Interoperable Distributed Ledger Technology

State of the Art Analysis

July 2020

Table of Contents

Introduction of DLTs

Distributed ledger technology (or a distributed ledger, blockchain, DLT) is a type of distributed database that has the following notable features:

- Distributed participation
- Decentralization
- Distributed consensus
- Public-Private key cryptography

In a DLT, every participant shares a replica of the network's transaction history. The information is updated to all nodes in near-real-time. However, due to network latencies, the arrival orde of these transactions to each node can be different order. This is why DLTs require advanced consensus protocols to agree upon the latest, common state of the database. This common state cannot be maliciously modified by a single party and even by small adversarial coalitions. Furthermore, it can be verified by reconciling one version of the database to another that is kept on a separate node. In short, unlike centralized solutions, there is no single point of failure for a DLT.

The consensus in a blockchain can take many forms including; Proof of Work, Proof of Stake, Proof of Authority, practical Byzantine Fault Tolerance, Single Authority, etc. Each form of consensus ensures transactional accuracy agreeable among network participants with varying degrees of assurance. This contrasts with traditional databases; whereby inputted information is assumed accurate until subsequently reviewed.

Public-private key cryptography allows participants to transact pseudonymously. To achieve this, the public key of a user is used to generate her address on a Blockchain. Transactions are sent to and from these addresses. Private keys and digital signatures provide authenticity for each transaction. That is only the owner of a digital asset or token can access and modify its state. In short, public-private key cryptography provides the necessary and sufficient security properties for the transactions on blockchain networks.

Because of their unique characteristics, such as establishing trust without a trusted third-party, DLTs are being used for a plethora of applications, such as cloud/fog computing, Internet of Things (IoT), data storage, network management, and digital content distribution. These foundational features are cornerstones on solving security (DoS attacks, collusion attacks, access control, confidentiality), privacy (anonymity, integrity), and trust (data credibility assessment) issues. Bitcoin as the first application and the first example of a common digital currency provides a solution to the lack of confidence in a decentralized, independent monetary system. It chronologically registers all valid transactions auditable by all network peers without

human intervention. In the context of Smart Grid, Gao et al. proposed a blockchain with smart contracts for creating a tamper-proof system, avoiding inconsistencies between electricity companies and consumers regarding electricity usage and bills [1]. For healthcare, Guo et al. introduced an attribute-based signature scheme to implement a blockchain-based electronic health system [2]. By using threshold secret and function sharing, the signature scheme can resist N-1 corrupted authorities' collusion attacks where N is the total number of authorities. The scheme is unforgeable in suffering a selective predicate attack. Throughput, latency, security, wasted resources, usability, multiple chains have been identified in [3] as the technical challenges and limitations for using blockchain. Aste et al. [7] conclude that the Blockchain has opened up new opportunities for businesses without intermediaries or central control points.

Conclusion 1: DLT can be applied in various environments with the potential to change the way transactions are conducted, provide benefits, such as security and privacy, and lower management costs.

Conclusion 2: DLT suffers from technical limitations and challenges. When applied in different environments, the throughput, latency, security, etc. issues of DLT must be re-considered.

Overview of DLT

There exist various DLT solutions currently in use differing in terms of read/write permissions, consensus algorithms, transaction latency, and throughput, security assumptions, etc. For instance,

- The Aion network [21] is a multi-tier blockchain designed to support a future where many specialized blockchains exist. The Aion protocol enables the development of a federated blockchain network, making it possible to seamlessly integrate dissimilar blockchain systems in a multi-tier hub-and-spoke model.
- Ethereum, developed by Vitalik Buterin, is using its own currency ETH. By providing a virtual machine as a secure environment, distributed Apps can be executed on the machines of voters. This enables the implementation of smart contracts.
- IOTA is a distributed ledger for the IoT. It represents a novel machine-to-machine communication suitable for industrial applications. It is based on multidimensional Directed-Acyclic-Graph technology.
- The Hyperledger-Project enables the users to develop their own blockchain implementations. Frameworks and tools that are tailored for specific applications, like bonds, financing, or digital identities were developed based on Ethereum Virtual Machine.

A traditional classification of DLT technologies based on ledger access and data validation policies is given in Table 1.

Permissioned		Permissionless	
Private	**Public**	**Private**	**Public**
Only members can validate and read	Only members can validate but data is open to everyone	Only pre-defined members can read and write	Every user can join and validate transactions
Ex: Hyperledger, R3 Corda	*Ex: Ripple*	*Ex: Test network of Ethereum, Bitcoin*	*Ethereum, Bitcoin*

Table 1: Categorization of Distributed Ledgers

Table 1 categorizes the DLTs in two ways; permissioned/permissionless ledgers and private/public ledgers. A **public** ledger is public in the sense that all the participants have read access to the stored data. On the other hand, for a **private** ledger, a single party (or a coalition) controls who will have this access. In a **public permissionless** ledger, all the participants can also be involved in the consensus protocols and validate the transactions. On the contrary, for a **public permissioned** ledger, only a set of predefined/permissioned participants can validate. Still, as mentioned above, all participants can read all the transactions. When the number of users involved in a consensus is large, and when the identities of the participants are unknown (as in public permissionless ledgers), consensus becomes expensive yielding less transaction throughput and more resource usage for consensus. However, especially for distributed ledgers among a few institutions, using a (consortium) private ledger is the natural solution. Here, performance and scalability problems of DLTs can be addressed more easily, since the network is more trusted than public and permissionless ledgers.

DLTs are recently supported by Cloud providers to be used as a database alternative for applications running on the Cloud: Microsoft offers Hyperledger, Ethereum, and Corda for Azure, Amazon has Blockchain as a Service, Oracle has Distributed Ledger in his cloud, SAP offers Leonardo blockchain [20]. As can be seen from the variety and richness of DLT support on Cloud, developers have many ledger alternatives with different characteristics. They are free to choose the one satisfying their applications' requirements while only providing the necessary level of security and trust and using the necessary amount of resources. From a single developer's point of view, this is natural to keep the application scalable. Unfortunately, from the beginning, a DLT is designed and proposed to work alone. It solves a unique, single problem (as establishing a cryptocurrency). Hence, it is hard for a ledger-based application to connect to the outside world and other applications running on different ledgers while keeping the same security guarantees its own ledger provides. Unfortunately, this restricts potential use-cases and exploitation of the full potential of DLTs.

Conclusion 3: Several platforms providing Blockchain have recently been developed. Thus, *interoperability* among these platforms put forward another challenge. In addition, with the increase of participants and ledgers, scalability becomes another issue to be considered when applying DLT.

Blockchain, IoT and Industry 4.0

The internet of things (IoT) is a technological phenomenon whereby devices, machines, objects, or even people are connected to the internet via unique identifiers (UIDs). These connected devices can automatically communicate with each other over the internet, without the need for human, manual interaction. In the context of an enterprise, IoT has significant implications when it comes to transparent tracking and tracing voluminous or complex interactions between automated processes. Blockchain is the underlay that records all transactional data in an immutable, auditable distributed ledger. Data is accessible by any stake-holding party whose connected device is part of the value chain. Several recent papers, blogs, and press reports describe the challenges, potentials, and use cases of using blockchains in combination with IoT [4, 6-9, 12].

Christidis et al. show that blockchains, as distributed and resilient peer-to-peer systems with their associated ability to interact with peers in a trust-less and verifiable manner, are generally suitable for IoT purposes [4]. The authors also give a summary of possible services between devices. It is possible for example to use micro-payment (Bitcoin or Ethereum) to enable the devices to rent disk space or to monetize API calls.

Conclusion 4: Without DLT, the data from all IoT interactions would be recorded and siloed by the party hosting each connected device. In the world of highly integrated supply chains, where stakeholders are numerous, sharing of complete data is paramount.

Conoscenti et al. introduce a list of 18 use cases of blockchains documented in different literature divided into the categories “Data storage management”, “Trade of goods and data”, “Identity management”, “Rating system” and “Other” [5]. By means of a Systematic Literature Review, the authors try to spot the main factors that affect the levels of integrity, anonymity, and adaptability of blockchains. It was found out that large blockchains systems like Bitcoin are most secure. At the same time, Bitcoin scalability issues make it poorly suitable for IoT. Also, only pseudonymity is guaranteed in the blockchain and no anonymity. The authors plan to test further different blockchains to find a solution suitable for IoT, in which the compromise between scalability and security is acceptable.

Huckle et al. discuss the use of IoT and blockchains to create secure, shared-economy distributed applications (Dapps) [6]. Uber and Airbnb are examples that belong to the shared economy.

Dorri et al. use the blockchain/bitcoin technology to secure a smart home environment [8]. Each smart home has a typical gateway component that acts as a miner and potentially also as a cluster head within an overlay compound. The smart home miner device handles its own private, secure blockchain, governing all internal and external communications between local storage, cloud storage, and smart devices. The paper shows the applicability of blockchain for Smart Home to secure transactions between the cloud store, service provider, and homeowner and the IoT devices. A comparison with established systems does not emerge from the paper.

Fremantle and Scott address mainly middleware blockchain is only partly addressed, but the authors conclude: “Blockchains are cryptographically secure ledgers that typically require a significant amount of memory, disk space and processor power to work [10]. These requirements go beyond typical IoT devices and even beyond more powerful systems in IoT networks such as hubs. One option to address this is to use remote attestation, but as yet there is little or no work in this space.”

Huh et al. use Ethereum to configure devices and manage public keys via blockchain [6]. They show a proof of concept for a small amount of IoT devices (air conditioner, temperature sensor, light, meter).

