Compressed Σ-protocol theory

Attema, T.

CHAPTER 1

1 Introduction

1.1 Background

1.1.1 Mathematical Proofs

Proofs and arguments are natural concepts; they provide evidence attesting the correctness of a claim. They are a cornerstone in every scientific discipline, systematizing and structuring our understanding of the universe. However, the exact form of a proof may differ between disciplines. In natural sciences, such as biology, chemistry and physics, a "proof" may consist of a set of experimental observations confirming a hypothesis. In mathematics, a proof consists of logical arguments inferring a statement from agreed upon ground rules, referred to as axioms. Some of the first mathematical proofs date back to the ancient Greeks, with Euclid introducing the axiomatic approach.

Mathematical proofs are not unique; typically, true statements admit many different proofs. The quality or elegance of proofs can be considered a matter of taste, but their formality and rigor certainly has an aesthetic appeal. Hardy explicitly argued the importance of elegance, and Erdős often referred to aesthetically pleasing proofs as *"proofs from the book."*

> *"The mathematician's patterns, like the painter's or the poet's must be beautiful; the ideas, like the colours or the words must fit together in a harmonious way. Beauty is the first test: there is no permanent place in this world for ugly mathematics."*
>
> — G.H. Hardy, A Mathematician's Apology (1940)

The elegance of a proof is strongly related to its verifiability. In an elegant proof it is much harder to hide a mistake in obscurity, intentionally or unintentionally. Elegance thus simplifies the verification of a proof.

Hilbert, one of the greatest mathematicians of the 19th and 20th centuries, envisioned that every mathematical truth admits a proof. More precisely, he conjectured the existence of a consistent system of axioms in which every mathematical truth can be proven. In fact, he even hoped for the existence of an automated procedure for finding or "computing" these proofs. However, his hope was in vain. First, Gödel's *Incompleteness Theorem* [Göd31] shows that any formal system

of axioms (sufficiently powerful to define certain elementary arithmetic) admits mathematical statements that are unprovable, i.e., statements that can neither be proved nor disproved. Second, Church and Turing independently showed that there does not exist a finite procedure to decide on the validity of arbitrary mathematical statements [Chu36; Tur36]. In particular, there cannot exist a procedure that yields a proof for every provable statement.

The work of Hilbert, Church, Gödel, Turing and others formalized notions such as computability and algorithms, marking the birth of computer science. However, it soon became clear that many problems that are theoretically computable remain intractable in practice, simply because of the overwhelming resources that appear to be required to compute a solution. Therefore, computer science extended its scope and, besides mere computability, it started to study the efficiency of computations, giving rise to the subfield of computational complexity theory. One could argue that the elegance and beauty of mathematical proofs, referred to by Hardy and Erdős, not only imply verifiability, but also an informal notion of *efficient* verifiability.

1.1.2 Cryptography

Also in cryptography, when aiming to realize certain functionalities in the presence of adversarial entities, proofs play an essential role; they allow provers to convince verifiers of their truthfulness and honesty. However, in many cryptographic scenarios, proofs contain secret information that needs to remain hidden from the adversary. This poses the question whether it is possible to prove a claim without revealing any information about the proof beyond its existence. To some extend this is indeed possible, and proofs with this property are referred to as *zero-knowledge proofs*. Not only the theory of proofs, but also computational complexity theory has a strong connection to cryptography. Let us first discuss the latter connection before returning to the theory of (cryptographic) proofs.

Traditionally, cryptography dealt with protecting communication channels against unwanted eavesdroppers. For instance, Julius Caesar encrypted his messages using a secret key such that only his generals, with knowledge of this secret key, would be able to decrypt the encrypted messages. The security of the established communication channel held under the assumption that messages can only be decrypted *efficiently* when given the secret key; without this key decrypting should be infeasible. In other words, the security depends on the *computational complexity* of decrypting a message without a key. Unfortunately, Caesar's cipher turned out to be broken; there exist efficient procedures for decrypting even without knowledge of the secret key. Still many modern encryption schemes follow the same principle; they rely on the computational complexity, or hardness, of decrypting a message without the secret key.

Caesar's cipher and its downfall present one of the first events in an everlasting arms race between cryptographers and cryptanalysts. Cryptographers aiming to develop new encryption schemes, and cryptanalysts aiming to break these schemes. The field of research containing both cryptography and cryptanalysis is referred to as *cryptology*. A notable example in this arms race is the Vigenère cipher [Bel53], designed in 1553 and dubbed "*le chiffrage indéchiffrable*" (the unbreakable cipher). After more than 300 years, in 1863, also this "unbreakable" cipher was

broken [Kas63]. Another famous example is the Enigma code, used by the German military during the Second World War. Turing's successful efforts in breaking this code are believed to have shortened the Second World War.[1] More generally, his foundational contributions to the field of computing forced cryptographers to design ciphers capable of withstanding attacks aided by electronic computing. Currently, the next chapter of this arms race has commenced; protecting communication channels against the looming threat of quantum computers. It is known that, once available, powerful enough quantum computers will be able to break some of the most commonly used encryption schemes [Sho94]. For this reason, cryptographers worldwide are developing novel schemes capable of withstanding attacks from both classical and quantum computers. This relatively young field of research is referred to as *post-quantum cryptography*.

Previously the security of encryption schemes mainly relied on heuristics; as long as it was unknown how to break a cipher it could be considered secure. However, the developments of the 20th century, such as the birth of computational complexity theory, turned cryptology into an exact science. Additionally, Shannon's information theory rigorously defined what it means for an encryption scheme to be perfectly secure [Sha48a; Sha48b; Sha49]. He proved that one-time-pad encryption admits this level of security. More precisely, even adversaries with unlimited resources will not be able to break a one-time-pad encryption. But he also showed that perfect security requires secret keys that are at least as long as the underlying message, deeming perfect security impractical for many application scenarios.

The Vigenère cipher, Enigma code and one-time-pad are *symmetric* encryption schemes; the same secret key is used for both encryption and decryption. An important limitation of these schemes is that they can only be used after the secret key has been distributed amongst the sender and recipient of the communication channel. Moreover, before distributing the secret key, the communication channel remains unprotected, i.e., the channel cannot be used to distribute the secret key. In the late 1960s, while working at the Government Communication Headquarters (GCHQ) of the United Kingdom, Ellis started working on a solution for this key distribution problem. He managed to prove that, in principle, it should be possible to secure a communication channel without pre-shared secret keys, but he did not find a cryptographic primitive for this task. In 1973, Cocks joined GCHQ and learned about Ellis' efforts. He soon realized that the integer factorization problem possesses the asymmetry required to secure a communication channel without pre-shared keys; it is easy to compute the product of two primes but it is (or at least appears to be) hard to find the prime factors of a composite integer.

Cocks' solution was the first public-key (or asymmetric) encryption scheme. A public-key encryption scheme uses two keys; a public key pk for encrypting messages and a secret key sk for decrypting encrypted messages. Because the public key can only be used for encryption, it does not need to remain secret. For this reason, a public-key encryption scheme is not subject to the key distribution problem; the public key can be distributed over insecure communication channels. Note that, to prevent an adversary from impersonating honest users, an authentication

[1] Prior to the outbreak of the Second World War, the Polish mathematicians Marian Rejewski, Jerzy Rózycki and Henryk Zygalski broke earlier versions of the Enigma code, thereby laying the foundation for ultimately breaking the Enigma code.

mechanism is still required.

