NPTEL ONLINE CERTIFICATION COURSES

Blockchain and its applications

Prof. Sandip Chakraborty
Department of Computer Science & Engineering
Indian Institute of Technology Kharagpur

Lecture 11: Bitcoin Mining and Beyond

CONCEPTS COVERED

- Bitcoin Mining
- The Economic Model of Bitcoin
- Popularity of Cryptocurrencies

KEYWORDS

- Mining and Miners
- Mining reward
- Bitcoin Price
- Beyond Bitcoins

Bitcoin Mining: The Key to Consensus

- There are special nodes, called the Miners
- Miners propose new blocks – solve the puzzle (find the nonce corresponding to a target block hash), and add the solution as a **proof** of solving the challenge to be the leader
  - Solving the challenge needs some work to be done – ***Proof of Work (PoW)***

Bitcoin Mining: The Key to Consensus

- There are special nodes, called the Miners
- Miners propose new blocks – solve the puzzle (find the nonce corresponding to a target block hash), and add the solution as a **proof** of solving the challenge to be the leader
  - Solving the challenge needs some work to be done – ***Proof of Work (PoW)***

Why someone would want to be the leader?

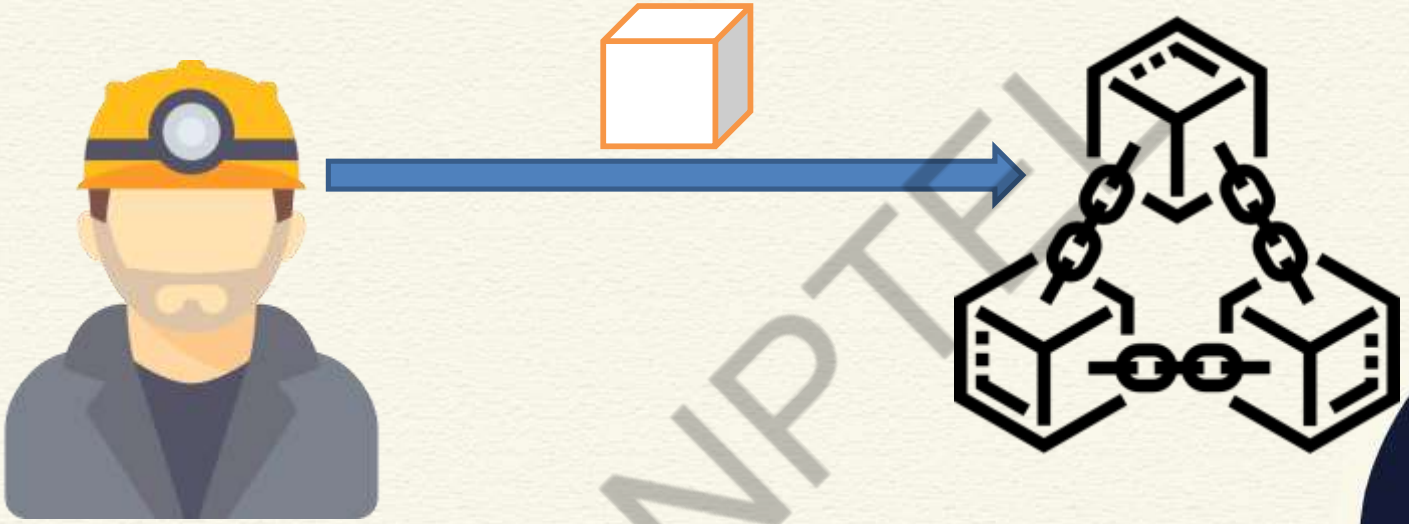
Bitcoin Mining: The Key to Consensus

- There are special nodes, called the Miners
- Miners propose new blocks – solve the puzzle (find the nonce corresponding to a target block hash), and add the solution as a **proof** of solving the challenge to be the leader
  - Solving the challenge needs some work to be done – ***Proof of Work (PoW)***

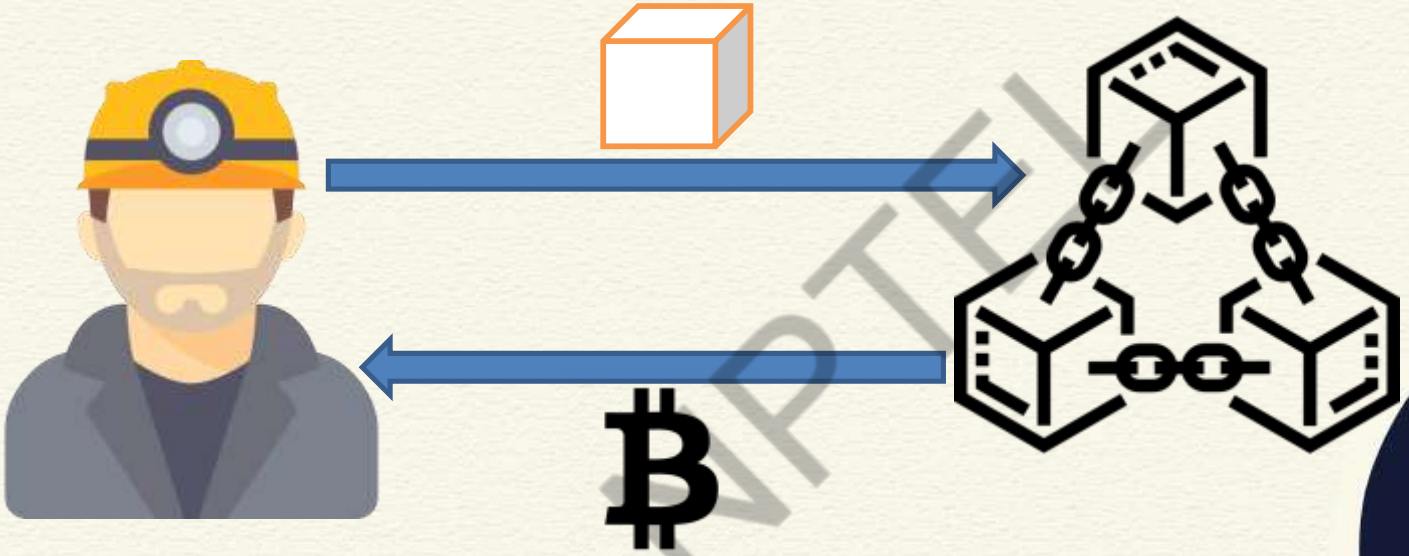
Why someone would want to be the leader?

Earn money (bitcoin) by solving the puzzle!