Esposito et al. deal with “the potential to use the Blockchain technology to protect healthcare data hosted within the cloud” [15]. They propose a distributed patient data store. They addressed the right-to-erasure and the storage of large data in the blockchain.

Fujitsu presented on Hannover Fair a demonstration of its envisioned smart factory of the future. According to [17], they have implemented IOTA DLT into their product portfolio and its IoT-Suite IntelliEdge [16]. Furthermore, four other partnerships that use IOTA in context with IoT, e.g. world’s first IOTA charging station and an automated Order Controlled Production Process, are outlined [17].

The tutorial from Chainskills shows the possibility to set up a private Ethereum blockchain on an IoT environment [18].

Currently lacking in the blockchain, IoT space is a tamper-proof digital replication of physical assets. Inherent in providing digital identifiers to physical assets, there are significant threats to accuracy. QR codes or other such identifiers can be replicated and applied to multiple physical assets, they can be rendered unreadable from wear and tear, they can be lost, they can be replaced with fraudulent identifiers, etc. However, the technology exists that when integrated with blockchain and IoT, users can ensure accuracy in physical-digital pairings.

Conclusion 5: It has been shown that the direct use of DLT is not practical for IoT. Further research work is needed to remove the limitations on the use of Blockchain in the IoT environment.

Conclusion 6: In the literature, many use case descriptions highlighting the advantages of using DLT in IoT applications. However, there are only a few publications that present real implementations and examine performance at the same time. Further research work is needed that shows concrete quantified advantages and evaluated demonstration.

Blockchain and Smart Contracts

A Smart contract (SC) is the set of actions that are self-executing without third parties, based on electronic contracts between network participants. It is a method that utilizes Blockchain technology to create contracts between two or more participants. When the terms described in the contracts are fulfilled, the terms will be executed by the Blockchain system. Practically various algorithms and functions can be activated and executed upon the reception of the fulfillment of contract terms. Smart contracts can be used in various environments and systems for different purposes. The blockchain's transparency offers a way to enforce the contract, therefore, benefits including low contracting, enforcement, and compliance costs can be achieved. Currently, smart contracts are being considered for a wide variety of uses, particularly for regulatory compliance, product traceability, service management, defeating counterfeit products, and fraud. For example, the function of the smart contract is to identify malicious usage of electrical power [4]. The consumer data being manipulated maliciously on the smart grid network will trigger the smart contract to send an encrypted message to the smart meter and displayed on the screen of the smart meter of the consumer.

Bitcoin itself only has limited SC support (non-Turing complete scripting). Nick Szabo presented the first approach in 1996 and Ethereum firstly implemented it in practice 2013 for an electronic payment system. Roman Beck discusses the potential of blockchains using the Ethereum blockchain platform [14]. Currently, Ethereum serves as the basis for today's implementations [13]. This implementation has the following features:

- self-tracking fulfillment of predefined requirements,
- decision making based on predefined algorithm,
- signable by human and machine.

Specialized sandbox styled programming languages for smart contracts are Solidity (similarities to C and JavaScript), Serpent (similar to Python), LLL (a Low-level Lisp-like Language), Mutan (Go-based language), Viper (a strongly-typed Python-derived decidable language) [13].

There exist papers in the literature that describe and discuss the use of blockchain technology for smart contracts [11, 13-14]. Ethereum first implemented the smart contract in practice for an electronic payment system using virtual machines. Another approach to creating smart contract support is using containers and integrating it with the internal API of the Blockchain (BC) platform. Instead of developing a specialized programming language with a custom virtual machine, regular programming languages like Go, Java, C# can be used with containerized

platforms. In this scenario, Docker containers create a safe environment for smart contracts which they can call internal API of the BC platform, so they can make operations on BC. Every supported language will have a proxy server, which translates functions into internal BC platform API calls. This approach cannot be said to be safer than the sandbox or virtual machine approach, but it is very practical specifically for private ledgers. The most popular example of this approach is Hyperledger Fabric, which is the most supported open source private ledger platform.

SC provides increasing network fault-tolerance and increasing autonomy of the Cyber-Physical Systems (CPS) participants. But there are some drawbacks like:

- privacy/confidentiality issues due to public contract details
- insufficient size of production networks compared to the global P2P network
- smart devices lack computing power for Nakamoto's consensus (proof of work)
- raised minimum cost of equipment to deal with cryptography
- IoT devices lack storage capacity for complete transaction log
- transaction-costs in Ethereum

Thus, using public systems is not profitable. Fortunately, there exist alternative solutions in the literature [13]; firstly, the use of private Blockchain solutions, secondly the use of specialized BC like https://iota.org/, and thirdly a private Ethereum BC in conjunction with the “Proof of Stake” consensus.

Magazzeni et al. introduce smart contracts in finance and government [11]. While the contract moves from natural language to formalized code, different questions for validation and verification are further discussed in the publication. Zhang and Wen describe the transaction of a smart property and paid data on the IoT by means of Blockchain and smart contracts [12].

Conclusion 7: Smart contracts rely on the computing system on which they execute. Security and trust for executing the smart contracts should be guaranteed. Especially when smart contracts involve multiple systems, how to provide a secure and trust environment is a challenge.

DLT and Interoperability

Today, there are thousands of blockchain platforms. Each serves its own use-case – from digital currencies to provenance tracking in supply chains. Each of these solutions operates in its siloed ecosystem. Different platforms cannot communicate.

Conclusion 8: Without interoperability, a robust web3.0 is impossible to achieve as value and information cannot seamlessly federate on-chain. Off-chain conversion re-introduces centrality, which contradicts blockchain's central tenet – decentralization.

Today, the protocols Aion/Mavennet, Cosmos, Polkadot, ICON, and Wanchain are developing solutions at the cutting edge of blockchain interoperability. Interoperability is the catalyst that will enable broad commercial adoption via increased scalability and transaction throughput. The following are some notable benefits to interoperability:

- Enable ecosystem applications such as identity, payment, and storage to interact across multiple blockchain platforms
- Enable enterprises to link public and private networks to optimize their cost, privacy. and security
- High-performance computing by spanning out workflows to fit-for-purpose blockchains
- Decentralized exchange of native coins and tokens across multiple blockchain platforms
- Assets/coins that outlive the network in which they were created

However, the following challenges complicate seamless communication via bridges and channels among heterogeneous networks:

- Bridges introduce longer finality time
- Different blockchains have different architectural designs (Bitcoin - 6 blocks, approx. 1-hour confirmation time. Aion - 90 blocks, approx. 15-minute confirmation time)
- Transaction signing is complex (Different networks use different cryptographic curves)
- Bridges need to be more secure but allow for more transaction throughput than the networks which they connect
- Pricing disparity between tokens trading on their native network and the same token on an external network

Conclusion 9*:* As of today, the problem of building a universal bridge remains unsolved.

Interoperability Approaches

Interoperability between different chains can generally be done in three ways. These are *notary schemes*, *relay schemes* and *hash-locking schemes*:

1. **Notary Schemes:** A notary scheme is a structure that validates a transaction on Ledger A and proves it to Ledger B. The group that performs these operations acts as a reliable notary. This notary structure is formed by multiple nodes and runs a consensus in itself.

2. **Relay (Sidechain) Schemes:** The smart contract on any ledger can read, validate, or act according to the incoming state or event on other ledgers. The contract can read information since part of the other ledger is stored in the ledger where the smart contract is located. It implies that there is no need for 3rd party interfaces; ledgers can talk directly among themselves. There are two types of relays; one-way and two-way.

    a. *One-way relays:* Ledger A can read the data on Ledger B; however, B cannot read A's. For example, BTC-Relay is a smart contract on Ethereum and can read data in Bitcoin; however, Bitcoin cannot read the data in Ethereum. For this reason, it is one-way.

    b. *Two-way relays:* Ledger A and B can read data mutually. For example, when Alice wants to buy something from the market with BTC and the merchant only uses ETH, if there is a two-way relay between the Ethereum and Bitcoin networks, Alice makes the payment and get the cashback in Bitcoin while the merchant takes ETH and give ETH as a cashback.

3. **Hash-Locking:** In Relay schemes, it is necessary to partially store the data of the other ledger. However, in hash-locking, just hash sharing is enough. Let's say that Alice wants to send ETH to Bob, and receive BTC from Bob. Alice sends her ETH through a smart contract, which is locked with a hash whose secret input is only known by Alice. Bob locks his BTC transaction using the hash he sees in the smart contract. When Alice sees this, she unlocks it using her hash lock's secret input. When Bob sees this, he unlocks the ETH that is sent to him. Bitcoin and Ethereum's timeout/timelock capabilities can be used to avoid infinite locks.

Polkadot

Blockchain scalability is an important problem, and platforms like Polkadot are attempting to become the next generation of networks that serve enhanced scalability and interoperability by expanding public blockchain design concepts and standardizing data transfer. Specifically, the three areas current blockchain ecosystems are struggling to deliver practical applications are i*nteroperability, scalability*, and *shared security*. Polkadot is a heterogeneous multi-chain framework built to promote the interoperability and scalability of blockchains contained in the 'Relay Network.' It leverages a form of proof-of-stake (PoS) consensus for the wider ecosystem of blockchains linked to it. To do this, it enables data structures to connect as 'parachains' which operate via a confidence-minimized federation structure.

