future internet

MDPI

Review

Distributed Ledger Technology Review and Decentralized Applications Development Guidelines

Claudia Antal, Tudor Cioara *, Ionut Anghel, Marcel Antal and Ioan Salomie

Computer Science Department, Technical University of Cluj-Napoca, Memorandumului 28, 400114 Cluj-Napoca, Romania; claudia.pop@cs.utcluj.ro (C.A.); ionut.anghel@cs.utcluj.ro (I.A.); marcel.antal@cs.utcluj.ro (M.A.); ioan.salomie@cs.utcluj.ro (I.S.)
* Correspondence: tudor.cioara@cs.utcluj.ro; Tel.: +40-264-202-352

Abstract: The Distributed Ledger Technology (DLT) provides an infrastructure for developing decentralized applications with no central authority for registering, sharing, and synchronizing transactions on digital assets. In the last years, it has drawn high interest from the academic community, technology developers, and startups mostly by the advent of its most popular type, blockchain technology. In this paper, we provide a comprehensive overview of DLT analyzing the challenges, provided solutions or alternatives, and their usage for developing decentralized applications. We define a three-tier based architecture for DLT applications to systematically classify the technology solutions described in over 100 papers and startup initiatives. Protocol and Network Tier contains solutions for digital assets registration, transactions, data structure, and privacy and business rules implementation and the creation of peer-to-peer networks, ledger replication, and consensus-based state validation. Scalability and Interoperability Tier solutions address the scalability and interoperability issues with a focus on blockchain technology, where they manifest most often, slowing down its large-scale adoption. The paper closes with a discussion on challenges and opportunities for developing decentralized applications by providing a multi-step guideline for decentralizing the design and implementation of traditional systems.

Keywords: distributed ledger technology; blockchain; decentralized applications; technology review; development guidelines; architecture

Citation: Antal, C.; Cioara, T.; Anghel, I.; Antal, M.; Salomie, I. Distributed Ledger Technology Review and Decentralized Applications Development Guidelines. *Future Internet* **2021**, *13*, 62. https://doi.org/10.3390/fi13030062

Academic Editors: Sk. Md. Mizanur Rahman and Ahad ZareRavasan

Received: 26 January 2021
Accepted: 24 February 2021
Published: 27 February 2021

1. Introduction

Distributed Ledger Technology (DLT) is a disruptive technology that provides an environment with no central authority for registering, sharing, and synchronizing transactions on digital assets. By joining several computer sciences disciplines such as distributed systems, cryptography, data structures, or consensus algorithms, it offers highly desirable features (decentralization, openness, immutability, transparency, traceability, security, availability, etc.). Gartner has included DLT technology in the hype cycle for the first time in 2016 at the phase of an innovation trigger [1] mostly due to the advent of its most popular type the blockchain technology. In 2017–2018, research has been committed to developing the mechanisms to accommodate the technology to requirements such as privacy, scalability, permissions, and interoperability which are essential to decentralized applications implementation perspective and emerging business models. By 2018 the DLT has passed the peak of inflated expectations, assuming to reach a plateau of productivity in the next 5 to 10 years [2], while in 2019 the innovation potential of the DLT is considered by Gartner to be driven not only by the technology expectations but also by the social ones [3].

In this sense, the development of decentralized autonomous organizations and implementation of decentralized applications and the decentralized web is of high interest for the next 10 years. Building such decentralized applications is not a straightforward process since many technological solutions have emerged, generating a confusing context with lots of challenges for the software industry [4].

The main contributions of this paper are a comprehensive overview of nowadays DLT solutions and a set of guidelines for decentralized application development. To streamline and organize the review we have defined and used a three-tier conceptual architecture [5,6] (see Figure 1). Each tier aggregates alternative DLT solutions to address specific issues in the implementation of decentralized applications:

- The Protocol and Network Tier (PN-Tier) aggregates the core DLT elements and organizes them in two layers. The Protocol Layer contains technology solutions for digital assets registration, transactions, data structures, privacy, and business rules implementation. The Network Layer contains technology solutions for creating a peer-to-peer network, ledger replication, and consensus-based validation.
- The Scalability Tier (S-Tier) runs most of the time a parallel DLT network and aggregates technological solutions for addressing the scalability issues raised by the PN-Tier. We have focused on solutions for blockchain ledger scalability problems such as storage scalability, transaction throughput, and computational scalability.
- The Interoperability Tier built on top of the previous two tiers addresses integration and interoperability of multiple DLT applications and systems deployments.

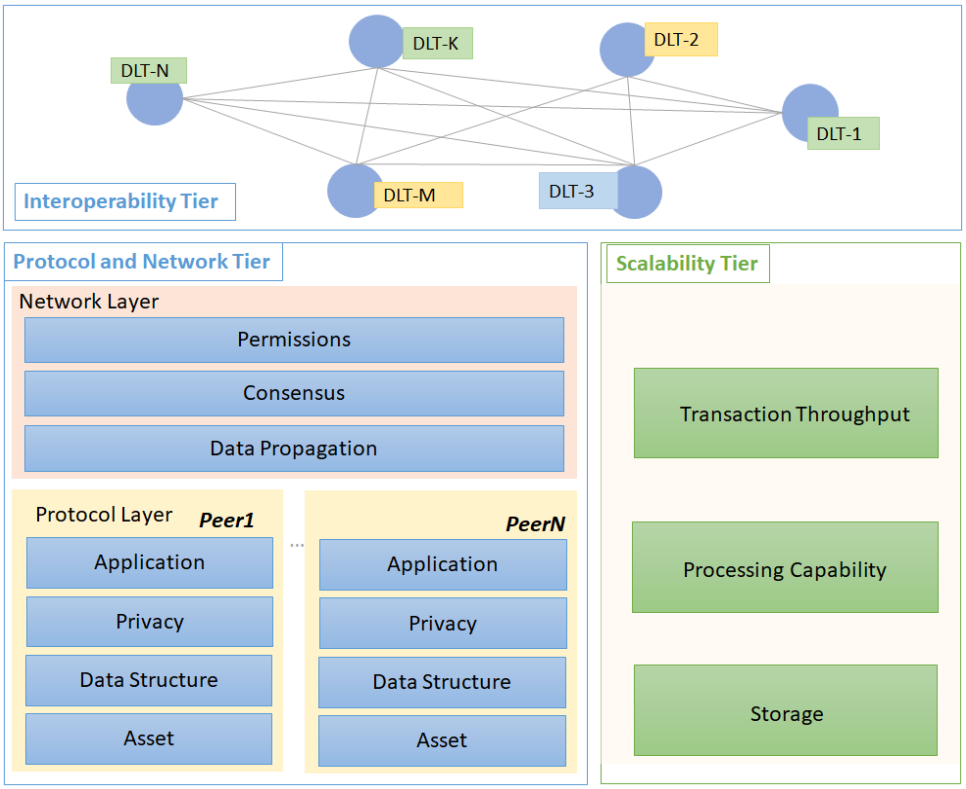

Figure 1. Three-tier architecture for decentralized applications development.

For each architectural tier, we have identified the main technological challenges and used them as criteria of including relevant solutions, and technologies aiming to create a consistent overview of the current technological state of the art. In the case of Scalability and Interoperability Tiers, the focus is mostly on blockchain technology. The reason for this is the higher level of maturity the applications developed using blockchain are often facing problems related to the transaction throughput, storage space, gas consumption, or interoperability with other chains.

Finally, we provide a guideline for building decentralized applications discussing the main steps and the technologies selection concerning the challenges that need to be addressed.

Most distributed ledger technology reviews found in the literature are focusing almost exclusively on the blockchain ledgers [7,8] and do not consider the other DLT variations such as Directed Acyclic Graphs (DAG) [9] and other combinations or hybrid solutions [10]. In our review, we have addressed these DLT variations in the Protocol and Network Tier. Even though in the Scalability and Interoperability Tiers we focus on the blockchain ledgers, being the more mature technology, some of the solutions and patterns referenced can be adopted by other DLT variations. At the same time, our review was driven by a decentralized application architecture that has been used and validated in previous publications. We have successfully used it for the implementation of decentralized applications in the domains of smart energy grid (demand response [6], peer to peer energy trading [11], flexibility management [12], etc.), stock exchange [13], or vaccine distribution [14]. As result, the criteria for selecting the technologies and organizing the review are strictly related to the development issues that may be encountered and need to be addressed in each tier. Finally, the guideline for decentralized application development provided on top of the reviewed literature can drive the selection of various DLT alternatives based on the issue encountered helping the community orienting in such an effervescent and confusing technological context. We could not find a similar guideline in the reviewed literature.

The rest of the paper is structured as follows: Section 2 presents the technological solutions for the PN-Tier, Section 3 reviews the mechanisms of S-Tier for improving the PN-Tier's scalability in three directions: storage, transaction throughput, and computational, Section 4 describes the interoperability solutions among federated blockchain ledgers, Section 5 discusses guidelines for developing decentralized applications, while Section 6 concludes the paper.

2. Protocol and Network Tier

Different solutions for PN-Tier have been proposed addressing specific challenges [4,15–19]. We have identified the technological components that are grouped at the protocol and network layers. The security of the entire tier is ensured as a result of integrating the public-private key cryptography for locking and unlocking transactions, with the tamper-resistant data structures and consensus algorithms.

2.1. Protocol Layer

The protocol layer represents the core of the technology that runs on each full node in the peer-to-peer network. It is governed by rules that specify what, when, how, and by whom the assets are operated on the chain and features four types of technological components: asset representation, data structure, privacy, and business rules enforcement.

2.1.1. Type of Asset and Data Structures

The tokenization process refers to the possibility of modeling different goods in a DLT system as digital assets that can be issued and transferred according to a predefined set of rules. The common terminology for an asset representation in DLT systems is Token or Coin. We have identified two types of tokens that can be represented in the system (see Table 1): native tokens and tokens based on real assets (asset-based tokens). Native tokens are tokens defined in the DLT system, completely independent of the real world, thus the rules governing the issuance and the transfer are completely defined in the system (through Initial Coin Offerings or mining reward schemes), and do not rely on any third trusted party. Bitcoin [4], Ether [15], EtherTulips [20], Grid [21], Rarible [22], CryptoKitties [23], NRGcoin [24] or Telcoin [25] are examples of such tokens that are completely virtual and have economic value based on supply and demand.

Table 1. Type of digital assets modeled using blockchain.

Type	Token Example	Fungibility	Issuers
Native tokens	Bitcoin, Ether, CryptoKitties [4,15,23]	Yes	Mining Reward Schemes
	ERC20, ERC223, ERC-621 [26]	Yes	Initial Coin Offerings
	ERC721 [26]	No	Initial Coin Offerings
Asset-Based Tokens	Real Estate [27]	No	Government Land Registries [28]
	Patents [29]	No	U.S. Patent & Trademark Office [30]
	Academic Records [31]	No	The Registrar's Office [32]
	Gold	Yes	Royal Mint Gold [33]

Real-life assets can also be represented through tokens in DLT systems, offering the opportunity to represent, transfer and track them. By using asset-based tokens, the chain can keep track of different kinds of assets, both tangible (real estate, cars, money, art, etc.) and intangible (patents, trademarks, copyrights, etc.). The DLT systems representing real-life assets must rely on a trusted third party to issue each token concerning the real object. Similarly, a transfer on-chain must be done under the governance of such a trusted party since any issue regarding a wrongful transfer on the chain can be verified only by communicating with the external systems.

In DLT systems, the real-life flow of assets is represented in the network as transactions between peers. The DLT requires specialized data structures at its core that can ensure three properties over the stored transactions: provenance, asset ownership validation, and immutability. Hash pointers have been frequently integrated with different data structures intended for DLT usages. Due to the hash functions' collision-resistant, data concealing, and data binding properties, the hash pointers are the best choice for adapting common data structures to the DLT requirements thus obtaining: linked list with hash pointers (e.g., blockchain), binary trees with hash pointers (e.g., Merkle Trees), graphs with hash pointers (e.g., Hash Graphs), etc.

Two main directions identified are related to distributed ledger data structures for representing peers' transactions, namely blockchain and direct acyclic graph (DAG).

The blockchain structure, as its name suggests, is a chain formed by linked back blocks, also known as Linked List using hash pointers (see Figure 2a). Each block contains all the transactions that occurred in the system in a short period (e.g., ~10 min for Bitcoin, or ~12 s for Ethereum). All the transactions contained in the block are hashed together in a Merkle Tree data structure, where the root of the tree is referenced in the block header and acts as a digital fingerprint of the entire collection. Thus, blockchain becomes an append-only data structure that gathers all the benefits of the hashing and cryptographic functions and, with the integration of a consensus algorithm, ensures an immutable history log of the entire activity of the network. The blockchain structure is implemented in well-known solutions such as Bitcoin [4], Ethereum [15], Litecoin [17], Hyperledger [34], CryptoNode [35], Ripple [36], and Zerocash [19].

A less popular data structure is DAG firstly mentioned in [37]. Since its proposal, several platforms have been developing solutions based on DAG variations: Dagcoin [38], IOTA [16], HashGraph [39], or hybrid systems like Holochain [40], and Flowchain [41]. Among the DAG-based systems, IOTA is the most used solution. It uses a DAG, called tangle, as a ledger for storing the transactions as depicted in Figure 2b. The entire graph starts with a genesis transaction that is approved directly or indirectly by all the transactions in the graph. Whenever a new transaction is submitted, it must validate and confirm two previous transactions from the graph that were not yet approved (i.e., tips). The tips selection algorithm is based on the family of Markov Chain Monte Carlo algorithms and it considers the cumulative weights of sub-tangles. Whenever a situation of conflicting transactions appears, the higher the cumulative weight of the transaction is, the more secure it is.

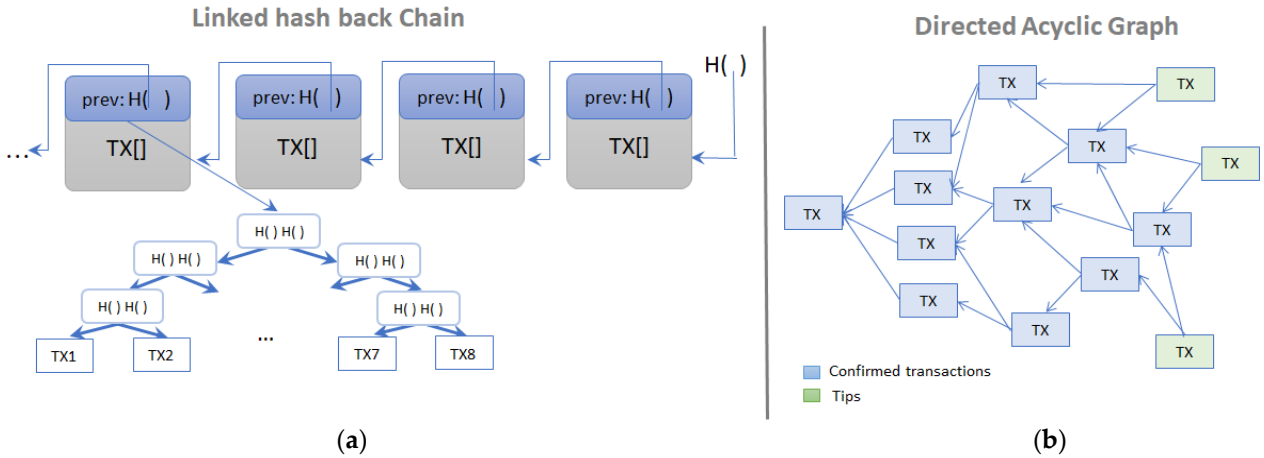

Figure 2. Data for representing peers' transactions structures: (**a**) blockchain using linked lists and (**b**) Directed Acyclic Graphs (DAG).

While the most successful solutions and research directions are focused on blockchain-based DLT, the DAG solutions have yet to overcome considerable shortcomings in terms of centralization and scalability to be considered suitable alternatives to blockchain-based DLTs [42].

