Federal Office
for Information Security

Towards Secure Blockchains

Concepts, Requirements, Assessments

Editorial

Blockchain is currently one of the most widely discussed topics in the area of information technology. This technology for distributed data storage originated from the cryptocurrency Bitcoin, which became famous especially due to the record highs its market value attained in 2017. Based on its promise of preventing manipulation of data on a purely technical level using its decentralised structure, offering maximum transparency and replacing intermediaries within business processes, many ideas for applying blockchain technology in fairly different areas have been developed in recent years.

Politics has also increasingly taken up blockchain technology. For instance, the term blockchain is used several times within the coalition agreement of the 19th election period of the German parliament dating from 2018, and the German federal government has set itself the target of developing a comprehensive blockchain strategy until summer 2019.

As with many topics receiving high attention by the media, it is important to take into account the technical foundations when discussing blockchain. In particular, this holds for IT security, since one often hopes to increase security by using blockchain. However, numerous incidents, which have caused damages in the millions, show that the mere use of blockchain does not imply that one can forgo measures for establishing IT security.

As a first step for pointing out the basic issues in this area, the Federal Office for Information Security (BSI) published a list of key points in February 2018. As a result of numerous discussions with different stakeholders offering blockchain solutions or considering their use, the BSI found a more in-depth analysis of the security features of blockchain was required.

Consequently, the analysis on the following pages presents blockchain technology in detail and extensively studies its aspects relevant to IT security. It also assesses to what extent blockchain technology is able to achieve the security properties ascribed to it and how it may be evaluated within the current legal framework.

This document thus supports developers and potential users of blockchain solutions in performing a well-founded assessment of risks and in taking IT security into account from the start. Furthermore, the dynamic development of blockchain technology offers the possibility of using the results of this analysis as a basis for future discussions on both the national and international level. With blockchain—as with other topics in IT security—the BSI thus strives to shape information security for government, business and society.

I wish you an insightful reading experience.

Arne Schönbohm, president of the Federal Office for Information Security (BSI)

Arne Schönbohm

Arne Schönbohm

Executive Summary

Blockchain is a new technology for data management which achieves the greatest possible independence from a central party using distributed storage of data in combination with cryptographic routines and additional technical measures. On the one hand, this serves to make targeted manipulations significantly harder. On the other hand, this feature of blockchains gave rise to proposals for their use in different applications in which several parties of conflicting interests are involved and cannot agree on a central party controlling the application. In other applications which do feature a central party, the use of blockchains was proposed for improving efficiency by making parties interact directly with each other.

Due to the expectations associated with blockchain technology, it has currently become a hot topic intensely studied by a plethora of parties from science, business and administration. Use of the technology is being discussed and studied in many areas. At the present time, the only area to practically employ blockchains to a somewhat larger extent though is that of finance, in particular cryptocurrencies.

In this document, the BSI studies blockchains primarily from the point of view of IT security, but also considers further implications of the basic technical design, e.g. on efficiency or compliance with requirements from data protection. In general, blockchains offer benefits as compared to databases in terms of availability and resilience against misuse, whereas they show disadvantages in terms of confidentiality and efficiency.

The mere use of blockchain does not solve IT security problems. On the contrary, well-known problems like hardware and software security continue to exist. In addition, new attack vectors targeting different components of the system arise. Besides consensus mechanisms, which ensure consistency of the distributed data copies, and smart contracts, which allow executing programs within the blockchain network, attacks may for instance concern external interfaces for entering and reading out data. Concrete incidents bear witness to the fact that the possibilities for attacks are not only theoretical.

It is important to note that blockchain technology exhibits a wide array of designs. Features often used for classification include the visibility of data (reading permissions) as well as the updatability (writing permissions) of the blockchain. However, a number of approaches, which may strongly differ among each other, likewise exist for consensus mechanisms and smart contracts. They correlate with the basic design of a blockchain to some extent, but also offer further degrees of freedom. If blockchain is to be used in a concrete application, it is hence necessary to carefully analyse which design is most suitable. The design of Bitcoin, which receives most attention in the media, will probably not be reasonable for most applications.

In the area of consensus mechanisms, the algorithm proof-of-work is predominant in public discussion. Proof-of-work is used among others by Bitcoin and criticised in particular for its enormous energy consumption. The algorithm allows keeping data consistent in a setting without authentication of individual parties while at the same time preventing manipulations. What often goes unnoticed is the fact that blockchains managing permissions more strictly and authenticating individual parties, which seem to be necessary for many applications, allow using so-called message-based consensus mechanisms, which are significantly more efficient and well-studied.

Several blockchain systems allow using smart contracts, which aim to make possible the tamper-proof execution of contracts between parties unknown to or mistrusting each other. Smart contracts are not equivalent to legal contracts though, and many contractual concepts cannot be represented via a smart contract. In addition, analyses of existing contracts revealed a large

number of security problems. They range from bugs in the code—which cannot be corrected for technological reasons—and manipulable random numbers to a lack of authenticity for data which is entered from the real world and processed in the contract. In order to handle smart contracts responsibly, it is imperative that these limitations and vulnerabilities be taken into account.

Since the security of blockchains strongly relies on the cryptographic algorithms used, these algorithms have to be carefully selected in order to attain the desired security level with respect to the security goals integrity, authenticity and confidentiality. For detailed recommendations, the reader is referred to several Technical Guidelines published by the BSI and cited in the text. Most stakeholders in the area of blockchain have paid hardly any attention to long-term security of data so far. In order to protect sensitive data for the long term, it is necessary to have at one's disposal measures for replacing cryptographic algorithms which are no longer secure. When doing so, it is imperative to bear in mind that replacing cryptographic routines does not automatically preserve the original security properties for old data. Besides cryptographic routines, the security properties of consensus mechanisms likewise need to be well-understood and taken into account. These aspects should hence be considered from the start in blockchain applications.

Legal questions concerning blockchains result among other things from the lack of a central, legally responsible party during normal operation, which has various implications. This topic is currently the subject of controversial discussion. Problems connected to data protection, like the implementation of requirements as defined in the General Data Protection Regulation (GDPR), arise from the transparency and tamper resistance of blockchains, which are among their main features. Such problems are also intensely studied at the moment.

Extensive and creative research projects targeting a broad range of technical limitations and problems of blockchain technology are being conducted at present. Current research topics include e.g. scalability, efficiency, pseudonymity and confidentiality. At the moment it is not possible to estimate when and if significant improvements may result from these projects.

A further issue to be noted is the lack of standards in the blockchain area. This leads to incompatibility among different blockchains and to a quite confusing plethora of solutions, which make choosing a concrete product for the longer term difficult for users. Regarding IT security, the BSI provides the document at hand as a decision-making aid and as a well-founded basis for future discussions.

Table of contents

Introduction

For some time now, blockchains have been known not only to experts, but also to a broader public thanks to frequent references in the media. The technology arose in 2009 in conjunction with Bitcoin, the first successful cryptocurrency that has given rise to many others. For some years now, the use of blockchains has also been proposed and tested in numerous other fields.

In particular with respect to applications outside cryptocurrencies, blockchains are a comparatively recent technology. Great hopes are swiftly ascribed to their use, but standardised technical approaches have not yet been established.

As with any new technology, the principle of *security by design* should be taken into account with blockchain as well. For this reason, in February 2018 the BSI published a list of key points indicating the general framework, requirements and measures which are necessary for a secure use of the technology [1]:

- Blockchain on its own does not solve IT security problems (cf. chapters 5 and 7).
- Choosing a suitable blockchain model is important (cf. chapters 1 and 2).
- When designing blockchains, security aspects must be taken into account early on (cf. chapters 3, 5, 6 and 10).
- Sensitive data requiring long-term protection must be specially protected in a blockchain (cf. chapter 6).
- Standardised security levels for blockchains must be defined and enforced (cf. chapter 12).

The document at hand addresses the same issues, but does so in much greater depth. It does not intend to explain the subject to the general public, but is primarily aimed at potential users that consider the use of blockchain and have acquired basic technical knowledge. It aims to provide a structured and comprehensive survey of the topic, focusing especially on the aspects linked to IT security. The document intends to enable readers to identify the questions of relevance to them, to evaluate their projects in terms of IT security and, if applicable, to derive concrete measures for the secure operation of blockchain solutions.

The contents of the document are structured as follows: The first part starts with an abstract description of the technology, which is complemented by some examples, and introduces general technical terms. Subsequently, a fundamental assessment of blockchains in terms of IT security is performed before the document goes on to present the technical core components of consensus mechanisms and smart contracts in more detail. This includes a comparison of different approaches as well as statements on security.

The following part intensely covers the security aspects of the topic. It provides general indications regarding the use of cryptographic routines. On the one hand, this concerns the choice of algorithms for attaining certain security guarantees, but on the other hand, more concrete indications for a secure implementation are also furnished. The following section is devoted to the topic of long-term security in blockchains, i.e. whether and how one can react to cryptographic routines becoming vulnerable to new types of attacks, and hence no longer being secure. Finally, a plethora of known attacks targeting blockchains are discussed and countermeasures are presented, if applicable.

The third part highlights legal questions. Apart from a general legal assessment, it covers aspects of data protection in particular.

Finally, the last part examines the practical application of blockchains in more detail. To this end, some frequently mentioned application scenarios are abstractly analysed and the aspects relevant in terms of IT security are emphasised. Subsequently, the document presents a number of approaches and ideas that are being worked out and discussed as a further development and generalisation of blockchains using the umbrella term of distributed ledger technology (DLT). Besides extensions of the data structure and combinations with other new technologies, interoperability is an important topic of research.

Technical terms from cryptography are explained in a glossary at the end of the document. The end of each chapter features a summary of the most important statements.

Part I Basics

1 Definitions and taxonomy

In order to reach a common understanding of the ingredients of the blockchain technology, and to provide a basis for the subsequent discussion, the first sections of this chapter will introduce important technical terms and basic mechanisms in the field of blockchains, and describe different approaches to their classification. To illustrate these fundamental ideas, the chapter concludes with a presentation of some well-known examples.

1.1 Basic terms

The basic idea of the blockchain technology is based on the more general notion of what is known as distributed ledger technology (a distributed digital analogue of the classical ledger in bookkeeping). This is a technique for distributed data storage in a peer-to-peer network (P2P-network), in which the nodes decide by agreement (consensus) on updating the stored data. The data may, for instance, contain balances of cryptocurrency accounts, proofs of origin of traded goods, or, more abstractly, the current states of smart contracts.

There is neither a central communication control nor a central data storage. Every network node manages its own local copy of the complete data set, to which it may add data. A suitable consensus mechanism ensures that the distributed data in each node is kept up-to-date and in agreement with the others, so that the distributed ledger remains in a consistent state at all times.

The security measures for network access control, data structure and consensus protocol all use cryptographic algorithms in order to reach the desired security goals (in particular integrity and authenticity). The rules for validating, storing and using the data (business logic) are inherently coded into the data itself. The network enforces these rules in an automated process.

In particular, in blockchain technology, all data—in the form of so-called transactions—are validated within the network and grouped into blocks. New blocks are chained to their respective predecessor by a tamper-proof cryptographic link, thereby establishing a chronological order of all transactions. This results in a continually growing chain of data blocks, the *blockchain*, as a special case of a distributed ledger (see figure 1).

Blockchain technology consists of five fundamental building blocks (see figure 2):

- peer-to-peer network
- blockchain as a data structure
- consensus building
- business logic
- cryptography

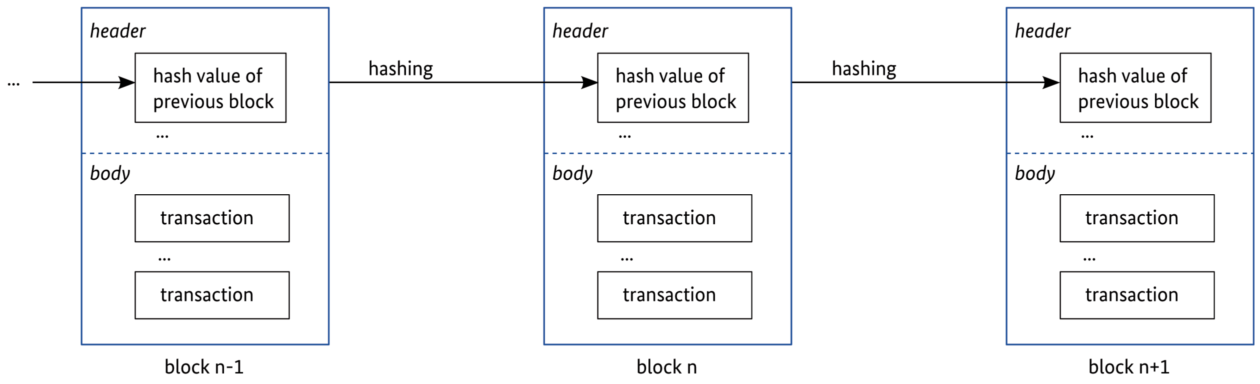

Figure 1: Blockchain as a data structure

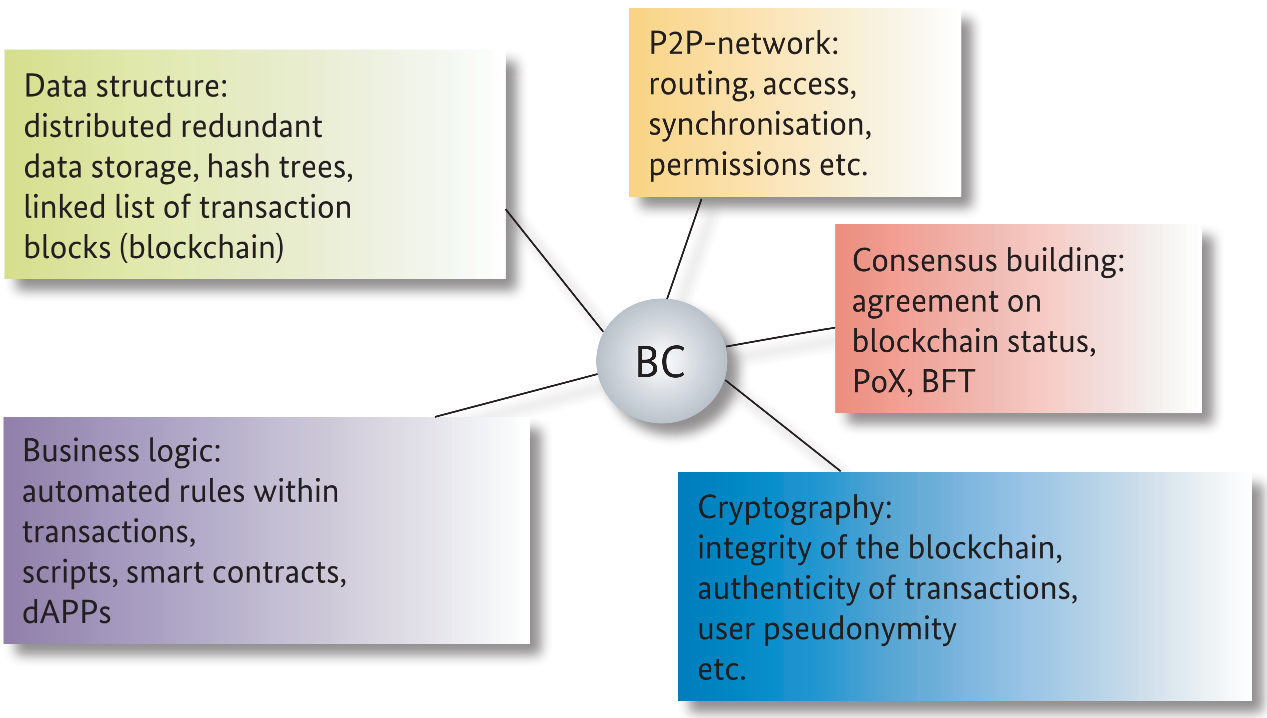
Data structure:
distributed redundant data storage, hash trees, linked list of transaction blocks (blockchain)
P2P-network:
routing, access, synchronisation, permissions etc.
BC
Consensus building:
agreement on blockchain status, PoX, BFT
Business logic:
automated rules within transactions, scripts, smart contracts, dAPPs
Cryptography:
integrity of the blockchain, authenticity of transactions, user pseudonymity etc.

Figure 2: Fundamental building blocks of the blockchain technology

In any specific use case, it is necessary to find an appropriate model for each of these building blocks, which means to decide on explicit network models, communication and data structures, consensus mechanism, business logic and cryptographic algorithms. For example, a blockchain-based cryptocurrency needs to be designed to deal with many different, unknown and possibly untrustworthy users. On the other hand, a blockchain that is used to track luxury goods must be able to reliably verify the origin and correctness of the data it processes. For every blockchain application, the choice and the design of a suitable blockchain model are essential in order to reach the intended functionality and security goals.

The chosen blockchain model, in turn, can be implemented in many possible ways. Among other things, it is necessary to choose a technological basis and suitable communication protocols, to find an accurate data representation and rule language, and to build a correct and secure implementation for the chosen algorithms. Such an implementation will in the following be referred to as a blockchain system.

Every blockchain system (see figure 3) contains certain core components, which technically define the blockchain and cannot be replaced. Among these are the network architecture, the blockchain as a data structure, the consensus mechanism, and the system control logic together with the underlying cryptographic algorithms. The blockchain core is supplemented by several

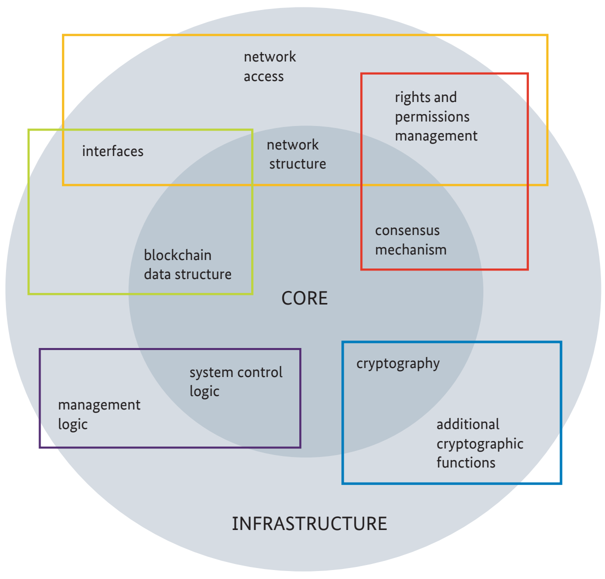

Figure 3: Multilayer model of a blockchain system

infrastructure components that are necessary for the system operation and certain additional functionality. These include, for example, the network access, interfaces to the surrounding environment, management logic, additional cryptographic functions, or a rights and permissions management.

So in brief, the term ‘blockchain technology’—or simply ‘blockchain’ as it is often called—covers a multitude of theoretical and practical variations and combinations of the blockchain building blocks. It is misleading to speak of ‘the’ blockchain technology or of ‘the’ blockchain, when instead every use case requires an individual blockchain model to be designed and a suitable blockchain system to be developed.

1.2 Taxonomy

Due to the many possibilities blockchain technology offers, specific blockchain models may differ significantly in design and thus exhibit completely different properties. To aid in a classification, the following terms have become widely accepted:

On the one hand, one commonly distinguishes between *private* and *public* blockchains. These terms refer to network access and data visibility. In public blockchains, anyone may dispatch data to the network and inspect all transactions in the blockchain, whereas in private blockchains these privileges are restricted to certain user groups, like organisations or consortia.

On the other hand, it is custom to differentiate *permissioned* from *unpermissioned/permissionless* blockchains when referring to the right to append new blocks to the chain. If all network nodes are authorised to validate transactions, to build new blocks, and to participate in establishing consensus, then the blockchain is called unpermissioned. If, however, these actions are only permitted to those nodes that have been appointed and authorised by a central authority, then the blockchain is said to be permissioned.

The choice of the appropriate blockchain category depends on the specific use case and should be considered as early as during concept development. Relevant aspects are, among others, the intended user group and the nature of the processed data, but also any economic and legal requirements that may affect the operation of the blockchain. A brief assessment of these factors reveals the implications for the further design of the blockchain model, in particular for the choice of the consensus mechanism and the cryptographic security measures.

The normal operation of a public unpermissioned blockchain (like Bitcoin) does not require a central authority. For the other blockchain models, it is essential that permissions and restrictions be more or less centrally managed. Therefore, they depend upon the existence of a suitable authority.

1.3 Examples

In the later chapters of this document, several real-life examples will be provided to illustrate the general discussion. To begin with, this section introduces some well-known examples which play a prominent role due to their widespread distribution, their maturity or their economic relevance.

Example: Bitcoin

The most popular example of a public unpermissioned blockchain application is undoubtedly the cryptocurrency Bitcoin [2], which has been online without interruption since 2009 and has by far the highest market capitalisation of all cryptocurrencies (67 billion US dollars as of March 2019).

Bitcoin uses a public-key cryptosystem both to generate the sender's and receiver's addresses and to authenticate all payments (transactions). Software tools called Bitcoin wallets can be used to store, manage and in some cases even generate the required key pairs. In simple terms, a Bitcoin address is the hash value of a public signature key (see upper part of figure 4). A full description of how Bitcoin addresses are generated can be found in [3].

Payments can be made by transmitting the sender's public key corresponding to the sender's address and a data record which is signed with the corresponding private key (see lower part of figure 4). By repeating the hashing procedure, the receiver verifies that the public key corresponds to the sender's address, and with the help of the public key he checks whether the signature is correct. If both verifications are successful, it is proven that the sender is indeed the owner of the coins to be transferred.

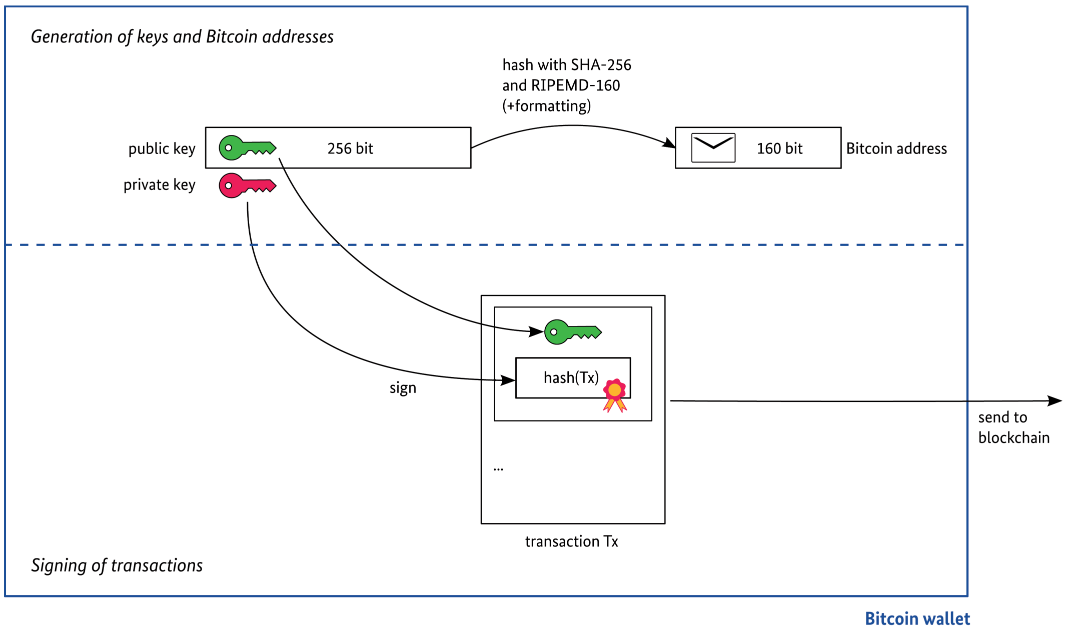

Figure 4: Schematic representation of the signature generation and the signing of transactions in a Bitcoin wallet

All transactions are broadcast throughout the Bitcoin network. Certain nodes, called miners, confirm the transactions, collect them into blocks, and append those to the end of the Bitcoin blockchain. A cryptographic hash function is used to link each block to its predecessor, thus securing the integrity of the entire chain. To establish consensus, Bitcoin uses a mechanism called proof-of-work (see also section 3.2.3): Before adding a new block, a miner has to find a hash value of a certain form, which is a computationally expensive task. As an incentive, the miner who is the first to solve this problem and appends his block to the blockchain receives a reward in the form of (newly generated) bitcoins and can also expect to get some transaction fees. This procedure is called mining.

For many, Bitcoin as an open, pseudonymous, and unregulated blockchain, available as open-source software, is the epitome of the pure blockchain idea. However, Bitcoin's approach is far from being suitable for every application, and therefore it should not obstruct one's view of the broad spectrum of other approaches that the blockchain technology offers.

The blockchain terminology originates in large parts from Bitcoin, but many terms are nowadays used in a broader sense. These include *blockchain* itself as the name of the data structure behind Bitcoin, but also, for example, the term *transaction*, which does not only refer to a transfer of bitcoins or other digital assets, but more generally to any piece of information that is to be processed by the blockchain. Likewise, a *wallet* is not necessarily a digital wallet. Instead, it may be any user interface to the blockchain network through which a user can manage his login data and certain secrets and access the blockchain system. The term *miner* is generally used for anyone having the permission to append new blocks to the blockchain.

Example: Ethereum

Ethereum [4], [5] is another public unpermissioned blockchain system. It provides the cryptocurrency Ether, which comes in second with a market capitalisation of more than 14 billion US dollars (as of March 2019). However, Ethereum has gained popularity mainly as a blockchain-based platform for smart contracts.

As a cryptocurrency, the mode of operation of Ethereum is comparable to that of Bitcoin. The underlying structures and processes as well as the consensus mechanism are similar. However, transactions in Ethereum can not only transfer assets, but also execute programs, known as smart contracts (see chapter 4). An individual address is assigned to each contract, which can be used for further interaction with the contract, but does not differ from user addresses in other respects.

In addition to the transaction data on the blockchain, Ethereum maintains a global system status *(world state)*, which contains, among others, the current account balances and, in the case of contract addresses, the hash value of the contract's bytecode. The world state is not stored on the blockchain itself, but locally in the memory of each node [6]. Valid transactions describe the transition from the current state to a new state, for example, when the balance of an Ethereum account is updated. To make sure that the local copies of the world state remain consistent on all nodes, every block contains the hash value of the current state. In order to compute this value, each smart contract initiated or called by a transaction in the block has to be executed during mining. Likewise, all nodes have to execute every smart contract a second time during verification, in order to confirm that the state hash matches the stored value.

Example: Hyperledger Fabric

Another well-known blockchain system is Hyperledger Fabric [7], [8]. It does not provide a cryptocurrency, but is first and foremost a platform for smart contracts. In contrast to the examples above, it is private and permissioned, thus allowing the use of message-based consensus mechanisms—in particular CFT and BFT algorithms (see section 3.2.2)—with only insignificant costs for the miners.

It is a primary goal of Hyperledger Fabric to provide a modular blockchain system in which several components can be selected or exchanged as needed. By this, the blockchain system may be adapted to meet individual requirements. In particular, the block validation and the consensus mechanism are separated from each other. In addition to offering an increased flexibility, this approach also reduces the consumption of resources (see section 4.1).

