Subproject C1: Robustness and Security

Johannes Blömer[1], Fabian Eidens[1], Tibor Jager[2], David Niehues[2], Christian Scheideler[1]

1 Department of Computer Science, Paderborn University, Paderborn, Germany
2 School of Electrical, Information and Media Engineering, University of Wuppertal, Wuppertal, Germany

1 Introduction

In Subproject C1 of the CRC, we developed methods and techniques that ensure high robustness and security in OTF markets, taking into account the specific characteristics and demands of such a market. Although robustness and security share the high level objectives of availability and acceptance, in practice, they require quite different techniques. Robustness usually (but not exclusively, think about denial-of-service attacks) deals with unforeseen but not necessarily malicious behavior of participants, whereas malicious attacks are at the very core of security. For a service to be secure, it needs to satisfy several key properties. First, there need to be mechanisms that guarantee the integrity of the communication and the authenticity of communication partners. Second, in many cases communication needs to be confidential. Third, privacy-preserving mechanisms are required to protect the identity of communication parties, i.e., to guarantee the anonymity of parties, and to protect sensitive data like business secrets or customer data. While these are the standard requirements and properties from security and privacy, and several techniques exist to realize these goals, even simultaneously, online markets such as the OTF market have special characteristics that necessitate specialized and enhanced methods. For example, the decentralized nature of OTF markets mostly prohibits the use of centralized techniques for access control. Furthermore, in some applications, e.g., machine learning based applications, the quality of services offered by service and software providers may depend heavily on the quality of (sensitive) data provided by customers, such as learning data. Hence there is a need to make sensitive data available for learning algorithms while still preserving their confidentiality and privacy.

For a service to be robust, it should be available 24/7 and withstand faults as well as Byzantine behavior as much as this is possible. This is particularly important for an open OTF market, where many different stakeholders need to interact without any prior trust relationships and customers are only willing to use it if its services are highly available. A

bloemer@upb.de (Johannes Blömer), fabian.eidens@uni-paderborn.de (Fabian Eidens), tibor.jager@uni-wuppertal.de (Tibor Jager), niehues@uni-wuppertal.de (David Niehues), scheideler@upb.de (Christian Scheideler)

critical aspect of such a market is, for example, any kind of information needed for the composition of software solutions, such as reputation data. Ideally, a solution for such information systems should even withstand insider attacks since the code used for the organization of the OTF market might be freely available to everyone. Another important aspect is an appropriate infrastructure interconnecting the market participants that has a low maintenance overhead and that can handle even large churn (i.e., a high arrival and departure rate of market participants) without seriously affecting the exchange of information.

In the area of robustness, we first focused on robust information systems. A huge problem for robust information systems are denial-of-service (DoS) attacks. There are basically two approaches to counter DoS attacks: stopping DoS attacks, for example, by filtering them, or setting up a system that remains available despite a DoS attack. Stopping DoS attacks is usually a hard problem as it requires interactions with Internet service providers or security agencies, so we focused on systems that remain available despite DoS attacks. Various such systems have already been proposed when DoS attacks are initiated by outsiders, i.e., attackers that do not know the setup of the system. We, instead, considered attacks by insiders, i.e., attackers that know everything about the system and can use that information in order to start DoS attacks on a limited number of its servers in order to make certain parts of the information unavailable to legitimate requests. Our findings are summarized in Section 2.4. Later, we also focused on the problem of protecting an overlay network against DoS attacks. For an overlay network to be scalable, its degree should be at most polylogarithmic in the number of nodes, since a high degree also means a high maintenance overhead. However, the lower the degree, the easier it is to start so-called Eclipse attacks, i.e., attacks that isolate parts of the network from the rest. These Eclipse attacks can, for example, be performed by starting a DoS attack on the neighborhood of a targeted node so that it cannot interact anymore with the rest of the network. Our approach to defend against such kinds of attacks is to continuously change the topology of the network, and to do this so quickly that an attacker cannot keep up with the changes. Our findings are summarized in Section 2.5.

In security, we focused on techniques that guarantee strong authentication of data and entities while also preserving user privacy. To achieve this we construct enhanced signature schemes and anonymous credentials. Finally, to help actors in OTF markets find appropriate services we study secure and anonymous reputation systems. We begin by briefly discussing reputation systems and their use in the context of OTF. In OTF markets, reputation systems can complement certificates and provide important information about the quality and trustworthiness of service or software providers. However, reputation systems as currently used in many online markets face several security and privacy issues. In particular, to avoid retribution for negative ratings, anonymity of ratings seems to be a desirable property. Anonymity itself, on the other hand, creates problems, such as skewing the reputation score of a service by a flood a negative ratings. To overcome these problems, we have identified key properties of secure and privacy-preserving reputation systems. These properties have been summarized into precise definitions of cryptographic reputation systems that did not exist prior to our work. We also provide efficient constructions of reputation systems. These results are described in more detail in Section 2.1. To authenticate data and users, one can use signatures and credentials, respectively. However, in the OTF market we often need enhanced versions of signatures or credentials to meet

the requirements of the markets. To meet these demands, we defined and constructed several novel cryptographic principles. These are described in more details in Section 2.2 (updatable anonymous credentials) and Section 2.3 (verifiable random functions). Whereas digital signatures authenticate data, credentials authenticate entities, usually based on attributes of entities. If credentials are used across many applications, they allow for tracking and may reveal private information of entities, e.g., in the OTF markets, entities may be users that contact many service providers and, given a composed service, have to use different compute centers. Hence anonymity is a concern. This is provided by anonymous credentials, a cryptographic technique developed in the last 20 years. In our research we realized efficient constructions for anonymous credentials with additional features, e.g., updatability. As an application of the credentials, in transfer project T2 of the CRC 901 we design privacy-preserving incentive systems (see page 237). Verifiable random functions are enhanced digital signatures whose additional properties make them attractive and useful for applications in the OTF market like consensus systems and public-key distribution systems. Usually, constructions of verifiable random functions rely on the so-called random oracle model. However, it is well-known that random oracles cannot be realized and need to be replaced by standard hash functions, leading to constructions whose security is only heuristic. We developed new verifiable random functions that do not rely on the random oracle model and are more efficient then previous constructions. The techniques introduced in this work also have applications beyond verifiable random functions.

2 Main Contributions

In this section, we review following five highlights from the research in Subproject C1:

- cryptographic reputation systems (Section 2.1)
- updatable anonymous credentials (Section 2.2)
- efficient verifiable random functions without random oracles (Section 2.3)
- insider-resistant distributed storage systems (Section 2.4)
- construction and maintenance of robust overlays (Section 2.5).

2.1 Cryptographic Reputation Systems

In an OTF market, users will contact OTF providers who, in turn, will contact service and/or software providers. In some cases, these contacts will not be based on substantial prior direct experience. In particular, OTF service providers rely on software providers that they use only occasionally. In these cases, as in others, it is valuable to have information about service and software providers that helps customers (users or service providers) to assess the quality and trustworthiness of other providers. For software and service providers, certificates may play an important role. However, the dynamics of an OTF market reduces the availability of trustworthy certificates compared to more traditional markets. Moreover, certificates may not say anything about the quality of (recent) products

and services offered by certified providers. As an alternative, reputation systems become more important, as witnessed for decades in online consumer markets.