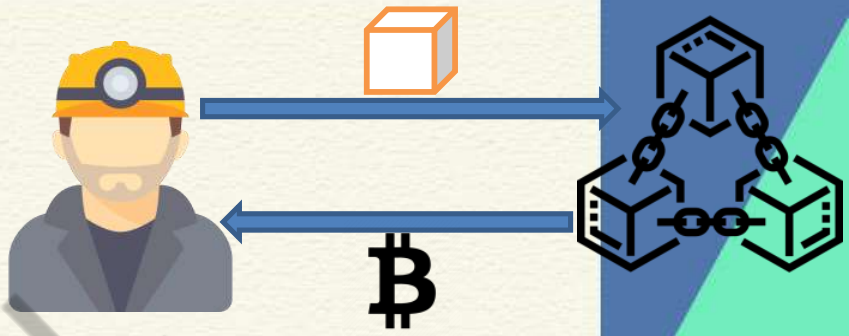
Mining a Block: The Reward



Mining a Block: The Reward

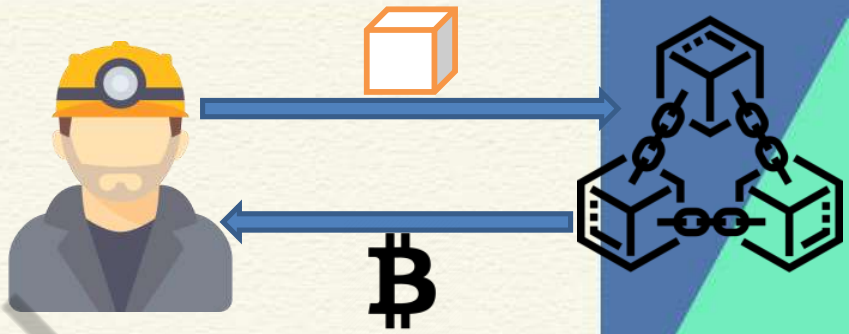


The Economics behind Reward



Encourage the community to participate in the mining through incentivization

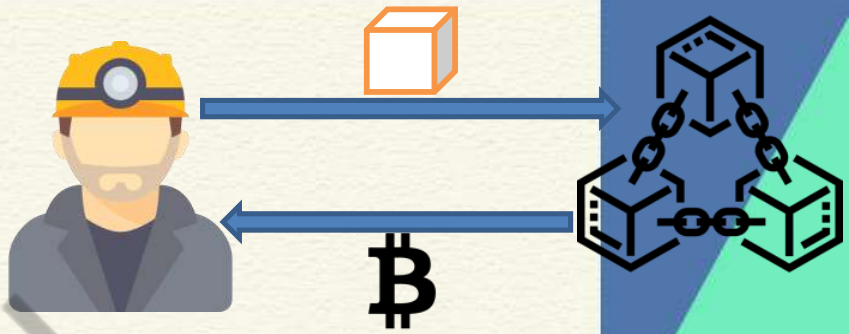
The Economics behind Reward



Encourage the community to participate in the mining through incentivization

Produces new Bitcoins in the System
(Similar to a Minting new Coins)

The Economics behind Reward



The Bitcoin network works like a Reserve Bank to regulate the flow of Money (Bitcoin) in the market, but without explicit governance

Blockchain 1.0: Distributed Ledger

- Use of the **Distributed Ledger Technology** (DLT) to design the "Money of the Internet" -- Bitcoin and other cryptocurrencies

Blockchain 1.0: Distributed Ledger

- Use of the **Distributed Ledger Technology** (DLT) to design the "Money of the Internet" -- Bitcoin and other cryptocurrencies
- 3rd January 2009: Nakamoto mined the first block of the Bitcoin network (called the genesis block)
  - 2013: Coinbase reported selling US$1 Million worth of Bitcoin
- Many other cryptocurrencies have been evolved after that
  - Ethereum
  - Litecoin
  - ...

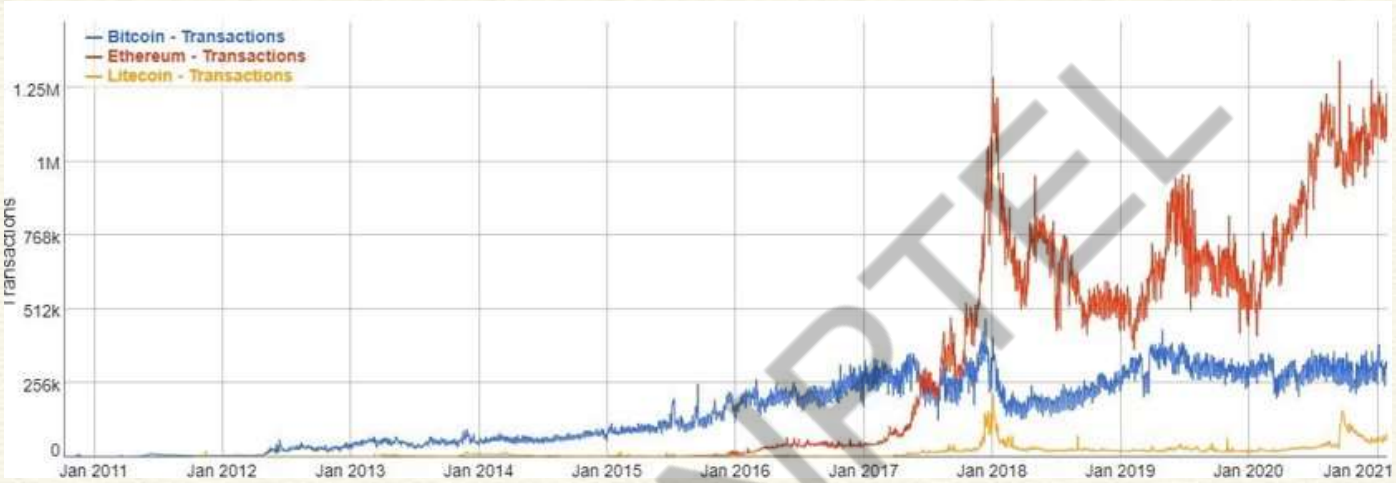
The Price of Bitcoins

- Bitcoin value increased drastically over time
  - May 2010: < $0.01
  - April 2014: $340 - $530
  - December 2017: ~$13800

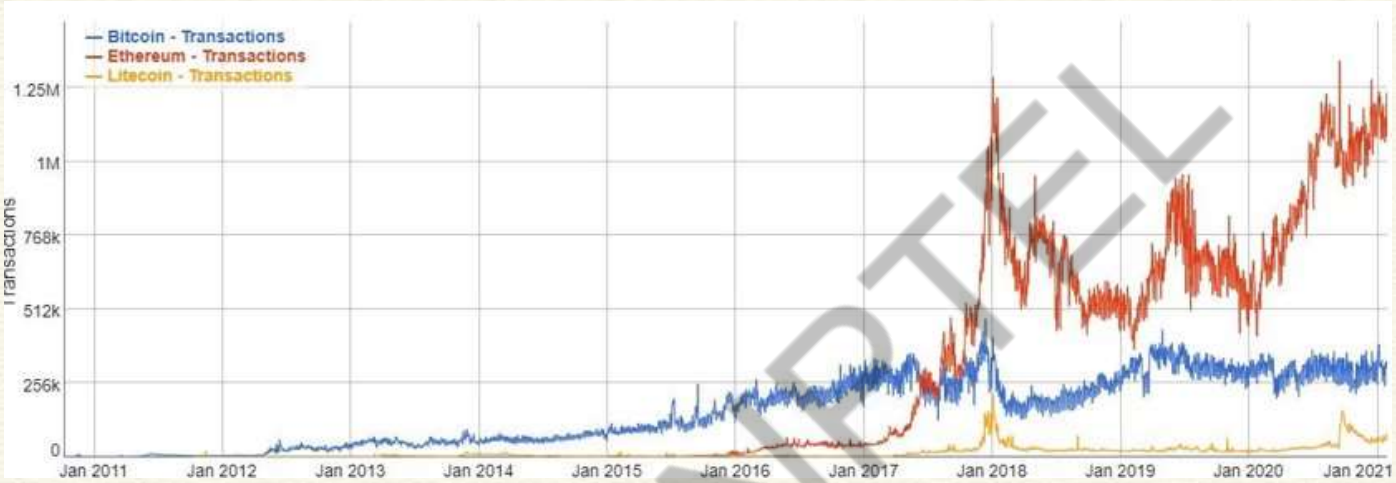
  - Today (24 Sept, 2021 10:30 am): **$44,285.50**

Bitcoin Millionaires

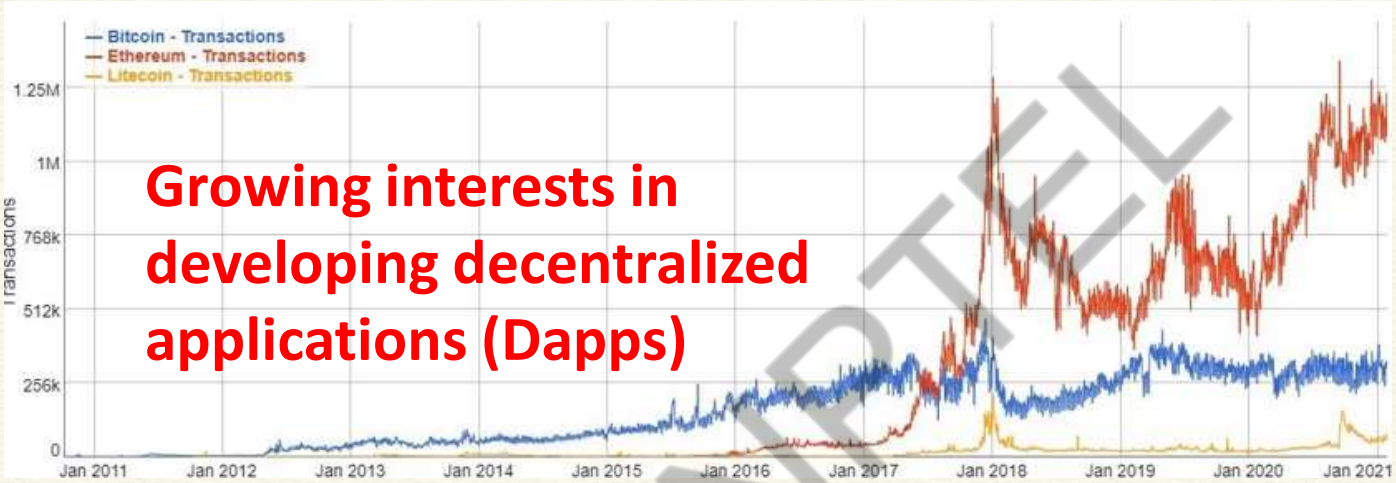
Popularity of Cryptocurrencies



Popularity of Cryptocurrencies



Popularity of Cryptocurrencies



Growing interests in developing decentralized applications (Dapps)

