Distributed Consensus / The Blockchain

Christopher Jämthagen

Department of Electrical and Information Technology
Lund University, Sweden

May 12, 2015

Outline

1. **Background**
   - Hash functions
   - Digital signatures
2. **Distributed consensus**
   - Definition
   - Byzantine Generals Problem
   - Dynamic Membership Multiparty Signatures (DMMS)
   - The Blockchain
3. **Practical applications**
   - Timestamping
   - Cryptocurrency (Bitcoin)
   - Smart contracts
4. **Conclusion**

Outline

1. **Background**
   - Hash functions
   - Digital signatures
2. **Distributed consensus**
   - Definition
   - Byzantine Generals Problem
   - Dynamic Membership Multiparty Signatures (DMMS)
   - The Blockchain
3. **Practical applications**
   - Timestamping
   - Cryptocurrency (Bitcoin)
   - Smart contracts
4. **Conclusion**

Cryptographic hash functions

- Arbitrary input data to fixed-length hash value

Cryptographic hash functions

- Arbitrary input data to fixed-length hash value
- Important attributes include:
  - **pre-image resistance**
    Given a hash h it should be difficult to find any message m such that $h = hash(m)$

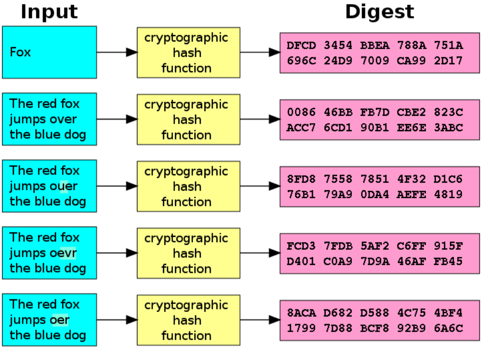
Cryptographic hash functions

- Arbitrary input data to fixed-length hash value
- Important attributes include:
  - **pre-image resistance**
    Given a hash h it should be difficult to find any message m such that $h = hash(m)$
  - **second pre-image resistance**
    Given an input m it should be difficult to find another input m' such that $m \neq m'$ and $hash(m) = hash(m')$

Cryptographic hash functions

- Arbitrary input data to fixed-length hash value
- Important attributes include:
  - **pre-image resistance**
    Given a hash h it should be difficult to find any message m such that $h = hash(m)$
  - **second pre-image resistance**
    Given an input m it should be difficult to find another input m' such that $m \neq m'$ and $hash(m) = hash(m')$
  - **collision resistance**
    It should be difficult to find two different messages m and m' such that $m \neq m'$ and $hash(m) = hash(m')$

Cryptographic hash functions



Input

Digest

Fox → cryptographic hash function → DFCD 3454 BBEA 788A 751A 696C 24D9 7009 CA99 2D17

The red fox jumps over the blue dog → cryptographic hash function → 0086 46BB FB7D CBE2 823C ACC7 6CD1 90B1 EE6E 3ABC

The red fox jumps ouer the blue dog → cryptographic hash function → 8FD8 7558 7851 4F32 D1C6 76B1 79A9 0DA4 AEFE 4819

The red fox jumps oevr the blue dog → cryptographic hash function → FCD3 7FDB 5AF2 C6FF 915F D401 C0A9 7D9A 46AF FB45

The red fox jumps oer the blue dog → cryptographic hash function → 8ACA D682 D588 4C75 4BF4 1799 7D88 BCF8 92B9 6A6C

Digital signatures

- A scheme for showing the authenticity of a piece of data

Digital signatures

- A scheme for showing the authenticity of a piece of data
- The signer creates a key pair which are mathematically linked

Digital signatures

- A scheme for showing the authenticity of a piece of data
- The signer creates a key pair which are mathematically linked
  - **Signing key** - used to create the signature and must be kept private

Digital signatures

- A scheme for showing the authenticity of a piece of data
- The signer creates a key pair which are mathematically linked
  - **Signing key** - used to create the signature and must be kept private
  - **Verification key** - used to verify a signature and may be published

