DS-06-2017: Cybersecurity PPP: Cryptography

PRIViLEDGE
Privacy-Enhancing Cryptography in Distributed Ledgers

D2.1 – State of the Art on Privacy-Enhancing Cryptography for Ledgers

Due date of deliverable: 30th June 2018
Actual submission date: 30th June 2018

Grant agreement number: 780477
Start date of project: 1 January 2018
Revision 1.0

Lead contractor: Guardtime AS
Duration: 36 months

Project funded by the European Commission within the EU Framework Programme for Research and Innovation HORIZON 2020	
Dissemination Level	
PU = Public, fully open	X
CO = Confidential, restricted under conditions set out in the Grant Agreement	
CI = Classified, information as referred to in Commission Decision 2001/844/EC	

D2.1

State of the Art on Privacy-Enhancing Cryptography for Ledgers

Editor
Toomas Krips (UT)

Contributors
Michele Ciampi(UEDIN)
Peter Gaži (IOHK)
Aggelos Kiayias (UEDIN)
Helger Lipmaa (UT)
Berry Schoenmakers(TUE)
Luisa Siniscalchi (UNISA)
Björn Tackmann(IBM)
Ahto Truu (GT)
Ivan Visconti(UNISA)
Michał Zajac (UT)
Reviewers
Berry Schoenmakers(TUE)
Ahto Truu (GT)
Niels de Vreede (TUE)

30th June 2018
Revision 1.0

The work described in this document has been conducted within the project PRIViLEDGE, started in January 2018. This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No. 780477.

The opinions expressed and arguments employed herein do not necessarily reflect the official views of the European Commission.

©Copyright by the PRIViLEDGE Consortium

Contents

Chapter 1

Introduction

Distributed ledgers or blockchains are a technology for storing data in a distributed manner, that is, without a central storage while keeping a consensus about the contents of the data. Data that has been entered to a blockchain is very hard to remove.

Distributed ledgers have seen great prominence in the recent years. This technology has many applications, such as cryptocurrencies, or assuring the transparency of documents.

The aim of the project PRIViLEDGE is to develop and advance techniques that increase privacy, anonymity and efficient decentralized consensus for distributed ledgers technologies (DLT). Thus it is important to collect knowledge about the state of the art of different privacy-enhancing cryptographic primitives and protocols that relate or may relate to the blockchain. This document aims to collect knowledge that might be applicable or relevant to this topic. It does not describe cryptographic ledgers or protocols built on ledgers, as such, however, for this is not the aim of this deliverable. They will be described in more detail in the deliverable D3.1.

The document is structured in the following way.

Chapter 2 describes the different cryptographic models. These are a necessary component of understanding cryptographic protocols. This is necessary as then later it is better to describe and compare different instantiations of the protocols. It does, however, not describe models of ledgers, such as assumptions about players being rational or that no party is assumed to control more than some amount of computational power. Those will be described in the deliverable D3.1.

Chapter 3 gives an overview of primitives related to authentication. Many of the benefits of the blockchain come from the power of being able to preserve data. However, keeping the integrity of the data is not very useful if the data itself is incorrect due to either error or malice. Thus authentication of the data is a necessary part of DLTs. This chapter gives an overview of different flavours of signatures, many of which have been used or been proposed for use in various blockchains. It also describes various other authentication-related primitives, such as verifiable random functions, anonymous credential chemes, hash chains, and others. A description of well-known elementary primitives is included for completeness.

Chapter 4 gives an overview on privacy-enhancing protocols which are of central importance in this project. This chapter begins with some elementary confidentiality-related primitives for completeness. Then it proceeds to encryption, which is perhaps the most straight-forward of all privacy-enhancing techniques. It views several different flavours of encryption — different encryption flavours allow different sets of parties to encrypt and decrypt the data. Distributed ledgers deal with many parties simultaneously which tends to lead to complicated types of interactions with each other. Thus it would be beneficial to provide different possibilities to account for the fact that the desires of the parties might be rather complicated. We also overview private information retrieval, which allows a client to obtain information from a database without the owner of the database learning what information was queried for. As distributed ledgers and blockchains are used often for storing data, cryptographic protocols related to databases is a natural area of interest.

Chapter 5 describes secure computation and verifiable computation settings. These are powerful tools that allow processing of data in secret and authenticated ways, respectively. On one hand, these can be important tools for building other protocols. On the other hand, if data is stored on the ledgers, then it is rather natural to want to compute on it, while still preserving the privacy of the data and the correctness of the computation.

Chapter 6 gives an overview of zero-knowledge techniques. Zero knowledge is an important tool for achieving many cryptographic goals – the ability to prove to another party that everything has been done correctly without revealing any private information is extremely useful for making protocols secure in the malicious setting — i.e. in the setting where it is not assumed that the parties behave only as the protocol dictates. Also, blockchain can help with improving zero-knowledge protocols as it can be used as an instantiation of a trusted third party.

This document ends with Chapter 7 where a number of open problems are presented. These can be seen as directions of possible research.

Chapter 2

Preliminaries

Cryptographic primitives are based on models — in order to do cryptography one needs make some assumptions — what is possible to compute efficiently, how some infrastructure is set up, who is trusted, what can we assume about hardware, and so on. It is useful to outright specify what is assumed about the system we use. Thus this chapter gives an overview about various cryptographic models.

2.1 Information-Theoretic and Computational Security

We remind these two very basic notions for the sake of completeness.

Information-theoretic security is understood to be the kind of security that even an adversary with unlimited computing power cannot break. Essentially, the information is, information-theoretically speaking, not there.

This is contrasted with computational security. Computational security protocols come with some complexity parameter κ that describes the computational bounds of the adversary — it is said that an adversary whose computational powers are bound by κ can not break the scheme, while not making statements about computationally more powerful adversaries.

2.2 Public Key Models

As shown in [GK90] achieving non-interactive zero-knowledge (NIZK, see 6.1) proofs in the standard model is impossible and some trust assumptions are a must. One may distinguish two main approaches in order to achieve NIZKs. One assumes existence of some basic public key infrastructure and can be further subdivided into models presented in this section (ordered from the strongest to the weakest): common reference string model (there exists a public key provided by a trusted third party), registered public key model (each party has her own key registration authority that makes sure that a public key is *well-formed*) and bare public key model (where each party provides her own public key and no one checks its *well-formedness*). The other approach assumes existence of Random Oracle and has been described in details in 2.3.

2.2.1 Common Reference String Model

The Common Reference String (CRS) model has been introduced by Blum, Feldman and Micali in [BFM88]. The model assumes that there exists a Trusted Third Party (TTP) that provides a common reference string for all players participating in the protocol. In particular, for NIZK proof systems, the CRS is known by both prover and verifier. A weaker version of NIZK proof system is NIZK argument system, the difference between the two has been explained in Sec. 6.1.

In the case of a NIZK argument system, each string crs comes with a pair of secret trapdoors (CRS trapdoor tc and simulation trapdoor ts, where $\mathsf{ts} \subseteq \mathsf{tc}$), such that tc is sampled from a well-defined distribution $\mathcal{D}$, and for some function f, we have $\mathsf{crs} \leftarrow f(\mathsf{tc})$. The simulation trapdoor ts allows the simulator to generate proofs

Functionality $\mathcal{F}_{\mathsf{crs}}^{\mathcal{D},f}$

$\mathcal{F}_{\mathsf{crs}}^{\mathcal{D}}$ is parametrized by distribution $\mathcal{D}$. It proceeds as follows, running with a set of parties and an adversary:

1. Choose a value $\mathsf{tc} \leftarrow_{\$} \mathcal{D}$;
2. Compute $\mathsf{crs} \leftarrow f(\mathsf{tc})$;
3. When receiving $(\texttt{retrieve}, \mathsf{sid})$ from some party send $(\texttt{CRS}, \mathsf{sid}, \mathsf{crs})$ to that party.

Figure 2.1: Functionality $\mathcal{F}_{\mathsf{crs}}^{\mathcal{D}}$ [BCNP04]

Functionality $\mathcal{F}_{\mathsf{rpk}}$

$\mathcal{F}_{\mathsf{rpk}}$ proceeds as follows, given function f and security parameter λ, and running with a set of parties and an adversary $\mathcal{A}$. At the first activation a set R of strings is initialized to be empty.
Registration: When receiving a message $(\texttt{register}, \mathsf{sid})$ from a party $\mathcal{P}_i$ (either corrupted or uncorrupted) send $(\texttt{register}, \mathsf{sid}, \mathcal{P}_i)$ to $\mathcal{A}$ and receive p' from $\mathcal{A}$. If $p' \in R$ then let $p \leftarrow p'$. Otherwise, $r \leftarrow_{\$} \{0,1\}^{\lambda}$, let $p \leftarrow f(r)$, and add p to R. Record $(\mathcal{P}_i, p)$ and return (sid, p) to $\mathcal{P}_i$ and to $\mathcal{A}$.
Registration by a corrupted party: When receiving a message $(\texttt{register}, \mathsf{sid}, r)$ from a corrupted party $\mathcal{P}_i$, record $(\mathcal{P}_i, f(r))$. In this case, $f(r)$ is not added to R.
Retrieval: When receiving a message $(\texttt{retrieve}, \mathsf{sid}, \mathcal{P}_i)$ from party $\mathcal{P}_j$, send $(\texttt{retrieve}, \mathsf{sid}, \mathcal{P}_i, \mathcal{P}_j)$ to $\mathcal{A}$, and obtain a value p from $\mathcal{A}$. If $(\mathcal{P}_i, p)$ is recorded then return $(\mathsf{sid}, \mathcal{P}_i, p)$ to $\mathcal{P}_j$. Else, return $(\mathsf{sid}, \mathcal{P}_i, \perp)$ to $\mathcal{P}_j$.

Figure 2.2: Functionality $\mathcal{F}_{\mathsf{rpk}}$ [BCNP04].

for instances which it does not know a witness for (i.e. it assures that the simulator can fulfil the task given to it in the definition of zero knowledge, see 6.1). Here, it is assumed that the TTP does not reveal any part of tc to anyone and only provides ts to the simulator. An ideal functionality realizing the model is presented in Fig. 2.1.

2.2.2 Registered Public Key Model

Registered Public Key (RPK) model is a model of establishing Public Key Infrastructure (PKI) proposed by Barak et al. in [BCNP04]. The model assumes that each participant $\mathcal{P}$ has their own trusted Key Registration Authority (KRA) $\mathcal{R}_{\mathcal{P}}$. Key Registration Authority stores and publishes keys following its functionality described on Fig. 2.2.

In the first phase (key registration phase), $\mathcal{P}$ sends her public key to $\mathcal{R}_{\mathcal{P}}$ and then proves to $\mathcal{R}_{\mathcal{P}}$ the knowledge of the secret key. (That is why we will later refer to this model as *RPK using traditional proofs of knowledge* model, RPKPoK.) It is assumed that if $\mathcal{P}$ is honest, then the secret key exists and the public key comes from correct distribution (it is *"safe"*). If $\mathcal{P}$ is dishonest, the secret key still exists and, since $\mathcal{P}$ has to provide a proof of knowledge of it, it is known to her. However, there is no guarantee about distribution (such key is called *"well-formed"*). See Fig. 2.2 for the original description of the functionality of the key registration from [BCNP04] and Fig. 2.3 for illustration of this model. Note, that if an honest party $\mathcal{P}$ wants to register her key, the key may be picked as well by the KRA, as depicted on Fig. 2.2 and 2.3.

Verifier-Only RPK We distinguish a special version of the RPK model, called the verifier-only RPK model. In this model, contrary to the standard RPK model, only a verifier has her trusted key registration authority that provides her a key.

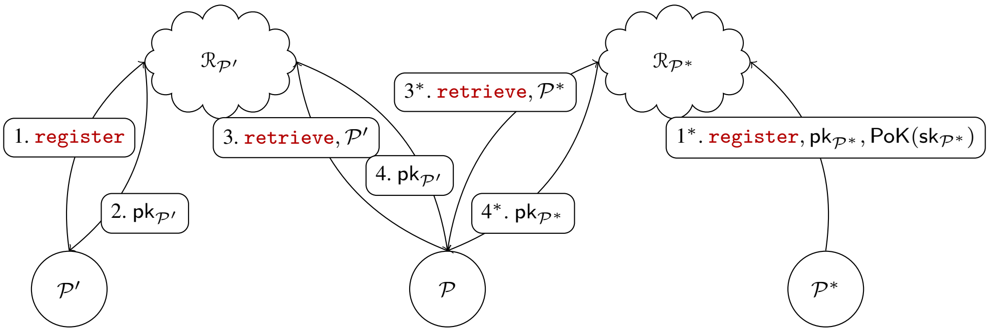
$\mathcal{R}_{\mathcal{P}'}$
$\mathcal{R}_{\mathcal{P}^*}$
3*. retrieve, $\mathcal{P}^*$
1. register
3. retrieve, $\mathcal{P}'$
4. $\mathsf{pk}_{\mathcal{P}'}$
1*. register, $\mathsf{pk}_{\mathcal{P}^*}$, $\mathsf{PoK}(\mathsf{sk}_{\mathcal{P}^*})$
2. $\mathsf{pk}_{\mathcal{P}'}$
4*. $\mathsf{pk}_{\mathcal{P}^*}$
$\mathcal{P}'$
$\mathcal{P}$
$\mathcal{P}^*$

Figure 2.3: RPKPoK with three parties – an honest $\mathcal{P}'$ with her KRA $\mathcal{R}_{\mathcal{P}'}$, a dishonest $\mathcal{P}^*$ with the KRA $\mathcal{R}_{\mathcal{P}^*}$ and a party $\mathcal{P}$ who wants to retrieve public keys of $\mathcal{P}'$ and $\mathcal{P}^*$. Party $\mathcal{P}'$ registers her key by sending register message to her authority $\mathcal{R}_{\mathcal{P}'}$. Dishonest $\mathcal{P}^*$ picks her public key, secret key pair $(\mathsf{pk}_{\mathcal{P}^*}, \mathsf{sk}_{\mathcal{P}^*})$, sends register to $\mathcal{R}_{\mathcal{P}^*}$ along with the picked $\mathsf{pk}_{\mathcal{P}^*}$ and $\mathsf{PoK}(\mathsf{sk}_{\mathcal{P}^*})$ – proof of knowledge of $\mathsf{sk}_{\mathcal{P}^*}$. Party $\mathcal{P}$ retrieves public keys of $\mathcal{P}', \mathcal{P}^*$ by querying $\mathcal{R}_{\mathcal{P}'}, \mathcal{R}_{\mathcal{P}^*}$ respectively.

2.2.3 Bare Public-Key Model

The Bare Public-Key (BPK) model was proposed by Canetti et al. in [CGGM00] as a very mild setup assumption that allows for round-efficient constructions of advanced notions of zero-knowledge proofs.

The model as described in [CGGM00] and later on formalized by [Rey01] assumes that the verifiers post their identities represented by public keys to a public repository. This is a form of preprocessing phase to be performed before any proof system starts.

The reason why the assumption is mild is that the public repository can be completely controlled by the adversary that can decide to discard identities and to add her own identities, even adaptively on the ones of the honest verifiers. The fact that the assumption is mild has been formally proven in [KL11] where it is shown that the impossibility of universally composable secure computation that holds in the plain model, holds also in the BPK model.

The BPK model has a natural extension to any cryptographic protocol by requiring that players post their identities before being engaged in protocol executions.

There is a connection between the BPK model and the execution of cryptographic protocols in presence of a ledger. Indeed in the BPK model honest players are guaranteed that after protocols start, the public repository will not change. There is therefore a clear similarity with the read-only property of past data added to the ledger.

2.3 Random Oracle Model

In the standard computational model all parties involved in a cryptographic protocol are assumed to be interactive Turing machines. In many cases, including consensus protocol design, it has been proven useful to describe properties in the Random Oracle model (ROM) [BR93]. In this model, the parties involved in the execution of a protocol have access to a shared functionality that takes an input x and returns a random value ρ. Moreover, whenever queried on the same input x, the function returns always the same value ρ. The ROM allows to construct highly efficient cryptographic schemes. Though, even if a protocol can be proved secure in the ROM, it is not implied that the security holds when the random oracle is replaced by a concrete, publicly computable hash function [Nie02, GK03, CGH04].

The Random Oracle model can be captured as an ideal functionality, cf. Figure 2.4.

The ROM is widely used to analyze the security of general protocols following the simulation-based security definitions. The ROM provides also some composable security guarantees. The work of Canetti et. al. [CJS14] shows how to obtain composability for different classes of cryptographic protocols in the Global Random Oracle model (GROM). In this model, all the protocol instances share access to the same RO. The recent work

Functionality $\mathcal{F}_{\mathrm{RO}}$

The functionality interacts with an adversary $\mathcal{S}$ and a set $\mathcal{P} = \{P_1, \dots, P_n\}$ of parties.

- Upon receiving $(\mathsf{Eval}, \mathsf{sid}, x)$ from P_i (resp. $\mathcal{S}$), return ρ to P_i (resp. $\mathcal{S}$) if $(x, \rho) \in T$. If no entry for x is in T, then choose $\rho \leftarrow \{0,1\}^\kappa$, add (x, ρ) in T and return ρ to P_i.

Figure 2.4: *The random oracle ideal functionality.*

of [CDG+18] proposes a revisited version of the GROM and extends the number of essential cryptographic primitives that can be proved secure when arbitrarily composed in this special GROM model.

The RO has been intensively used to describe the security of transaction ledger [GKL15, KZZ16, DGKR18]. For example, in many blockchains the miners, in order to generate a new block, need to compute a certain amount of hash values. In order to analyse the security of the underlying blockchain it is convenient to model the hash as a random oracle.

2.4 Universally Composable Security

The following description is from [CLOS02, Lin11].

Universal composability is a definition of security that considers a stand-alone execution of a protocol in a special setting involving an environment machine $\mathcal{Z}$, in addition to the honest parties and adversary. As with the classic definition of secure computation, ideal and real models are considered where a trusted party carries out the computation in the ideal model and the real protocol is run in the real model. The environment adaptively chooses the inputs for the honest parties, interacts with the adversary throughout the computation, and receives the honest parties' outputs. Security is formulated by requiring the existence of an ideal-model simulator Sim so that no environment $\mathcal{Z}$ can distinguish between the case that it runs with the real adversary $\mathcal{A}$ in the real model and the case that it runs with the ideal-model simulator Sim in the ideal model.

In slightly more detail, we denote by $\mathsf{IDEAL}_{\mathcal{F},\mathsf{Sim}^{\mathcal{A}},\mathcal{Z}}(\lambda, \mathsf{x})$ the output of the environment $\mathcal{Z}$ with input x after an ideal execution with the ideal adversary (simulator) Sim and functionality $\mathcal{F}$, with security parameter λ. We will only consider black-box simulators Sim, and so we denote the simulator by $\mathsf{Sim}^{\mathcal{A}}$ meaning that it works with the adversary $\mathcal{A}$ attacking the real protocol. We denote by $\mathsf{EXEC}_{\pi,\mathcal{A},\mathcal{Z}}(\lambda, \mathsf{x})$ the output of environment $\mathcal{Z}$ with input x after a real execution of the protocol π with adversary $\mathcal{A}$, with security parameter λ.

In addition, according to the definition in [CLOS02], all messages between the parties and between the parties and the ideal functionality are delivered by the adversary. We consider a model with ideally authenticated channels, and so the adversary is allowed to read the messages sent but cannot modify them. In contrast to messages sent between the parties which can be read by the adversary, messages sent between the parties and the ideal functionality are comprised of a public header and private content. The public header contains information that is not secret (like the message type, session identifier, the sending and receiving party), whereas the private content contains information that the adversary is not allowed to learn like the parties' private inputs. See [CLOS02] for more details.

A protocol π *UC-securely computes* $\mathcal{F}$ if there exists a probabilistic polynomial time (PPT) Sim such that for every non-uniform PPT (NUPPT) $\mathcal{Z}$ and every PPT $\mathcal{A}$, the following holds:

$$\{\mathsf{IDEAL}_{\mathcal{F},\mathsf{Sim}^{\mathcal{A}},\mathcal{Z}}(\lambda, \mathsf{x})\}_{\lambda\in\mathbb{N},\mathsf{x}\in\{0,1\}^*} \approx_c \{\mathsf{EXEC}_{\pi,\mathcal{A},\mathcal{Z}}(\lambda, \mathsf{x})\}_{\lambda\in\mathbb{N},\mathsf{x}\in\{0,1\}^*}.$$

The importance of this definition is a composition theorem that states that any protocol that is universally composable is secure when run concurrently with many other arbitrary protocols (see [Can01, CLOS02] for discussions and definitions). Indeed, one main challenge in formulating the security of cryptographic protocols is capturing the threats coming from the execution environment, and in particular potential "bad interactions" with other protocols that are running in the same system or network. One way to capture the security concerns that arise in a specific protocol environment or in a given application is to directly represent the given environment or application within an extended definition of security. Such an approach is taken, for instance in the cases of non-malleable commitments, concurrent zero knowledge and general concurrently secure protocols, where the definitions explicitly model several adversarially coordinated instances of the protocol in question. This approach, however, results in definitions with ever-growing complexity, and is inherently limited in scope since it addresses only specific environments and concerns [Can01].

Composable Treatment of Ledger Distributed ledgers can be used as a tool to design cryptographic protocols. So it is natural to define the security and the functionalities that, for example, Bitcoin can provide. More precisely, it is convenient to define models of computation and, in these models, an abstraction of Bitcoin as a distributed protocol, and proved that the output of this protocol satisfies certain security properties, for example the common prefix [GKL15] or consistency [PSS17] property. The reason to model a ledger functionality in the UC model relies on the fact that it is not known in what context the ledger will be used, and what kind of cryptographic protocols will be concurrently executed (see [GKL15, KZZ16] for more details about the UC treatment of ledger).

2.5 Leakage Resiliency and Tamper-Proofness

PRIViLEDGE will use the ledger to secure data providing confidentiality and integrity. This will necessary require some key management mechanisms. A major issue in key management is the possibility that some important information (i.e., a secret) could be leaked therefore threatening the desired security. Two main directions have been investigated to obtain a robust key management: the use of tamper-proof hardware, and the use of leakage-resilient cryptography.

2.5.1 Tamper-Proof Hardware Tokens

The use of smart cards in the last decades has been a concrete real-world example of the advantages of using tamper-proof hardware for running cryptographic protocols. Starting from smart cards, secure identification has been the standard application for tamper-proof hardware. Indeed RSA SecurID tokens are used all over the world and are nowadays part of multi-factor authentication systems.