Reputation systems provide valuable information about previous transactions and are popular tools to measure the trustworthiness of interacting parties. This measurement relies on the existence of a large number of ratings for one specific service or product. However, in most practical applications the process of rating reveals much information about the rater, besides the actual rating. Providers of reputation systems may use this information in many different ways that are not necessarily desired by the users, such as profiling users. Moreover, users can feel compelled to rate "dishonestly/benevolently" when they fear negative consequences from negative ratings. Therefore, it is important that raters at least have the choice to not reveal more information than the actual rating. Besides that, reputation systems need to be protected against various attacks to provide trustworthy, reliable, and honest ratings. These attacks include self-rating attacks (also known as self-promoting attacks), Sybil attacks, whitewashing attacks, bad mouthing attacks, ballot stuffing attacks, and value imbalance attacks. Both the privacy concerns and the prevention of attacks are discussed frequently in the literature, e.g., [BPS+17; ZWC+16], albeit they were often not considered simultaneously. Further important security properties for reputation systems are anonymity, (public) linkability, traceability, and non-frameability, as discussed in e.g., [BJK15; ZWC+16]. Anonymity means that ratings of honest users are indistinguishable, whereas public linkability requires that anyone can decide whether or not two ratings for the same product were created by the same user. Also, ratings need to be traceable: The identity of any rater can be determined by a designated system manager. In the course of this, non-frameability guarantees that honest parties are not blamed of having rated some product, when they did not. The combination of traceability and non-frameability enables penalizing dishonest behavior.

Two different approaches to define and prove the security of cryptographic primitives are used in the literature: experiment based and simulation-based. In experiment-based security, so-called security experiments and games are defined mathematically, in which an adversary plays against a challenger running a cryptographic primitive. In a security proof for a cryptographic primitive one then shows that no efficient algorithm or adversary can win this game, except with tiny probability of success. Simulation-based security definitions define security properties by describing ideal scenarios that make use of, practically not realizable, trusted parties. A security proof for a cryptographic primitive shows that an adversary trying to attack the primitive does not have a significantly better chance of succeeding than an adversary in the idealized scenario. Experiment-based security definitions and proofs tend to be more efficient and easier to understand. In comparison, simulation-based security usually offers better security guarantees, in particular if cryptographic primitives are combined or composed with each other. The important simulation-based framework for proving the security of composed cryptographic primitives is Ran Cannetti's *universal composability (UC)* framework [Can01]. In our research we considered both, experiment-based and simulation-based, definitions and constructions.

Experiment-Based Security of Reputation Systems

In [BJK15] we gave a first definition of a cryptographically secure and anonymous reputation systems. We also provided a construction of such a system based on group signature schemes. Group signatures are one of the most important primitives for privacy-preserving

cryptography. A group signature allows a group of users to sign documents on behalf of the group without revealing the identity of the specific member. In our system, we establish a separate group for each product or service, consisting of all users that bought the product, and hence have the right to rate the product. The reputation system provides anonymity, traceability, strong-exculpability, verifier-local revocation, and public linkability. Except for revocation, these properties have already been discussed above. Revocation mechanisms are a security check against the misuse of anonymity (or better pseudonymity), since it allows a system manager to revoke the rights of users, i.e., in group signatures the right to sign messages and in reputation systems the right to publish the rating. A system has verifier-local revocation, if revocation messages only have to be sent to signature or rating verifiers but not to individual signers or raters. It is well known how to realize anonymity, traceability, non-frameability, and revocation in the context of group signatures, although not necessarily simultaneously and for many groups in parallel. Our construction of a reputation system achieves this, and adds some properties specific to reputation systems. The construction is based on a group signature scheme by Boneh, Boyen, and Shacham [BBS04] and the dynamic version of the scheme presented by Delerablée and Pointcheval [PS16]. These schemes already give us anonymity, traceability, and strong-exculpability. To achieve verifier-local revocation we modify a technique by Nakanishi and Funabiki [NF06]. With the same technique we achieve public linkability. Note that anonymity of group signatures does not imply anonymity in our reputation system. This is due to the fact that providers control the groups corresponding to several products. Hence, they may combine information for different groups to violate anonymity. To prevent this, in our construction we employ a system manager that contributes a trustworthy component to each group public key.

UC-Secure Reputation Systems

Typically, reputation systems are used in combination with other applications. Common experiment-based security definitions, such as the ones defined and used in [BJK15], provide next to no security in such circumstances. With the universal composability framework (UC) of R. Canetti [Can01] there exists a methodology that guarantees security even in composed applications. Informally, in UC the execution of a real-life protocol is compared to the execution of an ideal protocol. If the real-life and ideal protocol executions are indistinguishable, then the real-life protocol is UC-secure. Based on this security definition, Canetti formulates a composition theorem that states that any UC-secure protocol is also secure when it is composed with other protocols.

In [BEJ18], first we present an ideal functionality for reputation systems in the UC framework. Our ideal functionality prevents all previously mentioned attacks and provides anonymity, public linkability, traceability, and non-frameability. Based on this, and extending the construction in [BJK15], we construct a reputation system where (a) users can rate each other's products, (b) there is no separation of customers and providers, (c) security is preserved under composition, i.e., we present an efficient protocol for reputation systems that realizes the ideal functionality for reputation systems. All three properties are highly relevant for the use of reputation systems in an OTF market. Here, reputation systems are embedded into a larger (security) system. Hence, universal composability is required. Moreover, participants in OTF markets may simultaneously or at different times play the roles of a user and of a service or software provider. Hence, it is mandatory,

that reputation systems satisfy properties (a) and (b), as well. On a technical level, our reputation system is influenced by techniques known from Σ-protocols and (dynamic) group signatures, similarly to the scheme in [BJK15]. However, to achieve UC-security we need to employ numerous advanced techniques known from other constructions of UC-secure cryptographic primitives. Somewhat surprisingly, the resulting system, in addition to being UC-secure, is also more efficient and more flexible than the scheme in [BJK15].

2.2 Updatable Anonymous Credentials

In current systems, classical authentication of a user at a provider usually involves that a user provides identifying information (e.g., name, email) combined with some user specific secret (e.g., passport, password). Presented with this, the provider grants the user access to its service. Furthermore, while the user is interacting with the provided service, the provider stores additional data of the user in a database. For example, profile information such as address, day of birth, and user's preferences. Already this simple example, is a threat to users' privacy, since they have no sovereignty over their data and data becomes stores and associated with their identity with no or limited benefit for the user. This becomes even more serious if we consider several providers that pool their databases (for example, after an acquisition), which allows them to link users across services.

Anonymous credentials employed in such a scenario enables anonymous authentication of users at providers such that no identifying information is revealed. For this, an anonymous credential encodes information about the user in certified attributes and providers can define (access) policies over attributes. Therefore, an authentication via an anonymous credential only proves that the user in question has a credential on certified attributes that satisfy the policy. For example, the policy checks if the user has a valid subscription for the service of the provider. With anonymous credentials, the provider then only learns that the end date of the subscription is before or after the current date. In general, a policy can be interpreted as a statement and through authentication a provider only learns the truthfulness of the statement based on the attributes of a user. Hence, authentication via anonymous credentials is more expressive than classical authentication and leaks minimal information. In the literature, this extension to anonymous credentials [CL01; CL04; PS16] is referred to as attribute-based anonymous credentials and many more extensions are given in the literature, e.g., delegation of credentials [BCC$^+$09; BB18; CL19; CDD17; MSBM22], revocation [CL01; CL02; CKS10], auditing [CLNR14], and expressive policies [CG08; BBB$^+$18].

