k1} & \dots & a_{kk} \\ a_{(k+1)1} & \dots & a_{(k+1)k} \end{pmatrix}$$

Definition 3 ($\mathscr{D}_k$-Matrix Diffie-Hellman ($\mathscr{D}_k$-MDDH) Assumption [122])**.** *The* $\mathscr{D}_k$-MDDH *assumption for* $\iota \in \{1,2\}$ *holds relative to* Pgen*, if for any PPT adversary* $\mathscr{A}$*,* $|\mathsf{Exp}^{\mathsf{MDDH}}_{\mathscr{A}}(\mathsf{p}) - 1/2| \approx_\lambda 0$*, where* $\mathsf{Exp}^{\mathsf{MDDH}}_{\mathscr{A}}(\mathsf{p}) :=$

$$\Pr\left[\begin{array}{l} \mathsf{p} \leftarrow_\$ \mathsf{Pgen}(1^\lambda); A \leftarrow_\$ \mathscr{D}_k; v \leftarrow_\$ \mathbb{Z}_p^k; \\ u \leftarrow_\$ \mathbb{Z}_p^{k+1}; b \leftarrow_\$ \{0,1\}; \\ b^* \leftarrow \mathscr{A}(\mathsf{p}, [A]_\iota, [u]_\iota, [b\cdot Av + (1-b)\cdot u]_\iota) \end{array} : \; b = b^* \right].$$

Definition 4 ($\mathscr{D}_k$-KerMDH Assumption [122])**.** *The* $\mathscr{D}_k$-KerMDH *assumption for* $\iota \in \{1,2\}$ *holds relative to* Pgen*, if for any PPT* $\mathscr{A}$*,*

$$\Pr\left[\begin{array}{l} \mathsf{p} \leftarrow \mathsf{Pgen}(1^\lambda); A \leftarrow_\$ \mathscr{D}_k; [s]_{3-\iota} \leftarrow \mathscr{A}(\mathsf{p}, [A]_\iota) : \\ s \neq 0 \wedge A^\top s = 0_k \end{array}\right] \approx_\lambda 0 \ .$$

Note that as it is shown in [122], if $\mathscr{D}_k$-MDDH holds then $\mathscr{D}_k$-KerMDH holds.

Definition 5 ($\mathscr{D}_k$-SKerMDH Assumption [87])**.** *The* $\mathscr{D}_k$-SKerMDH *assumption holds relative to* Pgen*, if for any PPT* $\mathscr{A}$*,*

$$\Pr\left[\begin{array}{l} \mathsf{p} \leftarrow \mathsf{Pgen}(1^\lambda); A \leftarrow_\$ \mathscr{D}_k; ([s_1]_1, [s_2]_2) \leftarrow \mathscr{A}(\mathsf{p}, [A]_1, [A]_2) : \\ s_1 - s_2 \neq 0 \wedge A^\top (s_1 - s_2) = 0_k \end{array}\right] \approx_\lambda 0 \ .$$

2.3. Knowledge Assumptions

In knowledge assumptions, we assume that if one (i.e, an adversary $\mathscr{A}$) can compute some value together with accompanying values which are called knowledge components, then she knows essentially how the values have been computed. In such definitions, we usually use an algorithm $\mathsf{Ext}_{\mathscr{A}}$ which extracts the knowledge components (the value) with polynomial overhead. Abe and Fehr in [16] introduced the notation $(z,y) \leftarrow (\mathscr{A}||\mathsf{Ext}_{\mathscr{A}})(x,r)$ meaning that if the algorithm $\mathscr{A}$ with input (x,r) outputs a tuple of values z, then the algorithm $\mathsf{Ext}_{\mathscr{A}}$ with the same input (x,r) outputs a tuple of values y. [3] In this thesis, we utilize some variants of the following knowledge assumptions.

[3] Where usually r is the random coins of $\mathscr{A}$.

Definition 6 (KoE Assumption [61])**.** *We say that* Pgen *is* Knowledge of Exponent (KoE) secure *if for any* $\mathsf{p} \leftarrow \mathsf{Pgen}(1^\lambda)$*,* $\iota \in \{1,2,T\}$*, and PPT adversary* $\mathscr{A}$ *there exists a PPT extractor* $\mathsf{Ext}_{\mathscr{A}}^{\mathsf{KoE}}$*, such that*

$$\Pr\left[\begin{array}{l} r \leftarrow_{\$} \mathsf{RND}_\lambda(\mathscr{A}), x \leftarrow_{\$} \mathbb{Z}_p; ([\alpha_1]_\iota, [\alpha_2]_\iota || a) \leftarrow (\mathscr{A} || \mathsf{Ext}_{\mathscr{A}}^{\mathsf{KoE}})(\mathsf{p}, [x]_\iota; r) : \\ [\alpha_1]_\iota = x[\alpha_2]_\iota \wedge [\alpha_2]_\iota \neq [a]_\iota \end{array}\right] \approx_\lambda 0 \ .$$

Intuitively, given only $[1]_\iota$ and $[x]_\iota$ it is assumed to be hard to generate a pair of the form $([a]_\iota, a[x]_\iota)$ unless one starts by simply choosing $a \in \mathbb{Z}_p$.

Definition 7 (BDH-KE Knowledge Assumption [9])**.** *Let* $\mathscr{R}$ *be a relation and* $\mathsf{aux}_{\mathscr{R}}$ *be the auxiliary information*[4] *related to* $\mathscr{R}$*. We say that* Pgen *is* Bilinear Diffie-Hellman Knowledge of Exponents (BDH-KE) secure *for* $\mathscr{R}$ *if for any* $(\mathscr{R}, \mathsf{aux}_{\mathscr{R}}) \in \mathrm{range}(\mathscr{R}(1^\lambda))$ *and PPT adversary* $\mathscr{A}$ *there exists a PPT extractor* $\mathsf{Ext}_{\mathscr{A}}^{\mathsf{BDH\text{-}KE}}$*, such that*

$$\Pr\left[\begin{array}{l} r \leftarrow_{\$} \mathsf{RND}_\lambda(\mathscr{A}); ([\alpha_1]_1, [\alpha_2]_2 || a) \leftarrow (\mathscr{A} || \mathsf{Ext}_{\mathscr{A}}^{\mathsf{BDH\text{-}KE}})(\mathscr{R}, \mathsf{aux}_{\mathscr{R}}; r) : \\ [\alpha_1]_1[1]_2 = [1]_1[\alpha_2]_2 \wedge a \neq \alpha_1 \end{array}\right] \approx_\lambda 0 \ .$$

Notice that one may view the BDH-KE assumption as a simple case of the Power Knowledge of Exponent (PKE) assumption of [64] (where $\mathscr{A}$ is given as an input the tuple $\{([x^i]_1, [x^i]_2)\}_{i=0}^n$ for some $n \geq 0$, and assumed that if $\mathscr{A}$ outputs $([\alpha]_1, [\alpha]_2)$ then she knows $(a_0, a_1, \ldots, a_n)$, such that $\alpha = \sum_{i=0}^{n} a_i x^i$.) as used in the case of asymmetric pairings in [64]. Therefore, BDH-KE can be viewed as an asymmetric-pairing version of the original KoE assumption [61].

2.4. Universal Composability

The Universal Composability (UC) framework was introduced by Canetti in [52]. In the UC framework, one analyzes the security of the protocol under the real-world and ideal-world paradigm. More precisely, in this setting, the real-world execution of a protocol is compared with an ideal-world interaction with the primitive that it implements. Then a composition theorem in this model states that the security of the UC-secure protocols remains if it is arbitrarily composed with other UC-secure protocols or the protocol itself. Additionally, the UC-secure property guarantees security in practical applications where individual instances of protocols are run in parallel, such as the internet. The entities in the UC framework in both ideal-word and real-word executions are modeled as PPT interactive Turing machines that send and receive messages through respectively their output and input tapes.

[4] In this thesis, the auxiliary information is the group description that contains the generator and the size of the group. In general, if we allow any auxiliary information then we have non-uniform PPT adversaries (a family of circuits). If we would not have such powerful adversaries, then we usually put some restrictions over the auxiliary information.

In the ideal world execution, dummy parties (possibly controlled by an ideal-word adversary Sim, also called simulator) communicate directly with the ideal functionality $\mathscr{F}$. The ideal functionality can be viewed as a trusted party that creates the primitives to implement the protocol. Correspondingly, in the real-world execution, parties (possibly corrupted by a real-world adversary $\mathscr{A}$) communicate with each other as a protocol Π that realizes the ideal functionality. Both the ideal and real executions are controlled by the environment $\mathscr{Z}$, an entity that sends inputs and receives the outputs of $\mathscr{A}$, the individual parties, and Sim. Finally, after seeing the ideal or real protocol execution, $\mathscr{Z}$ returns a bit, which is considered as the output of the execution. Then the rationale behind this framework lies in showing that the environment $\mathscr{Z}$ can not efficiently distinguish between the ideal and real executions, therefore meaning that the real-world protocol is as secure as the ideal-world (the ideal functionality).

Besides, the two aforementioned models (real-world and ideal-world) of computation, the UC framework considers the hybrid-world, where the executions are similar to the real-world but with the additional assumption that the parties are allowed to access to an auxiliary ideal functionality $\mathscr{G}$. More precisely, in this case, instead of honest parties interacting directly with the ideal functionality, the adversary passes all the messages from and to the ideal functionality. Also, the transmission channels are considered to be ideally authenticated, meaning that the adversary is not able to modify the messages but only able to read them. Unlike information transferred between parties, which can be read by the adversary, the information transferred between parties and the ideal functionality is split into a public and private header. The private header carries some information like as the private inputs of parties and it cannot be read by the adversary. The public header carries only some information that can be viewed publicly such as receiver, sender, type of a message, and session identifiers. Let denote the output of the environment $\mathscr{Z}$ that shows the execution of a protocol Π in a real-world model and a hybrid model, respectively as $\text{IDEAL}^{\mathscr{F}}_{\mathsf{Sim}}$ and $\text{HYBRID}^{\mathscr{G}}_{\Pi,\mathscr{A}}$. Then the UC security is formally defined as:

Definition 8. *A n-party ($n \in \mathbb{N}$) protocol Π UC-realizes an ideal functionality $\mathscr{F}$ in the hybrid model if, for every PPT adversary $\mathscr{A}$, there exists a simulator* Sim *such that for all environments $\mathscr{Z}$,*

$$\text{IDEAL}^{\mathscr{F}}_{\mathsf{Sim}} \approx_{\lambda} \text{HYBRID}^{\mathscr{G}}_{\Pi,\mathscr{A}}.$$

The protocol Π is statistically secure if the above definition holds for all unbounded $\mathscr{Z}$. In Chapter 4, we define the ideal functionality for a commitment scheme and provide its corresponding hybrid functionality to prove the UC security of the scheme.

2.5. Public-Key Cryptosystems

Labeled Public-Key Encryption. Labeled encryption is a variant of the public-key encryption where encryption and decryption take an additional label. Decryption should only work if the label used for decryption is identical to the one used when producing the ciphertext. More formally:

Definition 9. *A labeled (tagged) public-key encryption scheme* $\Pi = (\mathsf{KGen}, \mathsf{Enc}, \mathsf{Dec})$*, is defined by three algorithms:*

$\mathsf{KGen}(1^\lambda)$*: given a security parameter* λ*, generates a public key* pk *and a secret key* sk*.*

$\mathsf{Enc}^\tau_{\mathsf{pk}}(M)$*: given the public key* pk*, a label* τ *and a message* M*, outputs a ciphertext* C*.*

$\mathsf{Dec}^\tau_{\mathsf{sk}}(C)$*: given a label* τ*, the secret key* sk *and a ciphertext* C*, with* $C = \mathsf{Enc}^\tau_{\mathsf{pk}}(M)$*, outputs* M*.*

For correctness, it is required that for all $(\mathsf{pk}, \mathsf{sk}) \in \mathsf{KGen}(1^\lambda)$, all labels τ and all messages M, $\mathsf{Dec}^\tau_{\mathsf{sk}}(\mathsf{Enc}^\tau_{\mathsf{pk}}(M)) = M$. Henceforth, we use public-key encryption schemes that provide indistinguishability under chosen plaintext and adaptive chosen ciphertexts attacks, i.e., provide IND-CPA or IND-CCA security respectively.

(Labeled) Cramer-Shoup encryption [60]. The Cramer-Shoup (CS) IND-CCA2 secure public-key encryption scheme in group $\mathbb{G}_1$ of order p is defined as follows: the secret key sk is $(x_1, x_2, y_1, y_2, z) \leftarrow_\$ \mathbb{Z}_p^5$. Assume $[g_1]_1, [g_2]_1$ are two different generators of $\mathbb{G}_1$. Let H be a collision-resistant hash function. The public key is $\mathsf{pk} = ([g_1, g_2, h, c, d]_1, H)$, where $[c]_1 = x_1[g_1]_1 + x_2[g_2]_1$, $[d]_1 = y_1[g_1]_1 + y_2[g_2]_1$, $h = z[g_1]_1$. The encryption of $[m]_1$ with randomness $r \leftarrow_\$ \mathbb{Z}_p$ is defined as $[C]_1 = [u_1, u_2, e, v]_1$ where $[u_1]_1 = r[g_1]_1$, $[u_2]_1 = r[g_2]_1$, $[e]_1 = [m]_1 + r[h]_1$, $[v]_1 = r([c]_1 + \xi[d]_1)$, where $\xi = H([u_1]_1, [u_2]_1, [e]_1)$. In case of labeled CS with label τ, the hash value is computed as $\xi = H(\tau, [u_1]_1, [u_2]_1, [e]_1)$.

2.6. Commitment Schemes

Definition 10. *A commitment scheme* $\Pi = (\mathsf{KGen}, \mathsf{Commit}, \mathsf{Decommit})$*, is defined by the following three algorithms:*

$\mathsf{KGen}(1^\lambda)$*: given a security parameter* λ*, generates public parameter* p *of the scheme that implicitly passed as input to the other algorithms.*

$\mathsf{Commit}(\mathsf{p}, m, r)$*: given the public parameter* p*, a message* m *from message space* M*, and a randomness* r *from randomness space* R*, outputs a commitment* c *together with an opening information* δ[5]*.*

[5]The opening information will be used in the decommit phase to prove that the commitment c contains a valid message m.

$\mathsf{Decommit}(\mathsf{p},\mathsf{c},m,\delta)$*: given given the public parameter* p*, a commitment* c*, the message* m *and an opening information* δ*, outputs* m *or* $\perp$ *if the opening verification fails.*

Such a scheme must satisfy both *hiding* property (meaning that the commit phase does not disclose any information about the committed message m), and *binding* property (meaning that the decommit phase (opening phase) can successfully open to only one value). The aforementioned properties may be achieved in a perfect, statistical or computational, according to the power of the adversary against those properties. Besides, some additional strong properties are demanded in some systems like the UC-secure commitment scheme. The first is *extractability* which states that given a trapdoor, one (i.e, the simulator Sim in UC model) can recover the committed value m. The second one is *equivocability* which means that given a trapdoor, one (i.e, the simulator Sim in UC model) can open a commitment to any message $m' \neq m$.

Pedersen Commitment [127]. The Pedersen commitment is an equivocable commitment and defined as follows: (i) the $\mathsf{KGen}(1^\lambda)$ generates a group $\mathbb{G}_1$ of order p, with two generators $[g]_1$ and $[h]_1 = s[g]_1$ where $s \in \mathbb{Z}_p$. Sets $\mathsf{p} = ([g]_1, [h]_1)$. (ii) The $\mathsf{Commit}(\mathsf{p},m,r)$ algorithm for a message $m \in \mathbb{Z}_p$ and the random element $r \leftarrow_\$ \mathbb{Z}_p$, computes the commitment $\mathsf{c} = m[g]_1 + r[h]_1$; while the opening information $\delta = r$, and (iii) the $\mathsf{Decommit}(\mathsf{p},\mathsf{c},m,\delta)$ algorithm returns m and r and allows to check the validity of the commitment by verifying that $\mathsf{c} =^? m[g]_1 + r[h]_1$. Such a commitment is computationally binding under the discrete logarithm assumption. Also it is an equivocable commitment since given the secret value (the discrete logarithm s) as additional information (or a trapdoor), one can open the commitment to any message $m' \neq m$. Notice that the Cramer-Shoup encryption [60] can be viewed as an extractable commitment scheme.

2.7. Smooth Projective Hash Functions

Let $\mathscr{L}_{\mathtt{lpar}} \subset \mathscr{X}_{\mathtt{lpar}}$ be a language parameterized by $\mathtt{lpar}$ (the language parameter), where $\mathscr{X}_{\mathtt{lpar}}$ is the underlying domain, e.g., a group. A projective hash function (PHF, [60]) for $\{\mathscr{L}_{\mathtt{lpar}}\}$ is a tuple of PPT algorithms $\mathsf{PHF} = (\mathsf{Pgen}, \mathsf{Pgen.lpar}, \mathsf{hashkg}, \mathsf{projkg}, \mathsf{hash}, \mathsf{projhash})$, defined as follows. Here, $\mathscr{R}_{\mathtt{lpar}}$ is the relation defined by $\mathscr{L}_{\mathtt{lpar}} = \{x : \exists w, (x,w) \in \mathscr{R}_{\mathtt{lpar}}\}$.

$\mathsf{Pgen}(1^\lambda)$**:** takes a security parameter λ and generates the global parameters p (we assume that all algorithms have access to p).

$\mathsf{Pgen.lpar}(\mathsf{p})$**:** sets up the language parameters $\mathtt{lpar}$ (with its trapdoor $\mathtt{tds}$),

$\mathsf{hashkg}(\mathtt{lpar})$**:** inputs a language parameter $\mathtt{lpar}$. It generates and outputs a hashing key hk for $\mathscr{L}_{\mathtt{lpar}}$.

$\mathsf{projkg}(\mathtt{lpar};\mathsf{hk},x)$**:** inputs a language parameter $\mathtt{lpar}$, a hashing key hk, and possibly a word $x \in \mathscr{X}_{\mathtt{lpar}}$. It outputs deterministically a projection key hp.

$\mathsf{hash}(\mathtt{lpar};\mathsf{hk},x)$**:** inputs a language parameter $\mathtt{lpar}$, a hashing key hk, and a word $x \in \mathscr{X}_{\mathtt{lpar}}$. It outputs deterministically a hash value H.

$\mathsf{projhash}(\mathtt{lpar};\mathsf{hp},x,w)$**:** inputs a language parameter $\mathtt{lpar}$, a projection key hp, a word x, and a witness w for $x \in \mathscr{L}_{\mathtt{lpar}}$ (i.e., $(x,w) \in \mathscr{R}_{\mathtt{lpar}}$). It outputs deterministically a projected hash value pH.

The set of hash values is called the *range* of PHF and is denoted by Π. We assume Π to be efficiently sampleable and that Π has size that is exponential in λ.

In what follows, $\mathtt{lpar}$ contains p and some public parameters specifying the relation (e.g., an encryption key). A distribution $\mathscr{D}_{\mathsf{p}}$ (e.g., the output distribution of $\mathsf{Pgen.lpar}(\mathsf{p})$) on $\mathscr{L}_{\rho}$ is *witness-sampleable* [100] if there exists a PPT algorithm $\mathsf{Pgen.lpar}(\mathsf{p})$ that samples $(\mathtt{lpar},\mathsf{td}) \in \mathscr{R}_{\mathsf{p}}$ such that $\mathtt{lpar}$ is distributed according to $\mathscr{D}_{\mathsf{p}}$, and membership of x in *the parameter language* $\mathscr{L}_{\mathtt{lpar}}$ can be verified in PPT given td. We always assume that $\mathtt{lpar}$ can be efficiently computed from td. In the related work, $\mathscr{D}_{\mathsf{p}}$ is often assumed to be witness-sampleable, even if it is not always necessary.

A PHF is *perfectly correct* if for all parameters $\mathtt{lpar} \in \mathrm{im}(\mathsf{Pgen.lpar}(\mathsf{p}))$, $(x,w) \in \mathscr{R}_{\mathtt{lpar}}$, $\mathsf{hk} \in \mathrm{im}(\mathsf{hashkg}(\mathtt{lpar}))$, and $\mathsf{hp} \leftarrow \mathsf{projkg}(\mathtt{lpar};\mathsf{hk})$, the following holds, $\mathsf{hash}(\mathtt{lpar};\mathsf{hk},x) = \mathsf{projhash}(\mathtt{lpar};\mathsf{hp},x,w)$.

There are at least three types of smooth PHFs (SPHFs). Intuitively, in GL-SPHF [83], we want security even in the case hp can maliciously depend on x. On the other hand, in KV-SPHF [104], we want security even in the case x can maliciously depend on hp. The third type is CS-SPHF, [60]. See [24, Section 2.5] for more information.

A PHF $\mathsf{PHF} = (\mathsf{Pgen},\mathsf{Pgen.lpar},\mathsf{hashkg},\mathsf{projkg},\mathsf{hash},\mathsf{projhash})$ for a language $\mathscr{L} \subseteq \mathscr{X}$ is *ε-GL/CS-smooth* if for any $\mathtt{lpar}$ and any word $x \in \mathscr{X}_{\mathtt{lpar}} \setminus \mathscr{L}_{\mathtt{lpar}}$, the following distributions are ε-close:

$$\{(\mathsf{hp},\mathsf{H}) : \mathsf{hk} \leftarrow_{\$} \mathsf{hashkg}(\mathtt{lpar}); \mathsf{hp} \leftarrow \mathsf{projkg}(\mathtt{lpar};\mathsf{hk},x); \mathsf{H} \leftarrow \mathsf{hash}(\mathtt{lpar};\mathsf{hk},x)\}$$
$$\{(\mathsf{hp},\mathsf{H}) : \mathsf{hk} \leftarrow_{\$} \mathsf{hashkg}(\mathtt{lpar}); \mathsf{hp} \leftarrow \mathsf{projkg}(\mathtt{lpar};\mathsf{hk},x); \mathsf{H} \leftarrow_{\$} \Pi\}$$

A PHF is GL/CS-smooth if it is ε-GL/CS-smooth with ε negligible in λ.

An SPHF satisfying this smoothness definition is called

- a *CS-smooth projective hash function (CS-SPHF)* if hp does not depend on x (i.e., projkg does not use its input x),
- a *GL-smooth projective hash function (GL-SPHF)* if hp may depend on x.

A PHF $\mathsf{PHF} = (\mathsf{Pgen},\mathsf{Pgen.lpar},\mathsf{hashkg},\mathsf{projkg},\mathsf{hash},\mathsf{projhash})$ for a language $\mathscr{L} \subseteq \mathscr{X}$ is *ε-KV-smooth* if for any $\mathtt{lpar}$ and any (not necessarily computable in polynomial-time) function f from the set of possible projection keys hp to $\mathscr{X}_{\mathtt{lpar}} \setminus \mathscr{L}_{\mathtt{lpar}}$, the following distributions are ε-close:

$$\{(\mathsf{hp},\mathsf{H}) : \mathsf{hk} \leftarrow_{\$} \mathsf{hashkg}(\mathtt{lpar}); \mathsf{hp} \leftarrow \mathsf{projkg}(\mathtt{lpar};\mathsf{hk}); \mathsf{H} \leftarrow \mathsf{hash}(\mathtt{lpar};\mathsf{hk},f(\mathsf{hp}))\}$$
$$\{(\mathsf{hp},\mathsf{H}) : \mathsf{hk} \leftarrow_{\$} \mathsf{hashkg}(\mathtt{lpar}); \mathsf{hp} \leftarrow \mathsf{projkg}(\mathtt{lpar};\mathsf{hk}); \mathsf{H} \leftarrow_{\$} \Pi\}$$

A PHF is KV-smooth if it is ε-KV-smooth with ε negligible in λ. Since projkg does not depend on x in this case, we often omit x as an argument for projkg.

An ε-smooth CS-SPHF is not necessarily an ε-smooth KV-SPHF, but a 0-smooth CS-SPHF is always a 0-smooth KV-SPHF, [24, Remark 2.5.4]. In this thesis, we will explicitly deal with GL-SPHFs (GL-smooth PHFs) and KV-SPHFs (KV-smooth PHFs), ignoring CS-SPHFs (CS-smooth PHFs).

Smooth Projective Hash Function for (Labeled) CS Ciphertexts. Benhamouda et al. [27, 28] presented KV-SPHF for (Labeled) CS encryption [60]. Briefly in this construction, the hashing key is a vector $\mathsf{hk} = (\eta_1, \eta_2, \theta, \mu, \iota) \leftarrow_{\$} \mathbb{Z}_p^5$, and for a word for CS ciphertext $[C]_1$, the projection key is, $\mathsf{hp} = (\mathsf{hp}_1 = \eta_1[g_1]_1 + \theta[g_2]_1 + \mu[h]_1 + \iota[\iota]_1, \mathsf{hp}_2 = \eta_2[g_1]_1 + \iota[d]_1)$ and,

$$\begin{aligned}\mathsf{H} =& \mathsf{hash}(\mathsf{hk}, (\mathsf{pk}, [m]_1), [C]_1) = (\eta_1 + \xi\eta_2)[u_1]_1 + \theta[u_2]_1 + \mu([e]_1 - [m]_1) + \iota[v]_1 \\ =& r[\mathsf{hp}_1]_1 + r\xi[\mathsf{hp}_2]_1 = \mathsf{projhash}(\mathsf{hp}, (\mathsf{pk}, [m]_1), [C]_1, r) = \mathsf{pH}.\end{aligned}$$

2.8. Registered and Bare Public Key Model.

In the registered public key (RPK, [19]) model, it is assumed that each party $\mathscr{P}_{id}$ trusts *some* key-registration authority $\mathscr{R}_{id}$ and has registered her key with $\mathscr{R}_{id}$. (The same $\mathscr{R}_{id}$ can be used by several parties, or each party can decide to trust a separate authority.) If $\mathscr{P}_{id}$ is honest, then the secret key exists and the public key comes from correct distribution (in this case, the public key is said to be *"safe"*). If $\mathscr{P}_{id}$ is dishonest, the secret key still exists (and the public key has been computed from it honestly) but there is no guarantee about its distribution (in this case, the public key is said to be *"well-formed"*).

Different variants (most importantly, the "traditional proof-of-knowledge" version where the secret key and the public key are generated by $\mathscr{P}_{id}$ who then sends the public key to $\mathscr{R}_{id}$ and proves the knowledge of the secret key to $\mathscr{R}_{id}$ by using a stand-alone zero-knowledge proof) of the RPK model are known. We assume that each party knows the identities of all other parties and their key-registration authorities, see [19] for discussion.

In the Bare Public Key (BPK) model [54, 120], parties have access to a public file F, a polynomial-size collection of records (id, pk_{id}), where id is a string identifying a party (e.g., a verifier), and pk_{id} is her (alleged) public key. In a typical zero-knowledge protocol in the BPK model, a key-owning party $\mathscr{P}_{id}$ works in two stages. In stage one (the *key-generation stage*), on input a security parameter 1^{λ} and randomizer r, $\mathscr{P}_{id}$ outputs a public key pk_{id} and stores the corresponding secret key sk_{id}. We assume the *no-auxiliary-string BPK* model where from this it follows that $\mathscr{P}_{id}$ actually created pk_{id}. After that, F will include (id, pk_{id}). In stage two, each party has access to F, while $\mathscr{P}_{id}$ has possibly access to sk_{id} (however, the latter is not required by us). It is commonly assumed that only the verifier of a NIZK argument system in the BPK model has a public key [120].

2.9. Generic Model

In the *Generic Bilinear Group Model* (GBGM) [35, 118, 124, 131], one assumes that the adversary has only access to group elements via generic bilinear-group operations (group operations and the bilinear map) together with an equality test. In the *subversion GBGM* (Sub-GBGM, [9, 22, 43, 132]; named *generic group model with hashing into the group* in [22]), the adversary has an additional power of creating new random group elements. The Sub-GBGM is motivated by known elliptic curve hashing algorithms [42, 99, 133] that allow one to efficiently create elliptic-curve group elements without knowing their discrete logarithm.

Hence, Sub-GBGM is a weaker model than GBGM. As an important example, knowledge assumptions that state that the output group element must belong to the span of input group elements hold in the GBGM but not in the Sub-GBGM. This is since, in the Sub-GBGM, the adversary can create new group elements without knowing their discrete logarithms; indeed the output element might be equal to one such created group elements. Thus, a Sub-GBGM adversary is less restricted than a GBGM adversary. See [9] for a longer introduction to GBGM and Sub-GBGM.

