BLOCKCHAIN AND DISTRIBUTED LEDGER TECHNOLOGIES

Lecture 7 – Blockchain Security

Volker Skwarek
Hochschule für angewandte Wissenschaften Hamburg

WORLD BANK GROUP

VEERMATA JIJABAI TECHNOLOGICAL INSTITUTE
V.J.T.I.
ESTD 1887
MUMBAI 400019

DeLight Chain

HAW HAMBURG

LEARNING OUTCOMES

After this lecture students shall

- **know**
  - about security issues of different blockchains
- **be able to**
  - identify security problems
  - Propose defense approaches

CRYPTOGRAPHY AS SECURITY ENABLER (1/5)
CLASSIFICATION OF CRYPTOSECURITY

1. complete hack: a cryptoanalytic explores the key
2. global deduction: a similar algorithm to A can be derived without knowledge of the key
3. punctual or local deduction: a cyphertext can be decyphered into plaintext
4. information deduction: only partial information about the message or the key can be derived

Measure of security: algorithmic complexity. Ex.: a complexity of 2^{128} requires 2^{128} computations.
At 1 mio. ops/s at 1 mio. parallel processors, it requires 10^{19} years to try out all operations
→ billion times the age of the universe.

Security can be reached by
- complexity of data
- computational complexity
- memory requirements

CRYPTOGRAPHY AS SECURITY ENABLER (2/5)
BASIC ATTACKING PRINCIPLES

- cyphertext C only:
  get encrypted messages E(P) and decypher them to P – or even better: restore the keys K
  given: $C_1 = E_k(P_1), C_2 = E_k(P_2)$, wanted: P, algorithm E, keys K
- known plaintext P:
  analyse plaintext P and encrypted messages E(P), restore keys K
- chosen plaintext:
  enforce arbitrary plaintext messages and encrypted messages, restore keys
- adaptive chosen plaintext:
  enforce arbitrary plaintext messages and encrypted messages as a reaction according to previous messages, restore keys
- chosen cyphertext:
  feed arbitrary cyphertext into cryptosystem and obtain deciphered result, find algorithm, keys
- brute force:
  try out and find a solution

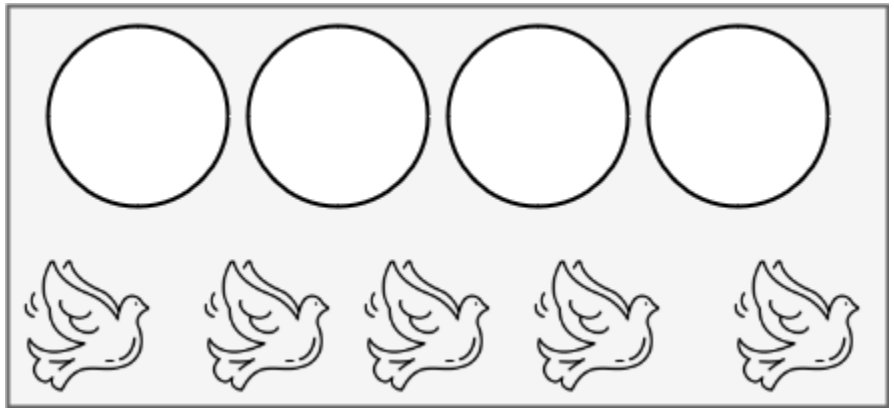
CRYPTOGRAPHY AS SECURITY ENABLER (3/5)
HASH FUNCTIONS

- Hash functions are used to exchange message securely.
- Hash functions need the following properties:

1. **Arbitrary message size:** hash function can be applied to any message size and message sizes can be handled
2. **Fixed output length:** hash functions create a fixed-length output
3. **Efficiency:** hash function is easy to compute in forward direction

4. **Preimage resistance**
5. **Weak collision resistance**
6. **Hard collision resistance**

CRYPTOGRAPHY AS SECURITY ENABLER (4/5)
HASH FUNCTIONS – PIGEONHOLE PRINCIPLE



With n pigeons and n-1 holes there will be at least one hole occupied by 2 pigeons.

- Fixed hash-function outputs of *n* bits

  2^n possible hash function output values

  Threat to weak collision resistance

How many people are needed in a room such that there is a reasonable chance that at least two people have the same birthday?

Piecewise approach:

1. Two people in the room: P(no collsion among 2 people)$= \left(1 - \frac{1}{365}\right)$
2. Three people in the room: P(no collision among 3 people)$= \left(1 - \frac{1}{365}\right) * \left(1 - \frac{2}{365}\right)$

.

.

n. N people in the room: P(no collision among N people) = $\left(1 - \frac{1}{365}\right) * \left(1 - \frac{2}{365}\right) \ldots * \left(1 - \frac{N-1}{365}\right)$

365 days a year lead to at least one collision with N=366 people

e.g. P(minimum one collision) = 1-P(no collision) = $1 - \left(1 - \frac{1}{365}\right) * \cdots * \left(1 - \frac{23-1}{365}\right) = 0.507 \sim 50\ \%$

PROOF OF WORK (1/6)

- **Proof of Work** is
  - a piece of data which is difficult to find but easy for others to verify and which satisfies certain requirements
  - a reward system to incentivize only one in the cyberspace to give a privilege to produce a block and get a reward

- Drawbacks of PoW:
  - Requires a big investment to be a miner
  - Consumes unnecessary electricity!
  - Handle only 3 transactions per second
  - Poor to resist double-spending various malicious attacks, and censorship attacks