Beyond that, anonymous credentials solve the problem that providers operate databases that includes user data that, from a privacy-preserving perspective, are better suited to be stored on the user side. Intuitively, instead of storing user information in a database row, one can employ an anonymous credential system. Then, the row is encoded as an attribute vector and certified by an anonymous credential which is then stored on a user's device. In this setting, users get their attributes certified by a provider acting as an issuer of anonymous credentials. However, the question arises, how providers and users can update their attributes as it is a common process in a system that uses a database. There, the provider would just update some entries in a row associated to a user. For this,

we introduced updatable anonymous credentials (UAC) in [BBDE19] allowing privacy-preserving updates of attributes certified a credential. To do this, the user has to contact the original issuer of the credential and both agree on an update that they want to execute. The result of this is a new credential on update attributes, where the update process does not leak the attributes or the credential to the issuer. The issuer only learns which update was performed, i.e., if the update was a "+7" on a point counter attribute the issuer only learns that it updated some point counter with "+7". With this short example in place, let us describe the roles and processes of an UAC system before we show how we can instantiate an UAC system.

Roles and Protocols

In an UAC system there are users, issuers, and verifiers. Users can get their attributes certified in a credential by an issuer. Hence, issuers are responsible for generating credentials on attributes and verifiers check the validity of credentials with respect to a policy.

To describe the protocols that the different parties (roles) execute, let us expand our subscription example. Here, a user wants to get a credential from an issuer certifying that the user has a valid subscription. For this first issuance of a credential the user shows a recipe of the subscription to validate his assertion of a valid subscription. This can also be realized with an anonymous payment. However, this is outside of the system and for the example we just assume that the issuer can be sure that the user has a valid subscription. Following this, the user and issuer execute a so-called issue protocol. Here, the issuer generates a credential on the subscription end date by encoding it in an attribute called `sub_end`. Additionally, the issuer adds a second attribute to the credential, called `actions`, which is initialized to be 0. The attribute `actions` = 0 is given to any user that joins the system. Next, using the issued credential, the user can authenticate to a verifier via a show protocol to get access to the subscribed service as described above. If the user now performs some predefined actions, e.g., by using a specific feature of the service, the issuer offers the user to update its credentials, i.e., it increments the `actions` attribute by 1. For this, the user and issuer agree on the update (in the form of an update function) and execute an update protocol. The result is a new credential for the user on attributes `sub_end` (unchanged) and `actions` = 1. Furthermore, a verifier can give a discount on other services or features of the service if the user has a valid subscription and has performed more than 50 actions. For this, the user and verifier execute a show protocol in which the user proves that it has a valid subscription and now also proves that its `actions` attribute is greater than 50. This show protocol does not leak any information about the actual `actions` count or the end date of the subscription. In general, the practical features of UAC combined with privacy-preserving protocols seem as though they require heavy cryptographic techniques that are inefficient in real-world applications. The contrary is the case and was shown by a formal analysis of the UAC system and a prototype implementation in [BBDE19; BEHF21].

Efficient Instantiation

In the following, we present how UAC can be efficiently instantiated with modern cryptographic building blocks in which efficient implementations are available. Up to now, we referred to the attributes as certified by a credential. Concretely, in UAC a credential is a

digital signature on a message vector that represents an attribute vector. That means an issuer generates its own public-secret key pair of a digital signature scheme under which it issues credentials. Furthermore, the proofs that the user has to generate, e.g., in a show protocol, are done via a proof system called (non-interactive) zero-knowledge arguments of knowledge. Zero-knowledge arguments of knowledge are systems that generate an efficiently verifiable proof string with two security properties: zero-knowledge and argument of knowledge. Zero-knowledge means that the proof does not leak any information about the secrets of the proof (also called witness), such as the attributes. Argument of knowledge guarantees that no adversarial user can convince a verifier, i.e., generate a valid proof, without having a valid witness. This means if the UAC policy to be proven cannot be satisfied by user's attributes, no adversarial user can generate a valid proof. Hence, the security properties of the proof system protects the privacy of the user and the interests of the verifier.

With these building blocks, in place we can start describing the technical details of issue and update protocols. Since an issue protocol is just a special case of an update protocol we just describe the latter. To get a credential on updated attributes, suppose a user with an existing credential on attribute vector containing `sub_end` and `actions` attributes agreed on an update function that updates the `actions` attribute by "+7". The user prepares this update by sending the issuer a commitment on the updated attribute vector. This commitment does not leak any information about its content and guarantees that the user cannot change the committed values later in the protocol. In case of an issue protocol, the update function just sets the attributes to be issued to its starting values. Next, the user computes a proof that shows that the commitment is correctly formed. This means that it contains attributes of a valid credential and the update was correctly prepared. Then, the issuer checks the proof and, if it is valid, it digitally signs the commitment and sends the result back to the user as his credentials.

2.3 Efficient Verifiable Random Functions without Random Oracles

We developed more efficient constructions of so-called *variable random functions* (VRFs), which can be seen as enhanced digital signature schemes with additional properties. Verifiable random functions play an important role in several applications relevant to Subproject C1 of the CRC-901. Specifically, VRFs are a core building block of the family modern consensus mechanisms called *proof-of-stake*, which are part of AP 1 of Subproject C1. Moreover, VRFs are used in constructions of verifiable distributed public-key distribution systems, such as CONIKS [MBB+15]. Such verifiable distributed public-key distributions systems are relevant to the decentralization of the components for On-the-Fly Computing described in AP 3.2.

The Random Oracle Model

In practical modern cryptography, the so-called *random oracle model* (ROM) introduced by Bellare and Rogaway is often used. In this model, one or several hash functions are modeled as so-called *random oracles* (ROs). A RO can be queried on specific inputs from the domain of the hash functions by all parties that are active in the context of the cryptographic scheme. Each random oracle maintains an initial list that maps hash function

inputs to outputs. Every time the RO is queried on an input x, it checks whether its list contains an entry for x. If this is not the case, it draws an element y from the range of hash functions uniformly at random and stores the mapping $x \mapsto y$ in its list and returns y. If such a mapping $x \mapsto y$ already exists in the list, y is retrieved and returned. Moreover, it is common to allow *programming* the RO in the proof of the security of a cryptographic scheme. That is, inside the proof specific mappings of inputs and outputs may be chosen as long as the distribution of the outputs remains provably indistinguishable from the distribution of a non-programmed oracle. Note that this is a very strong idealization of a cryptographic hash function, which provides not only all standard security properties such as collision resistance or (second) preimage resistance but also many further very strong properties beyond this, such as *programmability* in a security proof, which is not possible for a fixed concrete function such as SHA-3. Unfortunately, it is known that random oracles can not be instantiated in general [CGH04]. Therefore, a concrete practical instantiation of a construction that is only proven secure in the ROM only achieves heuristic security. As a result, from a practical perspective, it would be preferable to have efficient cryptographic schemes that can be proven secure outside the ROM. Similarly, from a theoretical perspective, constructing such cryptographic schemes helps advancing our understanding of what can be achieved outside the ROM and when the ROM is inherently necessary.

