

# Topics in Algebra: Cryptography

**Univ.-Prof. Dr. Goulmara ARZHANTSEVA**

WS 2018



# Digital currency

## Two major functionalities:

- A reliable process to **produce money**;
- A reliable process to **record transactions**.

Earlier currencies: **centralized** architecture, transactions are **secrete**.

Bitcoin etc.: **distributed** architecture, transactions are **public**.

# Digital currency

## Three major security requirements:

- **Non-repudiation** of transactions: if you commit to pay you cannot later deny it.
- **Integrity** of the entire transaction data set: it is correct and consistent, e.g. no double spending.
- **Pseudonymity** of transactions: one can dissociate the identity of a holder from any transaction.

# Digital currency

## Three major security requirements:

- **Non-repudiation** of transactions: if you commit to pay you cannot later deny it.
- **Integrity** of the entire transaction data set: it is correct and consistent, e.g. no double spending.
- **Pseudonymity** of transactions: one can dissociate the identity of a holder from any transaction.

The confidentiality of transactions is not required: they are public.

The non-repudiation is required for individual transactions.

The distributed architecture: it is easy to verify but not to modify, so the data origin authentication of a wider transaction data set is not required.

# Distributed ledger

Definition: **Distributed ledger**=Distributed Ledger Technology (DLT)

This is a digital **database** that is replicated, shared and synchronized by a **consensus algorithm**, across multiple nodes of a **peer-to-peer network**.

Distributed ledgers may be **unpermissioned** or **permissioned** regarding if anyone or only approved nodes can validate transactions.

# Distributed ledger

Each node replicates and saves an **identical** copy of the ledger and updates itself independently.

When a ledger update happens, each node constructs the new transaction, and then the nodes vote by the **consensus algorithm** on which copy is correct.

Once a consensus has been determined, all the other nodes update themselves with the new, **correct** copy of the ledger.

# Blockchain

A **blockchain** is one example of a distributed ledger. It can be either public or private. It is an unpermissioned ledger.

It contrasts with conventional ledgers as it sets rules about a transaction that are tied to the transaction itself (not to the entire database).

Security is through cryptographic **hash** functions and **digital signatures**.

The consensus algorithm is through **mining**.

# Block

## Definition: Block

This is a data item consisting of:

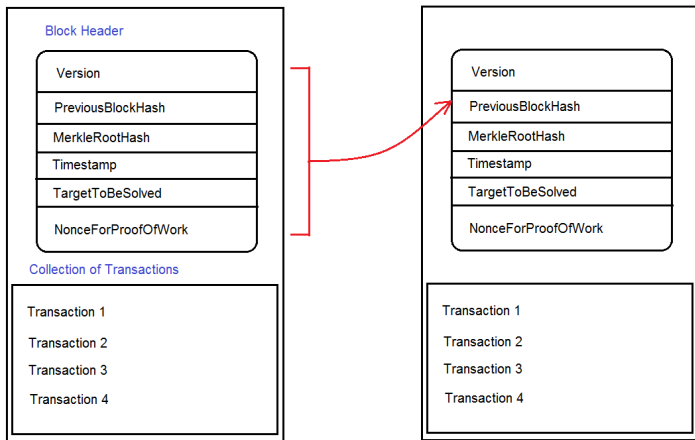
- A **block header**, including a **block hash** of the block header of the previous block in the **blockchain**;
- A list of **transactions**: each accepted transaction is in a block of the blockchain.

A **genesis block** is the very first block, created in 2009.

Every 10 minutes, on average, a new block is appended to the blockchain through **mining**.



# Block



A block in a blockchain. [image: blog.brakmic.com]

# Blockchain

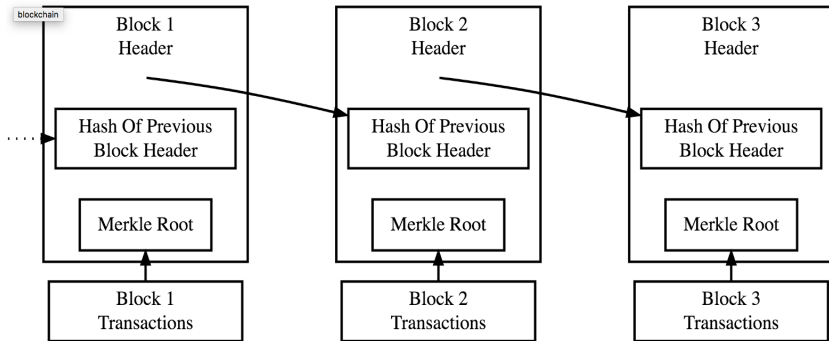
Definition: **Blockchain** is the distributed ledger supporting bitcoin:

It is a public record of a sequence of blocks (= block header + transactions) in chronological order.

It is used to verify the permanence of bitcoin transactions and to prevent double spending.