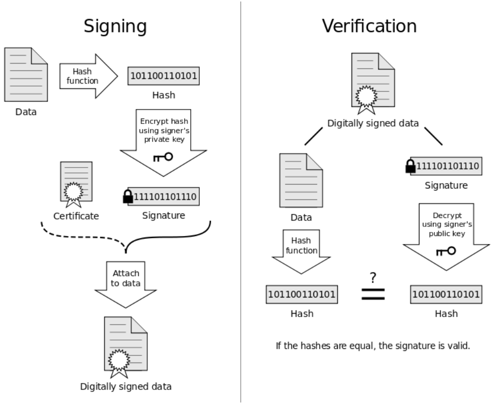
Digital signatures

- A scheme for showing the authenticity of a piece of data
- The signer creates a key pair which are mathematically linked
  - **Signing key** - used to create the signature and must be kept private
  - **Verification key** - used to verify a signature and may be published
- Security relies on unforgeability by computationally bounded adversary

Digital signatures

- A scheme for showing the authenticity of a piece of data
- The signer creates a key pair which are mathematically linked
    - **Signing key** - used to create the signature and must be kept private
    - **Verification key** - used to verify a signature and may be published
- Security relies on unforgeability by computationally bounded adversary
- **Proof-of-Knowledge** - Proof that you know the private key

Digital signature



Outline

1. **Background**
   - Hash functions
   - Digital signatures
2. **Distributed consensus**
   - Definition
   - Byzantine Generals Problem
   - Dynamic Membership Multiparty Signatures (DMMS)
   - The Blockchain
3. **Practical applications**
   - Timestamping
   - Cryptocurrency (Bitcoin)
   - Smart contracts
4. **Conclusion**

Distributed consensus
Definition

- A global agreement between many mutually-distrusting parties

Distributed consensus

Definition

- A global agreement between many mutually-distrusting parties
- Parties may:
  - Lack identities

Distributed consensus
Definition

- A global agreement between many mutually-distrusting parties
- Parties may:
  - Lack identities
  - Join and leave the network

Distributed consensus

Definition

- A global agreement between many mutually-distrusting parties
- Parties may:
  - Lack identities
  - Join and leave the network
    - At any time

Distributed consensus
Definition

- A global agreement between many mutually-distrusting parties
- Parties may:
  - Lack identities
  - Join and leave the network
    - At any time
    - At no cost

Distributed consensus
Definition

- A global agreement between many mutually-distrusting parties
- Parties may:
  - Lack identities
  - Join and leave the network
    - At any time
    - At no cost
    - Without any third parties permission

Distributed consensus
Definition

- A global agreement between many mutually-distrusting parties
- Parties may:
  - Lack identities
  - Join and leave the network
    - At any time
    - At no cost
    - Without any third parties permission
- Difficult problem
  - Ordinary consensus is an order of magnitude more efficient to solve with trusted third parties performing the signing

Distributed consensus
Definition

- A global agreement between many mutually-distrusting parties
- Parties may:
  - Lack identities
  - Join and leave the network
    - At any time
    - At no cost
    - Without any third parties permission
- Difficult problem
  - Ordinary consensus is an order of magnitude more efficient to solve with trusted third parties performing the signing
  - Efficiency trade-off for decentralization

Distributed consensus
Definition

- A global agreement between many mutually-distrusting parties
- Parties may:
  - Lack identities
  - Join and leave the network
    - At any time
    - At no cost
    - Without any third parties permission
- Difficult problem
  - Ordinary consensus is an order of magnitude more efficient to solve with trusted third parties performing the signing
  - Efficiency trade-off for decentralization
- The consensus problem illustrated - Byzantine Generals Problem

Distributed consensus

Two Generals Problem

- Players: Two generals, their respective armies and their messengers

Distributed consensus

Two Generals Problem

- Players: Two generals, their respective armies and their messengers
- Goal: Invade town

Distributed consensus

Two Generals Problem

- Players: Two generals, their respective armies and their messengers
- Goal: Invade town
- Strategy: Messengers used to communicate between the generals

Distributed consensus

Two Generals Problem

- Players: Two generals, their respective armies and their messengers
- Goal: Invade town
- Strategy: Messengers used to communicate between the generals
- Prerequisites:
  - For success both armies must attack at the same time