2.10. Non-Interactive Zero-Knowledge

Let RGen be a relation generator, such that $\mathsf{RGen}(1^\lambda)$ outputs a polynomial time decidable binary relation $\mathscr{R} = \{(x,w)\}$. Here, x and w are respectively the statement and the witness. The generator RGen also returns some auxiliary information $\mathsf{aux}_{\mathscr{R}}$ that is public for the adversary and the honest parties. Let $\mathscr{L}_{\mathscr{R}} = \{x : \exists w, (x,w) \in \mathscr{R}\}$ be an NP-language. NIZK proofs in the CRS model consists of the four algorithms $(\mathsf{K}, \mathsf{P}, \mathsf{V}, \mathsf{Sim})$ where K, P, V, and Sim are common reference strings (CRS) generator, prover, verifier, and the simulator, respectively. A NIZK must provide the following properties: (i) completeness (for all CRS crs generated by K and $(x,w) \in \mathscr{R}$, we have that $\mathsf{V}(\mathsf{crs}, x, \mathsf{P}(\mathsf{crs}, x, w)) = 1$), (ii) zero-knowledge (there exists a simulator Sim that returns a simulated proof such that it is distinguishable from proofs computed by $\mathsf{P}(\mathsf{crs}, x, w)$), and (iii) soundness (an adversary cannot output a proof π and an instance $x \notin \mathscr{L}_{\mathscr{R}}$ such that $\mathsf{V}(\mathsf{crs}, x, \pi) = 1$. Moreover, the knowledge soundness property goes further and says that for any prover P generating a valid proof there is an extractor Ext that can extract a valid witness w.

2.11. zk-SNARKs

2.11.1. Quadratic Arithmetic Programs

Quadratic Arithmetic Program (QAP) was introduced by Gennaro *et al.* [82] as a language $\mathscr{L}$ where for a witness w and input x, predicate $(x,w) \in \mathscr{R}$ can be verified with a parallel quadratic checking; and it also has an efficient reduction

from the language (either Boolean or Arithmetic) CIRCUIT-SAT. Therefore, an efficient zk-SNARK for QAP results in an efficient zk-SNARK for CIRCUIT-SAT.

Let m be the number of wires and n be the number of multiplication gates in an arithmetic circuit. Similarly to [82], we assume arithmetic circuits that consist only of fan-in-2 multiplication gates, where either input of each multiplication gate is a weighted sum of some wire values.

Let $\mathbb{F} = \mathbb{Z}_p$, such that ω is the nth primitive root of unity modulo p.

An QAP instance $\mathscr{Q}$ is specified by $(\mathbb{Z}_p, m_0, \{u_j, v_j, w_j\}_{j=1}^m)$, where $u_j(X) = \sum_{i=1}^n U_{ij}\ell_i(X)$, $v_j(X) = \sum_{i=1}^n V_{ij}\ell_i(X)$, and $w_j(X) = \sum_{i=1}^n W_{ij}\ell_i(X)$ are polynomials from $\mathbb{Z}_p[X]$ with degree $d \leq (n-1)$. Here, $U, V, W \in \mathbb{Z}_p^{n\times m}$ are public sparse matrices. The instance $\mathscr{Q}$ defines the following relation:

$$\mathscr{R}_{\mathscr{Q}} = \left\{ \begin{array}{l} (x,w) \colon x = (A_1, \ldots, A_{m_0})^\top \wedge w = (A_{m_0+1}, \ldots, A_m)^\top \wedge \\ (\sum_{j=1}^m A_j u_j(X))(\sum_{j=1}^m A_j v_j(X)) \equiv \sum_{j=1}^m A_j w_j(X) \pmod{\ell(X)} \end{array} \right\} .$$

Alternatively, $(x,w) \in \mathscr{R}$ if there exists a degree-$\leq (n-2)$ polynomial $h(X)$, s.t.

$$(\textstyle\sum_{j=1}^m A_j u_j(X))(\sum_{j=1}^m A_j v_j(X)) - \sum_{j=1}^m A_j w_j(X) = h(X)\ell(X) \ .$$

2.11.2. Definitions: zk-SNARKs

Now we recall a formal description of zk-SNARKs and their security properties. As in [94], we assume that the relation generator RGen also returns auxiliary information $\mathsf{aux}_{\mathscr{R}}$ that equals $\mathsf{p} \leftarrow \mathsf{Pgen}(1^\lambda)$. So, we give $\mathsf{aux}_{\mathscr{R}}$ as an input to all parties (including honest or adversarial).

Let $\mathscr{L}_{\mathscr{R}} = \{x : \exists w \text{ s.t. } |w| = \mathsf{poly}(|x|), (x,w) \in \mathscr{R}\}$ be an NP-language. Let $\tilde{\mathsf{z}}_{\mathscr{R}}$ be a common auxiliary information that is generated by using a relation generator RGen [31]. A NIZK system Ψ for RGen consists of four PPT algorithms:

CRS generator K**:** a *probabilistic* algorithm that, given $\tilde{\mathsf{z}}_{\mathscr{R}} \in \mathrm{range}(\mathsf{RGen}(1^\lambda))$ outputs a *CRS trapdoor* td and a CRS crs. Otherwise, it outputs $\perp$.

Prover P**:** a probabilistic algorithm that, given $(\tilde{\mathsf{z}}_{\mathscr{R}}, \mathsf{crs}, x, w)$, outputs an argument π if $(x,w) \in \mathscr{R}$. Otherwise, it outputs $\perp$.

Verifier V**:** a probabilistic algorithm that, given $(\tilde{\mathsf{z}}_{\mathscr{R}}, \mathsf{crs}, x, \pi)$, returns either 0 (reject) or 1 (accept).

Simulator Sim**:** a probabilistic algorithm that, given $(\tilde{\mathsf{z}}_{\mathscr{R}}, \mathsf{crs}, \mathsf{td}, x)$ outputs an argument π. Otherwise, it outputs $\perp$.

Definition 11 (SNARK)**.** *A non-interactive system* Ψ *is a succinct non-interactive argument of knowledge (SNARK) for relation generator* RGen *if it is complete and knowledge sound, and moreover succinct, meaning that for all* λ*, all* $(\mathscr{R}, \mathsf{aux}_{\mathscr{R}}) \in \mathrm{im}(\mathsf{RGen}(1^\lambda))$*, all* $\mathsf{crs} \leftarrow \mathsf{K}(\tilde{\mathsf{z}}_{\mathscr{R}})$*, all* $(x,w) \in \mathscr{R}$ *and all proofs* $\pi \leftarrow \mathsf{P}(\tilde{\mathsf{z}}_{\mathscr{R}}, \mathsf{crs}, x, w)$ *we have* $|\pi| = poly(\lambda)$ *and* $\mathsf{V}(\tilde{\mathsf{z}}_{\mathscr{R}}, \mathsf{crs}, x, \pi)$ *runs in time polynomial in* $\lambda + |x|$*.*

A NIZK argument Ψ must satisfy the following properties:

(i) **Completeness.** For any λ, and $(x,w) \in \mathscr{R}_\rho$,

$$\Pr\left[(\mathsf{crs},\mathsf{td}) \leftarrow \mathsf{K}(\tilde{\mathsf{z}}_{\mathscr{R}}) : \mathsf{V}(\tilde{\mathsf{z}}_{\mathscr{R}},\mathsf{crs},x,\mathsf{P}(\tilde{\mathsf{z}}_{\mathscr{R}},\mathsf{crs},x,w)) = 1 \quad \right] = 1 \ .$$

(ii) **Statistical Zero-Knowledge.** For any computationally unbounded adversary $\mathscr{A}$, $|\varepsilon_0^{zk} - \varepsilon_1^{zk}| \approx_\lambda 0$, where $\varepsilon_b^{zk} :=$

$$\Pr\left[(\mathsf{crs},\mathsf{ts}) \leftarrow \mathsf{K}(\tilde{\mathsf{z}}_{\mathscr{R}}), b \leftarrow_{\$} \{0,1\} : \mathscr{A}^{\mathsf{O}_b(\cdot,\cdot)}(\tilde{\mathsf{z}}_{\mathscr{R}},\mathsf{crs}) = 1 \quad \right].$$

The oracle $\mathsf{O}_0(x,w)$ returns $\perp$ (reject) if $(x,w) \notin \mathscr{R}_\rho$, and otherwise it returns $\mathsf{P}(\tilde{\mathsf{z}}_{\mathscr{R}},\mathsf{crs},x,w)$. Similarly, $\mathsf{O}_1(x,w)$ returns $\perp$ (reject) if $(x,w) \notin \mathscr{R}_\rho$, and otherwise it returns $\mathsf{Sim}(\tilde{\mathsf{z}}_{\mathscr{R}},\mathsf{crs},\mathsf{td},x)$.

(iii) **Computational Soundness.** For any non-uniform PPT $\mathscr{A}$,

$$\Pr\begin{bmatrix}(\mathsf{crs},\mathsf{td}) \leftarrow \mathsf{K}(\tilde{\mathsf{z}}_{\mathscr{R}}); (x,\pi) \leftarrow \mathscr{A}(\tilde{\mathsf{z}}_{\mathscr{R}},\mathsf{crs}) : \\ \mathsf{V}(\tilde{\mathsf{z}}_{\mathscr{R}},\mathsf{crs},x,\pi) = 1 \wedge \neg(\exists w : (x,w) \in \mathscr{R}_\rho)\end{bmatrix} \approx_\lambda 0 \ .$$

In zk-SNARK constructions, Ψ needs to satisfy a strong definition of the computational soundness called computationally *knowledge-soundness* [94], if for all non-uniform PPT $\mathscr{A}$, there is a non-uniform PPT extractor $\mathsf{Ext}_{\mathscr{A}}$,

$$\Pr\begin{bmatrix}(\mathsf{crs},\mathsf{td}) \leftarrow \mathsf{K}(\tilde{\mathsf{z}}_{\mathscr{R}}); r \leftarrow_{\$} \mathsf{RND}_\lambda(\mathscr{A}), \\ (x,\pi) \leftarrow \mathscr{A}(\tilde{\mathsf{z}}_{\mathscr{R}},\mathsf{crs};r), w \leftarrow \mathsf{Ext}_{\mathscr{A}}(\tilde{\mathsf{z}}_{\mathscr{R}},\mathsf{crs};r) : \\ (x,w) \notin \mathscr{R} \wedge \mathsf{V}(\tilde{\mathsf{z}}_{\mathscr{R}},\mathsf{crs},x,\pi) = 1\end{bmatrix} \approx_\lambda 0 \ .$$

Next we recall some addition (strong) properties of NIZKs.

Definition 12 (Statistically Composable Zero-Knowledge [95])**.** Ψ *is* statistically composable zero-knowledge (ZK) for RGen*, if for all* $(x,w) \in \mathscr{R}$ *and all computationally unbounded* $\mathscr{A}$*,* $\varepsilon_0^{comp} \approx_\lambda \varepsilon_1^{comp}$*, where*

$$\varepsilon_b^{comp} = \Pr\begin{bmatrix}(\mathsf{crs},\mathsf{td}) \leftarrow \mathsf{K}(\tilde{\mathsf{z}}_{\mathscr{R}}), \\ \textbf{if } b = 0 \textbf{ then } \pi \leftarrow \mathsf{P}(\tilde{\mathsf{z}}_{\mathscr{R}},\mathsf{crs},x,w) \\ \textbf{else } \pi \leftarrow \mathsf{Sim}(\tilde{\mathsf{z}}_{\mathscr{R}},\mathsf{crs},\mathsf{td},x) \textbf{ fi} : \mathscr{A}(\tilde{\mathsf{z}}_{\mathscr{R}},\mathsf{crs},\mathsf{td},\pi) = 1\end{bmatrix} \ .$$

Ψ *is* perfectly composable ZK for RGen *if one requires that* $\varepsilon_0^{comp} = \varepsilon_1^{comp}$.

The adversary $\mathscr{A}$ in composable ZK definition only can see an argument π (either real or simulated π) and is not allowed to make many queries to the oracle (as in the case of the following unbounded ZK). Next, we recall statistically unbounded ZK [90]. [6]

[6]Notice that Unbounded ZK was not defined in [90], presumably because it is a corollary of composable ZK as it is shown in [90].

Definition 13 (Statistically Unbounded ZK [91])**.** Ψ *is* statistically unbounded ZK for RGen*, if for all computationally unbounded* $\mathscr{A}$*,* $\varepsilon_0^{unb} \approx_\lambda \varepsilon_1^{unb}$*, where*

$$\varepsilon_b^{unb} = \Pr[(\mathsf{crs},\mathsf{td}) \leftarrow \mathsf{K}(\tilde{\mathsf{z}}_{\mathscr{R}}) : \mathscr{A}^{\mathsf{O}_b(\cdot,\cdot)}(\tilde{\mathsf{z}}_{\mathscr{R}},\mathsf{crs}) = 1] \ .$$

Here, the oracle $\mathsf{O}_0(x,w)$ *returns* $\perp$ *(reject) if* $(x,w) \notin \mathscr{R}$*, and otherwise it returns* $\mathsf{P}(\tilde{\mathsf{z}}_{\mathscr{R}},\mathsf{crs},x,w)$*. Similarly,* $\mathsf{O}_1(x,w)$ *returns* $\perp$ *(reject) if* $(x,w) \notin \mathscr{R}$*, and otherwise it returns* $\mathsf{Sim}(\tilde{\mathsf{z}}_{\mathscr{R}},\mathsf{crs},\mathsf{td},x)$*.* Ψ *is* perfectly unbounded ZK for RGen *if one requires that* $\varepsilon_0^{unb} = \varepsilon_1^{unb}$*.*

From [90], it is known that composable ZK is a stronger requirement than unbounded ZK since in the case of composable ZK, (i) the adversary gets access to td, and (ii) the simulated CRS was required to be indistinguishable from the real CRS.

Groth's SNARK construction [94]. We recall the construction of the most efficient zk-SNARK proposed by Groth [94] in Fig. 1.

Theorem 1 (Theorem 2 of [94])**.** *The construction in Fig. 1 is perfectly complete, perfect zero-knowledge and statistically knowledge sound.*

2.12. Quasi-Adaptive NIZK Arguments

We recall the definition of QA-NIZK arguments of Jutla and Roy [100]. A QA-NIZK proof system provides proof for membership of words x in accordance with witnesses w in a language $\mathscr{L}_\rho$ defined by a relation $\mathscr{R}_\rho$ which is parametrized by some language parameter $\mathtt{lpar} := \rho$ chosen from a distribution $\mathscr{D}_\mathsf{p}$. The distribution $\mathscr{D}_\mathsf{p}$ is witness samplable if there exists an efficient algorithm that samples (ρ,td_ρ) where td_ρ is a secret value associated to ρ, so that the parameter ρ is distributed according to $\mathscr{D}_\mathsf{p}$. The membership of the language parameter ρ can be efficiently verified with the trapdoor td_ρ. The CRS of QA-NIZKs depends on a language parameter ρ and as mentioned in [100], it has to be chosen from a correct distribution $\mathscr{D}_\mathsf{p}$.

A tuple of PPT algorithms $\Pi = (\mathsf{KGen},\mathsf{P},\mathsf{V},\mathsf{Sim})$ is a QA-NIZK argument in the CRS model for a set of witness-relations $\mathscr{R}_\mathsf{p} = \{\mathscr{R}_\rho\}_{\rho\in\mathrm{Supp}(\mathscr{D}_\mathsf{p})}$ with ρ sampled from a distribution $\mathscr{D}_\mathsf{p}$ over associated parameter language $\mathscr{L}_\mathsf{p}$, if the following properties (i-iii) hold. Here, KGen is the parameter and the CRS generation algorithm, more precisely, KGen consists of two algorithms Pgen (generates the parameter p) and K (generates the CRS), P is the prover, V is the verifier, and Sim is the simulator.

(i) **Completeness.** For any λ, and $(x,w) \in \mathscr{R}_\rho$,

$$\Pr\begin{bmatrix}\mathsf{p} \leftarrow \mathsf{Pgen}(1^\lambda); \rho \leftarrow_\$ \mathscr{D}_\mathsf{p};\\ (\mathsf{crs},\mathsf{td}) \leftarrow \mathsf{K}(\rho); \pi \leftarrow \mathsf{P}(\rho,\mathsf{crs},x,w) :\\ \mathsf{V}(\rho,\mathsf{crs},x,\pi) = 1\end{bmatrix} = 1 \ .$$

$\mathsf{K}(\check{\mathsf{z}}_{\mathscr{R}})$**:** sample $\mathsf{td} = (\chi, \alpha, \beta, \gamma, \delta) \leftarrow_{\$} (\mathbb{Z}_p^*)^5$. For $j \in [m]$, set $u_j(\chi) \leftarrow \sum_{i=1}^{n} U_{ij}\chi^i$, $v_j(\chi) \leftarrow \sum_{i=1}^{n} V_{ij}\chi^i$, $w_j(\chi) \leftarrow \sum_{i=1}^{n} W_{ij}\chi^i$;
compute

$$\mathsf{crs} \leftarrow \begin{pmatrix} [\alpha, \beta, \delta, (\chi^i)_{i=0}^{n-1}, ((u_j(\chi)\beta + v_j(\chi)\alpha + w_j(\chi))/\delta)_{j=m_0+1}^{m}, (\chi^i \ell(\chi)/\delta)_{i=0}^{n-2}]_1, \\ [((u_j(\chi)\beta + v_j(\chi)\alpha + w_j(\chi))/\gamma)_{j=1}^{m_0}]_1, [\beta, \gamma, \delta, (\chi^i)_{i=0}^{n-1}]_2 \end{pmatrix},$$

return $(\mathsf{crs}, \mathsf{ts})$.

$\mathsf{P}(\check{\mathsf{z}}_{\mathscr{R}}, \mathsf{crs}, x = (A_j)_{j=1}^{m_0}, w = (A_j)_{j=m_0+1}^{m})$**:** $a^\dagger(X) \leftarrow \sum_{j=1}^{m} A_j u_j(X)$; $b^\dagger(X) \leftarrow \sum_{j=1}^{m} A_j v_j(X)$; $c^\dagger(X) \leftarrow \sum_{j=1}^{m} A_j w_j(X)$;

$h(X) = \sum_{i=0}^{n-2} h_i X^i \leftarrow (a^\dagger(X)b^\dagger(X) - c^\dagger(X))/\ell(X)$; $r_a \leftarrow_{\$} \mathbb{Z}_p$; $r_b \leftarrow_{\$} \mathbb{Z}_p$;

$[h(\chi)\ell(\chi)/\delta]_1 \leftarrow \sum_{i=0}^{n-2} h_i [\chi^i \ell(\chi)/\delta]_1$;

$[\mathsf{a}]_1 \leftarrow \sum_{j=1}^{m} A_j [u_j(\chi)]_1 + [\alpha]_1 + r_a[\delta]_1$;

$[\mathsf{b}]_2 \leftarrow \sum_{j=1}^{m} A_j [v_j(\chi)]_2 + [\beta]_2 + r_b[\delta]_2$;

$[\mathsf{c}]_1 \leftarrow r_b[\mathsf{a}]_1 + r_a\left(\sum_{j=1}^{m} A_j [v_j(\chi)]_1 + [\beta]_1\right) + \sum_{j=m_0+1}^{m} A_j [(u_j(\chi)\beta + v_j(\chi)\alpha + w_j(\chi))/\delta]_1 + [h(\chi)\ell(\chi)/\delta]_1$;

return $\pi \leftarrow ([\mathsf{a}]_1, [\mathsf{b}]_2, [\mathsf{c}]_1)$.

$\mathsf{V}(\check{\mathsf{z}}_{\mathscr{R}}, \mathsf{crs}, x = (A_j)_{j=1}^{m_0}, \pi = ([\mathsf{a}]_1, [\mathsf{b}]_2, [\mathsf{c}]_1))$**:**

check $[\mathsf{a}]_1[\mathsf{b}]_2 \stackrel{?}{=} [\mathsf{c}]_1[\delta]_2 + (\sum_{j=1}^{m_0} A_j [(u_j(\chi)\beta + v_j(\chi)\alpha + w_j(\chi))/\gamma]_1)[\gamma]_2 + [\alpha]_1[\beta]_2$.

Figure 1. Groth's SNARK [94].

(ii) **Statistical Zero-Knowledge.** For any computationally unbounded adversary $\mathscr{A}$, $|\varepsilon_0^{zk} - \varepsilon_1^{zk}| \approx_\lambda 0$, where $\varepsilon_b^{zk} :=$

$$\Pr\left[\begin{array}{l} \mathsf{p} \leftarrow \mathsf{Pgen}(1^\lambda); \rho \leftarrow_{\$} \mathscr{D}_{\mathsf{p}}; (\mathsf{crs}, \mathsf{td}) \leftarrow \mathsf{K}(\rho); b \leftarrow_{\$} \{0,1\} : \\ \mathscr{A}^{\mathsf{O}_b(\cdot,\cdot)}(\rho, \mathsf{crs}) = 1 \end{array}\right].$$

The oracle $\mathsf{O}_0(x, w)$ returns $\perp$ (reject) if $(x, w) \notin \mathscr{R}_\rho$, and otherwise it returns $\mathsf{P}(\rho, \mathsf{crs}, x, w)$. Similarly, $\mathsf{O}_1(x, w)$ returns $\perp$ (reject) if $(x, w) \notin \mathscr{R}_\rho$, and otherwise it returns $\mathsf{Sim}(\rho, \mathsf{crs}, \mathsf{td}, x)$.

$\mathsf{K}(\rho := [M]_1 \in \mathbb{G}_1^{n\times m})$**:** $A \leftarrow_\$ \hat{\mathscr{D}}_k$; $K \leftarrow_\$ \mathbb{Z}_p^{n\times \hat{k}}$; $C \leftarrow KA \in \mathbb{Z}_p^{n\times k}$; $[P]_1 \leftarrow [M]_1^\top K \in \mathbb{Z}_p^{m\times \hat{k}}$; $\mathsf{crs} \leftarrow ([A,C]_2, [P]_1)$; $\mathsf{td} \leftarrow K$; **return** $(\mathsf{td}, \mathsf{crs})$.

$\mathsf{P}([M]_1, \mathsf{crs}, [y]_1, w)$**:** $[\pi]_1 \leftarrow [P]_1^\top w \in \mathbb{G}_1^{\hat{k}}$.

$\mathsf{V}([M]_1, \mathsf{crs}, [y]_1, [\pi]_1)$**:** - **if** $[y]_1^\top [C]_2 = [\pi]_1^\top [A]_2$ **return** 1 **else return** 0.

$\mathsf{Sim}([M]_1, \mathsf{crs}, \mathsf{td}, [y]_1)$**:** $[\pi]_1 \leftarrow K^\top [y]_1 \in \mathbb{G}_1^{\hat{k}}$.

Figure 2. Kiltz-Wee QA-NIZK Π_{as} ($\hat{k} = k+1$ and $\hat{\mathscr{D}}_k = \mathscr{D}_k$) and Π'_{as} ($\hat{k} = k$ and $\hat{\mathscr{D}}_k = \bar{\mathscr{D}}_k$) [106].

(iii) **Computational Soundness.** For any PPT $\mathscr{A}$,

$$\Pr\left[\begin{array}{l}\mathsf{p} \leftarrow \mathsf{Pgen}(1^\lambda); \rho \leftarrow_\$ \mathscr{D}_{\mathsf{p}}; (\mathsf{crs}, \mathsf{td}) \leftarrow \mathsf{K}(\rho); \\ (x, \pi) \leftarrow \mathscr{A}(\rho, \mathsf{crs}) : \\ \mathsf{V}(\rho, \mathsf{crs}, x, \pi) = 1 \wedge \neg(\exists w : (x, w) \in \mathscr{R}_\rho)\end{array}\right] \approx_\lambda 0 \ .$$

Kiltz-Wee's QA-NIZK Argument for Linear Spaces [106]. In this part we recall the most efficient constructions of QA-NIZK arguments of membership in linear spaces given by Kiltz and Wee [106] for the language

$$\mathscr{L}_{[M]_1} = \left\{ [y]_1 \in \mathbb{G}_1^n : \exists w \in \mathbb{Z}_p^m \text{ s.t. } y = Mw \right\} \ ,$$

where $\rho = [M]_1 \in \mathbb{G}_1^{n\times m}$. The corresponding relation is defined as $\mathscr{R}_{[M]_1} = \{([y]_1, w) \in \mathbb{G}_1^n \times \mathbb{Z}_p^m : y = Mw\}$. This language is useful in many applications (cf. [100] and follow up work). We recall the full construction of the Kiltz-Wee QA-NIZK arguments for linear subspaces in the CRS model in Fig. 2. Let $\mathscr{D}_k$ be the distribution is defined Definition 3. Let $\bar{\mathscr{D}}_k$ be taken as the uniform distribution over a set of all matrices in $\mathbb{Z}_p^{k\times k}$.

Theorem 2 (Theorem 1 of [106])**.** *If $\hat{\mathscr{D}}_k = \mathscr{D}_k$ and $\hat{k} = k+1$, Fig. 2 describes a QA-NIZK proof system Π_{as} with perfect completeness, computational adaptive soundness based on the $\mathscr{D}_k$-*KerMDH *assumption, perfect zero-knowledge, and proof size $k+1$.*

Theorem 3 (Theorem 2 of [106])**.** *If $\hat{\mathscr{D}}_k = \bar{\mathscr{D}}_k$, $\hat{k} = k$, the, and $\mathscr{D}_{\mathsf{p}}$ is a witness samplable distribution, Fig. 2 describes a QA-NIZK proof system Π_{as} with perfect completeness, computational adaptive soundness based on the $\mathscr{D}_k$-*KerMDH *assumption, perfect zero-knowledge, and proof size k.*

3. NIZKS WITH SUBVERTED SETUP

General Motivation for Subversion NIZKs. Zero-knowledge proofs were introduced by Goldwasser *et al.* [86]. A zero-knowledge proof is a protocol between a prover and a verifier that allows the prover to convince the verifier of the validity of a statement without revealing any more information. Zero-knowledge proofs and in particular *non-interactive zero-knowledge proofs* (NIZKs) are a critical primitive for building cryptographic protocols as they allow to have the correctness of some computations while valuing the privacy of the user. They play a pivotal role in both the practice and theory of cryptography and increasingly found their way into real-world applications. Some important applications are anonymous credentials [21, 46–49, 55, 78], electronic voting [63, 93, 129], and group signatures [18, 36, 37, 56, 66, 67]. In addition they are a core of verifiable computation [38, 81, 82, 126] and smart contracts [108]. Nowadays the use of some efficient NIZKs gets more attentions in real-world applications, more precisely in some systems such as Zcash [130] and Ethereum [44].

For the practical applications of NIZKs in the CRS model, an important issue is a need for trust in the generation of the CRS. In theory, one may simply assume that the CRS comes from some trusted party, but indeed in the real-word applications such a party is troublesome to implement. Recently, there has been an increasing interest to diminish trust in the generator of the CRS. One of this line of works is subversion zero-knowledge initiated by Bellare *et al.* in [22], where the zero-knowledge property even holds when the CRS is generated maliciously. They established several positive and negative results. More accurately, they proved that it is impossible to achieve simultaneously subversion soundness and zero-knowledge (even non-subversion). While subversion zero-knowledge (Sub-ZK) can be achieved. Following this initial work, in this chapter we investigate subversion version of two variants of NIZKs called zk-SNARK and QA-NIZK (respectively the results are published in [9] and [11]); and explain the intuition of constructing the Sub-ZK of the most efficient SNARKs [94] in Section 3.1, and the Sub-ZK of the most efficient QA-NIZK [106] in Section 3.5.

3.1. Subversion-Secure zk-SNARK

Motivation. As noted before, an important aspect of practical applications of zk-SNARKs is the challenge of the need of a trust for generating CRS. One way to deal with such an issue is to use the security notation of Bellare *et al.* [22] called subversion zero-knowledge where one can achieve zero-knowledge even when the CRS is subverted. Implementing this security notion signifies that the prover is not required to trust the CRS generator and so it diminishes the trust of one party in such systems. Following this idea, we investigate subversion zero-knowledge property of the most efficient zk-SNARKs [94].

3.2. Previous Works

The first NIZK with fast verification and small proof sizes (whose size is independent of its witness and the proven statement) was proposed by Groth [92] based on bilinear groups. Later it was improved by Lipmaa [113]. Gennaro *et al* [82] explicated how one can efficiently convert any boolean circuit into a Quadratic Span Program (QSP), and proposed a SNARK for QSPs whose proof is 8 group elements. They also proposed another construction based on quadratic arithmetic programs (QAP) [1]. In practice, usually QAP based constructions are more efficient and preferred. Later Parno *et al.* [126] modified the SNARK construction of Gennaro *et al* [82] by reducing the proof size by one group element and called it *Pinocchio*. The proof size of SNARK for boolean circuits was notably optimized by Danezis *et al* [64], by moving from QSP to square span programs (SSP), and constructed a system with the proof size contains only 4 group elements. Recently, Groth [94] finally proposed the most efficient SNARK, which is for QAP based arithmetic circuits and its proof contains only 3 group elements (and requires 3 pairings in verification phase).