New Techniques for Verifiable Random Functions and Further Applications

VRFs are a public-key primitive, where a public *verification key* vk identifies a function $F_{\mathsf{vk}} : X \to Y$ for some domain X and some range Y. However, vk does not allow to efficiently evaluate F_{vk}. The respective *secret key* sk then allows the following two functionalities:

1. Evaluating F_{vk} on any input $x \in X$ and thus obtaining $y = F_{\mathsf{vk}}(x)$ in an efficient way.
2. Generating a proof of correct evaluation π that can be efficiently verified with the help of vk. That is, vk and π together allow to verify that $y = F_{\mathsf{vk}}(x)$ holds without the need to know sk.

Finally, we require that even for an adversarially chosen input $x \in X$ it remains impossible for any efficient algorithm to distinguish $y = F_{\mathsf{vk}}(x)$ from a $y' \in Y$ that is chosen uniformly at random if π is not known.

We developed new verifiable random functions that do not rely on the ROM and are significantly more efficient than previously known constructions in terms of the size of vk, sk and π [JN19; JKN21]. The techniques also turned out to have further applications beyond VRFs. *Identity-Based Encryption* (IBE) is a type of public-key encryption where, there is only a single *master public-key* mpk known to all parties and a respective *master secret-key* msk only known to a trusted third party. Using msk, the trusted third party can then issue *user secret keys* sk for arbitrary identities, e.g., email addresses, and provide the respective users with them. It then suffices to know mpk and a user's identity to encrypt a message for the user such that only that user can decrypt the ciphertext efficiently. Based on the same techniques that we applied in the context of VRFs, we also developed more efficient IBE schemes. In [JKN21], we describe new more efficient IBE schemes with security under assumptions related to the hardness of the discrete logarithm problem in elliptic curve groups and under the learning with errors (LWE) problem, which provides

post-quantum security. Moreover, in [Nie21a] more efficient constructions of IBEs are described.

Verifiable Random Functions with Optimal Tightness

Another aspect of efficiency of cryptographic schemes is called *tightness*. Security in modern cryptography is often proven by reducing the security of the cryptographic scheme to the intractability of some computational problem, such as the discrete logarithm problem, the factorization problem, or the learning with errors problem. However, the *quality* of such reductions can vary in the tightness with which they relate the security of the cryptographic scheme to the intractability of the respective computational problem. That is, applying the reduction to an algorithm $\mathcal{A}$ running in time $t_{\mathcal{A}}$ that breaks the security of a cryptographic scheme with probability $\epsilon_{\mathcal{A}}$ yields an algorithm $\mathcal{B}$that solves the computational problem in time $t_{\mathcal{B}} \geq t_{\mathcal{A}}$ with probability $\epsilon_{\mathcal{B}} \leq \epsilon_{\mathcal{B}}$. In order to achieve efficient cryptographic schemes, we want to construct reductions that have a so-called *loss* $\ell := (t_{\mathcal{B}}/\epsilon_{\mathcal{B}})/(t_{\mathcal{A}}/\epsilon_{\mathcal{A}})$ that is as small as possible and ideally a small constant. Reductions with a small loss have the advantage that strong security guarantees can be achieved while relying on smaller instances of the computational problem. For example, a tight reduction could allow us to use groups of smaller size when relying on the intractability of the discrete logarithm problem, which would yield smaller keys and make algorithms more efficient. This approach is also known as *concrete security* and thoroughly discussed in [BR09]. This raises the natural question of how tight reductions for cryptographic schemes can be. We advanced the state of the art in the research area by providing the first lower bound for the loss of reductions from the security of VRFs to non-interactive hardness assumptions and providing the first construction of a VRF with an accompanying security proof that meets this bound [Nie21b].

2.4 Insider-Resistant Distributed Storage Systems

In recent years, the use of online services has increased significantly. For instance, communicating with friends via social media platforms, sharing videos via YouTube, or shopping online via Amazon. This induces the necessity to store large amounts of data online in such a way that they can be managed and accessed efficiently. Distributed storage systems constitute one of the most natural approaches for the implementation of such a storage. Popular examples include storage solutions offered by Google, Apple, and Amazon. We considered distributed storage systems that are defined as a network consisting of several servers that provide a lookup and update operation. If only a lookup operation but no update operation is provided, we call the system a *distributed information system*.

Since availability and retrievability of the stored data is a key aspect of distributed storage systems, these systems should have various mechanisms in place to protect them against adversarial attacks. One of the biggest threats distributed storage systems are exposed to are crash failures. A server that experiences a crash failure is not available anymore, meaning that it neither responds to any requests nor performs any further operations. Crash failures can be temporary or permanent and can have many causes, such as maintenance work, hardware or software failures, or DoS attacks. Especially crash failures caused by DoS attacks can pose a serious threat, since they usually are unpredictable, hard to prevent,

and can cause the unavailability of a server for some time. Besides crash failures, storage failures also constitute a big threat to distributed storage systems. A server that experiences a storage failure may hold arbitrarily corrupted data in its storage without being aware of that.

While a crash failure can easily be detected using crash failure detectors, this does not hold for massive storage failures. Instead, the distributed storage system needs to implement techniques and methods in order to work correctly despite the existence of servers with storage failures. Storage failures may not only be caused by malicious adversaries, they may also occur due to technical errors, such as disk faults or physical interconnect failures. For instance, in 2008 Amazon's S3 storage service experienced a multi-hour downtime due to a single bit corruption resulting in monetary loss for Amazon and the unavailability of data stored at the S3 storage service.

The predominant approach in distributed storage systems to deal with the threat of failures is to use redundancy and information hiding: The idea behind this is that information that is not only stored at a single server but also replicated on multiple servers is more likely to remain accessible during an attack, in particular if the adversary does not know the storage locations of the redundant data items. For example, if a logarithmic number of copies of each data item is distributed among the servers in the system, and the adversary is not aware of these locations, then it is easy to see that with high probability a copy of each data item is still accessible if the adversary crashes less than half of the servers. However, the situation is completely different when considering an insider adversary, i.e., someone who has complete knowledge of the system and may use this knowledge to crash a large fraction of the servers. Since information cannot be hidden anymore in this case, it seems unavoidable to replicate each data item across more than t servers in order to remain accessible if the system is under an attack that crashes t servers. Unfortunately, in this case the storage overhead becomes very large when considering adversaries that may crash a large fraction of the servers. However, it turns out that this dilemma can be circumvented when using coding, which is one of the key ideas we used in the development of robust storage systems.

Concretely, our goal was to develop distributed information and storage systems that provide efficient lookup and write protocols that work provably correctly despite the existence of an insider adversary that may attack a large fraction of the servers by causing crash failures or a special type of storage failure. In this context, by *efficient* we mean at most polylogarithmic in the number of servers, and by a *large fraction* of attacked servers we mean asymptotically much larger than polylogarithmic, such as $O(n^{1/\log\log n})$, with n being the number of servers, or even up to a constant fraction of all servers. At the same time, we ensure the additional amount of storage required by each server to be limited by at most a logarithmic factor.

Basic IRIS

Our first result was IRIS, a distributed information system that is provably robust against an insider adversary that crashes up to $O(n^{1/\log\log n})$ servers while requiring only a constant storage redundancy. The main innovation in this system is the development of a technique for the efficient encoding of the data items stored in the system with each other using a hierarchical coding strategy that is based on the structure of a k-ary butterfly ($k = \Theta(\log n)$)

and a simple parity-based code. This technique allows to specify a lookup protocol that guarantees to correctly serve each lookup request for any data item with polylogarithmic work at each server and polylogarithmic time, although the adversary may crash up to $O(n^{1/\log\log n})$ servers.