Also at GCHQ, Williamson learned about Cocks' breakthroughs. The somewhat counterintuitive notion of public-key encryption led him to believe that Cocks' solution must contain a flaw. Williamson did not manage to find a flaw, but in 1974, while trying to find one, he invented an alternative solution for the key distribution problem.

The results of GCHQ remained classified until the late 1990s, but nowadays Cocks, Ellis and Williamson are broadly recognized for their breakthroughs in cryptography. For instance, in 2010 the Institute of Electrical and Electronics Engineers (IEEE) awarded them the 100th IEEE Milestone Award.

Fortunately, in 1976, the revolutionizing notion of public-key encryption was independently put forward by Diffie and Hellman [DH76]. Without knowledge of the work done in secrecy at GCHQ, Diffie and Hellman reinvented Williamson's protocol, currently well known as the Diffie-Hellman (DH) key exchange protocol. In 1978, also Cocks' approach, i.e., basing public-key encryption on the hardness of factoring integers, was rediscovered by Rivest, Shamir and Adleman [RSA78]. Their protocol is now known as the RSA encryption scheme. Eventually, these solutions to the key distribution problem brought cryptography to the masses; nowadays encryption schemes are omnipresent in society.

In the 1970s, Diffie and Hellman not only invented public-key cryptography,[2] they also described how public-key encryption schemes rely on the existence of *trapdoor one-way functions*. These are functions that can be evaluated efficiently, but are hard to invert without knowledge of a secret trapdoor. In other words, the computational complexity of inverting certain functions underlies the security of public-key encryption schemes, again exemplifying the strong relation between cryptology and computational complexity theory.

Almost 50 years after their introduction, it still has not been proven that the functions underlying the Diffie-Hellman and RSA schemes indeed possess the required one-way property. Therefore, the security of these schemes relies on the *computational assumption* that inverting these functions is indeed intractable. Hence, this security notion still has a somewhat heuristic nature; security holds as long as no one finds an efficient procedure for solving the underlying computational problem. The confidence in a computationally secure scheme grows with the amount of research that has gone into solving the underlying problem. It is common practice to reduce breaking a cryptographic primitive to solving a well-studied computational problem. For instance, the security of RSA encryption scheme is related to the integer factorization problem; a problem that has been studied for at least 300 years. Hence, already at the time of its introduction, it had withstood cryptanalytic efforts. Based on this, and accounting for future developments, Rivest, Shamir and Adleman suggested the use of 200 digit (or 664 bit) public keys. However, the publication of this cryptographic primitive further incentivized the study of the integer factorization problem. Notably, in 1988, Pollard proposed a new algorithm for factoring integers. His approach, later improved and generalized, has become known as the *number field sieve* (NFS) [LL93]. The NFS is currently the most efficient (classical) approach known for factoring integers. In

[2] Only in 1997, GCHQ revealed that Ellis, Clifford and Williamson had already invented public-key cryptography, although in secrecy.

particular, it shows that 664 bit public keys do not offer a reasonable amount of security anymore. For this reason, it is recommended to use public RSA keys of at least 2048 bits. Similar progress has been made into solving the discrete logarithm problem underlying the Diffie-Hellman key exchange protocol. However, both the integer factorization and the discrete logarithm problem have remained classically intractable; there still does not exist an efficient, i.e., polynomial time, algorithm for solving these problems on classical computers. By contrast, Shor has shown how to solve both problems efficiently on a quantum computer [Sho94]. Hence, once powerful enough quantum computers become available, the security of the Diffie-Hellman and RSA schemes can no longer be guaranteed; post-quantum cryptography must be deployed well before this happens.

1.1.3 Multilateral Cryptography

Besides a solution for the key distribution problem, Diffie and Hellman also proposed a novel cryptographic functionality: *digital signature schemes*. A digital signature allows anyone to verify the authenticity of the sender, i.e., to verify its identity. It can also be used to show that a message has not been altered during transmission, i.e., guarantee its integrity. Diffie and Hellman therefore broadened the scope of cryptology beyond the confidentiality of communication channels. Today cryptology deals not only with confidentiality, but also with authenticity, integrity and non-repudiation.[3]

The broadened scope of cryptology inspired the development of many more advanced cryptographic functionalities. For instance, already in 1978 the concept of a *privacy homomorphism* was formulated [RAD78]. A privacy homomorphism, now known as a homomorphic encryption scheme, allows computations to be performed on encrypted data. This way the party performing the computations does not need to have access to the input data, but only to their encryption. While this concept has existed for decades, it took until 2009 before the first fully homomorphic encryption scheme, allowing *arbitrary* computations to be performed on encrypted data, was constructed [Gen09]. Further, Blum showed how two mutually distrustfully and physically separated parties can flip a coin without using a trusted third party [Blu81]. A protocol for playing a "mental" game of poker over the telephone was designed [SRA81; GM82]. And, more generally, it was shown how multiple parties can collaboratively evaluate arbitrary functions on their private inputs without revealing these inputs to each other, giving rise to the flourishing field of *multiparty computation* (MPC) [Yao82; Yao86; GMW87; CDG87; BGW88; CCD88].

A common denominator in these more advanced cryptographic primitives is that they aim to protect parties not only against external adversaries, but also against each other. For instance, guaranteeing that players are not cheating in a game of mental poker. This type of security is also referred to as *multilateral* security, whereas security against merely external adversaries is referred to as *unilateral* security. When aiming for multilateral security, it is desirable that parties *prove* that they behave honestly or, more generally, prove that the claims they make are

[3] Non-repudiation requires the identity of a sender to be verifiable not only by the sender but also by a third party. In this case, the sender cannot deny having sent the message.

valid.

1.1.4 Probabilistic Proof Systems

In cryptography, it is typically sufficient for provers to be able to prove the validity of certain subclasses or families of claims; they do not need to be able to prove the validity of all possible claims. For instance, the family of all integers composed of two prime factors; each integer in this family corresponds to the claim that it is indeed the product of two primes, and the prime factors constitute a proof for such a claim. By multiplying these factors the proof can be verified efficiently. This example describes a *proof system* for the family of all integers composed of two prime factors. More formally, a proof system for a family of valid claims L is defined by an efficiently computable and deterministic verification function V_L that, on input a claim x and a purported proof w, outputs either accept or reject. A family L and a claim x are also referred to as a *language* and a *statement*, respectively. Thus, in this formalization, a prover claims that a statement x is in the language L, i.e., $x \in L$. A proof w such that $V_L(x; w) = \mathsf{accept}$ is also called a *witness* for statement x. This formalization is due to Cook and Reckhow [CR79]. They required a proof system to be *complete* and *sound*. A proof system is complete if every valid statement $x \in L$ admits a witness w. It is sound if for every invalid statement $x \notin L$ and every w it holds that $V_L(x; w) = \mathsf{reject}$.

