Yobicash

A cryptocurrency for storing and sending encrypted data

Christian Nyumbayire
revised by Amelia Tomasicchio
March, 30, 2017

Abstract

Yobicash is a cryptocurrency for storing, managing and sending encrypted data. Data ownership and user's privacy are secured thanks to cryptography. Data replication and a memory hard proof-of-work ensure data availability and consistency in the presence of untrustable nodes. A new way to build blocks that makes it possible to register more than ten thousand transactions per second.

1. Introduction

Yobicash is a cryptocurrency that let users store, transfer and delete data in a verified and confidential way through a Bitcoin-like blockchain.
Thanks to encryption, only data signers can read a document, ensuring privacy of the owners.
Data can be co-owned and managed by groups of users with different permission levels and rights. Some users may be just allowed to read, others may have the right to write, delete and transfer contents, others may have to win weighted elections with their peers in order to receive permission to do any action.

To ensure its use value, Yobicash prioritize throughput and scalability, processing 10 thousand transactions per second for megabyte per block.

1.1. Historical background

Everything started back in 1988, when Mark S. Miller and K. Eric Drexler published three revolutionary papers appeared in "The Ecology of Computation" [1]. They envisioned trustless computer networks that were able to share data and computation in exchange of economical returns. The security - which any

common distributed system could not achieve - would have been granted by decentralization and market process. They called it the *agoric system*, from the ancient Greek for market, *agorà.*

In 2008 Satoshi Nakamoto, the pseudonymous user or group, published the paper of a cryptographically enforced digital currency called Bitcoin [2]. From then on, this currency tokens are produced through a shared computation effort called *mining* and transferred through a public shared ledger of transactions: the *blockchain*.

Bitcoin was the first successful embodiment of an agoric system thanks to its strength to provide a cryptographic and transparent environment.
Since its development Bitcoin has been the most successful cryptocurrency, reaching a market capitalization of 12 billion dollars.
After Bitcoin, new cryptocurrencies have been created, each one with a different market proposition.

This ecosystem of agoric systems based on the blockchain (from now on we'll call them "blockchains") is maturing and giving more and more returns.

Hopes are that the blockchain will change the face of every market and government function by reducing verification and network costs [3].
Although the high hopes, the blockchains we daily experience are still too specific in use, centralized and cannot scale to reach global usage.

1.2. Generality

Factom [4] and Ethereum [5] made attempts of generality and they could exemplify the way the above issues have been addressed so far: generalizing data or generalizing computation.

Factom incentivizes nodes to store files whose *checksum* – a cryptographic proof of existence – is shared on a blockchain. To address scalability issues, Factom does not compel nodes to host all the files, which are declared in its blockchain. This means that data access and ownership may not be ensured. Its blockchain guarantees only the checksum existence, ownership and transfer.
To increase data availability, this kind of currencies try to incentivize hosting. They gain a greater partition tolerance, but they loose maintainability.

Although this drawback, the approach of generalizing data – blockchains for data – letscomputation be done offline and verified through a proof-of-process. In a proof-of-process system for code execution the blockchain takes care only of recording code, inputs, mid-steps and outputs, so to make everyone interested in verifying the whole computation process.

On the other hand, Ethereum attempts to generalize computation, so it has a blockchain for computation. The data stored on Ethereum are only those that are needed in order to the define its computation – the

so-called " smart contracts" - so the Ethereum blockchain has a specific semantics and it is limited in scope and size.
These contracts are written in a specific programming language, Solidity, and each node in the network runs the contracts when they are called and it checks nodes' results by comparing them with its owns. The burden of computation is shared among all the nodes. Data that exceeds the size and semantics of Ethereum have to be put offline, and referenced by the system some way.
Many have attempted to use Ethereum as a generic programming environment, building on it token systems and complex contracts. Because of its limitations and the lack of safety of its programming language, Ethereum is more famous for its hacks [6] and missed chances [7, 8].

1.3 The Yobicash Answer

Differently from Factom and similar systems, Yobicash is a blockchain for data that allows users to put raw data directly on the blockchain, ensuring partition tolerance in a simple way. Data is treated as value and can be converted back to currency at any time, so it is not just a cost for nodes, but also an economic value.

