Lesson 12: BLOCKCHAIN: INTRODUCTION
31/3/2021
this course is on blockchains, from a

- scientific prospective
- we have not “to sell blockchains”!

so, let us begin with a general introduction to blockchains

- not only cryptocurrencies
  - other use cases of the blockchain
  - different consensus algorithms, not only PoW
  - show how the crypto tools introduced so far, are used for blockchains

in the next lesson, we will come back to cryptocurrencies

- cryptocurrencies are the killer application, after all
- the Bitcoin ecosystem
BLOCKCHAIN: BASIC CONCEPTS

- what is a ledger?
- consensus in a distributed environment
- tamper freeness
- proof of ownership
- permissioned and permissionless blockchains
THE FIRST ABSTRACTION: THE LEDGER

- a ledger
  - like a bulletin storing operations
  - maintains the order of operations
- which properties needed for a ledger?
  - append-only list of events
  - tamper-proof
    - auditability
  - everyone agrees on content
    - consensus
- not just financial!
  - any application which needs a log of events
THE FIRST ABSTRACTION: THE LEDGER

- Alice has a company
- she works as an intermediary between retail and wholesale
- needs a ledger to log the asset/values transfers
- several entities will access the ledger
  - wholesales sends good to Alice
  - Alice transfers good to retails
- ledger registers all these operations

Alice has a company

she works as an intermediary between retail and wholesale

needs a ledger to log the asset/values transfers

several entities will access the ledger

wholesales sends good to Alice

Alice transfers good to retails

ledger registers all these operations
THE LEDGER AS A BLOCKCHAIN

- if the ledger is organized as a list of blocks
- call it a blockchain
- but other structures are possible! For instance, graphs

A simplification: blocks contain single operations.
consensus is the mechanism which defines

- who decides which operation will be added to the blockchain
- which operation among those to be confirmed, will be added
A TAMPER FREE LEDGER

- compute the hash of each entry (block)
- store in each entry the predecessor's hash
- if one entry is tampered
  - need to recompute the hash of all the following ones
  - this must be computationally hard
  - hash the proof of work with the block
- enables auditability
WHAT IS CONSENSUS?

each node presents an item to add to the ledger
WHAT IS CONSENSUS?

- consensus is agreement on the same value
- nodes agree on one of the nodes's input
- validity: agree on someone's proposal
- but, faulty or malicious node may exist
CONSENSUS IS IMPORTANT!

- from Micali's talk: “what is really important in a blockchain? “It is how the next block is chosen, because if this is in the hand of a few entities, or even worst, of one entity, these entities have more power than Luigi XIV”
  - is the level of participation in the block generation that makes a blockchain really decentralized
  - a jungle of consensus algorithms proposed in the last years
  - the “yellow cloud” can be implemented in several ways
several challenges:

- maintain consistency in presence of different network jitter, delay,...
- the main challenge is that some parties can cheat: byzantine parties
- a classical result: if the “majority is honest”, the system works well
- but which notion of majority???
CONSENSUS IN A DISTRIBUTED NETWORK

- assume **honest majority**: nodes which follow correctly the protocol
- implement consensus by **voting**
  - broadcast every operation on the network and then collect votes

<table>
<thead>
<tr>
<th>Transaction id</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>ddb521239864k...</td>
<td>0.084 BTC</td>
</tr>
<tr>
<td>edd98763hn3nr...</td>
<td>1.2 BTC</td>
</tr>
<tr>
<td>mk8765g4g2j3...</td>
<td>0.036 BTC</td>
</tr>
</tbody>
</table>

- how to implement voting?
- a very well known problem in distributed systems
BEWARE OF THE SYBIL ATTACK!!

- I am the Delphic Sybil!
- I like voting a lot!
in cryptocurrencies, who prevents someone to spend twice or more times the same digital currency, if no central authority is present?