Summary.

- Blockchain technology allows a modular definition, and offers a variety of different designs and implementations.
- The choice of an appropriate blockchain model must depend on the specific use case.
- There are several possibilities to design and manage read and write permissions on a blockchain.

2 Assessment

In this chapter, blockchains are assessed with regard to several properties. They are also compared to their classical alternative, i.e. databases. The analysis allows a basic appraisal of the question of to what extent using blockchain technology in a certain setting is reasonable and what its benefits and downsides are as compared to databases.

The properties considered are, on the one hand, the security goals of IT security, which are introduced in more detail in the next section, and on the other hand practical requirements on throughput and scalability of the technology. These features stand partly in tension both with each other and with the fundamental design of blockchains, which wishes to establish decentrality, transparency and tamper resistance (see figure 5).

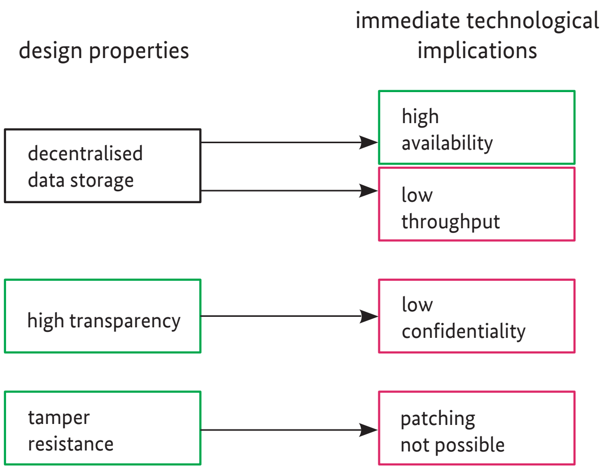

Figure 5: Design goals and their immediate technological implications

The deliberate decision for transparency, for example, makes assuring confidentiality and anonymity in blockchains very difficult (see sections 5.1 and 5.4). The same holds true for compliance with data protection requirements (see section 9.3).

Blockchain systems offer very good availability, which results from the decentralised way that data is stored. On the other hand, the latencies involved decrease the throughput of the system as compared to centralised solutions. Besides, decentralised data storage generates the need for additional storage and comes with associated costs. A close consideration of all of these aspects is hence always necessary.

2.1 IT security

2.1.1 Security goals

The classical security goals of IT security, which serve to assess technical solutions, are integrity, authenticity, availability and confidentiality. In addition, anonymity or pseudonymity are likewise often considered. The terms are defined as follows (cf. IT-Grundschutz [9]):

- Integrity means assuring the completeness and accuracy of data. A loss of integrity can mean that data was changed without permission or that information about the author or the creation time of data was tampered with.
- Authenticity is the property guaranteeing that a communication partner (a person or an IT component or application) is who he claims to be. Authentic information is guaranteed to have been created by the source indicated.
- Availability of services, IT applications or pieces of information means that users can always use them as intended.
- Confidentiality is the protection against unauthorised disclosure of information. Confidential data and information must only be accessible to authorised persons in the permissible way.
- An entity is anonymous if it cannot be identified. Anonymity guarantees in particular that data or actions of the same entity cannot be linked. In contrast, pseudonymity means that such a connection can be established using a

pseudonym, but at the same time it is not possible to link the pseudonym to a real identity.

The following assessment is limited to the blockchain itself. In particular, the different security guarantees for data only start holding once the data is stored inside the blockchain. For example, integrity protection does not extend to the input of data via an interface to the real world (e.g. sensor data, events in the real world). Just as with alternative technologies, additional measures would be necessary to achieve this goal.

The analysis distinguishes public and private blockchains where this is possible and makes sense.

2.1.2 Integrity

The integrity of data stored in a blockchain is basically assured by chaining individual blocks using a hash function. Provided a suitable hash function has been chosen, it is not possible for an attacker to subsequently tamper with data without being noticed.

It needs to be pointed out, however, that the security of hash functions may only be guaranteed for a certain period of time and that in the long term changes may need to be applied (see section 6.2). Furthermore, one should keep in mind that integrity protection for data often only comes into effect some time after the data is included into the blockchain. This is because several consensus mechanisms require a certain waiting time until the effort necessary to tamper with data included in blocks becomes prohibitive. The choice of the consensus mechanism is in turn linked to the blockchain model (see section 3.2). In general, private or permissioned blockchains allow assuring integrity faster than public unpermissioned blockchains do.

2.1.3 Availability

Thanks to the distributed and decentralised storage inherent to the technology, information is available with high probability at all times provided a sufficient number of nodes having stored the complete data is present. In this way, partial outages of the underlying network can be tolerated. Targeted attacks on a single central party are not possible. In private blockchains, however, an attacker making a large effort may well be able to reduce availability. In general, a higher rate of connections strengthens the network against outages due to targeted attacks or technical faults.

It is important to note that the good availability only covers the data directly stored in the blockchain. If data is outsourced, e.g. for reasons of data protection or confidentiality (see section 5.2), and the blockchain is only used to store references, it does not yield any guarantee for the availability of the actual data.

2.1.4 Confidentiality

The design of blockchains makes establishing confidentiality very hard. Confidentiality should thus not be a security goal when using blockchains. The reason is that transaction data must be available to all participant nodes and hence encryption of the data is very hard to implement. There are some complex proposals for establishing confidentiality that tend to be relatively inefficient though (see section 5.1).

On private blockchains, the number of nodes is usually much smaller than on public ones. Confidentiality is assured towards those without access to the network, provided communication within the network is adequately protected. Moreover, private blockchains provide the opportunity to further restrict visibility (e.g. [8]), but this requires more effort. Although data still remains visible for some nodes endowed with higher permissions when these solutions are used, they may be sufficient for many use cases.

A general approach for circumventing confidentiality issues is to store the relevant data externally (see section 5.2). However, in this case availability is not guaranteed by the blockchain according to section 2.1.3.

2.1.5 Authenticity

Transactions on a blockchain are usually protected by means of digital signatures using a public-key cryptosystem. If an appropriate cryptosystem is chosen, forging a signature for given data is extremely difficult and transactions are thus authentic. The considerations on long-term security from section 6.2 apply again though.

It is important that a blockchain can only ensure authenticity with respect to identities within the network. In order to assign the keys to a concrete communication partner, additional measures are required, just as with other technologies employing digital signatures. No approaches for doing this are established on public unpermissioned blockchains. On private or permissioned blockchains, the node operators would have to be identified in a suitable way when creating the keys, and the assignment of the keys would have to be documented, e.g. using a public key infrastructure (PKI).

The protection of the respective private signature keys is of paramount importance for ensuring the authenticity of transaction data. In particular when using a static (i.e. permanently used) key, losing this key is tantamount to forfeiting the authenticity of the node itself.

2.1.6 Anonymity/pseudonymity

Owing to the transparency provided by blockchains, different transactions of an entity may be linked. Thus, there is at most pseudonymity. On private blockchains this is not a problem, since usually an identification of the nodes is particularly desired. There are proposals to use different pseudonyms for different transactions on public blockchains in order to thwart the linking of transactions and attain anonymity. These measures, however, face fundamental limitations and can be reversed with sufficient effort. It also seems possible to remove pseudonymity by incorporating further information that often lies outside the blockchain itself. The reader is referred to section 5.4 for more details.

2.2 Efficiency

If blockchains are to be used on a large scale and for a large number of transactions, resource requirements and throughput are very important benchmarks in practice. The resources considered here are storage space and in particular the energy consumed. The throughput of a blockchain denotes the amount of transactions that can be included in new blocks during a fixed time interval.

2.2.1 Resource requirements

Blockchains provide very good availability for stored data. This results from the redundant distributed storage of all data starting from the first block. However, nodes operated by private parties would not be able to cope with the necessary, fast increasing storage requirements that would arise especially on public blockchains used on a large scale. The problem might be mitigated, mainly on private blockchains, by regularly deleting, at least partially, or compressing sufficiently old data.

An often cited obstacle limiting the scalability of blockchains is their energy consumption. For example, according to estimates from March 2019, Bitcoin is responsible for about 0.2 % of the global power consumption and thus uses an exorbitant amount of energy [10]. This is mainly due to its consensus mechanism. Other blockchain models, though, allow using consensus mechanisms that consume a negligible amount of energy (see section 3.2). Hence, energy consumption is not an issue with them. This applies in particular to private blockchains.

2.2.2 Throughput

The throughput a blockchain achieves is likewise strongly dependent on the consensus mechanism. On a lower level it also depends on the physical properties of the underlying network, since the decentralised storage and distribution of data require permanent communication between the different nodes inside the network. The first important property is the bandwidth, which specifies what amount of data can be distributed within the network per unit of time. Second, there is the latency, which denotes the time for transmitting a data package inside the network. Both values fluctuate to a certain extent over time and depend on which network nodes are communicating with each other.

In principle, private blockchains allow for a significantly higher throughput as compared to public ones. This is mostly due to the structure of the network, which usually contains much fewer nodes in private blockchains than in public ones. Furthermore, consensus mechanisms that can be used on private blockchains achieve a throughput which is orders of magnitude higher than the one attained by consensus mechanisms for public blockchains. For example, the throughput of Bitcoin was about 7 tps (transactions per second) in March 2019, whereas consensus mechanisms on private blockchains achieve 20,000 tps or more ([11], see section 3.2).

2.3 Comparison to databases

Blockchains are often considered an alternative to classical databases. Several criteria can govern the decision for using one of these two technologies. The rough comparison of the technologies presented below may serve as a first guidance. For reasons of clarity, only the two most prominent blockchain models, a public unpermissioned blockchain (e.g. Bitcoin) and a private permissioned blockchain (e.g. instances of Hyperledger), are contrasted to a classical client-server database (see table 1). In this setting, a central server stores

	Blockchain (public unpermissioned)	Blockchain (private permissioned)	Client-server database
Integrity	++	++	++
Authenticity	depends on implementation		
Availability	++	++	+
Confidentiality	--	o*	+
Pseudonymity	o	--	--
Decentrality	++	o	--
Resilience against misuse	++	+	o
Transparency	++	o	--
Resource requirements	(very) high†	low	very low
Throughput	--	+	++

Table 1: Comparison of blockchains and databases (cf. sections 2.1 and 2.2).

* This assessment reflects progress that has been made in some blockchains. For many solutions, confidentiality is weaker than assessed here.

† At the moment, all important public unpermissioned blockchains are using the consensus mechanism proof-of-work (see section 3.2). Some of them are planning to transition to less energy-intensive consensus mechanisms (as of March 2019).

the data, and clients can query the server for data. In general, individual aspects may vary to a certain extent depending on the concrete design chosen.

Databases employ a number of measures for ensuring integrity of stored data. These chiefly include the regular creation of backup copies and log files as well as strong access control to the data. The central server constitutes a single point of failure, and targeted attacks can thus greatly affect availability. However, this risk can be mitigated to the desired extent by storing data redundantly. In a database, the adequate use of cryptographic routines, where permissions are assigned to authorised users only, serves to establish data confidentiality (cf. also IT-Grundschutz [12]).

An essential difference is that a database—also when storing data redundantly—is controlled by a single operator. In contrast to blockchain solutions, misbehaviour of this entity can compromise security guarantees without this being prevented on the technical level. However, this risk is estimated to be small in practice, if one employs additional organisational security measures and takes into account the legal circumstances.

Due to the central infrastructure, the effort expended for computation and communication as well as the energy consumption required for operating a database is very small. At the same time, one can achieve very high throughput. For example, classical payment service providers reported achieving up to 56,000 tps as part of stress tests [13].

In a nutshell, when compared to databases blockchains offer benefits in terms of resilience against misuse and possibly availability, whereas they show significant disadvantages in terms of confidentiality and efficiency.

Summary.

- The security goals of IT security, the requirements on efficiency and the design goals of blockchains stand in tension with each other.
- There are large differences between public unpermissioned and private permissioned blockchains.
- Some security goals, in particular authenticity, need to be ensured by additional infrastructural measures.
- Blockchains offer benefits as compared to databases in terms of resilience against misuse and availability.
- Blockchains exhibit disadvantages as compared to databases in terms of confidentiality and efficiency.

3 Trust and consensus

Distributed storage of data is an essential blockchain feature. On the one hand, it increases resilience against misbehaviour on the technical level. Organisational measures are hence less necessary. On the other hand, distributed data storage requires procedures for guaranteeing the agreement of different data copies. The consensus mechanism used to this end is thus a core component of blockchains, and its design is fundamental for the security of the whole system.

3.1 Blockchains and trust

An essential property of a blockchain is the absence of a central party which serves to route and manage communication as in conventional solutions and which needs to be trusted by users. According to the original Bitcoin publication [2], this feature—i.e. that the security of blockchains is based 'on cryptographic proofs instead of trust'—constituted the fundamental motivation for devising Bitcoin. It is frequently described as the essential innovation of blockchain technology.

Unlike traditional solutions like databases, the actual operation of blockchains can do without a central party as described above. However, a significant amount of trust in some stakeholders and components is equally necessary on blockchains.

The programmers of the software that is used for the operation and participation in the blockchain hold an important position. Users need to trust them to actually implement the desired functionality and not to include vulnerabilities into the software by accident or on purpose. That also applies in case blockchains are developed and maintained as open source projects. In this case, the programmers govern the general development of the project and, for lack of time and expertise on the part of the users, public visibility of the source code need not necessarily result in the discovery of subtle flaws. Trusting in the correct implementation is all the more important, as on public blockchains it may take substantial time for updates correcting known vulnerabilities to be deployed by all nodes. Patching can thus be tedious.

Groups of prominent stakeholders often arise over time during the operation of public blockchains. In the area of cryptocurrencies, examples include exchanges for conversion to established fiat currencies, e.g. euro or US dollar, as well as mining pools, which are consortia of individual miners. For economic reasons, those areas exhibit a tendency towards centralisation (see section 3.2.5). Users have to trust exchanges to actually render their services [14] and to protect their infrastructure against attacks, for which they constitute prominent targets.

On private blockchains, the trust model is more akin to the one employed by classical centralised solutions, but at the same time they do not completely forgo decentralisation. Due to the various levels of permissions applicable, users need to trust those who hold higher positions than they themselves do. The fact that the blockchain is visible to its users, however, allows detecting various types of misbehaviour. Furthermore, private blockchains require a central administrative party for managing roles and permissions. Although it cannot directly interfere with the operation of the blockchain, by assigning or withdrawing permissions it might exert influence indirectly, but not without being noticed.

3.2 Consensus mechanisms

3.2.1 Problem statement

Blockchains need to ensure that data copies held by different nodes agree. That agreement also needs to extend to the ordering of blocks and at least of those transactions which are logically dependent on each other. A consensus mechanism is used to this end.

The situation as described is a special instance of the so-called consensus problem from the area of distributed systems. It requires several nodes in a network, some of which may be faulty, to agree on a value. Nodes start by proposing possibly different values. Using the consensus mechanism, all correct, i.e. non-faulty, nodes agree on ('decide') one of the proposed values. Formally, a solution to the consensus problem must fulfil the following conditions (cf. [15, pp. 203–205]):

- Validity: If a node decides a value, then this value was proposed by some node.
- Integrity: No node decides twice.
- Agreement: No two correct nodes decide differently.
- Termination: Every correct node eventually decides some value.

An algorithm fulfilling the first three conditions is said to guarantee safety. If it fulfils the fourth condition, it provides liveness. Informally, safety ensures that nothing bad happens, whereas liveness guarantees that eventually something good happens, namely that the procedure terminates.

The real difficulty with the problem stems from the conditions under which it needs to be solved. The two most important points to consider are the physical properties of the network and the behaviour of faulty nodes.

Concerning the network, one distinguishes the following three models:

- Synchronous network: Messages between two nodes are delivered within a known time interval.
- Asynchronous network: There is no upper bound for the time needed to deliver a message.
- Partially synchronous network: Messages are delivered within a bounded time interval, but this bound is unknown, or the network is synchronous from an unknown point in time.

The most important fault types are:

- Crash fault: Nodes crash but will usually be up and running again after some time.
- Byzantine fault (cf. figure 6): Nodes may exhibit arbitrary behaviour. Byzantine faults may be induced by deliberate misbehaviour of a node or may result from technical or human failures.

Algorithms that provide safety and liveness in the presence of the respective faults are called crash fault-tolerant (CFT) and Byzantine fault-tolerant (BFT), respectively.

The consensus problem has been extensively and successfully studied in the area of distributed systems starting from the 1980s [15]. Important results include general statements on the conditions under which the problem does or does not admit a solution [16], [17]. Practical algorithms which are proven to solve the problem have also been developed and are extensively being used. An important result states that the consensus

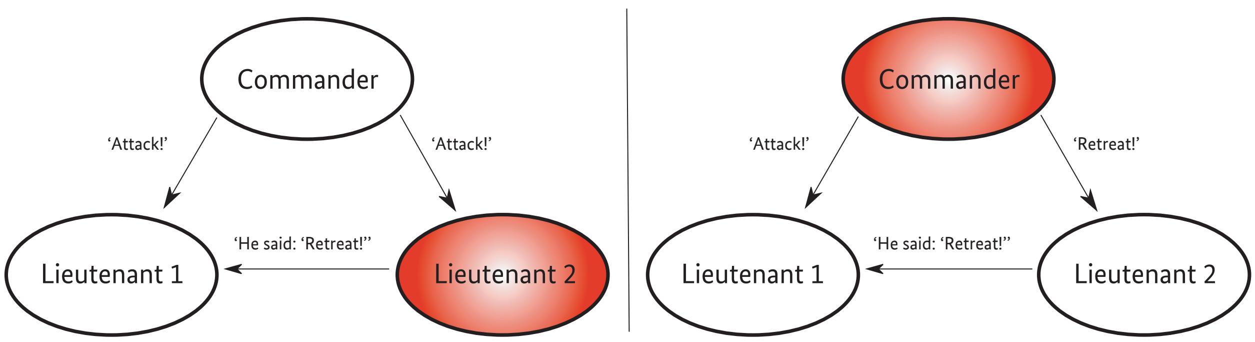

Figure 6: The Byzantine fault got its name from the Byzantine generals problem. Lieutenant 1 is unable to decide which of the other parties misbehaves.

problem in an asynchronous network cannot be solved by a deterministic algorithm even in the presence of just one crash fault [18]. As a result, a consensus mechanism needs to renounce one of the properties safety and liveness whenever the network is asynchronous, since it cannot prevent faults from occurring. Further results concern the number of nodes required to tolerate a certain amount of faulty nodes.

Apart from the statement under which conditions an algorithm can solve the consensus problem and which security guarantees it offers, efficiency is of paramount importance for widespread practical use. An important feature is the throughput the algorithm achieves. Throughput is always dependent on the behaviour of the network and the nodes, but in addition, there may be a strong dependence on the number of nodes participating in the consensus mechanism. If consensus mechanisms scale poorly to larger number of nodes, it is not reasonable to use them in unpermissioned blockchains.

3.2.2 Message-based algorithms

Over the years, a number of message-based algorithms solving the consensus problem under different conditions have been developed. Message-based means that nodes exchange messages according to a predetermined voting protocol in order to establish consensus. One node *(leader, primary)* holds a central position, whereas the other nodes behave passively. In case of crashes (CFT algorithms) or misbehaviour (BFT algorithms) of the leader, a new leader is elected.

For maximal robustness against adverse circumstances, these algorithms assume an asynchronous network. Since the consensus problem cannot be solved under these circumstances, they firstly guarantee only safety. In particular, data copies on different correct nodes always agree. Liveness, however, is only established as soon as the network is synchronous. In practice, asynchronous phases in a network are bounded in time, since problems are fixed after a certain period of time, which means the network is in fact partially synchronous.

Since they make the same assumptions regarding the network, the main difference among the practically relevant algorithms from the area of distributed systems lies in their fault model. There are CFT as well as BFT algorithms that are considered quasi-standards or at least the basis for further developments in the respective area.

In the field of crash faults, the algorithms Paxos and Raft [19], [20] are most well known. According to general theoretical results for this fault model, security guarantees for correct nodes may remain valid provided less than half of all nodes crash [16]. Both algorithms achieve this theoretical bound. As mentioned, they always ensure safety, and liveness as soon as the network is synchronous sufficiently long. The total amount of messages necessary for establishing consensus grows linearly with the number of participating nodes. Therefore, these algorithms do not scale to a large number of nodes. CFT algorithms like Paxos and Raft achieve a throughput of about 50,000 tps, if one requires that three faults be tolerated [11].

In the area of Byzantine faults, the algorithm PBFT (Practical Byzantine Fault Tolerance) [21] is fundamental. The guarantees for safety and liveness are the same as for Paxos and Raft and hold provided less than one third of all nodes exhibit Byzantine faults. That corresponds to the theoretical bound for this situation. However, since Byzantine faults are much more powerful than simple crash faults, significantly more messages are required for ensuring the security guarantees. The amount of messages grows quadratically in this case. Consequently, scalability is even poorer than with Paxos and Raft. PBFT and its enhancements achieve a throughput of about 20,000 tps, if one requires that three faults be tolerated [11]. Apart from increasing throughput in general, enhancements of PBFT in particular allow keeping throughput relatively stable in the presence of faults. The original PBFT algorithm can indeed tolerate Byzantine faults; however, throughput in this case drops considerably, which is unsatisfactory in practice.

In practice, distributed systems have almost always used CFT algorithms due to their higher throughput. At the same time, the occurrence of Byzantine faults was assumed highly improbable, since the consensus mechanisms were used for redundant data storage and all nodes were controlled by the same operator. Since the setting on blockchains is quite different, it is advisable to ponder whether resilience against Byzantine faults or throughput is more important. It should be noted that when using CFT algorithms, single nodes might purposefully sabotage the consensus mechanism. Due to their poor scalability, message-based CFT and BFT algorithms are not appropriate for use on unpermissioned blockchains. Another reason stems from the fact that message-based algorithms require verifying the identities of nodes in order to prevent participants from casting more than one vote during the voting protocol. These algorithms are frequently used on private blockchains though [22].

3.2.3 Proof-based algorithms

Numerous new proposals for consensus mechanisms accompanied the emergence of blockchain technology. Unlike message-based algorithms, in general they scale very well to large numbers of nodes and can also be used on public unpermissioned blockchains. A common feature of the algorithms is that the proof of a certain resource is fundamental in establishing consensus. The generic term proof-of-X (PoX) is sometimes used.

The most famous example is the algorithm Proof-of-Work (PoW) employed among others by Bitcoin and Ethereum [2]. For each block, it uses a computationally intensive mathematical puzzle (finding a string that hashes to a value below a certain bound) for establishing consensus on new data. The solution to this puzzle is the actual 'proof of work'. Essentially, the block proposed by the node who first finds a valid solution is distributed in the network and accepted by the other nodes. Due to latencies in message transmission within the network, different nodes may initially store diverging blocks at the same place in their local copy of the blockchain. This gives rise to diverging branches of the blockchain resulting in inconsistencies. Such an event is called a *fork*. Forks, however, are probabilistically consolidated after a short time.

One can abstractly model PoW as a consensus mechanism with a leader whose proposal is usually (provided no forks arise) accepted by all correct nodes. The computation of the proof of work amounts to the election of the leader, who can distribute his proposed block after a random time interval (the time he needs for computing a solution). In contrast to CFT and BFT algorithms, PoW always ensures liveness. In asynchronous phases, there are no safety guarantees whatsoever though. Because of latencies, safety is not even guaranteed in synchronous phases, and forks may cause temporary inconsistencies. This is called *eventual consistency*. In this context, the term *finality* of blocks or transactions is also used: With PoW, blocks added to the blockchain can subsequently become invalid by consolidating forks. That is not possible when using the CFT and BFT algorithms presented above, since they always guarantee safety, and blocks are hence final. The throughput one can achieve using PoW is comparatively small. For example, Bitcoin achieves about 7 tps, Ethereum about 15 tps (as of March 2019). Essentially, the latency of the network prevents one from choosing an arbitrarily small value for the time interval between different blocks, i.e. the average waiting time until the puzzle is first solved. Otherwise, forks would arise frequently and it would become virtually impossible to consolidate them.

The time a node needs to solve the puzzle is strongly dependent on the computing power of its hardware. PoW can thwart long-term attacks on the consistency of a blockchain provided less than 50 % of the computing power of the network are controlled by a (Byzantine) attacker (for details see section 7.1.2). The biggest issue with PoW is its enormous energy consumption (see section 2.2). Since the energy consumption and its cost serve to provide resilience against attacks, an incentive system (see section 3.2.5) incentivises correct nodes to participate in the PoW. In the case of cryptocurrencies, energy consumption can be assumed proportional to their market value for economic reasons.

Following PoW, further consensus mechanisms for public unpermissioned blockchains were proposed. They primarily aim to decrease energy consumption and to increase throughput.

In particular, the algorithm Proof-of-Stake (PoS) [23] is noteworthy. Ethereum has been planning to switch to it for quite some time, but this change has not yet taken place (as of March 2019). Instead of the computing power contributed, PoS uses funds in the underlying cryptocurrency as a prerequisite for creating new blocks. Those who would like to participate in the creation of new blocks as so-called validators have to deposit part of their funds. Depending on the implementation, one or more validators are randomly selected for each block and can submit proposals. The probability to be selected depends on the amount of funds deposited. In case several validators participate in the block creation, they now establish consensus among each other (e.g. using a BFT algorithm) as to which block should be added to the blockchain. The other nodes subsequently adopt this block. It is intended to implement certain rules in order to destroy the deposits of misbehaving validators. This aims to make several types of attacks significantly more expensive for an attacker and thus less probable. Power consumption is negligible in comparison to PoW. It must be pointed out that PoS is not in wide use right now and that its properties, especially concerning safety and liveness, strongly depend on the implementation. General statements are hence difficult to make.

Based on the abstract formalisation of PoW, some blockchains employ the consensus mechanism Proof-of-Elapsed-Time (PoET), which produces the random waiting time using a special hardware security module [22], [24]. Compared to PoW, this solution also has the advantage of negligible energy consumption. At the same time, it achieves a considerably higher throughput. The properties concerning liveness and safety are essentially the same as for PoW. On the other hand, in this case one needs to trust in the correct operation of the security module.

	Paxos/Raft	PBFT	PoW	PoS	PoET
Fault-tolerance	CFT	BFT	BFT	probably BFT	BFT
Safety	yes	yes	eventual consistency in phases of synchrony	depends on implementation	eventual consistency in phases of synchrony
Liveness	in phases of synchrony	in phases of synchrony	yes	depends on implementation	yes
Authentication of nodes required	yes	yes	no	no	no
Throughput	very high	high	low	medium/high*	medium
Energy consumption	very low	very low	very high	very low	very low
Scalability to large number of nodes	poor	poor	good	good	medium
Formal security proofs	yes	yes	no†	no	no

Table 2: Comparison of consensus mechanisms

* Reliable benchmarks from actual applications are not available, but cf. [28, p. 5].

† There are several formalisations and security proofs for PoW in the context of Bitcoin. However, they either make unrealistic assumptions or do not take into account certain aspects [29].