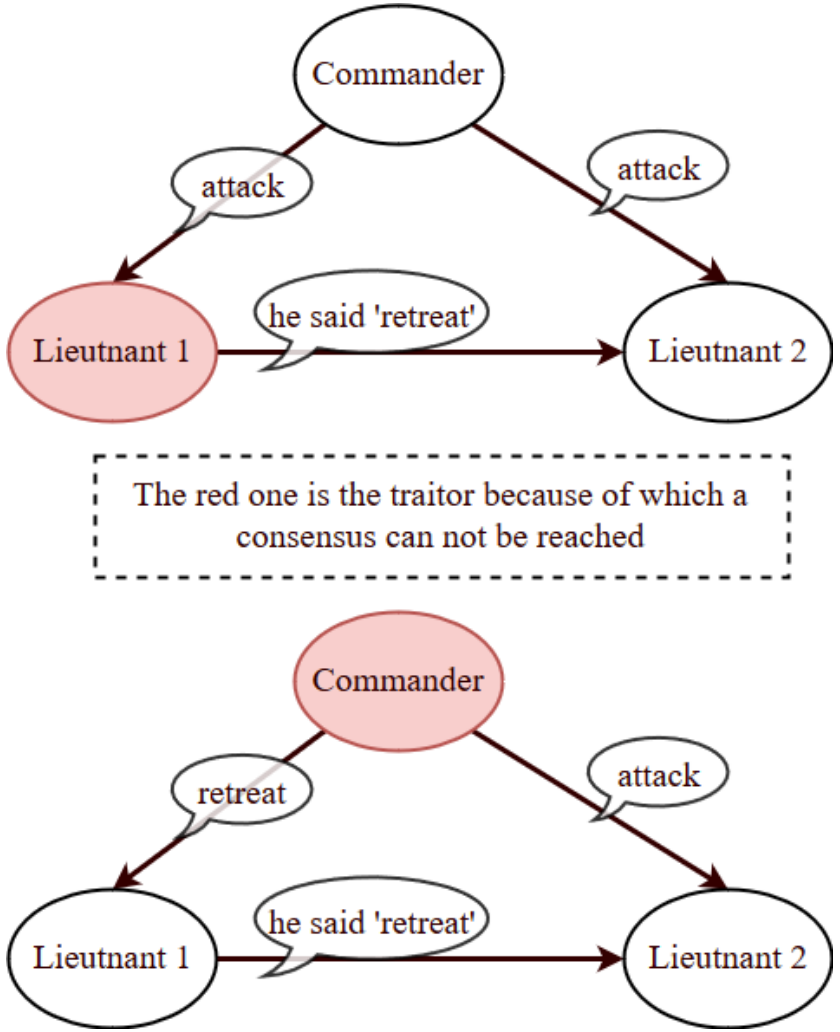
PROOF OF WORK (2/6)
BYZANTINE GENERALS PROBLEM

Problem:

- Impossibility results

Bitcoin‘s Solution:

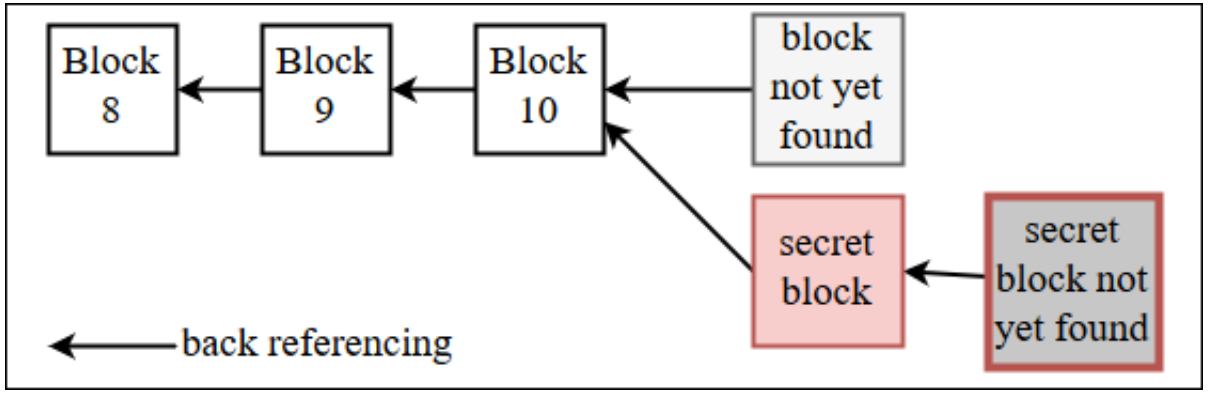
- Incentives
- Randomness



PROOF OF WORK (3/6)
SELFISH MINING

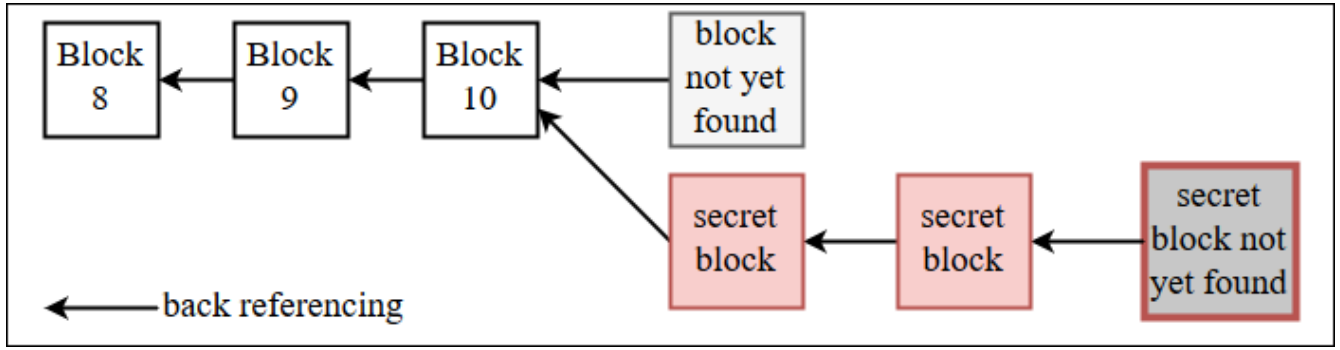
Selfish mining/ block-withholding:

- After you found a block, you do not announce it but keep it secret.
- You try to find a second block, before the network finds a block



The network has been fooled if you find a second block and keep it secret.