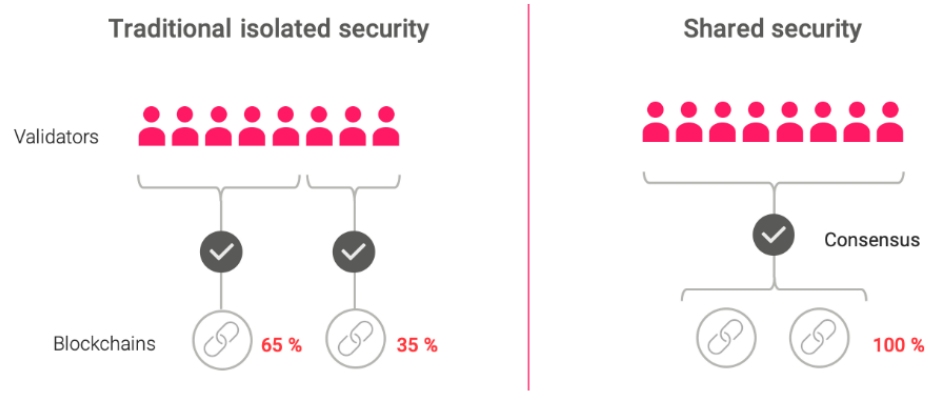

Fig. 1: Pooled security vs. Traditional isolated security [59]

The difference between the traditional, isolated security model and Polkadot's shared security model can be seen in Figure 1. Polkadot employs a central relay chain through which the parachains link and organize the consensus, as well as data and the messages in between are transferred. Notably, both public and permissioned blockchains can connect to the network, with the ability of permissioned chains to isolate themselves from the rest of the system while still retaining the ability to transfer data to other chains and leverage the security properties. For pooled security and interoperability with other chains, parachains can be blockchains or other data structures that plug into the relay chain. However, to be compatible with the Polkadot network **they must be able to form fast, compact client proofs** and **there must be a method for authorizing a transaction for a large number of independent authorities**.

Although they are the parts of the same relay network, parachains process their own transactions and this enables the network to scale based on the simultaneous independent processing of transactions per parachain. This process and the parachains are secured by the wider network consensus that is heavily influenced by Tendermint and HoneyBadgerBFT, but uses PoS as the primary way of motivating validators to be truthful in the network. However, the relay chain can also create 'bridges' with other chains which have their own consensus.

The Polkadot protocol's lower layers (the Wasm interpreter, consensus, and networking) are known as the *Polkadot Runtime Environment*, and are common throughout the network's parachains. On the other hand, each connected parachain has a unique set of upper layers. *Substrate* is the first implementation of the Polkadot Runtime Environment (PRE). Parachains will be written using the PRE, which is built on the Web3 technology stack. An important aspect of Polkadot is that it uses the networking stack of Libp2p and is the first real-world use of its implementation of Rust.

The dynamics of how Polkadot works are complex, so the best way to visualize the platform is through the ecosystem's four primary participant roles: a Polkadot network has four basic roles to maintain: *collator*, *fisherman*, *nominator*, and *validator*. In one possible implementation of Polkadot, the latter role can in fact be broken down into two roles: *basic validator* and *availability guarantor*.

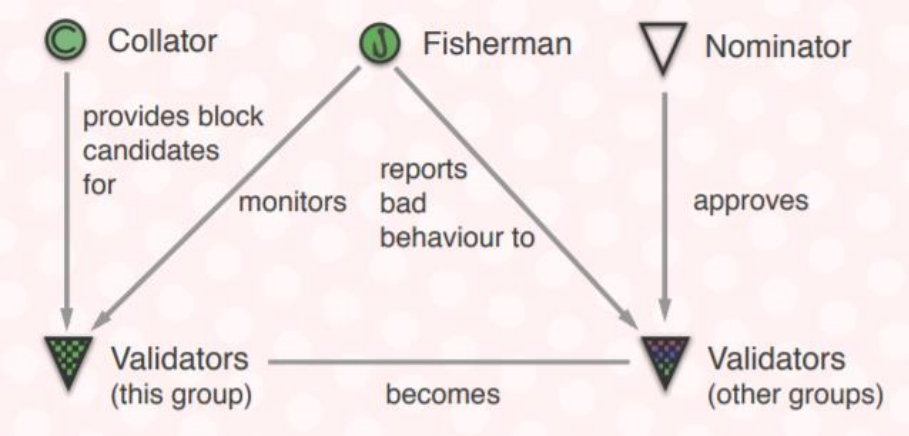

Fig. 2: Interaction of roles [58]

- **Validators** finalize blocks in the Polkadot ecosystem and are expected to run the complete relay chain application. A substantial 'bond' (in the native DOT token) must be staked to be a validator. Validators may also appoint other validators to act in their place. Validators receive candidate blocks from collators — who distribute selected blocks from parachains to validator subgroups — and finalize the blocks on the relay chain via a deterministic selection phase and final validation round of ratification.

- **Nominators** are parties that also hold a stake in the network, but function as a mechanism to select trusted validators by contributing their bond to the bond of a selected validator. Their job is very straightforward and helps strengthen the relay chain's pooled security.

- **Collators** work on the parachain level, rather than securing the relay chain directly. They collect parachain transactions, produce a proof along with an unsealed block, and send it to the appropriate validator charged to finalize a parachain block. The Polkadot white paper notes that collators' role may evolve, and they may eventually be contracted to

work closely with specific validators to check blocks from certain parachains. Collators can also work as an added layer of security to prove malicious conduct to validators on the network. The general role of collators is similar to the work of miners in PoW blockchains.

- **Fishermen** are independent of the process of block verification and look for malicious behavior on the network that they report about bad validators to validators. We are encouraged to pursue significant one-off incentives as 'bounty-hunters' by demonstrating that a bonded group (i.e., validator or collator) behaved maliciously beyond the rule set. Fishermen post small bonds to the network too, however. This is to prevent Sybil attacks, but is not nearly as high as validators and can be withdrawn at any point.

Cosmos

Cosmos aims to become an “internet of blockchains” to solve the scalability and interoperability problems once and for all. Cosmos’ architecture consists of several independent blockchains called “*Peg Zones*” attached to a central blockchain called “*Hub*”. That is there will be multiple parallel blockchains (peg zones) connected to one central Hub. The hub itself is a distributed ledger where individual users can keep their tokens, or the peg zones themselves. Using IBC (Inter Blockchain Communication) the peg zones will communicate with each other through the Hub. The basic layouts of the Cosmos architecture and ecosystem are given in Figure 3.

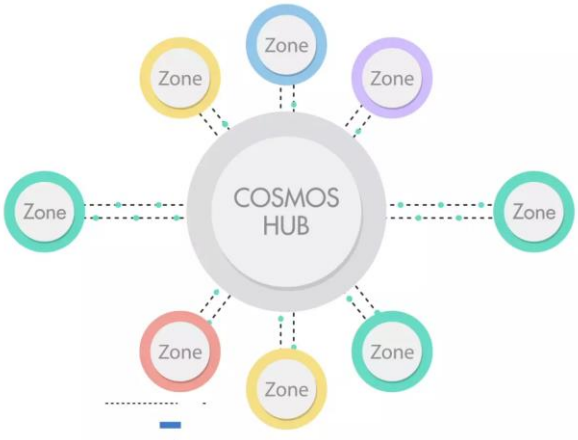

Fig. 3: Cosmos structure [60]

As the Hub plays such a critical role in the Cosmos blockchain network, it is extremely necessary to ensure its security. Because of that, a globally decentralized group of validators secures it. Similar to Polkadot, the heterogeneous chains can have a unique set of laters. That is the chains can differ in networking, consensus and application implementation. Hence, multiple consensus protocols can coexist in a Cosmos network. That being said, each chain must be compatible with IBC. Hence, the consensus layer of each chain must have fast finality (protocols with probabilistic finality such as PoW do not have this property).

The Cosmos blockchain is maintained by a set of validators who agree on the next block to commit. A *sovereign* blockchain is a blockchain with its own validator set. Usually, it is important for a chain to be sovereign; the validators are solely responsible for modifying the state. Since IBC allows the chains in the peg zones to transfer data/tokens, these chains having different applications and validator sets can be interoperable.

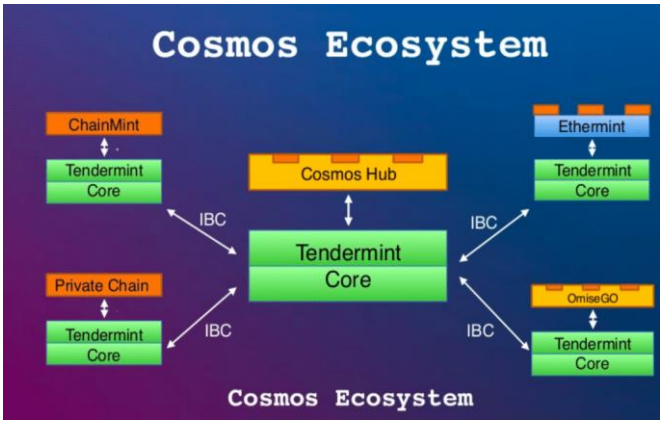

Fig. 4: Cosmos Ecosystem [61]

Wanchain

Wanchain claims to be the infrastructure connecting the decentralized financial world. It aims to provide deposit and loan services with cryptocurrencies. When a Wanchain transfer request is initiated, it issues and locks tokens on the target blockchain with an existing smart contract. The validators verify that the transaction is correctly placed. Then a new smart contract token, analogous to the original currency is created. When this set of tokens is required to send to another party, the locked tokens are released and the *exchange* of the assets are completed. The ecosystem contains *vouchers*, *proof nodes*, *validators*, *verifiers* and the *storeman*.