3.3. Problem Statement

What are the requirements to have the most efficient zk-SNARK, Groth's zk-SNARK [94] in a subverted setup (when the CRS is maliciously generated) ?. More precisely, can we construct a subversion version of Groth's zk-SNARK [94] while it still remains succinct? What would be the complexity of the prover and the verifier of subversion version of Groth's zk-SNARK? What is the cost of having the subversion zero-knowledge property for such a construction?

[1] In QAPs, the inputs of the gates of the arithmetic circuits are elements from a finite field $\mathscr{F}$ and so the gates multiply or add field elements. Also since the circuit satisfiability is NP-complete, SNARKs cover all NP languages.

3.4. Our Solution

Following the result of Bellare *et al.* [22] (the existence of Sub-ZK of NIZKs), we start with defining Sub-zk-SNARKs and their new security characteristics. Then in order to achieve Sub-ZK, we define a new algorithm CV for checking whether the CRS is well-formed. We implement the new security properties and the algorithm CV for Groth's zk-SNARK [94] by modifying its CRS and defining the corresponding algorithm CV that checks the well-formness of the new CRS. More accurately, we investigate the new security definitions for Groth's zk-SNARK [94] with the following steps: (i) add some new elements in the CRS to make it publicly verifiable and trapdoor extractable, (ii) introduce a new algorithm CV which takes the modified CRS and checks whether it is well-formed, (iii) define the following new security definitions for Sub-zk-SNARK, *perfect subversion-complete* (this comprises the requirement that the CRS verification accepts an honestly generated CRS), *computationally knowledge-sound*, and *composable (or statistically unbounded) Sub-zk SNARKs*, (iv) prove that the Sub-ZK version of Groth's zk-SNARK [94] provides the aforementioned security characteristics. A full description of our construction of Sub-zk-SNARK is provided in the paper [9], which is joint work with Baghery, Lipmaa, and Zajac.

3.4.1. New Algorithm: CRS Verification CV

When we are operating on a subverted setup where the CRS is generated maliciously, the need for an algorithm confirming that the CRS is well-formed is crucial. More precisely in NIZKs (i.e, SNARKs) setting, before generating the proof, an honest prover P needs *at least* to validate the correctness of the CRS. In other words, one requires that the proof generated by an honest prover P with the subverted CRS will not disclose any information about her witness (Sub-ZK property). Therefore to preserve zero-knowledge in subverted setup, the existence of *efficient* CRS verification algorithm CV is vital.

For the sake of clarity, we assume that CV is a separate algorithm and not a part of P *algorithm*. So that, an honest prover first checks the CRS by running the CV algorithm, if it accepts then runs the P *algorithm*. Such separation also is valuable in practice where one uses the same CRS and executes many zero-knowledge arguments; it makes sense that the prover only executes CV once. We perceive that an honest verifier does not necessitate to run the CRS verification algorithm since we are not aiming to deliver subversion (knowledge-)soundness.

Thus a subversion-resistant *NIZK argument system* Ψ for the relation $\mathscr{R}$ consists of the following five PPT algorithms K, P, V, Sim, and CV. The first four algorithms are similar to the ones in NIZK argument systems presented in Section 2.11.2. But the last one is a new algorithm that takes crs as input and outputs 1 if it is well-formed; otherwise outputs 0.

In the next subsection we use such CV algorithm to define the new security properties of Sub-zk-SNARKs.

3.4.2. Security Definition of Sub-zk-SNARK

In general, a Sub-zk-SNARK must provide the following security definitions: *perfect subversion-completeness* (a legitimate prover convinces a legitimate verifier, and an honestly generated CRS passes the CRS verification CV) *computational knowledge-soundness* (if a prover convinces a legitimate verifier, then he must know the corresponding witness), and *statistical Sub-ZK* (given a possibly subverted CRS, a proof generated by the legitimate prover exposes no side information). We note that in Sub-ZK definition we acknowledge the existence of efficient subverter and a computationally unbounded distinguisher. Furthermore, we consider the case when the subverter is unbounded and call it statistically unbounded Sub-ZK. In addition, following the definition of statistical composable ZK [91] (ZK is guaranteed against an adversary who has to distinguish a single argument from a simulated one), we define a stronger version of statistical Sub-ZK called *statistical composable Sub-ZK*.

We define the security definitions of Sub-ZK SNARK based non subversion-resistant security definitions from [91,95]. For the sake of simplicity, we *emphasize* the differences between subversion and non-subversion definitions. Similarly to non-subversion definitions mentioned in Section 2.11.2, we let all legitimate parties obtain $\mathsf{aux}_{\mathscr{R}}$ as an input. We split the CRS generation K algorithm into the following three algorithms, K_{td}, $\mathsf{K}_{\mathtt{tds}}$, and $\mathsf{K}_{\mathsf{crs}}$ where respectively output the CRS's trapdoor td, the simulation trapdoor $\mathtt{tds}$, and the CRS crs.[2] Additionally, we split the CRS into three parts indicating $\mathsf{crs} = (\mathsf{crs}_{\mathsf{P}}, \mathsf{crs}_{\mathsf{V}}, \mathsf{crs}_{\mathsf{CV}})$ where $\mathsf{crs}_{\mathsf{P}}$ (resp., $\mathsf{crs}_{\mathsf{V}}$) is the part of the CRS given to a legitimate prover (resp., a legitimate verifier), and $\mathsf{crs}_{\mathsf{CV}}$ is the part of CRS not required by the prover or the verifier except to run CV algorithm. Such splitting does not influence the security proof but in many instances, $\mathsf{crs}_{\mathsf{V}}$ is significantly shorter than $\mathsf{crs}_{\mathsf{P}}$.

Definition 14 (Perfect Subversion-Completeness)**.** *A non-interactive argument* Ψ *is* perfectly subversion-complete for RGen*, if for all* λ*,* $(x,w) \in \mathscr{R}$*,* $(\mathscr{R}, \mathsf{aux}_{\mathscr{R}}) \in \mathrm{range}(\mathsf{RGen}(1^{\lambda}))$*, and* $\mathsf{td} \in \mathrm{range}(\mathsf{K}_{\mathsf{td}}(\mathscr{R}, \mathsf{aux}_{\mathscr{R}})) \setminus \{\perp\}$*,*

$$\Pr\left[\begin{array}{l}\mathsf{crs} \leftarrow \mathsf{K}_{\mathsf{crs}}(\mathscr{R}, \mathsf{aux}_{\mathscr{R}}, \mathsf{td}) : \mathsf{CV}(\mathscr{R}, \mathsf{aux}_{\mathscr{R}}, \mathsf{crs}) = 1 \wedge \\ \mathsf{V}(\mathscr{R}, \mathsf{aux}_{\mathscr{R}}, \mathsf{crs}_{\mathsf{V}}, x, \mathsf{P}(\mathscr{R}, \mathsf{aux}_{\mathscr{R}}, \mathsf{crs}_{\mathsf{P}}, x, w)) = 1\end{array}\right] = 1 \ .$$

Definition 15 (Computational Knowledge-Soundness [95])**.** Ψ *is computationally (adaptively)* knowledge-sound for RGen*, if for every non-uniform PPT* $\mathscr{A}$*, there exists a non-uniform PPT extractor* $\mathsf{Ext}_{\mathscr{A}}$*,*

$$\Pr\left[\begin{array}{l}(\mathscr{R}, \mathsf{aux}_{\mathscr{R}}) \leftarrow \mathsf{RGen}(1^{\lambda}), (\mathsf{crs} \,\|\, \mathtt{tds}) \leftarrow \mathsf{K}(\mathscr{R}, \mathsf{aux}_{\mathscr{R}}), \\ r \leftarrow_{\$} \mathsf{RND}_{\lambda}(\mathscr{A}), ((x,\pi) \,\|\, w) \leftarrow (\mathscr{A} \,\|\, \mathsf{Ext}_{\mathscr{A}})(\mathscr{R}, \mathsf{aux}_{\mathscr{R}}, \mathsf{crs}; r) : \\ (x,w) \notin \mathscr{R} \wedge \mathsf{V}(\mathscr{R}, \mathsf{aux}_{\mathscr{R}}, \mathsf{crs}_{\mathsf{V}}, x, \pi) = 1\end{array}\right] \approx_{\lambda} 0 \ .$$

[2] Intuitively Groth's CRS generation algorithm returns $\mathsf{K}_{\mathsf{crs}}(\mathscr{R}, \mathsf{aux}_{\mathscr{R}}, \mathsf{K}_{\mathsf{td}}(\mathscr{R}, \mathsf{aux}_{\mathscr{R}}))$ as the CRS and $\mathsf{K}_{\mathtt{tds}}(\mathscr{R}, \mathsf{aux}_{\mathscr{R}}, \mathsf{K}_{\mathsf{td}}(\mathscr{R}, \mathsf{aux}_{\mathscr{R}}))$ as the simulation trapdoor

Where, $\mathsf{aux}_{\mathscr{R}}$ can be viewed as a common auxiliary input to $\mathscr{A}$ and $\mathsf{Ext}_{\mathscr{A}}$ that is generated by using a benign [32] relation generator ($\mathsf{aux}_{\mathscr{R}}$ could be the description of a secure bilinear group). A knowledge-sound argument system is called an *argument of knowledge*.

Next, we define *statistically unbounded Sub-ZK* such that in contrast to statistically unbounded ZK [90], the CRS crs is generated by a subverter Z who also returns $\mathsf{aux}_{\mathsf{Z}}$. Besides, the adversary $\mathscr{A}$ has access to $\mathsf{aux}_{\mathsf{Z}}$ meaning that $\mathscr{A}$ can cooperate with Z. Also, the extractor $\mathsf{Ext}_{\mathsf{Z}}$ can extract td from Z, and it will be used to compute the simulation trapdoor $\mathtt{tds}$ that is then given as an auxiliary input to $\mathscr{A}$ and the oracle O_1.

Definition 16 (Statistically Unbounded Sub-ZK)**.** *Ψ is* statistically unbounded Sub-ZK for RGen*, if for any PPT subverter Z there exists a PPT $\mathsf{Ext}_{\mathsf{Z}}$, such that for all $(\mathscr{R},\mathsf{aux}_{\mathscr{R}}) \in \mathsf{range}(\mathsf{RGen}(1^{\lambda}))$ and all computationally unbounded $\mathscr{A}$, $\varepsilon_0^{unb} \approx_{\lambda} \varepsilon_1^{unb}$, where ε_b^{unb} is defined as*

$$\Pr\left[\begin{array}{l} r \leftarrow_{\$} \mathsf{RND}_{\lambda}(\mathsf{Z}), (\mathsf{crs}, \mathsf{aux}_{\mathsf{Z}} \,\|\, \mathsf{td}) \leftarrow (\mathsf{Z} \,\|\, \mathsf{Ext}_{\mathsf{Z}})(\mathscr{R}, \mathsf{aux}_{\mathscr{R}}; r), \\ \mathtt{tds} \leftarrow \mathsf{K}_{\mathtt{tds}}(\mathscr{R}, \mathsf{aux}_{\mathscr{R}}, \mathsf{td}) : \mathsf{CV}(\mathscr{R}, \mathsf{aux}_{\mathscr{R}}, \mathsf{crs}) = 1 \wedge \\ \mathscr{A}^{\mathsf{O}_b(\cdot,\cdot)}(\mathscr{R}, \mathsf{aux}_{\mathscr{R}}, \mathsf{crs}, \mathtt{tds}, \mathsf{aux}_{\mathsf{Z}}) = 1 \end{array}\right].$$

Here, the oracle $\mathsf{O}_0(x,w)$ returns $\perp$ (reject) if $(x,w) \notin \mathscr{R}$, and otherwise it returns $\mathsf{P}(\mathscr{R}, \mathsf{aux}_{\mathscr{R}}, \mathsf{crs}_{\mathsf{P}}, x, w)$. Similarly, $\mathsf{O}_1(x,w)$ returns $\perp$ (reject) if $(x,w) \notin \mathscr{R}$, and otherwise it returns $\mathsf{Sim}(\mathscr{R}, \mathsf{aux}_{\mathscr{R}}, \mathsf{crs}, \mathtt{tds}, x)$. Ψ is perfectly unbounded Sub-ZK for RGen *if one requires that $\varepsilon_0^{unb} = \varepsilon_1^{unb}$.*

Finally, we define statistical composable Sub-ZK based on the definition of (computational) composable ZK from [90] but with some fundamental differences. More accurately, [91] defined the two properties: (i) *reference string indistinguishability*, meaning that the CRS created by a *valid* K is indistinguishable with the one simulated by a simulator $\mathsf{Sim}_{\mathsf{crs}}$, and (ii) *simulation indistinguishability* meaning that the proof created by a *legitimate* prover P should be indistinguishable with the one simulated by a simulator Sim.

In Sub-ZK case, since the CRS created by a subverter Z (in both the real and the simulated cases), therefore our statistical composable Sub-ZK is analogous to the definition of simulation indistinguishability but instead of simulating the CRS, we use the CRS crs generated by Z, assume that an extractor $\mathsf{Ext}_{\mathsf{Z}}$ extracts td from crs, calculate $\mathtt{tds}$ from td by using $\mathsf{K}_{\mathtt{tds}}$, and finally check that CV accepts crs.

Definition 17 (Statistically Composable Sub-ZK)**.** *Ψ is* statistically composable Sub-ZK for RGen*, if for any PPT subverter Z there exists a PPT $\mathsf{Ext}_{\mathsf{Z}}$, such that for all $(\mathscr{R}, \mathsf{aux}_{\mathscr{R}}) \in \mathsf{range}(\mathsf{RGen}(1^{\lambda}))$, all $(x,w) \in \mathscr{R}$, and all computationally*

// $i=1$
$[\zeta]_T \leftarrow ([\chi]_1[\chi^{n-1}]_2 - [1]_T)/n; [\omega']_2 \leftarrow [1]_2;$
if $[\chi]_2 = [\omega']_2$ **then** check $[a_1]_1 = [1]_1$;
else check $[a_1]_1([\chi]_2 - [\omega']_2) =^? [\zeta]_T$;
for $i = 2$ **to** n **do**
$[\zeta]_T \leftarrow \omega[\zeta]_T$; $[\omega']_2 \leftarrow \omega[\omega']_2$;
if $[\chi]_2 = [\omega']_2$ **then** check $[a_i]_1 =^? [1]_1$;
else check $[a_i]_1([\chi]_2 - [\omega']_2) =^? [\zeta]_T$; **endfor**

Figure 3. The algorithm $\mathsf{checkLag}([\chi, (a_i)_{i=1}^n]_1, [1, \chi, \chi^{n-1}]_2, [1]_T)$ for checking that $[a_i]_1 = [\ell_i(\chi)]_1$ for $i \in [n]$.

unbounded $\mathscr{A}$, $\varepsilon_0^{comp} \approx_\lambda \varepsilon_1^{comp}$, *where* ε_b^{comp} *is defined as*

$$\Pr\left[\begin{array}{l} r \leftarrow_\$ \mathsf{RND}_\lambda(\mathsf{Z}), (\mathsf{crs}, \mathsf{aux}_\mathsf{Z} \,\|\, \mathtt{td}) \leftarrow (\mathsf{Z} \,\|\, \mathsf{Ext}_\mathsf{Z})(\mathscr{R}, \mathsf{aux}_{\mathscr{R}}; r), \\ \mathtt{tds} \leftarrow \mathsf{K}_{\mathtt{tds}}(\mathscr{R}, \mathsf{aux}_{\mathscr{R}}, \mathtt{td}), \textbf{ if } b = 0 \textbf{ then } \pi \leftarrow \mathsf{P}(\mathscr{R}, \mathsf{aux}_{\mathscr{R}}, \mathsf{crs}_\mathsf{P}, x, w) \\ \textbf{else } \pi \leftarrow \mathsf{Sim}(\mathscr{R}, \mathsf{aux}_{\mathscr{R}}, \mathsf{crs}, \mathtt{tds}, x) \textbf{ fi} : \\ \mathsf{CV}(\mathscr{R}, \mathsf{aux}_{\mathscr{R}}, \mathsf{crs}) = 1 \wedge \mathscr{A}(\mathscr{R}, \mathsf{aux}_{\mathscr{R}}, \mathsf{crs}, \mathtt{tds}, \pi, \mathsf{aux}_\mathsf{Z}) = 1 \end{array}\right].$$

Ψ *is* perfectly composable Sub-ZK for RGen *if one requires that* $\varepsilon_0^{comp} = \varepsilon_1^{comp}$.

We note that in contrast to the definition of composable (non-Sub) ZK in [95], in composable Sub-ZK the CRS crs is created by a subverter Z who also returns aux_Z. Besides, the adversary $\mathscr{A}$ has access to aux_Z meaning that $\mathscr{A}$ can cooperate with Z. Also the extractor Ext_Z can extract $\mathtt{td}$ from Z, and it will be used to compute the simulation trapdoor $\mathtt{tds}$ that is then given as an auxiliary input to $\mathscr{A}$.

(i) first split the CRS generation algorithm into the three algorithms, $\mathsf{K}_\mathtt{td}$, $\mathsf{K}_\mathtt{tds}$, and K_crs. intuitively the CRS generation algorithm of Groth's zk-SNARK [94] returns $\mathsf{K}_\mathsf{crs}(\mathscr{R}, \mathsf{aux}_{\mathscr{R}}, \mathsf{K}_\mathtt{td}(\mathscr{R}, \mathsf{aux}_{\mathscr{R}}))$ as the CRS and $\mathsf{K}_\mathtt{tds}(\mathscr{R}, \mathsf{aux}_{\mathscr{R}}, \mathsf{K}_\mathtt{td}(\mathscr{R}, \mathsf{aux}_{\mathscr{R}}))$ as the simulation trapdoor, (ii) next add to the CRS $2n+3$ new elements (see the variable crs_CV in Fig. 4) that are needed for CV algorithm to work efficiently. Also for sake of simplicity, we split the CRS $\mathsf{crs} = (\mathsf{crs}_\mathsf{P}, \mathsf{crs}_\mathsf{V}, \mathsf{crs}_\mathsf{CV})$, where the crs_CV carries the new elements added in the CRS. (iii) then describe an efficient CRS verification algorithm CV (see Fig. 4), and (iv) replace Groth's CRS generation algorithm with the CRS generation algorithms, $\mathsf{K}_\mathtt{td}$, $\mathsf{K}_\mathtt{tds}$, and K_crs; and attach the CV algorithm as a new algorithm to Groth's zk-SNARK [94].

3.4.3. Sub-zk-SNARK Construction

In this part, we implement the aforementioned CRS verification algorithm CV for the most efficient zk-SNARK [94]. We give a concrete construction of Sub-ZK version of zk-SNARK [94] and depict the full construction in Fig. 4. In particular, we modify Groth's zk-SNARK [94] with the following steps:

Note that for the performance of CV algorithm, we add the Lagrange polynomials $\ell_i(.)$ evaluated in the trapdoor χ in the CRS. See Fig. 5 for the procedure

$\mathsf{K}_{\mathsf{td}}(\mathscr{R}, \mathsf{aux}_{\mathscr{R}})$: Generate $\mathsf{td} = (\chi, \alpha, \beta, \gamma, \delta) \leftarrow_{\$} \mathbb{Z}_p^3 \times (\mathbb{Z}_p^*)^2$.
$\mathsf{K}_{\mathtt{tds}}(\mathscr{R}, \mathsf{aux}_{\mathscr{R}}, \mathsf{td})$: Set $\mathtt{tds} \leftarrow (\chi, \alpha, \beta, \delta)$.
$\mathsf{K}_{\mathsf{crs}}(\mathscr{R}, \mathsf{aux}_{\mathscr{R}}, \mathsf{td})$: Compute $(\ell_i(\chi))_{i=1}^n$ by using the algorithm in Fig. 5. Set $u_j(\chi) \leftarrow \sum_{i=1}^n U_{ij}\ell_i(\chi)$, $v_j(\chi) \leftarrow \sum_{i=1}^n V_{ij}\ell_i(\chi)$, $w_j(\chi) \leftarrow \sum_{i=1}^n W_{ij}\ell_i(\chi)$ for all $j \in [m]$. Let

$$\mathsf{crs}_{\mathsf{P}} \leftarrow \left(\begin{array}{l} \left[\alpha, \beta, \delta, \left(\frac{u_j(\chi)\beta + v_j(\chi)\alpha + w_j(\chi)}{\delta} \right)_{j=m_0+1}^{m} \right]_1, \\ \left[(\chi^i \ell(\chi)/\delta)_{i=0}^{n-2}, (u_j(\chi), v_j(\chi))_{j=0}^{m} \right]_1, \left[\beta, \delta, (v_j(\chi))_{j=0}^{m} \right]_2 \end{array} \right) ,$$

$$\mathsf{crs}_{\mathsf{V}} \leftarrow \left(\left[\left(\frac{u_j(\chi)\beta + v_j(\chi)\alpha + w_j(\chi)}{\gamma} \right)_{j=0}^{m_0} \right]_1, [\gamma, \delta]_2, [\alpha\beta]_T \right) ,$$

$$\mathsf{crs}_{\mathsf{CV}} \leftarrow \left([\gamma, (\chi^i)_{i=1}^{n-1}, (\ell_i(\chi))_{i=1}^{n}]_1, [\alpha, \chi, \chi^{n-1}]_2 \right) .$$

Return $\mathsf{crs} \leftarrow (\mathsf{crs}_{\mathsf{CV}}, \mathsf{crs}_{\mathsf{P}}, \mathsf{crs}_{\mathsf{V}})$.
$\mathsf{K}(\mathscr{R}, \mathsf{aux}_{\mathscr{R}})$: let $\mathsf{td} \leftarrow \mathsf{K}_{\mathsf{td}}(\mathscr{R}, \mathsf{aux}_{\mathscr{R}})$.
Return $(\mathsf{crs} \,\|\, \mathtt{tds}) \leftarrow (\mathsf{K}_{\mathsf{crs}} \,\|\, \mathsf{K}_{\mathtt{tds}})(\mathscr{R}, \mathsf{aux}_{\mathscr{R}}, \mathsf{td})$.
$\mathsf{CV}(\mathscr{R}, \mathsf{aux}_{\mathscr{R}}, \mathsf{crs})$:

1. For $\iota \in \{\gamma, \delta\}$: check that $[\iota]_1 \neq [0]_1$
2. For $\iota \in \{\alpha, \beta, \gamma, \delta\}$: check that $[\iota]_1 [1]_2 = [1]_1 [\iota]_2$,
3. For $i = 1$ to $n-1$: check that $\left[\chi^i\right]_1 [1]_2 = \left[\chi^{i-1}\right]_1 [\chi]_2$,
4. Check that $([\ell_i(\chi)]_1)_{i=1}^n$ is correctly computed by using the algorithm in Fig. 3,
5. For $j = 0$ to m:
   (a) Check that $[u_j(\chi)]_1 = \sum_{i=1}^n U_{ij} [\ell_i(\chi)]_1$,
   (b) Check that $[v_j(\chi)]_1 = \sum_{i=1}^n V_{ij} [\ell_i(\chi)]_1$,
   (c) Set $[w_j(\chi)]_1 \leftarrow \sum_{i=1}^n W_{ij} [\ell_i(\chi)]_1$,
   (d) Check that $[v_j(\chi)]_1 [1]_2 = [1]_1 [v_j(\chi)]_2$,
6. For $j = m_0 + 1$ to m: check that $[(u_j(\chi)\beta + v_j(\chi)\alpha + w_j(\chi))/\delta]_1 [\delta]_2 = [u_j(\chi)]_1 [\beta]_2 + [v_j(\chi)]_1 [\alpha]_2 + [w_j(\chi)]_1 [1]_2$,
7. Check that $\left[\chi^{n-1}\right]_1 [1]_2 = [1]_1 \left[\chi^{n-1}\right]_2$,
8. For $i = 0$ to $n-2$: check that $\left[\chi^i \ell(\chi)/\delta\right]_1 [\delta]_2 = \left[\chi^{i+1}\right]_1 \left[\chi^{n-1}\right]_2 - \left[\chi^i\right]_1 [1]_2$,
9. Check that $[\alpha]_1 [\beta]_2 = [\alpha\beta]_T$.

$\mathsf{P}(\mathscr{R}, \mathsf{aux}_{\mathscr{R}}, \mathsf{crs}_{\mathsf{P}}, x = (A_1, \ldots, A_{m_0}), w = (A_{m_0+1}, \ldots, A_m))$: Works similar to the one of Groth's SNARK [94].
$\mathsf{V}(\mathscr{R}, \mathsf{aux}_{\mathscr{R}}, \mathsf{crs}_{\mathsf{V}}, x = (A_1, \ldots, A_{m_0}), \pi = ([\mathsf{a}]_1, [\mathsf{b}]_2, [\mathsf{c}]_1))$: Works similar to the one of Groth's SNARK [94].

Figure 4. The full construction of Sub-zk-SNARK.

$\zeta \leftarrow (\chi^n - 1)/n; \omega' \leftarrow 1;$
if $\chi = \omega'$ **then** $\ell_1(\chi) \leftarrow 1$; **else** $\ell_1(\chi) \leftarrow \zeta/(\chi - \omega')$; **fi**
  for $i = 2$ **to** n **do**
  $\zeta \leftarrow \omega\zeta; \omega' \leftarrow \omega\omega';$
  if $\chi = \omega'$ **then** $\ell_i(\chi) \leftarrow 1$; **else** $\ell_i(\chi) \leftarrow \zeta/(\chi - \omega')$; **fi endfor**

Figure 5. The algorithm $\mathsf{compLag}(\chi, n)$ for computing $(\ell_i(\chi))_{i=1}^n$

of the Lagrange generation algorithm. Thus one can efficiency check the well-formness of the polynomials $u_j(\chi)$, $v_j(\chi)$, and $w_j(\chi)$ of QAPs instance in CV algorithm.

3.4.4. Efficiency of Sub-zk-SNARK

In this section we investigate the performance of Sub-zk-SNARK. As presented earlier, we added some new elements in different groups to the CRS of [94] to make it publicly verifiable (by executing CV algorithm). We describe the CRS length of Sub-zk-SNARK with more details in Table 1. The total size of the CRS contains $4m + 3n + 13$ group elements.[3]

We recall that the CRS of Groth's zk-SNARK [95] consists of $m + 2n$ and n elements respectively in $\mathbb{G}_1$ and $\mathbb{G}_2$.

Computational complexity of CRS generation. Suppose gk has been already computed. We create crs by first generating the discrete logarithms of all CRS elements, and then their versions in $\mathbb{G}_\iota$ for $\iota \in \{1,2\}$. One can evaluate $u_j(\chi)$, $v_j(\chi)$, and $w_j(\chi)$ for each $j \in [m]$ in time $\Theta(n)$ by using precomputed values $\ell_i(\chi)$ for $i \in [n]$ and the fact that the matrices U,V,W contain $\Theta(n)$ non-zero elements. (The latter is a standard assumption, already made in [82].) The rest of the CRS can be created efficiently by using straightforward algorithms. By counting the compLag algorithm in Fig. 5, the whole CRS generation algorithm is dominated by $3m + 2n + 3$ exponentiations in $\mathbb{G}_1$, $m + 5$ exponentiations in $\mathbb{G}_2$, 1 exponentiation in $\mathbb{G}_T$ (thus, one exponentiation per each CRS element), and $\Theta(n)$ multiplications/divisions in $\mathbb{Z}_p$. Thus based on Fig. 5, the whole CRS generation algorithm is dominated by $3m + 3n + 5$ exponentiations in $\mathbb{G}_1$, $m + 7$ exponentiations in $\mathbb{G}_2$, and 1 exponentiation in $\mathbb{G}_T$ (one per CRS element) and $\Theta(n)$ multiplications/divisions in $\mathbb{Z}_p$.

CV's computational complexity. Suppose subverting gk is difficult; this holds by assuming that the SNARKs use well-known bilinear groups (i.e, the Barreto-Naehrig curves). In the the CRS verification algorithm in Fig. 4, we observe that all steps but step 4 can be efficiently computed in $\Theta(n)$ cryptographic operations.

[3] One element (namely, $[\delta]_2$) is both in crs_V and crs_P, and so the numbers in the *total* row are not equal to the sum of the numbers in previous rows.

	$\mathbb{G}_1$	$\mathbb{G}_2$	$\mathbb{G}_T$	Total
crs_P	$3m+n-m_0+4$	$m+3$	0	$4m+n-m_0+7$
crs_V	m_0+1	2	1	m_0+4
crs_CV	$2n$	3	0	$2n+3$
Total	$3m+3n+5$	$m+7$	1	$4m+3n+13$

Table 1. The CRS length of Sub-zk-SNARK.