- bits are also easier to copy than paper!
- difficult to avoid if the network is under a sybil attack
DOUBLE SPENDING AND SYBILS

- Alice
  - double spends the same bitcoin with B and C
  - performs a Sybil attack
    - assume more identities: easy in Bitcoin
DOUBLE SPENDING AND SYBILS

• store transaction in a ledger: each operation approved through consensus

• Alice will approve with her multiple identities the double spending of herself with Bob and Charlie

• the attack is successful
CONSENSUS IN POW BASED BLOCKCHAINS

• how to define majority in a context where everybody can easily join the network and assume multiple identities?
• if simply count the number of votes....
• easy to control the majority of the network by assuming multiple identities

• instead, use the computing power controlled by each identity.
• creating multiple identities is not useful, if you own a single computer
PROOF OF WORK

• needs voting scheme that is hard to fake!

• proof of Work
  • Dwork & Naor 1993
  • requires a lot of computation to fake

• winner of the lottery decides which is the next node of the blockchain
  • like a lottery to choose which node will decide the next block
    • tickets of the lottery are very expensive (proof of work)
    • winner of the lottery is paid when later winner endorses validity

• give incentives for well behaviour

• Sybil attacks expensive and pointless
A TAMPER FREE LEDGER

- this happened | hash
- this happened | hash
- this happened | hash

POW
- this happened
- this happened
- this happened
POW VERSUS CLASSICAL CONSENSUS

- PoW Consensus
  - might pick multiple winners at the same time, every now and then
  - what happens if near-simultaneous winners win the lottery?
    - this causes "forks": two winners attach their new block to the blockchain
    - two new blocks linked in parallel with the previous block of the blockchain
    - infrequent, but not impossible
  - later winner decide which forks wins
  - all entries of the chain start
    - provisional
    - become stable, merges over time
  - that is not classical consensus: classical approaches
    - establish a unique winner
    - once consensus is reached, it is permanent
Alice now decides to change work and opens a restaurant

- rental is high, venture capitalists are greedy
- Alice uses an ICO (Initial Coin Offering)
  - proposes a project that will be implemented on a blockchain
  - get fundings from people proposing to participate to the project
  - create tokens to be given to the funders, as a compensation
  - *cryptocopupons*: discount meals when the restaurant opens
PROOF OF OWNERSHIP

- Alice uses a ledger to register token transfers.
- Needs a solution to guarantee ownership of coupons:
  - How Alice can prove that she owns a coupon?
  - How a funder that is going to spend a coupon can prove that he/she has received it and he/she owns it?
- No certification authority:
  - No centralized entity certifying the identities
- A completely decentralized solution using asymmetric key cryptography
PROOF OF OWNERSHIP

- Alice generates a pair (public key, private key)
- anyone who knows the private key matching the public key of Alice, for instance af876f536..........., owns the Alice cryptocoupons
  - Alice themself
  - everyone Alice gives the private key to
  - everyone who steal Alice's private key
- private key gives ownership
  - possibility to sign the transfer operation
- public key gives the proof of ownership
  - prove that the emitter of the transfer is really the owner of the coupon
- register on the ledger the signed transactions
  - can be verified by the receiver
Alice decides to transfer 50% ownership of a coupon to each one of two different founders. She has a private key matching the public key af876f536..........

- Identifies the public key of the users she wants to transfer the coupon to:
  - 1FE1W2EEJE....
  - A5d65ab38.....

- Signs the transfer to prove she knows the private key and is therefore authorized to do the transfer.

- Transfers the 50% ownership:
  - To each one of two owners of the private key
  - Which corresponds to the previous public keys.

- The two users exploit their private key to collect and use half of the coupon.
SPENDING COUPONS

- no notion of account, balances, etc, like in a bank
- only coupon transfers
- ownership of coupons stored in the ledger
  - a user can retrieve all its coupons scrolling the ledger
- spending a coupon destroys it
- UTXO: unspent transaction output
  - a structure recording all the coupons owned by a user and not yet spent
Who stores the blockchain?