Enhanced IRIS

We then extended IRIS to Enhanced IRIS, which is able to tolerate even up to a constant fraction of all servers to be crashed. Except for the storage redundancy, which increases to a logarithmic factor, Enhanced IRIS still guarantees the same properties as Basic IRIS. The main idea behind this extension of Basic IRIS is to not only use a k-ary butterfly as the underlying topology for the encoding, but to additionally make use of permutations that fulfill certain expansion properties in order to spread the encoding information even further among the servers. IRIS and Enhanced IRIS were presented in [ES15].

RoBuSt

While Basic IRIS and Enhanced IRIS are distributed information systems that provide only a lookup functionality, we later developed RoBuSt, a distributed storage system that provides both lookup and update functionality [ESS14]. More precisely, RoBuSt is a distributed storage system that correctly handles lookup and write requests in polylogarithmic time and with polylogarithmic work despite the existence of an insider adversary that crashes up to $O(n^{1/\log\log n})$ servers. On top of that, RoBuSt requires only a logarithmic storage redundancy. RoBuSt reuses the k-ary butterfly encoding approach introduced with Basic IRIS with the additional ingredient of a clever arrangement of the data items stored in the system into so-called buckets and an appropriate strategy for traversing the buckets efficiently.

OSIRIS

We further strengthened the adversary considered in such a way that it now may not only crash servers, but instead even corrupt the storage of up to $O(n^{1/loglogn})$ servers. Here, we confined ourselves to the corruption of the data stored at the servers while assuming the protocols and main memory of the servers to be reliable. This kind of attack can also be interpreted as a DNS spoofing attack. The main challenge in this setting is that, in contrast to crashed servers, there is no way to efficiently detect corrupted servers. Hence, we needed to add techniques for verifying the validity of data. By appropriately interweaving techniques from the field of authenticated data structures, namely Merkle trees, with techniques developed for IRIS and RoBuSt, we developed OSIRIS, a distributed storage system that is provably robust against an insider adversary that may corrupt the storage of up to $O(n^{1/\log\log n})$ servers. At the same time, OSIRIS correctly answers any set of lookup and update requests in polylogarithmic time and with polylogarithmic work per server while requiring a logarithmic redundancy only [Eik16].

2.5 Construction and Maintenance of Robust Overlays

A key design goal for our OTF market infrastructure is to be open and permissionless. We desire this for two reasons. On the one hand, we want to put little to no boundaries on new parties entering the market to keep it competitive. In particular, established market participants should not be able to prevent new competitors from entering. On the other hand, any participant should be able to leave the market without affecting the functionality, i.e., the network should not be constructed *around* one powerful party that handles a significant amount of market transactions. Therefore, the OTF market infrastructure lends itself to be implemented in a peer-to-peer (P2P) fashion. The P2P approach has proven to be a useful technique for constructing resilient, decentralized systems. In a P2P architecture, the participants (which we will call nodes in the remainder) are connected via the Internet and form a logical network topology, also known as an overlay network or simply overlay. Within the overlay, each node has a logical address and logical links that allow it to search and store information in the network. Ideally, the topology has no single point of failure, so nodes can leave the network without disrupting the functionality. Furthermore, it is designed to scale with any number of nodes.

A fundamental requirement for all applications built on P2P networks is reliable communication between the nodes, i.e., each node should be able to send a message to another node at all times. Of course, this also holds for our OTF market infrastructure. Ensuring reliable communication is complicated by the fact that in every large-scale system, errors and attacks are the rule rather than the exception. Together with the fact that nodes may frequently leave or enter the system on their own accord, this implies a massive amount of so-called *churn*, i.e., changes in the set of nodes. Therefore, we investigated robust distributed protocols that maintain connected overlays despite heavy churn.

Throughout our work, we used the de-facto standard model for P2P algorithms. We assume that time proceeds in synchronous[16] rounds and observe a dynamic set of nodes $\mathcal{V} := (V_0, V_1, \dots)$ such that V_t is the set of nodes in round t. Each node is identified by a unique and immutable identifier denoted by ID. A node $u \in V_t$ can send a message to a node $v \in V_t$ only if it knows the ID of node v. In a real-world network, these IDs could, e.g., be the nodes' IP addresses. This results in series of graphs $\mathcal{G} := (G_0, G_1, \dots)$ with $G_i = (V_i, E_i)$ and $E_t := \{(u, v) \mid u \text{ knows the ID of } v \text{ in round } t\}$. We assume that a node can create edges to $O(\log n)$ different nodes in each round and can send $O(\text{polylog}\, n)$ bits via each edge.

Our research went in two directions that complement each other. First, we asked ourselves how to efficiently construct a robust overlay from any initial topology. Given any connected graph of n nodes representing our overlay, transform it into a low-diameter and high-expansion network. This protocol can be executed periodically to let the overlay recover from heavy but uncoordinated churn. Second, we assumed that the network already has a suitable topology but is attacked by a powerful adversary. This adversary tries to strategically disable nodes with crucial positions within the overlay. In this situation, the adversary's knowledge of the system and the system's reaction time, i.e., whether or not the nodes can detect if they get attacked, are crucial to the success probability of our defenses. We developed a nuanced model to study several types of adversaries and presented competitive protocols that are safe against powerful adversaries.

[16] Synchronicity is a standard assumption as nodes need to react to the adversary's changes promptly.

Result	Runtime	Init. Topology	Communication[a]
[AAC+05]	$O(d^b + \log^2 n)$ w.h.p	Any	$O(\log n)$
[GHSS17]*	$O(\log^2 n)$	Any	$O(d \log n)$
[GHS19]*	$O(\log^{3/2} n)$ w.h.p	Any	$O(d \log n)$
[GHSW21]*	$O(\log n)$ w.h.p	Any	$O(d \log n)$

[a] Number of messages per node and round.
[b] d denotes the initial graph's degree.
* Supported by the CRC 901.

Table 1: *An overview of the overlay construction algorithms.*

Fast Construction of Overlays

To the best of our knowledge, the first overlay construction algorithm with polylogarithmic time and communication complexity that can handle (almost) arbitrary initial states has been proposed by Angluin et al. [AAC+05]. Here, the authors assume a weakly connected initial graph of degree d. If in each round, each node can send and receive at most d messages, and new edges can be established by sending node identifiers, their algorithm transforms the graph into a binary search tree of depth $O(\log n)$ in $O(d + \log^2 n)$ time, w.h.p. A low-depth tree can easily be transformed into many other topologies, and fundamental problems such as sorting or routing can be easily solved from such a structure. This idea has sparked a line of research investigating how quickly such overlays can be constructed. In the context of the CRC, we contributed three results. First, Gmyr et al. presented a deterministic $O(\log^2 n)$ time algorithm [GHSS17]. The algorithm's key idea is to maintain a series of so-called *supernodes*, which are groups of nodes that act in coordination. The algorithm operates in phases of $O(\log(n))$ rounds where each supernode merges with at least one of its neighboring supernodes in each phase. Thus, the size of each supernode, i.e., the number of nodes it contains, grows by a constant factor in each phase. This results in a deterministic runtime of $O(\log^2(n))$. Using randomization, we further improved this procedure to time $O(\log^{3/2} n)$, w.h.p. [GHS19]. Our main idea was to merge several clusters of supernodes to increase the growth in each phase. Finally, we further optimized the runtime to the optimal value of $O(\log(n))$ in [GHSW21]. This approach is different from *all* previous algorithms in that it does *not* use any form of clustering to contract large portions of the graph into supernodes. On a high level, our algorithm progresses through $O(\log n)$ iterations, where the next graph is obtained by establishing random edges on the current graph. These random edges are simply sampled by constant-length random walks, resulting in an extremely simple yet fast algorithm. Table 1 provides an overview of all contributions.