What are these Dapps?

- DLTs can contain information beyond financial transactions
  - What about submitting an executable code as a transaction?

- Not a new idea, people already knows about **Remote Code Execution** or **Code Injection**

What are these Dapps?

- DLTs can contain information beyond financial transactions
  - What about submitting an executable code as a transaction?

- **Transaction gets executed ==** Your code is getting executed
  - And you have consensus on the code execution !

What are these Dapps?

Smart Contacts

Will see them in the next lecture!

NPTEL ONLINE CERTIFICATION COURSES

Blockchain and its applications

Prof. Sandip Chakraborty
Department of Computer Science & Engineering
Indian Institute of Technology Kharagpur

Lecture 12: Smart Contracts and the Permissioned Models of Blockchain

CONCEPTS COVERED

- Smart Contracts and automated code execution
- Permissioned Blockchain

KEYWORDS

- Smart Contracts
- Blockchain 2.0 and 3.0
- Permissioned Models
- Enterprise Blockchains

DLTs for Code Execution

- DLTs can contain information beyond financial transactions
  - What about submitting an executable code as a transaction?

- **Transaction gets executed ==** Your code is getting executed
  - And you have consensus on the code execution !

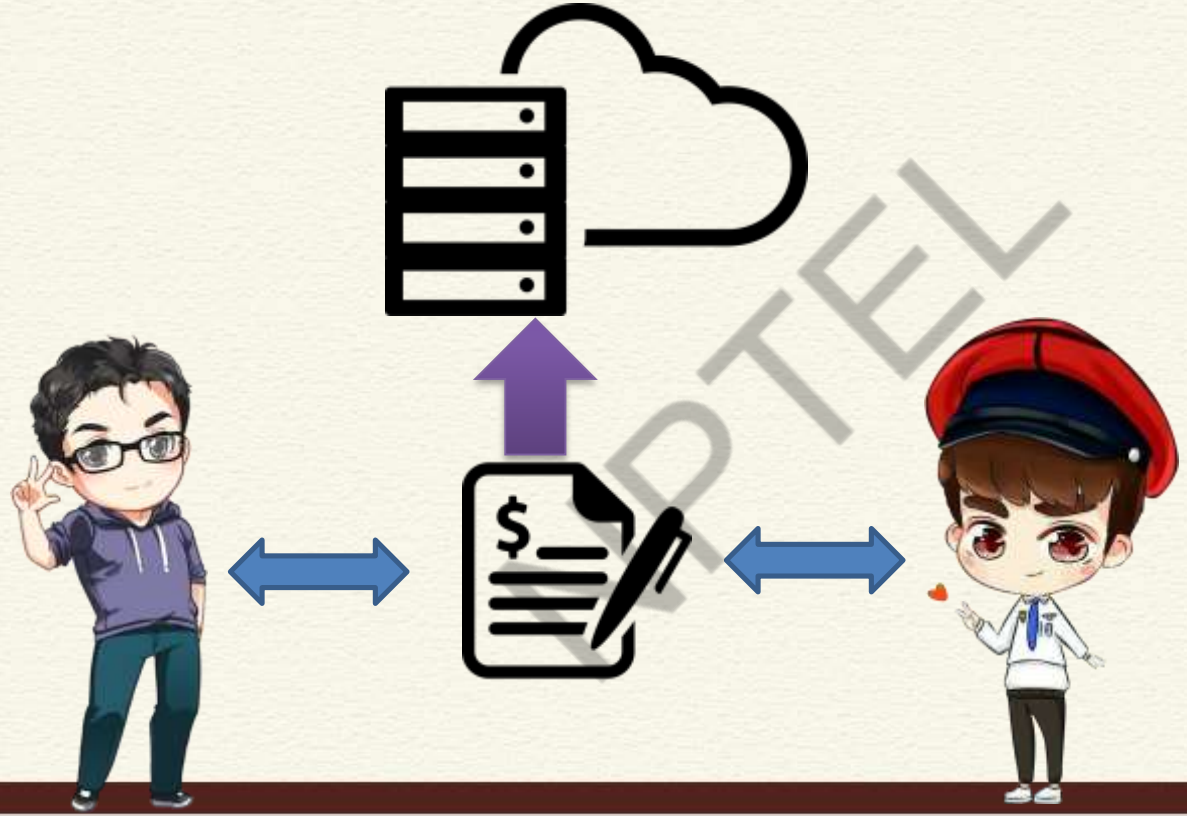
Smart Contracts

Smart Contacts

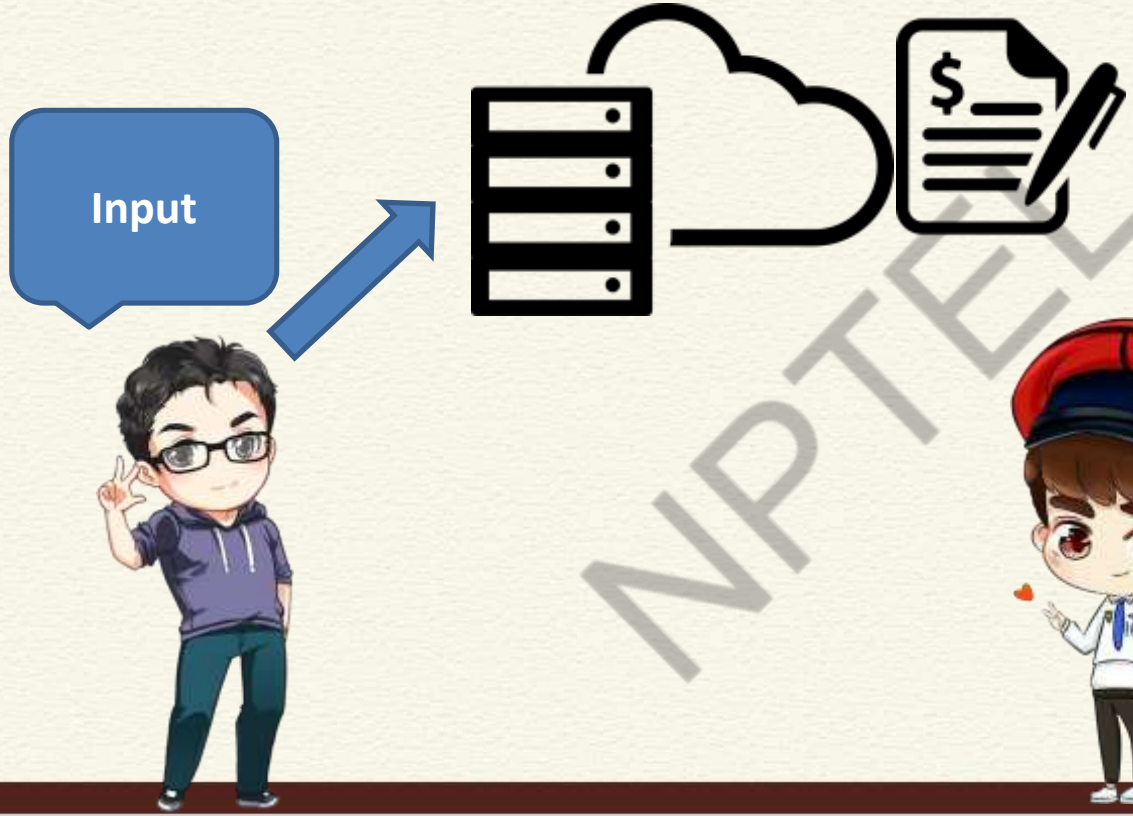
Automatically Execute Code over a Decentralized Platform