Katz showed in [Kat07] that tamper-proof hardware can be used to actually perform any universally composable (UC) secure computation. The feasibility result of Katz has then been exploited in multiple directions in particular focusing on reducing the requirements on tamper proofness (e.g., by focusing on stateless tokens), the number of tokens, the reuse of tokens in multiple sessions and minimizing the involved complexity-theoretic assumptions. In addition to the obvious attempts to securely realize any functionality by means of tokens, part of the recent research has focused on specific primitives that are relevant in real-world applications. In [MS08] Moran and Segev showed how to use tamper-proof tokens for designing advanced notions of commitment schemes. In [Kol10] Kolesnikov showed how to obtain an efficient protocol for oblivious transfer (OT) with stateless tamper-proof tokens. However this construction is proven secure under a relaxed security notion (i.e., covert) where the adversary cheats only if she is guaranteed not to be caught. Improvements were presented by [DKNS04] requiring only stateless tokens for any unbounded number of OTs and UC security. In [FPS$^+$11] it was shown that tamper-proof tokens allow for very efficient constructions for secure set intersection.

The reason why tamper-proof hardware tokens can be useful for ledgers is two-fold. First of all, tamper-proof hardware helps for key management and more in general for storing and using private information. Second, tamper-proof hardware allows for the design of non-interactive protocols. Non-interactiveness can be crucial in several applications that make use of ledgers. Indeed when using interaction different messages are published in different blocks, and the delays to confirm blocks in (at least some) ledgers would affect negatively the overall performance.

2.5.2 Primitives with leakage resilience

A recent research direction in cryptography consists of designing cryptographic primitives that remain secure even in case of leakage of some information from the state of a machine storing a secret. Since the seminal work of Dziembowski and Pietrzak [DP08], security against leakage attacks has been considered in the design of all cryptographic primitives studying both feasibility and infeasibility results.

Cryptocurrencies typically make use of digital signatures posted in a ledger in order to show the transfer of coins among wallets. As such, leakage-resilient signature schemes are a relevant privacy-enhancing cryptographic primitive for ledgers. In general a privacy-preserving ledger includes encrypted data and therefore there are secret keys stored out of the ledger that have a critical role for using the ledger. However since some sort of leakage is possible, the use of leakage-resilient encryption can be useful.

We finally mention a recent attempt to relax the security in the presence of leakage leading to the notion of leakage tolerance [BCH12]. Such a limited security notion is sometimes required because of impossibility results in presence of leakage [OPV15].

2.6 Simulation

The security of a cryptographic primitive can be defined via the notion of *simulation*. Let us consider an encryption scheme Π. A natural way to define the security of such a scheme would be to state that Π is secure if the adversary that receives a ciphertext learns nothing about the encrypted message. However, this definition does not consider that an adversary could know something about the encrypted message in advance, before receiving the ciphertext. The simulation paradigm captures this specific aspect. Indeed, to define the security of an encryption scheme we impose that whatever an adversary $\mathcal{A}_1$ can give as output given an encryption c of a message m, it can be given as output also by an adversary $\mathcal{A}_2$ that does not take c as input. Since $\mathcal{A}_2$ receives nothing, he can learn nothing about the message m. We refer to this scenario as the *ideal world*. The scenario where the $\mathcal{A}_1$ gets the encryption of m is instead called *real world*. More precisely, we say that an encryption scheme is secure if the behavior of an adversary $\mathcal{A}_1$ attacking Π in the real world can be simulated by an adversary $\mathcal{A}_2$ in the ideal world. Clearly $\mathcal{A}_2$ should get some help in order to simulate the output of $\mathcal{A}_1$. Indeed, $\mathcal{A}_2$ can have access to some information related to $\mathcal{A}_1$, for example he can access to its description.

More generally, to define the security of a cryptographic primitive, it is necessary to define an ideal world. Then, to prove that a scheme Π securely implements the cryptographic primitive, it is sufficient to construct a simulator (an ideal world adversary) that generates a view that is computationally indistinguishable from the real world adversary's view. We can summarize the tasks that a simulator must fulfill as follows [Lin17]: 1) it must generate a view for the real adversary that is indistinguishable from its real view; 2) it must extract the actual inputs used by the adversary in the execution; 3) it must make the generated view consistent with the output that is based on this input.

Understanding how to define security is the first step to capture the cryptographic primitives needed to improve the privacy of distributed ledgers. Moreover, this helps to understand and formalise the notion privacy that we want to achieve in the context of distributed ledgers and blockchains.

2.7 Post-Quantum Cryptography

Post-quantum cryptography concerns cryptographic schemes that are secure against an attacker making use of a large-scale quantum computer. Nearly all public key cryptography in use today is based on either RSA, or the discrete logarithm problem in cyclic groups or in elliptic curves over finite fields. In 1994 Shor proposed a quantum algorithm to factor integers or solve the discrete logarithm problem in time polynomial in the size of the problem [Sho94]. This means that, should quantum computers become reality in the future, the public key schemes in use today will be broken. Furthermore, any messages encrypted using these schemes that have been recorded in the past can then also be deciphered.

Another issue for post-quantum cryptography is to take into account the consequences of Grover's algorithm [Gro96]. Grover's algorithm is a quantum algorithm that is typically described as being capable of searching an unordered database in time which scales with the square root of the size of the database in contrast with the linear time required by a classical computer.

In actuality, Grover's algorithm performs a more general task of searching for solutions satisfying some efficiently computable predicate. Examples would be to find pre-images of hash functions, or find keys used for symmetric encryption.

Although the square root speed up of Grover's algorithm compared to classical search algorithms is not as dramatic as the speed-up of Shor's algorithm, it does mean that key and hash sizes for which brute force attacks are deemed infeasible using classical computers could be performed using a quantum computer. It is expected that key and hash sizes need to be doubled to provide the same level of security against an attacker using a quantum computer as currently offered against classical attackers.

To prepare for the eventuality of quantum computers, alternative asymmetric cryptographic schemes are being developed to withstand attackers with quantum computing capabilities. At the time of writing, the U.S. National Institute of Standards and Technology's Post-Quantum Cryptography Standardization project [Nat17], the first large scale standardization project of its kind, has finished accepting proposals for post-quantum cryptography standards and is in the evaluation stage of the proposals. Most proposals for post-quantum cryptography are based on one of a few problems which are thought to resist quantum computing. The most popular of these problems for candidate post-quantum cryptography standards are lattice-based, code-based and multivariate-based cryptography. For signature schemes hash-based cryptography is also suitable.

Not all cryptography is vulnerable to attacks using quantum computers. Information theoretically secure methods are secure even against unlimited computational resources. Examples thereof are one time pad encryption, secure multi-party computation with honest majority and information theoretically secure message authentication codes. Of course, in practice, these methods often make use of pseudo randomly generated numbers and are typically used in conjunction with non information theoretically secure cryptography, which may still make systems using such schemes vulnerable to quantum attacks.

Chapter 3

Authentication-related primitives

Chapter 3 gives an overview of primitives related to authentication. Many of the benefits of the blockchain come from the power of being able to preserve data. However, keeping the integrity of the data is not very useful if the data itself is incorrect due to either error or malice. Thus authentication of the data is a necessary part of DLTs. This chapter gives an overview of different flavours of signatures, many of which have been used or been proposed for use in various blockchains. It also views the verifiable random functions, anonymous credential schemes, hash chains, anonymous authentication, and chameleon hashes — primitives that have either been used or proposed to be used for blockchains. A description of well-known elementary primitives is included for completeness.

3.1 Elementary Primitives

Here we recall some elementary cryptographic protocols. As they are commonly known, we shall be brief about them.

Hash functions are deterministic functions that map arbitrary length inputs to fixed-length outputs. There are several properties that hash functions are supposed to have which vary depending on the context. Generally, it should be hard to find two different values x and x' so that $h(x) = h(x')$ where h is the hash function — this is called collision resistance. Given y, it should be hard to find a x so that $h(x) = y$ — this is called pre-image resistance. Also, given x, it should be hard to find $x' \neq x$ so that $h(x) = h(x')$ — this is called second pre-image resistance. In cryptographic contexts they are often used as instantiations of random oracles.

A *commitment scheme* is a scheme that, given a secret value x allows one to commit to it and publish the commitment c_x so that one one hand, others do not learn anything about x from the commitment, and on the other hand, there is a revealing phase where one reveals x and where others can verify that c_x was indeed computed from x.

Signatures can be created only by the owner of a private key but that can be verified by anyone that has the corresponding public key. They are used for showing the authenticity — if the owner publishes some data X and his signature $y = s_{sk}(X)$, then everybody can compute the verification $v_{pk}(y, X)$ to learn whether y corresponds to X or not. If it does not, then this suggests that X has been tampered with.

3.2 Signatures

3.2.1 Server-Supported Signature Schemes

Server-supported signatures, as the name implies, are characterized by the fact that the signer has to co-operate with a server to produce a signature. The two main motivations for such schemes are:

- performance: costly computations can be offloaded from an underpowered signing device (such as a smart-card) to a more powerful device (such as a desktop computer or a server);

- security: risks of key misuse can be reduced by either keeping the keys in a server environment which can presumably be managed better than an end-user's personal computer, or by having the server perform additional checks as part of the signature generation protocol.

When the signer fully trusts the server, an obvious solution is to let the server just handle all asymmetric-key operations based on requests from the signer authenticated using symmetric-key techniques. The drawback of this approach is that if the server abuses the key, the signer can't hold the server accountable. To solve this problem, several protocols have been proposed where the server's actions are verifiable: for any valid-looking signature, the signer either authorized it or can prove that the server has misbehaved.

The first such protocol by Asokan et al. [ATW96] combined an asymmetric-key signature primitive with one-time password authentication. For each message, the signer has to verify the signature the server produced, thus the protocol only saves the signer's resources in case verification is cheaper than signing in the underlying asymmetric-key primitive. Additionally, the signer has to keep copies of all signatures.

Bicakci and Baykal [BB04] replaced the one-time passwords with one-time hash-based signatures. In the resulting protocol the signer has to create a new one-time key pair and compute a one-time signature for each message, but does not have to verify the asymmetric-key signature produced by the server and also the burden of keeping communication history is shifted from the signer to the server. Goyal [Goy04] proposed a method to reduce the storage requirements.

A common drawback of the above protocols is that a malicious server can create signatures that appear to be valid to a verifier until challenged by the signer. As such, they may be usable for signing digital equivalents of traditional contracts, where a dispute resolution process exists, but are unsuitable for applications with immediate and irrevocable effects, such as authentication for access control purposes or accepting transactions to an append-only ledger.

As an alternative research avenue, several methods have been proposed for outsourcing more expensive computation steps of specific signature algorithms, starting with Matsumoto et al. [MKI88] for RSA in particular. However, many of the early proposals have subsequently been shown to be insecure [PW92, LL95].

In recent years, perhaps also due to increasing computational power of handheld devices and wider availability of hardware-accelerated implementations of popular cryptographic algorithms, attention has shifted to splitting keys between end-user devices and back-end servers to improve the security of the private keys, for example by Camenisch et al. [CLNS16] and Buldas et al. [BJKO17].

As a somewhat different approach, Buldas et al. [BLT17] combined hash-linked time-stamping and message authentication codes with one-time keys to obtain a server-assisted signature scheme.

3.2.2 Compact Types of Signature Schemes

This section reviews signature schemes with improved storage efficiency, which is important in the blockchain setting as it allows to save significant storage space.

Multi-Signatures A *multi-signature scheme* enables a group of signers to produce a joint signature on a common document. A trivial implementation of a multi-signature scheme concatenates the individual signatures of all signers; the size of such a multi-signature, however, grows linearly in the number of signers. Specialised multi-signature schemes achieve better efficiency in terms of signature size.

Existing multi-signature schemes. The first line of work on multi-signature schemes did not protect from rogue secret keys; these schemes are therefore not discussed in this overview. The first scheme protecting against adversarially created keys was developed by Micali et al. [MOR01], and uses a multi-party protocol for the key generation. As a consequence, no further users can join the protocol after the setup. The subsequent works of Boldyreva [Bol03] and Lu et al. [LOS$^+$06] solved this issue, but require that knowledge of the secret key is proven when registering the public keys. Ristenpart and Yilek [RY07] later showed that the proof of possession sometimes performed by CAs today is sufficient to justify the knowledge-of-secret-key assumption used in those schemes.

Scheme	**Setup**	**Registry**	**Rounds**	**Assumption**	**|sig|**	**|pk|**	**Sign**	**Verify**
[MOR01]	RO	interactive	2	DL	$2\|\mathbb{G}\|$	$(3+2\log N)\|\mathbb{G}\|$	1 exp.	N exp.
[Bol03]	RO, CRS	known sk	1	GapDH	$\|\mathbb{G}\|$	$\|\mathbb{G}\|$	1 exp.	$2N$ p.
[BGLS03]	RO	bare pk	1	co-GapDH	$\|\mathbb{G}\|$	$\|\mathbb{G}\|$	1 exp.	N p.
[LOS$^+$06]	CRS	known sk	1	co-CDH	$2\|\mathbb{G}\|$	$\|\mathbb{G}\|$	3 exp.	$2N$ p.
[BN06]	RO, CRS	bare pk	3	DL	$\|\mathbb{G}\| + \|q\|$	$\|\mathbb{G}\|$	1 exp.	N exp.
[BJ08]	RO, CRS	bare pk	3	DDH	$2\|q\|$	$2\|\mathbb{G}\|$	1 exp.	N exp.
[BCJ08]	RO, CRS	bare pk	2	DL	$3\|\mathbb{G}\| + 3\|q\|$	$\|\mathbb{G}\|$	3 exp.	N exp.
[Nev08]	RO	bare pk	1	RSA	$n + \|\mathbb{G}\|$	$\|\mathbb{G}\|$	1 exp.	$2N$ mult.
[MWLD10]	RO, CRS	bare pk	2	DL	$3\|q\|$	$\|\mathbb{G}\|$	1 exp.	N exp.
[MPSW18]	RO	bare pk	2	one more DL	$\|\mathbb{G}\| + \|q\|$	$\|\mathbb{G}\|$	$N+1$ exp.	$N+2$ exp.
[BDN18]	RO	bare pk	2	co-CDH	$\|\mathbb{G}\|$	$\|\mathbb{G}\|$	1 exp.	$N+1$ p.

Table 3.1: Comparison of multi-signature schemes from the literature. Size $|\mathbb{G}|$ refers to group elements, whereas $|q|$ refers to (possibly shorter) exponents. Size measurement in number of elements is only partially meaningful because necessary group sizes will differ in different settings. The overhead of [Nev08] can be up to $(n+1)|\mathbb{G}|$ for short messages, the table indicates the best case for large messages. The last two schemes [MPSW18, BDN18] additionally provide key aggregation. Verification is stated for *all* signatures.

The current state-of-the-art multi-signature schemes only require the bare public key model without the need to prove possession of the secret key. They differ in the number of communication rounds required for signing, the efficiency in terms of computation as well as key and signature size, and the assumptions under which they are proven secure. One early and somewhat special scheme is that of Boneh et al. [BGLS03], which can be used as a multi-signature scheme as discussed by Bellare et al. [BNN07]. The resulting scheme differs from most following schemes in that it does not require a CRS, is non-interactive, and is based on pairings. It has small signature and key sizes, but a comparably slow verification.

Most following schemes are based on the work of Bellare and Neven [BN06], which has a three-round signing protocol, is based on the discrete-logarithm problem, proven in the random-oracle model, and requires a CRS. Subsequent work achieves a smaller signature size [BJ08] or less communication rounds [BCJ08, MWLD10], at the cost of other parameters. Recently, Boneh et al. [BDN18] revisited the scheme of [BGLS03] and applied techniques of [MPSW18] to achieve an efficient multi-signature scheme with public-key aggregation. The scheme is also useful as an aggregate-signature scheme and as an accountable-subgroup multi-signature, which is similar to a threshold signature without the anonymity property.

The parameters of all schemes are summarized in Table 3.1. The assumptions required by the schemes differ. Some schemes are based on the discrete logarithm (DL) assumption in a cyclic group $\mathbb{G}$, meaning that given g^a for some known generator $g \in \mathbb{G}$, computing a is infeasible. Other schemes use stronger assumptions, such as the "one more DL" assumption which states it is hard to compute $q + 1$ discrete logarithms with q queries to a discrete-logarithm oracle, for any q (polynomial in the security parameter). The Decisional Diffie-Hellman (DDH) assumption states that for uniformly random $a, b, c \in \{1, \ldots, |\mathbb{G}|\}$, the triples (g^a, g^b, g^c) and (g^a, g^b, g^{ab}) are indistinguishable. The GapDH assumption requires that it be infeasible to compute, given g^a, g^b, the value g^{ab}, even in presence of an oracle that solves DDH. The co-CDH and co-GapDH assumptions are stated in the pairing setting, namely a setting with three groups $\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_{\mathrm{T}}$ and a bilinear map $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_{\mathrm{T}}$. Here, co-CDH requires that given $g, g^a \in \mathbb{G}_1$ for uniformly random $a \in \{1, \ldots, |\mathbb{G}_1|\}$ and $h \in \mathbb{G}_2$ it be hard to compute h^a. The assumption co-GapDH requires that this hold even given an oracle that distinguishes tuples g, g^b, h, h^b from tuples g, g^b, h, h^c for uniformly random $b, c \in \{1, \ldots, |\mathbb{G}_1|\}$. The columns **Sign** and **Verify** indicate the computational cost of the algorithms in terms of the most expensive operations required, which, depending on the setting, may be modular multiplication (mult.), modular exponentiation (exp.), or pairings (p.), i.e. evaluations of the mapping e described above.

Use of multi-signature schemes in DLT. Maxwell et al. [MPSW18] discuss the use of multi-signatures for Bitcoin. A multi-input multi-output (MIMO) transaction in Bitcoin contains signatures (on the same data) with

Scheme	Setup	Assumption	|sig|	|pk|	Sign	Verify	Type
[BGLS03]/[BNN07]	—	co-GapDH	$\|\mathbb{G}\|$	$\|\mathbb{G}\|$	1 exp.	n p.	general
[LMRS04]	RO	certified TDP	$\|\mathbb{G}\|$	$\|\mathbb{G}\|$	1 exp.	n exp.	sequential
[LOS$^+$06]	CRS	co-CDH	$2\|\mathbb{G}\|$	$(k+2)\|\mathbb{G}\|$	4 exp.	$2n$ p.	sequential
[Nev08]	RO	RSA	$\|\mathbb{G}\| + \ell$	$\|\mathbb{G}\|$	$n+1$ exp.	n exp.	sequential
[BGR12]	RO	RSA	$\|\mathbb{G}\| + (n+1)\ell$	$\|\mathbb{G}\|$	1 exp.	n exp.	sequential
[GOR18]	RO	RSA	$\|\mathbb{G}\|$	$\|\mathbb{G}\|$	$n+1$ exp.	n exp.	sequential
[FLS12]	—	co-GapDH+coll.res.	$3\|\mathbb{G}\| + \ell$	$\|\mathbb{G}\|$	n exp. + n p.	$n+1$ p.	sequential
[Sch11]	—	LRSW (interactive)	$4\|\mathbb{G}\|$	$2\|\mathbb{G}\|$	n p. + $2n$ exp.	n p. + $2n$ exp.	sequential
[LLY15]	—	SXDH, LW2, DBDH	$8\|\mathbb{G}\|$	$11\|\mathbb{G}\|$	8 p. + $4n$ exp.	8 p. + $5n$ exp.	sequential

Table 3.2: Comparison of aggregate signature schemes from the literature. Size $|\mathbb{G}|$ refers to group elements, whereas $|q|$ refers to (possibly shorter) exponents. Size measurement in number of elements is only partially meaningful because necessary group sizes will differ in different settings. Value k refers to the length of signed messages. Size ℓ is (approx.) the output length of a hash function used in [Nev08], [BGR12], and [FLS12].

all keys corresponding to the inputs of the transaction. A multi-signature can therefore decrease the size of the transactions stored in the blockchain. The aggregation of public keys claimed by [MPSW18] additionally decreases transaction size. Recent work of Drijvers et al. [DEFN18] shows, however, that the proof of [MPSW18] is flawed and is unlikely to be recovered under standard assumptions.

Aggregate Signatures An *aggregate signature scheme* enables a group of signers to combine their signatures into a compact representation. In contrast to multi-signatures, aggregate signatures do not require the signed messages to be equal.

An overview of schemes described below is given in Table 3.2. Most abbreviations are analogous to the ones used in Table 3.1 and described above, and we only describe the differences here. A trapdoor permutation (TDP) is a family of permutations that are easy to compute but difficult to invert, but which can be sampled together with a *trapdoor* that allows for easy inversion. The most prominent example of a TDP is RSA. A *certified* TDP is one where the public key, i.e. the description of the permutation, allows to determine whether it describes a valid instance of the TDP. The term "coll.res." indicates that the scheme additionally uses a hash function that is required to be collision resistant. The LRSW assumption, named after the paper introducing it [LRSW99], states that given an oracle that on input an integer s returns a triple (a, a^{sy}, a^{x+sxz}), it is difficult to produce a new integer t and triple (b, b^{ty}, b^{x+txz}) that is non-trivial, i.e., $t \neq s$ and b is not the neutral element. The symmetric external Diffie-Hellman (SXDH) assumption is in the pairing setting and can be seen as the DDH assumption in the group $\mathbb{G}_2$. The LW2 assumption was introduced by Lewko and Waters [LW10] assumes that, with generators $g_1 \in \mathbb{G}_1$ and $g_2 \in \mathbb{G}_2$, given g_1^a, g_1^b, g_1^c and $g_2^a, g_2^{a^2}, g_2^{bx}, g_2^{abx}, g_2^{a^2x}$, it is difficult to distinguish g_1^{bc} from g_1^d, where all exponents a, b, c, d, x are uniformly random. Finally, the decisional bilinear Diffie-Hellman (DBDH) assumption requires that it be difficult to distinguish, given g_1^a, g_1^b, g_1^c and g_2^a, g_2^b, g_3^c, the values $e(g_1, g_2)^{abc}$ and $e(g_1, g_2)^d$, with a, b, c, d uniformly random.

The first construction of aggregate signatures was given by Boneh et al. [BGLS03] and is based on pairings. Bellare et al. [BNN07] later showed that a certain condition (relating to messages being distinct) can be dropped for a certain variant of the scheme.

Sequential aggregate signatures. Introduced by Lysyanskaya et al. [LMRS04], a sequential aggregate signature scheme achieves the same security as a (general) aggregate signature scheme, but requires that the signatures be aggregated in a certain predetermined order. This restriction allows for more efficient implementations, in particular, they show a scheme based on certified trapdoor permutations which is secure in the random-oracle model. The scheme of Lu et al. [LOS$^+$06] achieves security in the standard model, based on pairings and at the cost of efficiency. Analogously to the multi-signature presented in the same paper, the scheme requires that signers prove knowledge of their secret key. The scheme of Neven [Nev08] realizes sequential aggregate signatures based on a weaker assumption, not requiring the trapdoor permutation to be certified. The scheme of Brogle et al. [BGR12] allows *lazy verification* in which each partial signature can be checked independently. Recent work of Gentry et al. [GOR18] improves said constructions. (Their proofs do not support instantiations for realistic parameter sizes.) Fischlin et al. [FLS12] provide a sequential aggregate signature scheme based on pairings, which additionally is *history-free* in the sense that a signer is not required to know the messages that are already signed in the partial signature they are amending. The schemes of Schröder [Sch11] and Lee et al. [LLY15] also assume that the signers prove possession of their secret keys.

