Blockchains and Distributed Ledgers Uncovered: Clarifications, Achievements, and Open Issues

Eder J. Scheid, Bruno B. Rodrigues, Christian Killer, Muriel F. Franco, Sina Rafati, Burkhard Stiller

Blockchains and Distributed Ledgers Uncovered: Clarifications, Achievements, and Open Issues

Eder J. Scheid*[0000−0002−7989−5286], Bruno B. Rodrigues[0000−0003−3277−3799], Christian Killer[0000−0001−7943−1185], Muriel F. Franco[0000−0002−0208−0521], Sina Rafati[0000−0001−6265−4657], and Burkhard Stiller[0000−0002−7461−7463]

Communication Systems Group CSG, Department of Informatics IfI, University of Zürich UZH, Binzmühlestrasse 14, CH—8050 Zürich, Switzerland
{scheid, rodrigues, killer, franco, rafati, stiller}@ifi.uzh.ch
* Corresponding Author

Abstract. Although the first Blockchain (BC) was proposed about a decade ago, BC achievements from a technical and functional perspective are measurable, but still face open issues. Since IFIP's Working Group 6.6 on the "Management of Networks and Distributed System" investigated BCs in various aspects, this contribution here summarizes and clarifies key characteristics of BCs and their related approach of Distributed Ledgers (DL).
While many properties are under discussion, the two approaches differ measurably. In turn, the value of BCs and DLs is outlined in combination with selected and exemplified application domains. However, a set of open issues has been observed, which possibly hinders a practical operation, *e.g.,* due to excessive expectations, missing interoperability, wrong scalability promises, or out-of-scope trust assumptions. Thus, the state-of-the-art in BCs and DLs is clarified and current as well necessary research steps to follow complement this state.

Keywords: Blockchains · Distributed Ledger · Scalability · Trust

1 Introduction

The International Federation for Information Processing (IFIP) Technical Committee (TC) 6, being focused on the "Communications Systems" broader area, (*i*) promotes the exchange of knowledge in this field, (*ii*) connects telecommunication users, manufacturers, and providers, and (*iii*) aids standardization bodies in their efforts. This work is achieved via nine Working Groups (WG) as well as Special Interest Groups (SIG), the majority of which are concerned either with specific aspects of communications systems themselves or with the application of communications systems. With its decentralized nature, IFIP TC6 is the supporter and all WGs operate autonomously, while tackling diverse aspects of telecommunication, ranging from architectures and protocols for distributed systems (WG6.1) via network and inter-network architectures (WG6.2), the performance of communication systems (WG6.3), the management of networks

and distributed systems (WG6.6), smart networks (WG6.7), mobile and wireless communications (WG6.8), communications systems for developing countries (WG6.9), photonic networking (WG6.10), communication aspects of the eWorld (WG6.11) to services-oriented systems (WG6.12). In this setting, WG6.6 had selected from 2014 onward the Blockchain (BC) topic as one, which is by itself determines an approach to be managed, resulting in certain security-related outcomes, and which manages data handling in a distributed setting, too, well beyond known approaches' characteristics. Thus, this chapter here discusses key characteristics of BCs and their related approach of a Distributed Ledger (DL), since the two approaches differ measurably.

Since BCs and DLs do provide due to their characteristics suitable means for the management of networks and distributed systems, too, respective umbrella work is performed as well in the context of IFIP by IFIP's TC6 on "Communication Systems", especially within the Working Group (WG) 6.6 on "Management of Networks and Distributed Systems". Thus, since 2017 the IFIP/IEEE Network Operations and Management Symposium (NOMS) [63], [58], the IFIP/IEEE International Symposium on Integrated Network Management (IM) [8], [64], [57], and the International Conference on Network and Service Management (CNSM) [45] included a set of BC- and DL-related keynotes, reviewed papers, and tutorials, which had become part of their respective scientific programs.

1.1 Blockchain History

A bit longer than a decade ago, the Computer Science community was presented with Bitcoin [48], which was, at that time, yet another proposal for electronic cash. However, unlike previous proposals [17, 3, 65], Bitcoin successfully solved the double-spending problem (*i.e.,* same coin spent twice) in practice and a fully decentralized manner, without the need for a Trusted Third Party (TTP), by combining Peer-to-Peer (P2P) mechanisms, applied cryptography, and time-stamping principles to form an immutable, decentralized, and publicly verifiable ledger of transactions, called *Blockchain* (BC).

Since then, BCs were applied in different areas and use cases [13] due to pseudonymity, low transaction fees, open access, and immutability. Currently, BCs in academia's and public's perception can be observed by more than 9,100 cryptocurrencies listed [20] and over 13,200 BC-related projects registered with versioning control providers [30]. Although time-stamping services existed in the past [33], too, their full decentralization became only possible with an underlying BC, which by itself benefited from earlier developments of a block-based Byzantine storage, applying cryptographic hashes on user data to be stored [46].

The extensive employment of the BC in a wide range of application domains and the exponential growth in popularity has led to the development and research of novel BC platforms [5], with many approaches tackling specific problems of earlier approaches and claiming to solve them. However, not all BC platforms follow the same core principles as the first BC proposal (*i.e.,* Bitcoin) did, as their deployment models, consensus mechanisms, and underlying data structures differ drastically. For example, Ripple [16] restricts the privilege to

write blocks to a set of defined nodes, EOS [24] employs a consensus mechanism, where miners are elected and selected based on the number of coins they hold (*i.e.,* their stake), and IOTA [38] implements a Directed Acyclic Graph (DAG) to store transactions. These and many other platforms cannot be considered BCs *per se* but should be defined as Distributed Ledgers (DL).

Both BCs and DLs are evolving currently at a fast pace due to academic research, industrial incentives, and intense funding of projects based on such approaches [35]. However, the differences between BCs and DLs are not easy to grasp and are still fuzzy for users. Although seemingly similar at first glance, BCs and DLs inherently features various advantages and disadvantages, which have to be balanced with the needs of the application area in question. Thus, it is essential to distinguish these two approaches and clarify key differences to users who plan to develop, apply, or invest in projects or companies at the doorstep to employ BCs or DLs in their current operation, *e.g.,* supply-chain [8].

Therefore, this paper contributes besides past work of [9] especially to the clarification of key characteristics of BCs and DLs, while presenting measurable advantages and drawbacks of the two approaches in combination with selected platforms and exemplified application domains. Based on this discernment, the paper (*a*) details those open issues concerning the applicability of BCs and DLs in real life, and (*b*) outlines further research directions to increase their maturity, contributing to the state-of-the-art in this area.

1.2 Paper Structure

The remainder of this paper is organized as follows. First, Section 2 explains the necessary background and definitions, and summarizes selected platforms. Second, Section 3 clarifies the differences between BCs and DLs, which is followed by Section 4 summarizing the value of BCs and DLs. While Section 5 discusses open issues, finally, Section 6 draws conclusions.

2 Laying the Groundwork — Technical Background

Several concepts, paradigms, and mechanisms build the foundations for BC and DL. Therefore, a concise understanding of major building blocks is required to understand the intricacies of BC and DL, which supports the remainder of this paper. Thus, their description and a set of selected BC implementations and their differences exemplify this understanding.

2.1 Public-Key Cryptography

Public-Key Cryptography (PKC) first emerged to solve the main drawback of symmetric cryptography: the exchange of secret keys [27, 66]. Asymmetric cryptography mitigates the key distribution issues by using a public and a private key. These keys are associated in a mathematical way. Knowledge about the

public key does not allow one to compute anything about the private key. However, knowing the private key allows unlocking the information encrypted with the public key [66]. In general, PKC relies on the hardness of reversing certain mathematical operations, often referred to as *trapdoors*.

For instance, the resilience of PKC systems (*e.g.,* RSA and ElGamal) stems from the hardness of inverting the exponentiation of prime numbers, often referred to as the Discrete Logarithm problem [66]. PKC is also used to build digital signature schemes. The Digital Signature Algorithm or DSA (often referred to as the Digital Signature Standard, DSS [66], too) preserves the integrity of messages exchanged between two parties. Originally, DSA was designed to work in finite fields. Today, it is common to use it with Elliptic Curves (EC), in which case it is referred to as EC-DSA. The EC variants of DSA run very fast and have smaller footprints as well as key sizes than almost all other signature algorithms [66], which is why EC-DSA is favored in the context of BCs.

Two fundamental processes of BCs make use of PKC: (*i*) Address Generation and (*ii*) Transaction Signing. In (*i*), a valid BC address is generated by using an algorithm based on a public key, which in turn is associated with a private key. In (*ii*), BC transactions are cryptographically signed to guarantee that funds can only be spent by their owners *i.e.,* who know the private key. Figure 1 depicts the generic BC address generation process. The process relies on cryptographic one-way hash functions. These functions take an arbitrary length bit string as input and produce a fixed-length bit string as output. In that context, *one-way* means that it should be computationally infeasible to retrieve the matching input with the usage of a hash digest. Thus, cryptographic hash functions are used to shield the public keys, and only hash digests serve as public identifiers, *i.e.,* BC addresses of the electronic wallet and can thus be used to receive funds. Only users in possession of the corresponding private key can generate authentic BC transactions, which control the funds associated with that wallet. Transactions are validated by verifying the digital signatures.

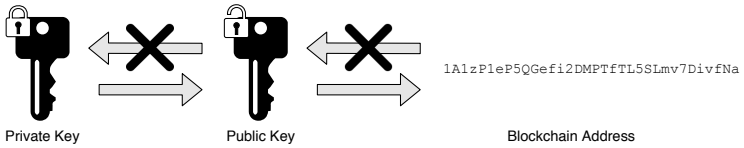

Figure 1. Address Generation based on Public Keys

Typically, to issue a valid transaction, a user must create a transaction containing the source address (*i.e.,* his/her BC address), the destination BC address, the amount to be transferred, and possibly, the miner fee (*cf.* Section 2.2). After this transaction is created, the user must digitally sign it with the corresponding BC address' private key to create a signed, raw transaction, which is broadcast to all computing devices (*i.e.,* peers and their nodes), connected through the Internet, that run applications which implement the BC protocol (*i.e.,* the

BC network). Such a transaction stays in the transaction pool until a miner includes it into a new block. To verify, if the owner of that BC address issued the transaction, anyone could use the public key of that user to confirm that this transaction was not modified by anyone else. Thus, this process provides, in general, the *integrity* and *authenticity* properties of the BC transaction.

