A Review Of Blockchain

Mahboubeh Heydari, Ali Noroozi, Narjes Akbari, Masoumeh Mohammadi, Kamal Yousefiyan, Dr. Mohammad Hossein Ahmadzadegan
Islamic Azad University E-Campus
m.heydari@iauec.ac.ir, Alinoroozi@ iauec.ac.ir

Abstract - Blockchain is an emerging technology for decentralized and transactional data sharing across a large network of untrusted participants. It enables new forms of distributed software architectures, where components can find agreements on their shared states without trusting a central integration point or any particular participating components. Considering the block- chain as a software connector helps make explicitly important architectural considerations on the resulting performance and quality attributes (for example, security, privacy, scalability and sustainability) of the system.

I. INTRODUCTION

In today's ‹Based on our experience in several projects using blockchain, in this paper we provide rationales to support the architectural decision on whether to employ a decentralized blockchain as opposed to other software solutions, like traditional shared data storage. Additionally, we explore specific implications of using the blockchain as a software connector including design trade-offs regarding quality attributes

Blockchain—a peer-to-peer network that sits on top of the internet—was introduced in October 2008 as part of a proposal for bitcoin, a virtual currency system that eschewed a central authority for issuing currency, transferring ownership, and confirming transactions. Bitcoin is the first application of blockchain technology.

In a blockchain system, the ledger is replicated in a large number of identical databases, each hosted and maintained by an interested party. When changes are entered in one copy, all the other copies are simultaneously updated. So as transactions occur, records of the value and assets exchanged are permanently entered in all ledgers. There is no need for third-party intermediaries to verify or transfer ownership. If a stock transaction took place on a blockchain-based system, it would be settled within seconds, securely and ver- ihably. (The infamous hacks that have hit bitcoin exchanges exposed weaknesses not in the blockchain itself but in separate systems linked to parties using the blockchain.)[1]

The blockchain data structure is a timestamped list of blocks, which records and aggregates data about transactions that have ever occurred within the blockchain network.

The first generation of blockchain is a public ledger for monetary transactions with very limited. capability to support programmable transactions. The second generation of blockchain became a generally programmable infrastructure with a public ledger that records computational results. apability to support programmable transactions.

The blockchain is a public ledger maintained by all the nodes within the cryptocurrency network. The blockchain stores all the transactions that have ever occurred in the cryptocurrency system. Later, the concept was generalized to a distributed ledger that exploits the blockchain to verify and store transactions without needing cryptocurrency or tokens.

The blockchain network does not rely on any central trusted authority, which has the power to control the system, like traditionally centralized banking and payment systems

we use blockchain to refer to the data structure replicated on the nodes and blockchain network to refer to the infrastructure composed of a decentralized peer-to-peer network of nodes.

Blocks and transactions are the two essential elements making up the blockchain. Blocks are the containers aggregating transactions. Every block is identifiable, and linked back to its previous block in the chain.

Transactions represent state transitions with ownership information, which could include new data records and transfer of control among participants.

The blockchain is a complex, network-based software connector, which provides communication, coordination (through transactions, smart contracts and validation oracles) and facilitation services. Additionally, a blockchain-based system can maintain a unique chain to record all types of transactions together or maintain multiple chains to isolate information of separate parties or of

separate concerns, for example, using one chain to store transactions, and using a separate chain to store access control information.

On the blockchain, there is a data set registry implemented as a smart contract, which stores all the data sets registered on the platfonn and allows data owners to register a new data set on the blockchain.

The blockchain provides communication and coordination services through transactions, validation oracles and smart contracts, and specific facilitation services, including permission management, cryptography-based secure payment, transaction validation, mining and incentives.

II. HOW BLOCKCHAIN WORK s

Here are five basic principles underlying the technology.

1- DISTRIBUTED DATABASE

Each party on a blockchain has access to the entire database and its complete history. No single party controls the data or the information. Every party can verify the records of its transaction partners directly, without an intermediary.

2- PEER-TO-PEER TRANSMISSION

Communication occurs directly between peers instead of through a central node. Each node stores and forwards information to all other nodes.

3- TRANSPARENCY WITH PSEUDONYMITY

Every transaction and its associated value are visible to anyone with access to the system. Each node, or user, on a blockchain has a unique 30-plus-character alphanumeric address that identifies it. Users can choose to remain anonymous or provide proof of their identity to others. Transactions occur between blockchain addresses.