Use of aggregate signature schemes in DLT. Aggregate signatures allow for a more compact representation of multiple signatures. Especially for data that is stored on many replicas, such as the blockchain itself, the overall savings in storage can be significant. Possible methods are, e.g., to aggregate the signatures of all transactions within each block.

3.2.3 Privacy-Enhancing Signature Schemes

Blind Signatures Blind signatures are a form of signatures where the signer does not see the message M signed as it has been disguised in some way so that he sees only a randomized version of M. However, the verification still checks the signature against the original message M. Unlike digital signatures, blind signatures are generated by means of an interactive protocol between the signer and a receiver. The signer does not see the message being signed, and, in addition, the signer does not learn any useful information on the signature being produced.

Blind signatures are due to Chaum, who also invented the well-known RSA-based blind signature scheme [Cha82, Cha83]. A blind signature scheme consists of three components:

Key generation. An algorithm that on input of a security parameter k, generates a key pair (sk, pk) consisting of a private key and a public key, respectively.

Signature generation. A two-party protocol between a signer $\mathcal{S}$ and a receiver $\mathcal{R}$ with a public key pk as common input. Private input of $\mathcal{S}$ is a private key sk, and private input of $\mathcal{R}$ is a message M. At the end of the protocol, $\mathcal{R}$ obtains a signature S on M as *private* output.

Signature verification. An algorithm that on input of a message M, a public key pk, and a signature S, determines whether S is a valid signature on M with respect to public key pk.

A blind signature scheme is required to be unforgeable and unlinkable, defined as follows. Let (sk, pk) be a key pair for a blind signature scheme. A pair (M, S) is valid if signature verification of M and S with respect to public key pk succeeds.

A blind signature scheme is **unforgeable** if for an adversary (not knowing sk) the only feasible way to obtain valid pairs (M, S) is to execute the signature generation protocol with a signer holding private key sk. More precisely, a blind signature scheme should withstand a **one-more forgery**: if an adversary is able to obtain ℓ valid pairs of messages and signatures, then the signer executed the signature generation protocol at least ℓ times. Preferably, we like this to hold for any positive ℓ bounded polynomially in the security parameter k.

A blind signature scheme is **unlinkable** if for an adversary (colluding with a signer) it is infeasible to link any valid pair (M, S) to the instance of the signature generation protocol in which it was created. More precisely, suppose a signer $\mathcal{S}$ and a receiver $\mathcal{R}$ play the following game. First they run the signature generation protocol resulting in a pair (M_0, S_0) and then they run it once more, resulting in (M_1, S_1). Then $\mathcal{R}$ flips a coin, that is, chooses $b \in_R \{0, 1\}$ and sends (M_b, S_b), (M_{1-b}, S_{1-b}) (in this order) to $\mathcal{S}$. Finally, $\mathcal{S}$ makes a guess for the value of b. Unlinkability means that the probability of $\mathcal{S}$ guessing b correctly is $\frac{1}{2}$, except for a difference negligible in the security parameter k.

Ring Signatures A *ring signature* allows a signer to sign a message such that the authenticity can be verified relative to a *ring* of users, that is, a spontaneous set of users chosen by the signer when creating the signature. The other members of the ring do not cooperate in the signing or may not even be aware of it. Ring signatures provide a certain level of anonymity since a verifier will only learn that the signer is among the users in the ring, but will not learn which user has signed. The term ring signature was coined by Rivest et al. [RST01], although earlier group-signature schemes already used similar schemes as building blocks of their constructions (cf. [HS03]).

Dodis et al. [DKNS04] describe the first ring signature scheme in which the signature size does not grow with the number of users, and show it secure in the random-oracle model. Ring-signature schemes based on pairings and secure without random oracles have been described by Shacham and Waters [SW07] with signatures growing linearly in the number of ring members, Chandran et al. [CGS07], with signature size growings sub-linearly in the number of ring members, and by Chow et al. [CWLY06] with constant-size signatures. The work of Bender et al. [BKM09] points out that previous constructions of ring signatures assume that public keys are generated honestly; they provide a stricter security definition. They then also describe a generic construction and two more efficient constructions based on specific computational assumptions and only for rings of size 2. A first (still not practical) lattice-based scheme has been given by Melchor et al. [MBB$^+$13]. Another scheme that achieves post-quantum security is described by Derler et al. [DRS18]. It is only based on symmetric primitives.

Ring signatures with advanced security properties exist and are interesting for use in combination with DLT. For instance, *threshold ring signature schemes* combine threshold signing with the anonymity guarantees of ring signatures. A first scheme has been described by Bresson et al. [BSS02], based on the RSA assumption and secure in the random-oracle model. Wong et al. [WFLW03] have described a construction extending the original work of Rivest et al. [RST01]. *Separable ring signature schemes* furthermore allow to combine several users' signatures into a threshold ring scheme, even although the keys are for different types of signature schemes [LWW03]. A *linkable ring signature scheme* allows to determine whether two ring signatures have been created by the same user [LWW04, LW05]; a scheme with constant-size signatures based on RSA has been proposed by Tsang and Wei [TW05]. Recently, Baum et al. [BLO18] have described a scheme based on hardness of lattice problems. The discussed properties can also be combined in a single scheme [TWC$^+$04].

Accountable ring signatures allow to specify one designated opener that can revoke the anonymity and have been introduced by Xu and Yung [XY04]. An implementation based on the DDH assumption, in which the signature size grows logarithmically with the number of ring users, has been proposed by Bootle et al. [BCC$^+$15]. This has been improved to constant-size signatures by Lai et al. [LZCS16] and Kumawat and Paul [KP17].

Traceable ring signatures, as introduced by Fujisaki and Suzuki [FS08] allow each user to only use the ring signature once for a certain context; this can be used for preventing double spending. The original scheme is based on the decisional Diffie-Hellman assumption and proven in the random-oracle model [FS08], a later work of Fujisaki [Fuj11] devises a scheme where the signature grows sub-linearly in the number of ring members, based on pairing-based cryptography and secure in the standard model. Recent work of Gu and Wu [GW18] describes a scheme with constant-size signatures, based on the CDH assumption.

Use of ring signatures in DLT. Various cryptocurrencies have used ring signatures to improve transaction privacy. The underlying idea is that a transaction is signed with a ring signature, which is signed relative to a certain number of transaction outputs, hiding which exact output was spent. Since a vanilla implementation would allow for double spending, ring signatures with additional features are used. CryptoNote [vS13], for instance, is based on the traceable ring signature scheme of Fujisaki and Suzuki [FS08]. The Monero cryptocurrency [Noe15] uses the scheme of Liu et al. [LWW04]. An improved protocol has recently been described and is based on similar technical ideas [SALY17].

Group Signatures Group signature were introduced by Chaum and van Heyst [CvH91] as a special type of digital signatures satisfying three extra requirements.

A group signature scheme for a group formed by a group leader $\mathcal{P}_0$ and group members $\mathcal{P}_1, \ldots, \mathcal{P}_\ell$, $\ell \geq 1$, consists of four components:

Key generation. A protocol between $\mathcal{P}_0, \mathcal{P}_1, \ldots, \mathcal{P}_\ell$ for generating a public key h for the group, a private key x_0 for the group leader $\mathcal{P}_0$ and a private key x_i for each group member $\mathcal{P}_i$, $1 \leq i \leq \ell$.

Signature generation. An algorithm that on input of a message M, the public key h of the group, and a private key x_i of a group member $\mathcal{P}_i$, outputs a group signature S.

Signature verification. An algorithm that on input of a message M, the public key h of the group, and a signature S, determines whether S is a valid group signature on M with respect to public key h.

Signature opening. An algorithm that on input of a message M, the public key h of the group, a valid group signature S, and the private key x_0 of the group leader, outputs the identity of the group member who generated S.

In addition to the requirements for a basic digital signature scheme, a group signature scheme should satisfy the following requirements related to the anonymity of a group member and the role of the group leader. Given a signature, no-one except the group leader should be able to tell which group member produced a given signature. More generally, given two signatures, no-one except the group leader should be able to tell whether these signatures were produced by the same group member or not (**unlinkability**). Of course, group members should not be able to produce signatures on behalf of other group members. Similarly, a group leader should not be able to *frame* a group member $\mathcal{P}_i$ by opening a signature produced by $\mathcal{P}_j$ as if it was produced by $\mathcal{P}_i$ ($i \neq j$).

Ring signatures and list signatures can be viewed as special types of group signatures. Ring signatures are well-known (see previous section) and can be seen as group signatures without a group leader. Indeed, as noted at the end of [CDS94], direct application of 1-out-of-ℓ proofs yields this type of group signatures.

List signatures. Canard et al. [CSST06] introduced list signatures as a variant of group signatures setting a limit on the number of signatures each group member may issue. The basic idea behind list signatures goes back to Stam's master's thesis [Sta99], which covers mechanisms to ensure that voters cannot cast more than one vote in an anonymous election. The limits must be enforced without having the group leader open signatures of honest group members—which excludes the trivial solution in which the group manager opens every signature to see whether some group members exceed their limits. Furthermore, list signatures enable public identification of group members who exceed their limits, also without involving the group manager. See [CSST06] for constructions, both for small groups (complexity grows with size of group) and large groups (complexity independent of group size).

Threshold Signatures Threshold signatures are a specific type of multi-signatures. For a set of n users and a specific threshold $t \leq n$, generation of a signature is possible if and only if t signers collaborate.

Threshold cryptosystems based on the discrete logarithm problem have been suggested early by Desmedt and Frankel [DF89]. Their scheme required a trusted party to generate all keys; this restriction is resolved by the scheme of Pedersen [Ped91]. Many protocols for generating the distributed keys have been developed later [GJKR07]. Gennaro et al. [GJKR96] describe a threshold signature based on DSA which is additionally *robust* in the sense that malformed shares contributed by dishonest parties will not affect the correctness of the signature. Later, Gennaro et al. [GGN16] presented a protocol for DSA signature generation that requires 6 rounds of interaction. All these solutions, however, require an interactive protocol between the participants for signature generation. This interactive signing phase appears in most of the early discrete-logarithm-based threshold signature schemes and may limit their applicability in distributed ledgers.

Shoup [Sho00] describes the first practical threshold signature scheme based on RSA that has a non-interactive signing phase. The protocol requires a trusted setup phase to generate the keys; this requirement has been overcome by Damgård and Koprowski [DK01]. Later, Gennaro et al [GHKR08] describe a variant of the scheme that works for dynamically changing groups of users. Boneh et al. (BLS) introduce a non-interactive threshold signature scheme in the random-oracle model using elliptic-curve cryptosystems with bilinear maps, which requires a pairing operation for verification [BLS04]. A non-interactive scheme based on the GapDH assumption[1] has been given by Boldyreva [Bol03], and builds on the BLS signature scheme [BLS04].

[1] The GapDH assumption requires that in a cyclic group $\mathbb{G}$ with generator g, given g^a, g^b with uniformly random a, b, it be difficult to compute g^{ab} even in presence of an oracle that distinguishes Diffie-Hellman triples g^x, g^y, g^{xy} from random triples g^x, g^y, g^z.

Recently, Boneh et al. [BGGK17] describe what they call a *universal thresholdizer*, a method to convert any signature scheme into a threshold signature scheme. This method leads to the first (non-interactive) lattice-based threshold signature scheme, but is based on computationally expensive techniques such as fully homomorphic encryption.

Use of threshold signature schemes in DLT. Gennaro et al. [GGN16] suggest the use of threshold DSA for generating signatures in Bitcoin transactions, as a measure against compromise of secret keys. A further area where threshold signatures are useful is permissioned blockchain systems, meaning systems in which all users have registered identities, in which a natural model is for a threshold of nodes to agree on a specific action.

Key-Evolving Signatures In regular digital signature schemes, an adversary who compromises the signing key of a user can generate signatures for any messages it wishes, including messages that were (or should have been) generated in the past. *Forward secure signature schemes* [BM99] prevent such an adversary from generating signatures for messages that were issued in the past, or rather allows honest users to verify that a given signature was generated at a certain point in time. Basically, such security guarantees are achieved by "evolving" the signing key after each signature is generated and erasing the previous key in such a way that the actual signing key used for signing a message in the past cannot be recovered but a fresh signing key can still be linked to the previous one. This notion is formalized through *key evolving signature schemes*, which allow signing keys to be evolved into fresh keys for a number of time periods. Efficient constructions of key evolving signature schemes with forward security are given e.g. in [IR01, MMM02, KR02]. An efficiency comparison of various forward secure signature schemes is given in [CJMM03].

Use of key-evolving signatures in DLT. The primary use of key-evolving signatures in blockchain protocols (more specifically: proof-of-stake protocols) is to achieve security against adaptive corruptions. In a nutshell, these schemes are used to sign blocks (or other protocol messages), and honest participants are mandated by the protocol to update their secret keys regularly. Typically, a participant that is selected to act by the protocol (e.g. to create a new block), signs this block with a key-evolving signature, evolves its key, and only then broadcasts this signed block. In this way, even if the adversary is capable to corrupt this party immediately after it discloses its role (by broadcasting the block), he can no longer take advantage of the secret key obtain through the corruption to forge a signature on an alternative block. This general approach is used in the protocols Algorand [Mic16], Ouroboros Praos [DGKR18] and Ouroboros Genesis [BGK$^+$18].

3.3 Verifiable Random Functions

A verifiable random function (VRF for short) is a public-key analogue to the well-known pseudorandom functions (PRFs), consisting of three algorithms F, P, V. It allows the owner of a secret key sk to evaluate the function on arbitrary input x, obtaining an output $y = F_{sk}(x)$ that is unique, and pseudorandom for anyone not holding the key sk. However, using the secret key sk it also allows to generate a proof $\pi = P_{sk}(x)$ that can later be used by anyone holding the public key pk corresponding to sk to verify that y is indeed the correct value of F on x, by checking $y = V_{pk}(y, x, \pi)$.

Verifiable random functions were introduced by Micali, Rabin and Vadhan in [MRV99]. A more efficient VRF construction was later proposed in [DY05].

Use of verifiable random functions in DLT. Verifiable random functions have been used in proof-of-stake protocols to realize a local, private lottery selecting participants eligible to act in the protocol proportionally to their stake share. The local and private nature of the lottery helps achieve the security of the protocol against instant adaptive corruptions, as the potential positive outcome of the lottery remains hidden to the other participants (and the attacker) until the winning party acts, at which point corrupting the party can no longer compromise the protocol. This use of VRFs has appeared in the NXT cryptocurrency [Com14], and the protocols Algorand [Mic16], Ouroboros Praos [DGKR18] and Ouroboros Genesis [BGK$^+$18].

Additionally, VRFs are also used in [Mic16, DGKR18, BGK^{+}18] in the process of generating protocol randomness in a way that is efficient (hashing-based, as opposed to a coin-tossing protocol in [KRDO17]) and yet the resulting randomness can only be biased by the attacker in a limited amount that can be contained by the protocols.

3.4 Anonymous Credential Schemes

The purpose of an *anonymous credential scheme* is to allow a user to prove claims about their identity in a privacy-preserving way. The user obtains a certificate that binds attributes related to the user, such as name, birth date, or employer, to the user's cryptographic key. Different from a traditional certificate scheme, an anonymous credential system allows the user to *selectively and partially* open attributes toward the verifier, and the verifier will not be able to infer more information about the user than the fact that the attribute is true. For instance, this allows a user to prove toward a web site that he is older than some minimum age required to access the web site, without disclosing his name or any other privacy-sensitive information.

Cryptographic schemes that provide anonymous credentials have been proposed a series of works by Camenisch and Lysyanskaya [CL01, CL02, CL04] and by Brands et al. [BDD07], and have been implemented in Identity Mixer [CH02] and U-Prove [PZ13].

Use of anonymous credential schemes in DLT. Anonymous credential schemes are useful in permissioned DLT to allow registered users to anonymously but authentically initiate transactions. More concretely, Identity Mixer technology is in the process of being implemented in Hyperledger Fabric [Dub17]. Another combination of DLT and anonymous credential schemes is by using a distributed ledger for providing credentials in a privacy-friendly way. The Sovrin project [Sov18] builds on the Hyperledger Indy platform, which also uses Identity Mixer technology.

3.5 Hash Chains

Hash chains are the fundamental concept underlying any blockchain: each block contains the hash of the preceding block. This also means that each block inherently authenticates the entire history up to that block. Hash chains are closely related to the concept of Merkle trees [Mer80] but appear explicitly first in the work of Haber and Stornetta [HS90] in the context of time stamping. The underlying idea is for a time stamping service to accept documents, hash them (together with the previous hash), and publish the hash value e.g. in a newspaper. Subsequent work by Benaloh and de Mare [BdM93] and Buldas et al. [BLLW98] extended the efficiency of the time stamping service using different tree shapes.

Hash chains have since been used in multiple distributed systems protocols, such as the Blind Stone Tablet (BST) by Williams et al. [WSS09], as a means to achieve fork-linearizability in cloud-based storage with multiple clients. In contrast to those protocols, and starting with Bitcoin [Nak09], blockchain systems collate multiple transactions into a block, and builds a hash chain over those blocks. All subsequent blockchain protocols follow the same approach.

Several newer schemes generalize the concept of a blockchain toward a directed acyclic graph of blocks, or block-DAG, as the total order of transactions achieved by a blockchain may be unnecessarily strict and can become a performance bottleneck. Such systems include Swirlds [swi], Iota [iot], Spectre [SLZ16] and Phantom [SZ18]. The consistency of these approaches is, however, not as well understood as that of blockchains.

3.6 Authenticated Data Types

An *authenticated data type* (ADT, also authenticated data structure) [Tam03] allows a client to outsource data to a server while guaranteeing the integrity of the data. In a nutshell, while the server stores the data, the client holds a small *authenticator* (or *digest*) that relates to it. Operations on the data are performed by the server, and for each operation the server computes a proof that, together with the authenticator, allows the client to check that the server performed the operation correctly.

Merkle trees [Mer80, Mer89] can be seen as the prototype realization ADT, efficiently implementing a bounded-length array and providing efficient proofs for array elements. Many ADTs for specific data structures and related specific types of queries have been described in the literature. Merkle trees apply to bounded-length arrays, where entries can be updated. Other instantiations exist for data structures such as sets [NN00, PTT11], dictionaries [AGT01, GTS01], range trees [MNG$^+$01], hash tables [PTT08] and many more. A recent construction of Cachin et al. [CGPT17] provides a construction for arbitrary abstract data types, based on verifiable computation. Generally, ADTs can be seen as special cases of (stateful) verifiable computation schemes [WB15, PHGR13, FFG$^+$16, CGPT17] and NIZKs, where the server can prove the correctness of an arbitrary computation. Furthermore, recent work has seamlessly integrated an ADT into a programming language [MHKS14].

Use of authenticated data types in DLT. Besides the appearance of a hash chain as the underlying data structure of a blockchain, Merkle trees are used in various ways for compact proofs of membership. As an example, Zerocash [BCG$^+$14a] keeps a Merkle tree of all commitments of assets seen on the ledger, and a membership proof to show validity.

3.7 Anonymous Authentication

Cryptography is useful to solve many real-world tasks that sometimes conflict with each other. A requirement often needed is that of obtaining entity authentication so that a player is ensured about the identity of another player. Another often-needed requirement is related to privacy protection, and the fact that a qualified entity would like to access to some remote services without being traced. The tension between such two requirements can be relaxed through anonymous authentication systems. Here a user can prove to be a qualified user without, however, revealing his identity. The verifier of such system is convinced that the user belongs to a qualified set of users and has no additional information about the identity of the user. This notion has been sometimes referred as anonymous group identification [DKNS04].

Anonymous authentication is clearly connected to group and ring signatures. Indeed such signatures schemes can be seen as building blocks for anonymous authentication. There are constructions of anonymous authentication schemes that leverage on interaction (that is clearly not available in the setting of signatures schemes). Some notable examples are the OR-composition technique of Cramer et al. [CDS94] recently improved in [CPS$^+$16a, CPS$^+$16b], and the deniable ring authentication of Naor [Nao02].

Anonymous authentication is a privacy-preserving primitive that can be useful for privacy-enhancing ledgers since it would allow a legitimate user of a ledger to perform a transaction keeping his identity private. Still the transaction can be verified and validated since it was originated by a legitimate user.

3.8 Chameleon Hashes

Chameleon hashes are similar to hash functions but additionally contain a trapdoor T. For a party who does not know T, a chameleon hash function is similar to a regular hash function in that it is hard to find collisions and a second preimage to a given value. However, if a party knows T, she can easily find collisions and second pre-images. [CDK$^+$17]

These were originally proposed as a mechanism for non-transferrable signatures — the idea is that the recipient of the signature should not be able to pass the signature along to third parties and be believed. This is achieved in the following way. Suppose that Alice signs a message m and obtains a signature s and sends them to Bob, who possesses the trapdoor T and that everybody knows that Bob knows T. Now, Bob is convinced that the message is authentic as other people do not know the trapdoor. However, since Bob knows T, he can find a different message m' the signature of which is also s. Everybody knows that Bob has such abilities and thus, if

Bob passes the pair (m, s) along to Charlie and claims that m was signed by Alice, then Charlie will not believe him, as Bob could have replaced m with any other m'.

Chameleon hashes are potentially interesting to blockchain as they introduce a way to redact a blockchain, for example, to remove some blocks. [AMVA17] This might be desirable if criminal content is stored on the blockchain, for example child pornography. Currently, this has already happened for Bitcoin [bit]. Other possible reasons to modify blockchain include the ”right to be forgotten” — if one has spent all their cryptocurrency, one might wish to remove ones spending history from the blockchain.

The immutability of blockchain relies on the collision resistance of the underlying hash function. However, if some committee is allowed to find collisions, then they can change the blockchain. Authors of [AMVA17] propose a way how it is possible to remove blocks from the blockchain.

The original approach to solve this problem suffered from the *key exposure problem* — namely, if Bob forges a signature, then Alice can obtain Bob's trapdoor from it.

A property that the chameleon hash scheme can have is enhanced collision resistance (ECR)— an adversary who does not know the trapdoor should not be able to produce collisions even after seeing polynomially many collisions. Not all chameleon hashes have this property which was introduced by Ateniese et al. in [AMVA17], along with a scheme that is ECR.