Traditional Way of Maintaining Digital Contacts

Traditional Way of Maintaining Digital Contacts

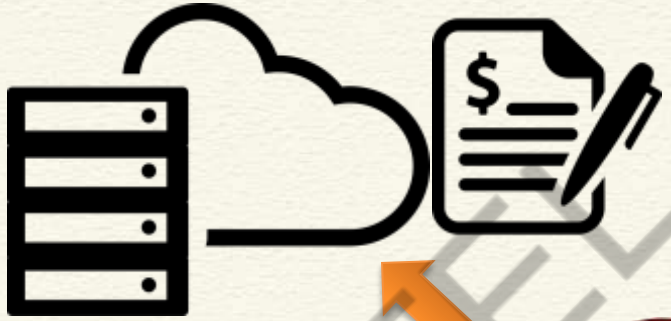


Traditional Way of Maintaining Digital Contacts



Traditional Way of Maintaining Digital Contacts

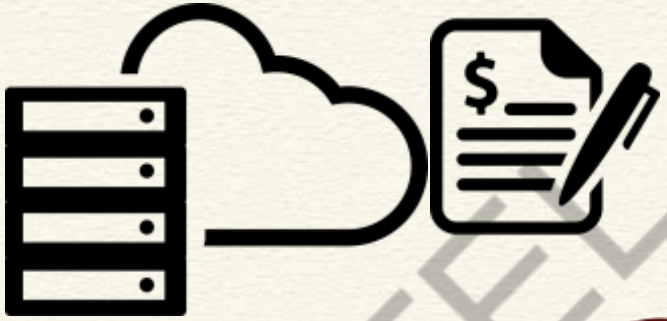
Traditional Way of Maintaining Digital Contacts



Input

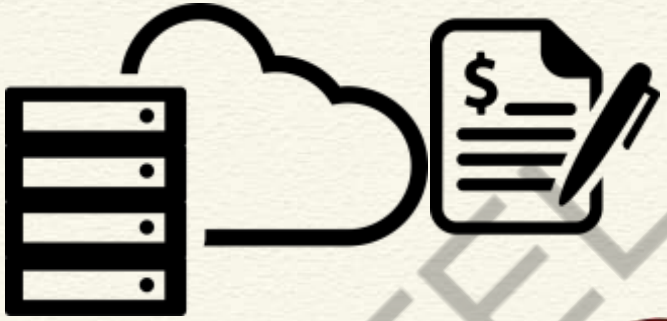
Traditional Way of Maintaining Digital Contacts

Traditional Way of Maintaining Digital Contacts



How will you check that the inputs are valid?

Traditional Way of Maintaining Digital Contacts



What if I deny about an input later on?

Decentralized Code Execution

```
int pay (float *sndAcc, float *rcvAcc, float amount) {
    if (*sndAcc < amount) return -1;
    else {
        *sndAcc -= amount;
        *rcvAcc += amount;
        return 1;
    }
}

int deliverGoods (int count, int pricePerC) {
    int success = pay (sender, receiver, count*pricePerC);
    if(success == 1) {
       sceduleLogistics();
       return 1;
    }
    Return 0;
}
```

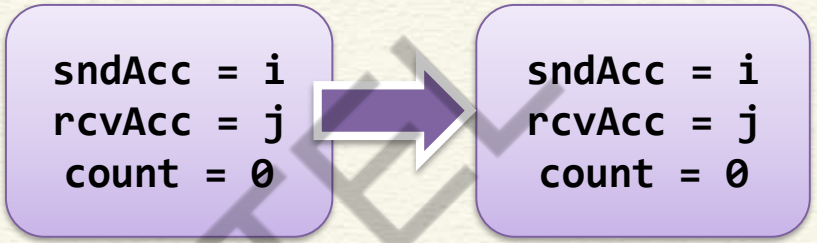
Decentralized Code Execution

```
int pay (float *sndAcc, float *rcvAcc, float amount) {
    if (*sndAcc < amount) return -1;
    else {
        *sndAcc -= amount;
        *rcvAcc += amount;
        return 1;
    }
}

int deliverGoods (int count, int pricePerC) {
    int success = pay (sender, receiver, count*pricePerC);
    if(success == 1) {
       sceduleLogistics();
       return 1;
    }
    Return 0;
}
```

sndAcc = i
rcvAcc = j
count = 0

Decentralized Code Execution

```
int pay (float *sndAcc, float *rcvAcc, float amount) {
    if (*sndAcc < amount) return -1;
    else {
        *sndAcc -= amount;
        *rcvAcc += amount;
        return 1;
    }
}

int deliverGoods (int count, int pricePerC) {
    int success = pay (sender, receiver, count*pricePerC);
    if(success == 1) {
       sceduleLogistics();
       return 1;
    }
    Return 0;
}
```

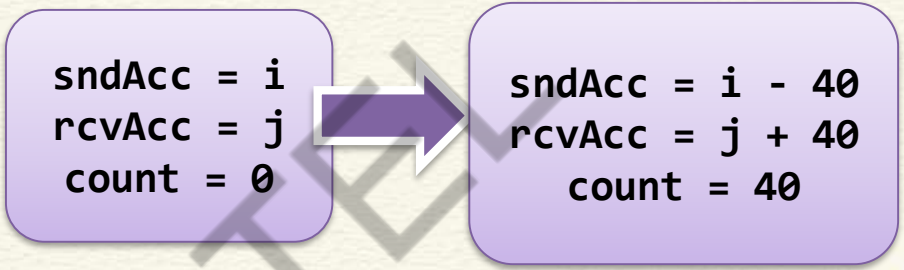


deliverGoods (10, 4)

Decentralized Code Execution

```
int pay (float *sndAcc, float *rcvAcc, float amount) {
    if (*sndAcc < amount) return -1;
    else {
         *sndAcc -= amount;
         *rcvAcc += amount;
         return 1;
    }
}

int deliverGoods (int count, int pricePerC) {
    int success = pay (sender, receiver, count*pricePerC);
    if(success == 1) {
       sceduleLogistics();
       return 1;
    }
    Return 0;
}
```

deliverGoods (10, 4)
pay(sndAcc, rcvAcc, 40) > 1



Decentralized Code Execution

```
int pay (float *sndAcc, float *rcvAcc, float amount) {
    if (*sndAcc < amount) return -1;
    else {
         *sndAcc -= amount;
         *rcvAcc += amount;
         return 1;
    }
}

int deliverGoods (int count, int pri[illegible] {
    int success = pay (sender, receive[illegible]ount*pricePerC);
    if(success == 1) {
       sceduleLogistics();
       return 1;
    }
    Return 0;
}
```

```
deliverGoods (10, 4)
pay(sndAcc, rcvAcc, 40) > 1
```

```
sndAcc = i
rcvAcc = j
count = 0
```

```
sndAcc = i - 40
rcvAcc = j + 40
count = 40
```

Put the states of execution in a blockchain

Smart Contract Execution

Smart Contract Execution

Submit the anonymized (through public key encryption) contract to a blockchain network

Smart Contract Execution

Everyone in the network can see and validate the execution steps

Cryptokitties – A Popular Game on Ethereum

Cryptokitties – A Popular Game on Ethereum

Permissioned Model of Blockchain

- PoW (Nakamoto Consensus) works good in an open network
  - But, transaction latency is very high
  - ~10 minutes in Bitcoin block commitment
  - Few seconds to few minutes for Ethereum (depending on the cost that you pay)

Permissioned Model of Blockchain

- PoW (Nakamoto Consensus) works good in an open network
  - But, transaction latency is very high
  - ~10 minutes in Bitcoin block commitment
  - Few seconds to few minutes for Ethereum (depending on the cost that you pay)

- **Can we think of any other Blockchain applications beyond cryptocurrency?**
  - The high latency makes them unsuitable for most of the real-time applications

Permissioned Model of Blockchain

- **Many decentralized applications do not demand an open environment**
  - The food supply chain
  - Know Your Customer (KYC)
  - Trade financing
  - ...