2.2 Mining on a Blockchain

The process of validating transactions, adding a new block, solving a crypto puzzle, and minting (*i.e.,* creating) new coins (which are just different terms for the exact same action) in Proof-of-Work (PoW)-based BCs is called "mining" [29]. Mining is performed by dedicated, but not pre-determined nodes in a BC network; these nodes are termed "miners", which store all transactions of that BC, starting from the "genesis block" (*i.e.,* block number 0) to the most recent block. These miners need to maintain this information in order to verify if new transactions are valid or not, which leads to a fully decentralized solution of the double-spending problem (*i.e.,* the same coins spent twice) [48].

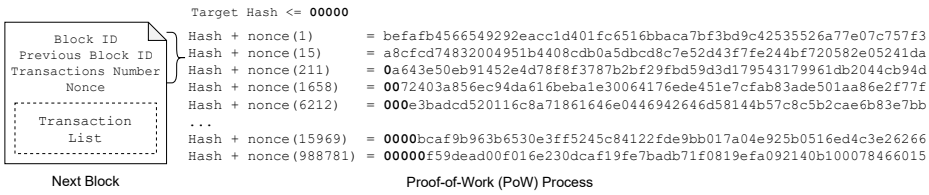

Figure 2. Hash-based Proof-of-Work (PoW)

For example, in the Bitcoin BC [48] a new block is added to the BC every 10 min by the fastest miner that provided a solution satisfying the crypto puzzle, which resembles a specific form of a hash value of that block with a predefined number of preceding zeros, all based on an agreed-upon hash function. In order to determine this result, a miner must spend significant computational power to calculate this hash values (*cf.* Figure 2) based on the content of that block proposed (*e.g.,* transactions, headers, and previous block number), and a random number used once (*i.e., nonce*).

Moreover, miners compete with other miners to find this hash value; thus, the more computational power a miner has, the larger likelihood it has of finding that value. As soon as the miner finds a suitable hash value, it broadcasts the same block, that hash value, and the nonce to the BC network, such that other nodes can verify the correctness of this hash value. This can be performed basically at "no cost", since this verification requires the calculation of this only hash value based on all data known, including the hash function itself. If the hash value is correct, the Bitcoin BC protocol creates an additional transaction containing the reward to that miner, who found this suitable hash value first. Such a reward

fee is paid for each successfully persisted block of BC transactions. Since this is an interactive process, it starts all over again once the block is persisted into the chain by taking newly available BC transactions from the mining pool.

2.3 A Few Handful of Blockchain Platforms

Since 2009, more than 9,000 BC platforms, cryptocurrencies, and tokens were proposed [20]. Not all systems proposed so far differ in all dimensions; they may be based on a few dozens of really distinct BC platforms. Thus, systems with the primary goal to offer cryptocurrencies and tokens (in general, an electronic representation of a digital asset) differ from systems, which exploit cryptocurrencies and tokens in an application-specific manner. Therefore, this paper selected such technically different BC platforms that (partially, due to space constraints) represent major BC and DL characteristics, namely Bitcoin, Ethereum, HyperLedger, IOTA, and Bazo (*cf.* Table 1).

Table 1. Platform Comparison

Platform	Type	Data Structure	Consensus	Tps	Notes
Bitcoin	Public	Blocks	PoW	7	First BC proposal
Ethereum	Public	Blocks	PoW	15	Turing-complete Smart Contracts
HyperLedger	Private	Blocks	Many	Variable	Umbrella project: enterprise focused platforms
IOTA	Public	DAG	Own	1,000	IoT BC
BAZO	Experimental	Blocks	PoS	50	Research and experimental BC

Transactions per Second (Tps), Internet-of-Things (IoT)

Bitcoin was originally proposed under the pseudonym "Satoshi Nakamoto" in 2009. This proposal of an experimental cryptocurrency created a fully public and decentralized one, which achieved over time a significant standing due to the disruptive potential of the technical platform on which it is based on, the BC [48]. The conceptual design based on the PoW consensus mechanism (*cf.* Section 2.2) providing the guarantee of security and immutability continues up to today as "secure" [28], since no BC-related error or flaw had been discovered.

One of the main factors that contribute to the current success of Bitcoin is the incentive scheme for peers participating in the BC network as miners [48]. Henceforth, incentives do not only ensure that nodes generate valid blocks through the PoW consensus mechanism but also ensure that malicious nodes are not encouraged to assemble more CPU power than all honest nodes. Following the success of Bitcoin, several other cryptocurrencies were created based on Bitcoin's source

code [20]. These are typically called *Altcoins*, in which their code includes several modifications in its parameters to tackle different parameters and use cases than Bitcoin (*e.g.,* block size or creation time). For instance, the Bitcoin protocol has a block time that can be considered too long for some financial applications; thus, certain Altcoins (*e.g.,* Litecoin [41] and Monero [40]) reduce this block creation time, as well as difficulty in PoW hashing competition. Monero, for instance, offers additional security features based on a ring-signature scheme in which transactions can be fully obscured.

Despite the experimental cryptocurrency's success, a performance drawback ensures the security of the PoW model and consensus mechanism [28]. Notably, the resulting large energy consumption by design being generated as an effect of these partial hash collisions, in which several nodes miners participate in a competition to find a suitable hash value above a certain target with X leading zeros, determines only one node as the winner [42]. Additionally, an "unwanted" side effect of these computational demands matures in lower scalability and throughput of the chains in terms of Tps achieved, *e.g.,* 7 Tps in the case of Bitcoin [21]. Thus, if many transactions are submitted at the same time, especially exceeding the storage capacity of a single block, some users' transactions will have to wait until their transactions are appended into a block by a miner, consequently increasing the delay for a confirmation.

Ethereum originated with the intent of creating a public BC platform, where general distributed applications can be built on. The main drivers for such a proposal and the development of Ethereum were the limitations of Bitcoin's scripting language, which does not provide for Turing-completeness or even states. These limitations were solved in Ethereum by relying on (*a*) an account-based BC, which is different from a UTXO model since it modifies the state of an account, and (*b*) an underlying built-in Smart Contract (SC) language (*cf.* Section 4.2) providing the capability to developers to write Turing-complete programs, which can operate as SCs [10, 75].

Blocks in Ethereum are smaller and are created faster (on average every 14 s) compared to *e.g.,* Bitcoin's block creation time being on average at 10 min. Thus, the overall Ethereum BC size grows faster, too. To alleviate this increase in size, Ethereum implemented apart from full nodes, which store the whole BC data, two other node types: (*i*) fast nodes and (*ii*) light nodes. Fast nodes only store block headers, and light nodes have to rely on full nodes to retrieve blocks when necessary. However, full nodes are always required to perform the mining and execution of SCs. Further, Ethereum's consensus mechanism "ethash" was developed to tackle the danger of centralization imposed by mining pools in Bitcoin [56]. These mining pools exist because the PoW algorithm of Bitcoin can be implemented and performed by Application-Specific Integrated Circuits (ASIC), and miners retrieve block headers from a central pool and perform the PoW on these headers, without needing to maintain the whole BC. Ethash tackles this problem by being more memory-hungry, which is quite expensive in ASICs, and by asking for miners to fetch random blocks, which requires the

storage of the entire BC. Nevertheless, "ethash" is based on PoW and, ultimately, is susceptible to centralization, too.

As of today, Ethereum does not scale either, achieving 15 Tps [7], and it is not energy-efficient by relying on PoW consensus mechanisms. However, plans exist to move within Ethereum from PoW to a Proof-of-Stake (PoS) consensus mechanism implementation (*cf.* Section 5.4) in the near future [12]. With such a change, sharding can be implemented, which can improve transaction rates by adding clusters of PoS block validators and which contributes to secure the entire Ethereum BC. Thus, the overall energy consumption can be reduced, and the overall scalability can be improved.

HyperLedger (HL) is a cross-industry, open-source project managed by The Linux Foundation. HL includes 14 projects, four are active, and ten are in the "Incubation" phase and not yet ready for full deployment [72]. Thus, this section focus on the four active DL projects by HL:

- *HL Fabric*'s development is mainly driven by IBM, and the initial vision was first described in [1]. HL Fabric supports multiple levels of privacy. Channels enable P2P data sharing, leading to complete data isolation between a set of participants, where each channel a private communication channel between two or more specific network members, enabling confidential transactions. Private data collections enable P2P transactions between authorized participants, keeping data private to a subset of transactors (and potentially regulators/auditors). Private data is only shared P2P, with hashes stored on the BC, offering verifiable evidence to all peers, which can validate transactions. Additionally, an optional Identity Mixer can be deployed to increase the anonymity of transaction submitters.
- *HL Indy* determines an independent project but is commonly associated with "The Sovrin Foundation" [68], which is a public utility for identity, built on top of Indy's codebase. HL Indy's goal is to develop a set of decentralized identity specifications independent of any particular ledger, thus, enabling interoperability across any DL that supports them. HL Indy allows for an identity-based authentication on attributes users are willing to store and share. This can reduce the risks of certain business cases because the data can be kept with the respective user and presented in a way that others can trust and verify the data independently.
- *HL Iroha* was initially developed by Japanese developers for an application-specific BC for a mobile use case. Iroha mainly differs from other DLs, because Iroha offers a novel Byzantine Fault Tolerance (BFT) consensus algorithm called YAC, which specifies a practical decentralized consensus algorithm, guaranteeing the fast finality of transactions with low latency measures. Further, Iroha offers built-in commands as a significant benefit compared to other platforms, since it is straightforward to specify and perform everyday tasks easily, such as creating digital assets, registering accounts, and transferring assets between accounts. Moreover, the narrow focus of Iroha decreases the attack surface, and thus, improves the overall security

of the system, since fewer steps may fail or can be exploited. Finally, Iroha is the only DL with robust access control, allowing permissions to be set for all commands, queries, and joining of the network.
- *HL Sawtooth* is similar to Fabric in many aspects, with the main difference that Intel is the company driving the development of Sawtooth, which uses the Proof-of-Elapsed-Time (PoET) consensus mechanism [36]. Thus, depending on Intel's Software Guard Extensions (SGX) leaders are elected to mine blocks based on random wait times. Also, peers have access to all transaction data, unlike in Fabric, where "private" channels can be established.