2.1.2. Privacy of Transacting Parties and Data

Most of the popular DLT solutions preach the properties of transparency and openness to be major benefits brought by the technology. By providing open information about the transacting parties (pseudo-anonymous most of the time) and transacted assets, the systems offer clear history and audit advantages. However, in many use cases where the privacy of data is highly required [43] (e.g., medical use cases [44,45]), many of the DLT solutions may prove to be unsuitable due to their transparency and openness. With the emerging General Data Protection Regulation (GDPR) restrictions, the privacy of data is one of the most controversial subjects in the DLT systems. In this direction, solutions have emerged that aim to hide the details regarding the transaction information, while at the same time keeping the reliability of the system by allowing consistency checks, validation, and audit regarding the previous actions and history [46].

In Bitcoin versions of the DLT, the value of the transacted asset is clearly stated, and the nodes can easily check the ownership over the asset by checking the history and identifying the unspent transaction outputs (UTXO). The biggest challenge is that by hiding the transaction data, the nodes in the system should still be able to validate the availability of the funds and the ownership of the sender over that asset. Coin Mixers have been developed over Bitcoin to hide information and prevent tracking and tracing of transacted assets. One of these approaches is CoinJoin [47] which aims to aggregate multiple users to agree upon several inputs and outputs and then sign the transactions. This makes it harder for the coins to be traced across the transactions since the group of inputs provided to a transaction will not belong to the actual sender. Similarly, Dash [48] aims to provide mixing services but instead of using a central point that is responsible for mixing inputs and outputs like CoinJoin, it provides a second layer of master nodes over Bitcoin.

A private-public key cryptographic mechanism is proposed by Quorum [49] to hide the transacted value. The encrypted transactions are shared point-to-point only to the involved parties, and a private state is defined by the Quorum node, where these encrypted transactions are stored. The encryption keys are shared between the involved parties, to validate the transaction. However, the shared blockchain (public state) only contains the hash of the encrypted transaction. This leads to a shortcoming of the system since the network cannot verify the validity of the private transactions, this being done only by the

involved parties. The shortcoming of Quorum is overcome by integrating Zero-Knowledge proof mechanisms [50] with DLT systems.

Zero-knowledge proofs have been added to digital currencies, such as Zerocoin [19,51] to create anonymous Bitcoin transactions without the need of third parties such as CoinJoin and Dash. The Zero-Knowledge proof mechanisms, mainly succinct Non-Interactive Zero-Knowledge proofs (ZkSnarks) [52] are used to ensure the transactions' validation without revealing actual information about the parties involved. The Zero-Knowledge proofs require that a Verifier could easily check that the Prover owns a secret, without revealing the actual secret to the Verifier. Consider a simple scenario, where the network owns a hash value H. An actor wants to prove to the network that he holds the secret s, that hashed offers the value of H. Normally, the actor would need to reveal the secret to the network. ZkSnarks aim to enhance this model, by allowing the actor to prove ownership over the data without disclosing any information regarding the secret. In this sense, ZkSnarks introduces a proving function, that can be used by the actor to issue a proof showing that the private secret is indeed corresponding to the public information H and a verification function that can be executed by any participant in the network to validate whether the proof corresponds to the public information H. ZCash [19] implements a ZkSnarks mechanism as an improvement of the Bitcoin system that ensures the privacy of the transactions. It requires any transaction to be locked by a secret, called a commitment, and unlocked by a participant that holds the secret and who can generate the relevant proof, called the nullifier. The commitment is issued off-chain by the sender of the transaction, having as secret information the value and the receiver's public key. Similarly, the nullifier is computed off-chain by the receiver, proving that he owns the necessary information to spend the locked value. The commitment and the nullifier are registered on-chain. The commitments are registered in the commitment tree, and each time a transfer is required, a nullifier for one of these commitments needs to be issued and then registered in the nullifier set as future proof for avoiding double-spending attacks. The verifiers of the nullifier are all mining nodes that need to validate the integrity of the transactions.

Homomorphic encryption is another approach that aims to improve the privacy of blockchain solutions in MimbleWimble [53], a system designed to provide an untraceable version of Bitcoin. It is a side chain that takes advantage of cryptographic properties to hide transacted values. In Bitcoin, each transaction is represented by an input amount that has an associated past UTXO that it unlocks, and an output amount locked by the receiver's key. The restriction imposed by the Bitcoin system is that the output amount of a transaction should never exceed the input amount of it. MimbleWimble uses Elliptic Curve Cryptography and Homomorphic encryption, leveraging on the fact the computations applied on the cyphertexts offer the same results as if applied on the plaintext. Therefore, in MimbleWimble the transacted amounts are blinded and by applying computations on the obtained cyphertexts they are further involved in mathematical operations proving that the input values of the transaction equal with the output values, obtaining the same results as they would have been performed directly in plaintext.

Another commonly used strategy, that hides the transferred asset amount of a transaction and the parties involved, is the ring signature. CryptoNote [54] is one of the first protocols that proposes the use of ring signatures for issuing transfers by specifying a group of possible signers to avoid the possibility of discovering the exact sender of the money. Considering a group of N parties (sub-group of network participants) involved in the ring, one of the parties can sign the message, resulting in a signature that can be verified by anyone in the network, but without the possibility of detecting the exact signing party. Furthermore, in Monero [18], the authors use Stealth addresses for the receiving parties. Each Monero account is composed of two private keys (view and spend key) and the public address. As their name suggests, the view key is used to track all the transactions that were published and are destined for that account. The spend key is used to send transactions and the public address is the one used by a sender to compute the one-time public key (stealth address), unique for each transaction. The stealth addresses offer non-linkable

transactions, which means that the outputs are not associated with the addresses of the wallets. However, this solution does not provide complete privacy since the sender can trace when the money is spent by the receiver. A completely private system would need to offer both non-linkable and non-traceable transactions.

Table 2 presents comparatively the main privacy-preserving techniques identified for DLT and the privacy features they are offering. The coin mixers do not offer complete privacy, although they offer non-traceability mechanisms, the actual values transmitted are still visible. In terms of non-traceability however, the Zero-Knowledge Proofs were not yet validated as completely untraceable, due to the complexity of the algorithms involved. ZCash aims to provide traceable capabilities for the system, to be able to detect malicious users, which leads to the conclusion the transactions may not be completely untraceable [19]. Private-Public Key cryptography and Coin mixers also have one main disadvantage, because they rely on central authorities to provide Key sharing services and mixing services respectively. Another important property is the advanced scripting capability, where Coin mixers and Homomorphic Encryption mechanisms prove not to be a good solution, while the Zero-Knowledge Proof solution although feasible, has the drawback of using high computational resources for applying the cryptographic algorithms.

Table 2. DLT main privacy presenting techniques.

Features	Private-Public Key Encryption	Zero-Knowledge Proofs	Ring Signatures	Homomorphic Encryption	Coin Mixers
Hidden Data	yes	yes	yes	yes	no
Non-traceable	yes	n/a	yes	yes	yes
Non-linkable	no	yes	yes	no	yes
Decentralized	no	yes	yes	yes	no
Private Business Enforcement	no	yes	no	no	no
Transaction validation by network	no	yes	yes	yes	yes

2.1.3. Business Rules Enforcement over Transactions

The capability of a DLT system to support business implementation that can be run in a decentralized way, and then be verified and audited by all the nodes in the system, is usually provided through smart contracts. Opposed to the concept suggested by their names, the smart contracts are not very smart and may not provide a contract in the legal sense, but rather they are pieces of code similar to the stored procedures that are executed and validated by the nodes in the system, whenever they are triggered. However, there are DLT systems such as Bitcoin that are offering few possibilities for scripts to be implemented. Not being a Turing Complete language, the Bitcoin script language does not permit the implementation of complex business logic required for more advanced use cases. As a result, new DLT systems have emerged that allow customizing decentralized applications and enforcement of business logic in a decentralized way regarding when, how, and by whom may an asset transfer be executed.

In Table 3 a comparison of the main state of the art approaches for enforcing the business rules on DLT is presented. There are two types of approaches that allow smart functionality to be implemented and run across the nodes of the network: stateless and stateful [55]. The Stateless implementation is offering the possibility to implement custom logic at the level of transactions. Whenever a transaction is issued there is a set of rules that can be verified before rendering the transaction valid. The Stateful systems, on the other hand, offer Business-oriented functionalities, by focusing on the rules that govern the use case, and keeping the state of the business in tamper-resistant structures (Patricia Merkle Trees, adapted from [56]) that are easily verifiable and audited by the entire distributed system. In the Stateful Systems, the transaction has the role of triggering changes and applying updates on the stored state.

Table 3. Business rules enforcement on DLT.

Type	Implementation	Platform	Enforcement Flexibility	Costs	Exploitation Risks
		Transactional Rules			
Stateless Transaction Oriented	Built-In Enforcement	Bitcoin [4], Litecoin [17]	Limited	None	Low
		Nxt [57]	Templates	None	Low
	Piggy Backed Enforcement	Counterparty [58]	Turing Complete	Fee per instruction	High
		State Storage			
Stateful Business Oriented	Smart Contracts & Merkle Patricia Tree	Ethereum [15]	Turing Complete	Fee per instruction	High
	Smart Contracts & NoSQL DB	HyperLedger [34]	Turing Complete	None	High

In terms of functional complexity, some systems allow full computation capabilities by supporting Turing Complete languages for smart contract implementations or partial capabilities by offering a limited range of operations or fixed predefined templates. Both approaches have their advantages. On one hand, full capabilities are desired to be able to model and enforce any complex business system. Turing Completeness allows this, but it requires higher transactional costs. The on-chain computation demands for each node to execute possibly complex scripts; thus, the costs are proportional with the number of instructions. Furthermore, a complex and flexible language makes the system susceptible to different kinds of attacks that can be caused by exploiting different language shortcomings or human errors during the business logic implementations like call depth attack, race conditions, timestamp dependency, transaction ordering dependency, etc. On the other hand, the risk of exploits is highly reduced by limiting the operations allowed or by providing predefined templates.

2.2. Network Layer

The network layer technologies are related to the peer-to-peer network formed by the nodes that hold copies of the ledger (full nodes or light nodes) and participate as active players in the network. The main technological components identified at this layer are targeting the data propagation among the nodes, the peer's registration and network permission, and the consensus among peers.

2.2.1. Data Propagation and Replication

In terms of transaction data propagation, the first generation of DLT systems (Bitcoin [4], Litecoin [17], Ethereum [15], etc.) relied on full-discovery or global disclosure. This is one of the strongest features of blockchain systems since a complete replication of the data offers high availability and reliability. However, there are use cases (e.g., banking, enterprise data) that impose restrictions regarding access to transaction information [59]. Two categories of systems have been identified based on how the transactions are propagated in the system. Firstly, the global disclosure mechanism, implemented by the systems where all the full nodes have access to all the transactions published in the system, and secondly, the selective disclosure mechanism where nodes have access only to exclusive transactions that are targeting either specific businesses or only the involved parties.

Most of the blockchain ledgers adopt a global disclosure approach to offer high reliability in an open system where any node can join. The entire system is a peer-to-peer network, where all the nodes are equal. Whenever a new event is issued (a new transaction, a new block) the data is propagated through the entire network, and each node can verify and validate the integrity of the data. The redundancy in storage and computation makes it very difficult for a malicious node to influence the system to its advantage. To attack (e.g., double-spending attack) on a globally disclosed DLT, an elaborate plan must be conducted by the malicious node. It must analyze the network topology (network segmentation) and issue contradictory actions for each half of the network, with the purpose of convincing half of the network to agree with the malicious action taken.

Having a global disclosure between all the peers in the network has obvious advantages since such a system benefits from the high replication and availability brought by a

large number of nodes, as well as Byzantine Fault Tolerant consensus between these nodes regarding the data. However, some clients/businesses prefer having more privacy and control over their data. This property is especially desired in private and consortium chains (e.g., banking systems), where the transactions are required to be shared only between the transacting parties. Although such a paradigm shift may lead to lower reliability in the system, the risks are highly attenuated if these requirements are implemented in permissioned systems where each stakeholder has its identity known and can be held accountable for his actions.

One of the selective disclosure approaches is presented in the Hyperledger Multichannel Architecture [60]. The system relies on third-party entities, called Orderers, which are required to order the transactions and publish them according to the category (business specific) in a corresponding channel. A Byzantine fault-tolerant consensus protocol is implemented between the Orderers, to ensure consistency between the decisions. A channel is a business-specific queue that broadcasts all the transactions to the subscribed parties. All the subscribers (peers) will receive the transactions in the same order in cryptographically linked blocks. A peer can be subscribed to more than one chain, but the chains do not interact with each other and each block received will contain only transactions corresponding to the corresponding business. Quorum [49] is another approach that aims to improve security by keeping the exclusive transactions shared only between the involved parties. The system is a hybrid between the global and selective disclosure paradigms, by allowing public transactions to be fully replicated and exclusive transactions to be shared only across the parties. The Quorum's privacy engine defines a private state tree that is updated with contracts and transactions that are sent point-to-point only to the interested parties. The private transaction contents are encrypted using Public Key cryptography, and only the users holding the private keys have access and can decrypt the actual content of the transaction. Proof of these events is also registered in the public chain, by hashing the encrypted private transaction. A similar permissioned implementation is also designed in Corda [61] where the network is formed of permission services, notary services, and peers. The system aims to provide redundancy while also keeping the transactions only known to the involving parts. Any transaction that occurs in the system must be signed and approved by both participants, and by the notary service responsible to validate transactions and prevent double-spending events. The notary service can be one entity or multiple entities that are coordinated by a consensus algorithm.

In Table 4, the comparison between the Data propagation patterns found in the literature is presented. One of the biggest disadvantages of the current selective disclosure systems is their trust in different central authorities. The Quorum system requires some level of trust between the private parties, and the other systems rely on central authorities that are responsible either for forwarding the messages like in the case of the Hyperledger MultiChannel system or on authorities responsible to validate the integrity of transactions like in the case of Corda [61] or Plasma [62]. Consequently, selective disclosure should be considered only in trusted environments, where the central authorities can be considered a source of truth, while for public environments, global disclosure should be considered such that any party involved in the network can validate the integrity of the transactions.

Table 4. DLT data propagation patterns.

Type	Platform	Trusted Parties	Global Disclosed		Selective Disclosed	
			Data	Structure	Data	Structure
Public DLTs	Ethereum [15], Bitcoin [4], etc.	-	All transactions	Blockchain	-	-
Business Specific Chains	HyperLedger Multi-channel [60]	Orderer	-	-	Exclusive transactions	Queues, Blockchain
	Plasma [62]	Central Authority, N delegates	Public Transactions + settlements	Blockchain	Exclusive transactions	Blockchain
Point-to-Point transactions	Corda [61]	Notary Service	-	-	Exclusive transactions	Local database
	Quorum [49]	Private parties	Public transactions, Hashes of Exclusive Transactions	Blockchain	Exclusive transactions	Merkle Patricia Tree

2.2.2. Permission Mechanisms

Over the years the private institutions that realized the potential of the systems behind DLT, started to evaluate the integration of such systems with their businesses. However, some key components rendered the public chain unsuitable for many institutional and enterprise solution requirements, so they started to investigate new systems that address the issues regarding the governance and the permissions of the system. Firstly, in an enterprise solution, the participants need to be known and vetted before given access. Such a decision has a great impact on the system, even in terms of security and consensus. Since the participants are known, thus can be held accountable for their actions, the need for a high energy-consuming algorithm like Proof-of-Work is no longer justified. Therefore, there is a strong relationship between the requirements regarding access rights and the consensus algorithms suitable for a specific business.

The difference between public, private, permissionless, and permissioned DLT/blockchain is given mainly by the rights of the users in the system. Based on the classification presented in Table 5, the difference between private and public chains is established according to the target audience that has access (reading rights) to the chain. Restricting the access of a group to the chain renders the chain private. According to the group of people accessing the chain, it can be a consortium or an enterprise solution, where the consortium solution operates under the leadership of a group of companies, and the enterprise solution is under the operation of a single entity.