Yobicash currency corresponds to what we commonly define a byte, in fact, it’s called a Yobicash byte. The multiples of a Yobicash byte are called as the “data” equivalent: Yobicash kilobyte, megabyte, gigabyte and so on.

Miners create Yobicash bytes, so then users can exchange them.
Users have the right to convert all or part of their Yobicash bytes into actual data, whose size has to be equal to the value of the converted bytes.
After the conversion, data gets automatically encrypted.

On the other hand, if a user wants to get its currency back, she/he can just convert the data back to Yobicash bytes. Then the converted data will be deleted from the blockchain, but some metadata remains, like the data checksum, in order to recognize the data when it’s found in the blockchain or outside it.

This way, if a user wants to store some data on the blockchain, he/she needs to send the data to him/herself, by executing a transaction with the equivalent amount of the data bytes in Yobicash.
The same thing happens if he/she wants to send a data to another user. In this case, he/she will have to do the same but sending the Yobicash transaction to the other user and not to him/herself.

Confidentiality is ensured through encryption while different scalability measures - the market for fees and the incentive of every node owner to invest in storage - make its usage sustainable over time.
Data can be co-owned by different signers having different voting weights. Signers with no voting weight will have no voice in the matter of data management, but will be able to see the decrypted data. Signers with the majority of the vote weights will have a major weight on management issues, if not the last word.

1.4 Decentralization

Decentralization assures resilience to the network and it is a guarantee of fault tolerance in case of geographically localized malicious actors interested in taking over the network or particular mistrusts among actors of different regions.

Although cryptocurrencies should be decentralized by nature, the increasing difficulty of a mining algorithm can favor geographically localized nodes and disfavor others. This kind of concerns has been moved against Bitcoin which ended to be prevalently mined in China. This centralization has increased the mistrust of many actors on the system.

Many cryptocurrencies addressed this problem by using mining algorithms which do not slightly favor anyone. Memory hard hashing algorithms are the most famous class from this group of alternative mining algorithms. Yobicash uses a modified version of randmemohash, by Damian Lerner [9].

1.5 Scalability

Scalability is paramount for a blockchain in order to reach global usage. Related to blockchains, it means the capacity of the network to increase the number of transactions per second when needed. VISANet can process 56,000 tps (transactions per second) [10], while Bitcoin a theoretical maximum of 7 tps [11] and a real maximum of 3 tps.

This feature is very important even for some applications to be viable, e.g. blockchain embedded financial markets and IOT applications.

Scalability was not a priority in the times of Satoshi, but he expressed the need to increase the size of transactions and blocks in future times [12]. Bitcoin, stacked to a limit of 1Mb for blocks and transactions, so it confirms 3 tps on average.

In order to address scalability issues, Yobicash blocks have a size limit, but it changes according to the usage.
Yobicash achieves 10 thousand transactions per megabyte of a block, which means that for a 4 MB block it can confirm 40 thousand transactions. The real limit on the real number of transactions per second is given by the actual average time for the execution of a single transaction.

This feature could open up to new applications in need of fast responses times.

2. Use cases

Yobicash is a means for securing data hosting, ownership and transfer in an unsafe channel – the cloud. Thanks to its generality and focus on scalability, privacy and decentralization, Yobicash allows for many use cases.

2.1. Safe hosting

Yobicash allows users to host data and co-own it with different voting weights. Some users may be allowed to be only readers, some may be considered as the editors while others may be defined administrators holding the majority of voting weights.

Data can be copied to other users by sending copies to them, while deletion are just conversions back to value.

2.2. Safe messaging

Messaging is just transferring data. Thanks to encryption, messaging is confidential by default.

2.3. Proof of existence

A proof of existence is an hosted checksum of a digital artifact or the artifact itself. While putting the entire data on the blockchain may be expensive, resorting it to its checksum may be a possible alternative when the host of the raw data is trusted. The transaction timestamp containing the proof may be considered as the timestamp of the digital artifact itself.

2.4. Proof of ownership

A proof of ownership or smart property is a signed proof of existence. By signing the proof, signer(s) declare to own the underlined digital artifact and, if present, its offline referee.
Trusted signers can co-sign the proof to reinforce the claim. The proof transfer from an user or group of users to another one represents a transfer of ownership.

