POLITECNICO
MILANO 1863

BLOCKCHAIN NOTARIZATION: EXTENSIONS TO THE OPENTIMESTAMPS PROTOCOL

Andrea Brandoli

Supervisors: Daniele Marazzina,
Ferdinando M. Ametrano

16th April 2019

INTRODUCTION

- Notarization of digital documents is traditionally achieved by a trusted certification authority.
- A notary public could be malevolent and represents a single point of failure.
- An application built upon a distributed system reaching consensus in a decentralized way can greatly strengthen the notarization process.
- The Bitcoin blockchain, the most reliable decentralized system, can be used as a notary in a procedure called timestamping.
- The open-source protocol OpenTimestamps aims to be a standard for timestamping and solves a scalability issue.

OUTLINE

Definition (distributed system)

A distributed system is a system whose components are located on different networked computers, which communicate and coordinate their actions by passing messages to one another. The components interact with one another in order to achieve a common goal.

Examples:

- ATM points.
- Computers in cloud.
- Modern smartphones.

DISTRIBUTED SYSTEMS: PROS VS. CONS

PROS:

- Scalability.
- Performance.
- Fault tolerance.
- Reliability.
- Availability

CONS:

- Security.
- Consensus.

Thus, the main challenge of distributed systems is to achieve distributed consensus in the presence of a number of faulty processes.

THE CONSENSUS PROBLEM

- Networks are composed by many agents, called nodes.
- Nodes are either honest or byzantine.
- They could also be correct and faulty, but not as alternatives to honest and byzantine.

Definition (consensus)

There are n nodes, of which at most f might crash, i.e., at least $n - f$ nodes are correct. Node i starts with a proposed value v_i. The nodes must decide for one of those values, satisfying the following properties:

- *Agreement: All correct nodes decide for the same proposal.*
- *Termination: All correct nodes terminate in finite time.*
- *Validity: The decision value must be the proposal of some proposer.*

BYZANTINE AGREEMENT IN AN ASYNCHRONOUS NETWORK

- Is it possible to achieve distributed consensus in an asynchronous network in presence of byzantine nodes ?
- The Byzantine General's Problem is the abstraction of such situation.

More precisely:

Definition (byzantine general's problem)
A commanding general must send an order to his $n-1$ lieutenant generals such that the following conditions are satisfied:

- *All loyal lieutenants obey the same order.*
- *If the commanding general is loyal, then every loyal lieutenant obeys the order he sends.*

IMPOSSIBILITY RESULT

- Achieving distributed consensus in an asynchronous network in presence of byzantine nodes is a very hard problem.
- Fisher, Lynch and Paterson [12] formally proved that it is impossible to reach distributed consensus in an asynchronous network with one faulty process, even for a binary variable.
- So how Satoshi Nakamoto reaches consensus in Bitcoin, a decentralized, distributed, peer-to-peer network ?

OUTLINE

BITCOIN AS AN EXAMPLE OF EVENTUAL CONSISTENCY

- Bitcoin is a distributed, decentralized, peer-to-peer electronic payment system based on cryptographic proof.
- The Bitcoin network is subjected to network partition.
- It is inherently characterized by a trade-off between:
    - Consistency: All nodes agree on the current state.
    - Availability: The system is operational and instantly processing incoming requests.
    - Partition tolerance: The ability to continue operating correctly even in the presence of a network partition.
- CAP theorem proves that is impossible to achieve simultaneously the three properties.
- Nakamoto’s intuition has been to relax the agreement property in the consensus definition to hold probabilistically, to reach eventual consistency.

DOUBLE SPENDING PROBLEM

- In a digital cash scheme, a single digital token, being just a file that can be duplicated, can be spent twice.

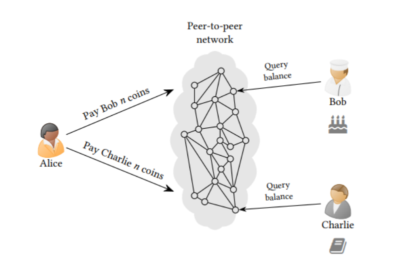

Figure 1: Illustration of the double spending problem.