A fundamental property of the algorithms PoW, PoS and PoET, which is inherent in their design, is that single nodes may increase their influence on the consensus mechanism by means of economic investments and may in extreme cases even manipulate it in their favour.

Furthermore, on unpermissioned blockchains alternatives based on committees have been proposed [25], [26], [27]. Roughly speaking, for each block a small group of nodes is selected. Those nodes establish consensus among each other, and the other nodes accept the result. Consensus is usually established using message-based CFT or BFT algorithms. The committee may be selected in different ways (e.g. one can also consider PoS with several validators as a committee algorithm). It is crucial that the selection process be tamper-proof. Proponents of committee algorithms expect them to yield a very high throughput as compared to PoW while at the same time having negligible energy consumption. It must be noted, however, that the security of these algorithms is based on the interaction of several complex routines which are usually not well studied, and can hence not be assessed at the moment.

Table 2 provides an overview of the properties of the algorithms discussed.

3.2.4 Statements on security

The security guarantees a consensus mechanisms offers constitute fundamental properties of a blockchain. When choosing an algorithm, one should very carefully check whether it actually provides the specified guarantees. This includes a careful formalisation of the problem including assumptions on network behaviour and fault types.

In addition, statements on security should always be checked and backed up by means of a broad discussion and peer-reviews by experts in the field. Formally proving the level of security would be desirable, but may turn out to be difficult, especially when considering algorithms including game-theoretic components (see also section 5.5). As a matter of principle, comprehensible security guarantees for blockchains demand an independent evaluation and certification of the algorithms used according to predetermined standards and test criteria (see section 12.2).

Concerning message-based CFT and BFT algorithms, the publications of Paxos, Raft and PBFT do include detailed proofs of security. Proof-based algorithms, however, have forgone this approach in nearly all cases. New algorithms have been and continue to be published in white papers, which often include neither formal statements on the problems solved nor a formal line of proof. Provided such statements exist, they are very rarely supported by scientific reviews by experts and should hence be considered mere assertions. In a number of cases, security guarantees as specified by their developers have turned out to be wrong [30], [31], [32], [33]. A further negative aspect is the insufficient technical documentation of new proposals, which makes assessing their security even harder.

The fundamental problem in this area is as follows: It is comparatively easy and hence tempting to devise a new algorithm that performs well under propitious circumstances concerning the network and the faults occurring. It is much harder though to devise an algorithm that continues to preserve its security guarantees when facing adverse circumstances and (in the case of Byzantine faults) potentially unpredictable events (cf. [22]).

3.2.5 Incentive systems

Since the consensus mechanism PoW is very energy-intensive and its participants incur high costs accordingly, a so-called incentive system is one of its core components. The basic idea of the incentive system is to economically reward nodes participating in PoW for their services. More precisely, a certain amount of the underlying cryptocurrency is credited to the node first solving the mathematical puzzle for a block. Depending on the market value of the cryptocurrency, the computing power of the rest of the network and local power costs, a node thus receives an economic incentive to participate in PoW. The higher the market value of the cryptocurrency, the higher the overall computing power of the network will be. This in turn translates to a stronger protection of the blockchain against long-term attacks on the level of the consensus mechanism, although the risks stemming from a strong centralisation of mining persist (see section 7.1.2).

The incentive system alone does not necessarily influence the miners' behaviour as intended (see section 5.5). It is quite conceivable that individual parties act against their economic interests inside the system in order to attain other goals (see section 7.1.2). Furthermore, it has been shown, e.g. for Bitcoin, that even if the assumptions hold, the incentives stop working as desired when a party controls much less than 50 % of the computing power of the network (see section 7.1.2). As a matter of principle, correctly creating long-term economic incentives for desired behaviour is a notoriously difficult problem, see e.g. [34], [35]. This can also be seen in Bitcoin. On the one hand, its incentive system does ensure that controlling a majority of the computing power as a basis for practical attacks becomes prohibitively expensive. On the other hand, it comes along with an enormous energy consumption and a strong centralisation of mining, which is diametrically opposed to Nakamoto's original goals [2].

Summary.

- Trust in certain stakeholders is still necessary.
- When choosing a consensus mechanism, the behaviour of the network and the fault-tolerance need to be carefully modelled.
- The decision whether to prioritise safety or liveness during consensus is fundamental.
- Established consensus mechanisms for public and private blockchains differ in many aspects.
- Statements on security are often imprecise and not reliable.
- Economic incentives may not work as intended.

4 Smart contracts

In 1994, Nick Szabo introduced the term *smart contract* to describe a 'computerized transaction protocol that executes the terms of a contract' [36]. The term was picked up when the Ethereum blockchain was developed, but other platforms that support executable programs may as well use different names. In Hyperledger Fabric, for example, they are known as *chaincode* or in a somewhat broader sense as *distributed applications* (dAPPs). To simplify the following discussion, the term *smart contract* or simply *contract* will be used to cover all similar concepts on other platforms. Although the term may suggest otherwise, smart contracts are no contracts in a strictly legal sense (see also section 8.4).

In the context of blockchains, a smart contract is an executable program. A transaction is used to send its code to the blockchain where it is executed by all peers that take part in the validation process. Typically, transactions are also used to call a contract or provide it with input data. The immutability of the blockchain prevents later modification of the code, and due to the validation process being automatically performed, its execution can neither be prevented nor halted.

4.1 Examples

To illustrate different concepts of smart contract design, this chapter starts with a presentation of some well-known examples followed by a more systematic treatment in the following section.

Example: Ethereum

An Ethereum smart contract (see section 1.3) is typically written in the programming language Solidity. The compiled bytecode is broadcast on the Ethereum network in a valid transaction but without specifying a recipient's address. In the mining process, a new contract address is generated and published in the blockchain together with the bytecode (e.g. [39]).

Subsequent transactions to a contract address cause its code to be executed—first by the miner when validating the transaction and adding it to a new block, and then by every other node during the block verification process. This is a bottleneck for efficiency and throughput in the Ethereum blockchain, especially in the presence of contracts that require very high computational effort. The code execution is performed locally in the EVM (Ethereum Virtual Machine) of the respective node and requires the provision of sufficient resources.

In order to render calls to computationally expensive contracts economically unattractive and at the same time remunerate the miners for their efforts, Ethereum charges a fee for every executed operation. The fees are billed in Ethereum's internal currency *gas* [5], specially designed for this very purpose. Whoever wants to initiate or call a contract determines a maximum fee *(gasLimit)* he is willing to pay as well as an exchange rate *(gasPrice)* between gas and Ether. The miner who successfully includes the transaction in a new block is reimbursed for the expended amount of gas at the predetermined exchange rate. Thus, the exchange rate can be used to indirectly adjust the block reward (see however section 7.2.2). If the actual costs exceed the maximum fee, the execution is reverted, but the fee is not refunded.

Example: Hyperledger Fabric

In Hyperledger Fabric (see section 1.3), it is envisaged that smart contracts can be written in arbitrary general-purpose programming languages (as of March 2019, only Go, Java and Node.js are supported) in order to build on the programmers' existing experience.

A significant difference to Ethereum is that Hyperledger Fabric separates consensus from contract execution and validation, as regards both the chronological order and the responsible nodes.

Every contract assigns the task of code execution to only a subset of all network nodes. In a first step, each of these nodes simulates the contract execution locally and independently, and returns the result to the contract owner without adding it to the blockchain. Once a sufficient number of identical results have been collected in this way, a transaction is set up and sent to a different group of nodes, the co-called *ordering service*. This service runs a consensus protocol to determine the order in which the received transactions are to be added to the blockchain. In particular, the ordering service neither validates nor executes any transaction. Validation is performed as a final step by all network peers when they update their local copies of the blockchain. In this step, transactions may even be discarded if some preceding contract made changes to the blockchain state such that it no longer matches the conditions under which the simulation had taken place. It is crucial, however, that the final step is deterministic and cannot induce a fork in the blockchain.

From using this transaction flow, Hyperledger expects a lower redundancy and an increased throughput since it does no longer require all of the network nodes to execute the contracts.

Example: Bitcoin

In Bitcoin (see section 1.3), it is possible to program simple scripts. For this purpose, Bitcoin is equipped with a stack-based scripting system with a limited number of commands [37]. Programmers can use these scripts to specify the requirements of the signature checks and adapt them to their own needs. During the validation process, miners will automatically execute and evaluate the scripts. Although there is no explicit limit on the stack size, it cannot grow arbitrarily because the size of the script itself is bounded [38].

Further examples illustrate the challenges that other smart contract platforms intend to solve. The following list is neither representative nor exhaustive.

Quorum [40] is a derivative of Ethereum. It is a permissioned blockchain system that offers both public and private smart contracts. In private contracts, transactions whose content is to be kept confidential can be encrypted. In consequence, only nodes in possession of the corresponding decryption key can perform the validation of such a private contract. All other nodes have to skip it, which will lead to discrepancies in their calculation of the system status. To avoid this problem, Quorum introduced a private system status in addition to the usual public status, which is shared by all nodes. The private status is generated individually by every node and is only affected by those contracts in whose validation the node was involved.

For further suggestions on how to achieve privacy in smart contracts see also section 5.1.

For some time, Ripple offered a service named Codius [41], which is concerned with the problem of interoperability (see also section 11.3). Codius transactions are designed for calling smart contracts on arbitrary blockchain platforms. However, the lack of standards for payments across different blockchain networks caused temporary difficulties, which were eventually solved by the specially devised Interledger Protocol [42].

In Nxt/Ardor [43], the actual contract logic is executed on an API outside the blockchain. Only blockchain-specific operations, such as the transfer of assets, are transmitted to and processed by the blockchain. This approach allows modifying the contract code at some later time since it is no longer secured by the blockchain. So the paradigm of immutability is abandoned in favour of future updates and bug-fixes. The trust model (see section 3.1) changes accordingly.

4.2 Comparison of different approaches

The examples above illustrate that there are significant differences between existing smart contract systems. In the following, some of the relevant properties are discussed in detail, providing an overview of what needs to be taken into consideration when choosing a suitable blockchain platform.

4.2.1 Programming language

The programming language in which a contract is written affects both the potential complexity and the error-proneness or vulnerability of the code. Some systems use scripting languages of limited scope suited only for simple programs. Restrictions may apply to the script size—in terms of the number of bytes or the number of operations—, but also to the comprehensiveness of the available instruction set. A language that allows the construction of loops is vulnerable to an attacker deliberately constructing an infinite loop and thus stalling the whole system. For security reasons, some scripting languages forgo the implementation of potentially vulnerable language elements such as loops, backward jumps or random write access to the stack. They accept limitations of the functionality in favour of a better verifiability of the correctness of the code.

Other blockchain platforms promote universally programmable smart contracts and therefore facilitate the use of Turing-complete programming languages. Some of these are new developments, still error-prone and suffering from design flaws, while others are technically mature programming languages. They allow writing arbitrarily complex smart contracts, but on the downside, the correctness of the code can no longer be proven. Vulnerabilities such as the above-mentioned infinite loops have to be prevented by other means (e.g. by implementing a gas limit in Ethereum, see section 4.1).

4.2.2 Ordering and validation

A smart contract is recorded in the blockchain whenever it is initialised or called by a transaction. In most existing blockchain systems, all miners validate blocks and transactions in the course of establishing consensus. As part of this validation process, all included smart contracts have to be executed and, if necessary, the system status has to be updated. Since validation is performed by all miners, it obviously consumes a considerable amount of computational power, especially on unpermissioned blockchains where arbitrarily many nodes can participate in the consensus mechanism.

More recent approaches separate consensus and execution by considering only such contracts in the consensus process that have already been successfully validated by a certain subset of nodes (see section 4.1). The nodes that participate in the consensus are informed of the successful validation and need not validate the contract themselves. This approach requires a permissioned blockchain because considerable trust has to be placed in the set of selected validators.

4.2.3 Runtime environment

A stack suffices for the execution of Bitcoin scripts. Other blockchain systems in which smart contracts can be complex programs need a more elaborate runtime environment. During the validation process, the miners have to execute the contract code, which was written by potentially untrustworthy programmers, locally on their own devices. It is therefore necessary to guarantee both security for the host device and platform interoperability. This can be achieved by isolated environments such as virtual machines or Docker containers.

4.2.4 Incentive and currency

Bitcoin scripts are very short programs whose execution requires only a negligible amount of resources. On the other hand, smart contracts that contain complex code face the problem that every validating node has to invest time and computing power in the validation process.

In unpermissioned blockchain systems it is to be expected that only an economic benefit will incentivise the nodes to perform a possibly resource-intensive validation process strictly following the rules. The implementation of an on-chain payment system may be straightforward if the blockchain already comes with an intrinsic currency as is the case, for example, for Bitcoin and Ethereum. If not, it could be expedient to introduce an internal currency or token solely for the purpose of paying the validators. It is to be noted that the precise design of the incentive system affects the security of the whole blockchain (see section 7.2.2).

By contrast, in permissioned blockchain systems all validating nodes are known. They can be bound by contract to correctly perform the validation, or else a remuneration for their efforts can be paid off-chain.

4.3 Frequently used elements

This section gives a brief description of two elements that are found in a vast number of smart contracts: oracle services and random numbers.

4.3.1 Oracles

Smart contracts have no access to data that does not reside inside their blockchain universe. Many contracts, however, have to react to external events, e.g. betting contracts that calculate the payouts according to certain sport results. So some functionality is required for importing arbitrary data from the real world into a blockchain. At present, this functionality is provided by various oracle services that serve as intermediaries between a data source and the calling contract.

Typically, the calling contract selects which data source to use. There can be differences as to whether arbitrary or only pre-selected sources are available; whether only one or several sources may be chosen simultaneously; and whether it is possible to change the selected data source at a later time.

Depending on the oracle service, data is transmitted either *on-chain* or *off-chain* from the data source to the calling contract. In the former case, the service writes the requested data to a special service contract in the blockchain where it can be queried. In the latter case, the data is not sent to the blockchain but, for example, provided on a private website, and a transaction is used to send the login credentials to the calling contract.

Furthermore, oracle services use different means to convince the user of the authenticity of the returned data. Some offer cryptographic proofs of authenticity, which can be verified by the calling contract (e.g. [44]); others determine the return value by means of a voting process that is published in the blockchain and thus minimises the incentive to return incorrect data (e.g. [45]); and still others allow users to contest the returned data and raise an objection if the outcome of the oracle is dubious (e.g. [46]).

While it may be straightforward to incorporate oracles into smart contracts, there are still security issues to bear in mind. Even though some services offer a proof of authenticity for the returned data, they do not vouch for the integrity of the data source. This means that the data provided by an oracle may have been incorrect beforehand or even deliberately manipulated. Therefore, users should always give serious thought to the trustworthiness of the data source they want to consult. It can increase security to request data from several independent data sources and then decide based on the majority vote.

Purely technical problems may also arise if, by the time a smart contract makes a call to an oracle service, that service or the data source referenced therein have ceased to exist. This problem will particularly affect oracle services with a static data source that cannot be changed.

4.3.2 Random numbers

Many blockchain platforms depend on the deterministic evaluation of all transactions because new blocks are only accepted in the blockchain if all nodes obtain identical results during validation. Smart contracts, on the other hand, often need random elements, for example gambling contracts, whose users trust in fair chances, especially in fair random numbers.

One approach to incorporate randomness into a deterministic system is to use deterministic references to values that are still unknown at the time of contract creation, for example block parameters such as the number of the block to which the contract will be added. However, since the entropy of some of the suggested values is rather low, they are unsuitable for certain applications. Moreover, though the parameters may be undefined at creation time, they may very well be known at the time of mining. But this renders them susceptible to manipulation by the miners [47]. It may also be the case that random numbers generated in this fashion are not independent: All transactions within the same block that use the same block parameter will derive the same random number from it. This can be exploited by an attacker if he is able to include his attacking contract into the same block as his victim [48].

Another approach makes use of commitment schemes. Here, each participant picks a secret random number and commits to it by publishing its hash value in the blockchain. In the next step, the secrets of all participants are revealed and used to generate a random number. An attacker can influence the result by deciding to ignore the protocol rules and not to reveal his secret. There are ideas to disincentivise this behaviour by demanding that a deposit be made along with the commitment, but it is a non-trivial task to determine a reasonable size for the deposit. Manipulation is also possible on the communication layer, for example by clogging up the network and thus preventing transactions of other participants from being processed in time [49].

Oracle services (see section 4.3.1) can likewise be used to import random numbers, e.g. from external web sites, into the blockchain. As in all oracle applications, there is generally no way to assure the quality of the data source and thus of the imported random numbers. If the oracle transmits the random number in an unencrypted transaction, an attacker can read it before it is permanently added to a block. He may try to outrun the oracle transaction with a transaction of his own in which he can take advantage of his a priori knowledge of the random number. This attack is known as *front-running* [48].

The extent to which the different approaches are suitable for specific applications depends not only on the required security level of the random numbers, but also on the blockchain system in use. Especially the consensus mechanism plays a crucial role: Wherever miners exert significant influence on the order in which transactions are processed or can be tempted to do so by external incentives, the above-mentioned attacks should be taken very seriously.

Summary.

- Smart contract design varies across blockchain systems with effects on efficiency and security.
- Without further measures, oracles cannot guarantee the authenticity of real-world data.
- Using random numbers in a deterministic blockchain system is non-trivial and poses security risks.

Part II Security

5 Data security

Security and trust in a blockchain system are largely based on cryptographic primitives such as signatures or hash functions. Even more complex cryptographic mechanisms can be used to achieve demanding security goals such as confidentiality or anonymity. For several years now, the BSI has maintained a Technical Guideline [50] which provides an assessment of the security of selected cryptographic procedures. It also contains concrete recommendations for algorithms and key lengths and provides orientation and a long-term perspective for users. When using cryptography in blockchains, compliance with these recommendations can help to achieve an adequate security level for the cryptographic applications used.

In the following, the most important security aspects of a blockchain are discussed from a cryptographic perspective and various implementation options are presented. Special legal requirements, such as those that arise when using legally binding secure electronic signatures [51], [52] are, however, not discussed.

5.1 Confidentiality

The security goal of confidentiality has already been addressed in section 2.1.4. Confidentiality is, in principle, diametrically opposed to the design principle of transparency of the blockchain technology. To realise both properties in one solution is, to say the least, very challenging. If sensitive (e.g. personal) data is to be stored or processed on a blockchain, it is out of the question to include this data in transactions without cryptographic protection.

An obvious classic approach to protecting confidentiality would be to encrypt all sensitive data. Then, however, a corresponding key management for the encryption keys would have to be available, which is conceptually not part of blockchain systems. In addition, such encrypted data would be particularly affected by the limited long-term security guarantees (see section 6.2) of classical cryptography. If an encryption algorithm becomes insecure or keys are compromised, the affected ciphertexts cannot simply be withdrawn or re-encrypted because they are distributed throughout the whole network and are no longer under the control of their originator. In addition, encrypted data are also protected from manipulation by the integrity mechanisms of the blockchain. A particularly serious obstacle to the use of encryption is the fact that encrypted transactions or transactions with encrypted data cannot be easily verified in the network because they are not visible to all nodes. A cryptographic approach to solving this problem is to use zero-knowledge (ZK) protocols (see also section 5.4). However, their realisation is complex and often very resource-intensive.

A simple practical solution to the deficiencies of confidentiality is not to store sensitive data directly in the blockchain, but to use only references of the data (e.g. in the form of a hash value). The data itself would then have to be stored and protected against unauthorised access in an external structure (e.g. in a database). In this solution, however, only the existence and integrity of the data can be ensured by the blockchain, its availability for the execution of operations (e.g. in smart contracts) is not guaranteed. In addition, the confidentiality of the data underlying the reference values is not always readily ensured (see section 5.2).

In permissioned blockchains, the handling of confidential data can also be resolved by organisational measures. Here, for example, separate protected data channels [53] to which only approved nodes have access can be used within the blockchain network. Within this (potentially very small) set of nodes data can then be considered confidential. This approach is similar to the concept of 'partial transaction visibility' [54], [40], where nodes have access only to those transactions in which they themselves are involved or which they need in order to verify the transaction history.

Another important aspect concerns confidential calculations in transactions or the confidential execution of smart contracts. In addition to very complex cryptographic concepts such as homomorphic encryption, which are hardly ready for practical use, approaches with trusted execution environments (TEE) have emerged [55]. For distinguished authorised applications, TEEs provide an isolated runtime environment that prevents other applications, the operating system, but also the host owner himself from seeing or manipulating the state of the application. In the context of blockchains, smart contracts could be executed on the network without their content having to be known. However, this approach can be problematic if the consensus mechanism is not final (e.g. in PoW, see section 3.2.3). Attacks could result from repeated resetting of a smart contract [56], which could allow the attacker to draw conclusions on secret contents. As always, additional hardware provides new attack vectors, e.g. in the form of side channels, and the blockchain system must be hardened against potential failures of the TEEs.

Confidential smart contracts can also be realised via secure multi-party computation (sMPC) [57]. In this case, calculations are carried out jointly by several nodes, each of which knows only part of the data and does not share its information with the other nodes. A smart contract can be executed on the network without any party gaining access to its full content. However, sMPC methods are relatively inefficient and thus unsuitable for many applications.

5.2 Data storage

For the security of the data in an IT system, a secure storage is particularly crucial. In a blockchain system, data is stored as part of transactions in the blockchain. Here, transaction data include both the metadata used for verification and administration in the blockchain (e.g. signature, transaction fee) as well as the content data processed by the transaction (e.g. payment orders, certificates). Normally, these transaction data are available in unencrypted form and can therefore be viewed by all nodes with appropriate rights.

Data integrity is achieved in the blockchain by concatenating transaction data blocks using a hash function. However, the security goal of data integrity can only be reliably achieved when using cryptographically secure hash functions. These are so-called collision-resistant one-way functions, i.e. it is virtually impossible to find a matching input for a given hash value, or two input values with the same hash value. Hash functions are by design not injective, i.e. it is possible that different input values are mapped to the same hash value, but in practice, for the recommended hash functions with the recommended output length [50], it is extremely unlikely to find such input values.

However, not all of the data accumulating or processed in the system must be stored as transaction data in the blockchain itself, if they are not immediately necessary in order to execute the transaction. When it comes to large amounts of data or sensitive data (see section 5.1), it often makes sense to store only references to the data in the blockchain. This can reduce the size of the blockchain and limit the amount of data which are openly accessible. Cryptographic mechanisms with integrity protection, such as hash functions, are recommended for referencing if the tamper resistance guaranteed by the blockchain should also cover the referenced data.

When using hash functions for referencing, a cryptographically secure hash function should be selected, as well. In this case, its properties guarantee that it is practically not possible to add data to a reference or change data behind a reference.

In the context of using hash functions for referencing, it is important to note that hash functions do not guarantee confidentiality, although a back calculation of the input value from its hash is virtually impossible. However, the search space of the possible meaningful input values is often limited in practice, since the data is often available in a structured form (e.g. personal data in a fill-in form in pdf format). Then it is possible to test all possible input values with some computational effort (brute-force attack). This can be done in advance of an attack by storing the hash values of all possible inputs in structured form (dictionary attack). In order to prevent or at least hamper the leakage of information about the data via the hash value, it is necessary to add entropy to the hash, which makes dictionary attacks more complex. Normally, this is done via a so-called *salt*, an additional random input value. This salt does not have to be secret but needs to be unpredictable.

If two or more parties on the blockchain which can all access the referenced data have a shared secret key, then the use of a message authentication code (MAC) for referencing data is also an option. A MAC guarantees the authenticity of data in addition to providing integrity. A salt is not necessary when creating a MAC because the reference values can only be calculated if the secret key is known.

If a public key infrastructure (PKI) is available, instead of a simple hash value, signature schemes can also be applied to the data and the signatures can be stored as references in the blockchain. In addition to the integrity of the data, this also proves that a signature has been generated by an entity which can be identified via the PKI. When using signatures, the use of a salt is recommended, as otherwise brute-force attacks on the underlying signed data are possible again.

5.3 Cryptographic keys

In blockchain systems, key-based procedures are not necessarily required to achieve the main objectives of decentralisation, transparency and tamper resistance. However, various properties of transactions can best be realised by using asymmetric cryptographic techniques. This concerns above all the proof of authorship of transactions (authenticity), for which electronic signature schemes are the classical solution.

The use of asymmetric cryptography is always associated with a need for key management. This includes generating key pairs, storing private keys, and distributing public keys.

The distribution of public keys depends in particular on the requirements of identification. If public keys are to be used in a pseudonymous fashion without linking them to the owner, as for example in Bitcoin (see section 5.4), they can be distributed in the network without checking (e.g. as part of transactions). However, if the identity (or possibly other properties) behind a public key is important, it must be certified and verified trustworthily. For this case, public key infrastructures have been established as a suitable solution, which, however, involves a certain amount of organisational and technical effort and must be operated in parallel to the blockchain structures. In addition, a trustworthy central authority (certificate authority) is necessary again.

In practice, most errors occur when storing private keys. If, as in the Bitcoin system, the private signature key is the only evidence of possession of property on the blockchain, then this key is highly at risk. Private keys should be stored securely, e.g. on a hardware token, in an encrypted disk container or even offline in a safe. To protect against loss, backup copies should also be created and managed separately. There are now a number of vendors providing solutions for electronic wallets to store keys. The security of these wallets varies and must be checked in each case. Among other things, the security of a wallet depends on whether it is a *hot wallet*, i.e. connected to the internet, or a *cold wallet* without an internet connection. Using a cold wallet can significantly reduce the risk of hacker attacks.

From a security perspective, the correct key generation also plays a major role. For the generation of private keys randomness is always needed. No secrecy is possible with a purely deterministic procedure. If a weak random number generator is used, whose output is predictable in some way, then attacks on the secret key can be significantly easier. Especially with online wallets, it is difficult to assess where the randomness which is needed for the key generation comes from and what quality it has. The recommendations of the BSI should also be taken into account when generating or using random numbers in cryptographic procedures [50], [58], [59], [60].

More information on practical threats and security incidents in key management can be found in section 7.3.2.

Example: Bitcoin's signature key

In section 1.3 it was explained how to calculate a Bitcoin address from the public signature key of a user. This method results in two unintended ways to take possession of the funds behind the address (see figure 7).

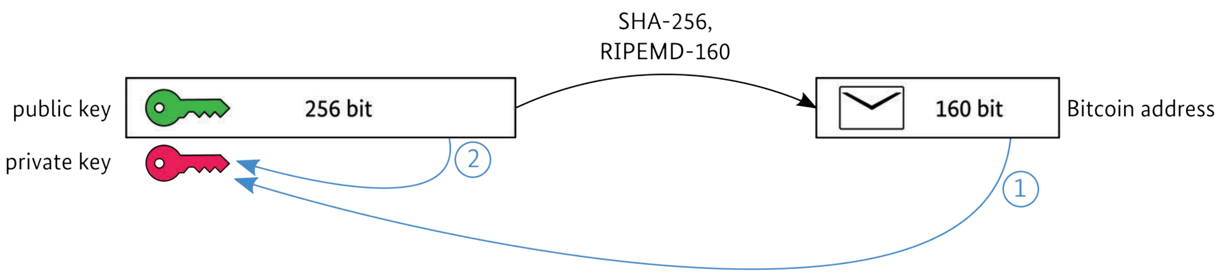

Figure 7: Two attacks against the private ECDSA signature key

First, due to the address length of 160 bits, for each address there are about 2^{96} valid secret signature keys of 256 bits in length that an attacker can try to find by testing. He can keep trying as long as funds are still stored at the address.