4-IRREVERSIBILITY OF RECORDS

Once a transaction is entered in the database and the accounts are updated, the records cannot be altered, because they're linked to every transaction record that came before them (hence the term "chain"). Various computational algorithms and approaches are deployed to ensure that the recording on the database is permanent, chronologically ordered, and available to all others on the network.

5-COMPUTATIONAL LOGIC

The digital nature of the ledger means that blockchain transactions can be tied to computational logic and in essence programmed. So users can set up algorithms and rules that automatically trigger transactions between nodes.[5]

III.Using Blockchain to Protect Personal Data

we describe a decentralized personal data management system that ensures users own and control their data. We implement a protocol that turns a blockchain into an automated access-control manager that does not require trust in a third party. we address the privacy concerns users face when using third-party services.

We focus specifically on mobile platforms, where services deploy applications for users to install. These applications constantly collect high-resolution personal data of which the user has no specific knowledge or control.

In light of this, our system protects against the following common privacy issues:

Data Ownership. Our framework focuses on ensuring that users own and control their personal data. As such, the system recognizes the users as the owners of the data and the services as guests with delegated permissions.

Data Transparency and Auditability. Each user has complete transparency over what data is being collected about her and how they are accessed.

Fine-grained Access Control. One major concern with mobile applications is that users are required to grant a set of permissions upon sign-up. These permissions are granted indefinitely and the only way to alter the agreement is by opting-out. Instead, in our framework, at any given time the user may alter the set of permissions and revoke access to previously collected data.

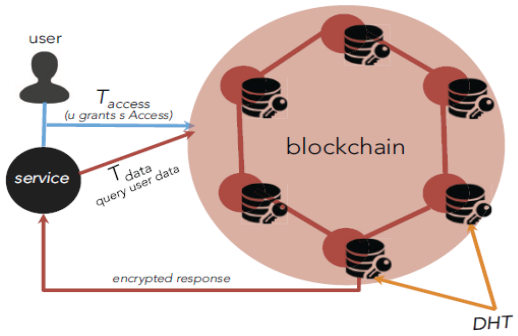

Fig. 1. Overview of the decentralized platform

As illustrated in Figure 1, the three entities comprising our system are mobile phone users, services, and nodes,. The system itself is designed as follows. The blockchain accepts two new types of transactions: T (access), used for access control management; and T (data), for data storage and retrieval. These network operations could be easily integrated into a mobile software development kit (SDK) that services can use in their development process. To illustrate, consider the following example: a user installs an application that uses our platform for preserving her privacy. As the user signs up for the first time, a new shared (user, service) identity is generated and sent, along with the associated permissions, to the blockchain in a T (access) transaction. Data collected on the phone (e.g., sensor data such as location) is encrypted using a shared encryption key and sent to the blockchain in a T (data) transaction

which subsequently routes it to an off-blockchain key-val-

ue store, while retaining only a pointer to the data on the public ledger. Both the service and the user can now query the data using a T (data) transaction with the pointer (key) associated to it.

The blockchain then verifies that the digital signature belongs to either the user or the service. For the service, its permissions to access the data are checked as well. Finally, the user can change the permissions granted to a service at any time by issuing a T (access) transaction with a new set of permissions, including revoking access to previously stored data.

Blockchain Protocols:. Protocol 1 is executed by nodes in the network when a T (access) transaction is received, and similarly, Protocol 2 is executed for T (data) transactions. As mentioned earlier in the paper, T (access) transactions allow users to change the set of permissions granted to a service, by sending a POLICYu, s set. Sending the empty set revokes all access-rights previously granted. Sending a T (access) transaction with a new compound identity for the first time is interpreted as a user signing up to a service. Similarly, T (data) transactions govern read/ write operations. With the help of CheckPolicy, only the user (always) or the service (if allowed) can access the data. Note that in lines 9 and 16 of Protocol 2 we used shorthand notation for accessing the DHT like a normal hashtable. In practice, these instructions result in an off-blockchain network message (either read or write) that is sent to the DHT.