Table 5. Public vs Private Blockchain permissions.

Action	Public Chain		Private Chain	
	Permission-Less	**Permissioned**	**Consortium**	**Enterprise**
Chain Access	Everyone	Everyone	Group Owner	Group Owner
Transactions	Everyone	Owners & Validated Users	Owners & Validated Users	Administrator
Commit to chain	Everyone	Owners & subset of Validated Users	Owners & subset of Validated Users	Administrator

In public DLTs, some restrictions can also be imposed regarding the users' access and permissions. In a permissioned ecosystem, the validators are known and accountable for their actions, thus a certain level of trust between the nodes can be considered. In a permission-less system, on the other hand, any user can perform any type of action (transactions of an asset, as well as commits of new blocks to the chain). Consequently, permission-less DLTs require Byzantine Fault Tolerant consensus algorithms, since the openness of the system allows even malicious nodes to join, making the network susceptible to a larger range of attacks.

2.2.3. Consensus Protocols

Figure 3 presents a taxonomy of the consensus algorithms which are classified in Non-Byzantine fault-tolerant algorithms and Byzantine fault-tolerant algorithms. The difference is given by the ability of algorithms to reach an agreement, integrity, and termination in case of existing faulty or attacker nodes in the distributed system, thus Non-Byzantine fault-tolerant ones rely on the assumption that all the nodes are fair, while the Byzantine fault-tolerant algorithms can handle situations when the number of malicious nodes is as high as half of the total number of nodes.

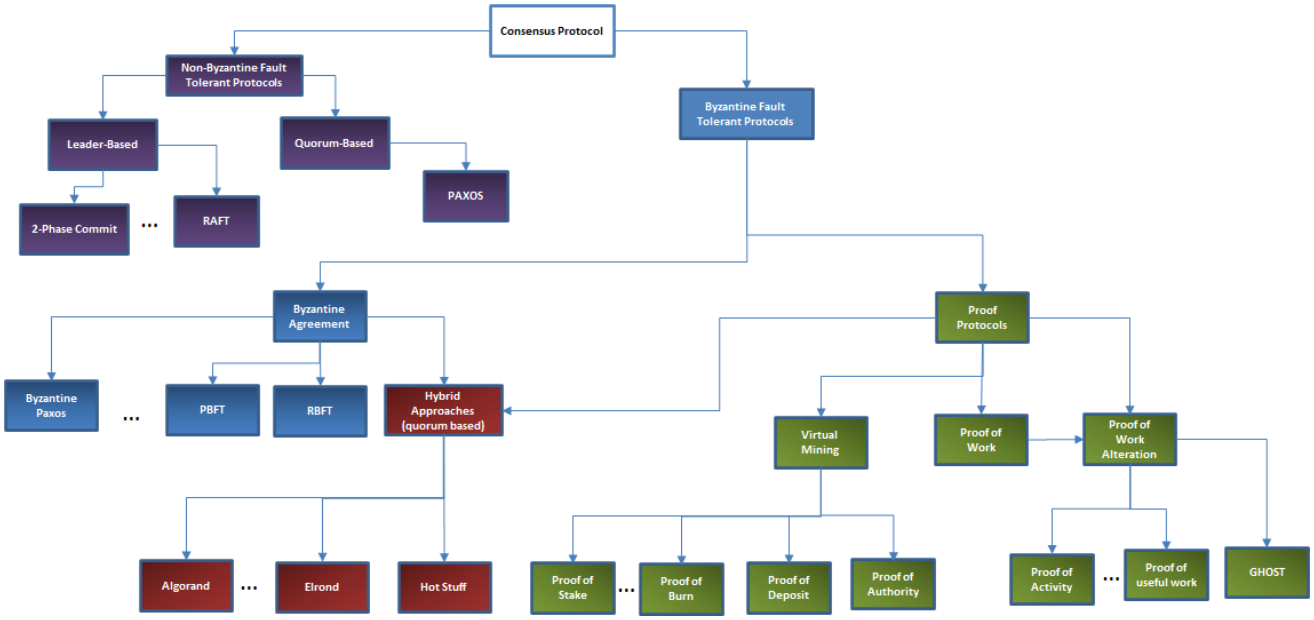

Figure 3. Consensus algorithms taxonomy.

The *Non-Byzantine fault-tolerant protocols* are leader-based, such as 2-Phase Commit [63] and RAFT [64], where a leader election algorithm is used to select a leader that will centralize the votes and commit the transaction. Furthermore, they are quorum-based, where a subset of the processes is selected to validate the transaction using a voting scheme. A well-known algorithm of this class is the Paxos algorithm [65] that solves consensus in a network of processes that may fail but are correct (there exist no faulty processes that may lie). In Quorum, RAFT algorithm is used, where a predetermined leader is creating a block that is sent to each node in the cluster [66].

The *Byzantine fault-tolerant protocols* aim to assure that the peers can agree on a system valid state even in case some of them feature faulty or malicious behaviors. The idea is to find a model and protocol for a network of message-passing processes, some of them being faulty, such that a general agreed state can be extracted from the distributed system. The Byzantine fault-tolerant protocols can be classified as Byzantine Agreement protocols and Proof Protocols [67]. In terms of finality, proof-protocols are known not to be final, however, they offer probabilistic finality, since once many blocks are sealed over, the probability of a block's state to change is very low.

The *Byzantine Agreement (BA)* protocols use a quorum-based mechanism where a subset of the nodes must agree on a transaction validity. Examples of such algorithms are the Byzantine Paxos algorithm [68], the Practical byzantine fault tolerance algorithm [69], and variants that address the robustness such as Ardvark [70] and RBFT [71] or that address the performance problems of PBFT, such as Q/U [72], HQ [73], Zyzzyva [74] and ABsTRACTs [75]. An interesting Byzantine fault tolerant distributed commit protocol is proposed in [76], where the authors enhance the classical 2-Phase Commit protocol by replicating the coordinator to successfully terminate when the coordinator failed and by building a quorum of coordinators to validate transactions and identify malicious participants.

The *Proof Protocols (PP)* are used by most of the public DLT systems [77,78] for supporting the consensus mechanisms to ensure the consistency of the ledger state across the network nodes. The Proof Protocols have defined two categories of nodes: Provers and Verifiers, where the Prover who may have unlimited resources needed to convince Verifier nodes with limited resources, about the truthfulness of a statement. As opposed to BA protocols, which use a quorum of participants to validate a transaction by voting, PP such as Proof of Work (PoW) and various alterations algorithms validate a transaction

(or a set of transactions) by solving a computationally-intensive problem by a Prover that requires a lot of physical resources and makes infeasible for an attacker to cast an erroneous vote. The time needed by the prover to solve the computationally intensive problem gives the mining rate and directly influences the throughput (number of transactions) and the network scalability. From the initial implementation of PoW in Bitcoin, where one block was generated every 10 min, PoW variations have been proposed aimed at improving the mining rate to obtain a higher throughput of transactions per second. The Greedy Heaviest Observed Subtree (GHOST) protocol [79] proposed by Ethereum increases the mining rate from 1 block per 10 min to 1 block per ~15 s. To avoid the potential problems that may arise due to delayed propagation of blocks, GHOST uses references to orphan blocks or uncles (valid blocks that were not accepted in the main chain due to network delays) to increase the weight of the longest chain. In this sense, each new block can contain references to previous uncles and for each of the referenced uncles, the miner will receive a small incentive, consequently, the miner of the uncle will also be rewarded when a new block refers to it. This mechanism discourages the faulty miners to mine on forked chains and from perusing long-range attacks. Other variations of PoW have been considered to impose some restrictions on the hardware devices used for mining by encouraging the implementation of ASIC (Application Specific Integrated Circuits) resistant algorithms for hashing. This came because of Bitcoin's early years when hardware companies started to profit from the popularity of blockchain solutions by developing ASICs to increase the hash rate of the computing nodes. However, one such circuit may cost around 3000 dollars [80], which makes it unprofitable for a simple user to invest in such hardware and gives more power and control to large companies and the manufacturer. To avoid this problem, the next generation of DLT solutions researched and applied new hash functions that are ASIC resistant. ASIC resistant algorithms try to shift their strategy from CPU intensive algorithms to memory intensive algorithms, called Memory hard puzzles. This came because the performance of processors has increased over time at an exponential rate, as opposed to the memory which has known a more linear increase. The purpose of these algorithms is to design a method that requires large amounts of data to be stored, that cannot be efficiently parallelized. Scrypt [81] is one of the first ASIC resistant algorithms and is currently widely used by many applications. However, Litecoin, which is one of the top platforms that use this algorithm set the memory size at 128 KB [82] thus making it possible to be stored at the CPU cache level. This restriction was applied since the Scrypt algorithm requires the same resources for solution verification as for the solution discovery and higher requirements would stress too much the regular non-mining nodes. Dagger Hashimoto [83] on the other hand, is an algorithm that provides an easy verification solution, thus allowing the Prover's requirements in memory size to increase up to 1 GB RAM. Equihash is also a widely used hashing algorithm. However, the main disadvantage, as the authors themselves state [84], is that the algorithm is parallelizable, which is not a quality desired in ASIC resistant algorithm. Finally, the Cuckoo hash cycles [85], used in [86,87], are also considered a reasonable solution when talking about ASIC resistance. Other relevant variations of PoW algorithms aim at giving a purpose for all the energy and computational resources of the network [88]. Since the network uses large computational resources whose only purpose is to prove and validate the next block of the blockchain, the concept of Proof of Useful Work is launched as an alternative to trying to use the computational power for a publicly beneficial domain. Such implementations aim to do research work (or Proof-of-Research). They gather the computational power across the network to provide solutions to some of the world's problems. CureCoin [89] is implementing an algorithm called SigmaX that aims to perform protein unfolding to find a cure for different diseases. Proof of Activity [90] is a PoW alteration algorithm found in Decred [91]. The algorithm starts as a simple PoW algorithm until one correct hash is found; the block is then transmitted in the network, but it is not yet added to the blockchain. To become a valid block, it needs to be signed by N holders in the network. The PoW obtained hash is used to generate N numbers that correspond to N coins generated since the genesis of the blockchain. Each of these coins

has one current stakeholder who will be required to sign the current block. The signature of all the N stakeholders is required to consider the block valid. In case that some of the stakeholders are not online and cannot sign, then the miners will continue their job to find a new hash and ask other stakeholders to sign the block. This approach makes attacks upon the network more difficult since it makes use of the advantages brought by both systems.

Virtual Mining Protocols offer an alternative to the PoW by keeping a high cost for the Prover, but changing the resource consumed. If the cost of the Prover in PoW is the energy consumed, which would be lost if the Prover does not offer honest work to be validated and rewarded by the network, in the virtual mining Protocols the cost is a deposit of coins that are offered as insurance for their honest work. If up until now the node was chosen based on its result to the computationally intensive problem, now the node will be elected in a pseudo-random way, and the chance of winning will be proportional to the number of coins/stakes of the owner of the system. Thus, in Virtual Mining Protocols, the clients have the mining potential proportional to the percentage of the stake they hold. Four virtual mining approaches have been identified across different solutions: Proof of Stake considers the age of the coin in the algorithm, thus requiring for some coins not to be spent for some time; Proof of Burn requires a relevant amount of coins to be destroyed and a proof of the destroying transaction to be provided; Proof of Deposit requires for some coins to be put away for some time in a vault; Proof of Authority suggests that only trusted parties are entitled to provide commits to the system, which can be required where high-security properties need to be implemented [92], like in the case of private Enterprise solutions. However, all four algorithms have the same purpose that is, incentivizing the honest work of the miner by promising as a reward a sum of coins greater than the initial insurance. According to [93], the Casper version of Proof-of-Stake (PoS) is considered a suitable alternative for the permissioned systems, by considering only a fixed set of users as validators of blocks. Another flavor of Proof-of-Stake commonly used for permissioned systems is the Delegated Proof of Stake (DPoS). In DPoS, N witnesses are periodically selected by stakeholders of the system, such that enough decentralization is ensured. Out of the N witnesses, each witness has its chance to propose the next block, and then be rewarded for its contribution. From existing Virtual Mining Protocols, the PoS a good potential of becoming the most used consensus protocol in DLTs because it addresses fundamental problems of the PoW protocol such as computational waste and high-power demand [94]. Anyway, in the case of the PoS algorithm, since the nodes propose a new block by guaranteeing with their stake it gives rise to the "nothing-at-stake" vulnerability. This means that when a fork appears in the context of a network partitioning, an attacker node can propose a block on either chain, hoping that at least one block will be accepted. The node guarantees each proposed block with its stake, but due to network partitioning, it is difficult for other nodes to observe and penalize this misbehavior. This situation can lead to other forks or to the fact that the attacker node receives rewards for proposing new blocks. In PoW algorithms, the "nothing-at-stake" vulnerability is avoided since when proposing a new block, the node has to solve a computational puzzle that consumes electrical energy, and by proposing two blocks on two chains from a fork means that the node has to solve twice the problem, thus doubling its costs. There are two categories of PoS mechanism: (i) chain-based PoS that mimics PoW by assigning pseudo-randomly the right to generate new blocks to various nodes and (ii) Byzantine Fault Tolerant PoS that is based on BFT research. They address the "nothing-at-stake" vulnerability in different ways. The chain-based PoS are penalizing nodes when sending multiple blocks on competing chains (e.g., Slasher [95,96], or Casper [93]). The BFT PoS mechanisms allow validators to vote on blocks by casting several messages, with two rules: finality condition (to determine when a hash is finalized) and slashing conditions (to determine when a validator misbehaved and must be excluded). A block is considered finalized once enough votes have been cast and all nodes from the DLT agree on adding it to the canonical history. This involves sending many messages in the network to make aware other nodes that a new block was proposed and running a version of the Byzantine Agreement on the new block.

Propagating many messages in the network impacts system scalability, thus methods to reduce the number of messages exchanged are needed leading to the development of hybrid approaches between Byzantine Agreement and Proof protocols [97]. Two techniques are found in the literature addressing this: (i) quorum based voting—when a node is selected randomly as the prover and a subset of nodes are selected to be verifiers that run a Byzantine Agreement protocol (Algorand [94]); and (ii) sharding-based approaches—where the blockchain is split into shards for inter-shard transactions and only transactions that involve nodes from two different shards need message propagation between shards (Elrond [98]). Algorand is based on a new and fast Byzantine Agreement Protocol used to generate a new block through a binary Byzantine Agreement (BA*) protocol that enhances the traditional BA protocol to work in rounds in a synchronous environment with at least 2/3 players being honest. Furthermore, cryptographic sortition based on Random Verifiable Functions is used to select a subset of the users to be members of the BA* algorithm. A cryptographic function is used to select a new leader based on a previous block. The leader will be in charge to propose the new block. A set of verifiers is used to check the validity of the new proposed block. The choice of the leader is not predictable, thus making it impossible for an attacker to alter the new block. Furthermore, leaders learn of their role without informing others only after proposing the new block, thus avoiding attacks. After a new block is proposed, the leader has no importance for the algorithm. However, the verifiers must agree on the new block, and they run the BA* algorithm in rounds, at each step players being replaced, thus avoiding cases when many verifiers are corrupt. Elrond is based on a sharding approach, splitting the blockchain and account state in several shards where parallel validation can occur using a consensus algorithm based on a secure PoS. The consensus algorithm follows a similar approach as Algorand with a prover and a set of validators chosen randomly within a shard and running a Byzantine Agreement algorithm to validate the proposed block. Finally, Hot Stuff [99] proposes a consensus algorithm using a leader-based Byzantine fault-tolerance protocol for partially synchronous distributed system models where a chosen leader drives the consensus decision at the rate of the maximum delay allowed by the network.