However, if addresses are reused, then the public key is known since it was transmitted in previous transactions, and the second option is to attack the DLP. According to the current state of research, one would use the Pollard-Rho algorithm [61] for such an attack. It is often asserted that an attacker only has ten minutes to complete this attack because it takes that long for a transaction to be included in a block. However, this is a fallacy, since in Bitcoin addresses are often reused and thus many public keys have been known for some time.

Even if the keys are securely created and managed, a crypto procedure can be vulnerable through its secret keys. The security of asymmetric methods is based on the fact that the secret keys (e.g. for decrypting or signing) cannot be calculated with realistic effort from the public keys. This is due to the difficulty of the underlying mathematical problem (e.g. discrete logarithm problem (DLP) or factorisation problem). However, if the difficulty of the problem can be reduced by cryptanalytic methods, an attacker can determine the secret key in whole or in part and thereby circumvent the procedure.

5.4 Pseudonymity and anonymity

Among other things, the Bitcoin system promised to protect its users' privacy by using anonymous keys [2]. Transaction addresses are calculated from hash values of the public signature key of the recipient (see section 1.3), whose corresponding private keys can later be used to authenticate the transfer of the received amount to another Bitcoin address. In Bitcoin, users do not have to use their real identity, but they need an address in the Bitcoin system to be contactable. For this reason, one cannot speak of anonymity in Bitcoin, but at best of pseudonymity.

However, pseudonymity cannot reliably be attained in practical applications either. Transactional and network analysis can be used to create user profiles, and interfaces to the real world, such as for example in payment services, can be used to reveal the identities of users [62], [63], [64], [65], [66]. There are already tools available, some of them exhibiting excellent success rates [67], [68], that can be used by researchers, law enforcement, or by individuals. Similar efforts to de-anonymise participants also exist for the Ethereum blockchain [69], [70].

There are various alternative cryptocurrencies that use concepts for anonymous users and/or anonymous transactions. Essentially, three different approaches have emerged:

- Mixing: Mixing is a method of combining transactions of different users in the blockchain such that the original source of a single transaction can no longer be traced. For some blockchain systems (e.g. Bitcoin), external service providers offer mixing services, but then users have to entrust their transactions to the mixing service. For other systems, such as the cryptocurrency Dash [71], mixing is an integral part of the protocol.

- Ring signatures: For ring signatures, digital signatures are created using a group of network participants who are not actively or knowingly involved in signing. The group size and the public keys of their members are part of the signature. As a result, signed transactions can be uniquely verified, but the authorship of the transactions can only be traced back to the signature group and not to the individual participant. For the practical anonymity of these procedures a sufficiently large signature group is decisive. But this, in turn, leads to a fast growing data volume of the transactions, which limits efficiency. Ring signatures are used for example in the cryptocurrency Monero [72].

- Zero-knowledge: ZK proofs allow one party to prove to another party that a given statement is true without revealing any other information about the statement itself. This can be used to put transactions in the blockchain while disclosing neither the sender nor the recipient nor the amount transferred. In these procedures, the data in the transactions can be kept confidential. Most of the methods already in use today (e.g. zk-SNARKs [73]) are relatively inefficient, as the complexity of the proofs increases greatly with data volume. In addition, some methods require a secret setup phase to generate certain parameters. Newer variants (e.g. zk-STARKs [74]) promise better scalability and transparency. ZK methods are for example used in the cryptocurrency Zcash [75].

However, many solutions that use these approaches have already encountered problems in the real-world use of the anonymity functionality. There are now a number of studies that have found ways to track and connect transactions despite the implemented protective mechanisms [76], [77], [78]. In most cases, side channel information or metadata which are not part of the transaction but created on the network level play a role.

5.5 Security assessments

The security of an IT system significantly depends on the security of the underlying cryptographic mechanisms. This is especially true for blockchain systems, as organisational security measures are often dispensed with (especially in public unpermissioned blockchains) in favour of decentralisation and disintermediation.

The security of cryptographic algorithms, in turn, can be specified as a security level in bits, depending on the selected parameters (e.g. key lengths). A security level of n bits means that the effort required to break the algorithm is 2^n elementary operations. In this setting, security is quantified as the computational power of a possible attacker. Starting from the desired security level or the expected attacker strength, the parameters of the cryptographic algorithm should be selected. The recommendations of the BSI [50], for example, aim for a security level of at least 100 bits. However, it must be taken into account that such considerations only reflect the current level of knowledge about possible attacks. If technical or cryptanalytic progress makes better attacks feasible, the parameter sizes have to be changed or even the algorithm has to be withdrawn (see section 6.1).

In addition to BSI's Technical Guideline [50], there are various international standards for cryptography (e.g. from ETSI, NIST, IEEE, ISO/IEC, IETF, ANSI) that offer recommendations and specifications for secure cryptographic mechanisms and define the state of the art. Known blockchain implementations (e.g. Bitcoin, Ethereum, Hyperledger Fabric) currently use classic cryptographic mechanisms (e.g. the hash function SHA-256, ECDSA signatures) that generally conform to the recommendations and achieve an adequate level of security. There are often no comprehensive or independently verified security assessments for newer or more specialised cryptographic methods (e.g. zk-SNARKs, sMPC). This makes assessing their level of security or providing recommendations very difficult.

Basically, an IT system is at most as secure as its weakest component. Its security is also dependent on the composition of the individual mechanisms and their integration into the overall system. In addition, the quality of the implementation and the operational conditions play a crucial role. Since blockchain systems can become very complex due to their distributed structure and their interfaces, statements about their overall security are difficult to achieve.

Another challenge, in particular in the quantitative assessment of the security of blockchains, are consensus mechanisms (see section 3.2). A valid consensus building is essential for data security in blockchains, especially for protecting against manipulation of data. In addition to cryptographic mechanisms (e.g. hash functions), in this setting economic incentives often contribute to security. This is why the term *crypto-economy* is also used. Game-theoretical considerations and findings often play a role [79]. At the same time, these factors have repercussions on the design of cryptographic mechanisms and protocols, which must be adapted to a changed attacker model [80]. The security contribution of the economic incentives is generally difficult to quantify, and a comparison or relation to the classic cryptographic security level is hardly possible (see also section 3.2.5).

Summary.

- Confidentiality is difficult to achieve in blockchains.
- Sensitive data should not be stored or processed directly or without protection on a blockchain.
- Cryptographic mechanisms should be state-of-the-art, and a good key management is required.
- Mechanisms for anonymisation and pseudonymisation on blockchains are often not reliable in practice.
- Concrete (especially quantitative) assessments of the security level achieved cannot be made for most blockchain applications.

6 Long-term security and crypto-agility

In a blockchain, sensitive data which require long-term protection need to be specially protected. Due to the long availability and potentially high sensitivity of data in a blockchain, the achievement of long-term security poses a particular challenge. A suitable concept for crypto-agility must ensure that the security mechanisms of the blockchain can be exchanged as needed. In particular, requirements arising from technical and mathematical advances in cryptanalysis or potential quantum computers must be taken into account.

However, the following considerations show that some security goals cannot be guaranteed in the long term by crypto-agility alone.

6.1 Crypto-agility and security guarantees

In principle, it is not possible to guarantee the security of cryptographic algorithms in the longer term—even if one disregards implementation aspects. In its Technical Guideline TR-02102 [50], the BSI lists algorithms that it expects to remain safe for at least six to seven years. Typical security proofs reduce the security of a cryptographic algorithm to a mathematical problem that is considered difficult. Prominent examples from today's asymmetric cryptography include the discrete logarithm problem (DLP) or the factorisation problem. In symmetric cryptography, one shows that known attacks are possible only with great effort. The underlying security assumptions can always turn out to be wrong. For example, it is known that (still hypothetical) quantum computers of sufficient size can easily solve cryptographically relevant instances of the factoring problem or the DLP. In addition, quantum computers may be used to speed up classic cryptographic attacks or to facilitate side-channel attacks. The state of development of quantum computers is being investigated by the BSI in a long-term study [81].

Currently, the field of post-quantum cryptography is developing. Post-quantum algorithms are safe against quantum computer-aided and classical attacks according to the current state of knowledge. However, the security assessment of these algorithms may change significantly over time. Therefore, long-term security statements come with considerable uncertainty. Nevertheless, post-quantum algorithms can be an alternative to classical cryptographic algorithms for use in blockchains.

At least, however, the blockchain must be set up in such a way that it is possible to exchange cryptographic mechanisms at runtime. This property is called *crypto-agility*. Blockchain crypto-agility is a current research area in which no fixed concepts are favoured yet [82].

Depending on the type of blockchain, such an exchange of cryptographic mechanisms can lead to the split-off of a new development branch, a so-called fork, if no agreement can be reached in the developer and user community. For cryptocurrencies, this option has already been discussed, see [83]. It becomes particularly problematic when a hardfork is necessary, i.e. a fork of the blockchain using the new mechanisms, which is no longer compatible with the old version and whose introduction therefore requires a highly synchronised approach within the network.

If a cryptocurrency uses a fixed hash function for PoW, a large proportion of the miners will use special hardware and thus be adversely affected when switching to another hash function that may offer better security. A change to long-term secure cryptographic algorithms can therefore fail due to economic reasons.

In general, even a successful exchange of cryptographic algorithms does not automatically preserve the original security guarantees for old data.

Because of the problems described above, long-term security should not be a core requirement for a blockchain application, and blockchains alone do not seem to be appropriate for resolving issues of long-term preservation. This is particularly true for public unpermissioned blockchains, where in principle everyone has access to the entire blockchain. The situation in private permissioned blockchains is better, especially if the communication between the partners is encrypted, the data is shared exclusively or is stored redundantly in a secured external environment.

6.2 Long-term security

Data in a blockchain is typically accessible for a very long (if not unlimited) time and may need to be protected for its whole lifetime. Once inserted in the blockchain, it will remain there for the entire lifespan of the blockchain, and will be available in unchanged form (at least that is the goal). In addition, any authorised user is always able to save the blockchain locally. Later on, he can access the data, for example, if a weakness in the cryptographic methods used has been found.

When assessing the long-term security of a blockchain, the following security goals must be taken into account:

6.2.1 Integrity

To ensure the integrity of stored data, blockchain systems must maintain the integrity of the entire blockchain at all times, even if the cryptographic algorithms used are no longer regarded secure at a later point in time. In the standard case of a blockchain which is linked by means of hash values this means that the security of the hash function used is to be ensured. One straightforward solution is to rehash old parts of the blockchain with a new hash function, add the hash value to a new block, and thus ensure integrity using a new, more secure hash function. The new data would have to reference the original data in the blockchain. However, rehashing requires access to the old data. In addition, their owner may need to be involved.

6.2.2 Authenticity

If electronically signed data is put into a blockchain, for example with the aim of confirming authorship in a legally binding form, then this aim must still be ensured in case the underlying signature algorithm becomes vulnerable. Replacing the signatures of data in their original transactions by new signatures is prevented by the immutability of the blockchain. A new signature must therefore be added to a new block and be appropriately linked to the original data. In order to guarantee the authenticity of signed data permanently, the signature of the data must be renewed with a more secure algorithm in a timely manner. In addition, it must be ensured that the signature algorithms used were still secure at least at the time of the respective signature creation. Blockchain applications with the goal of long-term security must therefore be able to determine afterwards when data has been inserted. So it is necessary to use a sufficiently accurate verifiable time stamp.

In the example of the signature algorithms typically used today, which only sign the hash value of data (see figure 8), it cannot be ruled out that the signature algorithm itself is still secure, but the hash function it uses has been broken. If, for example, it becomes possible to generate second preimages, i.e. to find another message with the same hash value for a fixed message, it is not sufficient to record only the hash value of the original message in the blockchain, because a potential attacker could be able to provide a second message with the same signature. This is true even if a link to a file containing the message is added to the chain, and the original message can be replaced by the second message in that file. Such attacks on authenticity must be prevented through appropriate measures, such as the rehashing of old data mentioned in section 6.2.1.

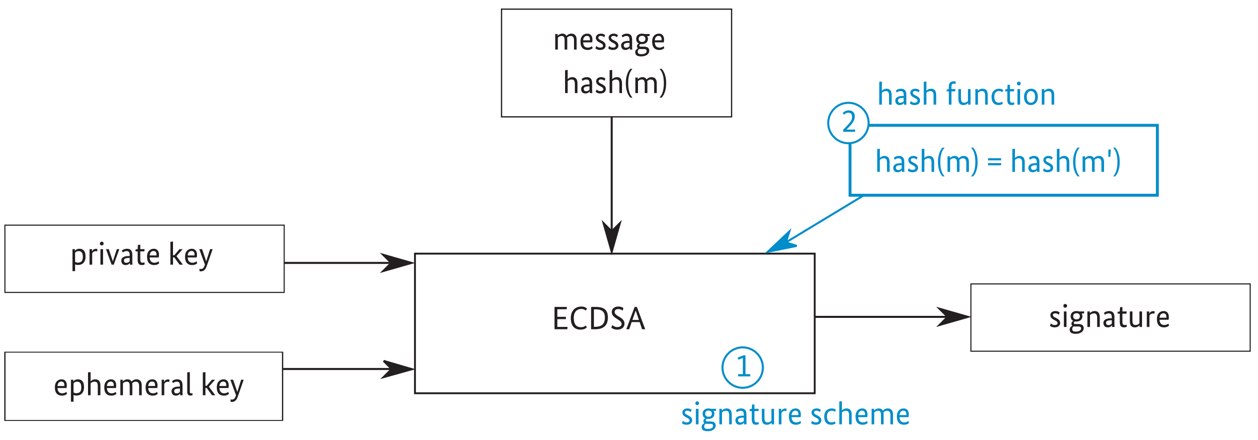

Figure 8: Over time, the signature algorithm and hash function may become vulnerable.

6.2.3 Confidentiality

In many cases, it does not make sense to overtly write data to a blockchain, on the one hand to protect the confidentiality of data, on the other hand since each storage of larger files—encrypted or unencrypted—greatly increases the amount of data in a long-lived blockchain.

In addition, directly storing signed encrypted data with the aim of achieving confidentiality also comes with challenges. For example, the encryption algorithm used could be broken in the future and thus the data could be obtained by decrypting it. A pure re-encryption of the data as a countermeasure is not enough, because a potential attacker generally has access to all data of the blockchain, which ideally are locally available in many places (see section 5.1).

Example: Bitcoin addresses

The example in section 5.3 explained that an attacker would need to successfully attack the DLP in order to compute the associated private key from a public Bitcoin key that has become public knowledge. However, it has already been proven that (still hypothetical) quantum computers of suitable size can solve the problem much more efficiently with Shor's algorithm [84] than all currently known algorithms can do. As a result of significant advances in the development of quantum computers, Bitcoin funds would thus be directly threatened.

One easily recognises that old or reused Bitcoin addresses can be particularly vulnerable and that it is sensible to occasionally transfer one's funds to new addresses (which, however, comes with costs) as soon as it becomes apparent that the DLP connected to Bitcoin is vulnerable. If the Bitcoin signature scheme has to be changed, that change is theoretically relatively easy for those bitcoins for which there is still an owner with access to the private key belonging to them. However, the Bitcoin community would have to agree on an algorithm change, which can be considered difficult based on past experiences. The owner would then have to initiate a transaction to his own address which is secured by the new algorithm. If he does not execute such a transaction, his bitcoins are at risk of theft. There are, however, already bitcoins for which the corresponding signature key has been lost. It is estimated that there are about three millions of such lost bitcoins [85]. These funds—less the cost of an attack—could be made available again.

Summary.

- The design of blockchains containing long-lived data is particularly demanding and requires a concept for crypto-agility.
- An exchange of cryptographic algorithms does not automatically preserve the original security guarantees.
- Sensitive long-lived data should not be stored directly—in encrypted or unencrypted form—in a blockchain.
- When using blockchain technology, the security of long-lived data is most likely to be achieved in private permissioned blockchains.

7 Attacks

As is the case with any technology, blockchain is vulnerable to certain attacks. Apart from new attack vectors targeting, e.g. the consensus mechanism, attacks which are well known from other areas are feasible on different levels (see figure 12). This section sets out the existing threats. Concrete incidents, which have often caused substantial damages, bear witness to the practical relevance of these attacks. Since every blockchain system comprises a complex IT infrastructure, precautionary measures as known from the area of IT security also apply in this context. They are provided in a short list at the end of the chapter.

7.1 Attacks on the blockchain core

7.1.1 Cryptographic routines

The security of a blockchain is based on the security of the cryptographic functions it uses (see section 5.3). Among others, they include methods for authenticating nodes and transactions. Authentication methods usually employ digital signatures using a public-key cryptosystem, as well as hash functions for chaining blocks, which thus ensure the integrity of the data stored in the blockchain. In addition, hash functions are used for tying up transaction data (hash trees) before concatenating blocks, and common signature schemes do not sign the transaction data itself but rather its hash value.

If an attacker is able to break the public-key cryptosystem used and to create unauthorised signatures for given data, he can initiate arbitrary transactions on behalf of other nodes. In cryptocurrencies this means that an attacker can steal other nodes' entire funds.

If an attacker can break the hash function and compute second preimages for given hash values, he is able to change blocks that have already been confirmed and firmly embedded into the blockchain. This completely destroys the integrity of the system. Apart from integrity, in this case authenticity of transactions can no longer be guaranteed either. An attacker can modify a transaction in a way which does not change its hash value. The original signature thus continues to be valid. For more details on the attacks discussed, the reader is referred to sections 5.2 and 6.2.

Many well-known blockchains like Bitcoin, Ethereum or Hyperledger Fabric use standardised cryptographic routines, which can currently be considered secure (e.g. the hash function SHA-256, ECDSA signatures). On the other hand, in some cases blockchains employed home-grown cryptographic algorithms. For example, the cryptocurrency IOTA initially used a self-designed hash function. This function was broken by cryptanalysts in a short time [30] and was only replaced by a standardised hash function as a result of further discussions.

7.1.2 Consensus mechanisms

The consensus mechanism ensures the state of the blockchain is in agreement among different nodes. Details may be found in section 3.2.1, which also discusses all algorithms presented below. Several attacks are conceivable depending on the implementation, see also section 3.2.4.

The basic options to attack CFT and BFT algorithms have already been discussed in section 3.2.2. Widespread CFT and BFT algorithms use public-key cryptosystems for message authentication. Consequently, an attack on these cryptographic functions directly translates to an attack on the consensus mechanisms.

Within the consensus mechanism PoW, the underlying mathematical puzzle (see section 3.2.3) is typically based on a hash function. If an attacker is able to break this hash function and to efficiently compute suitable preimages, he can completely control block creation. Besides that, an attacker controlling more than half of the computing power of the respective network can also attack PoW, as was already noted by Nakamoto in the original publication [2]. In this case, which is called a 51 % attack, the attacker can append new blocks containing the transactions he desires to previous blocks of the blockchain. Due to his superior computing power, the chain he creates will eventually be longer than the one worked on by the honest nodes. Once this happens, the honest nodes will accept the attacker's chain according to the protocol. It is important to note that a 51 % attack allows invalidating blocks already embedded in the blockchain. In particular, a successful 51 % attack targeting a cryptocurrency allows the attacker to retroactively annul payments for which he has already obtained a service in return. That way, he can spend funds multiple times, which gave rise to the term *double spending attack*.

Whether or not a 51 % attack is feasible, depends on the attacker's financial resources. According to estimates, 51 % attacks on minor cryptocurrencies (having market capitalisation of up to 30 million US dollars) would cost at most a few 100 US dollars per hour, if the attacker were not to buy but rather to rent the required hardware for a short time (as of March 2019) [86]. Among other cryptocurrencies, the Bitcoin fork Bitcoin Gold fell victim to such an attack in 2018, which caused losses amounting to 18 million US dollars [87]. If existing mining pools were to coordinate an attack, they would also not need to acquire hardware. According to [88], the net power cost for a 51 % attack targeting Bitcoin is about 210,000 US dollars per hour. Buying the hardware required for such an attack, however, would cost about 7 billion US dollars according to [88], and would thus be more expensive by several orders of magnitude (as of March 2019). Due to the geographical concentration of mining pools, coordinated attacks as mentioned might also be imposed and controlled by external stakeholders. One oft-cited argument asserts that attacks performed by miners do not make sense economically thanks to the incentive system that forms part of PoW. This does not take into account though that financial incentives outside the blockchain network may just as well motivate an attacker. Such *Goldfinger* attacks could, for example, aim to destabilise a cryptocurrency [89].

Besides, owing to random forks arising from latencies within the network, in practice the bound for 51 % attacks is less than half of the computing power [90]. Namely, a miner may already have found the solution for the current block and propagated it to the network, but miners who have not yet received it will still continue computing a solution of their own. Even apart from that, successful attacks are in principle possible using a smaller proportion of the computing power, but their probability of success will rapidly diminish in this case. Nevertheless, an attacker's expected gains may be positive if the blocks to be changed contain very valuable transactions [91].

Another strategy, called *selfish mining*, targets the incentive system of PoW [92]. An attacker doing selfish mining significantly delays the publication of some of the new blocks he found, thus deliberately breaking the protocol rules. The asymmetrical access to information, where the attacker knows all blocks found by the rest of the network, but the converse does not hold, causes honest nodes to waste computing power, which diminishes their expected revenue. Provided their actions are solely based on economic considerations, it makes sense for them to join the attacker, which eventually enables them to mount a 51 % attack. Even if the countermeasures as stated in [92] which are feasible in practice are incorporated into the PoW protocol, controlling no more than 25 % of the total computing power is sufficient for a successful selfish mining attack [92].

Figure 9 gives an impression of the distribution of computing power among mining pools within the Bitcoin network. Note that seemingly different

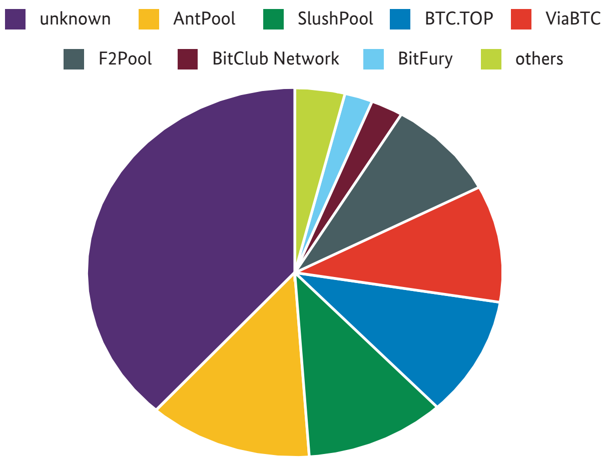

Figure 9: Distribution of hash rate among mining pools (average over the time period August 2018 till January 2019) [163]

pools might collude with each other or with the large section ‘unknown’.

Due to its conceptual similarity to PoW, the consensus mechanism PoET is vulnerable to similar attacks. An attacker needs to control at least half of the hardware security modules within the network in order to launch a 51 % attack. Since PoET does without an incentive system, it is not possible to transfer the selfish mining attack. On the other hand, the security module constitutes a new target for attacks. If an attacker is able to exploit vulnerabilities in the security module, he can virtually control the creation of blocks. The workaround of blacklisting security modules which generate the lowest waiting time much more often than expected provides a way to mitigate this problem [24].

The consensus mechanism PoS is also vulnerable to an analogue of the 51 % attack. In this case, an attacker would have to control a certain proportion of the funds deposited. The exact size of this proportion depends on the concrete design. It might however be lower than 51 %. Whether or not a 51 % attack can change blocks already embedded in the blockchain is also dependent on the concrete design. In case the chosen design does not guarantee finality of blocks, an attacker with sufficient funds at his disposal can tamper with old blocks without incurring additional costs, unless suitable countermeasures are applied. [23] discusses further attacks that might be possible, and how to combat them. In general, the algorithm is less well-studied than PoW as a result of being less widespread.

7.1.3 Network

Initiating a great many micro-transactions to be processed by the miners can be seen as a classical form of a distributed denial-of-service attack (DDoS attack). This DDoS attack targets the network as a whole and causes transactions of honest nodes to be processed and integrated into the blockchain much more slowly. In extreme cases, this leads to users losing confidence in the blockchain. An attacker may benefit from that, e.g. by speculating for a fall of a cryptocurrency. Countermeasures include charging transaction fees as well as assigning low priority to mass transactions by the same sender.

One can also try to systematically cut off individual nodes from the rest of the network using classical DDoS attacks and malware. If such an attack is successful against a PoW miner, his computing power is withdrawn from the network. The attacker thus obtains a higher share of the total computing power for a short time and increases his probability of mounting a 51 % attack successfully. By exploiting vulnerabilities in order to install a malware allowing remote code execution, it would even be possible to take over the computing power of a rival miner. If attacks target exchanges or other services connected to the network, the attacker may cause harm to his competitors. The Bitcoin network repeatedly witnessed such attacks in the past [93].

Another way to cut off a node from the network is to reroute his communication via servers belonging to and controlled by the attacker. This is called an *eclipse attack*. In this attack, when a node restarts its connection to the peer-to-peer network, it is tricked into establishing connections only to network nodes controlled by the attacker. The attacker achieves this by sending messages which systematically manipulate his victim’s view of the available network nodes.

Since the node cut off from the network is tricked into believing its connection is working as usual, it is not immediately aware of having fallen victim to an attack. If it renders services outside the blockchain, it can easily fall prey to a double spending attack [94]. An eclipse attack may be launched as an infrastructure attack or as a botnet attack. These two attacks differ in the IP address ranges used. In an infrastructure attack, the IP addresses are contiguous in blocks, whereas in a botnet attack they lie in diverse address ranges. According to [94], 32 IP address blocks or 4,600 bot addresses, respectively, suffice to launch an eclipse attack having success probability of 85 % in a network of arbitrary size.

The attacks presented mostly concern public unpermissioned blockchains. What makes them possible is that in this setting only data is authenticated, yet the keys used to this end are not assigned to a concrete communication partner (see section 2.1.5). In particular, eclipse attacks are not possible if the identities of individual nodes are known and documented using additional measures.

7.1.4 Implementation

Apart from the conceptual vulnerabilities inherent in the technology, developers may make mistakes when implementing a blockchain system. For example, [95] provides a list of well-known Bitcoin vulnerabilities. However, vulnerabilities may also have been incorporated into the code on purpose, for instance, if the operator of a blockchain application integrates a backdoor for his own benefit, as was the case with the cryptocurrency Soarcoin [96].

Faulty implementations are especially critical when they concern cryptographic routines, since they can break their security. One example is from the Bitcoin system. In this case, a faulty implementation of ECDSA allowed computing private keys. The reason was that the ephemeral keys used for signing different transactions were too short or not independent or were even reused multiple times [97], [98]. Besides such weaknesses in key generation, side-channel attacks during signature creation may allow obtaining keys (see figure 10). In case of a faulty implementation of the hash function, attacks may be possible although the hash function is theoretically secure (see section 7.1.1).