- Nakamoto solves the double spending problem combining cryptography and social incentives. But first, some protocol's mechanics.

TRANSACTIONS IN BITCOIN

- A bitcoin is defined as a chain of digital signatures.
- Coins cannot be combined, subdivided or transferred, but only entirely consumed as transaction inputs (TxIn) to create new output coins (TxOut).
- A TxOut can be in two states: spent or unspent.
- The UTXO is the set of current unspent transactions.
- A TxOut consists of an amount of bitcoins and a locking script, a cryptographic puzzle which determines the spending conditions.
- A TxIn consists of a pointer referencing the UTXO that it consumes and an unlocking script, the solution to solve the locking one.

THE BLOCKCHAIN

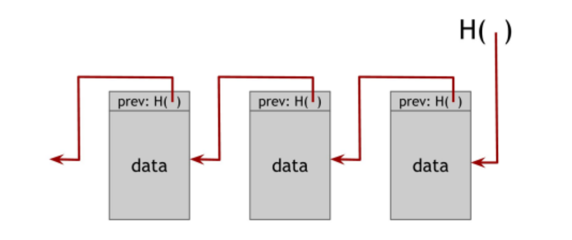

Figure 2: Blockchain as a hash pointer linked list.

- Transactions are recorded in a distributed ledger, the blockchain, formally a hash pointer linked list.
- The cryptographic link between blocks requires computing power to be created.
- A block is valid only if it includes valid transactions.

MINING & PROOF OF WORK

- All network nodes validate and clear all transactions.
- Special nodes, called miners, compete to finalize a new block of transactions, providing proof-of-work, which consists of finding a special number x called nonce s.t.

$$\mathrm{SHA256}(\mathrm{SHA256}(\underbrace{\ldots \| \mathrm{prev\_block\_header\_hash} \| \ldots \| x}_{\text{Candidate Block Header}})) < \frac{2^{224}}{d}.$$

- The miner who first find it is rewarded with the issuance of new bitcoins in a special coinbase transaction.
- Miners solve the double spending problem:
  - A double spending transaction invalidates a block.
  - The bitcoin reward would have removed.
  - The winning miner would have wasted his work.
  - Proof-of-work ensures blockchain immutability.

OUTLINE

NOTARIZATION

- Notarization is the official fraud-deterrend process that assures the parties of a transaction that a document is authentic and can be trusted.
- Traditionally, it is achieved by a trusted certification authority which represents a single point of failure.
- Need to decentralize the source of trust to be reliable even if the security of such central authority is violated, using the Bitcoin blockchain as a notary for timestamping.

Definition (timestamp)

A timestamp is a proof that some data d existed prior to a certain time t and in a certain state.

- To create a timestamp, data d has to cause an event that could not have been generated without the existence of d, must be attested to time t and can be publicly observed.
- A good timestamp must become invalid even if a single bit of the input data is modified.
- What is published on the blockchain is a commitment to d.

Definition (commitment operation)

A function $C : X \to Y$ is a commitment operation if given $x_1 \in X$ it is not feasible to compute $x_2 \in X$ s.t. $x_1 \neq x_2, C(x_1) = C(x_2)$.

Examples:

- append: "hello" $\xrightarrow{\text{append("world")}}$ "helloworld".
- prepend: "world" $\xrightarrow{\text{prepend("hello")}}$ "helloworld".

HASH FUNCTIONS

- However, both “append”and “prepend”commitment operations are not good for timestamping:
  - They do not hide input data.
  - Outputs are always bigger in size.
- Hash functions map input data of arbitrary length into hash values, i.e., outputs of fixed length:
  - Preimage resistant (one-wayness): given $h(x)$ it is not feasible to compute x.
  - Second-preimage resistant: given x it is not feasible to compute y s.t. $x \neq y$, $h(x) = h(y)$.
  - Collision resistant: it is not feasible to find x, y s.t. $x \neq y$, $h(x) = h(y)$.
- An hash value represents a digital fingerprint.
- Bitcoin uses SHA256: outputs have a fixed size of 256-bit.