At this moment, HL Fabric is the most active and prominent HL project. HL Fabric is actively researched and, in theory, able to scale up to 20,000 transactions per second [25]. HL Indy and The Sovrin Foundation are at the forefront of decentralized identifiers, and the paradigm "Privacy-by-Design" seems to be implemented well. In contrast, HL Iroha has a particular and narrow use case, focusing on mobile applications. Finally, HL Sawtooth is the most versatile of active HL projects. The main differences to Fabric cover the PoET consensus mechanism, which allows for a permissioned or permissionless deployment, whereas HL Fabric only allows for a permissioned deployment. However, the dependency on Intel's SGX for that specific consensus mechanism is unfavorable for permissionless settings, since it introduces another TTP, *i.e.,* Intel.

IOTA was developed by the IOTA community and made its public appearance in October 2015, while being supported by the IOTA Foundation. Its major difference from other BCs is that it utilizes a different data structure to store transactions, called the Tangle [55]. In the Tangle each transaction (rather than a block of transactions) references to two previous transactions, forming a complex Web structure known in mathematics as a Directed Acyclic Graph (DAG). The directed feature is needed to ensure that all reference pointers point in the same direction. Acyclicity is demanded because following a path from any transaction and arriving back at the same transaction is not allowed, (*i.e.,* no loops). Moreover, the deployment of a graph suits well, since all reference pointers and transactions form a graph of edges and vertices. Importantly, such a DAG structure allows transactions to be issued simultaneously, asynchronously, and continuously, as opposed to the discrete-time intervals and linear expansion of other BC [38].

By parallelizing transaction issuance and validation, IOTA can achieve a significantly higher transaction throughput at 1,000 Tps [15]. However, the introduction of such parallelization exposes the Tangle to uncontrolled growth attacks, such as parasite chains and splitting [55]. To solve these issues (until the development and testing of mitigation algorithms [37]), IOTA (a) introduced the concept of a central "coordinator", which confirms unreferenced transactions every minute, and (b) required that new transactions must directly or indirectly reference these confirmed transactions by the "coordinator". Ultimately, securing and controlling the Tangle at the cost of introducing a central entity [37].

In BCs, a bifurcation of roles between the miners and users exist, *i.e.,* interests are opposing (miners want slower transaction confirmation times and collect higher fees, whereas users want higher confirmation time and lower fees). In contrast, the Tangle aligns incentives of all participants equally [38], since every node issuing an IOTA transaction also actively participates in the consensus. Because there are no miners in the Tangle, transaction validation is an intrinsic part of transaction issuance, and thus, there are no transaction fees. The value which sent is always equal to the value which is received. This approach enables feeless micro and nano-payments, which emerging machine-to-machine communications, *e.g.,* for the Internet-of-Things (IoT), and a sharing economy will require to be able to operate at scale.

BAZO is a research-oriented, experimental BC, with the focus on providing an environment, where researchers and students can experiment with new consensus mechanisms, transaction and account models, sharding schemes, scalability investigations, and novel BC techniques [51]. BAZO's chain-based protocol is developed for simplicity reasons such that the PoS protocol works comparably to the PoW consensus mechanism with a key difference: BAZO designs a throttled PoW algorithm. Since a validator is limited to precisely one hash per second, BAZO defines a semi-synchronous system, such that every node can run on its own local time. If a node attempts to speed up his hashing power, the system detects the malicious node, and suggested blocks are rejected. BAZO chooses validators proportionally to the number of coins that each validator owns.

The basic design concept of the PoS protocol employed in BAZO includes a validator wishing to participate in the staking process, it has to publish a "Stake Transaction" well in advance in order to join the set of validators and he has to fulfill a PoS condition to determine if he or she is eligible to append a block to the BAZO BC. A block must contain information that proves the eligibility of the creator. However, a validator should be invulnerable if he published this information, and a block is considered orphaned in the event of a fork. Thus, a validator's secret information is used to generate publicly verifiable information to prove that the validator is eligible to append a block at height h. BAZO performs at 50 Tps within a network of 20 miners [50]. Research is being conducted on the scalability of BAZO applied sharding mechanisms to deliver a performance spectrum with transaction validations reaching up to 1,000 Tps.

3 A Tale of Two Approaches (BC vs. DL)

In their purest form, BCs act as a decentralized and public ledger, which transparently and immutably record blocks of transactions across a network of computers using a consensus algorithm. Therefore, a BC is, as proposed initially [48], open to all its participants concerning the rights of reading, writing, and participation in the consensus mechanism. A Linked List (LL), however, is a data structure that is traditionally managed by one or more trusted entities holding

write permissions. Thus, based on the process of composing new blocks of information (*i.e.,* the consensus mechanism), as well as guarantees of immutability and transparency, it can be stated that although the outcome of a BC and LL is similar, these structures are composed entirely different.

From an abstract point of view, a BC resembles an LL, which defines the abstract data structure, whose instances are interconnected by pointers. Thus, LL-based BC transactions are stored in the form of blocks within this LL by sequentially ordering new blocks (of transactions). However, these are all resemblances of LLs and BCs, since the process of composing a general chain can be very different. *I.e.,* the major differences lie in processes (*a*) of gathering information from the P2P [48] network, (*b*) of assembling information (*i.e.,* transactions) into such an abstract data structure (*i.e.,* blocks), and (*c*) on appending new blocks to the list of blocks (*i.e.,* consensus).

The capacity of a BC to provide a trustworthy, decentralized, and publicly available data storage makes it an interesting opportunity for organizations to increase business agility and to reduce costs by removing intermediaries in distributed applications [59]. However, one crucial aspect of classifying a distributed application as a BC, besides its underlying data structure, is that it should be fully public concerning (*i*) read, (*ii*) write, and (*iii*) consensus participation. Thus, if any of these actions are restricted to selected nodes, such distributed application cannot be classified as a BC, but rather as a DL.

Further, DLs can be classified in different types according to their read and write permissions. Figure 3 depicts quadrants representing a deployment type. The x-axis represents the two alternatives write permissions (permissioned or permissionless) and the y-axis represents read permissions (public or private). The light gray square represents the definition of a BC, while the dark gray squares represent DLs. Each deployment type is described as follows:

- **Public Permissioned** write permissions are restricted to selected entities, but anyone is able to read from the BC. For example, this deployment type can be used for use cases, where multiple trusted authorities want to publish public data, accessible to anyone, (*e.g.,* publishing hashes of academic certificates).
- **Public Permissionless** are the most prevalent type of BCs. Bitcoin [48], Ethereum [10], and most of their forks are considered to be public permissionless BCs, due to their read and write permissions, as well as the participation in the consensus, which is open to anyone with Internet access. Thus, public permissionless BCs are the standard type of BC deployment, and most cryptocurrencies are implemented as such.
- **Private Permissioned** trust models resemble traditional databases, where read and write permissions are restricted, and consequently, data can only be read by authorized parties. Restricting these permissions creates a hierarchy between participants (*e.g.,* role-based actions), where the main features of a BC (*e.g.,* transparency, immutability, and decentralization) may not be advantageous for a potential application.

– **Private Permissionless** are comparable to public permissionless BCs, but the notion of the reading access control is restricted to a particular (predefined) group or community. Therefore, writing and reading permissions are open to all participating members of such private groups. A dedicated supply chain BC would also act as a possible example since the information exchanged is only readable by its authorized members, but all members can issue transactions without limitations.

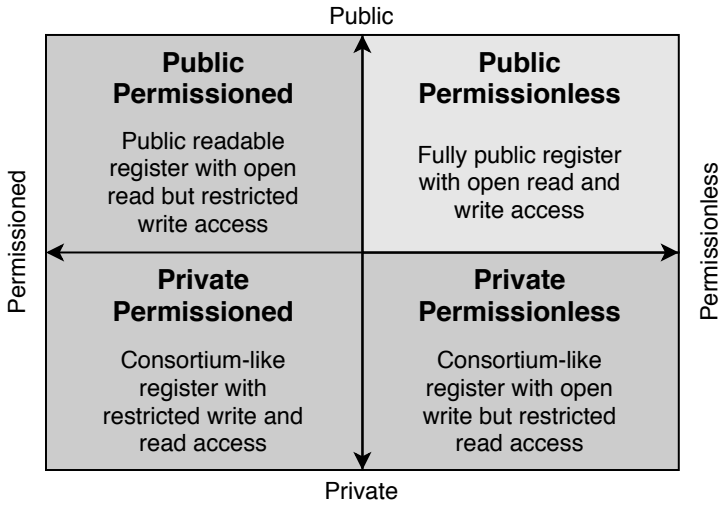

Figure 3. BC Deployment Types

Therefore, depending on the needs of the application domain, the inherent power of disintermediation can increase trust through transparency among those stakeholders involved. Nonetheless, while BCs have started their widespread adoption within the FinTech domain, many other application areas, use cases, and specific BC types are emerging, too. However, it is crucial to observe that the BC applicability relies on a multitude of different, partially contradicting facets, which are usually determined by dedicated application needs in terms of performance, security, and scalability. They have to be carefully considered in an analysis before the design decision for or against a BC or DL, respectively, can be taken.