1: **procedure** HANDLEACCESSTX(pk^k_{sig}, m)
2: $s \leftarrow 0$
3: $pk^{u,s}_{sig}, pk^{s,u}_{sig}, POLICY_{u,s} = Parse(m)$
4: **if** $pk^k_{sig} = pk^{u,s}_{sig}$ **then**
5: $L[\mathcal{H}(pk^k_{sig})] = m$
6: $s \leftarrow 1$
7: **end if**
8: **return** s
9: **end procedure**

Fig. 2. Protocol 1

1: **procedure** HANDLEDATATX(pk^k_{sig}, m)
2: $c, x_p, rw = Parse(m)$
3: **if** $CheckPolicy(pk^k_{sig}, x_p) =$ `True` **then**
4: $pk^{u,s}_{sig}, pk^{s,u}_{sig}, POLICY_{u,s} \leftarrow Parse(L[\mathcal{H}(pk^{u,s}_{sig})])$
5: $a_{x_p} = \mathcal{H}(pk^{u,s}_{sig} \parallel x_p)$
6: **if** $rw = 0$ **then** ▷ rw=0 for write, 1 for read
7: $h_c = \mathcal{H}(c)$
8: $L[a_{x_p}] \leftarrow L[a_{x_p}] \cup h_c$
9: (DHT) $ds[h_c] \leftarrow c$
10: **return** h_c
11: **else if** $c \in L[a_{x_p}]$ **then**
12: (DHT) **return** $ds[h_c]$
13: **end if**
14: **end if**
15: **return** $\emptyset$
16: **end procedure**

Fig. 3. Protocol 2

We rely on the blockchain being tamper-free, an assumption that requires a sufficiently large network of untrusted peers. In addition, we assume that the user manages her keys in a secure manner, for example using a secure-centralized wallet service. We now show how our system protects against adversaries compromising nodes in the system.

Given this model, only the user has control over her data. The decentralized nature of the blockchain combined with digitally-signed transactions ensure that an adversary cannot pose as the user, or corrupt the network, as that would imply the adversary forged a digital-signature, or gained control over the majority of the network's resources. Similarly, an adversary cannot learn anything from the public ledger, as only hashed pointers are stored in it.

1: **procedure** COMPOUNDIDENTITY(u, s)
2: *u and s form a secure channel*
3: *u* executes:
4: $(pk^{u,s}_{sig}, sk^{u,s}_{sig}) \leftarrow \mathcal{G}_{sig}()$
5: $sk^{u,s}_{enc} \leftarrow \mathcal{G}_{enc}()$
6: *u shares* $sk^{u,s}_{enc}, pk^{u,s}_{sig}$ *with s*
7: *s* executes:
8: $(pk^{s,u}_{sig}, sk^{s,u}_{sig}) \leftarrow \mathcal{G}_{sig}()$
9: *s shares* $pk^{s,u}_{sig}$ *with s*
10: // Both *u* and *s* have $sk^{u,s}_{enc}, pk^{u,s}_{sig}, pk^{s,u}_{sig}$
11: **return** $pk^{u,s}_{sig}, pk^{s,u}_{sig}, sk^{u,s}_{enc}$
12: **end procedure**

Fig. 4. Protocol 3

An adversary controlling one or more DHT nodes cannot learn anything about the raw data, as it is encrypted with keys that none of the nodes posses. Note that while data integrity is not ensured in each node, since a single node can tamper with its local copy or act in a byzantine way, we can still in practice minimize the risk with sufficient distribution and replication of the data.

Finally, generating a new compound identity for each user service pair guarantees that only a small fraction of the data is compromised in the event of an adversary obtaining both the signing and encryption keys. If the adversary obtains only one of

the keys, then the data is still safe. Note that in practice we could further split the identities to limit the exposure of a single compromised compound identity. For example, we can generate new keys for every hundred records stored.

As a result, Personal data, and sensitive data in general, should not be trusted in the hands of third-parties, where they are susceptible to attacks and misuse. Instead, users should own and control their data without compromising security or limiting companies' and authorities' ability to provide personalized services. Our platform enables

this by combining a blockchain, re-purposed as an access-control moderator, with an off blockchain storage solution. Users are not required to trust any third-party and are always aware of the data that is being collected about them and how it is used. In addition, the blockchain recognizes the users as the owners of their personal data. Companies, in turn, can focus on utilizing data without being overly concerned about properly securing and compartmentalizing them.[1]

IV. Architecture of the Hyperledger Blockchain Fabric

Hyperledger. The Hyperledger Project (www.hyperledger.org) is a collaborative effort to create an enterprise-grade, open-source distributed ledger framework and code base. It aims to advance blockchain technology by identifying and realizing a cross-industry open standard platform for distributed ledgers, which can transform the way business transactions are conducted globally. Established as a project of the Linux Foundation in early 2016, the Hyperledger Project currently has more than 50 members.