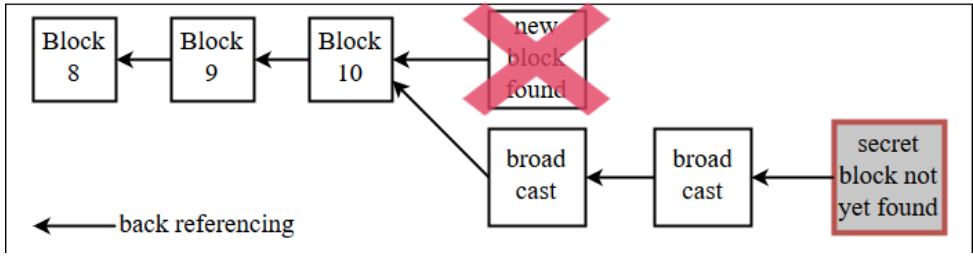
- You continue mining on your own chain
- BUT the network believes it is still mining on the longest proof of work chain



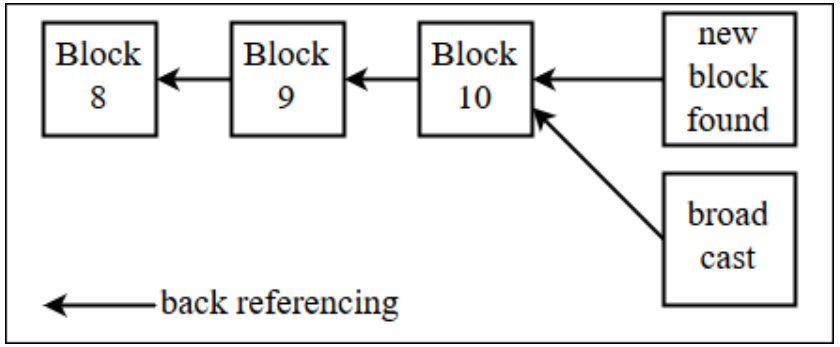
PROOF OF WORK (4/6)
SELFISH MINING

You keep your blocks secret until the network also finds a block. Then you broadcast your secret blocks and make the network block invalid.

- While the network was working on the invalid block, you had time to mine by yourself.
- You get the revenue for more than one block
- You get the higher effective proportion of hashrate, so you can expect **higher profits!**

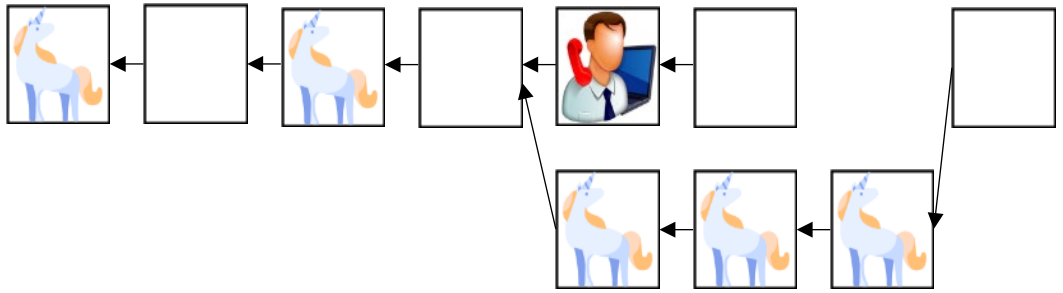


- If your chance to win the race is 50%, then malicious strategy is more profitable if you have more than 25% of the mining power
- If you have more than 33% of the mining power, the malicious strategy is more profitable even if you lose the race

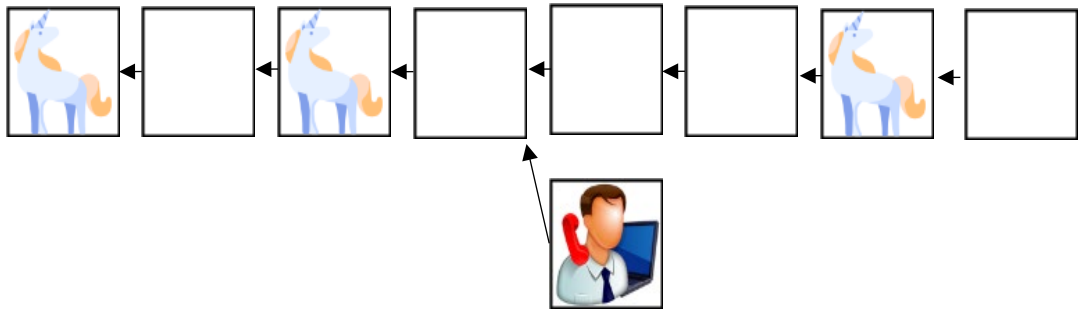


PROOF OF WORK (5/6)
BLACKLISTING BY PUNITIVE FORKING

- Wonderland owns more than 51% of the networks hashrate, than if non-Wonderland miners include transactions from Max Normal in a block, Wonderland will fork and create a longer proof of work chain
- The block containing Normal‘s transaction is now invalidated and can never be published