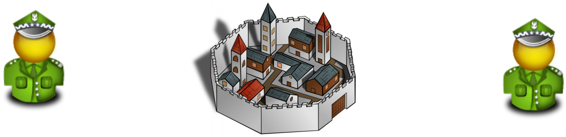
Distributed consensus

Two Generals Problem

- Players: Two generals, their respective armies and their messengers
- Goal: Invade town
- Strategy: Messengers used to communicate between the generals
- Prerequisites:
  - For success both armies must attack at the same time
  - Messengers must pass through town (insecure communication channel)

Distributed consensus

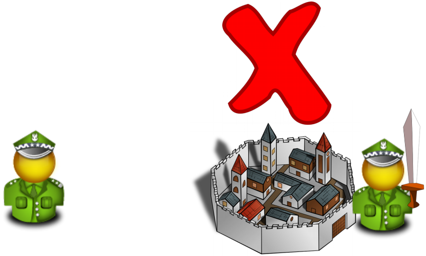
Two Generals Problem



Distributed consensus
Two Generals Problem

Distributed consensus

Two Generals Problem



Distributed consensus

Two Generals Problem

Alice

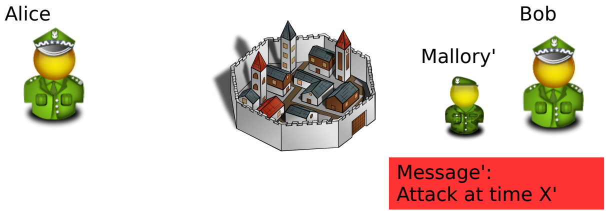
Mallory

Bob

Message:
Attack at time X

Distributed consensus

Two Generals Problem



Alice

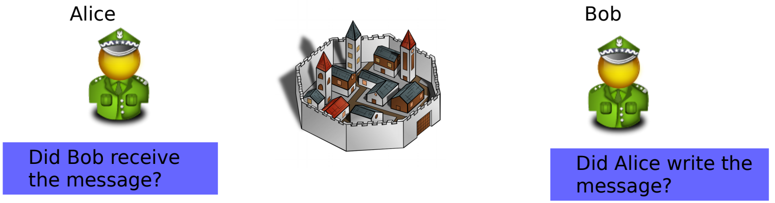
Mallory'

Bob

Message':
Attack at time X'

Distributed consensus

Two Generals Problem

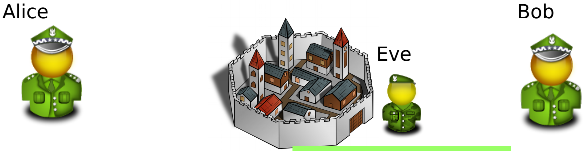


Did Bob receive the message?

Did Alice write the message?

Distributed consensus

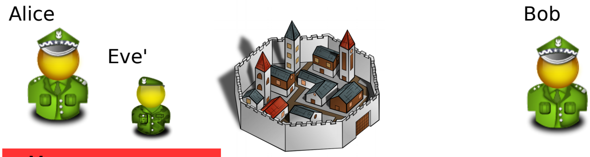
Two Generals Problem



Message:
Attack at time X'
ACK

Distributed consensus

Two Generals Problem



Alice

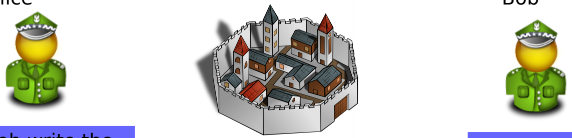
Eve'

Bob

Message:
Attack at time X''
ACK

Distributed consensus

Two Generals Problem



Alice

Bob

Did Bob write the message?

Did Alice receive the message?