The class of languages that admit a proof system as above is denoted by NP. Moreover, P denotes the class of languages for which claims can be efficiently verified without knowledge of a witness, i.e., even without a witness one can efficiently verify that $x \in L \subseteq \mathsf{P}$. For this reason, proof systems for languages that are not (known to be) in P are typically more interesting. However, since verifying a proof may require less resources than computing a proof from scratch, also proof systems for languages in P can be of interest. Clearly $\mathsf{P} \subseteq \mathsf{NP}$, however it is unknown whether $\mathsf{P} = \mathsf{NP}$, i.e., whether every problem that admits an efficiently verifiable solution can also be solved efficiently. The P versus NP problem [Coo71; Lev73] of computational complexity theory is one of the biggest open problems in mathematics and computer science.

In 1985, two seminal works independently generalized the notion of a proof system, by allowing randomness, interaction and errors [Bab85; GMR85]. In this generalization, called an *interactive* or *probabilistic* proof, two parties, a prover and a verifier, interact before the verifier decides whether to accept the prover's claim. In other words, the verifier is allowed to ask the prover a number of questions before making its decision. The verifier still has to be efficient, but is no longer required to be deterministic. In fact, since the prover can predict the questions asked by a deterministic verifier, any interactive proof with a deterministic verifier can be made non-interactive, i.e., by predicting the verifier's questions the prover can output all its answers without interacting with the verifier. Therefore, interaction can only give something new for *probabilistic* verifiers. Further, the verifier of an interactive proof is allowed to make errors, i.e., it might reject valid claims (completeness error) or accept false claims (soundness error). In many occasions, by deploying certain amplification techniques, the error probabilities of interactive proofs can be made arbitrarily small. Interestingly, these relaxations have opened a whole new world of possibilities.

First, interactive proofs can be constructed for certain languages that are not known to be in NP. For instance, while it is unknown whether there exists an efficiently verifiable proof attesting that two graphs are not isomorphic, there do exist interactive proofs for the graph non-isomorphism problem [GS86; GMW86].

Second, and perhaps more surprisingly, many interactive proofs can be made *zero-knowledge*. A zero-knowledge proof is an interactive proof in which the verifier learns nothing beyond the correctness of the prover's claim. For instance, it allows a prover to convince a verifier that an integer is the product of two primes without revealing the prime factors. The notion zero-knowledge was introduced by Goldwasser, Micali and Rackoff [GMR85]. They further gave the first zero-knowledge proof system. Zero-knowledge proofs have proven to be extremely powerful cryptographic primitives. They can for instance be used to prove knowledge of a secret password without revealing the password, or to prove that votes have been tallied honestly without revealing the individual votes. The existence of zero-knowledge proofs is related to the (conjectured) existence of one-way functions. More precisely, one-way functions exist if and only if all languages in NP admit a zero-knowledge proof system [GMW91; OW93].

Third, every claim in NP admits a proof that can be verified by checking only a small part of the proof, i.e., when given a statement x and its purported witness $w \in \{0,1\}^*$, represented as a bitstring, the verifier only needs to choose, at random, a small number of w's bits to verify. A proof or witness that can be verified in this manner is called a *Probabilistically Checkable Proof* (PCP) [AS92]. One of the most influential theorems in computational complexity theory, the PCP theorem, states that every statement in NP admits a PCP [ALM+98; Din07]. Unfortunately, even with a PCP, it is impossible to construct an interactive proof system for arbitrary NP-languages with *succinct* communication [GH98], i.e., an interactive proof with communication costs that grow only sublinearly in the size of the statement x. By contrast, interactive *arguments* do not suffer from this restriction. Interactive arguments relax the soundness property of interactive proofs; instead of requiring soundness against computationally unbounded provers, interactive arguments are only required to be sound against *computationally bounded* provers. By using a certain class of one-way functions, Kilian showed how to compile any PCP into an interactive argument with succinct communication [Kil92].

1.1.5 Proofs and Arguments of Knowledge

Interactive proofs and arguments only consider provers claiming that a public statement x is in a language L. If L is an NP-language, $x \in L$ implies that the statement x admits an efficiently verifiable witness w. An interactive proof for such a language merely allows a prover to convince a verifier of the *existence* of a witness, it does not necessarily allow proving *knowledge* of such a witness. In some cases the existence of a witness is trivially satisfied and therefore a void statement, whereas knowledge of a witness is a completely different story. For instance, consider a prover claiming that an integer has a prime factorization; clearly every integer has a prime factorization, but finding or knowing such a factorization can be highly nontrivial. This example demonstrates the need for a stronger functionality, allowing a prover to prove knowledge of a witness. While early interactive proofs seemed to satisfy this requirement intuitively, it took several years before

satisfactory definitions of *knowledge soundness* and *proofs of knowledge* (PoKs), as strengthenings of ordinary soundness and interactive proofs, were derived [GMR85; TW87; FFS88; BG92].

Informally, a prover is said to know a witness w, if there exists an efficient algorithm, also referred as the extractor, capable of extracting w from the prover. To this end, the extractor may invoke the prover and reply with arbitrary messages, playing the role of the verifier. Further, the extractor is allowed to rewind the prover to previous states. Hence, a dishonest prover knows a witness w if, by running the extractor, it can efficiently compute w.

As before, an *argument of knowledge* (AoK) is a relaxation of a PoK, in which knowledge soundness only holds against computationally bounded provers.

1.1.6 Σ-Protocol Theory

In the late 1980s and the early 1990s, various zero-knowledge proof systems were introduced [FS86; FFS88; GQ88; Sch91; Oka92]. Due to its efficiency, especially Schnorr's protocol [Sch91] is still broadly used today, e.g., as the main building block for many digital signature schemes. He proposed an elegant and practical interactive proof for proving knowledge of a discrete logarithm without revealing any information about the discrete logarithm itself. In his solution, the prover first sends a message to the verifier, who replies with a challenge sampled uniformly at random from some finite set, and after receiving the prover's response, the verifier decides whether to accept or reject the prover's claim. Nowadays interactive proofs that follow the same 3-round structure and design principle as Schnorr's protocol are referred to as Σ-*protocols* [Cra96].

Over the past decades, Σ-protocol theory has developed into a well-established and versatile theory for secure algorithmics. Loosely speaking, with secure algorithmics we refer to the design of cryptographic realizations of standard algorithmic tasks. In other words, this entails porting algorithms for standard tasks to cryptographic scenarios. For instance, in MPC where mutually distrustfully parties wish to collaboratively evaluate an algorithm without revealing their input values, or in zero-knowledge where a prover aims to convince a verifier that an algorithm has been evaluated honestly, again without revealing the input.

More generally, Schnorr's interactive proof is for proving knowledge of a homomorphism preimage [Cra96; CD98], i.e., it reveals a *linear* relation between a prover's secret witness w and a public statement x. The theory of Σ-protocols has been extended towards realizing a much broader class of (not necessarily linear) functionalities. For instance, there exist Σ-protocols for proving *partial* knowledge of a subset of discrete logarithms [CDS94]. Further, it is known how to prove the satisfiability of an arithmetic circuit by using Σ-protocols [CD98], i.e., for proving the existence of an input for which the arithmetic circuit evaluates to 0. The arithmetic circuit satisfiability problem is NP-complete, i.e., every problem in the complexity class NP can be written as a circuit satisfiability problem, demonstrating the power of Σ-protocols. Moreover, Σ-protocols have been instantiated based on various cryptographic hardness assumptions beyond the discrete logarithm assumption, e.g., based on lattice assumptions, plausibly providing post-quantum security [MV03].

