A Tutorial on Blockchain and Applications to Secure Network Control-Planes

Nikola Bozic[+,*], Guy Pujolle[*], Stefano Secci[*]
[+]SQUAD, Paris, France. Email: n.bozic@squad.fr
[*]Sorbonne Universites, UPMC Univ Paris 06, UMR 7606, LIP6, Paris, France. Email: firstname.lastname@upmc.fr

***Abstract*—Recent interest about the blockchain technology brought questions about its application to other systems than the cryptocurrency one. In this paper we present blockchain and discuss key applications to network systems in the literature.**

I. Introduction

Blockchain is a technology meant to store, read and validate transactions in a distributed data-base system. It was conceptualized in 2008 [1]. Closely after, it was used to enable a payment system for the Bitcoin cryptocurrency, released as a open source project.

Bitcoin is undergoing a fast deployment and adoption worldwide. Motivated by world economy crises, Bitcoin had the goal to run an independent payment system that is capable to store and track all transaction using distributed nodes (or ledger) executed by the participants of such a geographically distributed system. The aim of Bitcoin is to provide those transactions related to money transfers transparent, readable by anyone, while being secure. Security here is meant as the capability of avoiding false payments, or at least significantly decrease the probability of false, unauthorized, payments. All transaction being managed among the nodes of the Bitcoin network, there is no need for third trusted party like banks or other financial middle-man. To enable this, it was necessarily to build a system where records can be stored, easily verified and securely locked.

It turns out that blockchain was the technology chosen for running the Bitcoin transaction data-structure. Some countries have even started to consider the adoption of such technologies into public infrastructures [2]. Motivated by the Bitcoin success, interest raised in multiple potential application domains to attempt to utilize blockchain into different types of transactional systems. Recently many research efforts is carried out into this direction, in the communications networking field in particular.

The goal of this paper is to give a brief tutorial on the blockchain technology, classify blockchain techniques, and provide better understanding on its realizations through a few selected application to network systems. The rest of paper is organized as following. In Section II, the blockchain concept and block structure are briefly introduced. Section III resumes advantages and disadvantages of the technology, and issues mitigation proposals. Section IV introduces a blockchain classification. Section V describe a few selected applications to network systems. Section IV-C compare available blockchain systems. We conclude paper in Section VI.

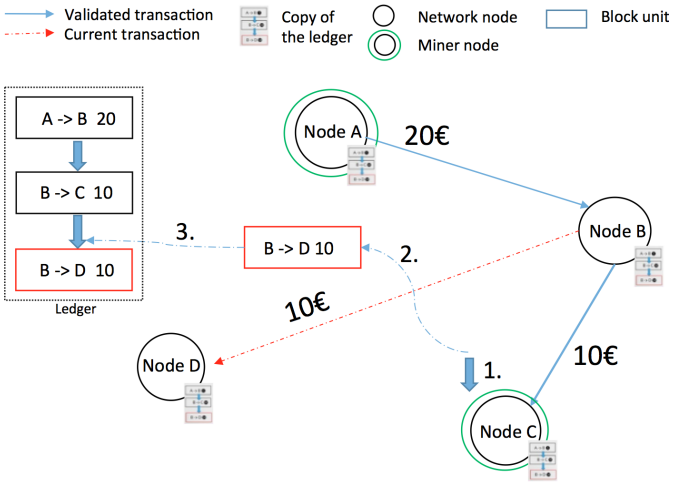

Fig. 1. An example of transactions management in the Bitcoin blockchain.

II. Blockchain Overview

Blockchain enables a new design approach for distributed database using a peer-to-peer (P2P) communications. It is meant to meet key service requirements [2]:

1) Making transactions whose authenticity is guaranteed (e.g., preventing duplicate payments);
2) Ensuring traceability of data by enabling transparent transactions (i.e., make falsification difficult);
3) Stably maintaining the ecosystem against attacks by malicious users without a central authority.

Let us first look how transaction are managed in a Bitcoin blockchain, noting that for the moment we are speaking about a 'public' blockchain alternative.

In blockchain jargon, a 'ledger' is a virtual book where all past transactions are stored. Making consensus among blockchain means provide a protocol that can provide unique state of a ledger, validated by system participants.

Any node with an internet connection can add blocks to the blockchain network, and there must be protocol providing consistent view of records in order to avoid the chaos.

In blockchain jargon, a 'miner' is a blockchain network node that has the role to validate new transactions [3]. When a miner validates a transaction, it places it in a new block, which it broadcasts to the others node in the network.

Fig. 1 illustrates the Bitcoin blockchain transaction process through a simple example. We have a network of 4 nodes

978-1-5090-4476-4/16/$31.00 ©2016 IEEE

marked from A to D. Suppose node A transferred 5 euros to node B, and node B transferred 10 euros to node C. The blockchain has to record those transactions into blocks linked together with a specific key. Let us now assume that B would like to transfer 10 euros to D. Node B will execute the transaction pushing it into all nodes in the network (using peer-to-peer communications).