Distributed consensus

Byzantine Generals Problem

Dynamic Membership Multiparty Signature

- Formed by a set of signers with no fixed size

Dynamic Membership Multiparty Signature

- Formed by a set of signers with no fixed size
- Proof-of-work instead of Proof-of-knowledge

Dynamic Membership Multiparty Signature

- Formed by a set of signers with no fixed size
- Proof-of-work instead of Proof-of-knowledge
- Signature of computational power

Dynamic Membership Multiparty Signature

- Formed by a set of signers with no fixed size
- Proof-of-work instead of Proof-of-knowledge
- Signature of computational power
    - Allows anonymous membership

Dynamic Membership Multiparty Signature

- Formed by a set of signers with no fixed size
- Proof-of-work instead of Proof-of-knowledge
- Signature of computational power
    - Allows anonymous membership
    - No risk of Sybil attacks

Dynamic Membership Multiparty Signature

- Formed by a set of signers with no fixed size
- Proof-of-work instead of Proof-of-knowledge
- Signature of computational power
  - Allows anonymous membership
  - No risk of Sybil attacks
    - One party joins many times

Dynamic Membership Multiparty Signature

- Formed by a set of signers with no fixed size
- Proof-of-work instead of Proof-of-knowledge
- Signature of computational power
  - Allows anonymous membership
  - No risk of Sybil attacks
    - One party joins many times
- DMMS signers are called miners

Dynamic Membership Multiparty Signature

Proof-of-Work

- A hash function produces fixed-size output from arbitrary input

Dynamic Membership Multiparty Signature

Proof-of-Work

- A hash function produces fixed-size output from arbitrary input
- Keep hashing data with small variations until some condition is met

Dynamic Membership Multiparty Signature
Proof-of-Work

- A hash function produces fixed-size output from arbitrary input
- Keep hashing data with small variations until some condition is met
  - Input is valid if its hash is lower than some target value

Dynamic Membership Multiparty Signature

Proof-of-Work

- A hash function produces fixed-size output from arbitrary input
- Keep hashing data with small variations until some condition is met
  - Input is valid if its hash is lower than some target value
  - When condition is met, verification is easy

Dynamic Membership Multiparty Signature

Proof-of-Work

- A hash function produces fixed-size output from arbitrary input
- Keep hashing data with small variations until some condition is met
  - Input is valid if its hash is lower than some target value
  - When condition is met, verification is easy
- Hash(Message||Nonce)

Dynamic Membership Multiparty Signature

Proof-of-Work

- A hash function produces fixed-size output from arbitrary input
- Keep hashing data with small variations until some condition is met
  - Input is valid if its hash is lower than some target value
  - When condition is met, verification is easy
- Hash(Message$||$Nonce)
- "Hello, world!0" => 1312af178c253f84028d48...

Dynamic Membership Multiparty Signature

Proof-of-Work

- A hash function produces fixed-size output from arbitrary input
- Keep hashing data with small variations until some condition is met
  - Input is valid if its hash is lower than some target value
  - When condition is met, verification is easy
- Hash(Message||Nonce)
- "Hello, world!0" => 1312af178c253f84028d48...
- "Hello, world!4250" => 0000c3af42fc31103f1fdc0...

The Blockchain
Introduction

- A collection of DMMS authenticated data

The Blockchain
Introduction

- A collection of DMMS authenticated data
- Data is included in blocks

The Blockchain
Introduction

- A collection of DMMS authenticated data
- Data is included in blocks
- Every block has a block header (80 bytes)

The Blockchain
Introduction

- A collection of DMMS authenticated data
- Data is included in blocks
- Every block has a block header (80 bytes)
  - Version

The Blockchain
Introduction

- A collection of DMMS authenticated data
- Data is included in blocks
- Every block has a block header (80 bytes)
  - Version
  - hashPrevBlock

The Blockchain
Introduction

- A collection of DMMS authenticated data
- Data is included in blocks
- Every block has a block header (80 bytes)
  - Version
  - hashPrevBlock
  - hashMerkleRoot

The Blockchain
Introduction

- A collection of DMMS authenticated data
- Data is included in blocks
- Every block has a block header (80 bytes)
  - Version
  - hashPrevBlock
  - hashMerkleRoot
  - Time

The Blockchain
Introduction

- A collection of DMMS authenticated data
- Data is included in blocks
- Every block has a block header (80 bytes)
  - Version
  - hashPrevBlock
  - hashMerkleRoot
  - Time
  - Bits

The Blockchain
Introduction

- A collection of DMMS authenticated data
- Data is included in blocks
- Every block has a block header (80 bytes)
  - Version
  - hashPrevBlock
  - hashMerkleRoot
  - Time
  - Bits - Target value for output hash