Hyperledger Fabric. Hyperledger Fabric (github.com/hyperledger/fabric) is an implementation of a distributed ledger platform for running smart contracts, leveraging familiar and proven technologies, with a modular architecture allowing pluggable implementations of various functions. It is one of multiple projects currently in incubation under the Hyperledger Project. A developerpreview of the Hyperledger Fabric (called "v0.5-developer-preview") has been released in June 2016 (github.com/hyperledger/fabric/wiki/Fabric-Releases).

Some key features of the current fabric release are:

- A permissioned blockchain with immediate finality;
- Runs arbitrary smart contracts (called chaincode) implemented in Go (golang.org):
- User-defined chaincode is encapsulated in a Docker container;
- System chaincode runs in the same process as the peer;
- Consensus protocol is pluggable, currently an implementation of Byzantine fault-tolerant consensus using the PBFT protocol is supported, a prototype of SIEVE [6] to address nondeterministic chaincode is available, and a protocol stub (named NOOPS) serves for development on a single node;
- Security support through certificate authorities (CAs) for TLS certificates, enrollment certificates, and transaction certificates;
- Persistent state using a key-value store interface, backed by RocksDB (rocksdb.org);
- An event framework that supports pre-defined and custom events;
- A client SDK (Node.js) to interface with the fabric;
- Support for basic REST APIs and CLIs.

Support for non-validating peers is minimal in the developer preview release.

V. A Scalable Blockchain Protocol : Blockchain-NG

Bitcoin-NG is a blockchain protocol that serializes transactions, much like Bitcoin, but allows for better latency and bandwidth without sacrificing other properties. The protocol divides time into epochs. In each epoch, a single leader is in charge of serializing state machine transitions. To facilitate state propagation, leaders generate blocks. The protocol introduces two types of blocks: key blocks for leader election and microblocks that contain the ledger entries. Each block has a header that contains, among other fields, the unique reference of its predecessor, namely a cryptographic hash of the predecessor header. The security of the protocol derives from its incentive compatibility, motivating the participants to follow the rules. In this section we detail the operation of the protocol.[3]

Until Bitcoin-NG, the thinking was that there were, essentially, two options for increasing Bitcoin's transaction throughput: increase the size of blocks, or decrease the block interval. Both options lead to various undesirable outcomes. Without rehashing (no pun intended) the entire blocksize debate, we'll quickly touch upon some of the key arguments.

In essence, all the protocol problems stem from the same fundamental issue. Due to the nature of the distribution algorithm, increasing the blocksize or reducing the block interval both lead to an increased rate of forks. In a fork, the blockchain is bifurcated into multiple branches, and there is no single blockchain. The system is therefore in an undecided state. Eventually, the fork is resolved, one branch is chosen and other branches are thereafter pruned, or simply, ignored.

Forks incur two significant security risks. First, they reduce security against attackers. Bitcoin is secured by mining power, and mining power in pruned branches does not participate in securing the system. If 1/4 of the blocks are pruned, then an attacker can be 1/4 smaller to perform selfish mining, or a 51% attack.

Second, forks reduce fairness. Bitcoin and all blockchain protocols compensate miners for their effort, and the compensation should be proportional to a miner's power. When forks are frequent, small miners and miners that are not well connected to the overlay network are at a dis-