Vouchers check the confirmation of the transactions on the source side and validators verify the above transaction is correctly placed, i.e., the asset is (created if new), registered and locked. After the exchange operations, the total number of coins in each blockchain remains constant as it is transferred from one chain to another, The Storeman nodes use an innovative, secure multi-party calculation method with a threshold-protected secret key to process the cross chain. For all transactions between chains, Storeman nodes create a locked account. These accounts can hold funds from the source blockchain forever. This causes the same value to be present in the target blockchain as mapping token. Original funds are released only when mapping tokens are returned to the original chain. Of course, mapping tokens are destroyed.

Ark

ARK is a fast, decentralized, scalable, collaborative system that enables building bridges between blockchains. Transactions on the network are around eight seconds, much faster than Bitcoin transactions, which takes about an hour. ARC was derived from Lisk, Crypti and BitShares and added new technologies to its core. It uses the Delegated Proof-of-Stake consensus algorithm. The ecosystem is expected to link blockchains together to create a blockchain network for any number of applications. The platform guarantees this through smart bridges.

Smart Contract Interoperability: A hashed timelock contract (HTCL) is a smart contract that allows transactions to be executed based on time. The recipient of a transaction must approve the payment with cryptographic evidence within a given time. Otherwise, the transaction will not take place. Cross-chain atomic exchange between cryptocurrencies is provided by HTCL. HTCL transactions use multiple signatures for verification and validation. Although this aspect is similar to existing cryptocurrency systems, there are two elements that it does not. The first one is *hashlock* which was mentioned above in this section. The second important element is *timelock* which serves to control transactions depending on time. Two different timelocks are used to set time constraints on HTCLs. The first one is CheckLockTimeVerify (CLTV). This is timelock that locks tokens based on time and releases them when the time comes. The second is CheckSequenceVerify (CSV). Locks and releases tokens based on the number of blocks, not time. This may create a synchronization problem between blockchains with different block production speeds. It should be noted that if HTCL is going to be used, one must open a channel between the chains.

HyperLedger Cactus

HyperLedger Cactus is a blockchain integration tool and one of the Hyperledger projects hosted by The Linux Foundation. The tool was designed to address the fragmentation and lack of interoperability between different distributed ledgers and allow users to securely integrate different blockchains.

Most importantly, Cactus is designed to be a pluggable architecture that would help enable the execution of ledger operations across multiple blockchain ledgers. Currently, the project supports integration with other HyperLedger frameworks such as Besu, Fabric, Corda and Quorum with the aim of adding support for integration with other blockchains as more and more ledger technologies are developed and/or enhanced

Hyperledger cactus will enable ledger operations to be executed across multiple blockchain ledgers by providing integrated services. These operations will be controlled by the Business Logic Plugin which will be provided by vendors as a module. This relates to the design goal of HyperLedger Cactus being a pluggable tool that can be changed and edited according to the desired needs of the user. The HyperLedger Cactus architecture has been specifically designed to enable users to integrate external blockchain platforms with HyperLedger cactus to form a communication bridge between said ledgers and overcome the fragmentation issues that are prevalent in blockchain technologies today. The tool will provide plugins which will be designed to provide a simple interface which can then be edited and/or changed according to the requirements of the user without the need for changes in the core code.

When the user requests an API call to the Cactus framework, the business logic plugin will determine the ledger operations that should be executed and cactus protocols ensure the reliability of the integrated service and its results are as expected. The diagram below summarizes the high-level architecture of HyperLedger Cactus:

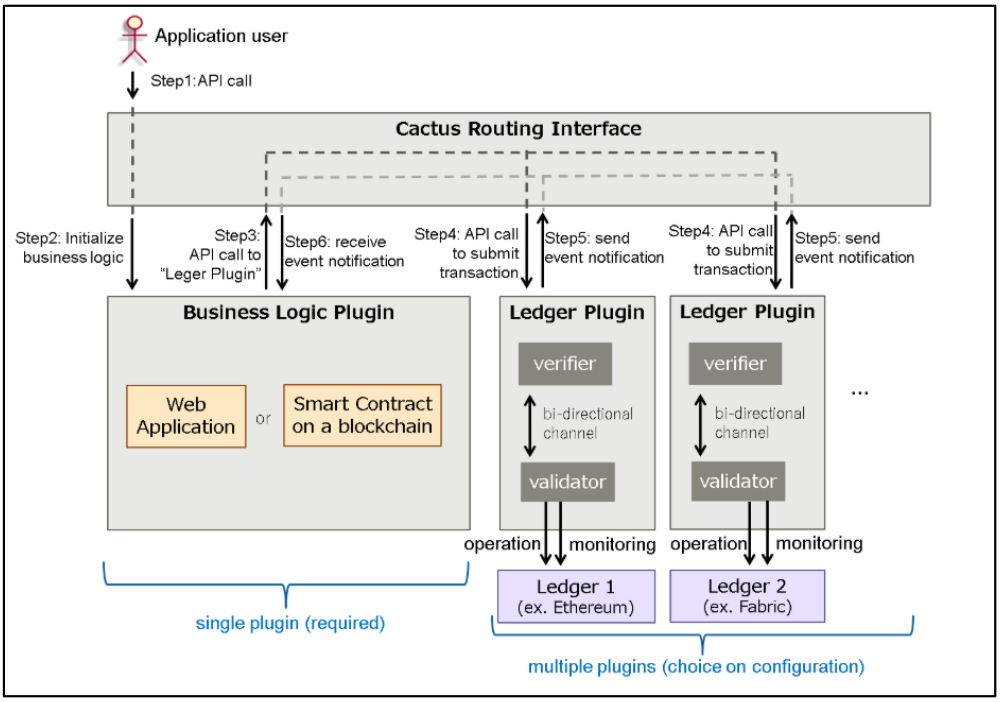

Fig. 5: HyperLedger Cactus structure [63]

Key Components pertaining to the architecture are:

- **Application user:** Submits API calls to "Cactus Routing Interface"
- **Business Logic Plugin:** Executes business logic and provides integration services that are connected with multiple blockchains.
- **Ledger Plugin:** Responsible for communicating the business logic plugin with each ledger. Consists of a validator and verifier to carry out successful communication of the business logic with the Business Logic plugin
- **Validator:** Valdiators are responsible for monitoring the transaction records of Ledger Operations and are responsible for determining the result (success, fail, timeout) from said transaction records. They are also responsible for ensuring the obtained result by attaching digital signatures with a “Validator Key” which are verified by the *Verifier.*

- **Verifier:** Responsible for verifying the digital signature of the validator and accepting successfully verified operations.
- **Cactus Routing Interface:** Routing service between "Business Logic Plugin" and "Ledger Plugin(s)".

As mentioned previously, HyperLedger Cactus provides a pluggable architecture where the different distributed ledgers can communicate with each other via API calls to plugins. By establishing a plugin architecture, cactus is able to provide an abstraction layer on top of the core components that allows operators of Cactus to swap out implementations as needed.

Cactus implements plugins as EcmaScript 6 (ES6) modules which can be loaded from the persistent data store at run time. Validation of code signatures to guarantee source code integrity is the responsibility of the core package.

The key plugins that make up the plugin architecture are:

- Ledger Connection Plugin:
  - Provide support for integration of ledgers without the need for core code changes
  - Provide feature checks that allow users to determine programmatically if the ledger in question supports some specific feature at runtime.

- X509 Certificate Plugin
  - Removes the need for operating authentication tokens by allowing users to authenticate themselves via “certificates”.
  - Extends passportJS by allowing the obtaining of CA certificates from validator nodes at runtime.

- Key/Value Storage Plugin
  - Enable high-level packages to store and retrieve configuration metadata for a Cactus cluster
  - Differentiate between active validators and hosts which allow access to such validators over the network.
- Server Side KeyChain Plugin
  - Extends the Key/Value Storage plugin functionality

When it comes to authentication protocols, Cactus bridges the gap between web applications and blockchain applications by providing a built-in OpenID Connect (OIDC) identity provider and server side key chain. Both of these authentication protocols are leveraged by Cactus to allow end users to authenticate just once against Cactus and manage their identities on other blockchains/ledgers through that single HyperLedger Cactus identity.

This innovation stems from the fact that traditional web applications mostly rely on a centralized authority storing hashed passwords (e.g: logging in with a password) and blockchain networks work with private keys belonging to a PKI to authenticate users. By leveraging both OIDC protocols and Cactus's server-side key chain protocols, Cactus is able to overcome the divide between the two worlds and allow for greater usability. This feature also allows for greater security for web applications which do not have secure offline storage APIs.

Example: A user of a Hyperledger Cactus account can import their private keys from their Fabric/Ethereum wallets and then have access to all of those identities by authenticating once only against Cactus which will result in a server-side session (HTTP cookie) containing a JSON Web Token (JWT).

Interoperability Case Studies

Use Case - 1: Deloitte connected two non-compatible blockchain platforms Ethereum and Hyperledger Fabric with two non-blockchain platforms Singapore Exchange (SGX) and Monetary Authority of Singapore using the combination of API-based approach and trusted agent-based approach (oracle). They focused on the reducing turnaround time of the delivery-versus-payment (DvP) process to lower the risk of counterparties and diminish the required liquidity in the targeted ecosystem.