3. Scalability Tier

The DLT scalability limitations are mainly driven by the restrictions imposed by the Protocol and Network Tier solutions (e.g., the consensus algorithms). To achieve higher scalability, one option would be to curtail some of the features of Protocol and Network Tier. This can be done either by compromising the security, the immutability, or the consensus of the DLT. Because most of the time this is not acceptable, the scalability challenges are open for research for all DLT variations. In this sense existing concepts such as distributed databases or file systems, have been reconsidered and integrated with Protocol and Network Tier, to allow the implementation of solutions for the Scalability Tier [7,100].

Anyway, due to the advent of blockchain platforms and applications, most of the nowadays literature is focused on the scalability limitations of this type of ledger. They are imposed either by maximum block size (e.g., 1 MB Bitcoin) or by a cost constraint (e.g., gas consumption and gas price in Ethereum). These constraints are combined with the strict periodicity of the block generation (e.g., 10 min for Bitcoin, 15 s for Ethereum) imposing limitations in the number of transactions processed. Moreover, they are impacting both the storage and the processing capabilities (e.g., due to the gas consumption costs in the case of Ethereum smart contract execution). Bitcoin reportedly can allow 7 transactions/second on average [101], while Ethereum registers 13 standard transactions/second or 7 transactions/second in case smart contract execution is involved [102]. Even private deployments reach certain limitations. Hyperledger is advertising 100,000 transactions/second, although reports show a lower limitation of 700 transactions/second [60].

3.1. Storage Size

With the increased storage capabilities of the systems, much of the paper documentation has been digitalized in domains like healthcare, intellectual properties, real estate, legislative contracts, etc. Furthermore, the media and social network use cases are more and more flexible, providing increased storage options for users to store their files (documents, photos, videos, etc.). DLTs caught the interest of these domains, aiming to maximize their potential, by ensuring immutability (legislative and real estate), provenance (intellectual properties), security (healthcare), etc. However, the greatest challenge of integrating DLT solutions with these domains is limited storage capabilities.

To improve the storage scalability several solutions have been proposed, such as Sharding or the integration of well-known file systems with existing DLTs. They are aiming to store all the data outside the DLT and keep only a digital fingerprint of the data on the Protocol and Network Tier. While the data kept on the Protocol and Network Tier benefits of all the advantages the system provides (consensus, immutability, security, etc.) it is considered a source of truth in the validation of data that is stored on the Scalability Tier.

Sharding is a solution implemented to improve storage scalability [103]. Different nodes are assigned to process and store only a corresponding sub-category of transactions [104]. A simple sharding technique is to split the network in shards corresponding to the transaction's prefix: 0×01 shard, 0×02 shard, etc.

For example, in the sharding mechanism proposed by Ethereum Sharding [105], the system defines objects at three different levels: level 0—transactions; level 1—collations; level 2—blocks. The collations are the data structures responsible for package transactions that belong to a shard. The collations are created and sealed by Collators that are nodes in the network registered on the main chain in the Validator Manager Contract. The Collator deposits a sum of coins on the main chain based on which they will be chosen in a Proof of Stake manner to validate the next collation. The header of the proposed collation will then be verified on-chain and added in the next block on the main chain. Cross-Sharding communication is also possible by providing Merkle-Proofs of existing transactions from the main chain. Similar approaches are investigated by Elrond Network [106], Hyperledger [34,60], Elastico [107], Omniledger [108] and Rapidchain [109].

The Scalability Tier solutions that use file systems as storage mechanisms allow large files to be stored by fragmenting, encrypting, and sharing chunks of the original file between the nodes, while the hash of the original file is stored in the Protocol and Network Tier. The nodes storing the data need to respond to periodic checks regarding the integrity of the stored data, and a reward scheme is implemented for their services.

Figure 4 shows an example of storage mechanism and integration with the Protocol and Network Tier in the case of blockchain ledgers. There are several successful implementations of such distributed file systems among which, worth mentioning are: Storj [110], IPFS [111], Filecoin [112], MediaChain [113], Decent [114], Sia [115], MadeSAFe [116], Swarm [117] and Arweave [118].

IPFS (InterPlanetary File System) is one of the most used Scalability Tier solutions for file storage. In this case, when a user publishes a large file using its own IPFS node, the node will first fragment the file in smaller chunks, the hash of each chunk becoming a node in a Merkle DAG, whose root is the hash of the initial file, thus making use of hash pointers to ensure tamper-evidence. For security reasons, the chunks stored are of standardized sizes, so that an attacker cannot extract any useful information by analyzing the size of a chunk. The owner of the data is responsible to hold the private key used to encrypt the chunks of data that are scattered across the network. This makes the system highly secured since even the data is stored across multiple nodes, the mechanisms make it impossible for anyone holding the data to use it since it is encrypted and fragmented. Moreover, it ensures security through encryption and no downtime since the file is shared across multiple users. The system offers the possibility to transfer data, check the availability and the integrity of the stored data, retrieve the data, and pay for the service provided.

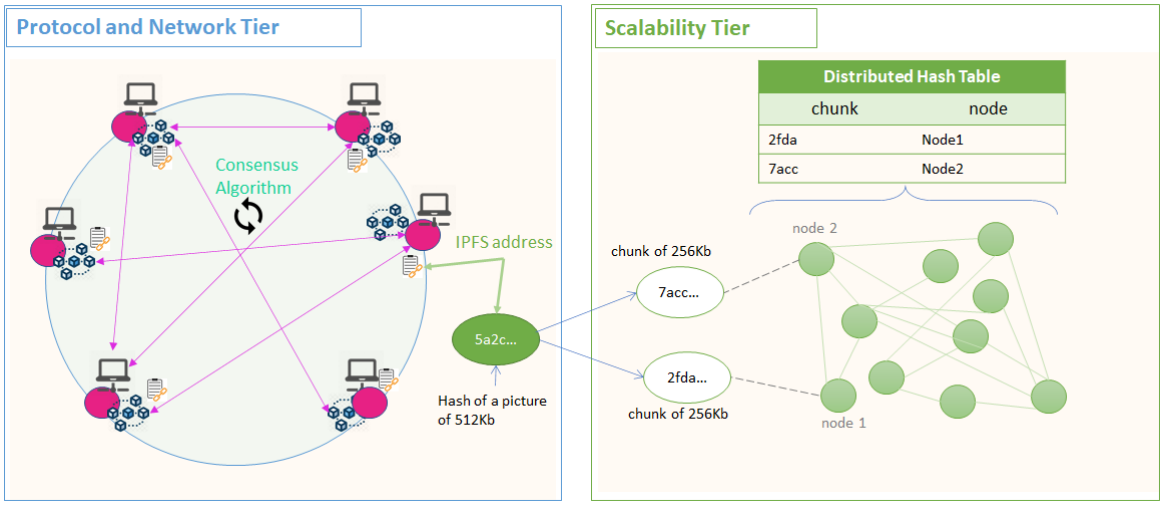

Figure 4. Scalability Tier—File System Storage Mechanism.

Similar implementations such as Storj [110] and Filecoin [112], are proposing to reward and motivate the decentralized nodes to act honestly regarding their storage services. Ethereum Swarm [119], is a peer-to-peer system that aims to store data in a decentralized way and relies on immutable content-addressable data. While IPFS needs Filecoin to validate storage proofs, Ethereum Swarm proofs are validated at the contract level and rely on incentive schemes based on the native coin, Ether.

In Table 6 a comparison between the identified storage scalability solutions is presented. Sharding presents a promising alternative to the classic DLTs, by providing increased storage capabilities. Using sharding the DLT storage capacity is multiplied with the number of shards, having the block sealing process parallelized with each shard. However, by increasing the number of shards, fewer nodes get to be assigned per each shard for validation. This can easily make the network susceptible to attacks since by attacking one shard the entire system can be compromised. The file systems solution even if it provides great scalability in terms of storage, also requires a degree of trust between the storing nodes. For example, in the case of IPFS, since it is not fault-tolerant on its own, the DLT storing the hash ensures only tamper-evidence in the system but does not make the system tamper-resistant. Each time an update is applied to one of the files, the hash pointer changes as well, requiring a transaction updating the entry on-chain as well. While this is desired to keep a tamper-evident system, a high frequency of updates will also lead to higher costs.

Table 6. Scalability tier storage solutions.

Features	Protocol & Network Tier	Scalability Tier	
	Fully Replicated	**Sharding**	**File Systems**
Immutability	Yes	Yes	No
Trusted Parties	None	None	Peer nodes
Byzantine Tolerant	Yes	A tradeoff with the no. of shards	No
Storage Scalability	Low	Medium	High
Cost	High	Medium	medium

3.2. Transaction Throughput

The number of transactions processed by blockchain ledger is important for implementing decentralized applications where micro-payments should be exploited (e.g., Energy Sector, Media Services, etc.). Micro-payments are online transactions involving small amounts of money. These small amounts of money are often used in exchange for

different goods or services, and most of the time require many transactions over a period. The problems that arise by integrating the micro-payments are the high cost accumulated as a result of the mining fees paid for each transaction and the congestions problems at the level of the Protocol and Network Tier due to the small size of the block. The most promising solutions analysed are the Sharding, Sidechains, and Payment Channels (see Table 7).

Table 7. Transaction scalability solutions.

	Protocol and Network Tier	**Scalability Tier**		
	Fully Replicated	**Sharding**	**Sidechains**	**Payment Channels**
Trusted Parties	None	None	Depending on implementation	None
Transaction Scalability	Low	Medium	Medium	High
Cost	High	Medium	Medium	Low

Sharding can be considered a suitable solution for increasing the transaction throughput as well as the storage [120]. Both improvements come because of introducing clusters of nodes responsible for specific categories of transactions. Higher transaction throughput may be provided by delegating the transactions to a different category of nodes and parallelizing the validation of these transactions. An issue arises when increasing the number of shards and transactions. When a transaction is sealed by a shard, it may reference a transaction that is not in the log of transactions of that shard. As a result, each time such a transaction needs to be validated, a cross-shard communication is required to issue Merkle Proofs of the referenced transaction. Consequently, increasing the number of shards and transactions will also introduce a communication overhead that may impact the scalability of the solution. More exact evaluations of the overhead introduced will be possible only after these currently researched solutions will offer a full specification and deployment on the public networks.

Sidechains [121,122] are proposed as alternatives for increasing the scalability of blockchain ledgers. A side chain is processing transactions in parallel with the main chain. The transactions of the sidechains are always rooted in a locking transaction in the main chain. Once proof of a locking transaction is made on the side chain, the actors may start using the assets by transacting on the side chain. To return to the main chain, proof of the latest state from the sidechain must be made to unlock the coins on the main chain. There are different reasons for connecting to side chains: testing new functionalities, extending the main chain functionalities (e.g., RootStock [123] enabling smart contract execution), or moving business-specific implementation to another less expensive chain (e.g., Plasma as a tree of sidechains [62]). In either case, moving part of the transactions from the main chain to side chains can lead to an improvement in the transaction throughput.

The sidechain implementation offers an alternative to sharding by fully relying on the information stored on the side chain and integrating with the main chain only when locking and unlocking the coins. This solution may offer some improvements to the underlying chain, by taking over some of the transitioning load. However, from a scalability perspective, the transaction throughput is still limited since the side chain is most of the time a blockchain ledger with the same constraints as any Protocol and Network Tier solutions. Additionally, some issues regarding the overall security of the systems need to be considered. Upon returning to the main chain, the transactions validated by the side chains are valid even if the validators' network of the side chain differs from the ones of the main chain.

The Payment Channels mainly implemented in Lightning Network solutions [124] are one of the best choices for improving transaction throughput. The mechanisms of the Lightning Networks were firstly defined for Bitcoin in [124], and future implementations have followed for other networks: Raiden for Ethereum [125] or Bolt for Zcash [126]. The Payment Channels aim to combine all the small payments into one large payment at the

end of a service period. The Lightning Network provides a point-to-point network that runs on top of the blockchain network as depicted in Figure 5.

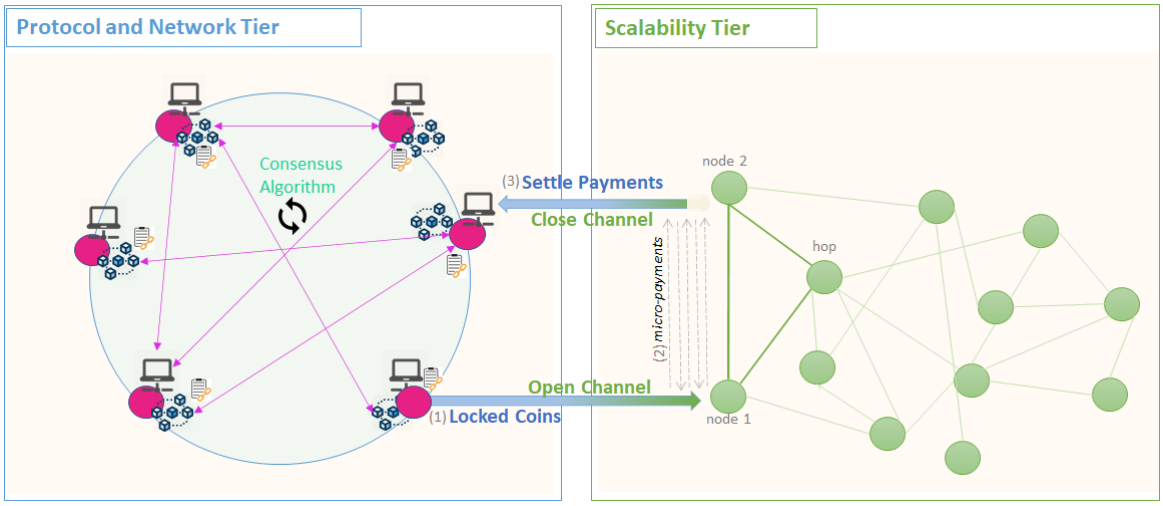

Figure 5. Scalability Tier—Payment Channels Mechanism.

It relies on hash time locks and cryptographic secrets to ensure the reliability of off-chain payments. The nodes in the Lightning Network exchange cryptographic signed transactions among them. They represent the payments exchanged between nodes offline. At any point, the payments can be deployed on the main chain to securely redeem the associated coins. Whenever two parties aim at exchanging many micro-transactions, the first step is to open a channel between them, represented on the blockchain as an opening transaction that locks the maximum amount of money that the two parties could transact off-chain. The transaction opening is signed by both parties involved. To unlock it both parties need to agree on the spending of the amount. Afterward, the parties will continue to exchange messages off-chain, namely, commitment transactions, which are designed as fail-safe mechanisms against cheating off-chain. Whenever a transfer occurs, determining a change in balances, each party creates one commitment transaction specifying the updated balances, signs it, and sends it offline to the other party. Upon receiving the commitment any party can successively sign it and publish it on-chain to redeem the coins or can wait and make other off-chain transfers and return only with a future commitment transaction on-chain. To prevent any party to return to the chain with a deprecated and more favorable commitment transaction, secret-locking and time-locking mechanisms are incorporated. They provide a fail-safe mechanism such that any party acting fraudulently risks losing all the money in favor of the other channel party.

The benefit of transferring over a route is the reduction of the cost associated with the opening transactions required on-chain each time a channel is required with a specific party. For sending the transactions through a path that requires intermediary parties (hop nodes) to route the messages, additional security mechanisms are required to ensure the correct delivery of the transacted assets. For successfully issuing a routed transfer, a commitment transaction is exchanged between every two parties involved in the path channels. Hash Time-Locked Contracts [127] have been defined over the previously presented mechanism, to prevent the intermediary to unlock the routed commitments before the receiver can confirm the payment. Upon confirmation, a proof is issued by the recipient and sent to the intermediary to unlock the hash-locked commitment. If the proof is not provided during an established period (time lock) then the intermediary will no longer be able to claim the payment.