advantage, earning less than their fair share. Miners are therefore incentivized to coalesce into larger and larger pools, and thereby pose a centralization threat.[4]
And of course, larger blocks typically require more resources, effectively cutting certain kinds of peers out of the network. Since the Bitcoin network is quite bursty, and at the network level, operates by lying idle for long periods of time, punctuated with sudden waves when a block has to be propagated throughout the globe, well-provisioned nodes are a necessity. And certain geographic regions may be at a permanent disadvantage.
The scalability debate has revolved around these issues, and has been caught in a morass, as these concerns are genuine and the tradeoffs difficult to resolve. And even if a compromise is found, the tradeoffs involved mean that the throughput gains will be modest. Under the currently prominent proposals, Bitcoin does not become competitive with today's VISA throughput for decades. The block-size/block-interval parameter adjustment is a difficult line to toe, as is clear from the tenor of the scalability debate.
Specifically, Bitcoin-NG chooses a leader at the beginning of an epoch, and she is in charge of serializing transactions until the next leader is chosen. NG maintains the overall blockchain structure, but has two types of blocks: key-blocks and microblocks. Key-blocks are used for leader election. They are generated by mining with Proof of Work, as in Bitcoin, and they occur at 10 minute intervals on average, as in Bitcoin; in fact, they are identical, in format, to Bitcoin blocks, except for a small twist on the coinbase transaction, explained below. Every key-block initiates a new epoch. Microblocks contain transactions; they are generated by the epoch leader; they contain no proof of work, and are signed with the leader's private key.

In a Bitcoin block, the first transaction, called the coinbase, rewards the miner for having solved a cryptopuzzle and thus for having contributed a block to the blockchain. All of the transactions are part of the same block and are contributed en masse. In between blocks, the traditional Bitcoin system appears idle to an onlooker, as miners are working to discover the next block, but without apparent progress on the consensus front. In contrast, in Bitcoin-NG, the key-blocks can be tiny because they need contain only the coinbase transaction, which names the public key that the miner will be using to sign microblocks. Because a key-block requires proof of work, competing miners cannot just manufacture one and usurp the leadership at will. Following the key-block, the lead miner can quickly issue microblocks, simply by signing them with the private key corresponding to the public key named in the key-block's coinbase. [4]

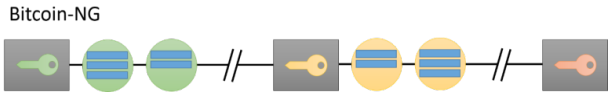

Fig. 5: Structure of the Bitcoin-NG chain. Microblocks (circles) are signed with the private key matching the public key in the last key block (squares). Fee is distributed 40% to the leader and 60% to the next one.

In short, Bitcoin-NG shifts the process of issuing blocks: instead of manufacturing a block at a time as in Bitcoin, an NG miner first acquires the right to issue microblocks, and can thereafter efficiently create a series of microblocks. Microblock creation is limited solely by signing speed (in the millisecond range) and network propagation speeds of small microblocks. Should the miner falter for any reason, other miners can take over when they discover a new key-block. Unlike Ripple and related protocols, leadership handover does not require participation from a quorum of existing nodes, and therefore maintains Bitcoin's decentralized Byzantine fault tolerance guarantees.
The keen reader may already notice several potential pitfalls, as it might seem difficult, at first glance, to incentivize miners to follow the protocol. Double-spend attacks by malicious miners are obviously a key concern.
We believe that Bitcoin-NG advances the science of blockchains by increasing throughput and reducing latency, without impacting miner fairness, the open architecture of Bitcoin, or the clients in any substantial way. It is worth noting that there is no conflict or contradiction between NG and Blockstream's sidechains; in fact, Blockstream's pegs allow for moving bitcoins among chains, and such chains can benefit from improved performance using Bitcoin-NG.
Anyway, it's very interesting to us that without sacrificing the capabilities of the Blockchain, we can create a better and more attractive layout than Blockchain.

References:

[1] G Zyskind, Oz Nathan, Decentralizing Privacy: Using Blockchain to Protect Personal Data, 2015 IEEE CS Security and Privacy Workshops

[2] Harvard Business Review. The Truth About BlockChain.Published In HBR January-February 2017,by Marco Iansity And Karim R.Lakhani

[3] Ittay, E., Gencer, A. E., Sirer, E. G., Renesse, R., Cornell University. A Scalable Blockchain Protocol. https://www.usenix.org/system/files/conference/nsdi16/nsdi16-paper-eyal.pdf, March 16–18, 2016

[4] Ittay, E., Sirer, E. G. Bitcoin-NG: A Secure, Faster, Better Blockchain. http://hackingdistributed.com/2015/10/14/bitcoin-ng

[5] Latanya Sweeney. k-anonymity: A model for protecting privacy. International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems, 10(05):557–570, 2002.

[6] M. Castro and B. Liskov. Practical Byzantine fault tolerance and proactive recovery. ACM Transactions on Computer Systems, 20(4):398–461, Nov. 2002.