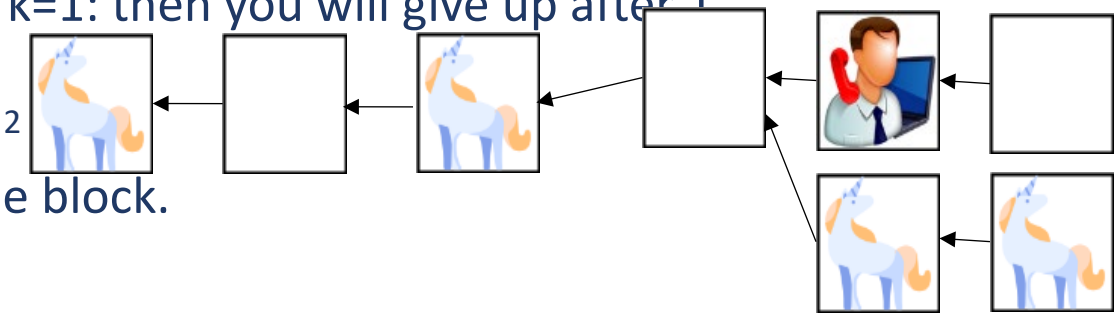
- Non-Wonderland miners might stop trying to include Max Normal‘s transactions when mining blocks, because they know that their block will be invalidated by Wonderland miners
- The strategy shown, how a 51% majority can prevent anyone from accessing the blocks, is called **punitive forking.**



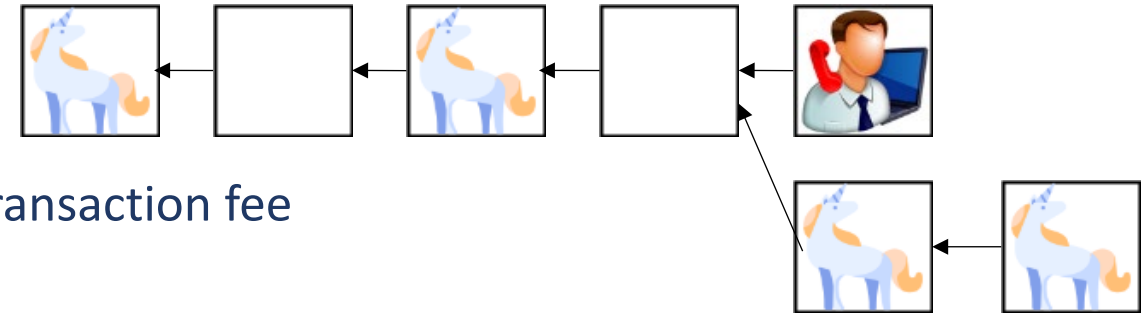
PROOF OF WORK (6/6)
BLACKLISTING BY FEATHER FORKING

Let q be the proportion of mining power you have 0<q<1 and k=1: then you will give up after 1 confirmation (one additional block).
Your chance to successfully invalidate Max Normal‘s block = q^2
If now q=0.2, then will q^2 = 4% as your chance of invalidate the block.

➡ A very low chance!



Because of your announcement other miners now know that their block has a q^2 chance of becoming invalildated. They now have to decide whether they should include Max Normal‘s transaction in their block!



Exp. Value(include) = $(2-q^2)$ * BlockReward + Max Normal‘s transaction fee
Exp. Value(don‘t include) = BlockReward

PROOF OF STAKE (1/5)

Proof of Stake is:

A consensus algorithm in which the member‘s voting (or mining) power is proportional to **stakes** in the network to select a node to **create** a block and **validate** the one created.

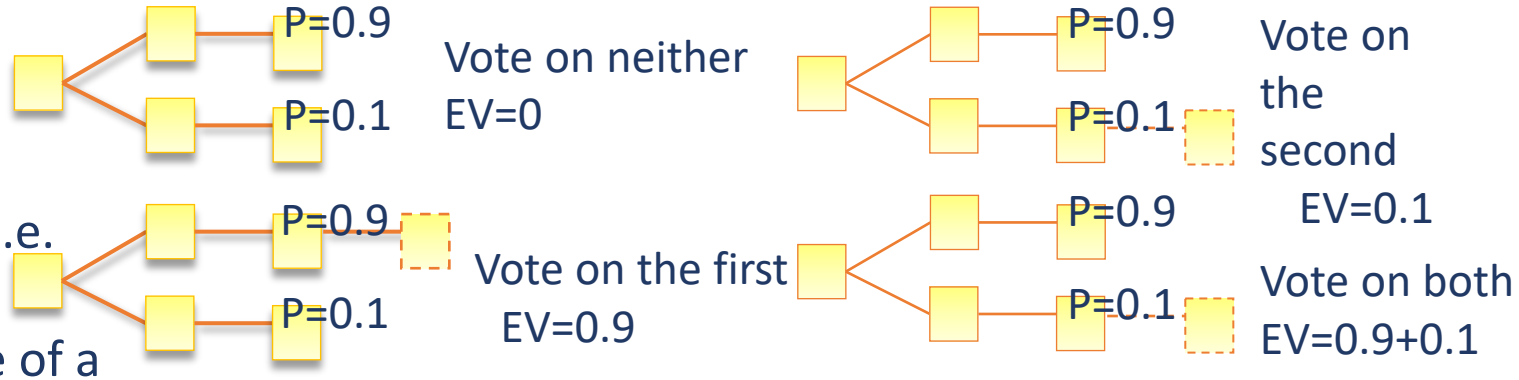
The stakes to select a node to make a block can be vary as follow:

- **the amount of tokens locked up**
- the number of days tokens held (Peercoin)
- random (nxt, Blockcoin)
- round robin (most of DPoS platforms)

PROOF OF STAKE (2/5)
NOTHING AT STAKE ATTACK

In Simple Minded PoS Algorithm 1, only rewards for creating blocks on multiple competing chains, no penalties.

1. If actors are all economically rational (i.e. pursuing self-interests only), the blockchain never converges in the case of a fork, thus never reaches consensus.
2. Economically rational actors choose both.



Defense - Simultaneous Slasher:

- At every round 64 validators gain signing priviledges based on a block 2000th blocks behind the current block
- The maximum length blockchain is weighted by the signatures of validators.
- If a malicious validator signs on two conflicting chains simultaneously, a future honest validator can include a proof of the double signing in a future block and slash the rewards of the malicious validator, taking 33% of the reward in return for honest behaviour.