Efficient Maintenance of Overlays

Drees et. al developed our first approach to handling high churn in [DGS16]. The core idea was to use random walks to reorganize the network continuously. However, instead of just using random walks in a standard fashion, which would take $\Omega(\log n)$ communication rounds in graphs of polylogarithmic degree to sample nodes uniformly at random, they

combine random walks with *pointer jumping*. Pointer jumping, i.e., letting a node introduce its neighbors to its neighbors, is a well-known technique in the area of parallel computing, but to great surprise, it seems that it has never been combined with random walks so far. This rather simple trick exponentially improves the running time needed to sample nodes uniformly at random via random walks. We refer to the technique of combining random walks with pointer jumping to sample nodes from a network as *rapid node sampling*.

Based on rapid node sampling, Drees et al. developed algorithms that maintain the connectivity of a network under heavy churn and DoS attacks. Their algorithm organizes the nodes of a network into an expander and maintains connectivity under adversarial churn by an omniscient adversary with a constant churn rate. An important assumption underlying this result is that a node that is prescribed to leave the network by the adversary does not have to leave immediately but can remain in the network for another $O(\log \log n)$ rounds.

On the flip side, rapid node sampling cannot be used if one wants to grant the adversary access to even more recent information than $O(\log \log n)$ rounds. To overcome this restriction, Götte et al. proposed a trade-off in[GVS19]. This trade-off comes in the form of a (a, b)-late omniscient adversary that has almost up-to-date information about the network topology, but it is more outdated concerning all other aspects. In particular, it has full knowledge of the topology after a rounds and complete knowledge of messages, internal states, etc., after b rounds. In the real world, an adversary with similar properties could, e.g., be an agency eavesdropping on Internet exchange points. They can see *who* communicates based on the involved IP addresses but cannot decrypt the messages (or take longer to decrypt them).

The main contribution is a distributed overlay maintenance algorithm that completely rearranges the network every 2 rounds and can handle a $(2, O(\log n))$-late adversary. Furthermore, the algorithm allows routing a message to a logical address $p \in [0, 1)$ within $O(\log n)$ rounds. The algorithms are randomized and the results hold *w.h.p.* Instead of a regular expander, they use a structured overlay topology, namely, an extension of the Linearized DeBruijn Graph presented in Richa et al. [RSS11]. Götte et al. present a robust algorithm that minimizes the number of messages sent in every step. The approach uses several structural properties of the overlay as well as a careful analysis of non-independent events to ensure fast reconfiguration of the network.

Table 2 provides an overview of our results and compares them to previous and concurrent works. As one can see, our results compare favorably with regard of the churn rate and adversarial knowledge they tolerate.

3 Impact and Outlook

With our research on reputation systems and anonymous credentials we contributed significantly to the rapidly increasing research on cryptographic privacy-preserving techniques. Given the dramatic growth of online services, web shops, social media apps, and other data demanding tools these techniques will become ever more important in the future. Our research certainly had significant impact on basic scientific research. But the more important contribution of our research is probably in reducing the gap between theory and practice. Although many advanced cryptographic privacy-preserving techniques exist, they

Paper	Lateness[a]	Churn Rate[b]	Immediate
[AS18]	$(O(\log\log n), O(\log\log n))$	$(\alpha n, O(\log\log n))$	Yes
[AMM+13]	$(O(\log n), O(\log n))$	$\left(O\left(\frac{n}{\log n}\right), O(\log n)\right)$	Yes
[DGS16]*	$(O(\log\log n), O(\log\log n))$	$(n - \frac{n}{\log n}, O(\log\log n))$	No[c]
[GVS19]*	$(2, O(\log n))$	$(\alpha n, O(\log n))$	Yes

[a] An adversary is (a, b)-late if it has full knowledge of the topology after a rounds and complete knowledge of all messages after b rounds.
[b] The churn rate is (C, T) if the adversary can perform C join/leaves in T rounds.
[c] Nodes remain in the network for additional $O(\log\log n)$ rounds.
* Supported by the CRC 901.

Table 2: *Overview of different models in the literature*

are rarely implemented in prototypes or even used in commercial applications. This is mainly due to two factors: advanced cryptographic techniques are often believed to be too inefficient and cumbersome, and cryptographic techniques are difficult to implement from scratch. To overcome these misconceptions and impediments we complemented our basic research by two more applied approaches:

1. Based on our research on updatable credentials we designed a so-called incentive systems and implemented it from scratch.
2. We built a cryptographic open-source Java library, called cryptimeleon, providing basic cryptographic primitives that are the backbone of many privacy-preserving techniques.

Since the incentive system is discussed in detail in the section on transfer project T2, we concentrate on cryptimeleon. The library allows users to build complex privacy-preserving primitives in the so-called bilinear group setting, currently the most powerful and efficient setting for cryptography (although not post-quantum secure, see below). It provides a general framework for the construction of primitives and a number of important basic primitives such as hash functions, pseudorandom functions and Schnorr-type zero-knowledge proofs. A detailed description of the library can be found in [BEHF21]. Although cryptimeleon is mainly targeted towards researchers in cryptography (which already use it in increasing numbers) we believe that it can also form the foundation for a library targeted at more general users.

Our research on reputation systems raises many questions for future research, e.g., how to realize such systems in a decentralized form. But from our perspective, the most pressing problem is to bring many more privacy-preserving techniques to the post-quantum world. Ever since the seminal algorithm of Peter Shor, we know that currently used cryptographic techniques are susceptible to quantum attacks. The development of quantum computers has seen tremendous progress in the last years. Although it is still unclear if and when quantum computers will be built on which Shor's algorithm can be implemented, we have to prepare our security infrastructure for this event: hence the standardization efforts for post-quantum secure cryptography by the NIST and other organizations worldwide.

For basic cryptographic primitives such as encryption schemes and digital signatures, we know many (hopefully) post-quantum secure constructions. For more advanced privacy-preserving techniques, such as reputation systems and anonymous credentials, the situation is quite different. Additionally, post-quantum cryptography tends to be technically much more challenging than classical cryptography. Therefore, it is even more urgent for cryptographers to provide users with easy to use post-quantum secure cryptographic libraries. In the context of verifiable random functions, our work has answered fundamental open questions, but also raised some further questions that are still open. While our constructions of VRFs are already much more efficient than previous constructions, they are still much less efficient than constructions in the ROM. Hence, an important open question is whether there are VRFs that are secure (under standard assumptions) in the standard model and that are as efficient as VRFs whose security proof requires the ROM. Since our techniques used to construct VRFs were also applicable to IBE, one can similarly ask the question whether standard model IBEs can be as efficient as IBEs that are proven secure in the ROM. One can also ask whether there exist IBEs that are secure in the standard model and as efficient as schemes that are proven secure in the ROM. With respect to tightness of security reductions for VRFs, our construction is proven secure under a so-called q-type assumption. These types of assumption have the negative aspect that they get stronger the larger q becomes [Che10]. For these reasons q-type assumptions are not considered standard assumptions. Thus, it would be preferable to achieve tightness with a security proof that relies on a standard assumption. Whether there are VRFs that can be proven secure under a standard assumption and achieve optimal tightness is another fundamental research question in this domain.