To find an effective solution to the interoperability issue they executed delivery leg on the permissioned Hyperledger Fabric platform and used a digital currency named as Ubin, which is backed with one SGD (Singapore Dollar) to run on crypto-enabled Ethereum platform.

To overcome the integration challenges of permissioned and permissionless blockchain platforms Deloitte used the smart contract of Hyperledger Fabric to trigger payment at the Ethereum network upon the change of title of the securities. The SGX server first shares a secret with the seller to lock and validate the ownership of the securities on Hyperledger Fabric. After this process SGX server generates a different secret for buyers to lock their payment on Ethereum. An event triggered smart contract will swap two generated secrets with buyer and seller simultaneously to enable them to unlock and receive securities and payment respectively.

With the benefit of this approach the need for intermediates such as custodians has been eliminated to reduce the counterparty risk [28].

Use Case - 2: EVRYTHNG Product Cloud created a powerful and scalable orchestration layer between a growing number of leading blockchain protocols and solutions. It's an approach which extends the smart capabilities of digitized products with the decentralized features of

integrated blockchains. They enabled data about products (e.g. supply chain history, consumer scans, temperature, current owner, etc.) to be replicated to, or collected from, different blockchains. This might be supply chain history, live tracking data or consumer engagement.

EVRYTHNG offers an API gateway called Blockchain Integration Hub to resolve supply-chain integrity issues and enables new direct-to-consumer applications triggered by end customers scanning products with their smartphones. To do this they transformed a physical item into a digital object that exists and interacts on the web. This approach made it possible to create scannable and interactive physical objects and given software intelligence to participate in new applications [28].

Use Case - 3: Interoperability in healthcare has traditionally been focused around data exchange between business entities, for example, different hospital systems. However, there has been a recent push towards patient-driven interoperability, in which health data exchange is patient-mediated and patient-driven. Patient-centered interoperability, however, brings with it new challenges and requirements around security and privacy, technology, incentives, and governance that must be addressed for this type of data sharing to succeed at scale.

Using a public or private blockchain to actually store clinical data is one example—for example, Yue et al. described a “Healthcare Data Gateway” (HDG) which would enable patients to manage their own health data stored on a private blockchain. Similarly, Ivan described a public blockchain implementation, where healthcare data is encrypted but stored publicly, creating a blockchain-based Personal Health Record [62]. MedChain is another example, where a permissioned network of medication stakeholders (including the patient) could be used to facilitate medication-specific data sharing between patients, hospitals, and pharmacies [62]. While we imagine that a model storing actual clinical data on a blockchain—permissioned or public—would have substantial privacy and scalability concerns, it is important to continue to understand the privacy and security implications of on-chain data storage.

In the healthcare space, FHIRChain is a smart-contract based system for exchanging health data based on the standard FHIR [62], where clinical data is stored off chain, and the blockchain itself stores encrypted meta-data which serve as pointers to the primary data source (like an EHR) [62]

Use Case - 4 : Ethereum to Quorum Asset Transfer - Hyperledger Cactus is to offer value transfer between two different blockchain ledger technologies. For example, a user can have assets stored in Ethereum ledger. But now, he wants to exchange it for assets on the Quorum ledger. In general circumstances without the exchanger solution, the user needs to sell his Ethereum assets and then buy the Quorum assets using the money. But, that’s not possible for all types of assets.To solve that particular issue, Hyperledger Cactus offerS Escrowed Asset Transfer social interaction. This interaction is important as it will give the user the flexibility to

choose the blockchain ledger of his choice.In this Hyperledger Cactus use case, the user once he transfers the assets from Ethereum will lose it from there. However, the asset will now be available on the Quorum ledger. Also, to make it work, the ledgers need to be provisioned and the exchanger should have its identities established before the exchange takes place.

Open Standards for Blockchain Interoperability

We are aware of standardization efforts of the following standardization organizations which are on specific blockchain topics as summarized below [23]:

- IEEE Blockchain Initiative [24] is mainly focusing on agriculture, medicine, and IoT.
- ISO/TC 307 technical committee [25] is working on distributed ledger technologies
- W3C community group [26] is working on the web Ledger Protocol which will be used to serve the protocol and format of the decentralized ledgers on the web

There are also studies like [27 -28] which are focusing on how open standards can be developed.

Recent report [28] is about the interoperability of supply chains which are enterprise permissioned based solutions. Some of these efforts on the interoperability can pave the way to standardization [28]:

- Blockchain Industrial Alliance (BIA): BIA works on cross-blockchain transactions and interconnectivity.
- Digital Container Shipping Association (DCSA): DCSA works for the interoperability in the container shipping industry.
- European Blockchain Partnership (EBP): EBP is working on a European Blockchain Services Infrastructure (EBSI) which will connect countries and be used for the delivery of the cross-border digital public services
- Enterprise Ethereum Alliance (EEA): EEA aims to develop open blockchain specifications for worldwide interoperability of the businesses.

Conclusion 10: We are not aware of an open standard but there are some initial standardization efforts on blockchain interoperability which is given in the section.

Security and Privacy on Blockchain

Privacy vs Security

The main security services provided by blockchain is integrity, availability, and fault tolerance. Privacy is not provided by design in many implementations [30]. However, privacy is usually a misunderstood subject. Most of the analyses take the cryptocurrency implementations into consideration. For these DLTs, there is the transparency of records as public ledgers are used in these implementations. However, these public records also do not keep any personal or private data. Indeed, such data should not be kept in private ledgers. Furthermore, it is not a common practice, especially in enterprise DLT implementations. Different implementations have different levels of privacy [29, 31]. Studies on privacy protection in blockchain are summarized in [29].

The problems of privacy and security are exacerbated with interoperability. As mentioned before, different DLTs may have different design rationales and characteristics. Hence, the privacy level or the decision of keeping the transactional data secret may differ from one DLT to another. Interoperability, enabling multiple DLTs working together, requires an utmost concern and advanced cryptographic techniques to guarantee the privacy and security levels of each DLT on common ground without wasting resources. This is why understanding the current problems, as well as the solutions in detail is important from the interoperability point of view.

Problems and Challenges

There are privacy challenges in collecting and storing personal data; PII (personally identifiable information) and PHI (Protected health information). The users should have trust in the system that their privacy is preserved. International laws like GDPR (General Data Protection Regulation of the EU - EU General Data Protection Regulation 2016/679) and national laws like KVKK (“Kişisel Verileri Koruma Kanunu” of Turkey) have strict regulations on the privacy of the citizens. The software developers of such systems are also responsible for these regulations [32]. There are challenges in satisfying Individuals' rights:

- Transparency and information
- Access, rectification, erasure
- Objection
- Automated individual decision-making
- Portability

There can also be restrictions on processing and exceptions to rights. The main challenges can be given as follows [33]:

- The approval of the share: The user should have the means to control the personal data and should know how that data is shared.
- Security of the collected data: The organizations are responsible for the security of the collected data. The data should not be revealed (when compromised) or sold to other parties.

- The right to be forgotten: The user can ask for the removal of the personal data and there should be some means to accomplish that.

Even when direct personal information that can be linked to a real identity is not stored on a DLT, depending on the use-cases, privacy still becomes an important issue. The pseudonyms, i.e., the DLT addresses provide limited privacy protection which can be defeated by using techniques such as transaction network analysis and behavioral analysis. Furthermore, regulations (e.g., anti-money-laundering) and policies such as KYC make these techniques easy to apply. For instance, when the user once uses his/her ledger address through an internet connection, the server can link the IP address with the DLT address. While sending a transaction to the DLT network, the broadcast and relay information can be used for the same task. In the literature, there are studies on the possibility of breaking the underlying P2P structure and decentralized protocols via classical attack patterns such as Denial-of-Service [41] and Sybil [42] attacks.

For better privacy protection, one can use multiple addresses, i.e., multiple pseudonyms. Even when multiple pseudonyms are used by the same person, the transaction network can be analyzed via advanced ML (e.g., clustering) techniques to link these addresses to each other. This analysis can further be improved by using behavioral patterns such as transaction time, day, hour, amount, etc. Especially for cryptocurrencies where *fungibility* is an important concern, privacy is not only crucial for keeping the identities secret, but also it is necessary to keep the value of each token the same. For instance, there is a chance for tokens produced/transferred by a transaction funded from an unethical/illegal action/source to be denied by some markets, authorities, etc. This will lessen the tokens' convertibility and hence, their value. It will also damage the internal structure and affect the integrity of the corresponding cryptocurrency.

Comentario [1]: ?

None of the above-mentioned problems are related to the secrecy of the data. DLT transactions form the network. They establish relations between the addresses and pseudonyms. This will happen regardless of the readability level of the information they carry. However, for some applications, one may want only the desired parties to read the data stored on the ledger including the assets (e.g., BTC, ETH, ALGO balances, retirement fund amount, financial information stored on transactions) and history (previous supply-chain steps, agreement details, electronic health records, etc.). Indeed, when the data is not intended to be used for validation purposes, it is a good practice not to store such data in the ledger. Instead, those data can be kept in encrypted form on legacy systems and the cloud [33]. However, when validation is necessary they must be processed by the smart contracts. Furthermore, they must be in an encrypted form and still be processable and verifiable. For instance, one should not be able to transfer an asset (e.g., crypto tokens) she does not own, and this must be verified by the third parties in DLT who do not know she had it.