A transaction needs to be validated and stored in a distributed ledger providing a unique view to all participants. In order to do so, miners intercepting a given transaction will try to validate it, which is trivial as all previous transactions are transparently stored in the ledger: validating that transaction means to verify in the ledger that, e.g., node B has gotten a sufficient amount of assets to transfer to the node D, prior to the actual transaction. As many miners independently go through the ledger, they compete in order to be first to find a specific key, i.e., an identifier that will enable to validate it quickly, result of a computational demanding task known as a 'Proof-of-Work (PoW)' explained hereafter. Finding such a key locks this transaction and allows storing it in a public ledger as follows.

1) The miner having found the key, e.g., node A, creates a block to store the transaction along with that key;
2) the new block is distributed to the network so that other miners, e.g., node B, can easily validate that solution is correct and stop trying to solve the PoW.
3) the new block is chained together with previous transactions blocks, as detailed in the next section. As a reward for the computationally task it accomplished, miner A gets a certain amount of Bitcoins (the amount halves every 210,000 blocks [4]). The blockchain gets updated and everyone on the P2P network becomes aware of it.

Before detailing the following the block construction process, we discuss security aspects of blockchain.

A. Security Aspects

In terms of security, blockchain uses already well-known security enforcement technologies, namely public key cryptography, digital signature, and hashing. Communication is done using a pair of private/public keys. Each node will sign its transactions it wants to levy with its private key. Nodes on the network are addressed by their public key, not their IP address. Communication security is not the strongest security feature of blockchain: the hashing process is the key feature as explained hereafter.

Each transaction is distributed across the whole network, arranged and encapsulated via a time-stamped block that has to pass through a mining process in order to became a valid transaction. For valid transactions nodes eventually always reach a consensus. Reaching a consensus among miners means agreeing on the way to link the new transaction block to the actual chain. Miners actually need to find a specific 'nonce', i.e., a random number that together with transaction data is given as input to a hash function: a good nonce is found when the hashing result gets less than a predefined value (target). The target in Bitcoin is a 256-bit number: the lower the target, the more difficult it is to generate a block; each hash gives a number between 0 and the maximum value of a 256-bit number - if the hash is below the target, mining is done, otherwise the miner tries again, incrementing nonce, which completely changes the hash value. The aim is to have an overall latency of about 10 min per block; in order to approach that, every 2016 blocks every node compares the time it took to generate these blocks with a 14-day goal (i.e., the time theoretically needed to make 2016 blocks). Then each node modifies its target accordingly.

It is important to stress that the hash function should be designed in such a way that given an output it is extremely hard to guess what was input: ideally, security-wise, the most efficient way to find a solution should be the random guess. The nonce is therefore a necessary key to lock a transaction in a ledger. Given the block with nonce, miners can easily verify that the hash result is less then a target. Because of its randomness, such a nonce computation task requires a certain computational power. The process of finding the nonce is referred to as the proof-of-work (PoW) and can be made more or less complex. The PoW complexity can be seen as a measure to prevent attacks by demanding some work to be done by the candidates before they can make any actions in the network.

Let us clarify how having a complex PoW computation can prevent attacks. During an attempt to add transaction records to a blockchain ledger, it is possible that someone would like to add exactly the same transaction in a new block. This would result in making a fork in the chain. Only one copy of the duplicated transaction is the authentic one, and it is necessary to prove which one. Considering this situation, there will be different chain versions. The 'right' transaction can be considered as the one being sent in the network first, generating the authentic chain. As it is not possible to determine which transaction exactly came first, the way to discriminate between the authentic chain and the bad one is looking at its length measured in number of blocks. The longest chain is going to be accepted and the other one will be declared not trustful and hence discarded. Indeed, as there are multiple nodes in the P2P network, multiple nodes almost simultaneously succeed in PoW; all miners working in favor of the authentic transaction will keep adding new blocks making the chain with the right transaction in the fork longer: the node that wants to make a fraud will need to run a lot of computational tasks in order to outrun the right chain and make its chain accepted as the longest. To successfully commit a fraud it requires at least 51% of computational power of all participant in the blockchain network. However authors in [**?**] argue that Bitcoin mining is actually vulnerable even if only 25% of the computing power is controlled by deceiver. In the Bitcoin network there are so many nodes [5] that the probability of being able to control sufficient percentage of the nodes is extremely small.