The versatility of Σ-protocol theory comes largely due to its *modularity*; ad-

vanced cryptographic primitives are composed of smaller abstract building blocks. These abstract building blocks are easy to analyze and can be instantiated from a wide variety of cryptographic hardness assumptions. By generic composition results, the (security) properties of cryptographic protocols are easily derived from the properties of their abstract building blocks.

1.1.7 Recent Efficiency Improvements in Proof Systems

The introduction of interactive proofs ignited a rich field of research. Notably, Wigderson, who played an influential role in the development of computational complexity theory and (interactive) proof theory, was awarded the 2021 Abel prize (along with Lovász) for his contributions to theoretical computer science and discrete mathematics. For instance, together with Goldreich and Micali, Wigderson showed that the validity of any NP-statement can be proven in zero-knowledge, assuming the existence of one-way functions [GMW86]. For an elaborate history of this field of research, we refer to his book [Wig19].

Additionally, the growing adoption of cloud and decentralized computing platforms has caused an increased interest in efficient (zero-knowledge) proof systems. Namely, in many scenarios, outsourcing computations to (untrusted) computing platforms requires verification. Verifiable computation deals with the integrity of computations outsourced to untrusted parties, i.e., it guarantees that computations have been executed correctly. The naive method for establishing *computational integrity* consists in redoing the computation and verifying its output. However, this approach has two major disadvantages. First, it is *inefficient*, i.e., it often completely beats the purpose of outsourcing computations to a party with more computational resources. Second, verifying a computation in this manner requires (private) input values to be revealed. Kilian's interactive proof for arbitrary NP-statements [Kil92] already demonstrated that zero-knowledge proof systems might offer a solution. His solution, although impractical due to a significant computational overhead, has succinct communication and is zero-knowledge. Alternatively, Σ-protocols offer concretely efficient zero-knowledge proofs for many languages [CD98]. However, their communication complexity scales linearly with the size of the statement, and the verification part of a Σ-protocol typically requires more computational resources than the computation that is to be verified. Hence, Σ-protocols only offer a partial solution for the computational integrity problem.

Recently, Bulletproofs [BCC+16; BBB+18] have been introduced as a "drop-in replacement" for Σ-Protocols in several important applications. Notably, this includes proving the satisfiability of an arithmetic circuit; protocols for this task are also referred to as *circuit zero-knowledge* protocols. The communication complexity of standard Σ-protocols is linear in the size of the circuit, whereas Bulletproofs reduce the communication complexity down to logarithmic. At the heart of Bulletproofs is an interactive proof of knowledge between a prover and verifier showing that a Pedersen commitment to a vector of large length n satisfies a multivariate polynomial equation of degree 2, defined with an inner product. This pivotal protocol stands out in that, by means of a split-and-fold technique, it ingeniously *compresses* the communication costs down to $\mathcal{O}(\log n)$ elements from $\mathcal{O}(n)$ via traditional Σ-protocols. Although this is at the expense of introducing a logarith-

mic number of communication rounds between the prover and verifier (instead of constant), its public-coin[4] nature ensures that it can be rendered non-interactive using the Fiat-Shamir heuristic [FS86]. However, applications following this novel paradigm meet a number of technical difficulties. First, this inner-product protocol is not zero-knowledge, and second, cryptographic protocol theory has to be reinvented with the quadratic constraint proved as its pivot. This leads to a deviation from the natural and well-established linearization strategy adopted by Σ-protocol theory.

Besides Bulletproofs, many novel interactive proof and, more generally, argument systems have recently been proposed. These systems offered practical computational integrity, even for lengthy and complicated computations. The current wealth of argument systems is partially due to the large number of distinctive features they possess. There does not exist a single argument system that outperforms its competitors on all terrains; the optimal solution depends largely on the application scenario. There are different performance metrics quantifying the efficiency of arguments, e.g., the computational complexities of the prover and the verifier, and the communication complexity or proof size. Moreover, most arguments require some set of public parameters known to all parties involved. Preferably this set of parameters, referred to as the *common reference string* (CRS), is as small as possible. Additionally, some argument systems enable efficiency improvements at the cost of requiring a *trusted* setup, i.e., a setup phase that is guaranteed to be executed honestly. When considering mutually distrustful parties, a trusted setup is challenging to realize. Argument systems that do not require a trusted setup are called *transparent*. Further, for their zero-knowledge and (knowledge) soundness properties, proofs and arguments may rely on different cryptographic assumptions. Some assumptions are more conservative and are even assumed to hold against quantum adversaries, e.g., the existence of one-way functions. While other assumptions, such as the knowledge of exponent (KEA) assumption, are unfalsifiable and could be considered more controversial.

1.2 Contributions

In this dissertation, we enhance Σ-protocol theory with a compression mechanism, allowing the communication complexity to be reduced from linear down to (poly)logarithmic. More precisely, we show how to combine compact commitments, arithmetic secret-sharing and an adaptation of Bulletproofs' split-and-fold technique to develop a versatile theory for the modular design of communication-efficient zero-knowledge proof systems: *Compressed Σ-Protocol Theory*. Further, we provide a number of applications and show that our approach is supported by various cryptographic platforms, including one plausibly offering post-quantum security.

A key design principle in our theory is *linearization*; we solve the linear problem instances first and then show how to linearize nonlinear ones. More precisely, our basic compressed Σ-protocols prove knowledge of homomorphism preimages,

[4] An interactive proof is *public-coin* if all of the verifiers random choices are made public, i.e., they are sent to the prover.

i.e., they prove a linear relation between a public element and its secret preimage. By a novel variation of an arithmetic secret-sharing based technique for Σ-protocols [CDP12], we then show how to linearize nonlinear problem instances, i.e., where the relation between the public statement and the secret witness is not captured by a linear mapping. Mathematically, solving the linear instances first and then linearizing the nonlinear ones is perhaps among the most natural problem solving strategies.

Additionally, we identify and close three gaps in the general theory of multi-round interactive proofs. First, we provide the first tight knowledge soundness analysis for the class of *special-sound* multi-round interactive proofs, containing Bulletproofs and compressed Σ-protocols. Second, we prove that the t-fold parallel repetition of special-sound multi-round interactive proofs optimally reduces the success probability of dishonest provers, or more precisely the knowledge error, from κ down to κ^t. Third, for special-sound interactive proofs, we show that the security loss of the Fiat-Shamir heuristic, rendering (public-coin) interactive proofs non-interactive, is independent of the number of rounds.

Below these contributions are described in more detail.