Current Solutions and Approaches

As stated above, regardless of the data being encrypted or not, transactions form a network whose topology can be exploited via machine learning tasks leveraged to link multiple pseudonyms as well as real-life identities. The best practice to avoid this is using each address only once. Although this solves many linkability problems, frequent payment patterns can still leak information when advanced techniques are used. The DLT can be more resistant to such techniques via practical solutions such as *mixing* [43]. A mixing service works as a post-office; each message/transaction, containing the data and a recipient address is encrypted by the mixing service's public-key and sent. The mixing service, who can decrypt the message, acts as a relay node, and directs the transaction to its real recipient. Assuming that the service has a large number of users and hence receiving many messages, it is easy to obfuscate the output message order so that it is not mappable to the input order.

In a DLT, the implementation of an effective and efficient mixing service is not an easy task. The service must be *secure* (no theft, double spending, DoS resistance), *provide a good level of anonymity* (large user base, unbiased randomness), *deniable* (plausibly deniable, no cryptographic evidence), *scalable* (cost-efficient, can support a large number of users), and *prevent misuses* (e.g., money laundering). Various mixing protocols have been proposed and employed for cryptocurrencies. The main idea is transferring the tokens to the mixing service addresses first and let the service either relay them to the actual recipient or provide a voucher to be redeemed later.

- In *centralized mixing*, all the mixing protocol is controlled by a trusted third-party. Such a scheme has many risks including *counterparty risk*, i.e., the mixer can steal funds, *logging risk,* i.e., the mixer can log the transaction details, centralization risk, i.e., the mixing service is a single point of failure, can be hacked, controlled by an adversary, etc. Mixcoin [44] and Blindcoin [45] are two examples of centralized mixing protocols. The former adds a signature-based accountability mechanism to expose theft so that users can unambiguously prove if the mixer has misbehaved. Besides, the latter uses the idea of blinding to keep the input-output address mapping hidden from the mixing service.
- In *altcoin exchange mixing*, the tokens, e.g., BTC, first sent to an exchange and spent to buy a different token, e.g., ETH. These tokens are then sent to another exchange and the operation is repeated multiple times. In the last step, the original token, e.g., BTC, is bought. The difference between the initial amount and the final one, which is spent as exchange fees, is the cost of mixing service. Although this process is plausibly deniable (just a set of financial transactions) and seems to be untraceable due to the involvement of many exchanges, a powerful authority can still link the addresses to real identities due to the KYC regulations.
- *Fair-exchange mixing* leverages advanced cryptographic fair-exchange protocols to avoid coin theft. In such a service, each asset can be fairly exchanged with a voucher

which later can be redeemed. A fair-exchange protocol is a multi-party protocol where parties accept to deliver an item if and only if they receive an item in return. Assuming such a service is used by many users at a time, finding the input-output mapping of the received and sent assets becomes hard. Although such schemes have stronger guarantees, it is not straightforward to implement them since they require a large amount of computation and a sufficient amount of asset transfer (e.g., liquidity for the assets are cryptocurrencies). TumbleBit [46] and Xim [47] are two examples of centralized fair-exchange mixing protocols.

- *Decentralized mixing protocols* try to remove counterparty risks and avoid fees by taking out the middleman (the centralized mixer). The idea is creating a network of peers who cooperate to make transactions that mix the assets, without relying on a trusted third party. *CoinJoin*, for instance, exploits the fact that a Bitcoin transaction can have multiple inputs and outputs [48]. Thus, although the anonymity set (i.e., the set of possible output addresses for a single input) is usually small, a single transaction can be used for mixing purposes. That is one of its inputs may be said to be directed to any of its outputs. Furthermore, the multisignature support (i.e., signing by multiple parties) in Bitcoin scripting can make multiple participants be in control of the process. Hence, the participants can avoid the counterparty risks of centralized mixing services. A participant only signs the transaction when it is correct, i.e., when one of the outputs is the desired output for the participant. However, without having private and anonymous communication channels for submitting the output addresses, the CoinJoin protocol is vulnerable to traffic analysis. Furthermore, since multi signature transactions with a large number of participants are not frequent in Bitcoin, these transactions are not plausibly deniable. Also, the protocol is not DoS-resistant since even a single participant can stop the protocol if s/he chooses not to sign her/his part. The CoinJoin idea is currently employed by cryptocurrencies such as *DASH* with extensions [49]. Another mixing protocol, *CoinShuffle* borrows the idea of using multisignatures and enhances it by a privacy-preserving, cryptographic shuffling protocol [50]. The protocol simply makes every participant shuffle the output addresses provided by the previous participants without seeing them. Hence, they cannot be mapped to input addresses.

Another way to hide the addresses from curious adversaries is a ring signature [50]. A ring signature is created using a group of keys formed by the participant's key, and a number of other public keys chosen by her/him. When the transaction is broadcasted, a third party can verify that it is signed by one of the private keys corresponding to the public keys in the group. However, the third party cannot know which key is used. That is a ring signature hides the sender/source of a transaction among a group of possible senders. Ring signatures are being used by popular privacy-focused cryptocurrencies such as Monero [51] along with other countermeasures such as one-time addresses.

As protecting the privacy of the participants, keeping the data secret is also a hard problem. As mentioned before, the read access to the data can be restricted by storing the details of the transactions "off-chain". That is another isolated system, database, storage, etc. can be used to store the plain data where only the *hash* (a short, compressed form of the data obtained via a one-way function with cryptographic guarantees) of the transaction details are kept "on-chain". This is a common practice since the transaction details cannot be obtained from the hash functions. In fact, if the data is not crucial for verification purposes, e.g., personal details of BTC users, this is believed to be the best practice. However, when "off-chain" storage must be used for verification and validation, this is not a straightforward task. Indeed, when required, a participant with read-privileges on the isolated storage can check whether the hash is generated from the correct transaction data. However, "off-chain" storage duplicates the "sources" of truth which is the main responsibility of a DLT. How the isolated storage is maintained is another problem; having a trusted-third party totally negates the benefits of a DLT. Hence, each party can keep and manage her/his own transaction data and grant access privileges to the required parties. Unfortunately, this is also against the fundamental motivations of using a DLT. This is why cryptographic techniques are mostly used to have confidential transaction support on blockchains [32]. These techniques, which have been mostly proposed for other applications, can be summarized as follows:

- **(Fully) Homomorphic Encryption** (HE) schemes enable both multiplication and addition operations directly over encrypted data without decryption. This means that any computation that can be expressed either as an arithmetic or logic (Boolean) circuit can be performed over encrypted data. For instance, let **E(4)** and **E(5)** be the ciphertexts for the values 4 and 5, obtained by a HE scheme with the encryption function **E**. Then one can have **E(9) = E(4 + 5)** or **E(20) = E(4 x 5)** without performing any decryption operations and performing the arithmetic on the range of the **E**, i.e. the encrypted domain. The use of HE on a distributed ledger unleashes exciting opportunities. For instance, with HE, a blockchain-based cryptocurrency can be implemented while hiding all the account balances from the public since each transaction can be performed by modifying the account balances and without revealing them.

  The first FHE scheme is proposed by Craig Gentry [35] followed by more practical schemes such as BGV, FV, CKKS, and TFHE [36-39]. While TFHE implements fast "bootstraping", which supports fully homomorphic operation; today, most homomorphic encryption schemes provide, in fact, "somewhat" homomorphic encryption (SWHE) capability, which simply means the number of homomorphic operations that can be applied on ciphertext is limited and the decryption is not possible if the limit, which is often referred as the *noise budget*, is exceeded. In particular, when a plaintext data is encrypted, the corresponding ciphertext contains a noise term, which increases after every homomorphic operation, i.e., addition or multiplication, over the ciphertext. Homomorphic multiplication is much more costly than homomorphic addition in terms of

the noise budget. Fortunately, for many applications such as cryptocurrencies and digital asset management, addition is the only, if not, the most frequent arithmetic operation used. That being said, homomorphic computation is practicable for all (arithmetic) circuits with low multiplicative depth and high-degree of parallelism. Therefore, there is an interest in the literature for research on novel (parallel) algorithms resulting in low-depth circuits for efficient homomorphic computation, even for rudimentary operations such as sorting [40] which can enlighten new avenues for the use of HE in DLT-based applications.

- **Secure Multi-Party Computation** (SMC) leverages advanced cryptographic primitives and protocols for the secure computation of functions over private inputs of two or more parties. At the end of the protocol, parties learn their outputs and nothing else except for what can be learned in the *ideal world*. In a possible application, the inputs to SMC are (multiple) data items stored on a DLT, or inputs coming from several participants who may not know and trust each other. These inputs will be processed to obtain a final result that will be revealed to the participants or to the public.

  In theory, the participants of an SMC protocol can be modeled as semi-honest, malicious, and covert adversaries. These models are closely related to the public/private, permissioned/permissionless characteristics of the underlying DLTs. For instance, a ***semi-honest adversary*** is a participant that follows the protocol but tries to learn from the exchange messages. Such a participant can exist even in private DLTs. In fact, one must assume that such participants **always** exist. A ***malicious party*** can behave in arbitrary ways to learn more about other parties' inputs. These participants exist and create security issues in permissionless DLTs. However, even for permissioned DLTs, one needs to have a monitoring and identification mechanism/tool for malicious actions. Once this exists, the read/write accesses of the participants who are detected to act maliciously can be revoked.

  SMCs that are secure against malicious parties are much harder to construct and more resource-demanding than those against semi-honest adversaries. As a compromise between these two extremes, covert adversaries are allowed to cheat but get caught by honest participants with some probability. If this probability is sufficiently large, the cheating party is deterred as it is reluctant to lose its reputation. SMC protocols rely on oblivious transfer~(OT) protocols and FHE. With a carefully selected adversary model depending on the application requirements, DLTs with various security guarantees can be implemented for decentralized applications.