It is worth noting that once a block is recorded, its integrity is intrinsically provided by the way it is chained to other blocks. An attempt to fake or modify a transaction which

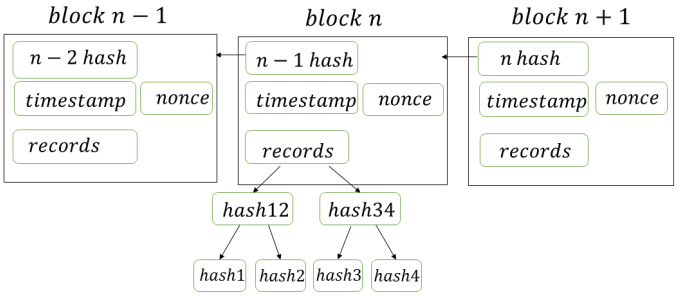

Fig. 2. A 3-block blockchain structure example.

is kept in the ledger will change its hash, which would require hash recalculation of all other blocks. Otherwise the change cannot be hidden. It is more likely that the honest candidate will succeed to add a new block with legitimate transaction to the end of the chain linking the previous hashes, before malicious candidate succeed to deceive all nodes to recalculate their hashes. Therefore, the **integrity** of the ledger is guaranteed by the participants that are part of blockchain.

The hashing process in the block construction is further detailed in the next section.

B. Block Construction

A blockchain can be seen as a database systems using blocks as unitary memory unity. A simple chain of three blocks is presented in Figure 2. Blocks are chained with each other by means of the hash identifier of the previous block in the chain, i.e, the one it attaches to. Moreover, each block is time-stamped and contains a list of its own transactions. Indeed, multiple transactions can be stored in one block, connected in a Merkle tree [6]. For example, in Fig. 2, hash1 and hash2 are computed for two distinct transactions; hash12 is the result of hashing the result of concatenating hash 1 and hash 2. Combining two by two recursively, one gets a hash of the root of Merkle transaction tree, also called Merkle hash.

Such a database systems is distributed in the sense that the complete blockchain is copied in multiple places in the blockchain network as is [7]. Every node has a copy of the blockchain and all copies are equals in terms of the confidentiality, i.e., there is no one copy that can be considered "better" than another one, each node is trustworthy at the same level [3]. Hence a blockchain system is seen as a distributed ledger system that verifies and stores transactions, ensuring its integrity, transparency, authenticity and availability.

In Bitcoin, a transaction is a certain amount of cryptocurrency transferred from one node to anther. In general, a transaction is a process of transferring certain assets that do not necessarily need to be a cryptocurrency. Given a certain block, you can find from all the previous blocks all the information that led to this one, back to the very first block, the 'genesis' block, jumping from one block to another one using the hash identifier.

More precisely, the link is the hash of the following elements: a hash of the previous block, the timestamp, the nonce and the Merkle hash of the transactions in the transaction tree. This explains that even a small change of previous transaction would result in a chaining reaction, changing all following hashes. While it is possible to calculate the Merkle hash of the transactions you would like to place in your new block, it is challenging to compute what nonce to place in the header of a new block, knowing that the hash have to start with certain leading zeros (as already mentioned, in order to make mining difficult, the target starts with leading zeros - smaller the target, more difficult to find the nonce). As already mentioned, the nonce computation is referred to as proof-of-work (PoW).

TABLE I
BLOCKCHAIN VS LEGACY CENTRALIZED AND DISTRIBUTED DATABASES

Features	Blockchain	Centralized Database	Distributed Database
records integrity	high	medium	medium
availability	high	low	medium
fault tolerance	high	low	high
confidentiality	low	high	medium
computing time	low	high	medium
trustless nodes collaboration	high	low	low

III. BLOCKCHAIN - PROS AND CONS

By the nature of its design, there are advantages that are granted by adopting a blockchain solution, as well as some flaws. We position in the following blockchain with respect to legacy database systems, highlighting strong and weak points.

A. Positioning with respect to legacy database systems

In Table I we resume key differences between blockchain, centralized databases and distributed databases.

The most simple way to achieve the basic function blockchain provides can be considered to be a centralized database system, to which blockchain can indeed be compared [8]. Table I provides a summary comparison in terms of key aspects.

Record integrity in blockchain is granted by public key cryptography communication [7], and reinforced by the heavy usage of hashing in block construction and transaction coding. In a distributed database scenario all the nodes are trustful, i.e, expected not to try an attack on the data integrity as they are owned by the same owner. Blockchain can be deployed among trust-less nodes, providing data integrity unless sufficient percentage of the total computational power is controlled by an attacker. Once a blockchain record is stored, any small change attempt would result in completely different block hash. In centralized databases, there one must trust the centralized entity, which however can be the vector to jeopardize data integrity.

In terms of availability, the geographical distribution of nodes clearly allows blockchain to be considered as superior than centralized databases, offering robustness against single-point-of-failure issues.

It is important to highlight that while in legacy database systems data storage is depended on a trusted third-party stakeholder, having access to them and being able to destroy or

corrupt the data, in blockchain there is an intrinsic guarantee of high availability and integrity: the blockchain consensus mechanism enforces all records to be separately and in parallel processed and verified by additional nodes, enabling those nodes to stay synchronized.