While short-term expectations of emerging technical solutions are often overestimated, BCs are expected to cause a rather evolutionary than revolutionary change, well in contrast to many popular science publications. However with that in mind, BCs may disrupt various industries, not from a technical side, but BCs may also challenge existing business models, business processes, and stakeholder interactions since the full distribution and decentralization allow for

the first time an interaction between individual users in a secured, public, and immutable P2P manner without any trusted intermediaries.

4 Blockchains' Suitability and Achievements

Since the BCs proposal in late 2008 as an underlying infrastructure for a cryptocurrency [48] and their later application in several application domains and use cases [26, 39, 13], the BC concept shows measurably a strong decentralization, transparency, and immutability features. However, for selected use cases concerns raised, whether they require a BC [76, 31]. As of today, there is still no clear consensus on such a debate [61]. However, undoubtedly, there are three BC application domains, in which these three features are essential: (*i*) Cryptocurrencies, (*ii*) SCs, and (*iii*) proof-of-existence. Thus, a BC is a perfectly suitable solution to maintain a decentralized, publicly accessible, and append-only ledger.

4.1 Cryptocurrencies and Related Developments

The idea of digital currencies is a concept explored well before the appearance of Bitcoin [48] and summarized as within Figure 4. Since the idea of creating electronic payment methods arose in the 1960s with the demand to optimize business processes, the system SABRE (Semi-Automated Business Research Environment) by IBM proposed the automation of the booking process of airline tickets by connecting the system directly with a banking credit system [18].

Although SABRE did not propose a digital currency as such, it had been an important milestone for the integration of existing banking systems into the starting digital world. The technical idea of electronic payments was presented in 1983 [18] and proposed a blind signature scheme to automate the payment of services and goods based on privacy standards backed by cryptographic means. The following major requirements for cryptocurrencies were raised:

- **Avoiding double-spending**: to avoid spending of the same coin, which is basically a bit string that could be copied easily, a trusted party is required to be online always. Such a trusted party was also a potential target of attacks.
- **Preserving privacy of transactions**: as transactions were required to be processed and validated by a trusted party, the observation of each transaction of each user has to be prevented.
- **Origin of currency**: related to issues concerned with the creation and value-added of the electronic cash, which impacts directly on the market acceptance and possible applications of a new currency.

DigiCash [17] was presented in 1990 based on this blind signatures scheme [18]. Although not being successful at the time, it was the first electronic currency based on cryptographic standards preventing double-spending and supporting privacy aspects of transactions. With Deutsche Bank and Credit Suisse on board and despite causing an impact on the financial market at the time, the currency

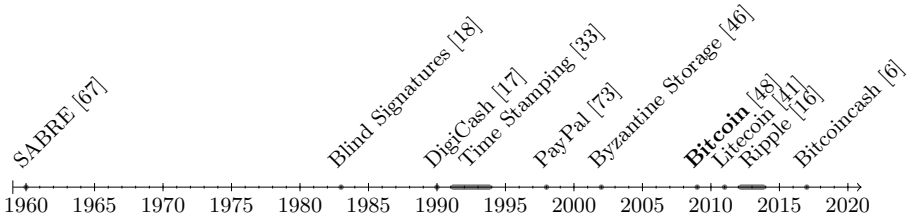

Figure 4. Timeline of Selected Digital Currencies and Related Developments

did not reach a successful position with the public. The company specialized in digital currency and payment systems operated for a few years only before declaring bankruptcy in 1996.

In the meantime, work on time-stamping digital documents, especially the data and not the medium, was proposed with computationally practical procedures (hash functions and digital signatures), such that it is infeasible for a user either to back-date or to forward-date a time-stamped document [33]. While at the same time, the complete privacy of documents themselves is reached, the approach does not require any record-keeping by the time-stamping service.

The popularity of the Internet, e-commerce, and commercial applications boosted at the turn of the century the increased demand for secure transactions over the Internet and digital currencies [73]. Based on this, PayPal [73] emerged in 1998 as an innovative money transfer service, differing from credit card services. Paypal integrated features to reduce fraud in electronic transactions, such as offering a virtual account, where a customer can add money that could be spent in any currency being converted and deducted from this virtual account.

While cryptographic storage techniques for data had been developed within the '90s, allowing users to keep data secret from non-trusted servers, the multi-user network file system SUNDR (Secure Untrusted Data Repository) focused on the detection of tampering attacks and stale data [46]. Whether or not the server obeys the protocol the ideal and immediate as well as unconditional notice of any misbehavior on the part of the server cannot be reached, however, fork consistency (a weaker form of data integrity) can be proven: if the server delays just one single user from seeing even a single change by another, the two users will never again see one another's changes. This can easily be detected with online communications since users own private keys. Thus, SUNDR's virtual nodes contain the file's metadata, the size, and cryptographic hashes of its blocks (the user data), while hash trees are applied to verify a file block's integrity without touching the entire file system. The server stores in addition to the blocks a signed version structure (which includes version data and the i-handle, a single cryptographic hash of the block) for each user and group, including information about operations in progress. SUNDR's practical consistency protocol enables the detection, whether the server has faithfully provided a consistent view of the entire file system to all clients. Thus, SUNDR defines a network file system,

whose protocol makes Byzantine file server failures readily detectable — a major prerequisite of a modern's BC functionality.

Until the Bitcoin proposal appeared [48], basically all security approaches deployed preventing double-spending and supporting transaction privacy followed a centralized path, many proposals relying on a TTP as a mediator of online operations. Their similar model compared to fiat currencies, did not trigger a widespread adoption. This was the case of DigiCash [17], which reached a global recognition within financial markets but could not grow its user base to a sufficient size. In contrast, PayPal's innovative financial characteristics, such as virtual wallets and the simplicity of previously complex financial operations (*e.g.,* currency exchange or online payments, marked the beginning of the FinTech area. Bitcoin introduced 2009 a cryptocurrency operating in a fully decentralized manner while being able to respond to concerns raised earlier [48]:

- **Double-spending**: a guarantee of detecting double-spending at any time without a trusted intermediary being involved is reached by the PoW consensus mechanism (*cf.* Section 2) deployed. Thus, participants wait for the creation of a certain number of blocks to ensure that a transaction is confirmed on the network, and a comparison against this chain reveals if a coin had been spent twice.
- **Privacy of transactions**: although the transaction history is public and verifiable, all authors of transactions are identified by a public key. Thus, the approach guarantees pseudonymity since it is not easily possible to track the public key to its owner directly.
- **Origin of currency**: the proposed approach seeks to add value to the currency by creating a relative scarcity, (*i.e.,* the number of coins is limited) and by ensuring that efforts to acquire Bitcoin by miners, (*i.e.,* by mining) require an investment in computing power (*cf.* Section 2.2).

Despite skepticism in early stages, the Bitcoin work and the underlying public DL, the BC, rose considerable interest due to its dis-intermediation feature, while aligning a balance between anonymity/privacy of users and the transparency of their transactions. As of 2011, several cryptocurrencies for specific purposes emerged based on parametric changes of the Bitcoin source code, *e.g.,* Litecoin [41], and new ones, such as Ripple [16], focused on the integration of banking systems. Public perception was secured at the moment the market capitalization reached 1 billion US$ in Bitcoins in March 2013 [54]. Thus, interest swapped from economic investors (on very shaky grounds) to industry, especially Small- and Medium-sized Enterprises (SME), and academia. The platform enabling Bitcoins to be exchanged safely in a fully distributed manner between unknown stakeholders and without any central element as a mediator, the public BC formed the start of the BC age.

4.2 Smart Contracts

The notion of a SC, as a computerized transaction protocol that executes the terms of a contract agreed between the parties involved, was proposed long

before the arrival of BCs [71]. The lack of a decentralized infrastructure providing a suitable dis-intermediation was resolved by BCs, and SCs now became implementable, since BCs outline the perfect distributed environment for their deployment and operation. The goal of an SC is to (*a*) satisfy common contractual conditions as with any regular paper-based contracts, *e.g.,* in terms of payments, liens, confidentiality, or even enforcement, (*b*) reduce malicious and accidental handling, and (*c*) avoid any trusted intermediaries. Thus, the SCs concept is a viable path to automate and ensure agreements reliably and more efficiently than paper-based contracts as of today. Henceforth, as specifically designed from scratch in Ethereum [10], the BC has proven to be a highly appropriate infrastructure for the fully decentralized and transparent execution of a mutual agreement between parties.

Despite Ethereum becoming the first and widely accessible platform that unveiled the entire potential of SCs, Bitcoin also features contracts, however, only in the form of simple scripts for performing transactions, which suite the finality of the application, *i.e.,* a cryptocurrency, in a simple and efficient manner [48]. In contrast, Ethereum proposed a sandbox environment through the Ethereum Virtual Machine (EVM) [10], in which it is possible to execute arbitrary and Turing-complete code directly on-chain, (*i.e.,* allowing for the execution of loops). An EVM defines an environment isolated from the host itself, being precisely the same for all Ethereum nodes (called "Ethereum Clients") in the BC network. A client software, *e.g.,* Geth and Parity, is used for external communications and interactions with the operating system of the host node.

Although many different BCs provide support for the execution of SCs, the majority follows the model determined by Ethereum, in which a sandboxed environment ensures that the execution of the SCs is precisely the same across all nodes of the network. Thus, Figure 5 illustrates the respective components involved in the creation of an SC in an EVM as well as the deployment steps needed. While the SC is defined in a high-level language, *e.g.,* by applying the Integrated Development Environment (IDE) Remix, it is transformed and interpreted in bytecode, until it is propagated on to the BC network according to the consensus algorithm configured in the EVM. The EVM itself operates on the respective operating system and runs the Ethereum protocol. Thus, the client communicates with the host's operating system to broadcast the transaction containing the bytecode corresponding to the SC, which is crafted into different IP packets and sent to the BC network. The role of the EVM (*i.e.,* the BC's "virtual machine") is crucial, since code must be identical across all Ethereum nodes in the BC network and must comply with well-defined interfaces. Therefore, it is possible to enable flexible support for different clients, which, in turn, can provide different abstraction levels for the development of applications.

