

BITCOIN

.

Bitcoin

Satoshi Nakamoto
satoshin@gmx.com
www.bitcoin.org

- Launched in 2009
- A *peer-to-peer* electronic cash system
- Why it is interesting to study Bitcoin?
 - ▣ It is a P2P network & protocol
 - ▣ It is a distributed comp. innovation
 - ▣ *New form of currency* that may take off or even replace existing currencies
 - Papers in economics and computer science
 - Many online store accept bitcoin it as a form of payment

Abstract. A purely peer-to-peer version of electronic cash would allow online payments to be sent directly from one party to another without going through a financial institution. Digital signatures provide part of the solution, but the main benefits are lost if a trusted third party is still required to prevent double-spending. We propose a solution to the double-spending problem using a peer-to-peer network. The network timestamps transactions by hashing them into an ongoing chain of hash-based proof-of-work, forming a record that cannot be changed without redoing the proof-of-work. The longest chain not only serves as proof of the sequence of events witnessed, but proof that it came from the largest pool of CPU power. As long as a majority of CPU power is controlled by nodes that are not cooperating to attack the network, they'll generate the longest chain and outpace attackers. The network itself requires minimal structure. Messages are broadcast on a best effort basis, and nodes can leave and rejoin the network at will, accepting the longest proof-of-work chain as proof of what happened while they were gone.

Overview

- Intro to Bitcoin
- Cryptographic hashes
- Public key crypto and digital signature
- Technical overview of Bitcoin
- The practice of mining Bitcoin
- “A glance of Smart Contracts”, next lesson

Many ATM machines exchange Bitcoins,
ex: in Tbilisi, Georgia !!



What is Bitcoin



- ❑ Bitcoin is a **Distributed Peer-to-Peer Network**
- ❑ Bitcoin is a **Peer-to-Peer Protocol**
- ❑ Bitcoin is a **Distributed Computing Innovation**
- ❑ Bitcoin as a **Currency** is only one of the many application of these concepts!
- ❑ Each participant maintains a **public decentralized ledger**
- ❑ There is **no central point of control**
- ❑ Bitcoin is secured by **Cryptography**
- ❑ Bitcoin are **created** through a process called **Mining**

Basic question for anyone accepting digital money

1. Can I be sure that no one else can claim this money ?
2. Can I trust that the money is authentic and not counterfeit ?
3. Can I trust that the digital money can only be spent once ?

(known as the **double-spend** problem)

Timeline: 2008---

- Bitcoin was invented in 2008
- Publication of a paper entitled *Bitcoin: A Peer-to-Peer Electronic Cash System* by Satoshi Nakamoto
- Key innovation: distributed computation system **Proof-of-Work** allowing to arrive at consensus about the state of transactions
- Bitcoin network started in 2009

Bitcoin: A Peer-to-Peer Electronic Cash System

Satoshi Nakamoto
satoshin@gmx.com
www.bitcoin.org

Abstract. A purely peer-to-peer version of electronic cash would allow online payments to be sent directly from one party to another without going through a financial institution. Digital signatures provide part of the solution, but the main benefits are lost if a trusted third party is still required to prevent double-spending. We propose a solution to the double-spending problem using a peer-to-peer network. The network timestamps transactions by hashing them into an ongoing chain of hash-based proof-of-work, forming a record that cannot be changed without redoing the proof-of-work. The longest chain not only serves as proof of the sequence of events witnessed, but proof that it came from the largest pool of CPU power. As long as a majority of CPU power is controlled by nodes that are not cooperating to attack the network, they'll generate the longest chain and outpace attackers. The network itself requires minimal structure. Messages are broadcast on a best effort basis, and nodes can leave and rejoin the network at will, accepting the longest proof-of-work chain as proof of what happened while they were gone.

Bitcoin as a solution to Byzantine Generals Problem

m

□ Trying to find a
information
compromise

□ *Proof of* \
problem with

The Byzantine Generals Problem

LESLIE LAMPORT, ROBERT SHOSTAK, and MARSHALL PEASE
SRI International

Reliable computer systems must handle malfunctioning components that give conflicting information to different parts of the system. This situation can be expressed abstractly in terms of a group of generals of the Byzantine army camped with their troops around an enemy city. Communicating only by messenger, the generals must agree upon a common battle plan. However, one or more of them may be traitors who will try to confuse the others. The problem is to find an algorithm to ensure that the loyal generals will reach agreement. It is shown that, using only oral messages, this problem is solvable if and only if more than two-thirds of the generals are loyal; so a single traitor can confound two loyal generals. With unforgeable written messages, the problem is solvable for any number of generals and possible traitors. Applications of the solutions to reliable computer systems are then discussed.

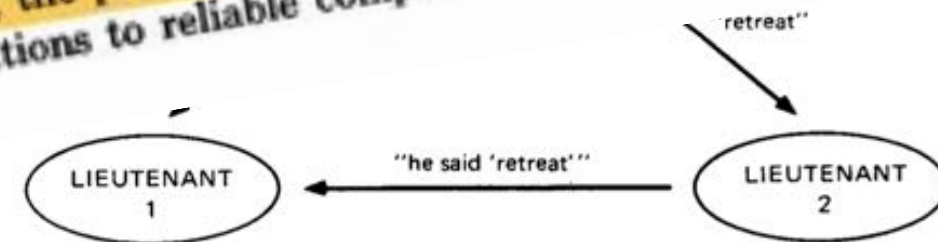
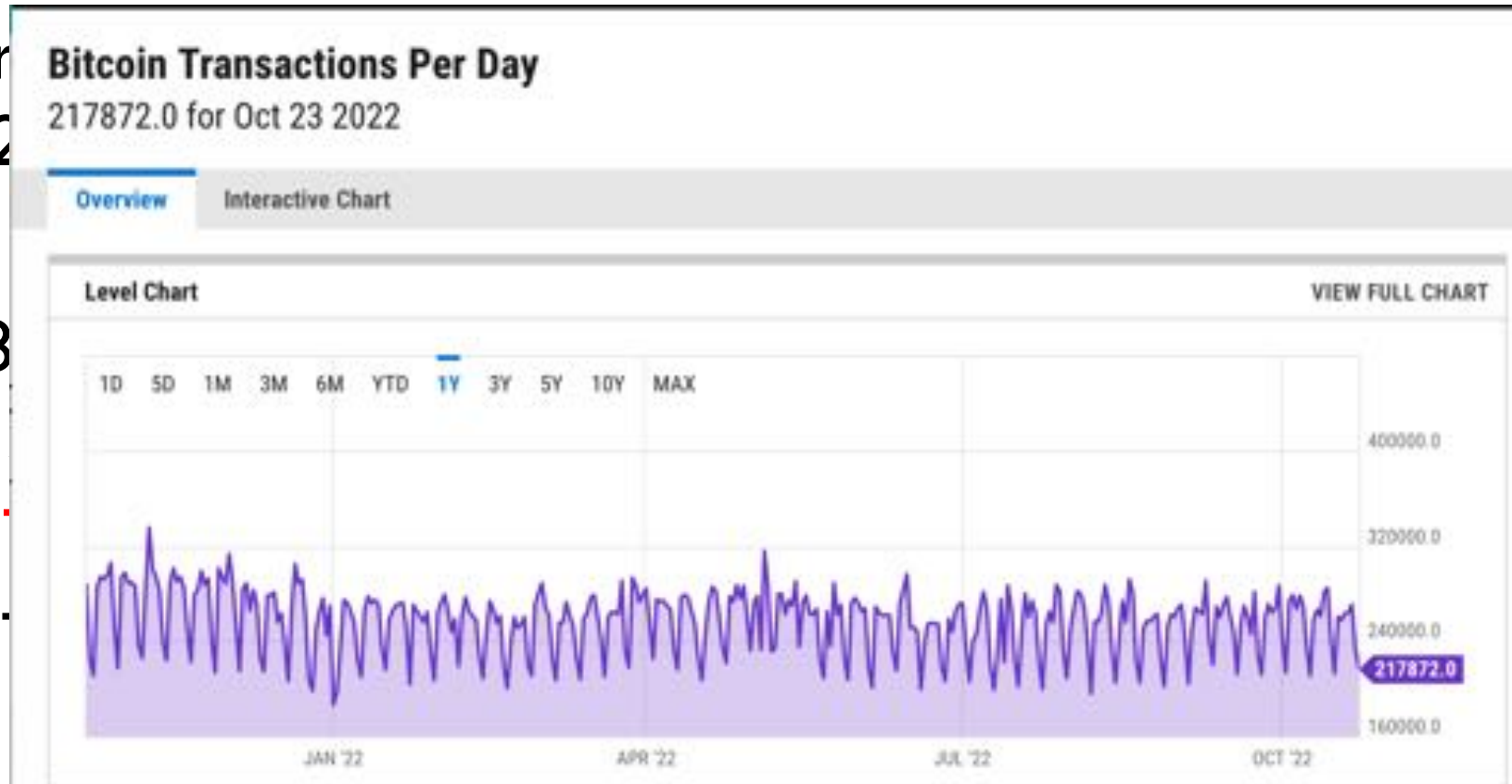


Fig. 2. The commander a traitor.

Size of the Bitcoin Economy

- Number of Bitcoins in circulation (avg) in Oct 2022
- Total number of Bitcoins
- 1 BTC ~ 6400\$ (Oct'18) ~ 193000\$ (Oct'21) ~ 193000\$ (Oct'22)
- Price is very unstable (BTC)
- Total balances held in Bitcoin wallets
- circulating in USD
- 200,000 – 300,000 (avg) transactions per day 4Q 2022



Bitcoin: Challenges

- All virtual currency must address the following challenges:
 - ▣ Creation of a virtual coin
 - How is it created in the first place? (how to bootstrap a new money)
 - How do you prevent inflation? (What prevents anyone from creating lots of coins?)
 - ▣ Validation
 - Is the coin legit?
 - How do you prevent a coin from **double-spending**?
- Bitcoin takes a infrastructure-less (i.e. P2P) approach
 - ▣ Rely on the concept of **proof-of-work** instead of a **dogmatic trust**
 - ▣ No central bank or clearing house

Overview



- Intro to Bitcoin
- Security Overview
 - ▣ Cryptographic hashes
 - ▣ Hash pointers and data structure
 - ▣ Digital signature
- Technical overview of Bitcoins
- The practice of mining Bitcoins (system's perspectives)

Four components in “secure communication”

- **Authentication** (Who am I talking to?)
 - ▣ Identification and assurance of the origin of information
- **Integrity** (Has my data been modified?)
 - ▣ Prevent improper and unauthorized changes
- **Availability** (Can I use my resources?)
 - ▣ The ability to use the information or resource desired
- **Confidentiality** (Is my data hidden?)
 - ▣ Concealment of information

From the Bitcoin's perspective

□ Authentication

- ▣ Am I paying the right person ? Not some other **impersonator** ?

□ Integrity

- ▣ Is the coin **double-spent** ?
- ▣ Can an attacker reverse or change Bitcoin transactions ?

□ Availability

- ▣ Can I make a transaction **anytime** I want ?

□ Confidentiality

- ▣ Are my transactions **private** ?

From the Bitcoin's perspective

- **Authentication** → Pub/Priv Key Crypto and Digital Signatures
 - ▣ Am I paying the right person ?
- **Integrity** → Digital Signatures and Cryptographic Hash
 - ▣ Is the coin **double-spent** ?
 - ▣ Can an attacker reverse or change transactions ?
- **Availability** → Broadcast messages to a P2P network
 - ▣ Can I make a transaction **anytime** I want ?
- **Confidentiality** → Pseudonymity (do not show your REAL identity)
 - ▣ Are my transactions **private** ?

Bitcoins: validation and creation of a Bitcoin

□ Validation

- Is the coin legit ?

 - Use of Cryptographic Hashes

- How do you prevent a coin from double-spending ?

 - Broadcast to all nodes

□ Creation

- How is it created in the first place ?