Another feature listed in Table I is fault tolerance. In blockchain, many nodes have a copy of the ledger. There is not a copy of higher trust than another copy: all copies are equal and there is no network node more important than others. Hence upon network failure, transactions can keep being updated thanks to the very distributed nature of the system, and upon failure restoration each node can get the missed transactions. Consequently, any node can be safely removed from the network, which is certainly not the case of centralized systems. In comparison to distributed database systems, one can say blockchain can be at least as fault tolerant as a legacy distributed database system.

Having transaction records content fully transparent to any node of a blockchain network is a drawback which can dishearten many users considering to opt for a blockchain solution rather than a legacy system. Indeed, confidentiality is definitely much lower than with legacy systems, and in particular centralized ones, where all request for database content is relayed by the database administrators. As public blockchains leverage on a public ledger, every Internet user can potentially have access to it, which differs from centralized/distributed databases where access is determined by central authority providing stronger confidentiality. Nonetheless, there are proposals on how to mitigate this problem; transacting under multiple blockchain addresses, zero-knowledge proofs are proposed as a way to mitigate confidentiality issues [9] [10]. Naturally these alternatives come with additional drawbacks by making blockchain more complicated or less scalable.

Computing time is definitely a feature for which legacy systems, and in particular centralized systems, can be considered as superior. It is worth recalling that blockchain communication is done through P2P, with records signed using public-key cryptography. This process has a computing overhead that can be avoided in centralized databases. Moreover, the mining process and the consensus mechanism among miners needed in blockchain, taking overall minutes due to ledger size and network latency, are absent in legacy systems.

In [11] it is argued that there is a semantic mistake saying that blockchain is just a shared database; blockchain can be seen instead as a technology that enables a new type of shared databases. The question is: what blockchain can provide that other (distributed) databases cannot? A database is collection of structured information, organized following rules and respecting constraints about how to store data, in particular in terms of consistency. A transaction is collection of instructions that make changes to that data following certain constraints and rules. If those rules are not satisfied, a transaction is simply refused. However, such rules in database management do not solve a problem of trust that is introduced when we have multiple nodes that do not have trust in each other. For example, suppose one would need to share a distributed data ledger among different companies: because of a common competitive market, they cannot trust each other.

P2P replication or multi-master replication enable writing rights to distributed nodes, with strong consistency policy able to manage possible writing conflicts at the expense of latency [12]. A more efficient way of keeping a high availability distributed database up-to-date under an acceptable server latency is to ensure strong consistent duplication and switching only to the unique master copy, with backup copies under eventual consistency. Yet, the existence of one master copy remains the very bottleneck. On the other hand, in terms of consistency vs availability, it can be said that blockchain technique enables higher availability, considering that multiple replications are stored on different entities, with consistency being downgraded to a second class priority, as it is eventually granted by consensus.

B. Drawbacks

As a public system, Blockchain continuously keep growing storing all previous transactions. This necessarily leads to storage increase since the ledger is kept by nodes; in end of 2016, blockchain size was more than 90 GB [13]. Scalability is therefore an important disadvantage.

Furthermore, every Internet node can participate to the network, and each node can have different computational power. It is therefore hard to think about ways to optimize power consumption in a blockchain network, with many redundant computations being done concurrently. The power consumption of a bitcoin transaction is certainly enormous if one attempts to compare it to classical transactions.

Another disadvantage worth being mentioned is that a threat can be introduced by the nature of the mining process: since the probability of finding a valid nonce grows with the machine computing power, stronger machines have better odds. Huge vendors can provide computationally more powerful infrastructures, in order to increase the odds, to be the first who will guess the nonce and get award. If the biggest miners conspired together and made mining alliances, they could represent a threat increasing the success probability in executing a fraud. Because of its public nature, there is no mechanisms for malicious node elimination in a standard blockchain network. As already discussed, confidentiality is a break-deal for many use-cases especially where confidentiality raises legal concerns.

In order to attempt to mitigate the above described drawbacks, efforts are being undergone to increase the efficiency and the protocol speed, with a particular attention on algorithms related to consensus, limited access and participant selection [14]. A natural question is: if we have limited access and listed participants, should a blockchain network serve as a public ledger or can it be developed otherwise to serve to specific use-cases? This led to appearance of blockchain systems that do no strictly follow the public blockchain example of Bitcoin, i.e., private blockchains.

IV. Beyond Legacy Blockchain

We describe in the following different directions in the evolution of blockchain systems.