Blockchain 3.0

- "**Trustless Decentralization**" over a closed network
  - Automatically transact assets among multiple organizations who do not trust each other
  - Run smart contracts within a consortium of various organizations – the individual organizations know each other but do not trust each other

Blockchain 3.0

- **Advantages:**
  - Go back to the classical distributed consensus protocols – low latency for commitment and high transaction throughput
  - Use "Witness Cosigning" instead of "Proof Mining" for new block generation
    - Classical Distributed Consensus + Digital Signature

Permissioned Blockchain

- The participants are pre-authenticated and pre-authorized
  - But they can still behave maliciously

- Run blockchain (and smart contracts) on top of this closed network
  - Ensure trusted computing among the participants

Conclusion

- Smart Contracts revolutionizes the blockchain/DLT applications
  - Supports automated code execution over a decentralized platform

- Permissioned blockchains have emerged for enterprise applications

Conclusion

- Smart Contracts revolutionizes the blockchain/DLT applications
  - Supports automated code execution over a decentralized platform
- Permissioned blockchains have emerged for enterprise applications
- **Now let us explore the detailed internal structure of a blockchain …. our next lecture**

NPTEL ONLINE CERTIFICATION COURSES

Blockchain and its applications

Prof. Shamik Sural
Department of Computer Science & Engineering
Indian Institute of Technology Kharagpur

Lecture 08: Blockchain Elements - I

CONCEPTS COVERED

- What is a Blockchain
- Blocks in a Blockchain
- Block Header

KEYWORDS

- Block Structure
- Block Header
- Mining a Block
- Block Generation Puzzle

What is Blockchain?

- A Platform for executing transactional services
- Spanned over multiple organizations or individuals who may not (need not) **trust** each other
- An append-only shared ledger of digitally signed and encrypted transactions replicated across a network of peer nodes

The Block in a Blockchain – Securing Data Cryptographically

- Digitally signed and encrypted transactions **"verified" by peers**
- **Cryptographic security** – Ensures that participants can only view information on the ledger that they are authorized to see

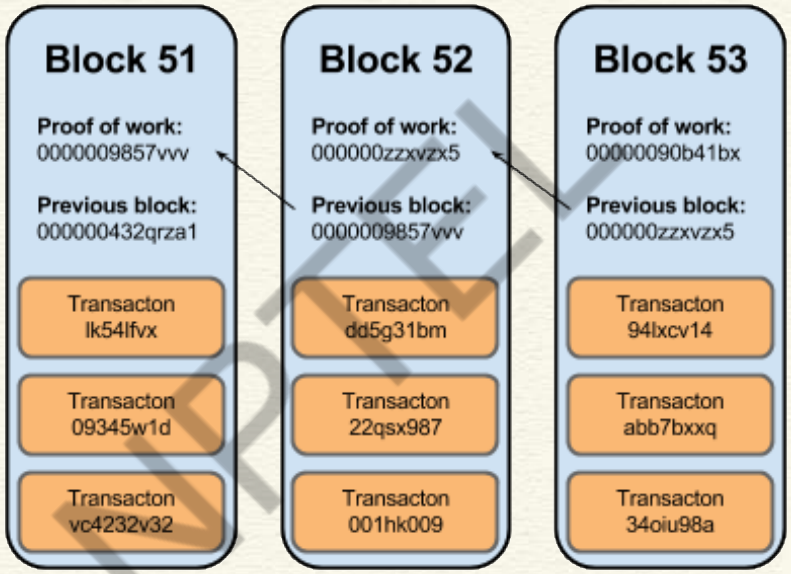


Image source: http://dataconomy.com/

Structure of a Block

- A block is a **container data structure** that contains a series of transactions
- **In Bitcoin:** A block may contain more than 500 transactions on average, the average size of a block is around 1 MB (an upper bound proposed by Satoshi Nakamoto in 2010)
    - May grow up to 8 MB or sometime higher (several conflicting views on this!!)
    - Larger blocks can help in processing large number of transactions in one go.
    - But longer time for verification and propagation

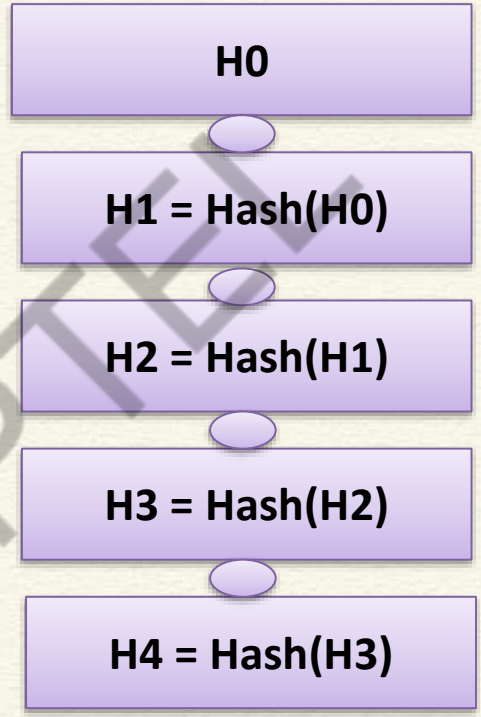
Structure of a Block (Reference: Bitcoin)

- Two components:
  - **Block Header**
  - **List of Transactions**

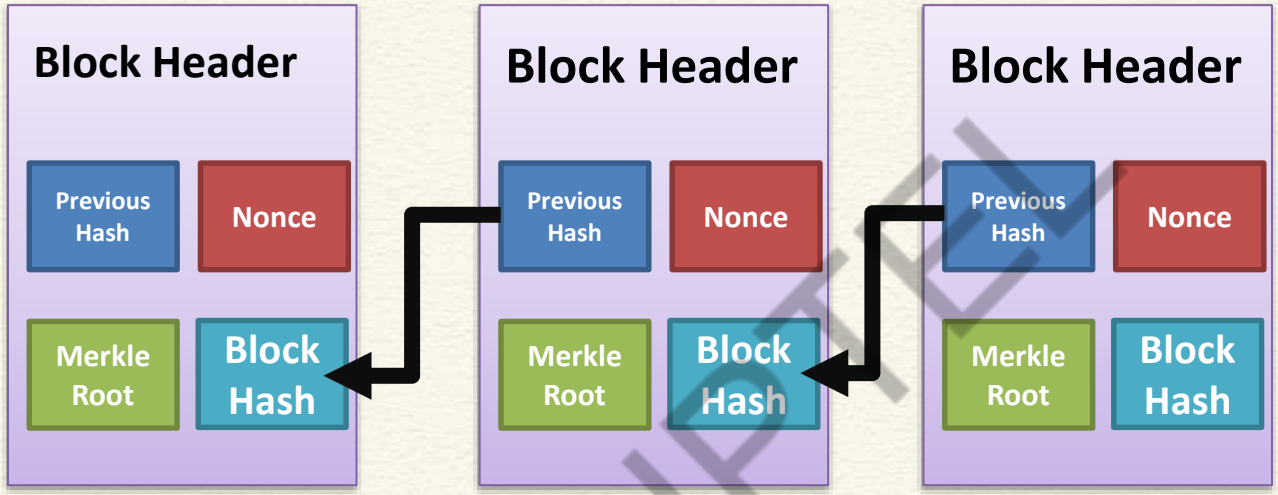
Block Source: https://btc.com/btc/blocks OR https://blockchain.com/explorer

Block Header (Reference: Bitcoin)

- Metadata about a block – (1) Previous block hash, (2) Mining statistics used to construct the block, (3) Merkle tree root
- **Previous block hash:** Every block inherits from the previous block – we use previous block's hash to create the new block's hash – make the blockchain **tamper proof**



Block Generation Puzzle



Find out the nonce which generates the desired hash (certain number of zero bits at the prefix) -

00000000000000004a2b84f93a285b7a7……..