- The blockchain is stored and maintained by the nodes (users) in the bitcoin network;
- Some nodes store the full blockchain, while others are **lightweight** and only store block headers;
- Two major functionalities are guaranteed by **mining**;

# Blockchain



Simplified Bitcoin Block Chain

Blockchain. [image: Satoshi Nakamoto]

# Bitcoin address

A bitcoin address is a string of 26-35 alphanumeric characters; it starts with 1 or 3 and represents a **destination** for a bitcoin payment.

It is a destination 'account', to which the bitcoin can be transferred.

It is available to anyone from whom the user wishes to receive a transaction (like a public key).

A unique address is used for each transaction. A user can have many addresses.

# Bitcoin address

A bitcoin address is a string of 26-35 alphanumeric characters; it starts with 1 or 3 and represents a **destination** for a bitcoin payment.

It is a destination 'account', to which the bitcoin can be transferred.

It is available to anyone from whom the user wishes to receive a transaction (like a public key).

A unique address is used for each transaction. A user can have many addresses.

## Definition: Bitcoin address

A **bitcoin address** is a 160-bit hash of the public portion of a public/private ECDSA keypair.

The verification key of a verification / signature key pair for the ECDSA.

The verification key is  $\sim 512$ -bits, the signature key is  $\sim 256$ -bits.

# Bitcoin address

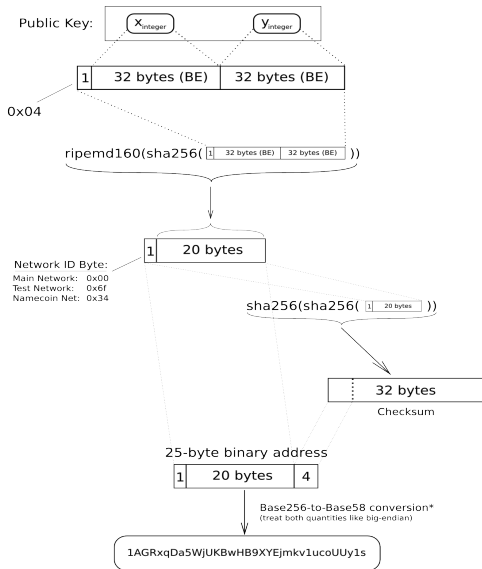
The bitcoin address is derived as follows:

- 1 Hash the verification key using SHA-256;
- 2 Hash the result using RIPEMD160;
- 3 Base58 **encode** the result (to convert binary to alphanumeric).

The verification key → a **pseudorandom** string of characters.

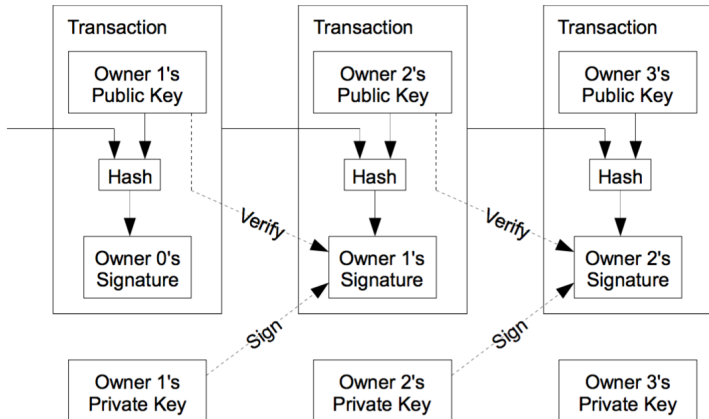
The encoding also provides a possibility for a **vanity** address.

## Elliptic-Curve Public Key to BTC Address conversion



# Bitcoin transaction

A **bitcoin transaction** is a digitally signed statement committing a payment to a bitcoin address.



Chaining of transactions. [image: Satoshi Nakamoto]



# Bitcoin transaction

A **bitcoin transaction** is a digitally signed statement committing a payment to a bitcoin address.

- 1 Input= the UTXO from a previous transaction, with scriptSig.
- 2 Output= the UTXO **binding** (with scriptPubKey) the recipient **bitcoin address** to a **transferred amount**.
- 3 Total input = transferred amount + fee (goes to miners).

Unspent Transaction Output=**UTXO**

scriptPubKey = locking, conditions required to spend the output.

scriptSig = unlocking, conditions allowing the output to be spent.

# Bitcoin transaction

A **bitcoin transaction** is a digitally signed statement committing a payment to a bitcoin address.

- 1 Input= the UTXO from a previous transaction, with scriptSig.
- 2 Output= the UTXO **binding** (with scriptPubKey) the recipient **bitcoin address** to a **transferred amount**.
- 3 Total input = transferred amount + fee (goes to miners).

Unspent Transaction Output=**UTXO**

scriptPubKey = locking, conditions required to spend the output.