Or Simple Minded PoS Algorithm 2

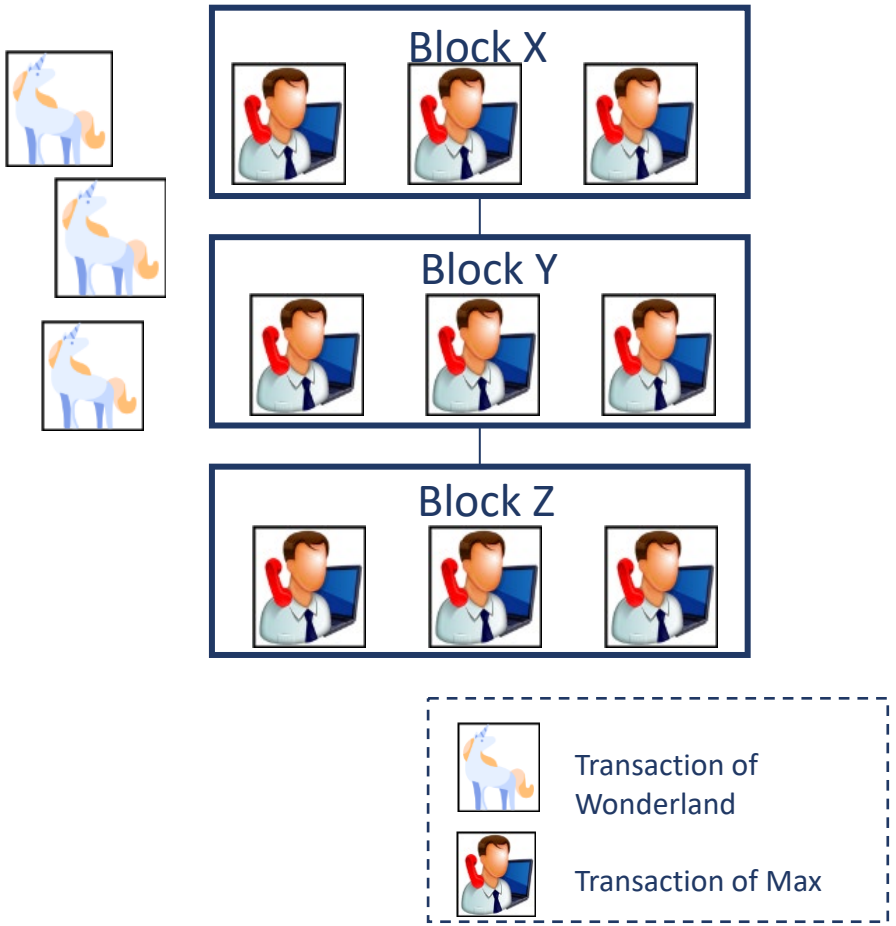
PROOF OF STAKE (3/5)
CENSORSHIP ATTACK

Censorchip

- ≥34% of voting power could effectively prevent from certain valid transactions to be included in the blockchain
- If the attacker has <67% power, the honest nodes can refuse to mine on blocks that they believe were censored, transforming this into a Liveness Denial attack.
- If the attacker has ≥67% voting power, it can work around above strategy.

Defense:

- Make censorship costly
- Timeclock Cryptography Algorithm



PROOF OF STAKE (4/5)
STAKE GRINDING ATTACK

Problem:

Next chosen validator depends on previous block‘s signature

- The current validator can repeatedly produce (a.k.a. „grind“) new signatures to improve his chance of being picked again.

Defense

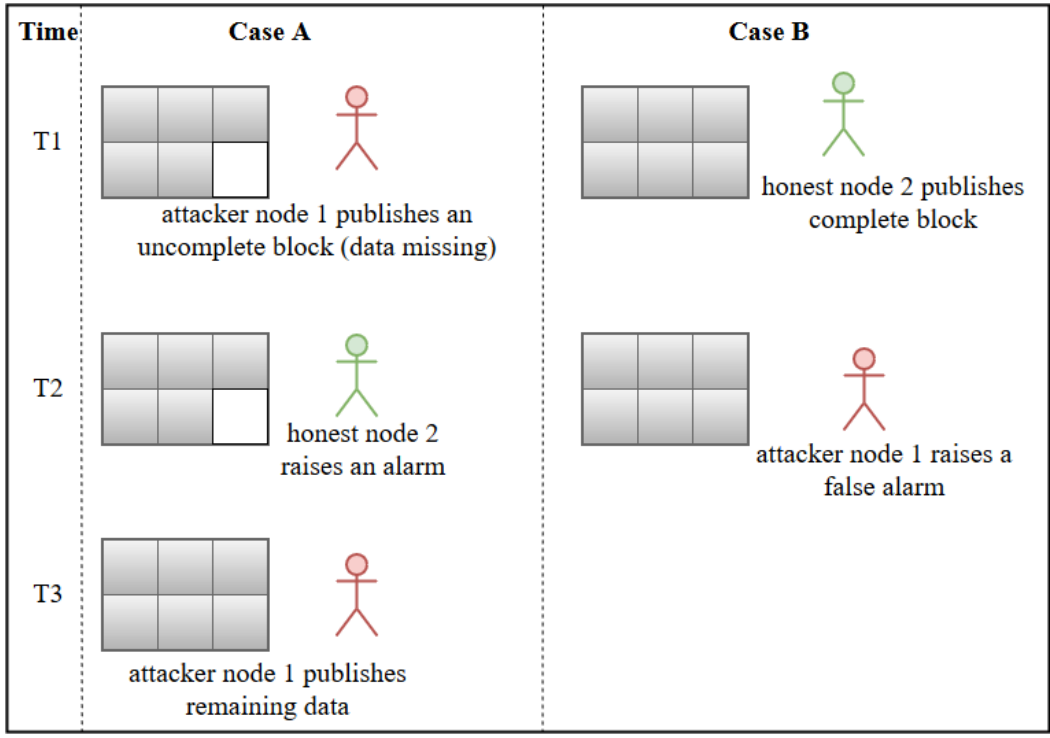
- Not to use information that can be easily manipulated as source data for the randomness
- Require all the validators deposit their stake well in advance
- Use some sort of secret sharing/threshold signature scheme, and have multiple validators collaboratively generate the random value.
  - Unless majority colludes, this is a safe scheme

PROOF OF STAKE (5/5)
DATA (UN)AVAILABILITY ATTACK

Attacker makes a block with invalid or malformed transactions, but does not include all the relevant data for someone constructing a fraud proof.