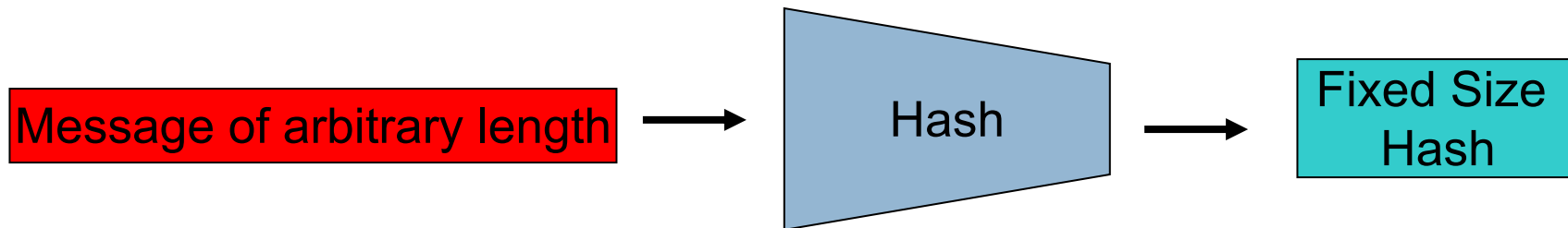
 - Provide incentives for miners

- How do you prevent inflation ? (What prevents anyone from creating lots of coins ?)

 - Limit the creation rate of the Bitcoins

Cryptographic Hash Functions (sha256, md5 ...)

- Consistent: $\text{hash}(x) = \text{hash}(y) \Rightarrow$ it is safe to assume $x = y$
always yields same result
- One-way/Hiding: given $\text{hash}(x)$, **hard to find** x
- Collision-free: given $\text{hash}(x)$, **hard to find** y different from x
such that $\text{hash}(x) = \text{hash}(y)$



- Hard to find / Puzzle-Friendly = use of **force brute** technique
given $\text{hash}(x)$ no solving strategy is better
than *force brute* to find x

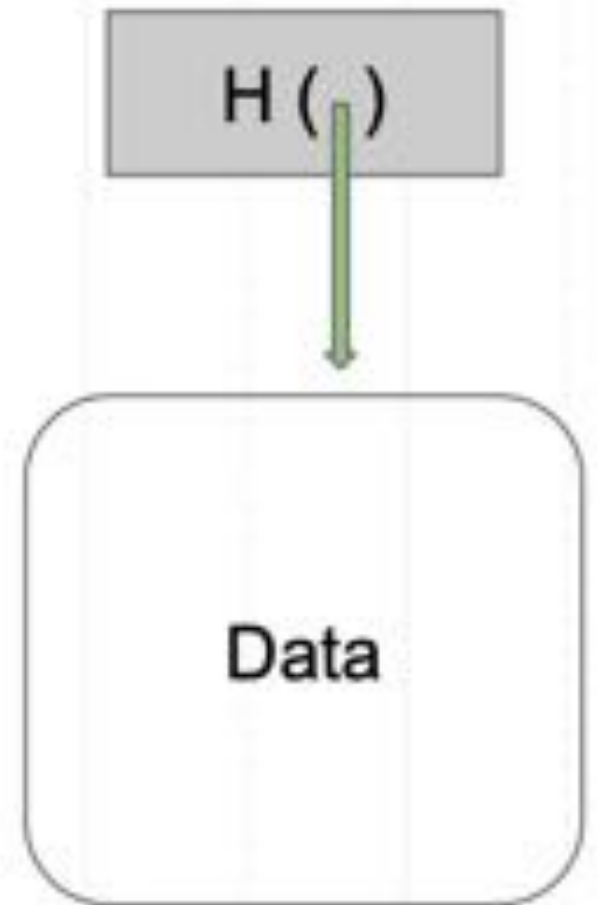
Hash functions as pointers 1/2

Hash Pointers

- Pointer to where information is stored
- Cryptographic hash of the information

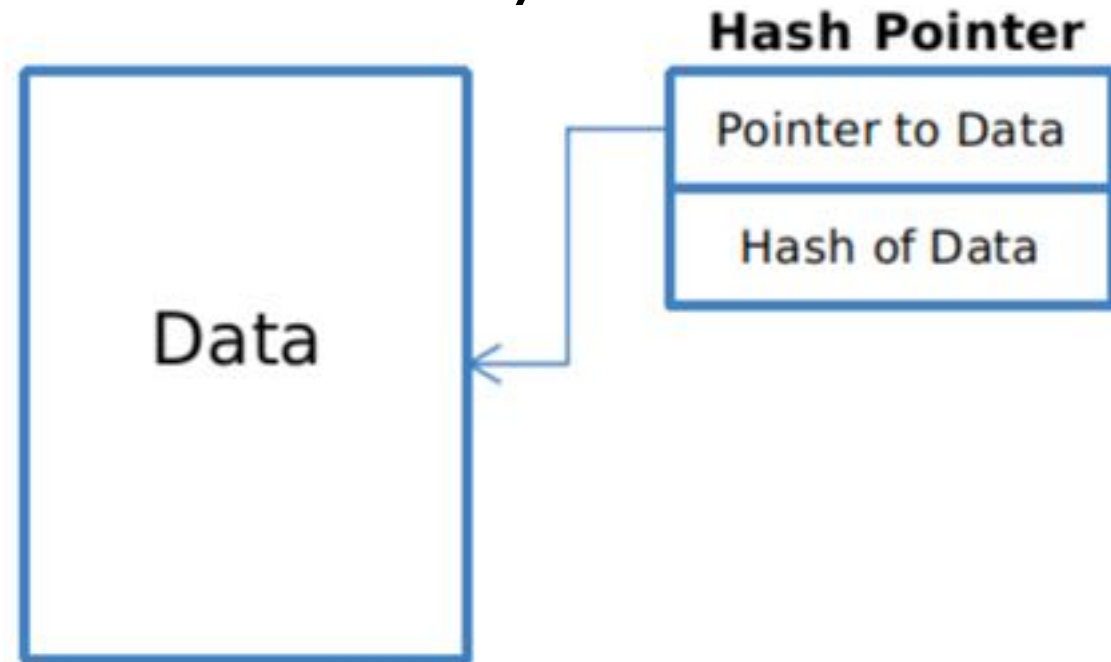
With Hash Pointers we can:

- Ask to get information back
- Verify that it hasn't changed

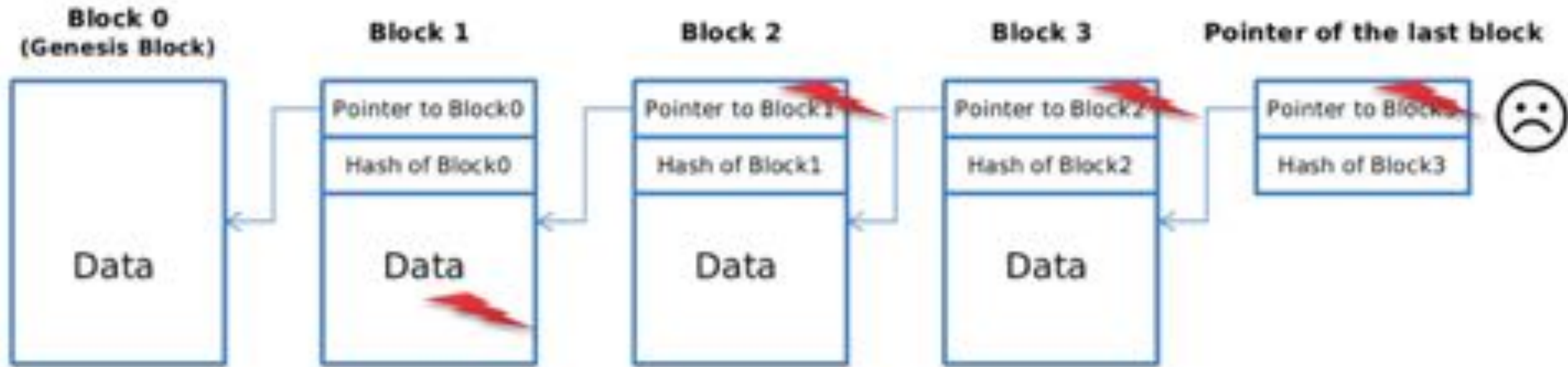


Hash functions as pointers 2/2

- A **hash pointer** is a data structure
- It is basically a pointer to the place where information is stored
- ... **plus** a cryptographic hash of the stored information (**digital signature**)
- A hash pointer will let us ask to retrieve the data and verify that the data hasn't changed
- So a hash pointer tells us **where** something and **what** it's value was
- In fact it also stores the hash of the value that this data had when we last saw it