A. Public vs Private Blockchains

A major open question is therefore whether we always need all Bitcoin blockchain primitives and features, or a distributed ledger can be designed differently to fit an individual use case [3]. In particular, confidentiality concerns are imposing the emergence of alternative blockchain architectures referred to as 'private blockchain'. The driver is to build a system that retains blockchain strong security, but making it smaller, faster, and not public. Figure 3 illustrates major differences between a public blockchain and a private one. First, permissionless

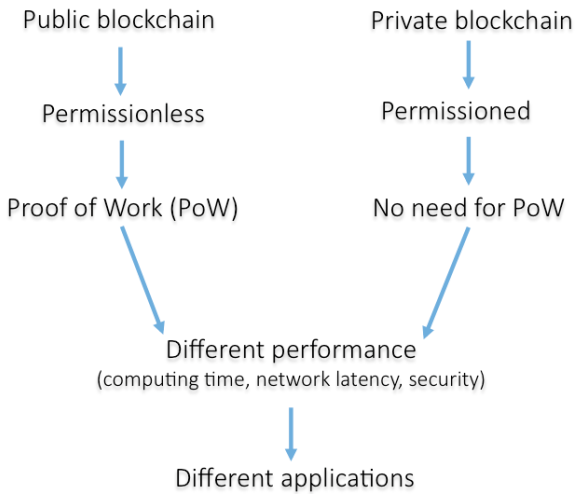

Fig. 3. Main differences between public and private blockchain approaches.

systems are public (such as Bitcoin): because everyone has the right to become part of the system, there is a need for transaction validation by other peer network participants. This process takes time and computational power, so in order to make the system incentive compatible there are transaction fees, and awards given to miners. Having a permissionless system directly implies that there cannot be trust between nodes, so a strongly distributed consensus mechanism must be imposed. In such a system, there is the possibility of a Sybil attack [15], where a network node tries to appear as several distinct nodes by creating a large number of pseudo-identities. Disproportionately large influence by a single node is a threat, so introduction of PoW in transaction validation is logically justified and needed.

Unlike permissionless system, permissioned systems use a participant whitelist. Speed, costs, censorship are just some of the important aspects balanced between permissioned systems and public ones. Features deriving from permissioned systems can also open the possibility to avoid computationally demanding consensus protocol, going beyond PoW, as discussed next.

B. Beyond the PoW

The PoW step in blockchain validation is a key step, but its performance implication in terms of computation time and power, consensus latency and block size, may be too serious to let blockchain fit the requirements of other applications.

In particular, to compensate for PoW complexity other possibilities to secure blockchain were taken in consideration. It is worth mentioning first the Proof-of-Stake (PoS) [16] in so-called 'simplex blockchains': while the PoW demands repeatedly running hashing algorithms, PoS demands proving ownership of a certain amount of currency, their "stake" in the currency, for the case of the Bitcoin cryptocurrency. Another proposed alternative to PoW is the Proof-of-Importance (PoI) [17], where nodes are clustered based on transaction indicators, i.e., number of transactions or contracts, that indicate their importance, hence affecting their credibility. Moreover, in [18] a hybrid method combines PoS and PoI-like indicators. The principle in the computation of PoI indicators is that the credibility should be a function of the number of validated transactions one actor has had. However, basing PoI on the number of transactions may be a vector for creating fake transactions to increase the score. Therefore, in [18] it is proposed that some miners generate a block using a PoS score, the next miner generates the next block using a PoI score, and the next block after that is generated using PoS again, and so on. Such hybrid strategies have an impact on the probability of succeeding an attack, lower than with simplex blockchains [1].

Finally, Practical Byzantin Fault Tolerance (PBFT) algorithm [19], one version of Byzantin Fault Tolerant algorithms (BFT), are also considered for transaction validation in private blockchains. BFT algorithms strive to reach agreement among distributed nodes tolerating Byzantine Faults (BF), i.e., faults where a replica sends different values to different destinations - BFT systems have a first stage of agreement, implemented by a consensus algorithm to enable majority voting on the same state among nodes, followed by the execution. In PBFT, one can tolerate up to f faulty nodes, where f is a known arbitrary fraction of the total number of nodes - with a state machine replicated across different nodes (one replica being set as primary and the others as backups), PBFT algorithm works as follows:

1) A client sends a service request to the primary machine.
2) The primary multicasts the request to the backups.
3) Replicas execute the request and send a reply back.
4) Client waits for $f + 1$ same replies with valid signature from different replicas to consider a result correct.

As total number of nodes needs to be known, PBFT is not fit for public systems; instead, it does for private ones. In this direction, in [14] it is proposed to oppose PoW-based blockchains to those based on BFT state machine replication, focusing on scalability arguments.

C. Open Source Implementations

Many open source projects have been initiated in order to make blockchain more popular and feasible for different

TABLE II
COMPARISON BETWEEN BLOCKCHAIN SYSTEMS

Blockchain	Bitcoin	Ethereum	Hyperledger	Ripple
Nature	public	public	private	public
Validation	PoW - SHA-256	PoW - ethash	PBFT	custom BFT (RPCA)
Purpose	cryptocurrency	smart contracts	chaincode	cryptocurrency
Language	stack-based scripting	Turing complete internal code	Go, Java	C++
Block processing time (appr)	600 s	15 s	quasi real-time	quasi real-time

business models and applications. Some, as Namecoin [20], are strongly based on Bitcoin implementation and network; it is used to register names and store associated values, in support of standard database operations. Other ones different substantially, such as Ethereum [21], Hyperledger [22], and Ripple [23]. The main difference between them is whether they are permissioned or permission-less. Bitcoin, Ethereum and Ripple are decentralized and public blockchains, while Hyperledger is decentralized but private. Between permissioned and permissionless, a key difference is about attack vector mitigation, via consensus techniques. The main differences are summarized in Table II.