Defense

- Require blocks to commit to the Merkle root of this „extended“ data (use erasure codes)
- Probabilistically check that the majority of the extended data is available.
- We know that one of three things is true:
  1. The entire extended data is available, the erasure code is constructed correctly and the block is valid
  2. The entire extended data is available, the erasure code is constructed correctyl, but the block is invalid
  3. The entire extended data is available, but the erasure code is constructed incorrectly



2 cases of data (un)availability attacks

DELEGATED PROOF OF STAKE PROS AND CONS

Pros

- Fast
  - Transactions can be confirmed as low as one second
  - More than 1000 TPS, adopted as a solution to scalability
- Efficient
  - Network parameters such as fee schedules, block intervals, transaction sizes can be decided by elected delegates.
- Used by many recent cryptocurreny platforms
- Double spending attack can be difficult

Cons

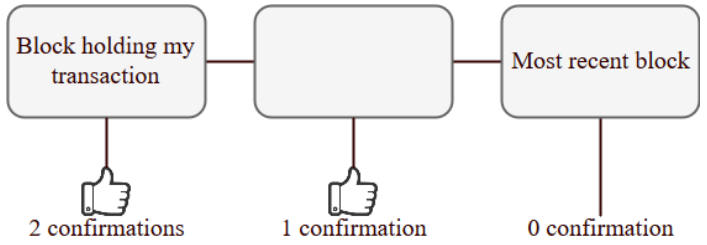
- Centralized
  - Vulnerable to network attacks such as DoS (Denial of Service)
- Delegates require to have high computing power and can not easily be replaced by ordinary stake holders
- Delegates can be colluded
  - Supposedly highly accountable

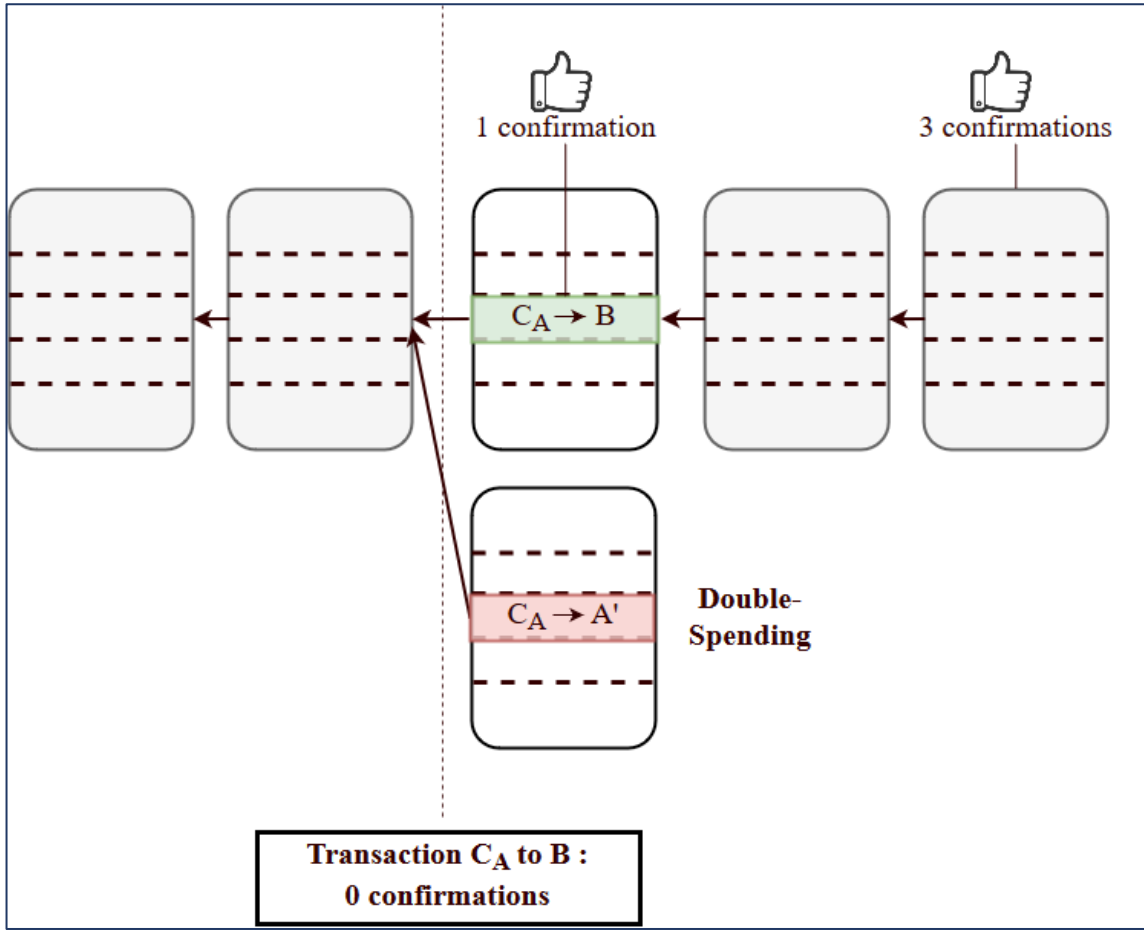
BITCOIN SECURITY (1/5)
DOUBLESPENDING ATTACK

Double-spending attack

- Alice creates two transactions: one to Bob, the other to herself
- Bitcoin‘s solution → make sure to wait for ***n*** confirmation.
  - **Confirmation:** The number of blocks created on top of the block a txn is in.