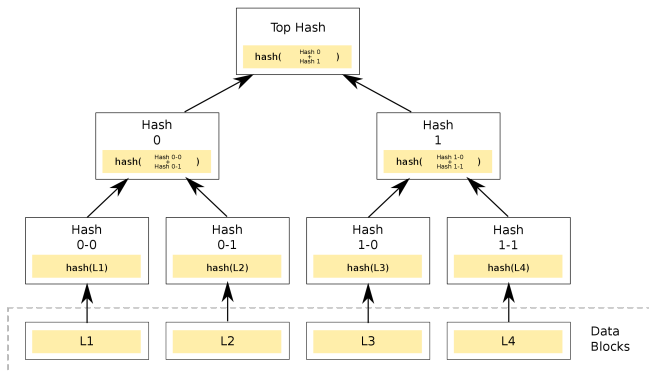
scriptSig = unlocking, conditions allowing the output to be spent.

One has to provide a **digital signature** and the corresponding **public key** as the unlocking script in the transaction input in order to allow the output (from a previous transaction) to be spend.

# Transaction verification

A **lightweight** node stores a block header.

Such nodes use a **Merkle tree**'1979 (a **hash tree**) to verify whether a transaction belongs to a given block.



Hash tree. [image: Wikimedia]

# Transaction verification

The hash function is the application of SHA-256 twice:

$$h(x) = \text{SHA-256}(\text{SHA-256}(x)).$$

A lightweight node stores a block header, a 256-bit root value.

Such a lightweight node gets a **verification** path of intermediate hash values, sufficient to compute the root from a transaction.

Example: a bitcoin block containing 1024 transactions requires a verification path of 10 intermediate hash values.

# Proof-of-work

Definition: **Proof-of-work** (PoW)

Dwork-Naor'1993

A consensus protocol, i.e. an irrefutable system to achieve agreement between various devices across a distributed network preventing exploitation.

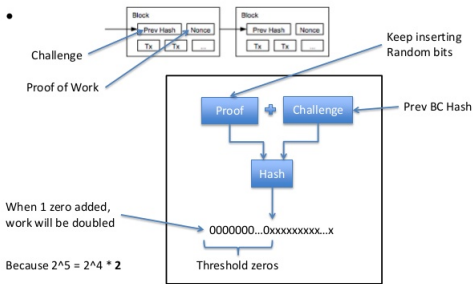
In bitcoin, it is used to achieve both of the two major functionalities: to produce money and to validate transactions.

21 million of bitcoins in total, by 2140.

# Bitcoin mining

The bitcoin **mining** is an 'exhaustive' search for preimages of the SHA-256(SHA-256(.)) hash function, trying to hash on random inputs.

## Proof of Work



Proof-of-work. [image: canardcoincoin.com]

As fo Aug. 2017, 72 of 256 hash bits of the **target** output must be zero.

# Bitcoin mining

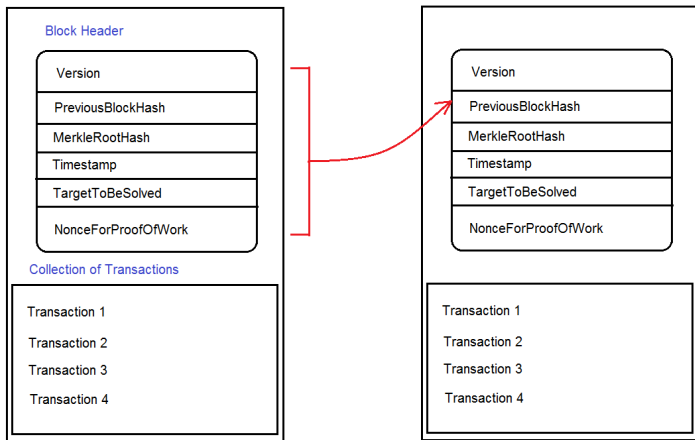
A **set preimage resistance**: given a set  $Z$  of potential outputs of the hash function  $h$ , it is difficult to find  $x$  with  $h(x) = z$  for some  $z \in Z$ .

The difficulty is controlled by the size of  $Z$ .

$Z$  is 'evenly spread' among possible outputs (the hash function provides a **pseudorandom** output).

$Z$  is defined to be the set of the hash function outputs **smaller** than a particular **target** output (the order is on binary numbers).

# Reminder: Block



A block in a blockchain. [image: blog.brakmic.com]



## More on design

Blockchain **forks** are possible but they can be managed:

- the longest blockchain version is accepted;
- the transactions from a ‘losing’ block are returned to the pool of ‘floating transactions’.

Flexibility:

- the difficulty of mining is adjusted to have a new block on average every 10min;
- 10min. is a trade-off between the speed of new block acceptance and the risks of forking.

Environmental impact!

# More on security

The main security issue is in logistic:

- loss of signing key;
- malicious mining farms.

The cryptographic security is in the hash functions:

- for data integrity;
- for pseudorandomness;
- for difficulty of mining.

and the ECDSA.

# Test questions

## Question 20

- (a) What other uses of cryptographic proofs-of-work do you know?
- (b) What are (dis)advantages of deploying distributed ledgers?

## Question 21

What is the length (=the number of intermediate hash values) of a verification path in the Merkle tree having  $n$  transactions? What is it for  $k$ -ary tree with  $n$ -leaves?

## Question 22

Why in your opinion is the difficulty of the proof of work in Bitcoin set to 10 minutes? What would go wrong if it was changed to 60 minutes or 10 seconds?