If users are allowed to overly influence security-related parts of the protocol, this may also be considered a faulty implementation. For instance, the security of PoW depends on the difficulty of the puzzle, which is usually determined by the interval between the last block publications. If it is possible to tamper with the block time stamp, an attacker can suggest that more time has passed between the creation of two blocks than is actually the case. Consequently, the difficulty is decreased and the attacker can create new blocks within short intervals and receive a reward in return. For example, in an incident in the cryptocurrency Verge an attacker was able to create one block per second for three hours and claim 1.3 million euro of block reward [99].

In a broader sense, the transaction malleability attack may also be considered a faulty implementation. The attack is possible if the same operation may be implemented using different commands from the blockchain scripting language. This allows transmitting blocks within the network that have logically equivalent contents, but different hash values. An attacker may use this feature for his benefit [100].

The firmware of the hardware components used may also exhibit security flaws. One incident was reported in the area of mining hardware in Bitcoin. In this incident, the manufacturer Bitmain

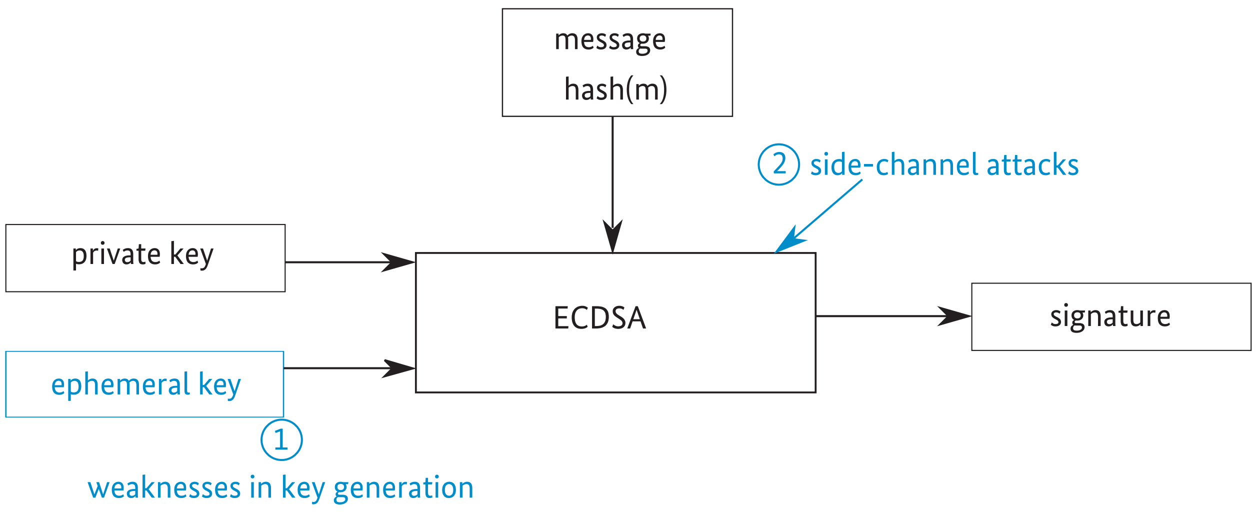

Figure 10: Faulty implementations may make cryptographic functions (in this case ECDSA) vulnerable to attacks.

was able to switch off devices via remote access. As the company is by far the largest producer of such hardware, exploiting the vulnerability would have rendered 51 % attacks on the Bitcoin blockchain possible [101], [102].

The faulty implementations described concern the core of the blockchain system. Additional functionality—like smart contracts—implemented by individual users needs to be considered separately (see section 7.2.1).

7.2 Security of smart contracts

For smart contracts, the immutability of blocks implies that source code can no longer be modified once it is embedded into the blockchain. It is thus essential that the code be free of bugs. However, analysis has shown that a large part of existing smart contracts exhibits bugs, which are sometimes elementary. Some studies estimate their rate at about 45 % of all Ethereum smart contracts [103].

The implications of buggy smart contracts are diverse. Sometimes, imprecise programming simply creates unfairness between contracting parties (for instance, see section 4.3.2). This usually only concerns the parties directly involved. In many cases though, bugs allow attackers unintended access to functions or data of a contract, enabling them, for example, to withdraw funds without authorisation. Other bugs may cause a contract to become completely unresponsive. The funds stored in this contract will thus be irrevocably blocked. In the past, such bugs have already accounted for losses in the millions, and at the same time affected other supposedly unrelated contracts (see Parity hack, section 7.2.1).

In the worst case, bugs in individual contracts can undermine whole systems, on the one hand by blocking functionality—like a simple typo which temporarily crippled the entire cryptocurrency ICON in June 2018 [104]—, on the other hand by making users lose trust in the blockchain—like the prominent DAO hack, which provoked a hard fork of the Ethereum blockchain including the reversal of many transactions.

[105] and [106] provide a good overview of security risks of Ethereum contracts in particular; [107] discusses more special vulnerabilities. The following sections illustrate some security risks that may arise during the life cycle of a smart contract.

7.2.1 Elementary programming errors

The extent to which a language is prone to programming errors has a significant influence on the security of the code. Immature or newly devised programming languages are typically more vulnerable in this respect than established ones. For instance, the language Solidity, which was specially crafted for use with Ethereum, has attracted criticism for its inconsistent design, and using the language Serpent once discussed as an alternative is not recommended either, since it exhibits numerous design flaws [108].

Besides classical pitfalls in programming, using hard-coded addresses or parameters is likewise risky, since the immutability of the source code prevents changes that might be required in case information becomes outdated. An incident on the Ethereum blockchain in 2017 (Parity hack [109]) demonstrates how several of these vulnerabilities may interact: A faulty initialisation made it possible to deactivate a contract inadvertently (or on purpose). Other smart contracts relying on that contract and having hard-coded its address could not get away from it. Consequently, they were no longer functioning and their funds of more than 150 million US dollars are blocked forever.

Several research groups study approaches to automatically detect bugs in program code [110] and to formally verify smart contracts [103], [111], [106]. Since in principle the compiler used may not be trustworthy or malware may be purposefully incorporated only at the bytecode level, the authors of [111] in particular argue that it is preferable to check the bytecode rather than the source code of a program. More work is required in this field though, especially for blockchains beyond Ethereum.

7.2.2 Block creation

Once a contract is created, it is added to the blockchain. However, it is usually not the programmer of the contract, but rather the miners (or an ordering service, see section 4.1) who decide in what order and at what time transactions are added to a block. Smart contracts should hence neither depend on a special order of execution nor on the time stamp [103] (see also section 4.3.2). Furthermore, the popular recommendation on the Bitcoin blockchain which advises one to wait for some time until a transaction is embedded into the blockchain deeply enough is equally valid for smart contract platforms using consensus mechanisms which allow forks.

A contract whose execution requires a large amount of computation is vulnerable to the so-called *verifier's dilemma* [112]. It states that every node having to validate this contract faces the question of whether it actually wants to consume the resources required or whether it accepts the contract without checking it. If the cost is too high, there is a risk that a significant proportion of the nodes decides to accept the contract without checking it. In this way, contracts might end up in the blockchain which, for example, carry out payments without authorisation, are not in agreement with the system state or undermine the entire credibility of the system. Ways to mitigate the dilemma might include a suitable design of the incentive system (see sections 3.2.5 and 4.2.4) that also rewards work expended for validation, or the option to contest unverified results [113].

7.2.3 Catching undesired behaviour at runtime

It is a big challenge to anticipate and correctly handle unexpected behaviour that might arise during the execution of a contract.

The programmer needs to know which sections of the code will automatically throw exceptions in case of runtime errors and at which places he needs to catch them by hand. The answer depends on the programming language and is not always obvious. For instance, the sketchy way Solidity handles exceptions has drawn criticism, since it fails to automatically catch runtime errors precisely in many of those operations which are often used [105].

It is important to check whether a contract calls foreign code anywhere and thus (temporarily) passes execution control to another entity. As is exemplified by Solidity, such a call of foreign code can happen without the programmer being even aware of it: Common operations used for transferring funds implicitly call the so-called *fallback* function of the receiver, which the calling contract cannot control. In particular, this function may contain code which is harmful to the calling contract.

The term *re-entrancy* denotes the option to call a function in a smart contract again before its former call is completed. For instance, this becomes possible if a contract passes execution control to another contract as described above. That contract may call the original function once again, thus creating an infinite loop. Even though the blockchain system includes precautions for automatically aborting loops at some point—in Ethereum this is done once the gas budget is exhausted—, an attack may already have caused great harm by then. The well-known DAO hack exploited precisely this vulnerability [105].

7.2.4 Visibility of smart contracts

A smart contract needs to receive all necessary input data via a transaction before it can be executed. In public blockchains, transactions are usually visible to all nodes. Consequently, it is not possible to call a contract in private without taking additional security and/or encryption measures. A direct way to exploit this lack of confidentiality exists with gambling contracts. In this case, the second player can read off how much the first one staked and use this information for his benefit [114].

The implications on confidentiality of so-called private variables, as provided by e.g. Solidity, are somewhat more subtle. Private variables are variables that contracts store in special memory areas, where they are protected from being accessed by other contracts. The value that is assigned to such a variable is announced in a transaction though, and thus publicly visible to all nodes. As a result, the content of private variables is not private despite their name.

Furthermore, the immutability of a blockchain prevents contracts from being deleted. It is sometimes claimed that Ethereum provides a way to delete smart contracts using the `selfdestruct` command. Its name notwithstanding, this command does not erase the bytecode from the blockchain though, but only removes its connection to a special Ethereum address within the system state. Nodes will henceforth consider the contract non-existent and stop calling it, but it is still possible to study its bytecode using the blockchain history. Therefore, this command does not yield a solution for implementing the right to be forgotten (see section 9.3.2) as specified in data protection frameworks like the GDPR.

7.3 Attacks on the blockchain infrastructure

7.3.1 Virtual-real interface

As mentioned in section 2.1, the security guarantees provided by a blockchain (as with other technologies for storing data) only start holding once the data has been added to a block. In case data and events from the real world outside the blockchain are used for triggering changes to the state of a blockchain, as may in particular be the case with smart contracts, an attack targeting the security of data before it is included into the blockchain may appeal to an attacker.

Oracles cannot completely solve this problem, since the data they retrieve may already have been tampered with (see section 4.3.1). When injecting sensor data into the blockchain as is sometimes proposed, targeted manipulations of the physical data captured by the sensors are also conceivable. Such attacks are usually hard to detect.

7.3.2 Security of keys

Apart from cryptographic routines and their technical implementations, processes for key creation and management constitute a main point of attack within blockchain applications (see figure 11). These attacks are comparable to those targeting other services, e.g. online banking. Survey articles like [115] illustrate to what extent vulnerabilities have been successfully transferred to and exploited within the context of blockchains.

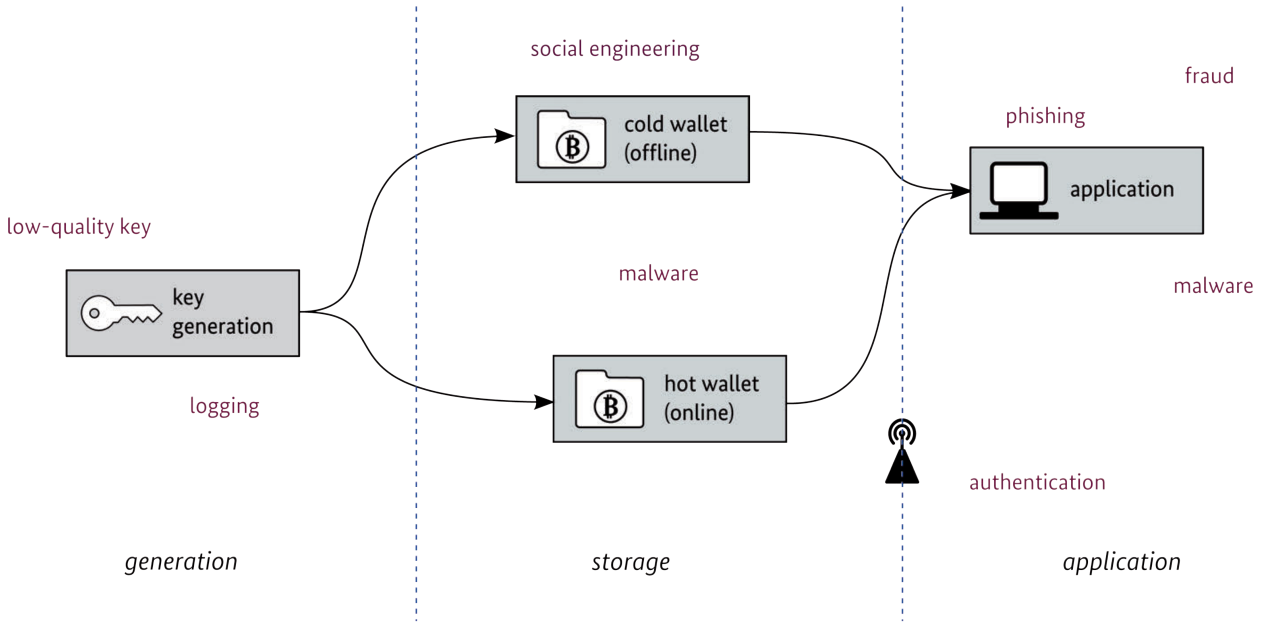

Figure 11: Practical attacks on cryptographic keys

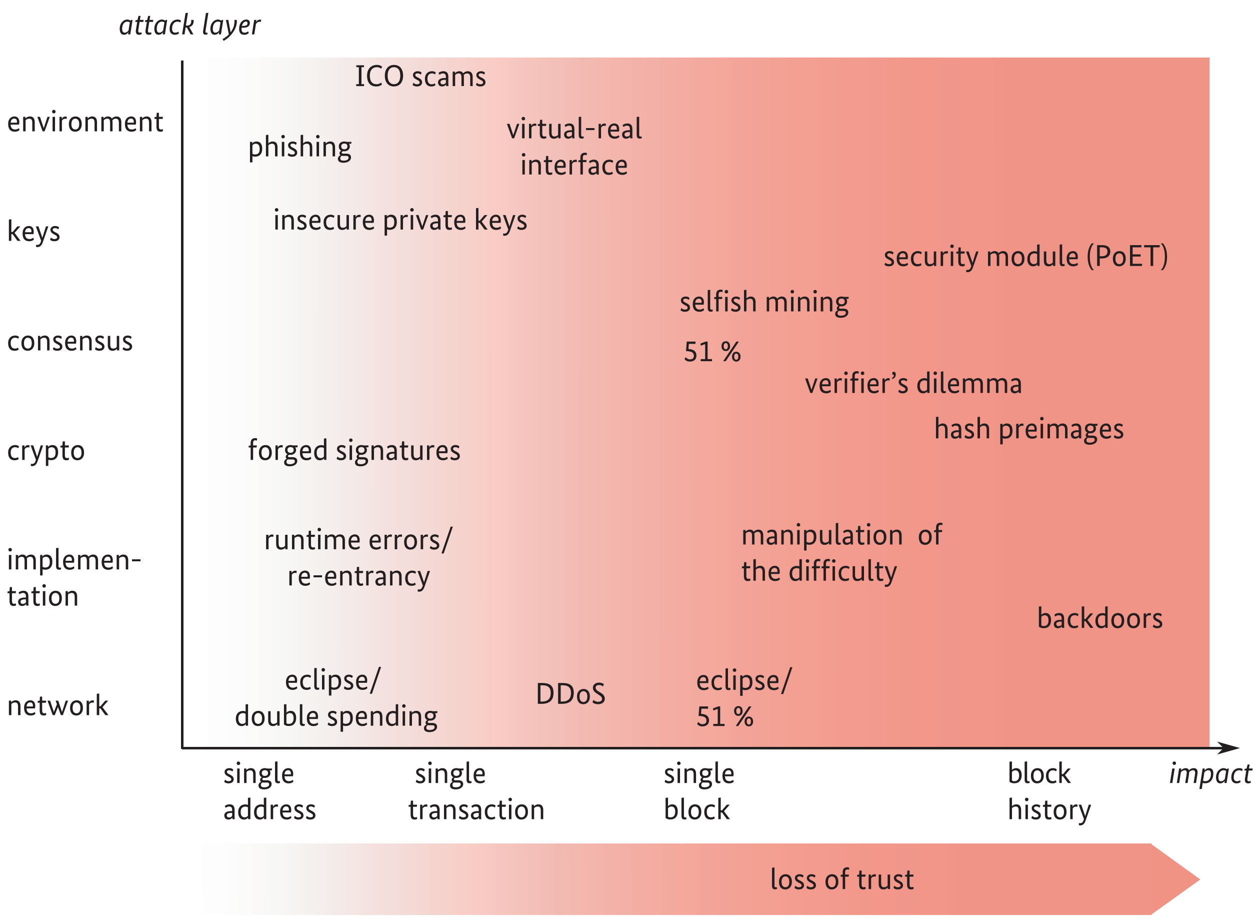

Figure 12: Overview of attacks on blockchain systems

Firstly, poor-quality keys may be used (see section 5.3). For instance, a so-called brain wallet is a method frequently used in cryptocurrencies for generating keys. It applies a hash function to a natural language password. In case the input values to the hash function exhibit too small an entropy, an attacker may however use a brute-force attack to recover the keys generated in this way [115].

Key creation and administration as well as interactions between users and blockchains are typically performed using third-party software (e.g. wallets), which most often does not provide any security guarantees. Several examples from the area of cryptocurrencies show that such software may well include backdoors integrated on purpose. One such case was the website iotaseed.io, which provided a service for key creation. During six months, that service seemed to work correctly. It recorded all the access data though, and eventually the entire funds of all affected wallets were stolen by the attackers [116].

Unintended vulnerabilities may be exploited using malware. The infrastructure of the blockchain system determines which parties an attacker may target. Depending on the system design, malware attacks may target users or directly be aimed at competing services (like wallet operators, crypto exchanges). Well-known examples include the campaigns Dridex and Trickbot, to which functionality for stealing cryptocurrency funds was added. Malware such as CryptoShuffler and Evrial read the clipboard for spying out access data [115]. The incident which caused the highest known damage involved an Infostealer implanted at the trading platform Coincheck. It allowed stealing funds in the cryptocurrency NEM from 260,000 accounts. The total funds stolen amounted to 532 million US dollars [117].

Furthermore, it is possible to interfere with the transmission of sensitive data. An attacker providing the address of his website as a response to a DNS request for the address of a crypto exchange may directly obtain the keys from his target.

7.3.3 Fraud

An attacker does not necessarily need to know his victim's private keys. Instead, he can trick a user into performing a legitimate transaction on the blockchain which benefits the attacker. Classical fraud methods may be used to this end. Within the context of blockchain, a prominent example are so-called ICO scams. Such scams work by embezzling investments into a new cryptocurrency. The incident involving the highest known damage happened in April 2018. The attackers misappropriated investments into the currencies Pincoin and Ifan amounting to 536 million euros [118]. Another option is to forge a party's public key—e.g. by compromising a website—and making users pay money to that key in exchange for services.

Apart from that, an attacker may use e.g. social engineering or phishing attacks to make a user reveal his private key. All the attacks mentioned in this section do not compromise the security of the underlying blockchain system.

7.3.4 General aspects of IT security

When using blockchains, all the typical measures for establishing IT security must be taken. The mere use of blockchains does not imply that one can dispense with them, as was already noted in [1]. What follows is a short list of the different general aspects to be taken into account and of the measures to be implemented. This list is not intended to be exhaustive.

- Security management: Security concept; documentation, auditing and updating of measures
- Emergency management: Emergency plan including responsibilities and immediate measures
- Infrastructure: Fire protection, (redundant) power supply
- Network security: Protection of internet access, firewalls
- Computer security: Access control, secure basic configuration, virus protection
- Operation: Logging including monitoring, updates and patches, backups
- Application level (permissioned blockchains): management of roles and permissions including access control, identification of parties
- Key management: secure creation, storage (including access control) and erasure

In order to determine in more detail which measures have to be taken, the specific protection requirements need to be assessed. The reader is referred to the BSI's IT-Grundschutz for a more comprehensive description [119].

Summary.

- Attacks on several components of the blockchain system are feasible and can have serious implications.
- The properties of the network directly influence the security of the entire blockchain system.
- Practical implementations can make algorithms vulnerable.
- The immutability of the code of smart contracts and their automatic execution require utmost care in programming.
- A great many security flaws in smart contracts are well known and have been observed in practice.
- General measures of IT security remain indispensable.

Part III Law

8 Legal aspects

Apart from technical issues, blockchain technology also raises legal questions. Assuming that the majority of its users is hardly looking into the general legal framework the technology is subject to, this section aims to set out a preliminary survey of aspects of civil and criminal law based on the current legal situation. Since this document does not focus on legal considerations, the following observations are only meant to provide a first overview. Aspects of data protection are discussed in chapter 9.

The following considerations are based on the current legal situation in Germany.

8.1 Aspects of civil law

A blockchain is designed to store all transactions in an immutable and tamper-proof fashion. This is where the potential of this technology lies, but at the same time the concept of immutability raises a variety of questions in civil law that have not yet been resolved conclusively.

For instance, one needs to address the question of what happens in case of erroneous transactions. Having initiated a transaction wrongly or erringly, is a party able to annul that transaction? May it declare avoidance or revocation? Furthermore, the question arises to whom such a declaration would have to be submitted.

One possible solution would be to exclude the instruments of civil law when using a blockchain. In this case, avoidances, withdrawals or revocations would have to be enforced outside the blockchain system. On public unpermissioned blockchains, one faces the additional problem that their participants at least employ pseudonyms. Consequently, an erring party may not even know who its counterpart is and which legal framework applies.

Other areas of civil law aiming to protect the parties to a contract cannot apply likewise. Legal transactions contrary to statutory prohibitions or public policy usually are void. However, whether the aforementioned is the case or not, is a matter of evaluation and needs to be assessed for each individual case, which the logic included in a smart contract is unable to do [120].

The blockchain does not take into account the protection of minors either. Whereas according to §§ 107 ff. BGB a contract is provisionally ineffective until ratified by a legal representative, transactions added to the blockchain will remain there. The notion of provisional ineffectiveness does not exist in a blockchain [120], [121].

Although it is possible to reverse transactions, doing so requires the other party's cooperation. If the original beneficiary of an erroneous transaction refuses to cooperate though, it would be possible to create a hard fork on purpose, which would replace the block containing the erroneous transaction. However, this would require convincing a sufficient number of nodes participating in the consensus mechanism that a certain block is to be considered erroneous [122] (cf. also section 9.3.2). That might turn out to be difficult or infeasible, in particular if several transactions were affected. In addition, all other transactions which happen to be contained in the same annulled blocks would also become void, even if they are in no way connected to the contested transaction. This fact might also have legal implications and possibly result in the annulment or reversal of uncontroversial transactions. Not only would this contradict the fundamental principle of the blockchain, but also the interests of the other parties affected.

8.2 Virtual-real interface

Regardless of the question of what a blockchain may achieve in technical terms, the legal background, which may be affected, must also be taken into account.

In particular, blockchains are used to ensure that data was not subsequently tampered with. From the perspective of IT security though, the question of whether real-world data entered into a blockchain is actually valid, i.e. agrees with reality, goes largely unnoticed. Although blockchains may guarantee that data has not been tampered with after its inclusion, the users do not have any means to check the validity of that data [123]. As remarked in section 7.3.1, potential attacks targeting the virtual-real interface must hence be taken into account. Briefly put, blockchains do improve transparency and auditability of data, yet these properties are only useful if the data entered is valid. A party checking the validity of data might be required to make sure this is the case.

The problem of the validity of data is closely linked to the authentication of nodes, more precisely to the assignment of keys to concrete communication partners (see section 2.1.5). It is probably impossible to solve without authenticating the nodes that enter data via the virtual-real interface or check its validity.

Considering this, one may dismiss the majority of ideas for replacing public registers at least regarding the question of whether by using a blockchain one can do without any intermediary. Whereas it would be possible in principle to run the land register based on blockchain technology, the author holds the opinion that a reliable intermediary would still be necessary for checking the validity of data entered into the land register.

Conversely, one needs to consider the question of to what extent entries in the blockchain concerning items in the real world are legally valid and enforceable. Unlike centralised solutions like databases, where the main tools to prevent misuse and manipulation of data are organisational measures and the legal framework (see section 2.3), blockchains do directly address these problems on the technical level. However, as with alternative technologies, registering in the blockchain a transfer of values outside the blockchain does not suffice in itself to ensure that this transfer is actually executed. This will rather require a legal system working to a sufficient extent. Considering this fact, proposals for using blockchains in states devoid of a functioning legal system, which aim to solve this problem by providing tamper resistance of data and transparency, need to be critically assessed as to the actual benefits they can yield.

8.3 Aspects of criminal law

Besides numerous questions from the area of civil law, one needs to ponder the question of whether a party operating a node may be liable to prosecution if the blockchain is abused for storing potentially criminal content.

The question arose in 2018 when researchers found that among other things the Bitcoin blockchain contains child pornography [124]. The German Criminal Code penalises several acts involving such content, e.g. displaying it publicly or even just possessing it (§§ 184 ff. StGB). In this context, data is already assumed to have been displayed publicly if it is possible to access it[1]. On the internet, it suffices to make data accessible over the internet for this to hold. It is hence not important whether the data was indeed shared, but the feasibility in itself is sufficient[2]. Since users of a blockchain download its content and provide it to others inside the respective network, at first glance one needs to assume that the content is displayed in the sense of the penal provisions cited. Prevailing case law acts on the assumption though that illegal content is only displayed publicly in the sense of §§ 184 ff. StGB if there is no need to overcome a ‘certain barrier’[3]. Such a barrier is typically supposed to exist if finding the content involves a significant effort (cf. also [125]). The case in question exhibits this feature. Although the data was stored inside the blockchain, it was not easily discernible. Consequently, the element of displaying data publicly should be supposed unfulfilled.

[1] BGH, order dated 12.11.2013 – 3 StR 322/13.

[2] BGHSt 47, 55, 60 with comments Gehrke ZUM 02, 283, Kudlich JZ 02, 310 and Lindemann/Wachsmuth JR 02, 206.

[3] BGH, judgment dated 22.05.2003, file number 1 70/03; BVerwG, judgment dated 20.02.2002 – file number 6 C 13/01.

The possession of pornographic content should also be ruled out in the example in question. In legal terms, possession is an actual power relation that in addition needs to be supported by an intention to possess[4]. Considering this definition, it is thus necessary to establish whether apart from the actual possession the node operator exhibits the respective intention to possess. This in turn must aim for the possibility of unhindered influence. Mere knowledge of the existence of a document or mere connivance of its possession by another party should hence not be sufficient[5] for being liable to prosecution under the provisions cited. This section considered the case of the node operator acting as the principal offender, whereas it did not explore other levels of participation.