A relevant factor for the popularity of general-purpose, decentralized applications (dApps) based on SCs, is the familiarization of developers with high-level SC programming languages and the *modus-operandi* of BC. Following the SC example as shown in Figure 5, it is possible to create a simple application returning a string the text *hello world* whenever it is called (*cf.* Listing 1). Ethereum pro-

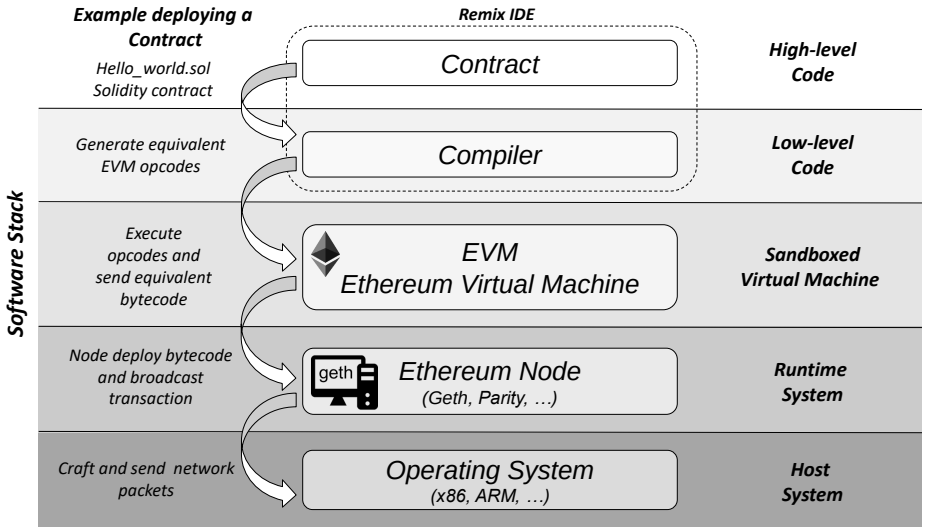

Figure 5. SC Deployment on the Ethereum Virtual Machine (EVM) [10]

vides direct support for its high-level language "Solidity", with a syntax based on JavaScript; however, support for different languages that developers are familiar with exist if that language is compiled into EVM op-codes, which are interpreted and executed by the EVM. Thus, such flexibilization is made possible through such abstraction layer allowing the use of different languages as long as your compiler generates EVM opcodes.

```
1 pragma solidity ^0.4.10;
2 contract HelloContract {
3       string helloWorld;
4       function getHello() public {
5            helloWorld = "Hello World";
6       }
7 }
```

Listing 1. A Simple "Hello World" SC Example in Solidity

Regardless of the high-level language of a SC, the EVM interprets and executes EVM opcodes based on an incentive scheme, the *gas*. Thus, the higher the complexity of an SC is, the higher will be the cost for its deployment and operation, demanding a higher amount of *gas* for its execution. It is important to note that such an incentive scheme is required for BCs, *i.e.,* their permissionless deployments, since anyone with Internet access can participate and, thus, a mechanism is needed to prevent the BC from DoS attacks, either maliciously or just by accident, *i.e.,* an endless loop within an SC. In contrast, for DLs, especially permissioned deployments, such a necessity of incentives may not be needed, since the BC network consists out of permissioned, *i.e.,* pre-selected

stakeholders. Once a sufficient amount of *Gas* is provided for the SC's deployment, the EVM generates the bytecode, which is sent to the client.

4.3 Proof-of-Existence and Time-stamping

One of the significant BC characteristics is data immutability. Once information is inserted and broadcast into the BC, its removal or subsequent alteration becomes practically impossible without the (unlikely) control of the majority of BC network nodes. Therefore, many solutions that require the registration of an event at a certain point in time exploit exactly this BC characteristics to reach an immutable time-stamping of this event, typically represented in a digital format. Within this immutability context, two use cases benefit from it: (*i*) Certificate Handling and (*ii*) Supply Chain Tracking (SCT).

Certificate Handling: certificates play an essential role, since as an official document they attest a fact. In case of academic certificates they determine knowledge in a particular area, are issued by accredited academic institutions, and often delivered on paper to students. To assess the authenticity of certificates today, companies or universities must call the issuing institution to perform this verification, which is a time-consuming and cumbersome process.

Due to the fact that in 2017 about 500 fake doctoral certificates were sold monthly in the US in "diploma mills" [53], a secure, efficient, and decentralized solution for the verification of certificates becomes relevant. Therefore, the role of a BC and its main characteristics indicate that a BC-based proof-of-existence for handling certificates is possible, and several projects follow this path [14, 60, 32]. In General, hashes of the certificates' PDF documents are immutably stored in the BC, persisted, and signed by the issuing institution with its private key. Thus, the verification process is simplified overall by matching the hash of the currently accessible PDF version of the certificate to the hash stored in the BC by the issuing institution. In addition, such BC-based certificate handling systems tackle the issue of fake certificates from "diploma mills" successfully, since they cannot store certificate hashes in the BC without the institution's private key, in turn, they are not able to issue certificates of counterfeited institutions.

Supply Chain Tracking: there are many use cases deploying Supply Chain Tracking (SCT) systems based on BCs. Since BC and SCT integration supports the persisting of data, their integrity, and, consequently, the establishment of trusted boundaries between different stakeholders for a product's life cycle. Since the production of a particular product encompasses a large number of actions and conditions during its life cycle, collecting throughout a product's supply chain time-stamped quantitative and qualitative data from the process, even including the geo-location of resources or times when certain steps had been performed, adds value beyond financial incentives such as region labels. To collect data required automatically, wireless sensors, IoT devices, and surveillance devices, such as cameras and drones, are integrated. Recent studies, [69],

[8], and [4] elaborate on several aspects employing BCs in SCT systems. The key takeaway is that the persistence of the supply chain data tracked within the BCs does seal SCT systems by protecting the data's integrity, however, while the time stamping mechanism and its related undeniable evidence of actions can be proven, the exact values persisted depend on the sensors in use. Supply chain tracking solutions at the enterprise level do include the food industry and many other vendor domains. However, the full decentralization of partially unknown stakeholders within a chain faces pioneer players in this area with challenges.

Even though certificates and SCT determine use cases benefiting from data immutability, modulo the sensor problem as sketched above, there exists an inherent problem with data structures in general concerning the semantics of data stored. Although existing a record that an event occurred at some point in time, it is not possible to verify at the BC's data structure level the semantics of data stored without specific application-level knowledge (*cf.* Section 5.3).

5 Missing Pieces and Open Issues

BCs and DLs serve as an ideal infrastructure for select use cases (*cf.* Section 4). However, there is still a considerable amount of research required, in order to fully answer the question, which systems could be replaced by a BC- or DL-based approach, such that efficiency gains compared to centralized approaches become measurable. Such efforts include especially (*a*) technical complexity, (*b*) interoperability and openness, (*c*) trust, and (*d*) sustainability, besides pure technical efficiency, manageability including identity management, and operations. The gap of processing sensed data, such as within IoT, and the trust in respectively digitized data will always depend on the calibration and certification of sensors.

5.1 Technical Complexity

The myriad of BC implementations emerging requires that users and companies desiring to adopt a BC —either by integrating it in their solution or creating novel solutions — need to understand the underlying BC technical details, *e.g.,* transaction structure, parameters, transaction signing methods, and address format, of each implementation under consideration. In this context, it is very difficult for non-technical users, *i.e.,* users without prior BC knowledge, to select the most appropriate BC implementation for their use cases. Thus, an approach which directly leads to an increase in the Operational Expenditures (OPEX) of the company is favored over an approach where the company must (*a*) train dedicated personnel or (*b*) hire external consultants to help in the BC selection process. The lack of *"in-house"* capabilities (*i.e.,* skills and understating) was recently highlighted in a global survey by Deloitte [44] as one barrier to more significant investments in BC. Thus, simplifying and abstracting technical details is a crucial challenge to foster the adoption of BCs on a wider scale and to generate valuable investments. An approach to tackle this issue is [63], where a framework to automatically select the best suitable BC for a determined

use-case is proposed. In the approach, users define policies that contain information regarding the BC requirements (*e.g.,* minimum Tps, required data size, deployment type, and costs) to store data from different users. The framework then automatically selects the BC that meets the requirements and manages the transactions to the selected BC. Hence, simplifying the complexity of managing data from multiple users in different BCs.

Moreover, besides this company-oriented view, public key and private key concepts (*cf.* Section 2.1) are not easy to be understood for non-technical individuals due to their underlying mathematical and computational complexity. This leads to the fact that many cryptocurrency wallets and funds are being managed by central online exchanges, which form a highly profitable target for attackers. Since such central exchanges hold all their clients' funds, the Mt-Gox hack in 2014 was economically successful, 460 million US$ were supposedly stolen [47]. Neither Bitcoin nor Ethereum faced fundamental technical attacks on their protocol and system-internals [19]; most of the attacks targeted external services around these two BCs had been exposed successfully to threats and attacks. Thus, besides BC's highly stable technical dimension, which typically includes the BC protocol, the mining, if applied, and the chain's storage, BCs also require a transparent and straightforward social dimension to be widely accepted as a secure mechanism for handling financial transactions. This does have to involve the BC's ecosystem around the pure BC specification; thus, the technical complexity even grows further.