2.5. Proof of authorship

A proof of authorship is a signed proof of existence. Differently from the proof of ownership, it is not transferable. This difference is neither enforced nor enforceable by the Yobicash protocol, but making it clear is a responsibility of the parties (e.g. by using particular semantics).

Proofs of authorship can be used to enforce intellectual property of digital products.

2.6. Proof of process

A proof of process is a series of chained proofs of existence or authorship, witnessing different phases of a continuous or discrete process. The process may range from the execution of an offline task by some human beings or robots to the execution of some code by some remote servers to the market data of a financial ticket.
The signer(s) of a step of a process are considered witnesses, verifiers or executors of that process.
A proof of process can be used for distributed crowd-sourcing, crowd-funding and task management. Multi-party computations can store their steps on the Yobicash blockchain using it as a safe channel.

2.7. Smart oracles

Smart oracles are proofs of existence or authorship made by trusted sources on raw data or checksums, dealing with particular events or general facts.
Financial and IOT applications can read time series data from smart oracles writing on the Yobicash blockchain.

2.8. Smart contracts

Smart contracts are proof of processes where the process is the execution of some code involving changes of rights, properties or balances of the parties involved. They are the coded version of offline contracts, run by servers online.
This contracts can reference as inputs data already present in the blockchain, or even future steps of some proven process. That data could it be registered by generally trusted parties like in smart oracles or be trusted just by the parties involved.
The executed code can be in any language the parties agree on, from JavaScript to Julia to Haskell. It is a responsibility of the parties choosing the language that fits more their needs.

3. Design

Yobicash design is similar to the Bitcoin's one, with some key differences to ensure simplicity, easy and confidential data management and scalability.

3.1 Overview

Yobicash is a Bitcoin-like peer-to-peer network. The network is composed by nodes, clients and wallets sharing transactions and blocks.

Nodes store and broadcast transactions and blocks. The data stored is used to build the blockchain, a verified chain of confirmed blocks. The blockchain has a predefined first block, the genesis block. Every block after the genesis one is selected according to a block selection algorithm, prioritizing blocks with more transactions and more data.
Nodes can choose to consume CPU and memory in the mining process, in return of a given amount of

Yobicash, the *coinbase*. This coinbase can be consumed only after an epoch, or 100 confirmed blocks. The coinbase amount decreases over time until the total amount of Yobicash produced reaches $2.1 * 2^{80}$, 2.1 yobibytes.

The mining algorithm used is a hardened version of randmemohash [9], a memory strict hashing algorithm. The consumed memory helps decentralization allowing common devices to be competitive in the mining process.
The algorithm has a varying difficulty to assure that the average block confirmation time is 17 seconds.

Users transfer value and data between themselves through transactions. They store transactions in personal wallets and send them to nodes in order to validate and include them in the blockchain.
Data can be converted back and forth to the value (the size of the data), which is denominated in bytes.

3.2. Cryptography

Yobicash relies on 4 cryptographic primitives: cryptographic hashes, public key signature, symmetric encryption and public key encryption. The following is not a complete specification of Yobicash, but delines its main features.

3.2.1. Cryptographic hashes

Cryptographic hashes are functions that map binary data of any size to equally sized collision-resistant (unique) short binary strings. Digests - the outputs of these functions - are used by Yobicash as checksums and IDs.

The cryptographic hashes used by Yobicash are SHA512 (also SHA3 or Keccak) and randmemohash [9].

Yobicash uses SHA512 to identify transactions, blocks and data in a certifiable way.
Digests obtained by hashing data are called *checksums*, while IDs are digest resulted from hashing entire transactions or blocks.
Randmemohash is used by miners to build the proof-of-work. This algorithm has the advantage of being memory hard, which means that it requires less CPU, so that even a common machine can compete in the mining process.

3.2.2 Public Key Signatures

A public key signature is a scheme for cryptographically signature of a binary data message. This scheme generates a couple of keys, the private and public keys, used accordingly for signing and verifying a signature. The private key (sk) must be kept in secret by its user, while the public key (pk) is publicly used to reference to its user and verify his signatures (sig).