More precisely its computation needs $6m+5n-4m_0+8$ pairings. Besides, one needs to compute $s(U)+s(V)+s(W)$ exponentiations in $\mathbb{G}_1$, where $s(M)$ is the number of "large" (i.e., large enough so that exponentiating with them is expensive) entries in the matrix M. Often, $s(M)$ are very small.

For the algorithm step 4, one needs $n-1$ exponentiations in $\mathbb{G}_2$, n exponentiations in $\mathbb{G}_T$, and $n+1$ pairings. Thus, the total complexity of the CV algorithm is dominated by $s(U)+s(V)+s(W)$ exponentiations in $\mathbb{G}_1$, $n-1$ exponentiations in $\mathbb{G}_2$, n exponentiations in $\mathbb{G}_T$, and $6m+6n-4m_0+9$ pairings. [4]

Prover's and verifier's computational complexity. Since the prover and verifier in Sub-zk-SNARK in Fig. 4 remain unchanged compare with the ones in Groth's zk-SNARK [95], their complexity is the same as Groth's zk-SNARK construction. More accurately, the prover demands to compute $h(X)$ (3 interpolations, 1 polynomial multiplication, and 1 polynomial division; in total $\Theta(n\log n)$ non-cryptographic operations in $\mathbb{Z}_p$), and $(n-1)+(s(A)+1)+1+(s(A)+1)+s(A_1,\ldots,A_{m_0})\leq n+3s(A)+2$ exponentiations in $\mathbb{G}_1$ and $s(A)+1$ exponentiations in $\mathbb{G}_2$, where $s(A)$ is the number of large elements in A (i.e., large enough so that exponentiating with them would be expensive). Thus, the whole computational complexity of prover is $\Theta(n)$ cryptographic operations and $\Theta(n\log n)$ non-cryptographic operations. In the verifier side, one needs to check a single pairing equation that takes m_0 exponentiations in $\mathbb{G}_1$ and 3 pairings. The exponentiations can be computed offline since they are independent of the argument π and only depends on the common input $(A_1,\ldots,A_{m_0})$. Hence, the whole online computational complexity of verifier is dominated only by 3 pairings.

Communication complexity. Similarly to Groth's zk-SNARK [95], the argument of Sub-zk-SNARK contains 2 elements in $\mathbb{G}_1$ and 1 element in $\mathbb{G}_2$.

[4] One can speed up CV by using batching technique [23]. Namely, clearly if $\sum_{i=1}^{s} t_i([a_i]_1\,[b_i]_2)=[c]_T$ for uniformly random t_i, then w.h.p., $[a_i]_1\,[b_i]_2=[c]_T$ for each individual $i\in[s]$. The speed up follows from the use of bilinear properties and from the fact that exponentiation is faster than pairing [73, 114]. Full batched version of CV is described in the full version [8, 12]

3.5. Subversion-Secure QA-NIZK

Motivation. Another important direction with the NIZKs is quasi-adaptive NIZKs (QA-NIZKs) introduced by Jutla and Roy [100]. In a QA-NIZK construction, the CRS depends on some language parameter `lpar` where `lpar` can be considered a correctly distributed public key of some cryptosystem. The first QA-NIZK [100] and also most of the other constructions in this line [2, 87, 100, 102, 106, 110] are known for linear subspace languages and have only been constructed in CRS model. The dependency between CRS and language parameter makes QA-NIZKs very efficient and they have a lot of applications such as signatures, commitments, PAKE schemes [2, 100, 102, 106, 110] etc. Besides, Fauzi *et al.* [72] coupled QA-NIZKs and SNARKs for linear subspaces to build efficient shuffle argument systems. Quite recently Campanelli *et al.* [50] used QA-NIZKs for constructing a new toolbox called LegoSNARK that enables one to construct complex zk-SNARKs, so-called commit-and-prove SNARKs (CP-SNARKs). In particular, QA-NIZK plays an influential role as a building block in several CP-SNARKs in Campanelli *et al.*'s system [50].

All the above QA-NIZK based constructions are in the CRS model where it brings the crucial need for having a trusted party to generate the CRS. Comparing with the subversion zk-SNARKs (in section Section 3.1), building Sub-ZK QA-NIZKs is more complicated since the dependency between CRS and language parameter `lpar`. Due to the broad applicability of QA-NIZKs in designing of various cryptographic primitives, we investigate QA-NIZKs in subverted setup, define corresponding security definitions, and finally explicate how one can construct a subversion zero-knowledge version of the most efficient QANIZKs [106].

3.6. Previous Works

QA-NIZKs were introduced by Jutla and Roy in [100]. For linear languages $\mathscr{L}$ (linear subspaces of vector spaces of bilinear groups), they have explicated that one can build more efficient computationally-sound NIZK proofs (so-called arguments) in a slightly different quasi-adaptive setting (where CRS depends of the language parameter)[5]. Compared to Groth-Sahai proofs [98], QA-NIZKs in [100] are established for linear languages $\mathscr{L}$ which can be used for many cryptographic primitives and real-world applications. In 2014, Julta and Roy in [102] gave an enhanced version of QA-NIZK where the size of the proof was independent of the number of equations and variables and so with a constant-size proof. In this line of research, there has been some efforts [2, 87, 100, 102, 106, 110] for constructing more efficient pairing-based QA-NIZKs in CRS model. Finally, Jutla

[5] In the quasi-adaptive setting, a class of parameterized languages $\{\texttt{lpar}\}$ is considered, parameterized by `lpar`, and the CRS generator is permitted to generate the CRS based on the language parameter `lpar`.

and Roy [102], and Kiltz and Wee [106] constructed the most efficient QA-NIZK with a constant-size proof (in the most optimized case of their construction, the proof size is only 1 group element) in the CRS model. Besides the linear languages, QA-NIZKs were constructed for some other languages like as the language of bit-strings [87] and the ones for shuffles [88] but they are inefficient and their CRS have quadratic-length. In addition Daza *et al.*'s [65] newly advanced QA-NIZK [65] for NP-complete SSP [64] that relies on non-succinct commitment. But due to the performance and the applications, the research of QA-NIZK is mainly focused on linear subspaces.

In the line of diminishing the trust in QA-NIZKs, Charanjit et al. [101] investigated the case of a subverted language parameter `lpar` but *honestly generated CRS*) (in an updated full version of [100] from September 2018) and explicated how zero-knowledge and soundness can be achieved for a large family of language subspaces in such setting. Abdolmaleki *et al.* [11] (the contribution of this section) studied QA-NIZKs in a more general subverted setup where both CRS and language parameters are subverted and built a subversion zero-knowledge version of Kiltz-Wee's [106] construction. Quite recently Lipmaa [115] proposed an updatable version of our construction [11].

3.7. Problem Statement

Can one obtain Sub-ZK QA-NIZKs for linear subspaces? what is the relation between Sub-ZK QA-NIZKs in the CRS model and QA-NIZKs in the BPK model [54, 120]? Are soundness and zero-knowledge achievable when both `lpar` and the CRS are subverted?

3.8. Our Solution

First, we observe that Sub-ZK in the CRS model corresponds to no-auxiliary-string non-black-box NIZK in BPK model [54, 120]. By the impossibility result of [22], we discern that one cannot achieve both soundness and zero-knowledge in the case both `lpar` and the CRS have been subverted. Following that, we essentially concentrate on the slightly more relaxed case when soundness holds if only language parameter `lpar` has been subverted (Sub-PAR) and zero-knowledge holds when both `lpar` and the CRS have been subverted (Sub-ZK). As the QA-NIZKs have a more complex structure compared to SNARKs [6], one can not employ the knowledge assumptions of [9, 76] or knowledge-of-exponent assumptions [61] immediately translated to the case of (Kiltz-Wee) QA-NIZK. We first define QA-NIZK in the BPK Model [7], and introduce a new algorithm PKV check-

[6], For example, the most efficient known QA-NIZK for linear subspaces by Kiltz and Wee [106] has a trapdoor matrix K but $[K]_1$ is not explicitly supplied in the CRS

[7]In the BPK model, we assume that the verifier has a public key pk; the key authority $\mathscr{R}_{id}$ performs the functionality of a bulletin board by collecting pk.

ing whether the language parameter lpar and the public key pk (correspond to the CRS in CRS model) are well-formed. Then we define new security definitions QA-NIZK arguments in the BPK model: (i) *persistent zero-knowledge*, meaning the zero-knowledge property when both lpar and pk have been subverted (Sub-ZK), and (ii) *Sub-PAR soundness* meaning the soundness property when lpar has been subverted. Besides, we define *Sub-PAR knowledge soundness* of QA-NIZK arguments in the BPK model.

Finally, to deliver a concrete Sub-ZK QA-NIZK, we implement the above security definitions together with the new algorithm PKV for the most-efficient known QA-NIZK by Kiltz and Wee [106] for linear subspaces. A full description of our construction is given in the paper [11], which is joint work with Lipmaa, Siim and Zajac.

3.8.1. Subversion ZK in the CRS Model and ZK in the BPK Model

A zero-knowledge argument in the BPK model can be perceived as either designated-verifier (the proof convinces only the designated verifier) when the public key pk is generated by the verifier or transferable (the verifier can transfer the argument to other verifiers and convince them of its legality) when pk is generated by a third party. We note that by viewing pk as the CRS, the latter case is equivalent to the CRS model and so the BPK model is significantly weaker than the CRS model.

Intuitively we first observe that *no-auxiliary-string non-black-box* zero knowledge [85] in the BPK model [54, 120] is equivalent to Sub-ZK in the CRS model, as defined in [9, 22, 76]. Such an important connection (between Sub-ZK and no-auxiliary-string non-black-box zero knowledge) was not contemplated in the prior work on the Sub-ZK area and might be useful in interpreting the future Sub-ZK systems. Due to such connection, we use the notation Sub-ZK to denote no-auxiliary-string non-black-box zero knowledge in the BPK model.

Additionally due to the impossibility results [85] of having three rounds auxiliary-string zero-knowledge (and so auxiliary-string non-black-box NIZK) in the plain model, in lemma 1 of [11] we conclude that one only can construct *no-auxiliary*-string non-black-box NIZK for languages outside of BPP. Thus in the rest of this section by QA-NIZK in BPK model we mean that it is no-auxiliary-string[8] non-black-box NIZK but to ease of readability we drop it.

3.8.2. Defining QA-NIZK in the BPK Model

A QA-NIZK argument is an important tool for proving membership in a language defined by a relation $\mathscr{R}_\rho = \{(x,w)\}$, with the language parameter $\mathtt{lpar} := \rho$ from a distribution $\mathscr{D}_\mathsf{p}$. As we argued before, the primary security definitions of QA-

[8] Auxiliary-string non-black-box ZK [85] means that definitions hold even if any $\mathsf{aux} \in \{0,1\}^{\mathsf{poly}(\lambda)}$ is given as an additional input to $\mathscr{A}$.

NIZK [100] were defined in the CRS model[9]. In this part, we strengthen the primary definitions by assuming that both language parameter ρ and public key pk can be subverted. We then lift the definitions to the weaker BPK model. Motivated by the Sub-zk-SNARKs definitions of our work [9] (see Section 3.1), we first define the new algorithm PKV that checks if ρ and pk in BPK model are well-formed. Additionally, we define the new algorithm PARV which is the ρ-verification algorithm. Then by employing a PKV algorithm, we lift the primary QA-NIZKs definition to the BPK model. Following [22], we assume that the system parameters (i.e, the description of the bilinear group) p is deterministically generated from λ and so p is not permitted to be subverted.

For a set of witness-relations $\mathscr{R}_{\mathsf{p}} = \{\mathscr{R}_\rho\}_{\rho\in\mathrm{Supp}(\mathscr{D}_{\mathsf{p}})}$[10], we say a tuple of PPT algorithms $\Pi = (\mathsf{Pgen},\mathsf{KK},\mathsf{PARV},\mathsf{PKV},\mathsf{P},\mathsf{V},\mathsf{Sim})$ is a *Sub-ZK QA-NIZK argument system* in the BPK model if the following Items i, ii, iv and v hold. In addition, Π is a *Sub-ZK QA-NIZK argument of knowledge*, if Item iii holds.

Let Pgen be the parameter generation algorithm, KK is the public key pk generation algorithm, PARV is the ρ-verification algorithm, PKV is the public key verification algorithm, P is the prover, V is the verifier, and Sim is the simulator.

(i) **Perfect Completeness:** for any λ, $\mathsf{p}\in\mathrm{im}(\mathsf{Pgen}(1^\lambda))$,

$$\Pr\left[\begin{array}{l}\rho\leftarrow_{\$}\mathscr{D}_{\mathsf{p}};(\mathsf{pk},\mathsf{sk})\leftarrow\mathsf{KK}(\rho);(x,w)\leftarrow\mathscr{A}(\mathsf{pk});\\ \pi\leftarrow\mathsf{P}(\rho,\mathsf{pk},x,w):\mathsf{PARV}(\rho)=1\wedge\mathsf{PKV}(\rho,\mathsf{pk})=1\wedge\\ ((x,w)\notin\mathscr{R}_\rho\vee\mathsf{V}(\rho,\mathsf{pk},x,\pi)=1)\end{array}\right]=1\ .$$

(ii) **Computational Quasi-Adaptive Sub-PAR Soundness:** for any PPT $\mathscr{A}$ and $\mathsf{p}\in\mathrm{im}(\mathsf{Pgen}(1^\lambda))$,

$$\Pr\left[\begin{array}{l}\rho\leftarrow\mathscr{A}(\mathsf{p});(\mathsf{pk},\mathsf{sk})\leftarrow\mathsf{KK}(\rho);(x,\pi)\leftarrow\mathscr{A}(\mathsf{pk}):\\ \mathsf{PARV}(\rho)=1\wedge\mathsf{V}(\rho,\mathsf{pk},x,\pi)=1\wedge\neg(\exists w:\mathscr{R}_\rho(x,w)=1))\end{array}\right]\approx_\lambda 0\ .$$

(iii) **Computational Quasi-Adaptive Sub-PAR Knowledge-Soundness:** for every PPT adversary adversary $\mathscr{A}$, there exist a PPT extractor $\mathsf{Ext}_{\mathscr{A}}$, s.t. for all $\mathsf{p}\in\mathrm{im}(\mathsf{Pgen}(1^\lambda))$,

$$\Pr\left[\begin{array}{l}r\leftarrow_{\$}\mathsf{RND}_\lambda(\mathscr{A});\rho\leftarrow\mathscr{A}(\mathsf{p};r);(\mathsf{pk},\mathsf{sk})\leftarrow\mathsf{KK}(\rho);\\ (x,\pi)\leftarrow\mathscr{A}(\mathsf{pk};r);w\leftarrow\mathsf{Ext}_{\mathscr{A}}(\mathsf{p},\mathsf{pk};r):\mathsf{PARV}(\rho)=1\wedge\\ \mathsf{V}(\rho,\mathsf{pk},x,\pi)=1\wedge\mathscr{R}_\rho(x,w)=0\end{array}\right]\approx_\lambda 0\ .$$

A knowledge-sound argument system is called an *argument of knowledge*.

[9] Later Jutla and Roy modified the definitions by assuming the case that the language parameter ρ is maliciously chosen in the full version of their paper [101]

[10] Recall that a distribution $\mathscr{D}_{\mathsf{p}}$ on $\mathscr{L}_\rho$ is *witness-sampleable* [100] if there exists a PPT algorithm $\mathscr{D}'_{\mathsf{p}}$ that samples $(\rho,\mathsf{td}_\rho)\in\mathscr{R}_{\mathsf{p}}$ such that ρ is distributed according to $\mathscr{D}_{\mathsf{p}}$.

(iv) **Statistical Zero Knowledge:** for any $\mathsf{p} \in \mathrm{im}(\mathsf{Pgen}(1^\lambda))$ and computationally unbounded adversary $\mathscr{A}$, $|\varepsilon_0^{zk} - \varepsilon_1^{zk}| \approx_\lambda 0$, where $\varepsilon_b^{zk} :=$

$$\Pr\left[\rho \leftarrow \mathscr{D}_{\mathsf{p}}; (\mathsf{pk},\mathsf{sk}) \leftarrow \mathsf{KK}(\rho) : \mathscr{A}^{\mathsf{O}_b(\cdot,\cdot)}(\rho,\mathsf{pk}) = 1 \quad \right] .$$

The oracle $\mathsf{O}_0(x,w)$ returns $\perp$ (reject) if $(x,w) \notin \mathscr{R}_\rho$, and otherwise it returns $\mathsf{P}(\rho,\mathsf{pk},x,w)$. Similarly, $\mathsf{O}_1(x,w)$ returns $\perp$ (reject) if $(x,w) \notin \mathscr{R}_\rho$, and otherwise it returns $\mathsf{Sim}(\rho,\mathsf{pk},\mathsf{sk},x)$.

(v) **Statistical Persistent Zero Knowledge:** for any PPT subverter Z, there exists a PPT extractor Ext_Z, s.t. for any $\mathsf{p} \in \mathrm{im}(\mathsf{Pgen}(1^\lambda))$ and computationally unbounded adversary $\mathscr{A}$, $|\varepsilon_0^{zk} - \varepsilon_1^{zk}| \approx_\lambda 0$, where

$$\varepsilon_b^{zk} := \Pr\left[\begin{array}{l} r \leftarrow_\$ \mathsf{RND}_\lambda(\mathsf{Z}); (\rho,\mathsf{pk},\mathsf{aux}) \leftarrow \mathsf{Z}(\mathsf{p};r); \mathsf{sk} \leftarrow \mathsf{Ext}_\mathsf{Z}(\mathsf{p};r) : \\ \mathsf{PARV}(\rho) = 1 \wedge \mathsf{PKV}(\rho,\mathsf{pk}) = 1 \wedge \mathscr{A}^{\mathsf{O}_b(\cdot,\cdot)}(\rho,\mathsf{pk},\mathsf{aux}) = 1 \end{array}\right] .$$

The oracle $\mathsf{O}_0(x,w)$ returns $\perp$ (reject) if $(x,w) \notin \mathscr{R}_\rho$, and otherwise it returns $\mathsf{P}(\rho,\mathsf{pk},x,w)$. Similarly, $\mathsf{O}_1(x,w)$ returns $\perp$ (reject) if $(x,w) \notin \mathscr{R}_\rho$, and otherwise it returns $\mathsf{Sim}(\rho,\mathsf{pk},\mathsf{sk},x)$.

We emphasize that Π is *statistically Sub-ZK* if both statistically zero-knowledge (with trusted ρ and pk generators) and statistically persistent zero-knowledge (with possibly subverted ρ and pk) hold. Importantly, the reason why we need both properties to hold is the following: let Z be a subverter that creates both the language parameter ρ and the public key pk. For persistent zero-knowledge we assume an efficient extractor Ext_Z, such that given Z's random coins r as an input, it returns the secret key sk corresponding to pk (since there is no auxiliary input, ρ and pk *have* to be generated by Z with the random coins r). However, since we allow both $\mathtt{lpar}$ and pk be chosen by Z, it is possible that Z sets $\mathsf{sk} = \mathsf{td}_{\mathtt{lpar}}$. Therefore, in this case, persistent zero-knowledge holds but standard zero-knowledge does not hold (see the corresponding example in section 4 of [11]).

3.8.3. Constructing QA-NIZK in the BPK Model

We commence with the most efficient QA-NIZK construction Π'_{as} of Kiltz-Wee QA-NIZK [106] and build a Sub-ZK version of Π'_{as} which serves the aforementioned security definitions in the BPK model. In particular, we implement the *PKV* algorithm for Kiltz-Wee QA-NIZK Π'_{as} such that one (prover) can check if the public key in BPK (or CRS of its version in the CRS model) and the language parameter ρ are well-formed. In addition, to satisfy zero-knowledge property, we define two different (tautological) knowledge assumptions, KWKE (Kiltz-Wee Knowledge of Exponent assumption) and SKWKE (Strong Kiltz-Wee Knowledge of Exponent assumption) enabling the simulator Sim to obtain the correct secret key $\mathsf{sk} = K$ of Π'_{as} (for computing the simulated proof).

check $[a_{11}]_2 \neq [0]_2 \wedge \ldots \wedge [a_{kk}]_2 \neq [0]_2$;
if $\mathscr{D}_k = \mathscr{L}_k$ **then** check $i \neq j \Rightarrow [a_{i,j}]_2 = [0]_2$;
elseif $\mathscr{D}_k = \mathscr{I}\mathscr{L}_k$ **then** check $i \neq j \Rightarrow [a_{ij}]_2 = [0]_2; \forall i, [a_{i,i}]_2 = [a_{1,1}]_2 + [i-1]_2$;
elseif $\mathscr{D}_k = \mathscr{C}_k$ **then** check $i \notin \{j, j+1\} \Rightarrow [a_{ij}]_2 = [0]_2; \forall i, [a_{i+1,i}]_2 = [1]_2$;
elseif $\mathscr{D}_k = \mathscr{S}\mathscr{C}_k$ **then** check $i \notin \{j, j+1\} \Rightarrow [a_{ij}]_2 = [0]_2$;
$\forall i\,([a_{i+1,i}]_2 = [1]_2 \wedge [a_{ii}]_2 = [a_{11}]_2)$; **fi**
return 1 if all checks pass and 0 otherwise.

Figure 6. Auxiliary procedure $\mathsf{MATV}([\bar{A}]_2)$ for $\mathscr{D}_k \in \{\mathscr{L}_k, \mathscr{I}\mathscr{L}_k, \mathscr{C}_k, \mathscr{S}\mathscr{C}_k\}$.

New algorithm PKV**.** We recall the CRS of Kiltz-Wee QA-NIZK [106] Π'_{as} includes $\mathsf{crs} = ([\bar{A}, C]_2, [P]_1)$ where $\bar{A} \in \mathbb{Z}_p^{k \times k}$ denotes the upper square matrix of $A \in \mathbb{Z}_p^{(k+1)\times k}$ (see Section 2.12). Then by lifting Π'_{as} into the BPK model, we define the public key $\mathsf{pk} = (\mathsf{pk}^{\mathsf{snd}}, \mathsf{pk}^{\mathsf{zk}})$ where $\mathsf{pk}^{\mathsf{snd}} = [\bar{A}, C]_2$ and $\mathsf{pk}^{\mathsf{zk}} = [P]_1$.

Inspired from CV algorithm of our work [9] (see Section 3.1), we propose the new algorithm PKV with the following steps: (i) add some new elements to the public key pk (where we denote these elements by $\mathsf{pk}^{\mathsf{pkv}}$), and (ii) efficiently verify the the membership of $\bar{A}$ in $\mathscr{D}_k$. For the latter case, additionally, we propose another algorithm $\mathsf{MATV}(.)$ checking invertibility of the matrix $\bar{A}$ (see Fig. 6). We say that the distribution $\mathscr{D}_k$ is *efficiently verifiable* if there is an algorithm $\mathsf{MATV}([\bar{A}]_2)$ that outputs 1 if $\bar{A}$ is invertible and well-formed with regard to $\mathscr{D}_k$ and otherwise outputs 0. Finally we construct the PKV algorithm that is depicted in Fig. 7.

$\mathsf{KK}(\rho := [M]_1 \in \mathbb{G}_1^{n\times m})$**:** $A \leftarrow_{\$} \mathscr{D}_k$; $K \leftarrow_{\$} \mathbb{Z}_p^{n\times k}$; $[C]_2 \leftarrow [K\bar{A}]_2 \in \mathbb{G}_2^{n\times k}$; $[P]_1 \leftarrow [M]_1^\top K \in \mathbb{G}_1^{m\times k}$;
if $\mathscr{D}_k$ is efficiently verifiable **then** $\mathsf{pk}^{\mathsf{pkv}} \leftarrow \varepsilon$; **elseif** $\mathscr{D}_k = \mathscr{U}_2$ **then** $\mathsf{pk}^{\mathsf{pkv}} \leftarrow [a_{11}, a_{12}]_1$; **fi** ; $\mathsf{pk}^{\mathsf{snd}} \leftarrow [\bar{A}, C]_2$; $\mathsf{pk}^{\mathsf{zk}} \leftarrow [P]_1$;
$\mathsf{pk} \leftarrow (\mathsf{pk}^{\mathsf{snd}}, \mathsf{pk}^{\mathsf{zk}}, \mathsf{pk}^{\mathsf{pkv}})$; $\mathsf{sk} \leftarrow K$; **return** $(\mathsf{pk}, \mathsf{sk})$;

$\mathsf{PKV}([M]_1, \mathsf{pk})$**:** Return 1 only if the following checks all succeed:
$\mathsf{pk} = (\mathsf{pk}^{\mathsf{snd}}, \mathsf{pk}^{\mathsf{zk}}, \mathsf{pk}^{\mathsf{pkv}}) \wedge \mathsf{pk}^{\mathsf{snd}} = [\bar{A}, C]_2 \wedge \mathsf{pk}^{\mathsf{zk}} = [P]_1$;
$[P]_1 \in \mathbb{G}_1^{m\times k} \wedge [\bar{A}]_2 \in \mathbb{G}_2^{k\times k} \wedge [C]_2 \in \mathbb{G}_2^{n\times k}$;
$(*)$ $[M]_1^\top [C]_2 = [P]_1 [\bar{A}]_2$;
if $\mathscr{D}_k$ is efficiently verifiable **then** $\mathsf{MATV}([\bar{A}]_2)$;
else check $\mathsf{pk}^{\mathsf{pkv}} = [a_{11}, a_{12}]_1 \in \mathbb{G}_1^{1\times 2} \wedge [a_{11}]_1 [1]_2 = [1]_1 [a_{11}]_2 \wedge$
$[a_{12}]_1 [1]_2 = [1]_1 [a_{12}]_2 \wedge [a_{11}]_1 [a_{22}]_2 - [a_{12}]_1 [a_{21}]_2 \neq [0]_T$; **fi**

Figure 7. Algorithm PKV for $[y]_1 = [M]_1 w$ in the BPK model, where either $\mathscr{D}_k$ is efficiently verifiable or $\mathscr{D}_k = \mathscr{U}_2$.

New knowledge assumptions. For providing *Sub-ZK property* in the BPK model (or in subverted setting) we define the two new knowledge assumptions, KWKE

and SKWKE. Roughly speaking the KWKE assumption guarantees that one can extract a secret key $\mathsf{sk} = K$ from which one can generate $\mathsf{pk}^{\mathsf{zk}} = [P]_1$ but not necessarily $\mathsf{pk}^{\mathsf{snd}}$. On the other hand, by the knowledge assumption SKWKE, one can extract the *unique* secret key K that was used to generate the *whole* public key $\mathsf{pk} = (\mathsf{pk}^{\mathsf{snd}}, \mathsf{pk}^{\mathsf{zk}})$. Notice that to deliver Sub-ZK, one only needs to guarantee that $\mathsf{pk}^{\mathsf{zk}}$ can be calculated form sk and so KWKE knowledge assumption is sufficient.

More formally, in KWKE assumption, we assume that if $\mathscr{A}$ outputs a $\rho \in \mathscr{D}_{\mathsf{p}}$ and a pk accepted by PKV, then there exists an extractor $\mathsf{Ext}_{\mathscr{A}}$ who, knowing the secret coins of $\mathscr{A}$, returns a secret key K that *could* have been used to compute $\mathsf{pk}^{\mathsf{zk}}$. SKWKE additionally assures that the same K has been used to compute $\mathsf{pk}^{\mathsf{snd}}$.

Definition 18. *Fix $k \geq 1$, $n > m \geq 1$, and a distribution $\mathscr{D}_k$. Let PKV be as in Fig. 7. Then $(\mathscr{D}_{\mathsf{p}}, k, \mathscr{D}_k)$-$\mathrm{KWKE}_{\mathbb{G}_1}$ (resp., $\boxed{(\mathscr{D}_{\mathsf{p}}, k, \mathscr{D}_k)\text{-}\mathrm{SKWKE}_{\mathbb{G}_1}}$) holds relative to Pgen if for any $\mathsf{p} \in \mathrm{im}(\mathsf{Pgen}(1^\lambda))$ and PPT adversary $\mathscr{A}$, there exists a PPT extractor $\mathsf{Ext}_{\mathscr{A}}$, s.t. $\mathsf{Adv}^{\boxed{S}\mathsf{kwke}}_{\mathscr{D}_{\mathsf{p}},k,\mathscr{D}_k,\mathbb{G}_1,\mathsf{Pgen},\mathscr{A},\mathsf{Ext}_{\mathscr{A}}}(\lambda) :=$*

$$\Pr\left[\begin{array}{l} r \leftarrow_{\$} \mathsf{RND}_\lambda(\mathscr{A}); (\rho := [M]_1, \mathsf{pk}) \leftarrow \mathscr{A}(\mathsf{p}; r); K \leftarrow \mathsf{Ext}_{\mathscr{A}}(\mathsf{p}; r) : \\ \mathsf{pk} = ([\bar{A}, C]_2, [P]_1, \mathsf{pk}^{\mathsf{pkv}}) \wedge [M]_1 \in \mathscr{D}_{\mathsf{p}} \wedge \\ \mathsf{PKV}([M]_1, \mathsf{pk}) = 1 \wedge (P \neq M^\top K \boxed{\vee C \neq K\bar{A}}) \end{array}\right] \approx_\lambda 0 \ .$$

Here, the $\boxed{\textit{boxed}}$ part is only present in the definition of SKWKE.