The Payment Channels offers the best scalability solution. It is implemented without relying on any third parties and provides higher transaction throughput, promising millions of transactions per second. However, once the network has many users it is more difficult to

find a viable route between two parties. The most desired topology of the lightning network would be a completely decentralized network. In this case, each node has connections with other nodes, together forming a mesh of channels that are part of a connected graph. However, to be able to route money from one point to another, all the nodes from the path need to have at least the same amount of money as the one requested by the initiator. Taking advantage of this issue motivates some actors of the system to open channels with a larger number of parties and fuel the channels with enough money to be able to route and connect different parts of the network, acting like large routing hubs in exchange for small fees.

3.3. Processing Capability

The processing capability limitation is extremely relevant for blockchain distributed ledgers that allow the implementation of complex functionality and computations through smart contracts. For example, in Ethereum, the concept of gas was introduced to measure the amount of computational effort. When running smart contracts, each executed operation and processed byte of data is paid for with gas. This mechanism prevents an attacker from running extremely long tasks or infinite loops since the attacker would need to provide enough reward to incentivize the miner to execute each operation. When the reward provided runs out, the computation stops, and the transaction is dropped, thus avoiding situations where the nodes become unavailable due to attacks or complex computations. By integrating the Protocol and Network Tier system with external services, this shortcoming can be solved.

The nowadays solutions for addressing complex computational problems in blockchain ledgers are Oracles [128] and Proof of Computation mechanisms [129].

The Oracles are mechanisms that provide a secure connection between the blockchain and the outside world. They act as a trusted third-party entity, or a network of entities, for the Protocol and Network Tier. The Oracles can be used to offer results from different URLs, [130], IPFS, or units responsible for running more complex algorithms. The problem with interacting with the Oracles, directly from the chain, is that the response must be the same across any number of requests issued by the nodes during mining. This proves to be almost impossible when accessing dynamic changing data regarding weather, stock prices, etc. One problem that can appear in the Oracle-based system is data tampering or man-in-the-middle attack. In this sense, the Oracles are responsible to ensure the authenticity of data through authenticity proofs. One problem that persists is the centralized nature of the Oracles. The mechanism is presented in Figure 6 for a blockchain ledger [131]. An event containing details about the request is issued from the blockchain and intercepted by the Oracle. The necessary information is retrieved from external services and published back on the chain through a callback transaction.

Other implementations aim to outsource computing-intensive problems to off-chain nodes by implementing a Proof of Computation mechanism. Compared to the Oracles the Proof of Computation is a better choice. It is implemented without relying on any trusted party and provides validation of the result implemented directly on-chain. The proposed solutions show great potential for use cases requiring security and correctness validation for more complex computations than the ones that can be handled on-chain.

Figure 7 shows the Proof of Computation mechanism of TrueBit [129]. It relies on Ethereum smart contracts and gives the possibility of peers to request solutions for complex computational tasks. A Solver that has enough computational resources will run the tasks outside the chain and submit its results.

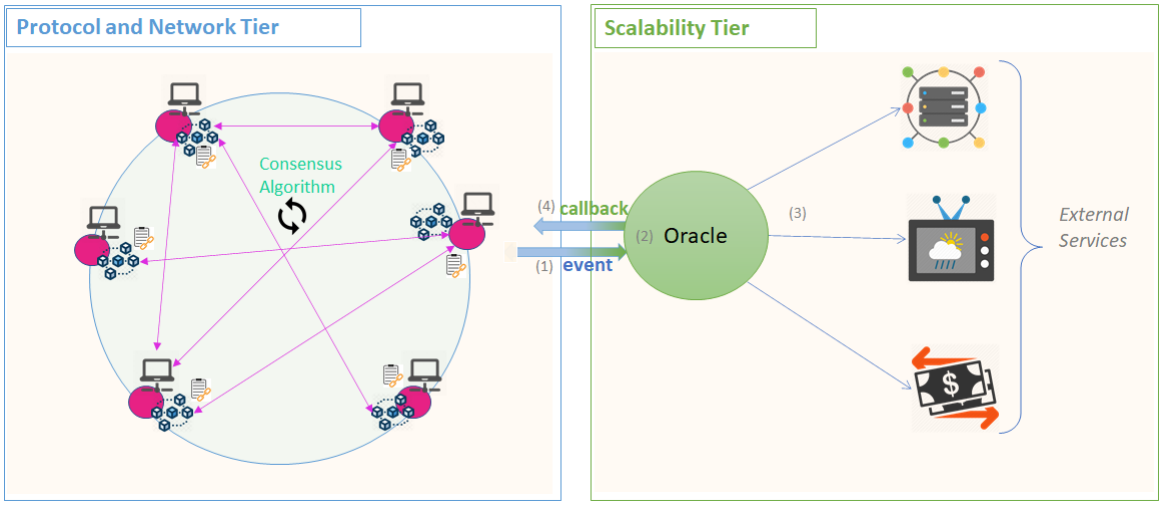

Figure 6. Scalability Tier—Oracle for External Services Integration.

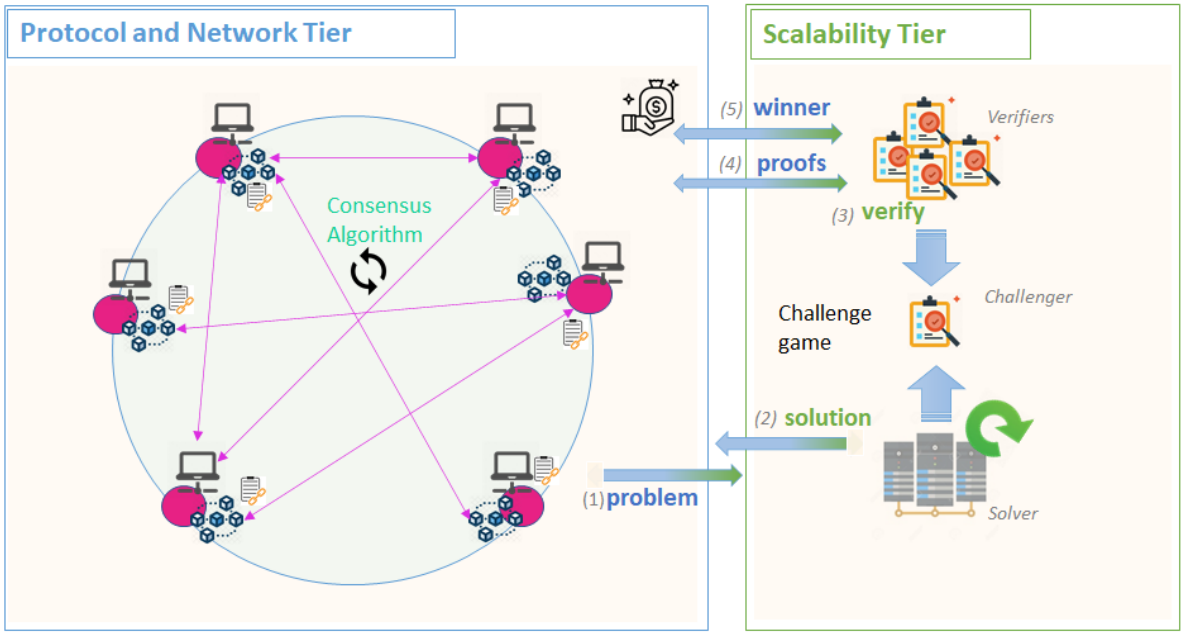

Figure 7. Scalability Tier—Proof of Computation mechanism.

Several Verifiers peers can evaluate the results, and if a disagreement occurs, a Challenger can contradict the result published, by starting a Challenge Game. Several rounds of proofs are registered on the chain to check whether the computation was done correctly. In the end, the winning part will receive a reward for its cooperation, while the part proven to be wrong will be charged for its actions. Another set of actors are de Judges that given the proof of the solution can easily verify the correctness of the game. However, a big drawback of the system is that it is limited to running tasks written in WASM [132]. Enigma [133] proposes a similar concept, of outsourcing the computation but with an added layer of privacy over the data and computation performed. By leveraging on secure multi-party computation, the proposed solution distributes the data across several nodes for computation. As a result, no central node will have access to the entire problem or solution, but to a seemingly unintelligible part of it. The proposed solution is not yet released in production, but it is currently tested on the Ethereum test network [134].

4. Interoperability Tier

By launching various domain-specific DLT applications for public use, each addressing different requirements, one problem that arises is the need for an interoperability protocol or mechanism. It should facilitate the transition from isolated DLT applications to networks of integrating DLT applications. Such a network could further unlock the potential of the DLT by allowing different types of businesses or applications to leverage on a different type of DLTs. Moreover, they will coexist and cooperate by sharing their ledgers.

The state-of-the-art solutions for DLTs integration are mostly referring to the blockchain ledgers [7]. Due to the development of a high number of blockchain platforms and applications, their interoperability is the main technological trend in the next years [135]. The most popular ones such as Ethereum, Bitcoin, or Hyperledger use different data formats and data interchange solutions making their integration difficult [136]. For example, Interledger [137] developed a protocol for allowing communication across different blockchain ledgers by using payment channels. The mechanism is called "AtomicSwap" and uses routes existing in the Lightning Network. Considering a hop that has Receiver and Sender channels opened on two networks, such as Bitcoin and Ethereum. Upon a transfer, the hop can agree to update its Bitcoin balance from the Receiver channel, in return for Ether on the Sender channel.

Most state-of-the-art interoperability solutions are addressing the transfer of coins from a parent blockchain ledger to a secondary one. Two-way peg systems [122] are mostly used for this type of Ledger-to-Ledger communication. The transfer is achieved by temporarily locking some coins on the parent blockchain and then unlocking the same amount of coins on the secondary blockchain. The transacting process is executed on the secondary chain until the user decides to return to the main chain. This is possible by repeating the same process on the secondary chain, thus locking the coins on the secondary chain, and then releasing them on the main chain. A central exchange that needs to have access to both chains is proposed as a solution for implementing the transfer of coins between two blockchain ledgers [123]. A user sends a request from the main chain specifying the number of coins and the address that should receive the coins in the secondary chain. The exchange can then simply send the same amount of coins to the specified address on the secondary chain. The central exchange needs to be a trusted entity; otherwise, the money could be easily stolen from the two chains. To address this issue a MultiSignature scheme can be used [138]. In this case, any transaction must be approved by N out of M participants, instead of relying on only one entity. It still relies on a middleman, but the risks are significantly reduced.

The entangling of the chains is presented as a potential solution for coin transfer between blockchain ledgers. It uses a secondary chain to monitor the main chain and includes all the block headers in the main chain blocks [139]. As a result, blocks from the secondary chain will have two parents: the previous block from the secondary chain and the last mined block on the main chain. The main disadvantage of this approach is that the main chain needs to create blocks at a lower rate than the secondary chain. An entangled model is used by BTC-relay that connects the Bitcoin chain with the Ethereum chain [140]. After every 10 min, proof of the last mined block in Bitcoin is provided in the Ethereum smart contract. Each time a user wants to prove the validity of a Bitcoin transaction on the Ethereum chain, it only needs to provide the Merkle Path of the transaction and the block it is contained in (a Simplified Payment Verification (SPV) proof [141]).

The sidechain solution for coni transfers aims to eliminate the middleman. The interoperability protocol of the chains implements a new way of unlocking coins, which is *proof of a locked transaction on the other chain*. The transacting parties will submit on the sidechain, an SPV proof of the transaction deployed on the main chain. The SPV proof will contain the Merkle path for the transaction submitted and mined and the hashes of all the blocks that followed. Upon receiving the proofs, the sidechain will enter a reorganization phase. This phase aims to provide the necessary time to avoid any possibility of a double-spending problem. In the case of a fork in the main chain network, anyone can offer a

new SPV proof that contains the same block as the one provided initially, but without the transaction in it. If the proof provides a longer list of block hashes that confirm the missing transaction block, it is concluded that a double-spending attack was committed on the main chain and the original transaction is ignored in the sidechain as well. Upon withdrawal on the main chain, the same process occurs. The main chain lockbox is defined as a new type of transaction that can only be unlocked using SPVs of locked transactions from the other chain. The system still has a notable drawback since the main chain needs to rely on the integrity of the sidechain. If the miners of the sidechain are not honest and coins are stolen, the main chain has no way to prove the honesty of the requests if a valid SPV is provided. Furthermore, if the amount of coins is split on the sidechain, to retrieve the coins on the mainchain all the owners from the sidechain must provide the SPV, thus no partial consumption of the coins on the main chain is permitted. Drivechain [142] is an improvement of the sidechain. The protocol is like the sidechain one until the step requiring withdrawal from the sidechain back to the main chain. At this point, the SPV proofs are not used directly to withdraw the coins on the chain. During the reorganization period, several withdrawal proofs are joined aiming to consume a given amount of coins from the main chain. The withdraw transaction id, (not the actual transaction) will then be mined in the next block on the main chain's transaction. The mined ID will be interpreted as the intent of spending the locked money but will give time (1008 blocks) for all the users from the sidechain to validate the transaction and give chance to the miners to vote whether the actual transaction should be mined on the mainchain. In this way, the miners of the sidechain are prevented to commit illegal transactions, since any commit to the main chain will first be evaluated by the corresponding actors and most miners. A hybrid model of Drivechain is proposed by Rootstock [123], implementing a combination of sidechain and multi-signature federation. It uses sidechain functionality for passing coins from the main chain to the secondary chain. However, when returning not only the miners have the right to vote but also specially delegated notaries that vouch for the integrity of the transactions.

The communication between blockchain ledgers is of much interest for Ethereum as well. A solution in this platform case is to offer different chains per application. CryptoKitties [23], is such an example of an application build on an alternative chain. It is completely independent of the state and data stored on the Ethereum main chain. However, the economic value of the application tokens need to be maintained, thus the system should allow purchasing the tokens on the main chain, paying with actual ethers for acquiring decentralized applications specific tokens, and then moving the tokens on a separate chain to use them in the actual application. Plasma [62] is an alternative proposal of the sidechains on Ethereum. An equivalent transaction is generated on the sidechain (plasma chain) from nothing, giving the corresponding coins to the plasma chain user. The validation mechanism is based on Fraud Proofs by periodically checking the main chain. If a user wants to spend its coins on the main chain in a fraudulent manner, its action can be proved to be malicious by submitting proof of a spending transaction that was previously registered on the plasma chain. This would prevent the main chain transaction from being validated. Since a user can issue a spending transaction directly from the main chain this offers a great advantage over the classical sidechain approach. That is, in case the plasma chain is compromised, the users can still issue their withdrawals. Moreover, Plasma is designed to permit the implementation of nested chains, thus creating a tree-structured system of chains that requires each user to monitor only the chains that can affect one's transactions.

Table 8 presents the main solutions for inter blockchain ledgers interoperability and communication comparing their main features. While hybrid models and Plasma may offer reliable solutions in terms of ledger-to-ledger interaction, the Lightning Network can be considered a viable alternative that can provide high transfer rates.

Table 8. Ledger to Ledger interoperability approaches.

	Implementation Level	Deposit Mechanism	Withdraw Mechanism	Trusted Entities	Chain Independence
Central Exchange	Escrow	TX to a central authority	TX to a central authority	Central Authority	Yes
MultiSig Federation [138]	Escrow	TX to a multi-signature federation	TX to a multi-signature federation	N Delegates	Yes
Entangled Chain [140]	Protocol Layer	SPV Proof from the main chain	-	Dependent on the withdraw	Mining rate restriction
Sidechains [121,122]	Network & Protocol Layer	SPV Proof from the main chain + proof of block validity	SPV Proof from the sidechain + proof of block validity	Sidechain miners	Yes
Drivechain [142]	Network & Protocol Layer	SPV Proof from the main chain + proof of block validity	SPV Proof from the sidechain + proof of block validity + miners votes	Sidechain miners	Yes
Hybrid Models [123]	Network & Protocol Layer	SPV Proof from the main chain + proof of block validity	SPV Proof from the sidechain + proof of block validity + miners votes + multi-signature notaries	Sidechain miners + Notaries	Yes
Plasma [62]	Network & Protocol Layer	Proof for TX on the main chain	Direct withdraw + Fault Proofs	Central Authority, N Delegates	Yes
Lightning Network, Interledger [137]	Scalability Tier Solution	Atomic Swap	Atomic Swap	None	Yes

5. Discussion and Development Guidelines

As presented in the previous sections there are a lot of distributed ledger technologies. Most of them have emerged in the last years to address specific development issues. The ecosystem is subject to rapid changes making the selection of technologies for the implementation of decentralized applications rather difficult and fuzzy. At the same time DLTs bring benefits concerning the implementation of decentralized applications. They eliminate the need for a mediator, having the capabilities to enforce contract rules on-chain, each participant being aware of the consequences of his actions. The hashed data structures allow for easy traceability of the assets and state updates in the ledger. Since the records are public and replicated, great transparency is provided. Even if all the transactions and all the actions are public the platforms provide high security through consensus, public-key cryptography, and tamper-resistant recording. Nevertheless, a lot of nowadays applications and management systems are rather centralized (e.g., utility grids, banking, stock exchange, etc.). Thus, efforts are committed to investigating how the DLT can be applied, integrated, and used for decentralizing such applications and systems.