1.2.1 Compressed Σ-Protocols

We start, in Chapter 3, by combining two essential components. First, as an abstract building block, or *pivot*, we consider a basic Σ-protocol for proving knowledge of the preimage of a group homomorphism $\Psi\colon \mathbb{G}^n \to \mathbb{H}$, where $n \in \mathbb{N}$. Hence, our pivot is a Σ-protocol for proving knowledge of an n-dimensional vector. The zero-knowledge property states that evaluating the Σ-protocol does not reveal any information about the preimage. The communication complexity of this pivot grows linearly in the input dimension n. More precisely, the final message of the Σ-protocol, sent from the prover to the verifier, is a vector of dimension n. Second, this Σ-protocol is *compressed* by replacing the final (long) prover-message with an appropriate adaptation of Bulletproofs' inner-product argument; instead of sending its final message to the verifier, the prover shows it knows it. For many homomorphisms of interest, namely if the size of the codomain $\mathbb{H}$ is constant or logarithmic in n, this *compression mechanism* has a communication complexity that is logarithmic or polylogarithmic in the dimension n. Note that the compression mechanism does not need to be zero-knowledge; it replaces a message that the prover would have revealed otherwise. As a result, the required soundness and zero-knowledge properties of the Σ-protocol are preserved, but the overall communication drops from linear down to (poly)logarithmic.

1.2.1.1 Opening Linear Forms on Compact Commitments

Compressed Σ-protocols can be instantiated for a broad class of homomorphisms. A notable example is given by homomorphisms of the form

$$\psi(\mathbf{x};\gamma) = \big(\textsc{com}(\mathbf{x};\gamma), L(\mathbf{x})\big)\,,$$

where $\mathbf{x} \in \mathbb{Z}_q^n$ is the prover's secret input vector, COM is a (homomorphic) commitment scheme, γ is the commitment randomness and $L\colon \mathbb{Z}_q^n \to \mathbb{Z}_q$ is a linear form. If the commitment scheme is *compact*, i.e., the size of a commitment is constant in

the dimension n of the committed vector $\mathbf{x}$, the compression mechanism reduces the communication complexity from linear down to (poly)logarithmic in n. This instantiation allows a prover to prove knowledge of a commitment opening $(\mathbf{x}; \gamma)$ that satisfies a linear constraint captured by the linear form L. Evaluating this (compressed) Σ-protocol reveals nothing beyond the value $y = L(\mathbf{x})$, and is therefore also referred to as *opening* linear form L on committed vector $\mathbf{x}$.

1.2.1.2 Functionality Enhancements

Many techniques known from Σ-protocol theory directly apply to compressed Σ-protocols. For instance, standard amortization techniques allow many linear forms $L_1, \ldots, L_s$ to be opened, instead of just one, without increasing the overall communication complexity. Similarly, a prover can open a single linear form on many different committed vectors for the price of one. Further, using this and by plug-and-play with our basic theory, we show how to handle the application scenario where the linear form takes as secret input a long vector that is initially dispersed across several commitments. We handle this scenario by *compactifying* these dispersed components into a single commitment first. This is useful in important applications, such as commit-and-prove zero-knowledge proofs for arithmetic circuit satisfiability, where the prover has committed to the input vector *before* the arithmetic circuit is provided. More precisely, in many relevant practical scenarios, we must assume that the commitment to the prover's secret input vector, about which something is to be proved in zero-knowledge, has already been produced before the zero-knowledge protocol is run. In these scenarios commit-and-prove functionality is required. Moreover, to prepare for Strong-RSA and lattice instantiations, we further extend the compressed Σ-protocols to provers additionally claiming that the preimage is *short*.

1.2.1.3 Higher Level Functionalities

In Chapter 4, the significance of opening linear forms surfaces. First, we integrate this basic functionality with a novel variation on arithmetic secret-sharing based techniques for Σ-Protocols [CDP12], inspired by MPC. These techniques allow for linearization of nonlinear relations. More precisely, we show how to prove the correctness of large sets of committed multiplication triples $(\alpha_i, \beta_i, \gamma_i := \alpha_i \beta_i)$. It will turn out that, combined with an appropriate adaptation of [CDP12], we only need black-box access to our basic functionality of opening linear forms. The (poly)logarithmic communication complexity of the compressed Σ-protocols is directly inherited by our protocol for proving the correctness of multiplication triples.

Second, we consider another scenario that cannot be handled directly with a basic compressed Σ-protocol. Namely, a prover claiming to know k-out-of-n homomorphism preimages. More precisely, for a fixed homomorphism ψ, the prover claims to know k preimages out of n public elements $P_1, \ldots, P_n$ in the codomain of ψ. As before, the prover wishes to convince a verifier of the veracity of this claim without revealing any additional information. In particular, it should remain a secret for which k elements the prover knows the preimages. *Proofs of*

partial knowledge were introduced in [CDS94]. In [CDS94], a k-out-of-n proof of partial knowledge Σ-protocol with *linear* (in n) communication complexity was presented. Unfortunately, their Σ-protocol cannot be compressed. For this reason, we construct a novel Σ-protocol for proving k-out-of-n partial knowledge. More precisely, we deploy a linear secret-sharing scheme to reduce the k-out-of-n scenario to the n-out-of-n scenario. For the n-out-of-n scenario, standard amortization techniques, together with our compression mechanism, apply. Altogether, this results in a k-out-of-n proof of partial knowledge with *logarithmic* (in k and n) communication complexity. Again we only need black-box access to basic compressed Σ-protocols.

These functionality enhancements explain why our basic compressed Σ-protocols do not need any *direct* provision to handle nonlinearity. In both cases, it is the combination of proving knowledge of homomorphism preimages and (arithmetic) secret-sharing that allows for linearizing nonlinear relations.

1.2.1.4 Suitable Cryptographic Platforms

In Chapter 5, we show that compressed Σ-protocols can be instantiated in a variety of cryptographic platforms. First, we consider a discrete logarithm based instantiation that starts from the Pedersen vector commitment scheme. This instantiation allows a prover to open linear forms on committed vectors with a logarithmic communication complexity. Further, we show that this instantiation can be extended to pairing based platforms. In addition, compressed Σ-protocols can be based on a Knowledge-of-Exponent Assumption (KEA), further reducing the communication complexity down to *constant* instead of logarithmic. Note that the KEA is unfalsifiable and its application is not completely without controversy [Nao03; BCP+14]. Moreover, this approach introduces a trusted set-up, which might be undesirable. Finally, we show how to base compressed Σ-protocols on the Strong-RSA and certain lattice assumptions. However, these instantiations are subject to a so called *soundness slack*. An interactive proof is said to have soundness slack if a prover can only convince the verifier of the correctness of a related, but somewhat relaxed, claim. More precisely, in these instantiations the prover claims to know not an arbitrary but a *short* ψ-preimage $\mathbf{x}$ of an element P, i.e., $\psi(\mathbf{x}) = P$ and $\|\mathbf{x}\| \leq \beta$ for some homomorphism ψ and some $\beta \in \mathbb{R}_{\geq 0}$. While such a witness $\mathbf{x}$ is required to convince the verifier, i.e., for completeness, knowledge soundness only guarantees the verifier that the prover knows an input $\tilde{\mathbf{x}}$ such that $\psi(\tilde{\mathbf{x}}) = \zeta \cdot P$ and $\|\tilde{\mathbf{x}}\| \leq \tau \cdot \beta$. The element ζ is referred to as the approximation factor and τ is referred to as the soundness slack. The source of the soundness slack is twofold. First, during the execution of the compressed Σ-protocol, while its dimension decreases, the norm of the preimage increases. Second, the protocol is proven to be knowledge sound by constructing an efficient algorithm capable of extracting a witness from any prover that convinces the verifier with large enough probability. The extraction algorithm contributes to the soundness slack and additionally introduces an approximation factor. In many application scenarios this relaxation is acceptable. However, selection of larger implementation parameters is warranted, causing the communication complexity to be *poly*-logarithmic instead of logarithmic or constant.