As a result, one cannot flatly exclude criminal liability in the case in question. A node operator will typically not be liable to prosecution though, since he will not exhibit either one of the elements of the crime or at least the intention. There is no settled case law on this topic yet. Present case law concerning internet file-sharing sites is not applicable to blockchains either. For instance, in an order dated 2009/05/08 (Az.: 1 Ss 46/09) the German court OLG Oldenburg ruled that a user of internet file-sharing sites did not necessarily need to expect that he will provide the files he downloaded to other users automatically, owing to the implementation of the file-sharing site. Users of a (public) blockchain cannot claim this, however, since providing content to third parties is an inherent feature of blockchain technology. As individual data cannot be deleted from the blockchain (see section 9.3.2), most of its users probably condone the distribution of the data, since otherwise it would be impossible to use the blockchain. Therefore, one needs to wait and see how the legal status will evolve.

In general, questions concerning the storage of potentially criminal content mostly affect unpermissioned blockchains. In private and permissioned blockchains, appropriately managed permissions prevent or at least strongly hinder the integration of such data. In addition, possible offenders should be deterred by the fact that legal prosecution is significantly facilitated by the improved identifiability of all parties. Furthermore, it is conceivable in principle to subsequently delete data, see section 9.3.2.

8.4 Smart contracts

Smart contracts face the same problems as cryptocurrencies in terms of responsibilities, questions of liability, data protection (see chapter 9), and so forth. As mentioned in chapter 4, they are not contracts in the legal sense, but rather pre-programmed processes. Other jurisdictions—e.g. the US states of Arizona and Tennessee—have adopted laws though which aim to make smart contracts legally binding in the same way as legal contracts [126].

It is not possible to translate vague legal terms as often used in legal contracts (e.g. ‘appropriate’, ‘reasonable’, ‘immediately’) into precise if-then conditions as used in smart contracts without running into problems. Furthermore, due to its automatic execution a smart contract does not leave any margin when it comes to interpreting the content of a contract, e.g. determining the intention of the parties to a contract (§ 133 BGB). This is of particular importance since smart contracts are not written in natural language, and an average user not possessing substantial knowledge of the programming language used will typically be hard put to check whether conditions as stipulated in the smart contract actually correspond to his intention.

[4] BT-Drs. 12/3001 p. 5 with reference to the element of possession of § 29 subpara. 3 BtMG and BGH 26, 117; 27, 380; 30, 279, respectively.

[5] Schönke/Schröder StGB, 29th ed., 2014, § 184 b recital 15 with additional references.

Summary.

- Numerous legal questions concerning blockchain are still awaiting clarification. There is no settled case law yet.
- Questions in terms of civil and criminal law result in particular from the immutability of the blockchain.
- Smart contracts are not equivalent to contracts in the legal sense.
- Organisational measures are required in order to guarantee the validity of data entered via the virtual-real interface.

9 Data protection and data sovereignty

Controversial debates are currently going on about how blockchain technology can be used in compliance with data protection regulations. In particular, the central property of transparency and the use of cryptography in blockchains provide apparently good prerequisites for a privacy-friendly technology. At the same time, the permanent traceability of transactions, which is inherent in blockchain design and cannot be modified, poses a significant challenge in terms of data protection. The following sections briefly discuss the most important aspects.

This chapter was written in collaboration with the German Federal Commissioner for Data Protection and Freedom of Information (BfDI). It is based on the current legal framework in Germany, in particular the General Data Protection Regulation as a part of EU law.

9.1 Goals of data protection and IT security

As opposed to IT security, which encompasses the protection of IT systems and the data they store or process, data protection is about the protection of personal data against misuse. IT security is a fundamental building block for establishing data protection. Hence, both areas considerably overlap whenever personal data are managed using IT systems. The common security goals

- integrity,
- confidentiality,
- availability

bear witness to this. Challenges for achieving these goals concern both IT security and data protection and often admit the same technical or cryptographic solutions.

In addition to the security goals, German data protection regulations defined the additional goals

- transparency,
- intervenability,
- unlinkability

as well as the superordinate requirement of data minimisation even before the European General Data Protection Regulation (GDPR, [127]) came into force [128]. Solutions for achieving these goals are likewise often of a technical or cryptographic nature and are thus closely related to approaches from IT security.

9.2 GDPR and blockchain

The German data protection goals did not establish themselves outside Germany. However, they are now either incorporated into the fundamental article 5 of the GDPR or follow from the data subject rights as set down in articles 15–18 as well as articles 13 and 14 (information to be provided), article 25 (data protection by design and by default) and article 32 (security of processing), among others. Since the GDPR has come into force, within a European context these data protection requirements are hence directly derived from the principles and data subject rights as set down in these articles.

If blockchain applications are operated by several organisations or persons, articles 26 (joint controllers) and 28 (processor) may likewise apply, as well as the requirements of chapter V (transfers of personal data to third countries or international organisations) of the GDPR in case parts of the blockchain are processed in countries outside the European Union. In general, with blockchain applications it may be difficult to determine a data controller altogether. Several stakeholders, for instance, developers, operators and miners, are eligible; the concrete decision on who should be considered a data controller strongly depends on the purpose and kind of the blockchain though.

If blockchains are used in the processing of personal data, the blockchain application in particular needs to allow implementing the data subject rights as set down in articles 15–18 and 20 of the GDPR. They are about

- the right of access by the data subject (article 15),
- the right to rectification (article 16),
- the right to erasure ('right to be forgotten', article 17),
- the right to restriction of processing (article 18) as well as
- the right to data portability (article 20).

Recital 26 of the GPDR includes a statement important in the context of blockchain technology, namely that personal data which have undergone pseudonymisation should still be considered personal data if they can be attributed to a natural person by the use of additional information. Hence, such data are equally subject to the principles of data protection. The GDPR does not apply to data that have been reliably anonymised though.

The magnitude of the challenge involved in achieving the data protection goals or in implementing the data subject rights depends on the technical realisation of a blockchain application.

9.3 Implementability of data protection requirements in blockchains

A general analysis of the security goals integrity, confidentiality and availability in blockchains can already be found in chapters 2 and 5. The results obtained in these chapters evidently also apply to the handling of personal data. Therefore, the subsequent sections will only consider the requirements on blockchain use that are specific to data protection, and examine technical-cryptographic approaches for implementing the data subject rights described in section 9.2.

The associated legal aspects (see also chapter 8) as well as the question of who should be considered a data controller and hence responsible for the implementation of the data subject rights (see section 9.2) do not fall within the scope of the discussion, since their direct implications for the technical implementation are minor.

9.3.1 Right of access

The right of access essentially extends the requirement for transparency, that is, for the comprehensibility of data collection and data processing. Transparency is generally taken to be one of the main features of blockchains. If the blockchain in question is public or the user is endowed with appropriate reading permissions on a private blockchain, he can inspect his own transaction data and trace their history and links within the blockchain. It needs to be noted though that this only applies to data which have actually been stored and not only referenced in the transactions.

The implementation of the right of access becomes more involved if information on the controller, recipient, purpose of processing or the like is sought. In this case—depending on the blockchain model, but especially on public and unpermissioned blockchains—, for lack of regulation valid information can often not be given.

9.3.2 Right to rectification and right to erasure

Both the right to rectification and the right to erasure ('right to be forgotten') (or, more generally speaking, the requirement of intervenability) contradict the fundamental blockchain goal of tamper resistance and are overall difficult to achieve in blockchains. A rectification of data can in general only be implemented by adding a new transaction providing a new version of the data. However, this can only serve to update and not to overwrite data. Older versions are still preserved in the blockchain. In order to erase transaction data or to rectify them in the sense of replacing them, it would be necessary to modify transactions already embedded in the blockchain afterwards, which should be impossible due to the chaining of transaction blocks.

Nevertheless, there are some approaches allowing for an intentional revision of data:

- Rollback: In permissioned blockchains, one can appoint a privileged group of nodes, which obtains the permission to rewrite the history of the blockchain under certain conditions and to modify transactions afterwards.

- Forks: In case of an urgent need for revision, e.g. in case of serious security incidents, it is possible to enforce a fork of the blockchain, i.e. to cause a blockchain branch to split off. Alternative versions of the transactions in question can then be introduced and confirmed in that branch. To this end, either the users in their entirety or at least a certain critical mass needs to agree on supporting the fork and on contributing to the establishment of consensus in the new branch, in order to create the necessary trust in the new blockchain. This can pose a challenge in particular with unpermissioned blockchains. For instance, in the past, after the DAO hack (see section 7.2), this led to the splitting of the Ethereum blockchain into different branches (Ethereum and Ethereum Classic), which are now continued independently from each other.

- Storing data off-chain: In order for data not to be covered by the tamper resistance of the blockchain, they can be stored externally and only be referenced within the blockchain (see section 5.2). If data are referenced using a hash value, that value is no longer valid after a modification, but the respective transaction block can still be verified and the integrity of the remaining transactions and of the blockchain as a whole is preserved. Such an approach is for instance used for storing certificates in blockchains [129].

- Chameleon hashes: Chameleon hashes [130] provide a possible cryptographic solution. They are hash functions that allow constructing collisions—that is, different data hashing to the same value—using a trapdoor. By means of such collisions, data on the blockchain can be replaced without infringing the integrity protection of the blockchain [131]. In order to still guarantee the tamper resistance of the blockchain in principle, a trustworthy party must securely manage the trapdoor (or the respective cryptographic keys) or the keys must be divided to several parties using a secret sharing scheme, allowing these parties to use the trapdoor only in collaboration with each other.

- Mutable blockchains: A further approach for inserting intentional changes into the blockchain is in controlling the visibility of transaction data using smart contracts [132]. In this case, every transaction may consist of several versions. Only the currently valid version is visible in the network, whereas the other ones are encrypted and hence not accessible. This approach likewise requires arrangements for who manages the keys corresponding to the locked versions. Furthermore, the limitations of encryption concerning long-term security and crypto-agility as already discussed in section 5.1 again apply.

A feature shared by all approaches for modifying or erasing data in blockchains is that they override the basic ideas of the blockchain technology in a certain way. That can be problematic insofar as it creates new possibilities for attacks, e.g. by abusing the mechanisms for modifying data. In addition, especially with the two last-mentioned cryptographic concepts it is unclear how to preserve the consistency of the blockchain despite a revision of data. The concepts indeed allow replacing data or making them invisible; however, due to the links between transactions, this can create problems when validating or processing other data. Furthermore, one needs to decide for every application individually on how to deal with the arising data relics.

If data revision requires additional permissions or a superior position for certain stakeholders in the blockchain, the underlying trust model changes accordingly. In addition, owing to the distributed data storage within the blockchain network many local copies of the blockchain of different dates may exist on individual nodes. The revision may not cover these copies and they may still contain the old data.

9.3.3 Right to restriction of processing

The right to restriction of processing can be considered together with the requirement for unlinkability and fundamentally contradicts the construction of a blockchain as well. As a matter of principle, transaction data are visible to all blockchain participants and can be used by them, and the chronological order induced by the chaining of blocks allows linking transactions logically. No provision is made for a restriction of processing or segregation according to the purpose.

A possible approach for resolving this contradiction would be anonymous transactions, or smart contracts (see sections 5.1 and 5.4) enabling users to control the access to their data themselves and to make linking of transactions more difficult to outsiders.

If data are only referenced within the blockchain, but stored externally, the blockchain can include a permissions management that attaches access to the data to certain prerequisites and limits it in time.

In private blockchains, options for limiting the availability of data are provided for by construction. However, a fine-grained and variable access control including an appropriate modelling of roles still requires considerable effort.

9.3.4 Right to data portability

The right to data portability can be implemented quite easily depending on the technical realisation of the blockchain, at least in case all data are directly stored in the blockchain. However, the requirement to compile the data concerning a person and to make them available to that person 'in a structured, commonly used (...) format' (article 20 GDPR) may well pose a challenge. Considering the current state of the art, it seems doubtful whether the statement that data are already available in such a format in public blockchains would be convincing in any case. One cannot necessarily expect all persons of whom the blockchain might contain personal data to be able to extract the corresponding data from the blockchain on their own. In case of off-chain storage, at any rate there would arise the additional obligation to suitably pre-process data stored outside the actual blockchain and to make them available.

In summary, one finds that the use of blockchain technology for data protection applications faces challenges. These can partly be solved by organisational measures, e.g. using the general measures of off-chain storage or forks, partly by choosing a restrictive blockchain model. As a rule, private or permissioned blockchains offer considerably more room for enforcing data protection requirements. Technically mature solutions for most problems are not yet available though. As matters stand, blockchain is hence in general not suitable as a privacy-enhancing technology.

9.4 Data sovereignty

An argument often employed in favour of blockchains states that they facilitate the *data sovereignty*—as a part of the digital sovereignty—of the users. Since the external circumstances—in particular data protection regulations—decisively influence the digital sovereignty, the following sections discuss this topic as an aspect of data protection in the broader sense.

Digital sovereignty is in this context taken to mean the ability to 'act and take decisions autonomously in the digital sphere' [133], [134], [135]. Data sovereignty means among other things that the user stays in control of his data at all times and can determine who handled or accessed them. It is questionable though to what extent a blockchain application can satisfy these requirements.

Firstly, like alternative technologies blockchains always feature parties with special roles or of special influence, as already mentioned in section 3.1. A user must have special trust in them for safeguarding his digital sovereignty. These parties may for example be regulators, miners or the programmers of the blockchain software.

Miners can use their influence in establishing consensus for blocking certain transactions in a targeted way, which can limit the control of the affected users. This issue arose for instance in the context of the sale of seized bitcoins by law enforcement agencies [136]. Using several pseudonyms, as is recommended on many blockchains, does not provide a remedy. As noted in section 5.4, real identities can be attributed in many cases to such pseudonyms by analysing transaction data. As a consequence, transaction information of a user may also become comprehensible to third parties against the user's will.

The programmers of the blockchain software can introduce bugs intentionally or inadvertently into the source code. Adequately implementing several security building blocks, e.g. the generation of secure signature key pairs, is a very demanding task. Without an independent evaluation of the products, users devoid of special knowledge in mathematics are thus hard put to assess the security of blockchain solutions. However, such an evaluation by a central party stands in tension again with the blockchain goal of decentrality.

How users may enforce their legal rights in case there is no responsible party in a blockchain or they cannot access it, is furthermore an unresolved question.

Finally, the tendencies towards centralisation prevalent in the area of identity management must be taken into account. Here, the identity data of users are managed by central service providers and used in different contexts although segregation would be reasonable. A motivation for the use of blockchain is often stated to be the dismantling of just these centralised structures. On the one hand, a centralised identity management creates a higher potential for abuse, on the other hand, it increases efficiency and is in many cases perceived as more user-friendly.

Summary.

- Data protection and IT security largely overlap with respect to their requirements on blockchain technology.
- Within its scope of application, the requirements of the GDPR must equally be observed for the use of blockchains.
- The implementation of important data protection goals is difficult in blockchain applications.
- Blockchains are not readily suitable for facilitating the data sovereignty of users.

Part IV Practical aspects

10 Applications

In various business areas, the application of blockchain technology has been a subject of discussion. This chapter introduces several abstract scenarios and their underlying ideas, thus serving as a general guideline rather than as a collection of specific real-world examples. A particular focus is on identifying those aspects in each scenario that are essential for a secure and reasonable implementation.

Beyond these fundamental aspects, there is also the question whether blockchains can really meet the stated objectives. In particular, one of the most frequently mentioned reasons for introducing blockchain technology is to make central authorities unnecessary, but the examples below show that in many cases central external services—such as trust service providers for the creation and validation of certificates, electronic signatures, seals and time stamps—remain indispensable. As a general rule, prior to implementing any blockchain solution, the participating nodes and their respective roles have to be identified and a suitable permissions management has to be agreed upon. Here, legal and regulatory requirements may have to be taken into consideration.

In practice, the decision in favour of, or against, an adoption of blockchain technology in general or a specific implementation in particular will always have to consider the potential costs. However, economic aspects will not be treated here, as costs depend heavily on the specific use case and the given situation.

10.1 Proof of ownership

The transfer of assets has been the first and primary use case, in which the blockchain serves as a means of documentation of the confirmed transactions. Apart from Bitcoin and other cryptocurrencies, the deployment of blockchain has repeatedly been suggested for the energy sector and for various public registers. Hope is expressed that through the use of blockchain and especially due to the lack of a central authority, transactions can be executed faster and at lower costs. In the context of the renewable energy transition, it seems moreover appropriate and advantageous to use a decentralised trading platform for the trade of locally generated energy. Both in the energy sector and in public registers, various proposals have been put forward over the last years. But when it comes to practical implementations, until now most of them have turned out to be mere pilot projects of limited scope.

Major problems with the blockchain technology arise from its deficiencies in throughput and confidentiality, which have been discussed in sections 2.1.4 and 2.2.2, respectively. The throughput that can be achieved by public unpermissioned blockchains is very low, which at present prevents any large-scale application, for instance in the financial sector. In private or permissioned blockchains the throughput may be considerably higher, but it still cannot rival that of centralised solutions. Confidentiality (see section 5.1) can only, if at all, be achieved with a high loss of efficiency. This is unacceptable especially in applications where the transactions contain sensitive data. In some cases, applications with only few nodes could resort to private blockchains and thus make use of more efficient algorithms (see section 5.1) in order to mitigate the above-mentioned problems.

10.2 Process flow management

The use of blockchain technology has also been discussed in the context of business processes. Here, detailed information about the intermediate process steps can be stored in the blockchain, thus providing a means of documentation of the entire process flow. A common example is that of monitoring supply chains in global trade.

For this scenario, the virtual-real interface (see section 7.3.1) is crucial. The authenticity, integrity and completeness of the information—e.g. sensor data—that is to be stored in the blockchain must be guaranteed. Moreover, some of the data may be confidential or at least not to be disclosed to all nodes in the network. If this is the case, storing the data in the blockchain is not recommended (see section 5.1).

The parties involved will typically not be private individuals, but companies or government agencies. Their number is limited, as is the expected amount of information to be processed. Therefore, the deployment of a private permissioned blockchain may increase transparency and allow a traceable, tamper-proof documentation of all process steps.

10.3 Proof of integrity

Another use case is concerned with ensuring the integrity of documents by storing their hash values in a blockchain. If required, the integrity of each document can be verified at a later time by re-computing its hash value and comparing it with the stored value. This approach has already been discussed for various areas of application, for example, for academic or other certificates (see example below) or documents containing copyrighted material. Note, however, that even a successful verification can only confirm an exact match with the original, but can never bear testimony to its authenticity.

In particular, whenever the hash value is checked by someone other than its originator, a proof of the document's authenticity is essential. To this end, additional measures have to be taken. For example, a trustworthy party can be put in charge of publishing data in the blockchain and of proving its authenticity by providing a signature. The type of signature to be used and any additional requirements imposed on the signer must be chosen in accordance with the intended security level. In the European Union, for example, if the data is to be accepted as legal evidence it must be secured by qualified electronic signatures, seals or time stamps in compliance with the eIDAS regulation [52]. Likewise, for a legally valid preservation of evidence, the long-term negotiability must be ensured, i.e. the possibility to transfer the data to a different system while maintaining validity of the integrity and authenticity proofs.

It is to be noted that most of the recent blockchain-related ideas of how to generate time stamps are inappropriate in this context. According to these ideas, a certain time label—e.g. the block generation time—is attached to the data in the blocks and is thus used to prove their existence at that specific point in time. As a side effect of the hash chain, the time labels are organised in a totally ordered sequence. However, this sequence need not coincide with the true sequence in which the data was generated. It only represents the order in which it was included in the blockchain. Moreover, without additional measures, it is not possible to link these blockchain time labels to timestamps from other sources because they are no absolute values in themselves. Neither need they show the correct time. Both the consensus mechanism and the latency of the network may allow for a certain margin, hence leaving some room for manipulations. Large deviations from the true time may well be detected, but not easily prevented by the nodes.

In an extension of this approach, some unpredictable and non-manipulable information (e.g. lottery winning numbers, detailed weather maps) is attached to each block in order to prove that it was indeed generated at the alleged time. However, this only shows that the block was generated *after* the respective information became accessible. The question whether the block already existed *before* a certain event *(proof of existence)*, which is far more important for practical use cases, cannot be answered by this method.

For a legally compliant preservation of evidence [137], it is essential to be able to ensure the integrity of documents over long periods of time. This calls for adequate methods to renew the security measures whenever the security guarantees of the hash functions, signatures, other cryptographic routines or parameters that are currently in use are about to expire. So far, no suitable concepts for such a renewal have been established on blockchain systems. Further details can be found in section 6.1.

Example: Blockchain for academic certificates

The following approach to prevent forgery of academic certificates has been discussed and tested [138], [139]: It is based on a public permissioned blockchain where write permission is only granted to academic institutions. Each institution signs the hash value of any certificate it issues and publishes the signed values in the blockchain. If there happen to exist earlier versions of the same document (e.g. with erroneous entries), a reference to these earlier versions is added. The same applies if a certificate needs to be revoked.

Note that a random value *(salt)* of fixed length must be added to the document in order to avoid stereotypical data, which could facilitate the reconstruction of contents. The salt may be printed on the document in the form of a QR code. Both the paper document and an electronic version thereof are handed over to the certificate holder. If later the holder is asked to produce the certificate, the verifying authority can recalculate the hash value and check it against the value stored in the blockchain. For this, the verifier need not resort to external data—except to those that are necessary for validating the signature key—or even contact the issuing institution. Therefore, there are no purely technical reasons why an academic institution should keep copies of all issued certificates. It may, however, be obliged to do so to satisfy legal regulations.

Based on the current state of knowledge, even if a hash function is broken, it is still considered difficult to find a second preimage of exactly the same format. In particular, this holds true in case of authenticated electronic certificates in which data must be inserted in a predefined format. Security can be increased by also storing a check value—e.g. a few bits of the hashed salt—in the blockchain. It is likely that blockchains can help in preventing forgery of academic certificates, but they cannot offer perfect security. In case of doubt, an additional inspection of the original paper document is still recommended.

10.4 Identity management

A general idea is to use blockchains for verifying identities in the internet and for controlling the access to user accounts (authentication and authorisation). A common approach is to compute a hash value from a person's identity data and store it in the blockchain where it cannot be tampered with. To verify the person's identity at a later time, the hash value can be re-computed from their personal data and compared with the value in the blockchain. If these values match, it can be deduced that the person does indeed know the original identity data.

It is to be noted, however, that the blockchain in itself is not able to authenticate a person. Rather, the users have to authenticate directly to their communication partners, while the blockchain merely serves as a storage medium.

This general idea can be extended by a granular permissions management to control the access to personal data or parts thereof. Only those attributes that are needed for a certain application—e.g. whether a person is of age—are provided, thus mitigating the risk of identity theft.

However, when implementing these concepts, several questions arise. One of the critical issues for online-authentication schemes is the initial identification or the initial verification of identity information and the generation of the corresponding electronic identities, i.e. the link between a person's identity data and a means of authentication. The identity verification must be protected from manipulations, and it has to be performed at a sufficiently high level of assurance [140]. Likewise, the process generating the electronic identities has to ensure the correctness and authenticity of the stored identity information at an appropriate security level. Both problems require additional measures to be taken outside the blockchain. This is because any security guarantee the blockchain offers, in particular on data authenticity (see section 2.1.5), is only valid once the data is published in the blockchain.

The above authentication process is susceptible to numerous attacks. Therefore, suitable security measures have to be taken in order to prevent attackers from authenticating with false identities. Standard attacks are brute-force attacks on hash values if they are based on structured data (see section 5.2), or the interception of messages and misuse of the contained information (replay attack).

Some of the existing projects are based on public unpermissioned blockchains. In this case, no guarantee can be given that the identity information is promptly included in one of the next blocks. Moreover, the transaction fees of most of the currently used blockchain systems may vary considerably over time and are thus difficult to predict [141].

Further, it is to be examined whether blockchain technology is at all compatible with the notion of disabling or withdrawing electronic identities. Since the security of identity data must be preserved for decades, practicable mechanisms to establish long-term security are of utmost importance (see section 6.2.3).

Legal questions, too, must be considered (see chapter 8). They concern legal responsibilities for the contents and the operation of the blockchain, as well as data protection issues, e.g. specifying who is entitled to process personal data, or proving that pseudonymised data cannot be attributed to specific individuals (see section 9.2). In the described scenario, anonymisation is impossible in principle. This implies that special care must be taken to prevent that blockchain data is used to track user activity.

Summary.

Depending on the use case:

- The limitations of blockchains concerning confidentiality and efficiency must be taken into consideration.
- The authenticity of data and communication partners can only be achieved by additional measures outside the blockchain.
- Legal requirements with respect to e.g. signatures and time stamps must be met.
- Concepts for achieving long-term security are essential.

11 Further development

Blockchain technology spans a very large area and comprises many different implementations. Nevertheless, blockchain itself is only a special case of the significantly wider term of distributed ledger technologies (DLT), as already mentioned in section 1.1. Essentially, in DLT, the data structure is defined in a more general way as compared to blockchain, and it requires neither the formation of blocks nor a chain structure.

The future development in the area of blockchain and DLT is very hard to foresee at the moment. However, there exist some approaches for implementing DLT using alternative data structures or for combining it with other currently investigated technologies (quantum computing, quantum communication, artificial intelligence (AI)). Most of these concepts are relatively new and part of them has not at all been practically implemented yet. The combination of blockchain with quantum technologies or AI may in itself rather be due to the fact that one seeks to link up several current topics. Therefore, mature concepts for many security issues do not yet exist, and in particular, a thorough examination by the expert community is missing. In very general terms, at present, it is not possible to assess whether and to what extent the different approaches will attain their technical goals.

The following sections introduce some further developments.

11.1 Block lattices and double chaining

An obvious generalisation of the blockchain data structure is the multi-dimensional extension of the chain structure, in particular by using a lattice structure. The cryptocurrency Nano [142] and the distributed applications platform Dexon [143] feature the first implementations of such a block lattice.

In Nano, every participant maintains his own asynchronous blockchain containing the history of his own account balance. Transactions among participants are publicly recorded in a global chain as changes to their current balances. The consensus mechanism of the global chain is a so-called delegated proof-of-stake (dPoS), a special instance of proof-of-stake (see section 3.2.3). In contrast, Dexon allows every node in parallel to build up its own blockchain containing all transactions distributed in the network. At the same time, consensus on the validity and the time of a block creation is established across blockchains using a BFT algorithm, and this consensus is recorded on a global chain.

Using such constructions one aims to increase the scalability of the system and to reduce the latency as well as the energy consumption within the network—especially compared to Bitcoin—, although the solutions continue to be public unpermissioned distributed ledgers.

A further generalisation consists in replacing the one-way (directed) chaining of blocks in a blockchain by two-way chaining [144]. Hence, unlike for instance in Bitcoin, there is no insecure end of the chain—the integrity of the last block is likewise protected, since it is chained to the penultimate block—, and in addition this solution is supposed to allow hiding blocks with sensitive contents from public access or deleting blocks from the chain.