The Blockchain
Introduction

- A collection of DMMS authenticated data
- Data is included in blocks
- Every block has a block header (80 bytes)
  - Version
  - hashPrevBlock
  - hashMerkleRoot
  - Time
  - Bits - Target value for output hash
  - Nonce

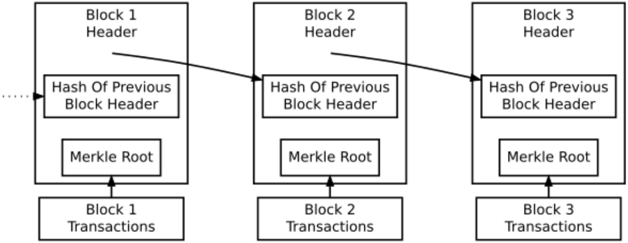
The Blockchain
Introduction

- A collection of DMMS authenticated data
- Data is included in blocks
- Every block has a block header (80 bytes)
  - Version
  - hashPrevBlock
  - hashMerkleRoot
  - Time
  - Bits - Target value for output hash
  - Nonce
- When Hash(blockheader) < bits, the proof-of-work for the block is valid
  - Start working on next block

The Blockchain
Introduction

- A collection of DMMS authenticated data
- Data is included in blocks
- Every block has a block header (80 bytes)
  - Version
  - hashPrevBlock
  - hashMerkleRoot
  - Time
  - Bits - Target value for output hash
  - Nonce
- When Hash(blockheader) < bits, the proof-of-work for the block is valid
  - Start working on next block
  - Not necessarily a valid DMMS

The blockchain
Illustrated



Simplified Bitcoin Block Chain

The blockchain
Security

- Clear differences between a DMMS and regular digital signatures

The blockchain
Security

- Clear differences between a DMMS and regular digital signatures
  - Forging is not an issue

The blockchain

Security

- Clear differences between a DMMS and regular digital signatures
  - Forging is not an issue
- Modifying data in the block invalidates the DMMS

The blockchain
Security

- Clear differences between a DMMS and regular digital signatures
  - Forging is not an issue
- Modifying data in the block invalidates the DMMS
  - Possible to produce another valid DMMS

The blockchain
Security

- A valid block receives a lock

The blockchain
Security

- A valid block receives a lock
- When another block has been created, previous block have two locks

The blockchain
Security

- A valid block receives a lock
- When another block has been created, previous block have two locks
- The longest chain with most work performed on it is the "real" chain

The blockchain
Security

- A valid block receives a lock
- When another block has been created, previous block have two locks
- The longest chain with most work performed on it is the "real" chain
- Secure under the assumption of an honest majority

The blockchain
Incentives

- Who will put work towards extending the blockchain?

The blockchain
Incentives

- Who will put work towards extending the blockchain?
  - Altruism is not secure

The blockchain

Incentives

- Who will put work towards extending the blockchain?
  - Altruism is not secure
  - Economic incentive necessary

The blockchain
Incentives

- Who will put work towards extending the blockchain?
  - Altruism is not secure
  - Economic incentive necessary
    - Issue scarce tokens in each block

The blockchain
Incentives

- Who will put work towards extending the blockchain?
  - Altruism is not secure
  - Economic incentive necessary
    - Issue scarce tokens in each block
    - Transparent issuing schedule

The blockchain
Incentives

- Who will put work towards extending the blockchain?
  - Altruism is not secure
  - Economic incentive necessary
    - Issue scarce tokens in each block
    - Transparent issuing schedule
    - Fees for including data in a block

The blockchain

Incentives

- Who will put work towards extending the blockchain?
  - Altruism is not secure
  - Economic incentive necessary
    - Issue scarce tokens in each block
    - Transparent issuing schedule
    - Fees for including data in a block
    - Double spending mitigation needed

The blockchain
Incentives

- Who will put work towards extending the blockchain?
  - Altruism is not secure
  - Economic incentive necessary
    - Issue scarce tokens in each block
    - Transparent issuing schedule
    - Fees for including data in a block
    - Double spending mitigation needed
    - Correct proof-of-work doesn’t equal a valid DMMS