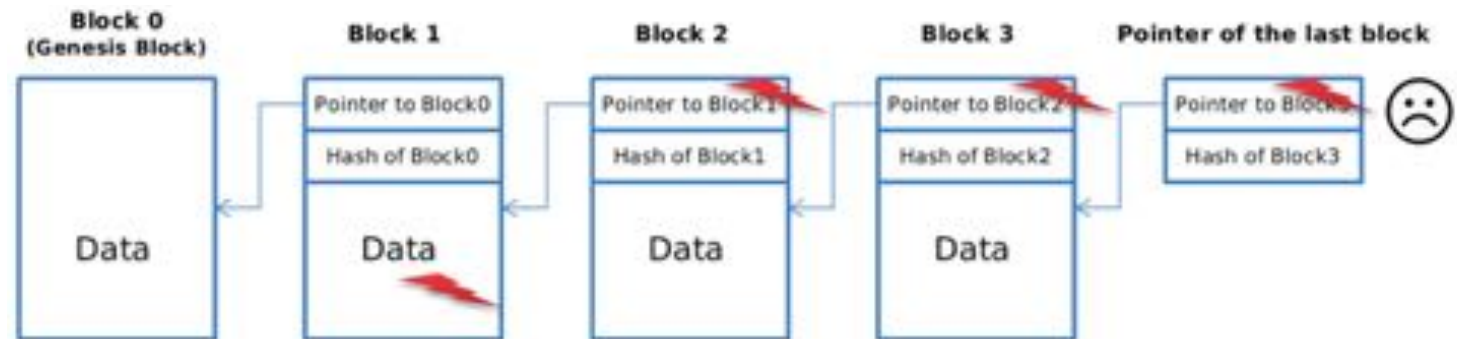


Tamper (alteration) evident property of a Blockchain

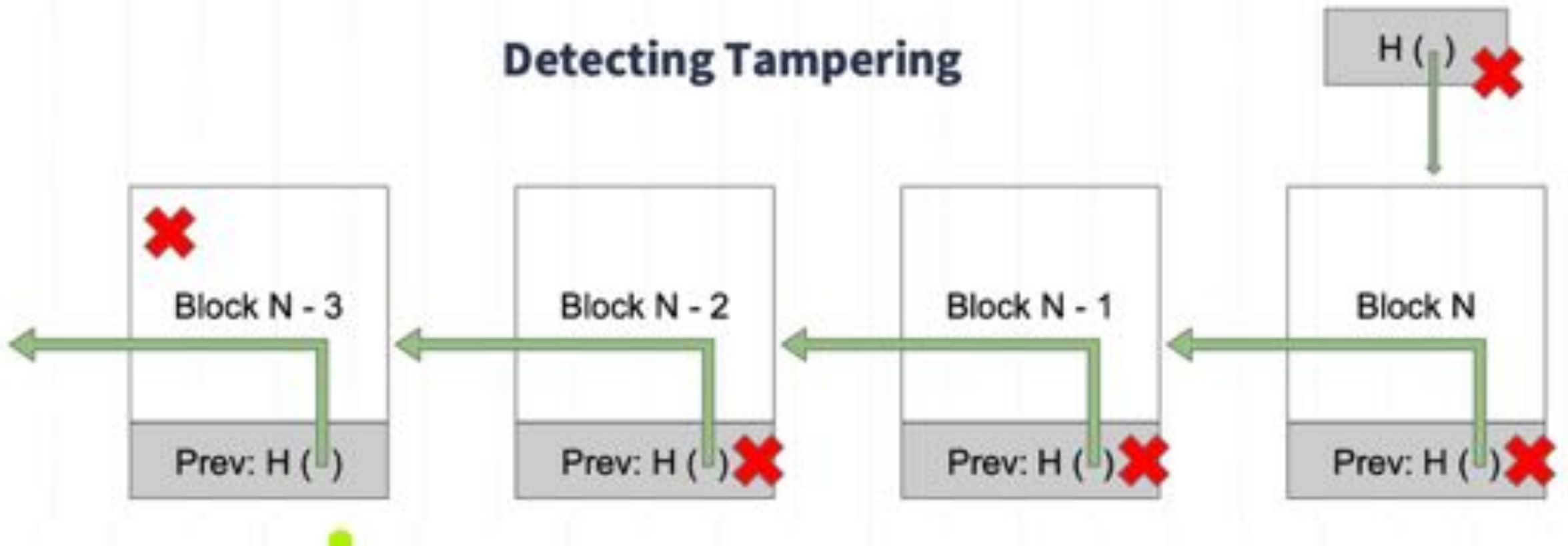


Tamper evident property of a Blockchain

- ❑ An attacker wants to tamper with one block of the chain, let's say, block 1
- ❑ The attacker changed the content of block 1! Because of **collision free property of the hash function**, he is not able to find another data which has the same hash with the old one. So now the hash of this modified block is also changed
- ❑ To avoid others noticing the inconsistency, he also needs to change the hash pointer of that block in the next block, which is block 2
- ❑ Now the content of block 2 is changed, so to make this story consistent, the hash pointer in block 3 must be changed
- ❑ Finally, the attacker goes to the hash pointer till the last block of the blockchain, which is a **nightmare for him**, because we keep and remember that hash pointer



Blockchain in a picture (again and again)

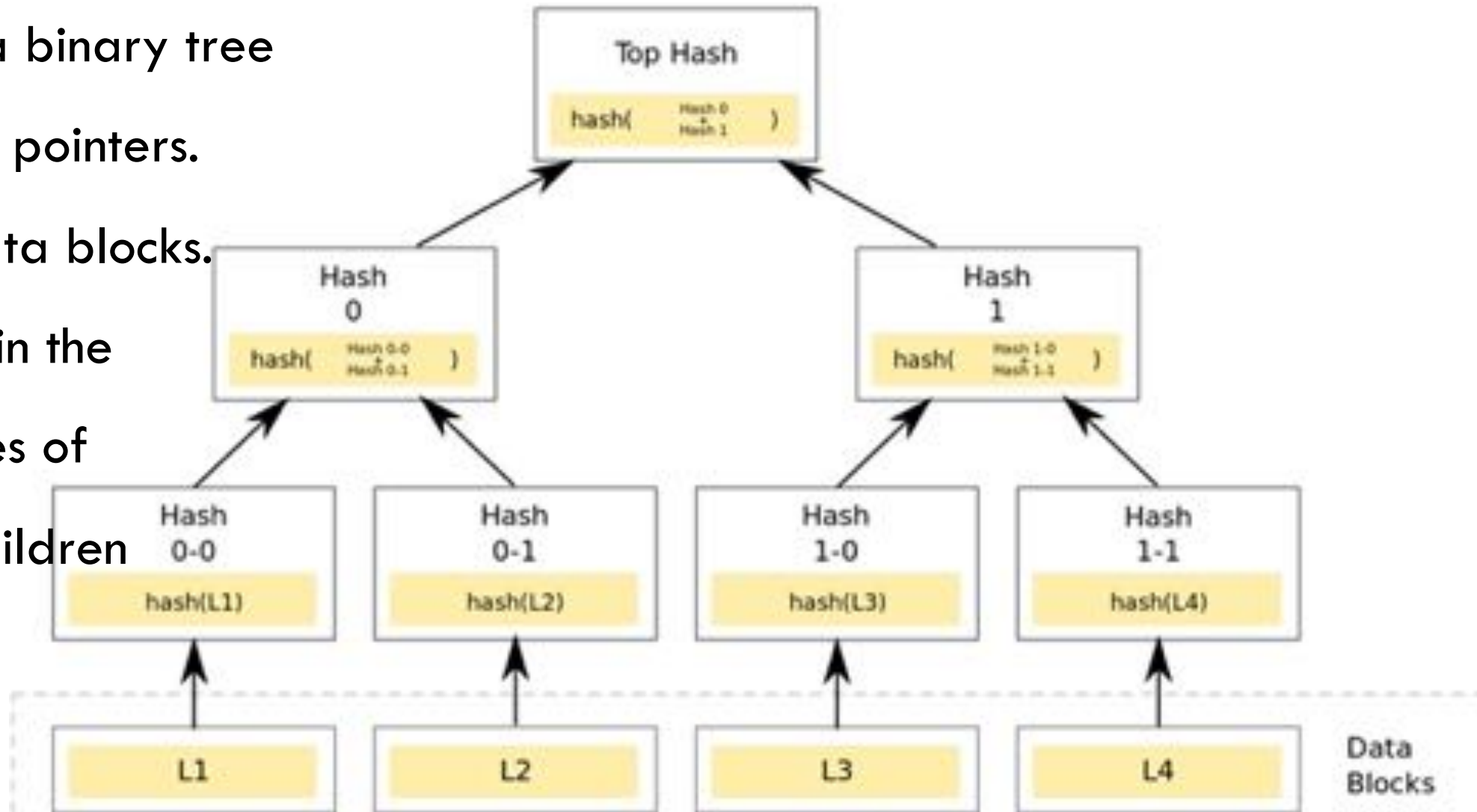


BC blocks : application of hash pointers : *Merkle Trees*

A Merkle Tree is a binary tree building with hash pointers.

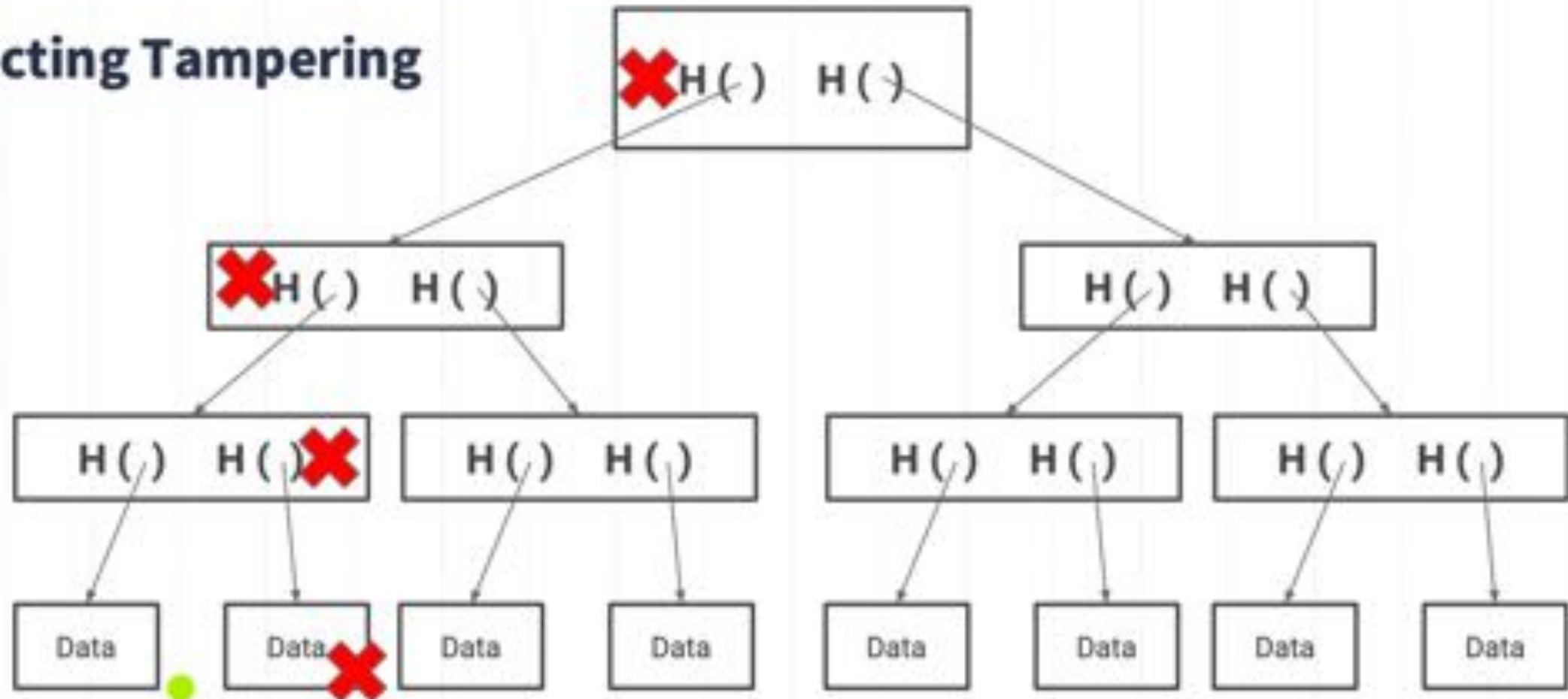
The leaves are data blocks.

Nodes further up in the tree are the hashes of their respective children



Markle tree and easy detecting tampering

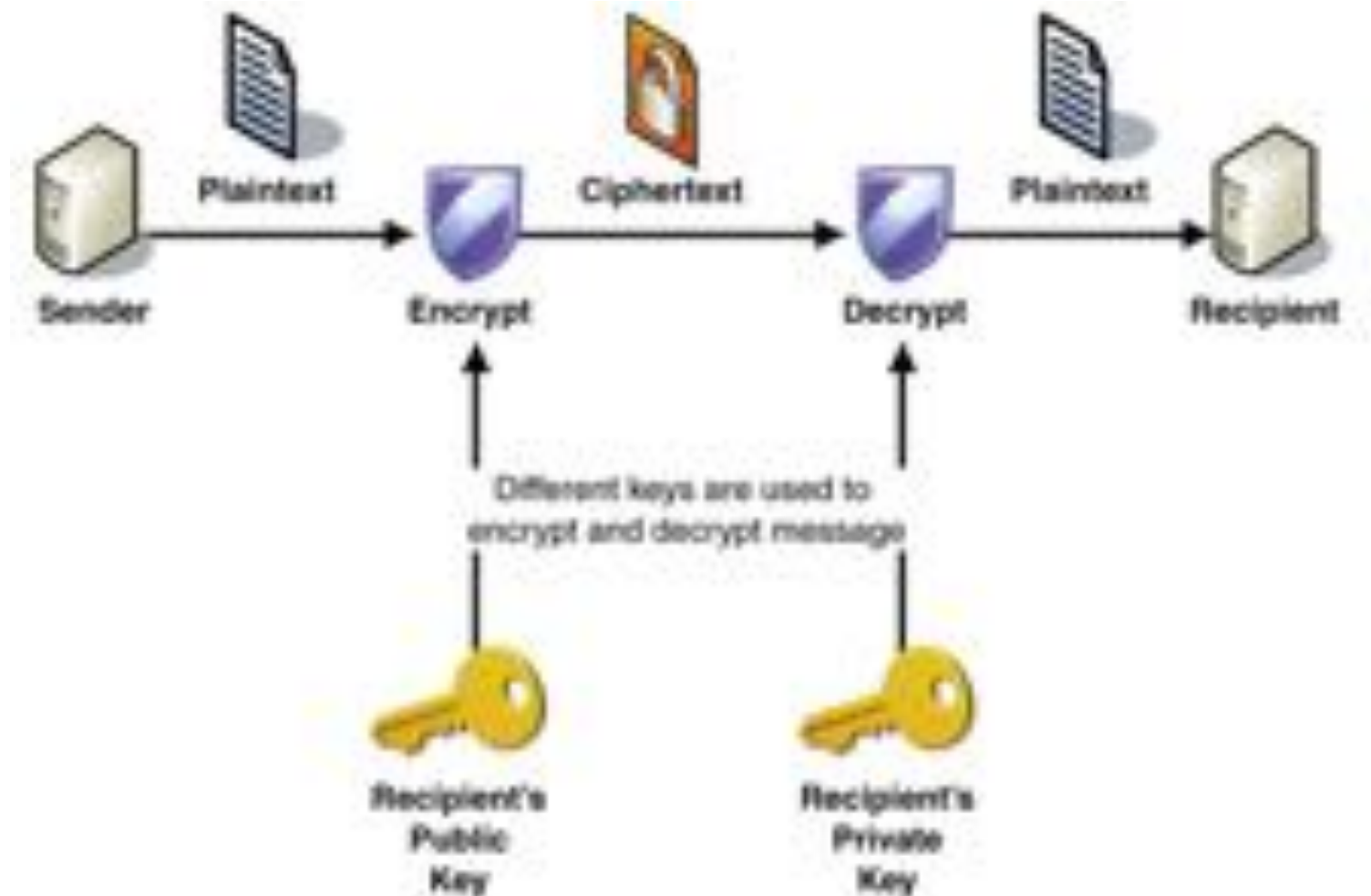
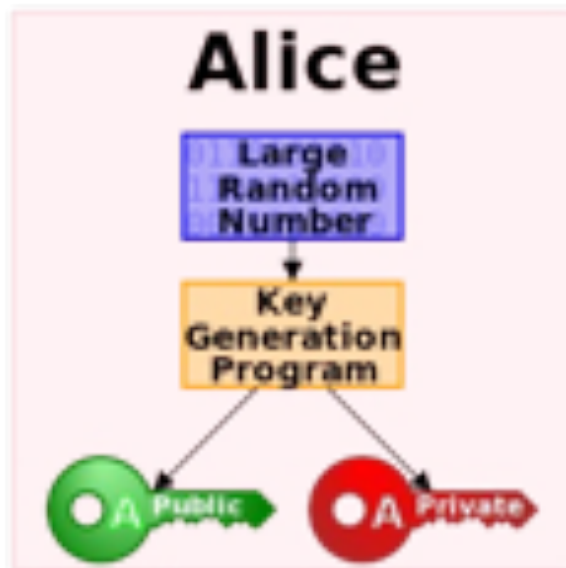
Detecting Tampering