Definition 19. *$\mathscr{D}_{\ell k}$-$\mathrm{WKerMDH}_{\mathbb{G}_1}$ holds relative to Pgen, if for any PPT $\mathscr{A}$, $\Pr[\mathsf{p} \leftarrow \mathsf{Pgen}(1^\lambda); A \leftarrow_{\$} \mathscr{D}_{\ell k}; c \leftarrow \mathscr{A}(\mathsf{p}, [A]_1) : A^\top c = 0_k \wedge c \neq 0_\ell] \approx_\lambda 0$.*

Admittedly, WKerMDH is a weaker variant of the KerMDH distribution. Notice that the assumption of WKerMDH-hardness often holds in practice, for example when ρ is a randomly chosen public key of a commitment scheme or a cryptosystem. We proved the security of KWKE and SKWKE under hash-algebraic knowledge (HAK) [116][11] and refer [11] for details of the proofs.

New interactive assumptions $\mathrm{KerMDH}^{\mathrm{dl}}$ and $\mathrm{SKerMDH}^{\mathrm{dl}}$. As we stated in Section 3.8.2, in Sub-PAR soundness, the language parameter $\rho = [M]_1$ is maliciously chosen. Thus in Sub-PAR soundness proof, we require that the trapdoor of ρ is extractable. To deal with this matter we define a new interactive non-falsifiable $\mathrm{KerMDH}^{\mathrm{dl}}$ (also $\mathrm{SKerMDH}^{\mathrm{dl}}$) assumption that ables one to extract the trapdoor of M from $[M]_1$. More precisely, in the soundness proof of Kiltz-Wee QA-NIZK [106], the KerMDH adversary $\mathscr{B}$ obtains $([M]_1, M)$ sampled from $\mathscr{D}'_{\mathsf{p}}$ (this relies on the witness-sampleability). But in our Sub-PAR soundness proof (we will see later in Theorem 4), $\mathscr{B}$ receives $[M]_1 \leftarrow \mathscr{A}(\mathsf{p})$ and then employs a

[11] Lipmaa [116] recently established the framework of HAK assumptions to make the algebraic group model (AGM) of Fuchsbauer *et al.* [79] more concrete and applicable. While in the AGM, it is assumed that every adversary is algebraic, a HAK assumption is defined concerning a concrete input distribution of the adversary.

non-adaptive DL oracle to extract M and so importantly, our proof does not need witness-sampleability. The SDL$^{\mathrm{dl}}$, KerMDH$^{\mathrm{dl}}$ and SKerMDH$^{\mathrm{dl}}$ assumptions are X^Y-type interactive assumptions as applied in [84, 112], where we contemplated that the assumption X holds if even the adversary can non-adaptively (i.e., before the X is picked) query an oracle that can solve the assumption Y.

The SDL$^{\mathrm{dl}}$ *assumption* holds relative to Pgen, if for any PPT $\mathscr{A}$,

$$\Pr\left[\mathsf{p} \leftarrow \mathsf{Pgen}(1^\lambda); st \leftarrow \mathscr{A}^{\mathrm{dl}(\cdot)}(\mathsf{p}); x \leftarrow_\$ \mathbb{Z}_p : \mathscr{A}(\mathsf{p}, st, [x]_1, [x]_2) = x\right] \approx_\lambda 0 \ .$$

Here, the oracle $\mathrm{dl}([y]_1)$ returns the discrete logarithm y of $[y]_1$.

The $\mathscr{D}_{\ell k}$-KerMDH$^{\mathrm{dl}}_{\mathbb{G}_1}$ *assumption* holds relative to Pgen, if for any PPT $\mathscr{A}$,

$$\Pr\left[\begin{array}{l}\mathsf{p} \leftarrow \mathsf{Pgen}(1^\lambda); st \leftarrow \mathscr{A}^{\mathrm{dl}(\cdot)}(\mathsf{p}); A \leftarrow_\$ \mathscr{D}_{\ell k}; [c]_2 \leftarrow \mathscr{A}(\mathsf{p}, st, [A]_1) : \\ A^\top c = 0_k \wedge c \neq 0_\ell\end{array}\right] \approx_\lambda 0 \ .$$

The $\mathscr{D}_{\ell k}$-SKerMDH$^{\mathrm{dl}}$ *assumption* holds relative to Pgen, if for any PPT $\mathscr{A}$,

$$\Pr\left[\begin{array}{l}\mathsf{p} \leftarrow \mathsf{Pgen}(1^\lambda); st \leftarrow \mathscr{A}^{\mathrm{dl}(\cdot)}(\mathsf{p}); A \leftarrow_\$ \mathscr{D}_{\ell k}; \\ ([c_1]_1, [c_2]_2) \leftarrow \mathscr{A}(\mathsf{p}, st, [A]_1, [A]_2) : A^\top (c_1 - c_2) = 0_k \wedge c_1 - c_2 \neq 0_\ell\end{array}\right] \approx_\lambda 0 \ .$$

Now by putting all the aforementioned algorithms and techniques together, we build a QA-NIZK in the BPK model.

QA-NIZK Construction in the BPK Model. Finally to build QA-NIZK Π_{bpk} in the BPK Model (or subversion ZK QA-NIZK in CRS model), one can take Kiltz-Wee QA-NIZK Π'_{as} for linear subspaces [106] and attach the new algorithm PKV to their construction. In Theorem 4 we prove the security of the construed QA-NIZK Π_{bpk} in the BPK model.

Theorem 4. *Assume that* $\mathscr{D}_{\mathsf{p}}$ *is such that* PARV *is efficient.*

(i) Π_{bpk} *is perfectly complete and perfectly zero-knowledge.*

(ii) If $(\mathscr{D}_{\mathsf{p}}, k, \mathscr{D}_k)$-KWKE$_{\mathbb{G}_1}$ *holds relative to* Pgen *then* Π_{bpk} *is statistically persistent zero-knowledge.*

(iii) Assume $\mathscr{D}_k$ *is efficiently verifiable (resp.,* $\mathscr{D}_k = \mathscr{U}_2$*). If* $\mathscr{D}_k$-KerMDH$^{\mathrm{dl}}$ *(resp.,* $\mathscr{D}_k$-SKerMDH$^{\mathrm{dl}}$*) holds relative to* Pgen *then* Π_{bpk} *is computationally quasi-adaptively Sub-PAR sound.*

(iv) Assume M has rank n ($y = Mw$ *always has a solution), and that* $\mathscr{D}_k$ *is robust. If* SDL$^{\mathrm{dl}}$ *and* $\mathsf{KGen}([M]_1)$*-HAK, for arbitrary efficiently computable* $[M]_1$*, hold relative to* Pgen *then* Π_{bpk} *is computationally quasi-adaptively Sub-PAR knowledge-sound.*

See the proof of Theorem 4 in [11].

4. UC-SECURE COMMITMENT CONSTRUCTIONS

General Motivation for UC-Secure commitment Scheme. A commitment scheme allows a committer C to transmit an analogue of a sealed envelope of her message m to a receiver R. Later the committer C can open her committed message m and together with some additional information. The receiver R verifies whether m is correctly enveloped. Such systems should assure that C cannot later change the enveloped message m to some $m' \neq m$ (binding property); while the receiver R must not learn any information of the enveloped message m (hiding property). One of the momentous aspects for practical applications of cryptographic protocols particularly in NIZKs or SNARKs (i.e, for implementing their CRS) is achieving Universal Composability (UC). This is a strong property was that introduced by in [52]. Informally, UC-secure protocols guarantee the security even if executed with other instances of the same protocol, or when composed with other protocols. To make commitment schemes UC-secure, one needs to provide the additional properties: (i) extractability, states that given a trapdoor, one (i.e, the simulator Sim in UC proof) can recover the committed value m, and (ii) equivocability means that given a trapdoor, one (i.e, the simulator Sim in UC proof) can open a commitment to any message $m' \neq m$.

In this chapter we investigate building efficient UC-secure commitments; one with utilizing a new primitive called publicly computable SPHF (PC-SPHF) which is currently the most efficient non-interactive UC-secure commitment [10]; and another one [6], which is an special UC-secure commitment called DL-extractable commitment that is compatible with MPC approach (aiming to generate the CRS of NIZKs in a UC secure manner)[1].

[1] Later Abdolmaleki *et al.* [7] used the UC-secure DL-extractable commitment to generate the CRS of SNARK [94] with MPC in a UC-secure manner.

4.1. UC-Secure Commitment from PC-SPHF

Motivation. In general UC-secure commitment scheme plays an influential role in many cryptographic protocols such as zero-knowledge proofs and multi-party computation. Therefore constructing an *efficient* UC-secure commitment scheme is an active area and so far there has been many efforts [4–6, 27, 74, 100] to achieve an *efficient* UC-secure commitment scheme. Constructing efficient UC-secure commitments falls into the following two categories: (i) the line following the ideas of Canetti-Fischlin [52] including [4–6, 27], and (ii) the lines employing non-interactive zero-knowledge proofs in the opening phase of the commitment as Fischlin *et al.*'s schemes [74] and improvements thereof [100]. In this part, we continue into the latter direction, and instead of non-interactive zero-knowledge proofs, we explicate how one can utilize the new primitive PC-SPHF and construct a framework for non-interactive UC-secure commitment scheme.

4.2. Problem Statement

Can we construct a compact framework for the UC-Secure commitment scheme? What tools could lead us to design efficient UC-Secure commitment respecting communication complexity and the number of rounds of the protocol? What are the best choices of cryptographic tools to instantiate such UC-secure commitments?

4.3. Our Solution

We start through the notion of SPHF in the bilinear group and extend it to build a new primitive called publicly computable SPHF (PC-SPHF). In particular, we append a new algorithm (the third mode of hashing) to SPHF systems. The new algorithm enables one to compute the hash value without knowing neither the witness of the language nor the hashing key, but some additional auxiliary information. We mainly define PC-SPHFs for languages of ciphertexts and show it can cover lots of important schemes such as ElGamal [70] and Cramer-Shoup (CS) [60] cryptosystems. We then employ PC-SPHFs primitive built from any proper SPHF for a (labeled) IND-CCA encryption scheme and construct a generic UC-secure commitment scheme. Finally, we instantiate this framework with CS encryption [60] and construct the currently most efficient non-interactive UC-secure commitment. We compare the performance of our UC-secure commitment with existing non-interactive UC-secure commitments in Table 4.3.4. A full description of our construction is given in the paper [10], that is joint work with Khoshakhlagh and Slamanig.

4.3.1. New Primitive: PC-SPHF

We recall that SPHFs [60] for a language $\mathscr{L}_{\mathtt{lpar}}$, parameterized by a language parameter $\mathtt{lpar}$, are cryptographic primitives with the following properties. Given a word x, one can compute a hash of it in two different ways: (i) either utilizing a projection key hp (an analogue of a public key), a word x, and a witness w, as $\mathsf{pH} \leftarrow \mathsf{projhash}(\mathtt{lpar}; \mathsf{hp}, x, w)$, or (ii) using a hashing key hk (an analogue of a secret key) and a word x, as $\mathsf{H} \leftarrow \mathsf{hash}(\mathtt{lpar}; \mathsf{hk}, x)$. where the algorithms projhash and hash are respectively the projection hash and the hash generators. If $x \in \mathscr{L}_{\mathtt{lpar}}$ and w is a valid witness of this fact, then the *correctness* (also known as *projectivity*) property guarantees that the two ways of computing the hash result in the same value, $\mathsf{pH} = \mathsf{H}$. If $x \notin \mathscr{L}$, then the *smoothness* property guarantees that, knowing hp but not hk, one cannot distinguish H from random.

We introduce a new feature (a third mode of hashing pcH) for SPHFs and define a new primitive called PC-SPHFs. More accurately, pcH algorithm computes the hash value in target group $\mathbb{G}_T$ without taking neither the witness w nor the hashing key hk as input, but some additional auxiliary information aux. Our PC-SPHF is similar to the trapdoor SPHFs in [27, 28] with some modifications, but has completely different motivations, and algorithms.

In particular, a PC-SPHF can be built upon an SPHF and then instantiated in the bilinear group $(p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, \hat{e})$. In contrast with SPHFs system, in PC-SPHFs, the projkg algorithm takes a hashing key hk, a word x and language $\mathscr{L}_{\mathsf{aux}}$, and returns a projection key hp that contains two parts $\mathsf{hp} = (\mathsf{hp}_1, \mathsf{hp}_2) \in \mathbb{G}_\iota^k \times \mathbb{G}_{3-\iota}^n$. The projection key hp_1 is the one underlying SPHF and the projection key hp_2 is intuitively some representation of the hashing key hk.

Definition 20. *A PC-SPHF for language $\mathscr{L}_{\mathsf{aux}}$ based upon SPHF is defined by the following algorithms:*

$\mathsf{Pgen}(1^\lambda, \mathscr{L}_{\mathsf{aux}})$**:** *Takes a security parameter λ and language $\mathscr{L}_{\mathsf{aux}}$ and generates the global parameters p, and the $\mathsf{crs}_{\mathscr{L}_{\mathsf{aux}}}$. It outputs $(\mathsf{p}, \mathsf{aux}, \mathsf{crs}_{\mathscr{L}_{\mathsf{aux}}})$.*

$\mathsf{hashkg}(\mathscr{L}_{\mathsf{aux}})$**:** *Takes a language $\mathscr{L}_{\mathsf{aux}}$ and outputs a hashing key $\mathsf{hk} = \alpha \leftarrow_\$ \mathbb{Z}_p^n$ for the language $\mathscr{L}_{\mathsf{aux}}$ of the underlying SPHF.*

$\mathsf{projkg}(\mathsf{hk}, \mathsf{crs}_{\mathscr{L}_{\mathsf{aux}}}, x)$**:** *Takes a hashing key hk, a CRS crs, and possibly a word x and outputs a projection key $\mathsf{hp} = (\mathsf{hp}_1, \mathsf{hp}_2) \in \mathbb{G}_\iota^k \times \mathbb{G}_{3-\iota}^n$, possibly depending on x, where hp_1 is the projection key of the underlying SPHF and hp_2 is some representation of hk.*

$\mathsf{hash}(\mathsf{hk}, \mathsf{crs}_{\mathscr{L}_{\mathsf{aux}}}, \mathsf{aux}, x)$**:** *Takes a hashing key hk, a CRS crs, aux, and a word x and outputs a hash $\mathsf{H} \in \mathbb{G}_\iota$, being the hash of the underlying SPHF.*

$\mathsf{projhash}(\mathsf{hp}, \mathsf{crs}_{\mathscr{L}_{\mathsf{aux}}}, \mathsf{aux}, x, w)$**:** *Takes a projection key hp, a CRS $\mathsf{crs}_{\mathscr{L}_{\mathsf{aux}}}$, aux, a word x, and a witness w for $x \in \mathscr{L}_{\mathsf{aux}}$ and outputs a hash $\mathsf{pH} \in \mathbb{G}_\iota$, being the projective hash of the underlying SPHF.*

$\mathsf{pchash}(\mathsf{hp}, \mathsf{crs}_{\mathscr{L}_{\mathsf{aux}}}, \mathsf{aux}, x)$**:** *Takes a projection key hp, a CRS $\mathsf{crs}_{\mathscr{L}_{\mathsf{aux}}}$, aux, and a word x, and outputs a hash $\mathsf{pcH} \in \mathbb{G}_T$.*

$(\mathsf{p}, \mathsf{aux}, \mathsf{crs}_{\mathscr{L}_{\mathsf{aux}}}) \leftarrow \mathsf{Pgen}(1^{\lambda}, \mathscr{L}_{\mathsf{aux}})$, $x \leftarrow_{\$} \mathscr{X}_{\mathsf{aux}} \setminus \mathscr{L}_{\mathsf{aux}}$, $\mathsf{hk} \leftarrow \mathsf{hashkg}(\mathscr{L}_{\mathsf{aux}})$, $\mathsf{hp} \leftarrow \mathsf{projkg}(\mathsf{hk}, \mathsf{crs}_{\mathscr{L}_{\mathsf{aux}}}, x)$;
If $b = 0$, then $\mathsf{H} \leftarrow \mathsf{hash}(\mathsf{hk}, \mathsf{crs}_{\mathscr{L}_{\mathsf{aux}}}, \mathsf{aux}, x)$, else $\mathsf{H} \leftarrow_{\$} \Omega$;
return $\mathscr{A}(\mathsf{crs}_{\mathscr{L}_{\mathsf{aux}}}, x, \mathsf{hp}, \mathsf{H})$

Figure 8. Experiments $\mathsf{Exp}^{csmooth\text{-}b}(\mathscr{A}, \lambda)$ for computational smoothness.

A PC-SPHF should satisfy the following properties:

Perfect correctness. For any $(\mathsf{p}, \mathsf{aux}, \mathsf{crs}_{\mathscr{L}_{\mathsf{aux}}}) \leftarrow \mathsf{Pgen}(1^{\lambda}, \mathscr{L}_{\mathsf{aux}})$, any word $x \in \mathscr{L}_{\mathsf{aux}}$ with witness w, any $\mathsf{hk} \leftarrow \mathsf{hashkg}(\mathscr{L}_{\mathsf{aux}})$, and $\mathsf{hp} \leftarrow \mathsf{projkg}(\mathsf{hk}, \mathsf{crs}_{\mathscr{L}_{\mathsf{aux}}}, x)$:

$$\mathsf{pH} \cdot [1]_{3-\iota} = \mathsf{pcH}.$$

The (t, ε)-soundness property. For any $(\mathsf{p}, \mathsf{aux}, \mathsf{crs}_{\mathscr{L}_{\mathsf{aux}}}) \leftarrow \mathsf{Pgen}(1^{\lambda}, \mathscr{L}_{\mathsf{aux}})$, given $\mathsf{crs}_{\mathscr{L}_{\mathsf{aux}}}$ and the projection key hp, no adversary running in time at most t can produce a value aux, a word x and valid witness w such that the following holds, $\mathsf{projhash}(\mathsf{hp}, \mathsf{crs}_{\mathscr{L}_{\mathsf{aux}}}, \mathsf{aux}, x, w) \neq \mathsf{hash}(\mathsf{hk}, \mathsf{crs}_{\mathscr{L}_{\mathsf{aux}}}, \mathsf{aux}, x)$, with probability at least ε. Perfect soundness requires that this holds for any t and any $\varepsilon > 0$.

Computational Smoothness. The computational smoothness experiment is provided in Fig. 8. For a language $\mathscr{L}_{\mathsf{aux}}$ and adversary $\mathscr{A}$, the advantage is defined as follows:

$$\mathsf{Adv}^{\mathrm{csmooth}}_{\mathscr{L}_{\mathsf{aux}}, \mathscr{A}}(\lambda) = |\Pr[\mathsf{Exp}^{csmooth-0}(\mathscr{A}, \lambda) = 1] - \Pr[\mathsf{Exp}^{csmooth-1}(\mathscr{A}, \lambda) = 1]|.$$

and we require that $\mathsf{Adv}^{\mathrm{csmooth}}_{\mathscr{L}_{\mathsf{aux}}, \mathscr{A}}(\lambda) \leq \mathsf{negl}(\lambda)$.

See the security analyzes of PC-SPHF in [10].

4.3.2. A Framework for UC-Secure Commitment Scheme

Now we present a generic UC-secure commitment scheme from any IND-CCA secure labeled public-key encryption scheme with an associated PC-SPHF. Intuitively, in our framework a committer C commits the message with an IND-CCA secure labeled ciphertext. Then by knowing the witness w (i.e., the randomness used in the encryption), C computes the projective hash pH and reveals it in the opening phase of the protocol. In the verification phase, the receiver R first computes the hash value pcH and verifies the commitment. Additionally in the the UC security proof, given the hashing key hk a simulator Sim computes the hash value H as the simulated proof of pH.

We utilize Canetti-Fischlin's ideal functionality of a commitment scheme [53] in the security proof of our generic UC-secure commitment. Let sid be a standard *commitment identifier* [2]. Following the functionality of [53] we define sid to be

[2]The standard *commitment identifier* sid shows the case when a committer C commits to a receiver R within a session.

$K_{crs}(1^{\lambda})$**:** generate a secret and public key $(\mathsf{sk},\mathsf{pk})$ for a labeled IND-CCA encryption scheme, set $\mathsf{crs}_{\mathscr{L}_{\mathsf{aux}}} = \mathsf{pk}$. Compute $\mathsf{hk} \leftarrow \mathsf{hashkg}(\mathscr{L}_{\mathsf{aux}})$ and $\mathsf{hp} \leftarrow \mathsf{projkg}(\mathsf{hk},\mathsf{crs}_{\mathscr{L}_{\mathsf{aux}}},\cdot)$ and set $\mathsf{crs} := (\mathsf{crs}_{\mathscr{L}_{\mathsf{aux}}},\mathsf{hp})$.

/ Commit phase:

$\mathsf{Commit}(\mathsf{crs},M,\mathsf{sid},\mathsf{cid},P_i,P_j)$**:** commit to message $M \in \mathbb{G}_1$ for party P_j, upon receiving a command $(\texttt{commit},\mathsf{sid},\mathsf{cid},P_i,P_j,M)$, party P_i chooses randomness r and computes $\mathsf{c} = \mathsf{Enc}^{\tau}_{\mathsf{pk}}(M;r)$ with $\tau = (\mathsf{sid},\mathsf{cid},P_i)$ and $\mathsf{pH} \leftarrow \mathsf{projhash}(\mathsf{hp},\mathsf{crs}_{\mathscr{L}_{\mathsf{aux}}},M\mathsf{c},r)$. P_i erases r and sends c to P_j and stores pH. Upon receiving $(\texttt{commit},\mathsf{sid},\mathsf{cid},P_i,P_j,\mathsf{c})$ from P_i, party P_j verifies c is well-formed. If yes, P_j outputs $(\texttt{receipt},\mathsf{sid},\mathsf{cid},P_i,P_j)$. Otherwise, P_j ignores the message.

/ Opening phase:

$\mathsf{Open}(M,\mathsf{pH},\mathsf{sid},\mathsf{cid},P_i,P_j)$**:** when receiving a command $(\texttt{open},\mathsf{sid},\mathsf{cid},P_i,P_j,M)$, party P_i reveals M and his state information pH.

/ Verification phase:

$\mathsf{Ver}(\mathsf{crs},(\texttt{commit},\mathsf{sid},\mathsf{cid},\mathsf{c}),M,\mathsf{pH},\mathsf{sid},\mathsf{cid},P_i,P_j)$: P_j computes $\mathsf{pcH} \leftarrow \mathsf{pchash}(\mathsf{hp},\mathsf{crs}_{\mathscr{L}_{\mathsf{aux}}},M,\mathsf{c})$ and verifies pH, i.e., whether $\mathsf{pH}\cdot[1]_2 = \mathsf{pcH}$, and ignores the opening if verification fails. If verification succeeds, P_j outputs $(\texttt{open},\mathsf{sid},\mathsf{cid},P_i,P_j,M)$ iff cid has not been used with this committer previously. Otherwise, P_j also ignores the message.

Figure 9. Generic UC-Secure Commitment from PC-SPHFs.

a standard *commitment identifier* [3]. We use another unique *commitment identifier* cid, for the case when a committer C commits to the same receiver R multiple times within a session. Besides, we assume that the combination of sid and cid is globally unique. Our framework for UC-secure commitment is secure against adaptive corruptions (assuming reliable erasure) and it is shown in Fig. 9. See the security proof of the Generic UC-Secure Commitment scheme of Fig. 9 in [10] (see Section 5.2).

4.3.3. Efficient UC-Secure Commitment Scheme

Finally in this section we build an efficient UC-secure commitment by instantiating the framework in Fig. 9 with labeled CS encryption scheme and PC-SPHF on labeled CS ciphertexts. The resulting scheme carries 4 elements in $\mathbb{G}_1$ in the commit phase and one element in $\mathbb{G}_1$ in the opening phase. To the best of our knowledge, this is currently the most efficient non-interactive UC-secure commitment scheme.

In the following we give a brief explanation of the PC-SPHF construction for labeled CS ciphertexts that is the core of building the efficient UC secure commit-

[3] The standard *commitment identifier* sid shows the case when a committer C commits to a receiver R within a session.

Scheme	|Commitment|	|Opening|	Assumption
[53]	$9 \times \mathbb{G}$	$2 \times \mathbb{Z}_p$	Plain DDH
[74],1	$5 \times \mathbb{G}_1$	$16 \times \mathbb{G}_1$	DLIN
[74],2	$37 \times \mathbb{G}_1$	$3 \times \mathbb{G}_1$	DLIN
[100]	$4 \times \mathbb{G}_1$	$3 \times \mathbb{G}_1 + 2 \times \mathbb{G}_2$	SXDH
[100]	$4 \times \mathbb{G}_1$	$4 \times \mathbb{G}_1$	DLIN
[1]	$8 \times \mathbb{G}_1 + \mathbb{G}_2$	$\mathbb{Z}_p$	SXDH
[4]	$7 \times \mathbb{G}$	$2 \times \mathbb{Z}_p$	Plain DDH
PC-SPHF$_{\mathsf{CS}}$	$4 \times \mathbb{G}_1$	$\mathbb{G}_1$	XDH

Table 2. Comparison with some existing non-interactive UC-secure commitments with a single global CRS when committing to a single message.

ment.

PC-SPHF on (Labeled) Cramer-Shoup Ciphertexts We explicate how to extend the SPHF on (labeled) CS ciphertexts into a PC-SPHF. The CRS $\mathsf{crs}_{\mathscr{L}_{\mathsf{aux}}}$ contains the encryption public key pk. With the hashing key $\mathsf{hk} = (\eta_1, \eta_2, \theta, \mu, \iota) \leftarrow_{\$} \mathbb{Z}_p^5$ and the projection key $\mathsf{hp} = ([\mathsf{hp}_1]_1, [\mathsf{hp}_2]_2)$, where $[\mathsf{hp}_{11}]_1 = \eta_1[g_1]_1 + \theta[g_2]_1 + \mu[h]_1 + \iota[c]_1$, and $[\mathsf{hp}_{12}]_1 = \eta_2[g_1]_1 + \iota[d]_1$, and $[\mathsf{hp}_2]_2 = [\eta_1, \eta_2, \theta, \mu, \iota]_2 \in \mathbb{G}_2^5$, and $\mathsf{aux} = [m]_1$, the hash values of the PC-SPHF are defined as follows:

$$
\begin{aligned}
\mathsf{H} =& \mathsf{hash}(\mathsf{hk}, \mathsf{crs}_{\mathscr{L}_{\mathsf{aux}}}, [m]_1, [\mathsf{c}]_1) \\
=& (\eta_1 + \xi\eta_2)[u_1]_1 + \theta[u_2]_1 + \mu([e]_1 - [m]_1) + \iota[v]_1 \in \mathbb{G}_1 \\
\mathsf{pH} =& \mathsf{projhash}(\mathsf{hp}, \mathsf{crs}_{\mathscr{L}_{\mathsf{aux}}}, [m]_1, [\mathsf{c}]_1, r) = r[\mathsf{hp}_{11}]_1 + r\xi[\mathsf{hp}_{12}]_1 \in \mathbb{G}_1 \\
\mathsf{pcH} =& \mathsf{pchash}(\mathsf{hp}, \mathsf{crs}_{\mathscr{L}_{\mathsf{aux}}}, [m]_1, [\mathsf{c}]_1) \\
=& [u_1]_1[\mathsf{hp}_{21}]_2 + [u_1]_1 \cdot \xi[\mathsf{hp}_{22}]_2 + [u_2]_1[\mathsf{hp}_{23}]_2 + ([e]_1 - [m]_1)[\mathsf{hp}_{24}]_2 \\
&+ [v]_1[\mathsf{hp}_{25}]_2 = [u_1]_1[\eta_1]_2 + [u_1]_1 \cdot \xi[\eta_2]_2 + [u_2]_1[\theta]_2 \\
&+ ([e]_1 - [m]_1)[\mu]_2 + [v]_1[\iota]_2 \in \mathbb{G}_T.
\end{aligned}
$$

4.3.4. Comparisons

We compare the performance of our UC-secure commitment PC-SPHF$_{\mathsf{CS}}$ with existing non-interactive UC-secure commitments in Table 4.3.4 [4]

For the comparison, we consider a type 3 bilinear group with the desired security level of 128 bit. Common preferences are Baretto-Naehrig (BN) or Barreto-Lynn-Scott (BLS) curves. A conservative estimate for this security level results elements in $\mathbb{G}_1$, $\mathbb{G}_2$ and $\mathbb{G}_T$ of size $2 \cdot 384$, $4 \cdot 384$ and $12 \cdot 384$ bits for BN and BLS12 and $2 \cdot 320$, $4 \cdot 320$ and $24 \cdot 320$ for BLS24 (without point compression)

[4] Notice that following existing literature we concentrate on the size of commitments and openings and exclude the message(s) in the opening information.

respectively [119]. Besides, we assume elliptic curves over prime fields to instantiate the plain DDH setting, elements of $\mathbb{G}$ will have at least $2 \cdot 256$ bits (without point compression) when targeting 128-bit security. Consequently, assuming point compression is utilized in both schemes, the commitment and opening size of the UC-commitment in [4] is 1799 and 512 bits respectively. Our UC-secure commitment has a commitment and opening size of 1284 and 321 bit respectively, improving [4] by about 30%. Moreover, compared to the most efficient construction in bilinear groups (i.e., [1]), we gain the opening size with a factor of 4.