1.2.2 Knowledge Extractor Analysis

In Chapter 6, we continue with the security analysis or, more precisely, the knowledge soundness analysis of compressed Σ-protocols. The goal of a compressed Σ-protocol is for a prover to convince a verifier that it knows some secret witness; a prover without knowledge of a witness should not be able to convince the verifier. This security property is formalized by the notion of knowledge soundness. Informally, knowledge soundness states that any prover, that succeeds in convincing the verifier with large enough probability, should be able to efficiently compute a witness satisfying the claimed properties. For this reason, to prove that an interactive proof or argument is knowledge sound, an efficient algorithm capable of extracting a witness from a prover must be constructed. The extractor may invoke the prover arbitrarily many times and also *rewind* the prover to previous states. In this process, the extractor plays the role of the verifier and provides the challenges to the prover. As such the extractor obtains different protocol transcripts, which it uses to compute a witness. The success probability and runtime of the extractor may, and typically do, depend on the success probability of the prover.

It is generally nontrivial to show that an interactive proof admits an extractor and, thus, is knowledge sound. By contrast, the weaker ordinary soundness notion does not require the existence of an extractor. More precisely, soundness only states that the existence of a prover with large enough success probability implies the existence of a witness; it does not require the witness to be efficiently computable. For this reason, it is typically much easier to prove ordinary soundness than knowledge soundness.

In the context of Σ-protocols, the more convenient notion *special-soundness* was introduced [Cra96]. A Σ-protocol is said to be k-special-sound if there exists an efficient algorithm that, on input k accepting protocol transcripts $(a, c_1, z_1), \ldots, (a, c_k, z_k)$ with common first message a and pairwise distinct challenges c_i, outputs a witness. Recall that a Σ-protocol transcript (a, c, z) contains three messages; the first message a is sent from the prover to the verifier, the verifier sends a challenge c sampled uniformly at random from some finite challenge set, and the prover sends the final response z. Subsequently, the verifier decides whether to accept or reject the transcript and thus the prover's claim. We also refer to k-special-soundness as k-out-of-N special-soundness, where N is the size of the verifier's challenge set.

In a k-out-of-N special-sound Σ-protocol, no matter what a dishonest prover does for the first message, if the statement does not admit a witness, there are at most $k-1$ challenges that the dishonest prover can possibly answer. Hence, since the challenges are sampled uniformly at random, a dishonest prover succeeds, on invalid statements without a witness, with probability at most $(k-1)/N$. This already shows that k-out-of-N special-soundness implies ordinary soundness with soundness error $(k-1)/N$. However, in the case of knowledge soundness, this line of reasoning does not apply, since in principle it is possible to answer all the challenges – and indeed the prover can do so if he knows a witness. The challenge is to show that the prover necessarily needs to know a witness to be able answer many challenges; formally, to show the existence of a knowledge extractor.

Although nontrivial to show, it is well known k-out-of-N special-sound

Σ-protocols admit a knowledge extractor. More precisely, k-out-of-N special-soundness implies knowledge soundness with knowledge error $(k-1)/N$, where the knowledge error is the optimal success probability of a dishonest prover. To prove knowledge soundness, it is thus sufficient to show that a Σ-protocol is special-sound, which is usually much easier than proving knowledge soundness directly. Namely, the special-soundness algorithm is given a set of accepting transcripts, whereas the knowledge extractor is only given access to a prover attacking the interactive proof.

Recently, and particularly for the aforementioned compression techniques, natural *multi-round* generalizations of special-soundness have become relevant. For instance, in Chapter 3, we show that compressed Σ-protocols satisfy a multi-round special-soundness notion. In fact, many recently introduced multi-round interactive proofs are special-sound, e.g., [BCC+16; BBB+18; MBK+19; BFS20; BLN+20]. However, known proof techniques, proving that special-soundness implies knowledge soundness, are no longer directly applicable. Namely, the nature of the compression mechanism significantly reduces the efficiency of the corresponding knowledge extractors. More precisely, the efficiencies of naive generalizations of known knowledge extractors scale *exponentially* in the number of rounds of the interactive proof. Several works have attempted to close this gap in the theory of multi-round interactive proofs [BCC+16; HKR19; PLS19; JT20; AL21]. However, their extractors either only provide an asymptotical analysis, requiring for instance exponentially large challenge sets, or their concrete security bounds are non-tight.

1.2.2.1 Special-Sound Multi-Round Interactive Proofs

We provide the first *tight* knowledge soundness analysis for multi-round special-sound interactive proofs and arguments. First, we construct a knowledge extractor that runs in *strict* polynomial time. Unfortunately, this extractor is only applicable to a portion of the full parameter space relevant to our applications. More precisely, it only applies to interactive proofs with a constant number of rounds, whereas Bulletproofs and compressed Σ-protocols have a logarithmic number of rounds. For this reason, we construct a second extractor for special-sound multi-round interactive proofs. In contrast to our first extractor, it runs in *expected* polynomial time. However, it is applicable to the full parameter space and therefore provides a complete solution to the aforementioned knowledge soundness problem. Along the way, we significantly simplify the knowledge soundness analysis of 3-round special-sound interactive proofs.

1.2.2.2 Parallel Repetition

In many occasions, the knowledge error κ, or the success probability of a dishonest prover, is not small enough, and thus needs to be reduced. This can be done generically by repeating the interactive proof in parallel. Naively, one expects that if a prover can cheat in a single instance with probability at most ϵ, then he can cheat at most with probability ϵ^t in a t-fold repetition. However, it is not immediately clear how to prove this – and in general it is actually not true [BIN97; PW07]. The issue is that the prover may potentially make the t runs dependent, and, for example, achieve that with probability ϵ he wins all of them (and thus he wins the parallel repetition) and with probability $1-\epsilon$ he loses all of them. This

situation does not contradict the security of a single run, because in each individual run he only wins with probability ϵ, and so it's not clear how to conclude security of the parallel repetition from the security of a single run (only).

In the case of k-out-of-N special-sound Σ-protocols, the t-fold parallel repetition is easily seen to be ℓ-out-of-N^t special-sound, with $\ell = (k-1)^t + 1$. This immediately implies that the soundness error is $(k-1)^t/N^t$, i.e., the t-fold parallel repetition reduces the soundness error from $\sigma = (k-1)/N$ down to σ^t. However, as before, this line of reasoning does not extend to the stronger notion of knowledge soundness. Namely, the expected runtime of the knowledge extractor for k-out-of-N special-sound interactive proofs is linear in k. Therefore, applying this knowledge extractor to the t-fold parallel repetition results in a runtime that is linear in $\ell = (k-1)^t + 1$, i.e., for $k > 2$ it is exponential in t which is too large. Therefore, to show that t-fold parallel repetition reduces the knowledge error from κ down to κ^t, one cannot merely rely on the special-soundness property. The situation becomes even more complicated when considering multi-round interactive proofs.