ship is a
tion.

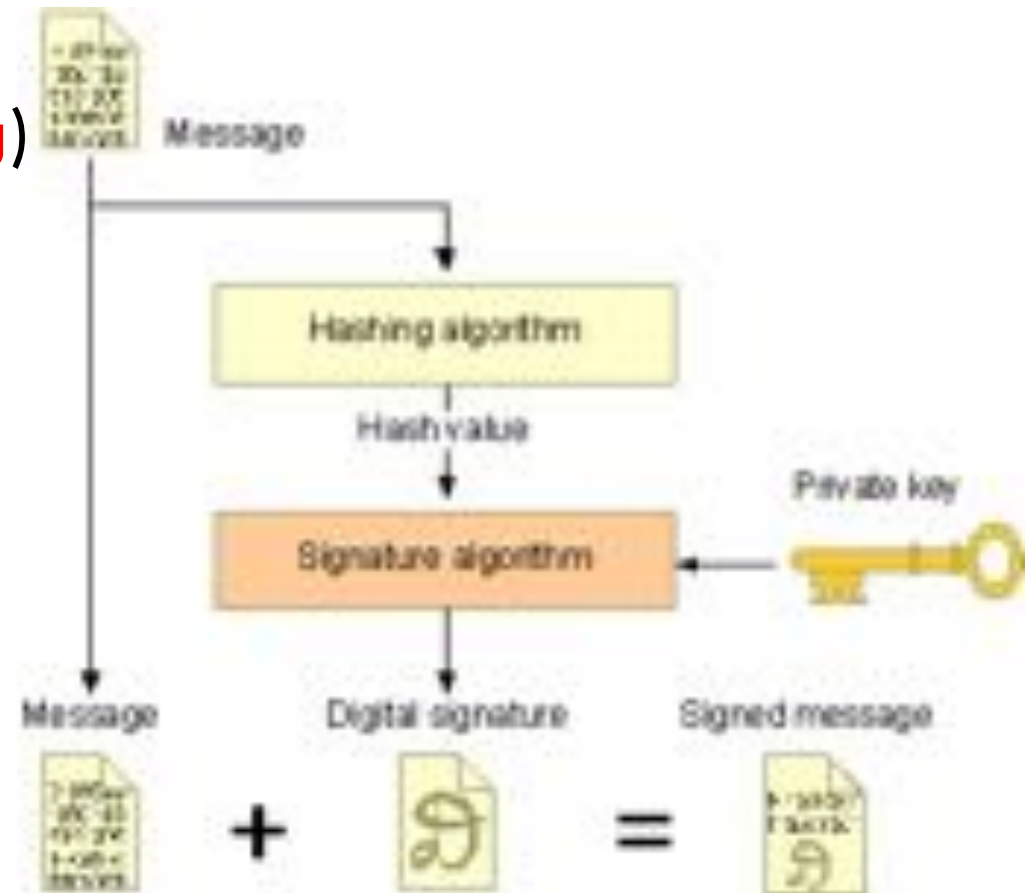
Encryption/Decryption via public/private keys

- Keys pair: public key and private key



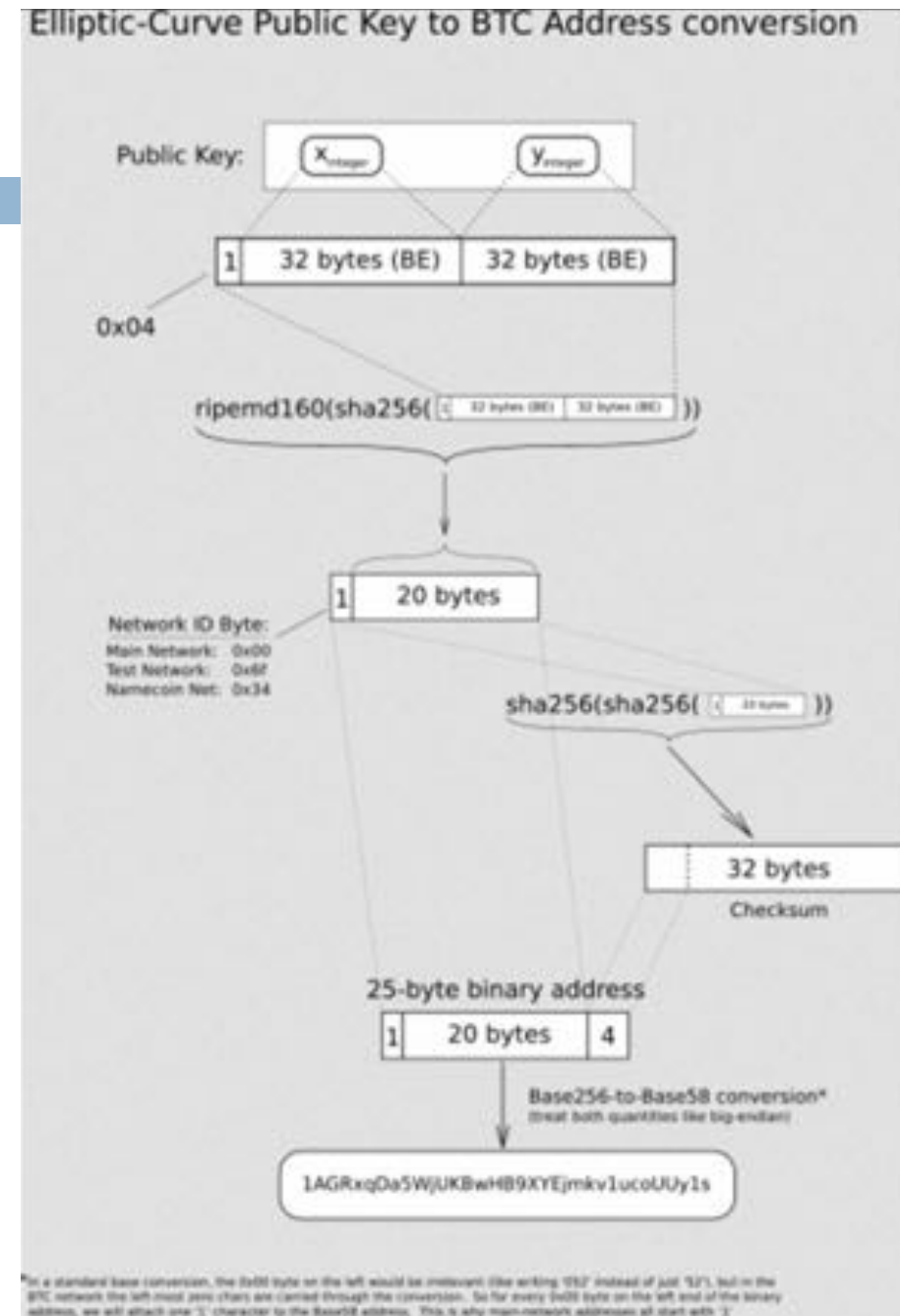
Digital Signature via public/private keys

- First, hash your Message using an **Hashing_Algorithm**
- Then, encrypt the hashed message with your **Private_key** obtaining the **Digital_signature** of your message
- Send message + Digital_signature (**Signed_msg**)
- By having the **Public_key** of the sender anyone can verify the integrity of the message
- First, decrypt the **Digital_signature** with the **Public_key** of the sender (digest)
- Then, compare obtained **Value** with **Hashing_Algorithm(Message)**
- If equals then integrity_ok else integrity_ko