- Alice does not want to host her blockchain
  - expensive
  - customer may not trust her
use a peer to peer network between all the users and Alice

the blockchain is replicated by each node

someone has to enforce operations validity, voting for the next operations to add to the blockchain

these are the miners

partecipate to the lottery

solve the proof of work

but why I should be a miner, since a need computing power and this costs a lot?

Alice offer extra coupons to the miners to maintain the network

a rewarding system to reward miners
Alice's cryptocoupons are permissionless

- anyone can participate
- anyone can be a miner
- no central authority
- based on reward
- may have some problems
  - version 0.7/0.8 fork
  - blocksize forks
  - $54M DAO Attack
A PERMISSIONED BLOCKCHAIN

- Alice sells her restaurant and opens a frozen yogurt business
- but her business is in trouble
  - shipments arrive melted
  - where is the problem?
ALICE SUPPLY CHAIN

Bob’s truck

Carol’s factory
1. I never transported that yogurt.
2. It was melted when I got it from Carol.
3. It was OK when I delivered it to Alice.
What does Bob say?

Carol’s factory
PUT A LEDGER IN THE CLOUD

- a distributed ledger to record the events
- put the ledger in the cloud

Bob and Carol

Sensors
A new consensus algorithm
- Practical Blockchain Fault Tolerance (PBFT)
- no problems with Sybils
- actually, based on voting
PERMISSIONED BLOCKCHAIN FOR SUPPLY CHAIN

-11°C at 13:10

-10°C at 13:20

Carol handoff to Bob, 13:30

PBFT

Bob signed

Alice signed
PERMISSIONED BLOCKCHAIN: WHAT IS DIFFERENT?

- parties have identities
- humans have passwords, keys
- sensors have keys
  - both humans and sensors are authenticated
- no Sybil attack
- different consensus mechanisms
- accountability if caught cheating
Practical Byzantine Fault Tolerance

Miguel Castro and Barbara Liskov
Laboratory for Computer Science,
Massachusetts Institute of Technology,
545 Technology Square, Cambridge, MA 02139
{castro, liskov}@lcs.mit.edu

Abstract
This paper describes a new replication algorithm that is able
to tolerate Byzantine faults. We believe that Byzantine-
fault-tolerant algorithms will be increasingly important in
the future because malicious attacks and software errors are
increasingly common and can cause faulty nodes to exhibit
arbitrary behavior. Whereas previous algorithms assumed a
synchronous system or were too slow to be used in practice,
the algorithm described in this paper is practical: it works in
asynchronous environments like the Internet and incorporates
several important optimizations that improve the response
time of previous algorithms by more than an order of magnitude. We
implemented a Byzantine-fault-tolerant NFS service using our
algorithm and measured its performance. The results show that
our service is only 3% slower than a standard unreplicated NFS.

and replication techniques that tolerate Byzantine faults
(starting with [19]). However, most earlier work (e.g.,
[3, 24, 10]) either concerns techniques designed to
demonstrate theoretical feasibility that are too inefficient
to be used in practice, or assumes synchrony, i.e.,
to rely on the synchrony assumption for correctness, which
is dangerous in the presence of malicious attacks. An
attacker may compromise the safety of a service by
delaying non-faulty nodes or the communication between
them until they are tagged as faulty and excluded from the
replica group. Such a denial-of-service attack is generally
easier than gaining control over a non-faulty node.

Our algorithm is not vulnerable to this type of
attack because it does not rely on synchrony for
safety. In addition, it improves the performance of
Rampart and SecureRing by more than an order of
magnitude as explained in Section 7. It uses only one
round trip to execute read-only operations and uses
a scheme based on message authentication and public-key
Open and Permissionless Blockchains

- Bitcoin
- Ethereum
- Steemit

Permission-based Blockchains

- Hyperledger
- Ethereum Quorum
- Ripple
- Stellar