TIMESTAMPING PROCEDURE

- A generic data file can be hashed to produce a short unique identifier.
- Such a digital fingerprint can be associated to a particular Bitcoin transaction, called null data transaction.
- These transactions make use of OP_RETURN, an opcode script which allows to add up to 80 bytes of arbitrary data in a provably unspendable transaction.
- Such transaction commits to a specific field of the block header, called merkleroot, which is the root of a binary tree that summarize all the transactions in a block.
- Blockchain immutability provides timestamping, proving the data file existance at that moment in time in that specific status.

OUTLINE

OPENTIMESTAMPS

- Blockchain timestamping have some downsides:
  - Not efficient: it requires one transaction per document.
  - Lack of standardization.
- OpenTimestamps is an open-source protocol that aims to be a standard for blockchain timestamping.
- It defines a set of rules for conveniently creating provable timestamps and later independently verifying them.
- A proof made with OpenTimestamps consists in a list of commitment operations applied in sequence to the document, ending with one or more time attestations.
- It proposes a solution to the scalability problem.

OPENTIMESTAMPS AS A SCALABILITY SOLUTION

- OpenTimestamps allows to aggregate up to an infinite number of document in a single transaction.
- The trick is to compose documents in a merkle tree and to timestamp only the root of that tree.

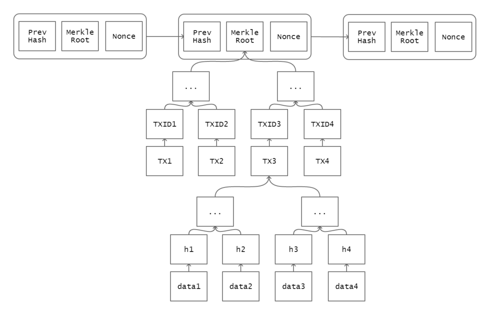

Figure 3: OTS scalability solution.

OPENTIMESTAMPS AS A SCALABILITY SOLUTION

- Moreover, aggregation servers collect hash values and periodically aggregate them in a single merkle tree.
- Aggregation servers are efficient but not convenient: to obtain a proof you must wait for the transaction to be included in a block.
- To provide proofs almost instantly, OpenTimestamps makes use of public calendar servers.
- Aggregation servers submit the merkleroot to a calendar server, which promise that every submitted digest will be timestamped with Bitcoin.
- Proofs made this way are incomplete, but they can be upgraded once the blockchain has completed the timestamp, adding the path up the the block header.

A PRACTICAL USE CASE: AIM OF THE PROJECT

- The author, in partnership with DGI (Digital Gold Institute) and ANIA (Associazione Nazionale fra le Imprese Assicuratrici) worked on a practical use case of blockchain timestamping.
- The aim of the project is to provide a fully operating timestamping service.
- It is build upon OpenTimestamps, while marginally improving and extending such protocol with additional features.
- For example, any insurance company could make use of it to grant authenticity of its associates policies.
- Simply, it could be useful whenever a digital signature is involved.

ARCHITECTURE OF THE SOLUTION

- The architecture of the solution consists of three cloud-servers, accessible from outside via traditional network elements (DNS server, reverse proxy, firewall, etc.).
- Such elements transparently remap host names under the domain “aniasafe.it”.
- The three servers are divided in a front-end server and two back-end servers:
  - Front-end: hosts the public web interface (`https://timestamp.aniasafe.it`).
  - Back-end: they both run a calendar server connected to a local Bitcoin node: (`https://calendar.aniasafe.it`) for mainnet, (`https://test-calendar.aniasafe.it`) for testnet.