1) Ethereum: is a platform which enables 'smart contracts', i.e., a blockchain network of distributed applications that mechanically execute tasks when certain conditions are met. It intends to enable a blockchain with a built-in fully fledged Turing-complete programming language to create contracts, allowing users to design own applications by writing up the logic in a few lines of code - this represents an innovation with respect to the rather rigid format of Bitcoin.

The Ethereum network currently uses a PoW consensus algorithm, called Ethash, specifically created for Ethereum. The way Ethash provides a PoW is by emphasizing memory hardness, i.e., the fact that memory access can also be a bottleneck, besides the computing power. Ethash is designed to consume nearly the entire available memory access bandwidth. PoW function is made to be sequential memory-hard, i.e., the function is made such that the determination of the nonce requires a lot of memory and memory access bandwidth, so that the memory cannot be used in parallel to discover multiple nonces simultaneously [21].

2) Hyperledger: is instead meant for private blockchains enabling 'chaincode', i.e., a piece of code that is deployed into a network of Hyperledger nodes enabling interaction with the shared ledger of that network. Like miners in Bitcoin, Hyperledger has validating peers (VP), which however do not 'mine' blocks and do not share the blocks between them. Instead, transaction is sent to one trusted VP that relays it to other VPs. Using PBFT, VPs reach consensus and build their own block. Following the total order, all such blocks will be the same as transaction execution is deterministic. System performances with Hyperledger go significantly ahead Bitcoin, as a computationally demanding PoW is not required.

3) Ripple: is meant for public cryptocurrency systems[24]; it uses an iterative consensus process taking only a few seconds to finalize transactions. The Ripple Protocol consensus algorithm (RPCA), can be considered as a custom BFT - it is applied every few seconds by all nodes, in order to maintain the correctness and agreement of the network. Once consensus is reached, the current ledger is considered "closed"and becomes the last-closed ledger. RPCA consists of several rounds, explained hereafter [24]:

1) Initially, each server (any entity running Ripple) takes all transactions it has seen that have not already been applied, and makes them public in the form of a list known as the "candidate set";
2) Each server then merges the candidate sets of all servers and votes on the transactions veracity: transactions that receive more than a minimum percentage of votes go to the next round;
3) The final round of consensus requires a minimum percentage of 80% of agreeing servers.

RPCA maintains correctness as long as there is no more than a certain fraction of f faulty nodes.

V. NETWORK CONTROL-PLANE USE-CASES

Although legacy blockchain is used for a cryptocurrency system, transaction records can concern other entities than a currency: a blockchain can be used to deploy distributed ledger for any kind of assets. Blockchain is now finding its place in many different fields, including finance, health-care, real-state, supply chains, government, and telecommunications.

We review in the following some blockchain use-cases in communication network control-plane applications. Most of these works are still at design and pre-specification phase, hence not all details are given; nevertheless, their description can be source of inspiration for improved applications or other control-plane applications, in particular related to system configuration (e.g., in IoT and ICN), routing management (e.g., in BGP or more broadly distance vector routing protocols) or entity mapping problems (e.g., in DNS and LISP or other mapping protocols).

A. IoT use-case

Authors in [7] examine how blockchains can meet the requirements of Internet of Things (IoT) communication infrastructures. More precisely, an application is the IoT device software update. Indeed, the current centralized model for software distribution suffers from high cost and lack of trust from the customer side; it is well known that such updates can be attack vectors against end-devices, and as IoT devices cannot embed heavy firewall systems they are even weaker in this respect. Making all IoT devices of a same network service part of the same blockchain, the manufacturer then

TABLE III
EXAMPLE OF BLOCKCHAIN APPLICATION TO NETWORK CONTROL-PLANE USE-CASES

	IoT	ICN	BGP	DNS	LISP
Scope	software updates; supply-chain transactions	System parameters (SP) delivery	Route announcements	Domain name delegation transaction	Routing locator mapping
Addressed issues	scalability	SP and block sizes	security	security	security
Implementation	not specified	Namecoin-Bitcoin fork	Bitcoin independent	Bitcoin independent	not specified
Hashed object	transactions	Content names with SP	Prefix-AS mapping	sub domain delegation	database map-entries

can deploy a smart-contract type of solution to allow the IoT devices to retrieve the latest firmware update [7]. Even if the manufacturer stop shipping the software, users can still retrieve desired firmware while being reassured that they get the right file.