4.4. UC-Secure Commitment with Integer Extraction

Motivation. As we noted before, an important goal, especially for practitioners is implementing the CRS mode. More precisely, the main challenge is diminishing trust in the CRS generation phase. In other words, how one can guarantee the CRS generated by a *single* party $\mathscr{R}$ is chosen from the correct distribution without any leakage of its trapdoors. A weaker setup model called registered public key (RPK) was introduced by Barak *et al.* [19]. In fact, in the RPK setting instead of having only one trusted party (in the CRS model), each party $\mathscr{G}_i$ should trust *some* key registration authority $\mathscr{R}_i$ who registers her key. In particular, the CRS model is a strong variant of the RPK model where the authority $\mathscr{R}_i$ of each party is the same and all parties must trust only one authority $\mathscr{R}$.

Kosba *et al.* [109] proposes a framework that transfers any zk-SNARK into a UC-secure SNARK with the cost of adding a trusted setup, run by a single party which is not UC secure. To have a fully UC-secure zk-SNARK with distributed trust in the setup phase, one needs to construct a UC-secure CRS-generation protocol by MPC. Due to the particular structure of pairing-based zk-SNARKs, in order to achieve that, one would require a particular UC-secure commitment scheme along with its corresponding UC-functionality. More precisely, consider the setting that the secrecy of a committed value x is important, namely revealing x enables to break of the privacy of the scheme, but to generate the CRS elements, the parties need to reveal g^x. On the other hand, in the UC setting the universal simulator needs to know x to simulate the corrupted committer C. This brought us to the point that we need a UC functionality that would allow the committer C to commit to x while later open to a group element g^x, and also allow the UC simulator to extract the integer x. In this chapter, we deal with the above research question and propose the mentioned ideal functionality along with a commitment (called DL-extractable commitment) that UC-securely realizes the proposed functionality. Also we note that Abdolmaleki *et al.* [7] uses our UC-secure commitment (see Section 4.8.2) to construct the CRS of Groth's SNARK [94]. In order to keep the consistency of the result in the paper [6] and this section, we use the standard notation g^x instead of the bracket notation $[x]_\iota$.

4.5. Problem Statement

Can we construct a UC-secure commitment that is compatible with MPC approach (i.e., for generating the CRS of UC-secure SNARKs [109] to have a fully UC-secure SNARK of [109])? More precisely, can we construct a UC-secure commitment that enables the committer to open a commitment to a group element (i.e, the group element g^x); however, the simulator could extract its discrete logarithm of the group element (i.e, the integer x)?

4.6. Our Solution

We start by defining a new ideal UC functionality $\mathscr{F}_{\mathsf{mcomdl}}$ that will be used for our final goal, constructing a UC-secure commitment scheme called *DL-extractable commitment*. Intuitively, in contrast with the standard functionality of UC-secure commitment schemes [53], the functionality $\mathscr{F}_{\mathsf{mcomdl}}$, in the committing phase, enables a committer C to send a message m to the functionality. While in the opening phase, $\mathscr{F}_{\mathsf{mcomdl}}$ passes $g^m \in \mathbb{G}$ to a receiver R (but the standard functionality of [53] passes the message m). As the functionality $\mathscr{F}_{\mathsf{mcomdl}}$ stores m, the UC simulator gets to know m and can simulate the corrupted committer.

Then, we construct a UC-secure commitment Γ_{dl} that realizes the functionality $\mathscr{F}_{\mathsf{mcomdl}}$ in the RPK model. Essentially, our UC-secure commitment Γ_{dl} is built on CRS-model based Fujisaki's UC-commitment scheme Fuj [80] with some major modifications (see Section 4.6.2). Briefly, we first split the CRS of the Fuj construction into two independent parts, one guarantees the binding property and another part is guaranteeing the hiding property. Then we represent it in the RPK model. [5]

After that, in the commit phase, we equip Fuj construction with the efficient IND-CPA secure Short Cramer-Shoup (SCS, [3]) public-key cryptosystem. We add a Σ-protocol Σ_{eq} (see Section 4.6.2) to prove the knowledge of the discrete logarithm m of the SCS-encrypted message. Additionally, in order to satisfy the extractability of m, we equip it with an integer commitment [62] and an additively homomorphic Paillier encryption [125] of m. More precisely we use the Paillier encryption for the extraction phase and use the integer commitment to prove that the SCS encryption and Paillier encryption of m are mutually consistent. Finally, by the combination of the above tools, we construct the DL-extractable UC-commitment scheme Γ_{dl} where committer can open a commitment to a group element g^m; while given the secret key of the Paillier encryption, the UC simulator can extract m from a corrupted committer, thanks to the Paillier encryption. A full description of our construction is given in the paper [6], which is joint work with Baghery, Lipmaa, Siim, and Zajac.

4.6.1. New Ideal functionality $\mathscr{F}_{\mathsf{mcomdl}}$

We define the new ideal functionality $\mathscr{F}_{\mathsf{mcomdl}}$, in a way that it lets parties commit to an integer m but open the commitment to g^m. Importantly the functionality $\mathscr{F}_{\mathsf{mcomdl}}$ stores the integer m which later enables the UC simulator Sim to extract m. Therefore, any commitment scheme should necessarily be DL-extractable if it realizes the $\mathscr{F}_{\mathsf{mcomdl}}$ functionality. See Fig. 10 for the details of the $\mathscr{F}_{\mathsf{mcomdl}}$

[5] Note that the RPK model is relatively unknown in the community, so reintroducing it and constructing an efficient commitment scheme in this model can be seen as another contribution of this work.

$\mathcal{F}_{\mathsf{mcomdl}}$ is parameterized by $\mathscr{M} = \mathbb{Z}_p$ and $\mathbb{G}$, interacts with the party $\mathcal{G}_i \in \{\mathsf{C}, \mathsf{R}\}$ for $i \in [\nu]$ (where ν is the number of the parties) as follows.

Upon receiving $(\texttt{commit}, \mathsf{sid}, \mathsf{cid}, \mathcal{G}_i, \mathcal{G}_j, m)$ from $\mathcal{G}_i$, where $m \in \mathbb{Z}_p$: if a tuple $(\mathsf{sid}, \mathsf{cid}, \cdots)$ with the same $(\mathsf{sid}, \mathsf{cid})$ was previously recorded, do nothing. Otherwise, record $(\mathsf{sid}, \mathsf{cid}, \mathcal{G}_i, \mathcal{G}_j, m)$ and send $(\texttt{rcpt}, \mathsf{sid}, \mathsf{cid}, \mathcal{G}_i, \mathcal{G}_j)$ to $\mathcal{G}_j$ and Sim.

Upon receiving $(\texttt{open}, \mathsf{sid}, \mathsf{cid})$ from $\mathcal{G}_i$, proceed as follows: if a tuple $(\mathsf{sid}, \mathsf{cid}, \mathcal{G}_i, \mathcal{G}_j, m)$ was previously recorded then send $(\texttt{open}, \mathsf{sid}, \mathsf{cid}, \mathcal{G}_i, \mathcal{G}_j, y \leftarrow g^m)$ to $\mathcal{G}_j$ and Sim. Otherwise do nothing.

Figure 10. DL-extractable functionality $\mathcal{F}_{\mathsf{mcomdl}}$ for committing multiple messages

functionality [6]. Following the functionality of [53] we define sid to be a standard *commitment identifier* [7]. We use another unique *commitment identifier* cid, for the case when a committer C commits to the same receiver R multiple times within a session. Besides, we assume that the combination of sid and cid is globally unique.

4.6.2. DL-Extractable UC-Commitment Scheme

We construct the DL-extractable UC-commitment scheme by implementing the functionality $\mathcal{F}_{\mathsf{mcomdl}}$ as follows: (i) for $m \in \mathbb{Z}_p$, by using the SCS [3], we encrypt the group element g^m, (ii) by using the additively homomorphic Paillier public-key cryptosystem [125], encrypt the integer m, and (iii) finally use Damgård-Fujisaki [62]'s commitment and commit to the integer m.

After that for proving the knowledge of integer m that was used in the aforementioned schemes, we equip the scheme with a Σ-protocol Σ_{eq}. Importantly, from Σ_{eq} one can only extract g^m and so m itself remains secret. As it is known that the UC-security property does not permit us to perform rewinding to retrieve m, we use straight-line extraction techniques from [80].

During the commit phase, we use the Σ-protocol, and after that, the used random coins will be erased (by the committer C). Then C opens the commitment to g^m by ending Σ_{eq} during the open phase. In the simulation phase, for simulating a corrupted committer C, given the secret key of the Paillier encryption, the UC simulator Sim decrypts the Paillier encryption of m and obtains m. (This satisfies extractability.) If C is honest, then by utilizing a trapdoor commitment scheme, the UC simulator Sim first commits to 0 and then simulates Σ_{eq}. (This satisfies equivocability.). Therefore, we reach a DL-extractable UC-commitment scheme.

In the following, we explain a more detailed intuition of the structure of the

[6] This is parameterized by $\mathbb{G}$ and $\mathbb{Z}_p$ which means $\mathbb{G}$ and $\mathbb{Z}_p$ are hard-coded into the functionality.

[7] The standard *commitment identifier* sid shows the case when a committer C commits to a receiver R within a session.

DL-extractable UC-commitment scheme Γ_{dl}. We first recall the Σ-protocol [59] in the RPK model.

Σ-protocols [59] in the RPK model. Let $\mathscr{R} = \{x, w\}$ be an NP-relation. A Σ-protocol $\Sigma = (\Sigma.\mathsf{P}_1, \Sigma.\mathsf{P}_2, \Sigma.\mathsf{Vf}, \Sigma.\mathsf{Sim})$ is a three-round protocol between the prover P and the verifier V, such that the first and the third messages are by the prover, and the second message is by the verifier. Let $\mathsf{rpk}_{\mathsf{V}}$ be the public key of the verifier. P has input $(\mathsf{rpk}_{\mathsf{V}}; x, w)$ and V has input $(\mathsf{rpk}_{\mathsf{V}}; x)$. The first message is denoted as $a \leftarrow \Sigma.\mathsf{P}_1(\mathsf{rpk}_{\mathsf{V}}; x, w; s)$, where $s \leftarrow_{\$} \mathsf{RND}_{\lambda}\Sigma$ is sampled from the randomizer space of the protocol. The second message e is chosen uniformly at random from $\{0,1\}^{\lambda}$, $e \leftarrow_{\$} \{0,1\}^{\lambda}$. The third message is denoted as $z \leftarrow \Sigma.\mathsf{P}_2(\mathsf{rpk}_{\mathsf{V}}; x, w; e; s)$. The verifier accepts iff $\Sigma.\mathsf{Vf}(\mathsf{rpk}_{\mathsf{V}}; x; a, e, z) = 1$.

A Σ-protocol is *complete* for $\mathscr{R}$ if a honest verifier always accepts a honest prover. A Σ-protocol is *specially sound* for $\mathscr{R}$ if given an input x and two acceptable views (a, e_1, z_1) and (a, e_2, z_2), $e_1 \neq e_2$, one can efficiently extract a witness w, such that $(x, w) \in \mathscr{R}$. A Σ-protocol is *statistically special honest-verifier zero-knowledge (SSHVZK)* for $\mathscr{R}$ if for any $\mathsf{rpk}_{\mathsf{V}}$, x and e, $\Sigma.\mathsf{Sim}(\mathsf{rpk}_{\mathsf{V}}; x, e)$ can first choose a z and then a, such that the simulated view (a, e, z) and the real view, given the same e, have negligible statistical distance.

Σ-Protocol Σ_{eq}. Assume Pai be the Paillier cryptosystem and SCS be the SCS cryptosystem. Recall that the plaintext space of SCS is $\mathbb{G}$ (of order p) and the plaintext space of Pai is $\mathbb{Z}_N$ for an $N > p$. The modulo $N = PQ$ where $P = 2P' + 1$ and $Q = 2Q' + 1$ are safe primes. (The case $N = p$ is straightforward to handle).

As stated before, the main core of the construction of the DL-extractable UC-commitment Γ_{dl} is the new Σ_{eq}-protocol. We utilize Σ_{eq} to prove the knowledge of the discrete logarithm m of the SCS encryption of g^m in Fuj construction [80] instead of its Σ-protocol. Let

$$\mathscr{R}_{\mathsf{eq}} = \left\{ \begin{array}{l} (x = (\mathsf{p}, \mathsf{SCS.pk}_{\mathsf{P}}, \mathsf{Pai.pk}_{\mathsf{P}}, g^m, \mathsf{c}_1, \mathsf{c}_2, \mathsf{lbl}), w = (m', r_1, r_2)) : \\ \mathsf{c}_1 = \mathsf{SCS.Enc}^{\mathsf{lbl}}_{\mathsf{SCS.pk}_{\mathsf{P}}}(g^m; r_1) \wedge \mathsf{c}_2 = \mathsf{Pai.Enc}_{\mathsf{Pai.pk}_{\mathsf{P}}}(m'; r_2) \wedge \\ m \equiv m' \pmod{p} \wedge m' < N \end{array} \right\},$$

where $\mathsf{p} \leftarrow \mathsf{Pgen}(1^{\lambda})$, lbl is a label of the SCS cryptosystem and $m', m \in \mathbb{Z}_p$ are the secret values. Here, $\mathsf{SCS.Enc}(.)$ and $\mathsf{SCS.pk}$ are respectively the encryption algorithm and the public key of the SCS cryptosystem. $\mathsf{Pai.Enc}(.)$ and $\mathsf{Pai.pk}$ are respectively the encryption algorithm and the public key of the Paillier cryptosystem. Let $\mathscr{L}_{\mathsf{eq}} = \{x : \exists w, (x, w) \in \mathscr{R}_{\mathsf{eq}}\}$ be the corresponding language. Thus, $x \in \mathscr{L}_{\mathsf{eq}}$ iff the two ciphertexts c_1 and c_2 encrypt g^m and m' respectively, such that $m \equiv m' \pmod{p}$. We depict the full of the Σ_{eq}-protocol for the relation $\mathscr{R}_{\mathsf{eq}}$ in Fig. 11. Here, $\mathsf{DF.Commit}(.)$ and $\mathsf{DF.ck}$ are respectively the commitment algorithm and the public key of the construction of Damgård-Fujisaki [62]. Let T be

1. Denote $\mathsf{RND}_\lambda\Sigma_{\mathsf{eq}} := \mathbb{Z}_{2^{B+\lambda}} \times \mathbb{Z}_p \times \{0,\ldots,\max(2^{2\lambda}p, T\cdot C(\lambda)\cdot 2^\lambda)-1\} \times \mathbb{Z}^*_{N^2} \times \{0,\ldots,C(\lambda)2^{B+2\lambda}-1\}$. P samples $s := (s_1,s_2,s_3,s_4,s_5) \leftarrow_\$ \mathsf{RND}_\lambda\Sigma_{\mathsf{eq}}$.
   P sets $\tilde{a}_1 \leftarrow \mathsf{DF.Commit}(\mathsf{DF.ck_V}; m; s_1)$, $a_2 \leftarrow (g^{s_2}, h^{s_2}, (cd^\tau)^{s_2})^\top$, $a_3 \leftarrow g^{s_3}$, $a_4 \leftarrow (1+s_3N)s_4^N \mod N^2$, $\tilde{a}_5 \leftarrow \tilde{g}^{s_3}\tilde{h}^{s_5}$.
   P sends $a \leftarrow \Sigma_{\mathsf{eq}}.\mathsf{P}_1(\mathsf{rpk_V}; x, w; s) := (\tilde{a}_1, a_2, a_3, a_4, \tilde{a}_5)$ to V.
2. V sends $e \leftarrow_\$ \{0,1\}^\lambda$ to P.
3. P sets $z_1 \leftarrow r_1 e + s_2$, $z_2 \leftarrow me + s_3$, $z_3 \leftarrow r_2^e s_4 \mod N^2$, $z_4 \leftarrow s_1 e + s_5$. Let $z := (z_1,z_2,z_3,z_4)$. P sends $\Sigma_{\mathsf{eq}}.\mathsf{P}_2(\mathsf{rpk_V}; x, w; e; s) := z$ to V.
4. V outputs 1 iff the following holds (otherwise, V outputs 0):
   (a) $\mathsf{c}_1^e a_2 = (g^{z_1}, g^{em}h^{z_1}, (cd^\tau)^{z_1})^\top$,
   (b) $g^{em}\cdot a_3 = g^{z_2}$,
   (c) $\mathsf{c}_2^e a_4 \equiv (1+N)^{z_2} z_3^N \pmod{N^2}$,
   (d) $\tilde{a}_1^e \tilde{a}_5 = \tilde{g}^{z_2}\tilde{h}^{z_4}$, and
   (e) $z_2 \in \{-T\cdot C(\lambda), \cdots, T\cdot C(\lambda)(2^\lambda+1)\}$.
   Denote this check by $\Sigma_{\mathsf{eq}}.\mathsf{Vf}(\mathsf{rpk_V}; x; a, e, z) \in \{0,1\}$.

Figure 11. Σ-protocol Σ_{eq} for $\mathscr{R}_{\mathsf{eq}}$.

a public constant such that $m < T$, e.g. $T = p$. Let $C(\lambda)$ be a function from $\mathbb{Z}^+$ to $\mathbb{Z}^+$, such that $C(\lambda)$ is superpolynomial ($C(\lambda) = 2^\lambda$). Let 2^B be a close upper bound on the order of the group $\tilde{\mathbb{G}}$.[8]

We note that g^m is public while the message m is not; this corresponds to the use of g^m in the new DL-extractable UC-commitment. We remark that in Fig. 11, in the honest case, $\mathsf{c}_1 = (\mathsf{c}_{11}, \mathsf{c}_{12}, \mathsf{c}_{13})^\top \leftarrow \mathsf{SCS.Enc}^{\mathsf{lbl}}_{\mathsf{SCS.pk_P}}(g^m; r_1) = (g^{r_1}, g^m h^{r_1}, (cd^\tau)^{r_1})^\top$ and $\mathsf{c}_2 \leftarrow \mathsf{Pai.Enc}_{\mathsf{Pai.pk_P}}(m; r_2) = (1+N)^m r_2^N \equiv (1+mN)r_2^N \mod N^2$. Here, $r_1 \leftarrow_\$ \mathbb{Z}_p$, $\tau = \mathsf{H}(\mathsf{lbl}, \mathsf{c}_{11}, \mathsf{c}_{12})$, and $r_2 \leftarrow_\$ \mathbb{Z}^*_N$. In Theorem 1 of [7], we prove the security of Σ-protocol Σ_{eq} in Fig. 11.

Construction of DL-extractable UC-commitment. Now we construct a DL-extractable UC-commitment scheme Γ_{dl} based on Fujisaki's UC-commitment scheme Fuj [80]. We depict the full construction of Γ_{dl} in Fig. 12. As noted before, we first move Fuj's setup to the RPK model and so use the public key rpk notation instead of the CRS crs. We split the public key rpk_i of $\mathscr{G}_i \in \{\mathsf{C}, \mathsf{R}\}$ into the binding part (rpk^b) and the hiding part (rpk^h). More precisely when the party $\mathscr{G}_i$ acts as the receiver R, the corresponding public key rpk_i contains the binding part. While rpk_i contains the hiding part when the party $\mathscr{G}_i$ acts as the committer C. Therefor $\mathsf{rpk} = (\mathsf{rpk}^b, \mathsf{rpk}^h)$.

[8] $\tilde{\mathbb{G}} = \mathbb{U} \times \mathbb{H}$ is a multiplicative abelian group such that $\mathbb{H}$ has order divisible only by large primes. Notice that if $\tilde{\mathbb{G}}$ is the multiplicative group modulo $N = PQ$ where $P = 2P'+1$ and $Q = 2Q'+1$ are safe primes, then the order of $\tilde{\mathbb{G}} = 4P'Q'$ (this setting is often recommended if one uses the Paillier cryptosystem [125].).

Then, we utilize SCS public-key cryptosystem [3], the additively homomorphic Paillier encryption Pai [125], the pederson commitment Ped, and the sigma-protocol Σ_{eq} during the *committing phase* $\Gamma_{\mathsf{dl}}.\mathsf{Commit}$. In the *opening phase* $\Gamma_{\mathsf{dl}}.\mathsf{Open}$. we employ the Ped and Σ_{eq} schemes. Finally, the extraction can be done by decrypting the Pai's ciphertext in the extraction phase $\Gamma_{\mathsf{dl}}.\mathsf{Ext}$.

We define the hiding part $\mathsf{rpk}^{\mathsf{h}} = (\mathsf{p}, \mathsf{SCS.pk}, \mathsf{Pai.pk}, \mathsf{H}^{\mathsf{h}})$ where $\mathsf{SCS.pk}$, $\mathsf{Pai.pk}$, and H^{h} are respectively the public key of SCS encryption scheme, the public key of Pai encryption scheme, and the collision-resistant hash function. The binding part $\mathsf{rpk}^{\mathsf{b}} = (\mathsf{p}, \mathsf{Ped.ck}, \mathsf{DF.ck}, \mathsf{H}^{\mathsf{b}})$ where $\mathsf{Ped.ck}$, $\mathsf{DF.ck}$, and H^{b} are respectively the public key of Ped commitment scheme, the public key of Damgård-Fujisaki's integer commitment DF [62], and the collision-resistant hash function. See Fig. 12 for the full description of the DL-extractable UC-commitment Γ_{dl}.

As usual in the RPK model, the algorithm $\Gamma_{\mathsf{dl}}.\mathsf{Gen}$ of Fig. 12 for party $\mathscr{G}_i \in \{\mathsf{C}, \mathsf{R}\}$ is run by the key registration authority $\mathscr{R}_i$, the algorithms $\Gamma_{\mathsf{dl}}.\mathsf{Commit}$ and $\Gamma_{\mathsf{dl}}.\mathsf{Open}$ are run by the committer C, and the algorithm $\Gamma_{\mathsf{dl}}.\mathsf{Vf}$ is run by the receiver R. We remark that the algorithms $\Gamma_{\mathsf{dl}}.\mathsf{tdOpen}$ and $\Gamma_{\mathsf{dl}}.\mathsf{Ext}$ are only used for the security proof of Γ_{dl} scheme. Finally, in theorem 2 of [7], we prove the Γ_{dl} scheme from Fig. 12 is a secure DL-extractable UC-commitment scheme in the RPK model.

$\Gamma_{\mathsf{dl}}.\mathsf{Gen}(1^\lambda)$: Generate new keys $(\mathsf{SCS.pk}, \mathsf{SCS.sk})$ for SCS, $(\mathsf{Pai.pk}, \mathsf{Pai.sk})$ for Pai, $(\mathsf{DF.ck}, \mathsf{DF.td})$ for DF, and $(\mathsf{Ped.ck}, \mathsf{Ped.td})$ for Ped. Choose collision-resistant hash functions $\mathsf{H}^{\mathsf{h}}, \mathsf{H}^{\mathsf{b}}$. Let $\mathsf{p} \leftarrow \mathsf{Pgen}(1^\lambda)$. Let $\mathsf{rpk} = (\mathsf{rpk}^{\mathsf{h}}, \mathsf{rpk}^{\mathsf{b}})$ where $\mathsf{rpk}^{\mathsf{h}} = (\mathsf{p}, \mathsf{SCS.pk}, \mathsf{Pai.pk}, \mathsf{H}^{\mathsf{h}})$ and $\mathsf{rpk}^{\mathsf{b}} = (\mathsf{p}, \mathsf{Ped.ck}, \mathsf{DF.ck}, \mathsf{H}^{\mathsf{b}})$. Let $\mathsf{td} = (\mathsf{td}^{\mathsf{h}}, \mathsf{td}^{\mathsf{b}})$, where $\mathsf{td}^{\mathsf{h}} = (\mathsf{SCS.sk}, \mathsf{Pai.sk})$ and $\mathsf{td}^{\mathsf{b}} = (\mathsf{Ped.td}, \mathsf{DF.td})$.

Return $(\mathsf{rpk}, \mathsf{td})$. // The equivocability td is Ped.td; The extraction td is Pai.sk;

$\Gamma_{\mathsf{dl}}.\mathsf{Commit}(\mathsf{rpk_C}; \mathsf{lbl}, m)$ **where** $\mathsf{lbl} = (\mathsf{sid}, \mathsf{cid}, \mathsf{C}, \mathsf{R})$**:** to commit to $m \in \mathbb{Z}_p$ for R upon receiving $(\texttt{commit}, \mathsf{lbl}, m)$, C does the following.

1. Obtain $\mathsf{rpk_R} = (\mathsf{rpk}^{\mathsf{h}}_{\mathsf{R}}, \mathsf{rpk}^{\mathsf{b}}_{\mathsf{R}})$ from $\mathscr{R}_{\mathsf{R}}$;
   $\mathsf{ck_{CR}} \leftarrow (\mathsf{rpk}^{\mathsf{h}}_{\mathsf{C}}, \mathsf{rpk}^{\mathsf{b}}_{\mathsf{R}})$; $\Sigma_{\mathsf{eq}}.\mathsf{rpk_R} \leftarrow \mathsf{DF.ck_R}$;
   $r_1 \leftarrow\!\!\$\, \mathsf{RND}_\lambda \mathsf{SCS}$; $r_2 \leftarrow\!\!\$\, \mathsf{RND}_\lambda \mathsf{Pai}$;
   $\mathsf{c}_1 \leftarrow \mathsf{SCS.Enc}^{\mathsf{lbl}}_{\mathsf{SCS.pk_C}}(g^m; r_1)$; $\mathsf{c}_2 \leftarrow \mathsf{Pai.Enc}_{\mathsf{Pai.pk_C}}(m; r_2)$;
   $\mathsf{c} \leftarrow (\mathsf{c}_1, \mathsf{c}_2)$;
   $x \leftarrow (\mathsf{p}, \mathsf{SCS.pk_C}, \mathsf{Pai.pk_C}, g^m, \mathsf{c}, \mathsf{lbl})$; $w \leftarrow (m, r_1, r_2)$;
   $s \leftarrow\!\!\$\, \mathsf{RND}_\lambda \Sigma_{\mathsf{eq}}$; $a \leftarrow \Sigma_{\mathsf{eq}}.\mathsf{P}_1(\Sigma_{\mathsf{eq}}.\mathsf{rpk_R}; x, w; s)$; $h_x \leftarrow \mathsf{H}^{\mathsf{b}}_{\mathsf{R}}(\mathsf{lbl}, x, a)$;
   (*) $r_3 \leftarrow\!\!\$\, \mathsf{RND}_\lambda \mathsf{Ped}$; $\mathsf{c}_3 \leftarrow \mathsf{Ped.Commit}(\mathsf{Ped.ck_R}; h_x; r_3)$;
   Send $(\mathsf{lbl}, \mathsf{c}_3)$ to R;
2. After obtaining $(\mathsf{lbl}, \mathsf{c}_3)$, R fetches $\mathsf{rpk_C} = (\mathsf{rpk}^{\mathsf{h}}_{\mathsf{C}}, \mathsf{rpk}^{\mathsf{b}}_{\mathsf{C}})$ from $\mathscr{R}_{\mathsf{C}}$, sets $\mathsf{ck_{CR}}$ as above, and checks that c_3 is a valid ciphertext. If yes, he sets $e \leftarrow\!\!\$\, \{0,1\}^\lambda$ and sends (lbl, e) to C. Otherwise, R ignores it.
3. After receiving (lbl, e), C does the following.
   $z \leftarrow \Sigma_{\mathsf{eq}}.\mathsf{P}_2(\Sigma_{\mathsf{eq}}.\mathsf{rpk_R}; x, w; e; s)$;
   Securely delete $(w = (m, r_1, r_2), s)$;
   $\mathsf{op} \leftarrow (\mathsf{lbl}, \mathsf{c}_3, e; a, z, r_3)$; Store $\mathsf{st_C} = (\mathsf{c}, g^m, \mathsf{op})$;
   Output $(\mathsf{c}, \mathsf{op})$ (privately); Send $(\texttt{com}, \mathsf{lbl}, \mathsf{c})$ to R;
4. R checks that $\mathsf{c} = (\mathsf{c}_1, \mathsf{c}_2) \in \mathbb{G}^3 \times \mathbb{Z}_{N^2}$. If yes, R outputs $(\texttt{rcpt}, \mathsf{lbl})$, and stores $\mathsf{st_R} \leftarrow (\mathsf{lbl}, \mathsf{c}_3, e, \mathsf{c} = (\mathsf{c}_1, \mathsf{c}_2))$. Otherwise, R ignores it.