Block Header (Reference: Bitcoin)

- **Mining** – the mechanism to generate the hash
  - The mechanism needs to be complicated enough, to make the blockchain **tamper proof**
  - **Bitcoin Mining:** $H_k = Hash(H_{k-1} \,||\, T \,||\, Nonce \,||$ *Something more*$)$
  - Find the nonce such that H_k has certain predefined **complexity** (number of zeros at the prefix)
- The header contains mining statistics – timestamp, nonce and difficulty

Block Header (Reference: Bitcoin)

- Understanding Difficulty and Bits
- “Bits” written in Hex, e.g., 0x170e2632
  - First byte is index and next three bytes form coefficient
  - Target = Coefficient*2^(8*(index-3))
- Difficulty is the largest possible target (0x00000000FFFF0000000000000000000000000000000000000000000000000000) divided by the current target , e.g.,
- (0x0000000000000000000E2631FFFFFFFFFFFFFFFFFFFFBB0C4B021913E000000)
- Remember: “Cost of Mining” – Pretty High (Computing Power and Energy)

[Number conversion utility: https://www.rapidtables.com/convert/number/hex-to-decimal.html]

Hashes in a Block Header (Reference: Bitcoin)

- Block identifier – the hash of the current block header (Hash algorithm: Double SHA256)
- Merkle Root
- Previous block hash is used to compute the current block hash
- Timestamp, Previous hash, Merkle root, Difficulty Bits, Nonce and Version used to compute current hash

Demonstration

https://dlt-repo.net/bitcoin-block-hash-verification-tool/

Block Source: https://btc.com/btc/blocks

CONCLUSIONS

- We have described the structure of a block in blockchain
- Main components of a block header
- How to solve block generation puzzle
- What is meant by mining of a block

REFERENCES

- Cryptography and Network Security – Principles and Practice by William Stallings, Pearson (2017)
- Blockchain Basics: A Non-Technical Introduction in 25 Steps by Daniel Drescher, Apress (2017)
- Any other standard textbook on blockchain/bitcoin

NPTEL ONLINE CERTIFICATION COURSES

Blockchain and its applications

Prof. Shamik Sural
Department of Computer Science & Engineering
Indian Institute of Technology Kharagpur

Lecture 14: Blockchain Elements - II

CONCEPTS COVERED

- Block Generation Cost
- Transactions in a Block
- Bitcoin Scripts

KEYWORDS

- Hash Generation Rate
- Transaction Input and Output
- Bitcoin Script

Block Generation Cost

- Energy efficiency ~0.098 J/GH = ~100 J/TH
- ASIC Hardware for bitcoin can perform about 750 TH/s
- Hash rate approx. 120M TH/s!! Many actually go waste ☹
- Network consumes about 80 TW-hours of electricity annually. Figures vary between sources and are some form of estimates
- Average household in Germany of four people consumes approx. 4,000 KW-hours of electricity per year.
- Can power about 20,000 households
- Concept of Pooling is used (https://btc.com/)
- What ensures tamperproof operation in terms of honest nodes??

Blockchain Replicas

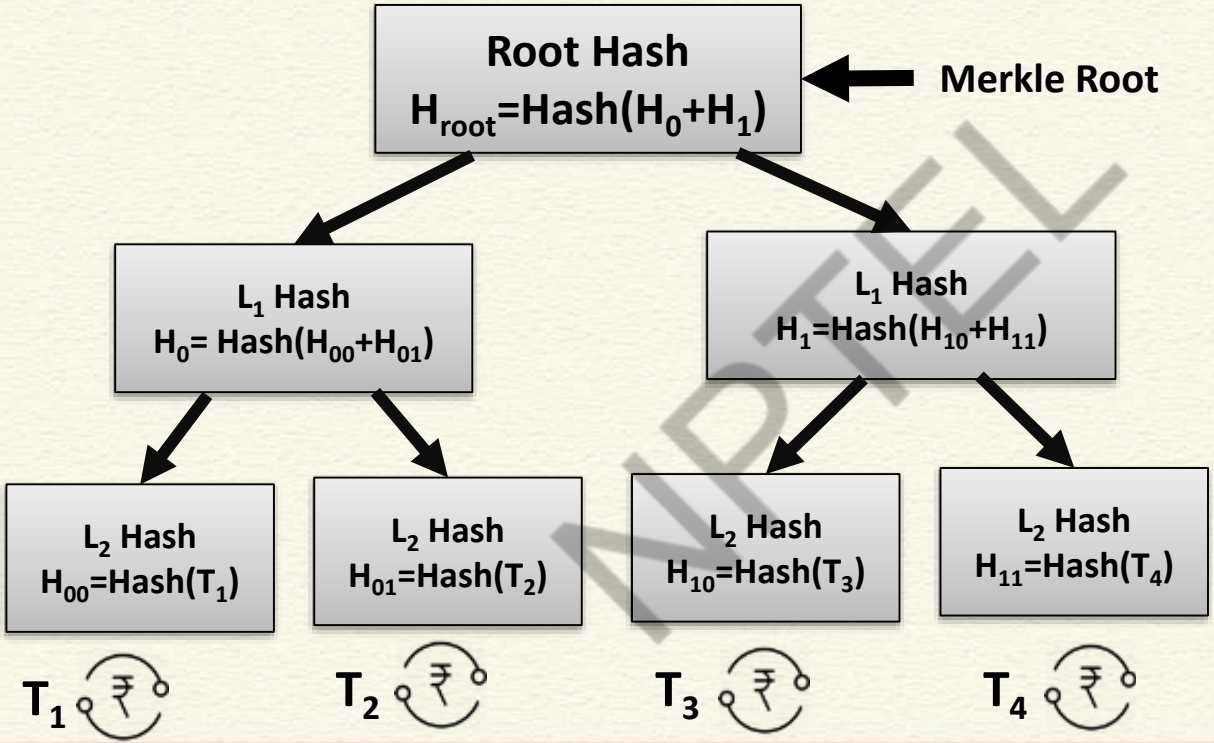
- Every peer in a Blockchain network maintains a local copy of the Blockchain.
- Size is just about 351 GB ☺
- As a new user joins the network, she can get the whole copy

- **Requirements**
  - All the replicas need to be **updated** with the last mined block
  - All the replicas need to be **consistent** – the copies of the Blockchain at different peers need to be **exactly similar**

Transactions in a Block

- Transactions are organized as a Merkle Tree. The Merkle Root is used to construct the block hash
- If you change a transaction, you need to change all the subsequent block hashes
- The **difficulty** of the mining algorithm determines the **toughness** of tampering with a block in a blockchain