Our results on overlays and data storage are (asymptotically) optimal or at least very close to their optimal solution, i.e., within polylogarithmic factors, concerning time and message complexity. Further, our random walk-based algorithms are heavily inspired by those used in practice in big P2P networks such as the one of Bitcoin, giving them a sound theoretical foundation. However, throughout all our contributions, we only considered benign failures of nodes. The nodes may unexpectedly crash but generally behave correctly and follow the protocol. For example, they do not intentionally alter any data item they store or introduce invalid identifiers to the system. So, a natural follow-up question is how our algorithms could also be extended to tolerate nodes exhibiting this and other malicious behavior. More precisely, we want to consider so-called byzantine nodes that deviate from the protocol and behave arbitrarily. Such nodes are not a niche phenomenon but are a very common threat in internet-scale applications: First, not everyone connected to the internet is trustworthy and may be controlled by an adversarial party with their own malicious goals. Second, even honest and well-protected nodes may be hacked by an attacker if the stakes are high enough.

A typical way to deal with byzantine nodes is to use so-called quorums. These are (randomly selected) subsets of nodes that act in coordination. In particular, the honest nodes outnumber the byzantine nodes in each quorum, so potentially malicious actions can be *overruled*. However, these quorums introduce a massive overhead because data must be replicated and passed to all members. Further, for all messages between quorums, the members of each quorum need to agree on the content. Reaching an agreement in the presence of byzantine nodes is notorious for being time and/or bandwidth-consuming as it typically requires all-to-all communication. To add insult to injury, using quorums is

a proactive approach. This means that we pay the high cost of maintaining the quorums even if there is no byzantine behavior (and we could have used our original algorithms). With our current models and assumptions, there seems to be no way around using quorums if one looks for proactive approaches. Because of this, we want to shift our attention to *reactive* approaches and design algorithms that are at least resource competitive. This means that the cost of executing the algorithm, e.g., the latency and message complexity, is directly related to the severity of the byzantine attack. The caveat of this approach is that the system needs some mechanism to recover from an attack, e.g., to retrieve dropped data items or reconnect parts of the overlay. This mechanism could be implemented through a trusted third party, e.g., a cloud provider, that offers these services for a price. Whenever the honest nodes notice byzantine behavior, they query the third party to repair the system. For example, this third party could store data items. If an honest node detects that data items have been dropped or tampered with in the P2P network, it retrieves a *fresh* copy from the trusted party. Ideally, the number of queries is linear in the number of tampered data items. However, there are many open questions about this approach that require nuanced answers: for instance, how the cost of queries is measured and what the exact capabilities of the trusted party are. In particular, the trusted party must be implementable in practice and not be too powerful to keep the problem interesting. We plan to investigate all these questions in the future.

Bibliography

[AAC+05] Angluin, D.; Aspnes, J.; Chen, J.; Wu, Y.; Yin, Y.: Fast Construction of Overlay Networks. In: *Proc. of the 17th Annual ACM Symposium on Parallelism in Algorithms and Architectures (SPAA)*. 2005, pp. 145–154

[AMM+13] Augustine, J.; Molla, A. R.; Morsy, E.; Pandurangan, G.; Robinson, P.; Upfal, E.: Storage and search in dynamic peer-to-peer networks. In: *Proc. of SPAA*. 2013, pp. 53–62

[AS18] Augustine, J.; Sivasubramaniam, S.: Spartan: A Framework For Sparse Robust Addressable Networks. In: *Proc. of IPDPS*. 2018, pp. 1060–1069

[BB18] Blömer, J.; Bobolz, J.: Delegatable Attribute-Based Anonymous Credentials from Dynamically Malleable Signatures. In: *ACNS 18: 16th International Conference on Applied Cryptography and Network Security*. Vol. 10892. Lecture Notes in Computer Science. Springer, 2018, pp. 221–239

[BBB+18] Bemmann, K.; Blömer, J.; Bobolz, J.; Bröcher, H.; Diemert, D.; Eidens, F.; Eilers, L.; Haltermann, J.; Juhnke, J.; Otour, B.; Porzenheim, L.; Pukrop, S.; Schilling, E.; Schlichtig, M.; Stienemeier, M.: Fully-Featured Anonymous Credentials with Reputation System. In: *Proceedings of the 13th International Conference on Availability, Reliability and Security, ARES*. ACM, 2018, 42:1–42:10

[BBDE19] Blömer, J.; Bobolz, J.; Diemert, D.; Eidens, F.: Updatable Anonymous Credentials and Applications to Incentive Systems. In: *ACM CCS 2019: 26th Conference on Computer and Communications Security*. ACM Press, 2019, pp. 1671–1685

[BBS04] Boneh, D.; Boyen, X.; Shacham, H.: Short Group Signatures. In: *Advances in Cryptology – CRYPTO 2004*. Vol. 3152. Lecture Notes in Computer Science. Springer, 2004, pp. 41–55

[BCC+09] Belenkiy, M.; Camenisch, J.; Chase, M.; Kohlweiss, M.; Lysyanskaya, A.; Shacham, H.: Randomizable Proofs and Delegatable Anonymous Credentials. In: *Advances in Cryptology – CRYPTO 2009*. Vol. 5677. Lecture Notes in Computer Science. Springer, 2009, pp. 108–125

[BEHF21] Bobolz, J.; Eidens, F.; Heitjohann, R.; Fell, J.: *Cryptimeleon: A Library for Fast Prototyping of Privacy-Preserving Cryptographic Schemes*. Cryptology ePrint Archive, Report 2021/961. https://eprint.iacr.org/2021/961. 2021

[BEJ18] Blömer, J.; Eidens, F.; Juhnke, J.: Practical, anonymous, and publicly linkable universally-composable reputation systems. In: *Cryptographers' Track at the RSA Conference*. Springer. 2018, pp. 470–490

[BJK15] Blömer, J.; Juhnke, J.; Kolb, C.: Anonymous and Publicly Linkable Reputation Systems. In: *FC 2015: 19th International Conference on Financial Cryptography and Data Security*. Vol. 8975. Lecture Notes in Computer Science. Springer, 2015, pp. 478–488

[BPS+17] Busom, N.; Petrlic, R.; Sebé, F.; Sorge, C.; Valls, M.: A privacy-preserving reputation system with user rewards. In: *Journal of Network and Computer Applications* 80 (2017)

[BR09] Bellare, M.; Ristenpart, T.: Simulation without the Artificial Abort: Simplified Proof and Improved Concrete Security for Waters' IBE Scheme. In: *Advances in Cryptology - EUROCRYPT 2009*. Vol. 5479. Lecture Notes in Computer Science. Springer, 2009, pp. 407–424.

