PLATINUM SPONSOR

GiG
GAMING INNOVATION GROUP

SILVER SPONSOR

playtech
SOURCE OF SUCCESS

EVENT SPONSOR

colorworks

VENUE PARTNER

sunborn
GIBRALTAR
5-STAR YACHT HOTEL

Marcin Zduniak

Blockchain Architect, Ethereum, Corda and Bitcoin Consultant

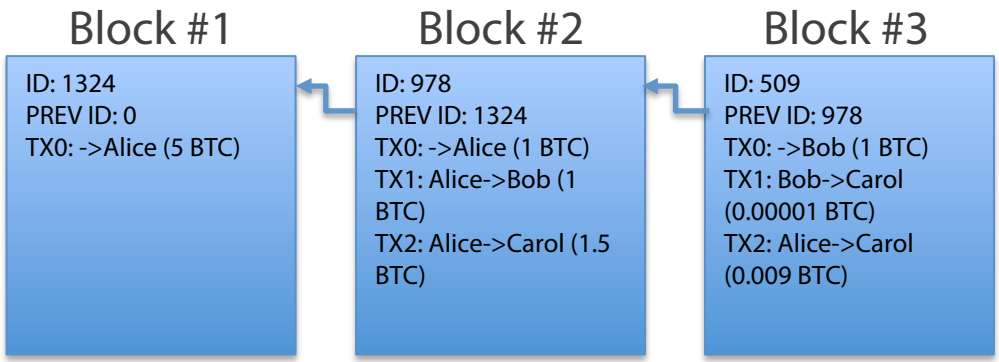
Girls in Tech
GIBRALTAR

Blockchain -> No need for trust

- It is all about removing reliance on any 3rd party trust providers.
- Parties can finally interact directly (P2P) with each other, trustlessly, without any proxies or indirect relationship.

Girls in Tech
GIBRALTAR

Blockchain/Ledger - Intuition



- Block contains transactions
- Block refers to its predecessor (preserving chronology)
- 1st transaction in every block creates Bitcoins “out of thin air”

Girls in Tech GIBRALTAR

Distributed Ledger - Intuition

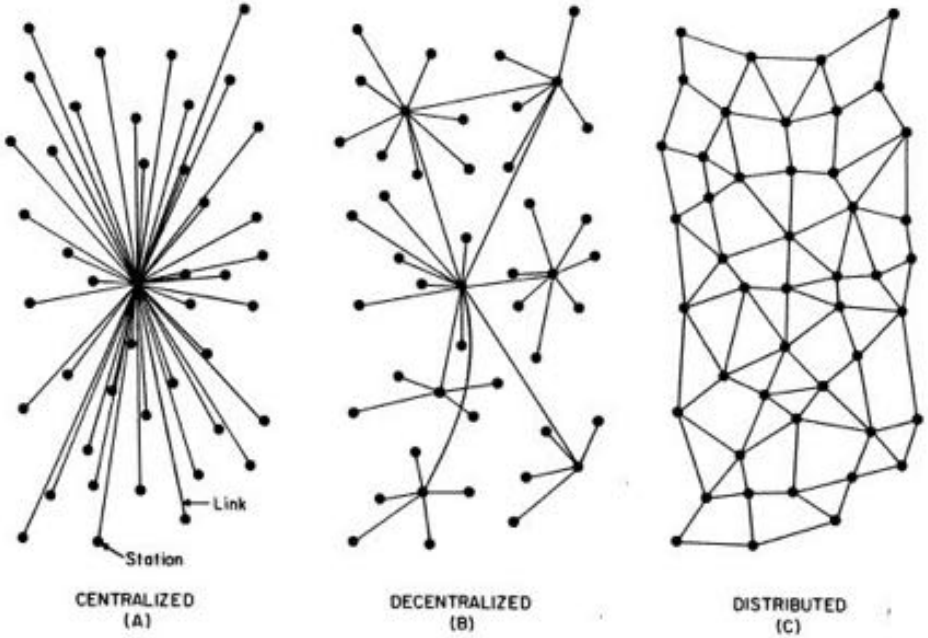


FIG. 1 — Centralized, Decentralized and Distributed Networks

- Centralized = single provider (Google, Facebook)
- Decentralized = some nodes still connected to single privileged node (e.g. banking system, single point of failure)
- Distributed = every node equal and connected to 2+ other nodes (e.g. Bittorent, Bitcoin [8+])
- Same ledger/blockchain shared by all nodes

Girls in Tech GIBRALTAR

Thank you Jameson Lopp: http://goo.gl/dNmG3s

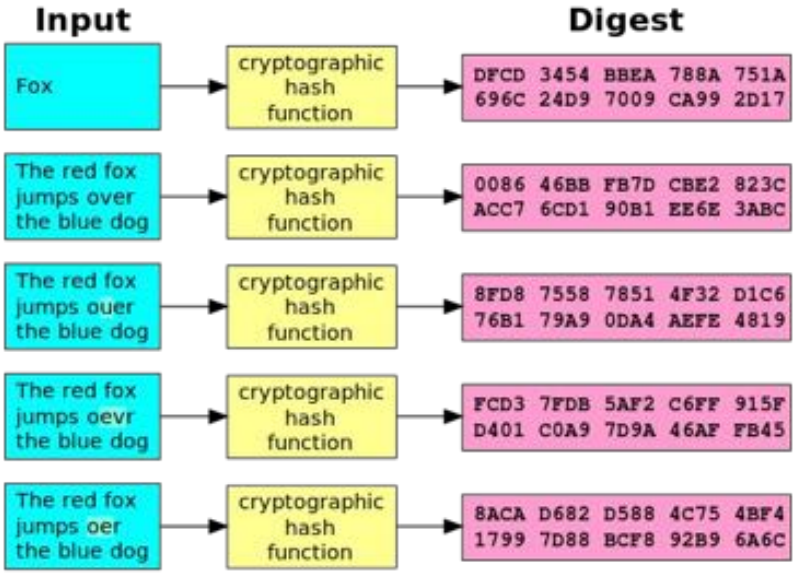
Cryptography Basics

In Bitcoin only two concepts

- Cryptographic hash functions (e.g. SHA-256, REPEMD-160, SHA3)
- Public Key Cryptography (ECDSA – Elliptic Curve Digital Signature Algorithm)