3.9 Cryptographic Bulletin Boards

Consider an application like the collection and storage of ballots during Internet voting and similar applications. Here, the data collection (e.g., voting) period has short timespan, but the data (e.g., encrypted ballots) should stay available for long time period for later auditing. The technology behind this type of ledger is called bulletin boards. A bulletin board system consists of several high-availability bulletin board servers, a large fraction of which is considered to be trusted.

Many of the e-voting schemes assume the existence of a bulletin board system, but the first efficient bulletin board systems were proposed only a few years ago [CS14, CZZ$^+$16]. The bulletin board system of [CS14] was proven secure by using formal methods and lacks, up to our knowledge, a cryptographic security proof. The bulletin board system of [CZZ$^+$16] assumes the existence of a highly trusted election authority.

Chapter 4

Confidentiality-related primitives

This chapter gives an overview of the primitives that strengthen confidentiality that either are or might be useful for blockchains or might benefit from blockchains. Confidentiality is one of the key goals of cryptography and thus a document about cryptographic primitives that failed to discuss confidentiality-related primitives would necessarily be incomplete.

We first remind a few elementary notions for the sake of completeness. Then we overview different flavours of encryption which allow different sets of parties to encrypt and decrypt the data. Distributed ledgers deal with many parties simultaneously which tends to lead to complicated types of interactions with each other. Thus it would be beneficial to provide different possibilities to account for the fact that the desires of the parties might be rather complicated. We also overview private information retrieval, which allows a client to obtain information from a database without the owner of the database learning what information was queried for. As distributed ledgers and blockchains are used often for storing data, cryptographic protocols related to databases is a natural area of interest.

4.1 Elementary Primitives

We here recall some elementary cryptographic protocols. As they are commonly known, we shall be brief about them. *Public key encryption schemes* (PKE) are schemes where there are two related keys — a public key and a private key. A public key is presumed to be known to the public. Anyone who knows the public key can encrypt a message that can be decrypted only by those that know the corresponding secret key. The phrase 'asymmetric encryption scheme' is used as a synonym for 'public key encryption scheme'.

Symmetric encryption schemes are encryption schemes where the same key is used for encrypting and decrypting.

Signcryption schemes are schemes where private-key-secret-key pairs are such that they can be used for both PKE and signature schemes.

An encryption scheme is said to have the ciphertext indistinguishability (IND-CPA) property if an adversary can pick two plaintexts, m_0 and m_1 and is presented an encrypted version of one of these and can not tell whether it is the encryption of m_0 or m_1.

4.2 Functional Encryption

Traditionally, encryption has been an all-or-nothing affair: either a recipient owns the secret key (and thus can decrypt) or she does not (and then learns nothing about the plaintext, except possibly its size). Functional encryption [SW05, KSW08, O'N10, BSW11] enables a much more fine-grained handling of encrypted data. Here, the owner of the master key can delegate partial secret keys to various recipients. In a functional encryption scheme for functionality $\mathcal{F}$, the knowledge of a secret key corresponding to some y enables one to decrypt an encryption of z to $\mathcal{F}(y, z)$. As such, functional encryption has many potential applications, and has spurred a long line of research, see the excellent survey [BSW12].

A functional encryption scheme can be required to satisfy several different security requirements [O'N10, BSW11]. In the case of the *adaptive* IND-FE-CPA security [O'N10, BSW11], it must be difficult for an adversary to distinguish functional ciphertexts of any two plaintexts z_0 and z_1. This must hold even if the adversary is given an oracle access to the partial secret key generator, where the secret key queries must satisfy the condition that $\mathcal{F}(y, z_0) = \mathcal{F}(y, z_1)$ for each queried y. In the weaker *selective security* model, the adversary is required to choose z_0 and z_1 before seeing the public key and answers to any of the secret key queries. See [O'N10, BSW11] for discussion.

4.3 Identity-Based Encryption

The public-key infrastructure has the problem of associating public keys to the identities — somebody has to guarantee that a public key claimed by a e-mail address really belongs to that e-mail address and not to an impostor. The common way to deal with this problem is relegating trust to keyservers.

Identity-based encryption aims for a situation where the public key can be any string (as opposed to the usual encryption schemes where public keys must have a specific structure), for example, the e-mail address that it is tied to. This, however poses a problem — we can presume that e-mail addresses are public knowledge — how can only the owner of that e-mail address obtain the secret key with what to decrypt messages sent to her?

Here the problem is solved with introducing an agency who has the power to create secret keys. That agency, however, requires even more trust as it has all the secret keys it has created. It has to be trusted not to read the mail sent to the users for which it has generated the secret keys and also not to be compromised.

IBE was proposed by Shamir in 1986 [Sha84]. A solution was given by Boneh and Franklin in 2001 [BF01]. Since then, the problem has been subject to a large amount of research.

4.4 Attribute-Based Encryption

Attribute-based encryption is a type of functional encryption where the owner of the secret key can only decrypt if he satisfies certain properties. This allows for fine-grained access control.

There are a certain number of attributes that each key either satisfies or does not satisfy. We can encrypt messages and require that it should be decryptable only by such keys the attributes of which satisfy certain logical statements of the arguments.

ABE was proposed by Goyal, Pandey et al. in 2006 [GPSW06].

4.5 Deniable Encryption

In [CDNO97] Canetti et al. introduced the concept of (sender) deniable encryption. It is a special encryption that allows the sender to generate fake randomness keys as evidence that a given ciphertexts is the encryption of a given plaintext. The original construction of Canetti et al. had the limitation of pre-planning in the sense that at encryption time it is required to choose either the standard encryption function or the deniable encryption function. This has been more recently improved by Sahai and Waters [SW14] that through the use of indistinguishable obfuscation showed how to perform deniable encryption without pre-planning. This was the first scheme with super-polynomial security, i.e., where an adversary has negligible advantage in distinguishing real and fake openings. In [Dac14] it is proven that there is no black-box construction of (sender) deniable public-key encryption with super-polynomial security from simulatable public-key encryption. This impossibility result shows that any construction should employ non-black-box techniques, stronger assumptions (this is the case of the construction of [SW14]), or interaction.

4.6 Witness Encryption

Let L be a language in NP. Witness encryption is a type of encryption where Alice encrypts a value y with respect to a value x. The ciphertext can only be decrypted by the one who manages to provide a witness w_x that witnesses that $x \in L$. If $x \notin L$, then the cyphertext can not be decrypted. Note that Alice does not have to know whether $x \in L$. Currently, there are no practical witness encryption schemes known.

Garg, Gentry et al. proposed the concept of witness encryption in [GGSW13]. They also gave a construction based on multilinear maps for a witness encryption scheme where the NP-problem under question is EXACT COVER. They also give an impossibility result — a statistically sound witness encryption is impossible unless the polynomial hierarchy collapses. The original construct also suffered not being able to rely on simple assumptions — basically, they had to assume that the scheme was secure.

There have been improvements of the original scheme. Gentry et al. [GLW14] propose a method for reducing the assumptions of a WE scheme. They demonstrate that the approach is viable by obtaining an assumption that depends on the length of the witness but otherwise does not depend on the NP-instance.

Also, Ananth et al. [AJN$^+$16] propose *combiners* for WE — a system that takes several constructions of a primitive and combines them and is secure if any of them is secure. Their constructions gives a combiner that is secure provided that one-way functions exist.

As multilinear maps are very expensive, WE based on those would also be expensive. Abusalah et al. [AFP16] propose a method where they add a setup phase, essentially offloading the cost to the setup phase that needs to be done only once. In that case encryption phase needs only two CPA encryptions and a NIZK proof. The setup required is still expensive, but needs to be done only once and can be used for arbitrarily many encryptions. The provider of the setup phase must be trusted though. Also, the decryption phase remains expensive.

Goyal and Goyal proposed using witness encryption in [GG17] in order to build a non-interactive zero-knowledge scheme that needs only blockchain as a trusted third party.

4.7 Private Information Retrieval

An m-out-of-n computationally private information retrieval (shortened to (n, m)-CPIR, [KO97]) protocol enables the receiver to obtain m elements from sender's database of n elements, without the sender getting to know which elements were obtained. An efficient CPIR protocol has to be implemented by virtually any two-party privacy-preserving database application, and hence CPIR protocols have received significant attention in the literature.

Let ℓ be the element length. Since there exists a trivial CPIR protocol with linear communication ℓn where the sender just forwards the whole database to the receiver, a major requirement in the design of new CPIR protocols is their communication efficiency. The first CPIR protocol with sublinear communication was proposed by Kushilevitz and Ostrovsky [KO97], and slightly optimized by Stern [Ste98]. The first CPIR protocol with polylogarithmic-in-n communication was proposed by Cachin, Micali and Stadler [CMS99]. The first CPIR protocols with *asymptotically* truly efficient communication complexity were proposed by Lipmaa [Lip05, Lip09] and Gentry and Ramzan [GR05]. Very recently, Kiayias et al. [KLL$^+$15, LP17] have proposed $(n, 1)$-CPIR protocols with optimal rate, that is, with communication $\ell + o(\ell)$.

Oblivious transfer is a strengthened version of CPIR that additionally requires that the user gain no information about the other items in the database that the user did not ask for. Differently from CPIR, oblivious transfer is also interesting in the case of linear communication (the mentioned trivial protocol where the sender forwards the whole database to the receiver does not preserver the privacy of the sender), in particular since it is known to be complete for multi-party computation [Kil88]. (This is since oblivious transfer can be used to securely evaluate arbitrary functions.) Moreover, oblivious transfer is often useful in the case when the trapdoor only consists of two elements since $(2, 1)$-oblivious transfer can be used together with garbled circuits to implement two-party computation [Yao82].

Chapter 5

Secure computation

Distributed ledgers have by definition multiple parties involved. These parties may not necessarily trust each other. The most basic cryptographic protocols necessary for blockchain —- signing and hashing — require the private data of either no party or just one party and can be verified by anybody. However, if we wish to build more complicated operations on top of the blockchain, we might need operations that take the private input of more than one party. For example, two parties may agree to put cryptocurrency to an account so that neither of those parties can take the currency when operating alone but they can move the currency when they collaborate. For this, a very natural solution is secure (multi-party) computation — a field that studies how to compute on values while preserving their secrecy. Another, related field that is relevant is the field of verifiable computing — other parties might need verification that all operations were performed correctly and this might be more complicated than simply verifying a signature or re-computing a hash.

5.1 Secret Sharing

Secret sharing is a method for taking a piece of data x and obtaining k values $x_1, \ldots x_k$ (called shares) in such a way that x can be learned from some subsets of $\{x_1, \ldots x_k\}$ but no information about x can be learned, if one possesses some other subsets. The values x_i are given to parties P_i.

Common examples are threshold secret sharing schemes where there is an integer t such that possessing t shares allows us to compute x, but possessing $t - 1$ shares gives no information about x. Shamir secret sharing [Sha79] is a popular option, it uses polynomial interpolation.

Secret sharing can be used for storing data in the cases where it is desirable that single parties can not access the data but committees of parties can.

A number of secret-sharing schemes enjoy homomorphic properties — given the sharings of values x and y, it is possible to manipulate the shares to obtain shares of $x + y$, xy or some other function of x and y (which functions can be computed this way and how efficient these operations are depend on the concrete setting). Secret-sharing based secure multi-party computation studies how to do this efficiently and with different security guarantees.

However, this property can be useful in using secret-sharing in other cryptographic protocols as well — for example, two parties may be able to create a key so that neither party knows the key nor can use it but that they can use the key when they collaborate [BDM16].

5.2 Secure (Multi-Party) Computation

Secure computation is a field that studies how to compute functions on values in such a way that the computing party or parties do not learn anything about the values they compute on.

Data is very useful in the modern world and much can be gained when multiple parties use their data together. However, data can also be private in nature. — for example, medical data or business data. To do medical research, data is highly needed, however, a person's medical information is considered private, and anonymisation might not provide enough privacy. (For example, when given the gender, birth date, and educational history, this might be sufficient to single out a person from a dataset even if their name is not given.) Similarly, companies may want to collaborate in a way where each provides some data of their own to achieve some common goal, but do not want to reveal that data as it is a business secret. Thus for the kinds of applications where at least some of the computing parties should not learn anything about the data they process secure computation methods are necessary.

Popular paradigms for secure computation include secret-sharing based approaches (SS), garbled-circuit based approaches (GC), and fully homomorphic encryption (FHE). An interesting property of FHE is that only a single party is needed to do the computation — in other cases we need to make assumptions about at least some sets of parties not collaborating, but as there is only one party in the case of FHE, we do not need any such assumption. However, whereas GC-based and SS-based approaches have been used in the real world, FHE is far too inefficient to be currently of any practical use [MSM17]. Thus, as this deliverable aims to describe the primitives that can be practically used for blockchain, we shall not describe FHE in any more detail.

5.3 Secret Sharing-Based Secure Computation

Some secret sharing schemes have structures that allow to perform different operations on shares. If two values x and y have been shared, then it is possible to manipulate those shares in such a way that the parties would hold shares of $x + y$ or xy — it is possible to add and multiply the shared values, thus theoretically allowing any computable function to be computed. This often includes the parties passing specific messages to each other, although some operations are possible to perform without sending any messages. Sending these messages is usually the bottleneck of this type of secure computation. As a consequence, these kinds of computations benefit very much from parallel composition.

Some variants of SS require at least three parties, while others require only two. Both have their advantages and disadvantages. Secure two-party computation can only be achieved with computational security which is usually slower than information-theoretical security. [DO10]. On the other hand, it can be more difficult to find three parties any two of which would not collaborate than to find two such parties.

Another approach to secret-shared based computation uses the idea of splitting computation into a precomputation phase and online phase. The idea is that it is possible to do some computations before the actual data enters the system. This makes computation faster when the actual data enters the system. This is useful when the application is an infrequent event with lots of data to process (for example an election), not a continuous process where computation needs to be done constantly (for example, keeping satellites from colliding). The original example of this are the Beaver triplets [Bea91] where one party prepares secret-shared multiplicative triples that help with the secret multiplication operation.

5.4 Garbled Circuits

Garbled circuits is a method for two-party computation that was originally proposed by Yao in 1986[Yao86]. Alice takes a circuit C that computes some function f and applies a 'garbling function' to it, obtaining the object C' and sends it to Bob. The nature of C' will be explained later. Bob now obtains the inputs from the Alice via oblivious transfer, enters his own input and evaluates the circuit, without learning anything more about Alice's input than can be learned from the output of C'.

A garbled circuit consists of garbled gates and wires. In an evaluation of a garbled circuit, wires carry wire labels corresponding to binary wire values in such a way that Bob does not learn anything about the wire values, unless they only depend on values known to Bob.

Generally it is done so that Alice picks two values w_0 and w_1 for every wire W that signify the respective bits. Bob should learn only one of them without knowing whether it corresponds to 0 or 1. In the case where these bits are the input bits, this can be easily solved with oblivious transfer.

However, suppose that we want to obtain the value of the wire w_o that leaves a gate, for clarity, let it be an AND-gate. Bob knows the values w_l and w_r — the values of the wires entering the gate.

Bob can learn the value w_o with the technique of the garbled gate. Essentially, there are four possibilities for the pair (w_l, w_r) — $(w_{l_0}, w_{r_0}), (w_{l_0}, w_{r_1}), (w_{l_1}, w_{r_0})$ and (w_{l_1}, w_{r_1}). In the first three cases, Bob should obtain w_{o_0} and in the last case w_{o_1}. Note that while Bob knows w_l and w_r, Bob does not know which of the four possible variants it is.

Thus Alice uses every possible pair of inputs as keys to symmetrically encrypt either w_{o_0} or w_{o_1}, respectively. When the output bit of the gate for the input (l, w) is supposed to be o, then the value that Alice encrypts with (w_l, w_r) is w_o.

In total she obtains four possible ciphertexts.

Now she computes a random permutation π of these four values and sends the values Bob. That is, essentially, a garbled gate.

Bob knows w_l and w_r which is one of the $(w_{l_0}, w_{r_0}), (w_{l_0}, w_{r_1}), (w_{l_1}, w_{r_0})$ and (w_{l_1}, w_{r_1}) — thus, when he tries to use the keys he knows to try to decrypt C_a, C_b, C_c and C_d, he succeeds precisely once, obtaining w_o without knowing whether the decrypted value is w_{o_0} or w_{o_1}.

Note that Alice can send all the quadruples (C_a, C_b, C_c, C_d) for every gate to Bob simultaneously. This makes the communication scheme of garbled circuits different from secret-shared based computations — in the first case, the parties need to communicate once, but the amount of data to be sent is very big. In the other case, the amount of data sent during every communication is smaller, but many rounds are needed. Also, in the case of garbled circuits, the amount of computation before and after communication is quite significant, whereas in some flavours of secret-sharing, they can be rather small. This gives the two approaches different strengths and weaknesses.

The method described above is the simple version of garbled circuits. Many optimizations have been proposed and implemented, but the specifics of those are not the goal of this deliverable.

5.5 Verifiable Computation

A verifiable computation scheme [GGP10, BGV11, FG12] allows for a client to outsource the computation of a function to an untrusted server; the server produces a proof of correctness along with the output of the function. The client checks the correctness proof to decide whether the output provided by the server is accepted or not. Let $\mathcal{F}$ be a function. A VC scheme $\mathcal{VC} = (\mathsf{KeyGen}, \mathsf{ProbGen}, \mathsf{Compute}, \mathsf{Verify})$ for function $\mathcal{F}$ consists, in general, of the algorithms described below.

- $(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{KeyGen}(\mathcal{F}, \lambda)$: The (randomized) key generation algorithm takes as input the function $\mathcal{F}$ and the security parameter λ, and outputs a public key pk and a secret key sk.

- $(\Sigma_X, \mathsf{vk}_X) \leftarrow \mathsf{ProbGen}_{\mathsf{sk}}(X)$: The (randomized) problem generation algorithm takes as input the value X and uses the secret key sk to compute an encoding Σ_X of X and a *secret* verification key vk_X.

- $\Sigma_Y \leftarrow \mathsf{Compute}_{\mathsf{pk}}(\Sigma_X)$: The (deterministic) compute algorithm takes as input the encoded value Σ_X and uses the public key pk to compute an encoding of $Y = \mathcal{F}(X)$.

- $Y \leftarrow \mathsf{Verify}_{\mathsf{sk}}(\mathsf{vk}_X, \Sigma_Y)$: The (deterministic) verify algorithm takes as input the verification key vk_X and the value Σ_Y; it uses the secret key sk and vk_X to compute a value $Y \in \{0,1\}^* \cup \{\perp\}$, where symbol $\perp$ denotes that the algorithm rejects the value Σ_Y.

A typical VC scheme needs to satisfy some properties that we informally discuss below.

- Correctness: The ProbGen algorithm produces problem instances that allow for a honest server to successfully compute a value Σ_Y such that $Y = \mathcal{F}(X)$.

- Soundness: No malicious server can “trick” a client into accepting an incorrect output, i.e, some value Y such that $Y \neq \mathcal{F}(X)$. We require this to hold even in the presence of so-called verification queries [FGP14].
- Outsourceability: The time to encode the input plus the time to run a verification is smaller than the time to compute the function itself.

The notion of verifiable computation combines well with smart contracts. In general one could delegate the verification process to a smart contract in order to start a transaction to pay for the computation.

Chapter 6

Zero Knowledge

The concept of zero knowledge [GMR85] refers to the capability of a user (the prover) to convince another user (the verifier) that some claim is true, without revealing any side information. and has been defined in several different ways. This is a fundamental privacy-preserving primitive for the constructions of privacy-enhancing ledgers since it allows to upload private data in ledger allowing later to prove statements about encrypted data without revealing any unnecessary private information. It has been defined with different flavors in order to capture different real-world scenarios and we discuss the main variants below.

6.1 Zero-Knowledge Proofs and Argument Systems

A zero-knowledge proof system guarantees the verifier that even an unbounded adversarial prover can not convince the verifier of a false claim with non-negligible probability. Instead the notion of zero-knowledge argument limits the security of the verifier to the fact that the prover can not break some computational assumption, and therefore applies only to computationally bounded (i.e., polynomial time) provers. Zero-knowledge proof and argument systems enable parties in a protocol to convince other users that they honestly followed the protocol without revealing their private data. E.g., in the case of electronic voting, a voter can prove that his encrypted ballot is for a registered candidate without revealing which candidate precisely he voted for.

6.1.1 Timing

A zero-knowledge protocol allows a prover to convince a verifier about the validity of an NP statement without providing additional knowledge to the verifier. This is formalized by requiring the existence of an efficient simulator S that can simulate the view of a malicious verifier interacting with the honest prover P. The notion of concurrent ZK (cZK), introduced in [DNS04], considers an adversary mounting a coordinated attack by acting as a verifier in many concurrent sessions and asking to receive proofs from multiple provers. cZK protocols are significantly harder to construct and analyze, and are often less efficient than the standalone ZK protocols [PTV10]. The difficulties in constructing and proving the security of a protocol in this setting is due to the message schedule that a malicious verifier could adopt. To help the simulator against such a strong malicious verifier, a timing model was introduced in [DNS04]. In this model it is assumed that every party has a local clock, and that all these local clocks are roughly synchronized. Using the clock the prover (and the simulator) can delay the response of certain messages by a given amount of time, thus limiting the power of the malicious verifier. In common ledger implementations, some synchronisation between the parties involved in the protocol is assumed, i.e., there exists a clock that all parties can access.

6.2 Non-Interactive Zero Knowledge

In many practical applications, one prover must convince many verifiers that he followed the protocol correctly. In such cases, one should use non-interactive zero-knowledge proofs (NIZK, [BFM88]), where the proof itself is a bit-string, created once by the prover and then verified independently by anybody who is interested in the correct outcome of the protocol.

It is impossible for a NIZK proof system to simultaneously satisfy its expected security requirements, namely the soundness and zero-knowledge properties, in the so called standard model [BFM88]. Because of that, universally verifiable NIZK proof systems are designed usually either in the common reference string model (the CRS model, [BFM88]) or in the random oracle model (the RO model, [BR93]). Those two models are orthogonal. The RO model is very often favored by practitioners due the availability of well-known heuristics like the Fiat-Shamir heuristic [FS86] that make it possible to construct very efficient RO-model NIZK proof systems for a wide variety of tasks. While it is well-known that the RO model is not always instantiable [CGH98, GK03], no concrete attacks are known against any sensibly designed NIZK proof systems. To avoid possible future attacks against heuristic RO-model NIZK proof systems, it is reasonable to design NIZK proof systems in the CRS model.

The two main drawbacks of the CRS model are

(i) one must trust the entity who has generated the CRS, and

(ii) CRS-model NIZK proof systems have traditionally been inefficient.

Currently, one can say that (ii) is not an issue, at least not in many cases; see Sect. 6.5.