ARCHITECTURE OF THE SOLUTION

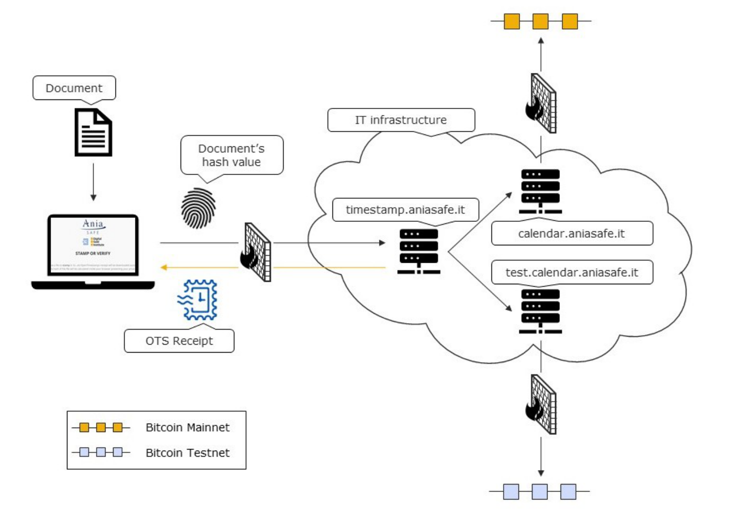

Figure 4: Architecture of the solution

WEB INTERFACE

- The front-end server hosts the web interface.
- The underlying JavaScript library defines the main functions invoked when a user interacts with the web interface: “stamp”, “verifyTimestamp”and “upgradeTimestamp ”.
- The JS library has been extended for additional features:
  - Testnet as an alternative chain.
  - Multi-validation: the original protocol supports timestamping on multiple chains, but the verification of one attestation at a time.
  - Esplora (block-explorer): a new class has been created to query Esplora’s API correctly.