In the rest of this section, we present a guideline for decentralizing systems by designing and implementing decentralized applications.

5.1. Decentralization of Design

The main steps required for decentralizing a centralized system by designing decentralized applications are detailed below discussing their role and specific technological requirements (see Figure 8).

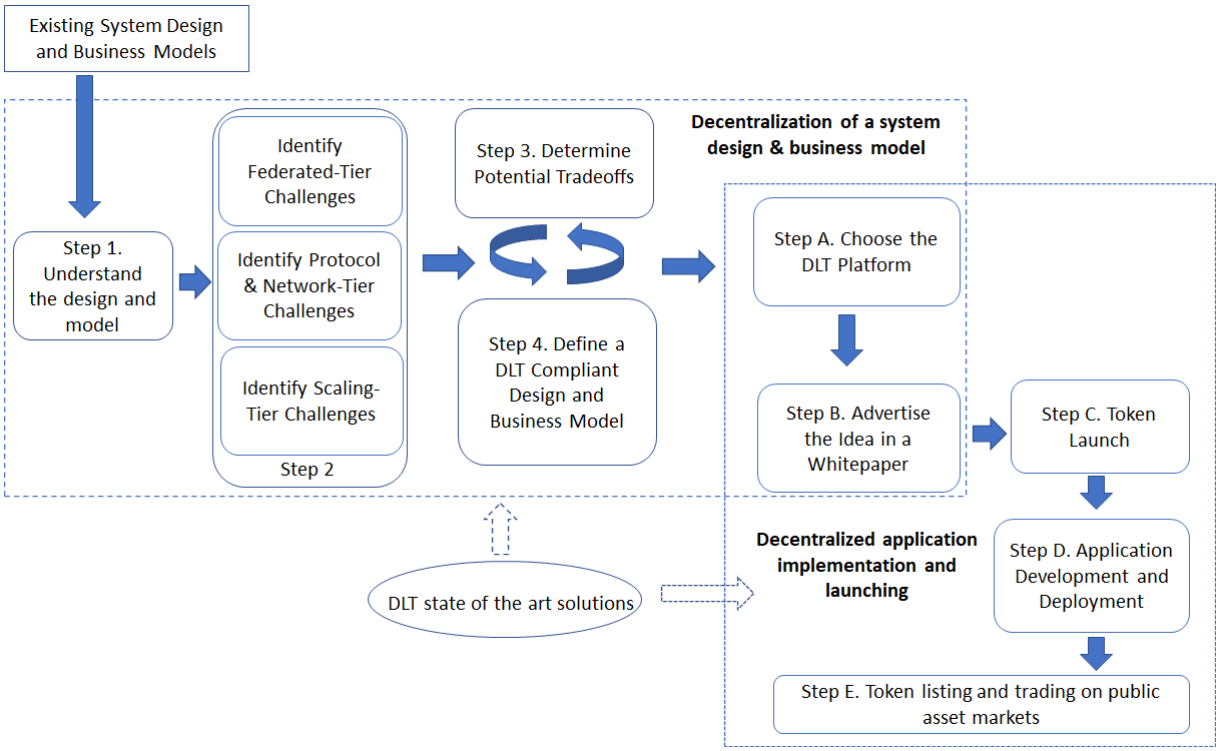

Figure 8. Decentralized application design and implementation steps.

Step 1. *Understanding the existing system design and business model.* An exhaustive understanding of the business is required to design a decentralized application. Good knowledge of the business model and rules that govern the domain, as well as of the architecture used in the centralized implementation, are prerequisites for starting the sequence of steps required to achieve a consistent and well-documented decentralized solution. One needs to identify the functional and non-functional requirements that need to be addressed for ensuring the completeness of the design. Most of the time, a complete and sound list of requirements can be obtained by holding several interview sessions with the future stakeholders of the system.

Step 2. *Identify potential challenges for each tier.* Once the functional and non-functional requirements are clearly defined, the challenges that can arise once mapping the model on a decentralized solution can already be identified. We classify the challenges based on the tiers identified at the level of architecture. Protocol and Network Tier challenges: What is the targeted network level, public or private? Does the system need to hide the transacted information? Is selective disclosure a requirement of the business? Etc. Scalability Tier challenges: Are the scalability requirements higher than what the Protocol and Network Tier solutions can offer? Are the scalability requirements targeting storage, throughput, or computation?

Step 3. *Determine potential tradeoffs*. Depending on the set of challenges identified, there might be situations where not all the requirements can be successfully ensured by the DLT solutions. In such situations, a tradeoff needs to be made. Most of the time, the application scalability may be affected while incorporating the reliability, immutability, and consensus properties ensured by a DLT. Furthermore, by outsourcing components of the system to the Scalability Tier, another tradeoff regarding the actual decentralization must be done since most of the Scalability Tier solution requires a trusted party to oversee the outsourced component.

Step 4. *Define a DLT compliant translation of the application design and business model.* It is important to conduct thorough research of the state of the art whenever a DLT solution is considered, for a system decentralization. Obtaining a clear picture of

all potential theologies alternatives to the problems identified can be quite difficult. A decentralized design of the system should be proposed, and the most suitable solution for each architectural component needs to be selected.

For example, if smart contracts technology is selected for implementing the application business logic several key features need to be carefully investigated. The smart contracts can keep track of the data stored and act as a financial escrow for the interacting parties. The functions of a smart contract can be used internally by the contract, or can be exposed as an API to the external modules. Most of the time the smart contracts are used as state machines, where the state is updated according to the latest input received. To interrogate a DLT event-based information is considered. The events are elements emitted by the contracts during processing and stored in tamper-resistant structures as well. Instructions executed by a smart contract in a public chain have a processing cost paid by the transaction issuer. The payment is directed to the miner, for using its computational resources. Consequently, an appropriate data structure must always be used to avoid high costs. The processing rules should be implemented, if possible, in the same contract where the data is stored, to avoid the cost of referencing other contracts. When deploying a contract, the storage of the code is also paid by the issuer. For multiple deployments of the same contract, a good approach is to use libraries containing static code that is deployed only once and then linked to each of the deployed contract instances.

From this point forward, a classical pipeline of implementation, verification, and maintenance can be applied to validate the final application.

5.2. Decentralized Application Development

Upon successful implementation of the above-presented steps, a decentralized design of the system and a DLT compliant translation of the business model is obtained. There are several steps to go through for launching such a decentralized application, from DLT platform selection, advertising, and crowdfunding up to the actual development and operationalization (see Figure 8).

Step A. *Choose the DLT platform.* From this point forward one needs to proceed with the evaluation of the existing DLT implementation platforms, since some decentralized applications may be compatible with existing systems, thus beneficiating from the already built network of miners. Two cases may emerge (Table 9): an existing platform matches the decentralized application requirements, or a new custom chain needs to be implemented. In the latter case, a custom chain is built and used as the base PN-Tier for the decentralized application implementation. Most of the time, this situation arises when the current frameworks' specifications are not compliant with the application requirements, thus a new DLT core platform must be developed, and a new network of nodes must be built. Furthermore, a tradeoff regarding the cost and the complexity of the implementation must be considered when choosing one of the two solutions.

Step B. *Publish and advertise the idea through a whitepaper.* One principle of decentralized application whitepaper is to provide complete transparency to build trust with the investors. Companies are encouraged to provide an honest technical and economic roadmap. They should explain in detail the technical feasibility of the solution as well as the investment and revenue plans, the shares among the company partners, the economic sustainability of the solution, etc.

Step C. *Token launch or Initial Coin Offering.* A fixed number of tokens should be released to attract investors and raise money for the development phase. The token distribution plan specifies the maximum number of tokens that will be ever generated by the decentralized application, the tokens unlocked for distribution, the tokens transferred to the founders or other participants, and the tokens locked for future use. The initial coin offering plans are advertised by the company, mentioning the initial price, start date, end date, the number of tokens unlocked,

pricing schemes (constant pricing, incremental pricing, etc.). The token registry and the distribution rules are programmed as smart contracts and deployed on-chain. Each buyer willing to invest in the decentralized application will need to acquire the necessary cryptocurrency and sign a transaction paying the requested sum in return for several tokens. The distribution contract will validate the deposited sum, and if all the rules hold, the token registry will update the buyer's account accordingly.

Step D. *Development and application deployment*. The decentralized application development will continue according to the roadmap presented in the whitepaper. Based on the alternatives presented in the previous chapters, a guideline for choosing the specific implementation solutions is presented in Table 10. Upon finalization, the application will be deployed and all the users that acquired tokens during the initial coin offering will be able to start using the decentralized application or will be able to sell the tokens to other interested parties.

Step E. *Token listing on public exchanges*. To facilitate the exchange of tokens between interested users, the company can list the token on one of the public exchanges. Sellers can make their offers and buyers can make their bids therefore, depending on the public interest in the launched business, the token can have the potential to raise its value. Being an off-chain exchange, only upon settlement the updated balance will be registered on-chain mirroring the transaction that happened between the seller and the buyer. However, some of these exchanges chose to act as a custodian for the exchanged tokens and the actual registry is not updated on-chain.

Table 9. DLT platform choice.

Option	Advantages	Disadvantages
Standalone Customized DLT	Target the business requirements, limitations, and challenges	Needs to build a network and gain trust in the mining nodes. Increased implementation complexity
Existing DLT Platform	Existing P2P Network and community Attacks are unlikely considering the high number of existing nodes	High costs imposed by the mining nodes Some tradeoffs may be made due to the core-properties of an existing DLT

Table 10. DLT technologies and alternatives.

Issue	Tier	Description	Alternative Solutions
Asset Representation	PN-Tier	Custom tokens in an existing DLT	**ERC-721**: CryptoKitties [23], Rarible [22], EtherTulips [20] **ERC20**: Grid [21], Telcoin [25], Storj [110]
		Native token in a custom build DLT	Filecoin [112], MediaChain [113]
Data Structure	PN-Tier	Selection of Security and Decentralisation over Scalability Selection of Scalability and Security over Decentralization	**Blockchain platforms**: Ethereum [15], Hyperledger [34], Quorum [49] **Directed Acyclic Graphs**: IOTA [16], HashGraph [39]
Data privacy	PN-Tier	Anonymity of identities involved	**Ring Signatures**: Monero [18], CryptoNote [54] **Mixers**: CoinJoin [47], Dash [48]
		Anonymity of content retaining verifiability	Homomorphic Encryptions (MimbleWimble [53]); **Secure Multi Party Computation**: Enigma [133]; **Zero-Knowledge Proofs**: Zcash [19], Zerocoin [51],
	PN-Tier/ S-Tier	Anonymity of data using references to locations in Scalability Tier	**External Storage System**: IPFS [111]

Table 10. *Cont.*

Issue	Tier	Description	Alternative Solutions
Business Enforcement	PN-Tier	Built-in custom business rules	Filecoin [112], NRGCoin [24]
		Generic rules and pluggable Turing Complete implementation	Ethereum [15], Hyperledger [34]
Data Propagation	PN-Tier	Selective disclosure	Hyperledger Multichannel (Corda [61], Plasma [62]), Quorum [49]
		Global disclosure	Any public permissionless DLT
Permissions & Consensus	PN-Tier	Public Permissionless & Byzantine Fault Tolerance	Ethereum with PoS or PoW
		Public Permissioned	Ethereum PoS or PoA; Hyperledger [34]
		Consortium or Private	Hyperledger [34], Corda [61]
Large Size data	PN-Tier	Selection of Scalability and Decentralisation over Security	BigChainDB [20]
	PN-Tier/ S-Tier	File System Storage	IPFS [111], Storj [110], Filecoin [112]
High-Frequency Data	PN-Tier/S-Tier	Selection of Scalability and Security over Decentralization	IOTA [16], Sharding [104], Sidechains [121]
		Overlay Networks; Payment/State Channels	Payment Channels [124], Raiden [125]
Computational Intensive Algorithms	PN-Tier/S-Tier	Selection of Scalability and Security over Decentralization	**Secure Multi-Party Computation**: Enigma [133]; TrueBit [129]
		Oracles	Provable [128]

6. Conclusions

The distributed ledger technology has the potential of being a game-changer in many domains, its recent developments being triggered not only by technology expectations but also by social ones. DLTs have enabled the development of decentralized applications but this process is still complicated due to the high availability of new and rather immature technological solutions that address different implementation issues. DLTs and especially blockchain are an effervescent innovation area. Therefore, the review of technological solutions and implementation guidelines are needed to improve understanding and to ease the development of decentralized applications.

Most of the similar initiatives found in the literature are focusing on the blockchain-based distributed ledger and pay little attention to other DLT implementations. In this paper, we have provided a review of the existing DLT solutions. We highlight their applicability, advantages, and disadvantages concerning other technologies. A significant number of solutions have been reviewed to provide a holistic image of the DLT implementation variations. In the technology review process, we have considered references from both academia and the private sector.

At the same time, our review was driven by a decentralized application architecture that has been used and validated in previous publications. The DLT solutions are categorized according to the 3-tiers of the architecture thus the criteria for selecting the technologies and organizing the review are strictly related to the development issues that may be encountered and need to be addressed in each tier.

Finally, we provide a guideline for decentralized application development defining specific steps for decentralizing a system design and business model using DLT and for implementing and launching it as a decentralized application. No similar guideline could be found in the reviewed literature. The guideline can be used to streamline the nowadays efforts for investigating how the DLT can be applied, integrated, and used for decentralizing systems such as medical systems, electricity grids, financial sector, etc., and for implementing new decentralized applications.

Author Contributions: Conceptualization, T.C. and I.S.; methodology, T.C. and I.A.; investigation, C.A. and M.A.; writing—original draft preparation, C.A. and M.A.; writing—review and editing, I.S. and T.C.; visualization, I.A.; supervision, I.S. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the European Union's Horizon 2020 research and innovation program grant number 957816 (BRIGHT) and grant number 774478 (eDREAM).

Data Availability Statement: Not Applicable, the study does not report any data.

Conflicts of Interest: The authors declare no conflict of interest. The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript, or in the decision to publish the results.