Another addressed application is the database function of typical supply-chain execution for novel applications unveiled by IoT operations such as smart-grid, garbage collectors, etc.. Instead of using internal database for each stakeholder as part of the supply-chain, a blockchain network can track a specific asset exposing a shared database, with cryptographic verifications, automatically propagated along the network, hence creating an auditable trail of information - under the assumption each stakeholder has network connection [7]. In the energy sector, IoT-blockchains integration might allow a peer-to-peer market where machines can buy and sell energy automatically, according to defined smart-contracts among users, as investigated in transactive grids [25], [26].

B. ICN use-case

Another investigated application domain is ICN (Information-Centric Networking) networks, in particular for a decentralized secure content distribution control-plane using device names as described in [27]. The idea is to use blockchain for System Parameters (SP) delivery in Hierarchical Identity Based Encryption (HIBE) systems [28]. The proposed ICN system is such that the content provider want to share its content with users, and it uses content names as HIBE public keys. HIBE consists of five steps, and the blockchain logic is applied to the intial SETUP and EXTRACT: during the SETUP, HIBE is executed by a Private Key Generator (PKG), taking as input a security parameter k hence returning a master-secret key (MSK) and some system parameters (SPs), keeping the MSK private whereas the SPs are made public. The EXTRACT phase takes as inputs the MSK, SPs and an identity ID, and returns a secret key. Each content provider generates the (public) SPs required by HIBE using his own PKG, as well as a secret key for each content name, and registers all globally unique names in the blockchain, including in the registration message the SP. The registration process is implemented as a new transaction in which the owner associates a content name with its SP [28]. Hence a blockchainis used to disseminate the SPs in a distributed way.

The proposed implementation of the ICN system uses Namecoin, which uses in turn the Bitcoin network [5]. An issue identified in this use-case is to have SPs of larger size than Namecoin fields (up to 520 bytes). Authors argue that further work shall consider alternative HIBE and blockchain implementations.

C. Internet use-cases

Authors in [29] propose to secure the Internet BGP (Border Gateway Protocol) and DNS (Domain Name System) control-plane infrastructure by the use of a blockchain based mechanism. They name the result 'Internet blockchain'. The idea is to allow Internet path announcements via BGP and domain name to IP mapping via DNS to be authenticated using a blockchain system, hence avoiding any Public Key Infrastructure (PKI) [30]. Peers on blockchain are addressable by its public key, unlike PKI, a peer does not need to get its address authenticated by any other entity, thereby eliminating the possibility of key tempering or spoofing by a third party. 'Internet blockchain' is used to authenticate route announcements in BGP and security extensions in DNS servers, technologies introduced to palliate to the lack of trust in BGP and DNS control-plane speakers. It is argued that its blockchain implementation should be completely independent from the Bitcoin blockchain, ideally providing Turing-complete programming language with multiple inputs/outputs transaction type providing fully scriptable transactions which trigger only when some specified events occur.

Another similar proposal is to investigate the application of blockchain for IP-to-IP mappings in the Locator/Identifier Separation Protocol (LISP) at the IETF, see [31]. The idea is to store EID-prefix delegations in a blockchain, wherein an EID-prefix delegations is equivalent to a bitcoin transaction.

It is a matter of fact that the Internet at large can strongly benefit from having such a verifiable and distributed transaction history log, result of a multi-signature based authorization granted by blockchain for enhanced security, with easy extensibility and possible scriptable programmability to secure even other types of Internet resources than address/name mappings, routing path announcements, etc. These investigations are being conveyed also in discussion groups in (pre-)standardization bodies, as for example the Distributed data & Service Federation BOF at the IRTF [32], rising the importance of Ethereum-like scriptable blockchains. Furthermore, as accessory blockchains one could have even an ad-hoc Internet infrastructure cryptocurency to make related services chargeable. Nevertheless, as evidenced in these studies about Internet control-plane applications, scalability remains the addressed challenge ahead, and active research is thus likely to enable a blockchain-based tamper-resistant Internet.

Table III summarizes different network control-plane use-cases explained in this section. These ideas being still at a preliminary design and specification phase, there are no identified or decided implementations and block validation protocols (PoW, PoI, PoS, BFT, etc), except for the ICN study for the moment strongly rooted in the Bitcoin architecture.

VI. Conclusion

Block-chain introduction into existing systems should first goes through detailed analysis in order to assess the benefits and practical constraints of such systems. We wrote this paper with the goal to provide a brief tutorial on blockchain idea and functioning, highlighting some preliminaries studies about its application to network control-plane systems. We believe further effort is needed in making blockchain implementations more open in term of customization than existing ones, in order to further stimulate their adoption in both private and public use-cases.

References

[1] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," Oct. 2008.

[2] "Survey on Blockchain Technologies and Related Services FY2015 Report," Nomura Research Institute, Tech. Rep., 03 2016.

[3] T. Swanson, "Consensus-as-a-service: a brief report on the emergence of permissioned, distributed ledger systems," 2015.