Parallel repetition is a fundamental technique in the theory of probabilistic proofs, and its effect on the ordinary soundness error has been studied extensively in many contexts [BIN97; PV07; Hai09; HPW+10; CL10; PV12; CP15]. However, somewhat surprisingly, the effect of parallel repetition on the knowledge error has largely remained unstudied. In this dissertation, we show that t-fold parallel repetition reduces the knowledge error of special-sound multi-round interactive proofs at an optimal rate; from κ down to κ^t. At the core of our results is an alternative, in some sense more fine-grained, measure of quality of a dishonest prover than its success probability, for which we show that it characterizes when knowledge extraction is possible. This new measure then turns out to be very convenient when it comes to analyzing the parallel repetition of such interactive proofs.

Additionally, we provide a novel knowledge extractor that is not only applicable to special-sound interactive proofs, but to the larger class of public-coin interactive proofs. This generality comes at a cost; for public-coin interactive proofs, we show that t-fold parallel repetition reduces the knowledge error from κ down to $\kappa^t + \nu$, for any arbitrary non-negligible ν.

1.2.2.3 The Fiat-Shamir Transformation

Public-coin interactive proofs are typically made non-interactive before being deployed in practice. This can be done by applying the widely used Fiat-Shamir transformation [FS86]. The general idea is to compute the verifier's i-th challenge c_i as a hash of the i-th prover message a_i and (some part of) the previous communication transcript. Recall that, since the interactive proof is public-coin, the i-th challenge c_i is sampled uniformly at random from some finite set. The security of the Fiat-Shamir transformation is usually proven in the idealized *random oracle model* (ROM), where it is assumed that the hash function behaves as a random function. More precisely, in this model the only way to compute the evaluation $H(x)$, of hash function H on input x, is by querying a "random oracle" that has sampled the function table of H uniformly at random. The security of the Fiat-Shamir transformation thus relies on the assumption that the hash

function H behaves as a random oracle in the context of the considered scheme. There exist contrived counterexamples of protocols that are secure in the ROM, but insecure when the random oracle is instantiated with any concrete hash function [CGH04]. However, this transformation is broadly used and, in practice, it appears to withstand all known attacks.

Unfortunately, the Fiat-Shamir transformation introduces a security loss. Namely, in the interactive setting, a dishonest prover must succeed on the challenges it receives from the verifier. By contrast, in the non-interactive setting, a dishonest honest prover may invoke the hash function several times and try multiple sets of challenges when forging a proof. Clearly, the security loss depends on the number of queries Q the prover is allowed to make to the hash function, which is thus modeled as a random oracle.

This also makes the security or extractor analysis of non-interactive Fiat-Shamir transformations significantly more complicated than the analysis of interactive proofs. In the interactive setting, the extractor determines which challenges to provide to the prover. In the non-interactive setting, the extractor does not know for which challenges the prover will output a proof.

The Fiat-Shamir transformation of Σ-protocols has been well-studied. In particular, it is known that the Fiat-Shamir transformation preserves the relevant security properties of a Σ-protocol (in the ROM), with a security loss that is linear in the prover's query complexity Q. However, in general, the security loss of the Fiat-Shamir transformation is *exponential* in the number of rounds of the interactive proof. In fact, it is easy to find interactive proofs that are indeed subject to this exponential security loss.

For multi-round interactive proofs, such as Bulletproofs and compressed Σ-protocols, this is a very unfortunate situation when it comes to choosing concrete security parameters. If one wants to rely on the proven security reduction, one needs to choose a large security parameter for the interactive proof, in order to compensate for exponential security loss, affecting its efficiency; alternatively, one has to give up on proven security and simply assume that the security loss is much milder than what the general bound suggests – indeed, for many interactive proofs, the known attacks do not feature such a large security loss. The latter, of simply assuming the loss to be milder, has become common practice.

This raises the question whether certain (natural) classes of interactive proofs feature a milder security loss. Ideally, the exponential loss appears for contrived examples only. So far, the only positive results in that direction are [CCH+19; GT21]. They show that, in some restricted settings and for certain specific interactive proofs, the Fiat-Shamir security loss is independent of the number of rounds. These results require additional cryptographic assumptions and only apply to a subclass of compressed Σ-protocols.

In this work, we resolve the state-of-affairs by giving both positive and negative answers to the above question. On the positive side, we show that for special-sound interactive proofs the security loss is *independent* of the number of rounds. One can now rely on proven security without choosing overly conservative, and hence inefficient, protocol parameters. On the negative side, we show that for t-fold parallel repetitions of typical special-sound interactive proofs the security loss is *exponential* in the number of rounds. This shows that the exponential security loss

is not only exhibited by contrived interactive proofs.

The extractor analyses of Chapter 6 immediately generalizes from interactive proofs to interactive arguments.

1.2.3 Applications

Finally, in Chapter 7, we discuss two applications of compressed Σ-protocol. First, we consider the circuit satisfiability problem. An interactive proof for circuit satisfiability allows a prover to prove, for any arithmetic circuit $C\colon \mathbb{Z}_q^n \to \mathbb{Z}_q^s$, that it knows a satisfiable input $\mathbf{x} \in \mathbb{Z}_q^n$, i.e., an input $\mathbf{x}$ such that $C(\mathbf{x}) = 0$. With a specialized reduction, we reduce proving the satisfiability of an arithmetic circuit to proving the correctness of a list of multiplication triples. For the latter task, the linearization strategy of Section 4.2 suffices.

Recall that the circuit satisfiability problem is NP-complete, i.e., every problem in NP can be written as a circuit satisfiability problem. However, oftentimes a significant overhead can be avoided by solving a specific problem *directly*, i.e., without reducing it to the standard circuit satisfiability scenario. For instance, by plug-and-play with our basic theory, we construct a *commit-and-prove* zero-knowledge protocol for circuit satisfiability directly. In a commit-and-prove protocol, the prover has already committed to the input vector $\mathbf{x} \in \mathbb{Z}_q^n$ before the start of the protocol, and claims that the committed vector satisfies the constraint $C(\mathbf{x}) = 0$ for some arithmetic circuit C. The naive solution reduces the commit-and-prove scenario to the standard circuit satisfiability scenario. This solution requires the commitment function to be described by a (typically large) arithmetic circuit, and therefore introduces an overhead. We avoid this reduction, and the corresponding overhead, and handle the commit-and-prove scenario directly via plug-and-play with compressed Σ-protocol theory.

Second, we construct a novel transparent and succinct threshold signature scheme (TSS). A k-out-of-n TSS allows any subset of at least k players to sign a message. Our TSS is transparent, because it does not require a trusted setup, and it is succinct, because the size of a threshold signature grows only logarithmically in the total number of players n. A TSS can be constructed immediately, by translating the TSS problem to a circuit satisfiability problem and applying a circuit zero-knowledge protocol. However, we again follow a direct approach and combine a carefully chosen signature scheme with the proofs of partial knowledge of Section 4.3. In contrast to the naive circuit zero-knowledge approach, this direct solution avoids a significant concrete overhead.