11.2 Directed acyclic graphs

The concept of directed acyclic graphs (DAG), which replaces the chain structure of a blockchain by a mesh of transactions, is even wider. The most well-known example is the cryptocurrency IOTA [145] targeting communication and payments in the Internet of things (IoT). In this setting, transactions form the vertices of a graph. The consensus mechanism requires a node to confirm two transactions and their conflict-freeness for every new transaction it wishes to include. In the graph, directed edges run from the new transaction to the confirmed transactions (see figure 13).

At the same time, creating a valid transaction requires computing a proof-of-work (for easy parameters compared to Bitcoin) which is based on computing special hash values that also include information on the confirmed transactions. Transactions that have been confirmed by sufficiently many new transactions either directly or indirectly (as represented by an edge or a path in the graph) are considered verified and hence secure. This structure aims to improve scalability and throughput and to reduce transaction costs in comparison with blockchain-based cryptocurrencies. The stability and the resiliency of a DAG, however, strongly rely on the choice of the preceding transaction to be verified, which calls for the development of complex probabilistic models. In addition, the graph structure gives rise to a range of new attack scenarios, for instance for double spending attacks [146].

11.3 Interoperability

In the area of blockchain and DLT, there already exists a plethora of possible models and application-specific implementations of the technology. Since international standardisation has started relatively late, in recent years in several fields of application both quasi-standards and a high number of individual solutions have emerged, which are mostly not mutually compatible (see section 12.1). Therefore, the same information may be used in different blockchains without synchronisation or users may have to run their applications on several blockchains in parallel.

By now, there are some technical approaches for consolidating the proliferation of blockchains or for controlling it by a coherent superstructure. Such solutions are often termed ‘cross-chain communication’ or ‘interchain communication’. Examples for respective frameworks include Cosmos [147], Polkadot [148] or Overledger [149], which promise interoperability and a secure communication across different blockchains. In many cases, there is a superordinate blockchain using a global consensus mechanism and many subordinate specialised blockchains with independent data processing. Creating trust (see also section 3.1) can turn out to be very complex in such concepts, though.

11.4 Channel networks

One option for improving scalability and privacy in a (public) blockchain is to outsource transactions from the blockchain and to process them in a separate network. To this end, a channel is created between two participants in the network, which allows exchanging payments or, more generally, data up to an agreed upon limit. Only the opening and closure of the channel and the respective balances are recorded on the blockchain by a transaction. Using a smart contract on the blockchain, one can force correct behaviour of the participants in the payment channel. Such a channel network provides the advantage that payments inside the channels are free of charge (which also allows making micropayments), the

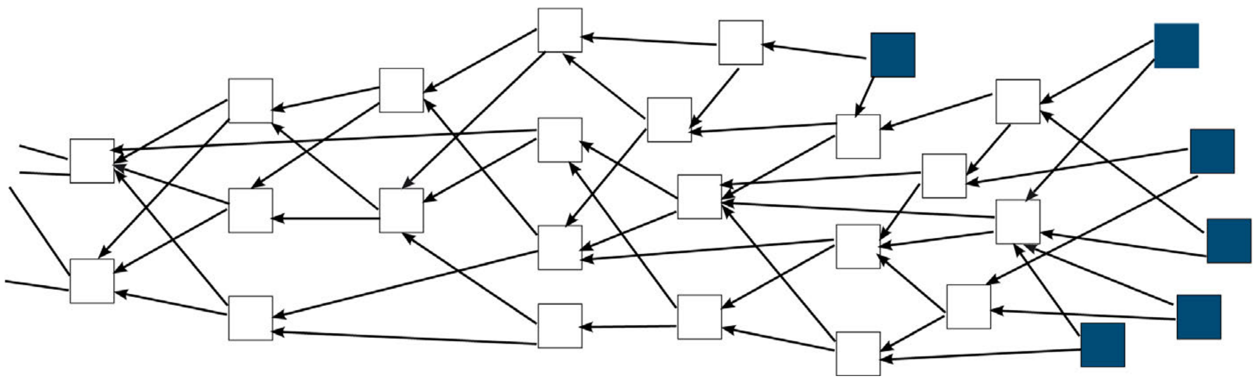
Figure 13: Directed acyclic graph in IOTA (‘Tangle’), cf. [146]

throughput is in principle only limited by the capacity of the network (and not by the block size or rate of the blockchain) and at the same time the privacy of the participants is protected better than on the blockchain, since not every single transaction is stored on the blockchain. The most well-known example for such a network of payment channels is Bitcoin's Lightning network [150].

11.5 Quantum blockchain

Section 6.1 already presented the potential threat to blockchains posed by quantum computers and a possible mitigation by using quantum computer resistant cryptography and crypto-agile solutions.

On the other hand, there already exist ideas for enhancing the blockchain technology by using quantum-mechanic concepts. The keyword 'quantum blockchain' may denote different approaches:

- The blockchain is based on a quantum key distribution (QKD) network. As a result, the blockchain participants can securely exchange keys and authenticate each other. The ensuing network communication is then protected by these keys. Such a system has already been implemented and tested [151].
- The blockchain network itself consists of quantum computers, the transactions in the blocks are represented by entangled photons and consecutive blocks are linked using entanglement in time [152]. Since fully-fledged quantum computers are not yet available and the concrete quantum theoretical properties of photons entangled in time have not been investigated yet, such ideas are so far purely theoretical.
- Blockchain transactions are transmitted using quantum teleportation in a classical network [153]. This prevents double spending attacks (see section 7.1.2), since it is in general impossible to clone quantum information reliably. The transactions are verified and protected using quantum cryptography (cryptographic schemes based on quantum mechanical effects, e.g. QKD). At present, such concepts are still very speculative, as the quantum theoretical foundations for such applications are not yet mature and are partially still in an experimental stage.

11.6 Blockchain and artificial intelligence

A further possible combination of technologies consists in connecting blockchains with artificial intelligence.

The approach of cooperative machine learning, which is relatively new in the area of AI research, allows devices in a distributed network to jointly compute a shared prediction model, where the training data remain locally on the devices. One problem in this context are malicious participants, who can perturb the correctness of the training. A blockchain system as a basis might provide an incentive for participating in cooperative learning in accordance with the rules and yield the opportunity for a reliable audit of the system. DeepChain [154], for instance, is a prototypical implementation of this idea based on the blockchain system Corda.

Another approach consists in establishing a blockchain network bringing together nodes that want to set up and train neural networks and to use existing neural networks (against payment of a fee), nodes that can provide their neural networks or training data (for remuneration), or nodes that provide their computing power for neural networks as miners (for remuneration). In such an application scenario, appropriate consensus mechanisms for the blockchain must be developed that can guarantee, apart from classical proof-of-work for miners, that training data were used for training a neural network or that a neural network arrives at the right conclusions. An example from healthcare is the project Skychain [155], which is still being tested.

Summary.

- There are several ideas for generalising blockchain technology or combining it with other new technologies.
- Most further developments based on blockchain are not yet mature and have not been investigated from a security point of view.
- Interoperability between blockchains is hard to achieve without standardisation.

12 Standards and regulation

As blockchain is a relatively new technology, few results are available so far in the field of standardisation and regulation. Due to the wide scope of application provided by blockchains, however, harmonising and controlling the technology would be sensible and advantageous in many areas. Besides IT security, questions arise in particular in legal, economic and socio-technical terms. The purpose of this chapter is to briefly present the existing endeavours and first results from standardisation and regulation in the national and international context (as of January 2019). The focus is on issues of IT security. Sector-specific regulations are out of scope in this chapter.

12.1 Standardisation

In the area of standardisation of blockchain, plenty of activity has been going on recently. By now, blockchain technology has reached a certain level of maturity and distribution, and in many fields of application the use of blockchain is contemplated. At the same time, it is becoming obvious that the understandings of the definitions and limits of the technology sometimes significantly differ and that there are very diverse implementations which are not interoperable (see section 11.3).

In practical applications, some quasi-standards have emerged by now, e.g. R3's open source blockchain platform Corda [54] or the Linux foundation's blockchain framework Hyperledger [8]. These solutions are not compatible among each other either.

The development of standards for blockchain technology can contribute to clarifying the general framework of the technology and to facilitating the use of blockchains.

At the International Organization for Standardization (ISO), a commission (TC307, [156]) led by Australia and founded as early as the beginning of 2016 has been charged with working out a standard for blockchain and distributed ledger technologies. The German Institute for Standardization (Deutsches Institut für Normung, DIN) has established a corresponding national mirror committee, which includes the BSI as an observing member. Issues of IT security are treated by the working group 'security, privacy and identity' and prepared for the standard. According to the plan, standardisation work should be finished by 2021.

At the European Telecommunications Standards Institute (ETSI), a committee (ISG PDL, [157]) is working on standardising permissioned distributed ledgers. The Institute of Electrical and Electronics Engineers (IEEE) is conducting standardisation projects in the blockchain context (publication series P2418, [158]), which among other things are developing blockchain frameworks for the areas of IoT and automotive. The US National Institute of Standards and Technology (NIST) as well as the American National Standards Institute (ANSI) are likewise working on blockchain. At the network level, the Internet Engineering Task Force (IETF) and the World Wide Web Consortium (W3C) are engaged in work on internet standards for a consistent identification of resources accessible in the network and for addressing blockchains [159], [160], which are aimed at creating interoperability of network communication between different blockchain systems.

12.2 Regulation

One of the frequently emphasised advantages of blockchain technology is the possibility to conduct transactions between participants without a central party (or at least reducing the dependency on central authorities, see section 3.1). However, this entails problems and insecurities, for instance in case of damage or a dispute (see section 8.1) or when reacting to security issues. Furthermore, the intentional pseudonymisation inherent to many public blockchains engenders insecurities for certain applications. For this reason among others, governmental regulation is sensible for many fields of application. As blockchains are often transnational, an international coordination concerning regulation and control of blockchain technology is necessary in many cases.

In the European Union (EU), a collaboration of EU member states and Norway was started in 2018 within the framework of the European Blockchain Partnership (EBP, [161]). It aims to create a transnational European blockchain infrastructure (European Blockchain Services Infrastructure, EBSI) for supporting the establishment of transnational digital public administration services. Within this collaboration, one intends to develop interoperable frameworks, standardised solutions as well as regulation and control structures for blockchain use. Besides, it wants to foster research and development activities in Europe in the context of blockchain. A focus is on security and data protection. First results are expected for 2019.

In Germany, the federal government is working out a national blockchain strategy—based on the coalition agreement from 2018—, which it plans to present in summer 2019. The focus of this strategy lies on establishing a secure legal framework, supporting pilot projects, identifying fields of application in the public administration, fostering research and connecting administration, industry and science. A public consultation served to obtain the opinion and expertise of different stakeholders in the blockchain area, in order to take into account the needs of all affected parties. During the elaboration of the blockchain strategy of the federal government, the BSI was consulted for topics from IT security. In addition, in Germany several government agencies are responsible for the concrete regulation of blockchain technology in different application areas. For instance, the Federal Financial Supervisory Authority (Bundesanstalt für Finanzdienstleistungsaufsicht, BaFin) has already worked out a regulatory assessment of cryptocurrencies and initial coin offerings (ICOs) [162].

In the area of IT security, the BSI in its capacity as Germany's national cyber security authority is responsible for investigating and assessing the blockchain technology. The document at hand provides a first compilation of the potentials and challenges of blockchains with respect to their security-related properties as well as an assessment of the technical-cryptographic measures and corresponding recommendations for a secure use of blockchains. Besides, a security certification of blockchain products or selected components may be useful for certain applications in the future. This requires the use of cryptographic algorithms that have been approved for blockchain applications by the BSI, since an individual assessment of proprietary algorithms is infeasible within the framework of certification.

Summary.

- A common understanding of blockchain technology is important for secure and interoperable blockchain applications.
- Several organisations promote standardisation in the blockchain area, but so far, it has seen few results.
- Intense work in the regulation of blockchains is being performed on the national as well as international level.

Glossary

This section introduces some basic cryptographic terms to the extent which is necessary for understanding the document at hand. The explanations often closely follow BSI's Technical Guideline TR-02102 [50].

Asymmetric encryption schemes can be used for the encryption of data and for creating *digital signatures* for authenticating data. Unlike symmetric encryption schemes, they do not require the communicating parties to have a common secret key. Instead, each user creates a pair of keys consisting of a public key and a secret key only known to himself. It has to be practically impossible to reconstruct the secret key from the public key. The public key can be used to encrypt data for the owner of the private key or to check his signatures.

Collision resistance of a hash function means that it is practically impossible to find two different bit strings hashing to the same value.

Cryptanalysis denotes the analysis of cryptographic mechanisms for breaking them or proving their security.

Digital signatures are used for authenticating data. In signature algorithms based on *asymmetric encryption* schemes the data to be signed is hashed first and, based on this hash value, the signature is then calculated using the secret key of the prover. The verifier then checks the signature using the corresponding public key. If the check is successful, this guarantees that the respective data has not been tampered with by unauthorised parties in the meantime. At the moment, the algorithms *RSA* and *ECDSA* are commonly used for digital signatures.

The **discrete logarithm problem (DLP)** on an elliptic curve E consists in computing the value $a \in \mathbb{N}$ from the points $P \in E$ and $aP \in E$.

ECDSA (Elliptic Curve Digital Signature Algorithm) is an asymmetric encryption scheme for *digital signatures*. Its security is based on the assumed difficulty of the DLP on elliptic curves.

Entropy of data denotes the average information they contain. Data of low entropy are redundant or exhibit statistical regularities.

The **factorisation problem** within the context of RSA consists in factoring a large number $n = pq$ into its prime divisors p and q.

Hash functions map a bit string of any length to a bit string of a fixed length. These functions play an important role in many cryptographic mechanisms, for example when deriving cryptographic keys or when authenticating data. Hash functions which are used in cryptographic mechanisms must meet the three conditions *one-way property*, *second preimage property* and *collision resistance* depending on the application. A hash function which meets all three conditions is called cryptographically strong.

Homomorphic encryption denotes encryption schemes in which a mathematical operation like addition or multiplication on the ciphertext results in the respective operation on the plaintext after decryption.

A **message authentication code (MAC)** is a symmetric method for data authentication. Therefore, the communicating parties must have agreed upon a shared symmetric key beforehand.

One-way property of a hash function means that, given a hash value, it is practically impossible to find a bit string hashing to that value.

A **public key infrastructure (PKI)** is used in asymmetric encryption schemes for guaranteeing the assignment of a public key to a fixed communication partner and to fend off so-called man-in-the-middle attacks. In a man-in-the-middle attack, an attacker intercepts communication between two parties and tampers with it. In a PKI, the assignment of a public key to a communication partner is attested by a central trustworthy party.

RSA is an asymmetric encryption scheme for *digital signatures*. Its security is based on the assumed difficulty of factoring large numbers $n = pq$ into their prime divisors p and q.

The **second preimage property** of a hash function means that, given a bit string, it is practically impossible to find another bit string hashing to the same value as the given bit string.

Secret sharing schemes allow dividing a secret into shared secrets so that all shared secrets (or a certain subset thereof) are required for reconstructing the secret. This can be used, for instance, to distribute a secret key to different users so that a certain quorum of users is required for reconstructing the key.

Index

List of abbreviations

AI	artificial intelligence	66
API	application programming interface	27
BFT	Byzantine Fault Tolerance	20
BGB	Bürgerliches Gesetzbuch (German Civil Code)	52
BGH	Bundesgerichtshof (Federal Court of Justice)	53
CFT	Crash Fault Tolerance	20
DAG	directed acyclic graph	66
dAPP	distributed application	26
DDoS	distributed denial of service	44
DLP	discrete logarithm problem	72
DLT	distributed ledger technology	66
dPoS	delegated proof-of-stake	66
ECDSA	Elliptic Curve Digital Signature Algorithm	72
EVM	Ethereum Virtual Machine	26
GDPR	General Data Protection Regulation	56
ICO	initial coin offering	71
IoT	Internet of things	66
MAC	message authentication code	72
P2P	peer-to-peer	9
PBFT	Practical Byzantine Fault Tolerance	21
PKI	public key infrastructure	73
PoET	proof-of-elapsed-time	23
PoS	proof-of-stake	23
PoW	proof-of-work	22
PoX	proof-of-X	22
QKD	quantum key distribution	68
sMPC	secure multi-party computation	33
StGB	Strafgesetzbuch (German Criminal Code)	53
TEE	trusted execution environment	33
tps	transactions per second	17
ZK	zero-knowledge	32

Bibliography

[1] BSI, *Blockchain sicher gestalten – Eckpunkte des BSI*, 2018. https://www.bsi.bund.de/SharedDocs/Downloads/DE/BSI/Krypto/Blockchain_Eckpunktepapier.pdf.

[2] S. Nakamoto, *Bitcoin: A Peer-to-Peer Electronic Cash System*, 2008.

[3] Bitcoin Wiki, *Technical background of version 1 Bitcoin addresses*, 2018. https://en.bitcoin.it/wiki/Technical_background_of_version_1_Bitcoin_addresses, accessed: 2019-01-30.

[4] Ethereum, *White Paper: A Next-Generation Smart Contract and Decentralized Application Platform.* https://github.com/ethereum/wiki/wiki/White-Paper, accessed: 2018-04-17.

[5] G. Wood, *Yellow Paper: A Secure Decentralised Generalised Transaction Ledger, EIP-150 Revision.* http://paper.gavwood.com/, accessed: 2018-04-11.

[6] M. Dameron, *Beigepaper: An Ethereum Technical Specification, Version 0.7.2*, 2018. https://github.com/chronaeon/beigepaper/blob/master/beigepaper.pdf, accessed: 2018-04-16.

[7] E. Androulaki, A. Barger, V. Bortnikov, C. Cachin, K. Christidis, A. De Caro, D. Enyeart, C. Ferris, G. Laventman, Y. Manevich, S. Muralidharan, C. Murthy, B. Nguyen, M. Sethi, G. Singh, K. Smith, A. Sorniotti, C. Stathakopoulou, M. Vukolic, S. W. Cocco and J. Yellick, *Hyperledger Fabric: A Distributed Operating System for Permissioned Blockchains*, 2018. https://arxiv.org/pdf/1801.10228.

[8] Hyperledger Fabric, *hyperledger-fabricdocs master documentation*, 2019. http://hyperledger-fabric.readthedocs.io/en/latest/blockchain.html, accessed: 2019-01-30.

[9] BSI, *IT-Grundschutz-Kompendium – Glossar*, 2019. https://www.bsi.bund.de/SharedDocs/Downloads/DE/BSI/Grundschutz/Kompendium/IT_Grundschutz_Kompendium_Edition2021.html.

[10] Digiconomist, *Bitcoin Energy Consumption Index*, 2017. https://digiconomist.net/bitcoin-energy-consumption, accessed: 2018-07-31.

[11] A. N. Bessani, J. Sousa and E. A. Pelinson, *State Machine Replication for the Masses with BFT-SMART*, in *44th Annual IEEE/IFIP International Conference on Dependable Systems and Networks, DSN 2014*, Atlanta, GA, USA, June 23–26, 2014, 2014, pp. 355–362.

[12] BSI, *IT-Grundschutz-Kompendium – APP.4.3 Relationale Datenbanksysteme*, 2019. https://www.bsi.bund.de/SharedDocs/Downloads/DE/BSI/Grundschutz/Kompendium_Einzel_PDFs_2021/06_APP_Anwendungen/APP_4_3_Relationale_Datenbanksysteme_Edition_2021.html.

[13] Visa Inc., *Visa Inc. at a Glance*, 2015. https://usa.visa.com/dam/VCOM/download/corporate/media/visa-fact-sheet-Jun2015.pdf, accessed: 2017-08-21.

[14] R. Anderson, I. Shumailov, M. Ahmed and A. Rietmann, *Bitcoin Redux*, 2013. https://www.cl.cam.ac.uk/~rja14/Papers/bitcoin-redux.pdf.

[15] C. Cachin, R. Guerraoui and L. E. T. Rodrigues, *Introduction to Reliable and Secure Distributed Programming (2nd edition)*, Springer, 2011.

[16] C. Dwork, N. A. Lynch and L. J. Stockmeyer, *Consensus in the presence of partial synchrony*, J. ACM, 35, no. 2, pp. 288–323, 1988.

[17] L. Lamport, R. E. Shostak and M. C. Pease, *The Byzantine Generals Problem*, ACM Trans. Program. Lang. Syst., 4, no. 3, pp. 382–401, 1982.

[18] M. J. Fischer, N. A. Lynch and M. Paterson, *Impossibility of Distributed Consensus with One Faulty Process*, J. ACM, 32, no. 2, pp. 374–382, 1985.

[19] L. Lamport, *The Part-Time Parliament*, ACM Trans. Comput. Syst., 16, no. 2, pp. 133–169, 1998.

[20] D. Ongaro and J. K. Ousterhout, *In Search of an Understandable Consensus Algorithm*, in *2014 USENIX Annual Technical Conference, USENIX ATC '14*, Philadelphia, PA, USA, June 19–20, 2014, 2014, pp. 305–319.

[21] M. Castro and B. Liskov, *Practical byzantine fault tolerance and proactive recovery*, ACM Trans. Comput. Syst., 20, no. 4, pp. 398–461, 2002.

[22] C. Cachin and M. Vukolic, *Blockchain Consensus Protocols in the Wild*, 2017. http://arxiv.org/abs/1707.01873.

[23] V. Buterin et al., *Proof of Stake FAQ*, 2018. https://github.com/ethereum/wiki/wiki/Proof-of-Stake-FAQs, accessed: 2018-10-01.

[24] Hyperledger Sawtooth, *PoET 1.0 Specification*, 2017. https://sawtooth.hyperledger.org/docs/core/releases/1.0/architecture/poet.html, accessed: 2018-09-28.

[25] D. Schwartz, N. Youngs and A. Britto, *The Ripple Protocol Consensus Algorithm*, 2014. https://ripple.com/files/ripple_consensus_whitepaper.pdf, accessed: 2018-01-05.

[26] D. Mazieres, *Stellar Consensus Protocol: A Federated Model for Internet-level Consensus*, 2015. https://www.stellar.org/papers/stellar-consensus-protocol.pdf.

[27] Y. Gilad, R. Hemo, S. Micali, G. Vlachos and N. Zeldovich, *Algorand: Scaling Byzantine Agreements for Cryptocurrencies*, 2017. http://eprint.iacr.org/2017/454.

[28] S. Bano, A. Sonnino, M. Al-Bassam, S. Azouvi, P. McCorry, S. Meiklejohn and G. Danezis, *Consensus in the Age of Blockchains*, 2017. http://arxiv.org/abs/1711.03936.

[29] N. Stifter, A. Judmayer, P. Schindler, A. Zamyatin and E. R. Weippl, *Agreement with Satoshi – On the Formalization of Nakamoto Consensus*, 2018. https://eprint.iacr.org/2018/400.

[30] E. Heilman, N. Narula, T. Dryja and M. Virza, *IOTA Vulnerability Report: Cryptanalysis of the Curl Hash Function Enabling Practical Signature Forgery Attacks on the IOTA Cryptocurrency*, 2017. https://github.com/mit-dci/tangled-curl/blob/master/vuln-iota.md, accessed: 2018-01-18.

[31] S. De Angelis, L. Aniello, R. Baldoni, F. Lombardi, A. Margheri and V. Sassone, *PBFT vs Proof-of-Authority: Applying the CAP Theorem to Permissioned Blockchain*, in *Proceedings of the Second Italian Conference on Cyber Security*, Milan, Italy, February 6th–9th, 2018, 2018.

[32] F. Armknecht, G. O. Karame, A. Mandal, F. Youssef and E. Zenner, *Ripple: Overview and Outlook*, in *Trust and Trustworthy Computing – 8th International Conference, TRUST 2015*, Heraklion, Greece, August 24–26, 2015, Proceedings, 2015, pp. 163–180.

[33] S. Thomas, *Correction to Ripple White Paper*, 2015. https://ripple.com/dev-blog/correction-to-ripple-white-paper/, accessed: 2018-01-05.

[34] C. A. E. Goodhart, *Problems of Monetary Management: The UK experience*, Papers in Monetary Economics, 1, 1975.

[35] H. Siebert, *Der Kobra-Effekt: Wie man Irrwege der Wirtschaftspolitik vermeidet*, Piper, 2003.

[36] N. Szabo, *Smart Contracts*, 1994. http://www.fon.hum.uva.nl/rob/Courses/InformationInSpeech/CDROM/Literature/LOTwinterschool2006/szabo.best.vwh.net/smart.contracts.html, accessed: 2018-08-13.

[37] Bitcoin Wiki, *Script*, 2018. https://en.bitcoin.it/wiki/Script, accessed: 2018-08-15.

[38] P. L. Seijas, S. Thompson and D. McAdams, *Scripting Smart Contracts for distributed ledger technology*, 2016. https://eprint.iacr.org/2016/1156.pdf.

[39] Etherscan, *The Ethereum Block Explorer.* https://etherscan.io/.

[40] Quorum, *Whitepaper*, 2016. https://github.com/jpmorganchase/quorum-docs/raw/master/Quorum%20Whitepaper%20v0.1.pdf, accessed: 2018-08-08.

[41] Codius, 2018. https://codius.org/, accessed: 2018-07-30.

[42] S. Thomas and E. Schwartz, *A Protocol for Interledger Payments*, 2015. https://interledger.org/interledger.pdf, accessed: 2019-02-08.

[43] Nxt, *The Nxt API*, 2018. https://nxtwiki.org/wiki/The_Nxt_API, accessed: 2018-08-03.

[44] Oraclize, *A Scalable Architecture for On-Demand, Untrusted Delivery of Entropy.* http://www.oraclize.it/papers/random_datasource-rev1.pdf, accessed: 2019-02-08.

[45] J. Peterson, J. Krug, M. Zoltu, A. K. Williams and S. Alexander, *Augur: a Decentralized Oracle and Prediction Market Platform*, 2018. http://www.augur.net/whitepaper.pdf, accessed: 2018-04-24.

[46] realitio, *Realitio – Crowd-sourced verification for smart contracts.* https://realit.io/, accessed: 2019-02-12.

[47] C. Pierrot and B. Wesolowski, *Malleability of the blockchain's entropy*, 2016. https://hal.archives-ouvertes.fr/hal-01364045/file/paper.pdf.

[48] A. Reutov, *Predicting Random Numbers in Ethereum Smart Contracts*, 2018. https://blog.positive.com/predicting-random-numbers-in-ethereum-smart-contracts-e5358c6b8620, accessed: 2018-05-22.

[49] J. Lung, *Ethereum network-level transaction spamming attack against honest participants*, 2018. https://github.com/randao/randao/issues/18, accessed: 2018-05-17.

[50] BSI, *Technical Guideline TR-02102 Cryptographic Mechanisms: Recommenda-tions and Key Lengths*, 2021. https://www.bsi.bund.de/EN/Service-Navi/Publications/TechnicalGuidelines/tr02102/tr02102_node.html.

[51] European Telecommunications Stand-ards Institute (ETSI), *TS 119 312 Electronic Signatures and Infrastructures (ESI); Cryptographic Suites*, 2017. https://www.etsi.org/deliver/etsi_ts/119300_119399/119312/01.02.01_60/ts_119312v010201p.pdf.