$$confirmations = block_{depth} - 1$$

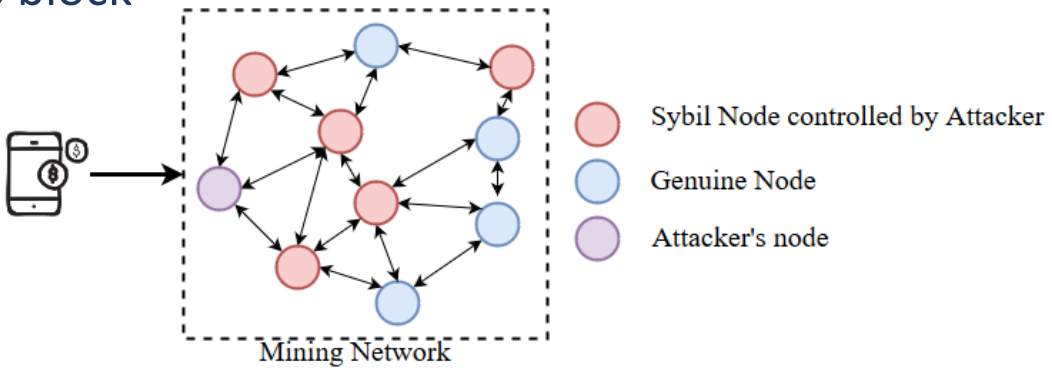




BITCOIN SECURITY (2/5)

- **Sybil Attack** – Lack of robust identity management
  - Attacker creates multiple identities (maybe virtual) and takes control of the network
  - to forward attackers block faster than the genuine users block

- **DoS/DDoS Attack** – Inherent from the Sybil Attack
  - A denies transactions from B‘s address



- **Majority Attack** – Bitcoin blockchain assumes an honest majority
  - 80 % of mining pools located in China, 20 % distributed over Iceland, Japan, Czech Republic, India

- **Indentity Theft** – Due to weak password for wallet
  - Stealing of private keys/wallet passwords through phishing attack

Attack:

An attacker launches a **sybil attack** against a target node and every other node.
The attacker aims to set the internal clock of the target node 70 minutes behind the wall clock and the internal clock of every other node 70 minutes ahead of the wall clock.

Now the attacker mines a block with a timestamp which is set 190 minutes ahead of the real time. Then every other node accepts the block, because it is in their 120 minute validation bound.
But the target node rejects it, because the block is past its 120 minute validation bound.

The network has now been effectively partitioned by the attacker, because the targed node thinks that every new block is invalid, while the rest of the network continues on.

As long as the attacker can keep timejacking the target, the attacker has indefinite amount of time to mine blocks for double spending.

BITCOIN SECURITY (4/5)
TIMEJACKING

- **Attacker gets indefinite amount of time to mine blocks for double spending attacks!**

  This still requires a non-trivial amount of hash power to maintain timejack.

- **The attacker gets several confirmations worth of time to mine no their double spending chain, even with a restricted time window!**

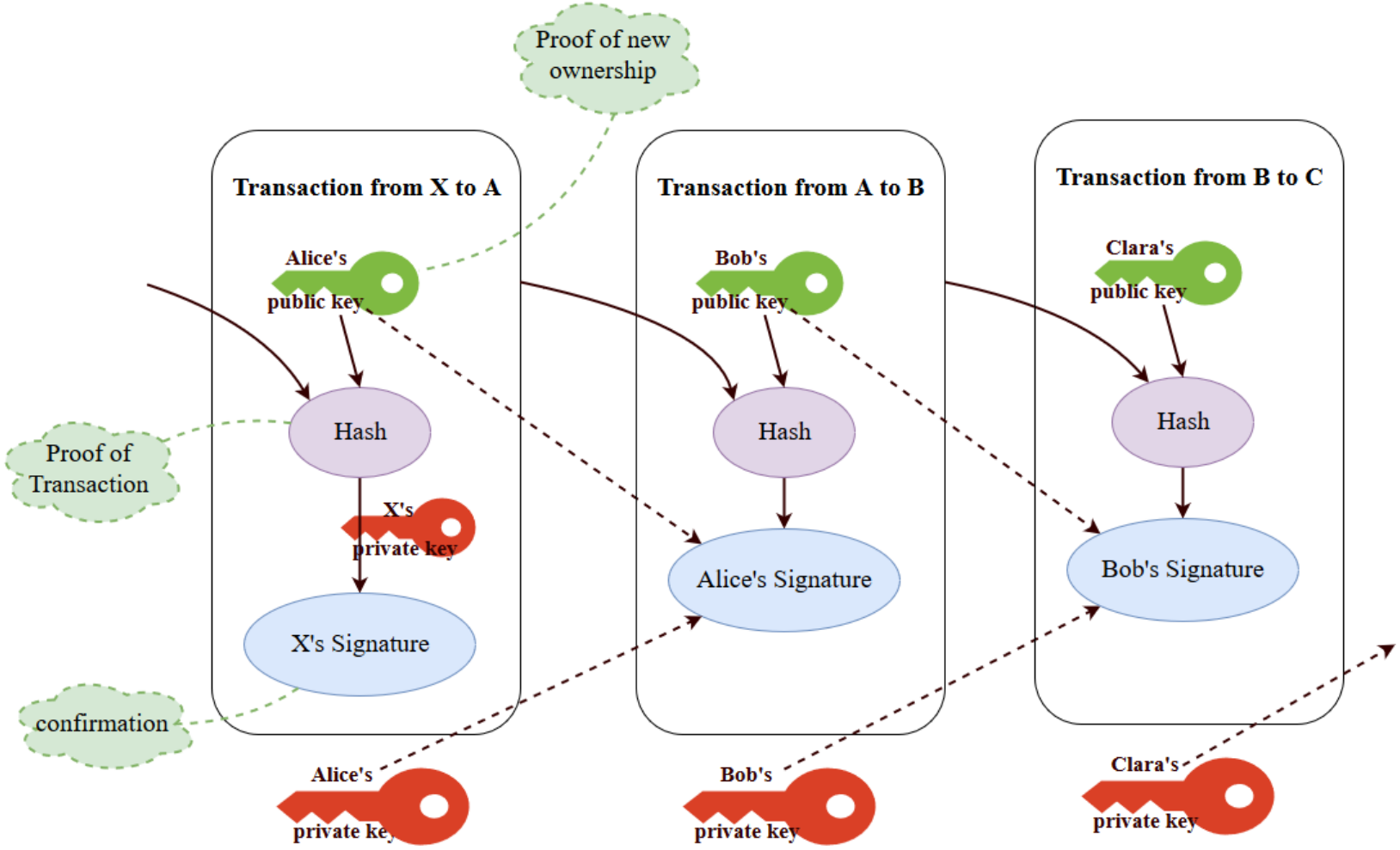
BITCOIN SECURITY (5/5)
TRANSACTIONS AND SIGNATURES