[Can01] Canetti, R.: Universally Composable Security: A New Paradigm for Cryptographic Protocols. In: *42nd Annual Symposium on Foundations of Computer Science*. IEEE Computer Society Press, 2001, pp. 136–145

[CDD17] Camenisch, J.; Drijvers, M.; Dubovitskaya, M.: Practical UC-Secure Delegatable Credentials with Attributes and Their Application to Blockchain. In: *ACM CCS 2017: 24th Conference on Computer and Communications Security*. ACM Press, 2017, pp. 683–699

[CG08] Camenisch, J.; Gross, T.: Efficient attributes for anonymous credentials. In: *ACM CCS 2008: 15th Conference on Computer and Communications Security*. ACM Press, 2008, pp. 345–356

[CGH04] Canetti, R.; Goldreich, O.; Halevi, S.: The random oracle methodology, revisited. In: *J. ACM* 51 (2004), no. 4, pp. 557–594.

[Che10] Cheon, J. H.: Discrete Logarithm Problems with Auxiliary Inputs. In: *J. Cryptol.* 23 (2010), no. 3, pp. 457–476.

[CKS10] Camenisch, J.; Kohlweiss, M.; Soriente, C.: Solving Revocation with Efficient Update of Anonymous Credentials. In: *SCN 10: 7th International Conference on Security in Communication Networks*. Vol. 6280. Lecture Notes in Computer Science. Springer, 2010, pp. 454–471

[CL01] Camenisch, J.; Lysyanskaya, A.: An Efficient System for Non-transferable Anonymous Credentials with Optional Anonymity Revocation. In: *Advances in Cryptology – EUROCRYPT 2001*. Vol. 2045. Lecture Notes in Computer Science. Springer, 2001, pp. 93–118

[CL02] Camenisch, J.; Lysyanskaya, A.: Dynamic Accumulators and Application to Efficient Revocation of Anonymous Credentials. In: *Advances in Cryptology – CRYPTO 2002*. Vol. 2442. Lecture Notes in Computer Science. Springer, 2002, pp. 61–76

[CL04] Camenisch, J.; Lysyanskaya, A.: Signature Schemes and Anonymous Credentials from Bilinear Maps. In: *Advances in Cryptology – CRYPTO 2004*. Vol. 3152. Lecture Notes in Computer Science. Springer, 2004, pp. 56–72

[CL19] Crites, E. C.; Lysyanskaya, A.: Delegatable Anonymous Credentials from Mercurial Signatures. In: *Topics in Cryptology – CT-RSA 2019*. Vol. 11405. Lecture Notes in Computer Science. Springer, 2019, pp. 535–555

[CLNR14] Camenisch, J.; Lehmann, A.; Neven, G.; Rial, A.: Privacy-Preserving Auditing for Attribute-Based Credentials. In: *ESORICS 2014: 19th European Symposium on Research in Computer Security, Part II*. Lecture Notes in Computer Science. Springer, 2014, pp. 109–127

[DGS16] Drees, M.; Gmyr, R.; Scheideler, C.: Churn- and DoS-resistant Overlay Networks Based on Network Reconfiguration. In: *Proc. of SPAA*. 2016, pp. 417–427

[Eik16] Eikel, M.: Insider-resistant distributed storage systems. PhD thesis. University of Paderborn, 2016.

[ES15] Eikel, M.; Scheideler, C.: IRIS: A Robust Information System Against Insider DoS Attacks. In: *ACM Trans. Parallel Comput.* 2 (2015), no. 3, 18:1–18:33

[ESS14] Eikel, M.; Scheideler, C.; Setzer, A.: RoBuSt: A Crash-Failure-Resistant Distributed Storage System. In: *Principles of Distributed Systems - 18th International Conference, (OPODIS)*. 2014, pp. 107–122

[GHS19] Götte, T.; Hinnenthal, K.; Scheideler, C.: Faster Construction of Overlay Networks. In: *International Colloquium on Structural Information and Communication Complexity (SIROCCO)*. Springer. 2019, pp. 262–276

[GHSS17] Gmyr, R.; Hinnenthal, K.; Scheideler, C.; Sohler, C.: Distributed Monitoring of Network Properties: The Power of Hybrid Networks. In: *Proc. of the 44th International Colloquium on Automata, Languages, and Programming (ICALP)*. 2017, 137:1–137:15

[GHSW21] Götte, T.; Hinnenthal, K.; Scheideler, C.; Werthmann, J.: Time-Optimal Construction of Overlay Networks. In: *PODC '21: ACM Symposium on Principles of Distributed Computing*. ACM, 2021, pp. 457–468.

[GVS19] Götte, T.; Vijayalakshmi, V. R.; Scheideler, C.: Always be Two Steps Ahead of Your Enemy. In: *2019 IEEE International Parallel and Distributed Processing Symposium, IPDPS*. IEEE, 2019, pp. 1073–1082.

[JKN21] Jager, T.; Kurek, R.; Niehues, D.: Efficient Adaptively-Secure IB-KEMs and VRFs via Near-Collision Resistance. In: *Public-Key Cryptography - PKC 2021 - 24th IACR*. Vol. 12710. Lecture Notes in Computer Science. Springer, 2021, pp. 596–626.

[JN19] Jager, T.; Niehues, D.: On the Real-World Instantiability of Admissible Hash Functions and Efficient Verifiable Random Functions. In: *Selected Areas in Cryptography - SAC 2019*. Vol. 11959. Lecture Notes in Computer Science. Springer, 2019, pp. 303–332.

[MBB+15] Melara, M. S.; Blankstein, A.; Bonneau, J.; Felten, E. W.; Freedman, M. J.: CONIKS: Bringing Key Transparency to End Users. In: *24th USENIX Security Symposium, USENIX Security 15*. USENIX Association, 2015, pp. 383–398.

[MSBM22] Mir, O.; Slamanig, D.; Bauer, B.; Mayrhofer, R.: *Practical Delegatable Anonymous Credentials From Equivalence Class Signatures*. Cryptology ePrint Archive, Report 2022/680. `https://eprint.iacr.org/2022/680`. 2022

[NF06] Nakanishi, T.; Funabiki, N.: A Short Verifier-Local Revocation Group Signature Scheme with Backward Unlinkability. In: *Advances in Information and Computer Security*. Vol. 4266. LNCS. Springer, 2006, pp. 17–32

[Nie21a] Niehues, D.: More Efficient Techniques for Adaptively-Secure Cryptography. PhD thesis. University of Wuppertal, Germany, 2021.

[Nie21b] Niehues, D.: Verifiable Random Functions with Optimal Tightness. In: *Public-Key Cryptography - PKC 2021 - 24th IACR*. Vol. 12711. Lecture Notes in Computer Science. Springer, 2021, pp. 61–91.

[PS16] Pointcheval, D.; Sanders, O.: Short Randomizable Signatures. In: *Topics in Cryptology – CT-RSA 2016*. Vol. 9610. Lecture Notes in Computer Science. Springer, 2016, pp. 111–126

[RSS11] Richa, A. W.; Scheideler, C.; Stevens, P.: Self-Stabilizing De Bruijn Networks. In: *Proc. of SSS*. 2011, pp. 416–430

[ZWC+16] Zhai, E.; Wolinsky, D. I.; Chen, R.; Syta, E.; Teng, C.; Ford, B.: AnonRep: Towards Tracking-Resistant Anonymous Reputation. In: *NSDI*. 2016, pp. 583–596