5.2 Interoperability

As of today, BCs form separated islands of technical solutions, which do not enable a native communication between any two ones. Thus, efforts toward the standardization of BCs and DLs are required and they include international organizations, *e.g.,* International Organization for Standardization (ISO) and the Internet Research Task Force (IRTF), and also focus groups, *e.g.,* ITU Telecommunication Standardization Sector (ITU-T) and Securities Association for Institutional Trade Communication (ISITC) Europe. Standardization are in the process to define a common foundation and set of protocols for a data exchange between BCs and DLs, proving a *de facto* standard for BCs and DLs. Thus, standardization can be viewed as an enabler for interoperability [11, 5].

Existing interoperability schemes cover (*i*) a Notary Scheme, (*ii*) Side-chains and Relays, and (*iii*) Hash Locking [11]. Notary schemes introduce one or more trusted entities that verify events in different BCs and provide information to external applications. With the deployment of a notary, applications are not restricted to a single BC platform, technical details of BC interaction can be abstracted, which simplifies the BC integration with legacy systems [62], however, at the cost of the re-introduction of a trusted, this, central entity. In contrast, side-chains and relays propose to maintain a parallel BC, which includes a consensus mechanism, where multiple nodes, *i.e.,* miners, validate events in the different chains and include the blocks into the parallel chain. In this sense, trust can be provided without relying on a central entity. However, it comes at

the cost of introducing additional technical complexity to reach interoperability, including the demand for additional storage for the parallel chain. Hash locking allows for an exchange of funds between two different BCs without any intermediary. However, hash locking requires that both BCs support Hashed TimeLock Contracts (HTLC), and these BCs are not able to exchange data or act upon generic events. Thus, a generic, decentralized, efficient, and open interoperability scheme for arbitrary BCs has not been found yet—which is very well visible today in all currently ongoing standardization activities on interoperability.

5.3 Trust Assumptions

BCs as of today can provide the integrity of the *data-at-rest* (data stored in persistent storage) and *data-in-transit* (information flowing over private or public networks), but not of *data-in-use* (active data, which is stored in a non-persistent digital state). The integrity of *data-at-rest* is secured via the consensus mechanism, the full distribution, and hash-based linking, (*i.e.,* immutability). In contrast, the integrity of *data-in-transit* is ensured by the public-private key pairs in use, *e.g.,* one cannot change a transaction already signed without being detected. However, signing a transaction does not mean that the transaction will be accepted or processed, *e.g.,* it can be rejected due to lack of funds or low fees. Further, legit transactions and legit BC traffic might be rerouted to fake nodes with a Border Gateway Protocol (BGP) hijacking attack [2] or by monopolizing connections with eclipse attacks [34].

In terms of *data-in-use*, its trustworthiness depends on the application creating the transaction. Thus, any BC cannot guarantee that data included in the transaction are trustworthy unless the generation process of these data is public, fully electronic, and observable by anyone - the exact case of a cryptocurrency, like Bitcoins. This problem of providing trust in general terms to *data-in-use* is that the input to the BC, if highly depending on IoT environments, originates from physical sensors. As these sensors are susceptible to a myriad of attacks, *e.g.,* node capture, false data injection, spoofing, sibyl attacks, and physical tampering [43], there is no guarantee that the *data-in-use* from IoT sensors is trustworthy, (*i.e.,* was not modified). However, the date and time as well as possibly the geo-location may be trustworthy, since respective control data is based on trustworthy algorithms. Thus, the verification of data from IoT and other sensors and their related integrity must be performed outside the BC before being immutability persisted within the BC.

Overall, a BC can be viewed as an enabler of the "Security Through Publicity" concept, which is defined as "*Acting in public does not guarantee that actions will be correct, but it does provide users with a quantifiable set of information and semantics that enable applications to construct meaningful security mechanisms*" [52]. In this sense, BCs acting as a public and open record of transactions allow for the building of secure systems on top of applications. Users interacting with a BC are acting in public (user interacting with a DL may most likely not fully act in public), which means that their actions (*e.g.,* transactions sent and SC calls), and their data (*e.g.,* account balance) can be verified by any peer or

application interested. Of course, this does not guarantee that their behavior or data is correct. However, it provides the guarantee that, following the BC protocol, their actions cannot be forged time- or date-wise or changed otherwise. Hence, reinforcing the statement: BCs serve as a trusted source of information, but solely for information generated within the BC protocol (such as for cryptocurrencies); data generated from external sources may not be trustworthy.

For any SC deployed on a BC, if that is possible, the trust assumptions made are comparable: if the SC code is provided is very basic and simple statements, which can be formally verified, the result of such an SC operation can be trusted. However, SC code potentially belongs to the category of "non-trusted" code, since a formal verification is not attached to an SC such that their integrity could be traced automatically. Note that this relates to the discussion in which the statement "Code is the law" suggests that SCs are final. In the sense of computation of that SC's opcodes, one by one in the given order, this holds true; thus, the result is trustworthy, and any node in the BC network can verify it. However, the programming language-based semantics of this SC are different, since the interpretation of the validity of a certain sequence of functions to be called to be considered right or not being allowed varies. In case these calls violate the intention of the SC, such a situation is much harder, if not impossible in general, to be formalized as the DAO attack [22] on Ethereum shows.

5.4 Sustainability and Consensus Mechanisms

The mining process based on PoW as described in Section 2.2 secures the BC network against double-spending. This is achieved by an investment into a larger amount of energy to find the correct solution to the crypto puzzle. As of April 2021, the Bitcoin network consumed 94.92 TWh of energy as estimated by [23], which is comparable to the annual consumption of the entire country of Kazakhstan. Therefore, sustainability concerns arise due the PoW consensus mechanisms in use. To tackle this concern of power consumption and to secure the BC at the same level of security, the validation of transactions may be possible via alternative mining schemes [74]. Among the myriad of alternatives some of them are already deployed in operational and research BCs, such as Proof-of-Stake (PoS) in Bazo, delegated PoS (dPoS) in EOS, and Proof-of-Authority (PoA) in private deployments of Ethereum. Thus, a discussion of respective mechanisms is required to determine the current state-of-the-art.

Proof-of-Stake (PoS): In PoS-based mining the probability of a miner to solve a crypto puzzle depends solely on the number of coins a mining node is willing to put at stake to secure the BC. This is in contrast to PoW, where the probability of a miner to find a solution to the crypto puzzle is related to the amount of computational power it possesses. Therefore, nodes with larger stake are more likely to be chosen to validate transactions and to include a block into the BC compared to nodes holding fewer coins. This main idea of PoS is herewith based on the fact that nodes holding more coins in a BC will contribute to its security,

since they have more to lose. In turn, wealthier nodes tend to become wealthier as they verify more blocks more time and collect additional transaction fees.

Since the choice of the next miner is not based on computational power, but on the stake (*i.e.,* wealth), the energy consumption of PoS is negligible compared to PoW [49]. However, a key problem with PoS is the "nothing-at-stake" problem. This arises because nodes potentially can verify concurrently blocks in several branches of the BC. After all, it costs a node nothing to do so, since the stake offered is valid for the entire BC. Thus, with nodes verifying blocks in several branches of the same BC at the same time, such branches show a larger likelihood to be never abandoned, and in turn, no consensus could be reached to which of the chain's branches all nodes have to convert to.

Delegated PoS (dPoS): By adding a democratic process to PoS, dPoS was proposed to solve the "nothing-at-stake" problem. This democratic process defines the selection of nodes that will add blocks to the BC. For dPoS the probability of a node to be selected as the node to persist a block int othe BC depends on (a) the wealth staked and (b) on the votes provided by other nodes in the BC. Therefore in contrast to PoS, it is not sufficient for a node to have wealth staked, a node must also behave in a manner that ensures a correct and fair overall state of the BC, *e.g.,* by only verifying transactions in a single branch. If a node starts to behave maliciously, other nodes in the BC can remove their votes and place them on another node, which behaves correctly. However, this democratic process comes at a cost, voting for a node might be impacted by financial rewards for other nodes to place their votes for a particular node [70], *i.e.,* buying votes. In practice, the set of verifier nodes in dPoS-based BCs is often restricted to a predefined number, *e.g.,* 21 verifiers in the case of EOS. Thus, while the sustainability argument has seen a relieve of pressure, a possible centralization of mining power in a dPoS-based BC moves in instead.

Proof-of-Authority (PoA): If only a set of "pre-authorized" nodes hold the rights to write a new block into a BC, the developed PoA miming scheme introduces "pre-authorized" nodes. This scheme is often found in DLs, where a selected group of individuals (*i.e.,* organizations, participants, or selected nodes) is allowed to interact with the BC. While this approach relaxes concerns on a higher energy consumption as for PoS, this advantage comes at the cost of a high centralization. Since the number of block verifiers (*i.e.,* miners) is limited and known, they potentially can collude and perform malicious actions, such as censuring transactions or delaying the same.

In summary, Table 2 compares these alternatives described. As the energy consumption lowers, the level of centralization increases. This is due to the fact that whenever a set of trusted participants act to secure the BC, the consensus mechanism can be simplified, *e.g.,* from PoW to PoS or PoA. In contrast, if participants are not known and do not trust each other, any useful consensus mechanism must ensure that it is computationally (*e.g.,* PoW) or financially

(*e.g.,* PoS) expensive to propose a new block to be added to that BC. Consequently, academia and industry is engaged in developing and experimenting consensus mechanisms that are both secure and energy-efficient at the same time and could replace PoW-based schemes [74].