The blockchain
Operating modes

- Nodes connect to eachother in a P2P network

The blockchain
Operating modes

- Nodes connect to eachother in a P2P network
- Two security levels of operation

The blockchain
Operating modes

- Nodes connect to eachother in a P2P network
- Two security levels of operation
  - Full verifying node

The blockchain

Operating modes

- Nodes connect to eachother in a P2P network
- Two security levels of operation
  - Full verifying node
    - Validates all data in the blocks according to given rules

The blockchain
Operating modes

- Nodes connect to eachother in a P2P network
- Two security levels of operation
  - Full verifying node
    - Validates all data in the blocks according to given rules
  - SPV (Simplified Payments Verification) node

The blockchain

Operating modes

- Nodes connect to eachother in a P2P network
- Two security levels of operation
  - Full verifying node
    - Validates all data in the blocks according to given rules
  - SPV (Simplified Payments Verification) node
    - Validates only the proof-of-work

The blockchain

Operating modes

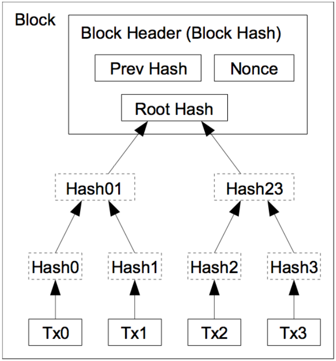
- Nodes connect to eachother in a P2P network
- Two security levels of operation
  - Full verifying node
    - Validates all data in the blocks according to given rules
  - SPV (Simplified Payments Verification) node
    - Validates only the proof-of-work
    - Merkle tree branches used to proove data in a block

The blockchain
Operating modes

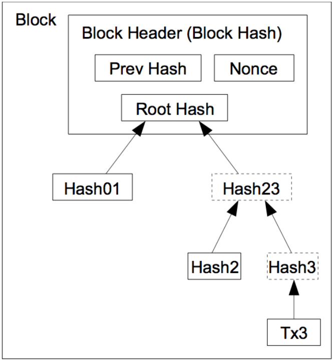
- Nodes connect to eachother in a P2P network
- Two security levels of operation
    - Full verifying node
        - Validates all data in the blocks according to given rules
    - SPV (Simplified Payments Verification) node
        - Validates only the proof-of-work
        - Merkle tree branches used to proove data in a block
        - Privacy issues

The Blockchain

Merkle trees and proofs



Transactions Hashed in a Merkle Tree



After Pruning Tx0-2 from the Block

The blockchain

Consistency vs. Correctness

- Normally protocol specification is the definition

The blockchain
Consistency vs. Correctness

- Normally protocol specification is the definition
- For distributed consensus systems, implementation defines the protocol

The blockchain

Consistency vs. Correctness

- Normally protocol specification is the definition
- For distributed consensus systems, implementation defines the protocol
- Bugs may not always be fixable

The blockchain
Consistency vs. Correctness

- Normally protocol specification is the definition
- For distributed consensus systems, implementation defines the protocol
- Bugs may not always be fixable
  - May cause split in the blockchain

The blockchain
Consistency vs. Correctness

- Normally protocol specification is the definition
- For distributed consensus systems, implementation defines the protocol
- Bugs may not always be fixable
  - May cause split in the blockchain
  - Bugs may become part of the specification

The blockchain

Forking

- Forking occurs either

The blockchain
Forking

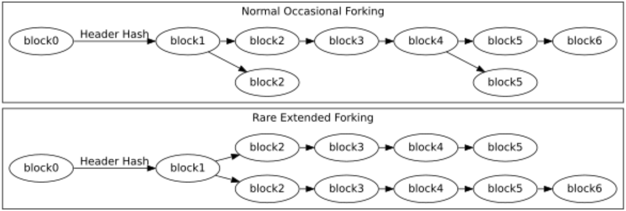
- Forking occurs either
  - When two miners finds a block at the same time

The blockchain
Forking

- Forking occurs either
  - When two miners finds a block at the same time
  - When the network nodes cant agree on what rules apply

The blockchain
Forking

- Forking occurs either
  - When two miners finds a block at the same time
  - When the network nodes cant agree on what rules apply
    - Can be intentional and unintentional