[52] *Regulation (EU) No 910/2014 of the European Parliament and of the Council of 23 July 2014 on electronic identification and trust services for electronic transactions in the internal market and repealing Directive 1999/93/EC*, 2014. http://eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX:32014R0910.

[53] Hyperledger Fabric, *Channels*, 2018. https://hyperledger-fabric.readthedocs.io/en/release-1.3/channels.html, accessed: 2019-01-25.

[54] R. G. Brown, *The Corda Platform: An Introduction*, 2018. https://www.corda.net/content/corda-platform-whitepaper.pdf, accessed: 2019-01-25.

[55] R. Cheng, F. Zhang, J. Kos, W. He, N. Hynes, N. M. Johnson, A. Juels, A. Miller and D. Song, *Ekiden: A Platform for Confidentiality-Preserving, Trustworthy, and Performant Smart Contract Execution*, 2018. http://arxiv.org/abs/1804.05141.

[56] M. Brandenburger, C. Cachin, R. Kapitza and A. Sorniotti, *Blockchain and Trusted Computing: Problems, Pitfalls, and a Solution for Hyperledger Fabric*, 2018. http://arxiv.org/abs/1805.08541.

[57] G. Zyskind, O. Nathan and A. Pentland, *Enigma: Decentralized Computation Platform with Guaranteed Privacy*, 2015. http://arxiv.org/abs/1506.03471.

[58] BSI, *AIS 46 Informationen zur Evaluierung von kryptographischen Algorithmen und ergänzende Hinweise für die Evaluierung von Zufallszahlengeneratoren*, 2013. https://www.bsi.bund.de/SharedDocs/Downloads/DE/BSI/Zertifizierung/Interpretationen/AIS_46_pdf.pdf.

[59] BSI, *AIS 20 Funktionalitätsklassen und Evaluationsmethodologie für deterministische Zufallszahlengeneratoren*, 2013. https://www.bsi.bund.de/SharedDocs/Downloads/DE/BSI/Zertifizierung/Interpretationen/AIS_20_pdf.pdf.

[60] BSI, *AIS 31 Funktionalitätsklassen und Evaluationsmethodologie für physikalische Zufallszahlengeneratoren*, 2013. https://www.bsi.bund.de/SharedDocs/Downloads/DE/BSI/Zertifizierung/Interpretationen/AIS_31_pdf.pdf.

[61] S. D. Galbraith, *Mathematics of Public Key Cryptography*, New York, NY, USA: Cambridge University Press, 2012.

[62] A. Biryukov, D. Khovratovich and I. Pustogarov, *Deanonymisation of Clients in Bitcoin P2P Network*, in *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*, Scottsdale, AZ, USA, November 3–7, 2014, 2014, pp. 15–29.

[63] H. Al Jawaheri, M. Al Sabah, Y. Boshmaf and A. Erbad, *When A Small Leak Sinks A Great Ship: Deanonymizing Tor Hidden Service Users Through Bitcoin Transactions Analysis*, 2018. http://arxiv.org/abs/1801.07501.

[64] S. Goldfeder, H. A. Kalodner, D. Reisman and A. Narayanan, *When the cookie meets the blockchain: Privacy risks of web payments via cryptocurrencies*, 2017. http://arxiv.org/abs/1708.04748.

[65] S. Meiklejohn, M. Pomarole, G. Jordan, K. Levchenko, D. McCoy, G. M. Voelker and S. Savage, *A fistful of Bitcoins: characterizing payments among men with no names*, Commun. ACM, 59, no. 4, pp. 86–93, 2016.

[66] D. Ron and A. Shamir, *Quantitative Analysis of the Full Bitcoin Transaction Graph*, in *Financial Cryptography and Data Security – 17th International Conference, FC 2013*, Okinawa, Japan, April 1–5, 2013, Revised Selected Papers, 2013, pp. 6–24.

[67] C. Kinkeldey, J.-D. Fekete and P. Isenberg, *BitConduite: Visualizing and Analyzing Activity on the Bitcoin Network*, in *Eurographics Conference on Visualization, EuroVis 2017*, Posters, Barcelona, Spain, 12–16 June 2017, 2017, pp. 25–27.

[68] Chainalysis, *Chainalysis – Blockchain analysis*, 2019. https://www.chainalysis.com/, accessed: 2019-01-25.

[69] R. Klusman and T. Dijkhuizen, *Deanonymisation in Ethereum Using Existing Methods for Bitcoin*, 2018. https://pdfs.semanticscholar.org/bb20/de5fca19392e92c945685de190337bc1e0bf.pdf, accessed: 2019-01-25.

[70] A. E. Gencer, S. Basu, I. Eyal, R. van Renesse and E. G. Sirer, *Decentralization in Bitcoin and Ethereum Networks*, 2018. http://arxiv.org/abs/1801.03998.

[71] E. Duffield and D. *Diaz, Dash: A Payments-Focused Cryptocurrency*, 2018. https://github.com/dashpay/dash/wiki/Whitepaper, accessed: 2019-01-25.

[72] Monero Research Lab, *Monero – secure, private, untraceable.* https://ww.getmonero.org/resources/research-lab/, accessed: 2019-01-25.

[73] E. Ben-Sasson, A. Chiesa, E. Tromer and M. Virza, *Succinct Non-Interactive Zero Knowledge for a von Neumann Architecture*, in *Proceedings of the 23rd USENIX Security Symposium*, San Diego, CA, USA, August 20–22, 2014, 2014, pp. 781–796.

[74] E. Ben-Sasson, I. Bentov, Y. Horesh and M. Riabzev, *Scalable, transparent, and post-quantum secure computational integrity*, 2018. http://eprint.iacr.org/2018/046.

[75] Zcash, *How it works*, 2018. https://z.cash/technology/, accessed: 2019-01-25.

[76] G. Kappos, H. Yousaf, M. Maller and S. Meiklejohn, *An Empirical Analysis of Anonymity in Zcash*, in *27th USENIX Security Symposium, USENIX Security 2018*, Baltimore, MD, USA, August 15–17, 2018, 2018, pp. 463–477.

[77] J. Quesnelle, *On the linkability of Zcash transactions*, 2017. http://arxiv.org/abs/1712.01210.

[78] M. Möser, K. Soska, E. Heilman, K. Lee, H. Heffan, S. Srivastava, K. Hogan, J. Hennessey, A. Miller, A. Narayanan and N. Christin, *An Empirical Analysis of Traceability in the Monero Blockchain*, PoPETs, 2018, no. 3, pp. 143–163, 2018.

[79] N. Houy, *The Bitcoin Mining Game*, Ledger, 1, pp. 53–68, 2016. https://ledgerjournal.org/ojs/index.php/ledger/article/view/13.

[80] J. A. Garay, J. Katz, U. Maurer, B. Tackmann and V. Zikas, *Rational Protocol Design: Cryptography against Incentive-Driven Adversaries*, in *54th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2013*, Berkeley, CA, USA, 26–29 October, 2013, 2013, pp. 648–657.

[81] F. K. Wilhelm, R. Steinwandt, B. Langenberg, P. J. Liebermann, A. Messinger and P. K. Schuhmacher, *Entwicklungsstand Quantencomputer*, 2017. www.bsi.bund.de/qcstudie.

[82] C. Berghoff, U. Korte, T. Kusber and S. Schwalm, *Langfristige Beweiswerterhaltung und Datenschutz in der Blockchain*, in *Dach Security 2018*, Gelsenkirchen, 2018.

[83] J. Wilmoth, *'Nuclear Option': ABC Dev Won't Rule out Changing Bitcoin Cash PoW Algorithm*, 2018. https://www.ccn.com/nuclear-option-abc-dev-wont-rule-out-changing-bitcoin-cash-pow-algorithm/, accessed: 2019-01-30.

[84] P. W. Shor, *Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer*, SIAM J. Comput., 26, no. 5, pp. 1484–1509, 1997.

[85] J. J. Roberts and N. Rapp, *Exclusive: Nearly 4 Million Bitcoins Lost Forever, New Study Says*, 2017. fortune.com/2017/11/25/lost-bitcoins/, accessed: 2019-01-30.

[86] *PoW 51% Attack Cost*, 2018. https://www.crypto51.app/, accessed: 2018-09-28.

[87] Trustnodes, *Bitcoin Gold 51% Attacked, $18 Million Stolen Through Double Spends*, 2018. https://www.trustnodes.com/2018/05/24/bitcoin-gold-51-attacked-18-million-stolen-double-spends.

[88] GoBitcoin.io, *Cost of a 51% attack*, 2019. https://gobitcoin.io/tools/cost-51-attack/, accessed: 2019-01-10.

[89] J. Kroll, I. Davey and E. Felten, *The Economics of Bitcoin Mining, or Bitcoin in the Presence of Adversaries*, Workshop on the Economics of Information Security, 2013.

[90] C. Decker and R. Wattenhofer, *Information Propagation in the Bitcoin Network*, 2013 IEEE Thirteenth International Conference on Peer-to-Peer Computing (P2P), 2013.

[91] G. Bissas, B. N. Levine, A. P. Ozisik, G. Andresen and A. Houmansadr, *An Analysis of Attacks on Blockchain Consensus*, 2016. http://arxiv.org/abs/1610.07985.

[92] I. Eyal and E. Sirer, *Majority Is Not Enough: Bitcoin Mining Is Vulnerable*, Lecture Notes in Computer Science, 8437, pp. 436–454, 2014.

[93] M. Vasek, M. Thornton and T. Moore, *Empirical Analysis of Denial-of-Service Attacks in the Bitcoin Ecosystem*, Lecture Notes in Computer Science, 8438, pp. 57–71, 2014.

[94] E. Heilman, A. Kendler, A. Zohar and S. Goldberg, *Eclipse Attacks on Bitcoin's Peer-to-Peer Network*, in *24th USENIX Security Symposium, USENIX Security 15*, Washington, D.C., USA, August 12–14, 2015, 2015, pp. 129–144.

[95] Bitcoin Wiki, *Common Vulnerabilities and Exposures*, 2019. https://en.bitcoin.it/w/index.php?title=Common_Vulnerabilities_and_Exposures&oldid=66007, accessed: 2019-01-10.

[96] Coinwire, *Australian Firm Loses $6.6 Million from Soarcoin „Backdoor“*, 2018. https://www.coinwire.com/australian-firm-losses-6-6-million-from-soarcoin-backdoor.

[97] J. W. Bos, J. A. Halderman, N. Heninger, J. Moore, M. Naehrig and E. Wustrow, *Elliptic Curve Cryptography in Practice*, Financial Cryptography and Data Security — 18th International Conference, pp. 157–175, 2014.

[98] J. Breitner and N. Heninger, *Biased Nonce Sense: Lattice Attacks against Weak ECDSA Signatures in Cryptocurrencies*, 2019. https://eprint.iacr.org/2019/023.pdf.

[99] A. Roos, *Verge fällt Hacker zum Opfer – schon wieder*, 2018. https://www.btc-echo.de/verge-xvg-faellt-hacker-zum-opfer-schon-wieder/.

[100] C. Decker and R. Wattenhofer, *Bitcoin Transaction Malleability and MtGox*, in *Computer Security – ESORICS 2014*, Proceedings, Part II, 19th European Symposium on Research in Computer Security, Wroclaw, Poland, September 7–11, 2014, 2014, pp. 313–326.

[101] WhaleCalls, *A tl;dr on Antbleed*, 2017. https://medium.com/@whalecalls/a-tl-dr-antbleed-2406a5e34fcd, accessed: 2019-01-25.

[102] Bitmain, *Antminer Firmware Update*, April 2017. https://blog.bitmain.com/en/antminer-firmware-update-april-2017/, accessed: 2019-01-25.

[103] L. Luu, D.-H. Chu, H. Olickel, P. Saxena and A. Hobor, *Making Smart Contracts Smarter*, 2016. https://eprint.iacr.org/2016/633.

[104] Trustnodes, *Smart Contract Bug Nearly Freezes Transfers in $800 Million Worth of Icon Tokens*, 2018. https://www.trustnodes.com/2018/06/17/smart-contract-bug-nearly-freezes-transfers-800-million-worth-icon-tokens, accessed: 2018-08-22.

[105] N. Atzei, M. Bartoletti and T. Cimoli, *A survey of attacks on Ethereum smart contracts*, in *Principles of Security and Trust*, LNCS 10204, M. Maffei and M. Ryan, Eds., Springer, 2017, pp. 164–186.

[106] S. Kalra, S. Goel, M. Dhawan and S. Sharma, *Zeus: Analyzing Safety of Smart Contracts*, Network and Distributed System Security Symposium, 2018. http://wp.internetsociety.org/ndss/wp-content/uploads/sites/25/2018/02/ndss2018_09-1_Kalra_paper.pdf.

[107] M. Bartoletti, S. Carta, T. Cimoli and R. Saia, *Dissecting Ponzi schemes on Ethereum: identification, analysis, and impact*, 2017. http://arxiv.org/abs/1703.03779.

[108] M. Araoz, *Serpent Compiler Audit v1.0.0*, 2017. https://github.com/AugurProject/augur-audits/blob/master/serpent-compiler/Zeppelin Solutions – Serpent Compiler Audit v1.0.0.pdf, accessed: 2018-08-15.

[109] G. Destefanis, A. Bracciali, M. Marchesi, M. Ortu, R. Tonelli and R. Hierons, *Smart Contracts Vulnerabilities: A Call for Blockchain Software Engineering?*, 2018. https://www.researchgate.net/publication/323545752_Smart_Contracts_Vulnerabilities_A_Call_for_Blockchain_Software_Engineering.

[110] I. Nikolic, A. Kolluri, I. Sergey, P. Saxena and A. Hobor, *Finding The Greedy, Prodigal, and Suicidal Contracts at Scale*, 2018. http://arxiv.org/abs/1802.06038.

[111] K. Bhargavan, A. Delignat-Lavaud, C. Fournet, A. Gollamudi, G. Gonthier, N. Kobeissi, N. Kulatova, A. Rastogi, T. Sibut-Pinote, N. Swamy and S. Zanella-Béguelin, *Formal Verification of Smart Contracts: Short Paper*, in *Proceedings of the 2016 ACM Workshop on Programming Languages and Analysis for Security*, ACM, 2016, pp. 91–96.

[112] L. Luu, J. Teutsch, R. Kulkarni and P. Saxena, *Demystifying Incentives in the Consensus Computer*, Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security, pp. 706–719, 2015.

[113] S. Jain, P. Saxena, F. Stephan and J. Teutsch, *How to verify computation with a rational network*, 2016. http://arxiv.org/abs/1606.05917.

[114] K. Delmolino, M. Arnett, A. Kosba, A. Miller and E. Shi, *Step by Step Towards Creating a Safe Smart Contract: Lessons and Insights from a Cryptocurrency Lab*, in *Financial Cryptography and Data Security*, J. Clark, S. Meiklejohn, P. Y. Ryan, D. Wallach, M. Brenner and K. Rohloff, Eds., Springer, 2016, pp. 79–94.

[115] C. McFarland, T. Hux, E. Wuehler and S. Campbel, *Blockchain Threat Report*, 2018. https://www.mcafee.com/enterprise/en-us/assets/reports/rp-blockchain-security-risks.pdf.

[116] C. Cimpanu, *IOTA Cryptocurrency Users Lose $4 Million in Clever Phishing Attack*, 2018. https://www.bleepingcomputer.com/news/security/iota-cryptocurrency-users-lose-4-million-in-clever-phishing-attack/.

[117] Kyodo News, *Police start investigating Coincheck cryptocurrency theft*, 2018. https://english.kyodonews.net/news/2018/01/021a8a3c5844-update2-regulators-take-action-against-coincheck-after-cryptocurrency-theft.html.

[118] W. Suberg, *Vietnam: Betrügerische ICOs von Pincoin und Ifan stehlen rund 536 Mio*, 2018. https://de.cointelegraph.com/news/vietnam-pincoin-ifan-icos-exposed-as-scams-that-allegedly-stole-660-million.

[119] BSI, *IT-Grundschutz-Kompendium – Edition 2019*, 2019. https://www.bsi.bund.de/DE/Themen/Unternehmen-und-Organisationen/Standards-und-Zertifizierung/IT-Grundschutz/IT-Grundschutz-Kompendium/it-grundschutz-kompendium_node.html.

[120] VDI Technologiezentrum GmbH, *Blockchain – Eine Technologie mit disruptivem Charakter, Kap. 7: Rechtliche Rahmenbedingungen der Blockchain, Version 1.0*, 2018. https://www.vditz.de/fileadmin/media/news/documents/Blockchain_-_Eine_Technologie_mit_disruptivem_Charakter.pdf, accessed: 2018-11-20.

[121] J. Schrey and T. Thalhofer, *Rechtliche Aspekte der Blockchain*, Neue juristische Wochenschrift, p. 1431, 2017.

[122] D. Saive, *Rückabwicklung von Blockchain-Transaktionen*, in *Tagungsband Herbstakademie 2018, Rechtsfragen digitaler Transformationen – Gestaltung digitaler Veränderungsprozesse durch Recht*, OLWIR Verlag, 2018, pp. 371–380.

[123] P. J. Pesch, *Blockchain, Smart Contracts und Datenschutz*, in *Smart Contracts*, M. Fries and B. P. Paal, Eds., Mohr Siebeck, 2019, pp. 13–23.

[124] R. Matzutt, J. Hiller, M. Henze, J. H. Ziegeldorf, D. Müllmann, O. Hohlfeld and K. Wehrle, *A Quantitative Analysis of the Impact of Arbitrary Blockchain Content on Bitcoin*, in *Proceedings of the 22nd International Conference on Financial Cryptography and Data Security (FC), Santa Barbara*, Springer, 2018.

[125] A. Peters, *Sabotage von Blockchains durch Einschleusung strafrechtsrelevanter Inhalte?*, in *Tagungsband Herbstakademie 2018, Rechtsfragen digitaler Transformationen – Gestaltung digitaler Veränderungsprozesse durch Recht*, OLWIR Verlag, 2018, pp. 381–396.

[126] S. H. Kimpel and C. Adcock, *The State of Smart Contract Legislation*, 2018. https://www.blockchainlegalresource.com/2018/09/state-smart-contract-legislation/, accessed: 2018-12-05.

[127] *Regulation (EU) 2016/679 of the European Parliament and of the Council on the protection of natural persons with regard to the processing of personal data and on the free movement of such data, and repealing Directive 95/46/EC (General Data Protection Regulation)*, 2016. https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=celex:32016R0679.

[128] M. Rost, *Standardisierte Datenschutzmodellierung*, Datenschutz und Datensicherheit, 6, pp. 433–438, 2012.

[129] Blockcerts, *Introduction – Blockcerts: The Open Standard for Blockchain Credentials.* https://www.blockcerts.org/guide/, accessed: 2019-02-01.

[130] H. Krawczyk and T. Rabin, *Chameleon Hashing and Signatures*, 1998. http://eprint.iacr.org/1998/010.

[131] G. Ateniese, B. Magri, D. Venturi and E. R. Andrade, *Redactable Blockchain – or – Rewriting History in Bitcoin and Friends*, in *2017 IEEE European Symposium on Security and Privacy, Euro S&P 2017*, Paris, France, April 26–28, 2017, 2017, pp. 111–126.

[132] I. Puddu, A. Dmitrienko and S. Capkun, *µchain: How to Forget without Hard Forks*, 2017. http://eprint.iacr.org/2017/106.

[133] G. Goldacker, *Digitale Souveränität*, 2017. https://www.oeffentliche-it.de/documents/10181/14412/Digitale+Souver%C3%A4nit%C3%A4t.

[134] BMWi, *Kompetenzen für eine digitale Souveränität*, 2017. https://www.bmwi.de/Redaktion/DE/Publikationen/Studien/kompetenzen-fuer-eine-digitale-souveraenitaet.pdf.

[135] Plattform „Innovative Digitalisierung der Wirtschaft“, *Digitale Souveränität und Künstliche Intelligenz – Voraussetzungen, Verantwortlichkeiten und Handlungsempfehlungen*, 2018. https://www.de.digital/DIGITAL/Redaktion/DE/Textsammlung/digital-gipfel-plattform-digitalisierung-der-wirtschaft-fg1.html.

[136] D. Ron and A. Shamir, *How Did Dread Pirate Roberts Acquire and Protect his Bitcoin Wealth?*, in *Financial Cryptography and Data Security – FC 2014 Workshops, BITCOIN and WAHC 2014*, Christ Church, Barbados, March 7, 2014, Revised Selected Papers, 2014, pp. 3–15.

[137] BSI, *Technical Guideline TR-03125 TR-ESOR Preservation of Evidence of Cryptographically Signed Documents*, 2019 (2021 for updated German version). https://www.bsi.bund.de/EN/Service-Navi/Publications/TechnicalGuidelines/TR03125/BSITR03125.html.

[138] IT-Finanzmagazin, *Blockchain-Anwendung: Frankfurt School – fälschungssichere Abschlusszeugnisse für Studierende*, 2018. https://www.it-finanzmagazin.de/blockchain-frankfurt-school-zeugnis-77708/, accessed: 2019-01-30.

[139] regio IT, *Showcase „Zeugnisvalidierung über Blockchains“*, 2017. https://www.uni-speyer.de/files/de/Lehrst%C3%BChle/Hill/Neue_Veranstaltungen/Showcase_Niehues.pdf, accessed: 2019-01-30.

[140] BSI, *Technische Richtlinie TR-03147 Vertrauensniveaubewertung von Verfahren zur Identitätsprüfung natürlicher Perso-nen*, 2017. https://www.bsi.bund.de/SharedDocs/Downloads/DE/BSI/Publikationen/TechnischeRichtlinien/TR03147/TR03147.html.

[141] F. Kirstein, M. Polzhofer and K.-P. Eckert, *Digitale Identitäten in der Blockchain – Erfahrungen aus der Entwicklung*, 2018.

[142] Nano, *Nano – an instant, zero-fee, scalable currency*, 2019. https://nano.org/en, accessed: 2019-01-25.

[143] Dexon, *DEXON – Empowering the Decentralised Future*, 2019. https://dexon.org/, accessed: 2019-01-25.

[144] Bundesdruckerei, *Blockchain – Verbesserungspotential der Technologie*, 2018. https://www.bafin.de/SharedDocs/Downloads/DE/Veranstaltung/dl_180410_BaFinTech_2018_WS3_2.pdf?__blob=publicationFile&v=3, accessed: 2019-01-25.

[145] IOTA, *The Next Generation of Distributed Ledger Technology*, 2018. https://www.iota.org/, accessed: 2019-01-25.

[146] S. Popov, *The Tangle*, 2018. https://assets.ctfassets.net/r1dr6vzfxhev/2t4uxvsIqk0EUau6g2sw0g/45eae33637ca92f85dd9f4a3a218e1ec/iota1_4_3.pdf, accessed: 2019-01-25.

[147] Cosmos Network, *Internet of Blockchains.* https://cosmos.network/, accessed: 2019-01-25.

[148] Polkadot, *Polkadot*, 2019. https://polkadot.network/, accessed: 2019-01-25.

[149] Quant Network, *Overledger.* https://www.quant.network/our-technology/overledger/, accessed: 2019-01-25.

[150] J. Poon and T. Dryja, *The Bitcoin Lightning Network: Scalable Off-Chain Instant Payments*, 2016. https://lightning.network/lightning-network-paper.pdf, accessed: 2019-01-25.

[151] E. O. Kiktenko, N. O. Pozhar, M. N. Anufriev, A. S. Trushechkin, R. R. Yunusov, Y. V. Kurochkin, A. I. Lvovsky and A. K. Fedorov, *Quantum-secured blockchain*, 2017. http://arxiv.org/abs/1705.09258.

[152] D. Rajan and M. Visser, *Quantum Blockchain using entanglement in time*, 2018. http://arxiv.org/abs/1804.05979.

[153] K. Ikeda, *qBitcoin*, 2017. http://arxiv.org/abs/1708.04955.

[154] J.-S. Weng, J. Weng, M. Li, Y. Zhang and W. Luo, *DeepChain: Auditable and Privacy-Preserving Deep Learning with Blockchain-based Incentive*, 2018. https://eprint.iacr.org/2018/679.

[155] G. Popov, *SKYCHAIN – The future of artificial intelligence in healthcare!*. https://skychain.global/upload/iblock/89a/wp_english_Newest.pdf, accessed: 2019-01-25.

[156] International Organization for Standardization, *ISO/TC 307 Blockchain and distributed ledger technologies.* https://www.iso.org/committee/6266604.html, accessed: 2019-02-01.

[157] European Telecommunications Standards Institute (ETSI), *Terms of Reference (ToR) for ETSI ISG ‚Permissioned Distributed Ledger' (ISG PDL)*, 2018. https://portal.etsi.org/Portals/0/TBpages/PDL/Docs/ISG_PDL_ToR_DG_Approved_20181130.pdf, accessed: 2019-02-01.

[158] Institute of Electrical and Electronics Engineers, *Standards – IEEE Blockchain Initiative*, 2019. https://blockchain.ieee.org/standards, accessed: 2019-02-01.

[159] World Wide Web Consortium, *Blockchain – W3C Blog*, 2019. https://www.w3.org/blog/category/technology/blockchain/, accessed: 2019-02-01.

[160] Internet Research Task Force, *blockchain federation – IRTF wiki*, 2018. https://trac.ietf.org/trac/irtf/wiki/blockchain-federation, accessed: 2019-02-01.

[161] Europäische Kommission, *European countries join Blockchain Partnership*, 2018. https://ec.europa.eu/digital-single-market/en/news/european-countries-join-blockchain-partnership, accessed: 2019-02-01.

[162] O. Fußwinkel and C. Kreiterling, *Blockchain-Technologie – Gedanken zur Regulierung*, 2018. https://www.bafin.de/SharedDocs/Veroeffentlichungen/DE/BaFinPerspektiven/2018/bp_18-1_Beitrag_Fusswinkel.html, accessed: 2019-02-01.

[163] Bitcoinity.org, *Bitcoin network hashrate*, 2019. https://data.bitcoinity.org/bitcoin/hashrate/6m?c=m&g=15&t=a, accessed: 2019-01-10.

Legal notice

Published by:
Federal Office for Information Security (BSI)
53175 Bonn, Germany

Source:
Federal Office for Information Security (BSI)
Godesberger Allee 185–189
53175 Bonn, Germany
telephone: +49 (0) 228 999582-0
email: blockchain@bsi.bund.de
internet: www.bsi.bund.de

Last updated:
March 2019

First published:
May 2019 in German

Content, editing and concept:
Dr. Christian Berghoff, Dr. Ute Gebhardt, Dr. Manfred Lochter, Dr. Sarah Maßberg,
Federal Office for Information Security (BSI)
with contributions by Julia Braam, Sandra Häberer, BSI
and by Thomas Häberlen, Rainer Oberweis, Dr. Tobias Stadler,
The Federal Commissioner for Data Protection and Freedom of Information (BfDI)

Design and print:
Appel & Klinger Druck und Medien GmbH, Schneckenlohe

Image credits:
Title: Getty Images, AF-studio
Figures 1–13: BSI

www.bsi.bund.de