Table 2. Non-Exhaustive Comparison of Alternative Consensus Mechanisms

Consensus Mechanism	Key Principle	Energy Consumption	Probability of Centralization
Proof-of-Work (PoW)	Computational power determines chance of adding a block	High	Low
Proof-of-Stake (PoS)	Staked wealth determines the chance of adding a block	Low	Medium
Delegated Proof-of-Stake (dPoS)	Votes by nodes and staked wealth determine the chance of adding a block	Low	Medium
Proof-of-Authority (PoA)	Only a set of authorized nodes are able to add blocks	Low	High

6 Observations and Conclusions

This article provides clarification on two concepts that received high attention over the past decade, namely Blockchains (BC) and Distributed Ledgers (DL). Both BCs and DLs deploy identical technical concepts (*cf.* Section 2), especially Peer-to-Peer (P2P) principles, applied strong cryptography, and consensus mechanisms. Overall, BCs and DLs achieve an immutable and distributed record of transactions. However, DLs often rely on a single Trusted Third Party (TTP) or on a consortium of entities (pre-defined stakeholders) to add new transactions and to maintain the ledger. In contrast, BCs offer a fully public and open participation in terms of reads, writes, and consensus mechanisms. Thus, BCs fully remove any intermediaries, thus, any need for TTPs, while still providing a fully decentralized and immutable ledger. While every BC is a DL, not all DLs are BCs. The respective technical details on this distinction were developed in Section 3, and respective classification were presented.

As of today, not a single IT system existing was fully replaced by a DL or by a BC. This leads to the conclusion that BCs determine an evolution of databases and linked lists, they do not result in a revolution, since a full replacement of a system or technology did not happen, yet. However, those overall benefits and technical developments, as well as refinements as provided by DLs and BCs (*cf.* Section 4), are remarkable since the proposal of Bitcoin in 2009. Moreover, organizational impacts are vast, especially providing in selected instances, *e.g.,* cryptocurrencies, SCs, and time-stamping, an evolution in the way stakeholders interact to reach transparency and traceability of different events and actions.

However, the key question to be answered in case of a new development foreseen reads as: "Is a distribution and decentralization of a solution really sensible?" Additionally, for many use cases, the deployment of a DL- or BC-based solution can benefit from those advantages discussed; however, this may come at the cost of an application's inefficiency due to the distributed complexity of a distributed system. Furthermore, incentives and counter-incentivization need to be considered in detail with respect to economic impacts.

In conclusion, by missing a clear definition of the differences between a BC and DL (now being provided above with measurable characteristics, especially with respect to very different trust assumptions on stakeholders involved), a culprit for their inefficient, incomplete, and possibly incorrect deployment combined with misleading expectations may have been found. However, the novelty of these new concepts and their partially "unknown" technical details are possible obstacles as well. A key concern remains in terms of trust assumptions made, because data persisted within a BC are only trustworthy, if they had been generated by an electronically computed algorithm, thus, trust depends in case of IoT applications deploying BCs on the calibration and certification of sensors. Any IoT-driven data, originating from sensors, can only be persisted in a trusted manner with respect to time and geo-location; any further correctness or validity of such data depends on the application and the sensors in use.

Generally, DLs and BCs still require a considerable research effort to optimize technical details, especially lowering technical complexity such that solutions can be integrated easily, defining standards and guidelines for their deployments and interoperability, demystifying trust assumptions, and solving the sustainability question of secure consensus protocols without losing data integrity. These selected research efforts as of IFIP's WG6.6 focus on concepts and protocols, which will need to reach identical maturity levels as known for databases and linked lists. Additionally, probabilities play an important role in BCs and DLs, since many security and consensus mechanisms in a distributed setting are built on probabilities. Thus, hash functions, public-private key pairs, *nonces*, mining power and success, and mining pools do not show a deterministic behavior, which seems to be highly relevant for users and their perceptions in today's societies. However, mathematically proven likelihoods indicate that strong security reached is already accepted within many IT systems of today's use in the FinTech and other domains.

Thus, the question of which IT systems can be (entirely) replaced by a BC- or a DL-based approach or which applications — besides those generating electronically digitally represented values — can benefit in full from these BC and DL advantages outlined, will remain open until answers with many variables to be considered and as addressed above will have been found.

Acknowledgements

This paper was driven by the IFIP WG6.6 involvement of the authors, and thus supported partially by *(a)* the University of Zürich UZH, Switzerland, and *(b)* the European Union's Horizon 2020 Research and Innovation Program under Grant Agreement No. 830927, the CONCORDIA Project.

References

1. Androulaki, E., Barger, A., Bortnikov, V., Cachin, C., Christidis, K., De Caro, A., Enyeart, D., Ferris, C., Laventman, G., Manevich, Y., Muralidharan, S., Murthy, C., Nguyen, B., Sethi, M., Singh, G., Smith, K., Sorniotti, A., Stathakopoulou, C., Vukolić, M., Cocco, S.W., Yellick, J.: Hyperledger Fabric: A Distributed Operating System for Permissioned Blockchains. In: ACM EuroSys Conference (EuroSys 2018). pp. 1–15. Porto, Portugal (April 2018)
2. Apostolaki, M., Zohar, A., Vanbever, L.: Hijacking Bitcoin: Routing Attacks on Cryptocurrencies. In: IEEE Symposium on Security and Privacy (SP 2017). pp. 375–392. San Jose, USA (May 2017)
3. Asokan, N., Janson, P.A., Steiner, M., Waidner, M.: The State-of-the-Art in Electronic Payment Systems. In: IEEE Computer. vol. 30, pp. 28–35 (September 1997)
4. Banerjee, A.: Blockchain with IoT: Applications and Use Cases for a New Paradigm of Supply Chain Driving Efficiency and Cost. In: Role of Blockchain Technology in IoT Applications, Advances in Computers, vol. 115, pp. 259–292. Elsevier (2019)
5. Belotti, M., Božić, N., Pujolle, G., Secci, S.: A Vademecum on Blockchain Technologies: When, Which and How. In: IEEE Communications Surveys and Tutorials. vol. 21, pp. 3796–3838 (July 2019)
6. bitcoincash.org: Bitcoin Cash - Peer-to-Peer Electronic Cash (2017), https://www.bitcoincash.org/
7. Blockchair: Ethereum Transaction per Second Graph (2021), https://blockchair.com/ethereum/charts/transactions-per-second
8. Bocek, T., Rodrigues, B.B., Strasser, T., Stiller, B.: Blockchains Everywhere-a Use-case of Blockchains in the Pharma Supply-chain. In: IFIP/IEEE Symposium on Integrated Network and Service Management (IM 2017). pp. 772–777. Lisbon, Portugal (May 2017)
9. Bocek, T., Stiller, B.: Smart Contracts – Blockchains in the Wings. In: Linnhoff-Popien, C., Schneider, R., Zaddach, M. (eds.) Digital Marketplaces Unleashed. pp. 169–184. Springer, Berlin, Germany (2017)
10. Buterin, V.: A Next-Generation Smart Contract and Decentralized Application Platform (2014), https://ethereum.org/en/whitepaper/
11. Buterin, V.: Chain Interoperability (September 2016), https://bit.ly/2WH8BM7
12. Buterin, V., Reijsbergen, D., Leonardos, S., Piliouras, G.: Incentives in Ethereum's Hybrid Casper Protocol. In: IEEE International Conference on Blockchain and Cryptocurrency (ICBC 2019). pp. 236–244. Seoul, South Korea (May 2019)
13. Casino, F., Dasaklis, T.K., Patsakis, C.: A Systematic Literature Review of Blockchain-based Applications: Current Status, Classification and Open Issues. In: Telematics and Informatics Journal. vol. 36, pp. 55–81. Elsevier (March 2019)
14. Castor, A.: Cardano Blockchain's First Use Case: Proof of University Diplomas in Greece (January 2018), https://bit.ly/2DVsrYt

15. Chandel, S., Zhang, S., Wu, H.: Using Blockchain in IoT: Is it a Smooth Road Ahead for Real? In: Future of Information and Communication Conference (FICC 2020). pp. 159–171. San Francisco, USA (March 2020)
16. Chase, B., MacBrough, E.: Analysis of the XRP Ledger Consensus Protocol (2018), http://arxiv.org/abs/1802.07242
17. Chaum, D., Fiat, A., Naor, M.: Untraceable Electronic Cash. In: IACR Advances in Cryptology (CRYPTO '88). pp. 319–327. Springer-Verlag, Santa Barbara, USA (August 1988)
18. Chaum, D.: Blind Signatures for Untraceable Payments. In: Advances in cryptology. pp. 199–203. Springer (1983)
19. Chen, H., Pendleton, M., Njilla, L., Xu, S.: A Survey on Ethereum Systems Security: Vulnerabilities, Attacks, and Defenses. ACM Computing Surveys **53**(3) (June 2020)
20. CoinMarketCap: Market Capitalizations (2021), https://coinmarketcap.com/
21. Croman, K., Decker, C., Eyal, I., Gencer, A.E., Juels, A., Kosba, A., Miller, A., Saxena, P., Shi, E., Gün Sirer, E., Song, D., Wattenhofer, R.: On Scaling Decentralized Blockchains. In: International Conference on Financial Cryptography and Data Security (FC 2016). pp. 106–125. Christ Church, Barbados (February 2016)
22. Daian, P.: Analysis of the DAO Exploit (2016), https://bit.ly/3ju13GC
23. Digiconomist: Bitcoin Energy Consumption Index (2021), https://digiconomist.net/bitcoin-energy-consumption
24. EOS.IO: Technical White Paper v2 (March 2018), https://bit.ly/3fGmn9k
25. Ferris, C.: Does Hyperledger Fabric Perform at Scale? (April 2019), https://ibm.co/3hkTfov
26. Franco, M.F., Scheid, E., Granville, L., Stiller, B.: BRAIN: Blockchain-based Reverse Auction for Infrastructure Supply in Virtual Network Functions-as-a-Service. In: IFIP Networking (Networking 2019). pp. 1–9. Warsaw, Poland (May 2019)
27. Garfinkel, S.L.: Public Key Cryptography. In: IEEE Computer. vol. 29, pp. 101–104 (June 1996)
28. Gervais, A., Karame, G.O., Wüst, K., Glykantzis, V., Ritzdorf, H., Capkun, S.: On the Security and Performance of Proof of Work Blockchains. In: ACM SIGSAC Conference on Computer and Communications Security (CCS 2016). pp. 3–16. Vienna, Austria (October 2016)
29. Ghimire, S., Selvaraj, H.: A Survey on Bitcoin Cryptocurrency and its Mining. In: International Conference on Systems Engineering (ICSEng 2018). pp. 1–6. Sydney, Australia (2018)
30. GitHub Inc.: Blockchain - GitHub Topics (2021), https://github.com/topics/blockchain
31. Greenspan, G.: Avoiding the Pointless Blockchain Project (2015), https://bit.ly/2Bj8wH3
32. Gresch, J., Rodrigues, B., Scheid, E., Kanhere, S.S., Stiller, B.: The Proposal of a Blockchain-based Architecture for Transparent Certificate Handling. In: International Conference on Business Information Systems (BIS 2018). pp. 1–12. Berlin, Germany (July 2018)
33. Haber, S., Stornetta, W.S.: How to Time-Stamp a Digital Document. vol. 3, pp. 99–111. Springer-Verlag, Berlin, Heidelberg (January 1991)
34. Heilman, E., Kendler, A., Zohar, A., Goldberg, S.: Eclipse Attacks on Bitcoin's Peer-to-Peer Network. In: USENIX Security Symposium (USENIX Security 15). pp. 129–144. Washington D.C, USA (August 2015)
35. ICODATA.io: ICO Funds Raised (2021), https://www.icodata.io/stats/