References

1. Top Trends in the Gartner Hype Cycle for Emerging Technologies. 2016. Available online: https://www.gartner.com/en/newsroom/press-releases/2016-08-16-gartners-2016-hype-cycle-for-emerging-technologies-identifies-three-key-trends-that-organizations-must-track-to-gain-competitive-advantage (accessed on 24 February 2021).
2. Top Trends in the Gartner Hype Cycle for Emerging Technologies. 2018. Available online: https://www.gartner.com/smarterwithgartner/5-trends-emerge-in-gartner-hype-cycle-for-emerging-technologies-2018/ (accessed on 24 February 2021).
3. Top Trends in the Gartner Hype Cycle for Emerging Technologies. 2019. Available online: https://www.gartner.com/smarterwithgartner/5-trends-appear-on-the-gartner-hype-cycle-for-emerging-technologies-2019/ (accessed on 24 February 2021).
4. Nakamoto, S. Bitcoin: A Peer-To-Peer Electronic Cash System. 2008. Available online: https://bitcoin.org/bitcoin.pdf (accessed on 24 February 2021).
5. Pop, C.; Antal, M.; Cioara, T.; Anghel, I.; Sera, D.; Salomie, I.; Raveduto, G.; Ziu, D.; Croce, V.; Bertoncini, M. Blockchain-Based Scalable and Tamper-Evident Solution for Registering Energy Data. *Sensors* **2019**, *19*, 3033. [CrossRef]
6. Pop, C.; Cioara, T.; Antal, M.; Anghel, I.; Salomie, I.; Bertoncini, M. Blockchain Based Decentralized Management of Demand Response Programs in Smart Energy Grids. *Sensors* **2018**, *18*, 162. [CrossRef] [PubMed]
7. Farahani, B.; Firouzi, F.; Luecking, M. The convergence of IoT and distributed ledger technologies (DLT): Opportunities, challenges, and solutions. *J. Netw. Comput. Appl.* **2021**, *177*, 102936. [CrossRef]
8. Maesa, D.; Mori, P. Blockchain 3.0 applications survey. *J. Parallel Distrib. Comput.* **2020**, *138*, 99–114. [CrossRef]
9. FBenčić, M.; Žarko, I.P. Distributed Ledger Technology: Blockchain Compared to Directed Acyclic Graph. In Proceedings of the 2018 IEEE 38th International Conference on Distributed Computing Systems (ICDCS), Vienna, Austria, 2–6 July 2018; pp. 1569–1570. [CrossRef]
10. Zia, M.F.; Benbouzid, M.; Elbouchikhi, E.; Muyeen, S.M.; Techato, K.; Guerrero, J.M. Microgrid Transactive Energy: Review, Architectures, Distributed Ledger Technologies, and Market Analysis. *IEEE Access* **2020**, *8*, 19410–19432. [CrossRef]
11. Pop, C.; Antal, M.; Cioara, T.; Anghel, I. Trading Energy as a Digital Asset: A Blockchain based Energy Market. In *Cryptocurrencies and Blockchain Technologies and Applications: Decentralization and Smart Contracts*; Gulshan, S., Nhuong, L., Kavita, S., Eds.; Wiley-Scrivener: Hoboken, NJ, USA, 2020; ISBN 978-1-119-62116-4.
12. Pop, C.D.; Antal, M.; Cioara, T.; Anghel, I.; Salomie, I. Blockchain and Demand Response: Zero-Knowledge Proofs for Energy Transactions Privacy. *Sensors* **2020**, *20*, 5678. [CrossRef]
13. Pop, C.; Pop, C.; Marcel, A.; Vesa, A.; Petrican, T.; Cioara, T.; Anghel, I.; Salomie, I. Decentralizing the Stock Exchange using Blockchain An Ethereum-based implementation of the Bucharest Stock Exchange. In Proceedings of the 2018 IEEE 14th International Conference on Intelligent Computer Communication and Processing (ICCP), Cluj-Napoca, Romania, 6–8 September 2018; pp. 45–466. [CrossRef]
14. Pop, C.D.A.; Cioara, T.; Antal, M.; Anghel, I. Blockchain Platform for COVID-19 Vaccine Supply Management. 2020. Available online: https://arxiv.org/abs/2101.00983 (accessed on 24 February 2021).
15. Wood, G. Ethereum: A Secure Decentralised Generalised Transaction Ledger. Ethereum Project Yellow Paper 151.2014 (2014): 1–32. Available online: http://gavwood.com/paper.pdf (accessed on 24 February 2021).
16. Pervez, H.; Muneeb, M.; Irfan, M.U.; Haq, I.U. A Comparative Analysis of DAG-Based Blockchain Architectures. In Proceedings of the 12th International Conference on Open Source Systems and Technologies (ICOSST), Lahore, Pakistan, 19–21 December 2018; pp. 27–34. [CrossRef]
17. Branson, E. *Litecoin: The Ultimate Beginner's Guide for Understanding Litecoins and What You Need to Know*; CreateSpace Independent Publishing Platform: Scotts Valley, CA, USA, 2014; ISBN 1507878192.
18. Alonso, K.M. Zero to Monero. Available online: https://www.getmonero.org/library/Zero-to-Monero-1-0-0.pdf (accessed on 24 February 2021).
19. Sasson, E.B.; Chiesa, A.; Garman, C.; Green, M.; Miers, I.; Tromer, E.; Virza, M. Zerocash: Decentralized anonymous payments from bitcoin. In Proceedings of the 2014 IEEE Symposium on Security and Privacy, Berkeley, CA, USA, 18–21 May 2014. [CrossRef]
20. EtherTulips. Available online: https://ethertulips.com/ (accessed on 24 February 2021).

21. Grid. Available online: https://web.gridplus.io/grid-token (accessed on 24 February 2021).
22. Rarible. Available online: https://rarible.com/ (accessed on 24 February 2021).
23. CryptoKitties: Collectible and Breedable Cats Empowered by Blockchain Technology. Available online: http://upyun-assets.ethfans.org/uploads/doc/file/25583a966d374e30a24262dc5b4c45cd.pdf?_upd=CryptoKitties_WhitePapurr_V2.pdf (accessed on 24 February 2021).
24. NRGcoin. Available online: https://nrgcoin.org/ (accessed on 24 February 2021).
25. TelCoin. Available online: https://www.telco.in/ (accessed on 24 February 2021).
26. Ethereum Improvement Proposals. Available online: http://eips.ethereum.org/erc (accessed on 24 February 2021).
27. Blocksquare. Available online: https://blocksquare.io/ (accessed on 24 February 2021).
28. Blockchain in Commercial Real Estate: The Future Is Here. Available online: https://www2.deloitte.com/us/en/pages/financial-services/articles/blockchain-in-commercial-real-estate.html (accessed on 24 February 2021).
29. Zeilinger, M. Digital art as 'onetised graphics': Enforcing intellectual property on the blockchain. *Philos. Technol.* **2018**, *31*, 15–41. [CrossRef]
30. Nielson, B. Blockchain Ownership of Intellectual Property. Available online: http://www.yourtrainingedge.com/blockchain-ownership-of-intellectual-property/ (accessed on 24 February 2021).
31. Turkanovi, M.; Hölbl, M.; Koši, K.; Heriko, M.; Kamišali, A. EduCTX: A blockchain-based higher education credit platform. *IEEE Access* **2018**, *6*, 5112–5127. [CrossRef]
32. Durant, E.; Trachy, A. Digital Diploma Debuts at MIT. Available online: http://news.mit.edu/2017/mit-debuts-secure-digital-diploma-using-bitcoin-blockchain-technology-1017 (accessed on 24 February 2021).
33. del Castillo, M. Britain's Royal Mint Reveals Details on "Live" Blockchain for Tracking Gold. Available online: https://www.coindesk.com/britains-royal-mint-reveals-details-on-live-blockchain-for-tracking-gold/ (accessed on 24 February 2021).
34. Androulaki, E.; Barger, A.; Bortnikov, V.; Cachin, C.; Christidis, K.; De Caro, A. Hyperledger fabric: A distributed operating system for permissioned blockchains. In Proceedings of the Thirteenth EuroSys Conference, Porto, Portugal, 23–26 April 2018. [CrossRef]
35. Nicolas van Saberhagen, CryptoNode v 2.0, Monero White Paper. 2013. Available online: https://bytecoin.org/old/whitepaper.pdf (accessed on 24 February 2021).
36. David, S.; Youngs, N.; Britto, A. The Ripple Protocol Consensus Algorithm. Ripple Labs Inc. White Paper 5 2014. Available online: https://ripple.com/files/ripple_consensus_whitepaper.pdf (accessed on 24 February 2021).
37. Churyumov, A. Byteball: A Decentralized System for Storage and Transfer of Value. Available online: https://byteball.org/Byteball.pdf (accessed on 24 February 2021).
38. Dagcoin Whitepaper. Available online: https://dagcoin.org/whitepaper/ (accessed on 24 February 2021).
39. Baird, L. The Swirlds Hashgraph Consensus Algorithm: Fair, Fast, Byzantine Fault Tolerance. Available online: https://www.swirlds.com/downloads/SWIRLDS-TR-2016-01.pdf (accessed on 24 February 2021).
40. Braun, E.H.; Luck, N.; Brock, A. Holochain-Scalable Agent-Centric Distributed Computing. 2018. Available online: https://github.com/holochain/holochain-proto/blob/whitepaper/holochain.pdf (accessed on 24 February 2021).
41. Chen, J. Flowchain: A Distributed Ledger Designed for Peer-to-Peer IoT Networks and Real-Time Data Transactions. In Proceedings of the 2nd International Workshop on Linked Data and Distributed Ledgers, Portoroz, Slovenia, 29 May 2017.
42. The Coordicide. 2019. Available online: https://cdn0.tnwcdn.com/wp-content/blogs.dir/1/files/2019/05/Coordicide_WP.pdf (accessed on 24 February 2021).
43. Kosba, A.; Miller, A.; Shi, E.; Wen, Z.; Papamanthou, C. Hawk: The blockchain model of cryptography and privacy-preserving smart contracts. In Proceedings of the 2016 IEEE symposium on security and privacy (SP), San Jose, CA, USA, 22–26 May 2016. [CrossRef]
44. Azaria, A.; Ekblaw, A.; Vieira, T.; Lippman, A. Medrec: Using blockchain for medical data access and permission management. In Proceedings of the 2016 2nd International Conference on Open and Big Data (OBD), Vienna, Austria, 22–24 August 2016; pp. 25–30. [CrossRef]
45. Griggs, K.N.; Ossipova, O.; Kohlios, C.P.; Baccarini, A.N.; Howson, E.A.; Hayajneh, T. Healthcare blockchain system using smart contracts for secure automated remote patient monitoring. *J. Med Syst.* **2018**, *42*, 130. [CrossRef]
46. Elagin, V.; Spirkina, A.; Levakov, A.; Belozertsev, I. Blockchain Behavioral Traffic Model as a Tool to Influence Service IT Security. *Future Internet* **2020**, *12*, 68. [CrossRef]
47. Maurer, F.K.; Neudecker, T.; Florian, M. Anonymous CoinJoin transactions with arbitrary values. In Proceedings of the 2017 IEEE Trustcom/BigDataSE/ICESS, Sydney, Australia, 1–4 August 2017; pp. 522–529. [CrossRef]
48. Duffield, E.; Diaz, D. Dash: A Payments-Focused Cryptocurrency. 2018. Available online: https://github.com/dashpay/dash/wiki/Whitepaper (accessed on 24 February 2021).
49. Quorum Whitepaper. Available online: https://github.com/jpmorganchase/quorum-docs/blob/master/Quorum%20Whitepaper%20v0.1.pdf (accessed on 24 February 2021).
50. Goldreich, O.; Micali, S.; Wigderson, A. Proofs that yield nothing but the validity of their assertion. *Preprint* **1986**.
51. Miers, I.; Garman, C.; Green, M.; Rubin, A.D. Zerocoin: Anonymous distributed e-cash from bitcoin. In Proceedings of the 2013 IEEE Symposium on Security and Privacy, Berkeley, CA, USA, 19–22 May 2013. [CrossRef]

52. Ben-Sasson, E.; Chiesa, A.; Tromer, E.; Virza, M. Succinct non-interactive zero knowledge for a von Neumann architecture. In Proceedings of the 23rd {USENIX} Security Symposium ({USENIX} Security 14, San Diego, CA, USA, 20–22 August 2014; pp. 781–796.
53. Poelstra, A. Mimblewimble. Self-Published in October 2016. Available online: https://download.wpsoftware.net/bitcoin/wizardry/mimblewimble.pdf (accessed on 24 February 2021).
54. van Saberhagen, N. CryptoNode v 2.0, Monero White Paper. 2016. Available online: https://github.com/monero-project/research-lab/blob/master/whitepaper/whitepaper.pdf (accessed on 24 February 2021).
55. Garrick, H.; Rauchs, M. 2017 Global Blockchain Benchmarking Study. Available online: https://papers.ssrn.com/sol3/papers.cfm?abstract_id=3040224 (accessed on 24 February 2021).
56. Morrison, D.R. PATRICIA-practical algorithm to retrieve information coded in alphanumeric. *J. ACM* **1968**, *15*, 514–534. [CrossRef]
57. Nxt. Available online: https://nxtdocs.jelurida.com/Nxt_Whitepaper (accessed on 24 February 2021).
58. Counterparty. Available online: https://counterparty.io/docs/ (accessed on 24 February 2021).
59. Saia, R.; Carta, S.; Recupero, D.; Fenu, G. Internet of Entities (IoE): A Blockchain-based Distributed Paradigm for Data Exchange between Wireless-based Devices. In Proceedings of the 8th International Conference on Sensor Networks (SENSORNETS 2019, Prague, Czech Republic, 26–27 February 2019; pp. 77–84. [CrossRef]
60. Honar Pajooh, H.; Rashid, M.; Alam, F.; Demidenko, S. Hyperledger Fabric Blockchain for Securing the Edge Internet of Things. *Sensors* **2021**, *21*, 359. [CrossRef] [PubMed]
61. Palm, E.; Bodin, U.; Schelén, O. Approaching Non-Disruptive Distributed Ledger Technologies via the Exchange Network Architecture. *IEEE Access* **2020**, *8*, 12379–12393. [CrossRef]
62. Joseph, P.; Buterin, V. Plasma: Scalable Autonomous Smart Contracts. White Paper. 2017, pp. 1–47. Available online: https://plasma.io/plasma.pdf (accessed on 24 February 2021).
63. George, C.; Dollimore, J.; Kindberg, T. *Distributed Systems: Concepts and Design*, 3rd ed.; Addison-Wesley: Boston, MA, USA, 2001; p. 452, ISBN 978-0201-61918-8.
64. Huang, D.; Ma, X.; Zhang, S. Performance Analysis of the Raft Consensus Algorithm for Private Blockchains. *IEEE Trans. Syst. ManCybern. Syst.* **2020**, *50*, 172–181. [CrossRef]
65. Lamport, L. *The Part-Time Parliament, ACM Transactions on Computer Systems 16*. 1998. Available online: https://lamport.azurewebsites.net/pubs/lamport-paxos.pdf (accessed on 24 February 2021).
66. Diego, O.; Ousterhout, J. In search of an understandable consensus algorithm. In Proceedings of the 2014 {USENIX} Annual Technical Conference, Philadelphia, PA, USA, 19–20 June 2014.
67. Longo, R.; Podda, A.S.; Saia, R. Analysis of a Consensus Protocol for Extending Consistent Subchains on the Bitcoin Blockchain. *Computation* **2020**, *8*, 67. [CrossRef]
68. Pires, M.; Ravi, S.; Rodrigues, R. Generalized Paxos Made Byzantine (and Less Complex). *Algorithms* **2018**, *11*, 141. [CrossRef]
69. Castro, M.; Liskov, B. Practical Byzantine Fault Tolerance and Proactive Recovery. *ACM Trans. Comput. Syst.* **2002**, *20*, 398–461. [CrossRef]
70. Clement, A.; Wong, E.; Alvisi, L.; Dahlin, M.; Marchetti, M. Making Byzantine Fault Tolerant Systems Tolerate Byzantine Faults. In *Networked Systems Design and Implementation*; USENIX: Berkeley, CA, USA, 2009.
71. Aublin, P.-L.; Mokhtar, S.B.; Quéma, V. RBFT: Redundant Byzantine Fault Tolerance. In Proceedings of the 33rd IEEE International Conference on Distributed Computing Systems, Philadelphia, PA, USA, 8–11 July 2013. [CrossRef]
72. Abd-El-Malek, M.; Ganger, G.; Goodson, G.; Reiter, M.; Wylie, J. Fault-scalable Byzantine Fault-Tolerant Services. *ACM Sigops Oper. Syst. Rev.* **2005**, *39*, 59. [CrossRef]
73. Cowling, J.; Myers, D.; Liskov, B.; Rodrigues, R.; Shrira, L. HQ Replication: A Hybrid Quorum Protocol for Byzantine Fault Tolerance. In Proceedings of the 7th USENIX Symposium on Operating Systems Design and Implementation, Seattle, WA, USA, 6–8 November 2006; pp. 177–190, ISBN 1-931971-47-1.
74. Kotla, R.; Alvisi, L.; Dahlin, M.; Clement, A.; Wong, E. Zyzzyva: Speculative Byzantine Fault Tolerance. *ACM Trans. Comput. Syst.* **2009**, *27*, 1–39. [CrossRef]
75. Guerraoui, R.; Kneževic, N.; Vukolic, M.; Quéma, V. The Next 700 BFT Protocols. In Proceedings of the 5th European conference on Computer systems, Paris, France, 30 March–2 April 2010. [CrossRef]
76. Zhao, W. A Byzantine Fault Tolerant Distributed Commit Protocol. In Proceedings of the Third IEEE International Symposium on Dependable, Autonomic and Secure Computing (DASC 2007), Columbia, MD, USA, 25–26 September 2007; pp. 37–46. [CrossRef]
77. Markus, J.; Ari, J. *Proofs of Work and Bread Pudding Protocols, Communications and Multimedia Security*. 1999. Available online: http://www.hashcash.org/papers/bread-pudding.pdf (accessed on 24 February 2021).
78. Goldwasser, S.; Micali, S.; Rackoff, C. The Knowledge Complexity of Interactive Proof-Systems. Available online: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.419.8132&rep=rep1&type=pdf (accessed on 24 February 2021).
79. Sompolinsky, Y.; Zohar, A. Secure High-Rate Transaction Processing in Bitcoin. In *Financial Cryptography and Data Security*; Böhme, R., Okamoto, T., Eds.; Springer: Berlin/Heidelberg, Germany, 2015.
80. Cocco, L.; Pinna, A.; Marchesi, M. Banking on Blockchain: Costs Savings Thanks to the Blockchain Technology. *Future Internet* **2017**, *9*, 25. [CrossRef]
81. Kim, H.; Kim, K.; Kwon, H.; Seo, H. ASIC-Resistant Proof of Work Based on Power Analysis of Low-End Microcontrollers. *Mathematics* **2020**, *8*, 1343. [CrossRef]