The phases of a public key signature are: the keys generation (keygen) through a security parameter 1^{λ}, the actual signing process (sign) and the signature verification (verify).

$$\text{keygen}(1^{\lambda}) = (sk, pk)$$
$$\text{sign}(sk, msg) = sig$$
$$\text{verify}(msg, sig, pk) = 0/1$$

This scheme is used to sign and verify Yobicash wire messages and as a base for transaction weighted multisignatures.
Weighted multisignatures allow more users to sign a single message, and on Yobicash this is used to sign transactions.
According to this scheme, when the sum of the weights of the actual signers is greater than a user-defined threshold, the signed message is considered valid.

The scheme used by Yobicash is ed25519 [13], which is based on Schnorr signatures and on curve25519. This scheme is more efficient and safer than secp256k1 [14] used by Bitcoin and Ethereum, and produces smaller signatures.
Yobicash uses this scheme to define the users and/or groups addresses and to sign transactions and blocks.

3.2.3. Symmetric Encryption

A symmetric encryption is a cryptographic scheme for encrypting binary data. This scheme generates a secret key which is used to encrypt and decrypt data.

This scheme has 3 phases: generation of a secret key sk (keygen) through a security parameter 1^{λ}, encryption (encrypt) of a plaintext message msg and decryption (decrypt) of the resulting ciphertext ciph.

$$\text{keygen}(1^{\lambda}) = \text{sk}$$
$$\text{encrypt}(\text{msg}, \text{sk}) = \text{ciph}$$
$$\text{decrypt}(\text{ciph}, \text{sk}) = \text{msg}$$

Yobicash uses XSalsa20 as symmetric encryption scheme. This scheme is used to encrypt the data stored and sent by groups.

3.2.4. Public Key Encryption

A public key encryption is a cryptographic scheme for encrypting binary data. The schemes generates a pair of keys, a secret key and a public key, used to encrypt and decrypt data. Data is encrypted through the encryptor secret key and the decryptor public key; and it is decrypted through the decryptor secret key and the encryptor public key.

In this way communication confidentiality is ensured. In some cases, a public label may be used by both the parties to authenticate the data.

From a single user point of view, the scheme is composed by four phases: key generation (keygen) through a security parameter 1^{λ}, encryption (encrypt) of the plaintext message msg and decryption (decrypt) of the resulted ciphertext ciph.

$$\text{keygen}(1^{\lambda}) = (\text{sk}, \text{pk})$$
$$\text{encrypt}(\text{msg}, [\text{label},] \text{sk}, \text{pk}') = \text{ciph}$$
$$\text{decrypt}(\text{ciph}, [\text{label},] \text{sk}', \text{pk}) = \text{msg}$$

Yobicash uses curve25519XSalsaPoly1305 [15] as its public key encryption scheme.

Users encrypt the symmetric encryption key used to encrypt data by using this algorithm. Each user has different public key encryption keys for each of their own wallets.

3.3 Transactions

A transaction is a data structure used to register the transfer of value and encrypted data between groups of users.

Groups are sets of ed25519 public keys associated with a curve25519XSalsaPoly1305 encrypted XSalsa20 private key common to all group members and created by the transaction creator, a weight and a threshold for the signing process. Only if the sum of the signing members reaches the threshold the transaction is considered signed.

Money in Yobicash is denominated in bytes, so one byte in Yobicash money is convertible to one byte of encrypted data.

A transaction has a fee, which is collected by miners. Fees are defined per byte of a transaction. Miners add first the transactions with the higher fees, increasing their revenues. When the costs of maintaining a node are higher than their revenues, the fees value accepted by miners increases. Competition to win the rush for the coinbase drives miners to increase the number of transactions confirmed and the total amount of data registered (see “blocks”).
In order to do that, miners have to decrease their storage costs and increase their capacity, lowering the required fees. The result is that the fee market incentivizes the increase of the available storage and intend to keep fees low.

3.4 Blocks

Blocks are bundles of IDs of confirmed transactions plus a proof-of-work and an associated coinbase.

The use of IDs - instead of full transactions - increases scalability, ensuring that in 17 seconds (the time required for the confirmation of two blocks) and with 1 Mb block size, Yobicash can confirm up to 10000 transactions per second. Nodes won’t accept blocks having transaction IDs they can’t find in their store or by querying other nodes.