Outline

1. **Background**
   - Hash functions
   - Digital signatures
2. **Distributed consensus**
   - Definition
   - Byzantine Generals Problem
   - Dynamic Membership Multiparty Signatures (DMMS)
   - The Blockchain
3. **Practical applications**
   - Timestamping
   - Cryptocurrency (Bitcoin)
   - Smart contracts
4. **Conclusion**

Proof of existence

- Proof of data existence at specific point of time

Proof of existence

- Proof of data existence at specific point of time
- Hash of data included in a specific block

Proof of existence

- Proof of data existence at specific point of time
- Hash of data included in a specific block
- Block has a timestamp

Proof of existence

- Proof of data existence at specific point of time
- Hash of data included in a specific block
- Block has a timestamp
- http://www.proofofexistence.com/

Proof of existence

- Proof of data existence at specific point of time
- Hash of data included in a specific block
- Block has a timestamp
- http://www.proofofexistence.com/
- http://factom.org/ - Large scale timestamping

Bitcoin

- Currency, commodity and platform

Bitcoin

- Currency, commodity and platform
- 10 minutes between blocks

Bitcoin

- Currency, commodity and platform
- 10 minutes between blocks
  - Target for DMMS adjusted every 2016 blocks (2 weeks)

Bitcoin

- Currency, commodity and platform
- 10 minutes between blocks
  - Target for DMMS adjusted every 2016 blocks (2 weeks)
- 25 bitcoins + fees awarded per block

Bitcoin

- Currency, commodity and platform
- 10 minutes between blocks
  - Target for DMMS adjusted every 2016 blocks (2 weeks)
- 25 bitcoins + fees awarded per block
  - Reward halves every four years

Bitcoin

- Currency, commodity and platform
- 10 minutes between blocks
  - Target for DMMS adjusted every 2016 blocks (2 weeks)
- 25 bitcoins + fees awarded per block
  - Reward halves every four years
  - Maximum 21 million bitcoins

Bitcoin

- Currency, commodity and platform
- 10 minutes between blocks
  - Target for DMMS adjusted every 2016 blocks (2 weeks)
- 25 bitcoins + fees awarded per block
  - Reward halves every four years
  - Maximum 21 million bitcoins
- Total hashrate ~350 Petahashes/second

Bitcoin

- Currency, commodity and platform
- 10 minutes between blocks
  - Target for DMMS adjusted every 2016 blocks (2 weeks)
- 25 bitcoins + fees awarded per block
  - Reward halves every four years
  - Maximum 21 million bitcoins
- Total hashrate ~350 Petahashes/second
  - "If all Googles servers would start hashing they would have <1% of total network hashrate"

Smart contracts

- Protocols that enforces agreements between participants

Smart contracts

- Protocols that enforces agreements between participants
- Transactions written in a scripting language

Smart contracts

- Protocols that enforces agreements between participants
- Transactions written in a scripting language
- Smart property

Smart contracts

- Protocols that enforces agreements between participants
- Transactions written in a scripting language
- Smart property
- Payment Channels

Smart contracts

- Protocols that enforces agreements between participants
- Transactions written in a scripting language
- Smart property
- Payment Channels
- Stock exchange

Smart contracts

- Protocols that enforces agreements between participants
- Transactions written in a scripting language
- Smart property
- Payment Channels
- Stock exchange
- Decentralized Autonomous Company/Organization (DAC/DAO)

Smart contracts

- Protocols that enforces agreements between participants
- Transactions written in a scripting language
- Smart property
- Payment Channels
- Stock exchange
- Decentralized Autonomous Company/Organization (DAC/DAO)
- Futarchy

Outline

1. **Background**
   - Hash functions
   - Digital signatures
2. **Distributed consensus**
   - Definition
   - Byzantine Generals Problem
   - Dynamic Membership Multiparty Signatures (DMMS)
   - The Blockchain
3. **Practical applications**
   - Timestamping
   - Cryptocurrency (Bitcoin)
   - Smart contracts
4. **Conclusion**

Conclusion

- The blockchain achieves
  - Distributed consensus
  - Decentralization
  - A complete public record of immutable history
  - Censorship resistance

Questions?