WEB INTERFACE: STAMPING

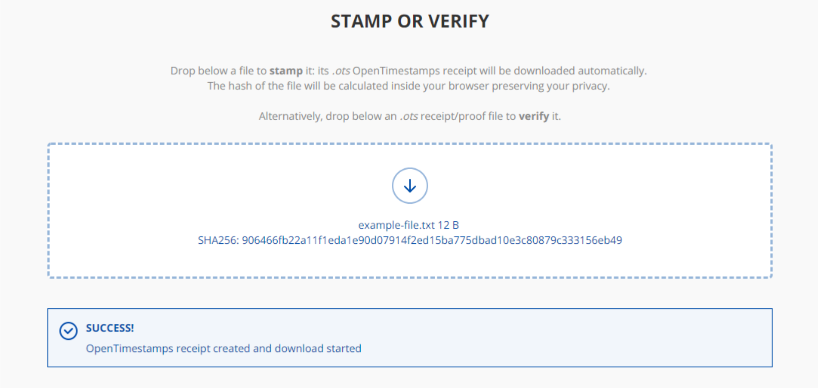

Figure 5: Stamping

WEB INTERFACE: VERIFYING

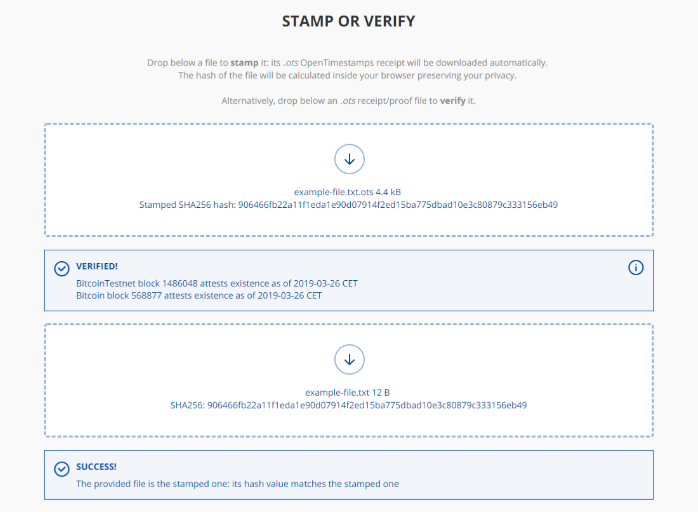

Figure 6: Multi-validation

OUTLINE

CONCLUSIONS AND FUTURE WORK

- The notarization process can be strengthen by means of a decentralized solution built upon a blockchain technology.
- The Bitcoin blockchain can be used as a notary for timestamping.
- Timestamping provides only proof-of-existence at a given date; it does not convey authorship, non-repudiation, veracity.
- OpenTimestamps provides a standardization for timestamping and it is scalable.
- Elliptic curve commitments could improve the timestamping process, allowing to push data into the payee public key or in a digital signature, avoiding fees.

REFERENCES i

[1] Bitcoin developer guide. **https://bitcoin.org/en/developer-reference**.

[2] Blockstream official website. **https://blockstream.com/**.

[3] Digital gold institute website. **https://www.dgi.io/**.

[4] Javascript library source code. **https://github.com/federicoon/javascript-opentimestamps**.

[5] Opentimestamps source code. **https://github.com/opentimestamps**.

[6] Opentimestamps website. **https://opentimestamps.org/**.

[7] Christopher Allen. Learning-bitcoin-from-the-command-line source code. https://github.com/ChristopherA/Learning-Bitcoin-from-the-Command-Line.

[8] Andreas M. Antonopoulos. ***Mastering Bitcoin: Programming the Open Blockchain.*** O'Reilly Media, Inc., 2nd edition, 2017.

[9] Adam Back. **Hashcash - a denial of service counter-measure.** Technical report, 2002.

[10] Michael Ben-Or. **Another advantage of free choice (extended abstract): Completely asynchronous agreement protocols.** In *Proceedings of the Second Annual ACM Symposium on Principles of Distributed Computing*, PODC '83, pages 27–30, New York, NY, USA, 1983. ACM.

[11] Leonardo Comandini. **sign-to-contract: how to achieve trustless digital timestamping with zero marginal cost.** Master's thesis, Politecnico di Milano, 2018.

[12] Michael J. Fischer, Nancy A. Lynch, and Michael S. Paterson. **Impossibility of distributed consensus with one faulty process.** *J. ACM*, 32(2):374–382, April 1985.

[13] Stuart Haber and W. Scott Stornetta. **How to time-stamp a digital document.** *Journal of Cryptology*, 3:99–111, 1991.

[14] Leslie Lamport, Robert Shostak, and Marshall Pease. **The byzantine generals problem.** *ACM Trans. Program. Lang. Syst.*, 4(3):382–401, July 1982.

[15] Satoshi Nakamoto. **Bitcoin: A peer-to-peer electronic cash system, 2009.**

[16] Arvind Narayanan, Joseph Bonneau, Edward Felten, Andrew Miller, and Steven Goldfeder. ***Bitcoin and Cryptocurrency Technologies: A Comprehensive Introduction.*** Princeton University Press, Princeton, NJ, USA, 2016.

[17] Christof Paar and Jan Pelzl. ***Understanding Cryptography: A Textbook for Students and Practitioners.*** Springer Publishing Company, Incorporated, 1st edition, 2009.

[18] Yee Jiun Song, Robbert van Renesse, Fred B. Schneider, and Danny Dolev. **The building blocks of consensus.** In *Proceedings of the 9th International Conference on Distributed Computing and Networking*, ICDCN'08, pages 54–72, Berlin, Heidelberg, 2008. Springer-Verlag.

[19] Peter Todd. **Opentimestamps: Scalable, trust-minimized, distributed timestamping with bitcoin.** `https://petertodd.org/2016/opentimestamps-announcement`, 2016.

[20] Saravanan Vijayakumaran. **The security of the bitcoin protocol.** `https://www.zebpay.com/pdf/Bitcoin-Security-White-Paper.pdf`, 2018.

[21] Roger Wattenhofer. ***The Science of the Blockchain.*** CreateSpace Independent Publishing Platform, USA, 1st edition, 2016.