Digital Signature:

- always changing, but encrypted always with the same key

Proof of new ownership:

- with the public key address contained in a transaction everybody accepts these address as new owner



HYPERLEDGER FABRIC SECURITY (1/2)
REMOTE ACCESS TROJAN

- **Problem:** Installation of Malware – Remote Access Trojan (RAT)
  - Chaincode runs on Docker container
  - Chaincode has access to networking – can very easily download and install further software packages (including security tools) and can runf for long periods of time
  - Installation of RAT will act as a base from which a threat actor could undertacke a more comprehensive attack.
    - A threat actor could create a new ledger with associated malicious chaincode, and persuade others to participate
    - A threat actor could infiltrate an organization responsible for developing and maintaining the chaincode for an existing ledger, then publish an update

HYPERLEDGER FABRIC SECURITY (2/2)
LOG INJECTION & CODE INJECTION

- **Problem:** Log Injection
  - Unvalidated inputs are written verbatim to a log
  - Indirect threat to the business model
  - Can be used to fabricate log entries to mislead incident response efforts, or corrupt the log to prevent it from being processed by automated monitoring systems.

- **Problem:** Code Injection
  - One function was found to be vulnerable to Code injection
  - Subcomponents of the system (functions and data structures) do not have detailed interface specifications which can allow one to determine:
    - whether a function body correctly implements the required behaviour and
    - whether calls to that function elsewhere in the program are using it correctly and appropriately.

- **Problems:**
  - According to the European Economic Area (EEA) Repord, CORDA was vulnerable to:
    (dated 15th April, 2016)

    - Cross site scripting (XSS)
    - Sensitive Data Exposure – not using TLS
    - Missing Function Level Access Control – server side code validation was not implemented
    - Cross Site Request Forgery (CSRF)

RIPPLE SECURITY (1/2)
FAULTY PAYMENT GATEWAY

- **Gateway:** A gateway is any person or organization that enables users to put money into and take money out of Ripple‘s liquidity pool
- Gateway wallets:
  - Included in the core of the Ripple network
  - Significantly contribute to the liquidity of the network
- A faulty gateway can disable rippling on most credit links of its wallet, ensuring that transactions routed through it are no longer possible and effectively freezing the balance held at credit of its wallet.
- This affects
  - Liquidity of the network
  - Lead to monetary losses to the neighboring wallets

RIPPLE SECURITY (2/2)

- **Problem:**
  - Ripple Labs owns 60% of all XRP in circulation (60 billion out of total 100 billion)

- This did not follow the goal of making XRP a decentralized peer-to-peer currency

- To answer this and to create Supply Predictability, Ripple placed 55 billion XRP in a cryptographically secure wallet
  - 55 escrow contracts of 1 billion each were created
  - In the beginning of a month a contract expires and 1 billion XRP is made available for Ripple‘s use.
  - Unused XRP at the end of the month are returned back to the escrow.

IOTA SECURITY (1/3)
COLLISIONS

- **Problem:** Hash function ‚Curl‘ has produced collisions
  - Collision in hash function was found using differential cryptoanalysis
  - Here, two different payments in IOTA (bundles) with same hash value are produced
  - Thus have the same signature.
  - In such attack, a bad actor can destroy users‘ funds, or possibly, get user funds.
- Fixed on Patch ussued on August 7, 2017

IOTA SECURITY (2/3)
REPLAY ATTACK

- **Problem:** Replay Attack

  - IOTA utilizes one time signatures, combined with low confirmation rates of transactions – „replayBundle“ feature.
  - Reattaching is required to get a transaction through
  - Bundles can only be safely signed a single time.
  - Thus, a user is allowed to reattach any bundle of transactions without any proof of ownership.
  - The expected behaviour – only one use of the same bundle hash should be allowed inside a consistend transaction history (subtangle).

- **Problem** – The replays of a previously confirmed bundle will not get confirmed again. The coordinator will repeatedly approve the same bundle hash
  - This means that while a user has signed a transactions to send 500 Miota it can be attached to the network 10 times draining the account of 5000 Miota.

IOTA SECURITY (3/3)
PHISHING ATTACK

- **Problem:** Phishing Attack

  - In August 2017, the hacker registered the domain iotaseed.io and advertised it as an IOTA seed online generator.
  - He linked the iotaseed.io website to a GitHub repository
  - He ran mostly the same code from the GitHub repository but made modifications to the Nitifier.js library
  - This code always used a fixed seed „4782588875512803642“ plus a counter variable that increases by one every time seedrandom is run
  - IOTA users visiting the iotaseed.io website received predictable seeds
  - On January 19, the hacker utilized the logs to access IOTA accounts with the seeds (private keys) he collected and started transferring funds out of owner‘s wallets which amounted to $4 million