[4] Bitcoin block reward halving countdown. [Online]. Available: http://www.bitcoinblockhalf.com/

[5] Bitnodes. [Online]. Available: https://bitnodes.21.co/

[6] R. C. Merkle, "A digital signature based on a conventional encryption function," in *Conference on the Theory and Application of Cryptographic Techniques*. Springer, 1987, pp. 369–378.

[7] K. Christidis and M. Devetsikiotis, "Blockchains and smart contracts for the internet of things," *IEEE Access*, vol. 4, pp. 2292–2303, 2016.

[8] G. Greenspan. (2016, March) Blockchains vs centralized databases. [Online]. Available: http://www.multichain.com/blog/2016/03/blockchains-vs-centralized-databases/

[9] E. B. Sasson, A. Chiesa, C. Garman, M. Green, I. Miers, E. Tromer, and M. Virza, "Zerocash: Decentralized anonymous payments from bitcoin," in *2014 IEEE Symposium on Security and Privacy*, May 2014, pp. 459–474.

[10] I. Miers, C. Garman, M. Green, and A. D. Rubin, "Zerocoin: Anonymous distributed e-cash from bitcoin," in *2013 IEEE Symposium on Security and Privacy*, May 2013, pp. 397–411.

[11] Private blockchains are more than just shared databases. [Online]. Available: http://www.multichain.com/blog/2015/10/private-blockchains-shared-databases/

[12] T. Cui, "Ldap directory template model on multi-master replication," in *2010 IEEE Asia-Pacific Services Computing Conference*, Dec 2010, pp. 479–484.

[13] Blockchain size. [Online]. Available: https://blockchain.info/charts/blocks-size

[14] M. Vukolić, "The quest for scalable blockchain fabric: Proof-of-work vs. bft replication," in *International Workshop on Open Problems in Network Security*. Springer, 2015, pp. 112–125.

[15] J. R. Douceur, "The sybil attack," in *International Workshop on Peer-to-Peer Systems*. Springer, 2002, pp. 251–260.

[16] S. King and S. Nadal, "Ppcoin: Peer-to-peer crypto-currency with proof-of-stake," *self-published paper, August*, vol. 19, 2012.

[17] "Technical Reference, NEM, Version 1.0 ," Tech. Rep., May 2015.

[18] H. Watanabe, S. Fujimura, A. Nakadaira, Y. Miyazaki, A. Akutsu, and J. Kishigami, "Blockchain contract: Securing a blockchain applied to smart contracts," in *2016 IEEE International Conference on Consumer Electronics (ICCE)*, Jan 2016, pp. 467–468.

[19] M. Castro and B. Liskov, "Practical byzantine fault tolerance and proactive recovery," *ACM Transactions on Computer Systems (TOCS)*, vol. 20, no. 4, pp. 398–461, 2002.

[20] H. Kalodner, M. Carlsten, P. Ellenbogen, J. Bonneau, and A. Narayanan, "An empirical study of namecoin and lessons for decentralized namespace design," in *Workshop on the Economics of Information Security (WEIS)*. Citeseer, 2015.

[21] G. Wood, "Ethereum: A secure decentralised generalised transaction ledger," *Ethereum Project Yellow Paper*, 2014.

[22] C. Cachin, "Architecture of the hyperledger blockchain fabric," in *Workshop on Distributed Cryptocurrencies and Consensus Ledgers*, July, 2016.

[23] Ripple. [Online]. Available: https://http://www.multichain.com/

[24] D. Schwartz, N. Youngs, and A. Britto, "The ripple protocol consensus algorithm," *Ripple Labs Inc White Paper*, p. 5, 2014.

[25] (2016) Transactive grid. [Online]. Available: http://transactivegrid.net/

[26] Blockchain-based microgrid gives power to consumers in new york. [Online]. Available: https://trac.ietf.org/trac/irtf/wiki/blockchain-federation

[27] N. Fotiou and G. C. Polyzos, "Decentralized name-based security for content distribution using blockchains," in *2016 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, April 2016, pp. 415–420.

[28] A. Lewko and B. Waters, "Unbounded hibe and attribute-based encryption," in *Annual International Conference on the Theory and Applications of Cryptographic Techniques*. Springer, 2011, pp. 547–567.

[29] A. Hari and T. Lakshman, "The internet blockchain: A distributed, tamper-resistant transaction framework for the internet," in *Proceedings of the 15th ACM Workshop on Hot Topics in Networks*. ACM, 2016, pp. 204–210.

[30] U. Maurer, "Modelling a public-key infrastructure," in *European Symposium on Research in Computer Security*. Springer, 1996, pp. 325–350.

[31] V. E. F. M. Jordi Pailliss, Albert Cabellos, "A blockchain-based mapping system," in *IETF 97, Seoul*, Nov. 2016.

[32] Blockchain-federation. Internet Research Tasks Force (IRTF). [Online]. Available: https://trac.ietf.org/trac/irtf/wiki/blockchain-federation