These applications demonstrate the advantage of a modular theory for secure algorithmics. There is a set of standard and abstract scenarios that can be handled with basic (compressed) Σ-protocols directly. The basic theory is appended with certain functionality enhancements, increasing its versatility. Application scenarios are handled via a plug-and-play with the abstract building blocks, and by appropriate instantiations thereof. Further, the (security) properties of compound protocols, handling (complex) application scenarios, follow directly from the properties of the basic building blocks. This approach resembles the design principle of Σ-protocol theory, which has now been strengthened with a compression mechanism. We believe this perspective to be useful for handling many more application scenarios in an intuitive manner.

1.3 Publications

This dissertation provides a conceptual framework for the modular design of communication-efficient zero-knowledge proofs. The results from the publications listed below are presented within this framework. There is no one-to-one correspondence between the chapters and the publications. All chapters are based on results that have appeared in different publications and, vice versa, most publications contribute to multiple chapters. In each chapter, it is indicated which publications it is based on. Moreover, this dissertation provides additional details and proofs that have not been published before.

[AFK22] Thomas Attema, Serge Fehr, and Michael Klooß. "Fiat-Shamir Transformation of Multi-Round Interactive Proofs." In: *Theory of Cryptography Conference (TCC)*. Vol. 13747. Lecture Notes in Computer Science. Springer, 2022, pp. 113–142.

[AF22] Thomas Attema and Serge Fehr. "Parallel Repetition of $(k_1, \ldots, k_\mu)$-Special-Sound Multi-Round Interactive Proofs." In: *CRYPTO*. Vol. 13507. Lecture Notes in Computer Science. Springer, 2022, pp. 415–443.

[ACR21] Thomas Attema, Ronald Cramer, and Matthieu Rambaud. "Compressed Σ-Protocols for Bilinear Group Arithmetic Circuits and Application to Logarithmic Transparent Threshold Signatures." In: *ASIACRYPT*. Vol. 13093. Lecture Notes in Computer Science. Springer, 2021, pp. 526–556.

[ACF21] Thomas Attema, Ronald Cramer, and Serge Fehr. "Compressing Proofs of k-out-of-n Partial Knowledge." In: *CRYPTO*. Vol. 12828. Lecture Notes in Computer Science. Springer, 2021, pp. 65–91.

[ACK21] Thomas Attema, Ronald Cramer, and Lisa Kohl. "A Compressed Σ-Protocol Theory for Lattices." In: *CRYPTO*. Vol. 12826. Lecture Notes in Computer Science. Springer, 2021, pp. 549–579.

[AC20] Thomas Attema and Ronald Cramer. "Compressed Σ-Protocol Theory and Practical Application to Plug & Play Secure Algorithmics." In: *CRYPTO*. Vol. 12172. Lecture Notes in Computer Science. Springer, 2020, pp. 513–543.

Further, the author has co-authored the following publications in the field of zero-knowledge proof systems. The results of these publications, although strongly related, are not covered in this dissertation.

[ACC+22] Thomas Attema, Ignacio Cascudo, Ronald Cramer, Ivan Damgård, and Daniel Escudero. "Vector Commitments over Rings and Compressed Σ-Protocols." In: *Theory of Cryptography Conference (TCC)*. Vol. 13747. Lecture Notes in Computer Science. Springer, 2022, pp. 173–202.

[ACX21] Thomas Attema, Ronald Cramer, and Chaoping Xing. "A Note on Short Invertible Ring Elements and Applications to Cyclotomic and Trinomials Number Fields." In: *Mathematical Cryptology* (2021), pp. 45–70.

[ALS20] Thomas Attema, Vadim Lyubashevsky, and Gregor Seiler. "Practical Product Proofs for Lattice Commitments." In: *CRYPTO*. Vol. 12171. Lecture Notes in Computer Science. Springer, 2020, pp. 470–499.

Finally, the author has co-authored the following publications, which are not directly related to compressed Σ-protocol theory. Therefore, also the results of these publications are not covered in this dissertation.

[ACK23] Thomas Attema, Pedro Capitão, and Lisa Kohl. "On Homomorphic Secret Sharing from Polynomial-Modulus LWE." In: *Practice and Theory of Public-Key Cryptography (PKC)*. 2023.

[SMA+22] Gabriele Spini, Emiliano Mancini, Thomas Attema, Mark Abspoel, Jan de Gier, Serge Fehr, Thijs Veugen, Maran van Heesch, Daniël Worm, Andrea De Luca, Ronald Cramer, and Peter M. A. Sloot. "A New Approach to Privacy-Preserving Clinical Decision Support Systems for HIV Treatment." In: *Journal of Medical Systems (JMS)* 46.84 (2022), pp. 1–11.

[ADE+22] Thomas Attema, Vincent Dunning, Maarten H. Everts, and Peter Langenkamp. "Efficient Compiler to Covert Security with Public Verifiability for Honest Majority MPC." In: *Applied Cryptography and Network Security (ACNS)*. Vol. 13269. Lecture Notes in Computer Science. Springer, 2022, pp. 663–683.

[ABN21] Thomas Attema, Joost W. Bosman, and Niels M. P. Neumann. "Optimizing the Decoy-State BB84 QKD Protocol Parameters." In: *Quantum Information Processing (QIP)* 20.154 (2021), pp. 1–26.

[HMA+20] Roy van Houte, Jesse Mulderij, Thomas Attema, Irina Chiscop, and Frank Phillipson. "Mathematical Formulation of Quantum circuit Design Problems in Networks of Quantum Computers." In: *Quantum Information Processing (QIP)* 19.5 (2020), pp. 1–22.

[NHA20] Niels M. P. Neumann, Roy van Houte, and Thomas Attema. "Imperfect Distributed Quantum Phase Estimation." In: *International Conference on Computational Science (ICCS)*. Vol. 12142. Lecture Notes in Computer Science. Springer, 2020, pp. 605–615.

[SHA+19] Alex Sangers, Maran van Heesch, Thomas Attema, Thijs Veugen, Mark Wiggerman, Jan Veldsink, Oscar Bloemen, and Daniël Worm. "Secure Multiparty PageRank Algorithm for Collaborative Fraud Detection." In: *Financial Cryptography and Data Security (FC)*. Vol. 11598. Lecture Notes in Computer Science. Springer, 2019, pp. 605–623.

[AGM+21] Thomas Attema, Nicole Gervasoni, Michiel Marcus, and Gabriele Spini. "Post-Quantum Cryptography: Computational-Hardness Assumptions and Beyond." In: *IACR Cryptology ePrint Archive* (2021). IACR ePrint: 2021/571.

[VAS19] Thijs Veugen, Thomas Attema, and Gabriele Spini. "An Implementation of the Paillier Cryptosystem with Threshold Decryption without a Trusted Dealer." In: *IACR Cryptology ePrint Archive* (2019). IACR ePrint: 2019/1136.

[HAA+19] Maran van Heesch, Niels L. M. van Adrichem, Thomas Attema, and Thijs Veugen. "Towards Quantum-Safe VPNs and Internet." In: *IACR Cryptology ePrint Archive* (2019). IACR ePrint: 2019/1277.