6.3 Zero-Knowledge Proofs and Arguments of Knowledge

A zero-knowledge proof/argument of knowledge is a strengthened notion of a zero-knowledge proof/argument system. The prover proves not only that a claim is true but also that he knows a witness that allows to check in polynomial time that the claim is true, without any interaction [BG92]. This advanced security notion for a verifier is formalized by requiring that there exists an efficient procedure named extractor that having access to the prover outputs the witness except with negligible probability. The difference between proof and argument is again a consequence of the assumed computational power of the adversarial prover, that is unbounded in the former case and efficient (i.e., polynomial time) in the latter.

The witness extraction property is fundamental in several applications where in some cases the claim is true by definition and the only point is to prove possession of a witness (e.g., knowledge of a discrete logarithm, of a private key, of a signature).

Witness extraction is performed using rewinds in case of a black-box extractor but can also be straight-line in case non-black-box extraction is possible [BL02]. In the non-interactive case witness extraction can be straight-line in the CRS model [SP92] while instead it requires rewinds in the random oracle model when the Fiat-Shamir heuristic is used. Using instead a heuristic due to Fischlin [Fis05], one cane have a NIZK proof of knowledge in the random oracle model with straight-line extraction.

6.4 Honest-Verifier Zero Knowledge and Witness Indistinguishability

There exists two weaker security notions for proofs and argument systems, namely: honest-verifier zero-knowledge and witness indistinguishability.

Honest-verifier zero knowledge refers to preserving the privacy of the input of the prover only w.r.t. a verifier that follows the protocol honestly. The adversary is in this case a distinguisher that studies the transcript of an interaction in order to extract some private information about the input of the prover.

Witness indistinguishability refers to hiding which witness is used by prover out of the several existing witness for proving that a certain claim is true. This notion has been introduced by Feige and Shamir in [FS90] and has been proved in [FLS90] to be a major building block for the design obtaining zero knowledge. This last construction introduced the so called *FLS paradigm* that allows to obtain zero knowledge by artificially creating another statement that is false during the real execution, but true during the simulation.

claims is true.

The most practical constructions of zero-knowledge proofs/arguments are based on Σ protocols. Every such protocol consists of 3 messages where the message of the verifier is a random string. They are honest-verifier zero-knowledge proofs of knowledge. The reason why Σ protocols are very popular in cryptography is twofold. First of all, there exist efficient constructions for several languages of practical relevance (e.g., proving that a triple is a Diffie-Hellman tuple). Second, the existence of the Fiat-Shamir heuristic [FS86] that transforms such protocol to non-interactive zero-knowledge arguments of knowledge. The reason why this is just a heuristic is that it relies on the assumption that the output of a collision-resistant hash function can be considered random in a security proof. Finally in [CDS94] and later on in [CPS$^+$16a, CPS$^+$16b] it has been shown how to obtain witness indistinguishability from Σ-protocols.

6.5 SNARKs

A succinct non-interactive argument of knowledge (SNARK) is a non-interactive argument of knowledge where the argument π is succinct. More precisely, it is required that the argument length is $\mathsf{poly}(\lambda)\,(|x| + |w|)^{o(1)}$, where $|x|$ is the input length and $|w|$ is the witness length, [GW11]. As shown in [GW11], SNARKs exist only under very strong ("non-falsifiable") assumptions.

Recently, many pairing-based (zero-knowledge) SNARKs (zk-SNARKs) in the CRS model have been proposed. In most of such SNARKs (see, e.g., [Gro10, Lip12, GGPR13, Lip13, DFGK14, Gro16]), the verifier's computation is dominated by a small number of exponentiations and pairings in a bilinear group, while the argument consists of a small number of group elements. Importantly, such SNARKs have a general-purpose feature: it is enough to construct a secure and efficient SNARK once; after that, one is only left to design an application-specific arithmetic circuit. If the functionality changes, one only has to redesign the arithmetic circuit (and the CRS) but not the SNARK. Because of the mentioned benefits, SNARKs have been implemented in contexts like verifiable computation [PGHR13] and, perhaps most importantly, cryptocurrencies [BCG$^+$14b].

However, one drawback in the mentioned pairing-based SNARKs is their reliance on the CRS model (Existing*fully-succinct* SNARKs with a short CRS, see e.g. [BCCT13, BCTV14], are impractical.) where all parties of the protocol have to trust that the CRS generator is honest. This is especially troublesome since (usually) one has to generate a new CRS each time the functionality changes. Reducing such trust has been a long-standing open question. Several different approaches for this are known, but each one has its own problems. The Registered Public Key (RPK, [BCNP04]) model is a weaker trust model where each party P_i has her own trusted authority $\mathcal{R}_i$ that registers her public key. While it is known how to construct NIZK arguments in the RPK model, [BCNP04], in the (at least, standard) RPK model the arguments are not transferable since either the malicious verifier or her key authority knows the simulation trapdoor. Thus, a third party does not know if the argument was created by the prover or the designated verifier (or, her key authority). Moreover, existing NIZK arguments in the RPK model are not efficient, and in particular, no pairing-based SNARKs (even designated-verifier ones) are known to exist in the RPK model.

Currently zk-SNARKs are one of the most promising (or at least, best known) existing approaches to efficiently solve some of the privacy issues surrounding cryptocurrency ledgers. In particular, Zerocash [BCG$^+$14a] seems to be quite efficient.

While RO-model NIZK proof systems are often much more efficient than CRS-model NIZK proof systems, very little work has been done in RO-model zk-SNARKs (see, e.g., [GK15, BCC$^+$16]).

The first problem of the CRS-model (the need to trust the generator of CRS) has recently been studied by several groups. One well-known approach here is to use multi-party computation, but this has been only recently made practical [BCG$^+$15]. Another approach is to construct NIZK proof systems so that at least some of the security guarantees hold even if the CRS has been subverted [BFS16]. This approach sounds very promising for future research, see, e.g., [ABLZ17, Fuc18].

Chapter 7

Open problems

Zero-Knowledge SNARKs The first open problem related to zkSNARKs we would like to deal with is to minimize the trust in the CRS creator for pairing-based zk-SNARKs (implemented say in, e.g. Zerocash, [BCG$^+$14a]). (Cf. [ABLZ17].) and study zk-SNARKs in different cryptographic settings, especially to see if one can avoid complete trust in the CRS creator and/or get better efficiency.

Another open problem is to construct cryptographic primitives that are SNARK-friendly, i.e., that can be implemented efficiently by using a SNARK. For example, construct commitment schemes and hash functions that have a small multiplication complexity.

Furthermore, one should study quantum-secure SNARKs, with the aim to obtain security even in the case a quantum computer is constructed.

Functional Encryption and Witness Encryption The first open problem to deal with would be to construct a functional encryption and attribute-based encryption scheme for stronger security notions. Stronger notions of functional encryption (FE) allow a more fine-grained access to encryption data. A multi-input secret key FE scheme, for example, allow different parties to encrypt data. Later on another party, given a functional key for a function f, can compute the output of f on such encrypted data. In a similar way the notion of attribute-based encryption (ABE) can be extended to the multi-input case. It is an interesting problem to construct efficient FE and ABE in the multi-input setting already for the two-input case.

Another task could focus on improving the efficiency/assumption of Witness Encryption (WE). Constructing efficient WE scheme without relying on setup represents an interesting problem for the blockchain context. A recent work [BJK$^+$17] goes into the direction of constructing WE under more standard assumptions (LWE). The drawback of this scheme is that the encryption procedure takes sub-exponential time. It could be interesting to study what is the best balance between efficiency and cryptographic assumption that one can achieve in the context of WE.

Multi-Party Computation An important open problem related to MPC and DLT is to verify whether secure multi-party computation may be used to reduce the amount of trust needed in trusted set up (CRS generation for zk-SNARKs)? To do this verifiable less efficient proof techniques that do not require the same kind of set up may be employed.

Another open problem is to design a system in which the CRS is regularly refreshed, i.e., the set up for zk-SNARKs is regularly repeated in such a way that a compromised CRS cannot be exploited to alter history, nor affect the security of the system after it expired.

Authentication and Credentials The first open problem in this area is to construct authenticated data type without trusted setup. The generic authenticated data type in [CGPT17] requires trusted setup. For use in the DLT setting, a construction without this requirement will be needed.

Next one would be to devise efficient implementations of authenticated data types for specific applications relevant to distributed ledger technology, including payments, asset transfers, financial clearance, netting and so on.

Furthermore, one could improve anonymous credential systems ([CDD17]) by including functionalities like key life-cycle management, revocation, and support for auditable tokens.

Last but not least, an improvement in compositional treatment of practical protocols would be appreciated. Protocols in the area of DLT are often composed of different schemes such as NIZKs, signatures, commitment, and so forth. This calls for a compositional treatment, but at the moment *practical* protocols do not quite fit into the existing *compositional* frameworks.

Generally, work on NIZKs and zkSNARKs targeted at efficiency and post-quantum assumptions would be of interest; we expect that this also holds for other partners.

Non-Interactive Computation An important open problem in a field of non-interactive computation is to find useful models where Non-Interactive Secure Computation (NISC) can be achieved. The work of Ishai et al. [IKO$^+$11] gave the first solution for NISC assuming that parties have access to the oblivious transfer functionality. Subsequently, efficient solutions for NISC based on cut-and-choose techniques were investigated in the common reference string (CRS) model. [AMPR14, MR17] the global random oracle model [CJS14], as well as the plain model with super-polynomial-time simulation [BGI$^+$17]. An interesting open question is to find other interesting models where NISC can be achieved.

For NIZK, an interesting task is to design Non-Interactive Zero-Knowledge proofs avoiding trusted parties (and thus avoiding the CRS). Efficient NIZK proofs have been constructed under somewhat unsatisfying assumptions, involving a CRS or using heuristic security. An interesting open question is to design efficient NIZK proofs avoiding trusted parties (and thus avoiding the CRS) and at the same time reducing the heuristic security.

In case of Sigma-protocols, further investigate CDS-OR and CPSSV techniques in zero-knowledge proofs is necessary. The CDS-OR technique [CDS94] allows to compose sigma protocols efficiently and unconditionally but requiring the statements to be known already in the first round. The recent work of [CPS$^+$16a] allows a partial knowledge in the last round, still unconditionally. The [CPS$^+$16b] showed how to postpone knowledge of all theorems to the last round obtaining better performance in the online-offline case, however assuming computational assumptions (e.g., DDH). An interesting open question is the possibility or impossibility of obtaining the best of both worlds, having all theorems postponed to the last round without requiring computational assumptions.

Authenticated Data Types An important open problem in this area is to provide a mechanism for tracking key states where it's possible to verify the current state of each key and to prove either for a single key or for the whole set that all state updates have been in agreement with a predefind state transtition graph. In most practical use cases the states could be integers from a predefined range — for example 0 for "enrolled" and 1 for "revoked" — and valid transitions always increase the value, but the more general case of arbitrary finite state machines is also interesting.

Bibliography

[ABLZ17] Behzad Abdolmaleki, Karim Baghery, Helger Lipmaa, and Michal Zajac. A subversion-resistant SNARK. In Tsuyoshi Takagi and Thomas Peyrin, editors, *Advances in Cryptology - ASIACRYPT 2017 - 23rd International Conference on the Theory and Applications of Cryptology and Information Security, Hong Kong, China, December 3-7, 2017, Proceedings, Part III*, volume 10626 of *Lecture Notes in Computer Science*, pages 3–33. Springer, 2017. 6.5, 7

[ACJT07] Lars Arge, Christian Cachin, Tomasz Jurdzinski, and Andrzej Tarlecki, editors. *Automata, Languages and Programming, 34th International Colloquium, ICALP 2007, Wroclaw, Poland, July 9-13, 2007, Proceedings*, volume 4596 of *Lecture Notes in Computer Science*. Springer, 2007. 7

[AFP16] Hamza Abusalah, Georg Fuchsbauer, and Krzysztof Pietrzak. Offline witness encryption. In Manulis et al. [MSS16], pages 285–303. 4.6

[AGT01] Aris Anagnostopoulos, Michael T. Goodrich, and Roberto Tamassia. Persistent authenticated dictionaries and their applications. In George I. Davida and Yair Frankel, editors, *Information Security, 4th International Conference, ISC 2001, Malaga, Spain, October 1-3, 2001, Proceedings*, volume 2200 of *Lecture Notes in Computer Science*, pages 379–393. Springer, 2001. 3.6

[AJN$^+$16] Prabhanjan Ananth, Aayush Jain, Moni Naor, Amit Sahai, and Eylon Yogev. Universal constructions and robust combiners for indistinguishability obfuscation and witness encryption. In Matthew Robshaw and Jonathan Katz, editors, *Advances in Cryptology - CRYPTO 2016 - 36th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 14-18, 2016, Proceedings, Part II*, volume 9815 of *Lecture Notes in Computer Science*, pages 491–520. Springer, 2016. 4.6

[AMPR14] Arash Afshar, Payman Mohassel, Benny Pinkas, and Ben Riva. Non-interactive secure computation based on cut-and-choose. In Phong Q. Nguyen and Elisabeth Oswald, editors, *Advances in Cryptology - EUROCRYPT 2014 - 33rd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Copenhagen, Denmark, May 11-15, 2014. Proceedings*, volume 8441 of *Lecture Notes in Computer Science*, pages 387–404. Springer, 2014. 7

[AMVA17] Giuseppe Ateniese, Bernardo Magri, Daniele Venturi, and Ewerton R. Andrade. Redactable blockchain - or - rewriting history in bitcoin and friends. In *2017 IEEE European Symposium on Security and Privacy, EuroS&P 2017, Paris, France, April 26-28, 2017*, pages 111–126. IEEE, 2017. 3.8

[ATW96] N. Asokan, Gene Tsudik, and Michael Waidner. Server-supported signatures. In Elisa Bertino, Helmut Kurth, Giancarlo Martella, and Emilio Montolivo, editors, *Computer Security - ESORICS 96, 4th European Symposium on Research in Computer Security, Rome, Italy, September 25-27, 1996, Proceedings*, volume 1146 of *Lecture Notes in Computer Science*, pages 131–143. Springer, 1996. 3.2.1

[AYL14] Gail-Joon Ahn, Moti Yung, and Ninghui Li, editors. *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security, Scottsdale, AZ, USA, November 3-7, 2014*. ACM, 2014. 7

[BB04] Kemal Bicakci and Nazife Baykal. Server assisted signatures revisited. In Tatsuaki Okamoto, editor, *Topics in Cryptology - CT-RSA 2004, The Cryptographers' Track at the RSA Conference 2004, San Francisco, CA, USA, February 23-27, 2004, Proceedings*, volume 2964 of *Lecture Notes in Computer Science*, pages 143–156. Springer, 2004. 3.2.1

[BCC$^+$15] Jonathan Bootle, Andrea Cerulli, Pyrros Chaidos, Essam Ghadafi, Jens Groth, and Christophe Petit. Short accountable ring signatures based on DDH. In Günther Pernul, Peter Y. A. Ryan, and Edgar R. Weippl, editors, *Computer Security - ESORICS 2015 - 20th European Symposium on Research in Computer Security, Vienna, Austria, September 21-25, 2015, Proceedings, Part I*, volume 9326 of *Lecture Notes in Computer Science*, pages 243–265. Springer, 2015. 3.2.3

[BCC$^+$16] Jonathan Bootle, Andrea Cerulli, Pyrros Chaidos, Jens Groth, and Christophe Petit. Efficient zero-knowledge arguments for arithmetic circuits in the discrete log setting. In Fischlin and Coron [FC16], pages 327–357. 6.5

[BCCT13] Nir Bitansky, Ran Canetti, Alessandro Chiesa, and Eran Tromer. Recursive Composition and Bootstrapping for SNARKs and Proof-Carrying Data. In Dan Boneh, Tim Roughgarden, and Joan Feigenbaum, editors, *STOC 2013*, pages 241–250, Palo Alto, CA, USA, June 1–4, 2013. ACM Press. 6.5

[BCG$^+$14a] 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, SP 2014, Berkeley, CA, USA, May 18-21, 2014*, pages 459–474. IEEE Computer Society, 2014. 3.6, 6.5, 7

[BCG$^+$14b] Eli Ben-Sasson, Alessandro Chiesa, Christina Garman, Matthew Green, Ian Miers, Eran Tromer, and Madars Virza. Zerocash: Decentralized Anonymous Payments from Bitcoin. In *IEEE SP 2014*, pages 459–474, Berkeley, CA, USA, May 18–21, 2014. IEEE Computer Society. 6.5

[BCG$^+$15] 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, SP 2015, San Jose, CA, USA, May 17-21, 2015*, pages 287–304. IEEE Computer Society, 2015. 6.5

[BCH12] Nir Bitansky, Ran Canetti, and Shai Halevi. Leakage-tolerant interactive protocols. In Ronald Cramer, editor, *Theory of Cryptography - 9th Theory of Cryptography Conference, TCC 2012, Taormina, Sicily, Italy, March 19-21, 2012. Proceedings*, volume 7194 of *Lecture Notes in Computer Science*, pages 266–284. Springer, 2012. 2.5.2

[BCJ08] Ali Bagherzandi, Jung Hee Cheon, and Stanislaw Jarecki. Multisignatures secure under the discrete logarithm assumption and a generalized forking lemma. In Ning et al. [NSJ08], pages 449–458. 3.2.2, 3.2.2

[BCNP04] Boaz Barak, Ran Canetti, Jesper Buus Nielsen, and Rafael Pass. Universally Composable Protocols with Relaxed Set-Up Assumptions. In *FOCS 2004*, pages 186–195, Rome, Italy, October, 17–19 2004. IEEE, IEEE Computer Society Press. 2.1, 2.2, 2.2.2, 6.5

[BCTV14] Eli Ben-Sasson, Alessandro Chiesa, Eran Tromer, and Madars Virza. Scalable Zero Knowledge via Cycles of Elliptic Curves. In Juan A. Garay and Rosario Gennaro, editors, *CRYPTO (2) 2014*, volume 8617, pages 276–294, Santa Barbara, California, USA, August 17–21, 2014. Springer, Heidelberg. 6.5

[BDD07] Stefan Brands, Liesje Demuynck, and Bart De Decker. A practical system for globally revoking the unlinkable pseudonyms of unknown users. In Josef Pieprzyk, Hossein Ghodosi, and Ed Dawson, editors, *Information Security and Privacy, 12th Australasian Conference, ACISP 2007, Townsville, Australia, July 2-4, 2007, Proceedings*, volume 4586 of *Lecture Notes in Computer Science*, pages 400–415. Springer, 2007. 3.4

[BdM93] Josh Cohen Benaloh and Michael de Mare. One-way accumulators: A decentralized alternative to digital sinatures (extended abstract). In Tor Helleseth, editor, *Advances in Cryptology - EUROCRYPT '93, Workshop on the Theory and Application of of Cryptographic Techniques, Lofthus, Norway, May 23-27, 1993, Proceedings*, volume 765 of *Lecture Notes in Computer Science*, pages 274–285. Springer, 1993. 3.5

[BDM16] Waclaw Banasik, Stefan Dziembowski, and Daniel Malinowski. Efficient zero-knowledge contingent payments in cryptocurrencies without scripts. In Ioannis G. Askoxylakis, Sotiris Ioannidis, Sokratis K. Katsikas, and Catherine A. Meadows, editors, *Computer Security - ESORICS 2016 - 21st European Symposium on Research in Computer Security, Heraklion, Greece, September 26-30, 2016, Proceedings, Part II*, volume 9879 of *Lecture Notes in Computer Science*, pages 261–280. Springer, 2016. 5.1

[BDN18] Dan Boneh, Manu Drijvers, and Gregory Neven. Compact multi-signatures for smaller blockchains. Cryptology ePrint Archive, Report 2018/483, 2018. `https://eprint.iacr.org/2018/483`. 3.2.2, 3.1, 3.2.2

[Bea91] Donald Beaver. Efficient multiparty protocols using circuit randomization. In Joan Feigenbaum, editor, *Advances in Cryptology - CRYPTO '91, 11th Annual International Cryptology Conference, Santa Barbara, California, USA, August 11-15, 1991, Proceedings*, volume 576 of *Lecture Notes in Computer Science*, pages 420–432. Springer, 1991. 5.3

[BF01] Dan Boneh and Matthew K. Franklin. Identity-based encryption from the weil pairing. In Kilian [Kil01], pages 213–229. 4.3

[BFM88] Manuel Blum, Paul Feldman, and Silvio Micali. Non-interactive zero-knowledge and its applications (extended abstract). In Simon [Sim88], pages 103–112. 2.2.1, 6.2

[BFS16] 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, *Advances in Cryptology - ASIACRYPT 2016 - 22nd International Conference on the Theory and Application of Cryptology and Information Security, Hanoi, Vietnam, December 4-8, 2016, Proceedings, Part II*, volume 10032 of *Lecture Notes in Computer Science*, pages 777–804, 2016. 6.5

[BG92] Mihir Bellare and Oded Goldreich. On defining proofs of knowledge. In Ernest F. Brickell, editor, *Advances in Cryptology - CRYPTO '92, 12th Annual International Cryptology Conference, Santa Barbara, California, USA, August 16-20, 1992, Proceedings*, volume 740 of *Lecture Notes in Computer Science*, pages 390–420. Springer, 1992. 6.3

[BGGK17] Dan Boneh, Rosario Gennaro, Steven Goldfeder, and Sam Kim. A lattice-based universal thresholdizer for cryptographic systems. Cryptology ePrint Archive, Report 2017/251, 2017. `https://eprint.iacr.org/2017/251`. 3.2.3

[BGI$^+$17] Saikrishna Badrinarayanan, Sanjam Garg, Yuval Ishai, Amit Sahai, and Akshay Wadia. Two-message witness indistinguishability and secure computation in the plain model from new assumptions. In Tsuyoshi Takagi and Thomas Peyrin, editors, *Advances in Cryptology - ASIACRYPT 2017 - 23rd International Conference on the Theory and Applications of Cryptology and Information Security, Hong Kong, China, December 3-7, 2017, Proceedings, Part III*, volume 10626 of *Lecture Notes in Computer Science*, pages 275–303. Springer, 2017. 7

[BGK$^+$18] Christian Badertscher, Peter Gazi, Aggelos Kiayias, Alexander Russell, and Vassilis Zikas. Ouroboros genesis: Composable proof-of-stake blockchains with dynamic availability. *IACR Cryptology ePrint Archive*, 2018:378, 2018. 3.2.3, 3.3

[BGLS03] Dan Boneh, Craig Gentry, Ben Lynn, and Hovav Shacham. Aggregate and verifiably encrypted signatures from bilinear maps. In Eli Biham, editor, *Advances in Cryptology - EUROCRYPT 2003, International Conference on the Theory and Applications of Cryptographic Techniques, Warsaw, Poland, May 4-8, 2003, Proceedings*, volume 2656 of *Lecture Notes in Computer Science*, pages 416–432. Springer, 2003. 3.2.2, 3.2.2, 3.2.2, 3.2.2