- **Zero-Knowledge Proofs (ZKP)** are used when someone wants to prove the correctness of a statement without revealing any other information other than the statement is correct. In a DLT, especially when assets/tokens/etc. are transferred from

one address to another, the ownership of a sufficient amount of resources must be verified. With zero-knowledge proofs, the sender can prove that s/he has enough resources without revealing her/his actual balance, i.e., the amount of resources s/he has. Formally, a zero-knowledge proof has three important properties:

- *Completeness:* The verifier will be “always” convinced if the statement is correct.
- *Soundness:* The verifier will "never" be cheated if the statement is not correct.
- *Zero-Knowledge:* The verifier will not learn anything else except whether the statement is correct or not.

Although the zero-knowledge proofs are usually designed as interactive protocols, techniques such as Fiat-Shamir heuristic [52] are frequently used to make them as non-interactive. These protocols, which have many applications in practice, are called Non-Interactive ZKPs (NIZKs). However, due to the computation required to generate and/or verify such proofs, NIZKs can incur significant computational overheads and can be a burden to implement a scalable system. For DLTs having many transactions, this limits the use of traditional NIZKs in the literature. This is why many researchers have been trying to reduce the proving and verification complexity of NIZKs.

In the literature, the terms SNARGs (succinct non-interactive adaptive arguments) and SNARKs (succinct non-interactive adaptive argument of knowledge) have been used to formally define efficient and computationally sound proofs (of knowledge). A succinct, e.g., short, cheap in communication and computation, NIZK is, therefore, a SNARK and called as zk-SNARK. The use of zk-SNARKs in Bitcoin has been discussed by Ben-Sasson in 2013 [53] and the Zerocash protocol, which is also based on zk-SNARKs, is proposed in 2014 to have decentralized anonymous payments [54]. Bulletproofs, which can prove that a committed value is in a range with a logarithmic complexity of the range length, are proposed in 2017 [55]. Furthermore, in 2018, the *zk-STARK* protocol was introduced which uses a setup with no trusted parties, quasi-linear proving time, and poly-logarithmic verification time [56]. Thanks to these attempts, these cryptographic marvels have been currently in use in cryptocurrencies such as Zcash, which is based on the Zerocash protocol and zk-SNARKs [57].

Link to previous and/or current collaborative research projects:

Project Name	CooperativeProgramme	Time period (approx.)	Technical Focus	Relationship
SealedGRID	H2020	2018-2021	Combine technologies like Blockchain, Distributed Hash Tables, Trusted Execution Environments, and OpenID Connect for Smart Grids.	BEIA will bring experience for scalable, trusted, and interoperable platform for secure smart grid using blockchain as transaction and security layer
SAFECARE	H2020	2018-2021	Safeguard of critical health infrastructure	BEIA is working on using blockchain for securing hospitals against ransomware

BROS	H2020	2017-2020	Use blockchain to solve robotics issues: data confidentiality, decision design,	Using blockchain technology and swarm robotics systems, BROS will generate new models for solving the issues.
BILLON	H2020	2017 - 2019	It is proposed a distributed ledger technology for creating free current accounts enabling making ultra-low-cost payments with real currencies (EUR, GBP, PLN) in a regulated manner.	Their distributed architecture developed on blockchain for FinTech
CHARIOT	H2020-EU.2.1.1	2018 - 2020	Provides a design method and cognitive computing platform and unites Privacy, Security and Safety (PSS) of IoT Systems	How they implemented blockchain for IoT.

AnticipatoryLedgers	H2020-EU.1.3.2	2018 - 2020	Provides design and ethical framework for distributed ledgers	Take-up policy analysis
PRIViLEDGE	H2020-EU.2.1.1	2018 - 2020	PRIViLEDGE realizes cryptographic protocols supporting privacy, anonymity, and efficient decentralized consensus for distributed ledgers	Encryption for DLT
BLOCKCHAINSOCIETY	ERC-2017-STG	2018-2022	The Blockchain Society project focuses on three research questions. (1) What internal factors contribute to the success of a blockchain application? (2) How does society adopt blockchain? (3) How to regulate blockchain?	Methods on how to regulate blockchain and mitigate the impact in society

BLOOMEN	H2020-ICT-2016-2	2017-2020	The main goal of the Bloomen proposal is to extend the use of blockchain technology to handle different online user transactions, provide an innovative way of content creation, sharing, personalized consumption, monetization and copyrighting.	How blockchains will be used as a distributed database for media copyright information, for fast micropayments of media content, and for transparency in copyright management and monetization.
PTwist	H2020-ICT-2017-1	2018-2019	PTwist aims to design, deploy, and validate an open platform which will twist plastic reuse practices, by boosting citizens' awareness, circular economy practices, and sustainable innovation in line with the new plastics economy vision.	How to use DLT, open source, blockchain, gaming, Crowdsourcing components, open data solutions and developments to the largest possible extent.

DLInnociate	H2020-INNOSUP-02-2016	2017-2018	Blockchain technology for secured Real Time Economy in cloud, both in general for providing applications to any client and for piloting the DEEP, Digital Enterprise Ecosystem Platform business plan on open innovation with a client in the FinTech	Blockchain in FinTech
Ontario Digital Identity Blockchain Application	Small Business Innovation Challenge (SBIC) – Ontario Centres for Excellence	2017-2018	Built a single-source blockchain based digital identity wallet that focuses on privacy and convenience. Product consists of features that allow for more efficient sharing of identity data between citizens	Blockchain for Identity

References

[1] J. Gao, K. O. Asamoah, E. B. Sifah, A. Smahi, Q. Xia, H. Xia, X. Zhang, and G. Dong, "GridMonitoring: Secured sovereign blockchain based monitoring on smart grid," IEEE Access, vol. 6, pp. 9917–9925, 2018.

[2] R. Guo, H. Shi, Q. Zhao, and D. Zheng, "Secure Attribute-Based Signature Scheme With Multiple Authorities for Blockchain in Electronic Health Records Systems," IEEE Access, vol. 6, pp. 11 676–11 686, 2018

[3] M. Swan, Blockchain: Blueprint for a New Economy, O'Reilly Media, Inc., 2015.

[4] K. Christidis and M. Devetsikiotis, "Blockchains and Smart Contracts for the Internet of Things," *IEEE Access*, vol. 4, pp. 2292–2303, 2016.

[5] M. Conoscenti, A. Vetro, and J. C. De Martin, "Blockchain for the Internet of Things: A systematic literature review," in *2016 IEEE/ACS 13th International Conference of Computer Systems and Applications (AICCSA)*, Agadir, Morocco, 2016, pp. 1–6.

[6] S. Huckle, R. Bhattacharya, M. White, and N. Beloff, "Internet of Things, Blockchain and Shared Economy Applications," *Procedia Computer Science*, vol. 98, pp. 461–466, 2016.

[7] T. Aste, P. Tasca, and T. Di Matteo, "Blockchain Technologies: The Foreseeable Impact on Society and Industry," *Computer*, vol. 50, no. 9, pp. 18–28, 2017.

[8] A. Dorri, S. S. Kanhere, R. Jurdak, and P. Gauravaram, "Blockchain for IoT security and privacy: The case study of a smart home," in *2017 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops)*, Kona, Big Island, HI, USA, 2017, pp. 618–623.

[9] S. Huh, S. Cho, and S. Kim, "Managing IoT devices using blockchain platform," in *2017 19th International Conference on Advanced Communication Technology (ICACT)*, Pyeongchang, Kwangwoon Do, South Korea, 2017, pp. 464–467.

[10] P. Fremantle and P. Scott, "A survey of secure middleware for the Internet of Things," *PeerJ Computer Science*, vol. 3, p. e114, May 2017.

[11] D. Magazzeni, P. McBurney, and W. Nash, "Validation and Verification of Smart Contracts: A Research Agenda," *Computer*, vol. 50, no. 9, pp. 50–57, 2017.

[12] Y. Zhang and J. Wen, "The IoT electric business model: Using blockchain technology for the internet of things," *Peer-to-Peer Networking and Applications*, vol. 10, no. 4, pp. 983–994, Jul. 2017.

[13] M. Y. Afanasev, Y. V. Fedosov, A. A. Krylova, and S. A. Shorokhov, "An application of blockchain and smart contracts for machine-to-machine communications in cyber-physical production systems," in *2018 IEEE Industrial Cyber-Physical Systems (ICPS)*, St. Petersburg, 2018, pp. 13–19.

[14] R. Beck, "Beyond Bitcoin: The Rise of Blockchain World," *Computer*, vol. 51, no. 2, pp. 54–58, Feb. 2018.