82. Franco, P. *Understanding Bitcoin: Cryptography, Engineering and Economics*; Wiley: Hoboken, NJ, USA, 2014; ISBN 978-1-119-01916-9.
83. Vujičić, D.; Jagodić, D.; Ranđić, S. Blockchain technology, bitcoin, and Ethereum: A brief overview. In Proceedings of the 2018 17th International Symposium INFOTEH-JAHORINA (INFOTEH), East Sarajevo, Bosnia and Herzegovina, 21–23 March 2018; pp. 1–6. [CrossRef]
84. Biryukov, A.; Khovratovich, D. Equihash: Asymmetric Proof-of-Work Based on the Generalized Birthday Problem, Network and Distributed System Security Symposium. 2016. Available online: https://www.ndss-symposium.org/wp-content/uploads/2017/09/equihash-asymmetric-proof-of-work-based-generalized-birthday-problem.pdf (accessed on 24 February 2021).
85. Tromp, J. Cuckoo Cycle: A memory bound graph-theoretic proof-of-work. In *International Conference on Financial Cryptography and Data Security*; Springer: Berlin/Heidelberg, Germany, 2015; pp. 49–62.
86. GRIN. Available online: https://github.com/ignopeverell/grin (accessed on 24 February 2021).
87. AEternity. Available online: http://www.aeternity.com/ (accessed on 24 February 2021).
88. Yang, Z.; Yang, K.; Lei, L.; Zheng, K.; Leung, V.C.M. Blockchain-Based Decentralized Trust Management in Vehicular Networks. *IEEE Internet Things J.* **2018**, *6*, 1495–1505. [CrossRef]
89. CureCoin. Available online: https://www.curecoin.net/ (accessed on 24 February 2021).
90. Bentov, I.; Lee, C.; Mizrahi, A.; Rosenfeld, M. Proof of Activity: Extending Bitcoin's Proof of Work via Proof of Stake. *Acm Sigmetrics Perform. Eval. Rev.* **2014**, *42*, 34–37. [CrossRef]
91. Christina, J. DTB001: Decred Technical Brief. Available online: https://cryptorating.eu/whitepapers/Decred/decred.pdf (accessed on 24 February 2021).
92. Boni, K.R.C.; Xu, L.; Chen, Z.; Baddoo, T.D. A Security Concept Based on Scaler Distribution of a Novel Intrusion Detection Device for Wireless Sensor Networks in a Smart Environment. *Sensors* **2020**, *20*, 4717. [CrossRef] [PubMed]
93. Buterin, V.; Griffith, V. Casper the friendly finality gadget. *arXiv* **2017**, arXiv:1710.09437. Available online: https://arxiv.org/abs/1710.09437 (accessed on 24 February 2021).
94. Chen, J.; Micali, S. Algorand: A secure and efficient distributed ledger. *Theor. Comput. Sci.* **2019**. [CrossRef]
95. Buterin, V. Slasher: A Punitive Proof-of-Stake Algorithm. 2014. Available online: https://blog.ethereum.org/2014/01/15/slasher-a-punitive-proof-of-stake-algorithm/ (accessed on 24 February 2021).
96. Bentov, I. Cryptocurrencies without Proof of Work. 2017. Available online: https://fc16.ifca.ai/bitcoin/papers/BGM16.pdf (accessed on 24 February 2021).
97. Zhu, S.; Cai, Z.; Hu, H.; Li, Y.; Li, W. zkCrowd: A Hybrid Blockchain-Based Crowdsourcing Platform. *IEEE Trans. Ind. Inform.* **2020**, *16*, 4196–4205. [CrossRef]
98. Elrond. Available online: https://elrond.com/ (accessed on 24 February 2021).
99. Maofan, Y.; Dahlia, M.; Michael, R.; Guy, G.; Ittai, A. HotStuff: BFT Consensus with Linearity and Responsiveness. In Proceedings of the 2019 ACM Symposium on Principles of Distributed Computing, Toronto, ON, Canada, 29 July–2 August 2019; pp. 347–356. [CrossRef]
100. Kuhn, R.; Yaga, D.; Voas, J. Rethinking Distributed Ledger Technology. *Computer* **2019**, *52*, 68–72. [CrossRef]
101. Bitcoin: Maximum Transactions Rate. Available online: https://en.bitcoin.it/wiki/Maximum_transaction_rate (accessed on 21 January 2021).
102. Ehrsam, F. Scalability Ethereum to Billions of Users. Available online: https://medium.com/@FEhrsam/scalability-ethereum-to-billions-of-users-f37d9f487db1 (accessed on 24 February 2021).
103. Costa, C.H.; Vianney, B.M.; Filho, J.; Henrique, M.; Maia, P.; Carlos, M.B.; Oliveira, F. Sharding by Hash Partitioning—A Database Scalability Pattern to Achieve Evenly Sharded Database Clusters. In Proceedings of the 17th International Conference on Enterprise Information Systems, Barcelona, Spain, 27–30 April 2015; pp. 313–320, ISBN 978-989-758-096-3. [CrossRef]
104. Yu, G.; Wang, X.; Yu, K.; Ni, W.; Zhang, J.A.; Liu, R.P. Survey: Sharding in Blockchains. *IEEE Access* **2020**, *8*, 14155–14181. [CrossRef]
105. Chow, S.S.M.; Lai, Z.; Liu, C.; Lo, E.; Zhao, Y. Sharding Blockchain. In Proceedings of the 2018 IEEE International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData), Halifax, NS, Canada, 30 July–3 August 2018; p. 1665. [CrossRef]
106. Elrond-A Highly Scalable Public Blockchain via Adaptive State Sharding and Secure Proof of Stake. 2019. Available online: https://elrond.com/assets/files/elrond-whitepaper.pdf (accessed on 24 February 2021).
107. Luu, L.; Narayanan, V.; Zheng, C.; Baweja, K.; Gilbert, S.; Saxena, P. A secure sharding protocol for open blockchains. In Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security, Vienna, Austria, 24–28 October 2016; pp. 17–30. [CrossRef]
108. Kokoris-Kogias, E.; Jovanovic, P.; Gasser, L.; Gailly, N.; Syta, E.; Ford, B. Omniledger: A secure, scale-out, decentralized ledger via sharding. In Proceedings of the 2018 IEEE Symposium on Security and Privacy (SP), San Francisco, CA, USA, 20–24 May 2018; pp. 583–598. [CrossRef]
109. Zamani, M.; Movahedi, M.; Raykova, M. Rapidchain: Scalability blockchain via full sharding. In Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security, Toronto, ON, Canada, 15–19 October 2018; pp. 931–948. [CrossRef]
110. Wilkinso, S.; Boshevski, T.; Brandoff, J.; Prestwich, J.; Hall, G.; Gerbes, P.; Hutchins, P.; Pollard, C. Storj: A Peer-to-Peer Cloud Storage Network. Available online: https://storj.io/storj.pdf (accessed on 24 February 2021).

111. Benet, J. IPFS —Content Addressed, Versioned, P2P File Systems. Available online: https://ipfs.io/ipfs/QmR7GSQM93Cx5eAg6a6yRzNde1FQv7uL6X1o4k7zrJa3LX/ipfs.draft3.pdf (accessed on 24 February 2021).
112. Labs, P. Filecoin: A Decentralized Storage Network. Available online: https://filecoin.io/filecoin.pdf (accessed on 24 February 2021).
113. MediaChain. Available online: http://www.mediachain.io/ (accessed on 24 February 2021).
114. Sevcik, J. DECENT Whitepaper. 2015. Available online: https://www.allcryptowhitepapers.com/decent-whitepaper/ (accessed on 24 February 2021).
115. Vorick, D. Luke Champine, Sia: Simple Decentralizes Storage. Available online: https://sia.tech/sia.pdf (accessed on 24 February 2021).
116. Nick, L.; Ma, Q.; Irvine, D. Safecoin: The Decentralised Network Token. Maidsafe. *Tech. Rep.* **2015**. Available online: https://docs.maidsafe.net/Whitepapers/pdf/Safecoin.pdf (accessed on 24 February 2021).
117. Trón, V.; Fischer, A.; Nagy, D.A.; Felföldi, Z.; Johnson, N. Swap, Swear and Swindle Incentive System for Swarm. Available online: https://ethersphere.github.io/swarm-home/ethersphere/orange-papers/1/sw%5E3.pdf (accessed on 24 February 2021).
118. Arweave. Available online: https://github.com/ArweaveTeam/arweave (accessed on 24 February 2021).
119. Ozyilmaz, K.R.; Yurdakul, A. Designing a Blockchain-Based IoT With Ethereum, Swarm, and LoRa: The Software Solution to Create High Availability With Minimal Security Risks. *IEEE Consum. Electron. Mag.* **2019**, *8*, 28–34. [CrossRef]
120. Wang, S.; Li, G.; Yao, X.; Zeng, Y.; Pang, L.; Zhang, L. A Distributed Storage and Access Approach for Massive Remote Sensing Data in MongoDB. *ISPRS Int. J. Geo-Inf.* **2019**, *8*, 533. [CrossRef]
121. Back, A.; Corallo, M.; Dashjr, L.; Friedenbach, M.; Maxwell, G.; Miller, A.; Poelstra, A.; Timón, J.; Wuille, P. Enabling Blockchain Innovations with Pegged Sidechains. 2014. Available online: https://blockstream.com/sidechains.pdf (accessed on 24 February 2021).
122. Back, A.; Maxwell, G. Transferring Ledger Assets between Blockchains via Pegged Sidechains. U.S. Patent Application No. 15/150,032, 10 November 2016. Available online: https://patents.google.com/patent/US20160330034A1/en (accessed on 24 February 2021).
123. Fallis, A. Rootstock Platform: Bitcoin Powered Smart Contracts—White Paper. *J. Chem. Inf. Model* **2013**, *53*, 1689–1699.
124. Joseph, P.; Dryja, T. The Bitcoin Lightning Network: Scalable off-Chain Instant Payments. 2016. Available online: https://lightning.network/lightning-network-paper.pdf (accessed on 24 February 2021).
125. Raiden. Available online: https://raiden.network/ (accessed on 24 February 2021).
126. Bolt. Available online: https://boltlabs.tech/ (accessed on 24 February 2021).
127. Siris, V.A.; Dimopoulos, D.; Fotiou, N.; Voulgaris, S.; Polyzos, G.C. Decentralized authorization in constrained IoT environments exploiting interledger mechanisms. *Comput. Commun.* **2020**, *152*, 243–251. [CrossRef]
128. Provable. Available online: http://provable.xyz/ (accessed on 24 February 2021).
129. Jason, T.; Reitwießner, C. A Scalable Verification Solution for Blockchains. 2017. Available online: https://people.cs.uchicago.edu/~{}teutsch/papers/truebit.pdf (accessed on 24 February 2021).
130. WolframAlpha. Available online: https://www.wolframalpha.com/ (accessed on 24 February 2021).
131. Peterson, J. Augur: A Decentralized Oracle and Prediction Market Platform. *arXiv* **2015**, arXiv:1501.01042. Available online: https://arxiv.org/abs/1501.01042 (accessed on 24 February 2021).
132. Haas, A.; Rossberg, A.; Schuff, D.L.; Titzer, B.L.; Holman, M.; Gohman, D.; Bastien, J.F. Bringing the web up to speed with WebAssembly. In Proceedings of the 38th ACM SIGPLAN Conference on Programming Language Design and Implementation, Barcelona, Spain, 18–23 June 2017.
133. Zyskind, G.; Nathan, O. Decentralizing privacy: Using blockchain to protect personal data. In Proceedings of the 2015 IEEE Security and Privacy Workshops, San Jose, CA, USA, 21–22 May 2015; pp. 180–184. [CrossRef]
134. Enigma- Testnet. Available online: https://github.com/enigmampc?language=javascript (accessed on 24 February 2021).
135. AHrga; Capuder, T.; Žarko, I.P. Demystifying Distributed Ledger Technologies: Limits, Challenges, and Potentials in the Energy Sector. *IEEE Access* **2020**, *8*, 126149–126163. [CrossRef]
136. Li, D.; Wong, W.E.; Guo, J. A Survey on Blockchain for Enterprise Using Hyperledger Fabric and Composer. In Proceedings of the 2019 6th International Conference on Dependable Systems and Their Applications (DSA), Harbin, China, 3–6 January 2020; pp. 71–80. [CrossRef]
137. The Interledger Protocol. Available online: https://interledger.org/rfcs/0027-interledger-protocol-4/ (accessed on 24 February 2021).
138. Le, D.; Yang, G.; Ghorbani, A. A New Multisignature Scheme with Public Key Aggregation for Blockchain. In Proceedings of the 17th International Conference on Privacy, Security and Trust (PST), Fredericton, NB, Canada, 26–28 August 2019; pp. 1–7. [CrossRef]
139. Rajan, D.; Visser, M. Quantum Blockchain Using Entanglement in Time. *Quantum Rep.* **2019**, *1*, 2. [CrossRef]
140. Otsuki, K.; Banno, R.; Shudo, K. Quantitatively Analyzing Relay Networks in Bitcoin. In Proceedings of the 2020 IEEE International Conference on Blockchain (Blockchain), Rhodes Island, Greece, 2–6 November 2020; pp. 214–220. [CrossRef]
141. Dai, W.; Deng, J.; Wang, Q.; Cui, C.; Zou, D.; Jin, H. SBLWT: A Secure Blockchain Lightweight Wallet Based on Trustzone. *IEEE Access* **2018**, *6*, 40638–40648. [CrossRef]
142. DriveChain: Enabling Bitcoin Sidechains. Available online: http://www.drivechain.info/ (accessed on 24 February 2021).