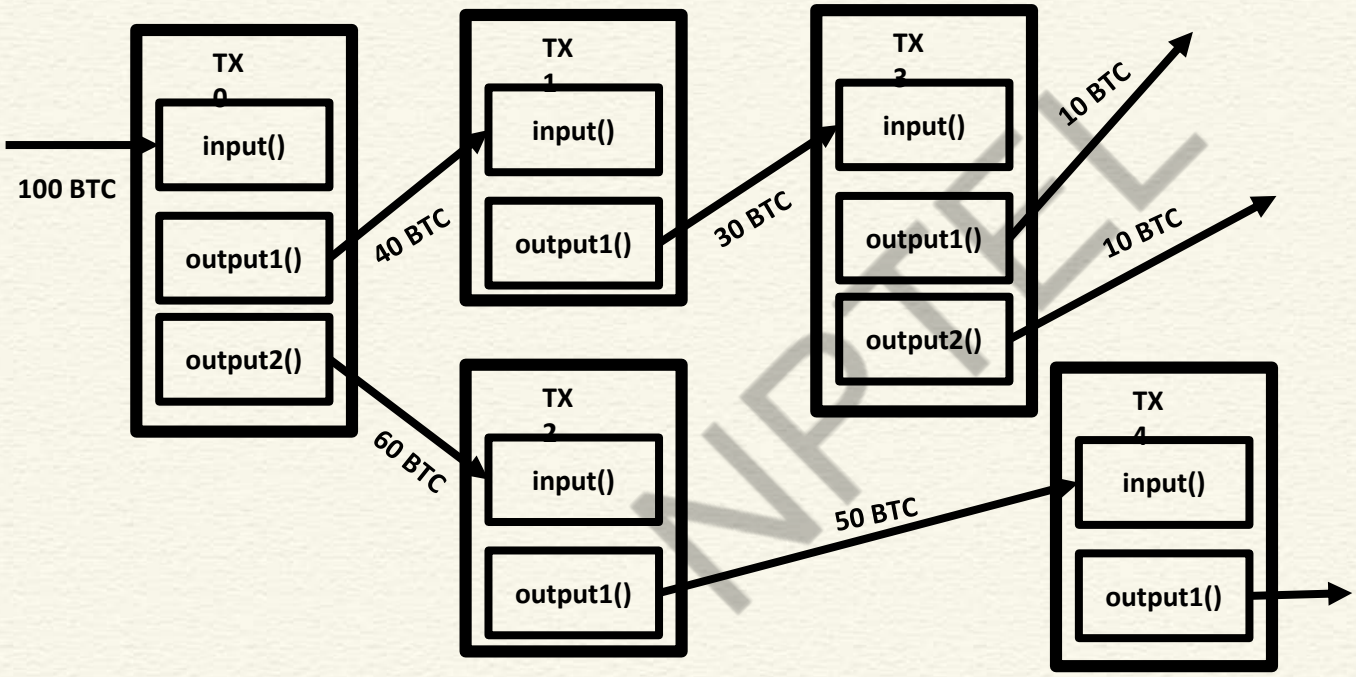
Merkle Tree – A Quick Recap



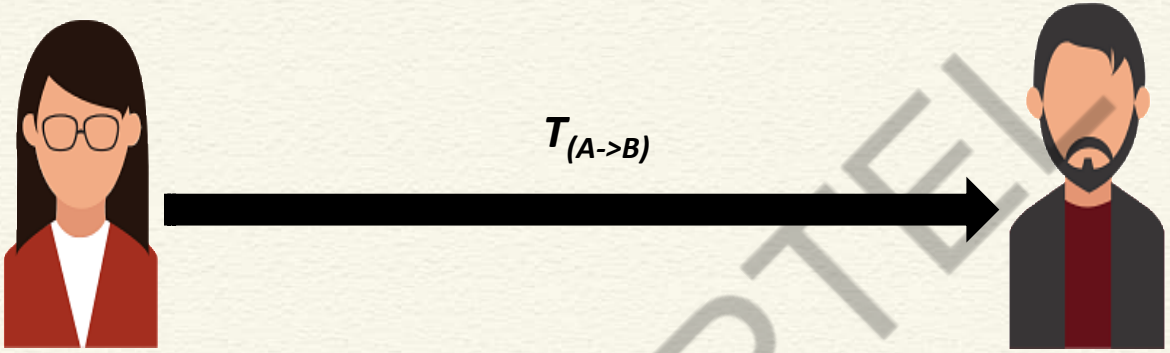
Transactions in a Block

Block Source: https://btc.com/btc/blocks

Bitcoin Transactions and Input and Output

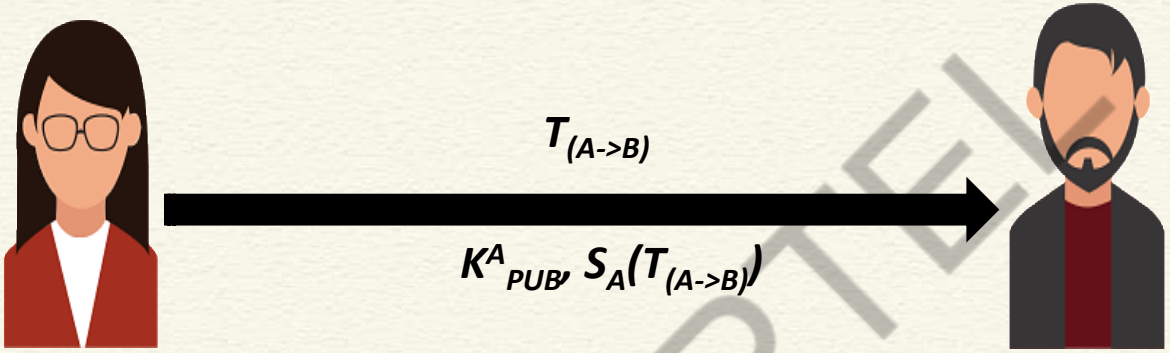


Bitcoin Scripts – A Simple Example



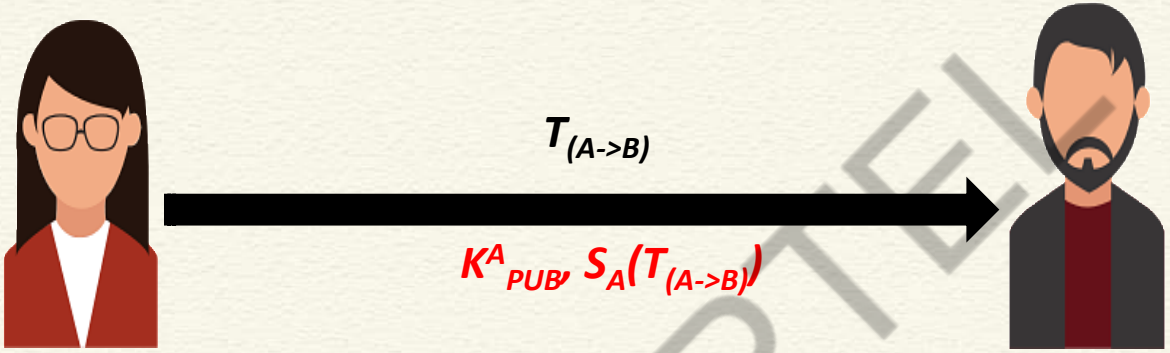
How Bob will verify that the transaction is actually originated from Alice?

Bitcoin Scripts – A Simple Example



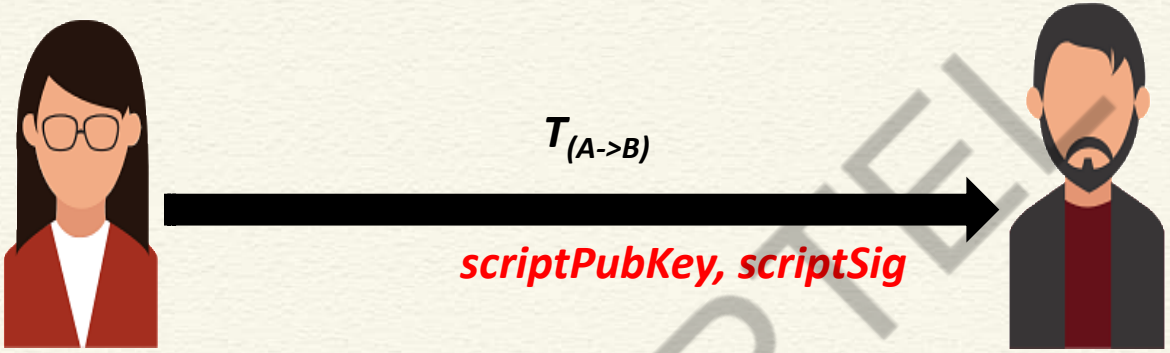
Send the public key of Alice along with the signature -> Bob can verify this

Bitcoin Scripts – A Simple Example



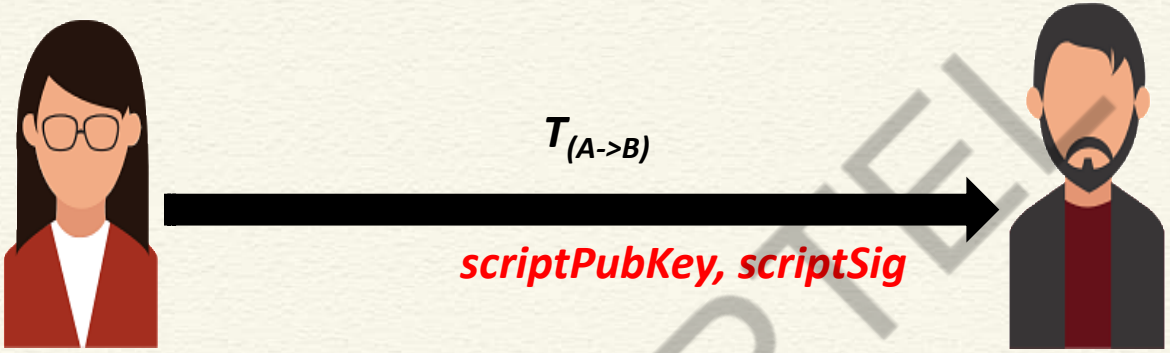
Bitcoin indeed transfers scripts instead of the signature and the public key