[15] C. Esposito, A. De Santis, G. Tortora, H. Chang, and K.-K. R. Choo, "Blockchain: A Panacea for Healthcare Cloud-Based Data Security and Privacy?," *IEEE Cloud Computing*, vol. 5, no. 1, pp. 31–37, Jan. 2018.
[16] FUJITSU IoT Solution INTELLIEDGE https://www.fujitsu.com/emeia/products/computing/pc/edge-computing/
[17] Chris Mueller, How IOTA is enabling Industry 4.0, https://medium.com/@iotasuppoter/how-iota-is-enabling-industry-4-0-b867564f57a3
[18] Chainskills, Tutorial, Create a private Ethereum blockchain with IoT devices http://chainskills.com/2017/02/24/create-a-private-ethereum-blockchain-with-iot-devices-16/
[19] C. Khan, A. Lewis, E. Rutland, C. Wan, K. Rutter, and C. Thompson, "A Distributed-Ledger Consortium Model for Collaborative Innovation," *Computer*, vol. 50, no. 9, pp. 29–37, 2017.
[20] J. Seeger, "Sicher verkettet- Hyperledger-Implementierungen im Vergleich," *iX Magazin*, no. 07/2018, pp. 44–48, 2018.
[21] M. Spoke, "Aion: Enabling the decentralized Internet," AION, White Paper, Jul. 2017. https://aion.network/media/en-aion-network-technical-introduction.pdf
[22] T. N. Dinh and M. T. Thai, "AI and Blockchain: A Disruptive Integration," *Computer*, vol. 51, no. 9, pp. 48–53, Sep. 2018.
[23] Karaarslan, E., Konacaklı, E. "Data Storage in the Decentralized World: Blockchain and Derivatives". "Who Run The World: DATA" Book. Istanbul University Press, 2020 (in press)
[24] IEEE, 2019. IEEE Blockchain Standards. https://blockchain.ieee.org/standards
[25] ISO, 2019. ISO/TC 307 technical committee on blockchain and distributed ledger technologies, https://www.iso.org/committee/6266604.html
[26] W31C, 2019.The Web Ledger Protocol 1.0, Draft Community Group Report 18 June 2019, https://w3c.github.io/web-ledger/
[27] C. Lima, "Developing Open and Interoperable DLT/Blockchain Standards [Standards]" in *Computer*, vol. 51, no. 11, pp. 106-111, 2018. doi: 10.1109/MC.2018.2876184, url: https://doi.ieeecomputersociety.org/10.1109/MC.2018.2876184
[28] Hewett, N., Lehmacher, W., & Wang, Y. (2019, April). Inclusive deployment of blockchain for supply chains. World Economic Forum.
[29] Feng, Q., He, D., Zeadally, S., Khan, M. K., & Kumar, N. (2019). A survey on privacy protection in blockchain system. *Journal of Network and Computer Applications*, *126*, 45-58.
[30] Karaarslan E., and Akbaş, M.F. 2016. Blok Zinciri Tabanlı Siber Güvenlik Sistemleri [Blockchain Based Cyber Security Systems]. Uluslararası Bilgi Güvenliği Mühendisliği Dergisi, Volume 3, Issue 2, Pages 16-21, DOI:10.18640/ubgmd.373297, http://dergipark.gov.tr/ubgmd/issue/33645/373297.
[31] Halpin, H., and Piekarska, M. 2017. Introduction to Security and Privacy on the Blockchain. In 2017 IEEE European Symposium on Security and Privacy Workshops (EuroS&PW) (pp. 1-3). IEEE.
[32] Authority GR. Contact tracing and location data Guidance on the EU General Data Protection Regulation 2016/679 & Data Protection Act 2004, Guidance Note IR01/20. 2020.

Comentario [2]: not used

Comentario [3]: not used

Comentario [4]: Couldn't find proper citation

Comentario [5]: Couldnt find proper citation

[33] Karaarslan, E., Konacaklı, E. "Decentralized Solutions for Data Collection and Privacy in Healthcare". "Artificial Intelligence For Data-Driven Medical Diagnosis" Book. De Gruyter, 2020 (in press)
[34] Karaarslan, E., Aydin, D, "An AI Based Decision Support and Resource Management System for COVID-19 Pandemic", DS for COVID-19 book, Elsevier, 2020 (in press)
[35] Craig Gentry. "Fully Homomorphic Encryption Using Ideal Lattices". Proc.41st Annual ACM Symp. on Theory of Computing. Bethesda, MD, USA, 2009.
[36] Zvika Brakerski, Craig Gentry, and Vinod Vaikuntanathan. "(Leveled) Fully Homomorphic Encryption without Bootstrapping". Proc. 3rd ACM Innovations in Theoretical Computer Science Conf. Cambridge, Massachusetts, 2012;
[37] Zvika Brakerski. "Fully Homomorphic Encryption without Modulus Switching from Classical Gap SVP". CRYPTO 2012. Springer;
[38] Jung Hee Cheon et al. "Homomorphic Encryption for Arithmetic of Approximate Numbers". ASIACRYPT 2017. Springer;
[39] Ilaria Chillotti et al., "Faster Fully Homomorphic Encryption: Bootstrapping in Less Than 0.1 Seconds". ASIACRYPT 2016. Springer.
[40] Gizem S. Cetin et al. "Depth Optimized Efficient Homomorphic Sorting". LATINCRYPT 2015- 4th Int. Conf. on Cryptology and Information Security in Latin America, 2015. Vol. 9230. Springer
[41] A. Biryukov, I. Pustogarov, "Bitcoin over tor isn't a good idea", Security and Privacy (SP), 2015 IEEE Symposium on, IEEE (2015), pp. 122-134
[42] G. Bissias, A.P. Ozisik, B.N. Levine, M. Liberatore, "Sybil-resistant mixing for bitcoin", The Workshop on Privacy in the Electronic Society (2014), pp. 149-158
[43] D. Chaum, "Untraceable electronic mail, return addresses, and digital pseudonyms**",** Commun. ACM, 24 (2) (1981), pp. 84-90
[44] Bonneau, A. Narayanan, A. Miller, J. Clark, J.A. Kroll, E.W. Felten, "Mixcoin: anonymity for bitcoin with accountable mixes**",**
International Conference on Financial Cryptography and Data Security, Springer (2014), pp. 486-504
[45] L. Valenta, B. Rowan, "Blindcoin: blinded, accountable mixes for bitcoin**",** International Conference on Financial Cryptography and Data Security, Springer (2015), pp. 112-126
[46] E. Heilman, L. Alshenibr, F. Baldimtsi, A. Scafuro, S. Goldberg, "Tumblebit: An untrusted bitcoin-compatible anonymous payment hub**",** Network and Distributed System Security Symposium (2017)
[47] G. Bissias, A.P. Ozisik, B.N. Levine, M. Liberatore "Sybil-resistant mixing for bitcoin" The Workshop on Privacy in the Electronic Society (2014), pp. 149-158
[48] G. Maxwell, "Coinjoin: Bitcoin privacy for the real world," Available: https:// bitcointalk.org/index.php?topic=279249.0, Mar. 2013.
[49] T. Ruffing, P. Moreno-Sanchez, and A. Kate, "Coinshuffle: Practical decentralized coin mixing for bitcoin," in ESORICS 2014: 19th European Symposium on Research in Computer Security. Springer International Publishing, 2014, pp. 345–364.

Comentario [6]: not used

Comentario [7]: not used

Comentario [8]: not used

Comentario [9]: repeated

[50] Ronald L. Rivest, Adi Shamir, Yael Tauman, “How to Leak a Secret”, International Conference on the Theory and Application of Cryptology and Information Security, ASIACRYPT 2001, pp 552-565
[51] Kurt M. Alonso, “Zero to Monero: First Edition a technical guide to a private digital currency; for beginners, amateurs, and experts” Published June 26, 2018 (v1.0.0), https://www.getmonero.org/library/Zero-to-Monero-1-0-0.pdf
[52] Amos Fiat, Adi Shamir. "How To Prove Yourself: Practical Solutions to Identification and Signature Problems", Conference on the Theory and Application of Cryptographic Techniques CRYPTO 1986: Advances in Cryptology, pp 186-194
[53] E. Ben-Sasson, "Universal and affordable computational integrity," May 2013, bitcoin 2013: The Future of Payments. [Online]. Available: http://www.youtube.com/watch?v=YRcPReUpkcU
[54] Eli Ben Sasson, Alessandro Chiesa, Christina Garman, Matthew Green, Ian Miers, Eran Tromer, Madars Virza. "Zerocash: Decentralized Anonymous Payments from Bitcoin", 2014 IEEE Symposium on Security and Privacy. San Jose, CA, USA.
[55] Benedikt Bünz, Jonathan Bootle, Dan Boneh, Andrew Poelstra, Pieter Wuille, Gregory Maxwell: Bulletproofs: Short Proofs for Confidential Transactions and More. IEEE Symposium on Security and Privacy 2018: 315-334
[56] Eli Ben-Sasson, Iddo Bentov, Yinon Horesh, Michael Riabzev: Scalable, transparent, and post-quantum secure computational integrity. IACR Cryptol. ePrint Arch. 2018: 46 (2018)
[57] Zcash, "A privacy-protecting, digital currency built on strong science", online, https://z.cash/
[58] Wood, Gavin. "Polkadot: Vision for a heterogeneous multi-chain framework." *White Paper* (2016).
[59] Parity, “A brief summary of everything substrate polkadot”, online, https://www.parity.io/a-brief-summary-of-everything-substrate-polkadot/
[60] Obi Wan, “Cosmos network ve Tendermint incelemesi”, online, https://medium.com/@obiwancoin/cosmos-network-ve-tendermint-i%CC%87ncelemesi-bb9518c4c167
[61] Marshall Taylor, “Tendermint guide”, online, https://coincentral.com/tendermint-guide/
[62] William J. Gordon, Christian Catalini, “Blockchain Technology for Healthcare: Facilitating the Transition to Patient-Driven Interoperability”, online, https://www.sciencedirect.com/science/article/pii/S200103701830028X
[63]https://github.com/hyperledger/cactus/blob/master/whitepaper/whitepaper.md#464-serverside-keychain-plugins