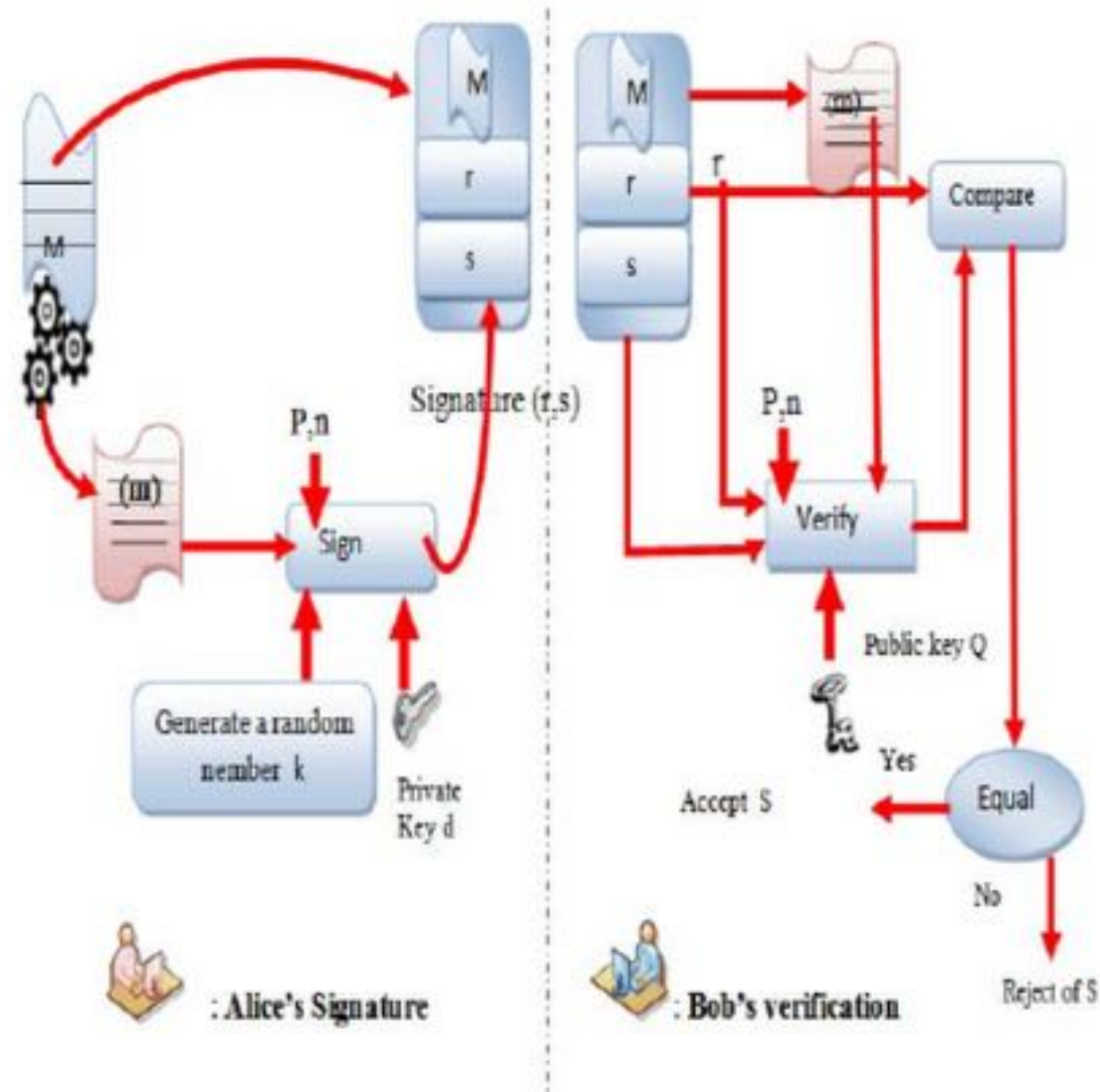
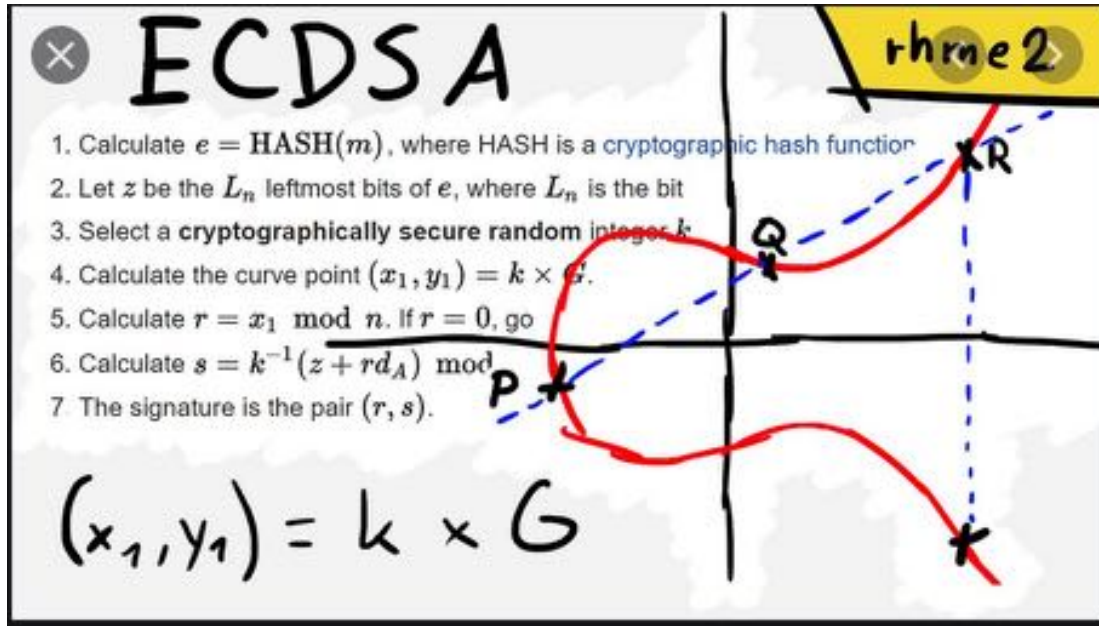
Bitcoin Address

- A **Bitcoin address**, or simply **address**, is an identifier of 26-35 alphanumeric characters, beginning with the number 1 or 3, that represents a possible destination for a bitcoin payment. Addresses can be generated at no computational cost by any user of Bitcoin using ECDSA algorithm
- 1H6R4cp58LYqFWH4t5GDJbxsH1ePEfRnmB
- 19zjb9P3yWS13U8wovKjuTUDFkDMeW68FN
- N.B. It is computationally unfeasible to compute the **public key** of an address, as the address is computed from the hash of the public key



Elliptic Curve Digital Signature Algorithm (ECDSA)

□ Trust me... you don't want to see all the details because there is no enough coffee in your blood ... but you are free to look at it if you care....



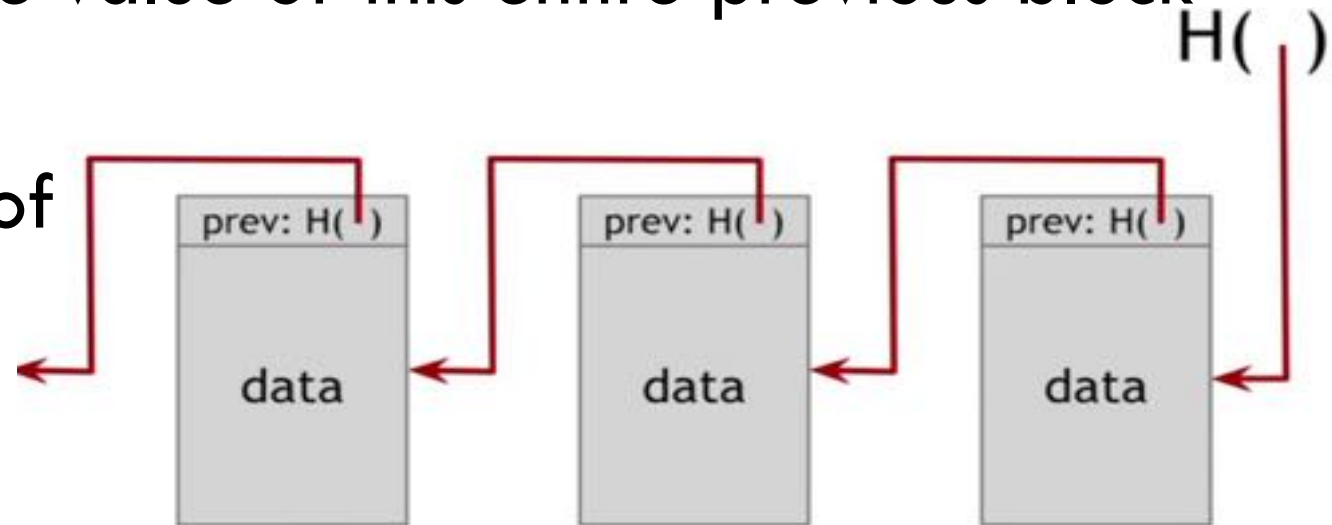
Overview



- Intro to Bitcoin
- Security overview
- Bitcoin: (a bit of) technical details (i hope not to be too much repetitive)
- The practice of mining Bitcoins (system's perspectives)

Blockchain implemented via Hash Pointers

- A Blockchain is a **linked_list** that we built with **hash pointers**
- It is just like a regular **linked_list** where you have a series of blocks containing **data** and a pointer to the previous block in the list
- Here the block pointer will be implemented with a **hash pointer**
- So it says where it is and what the value of this entire previous block was
- We are going to store the head of the list, just as a regular hash pointer

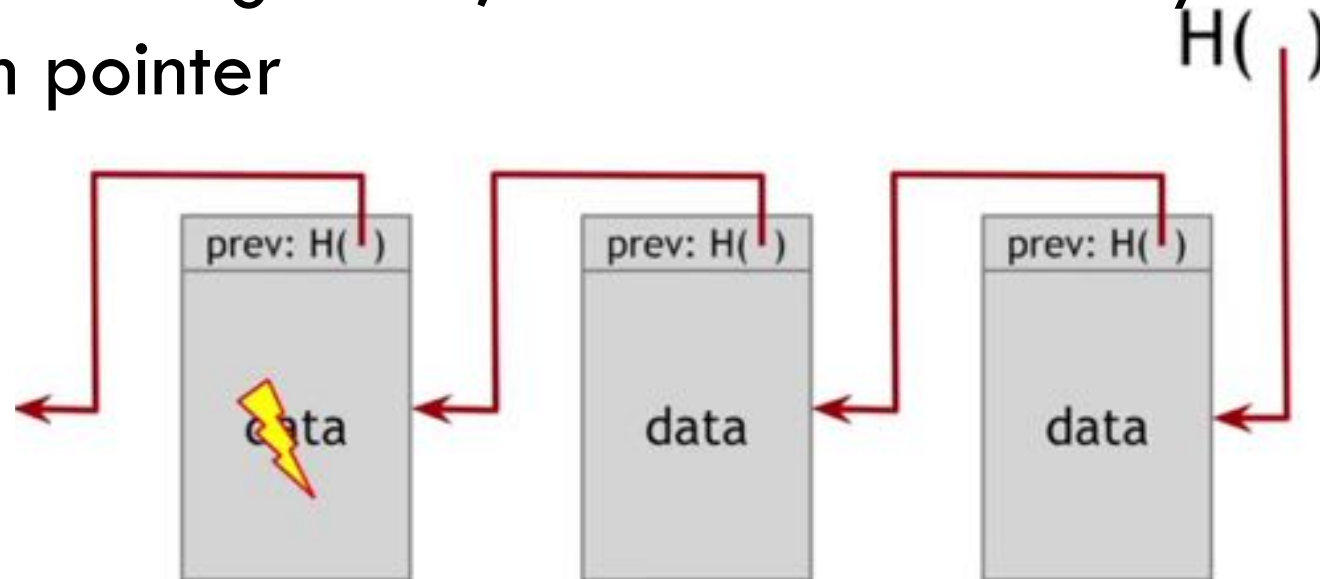


Blockchain detecting tampering

- Blockchain is a **tamper evident log**, used if we want to build a log data structure that stores a bunch of data that are not easy to modify
- **Let's try to add/modify data at the end of the log**, and detect if somebody tries to mess up with data already present in some block of the log
- Let's see what happens if an **adversary** wants to go back and tamper with data that's in the middle of the chain. And he wants to do it in such a way that we, the holders of the hash pointer at the head, won't be able to detect it

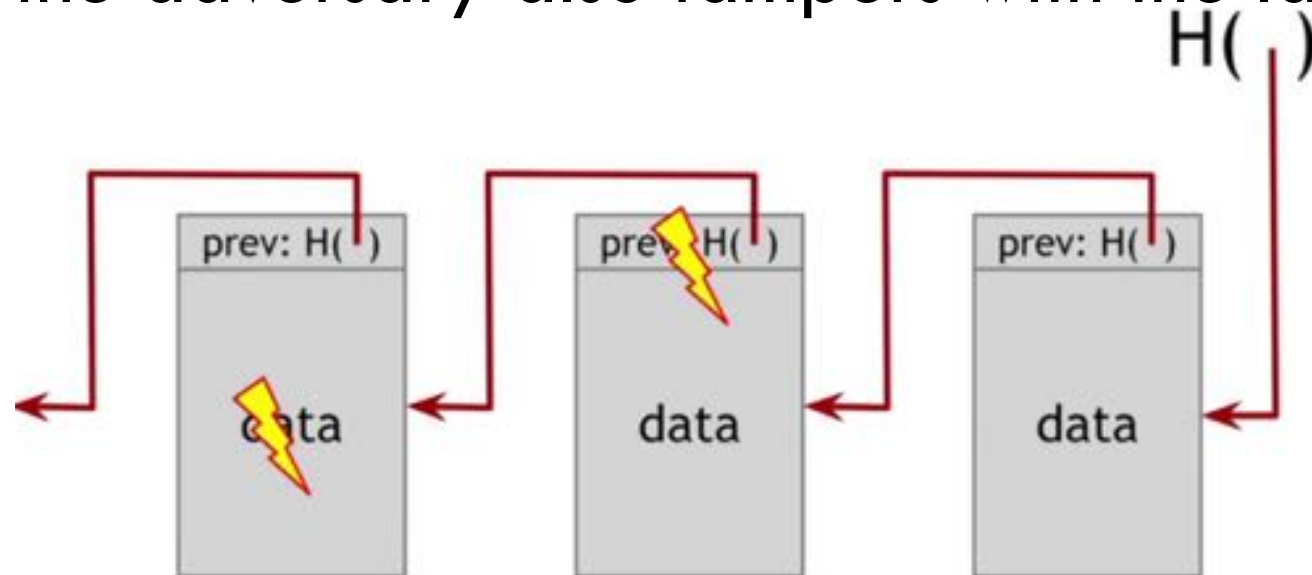
Tamper trial

- The adversary changes the contents of the block with the **light** symbol
- Therefore, the hash of this entire block changes, since the hash function is collision free
- So we could detect the inconsistency between this data and the hash pointer of the following block, unless the adversary also tampers it with a different hash pointer