[BGR12] Kyle Brogle, Sharon Goldberg, and Leonid Reyzin. Sequential aggregate signatures with lazy verification from trapdoor permutations - (extended abstract). In Xiaoyun Wang and Kazue Sako, editors, *Advances in Cryptology - ASIACRYPT 2012 - 18th International Conference on the Theory and Application of Cryptology and Information Security, Beijing, China, December 2-6, 2012. Proceedings*, volume 7658 of *Lecture Notes in Computer Science*, pages 644–662. Springer, 2012. 3.2.2, 3.2, 3.2.2

[BGV11] Siavosh Benabbas, Rosario Gennaro, and Yevgeniy Vahlis. Verifiable delegation of computation over large datasets. In Phillip Rogaway, editor, *Advances in Cryptology - CRYPTO 2011 - 31st Annual Cryptology Conference, Santa Barbara, CA, USA, August 14-18, 2011. Proceedings*, volume 6841 of *Lecture Notes in Computer Science*, pages 111–131. Springer, 2011. 5.5

[bit] `https://www.bbc.com/news/technology-43485572`. Online; accessed 30-June-2018. 3.8

[BJ08] Ali Bagherzandi and Stanislaw Jarecki. Multisignatures using proofs of secret key possession, as secure as the diffie-hellman problem. In Rafail Ostrovsky, Roberto De Prisco, and Ivan Visconti, editors, *Security and Cryptography for Networks, 6th International Conference, SCN 2008, Amalfi, Italy, September 10-12, 2008. Proceedings*, volume 5229 of *Lecture Notes in Computer Science*, pages 218–235. Springer, 2008. 3.2.2, 3.2.2

[BJK$^+$17] Zvika Brakerski, Aayush Jain, Ilan Komargodski, Alain Passelègue, and Daniel Wichs. Non-trivial witness encryption and null-io from standard assumptions. *IACR Cryptology ePrint Archive*, 2017:874, 2017. 7

[BJKO17] Ahto Buldas, Aivo Jürgenson, Aivo Kalu, and Mart Oruaas. Server-supported RSA signatures for mobile devices. In Simon N. Foley, Dieter Gollmann, and Einar Snekkenes, editors, *Computer Security - ESORICS 2017 - 22nd European Symposium on Research in Computer Security, Oslo, Norway, September 11-15, 2017, Proceedings, Part I*, volume 10492 of *Lecture Notes in Computer Science*, pages 315–333. Springer, 2017. 3.2.1

[BKM09] Adam Bender, Jonathan Katz, and Ruggero Morselli. Ring signatures: Stronger definitions, and constructions without random oracles. *J. Cryptology*, 22(1):114–138, 2009. 3.2.3

[BL02] Boaz Barak and Yehuda Lindell. Strict polynomial-time in simulation and extraction. In John H. Reif, editor, *Proceedings on 34th Annual ACM Symposium on Theory of Computing, May 19-21, 2002, Montréal, Québec, Canada*, pages 484–493. ACM, 2002. 6.3

[BLLW98] Ahto Buldas, Peeter Laud, Helger Lipmaa, and Jan Willemson. Time-stamping with binary linking schemes. In Hugo Krawczyk, editor, *Advances in Cryptology - CRYPTO '98, 18th Annual International Cryptology Conference, Santa Barbara, California, USA, August 23-27, 1998, Proceedings*, volume 1462 of *Lecture Notes in Computer Science*, pages 486–501. Springer, 1998. 3.5

[BLO18] Carsten Baum, Huang Lin, , and Sabine Oechsner. Towards practical lattice-based one-time linkable ring signatures. Cryptology ePrint Archive, Report 2018/107, 2018. `https://eprint.iacr.org/2018/107`. 3.2.3

[BLS04] Dan Boneh, Ben Lynn, and Hovav Shacham. Short signatures from the weil pairing. *J. Cryptology*, 17(4):297–319, 2004. 3.2.3

[BLT17] Ahto Buldas, Risto Laanoja, and Ahto Truu. A server-assisted hash-based signature scheme. In Helger Lipmaa, Aikaterini Mitrokotsa, and Raimundas Matulevicius, editors, *Secure IT Systems - 22nd Nordic Conference, NordSec 2017, Tartu, Estonia, November 8-10, 2017, Proceedings*, volume 10674 of *Lecture Notes in Computer Science*, pages 3–17. Springer, 2017. 3.2.1

[BM99] Mihir Bellare and Sara K. Miner. A forward-secure digital signature scheme. In Michael J. Wiener, editor, *Advances in Cryptology - CRYPTO '99, 19th Annual International Cryptology Conference, Santa Barbara, California, USA, August 15-19, 1999, Proceedings*, volume 1666 of *Lecture Notes in Computer Science*, pages 431–448. Springer, 1999. 3.2.3

[BN06] Mihir Bellare and Gregory Neven. Multi-signatures in the plain public-key model and a general forking lemma. In Juels et al. [JWdV06], pages 390–399. 3.2.2, 3.2.2

[BNN07] Mihir Bellare, Chanathip Namprempre, and Gregory Neven. Unrestricted aggregate signatures. In Arge et al. [ACJT07], pages 411–422. 3.2.2, 3.2.2, 3.2.2

[Bol03] Alexandra Boldyreva. Threshold signatures, multisignatures and blind signatures based on the gap-diffie-hellman-group signature scheme. In Yvo Desmedt, editor, *Public Key Cryptography - PKC 2003, 6th International Workshop on Theory and Practice in Public Key Cryptography, Miami, FL, USA, January 6-8, 2003, Proceedings*, volume 2567 of *Lecture Notes in Computer Science*, pages 31–46. Springer, 2003. 3.2.2, 3.2.3

[BR93] Mihir Bellare and Phillip Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In Dorothy E. Denning, Raymond Pyle, Ravi Ganesan, Ravi S. Sandhu, and Victoria Ashby, editors, *CCS '93, Proceedings of the 1st ACM Conference on Computer and Communications Security, Fairfax, Virginia, USA, November 3-5, 1993.*, pages 62–73. ACM, 1993. 2.3, 6.2

[Bra90] Gilles Brassard, editor. *Advances in Cryptology - CRYPTO '89, 9th Annual International Cryptology Conference, Santa Barbara, California, USA, August 20-24, 1989, Proceedings*, volume 435 of *Lecture Notes in Computer Science*. Springer, 1990. 7

[BSS02] Emmanuel Bresson, Jacques Stern, and Michael Szydlo. Threshold ring signatures and applications to ad-hoc groups. In Yung [Yun02], pages 465–480. 3.2.3

[BSW11] Dan Boneh, Amit Sahai, and Brent Waters. Functional encryption: Definitions and challenges. In Yuval Ishai, editor, *Theory of Cryptography - 8th Theory of Cryptography Conference, TCC 2011, Providence, RI, USA, March 28-30, 2011. Proceedings*, volume 6597 of *Lecture Notes in Computer Science*, pages 253–273. Springer, 2011. 4.2

[BSW12] Dan Boneh, Amit Sahai, and Brent Waters. Functional encryption: a new vision for public-key cryptography. *Commun. ACM*, 55(11):56–64, 2012. 4.2

[Can01] Ran Canetti. Universally Composable Security: A New Paradigm for Cryptographic Protocols. In *FOCS 2001*, pages 136–145, Las Vegas, Nevada, USA, 14–17 October 2001. IEEE, IEEE Computer Society Press. 2.4

[CC04] Christian Cachin and Jan Camenisch, editors. *Advances in Cryptology - EUROCRYPT 2004, International Conference on the Theory and Applications of Cryptographic Techniques, Interlaken, Switzerland, May 2-6, 2004, Proceedings*, volume 3027 of *Lecture Notes in Computer Science*. Springer, 2004. 7

[CDD17] Jan Camenisch, Manu Drijvers, and Maria Dubovitskaya. Practical uc-secure delegatable credentials with attributes and their application to blockchain. In Bhavani M. Thuraisingham, David Evans, Tal Malkin, and Dongyan Xu, editors, *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security, CCS 2017, Dallas, TX, USA, October 30 - November 03, 2017*, pages 683–699. ACM, 2017. 7

[CDG$^+$18] Jan Camenisch, Manu Drijvers, Tommaso Gagliardoni, Anja Lehmann, and Gregory Neven. The wonderful world of global random oracles. In Jesper Buus Nielsen and Vincent Rijmen, editors, *Advances in Cryptology - EUROCRYPT 2018 - 37th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Tel Aviv, Israel, April 29 - May 3, 2018 Proceedings, Part I*, volume 10820 of *Lecture Notes in Computer Science*, pages 280–312. Springer, 2018. 2.3

[CDK$^+$17] Jan Camenisch, David Derler, Stephan Krenn, Henrich C. Pöhls, Kai Samelin, and Daniel Slamanig. Chameleon-hashes with ephemeral trapdoors - and applications to invisible sanitizable signatures. In Serge Fehr, editor, *Public-Key Cryptography - PKC 2017 - 20th IACR International Conference on Practice and Theory in Public-Key Cryptography, Amsterdam, The Netherlands, March 28-31, 2017, Proceedings, Part II*, volume 10175 of *Lecture Notes in Computer Science*, pages 152–182. Springer, 2017. 3.8

[CDNO97] Ran Canetti, Cynthia Dwork, Moni Naor, and Rafail Ostrovsky. Deniable encryption. In Burton S. Kaliski Jr., editor, *Advances in Cryptology - CRYPTO '97, 17th Annual International Cryptology Conference, Santa Barbara, California, USA, August 17-21, 1997, Proceedings*, volume 1294 of *Lecture Notes in Computer Science*, pages 90–104. Springer, 1997. 4.5

[CDS94] Ronald Cramer, Ivan Damgård, and Berry Schoenmakers. Proofs of partial knowledge and simplified design of witness hiding protocols. In Yvo Desmedt, editor, *Advances in Cryptology - CRYPTO '94, 14th Annual International Cryptology Conference, Santa Barbara, California, USA, August 21-25, 1994, Proceedings*, volume 839 of *Lecture Notes in Computer Science*, pages 174–187. Springer, 1994. 3.2.3, 3.7, 6.4, 7

[CGGM00] Ran Canetti, Oded Goldreich, Shafi Goldwasser, and Silvio Micali. Resettable zero-knowledge (extended abstract). In F. Frances Yao and Eugene M. Luks, editors, *Proceedings of the Thirty-Second Annual ACM Symposium on Theory of Computing, May 21-23, 2000, Portland, OR, USA*, pages 235–244. ACM, 2000. 2.2.3

[CGH98] Ran Canetti, Oded Goldreich, and Shai Halevi. The random oracle methodology, revisited (preliminary version). In Jeffrey Scott Vitter, editor, *Proceedings of the Thirtieth Annual ACM Symposium on the Theory of Computing, Dallas, Texas, USA, May 23-26, 1998*, pages 209–218. ACM, 1998. 6.2

[CGH04] Ran Canetti, Oded Goldreich, and Shai Halevi. The random oracle methodology, revisited. *J. ACM*, 51(4):557–594, 2004. 2.3

[CGP03] Stelvio Cimato, Clemente Galdi, and Giuseppe Persiano, editors. *Security in Communication Networks, Third International Conference, SCN 2002, Amalfi, Italy, September 11-13, 2002. Revised Papers*, volume 2576 of *Lecture Notes in Computer Science*. Springer, 2003. 7

[CGPT17] Christian Cachin, Esha Ghosh, Dimitrios Papadopoulos, and Björn Tackmann. Stateful multi-client verifiable computation. Cryptology ePrint Archive, Report 2017/901, 2017. `https://eprint.iacr.org/2017/901`. 3.6, 7

[CGS07] Nishanth Chandran, Jens Groth, and Amit Sahai. Ring signatures of sub-linear size without random oracles. In Arge et al. [ACJT07], pages 423–434. 3.2.3

[CH02] Jan Camenisch and Els Van Herreweghen. Design and implementation of the *idemix* anonymous credential system. In Vijayalakshmi Atluri, editor, *Proceedings of the 9th ACM Conference on Computer and Communications Security, CCS 2002, Washington, DC, USA, November 18-22, 2002*, pages 21–30. ACM, 2002. 3.4

[Cha82] David Chaum. Blind signatures for untraceable payments. In David Chaum, Ronald L. Rivest, and Alan T. Sherman, editors, *Advances in Cryptology: Proceedings of CRYPTO '82, Santa Barbara, California, USA, August 23-25, 1982.*, pages 199–203. Plenum Press, New York, 1982. 3.2.3

[Cha83] David Chaum. Blind signature system. In David Chaum, editor, *Advances in Cryptology, Proceedings of CRYPTO '83, Santa Barbara, California, USA, August 21-24, 1983.*, page 153. Plenum Press, New York, 1983. 3.2.3

[CJMM03] Eric Cronin, Sugih Jamin, Tal Malkin, and Patrick D. McDaniel. On the performance, feasibility, and use of forward-secure signatures. In Sushil Jajodia, Vijayalakshmi Atluri, and Trent Jaeger, editors, *Proceedings of the 10th ACM Conference on Computer and Communications Security, CCS 2003, Washington, DC, USA, October 27-30, 2003*, pages 131–144. ACM, 2003. 3.2.3

[CJS14] Ran Canetti, Abhishek Jain, and Alessandra Scafuro. Practical UC security with a global random oracle. In Ahn et al. [AYL14], pages 597–608. 2.3, 7

[CL01] Jan Camenisch and Anna Lysyanskaya. An efficient system for non-transferable anonymous credentials with optional anonymity revocation. In Pfitzmann [Pfi01], pages 93–118. 3.4

[CL02] Jan Camenisch and Anna Lysyanskaya. A signature scheme with efficient protocols. In Cimato et al. [CGP03], pages 268–289. 3.4

[CL04] Jan Camenisch and Anna Lysyanskaya. Signature schemes and anonymous credentials from bilinear maps. In Matthew K. Franklin, editor, *Advances in Cryptology - CRYPTO 2004, 24th Annual International CryptologyConference, Santa Barbara, California, USA, August 15-19, 2004, Proceedings*, volume 3152 of *Lecture Notes in Computer Science*, pages 56–72. Springer, 2004. 3.4

[CLNS16] Jan Camenisch, Anja Lehmann, Gregory Neven, and Kai Samelin. Virtual smart cards: How to sign with a password and a server. In Vassilis Zikas and Roberto De Prisco, editors, *Security and Cryptography for Networks - 10th International Conference, SCN 2016, Amalfi, Italy, August 31 - September 2, 2016, Proceedings*, volume 9841 of *Lecture Notes in Computer Science*, pages 353–371. Springer, 2016. 3.2.1

[CLOS02] Ran Canetti, Yehuda Lindell, Rafail Ostrovsky, and Amit Sahai. Universally Composable Two-Party and Multi-Party Secure Computation. In John H. Reif, editor, *STOC 2002*, pages 494–503, Montréal, Québec, Canada, May 19–21 2002. ACM Press. 2.4

[CMS99] Christian Cachin, Silvio Micali, and Markus Stadler. Computationally private information retrieval with polylogarithmic communication. In Jacques Stern, editor, *Advances in Cryptology - EUROCRYPT '99, International Conference on the Theory and Application of Cryptographic Techniques, Prague, Czech Republic, May 2-6, 1999, Proceeding*, volume 1592 of *Lecture Notes in Computer Science*, pages 402–414. Springer, 1999. 4.7

[Com14] The NXT Community. Nxt whitepaper. `https://bravenewcoin.com/assets/Whitepapers/NxtWhitepaper-v122-rev4.pdf`, July 2014. 3.3

[CPS$^+$16a] Michele Ciampi, Giuseppe Persiano, Alessandra Scafuro, Luisa Siniscalchi, and Ivan Visconti. Improved or-composition of sigma-protocols. In Eyal Kushilevitz and Tal Malkin, editors, *Theory of Cryptography - 13th International Conference, TCC 2016-A, Tel Aviv, Israel, January 10-13, 2016, Proceedings, Part II*, volume 9563 of *Lecture Notes in Computer Science*, pages 112–141. Springer, 2016. 3.7, 6.4, 7

[CPS$^+$16b] Michele Ciampi, Giuseppe Persiano, Alessandra Scafuro, Luisa Siniscalchi, and Ivan Visconti. Online/offline OR composition of sigma protocols. In Marc Fischlin and Jean-Sébastien Coron, editors, *Advances in Cryptology - EUROCRYPT 2016 - 35th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Vienna, Austria, May 8-12, 2016, Proceedings, Part II*, volume 9666 of *Lecture Notes in Computer Science*, pages 63–92. Springer, 2016. 3.7, 6.4, 7

[CS14] Chris Culnane and Steve A. Schneider. A peered bulletin board for robust use in verifiable voting systems. In *IEEE 27th Computer Security Foundations Symposium, CSF 2014, Vienna, Austria, 19-22 July, 2014*, pages 169–183. IEEE Computer Society, 2014. 3.9

[CSST06] Sébastien Canard, Berry Schoenmakers, Martijn Stam, and Jacques Traoré. List signature schemes. *Discrete Applied Mathematics*, 154(2):189–201, 2006. 3.2.3

[CvH91] David Chaum and Eugène van Heyst. Group signatures. In Davies [Dav91], pages 257–265. 3.2.3

[CWLY06] Sherman S. M. Chow, Victor K.-W. Wei, Joseph K. Liu, and Tsz Hon Yuen. Ring signatures without random oracles. In Ferng-Ching Lin, Der-Tsai Lee, Bao-Shuh Paul Lin, Shiuhpyng Shieh, and Sushil Jajodia, editors, *Proceedings of the 2006 ACM Symposium on Information, Computer and Communications Security, ASIACCS 2006, Taipei, Taiwan, March 21-24, 2006*, pages 297–302. ACM, 2006. 3.2.3

[CZZ$^+$16] Nikos Chondros, Bingsheng Zhang, Thomas Zacharias, Panos Diamantopoulos, Stathis Maneas, Christos Patsonakis, Alex Delis, Aggelos Kiayias, and Mema Roussopoulos. D-DEMOS: A distributed, end-to-end verifiable, internet voting system. In *36th IEEE International Conference on Distributed Computing Systems, ICDCS 2016, Nara, Japan, June 27-30, 2016*, pages 711–720. IEEE Computer Society, 2016. 3.9

[Dac14] Dana Dachman-Soled. On minimal assumptions for sender-deniable public key encryption. In Hugo Krawczyk, editor, *Public-Key Cryptography - PKC 2014 - 17th International Conference on Practice and Theory in Public-Key Cryptography, Buenos Aires, Argentina, March 26-28, 2014. Proceedings*, volume 8383 of *Lecture Notes in Computer Science*, pages 574–591. Springer, 2014. 4.5

[Dav91] Donald W. Davies, editor. *Advances in Cryptology - EUROCRYPT '91, Workshop on the Theory and Application of of Cryptographic Techniques, Brighton, UK, April 8-11, 1991, Proceedings*, volume 547 of *Lecture Notes in Computer Science*. Springer, 1991. 7

[DEFN18] Manu Drijvers, Kasra Edalatnejad, Bryan Ford, and Gregory Neven. Okamoto beats schnorr: On the provable security of multi-signatures. Cryptology ePrint Archive, Report 2018/417, 2018. `https://eprint.iacr.org/2018/417`. 3.2.2

[DF89] Yvo Desmedt and Yair Frankel. Threshold cryptosystems. In Brassard [Bra90], pages 307–315. 3.2.3

[DFGK14] 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, *Advances in Cryptology - ASIACRYPT 2014 - 20th International Conference on the Theory and Application of Cryptology and Information Security, Kaoshiung, Taiwan, R.O.C., December 7-11, 2014. Proceedings, Part I*, volume 8873 of *Lecture Notes in Computer Science*, pages 532–550. Springer, 2014. 6.5

[DGKR18] Bernardo David, Peter Gazi, Aggelos Kiayias, and Alexander Russell. Ouroboros praos: An adaptively-secure, semi-synchronous proof-of-stake blockchain. In *Advances in Cryptology - EUROCRYPT 2018 - 37th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Tel Aviv, Israel, April 29 - May 3, 2018 Proceedings, Part II*, pages 66–98, 2018. 2.3, 3.2.3, 3.3

[DK01] Ivan Damgård and Maciej Koprowski. Practical threshold RSA signatures without a trusted dealer. In Pfitzmann [Pfi01], pages 152–165. 3.2.3

[DKNS04] Yevgeniy Dodis, Aggelos Kiayias, Antonio Nicolosi, and Victor Shoup. Anonymous identification in ad hoc groups. In Cachin and Camenisch [CC04], pages 609–626. 2.5.1, 3.2.3, 3.7

[DNS04] Cynthia Dwork, Moni Naor, and Amit Sahai. Concurrent zero-knowledge. *J. ACM*, 51(6):851–898, 2004. 6.1.1

[DO10] Ivan Damgård and Claudio Orlandi. Multiparty computation for dishonest majority: From passive to active security at low cost. In Tal Rabin, editor, *Advances in Cryptology - CRYPTO 2010, 30th Annual Cryptology Conference, Santa Barbara, CA, USA, August 15-19, 2010. Proceedings*, volume 6223 of *Lecture Notes in Computer Science*, pages 558–576. Springer, 2010. 5.3

[DP08] Stefan Dziembowski and Krzysztof Pietrzak. Leakage-resilient cryptography. In *49th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2008, October 25-28, 2008, Philadelphia, PA, USA*, pages 293–302. IEEE Computer Society, 2008. 2.5.2

[DRS18] David Derler, Sebastian Ramacher, and Daniel Slamanig. Post-quantum zero-knowledge proofs for accumulators with applications to ring signatures from symmetric-key primitives. In Tanja Lange and Rainer Steinwandt, editors, *Post-Quantum Cryptography - 9th International Conference, PQCrypto 2018, Fort Lauderdale, FL, USA, April 9-11, 2018, Proceedings*, volume 10786 of *Lecture Notes in Computer Science*, pages 419–440. Springer, 2018. 3.2.3

[Dub17] Maria Dubovitskaya. Integrate the Identity Mixer technology to support unlinkability for signing transactions (MVP). Hyperledger Jira Item, February 2017. 3.4

[DY05] Yevgeniy Dodis and Aleksandr Yampolskiy. A verifiable random function with short proofs and keys. In Serge Vaudenay, editor, *Public Key Cryptography - PKC 2005, 8th International Workshop on Theory and Practice in Public Key Cryptography, Les Diablerets, Switzerland, January 23-26, 2005, Proceedings*, volume 3386 of *Lecture Notes in Computer Science*, pages 416–431. Springer, 2005. 3.3

[FC16] Marc Fischlin and Jean-Sébastien Coron, editors. *Advances in Cryptology - EUROCRYPT 2016 - 35th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Vienna, Austria, May 8-12, 2016, Proceedings, Part II*, volume 9666 of *Lecture Notes in Computer Science*. Springer, 2016. 7