$\Gamma_{\mathsf{dl}}.\mathsf{Open}(\mathsf{st_C})$**:** upon receiving $(\texttt{open}, \mathsf{sid}, \mathsf{cid})$, C sends (g^m, op) to R.

$\Gamma_{\mathsf{dl}}.\mathsf{Vf}(\mathsf{ck_{CR}}; \mathsf{c}, g^m, \mathsf{op})$ **where** $\mathsf{c} = (\mathsf{c}_1, \mathsf{c}_2)$**:** upon receiving $(g^m, \mathsf{op} = (\mathsf{lbl}, \mathsf{c}_3, e; a, z, r_3))$ where $\mathsf{lbl} = (\mathsf{sid}, \mathsf{cid}, \mathsf{C}, \mathsf{R})$, R does the following.

1. $x \leftarrow (\mathsf{p}, \mathsf{SCS.pk_C}, \mathsf{Pai.pk_C}, g^m, \mathsf{c}, \mathsf{lbl})$; $h_x \leftarrow \mathsf{H}^{\mathsf{b}}_{\mathsf{R}}(\mathsf{lbl}, x, a)$;
2. If cid has not been used with C, $(\mathsf{lbl}, \mathsf{c}_3, e)$ are the same as in the commit phase, $\mathsf{c}_3 = \mathsf{Ped.Commit}(\mathsf{Ped.ck_R}; h_x; r_3)$, and $\Sigma_{\mathsf{eq}}.\mathsf{Vf}(\Sigma_{\mathsf{eq}}.\mathsf{rpk_R}; x; a, e, z) = 1$, then output $(\texttt{open}, \mathsf{lbl}, g^m)$. Otherwise, ignore the message.

$\Gamma_{\mathsf{dl}}.\mathsf{tdOpen}(\mathsf{rpk_C}, \mathsf{td}^{\mathsf{b}}_{\mathsf{R}}; \mathsf{c}, g^m, \mathsf{op}, g^{m'})$:

$x \leftarrow (\mathsf{p}, \mathsf{SCS.pk_C}, \mathsf{Pai.pk_C}, g^m, \mathsf{c}, \mathsf{lbl})$; $x' \leftarrow (\mathsf{p}, \mathsf{SCS.pk_C}, \mathsf{Pai.pk_C}, g^{m'}, \mathsf{c}, \mathsf{lbl})$;
$(a', z') \leftarrow \Sigma_{\mathsf{eq}}.\mathsf{Sim}(\Sigma_{\mathsf{eq}}.\mathsf{rpk_R}; x', e)$;
$\mathsf{op}' \leftarrow \mathsf{Ped.tdOpen}(\mathsf{Ped.td_R}; \mathsf{H}^{\mathsf{b}}_{\mathsf{R}}(\mathsf{lbl}, x, a), r_3, \mathsf{H}^{\mathsf{b}}_{\mathsf{R}}(\mathsf{lbl}, x', a'))$;
return op';

$\Gamma_{\mathsf{dl}}.\mathsf{Ext}(\mathsf{td}^{\mathsf{h}}_{\mathsf{C}}; \mathsf{c})$**:** **return** $\mathsf{Pai.Dec}_{\mathsf{Pai.sk}}(\mathsf{c}_2) \mod p$;

Figure 12. The commitment scheme Γ_{dl} in the RPK model

5. CONCLUSION

5.1. Conclusion

In this thesis, we investigate the important issues for the practical use of NIZKs that are (i) the need of a trusted setup for generating CRS of NIZKs and (ii) adding universal composability to guarantee that security is preserved if the protocol is concurrently run with other protocols or even copies of itself.

In the line of these subjects, we present some concrete constructions that provide the aforementioned properties. For the case (i) and in the line of diminishing the trust, we show how one can construct a subversion zero-knowledge version of the most efficient zk-SNARK [94] and the most efficient QA-NIZK [106] constructions where the prover does not need to trust the CRS generator. More formally the zero-knowledge property of the constructions remains when the CRS generator is subverted. In line with the case (ii), in terms of universal composability issue, we first utilize SPHFs and introduce a new cryptographic primitive called publicly computable SPHFs (PC-SPHFs) that has a lot of uses in building cryptography protocols. Then we explicate how one can construct the currently most efficient non-interactive UC-secure commitment by using labeled CS encryption scheme and PC-SPHF on labeled CS ciphertexts. Finally, we develop a new technique for constructing a UC-secure commitment scheme called DL-extractable commitment that that is compatible with MPC approach (enables one to generate CRS of zk-SNARKs in a UC secure manner).

5.2. Open Questions

In this part, we highlight some open questions left in the research line of the thesis.

I Both subversion constructions of zk-SNARKs and QA-NIZK are built in an ad-hoc manner. Thus the interesting question left is whether we can design such construction in a generic way?

II In the line of constructing (generic) UC-secure commitment and in the lattice-based direction, Katz and Vaikuntanathan [103] constructed the first (approximate) SPHF for a lattice-based language. Later Benhamouda *et al.* [29] improved the Katz-Vaikuntanathan construction, where the construction is over a tag-based IND-CCA encryption scheme a la Micciancio-Peikert [121]. Now an interesting question is if we can construct PC-SPHFs for the class of lattice-based languages? Then can we construct a post-quantum UC-secure commitment scheme?

III As stated before the DL-extractable commitment scheme is the first commitment scheme that allows one to open a commitment to a group element g^x; but the UC simulator is able to extract its discrete logarithm x. In fact, our construction is interactive and so the question left here is whether we can construct a non-interactive version of it?

BIBLIOGRAPHY

[1] Michel Abdalla, Fabrice Benhamouda, Olivier Blazy, Céline Chevalier, and David Pointcheval. SPHF-friendly non-interactive commitments. In Kazue Sako and Palash Sarkar, editors, *ASIACRYPT 2013, Part I*, volume 8269 of *LNCS*, pages 214–234. Springer, Heidelberg, December 2013.

[2] Michel Abdalla, Fabrice Benhamouda, and David Pointcheval. Disjunctions for hash proof systems: New constructions and applications. In Elisabeth Oswald and Marc Fischlin, editors, *EUROCRYPT 2015, Part II*, volume 9057 of *LNCS*, pages 69–100. Springer, Heidelberg, April 2015.

[3] Michel Abdalla, Fabrice Benhamouda, and David Pointcheval. Public-key encryption indistinguishable under plaintext-checkable attacks. In Jonathan Katz, editor, *PKC 2015*, volume 9020 of *LNCS*, pages 332–352. Springer, Heidelberg, March / April 2015.

[4] Michel Abdalla, Fabrice Benhamouda, and David Pointcheval. Removing erasures with explainable hash proof systems. In Serge Fehr, editor, *PKC 2017, Part I*, volume 10174 of *LNCS*, pages 151–174. Springer, Heidelberg, March 2017.

[5] Michel Abdalla, Céline Chevalier, and David Pointcheval. Smooth projective hashing for conditionally extractable commitments. In Shai Halevi, editor, *CRYPTO 2009*, volume 5677 of *LNCS*, pages 671–689. Springer, Heidelberg, August 2009.

[6] Behzad Abdolmaleki, Karim Baghery, Helger Lipmaa, Janno Siim, and Michal Zajac. DL-extractable UC-commitment schemes. In Robert H. Deng, Valérie Gauthier-Umaña, Martín Ochoa, and Moti Yung, editors, *ACNS 19*, volume 11464 of *LNCS*, pages 385–405. Springer, Heidelberg, June 2019.

[7] Behzad Abdolmaleki, Karim Baghery, Helger Lipmaa, Janno Siim, and Michal Zajac. UC-secure CRS generation for SNARKs. In Johannes Buchmann, Abderrahmane Nitaj, and Tajje eddine Rachidi, editors, *AFRICACRYPT 19*, volume 11627 of *LNCS*, pages 99–117. Springer, Heidelberg, July 2019.

[8] Behzad Abdolmaleki, Karim Baghery, Helger Lipmaa, and Michał Zajac. A Subversion-Resistant SNARK. Technical Report 2017/599, IACR, June 21, 2017. Available from `http://eprint.iacr.org/2017/599`.

[9] Behzad Abdolmaleki, Karim Baghery, Helger Lipmaa, and Michal Zajac. A subversion-resistant SNARK. In Tsuyoshi Takagi and Thomas Peyrin, editors, *ASIACRYPT 2017, Part III*, volume 10626 of *LNCS*, pages 3–33. Springer, Heidelberg, December 2017.

[10] Behzad Abdolmaleki, Hamidreza Khoshakhlagh, and Daniel Slamanig. A framework for uc-secure commitments from publicly computable smooth

projective hashing. In Martin Albrecht, editor, *Cryptography and Coding*, pages 1–21, Cham, 2019. Springer International Publishing.

[11] Behzad Abdolmaleki, Helger Lipmaa, Janno Siim, and Michal Zajac. On QA-NIZK in the BPK model. In Aggelos Kiayias, Markulf Kohlweiss, Petros Wallden, and Vassilis Zikas, editors, *PKC 2020, Part I*, volume 12110 of *LNCS*, pages 590–620. Springer, Heidelberg, May 2020.

[12] Behzad Abdolmaleki, Helger Lipmaa, Janno Siim, and Michal Zajac. On subversion-resistant snarks. Cryptology ePrint Archive, Report 2020/668, 2020. `https://eprint.iacr.org/2020/668`.

[13] Behzad Abdolmaleki, Sebastian Ramacher, and Daniel Slamanig. Lift-and-shift: Obtaining simulation extractable subversion and updatable snarks generically. Cryptology ePrint Archive, Report 2020/062, 2020. `https://eprint.iacr.org/2020/062`.

[14] Behzad Abdolmaleki, Sebastian Ramacher, and Daniel Slamanig. Sok: Lifting transformations for simulation extractable subversion and updatable snarks. The 3rd ZKProof Workshop, 2020. `https://docs.zkproof.org/pages/standards/accepted-workshop3/sok-lifting_transformations_se_snarks.pdf`.

[15] Behzad Abdolmaleki and Daniel Slamanig. Unbounded simulation-sound subversion resistant quasi-adaptive nizk proofs and applications to modular zk-snarks. Cryptology ePrint Archive, Report 2020/364, 2020. `https://eprint.iacr.org/2020/364`.

[16] Masayuki Abe and Serge Fehr. Perfect NIZK with adaptive soundness. In Salil P. Vadhan, editor, *TCC 2007*, volume 4392 of *LNCS*, pages 118–136. Springer, Heidelberg, February 2007.

[17] Masayuki Abe, Charanjit S. Jutla, Miyako Ohkubo, and Arnab Roy. Improved (almost) tightly-secure simulation-sound QA-NIZK with applications. In Thomas Peyrin and Steven Galbraith, editors, *ASIACRYPT 2018, Part I*, volume 11272 of *LNCS*, pages 627–656. Springer, Heidelberg, December 2018.

[18] Giuseppe Ateniese, Jan Camenisch, Marc Joye, and Gene Tsudik. A practical and provably secure coalition-resistant group signature scheme. In Mihir Bellare, editor, *CRYPTO 2000*, volume 1880 of *LNCS*, pages 255–270. Springer, Heidelberg, August 2000.

[19] Boaz Barak, Ran Canetti, Jesper Buus Nielsen, and Rafael Pass. Universally composable protocols with relaxed set-up assumptions. In *45th FOCS*, pages 186–195. IEEE Computer Society Press, October 2004.

[20] Razvan Barbulescu and Sylvain Duquesne. Updating key size estimations for pairings. *Journal of Cryptology*, 32(4):1298–1336, Oct 2019.

[21] Mira Belenkiy, Jan Camenisch, Melissa Chase, Markulf Kohlweiss, Anna Lysyanskaya, and Hovav Shacham. Randomizable proofs and delegatable

anonymous credentials. In Shai Halevi, editor, *CRYPTO 2009*, volume 5677 of *LNCS*, pages 108–125. Springer, Heidelberg, August 2009.

[22] Mihir Bellare, Georg Fuchsbauer, and Alessandra Scafuro. NIZKs with an untrusted CRS: Security in the face of parameter subversion. In Jung Hee Cheon and Tsuyoshi Takagi, editors, *ASIACRYPT 2016, Part II*, volume 10032 of *LNCS*, pages 777–804. Springer, Heidelberg, December 2016.

[23] Mihir Bellare, Juan A. Garay, and Tal Rabin. Batch Verification with Applications to Cryptography and Checking. In Claudio L. Lucchesi and Arnaldo V. Moura, editors, *LATIN 1998*, volume 1380 of *LNCS*, pages 170–191, Campinas, Brazil, April 20–24, 1998. Springer, Heidelberg.

[24] Fabrice Ben Hamouda--Guichoux. *Diverse modules and zero-knowledge*. PhD thesis, École Normale Supérieure, Paris, France, 2016.

[25] Eli Ben-Sasson, Alessandro Chiesa, Christina Garman, Matthew Green, Ian Miers, Eran Tromer, and Madars Virza. Zerocash: Decentralized anonymous payments from bitcoin. In *2014 IEEE Symposium on Security and Privacy*, pages 459–474. IEEE Computer Society Press, May 2014.

[26] Eli Ben-Sasson, Alessandro Chiesa, Matthew Green, Eran Tromer, and Madars Virza. Secure sampling of public parameters for succinct zero knowledge proofs. In *2015 IEEE Symposium on Security and Privacy*, pages 287–304. IEEE Computer Society Press, May 2015.

[27] Fabrice Benhamouda, Olivier Blazy, Céline Chevalier, David Pointcheval, and Damien Vergnaud. New techniques for SPHFs and efficient one-round PAKE protocols. In Ran Canetti and Juan A. Garay, editors, *CRYPTO 2013, Part I*, volume 8042 of *LNCS*, pages 449–475. Springer, Heidelberg, August 2013.

[28] Fabrice Benhamouda, Olivier Blazy, Céline Chevalier, David Pointcheval, and Damien Vergnaud. New techniques for SPHFs and efficient one-round PAKE protocols. Cryptology ePrint Archive, Report 2015/188, 2015. `http://eprint.iacr.org/2015/188`.

[29] Fabrice Benhamouda, Olivier Blazy, Léo Ducas, and Willy Quach. Hash proof systems over lattices revisited. In Michel Abdalla and Ricardo Dahab, editors, *PKC 2018, Part II*, volume 10770 of *LNCS*, pages 644–674. Springer, Heidelberg, March 2018.

[30] Fabrice Benhamouda and David Pointcheval. Trapdoor smooth projective hash functions. Cryptology ePrint Archive, Report 2013/341, 2013. `http://eprint.iacr.org/2013/341`.

[31] Nir Bitansky, Ran Canetti, Omer Paneth, and Alon Rosen. On the existence of extractable one-way functions. In David B. Shmoys, editor, *46th ACM STOC*, pages 505–514. ACM Press, May / June 2014.

[32] Nir Bitansky, Ran Canetti, Omer Paneth, and Alon Rosen. On the Existence of Extractable One-Way Functions. In David Shmoys, editor, *STOC 2014*,

pages 505–514, New York, NY, USA, May 31 – Jun 3, 2014. ACM Press.

[33] Manuel Blum, Paul Feldman, and Silvio Micali. Non-interactive zero-knowledge and its applications (extended abstract). In *20th ACM STOC*, pages 103–112. ACM Press, May 1988.

[34] Dan Boneh. The decision diffie-hellman problem. In *International Algorithmic Number Theory Symposium*, pages 48–63. Springer, 1998.

[35] Dan Boneh, Xavier Boyen, and Eu-Jin Goh. Hierarchical identity based encryption with constant size ciphertext. In Ronald Cramer, editor, *EUROCRYPT 2005*, volume 3494 of *LNCS*, pages 440–456. Springer, Heidelberg, May 2005.

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

[37] Jonathan Bootle, Andrea Cerulli, Pyrros Chaidos, Essam Ghadafi, and Jens Groth. Foundations of fully dynamic group signatures. In Mark Manulis, Ahmad-Reza Sadeghi, and Steve Schneider, editors, *ACNS 16*, volume 9696 of *LNCS*, pages 117–136. Springer, Heidelberg, June 2016.

[38] Jonathan Bootle, Andrea Cerulli, Jens Groth, Sune K. Jakobsen, and Mary Maller. Arya: Nearly linear-time zero-knowledge proofs for correct program execution. In Thomas Peyrin and Steven Galbraith, editors, *ASIACRYPT 2018, Part I*, volume 11272 of *LNCS*, pages 595–626. Springer, Heidelberg, December 2018.

[39] Sean Bowe, Ariel Gabizon, and Matthew D. Green. A multi-party protocol for constructing the public parameters of the pinocchio zk-SNARK. In Aviv Zohar, Ittay Eyal, Vanessa Teague, Jeremy Clark, Andrea Bracciali, Federico Pintore, and Massimiliano Sala, editors, *FC 2018 Workshops*, volume 10958 of *LNCS*, pages 64–77. Springer, Heidelberg, March 2019.

[40] Ernest F. Brickell, Jan Camenisch, and Liqun Chen. Direct anonymous attestation. In Vijayalakshmi Atluri, Birgit Pfitzmann, and Patrick McDaniel, editors, *ACM CCS 2004*, pages 132–145. ACM Press, October 2004.

[41] Ernie Brickell and Jiangtao Li. Enhanced privacy ID from bilinear pairing. Cryptology ePrint Archive, Report 2009/095, 2009. `http://eprint.iacr.org/2009/095`.

[42] Eric Brier, Jean-Sébastien Coron, Thomas Icart, David Madore, Hugues Randriam, and Mehdi Tibouchi. Efficient indifferentiable hashing into ordinary elliptic curves. In Tal Rabin, editor, *CRYPTO 2010*, volume 6223 of *LNCS*, pages 237–254. Springer, Heidelberg, August 2010.

[43] Daniel R. L. Brown. The exact security of ECDSA. Contributions to IEEE P1363a, January 2001. `http://grouper.ieee.org/groups/1363/`.

[44] Jon Buck. Ethereum upgrade byzantium is live, verifes'first zk-snark proof, 2017. https://cointelegraph.com/news/ethereum-upgrade-byzantium-is-live-verifies-first-zk-snark-proof.

[45] Jan Camenisch, Liqun Chen, Manu Drijvers, Anja Lehmann, David Novick, and Rainer Urian. One TPM to bind them all: Fixing TPM 2.0 for provably secure anonymous attestation. In *2017 IEEE Symposium on Security and Privacy*, pages 901–920. IEEE Computer Society Press, May 2017.

[46] Jan Camenisch, Stephan Krenn, Anja Lehmann, Gert Læssøe Mikkelsen, Gregory Neven, and Michael Østergaard Pedersen. Formal treatment of privacy-enhancing credential systems. In Orr Dunkelman and Liam Keliher, editors, *SAC 2015*, volume 9566 of *LNCS*, pages 3–24. Springer, Heidelberg, August 2016.

[47] Jan Camenisch and Anna Lysyanskaya. An efficient system for nontransferable anonymous credentials with optional anonymity revocation. In Birgit Pfitzmann, editor, *EUROCRYPT 2001*, volume 2045 of *LNCS*, pages 93–118. Springer, Heidelberg, May 2001.

[48] Jan Camenisch and Anna Lysyanskaya. A signature scheme with efficient protocols. In Stelvio Cimato, Clemente Galdi, and Giuseppe Persiano, editors, *SCN 02*, volume 2576 of *LNCS*, pages 268–289. Springer, Heidelberg, September 2003.

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

[50] Matteo Campanelli, Dario Fiore, and Anaïs Querol. LegoSNARK: Modular design and composition of succinct zero-knowledge proofs. In Lorenzo Cavallaro, Johannes Kinder, XiaoFeng Wang, and Jonathan Katz, editors, *ACM CCS 2019*, pages 2075–2092. ACM Press, November 2019.

[51] Matteo Campanelli, Rosario Gennaro, Steven Goldfeder, and Luca Nizzardo. Zero-knowledge contingent payments revisited: Attacks and payments for services. In Bhavani M. Thuraisingham, David Evans, Tal Malkin, and Dongyan Xu, editors, *ACM CCS 2017*, pages 229–243. ACM Press, October / November 2017.

[52] Ran Canetti. Universally composable security: A new paradigm for cryptographic protocols. In *42nd FOCS*, pages 136–145. IEEE Computer Society Press, October 2001.

[53] Ran Canetti and Marc Fischlin. Universally composable commitments. In Joe Kilian, editor, *CRYPTO 2001*, volume 2139 of *LNCS*, pages 19–40. Springer, Heidelberg, August 2001.

[54] Ran Canetti, Oded Goldreich, Shafi Goldwasser, and Silvio Micali. Resettable zero-knowledge (extended abstract). In *32nd ACM STOC*, pages 235–244. ACM Press, May 2000.

[55] David Chaum. Showing credentials without identification: Signatures transferred between unconditionally unlinkable pseudonyms. In Franz Pichler, editor, *EUROCRYPT'85*, volume 219 of *LNCS*, pages 241–244. Springer, Heidelberg, April 1986.

[56] David Chaum and Eugène van Heyst. Group signatures. In Donald W. Davies, editor, *EUROCRYPT'91*, volume 547 of *LNCS*, pages 257–265. Springer, Heidelberg, April 1991.

[57] Alessandro Chiesa, Matthew Green, Jingcheng Liu, Peihan Miao, Ian Miers, and Pratyush Mishra. Decentralized anonymous micropayments. In Jean-Sébastien Coron and Jesper Buus Nielsen, editors, *EUROCRYPT 2017, Part II*, volume 10211 of *LNCS*, pages 609–642. Springer, Heidelberg, April / May 2017.

[58] Alessandro Chiesa, Yuncong Hu, Mary Maller, Pratyush Mishra, Noah Vesely, and Nicholas Ward. Marlin: Preprocessing zkSNARKs with universal and updatable SRS. Cryptology ePrint Archive, Report 2019/1047, 2019. `https://eprint.iacr.org/2019/1047`.

[59] Ronald Cramer, Ivan Damgård, and Berry Schoenmakers. Proofs of partial knowledge and simplified design of witness hiding protocols. In Yvo Desmedt, editor, *CRYPTO'94*, volume 839 of *LNCS*, pages 174–187. Springer, Heidelberg, August 1994.

[60] Ronald Cramer and Victor Shoup. Universal hash proofs and a paradigm for adaptive chosen ciphertext secure public-key encryption. In Lars R. Knudsen, editor, *EUROCRYPT 2002*, volume 2332 of *LNCS*, pages 45–64. Springer, Heidelberg, April / May 2002.

[61] Ivan Damgård. Towards practical public key systems secure against chosen ciphertext attacks. In Joan Feigenbaum, editor, *CRYPTO'91*, volume 576 of *LNCS*, pages 445–456. Springer, Heidelberg, August 1992.

[62] Ivan Damgård and Eiichiro Fujisaki. A statistically-hiding integer commitment scheme based on groups with hidden order. In Yuliang Zheng, editor, *ASIACRYPT 2002*, volume 2501 of *LNCS*, pages 125–142. Springer, Heidelberg, December 2002.

[63] Ivan Damgård, Jens Groth, and Gorm Salomonsen. The theory and implementation of an electronic voting system. In Dimitris Gritzalis, editor, *Secure Electronic Voting*, volume 7 of *Advances in Information Security*, pages 77–98. Springer, 2003.

[64] George Danezis, Cédric Fournet, Jens Groth, and Markulf Kohlweiss. Square span programs with applications to succinct NIZK arguments. In Palash Sarkar and Tetsu Iwata, editors, *ASIACRYPT 2014, Part I*, volume 8873 of *LNCS*, pages 532–550. Springer, Heidelberg, December 2014.

[65] Vanesa Daza, Alonso González, Zaira Pindado, Carla Ràfols, and Javier Silva. Shorter quadratic QA-NIZK proofs. In Dongdai Lin and Kazue Sako, editors, *PKC 2019, Part I*, volume 11442 of *LNCS*, pages 314–343. Springer, Heidelberg, April 2019.

[66] Cécile Delerablée and David Pointcheval. Dynamic fully anonymous short group signatures. In Phong Q. Nguyen, editor, *Progress in Cryptology - VIETCRYPT 06*, volume 4341 of *LNCS*, pages 193–210. Springer, Heidelberg, September 2006.

[67] David Derler and Daniel Slamanig. Highly-efficient fully-anonymous dynamic group signatures. In Jong Kim, Gail-Joon Ahn, Seungjoo Kim, Yongdae Kim, Javier López, and Taesoo Kim, editors, *ASIACCS 18*, pages 551–565. ACM Press, April 2018.

[68] David Derler and Daniel Slamanig. Key-homomorphic signatures: definitions and applications to multiparty signatures and non-interactive zero-knowledge. *Designs, Codes and Cryptography*, 87(6):1373–1413, 2019.

[69] Whitfield Diffie and Martin Hellman. New directions in cryptography. *IEEE transactions on Information Theory*, 22(6):644–654, 1976.

[70] Taher ElGamal. A public key cryptosystem and a signature scheme based on discrete logarithms. In G. R. Blakley and David Chaum, editors, *CRYPTO'84*, volume 196 of *LNCS*, pages 10–18. Springer, Heidelberg, August 1984.

[71] Alex Escala, Gottfried Herold, Eike Kiltz, Carla Ràfols, and Jorge Villar. An algebraic framework for Diffie-Hellman assumptions. In Ran Canetti and Juan A. Garay, editors, *CRYPTO 2013, Part II*, volume 8043 of *LNCS*, pages 129–147. Springer, Heidelberg, August 2013.

[72] Prastudy Fauzi, Helger Lipmaa, Janno Siim, and Michal Zajac. An efficient pairing-based shuffle argument. In Tsuyoshi Takagi and Thomas Peyrin, editors, *ASIACRYPT 2017, Part II*, volume 10625 of *LNCS*, pages 97–127. Springer, Heidelberg, December 2017.

[73] Prastudy Fauzi, Helger Lipmaa, and Michał Zając. A Shuffle Argument Secure in the Generic Model. In Jung Hee Cheon and Tsuyoshi Takagi, editors, *ASIACRYPT 2016 (2)*, volume 10032 of *LNCS*, pages 841–872, Hanoi, Vietnam, December 4–8, 2016. Springer, Cham.

[74] Marc Fischlin, Benoît Libert, and Mark Manulis. Non-interactive and re-usable universally composable string commitments with adaptive security. In Dong Hoon Lee and Xiaoyun Wang, editors, *ASIACRYPT 2011*, volume 7073 of *LNCS*, pages 468–485. Springer, Heidelberg, December 2011.

[75] Jonathan Frankle, Sunoo Park, Daniel Shaar, Shafi Goldwasser, and Daniel J. Weitzner. Practical accountability of secret processes. In William Enck and Adrienne Porter Felt, editors, *USENIX Security 2018*, pages 657–674. USENIX Association, August 2018.

[76] Georg Fuchsbauer. Subversion-zero-knowledge SNARKs. In Michel Abdalla and Ricardo Dahab, editors, *PKC 2018, Part I*, volume 10769 of *LNCS*, pages 315–347. Springer, Heidelberg, March 2018.

[77] Georg Fuchsbauer. WI is not enough: Zero-knowledge contingent (service) payments revisited. In Lorenzo Cavallaro, Johannes Kinder, XiaoFeng Wang, and Jonathan Katz, editors, *ACM CCS 2019*, pages 49–62. ACM Press, November 2019.

[78] Georg Fuchsbauer, Christian Hanser, and Daniel Slamanig. Structure-preserving signatures on equivalence classes and constant-size anonymous credentials. *Journal of Cryptology*, 32(2):498–546, April 2019.

[79] Georg Fuchsbauer, Eike Kiltz, and Julian Loss. The algebraic group model and its applications. In Hovav Shacham and Alexandra Boldyreva, editors, *CRYPTO 2018, Part II*, volume 10992 of *LNCS*, pages 33–62. Springer, Heidelberg, August 2018.

[80] Eiichiro Fujisaki. Improving practical UC-secure commitments based on the DDH assumption. In Vassilis Zikas and Roberto De Prisco, editors, *SCN 16*, volume 9841 of *LNCS*, pages 257–272. Springer, Heidelberg, August / September 2016.