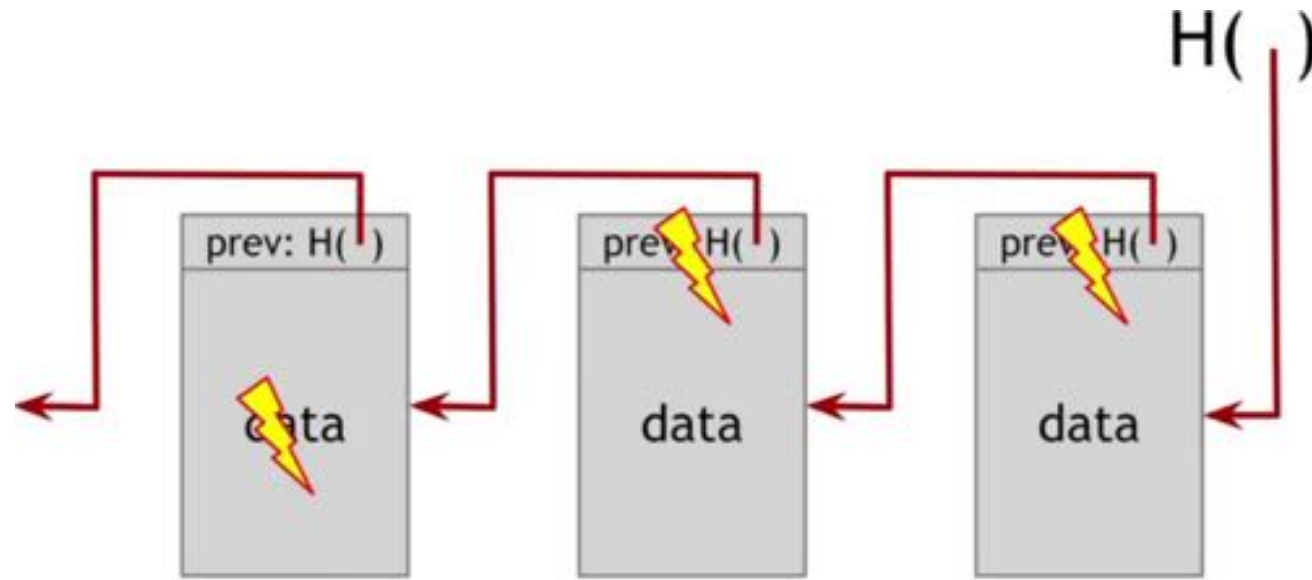
Tamper trial

- If the adversary tampers with the hash pointer then these two match up. But the content of the following block is changed: that means that its hash is not going to match the hash pointer of the following block
- So we detect the inconsistency between the contents of this block and the hash, unless the adversary also tampers with the last block

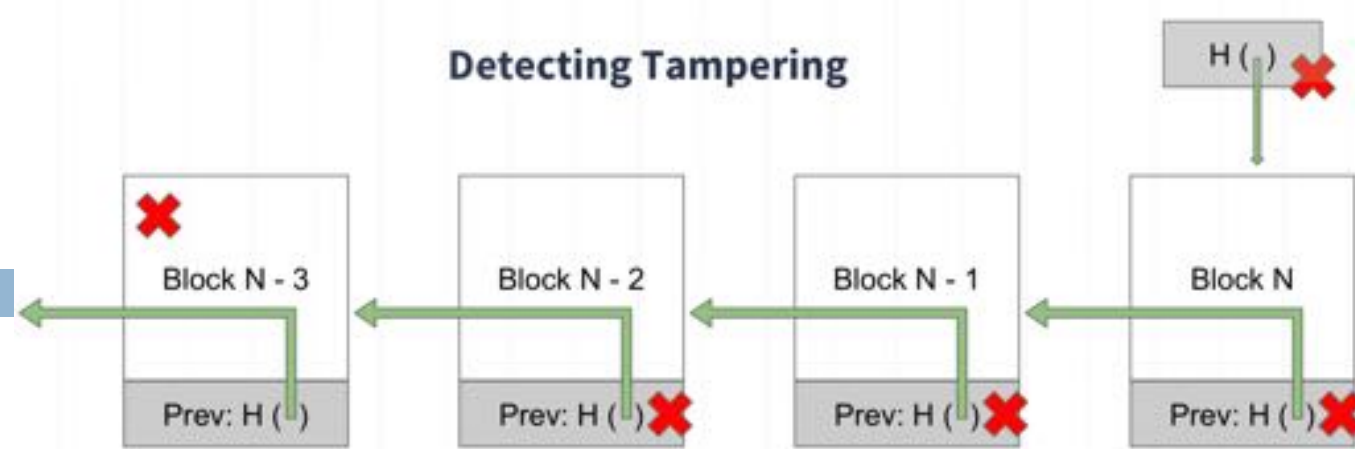


Tamper trial

- But now, the hash of this block doesn't match the hash that we hold
- The adversary can't tamper with that, because this is the value we remembered as being the head of the list



Recap tampering



- If the adversary wants to tamper with data **anywhere** in this entire chain, in order to keep the story consistent, he's going to have to tamper with hash pointers **all the way back to the beginning**
- And he's ultimately going to run into a **roadblock**, because he won't be able to tamper with the head of the list
- So we can build a blockchain like this containing as many blocks as we want, going back to some special block at the beginning of the list which we might call the **genesis block**. And that's a tamper evidence log built out of the block chamber

Digital Signature requirements for Bitcoin Addresses

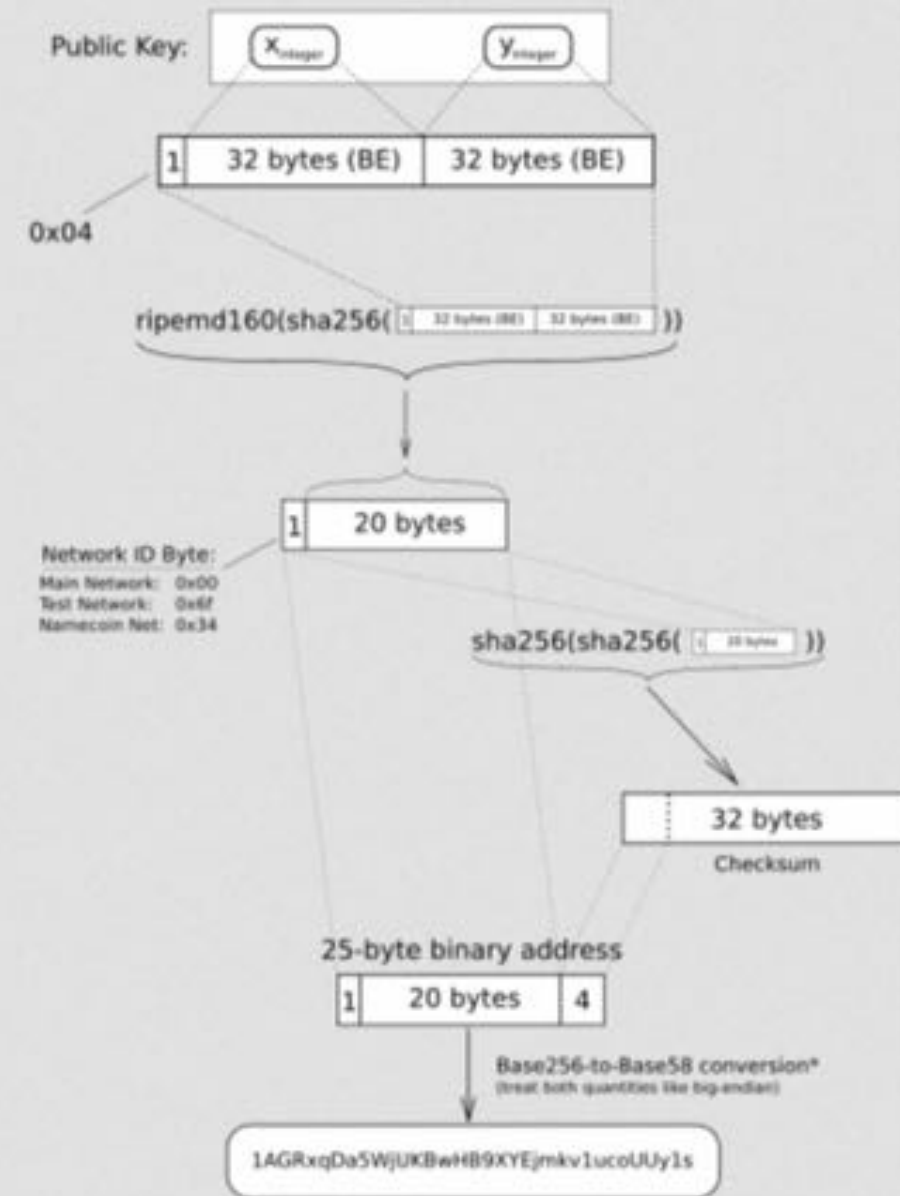


- ❑ Valid and **fast** signature verification
- ❑ It must be **almost impossible** to forge signatures
- ❑ Randomized Algorithms (ECDSA) offers a good source of randomness
- ❑ Limit on message size thanks of hash functions
- ❑ **Signing a Hash Pointer, it is like signing the whole structure behind**

Digital Signature ECDSA

- A Bitcoin address, or simply address, is an identifier of 26-35 alphanumeric characters, beginning with the number 1 or 3, that represents a possible destination for a Bitcoin payment
- Addresses can be generated at no cost by any user of Bitcoin

Elliptic-Curve Public Key to BTC Address conversion



*In a standard base conversion, the 0x00 byte on the left would be irrelevant (like writing '052' instead of just '52'), but in the BTC network the left-most zero bytes are carried through the conversion. So for every 0x00 byte on the left end of the binary address, we will attach one '0' character to the Base58 address. This is why main network addresses all start with '1'.