The proof-of-work system is built on the checksum of the serialized signed block (a binary representation of the block) in order to make the proof strictly linked to that block and to its signer. The coinbase is the mined value, and it depends on the height of the block and the amount planned to be produced at that height by a programmed money supply function.

Every block defines the size limit of blocks and transactions. This limit can grow by 10% if the sum of the previous block transactions is over 90% of the size of the previous block itself.

Blocks are selected according to their *activity*, or the product of the sum of their transactions and of the sum of their transactions size. The result is that miners are incentivized to: 1) increase the number of transactions, 2) increase the transaction sizes they register, 3) decrease storage costs and fees.

This, together with the increase of the working memory in order to be competitive in the mining process, has the effect of increasing scalability.

4. Future work

Yobicash is designed to be simple and to be used as a platform for more complex services. Security and scalability will remain the main areas of improvement to guarantee its value proposition.

4.1. More scalability

The current scalability measures may do a lot for some time, but throughput is never too high. The more can be made to raise throughput in order to maintain the trustless and Bitcoin-like structure, the better the users will be served. A lot could be done just through network simulations, fastening the development. Increasing scalability and sustainability will open to new applications, improving benefits for both users and the network.

4.2. Quantum Cryptography

Quantum computers threaten all the value stored by current cryptocurrencies. From public key signatures to currently used zero knowledge proofs to public key encryption, everything used so far except for Keccak hashes (sha3 hashes) will be easily broken [16].

Current research on post-quantum cryptography is still immature. A few attempts to suggest already usable algorithms [17] have been made, but a little bit of research will make it clear that many of the reported algorithms are far from being safe.

Nonetheless — when needed — switching to post-quantum cryptographic algorithms will be important. Supersingular isogeny elliptic curve cryptography and lattice based cryptography seem the more promising solutions so far.

5. Conclusions

Yobicash is currently the first attempt to make data a first class citizen within blockchains and to embody the idea of the blockchain as a general purpose technology. This is achieved by making data a value, ensuring its confidentiality, letting users co-own it and taking care of potential scalability issues.

This kind of blockchain – blockchain for data – enables a number of applications that ranges from simple notaries to complex multi-party computations.

More will have to be done in order to track and increase scalability and to ensure new alternatives to the current cryptography when quantum computers will be available. But right now the post-quantum cryptography we quoted above is still immature.

References

[1] M. S. Miller, K. E. Drexler, *The Ecology of Computation*, 1988
[2] S. Namamoto, *Bitcoin: A peer-to-peer electronic cash system*, https://www.bitcoin.org/bitcoin.pdf, 2008
[3] C. Catalini, J. S. Gans, *Some Simple Economics of the Blockchain*, 2016
[4] P. Snow, B. Deery, J. Lu, D. Johnston, P. Kirby, *Factom: Business Processes Secured by Immutable Audit Trails on the Blockchain*, 2014
[5] https://github.com/ethereum/wiki/wiki/White-Paper
[6] N. Atzei, M. Bartoletti, and T. Cimoli, *A survey of attacks on Ethereum smart contracts*, 2016
[7] http://blog.velocity.technology/velocity-is-shutting-downs
[8] G. Greenspan, *Why many smart contract use cases are simply impossible*, http://www.coindesk.com/three-smart-contract-misconceptions, 2016
[9] S. D. Lerner, *Strict Memory Hard Hashing Functions*, http://www.hashcash.org/papers/memohash.pdf, 2014
[10] https://usa.visa.com/dam/VCOM/download/corporate/media/visa-fact-sheet-Jun2015.pdf
[11] https://en.bitcoin.it/wiki/Scalability
[12] https://bitcointalk.org/index.php?topic=1347.msg15139#msg15139
[13] D. J. Bernstein, N. Duif, T. Lange, P. Schwabe, B. Yang, *High-speed high-security signatures*, 2011

[14] H. Mayer, *ECDSA Security in Bitcoin and Ethereum: a Research Survey*, 2016
[15] D. J. Bernstein, *Cryptography in NaCl*, 2009
[16] M. S. Brown, *Classical cryptosystems in a quantum setting*, 2004
[17] PQCrypto, *Post-Quantum Cryptography for Long-Term Security*, 2016