[FFG$^+$16] Dario Fiore, Cédric Fournet, Esha Ghosh, Markulf Kohlweiss, Olga Ohrimenko, and Bryan Parno. Hash first, argue later: Adaptive verifiable computations on outsourced data. In Edgar R. Weippl, Stefan Katzenbeisser, Christopher Kruegel, Andrew C. Myers, and Shai Halevi, editors, *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security, Vienna, Austria, October 24-28, 2016*, pages 1304–1316. ACM, 2016. 3.6

[FG12] Dario Fiore and Rosario Gennaro. Publicly verifiable delegation of large polynomials and matrix computations, with applications. In Ting Yu, George Danezis, and Virgil D. Gligor, editors, *the ACM Conference on Computer and Communications Security, CCS'12, Raleigh, NC, USA, October 16-18, 2012*, pages 501–512. ACM, 2012. 5.5

[FGP14] Dario Fiore, Rosario Gennaro, and Valerio Pastro. Efficiently verifiable computation on encrypted data. In Ahn et al. [AYL14], pages 844–855. 5.5

[Fis05] Marc Fischlin. Communication-efficient non-interactive proofs of knowledge with online extractors. In Victor Shoup, editor, *Advances in Cryptology - CRYPTO 2005: 25th Annual International Cryptology Conference, Santa Barbara, California, USA, August 14-18, 2005, Proceedings*, volume 3621 of *Lecture Notes in Computer Science*, pages 152–168. Springer, 2005. 6.3

[FLS90] Uriel Feige, Dror Lapidot, and Adi Shamir. Multiple non-interactive zero knowledge proofs based on a single random string (extended abstract). In *31st Annual Symposium on Foundations of Computer Science, St. Louis, Missouri, USA, October 22-24, 1990, Volume I*, pages 308–317. IEEE Computer Society, 1990. 6.4

[FLS12] Marc Fischlin, Anja Lehmann, and Dominique Schröder. History-free sequential aggregate signatures. In Ivan Visconti and Roberto De Prisco, editors, *Security and Cryptography for Networks - 8th International Conference, SCN 2012, Amalfi, Italy, September 5-7, 2012. Proceedings*, volume 7485 of *Lecture Notes in Computer Science*, pages 113–130. Springer, 2012. 3.2.2, 3.2, 3.2.2

[FPS$^+$11] Marc Fischlin, Benny Pinkas, Ahmad-Reza Sadeghi, Thomas Schneider, and Ivan Visconti. Secure set intersection with untrusted hardware tokens. In Aggelos Kiayias, editor, *Topics in Cryptology - CT-RSA 2011 - The Cryptographers' Track at the RSA Conference 2011, San Francisco, CA, USA, February 14-18, 2011. Proceedings*, volume 6558 of *Lecture Notes in Computer Science*, pages 1–16. Springer, 2011. 2.5.1

[FS86] Amos Fiat and Adi Shamir. How to prove yourself: Practical solutions to identification and signature problems. In Andrew M. Odlyzko, editor, *Advances in Cryptology - CRYPTO '86, Santa Barbara, California, USA, 1986, Proceedings*, volume 263 of *Lecture Notes in Computer Science*, pages 186–194. Springer, 1986. 6.2, 6.4

[FS90] Uriel Feige and Adi Shamir. Witness indistinguishable and witness hiding protocols. In Harriet Ortiz, editor, *Proceedings of the 22nd Annual ACM Symposium on Theory of Computing, May 13-17, 1990, Baltimore, Maryland, USA*, pages 416–426. ACM, 1990. 6.4

[FS08] Eiichiro Fujisaki and Koutarou Suzuki. Traceable ring signature. *IEICE Transactions*, 91-A(1):83–93, 2008. 3.2.3

[Fuc18] Georg Fuchsbauer. Subversion-zero-knowledge snarks. In Michel Abdalla and Ricardo Dahab, editors, *Public-Key Cryptography - PKC 2018 - 21st IACR International Conference on Practice and Theory of Public-Key Cryptography, Rio de Janeiro, Brazil, March 25-29, 2018, Proceedings, Part I*, volume 10769 of *Lecture Notes in Computer Science*, pages 315–347. Springer, 2018. 6.5

[Fuj11] Eiichiro Fujisaki. Sub-linear size traceable ring signatures without random oracles. In Aggelos Kiayias, editor, *Topics in Cryptology - CT-RSA 2011 - The Cryptographers' Track at the RSA Conference 2011, San Francisco, CA, USA, February 14-18, 2011. Proceedings*, volume 6558 of *Lecture Notes in Computer Science*, pages 393–415. Springer, 2011. 3.2.3

[GG17] Rishab Goyal and Vipul Goyal. Overcoming cryptographic impossibility results using blockchains. In Yael Kalai and Leonid Reyzin, editors, *Theory of Cryptography - 15th International Conference, TCC 2017, Baltimore, MD, USA, November 12-15, 2017, Proceedings, Part I*, volume 10677 of *Lecture Notes in Computer Science*, pages 529–561. Springer, 2017. 4.6

[GGN16] Rosario Gennaro, Steven Goldfeder, and Arvind Narayanan. Threshold-optimal DSA/ECDSA signatures and an application to bitcoin wallet security. In Manulis et al. [MSS16], pages 156–174. 3.2.3

[GGP10] Rosario Gennaro, Craig Gentry, and Bryan Parno. Non-interactive verifiable computing: Outsourcing computation to untrusted workers. In Tal Rabin, editor, *Advances in Cryptology - CRYPTO 2010, 30th Annual Cryptology Conference, Santa Barbara, CA, USA, August 15-19, 2010. Proceedings*, volume 6223 of *Lecture Notes in Computer Science*, pages 465–482. Springer, 2010. 5.5

[GGPR13] 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, *Advances in Cryptology - EUROCRYPT 2013, 32nd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Athens, Greece, May 26-30, 2013. Proceedings*, volume 7881 of *Lecture Notes in Computer Science*, pages 626–645. Springer, 2013. 6.5

[GGSW13] Sanjam Garg, Craig Gentry, Amit Sahai, and Brent Waters. Witness encryption and its applications. In Dan Boneh, Tim Roughgarden, and Joan Feigenbaum, editors, *Symposium on Theory of Computing Conference, STOC'13, Palo Alto, CA, USA, June 1-4, 2013*, pages 467–476. ACM, 2013. 4.6

[GHKR08] Rosario Gennaro, Shai Halevi, Hugo Krawczyk, and Tal Rabin. Threshold RSA for dynamic and ad-hoc groups. In Smart [Sma08], pages 88–107. 3.2.3

[GJKR96] Rosario Gennaro, Stanislaw Jarecki, Hugo Krawczyk, and Tal Rabin. Robust threshold DSS signatures. In Ueli M. Maurer, editor, *Advances in Cryptology - EUROCRYPT '96, International Conference on the Theory and Application of Cryptographic Techniques, Saragossa, Spain, May 12-16, 1996, Proceeding*, volume 1070 of *Lecture Notes in Computer Science*, pages 354–371. Springer, 1996. 3.2.3

[GJKR07] Rosario Gennaro, Stanislaw Jarecki, Hugo Krawczyk, and Tal Rabin. Secure distributed key generation for discrete-log based cryptosystems. *J. Cryptology*, 20(1):51–83, 2007. 3.2.3

[GK90] Oded Goldreich and Hugo Krawczyk. On the composition of zero-knowledge proof systems. *SIAM Journal on Computing*, 25:169–192, 1990. 2.2

[GK03] Shafi Goldwasser and Yael Tauman Kalai. On the (in)security of the fiat-shamir paradigm. In *44th Symposium on Foundations of Computer Science (FOCS 2003), 11-14 October 2003, Cambridge, MA, USA, Proceedings*, pages 102–113. IEEE Computer Society, 2003. 2.3, 6.2

[GK15] Jens Groth and Markulf Kohlweiss. One-out-of-many proofs: Or how to leak a secret and spend a coin. In Elisabeth Oswald and Marc Fischlin, editors, *Advances in Cryptology - EUROCRYPT 2015 - 34th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Sofia, Bulgaria, April 26-30, 2015, Proceedings, Part II*, volume 9057 of *Lecture Notes in Computer Science*, pages 253–280. Springer, 2015. 6.5

[GKL15] Juan A. Garay, Aggelos Kiayias, and Nikos Leonardos. The bitcoin backbone protocol: Analysis and applications. In *EUROCRYPT (2)*, volume 9057 of *Lecture Notes in Computer Science*, pages 281–310. Springer, 2015. 2.3, 2.4

[GLW14] Craig Gentry, Allison B. Lewko, and Brent Waters. Witness encryption from instance independent assumptions. In Juan A. Garay and Rosario Gennaro, editors, *Advances in Cryptology - CRYPTO 2014 - 34th Annual Cryptology Conference, Santa Barbara, CA, USA, August 17-21, 2014, Proceedings, Part I*, volume 8616 of *Lecture Notes in Computer Science*, pages 426–443. Springer, 2014. 4.6

[GMR85] Shafi Goldwasser, Silvio Micali, and Charles Rackoff. The knowledge complexity of interactive proof-systems (extended abstract). In Robert Sedgewick, editor, *Proceedings of the 17th Annual ACM Symposium on Theory of Computing, May 6-8, 1985, Providence, Rhode Island, USA*, pages 291–304. ACM, 1985. 6

[GOR18] Craig Gentry, Adam O'Neill, and Leonid Reyzin. A unified framework for trapdoor-permutation-based sequential aggregate signatures. In Michel Abdalla and Ricardo Dahab, editors, *Public-Key Cryptography - PKC 2018 - 21st IACR International Conference on Practice and Theory of Public-Key Cryptography, Rio de Janeiro, Brazil, March 25-29, 2018, Proceedings, Part II*, volume 10770 of *Lecture Notes in Computer Science*, pages 34–57. Springer, 2018. 3.2.2, 3.2.2

[Goy04] Vipul Goyal. More efficient server assisted one time signatures. Cryptology ePrint Archive, Report 2004/135, 2004. `https://eprint.iacr.org/2004/135`. 3.2.1

[GPSW06] Vipul Goyal, Omkant Pandey, Amit Sahai, and Brent Waters. Attribute-based encryption for fine-grained access control of encrypted data. In Juels et al. [JWdV06], pages 89–98. 4.4

[GR05] Craig Gentry and Zulfikar Ramzan. Single-database private information retrieval with constant communication rate. In Lu&apos;ıs Caires, Giuseppe F. Italiano, Lu&apos;ıs Monteiro, Catuscia Palamidessi, and Moti Yung, editors, *Automata, Languages and Programming, 32nd International Colloquium, ICALP 2005, Lisbon, Portugal, July 11-15, 2005, Proceedings*, volume 3580 of *Lecture Notes in Computer Science*, pages 803–815. Springer, 2005. 4.7

[Gro96] Lov K. Grover. A fast quantum mechanical algorithm for database search. In Gary L. Miller, editor, *Proceedings of the Twenty-Eighth Annual ACM Symposium on the Theory of Computing, Philadelphia, Pennsylvania, USA, May 22-24, 1996*, pages 212–219. ACM, 1996. 2.7

[Gro10] Jens Groth. Short pairing-based non-interactive zero-knowledge arguments. In Masayuki Abe, editor, *Advances in Cryptology - ASIACRYPT 2010 - 16th International Conference on the Theory and Application of Cryptology and Information Security, Singapore, December 5-9, 2010. Proceedings*, volume 6477 of *Lecture Notes in Computer Science*, pages 321–340. Springer, 2010. 6.5

[Gro16] Jens Groth. On the size of pairing-based non-interactive arguments. In Fischlin and Coron [FC16], pages 305–326. 6.5

[GTS01] Michael T. Goodrich, Roberto Tamassia, and Andrew Schwerin. Implementation of an authenticated dictionary with skip lists and commutative hashing. In *DISCEX '01*, 2001. 3.6

[GW11] Craig Gentry and Daniel Wichs. Separating succinct non-interactive arguments from all falsifiable assumptions. In Lance Fortnow and Salil P. Vadhan, editors, *Proceedings of the 43rd ACM Symposium on Theory of Computing, STOC 2011, San Jose, CA, USA, 6-8 June 2011*, pages 99–108. ACM, 2011. 6.5

[GW18] Ke Gu and Na Wu. Constant size traceable ring signature scheme without random oracles. Cryptology ePrint Archive, Report 2018/288, 2018. `https://eprint.iacr.org/2018/288`. 3.2.3

[HS90] Stuart Haber and W. Scott Stornetta. How to time-stamp a digital document. In Alfred Menezes and Scott A. Vanstone, editors, *Advances in Cryptology - CRYPTO '90, 10th Annual International Cryptology Conference, Santa Barbara, California, USA, August 11-15, 1990, Proceedings*, volume 537 of *Lecture Notes in Computer Science*, pages 437–455. Springer, 1990. 3.5

[HS03] Javier Herranz and Germán Sáez. Forking lemmas for ring signature schemes. In Thomas Johansson and Subhamoy Maitra, editors, *Progress in Cryptology - INDOCRYPT 2003, 4th International Conference on Cryptology in India, New Delhi, India, December 8-10, 2003, Proceedings*, volume 2904 of *Lecture Notes in Computer Science*, pages 266–279. Springer, 2003. 3.2.3

[IKO$^+$11] Yuval Ishai, Eyal Kushilevitz, Rafail Ostrovsky, Manoj Prabhakaran, and Amit Sahai. Efficient non-interactive secure computation. In Kenneth G. Paterson, editor, *Advances in Cryptology - EUROCRYPT 2011 - 30th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Tallinn, Estonia, May 15-19, 2011. Proceedings*, volume 6632 of *Lecture Notes in Computer Science*, pages 406–425. Springer, 2011. 7

[iot] `https://iota.org`. 3.5

[IR01] Gene Itkis and Leonid Reyzin. Forward-secure signatures with optimal signing and verifying. In Kilian [Kil01], pages 332–354. 3.2.3

[JWdV06] Ari Juels, Rebecca N. Wright, and Sabrina De Capitani di Vimercati, editors. *Proceedings of the 13th ACM Conference on Computer and Communications Security, CCS 2006, Alexandria, VA, USA, Ioctober 30 - November 3, 2006*. ACM, 2006. 7

[Kat07] Jonathan Katz. Universally composable multi-party computation using tamper-proof hardware. In Moni Naor, editor, *Advances in Cryptology - EUROCRYPT 2007, 26th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Barcelona, Spain, May 20-24, 2007, Proceedings*, volume 4515 of *Lecture Notes in Computer Science*, pages 115–128. Springer, 2007. 2.5.1

[Kil88] Joe Kilian. Founding cryptography on oblivious transfer. In Simon [Sim88], pages 20–31. 4.7

[Kil01] Joe Kilian, editor. *Advances in Cryptology - CRYPTO 2001, 21st Annual International Cryptology Conference, Santa Barbara, California, USA, August 19-23, 2001, Proceedings*, volume 2139 of *Lecture Notes in Computer Science*. Springer, 2001. 7

[KL11] Dafna Kidron and Yehuda Lindell. Impossibility results for universal composability in public-key models and with fixed inputs. *J. Cryptology*, 24(3):517–544, 2011. 2.2.3

[KLL$^+$15] Aggelos Kiayias, Nikos Leonardos, Helger Lipmaa, Kateryna Pavlyk, and Qiang Tang. Optimal rate private information retrieval from homomorphic encryption. *PoPETs*, 2015(2):222–243, 2015. 4.7

[KO97] Eyal Kushilevitz and Rafail Ostrovsky. Replication is NOT needed: SINGLE database, computationally-private information retrieval. In *38th Annual Symposium on Foundations of Computer Science, FOCS '97, Miami Beach, Florida, USA, October 19-22, 1997*, pages 364–373. IEEE Computer Society, 1997. 4.7

[Kol10] Vladimir Kolesnikov. Truly efficient string oblivious transfer using resettable tamper-proof tokens. In Daniele Micciancio, editor, *Theory of Cryptography, 7th Theory of Cryptography Conference, TCC 2010, Zurich, Switzerland, February 9-11, 2010. Proceedings*, volume 5978 of *Lecture Notes in Computer Science*, pages 327–342. Springer, 2010. 2.5.1

[KP17] Sudhakar Kumawat and Souradyuti Paul. A new constant-size accountable ring signature scheme without random oracles. In Xiaofeng Chen, Dongdai Lin, and Moti Yung, editors, *Information Security and Cryptology - 13th International Conference, Inscrypt 2017, Xi'an, China, November 3-5, 2017, Revised Selected Papers*, volume 10726 of *Lecture Notes in Computer Science*, pages 157–179. Springer, 2017. 3.2.3

[KR02] Anton Kozlov and Leonid Reyzin. Forward-secure signatures with fast key update. In Cimato et al. [CGP03], pages 241–256. 3.2.3

[KRDO17] Aggelos Kiayias, Alexander Russell, Bernardo David, and Roman Oliynykov. Ouroboros: A provably secure proof-of-stake blockchain protocol. In Jonathan Katz and Hovav Shacham, editors, *Advances in Cryptology - CRYPTO 2017 - 37th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 20-24, 2017, Proceedings, Part I*, volume 10401 of *Lecture Notes in Computer Science*, pages 357–388. Springer, 2017. 3.3

[KSW08] Jonathan Katz, Amit Sahai, and Brent Waters. Predicate encryption supporting disjunctions, polynomial equations, and inner products. In Smart [Sma08], pages 146–162. 4.2

[KZZ16] Aggelos Kiayias, Hong-Sheng Zhou, and Vassilis Zikas. Fair and robust multi-party computation using a global transaction ledger. In *EUROCRYPT (2)*, volume 9666 of *Lecture Notes in Computer Science*, pages 705–734. Springer, 2016. 2.3, 2.4

[Lin11] Yehuda Lindell. Highly-Efficient Universally-Composable Commitments Based on the DDH Assumption. In Kenny Paterson, editor, *EUROCRYPT 2011*, volume 6632, pages 446–466, Tallinn, Estonia, May15–19, 2011. Springer, Heidelberg. 2.4

[Lin17] Yehuda Lindell. How to simulate it - A tutorial on the simulation proof technique. In Yehuda Lindell, editor, *Tutorials on the Foundations of Cryptography.*, pages 277–346. Springer International Publishing, 2017. 2.6

[Lip05] Helger Lipmaa. An oblivious transfer protocol with log-squared communication. In Jianying Zhou, Javier L&apos;opez, Robert H. Deng, and Feng Bao, editors, *Information Security, 8th International Conference, ISC 2005, Singapore, September 20-23, 2005, Proceedings*, volume 3650 of *Lecture Notes in Computer Science*, pages 314–328. Springer, 2005. 4.7

[Lip09] Helger Lipmaa. First CPIR protocol with data-dependent computation. In Dong Hoon Lee and Seokhie Hong, editors, *Information, Security and Cryptology - ICISC 2009, 12th International Conference, Seoul, Korea, December 2-4, 2009, Revised Selected Papers*, volume 5984 of *Lecture Notes in Computer Science*, pages 193–210. Springer, 2009. 4.7

[Lip12] Helger Lipmaa. Progression-free sets and sublinear pairing-based non-interactive zero-knowledge arguments. In Ronald Cramer, editor, *Theory of Cryptography - 9th Theory of Cryptography Conference, TCC 2012, Taormina, Sicily, Italy, March 19-21, 2012. Proceedings*, volume 7194 of *Lecture Notes in Computer Science*, pages 169–189. Springer, 2012. 6.5

[Lip13] Helger Lipmaa. Succinct non-interactive zero knowledge arguments from span programs and linear error-correcting codes. In Kazue Sako and Palash Sarkar, editors, *Advances in Cryptology - ASIACRYPT 2013 - 19th International Conference on the Theory and Application of Cryptology and Information Security, Bengaluru, India, December 1-5, 2013, Proceedings, Part I*, volume 8269 of *Lecture Notes in Computer Science*, pages 41–60. Springer, 2013. 6.5

[LL95] Chae Hoon Lim and Pil Joong Lee. Security and performance of server-aided RSA computation protocols. In Don Coppersmith, editor, *Advances in Cryptology - CRYPTO '95, 15th Annual International Cryptology Conference, Santa Barbara, California, USA, August 27-31, 1995, Proceedings*, volume 963 of *Lecture Notes in Computer Science*, pages 70–83. Springer, 1995. 3.2.1

[LLY15] Kwangsu Lee, Dong Hoon Lee, and Moti Yung. Sequential aggregate signatures with short public keys without random oracles. *Theor. Comput. Sci.*, 579:100–125, 2015. 3.2.2, 3.2.2

[LMRS04] Anna Lysyanskaya, Silvio Micali, Leonid Reyzin, and Hovav Shacham. Sequential aggregate signatures from trapdoor permutations. In Cachin and Camenisch [CC04], pages 74–90. 3.2.2, 3.2.2

[LOS$^+$06] Steve Lu, Rafail Ostrovsky, Amit Sahai, Hovav Shacham, and Brent Waters. Sequential aggregate signatures and multisignatures without random oracles. In Serge Vaudenay, editor, *Advances in Cryptology - EUROCRYPT 2006, 25th Annual International Conference on the Theory and Applications of Cryptographic Techniques, St. Petersburg, Russia, May 28 - June 1, 2006, Proceedings*, volume 4004 of *Lecture Notes in Computer Science*, pages 465–485. Springer, 2006. 3.2.2, 3.2.2, 3.2.2

[LP17] Helger Lipmaa and Kateryna Pavlyk. A simpler rate-optimal CPIR protocol. In Aggelos Kiayias, editor, *Financial Cryptography and Data Security - 21st International Conference, FC 2017, Sliema, Malta, April 3-7, 2017, Revised Selected Papers*, volume 10322 of *Lecture Notes in Computer Science*, pages 621–638. Springer, 2017. 4.7

[LRSW99] Anna Lysyanskaya, Ronald L. Rivest, Amit Sahai, and Stefan Wolf. Pseudonym systems. In Howard M. Heys and Carlisle M. Adams, editors, *Selected Areas in Cryptography, 6th Annual International Workshop, SAC'99, Kingston, Ontario, Canada, August 9-10, 1999, Proceedings*, volume 1758 of *Lecture Notes in Computer Science*, pages 184–199. Springer, 1999. 3.2.2

[LW05] Joseph K. Liu and Duncan S. Wong. Linkable ring signatures: Security models and new schemes. In Osvaldo Gervasi, Marina L. Gavrilova, Vipin Kumar, Antonio Laganà, Heow Pueh Lee, Youngsong Mun, David Taniar, and Chih Jeng Kenneth Tan, editors, *Computational Science and Its Applications - ICCSA 2005, International Conference, Singapore, May 9-12, 2005, Proceedings, Part II*, volume 3481 of *Lecture Notes in Computer Science*, pages 614–623. Springer, 2005. 3.2.3