Cryptocoin as in “Satoshi Nakamoto” paper

A cryptocoin is a **chain of digital signatures** using **public/private keys**

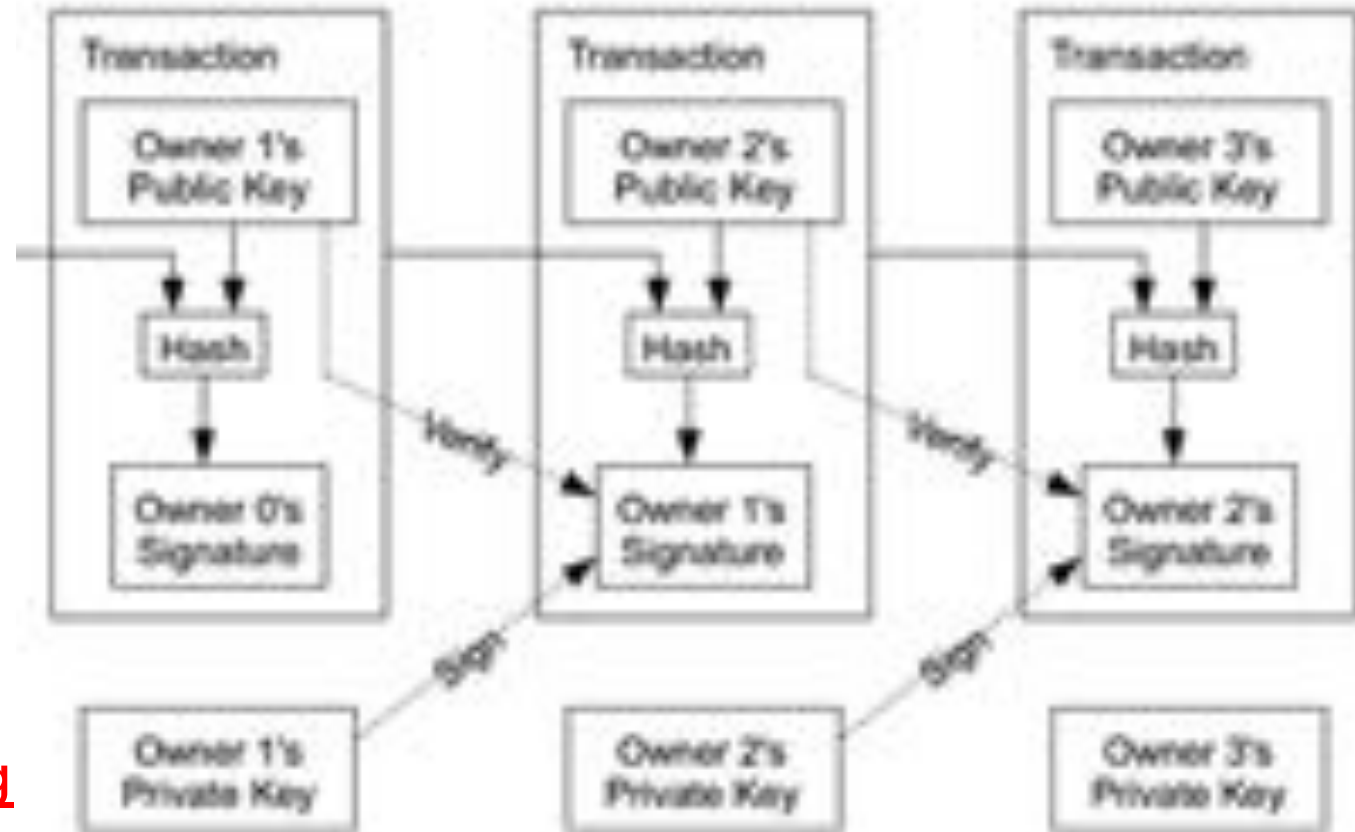
□ Bitcoin transfer: **Hash(Previous transaction, new Owner's public key)**

□ A payee can verify that the n^{th} Owner transferred some bitcoins to the $(n+1)^{\text{th}}$ Owner

□ Anyone can follow on the chain of ownership

□ All transactions are recorded in a Blockchain (the “paybook”)

□ The payee cannot check if received bitcoin are legitims: **double spending**



At this point enter the Double Spending Problem

- We need a way for the payee to know that the previous owners **did not sign any earlier transactions**
- The only way to confirm the absence of a transaction is to be **aware of all transactions**
- Transactions must be publicly announced, ie. **BROADCASTED**
- We need a system for participants **to agree on a single history** of the order in which they were received
- The payee needs proof that at the time of each transaction, the majority of nodes agreed it was the first received
- The solution is to implement a **distributed & peer-to-peer Timestamp Server**

About Distributed Consensus Protocols



- Remember we are in a potentially unreliable network with **faulty and malicious** nodes
 - ▣ Nodes may crash and may be malicious
- Network itself is **imperfect**
 - ▣ Not all pairs of nodes are connected
 - ▣ Faults in network
 - ▣ Latency

Proof of Work : Distributed Consensus in Bitcoin

- Bitcoin as a solution to **Byzantine Generals' Problem**
- Trying to find an agreement by exchanging information over an unreliable and potentially compromised network
- **Proof of Work** is a solution **without a central trusted authority**
- Remember: Bitcoin is a P2P system
 - ▣ When Alice want to pay Bob, she **BROADCAST** the transaction to the network i.e. to all Bitcoin connected nodes
 - ▣ Bob doesn't need to be connected to receive the transaction (i.e. the money)

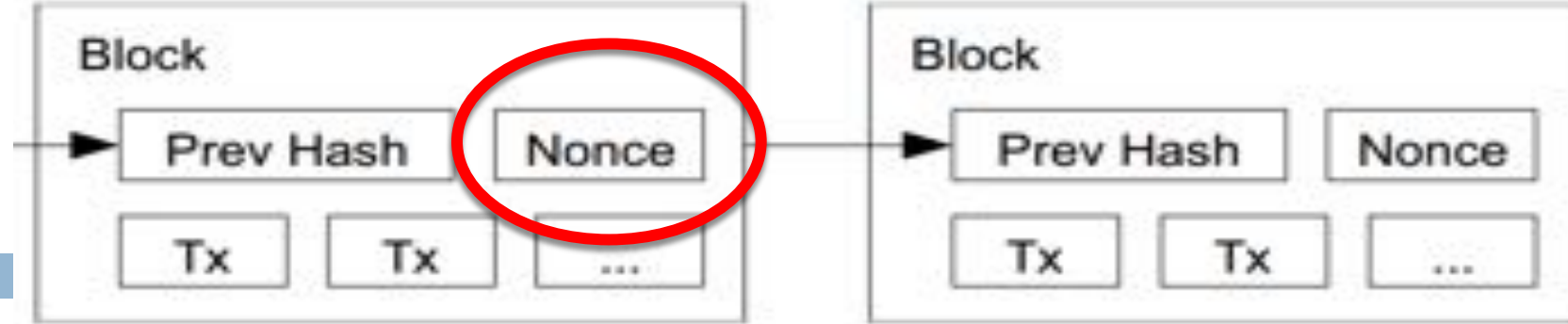
How Bitcoin solves Distributed Consensus?

- Bitcoin introduces **incentives to act honestly**
- **Consensus happens** over long time scale (**about 1 hour**)
- What is **double spending** : use your money twice
- **Double spend** probability **decrease exponentially** with the # of transactions
- Most common heuristics: ***6 confirmations*** of nodes that trust your transaction
- You're never 100% sure of a transaction is in consensus branch
- **Guarantee becomes a probabilistic notion**

Solving Distributed Consensus : proof of work

- New transactions are **broadcasted** to all nodes
- Each bitcoin node collects new transactions into a **block**
- Then, **Miners** try to solve an **Hash-Puzzle** using the **Force Bruce** algorithm
- In each “round”, a **random** node start broadcast its block
- Other nodes accept the block **only if** all transactions in it are valid (unspent, valid signatures)
- Nodes express their acceptance of the block by including its Hash in the next block they create

Proof-of-work



- Proof-of-work and activity related with mining and hash-puzzle
 - A new block contains **Tx** to validate and **Previous_Hash_Signature**
 - Hash Puzzle: find **nonce** such that

$$\text{SHA256}(\text{Previous_Hash_Signature} \mid \text{nonce} \mid \text{Tx}) < \mathbf{N}$$

- **..run..** when a Miner found the **nonce** (lucky) then broadcast it !!!

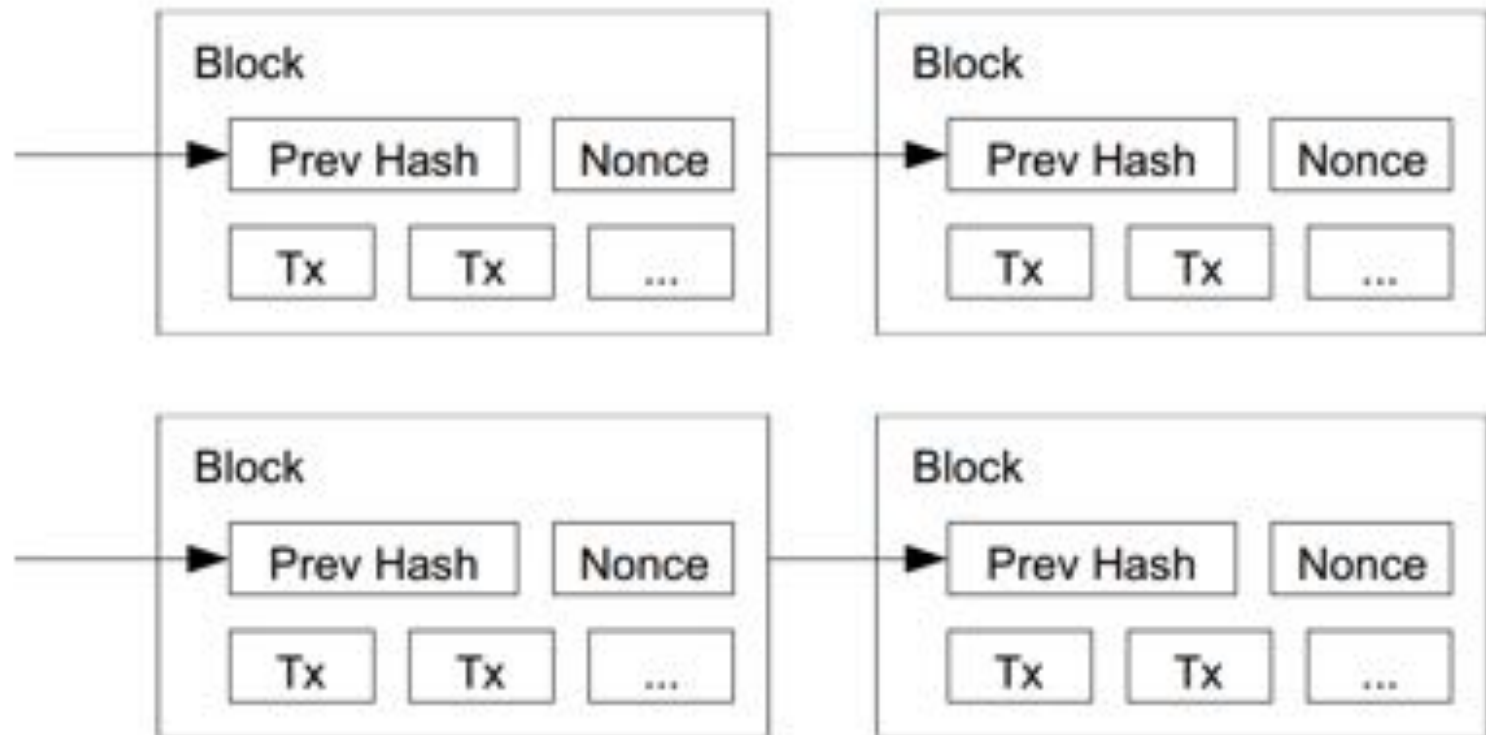
N is fixed by the BC protocol. Basically, this amounts to finding a hash value whose leading bits are zero. The work required is **exponential** in the number of zero bits required. Ex : $\mathbf{N} = \underbrace{0000\dots000}_{128 \text{ bit}} \underbrace{1111\dots1111}_{128 \text{ bit}}$

- Finding **N** is VERY hard, but verification is VERY fast (XOR)

About “Tie breaking”

- Two Miners may find a correct block simultaneously
 - ▣ Keep both and work on the first one
 - ▣ If one grows longer than the other, take the longer one

Two **different**
announced blocks may
satisfy the required
proof-of-work
(i.e. the hash-puzzle)