Bitcoin Scripts – A Simple Example



Bitcoin indeed transfers scripts instead of the signature and the public key

Bitcoin Scripts – A Simple Example



Bob can spend the bitcoins only if both the scripts return `TRUE` after execution

Bitcoin Scripts

- Simple, compact, stack-based and processed left to right
  - FORTH like language

- **Not Turing Complete** (no loops)
  - Halting problem is not there

Bitcoin Scripts

- With every transaction Bob must provide
  - A public key that, when hashed, yields the address of Bob embedded in the script
  - A signature to provide ownership of the private key corresponding to the public key of Bob

CONCLUSIONS

- Discussed the cost of block generation
- How transactions are included in blocks
- Use of scripts for making and claiming payments

REFERENCES

- Blockchain Basics: A Non-Technical Introduction in 25 Steps by Daniel Drescher, Apress (2017)
- Any other standard textbook on blockchain/bitcoin

NPTEL ONLINE CERTIFICATION COURSES

Blockchain and its applications

Prof. Shamik Sural
Department of Computer Science & Engineering
Indian Institute of Technology Kharagpur

Lecture 15: Blockchain Elements - III

CONCEPTS COVERED

- Understanding Bitcoin Scripts
- Some Interesting Bitcoin Scripts

KEYWORDS

- Bitcoing Script
- scriptPubKey
- scriptSig
- Stack

Bitcoin Scripts

Transaction Input → **scriptSig:**

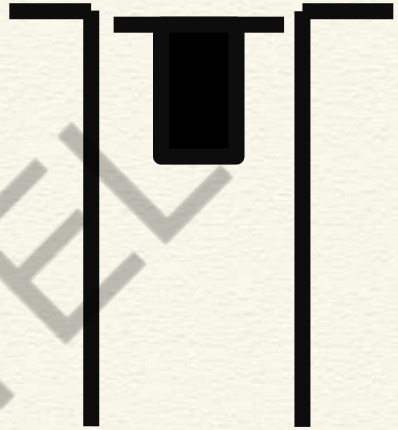
```
18E14A7B6A30…
D61967F63C7DD…
```

Transaction Output → **scriptPubKey:**

```
OP_DUP
OP_HASH160
16UwLL9Risc3QfPqBUvKof…
OP_EQUALVERIFY
OP_CHECKSIG
```

See for detailed steps:
https://developer.bitcoin.org/devguide/transactions.html

Bitcoin Scripts



```
scriptPubKey: OP_DUP
OP_HASH160 <pubKeyHash>
OP_EQUALVERIFY
OP_CHECKSIG
```

```
scriptSig: <sig>
<pubKey>
```

The stack is initially empty. Both the scripts are combined – input followed by output

```
<sig> <pubKey> OP_DUP
OP_HASH160 <pubKeyHash>
OP_EQUALVERIFY OP_CHECKSIG
```

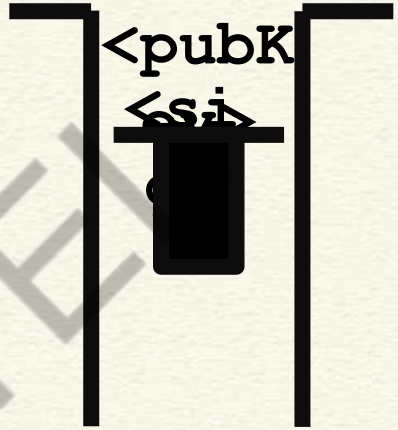
A real example from Bitcoin

For more examples to explore: https://btc.com/btc/blocks

Bitcoin Scripts

<sig> <pubKey> OP_DUP OP_HASH160 <pubKeyHash> OP_EQUALVERIFY OP_CHECKSIG

The top two items are pushed to Stack one after another



OP_DUP OP_HASH160 <pubKeyHash> OP_EQUALVERIFY OP_CHECKSIG

Bitcoin Scripts

```
OP_DUP OP_HASH160
<pubKeyHash> OP_EQUALVERIFY
OP_CHECKSIG
```

Top stack item is duplicated


```
OP_HASH160 <pubKeyHash>
OP_EQUALVERIFY OP_CHECKSIG
```

Bitcoin Scripts

```
OP_HASH160 <pubKeyHash>
OP_EQUALVERIFY OP_CHECKSIG
```

Top stack item is hashed (RIPEMD-160 hashing)

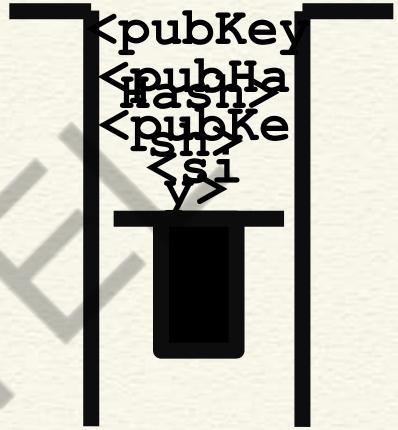

```
<pubKeyHash> OP_EQUALVERIFY
OP_CHECKSIG
```

Bitcoin Scripts

<pubKeyHash>

OP_EQUALVERIFY OP_CHECKSIG

The constant is pushed in the stack



OP_EQUALVERIFY OP_CHECKSIG

Bitcoin Scripts

OP_EQUALVERIFY OP_CHECKSIG

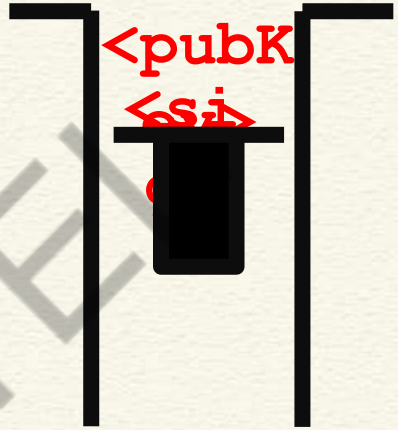
Equality is checked between the top two items in the stack

OP_CHECKSIG

Bitcoin Scripts

OP_CHECKSIG

Signature is checked based on the top two stack item



TRUE

Bitcoin Script Instructions

- Total 256 opcodes (15 disabled, 75 reserved)
  - Arithmetic operations
  - if-then conditions
  - Logical operators
  - Data handling (like OP_DUP)
  - Cryptographic operations
    - Hash functions
    - Signature verification
    - Multi-signature verification

Interesting Bitcoin Scripts

- **Provably un-spendable or prunable outputs**

```
scriptPubKey: OP_RETURN
{zero or more ops}
```

- **Anyone-can-spend outputs**

```
scriptPubKey: {empty}
scriptSig: OP_TRUE
```

Source: https://en.bitcoin.it/wiki/Script

Interesting Bitcoin Scripts

- **Freezing funds until a time in the future**

```
scriptPubKey: <expiry_time>
OP_CHECKLOCKTIMEVERIFY OP_DROP
OP_DUP OP_HASH160 <pubKeyHash>
OP_EQUALVERIFY OP_CHECKSIG
scriptSig: <sig> <pubKey>
```

Source: https://en.bitcoin.it/wiki/Script

CONCLUSIONS

- Use of scripts in generating input and output of bitcoin transactions
- Public key cryptography and digital signature for cryptographically protecting transactions

REFERENCES

- Blockchain Basics: A Non-Technical Introduction in 25 Steps by Daniel Drescher, Apress (2017)
- Any other standard textbook on blockchain/bitcoin