[81] Rosario Gennaro, Craig Gentry, and Bryan Parno. Non-interactive verifiable computing: Outsourcing computation to untrusted workers. In Tal Rabin, editor, *CRYPTO 2010*, volume 6223 of *LNCS*, pages 465–482. Springer, Heidelberg, August 2010.

[82] Rosario Gennaro, Craig Gentry, Bryan Parno, and Mariana Raykova. Quadratic span programs and succinct NIZKs without PCPs. In Thomas Johansson and Phong Q. Nguyen, editors, *EUROCRYPT 2013*, volume 7881 of *LNCS*, pages 626–645. Springer, Heidelberg, May 2013.

[83] Rosario Gennaro and Yehuda Lindell. A framework for password-based authenticated key exchange. In Eli Biham, editor, *EUROCRYPT 2003*, volume 2656 of *LNCS*, pages 524–543. Springer, Heidelberg, May 2003. `http://eprint.iacr.org/2003/032.ps.gz`.

[84] Kristian Gjøsteen. A new security proof for damgård's ElGamal. In David Pointcheval, editor, *CT-RSA 2006*, volume 3860 of *LNCS*, pages 150–158. Springer, Heidelberg, February 2006.

[85] Oded Goldreich and Yair Oren. Definitions and properties of zero-knowledge proof systems. *Journal of Cryptology*, 7(1):1–32, December 1994.

[86] Shafi Goldwasser, Silvio Micali, and Charles Rackoff. The knowledge complexity of interactive proof-systems (extended abstract). In *17th ACM STOC*, pages 291–304. ACM Press, May 1985.

[87] Alonso González, Alejandro Hevia, and Carla Ràfols. QA-NIZK arguments in asymmetric groups: New tools and new constructions. In Tetsu

Iwata and Jung Hee Cheon, editors, *ASIACRYPT 2015, Part I*, volume 9452 of *LNCS*, pages 605–629. Springer, Heidelberg, November / December 2015.

[88] Alonso González and Carla Ràfols. New techniques for non-interactive shuffle and range arguments. In Mark Manulis, Ahmad-Reza Sadeghi, and Steve Schneider, editors, *ACNS 16*, volume 9696 of *LNCS*, pages 427–444. Springer, Heidelberg, June 2016.

[89] Alonso González and Carla Ràfols. Sublinear pairing-based arguments with updatable crs and weaker assumptions. *IACR Cryptology ePrint Archive*, 2019:326, 2019.

[90] Jens Groth. Simulation-sound NIZK proofs for a practical language and constant size group signatures. In Xuejia Lai and Kefei Chen, editors, *ASIACRYPT 2006*, volume 4284 of *LNCS*, pages 444–459. Springer, Heidelberg, December 2006.

[91] Jens Groth. Simulation-Sound NIZK Proofs for a Practical Language and Constant Size Group Signatures. In Xuejia Lai and Kefei Chen, editors, *ASIACRYPT 2006*, volume 4284 of *LNCS*, pages 444–459, Shanghai, China, December 3–7, 2006. Springer, Heidelberg.

[92] Jens Groth. Short pairing-based non-interactive zero-knowledge arguments. In Masayuki Abe, editor, *ASIACRYPT 2010*, volume 6477 of *LNCS*, pages 321–340. Springer, Heidelberg, December 2010.

[93] Jens Groth. A verifiable secret shuffle of homomorphic encryptions. *Journal of Cryptology*, 23(4):546–579, October 2010.

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

[95] Jens Groth. On the Size of Pairing-based Non-interactive Arguments. In Marc Fischlin and Jean-Sebastien Coron, editors, *EUROCRYPT 2016*, volume 9666 of *LNCS*, pages 305–326, Vienna, Austria, May 8–12, 2016. Springer, Cham.

[96] Jens Groth, Markulf Kohlweiss, Mary Maller, Sarah Meiklejohn, and Ian Miers. Updatable and universal common reference strings with applications to zk-SNARKs. In Hovav Shacham and Alexandra Boldyreva, editors, *CRYPTO 2018, Part III*, volume 10993 of *LNCS*, pages 698–728. Springer, Heidelberg, August 2018.

[97] Jens Groth, Rafail Ostrovsky, and Amit Sahai. Perfect non-interactive zero knowledge for NP. In Serge Vaudenay, editor, *EUROCRYPT 2006*, volume 4004 of *LNCS*, pages 339–358. Springer, Heidelberg, May / June 2006.

[98] Jens Groth and Amit Sahai. Efficient non-interactive proof systems for bilinear groups. In Nigel P. Smart, editor, *EUROCRYPT 2008*, volume

4965 of *LNCS*, pages 415–432. Springer, Heidelberg, April 2008.

[99] Thomas Icart. How to hash into elliptic curves. In Shai Halevi, editor, *CRYPTO 2009*, volume 5677 of *LNCS*, pages 303–316. Springer, Heidelberg, August 2009.

[100] Charanjit S. Jutla and Arnab Roy. Shorter quasi-adaptive NIZK proofs for linear subspaces. In Kazue Sako and Palash Sarkar, editors, *ASIACRYPT 2013, Part I*, volume 8269 of *LNCS*, pages 1–20. Springer, Heidelberg, December 2013.

[101] Charanjit S. Jutla and Arnab Roy. Shorter Quasi-Adaptive NIZK Proofs for Linear Subspaces. Technical Report 2013/109, IACR, February 24, 2013. Available at `http://eprint.iacr.org/2013/109`, last retrieved version from 14 Sep 2018.

[102] Charanjit S. Jutla and Arnab Roy. Switching lemma for bilinear tests and constant-size NIZK proofs for linear subspaces. In Juan A. Garay and Rosario Gennaro, editors, *CRYPTO 2014, Part II*, volume 8617 of *LNCS*, pages 295–312. Springer, Heidelberg, August 2014.

[103] Jonathan Katz and Vinod Vaikuntanathan. Smooth projective hashing and password-based authenticated key exchange from lattices. In Mitsuru Matsui, editor, *ASIACRYPT 2009*, volume 5912 of *LNCS*, pages 636–652. Springer, Heidelberg, December 2009.

[104] Jonathan Katz and Vinod Vaikuntanathan. Round-optimal password-based authenticated key exchange. In Yuval Ishai, editor, *TCC 2011*, volume 6597 of *LNCS*, pages 293–310. Springer, Heidelberg, March 2011.

[105] Joe Kilian. A note on efficient zero-knowledge proofs and arguments (extended abstract). In *24th ACM STOC*, pages 723–732. ACM Press, May 1992.

[106] Eike Kiltz and Hoeteck Wee. Quasi-adaptive NIZK for linear subspaces revisited. In Elisabeth Oswald and Marc Fischlin, editors, *EUROCRYPT 2015, Part II*, volume 9057 of *LNCS*, pages 101–128. Springer, Heidelberg, April 2015.

[107] Jihye Kim, Jiwon Lee, and Hyunok Oh. Updatable crs simulation-extractable zk-snarks with a single verification. Cryptology ePrint Archive, Report 2019/586, 2019. `https://eprint.iacr.org/2019/586`.

[108] Ahmed E. Kosba, Andrew Miller, Elaine Shi, Zikai Wen, and Charalampos 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.

[109] Ahmed E. Kosba, Zhichao Zhao, Andrew Miller, Yi Qian, T.-H. Hubert Chan, Charalampos Papamanthou, Rafael Pass, Abhi Shelat, and Elaine Shi. C∅C∅: A Framework for Building Composable Zero-Knowledge

Proofs. Technical Report 2015/1093, IACR, November 10, 2015. http://eprint.iacr.org/2015/1093, last accessed version from 9 Apr 2017.

[110] Benoît Libert, Thomas Peters, Marc Joye, and Moti Yung. Non-malleability from malleability: Simulation-sound quasi-adaptive NIZK proofs and CCA2-secure encryption from homomorphic signatures. In Phong Q. Nguyen and Elisabeth Oswald, editors, *EUROCRYPT 2014*, volume 8441 of *LNCS*, pages 514–532. Springer, Heidelberg, May 2014.

[111] Benoît Libert, Thomas Peters, Marc Joye, and Moti Yung. Compactly hiding linear spans - tightly secure constant-size simulation-sound QA-NIZK proofs and applications. In Tetsu Iwata and Jung Hee Cheon, editors, *ASIACRYPT 2015, Part I*, volume 9452 of *LNCS*, pages 681–707. Springer, Heidelberg, November / December 2015.

[112] Helger Lipmaa. On the CCA1-Security of Elgamal and Damgård's Elgamal. In Xuejia Lai, Moti Yung, and Dongdai Lin, editors, *Inscrypt 2010*, volume 6584 of *LNCS*, pages 18–35, Shanghai, China, October 20–23, 2010. Springer, Heidelberg.

[113] Helger Lipmaa. Progression-free sets and sublinear pairing-based non-interactive zero-knowledge arguments. In Ronald Cramer, editor, *TCC 2012*, volume 7194 of *LNCS*, pages 169–189. Springer, Heidelberg, March 2012.

[114] Helger Lipmaa. Prover-Efficient Commit-And-Prove Zero-Knowledge SNARKs. In David Pointcheval, Abderrahmane Nitaj, and Tajjeeddine Rachidi, editors, *AFRICACRYPT 2016*, volume 9646 of *LNCS*, pages 185–206, Fes, Morocco, April 13–15, 2016. Springer, Heidelberg.

[115] Helger Lipmaa. Key-and-argument-updatable qa-nizks. Cryptology ePrint Archive, Report 2019/333, 2019. https://eprint.iacr.org/2019/333.

[116] Helger Lipmaa. Simulation-Extractable ZK-SNARKs Revisited. Technical Report 2019/612, IACR, May 31, 2019. Available from https://eprint.iacr.org/2019/612, updated on 13 Jul 2019.

[117] Mary Maller, Sean Bowe, Markulf Kohlweiss, and Sarah Meiklejohn. Sonic: Zero-knowledge SNARKs from linear-size universal and updatable structured reference strings. In Lorenzo Cavallaro, Johannes Kinder, XiaoFeng Wang, and Jonathan Katz, editors, *ACM CCS 2019*, pages 2111–2128. ACM Press, November 2019.

[118] Ueli M. Maurer. Abstract models of computation in cryptography (invited paper). In Nigel P. Smart, editor, *10th IMA International Conference on Cryptography and Coding*, volume 3796 of *LNCS*, pages 1–12. Springer, Heidelberg, December 2005.

[119] Alfred Menezes, Palash Sarkar, and Shashank Singh. Challenges with assessing the impact of NFS advances on the security of pairing-based cryptography. In *Paradigms in Cryptology - Mycrypt 2016*, pages 83–108, 2016.

[120] Silvio Micali and Leonid Reyzin. Soundness in the public-key model. In Joe Kilian, editor, *CRYPTO 2001*, volume 2139 of *LNCS*, pages 542–565. Springer, Heidelberg, August 2001.

[121] Daniele Micciancio and Chris Peikert. Trapdoors for lattices: Simpler, tighter, faster, smaller. In David Pointcheval and Thomas Johansson, editors, *EUROCRYPT 2012*, volume 7237 of *LNCS*, pages 700–718. Springer, Heidelberg, April 2012.

[122] Paz Morillo, Carla Ràfols, and Jorge Luis Villar. The kernel matrix Diffie-Hellman assumption. In Jung Hee Cheon and Tsuyoshi Takagi, editors, *ASIACRYPT 2016, Part I*, volume 10031 of *LNCS*, pages 729–758. Springer, Heidelberg, December 2016.

[123] Alexander Mühle, Andreas Grüner, Tatiana Gayvoronskaya, and Christoph Meinel. A survey on essential components of a self-sovereign identity. *Computer Science Review*, 30:80–86, 2018.

[124] V. I. Nechaev. Complexity of a determinate algorithm for the discrete logarithm. *Mathematical Notes*, 55(2):165–172, 1994.

[125] Pascal Paillier. Public-key cryptosystems based on composite degree residuosity classes. In Jacques Stern, editor, *EUROCRYPT'99*, volume 1592 of *LNCS*, pages 223–238. Springer, Heidelberg, May 1999.

[126] Bryan Parno, Jon Howell, Craig Gentry, and Mariana Raykova. Pinocchio: Nearly practical verifiable computation. In *2013 IEEE Symposium on Security and Privacy*, pages 238–252. IEEE Computer Society Press, May 2013.

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

[128] Ronald L Rivest, Adi Shamir, and Leonard Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, 21(2):120–126, 1978.

[129] Kazue Sako and Joe Kilian. Receipt-free mix-type voting scheme - a practical solution to the implementation of a voting booth. In Louis C. Guillou and Jean-Jacques Quisquater, editors, *EUROCRYPT'95*, volume 921 of *LNCS*, pages 393–403. Springer, Heidelberg, May 1995.

[130] Eli Ben Sasson, Alessandro Chiesa, Christina Garman, Matthew Green, Ian Miers, Eran Tromer, and Madars Virza. Zerocash: Decentralized anonymous payments from bitcoin. In *2014 IEEE Symposium on Security and Privacy*, pages 459–474. IEEE, 2014.

[131] Victor Shoup. Lower bounds for discrete logarithms and related problems. In Walter Fumy, editor, *EUROCRYPT'97*, volume 1233 of *LNCS*, pages 256–266. Springer, Heidelberg, May 1997.

[132] Jacques Stern, David Pointcheval, John Malone-Lee, and Nigel P. Smart. Flaws in applying proof methodologies to signature schemes. In Moti Yung, editor, *CRYPTO 2002*, volume 2442 of *LNCS*, pages 93–110. Springer, Heidelberg, August 2002.

[133] Mehdi Tibouchi and Taechan Kim. Improved elliptic curve hashing and point representation. *Des. Codes Cryptography*, 82(1–2):161–177, 2017.

ACKNOWLEDGEMENT

I appreciate the support of my supervisor, Helger Lipmaa, during my Ph.D. studies. Many thanks to him for giving me an opportunity to join his group and for teaching me how to be a researcher. I also thank everyone else that was part of the cryptographic protocols research group in Tartu. I thank all the other numerous co-authors that I had over the years. In particular Daniel Slamanig, whom I learned a lot and gave an opportunity to visit his group (AIT Austrian Institute of Technology in Vienna) and collaborate with them. Many thanks to Janno Siim and Toomas Krips for their Estonian translation of the abstract of the thesis.

I am grateful to have excellent reviewers whose feedback has greatly improved the quality of this thesis. For this, I thank the University of Tartu internal reviewer Vitaly Skachek, and pre-reviewers Carla Ràfols Salvador and Olivier Blazy.

Throughout my studies, I have been funded by European Union's Horizon 2020 research and innovation program under grant agreement No 653497 (project PANORAMIX) and No. 780477 (project PRIViLEDGE), and by the Estonian Research Council grant (PRG49).

SUMMARY IN ESTONIAN

Mitte-interaktiivsed nullteadmusprotokollid nõrgemate usalduseeldustega

Kinnitusskeemid ja nullteadmustõestused on ühed põhilisemad krüptograafilised primitiivid, millel on hulgaliselt päriselu rakendusi. Kinnitusskeem võimaldab osapoolel arvutada salajasest sõnumist kinnitus ja hiljem see avada (avalikustada sõnum) verifitseeritaval viisil. Erilist huvi pakuvad täieliku koosluskindlusega kinnitusskeemid kuna nende turvalisus säilib isegi kui kinnitusskeem on kombineeritud suvaliste teiste protokollidega. Tõhusad koosluskindlusega kinnitusskeemid võimaldavad paremat tõhusust rakendustes nagu kahe osapoole sasiahelad, nullteadmustõestused ja turvaline ühistöötlus. Kahjuks tüüpiliselt koosluskindluse nõue tähendab, et kinnitusskeem ei ole niivõrd tõhus kui koosluskindluseta alternatiiv. Nullteadmustõestus on protokoll tõestaja ja verifitseerija vahel, mis võimaldab tõestajal veenda verifitseerijat mingi väite paikapidavuses ilma rohkema informatsiooni lekitamiseta. Eelkõige just mitteinteraktiivsed nullteadmustõestused mängivad kriitlist rolli paljudes krüptograafilistes protokollides võimaldades verifitseerida mingi arvutuse korrektsust ilma, et kasutaja privaatsus oleks rikutud. Mitteinteraktiivsed nullteadmustõestused leiavad aina enam päriselu kasutust näiteks krüptorahades ja muudes hajustalletussüsteemides. Seega on väga oluline, et sellised tõestused oleks lühikesed ja kiire verifitseeritavusega. Just selliste omadustega on SNARK tüüpi nullteadmustõestused, mis on viimasel ajal olnud tohutu huvi all nii teaduses kui ka praktikas. Lisaks sellele on viimasel ajal aktiivselt uuritud kvaasi-adaptiivseid mitteinteraktiivseid nullteadmustõestusi lineaarsete keelte jaoks. Nendel süsteemidel on samuti kompaktsed tõestused ja seega pakuvad suurt huvi päris maailmas. Mitteinteraktiivsete nullteadmustõestuste juures on üks suuremaid praktilisi nõrkusi usaldatud seadistusfaas osapoolte ühisstringi genereerimiseks. Sellel suunal on välja pakutud kaks peamist praktilist lahendust: (i) Bellare ja teised [22] pakkusid välja õõnestuskindla nullteadmustõestuse mõiste, kus nullteadmuse omadus säilib ka siis kui ühisstring tuleb pahatahlikult osapoolelt, (ii) ühisarvutuse kasutamine ühisstringi loomiseks. Teise lahenduse puhul on koosluskindlus äärmiselt oluline kuna me soovime kombineerida ühisarvutuse protokolli nullteadmusprotokolliga. Selles doktoritöös uurime eelmainitud suundi ja pakume välja konkreetsed konstruktsioonid nende realiseerimiseks. Esiteks uurime õõnestuskindlaid SNARKe ja pakume välja õõnestuskindla versiooni seni kõige tõhusamast SNARKist [94]. Seejärel me uurime õõnestuskindlaid kvaasi-adaptiivseid tõestusi ja samuti pakume välja õõnestusekindla versiooni kõige efektiivsemast kvaasi-adaptiivsest nullteadmustõestusest [106]. Teisel suunal me kõigepealt kasutame pidevaid projektiivseid räsifunktsioone ja pakume välja uue primitiive, kus eelmainitud räsifunktsioonid on avalikult verifitseerita-

vad. Nende abil me konstrueerime seni kõige tõhusama mitteinteraktiivse koosluskindla kinnitusskeemi. Lõpetuseks me töötame välja uue võtte koosluskindlate kinnitusskeemide jaoks, mis võimaldab ühisarvutuse abil luua nullteadmustõestuste ühisstringe.

PUBLICATIONS

CURRICULUM VITAE

Personal data

Name:	Behzad Abdolmaleki
Citizenship:	Iranian
Languages:	Persian, Luri, English
Contact:	abdolmaleki.behzad@yahoo.com

Education

2016–2020	Ph.D. student in Computer Science, University of Tartu, Estonia.
2012–2014	M. Sc. in Electrical Engineering, Shahed University, Tehran, Iran.
2007–2011	in Physics and Electrical Engineering, Kurdestan University, Iran.

Employment

2016–2021	Researcher in Cryptography group, University of Tartu, Estonia.
2013–2016	Research Assistant in Information Systems and Security Lab (ISSL), Sharif University of Technology, Tehran, Iran

Scientific work

Main fields of interest:

- Cryptography
- Zero-knowledge proof, Non-Interactive Zero-Knowledge Arguments (NIZK)
- Smooth Projective Hash Functions (SPHFs)
- Lattice based cryptography (lattice based NIZK, $\cdots$)

ELULOOKIRJELDUS

Isikuandmed

Nimi:	Behzad Abdolmaleki
Kodakondsus:	Iraan
Keeled:	pärsia, luri, inglise
Email:	abdolmaleki.behzad@yahoo.com

Haridus

2016–2020	doktorant informaatikas, Tartu Ülikool, Eesti.
2012–2014	magistrikraad elektrotehnikas, Shahedi Ülikool, Teheran, Iraan.
2007–2011	bakalaureusekraad füüsikas ja elektrotehnikas, Kurdestani Ülikool, Iraan.

Teenistuskäik

2016–2021	krüptograafia teadur, Tartu Ülikool, Eesti.
2013–2016	teadusassistent infosüsteemide ja turvalisuse laboris, Sharifi Tehnoloogiaülikool, Teheran, Iraan.

Teadustegevus

Peamised uurimisvaldkonnad:

- Krüptograafia
- Nullteadmustõestused
- Pidevad projektiivsed räsifunktsioonid
- Võrepõhine krüptograafia

DISSERTATIONES INFORMATICAE PREVIOUSLY PUBLISHED IN DISSERTATIONES MATHEMATICAE UNIVERSITATIS TARTUENSIS

19. **Helger Lipmaa.** Secure and efficient time-stamping systems. Tartu, 1999, 56 p.
22. **Kaili Müürisep.** Eesti keele arvutigrammatika: süntaks. Tartu, 2000, 107 lk.
23. **Varmo Vene.** Categorical programming with inductive and coinductive types. Tartu, 2000, 116 p.
24. **Olga Sokratova.** Ω-rings, their flat and projective acts with some applications. Tartu, 2000, 120 p.
27. **Tiina Puolakainen.** Eesti keele arvutigrammatika: morfoloogiline ühestamine. Tartu, 2001, 138 lk.
29. **Jan Villemson.** Size-efficient interval time stamps. Tartu, 2002, 82 p.
45. **Kristo Heero.** Path planning and learning strategies for mobile robots in dynamic partially unknown environments. Tartu 2006, 123 p.
49. **Härmel Nestra.** Iteratively defined transfinite trace semantics and program slicing with respect to them. Tartu 2006, 116 p.
53. **Marina Issakova.** Solving of linear equations, linear inequalities and systems of linear equations in interactive learning environment. Tartu 2007, 170 p.
55. **Kaarel Kaljurand.** Attempto controlled English as a Semantic Web language. Tartu 2007, 162 p.
56. **Mart Anton.** Mechanical modeling of IPMC actuators at large deformations. Tartu 2008, 123 p.
59. **Reimo Palm.** Numerical Comparison of Regularization Algorithms for Solving Ill-Posed Problems. Tartu 2010, 105 p.
61. **Jüri Reimand.** Functional analysis of gene lists, networks and regulatory systems. Tartu 2010, 153 p.
62. **Ahti Peder.** Superpositional Graphs and Finding the Description of Structure by Counting Method. Tartu 2010, 87 p.
64. **Vesal Vojdani.** Static Data Race Analysis of Heap-Manipulating C Programs. Tartu 2010, 137 p.
66. **Mark Fišel.** Optimizing Statistical Machine Translation via Input Modification. Tartu 2011, 104 p.
67. **Margus Niitsoo**. Black-box Oracle Separation Techniques with Applications in Time-stamping. Tartu 2011, 174 p.
71. **Siim Karus.** Maintainability of XML Transformations. Tartu 2011, 142 p.
72. **Margus Treumuth.** A Framework for Asynchronous Dialogue Systems: Concepts, Issues and Design Aspects. Tartu 2011, 95 p.
73. **Dmitri Lepp.** Solving simplification problems in the domain of exponents, monomials and polynomials in interactive learning environment T-algebra. Tartu 2011, 202 p.

74. **Meelis Kull.** Statistical enrichment analysis in algorithms for studying gene regulation. Tartu 2011, 151 p.
77. **Bingsheng Zhang.** Efficient cryptographic protocols for secure and private remote databases. Tartu 2011, 206 p.
78. **Reina Uba.** Merging business process models. Tartu 2011, 166 p.
79. **Uuno Puus.** Structural performance as a success factor in software development projects – Estonian experience. Tartu 2012, 106 p.
81. **Georg Singer.** Web search engines and complex information needs. Tartu 2012, 218 p.
83. **Dan Bogdanov.** Sharemind: programmable secure computations with practical applications. Tartu 2013, 191 p.
84. **Jevgeni Kabanov.** Towards a more productive Java EE ecosystem. Tartu 2013, 151 p.
87. **Margus Freudenthal.** Simpl: A toolkit for Domain-Specific Language development in enterprise information systems. Tartu, 2013, 151 p.
90. **Raivo Kolde.** Methods for re-using public gene expression data. Tartu, 2014, 121 p.
91. **Vladimir Šor.** Statistical Approach for Memory Leak Detection in Java Applications. Tartu, 2014, 155 p.
92. **Naved Ahmed.** Deriving Security Requirements from Business Process Models. Tartu, 2014, 171 p.
94. **Liina Kamm.** Privacy-preserving statistical analysis using secure multi-party computation. Tartu, 2015, 201 p.
100. **Abel Armas Cervantes.** Diagnosing Behavioral Differences between Business Process Models. Tartu, 2015, 193 p.
101. **Fredrik Milani.** On Sub-Processes, Process Variation and their Interplay: An Integrated Divide-and-Conquer Method for Modeling Business Processes with Variation. Tartu, 2015, 164 p.
102. **Huber Raul Flores Macario.** Service-Oriented and Evidence-aware Mobile Cloud Computing. Tartu, 2015, 163 p.
103. **Tauno Metsalu.** Statistical analysis of multivariate data in bioinformatics. Tartu, 2016, 197 p.
104. **Riivo Talviste.** Applying Secure Multi-party Computation in Practice. Tartu, 2016, 144 p.
108. **Siim Orasmaa.** Explorations of the Problem of Broad-coverage and General Domain Event Analysis: The Estonian Experience. Tartu, 2016, 186 p.
109. **Prastudy Mungkas Fauzi.** Efficient Non-interactive Zero-knowledge Protocols in the CRS Model. Tartu, 2017, 193 p.
110. **Pelle Jakovits.** Adapting Scientific Computing Algorithms to Distributed Computing Frameworks. Tartu, 2017, 168 p.
111. **Anna Leontjeva.** Using Generative Models to Combine Static and Sequential Features for Classification. Tartu, 2017, 167 p.
112. **Mozhgan Pourmoradnasseri.** Some Problems Related to Extensions of Polytopes. Tartu, 2017, 168 p.

113. **Jaak Randmets.** Programming Languages for Secure Multi-party Computation Application Development. Tartu, 2017, 172 p.
114. **Alisa Pankova.** Efficient Multiparty Computation Secure against Covert and Active Adversaries. Tartu, 2017, 316 p.
116. **Toomas Saarsen.** On the Structure and Use of Process Models and Their Interplay. Tartu, 2017, 123 p.
121. **Kristjan Korjus.** Analyzing EEG Data and Improving Data Partitioning for Machine Learning Algorithms. Tartu, 2017, 106 p.
122. **Eno Tõnisson.** Differences between Expected Answers and the Answers Offered by Computer Algebra Systems to School Mathematics Equations. Tartu, 2017, 195 p.

DISSERTATIONES INFORMATICAE UNIVERSITATIS TARTUENSIS

1. **Abdullah Makkeh**. Applications of Optimization in Some Complex Systems. Tartu 2018, 179 p.
2. **Riivo Kikas**. Analysis of Issue and Dependency Management in Open-Source Software Projects. Tartu 2018, 115 p.
3. **Ehsan Ebrahimi**. Post-Quantum Security in the Presence of Superposition Queries. Tartu 2018, 200 p.
4. **Ilya Verenich**. Explainable Predictive Monitoring of Temporal Measures of Business Processes. Tartu 2019, 151 p.
5. **Yauhen Yakimenka**. Failure Structures of Message-Passing Algorithms in Erasure Decoding and Compressed Sensing. Tartu 2019, 134 p.
6. **Irene Teinemaa**. Predictive and Prescriptive Monitoring of Business Process Outcomes. Tartu 2019, 196 p.
7. **Mohan Liyanage.** A Framework for Mobile Web of Things. Tartu 2019, 131 p.
8. **Toomas Krips.** Improving performance of secure real-number operations. Tartu 2019, 146 p.
9. **Vijayachitra Modhukur.** Profiling of DNA methylation patterns as biomarkers of human disease. Tartu 2019, 134 p.
10. **Elena Sügis.** Integration Methods for Heterogeneous Biological Data. Tartu 2019, 250 p.
11. **Tõnis Tasa.** Bioinformatics Approaches in Personalised Pharmacotherapy. Tartu 2019, 150 p.
12. **Sulev Reisberg.** Developing Computational Solutions for Personalized Medicine. Tartu 2019, 126 p.
13. **Huishi Yin.** Using a Kano-like Model to Facilitate Open Innovation in Requirements Engineering. Tartu 2019, 129 p.
14. **Faiz Ali Shah.** Extracting Information from App Reviews to Facilitate Software Development Activities. Tartu 2020, 149 p.
15. **Adriano Augusto**. Accurate and Efficient Discovery of Process Models from Event Logs. Tartu 2020, 194 p.
16. **Karim Baghery.** Reducing Trust and Improving Security in zk-SNARKs and Commitments. Tartu 2020, 245 p.