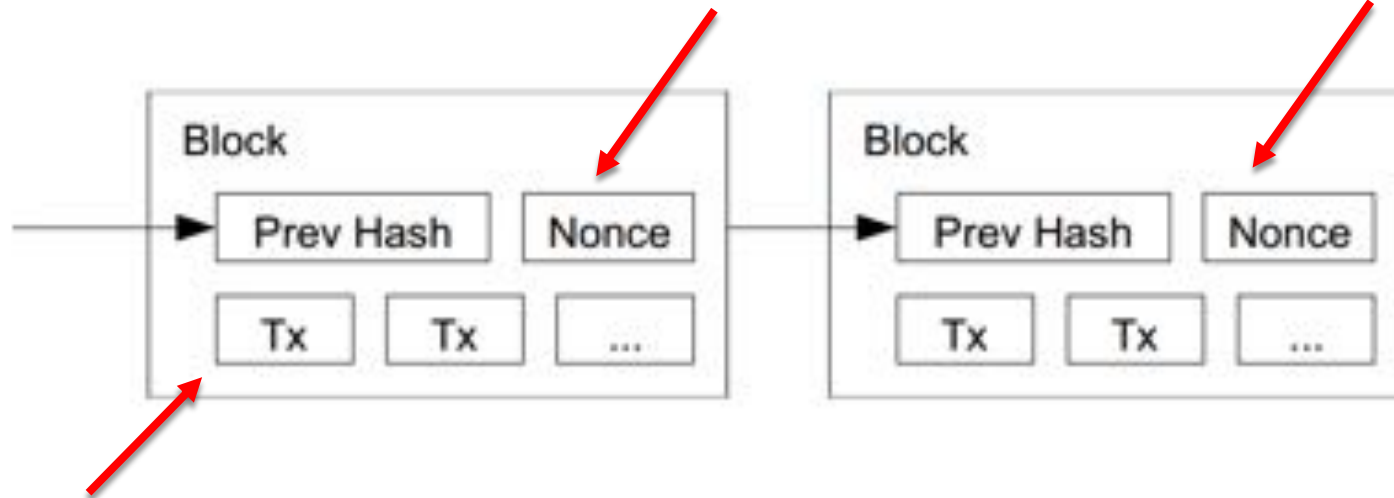


Reverting transactions is very very very hard...

- Reverting gets exponentially hard as the chain grows

2. Recompute nonce

3. Recompute the next nonce



1. Modify the transaction (revert or change the payer)

Resuming : Bitcoin & proof of work (timestamp server)

The steps to run the network are as follows:

- 1) New transactions are broadcast to all nodes. **of the Bitcoin P2P network**
- 2) Each node collects new transactions into a block.
- 3) Each node works on finding a difficult proof-of-work for its block. **CPU and Watt**
- 4) When a node finds a proof-of-work, it broadcasts the block to all nodes. **BTC P2P**
- 5) Nodes accept the block only if all transactions in it are valid and not already spent. **Easy**
- 6) Nodes express their acceptance of the block by working on creating the next block in the chain, using the hash of the accepted block as the previous hash. **BTC P2P**

Practical Limitation

- At least 10 mins to verify a transaction \$ (the block containing the Tx need to be published)
 - ▣ Agree to pay
 - ▣ Wait for the publication of one block (avg 10 mins) for the transaction \$ to go through
 - ▣ But, for a large transaction (\$\$) wait longer. Because if you wait longer it becomes more secure. For \$\$, you wait for six blocks (avg 1 hour)
 - NB: Bitcoin is a "Fiduciary currency". Bitcoin is not a "Fiat/Commodity" money
 - ▣ No intrinsic value
 - ▣ Based in that it will be generally accepted as a medium of exchange
- [FYI there 4 kinds of money : fiat, commodity, commercial-bank, and fiduciary]

Bitcoin Economics

- Rate limiting on the creation of a new block
 - Adapt to the “P2P network’s capacity”
 - A block created every 10 mins (approx. six blocks every hour)
 - How? Difficulty is adjusted every two weeks to keep the rate fixed as capacity/computing power increases
- N used in the puzzle to validate each new block: credited to the miner → **incentives for miners**
 - 50BTC initially. In 2013, 25BTC. In Feb 2018, **12.5BTC**, now is 6.25
 - Reward halved every 210,000 blocks (approx. every four years)
 - Reward can be spent only after 100 confirmations
 - Thus, the total number of Bitcoins will not exceed **21 million BTC then STOP !!!**
 - After this, we could experiments other reward mechanisms, i.e. a **transaction fee**

Overview

- Intro to Bitcoin
- Security Overview
- Bitcoin: Technical Details
- The practice of mining Bitcoin (computer HW perspective)

Far west (2006-2010)

- GPU: Radeon HD 6990 about 700 MH/s
- Butterfly Labs:
 - ▣ FPGA, ASIC



Hardware AMAZON

65nm Products - Units in Stock. Shipping Immediately.

1 GH/S BITCOIN MINER



Out of stock

50 GH/S BITCOIN MINER



\$2,499.00

Buy

500 GH/S BITCOIN MINER



\$22,484.00

Buy

28nm Products

BITCOIN MINING BY THE GH



\$10.83 / GH

Pre-Order

300 GH BITCOIN MINING CARD



\$2,800.00

Pre-Order

600 GH BITCOIN MINING CARD



\$4,680.00

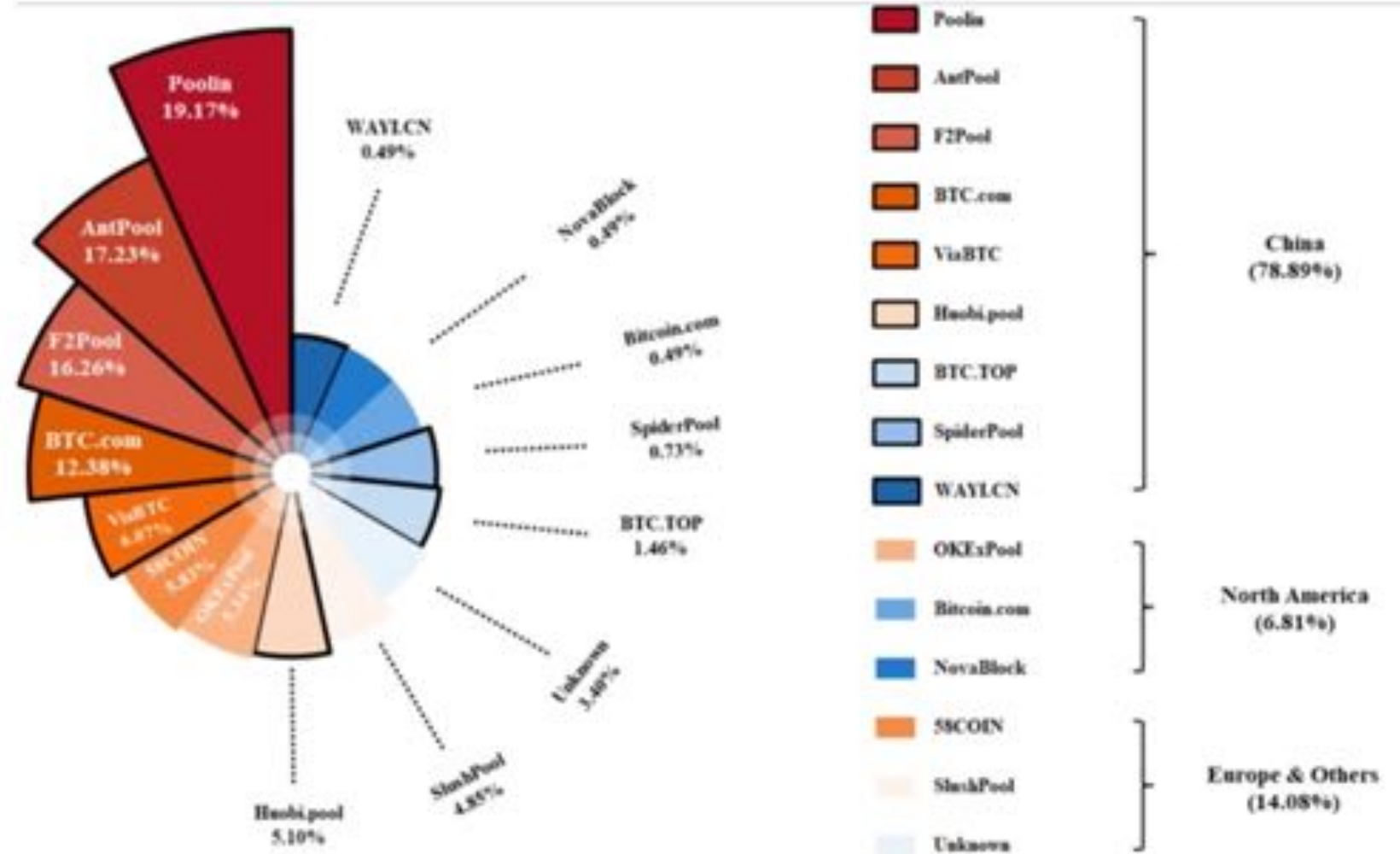
Pre-Order

Mining Now (Mining factory)









Who is mining now?

Where ?
Cina,
Georgia,
Sweden,
Ceck,
USA,
...



51% attack

What can a “51% attacker” do?

- Steal coins from existing address? 
- Suppress transactions from the Blockchain? 
- Suppress transactions from P2P Network? 
- Change the Block Reward? 
- Destroy confidence in Bitcoin?  

Summary

- Bitcoin combines techniques from cryptography and the right incentives
 - ▣ Nice design
 - ▣ A trait for popular systems
- Bitcoin is becoming industrialized
 - ▣ Miners form a pool
 - ▣ Mining hardware becomes sophisticated
 - ▣ Bitcoin exchange *piazza*
 - Derivative market, etc.
 - ▣ Government agencies are keeping an eye on them
 - ▣ Cloud mining
- Who will control Bitcoin in the end?

About Wallets

The wallet is a software able to store your private. It's also able to connect to P2P Bitcoin Network in order to receive and send transactions.

There different type of wallets:

- Full Nodes: Bitcoin Core, btc1, BitCore
- SPV Nodes: Electrum
- Android/iOS: Mycelium
- Hardware Wallet: Ladger Nano S, Trezor, KeepKey
- Online wallets: ... please don't!

...

Full node



- ❑ What is a Fully Validating Node?
- ❑ Permanently connected
- ❑ Store the entire Blockchain (250+GB)
- ❑ Hear and forward every node/transaction
- ❑ Keep track the UTXO set (Unspent Transaction Output)
- ❑ Validate and accept new blocks

Thin node



- ❑ Thin/SPV client (Simply Payment Verification)
- ❑ Idea: Don't store everything
- ❑ Store block headers only (only Markle Root for TXs)
- ❑ Request transactions as needed (verify incoming payment)
- ❑ Trust fully-validating nodes
- ❑ 1000x cost savings

About privacy

Bitcoin is not anonymous

Bitcoin is **pseudo-anonymous**.

The ledger is public. If someone find a connection between you and one of your bitcoin addresses, He can retrieve all the history of that particular address.

IF you buy from a KYC Exchange and withdraw your coin, you are leaving a trace that connect your person to that particular address.

Create as many Bitcoin addresses as you want! Even 1 every income payment!

About scam alert

Bitcoin is exploding and everybody is starting to look at it.

Bitcoin is also an innovative technology and speculation goals are, sadly, more effective than the innovation itself.

A lot of people see bitcoin as an opportunity to gain money.

... and a lot of people is ready to steal your investments.

About scam alert

What you should avoid:

- People who propose you methods to easily gain money.
- People who insist to give you hint how to invest your money.
- Ponzi schemes.
- Multilevel Networks.
- Pyramidal schemes.
- Genesis Mining.
- ... many more!

Simply, don't trust anyone. Do your own research and think critically.

About blockchain usecases

Bitcoin is the culmination of decades of research and it solve some important issues: **Double-Spending** and **The Byzantine Problem**.

Blockchain is also a reliable and immutable way to store publicly informations. After Bitcoin, others uses of the blockchain popped up:

- Proof of Ownership, Patent Records
- Digital Identity that protects privacy
- Smart Contracts
- Decentralized Marketplace
- Tokenization, Authentication and Authorization
- Governance
- ... many more!