[LW10] Allison B. Lewko and Brent Waters. New techniques for dual system encryption and fully secure HIBE with short ciphertexts. In Micciancio [Mic10], pages 455–479. 3.2.2

[LWW03] Joseph K. Liu, Victor K. Wei, and Duncan S. Wong. A separable threshold ring signature scheme. In Jong In Lim and Dong Hoon Lee, editors, *Information Security and Cryptology - ICISC 2003, 6th International Conference, Seoul, Korea, November 27-28, 2003, Revised Papers*, volume 2971 of *Lecture Notes in Computer Science*, pages 12–26. Springer, 2003. 3.2.3

[LWW04] Joseph K. Liu, Victor K. Wei, and Duncan S. Wong. Linkable spontaneous anonymous group signature for ad hoc groups (extended abstract). In Huaxiong Wang, Josef Pieprzyk, and Vijay Varadharajan, editors, *Information Security and Privacy: 9th Australasian Conference, ACISP 2004, Sydney, Australia, July 13-15, 2004. Proceedings*, volume 3108 of *Lecture Notes in Computer Science*, pages 325–335. Springer, 2004. 3.2.3

[LZCS16] Russell W. F. Lai, Tao Zhang, Sherman S. M. Chow, and Dominique Schröder. Efficient sanitizable signatures without random oracles. In Ioannis G. Askoxylakis, Sotiris Ioannidis, Sokratis K. Katsikas, and Catherine A. Meadows, editors, *Computer Security - ESORICS 2016 - 21st European Symposium on Research in Computer Security, Heraklion, Greece, September 26-30, 2016, Proceedings, Part I*, volume 9878 of *Lecture Notes in Computer Science*, pages 363–380. Springer, 2016. 3.2.3

[MBB$^+$13] Carlos Aguilar Melchor, Slim Bettaieb, Xavier Boyen, Laurent Fousse, and Philippe Gaborit. Adapting lyubashevsky's signature schemes to the ring signature setting. In Amr Youssef, Abderrahmane Nitaj, and Aboul Ella Hassanien, editors, *Progress in Cryptology - AFRICACRYPT 2013, 6th International Conference on Cryptology in Africa, Cairo, Egypt, June 22-24, 2013. Proceedings*, volume 7918 of *Lecture Notes in Computer Science*, pages 1–25. Springer, 2013. 3.2.3

[Mer80] Ralph C. Merkle. Protocols for public key cryptosystems. In *Proceedings of the 1980 IEEE Symposium on Security and Privacy, Oakland, California, USA, April 14-16, 1980*, pages 122–134. IEEE Computer Society, 1980. 3.5, 3.6

[Mer89] Ralph C. Merkle. A certified digital signature. In Brassard [Bra90], pages 218–238. 3.6

[MHKS14] Andrew Miller, Michael Hicks, Jonathan Katz, and Elaine Shi. Authenticated data structures, generically. In Suresh Jagannathan and Peter Sewell, editors, *The 41st Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages, POPL '14, San Diego, CA, USA, January 20-21, 2014*, pages 411–424. ACM, 2014. 3.6

[Mic10] Daniele Micciancio, editor. *Theory of Cryptography, 7th Theory of Cryptography Conference, TCC 2010, Zurich, Switzerland, February 9-11, 2010. Proceedings*, volume 5978 of *Lecture Notes in Computer Science*. Springer, 2010. 7

[Mic16] Silvio Micali. ALGORAND: the efficient and democratic ledger. *CoRR*, abs/1607.01341, 2016. 3.2.3, 3.3

[MKI88] Tsutomu Matsumoto, Koki Kato, and Hideki Imai. Speeding up secret computations with insecure auxiliary devices. In Shafi Goldwasser, editor, *Advances in Cryptology - CRYPTO '88, 8th Annual International Cryptology Conference, Santa Barbara, California, USA, August 21-25, 1988, Proceedings*, volume 403 of *Lecture Notes in Computer Science*, pages 497–506. Springer, 1988. 3.2.1

[MMM02] Tal Malkin, Daniele Micciancio, and Sara K. Miner. Efficient generic forward-secure signatures with an unbounded number of time periods. In Lars R. Knudsen, editor, *Advances in Cryptology - EUROCRYPT 2002, International Conference on the Theory and Applications of Cryptographic Techniques, Amsterdam, The Netherlands, April 28 - May 2, 2002, Proceedings*, volume 2332 of *Lecture Notes in Computer Science*, pages 400–417. Springer, 2002. 3.2.3

[MNG$^+$01] Chip Martel, Glen Nuckolls, Michael Gertz, Premkumar T. Devanbu, April Kwong, and Stuart G. Stubblebine. A general model for authenticated data publication. Available from `http://web.cs.ucdavis.edu/~devanbu/files/model-paper.pdf`, 2001. 3.6

[MOR01] Silvio Micali, Kazuo Ohta, and Leonid Reyzin. Accountable-subgroup multisignatures: extended abstract. In Michael K. Reiter and Pierangela Samarati, editors, *CCS 2001, Proceedings of the 8th ACM Conference on Computer and Communications Security, Philadelphia, Pennsylvania, USA, November 6-8, 2001.*, pages 245–254. ACM, 2001. 3.2.2

[MPSW18] Gregory Maxwell, Andrew Poelstra, Yannick Seurin, and Pieter Wuille. Simple schnorr multi-signatures with applications to bitcoin. Cryptology ePrint Archive, Report 2018/068, 2018. https://eprint.iacr.org/2018/068. 3.2.2, 3.1, 3.2.2

[MR17] Payman Mohassel and Mike Rosulek. Non-interactive secure 2pc in the offline/online and batch settings. In Jean-Sébastien Coron and Jesper Buus Nielsen, editors, *Advances in Cryptology - EUROCRYPT 2017 - 36th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Paris, France, April 30 - May 4, 2017, Proceedings, Part III*, volume 10212 of *Lecture Notes in Computer Science*, pages 425–455, 2017. 7

[MRV99] Silvio Micali, Michael O. Rabin, and Salil P. Vadhan. Verifiable random functions. In *40th Annual Symposium on Foundations of Computer Science, FOCS '99, 17-18 October, 1999, New York, NY, USA*, pages 120–130. IEEE Computer Society, 1999. 3.3

[MS08] Tal Moran and Gil Segev. David and goliath commitments: UC computation for asymmetric parties using tamper-proof hardware. In Nigel P. Smart, editor, *Advances in Cryptology - EUROCRYPT 2008, 27th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Istanbul, Turkey, April 13-17, 2008. Proceedings*, volume 4965 of *Lecture Notes in Computer Science*, pages 527–544. Springer, 2008. 2.5.1

[MSM17] Paulo Martins, Leonel Sousa, and Artur Mariano. A survey on fully homomorphic encryption: An engineering perspective. *ACM Comput. Surv.*, 50(6):83:1–83:33, 2017. 5.2

[MSS16] Mark Manulis, Ahmad-Reza Sadeghi, and Steve Schneider, editors. *Applied Cryptography and Network Security - 14th International Conference, ACNS 2016, Guildford, UK, June 19-22, 2016. Proceedings*, volume 9696 of *Lecture Notes in Computer Science*. Springer, 2016. 7

[MWLD10] Changshe Ma, Jian Weng, Yingjiu Li, and Robert H. Deng. Efficient discrete logarithm based multi-signature scheme in the plain public key model. *Des. Codes Cryptography*, 54(2):121–133, 2010. 3.2.2, 3.2.2

[Nak09] Satoshi Nakamoto. Bitcoin: A peer-to-peer electronic cash system. Whitepaper, 2009. http://bitcoin.org/bitcoin.pdf. 3.5

[Nao02] Moni Naor. Deniable ring authentication. In Moti Yung, editor, *Advances in Cryptology - CRYPTO 2002, 22nd Annual International Cryptology Conference, Santa Barbara, California, USA, August 18-22, 2002, Proceedings*, volume 2442 of *Lecture Notes in Computer Science*, pages 481–498. Springer, 2002. 3.7

[Nat17] National Institute of Standards and Technology. Post-quantum cryptography standardization. https://csrc.nist.gov/Projects/Post-Quantum-Cryptography/Post-Quantum-Cryptography-Standardization, 2017. Accessed: 2018-06-25. 2.7

[Nev08] Gregory Neven. Efficient sequential aggregate signed data. In Smart [Sma08], pages 52–69. 3.2.2, 3.1, 3.2.2, 3.2, 3.2.2

[Nie02] Jesper Buus Nielsen. Separating random oracle proofs from complexity theoretic proofs: The non-committing encryption case. In Yung [Yun02], pages 111–126. 2.3

[NN00] Moni Naor and Kobbi Nissim. Certificate revocation and certificate update. *IEEE Journal on Selected Areas in Communications*, 18(4):561–570, 2000. 3.6

[Noe15] Shen Noether. Ring signature confidential transactions for monero. Cryptology ePrint Archive, Report 2015/1098, 2015. https://eprint.iacr.org/2015/1098. 3.2.3

[NSJ08] Peng Ning, Paul F. Syverson, and Somesh Jha, editors. *Proceedings of the 2008 ACM Conference on Computer and Communications Security, CCS 2008, Alexandria, Virginia, USA, October 27-31, 2008*. ACM, 2008. 7

[O'N10] Adam O'Neill. Definitional issues in functional encryption. *IACR Cryptology ePrint Archive*, 2010:556, 2010. 4.2

[OPV15] Rafail Ostrovsky, Giuseppe Persiano, and Ivan Visconti. Impossibility of black-box simulation against leakage attacks. In Rosario Gennaro and Matthew Robshaw, editors, *Advances in Cryptology - CRYPTO 2015 - 35th Annual Cryptology Conference, Santa Barbara, CA, USA, August 16-20, 2015, Proceedings, Part II*, volume 9216 of *Lecture Notes in Computer Science*, pages 130–149. Springer, 2015. 2.5.2

[Ped91] Torben P. Pedersen. A threshold cryptosystem without a trusted party (extended abstract). In Davies [Dav91], pages 522–526. 3.2.3

[Pfi01] Birgit Pfitzmann, editor. *Advances in Cryptology - EUROCRYPT 2001, International Conference on the Theory and Application of Cryptographic Techniques, Innsbruck, Austria, May 6-10, 2001, Proceeding*, volume 2045 of *Lecture Notes in Computer Science*. Springer, 2001. 7

[PGHR13] Bryan Parno, Craig Gentry, Jon Howell, and Mariana Raykova. Pinocchio: Nearly Practical Verifiable Computation. In *IEEE SP 2013*, pages 238–252, Berkeley, CA, USA, May 19-22, 2013. IEEE Computer Society. 6.5

[PHGR13] Bryan Parno, Jon Howell, Craig Gentry, and Mariana Raykova. Pinocchio: Nearly practical verifiable computation. In *2013 IEEE Symposium on Security and Privacy, SP 2013, Berkeley, CA, USA, May 19-22, 2013*, pages 238–252. IEEE Computer Society, 2013. 3.6

[PSS17] Rafael Pass, Lior Seeman, and Abhi Shelat. Analysis of the blockchain protocol in asynchronous networks. In *EUROCRYPT (2)*, volume 10211 of *Lecture Notes in Computer Science*, pages 643–673, 2017. 2.4

[PTT08] Charalampos Papamanthou, Roberto Tamassia, and Nikos Triandopoulos. Authenticated hash tables. In Ning et al. [NSJ08], pages 437–448. 3.6

[PTT11] Charalampos Papamanthou, Roberto Tamassia, and Nikos Triandopoulos. Optimal verification of operations on dynamic sets. In Phillip Rogaway, editor, *Advances in Cryptology - CRYPTO 2011 - 31st Annual Cryptology Conference, Santa Barbara, CA, USA, August 14-18, 2011. Proceedings*, volume 6841 of *Lecture Notes in Computer Science*, pages 91–110. Springer, 2011. 3.6

[PTV10] Rafael Pass, Wei-Lung Dustin Tseng, and Muthuramakrishnan Venkitasubramaniam. Eye for an eye: Efficient concurrent zero-knowledge in the timing model. In Micciancio [Mic10], pages 518–534. 6.1.1

[PW92] Birgit Pfitzmann and Michael Waidner. Attacks on protocols for server-aided RSA computation. In Rainer A. Rueppel, editor, *Advances in Cryptology - EUROCRYPT '92, Workshop on the Theory and Application of of Cryptographic Techniques, Balatonfüred, Hungary, May 24-28, 1992, Proceedings*, volume 658 of *Lecture Notes in Computer Science*, pages 153–162. Springer, 1992. 3.2.1

[PZ13] Christian Paquin and Greg Zaverucha. U-prove cryptographic specification V1.1. Technical report, Microsoft Corporation, 2013. 3.4

[Rey01] Leonid Reyzin. *Zero-Knowledge with Public Keys.* PhD thesis, MIT Press, Cambridge, 2001. 2.2.3

[RST01] Ronald L. Rivest, Adi Shamir, and Yael Tauman. How to leak a secret. In Colin Boyd, editor, *Advances in Cryptology - ASIACRYPT 2001, 7th International Conference on the Theory and Application of Cryptology and Information Security, Gold Coast, Australia, December 9-13, 2001, Proceedings*, volume 2248 of *Lecture Notes in Computer Science*, pages 552–565. Springer, 2001. 3.2.3

[RY07] Thomas Ristenpart and Scott Yilek. The power of proofs-of-possession: Securing multiparty signatures against rogue-key attacks. In Moni Naor, editor, *Advances in Cryptology - EUROCRYPT 2007, 26th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Barcelona, Spain, May 20-24, 2007, Proceedings*, volume 4515 of *Lecture Notes in Computer Science*, pages 228–245. Springer, 2007. 3.2.2

[SALY17] Shifeng Sun, Man Ho Au, Joseph K. Liu, and Tsz Hon Yuen. Ringct 2.0: A compact accumulator-based (linkable ring signature) protocol for blockchain cryptocurrency monero. In Simon N. Foley, Dieter Gollmann, and Einar Snekkenes, editors, *Computer Security - ESORICS 2017 - 22nd European Symposium on Research in Computer Security, Oslo, Norway, September 11-15, 2017, Proceedings, Part II*, volume 10493 of *Lecture Notes in Computer Science*, pages 456–474. Springer, 2017. 3.2.3

[Sch11] Dominique Schröder. How to aggregate the CL signature scheme. In Vijay Atluri and Claudia Díaz, editors, *Computer Security - ESORICS 2011 - 16th European Symposium on Research in Computer Security, Leuven, Belgium, September 12-14, 2011. Proceedings*, volume 6879 of *Lecture Notes in Computer Science*, pages 298–314. Springer, 2011. 3.2.2, 3.2.2

[Sha79] Adi Shamir. How to share a secret. *Commun. ACM*, 22(11):612–613, 1979. 5.1

[Sha84] Adi Shamir. Identity-based cryptosystems and signature schemes. In G. R. Blakley and David Chaum, editors, *Advances in Cryptology, Proceedings of CRYPTO '84, Santa Barbara, California, USA, August 19-22, 1984, Proceedings*, volume 196 of *Lecture Notes in Computer Science*, pages 47–53. Springer, 1984. 4.3

[Sho94] Peter W. Shor. Algorithms for quantum computation: Discrete logarithms and factoring. In *35th Annual Symposium on Foundations of Computer Science, Santa Fe, New Mexico, USA, 20-22 November 1994*, pages 124–134. IEEE Computer Society, 1994. 2.7

[Sho00] Victor Shoup. Practical threshold signatures. In Bart Preneel, editor, *Advances in Cryptology - EUROCRYPT 2000, International Conference on the Theory and Application of Cryptographic Techniques, Bruges, Belgium, May 14-18, 2000, Proceeding*, volume 1807 of *Lecture Notes in Computer Science*, pages 207–220. Springer, 2000. 3.2.3

[Sim88] Janos Simon, editor. *Proceedings of the 20th Annual ACM Symposium on Theory of Computing, May 2-4, 1988, Chicago, Illinois, USA*. ACM, 1988. 7

[SLZ16] Yonatan Sompolinsky, Yoad Lewenberg, and Aviv Zohar. Spectre: A fast and scalable cryptocurrency protocol. Cryptology ePrint Archive, Report 2016/1159, 2016. `https://eprint.iacr.org/2016/1159`. 3.5

[Sma08] Nigel P. Smart, editor. *Advances in Cryptology - EUROCRYPT 2008, 27th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Istanbul, Turkey, April 13-17, 2008. Proceedings*, volume 4965 of *Lecture Notes in Computer Science*. Springer, 2008. 7

[Sov18] The Sovrin Foundation. Sovrin: A protocol and token for self- sovereign identity and decentralized trust. White paper, January 2018. 3.4

[SP92] Alfredo De Santis and Giuseppe Persiano. Zero-knowledge proofs of knowledge without interaction (extended abstract). In *33rd Annual Symposium on Foundations of Computer Science, Pittsburgh, Pennsylvania, USA, 24-27 October 1992*, pages 427–436. IEEE Computer Society, 1992. 6.3

[Sta99] M. Stam. Toggling schemes for electronic voting. Master's thesis, Dept of Mathematics and Computer Science, TU Eindhoven, The Netherlands, June 1999. 3.2.3

[Ste98] Julien P. Stern. A new efficient all-or-nothing disclosure of secrets protocol. In Kazuo Ohta and Dingyi Pei, editors, *Advances in Cryptology - ASIACRYPT '98, International Conference on the Theory and Applications of Cryptology and Information Security, Beijing, China, October 18-22, 1998, Proceedings*, volume 1514 of *Lecture Notes in Computer Science*, pages 357–371. Springer, 1998. 4.7

[SW05] Amit Sahai and Brent Waters. Fuzzy identity-based encryption. In Ronald Cramer, editor, *Advances in Cryptology - EUROCRYPT 2005, 24th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Aarhus, Denmark, May 22-26, 2005, Proceedings*, volume 3494 of *Lecture Notes in Computer Science*, pages 457–473. Springer, 2005. 4.2

[SW07] Hovav Shacham and Brent Waters. Efficient ring signatures without random oracles. In Tatsuaki Okamoto and Xiaoyun Wang, editors, *Public Key Cryptography - PKC 2007, 10th International Conference on Practice and Theory in Public-Key Cryptography, Beijing, China, April 16-20, 2007, Proceedings*, volume 4450 of *Lecture Notes in Computer Science*, pages 166–180. Springer, 2007. 3.2.3

[SW14] Amit Sahai and Brent Waters. How to use indistinguishability obfuscation: deniable encryption, and more. In David B. Shmoys, editor, *Symposium on Theory of Computing, STOC 2014, New York, NY, USA, May 31 - June 03, 2014*, pages 475–484. ACM, 2014. 4.5

[swi] `https://www.swirlds.com`. 3.5

[SZ18] Yonatan Sompolinsky and Aviv Zohar. Phantom: A scalable blockdag protocol. Cryptology ePrint Archive, Report 2018/104, 2018. `https://eprint.iacr.org/2018/104`. 3.5

[Tam03] Roberto Tamassia. Authenticated data structures. In Giuseppe Di Battista and Uri Zwick, editors, *Algorithms - ESA 2003, 11th Annual European Symposium, Budapest, Hungary, September 16-19, 2003, Proceedings*, volume 2832 of *Lecture Notes in Computer Science*, pages 2–5. Springer, 2003. 3.6

[TW05] Patrick P. Tsang and Victor K. Wei. Short linkable ring signatures for e-voting, e-cash and attestation. In Robert H. Deng, Feng Bao, HweeHwa Pang, and Jianying Zhou, editors, *Information Security Practice and Experience, First International Conference, ISPEC 2005, Singapore, April 11-14, 2005, Proceedings*, volume 3439 of *Lecture Notes in Computer Science*, pages 48–60. Springer, 2005. 3.2.3

[TWC$^+$04] Patrick P. Tsang, Victor K. Wei, Tony K. Chan, Man Ho Au, Joseph K. Liu, and Duncan S. Wong. Separable linkable threshold ring signatures. In Anne Canteaut and Kapalee Viswanathan, editors, *Progress in Cryptology - INDOCRYPT 2004, 5th International Conference on Cryptology in India, Chennai, India, December 20-22, 2004, Proceedings*, volume 3348 of *Lecture Notes in Computer Science*, pages 384–398. Springer, 2004. 3.2.3

[vS13] Nicolas van Saberhagen. Cryptonote v 2.0. `https://cryptonote.org/whitepaper.pdf`, October 2013. 3.2.3

[WB15] Michael Walfish and Andrew J. Blumberg. Verifying computations without reexecuting them. *Commun. ACM*, 58(2):74–84, 2015. 3.6

[WFLW03] Duncan S. Wong, Karyin Fung, Joseph K. Liu, and Victor K. Wei. On the rs-code construction of ring signature schemes and a threshold setting of RST. In Sihan Qing, Dieter Gollmann, and Jianying Zhou, editors, *Information and Communications Security, 5th International Conference, ICICS 2003, Huhehaote, China, October 10-13, 2003, Proceedings*, volume 2836 of *Lecture Notes in Computer Science*, pages 34–46. Springer, 2003. 3.2.3

[WSS09] Peter Williams, Radu Sion, and Dennis E. Shasha. The blind stone tablet: Outsourcing durability to untrusted parties. In *Proceedings of the Network and Distributed System Security Symposium, NDSS 2009, San Diego, California, USA, 8th February - 11th February 2009*. The Internet Society, 2009. 3.5

[XY04] Shouhuai Xu and Moti Yung. Accountable ring signatures: A smart card approach. In Jean-Jacques Quisquater, Pierre Paradinas, Yves Deswarte, and Anas Abou El Kalam, editors, *Smart Card Research and Advanced Applications VI, IFIP 18th World Computer Congress, TC8/WG8.8 & TC11/WG11.2 Sixth International Conference on Smart Card Research and Advanced Applications (CARDIS), 22-27 August 2004, Toulouse, France*, volume 153 of *IFIP*, pages 271–286. Kluwer/Springer, 2004. 3.2.3

[Yao82] Andrew Chi-Chih Yao. Protocols for secure computations (extended abstract). In *23rd Annual Symposium on Foundations of Computer Science, Chicago, Illinois, USA, 3-5 November 1982*, pages 160–164. IEEE Computer Society, 1982. 4.7

[Yao86] Andrew Chi-Chih Yao. How to generate and exchange secrets (extended abstract). In *27th Annual Symposium on Foundations of Computer Science, Toronto, Canada, 27-29 October 1986*, pages 162–167. IEEE Computer Society, 1986. 5.4

[Yun02] Moti Yung, editor. *Advances in Cryptology - CRYPTO 2002, 22nd Annual International Cryptology Conference, Santa Barbara, California, USA, August 18-22, 2002, Proceedings*, volume 2442 of *Lecture Notes in Computer Science*. Springer, 2002. 7