36. Intel Corporation: Proof of Elapsed Time (PoET) (2017), https://bit.ly/2OG1ejJ
37. IOTA Foundation: The Coordicide (May 2019), https://bit.ly/3cX1QOO
38. IOTA Foundation: IOTA (2021), https://www.iota.org/
39. Killer, C., Rodrigues, B., Scheid, E.J., Franco, M., Eck, M., Zaugg, N., Scheitlin, A., Stiller, B.: Provotum: A Blockchain-Based and End-to-End Verifiable Remote Electronic Voting System. In: IEEE 45th Conference on Local Computer Networks (LCN 2020). pp. 1–12. Sidney, Australia (November 2020)
40. Kurt, A.M.: Zero to Monero - First Edition (2018), https://bit.ly/3fOpBYC
41. Lee, C.: Litecoin - The Cryptocurrency for Payments (2011), https://litecoin.org/
42. Lin, I.C., Liao, T.C.: A Survey of Blockchain Security Issues and Challenges. In: International Journal of Network Security. vol. 19, pp. 653–659 (September 2017)
43. Lin, J., Yu, W., Zhang, N., Yang, X., Zhang, H., Zhao, W.: A Survey on Internet of Things: Architecture, Enabling Technologies, Security and Privacy, and Applications. In: IEEE Internet of Things Journal. vol. 4, pp. 1125–1142 (March 2017)
44. Linda Pawczuk, Rob Massey, J.H.: Deloitte's 2019 Global Blockchain Survey (2019), https://bit.ly/32E67SA
45. Mafakheri, B., Subramanya, T., Goratti, L., Riggio, R.: Blockchain-based Infrastructure Sharing in 5G Small Cell Networks. In: International Conference on Network and Service Management (CNSM 2018). pp. 313–317. Rome, Italy (November 2018)
46. Mazières, D., Shasha, D.: Building Secure File Systems out of Byzantine Storage. In: Annual Symposium on Principles of Distributed Computing (PODC 2002). pp. 108–117. Monterey, California (2002)
47. McMillan, R.: The Inside Story of Mt. Gox, Bitcoin's $460 Million Disaster (2014), https://bit.ly/3jqaiaz
48. Nakamoto, S.: Bitcoin: A Peer-to-Peer Electronic Cash System (2009), https://bitcoin.org/bitcoin.pdf
49. Nguyen, C.T., Hoang, D.T., Nguyen, D.N., Niyato, D., Nguyen, H.T., Dutkiewicz, E.: Proof-of-Stake Consensus Mechanisms for Future Blockchain Networks: Fundamentals, Applications and Opportunities. In: IEEE Access. vol. 7, pp. 85727–85745 (January 2019)
50. Niya, S.R., Maddaloni, F., Bocek, T., Stiller, B.: Toward Scalable Blockchains with Transaction Aggregation. In: ACM Symposium on Applied Computing (SAC 20). pp. 308–315. Brno, Czech Republic (2020)
51. Niya, S.R., Stiller, B.: BAZO: A Proof-of-Stake (PoS) Based Blockchain. Tech. rep., Zürich, Switzerland (May 2019), https://bit.ly/2G3odoh
52. Osterweil, E., Massey, D., Tsendjav, B., Zhang, B., Zhang, L.: Security Through Publicity. In: 1st USENIX Workshop on Hot Topics in Security (HOTSEC 2006). pp. 13–18. USENIX Association, Vancouver, Canada (August 2006)
53. Park, H., Craddock, A.: Diploma Mills: 9 Strategies for Tackling One of Higher Education's Most Wicked Problems (2017), https://bit.ly/2DoEeyu
54. Peck, M.E.: Bitcoin Hits $1 Billion (2013), https://bit.ly/39ab6vE
55. Popov, S.: The Tangle (2018), https://bit.ly/3bz3KSI
56. Ren, L., Ward, P.A.S.: Pooled Mining is Driving Blockchains Toward Centralized Systems. In: International Symposium on Reliable Distributed Systems Workshops (SRDSW 2019). pp. 43–48. Lyon, France (October 2019)
57. Rodrigues, B., Eisenring, L., Scheid, E., Bocek, T., Stiller, B.: Evaluating a Blockchain-based Cooperative Defense. In: IFIP/IEEE Symposium on Integrated Network and Service Management (IM 2019). pp. 533–538. Washington DC, USA (April 2019)

58. Rodrigues, B., Scheid, E.J., Stiller, B.: Blockchains in the Age of Softwarization – Hands-on Experiences with Programming Smart Contracts and Their Security Pitfalls (Tutorial 1). In: IFIP/IEEE Network Operations and Management Symposium (NOMS 2020). IEEE, Budapest, Hungary (April 2020)
59. Rodrigues, B., Bocek, T., Stiller, B.: The Use of Blockchains: Application-Driven Analysis of Applicability. In: Raj, P., Deka, G.C. (eds.) Advances in Computers - Blockchain Technology: Platforms, Tools and Use Cases. vol. 111, pp. 163–198. Elsevier (2018)
60. Rodrigues, B., Franco, M.F., Scheid, E.J., Kanhere, S.S., Stiller, B.: A Technology-driven Overview on Blockchain-based Academic Certificate Handling. In: Blockchain Technology Applications in Education. pp. 197–224. IGI Global (2020)
61. Ruoti, S., Kaiser, B., Yerukhimovich, A., Clark, J., Cunningham, R.: Blockchain Technology: What is it Good For? Communications of the ACM **63**(1), 46–53 (December 2019)
62. Scheid, E.J., Hegnauer, T., Rodrigues, B., Stiller, B.: Bifröst: a Modular Blockchain Interoperability API. In: IEEE Conference on Local Computer Networks (LCN 2019). pp. 332–339. Osnabrück, Germany (October 2019)
63. Scheid, E.J., Lakic, D., Rodrigues, B.B., Stiller, B.: PleBeuS: a Policy-based Blockchain Selection Framework. In: IEEE/IFIP Network Operations and Management Symposium (NOMS 2020). pp. 1–8. Budapest, Hungary (April 2020)
64. Scheid, E.J., Rodrigues, B.B., Granville, L.Z., Stiller, B.: Enabling Dynamic SLA Compensation Using Blockchain-based Smart Contracts. In: IFIP/IEEE Symposium on Integrated Network and Service Management (IM 2019). pp. 53–61. Washington DC, USA (April 2019)
65. Simplot-Ryl, I., Traore, I., Everaere, P.: Distributed Architectures for Electronic Cash Schemes: A Survey. In: International Journal of Parallel, Emergent and Distributed Systems. vol. 24, pp. 243–271. Taylor & Francis (June 2009)
66. Smart, N.P.: Cryptography Made Simple. Springer International Publishing (2016)
67. Smith, B.C., Leimkuhler, J.F., Darrow, R.M.: Yield Management at American Airlines. ACM Interfaces **22**(1), 8–31 (February 1992)
68. Sovrin Foundation: Control Your Digital Identity (2021), https://sovrin.org/
69. Stiller, B., Rafati, S., Grossenbacher, S.: Application of Blockchain Technology in the Swiss Food Value Chain (Foodchains Project Report) (June 2019), https://bit.ly/3hjxaGX
70. Suberg, W.: EOS Node Offers Users Financial Rewards for Votes, Reignites Decentralization Debate (2018), https://bit.ly/2WCrrEc
71. Szabo, N.: Formalizing and Securing Relationships on Public Networks. First Monday **2**(9) (September 1997)
72. The Linux Foundation: Hyperledger Wiki (2020), https://wiki.hyperledger.org/
73. Trautman, L.J.: E-Commerce, Cyber, and Electronic Payment System Risks: Lessons from PayPal. UC Davis Bus. LJ **16**, 261–307 (2015)
74. Wang, W., Hoang, D.T., Hu, P., Xiong, Z., Niyato, D., Wang, P., Wen, Y., Kim, D.I.: A Survey on Consensus Mechanisms and Mining Strategy Management in Blockchain Networks. In: IEEE Access. vol. 7, pp. 22328–22370 (January 2019)
75. Wood, G.: Ethereum: A Secure Decentralised Generalised Transaction Ledger (2020), https://ethereum.github.io/yellowpaper/paper.pdf
76. Wüst, K., Gervais, A.: Do you Need a Blockchain? In: Crypto Valley Conference on Blockchain Technology (CVCBT 2018). pp. 45–54. Zug, Switzerland (November 2018)

All links provided above were last accessed on April 8, 2021.