Girls in Tech
GIBRALTAR

Cryptographic Hash Functions



- Algo that maps data of arbitrary size to fixed-size bit strings
- One-way functions (infeasible to invert; only by brute-force search)
- Deterministic
- Quick to compute
- Small change in input message causes avalanche effect to the resulting hash value
- It is infeasible to find two different messages with the same hash value
- Bitcoin uses:
  SHA-256, REPEMD-160

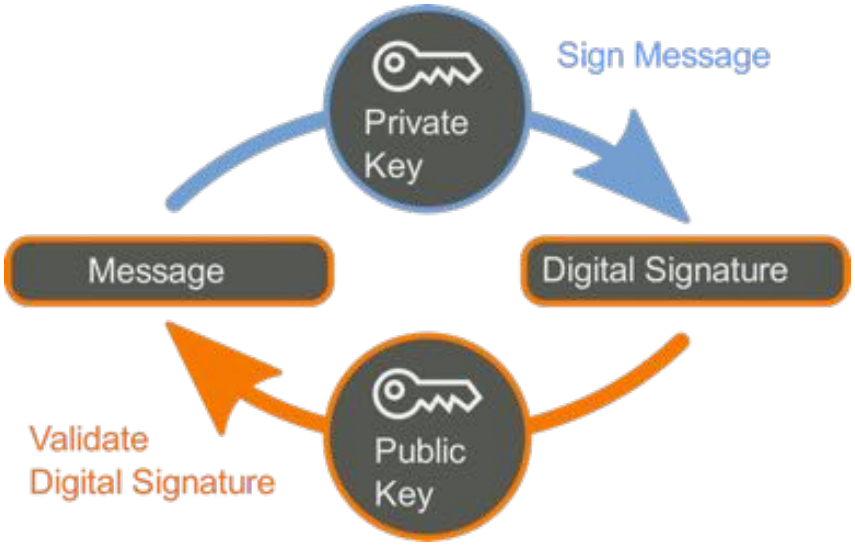
Thank you Wikipedia: https://en.wikipedia.org/wiki/Cryptographic_hash_function

Public Key Cryptography

- Generally used for encryption of the data and for signing/ authenticating the data
- In Bitcoin all data are unencrypted and publicly visible to everyone in the network
- In Bitcoin PKC is used to produce digital signatures to verify transactions integrity and authenticity

Girls in Tech
GIBRALTAR

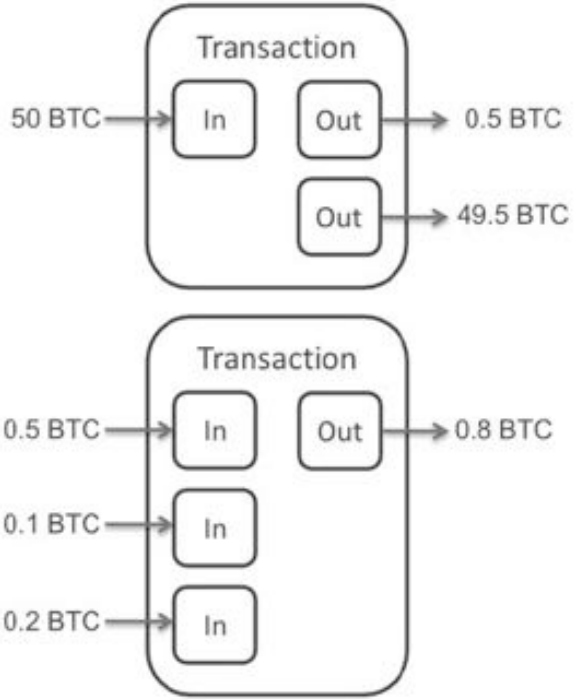
Public Key Cryptography



- Private key is effectively a very large random number with certain properties
- Public key can be derived from private key (but not other way round)
- Private key **(to be kept secret by the owner)** can be used for decryption and for signing messages
- Public key **(visible to everyone)** can be used for encryption and for validating signatures

Thank you Wikiversity: https://en.wikiversity.org/wiki/Digital_Signature

Bitcoin Transaction

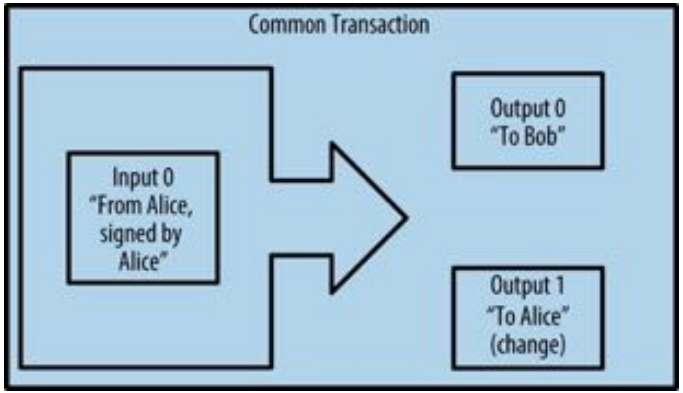


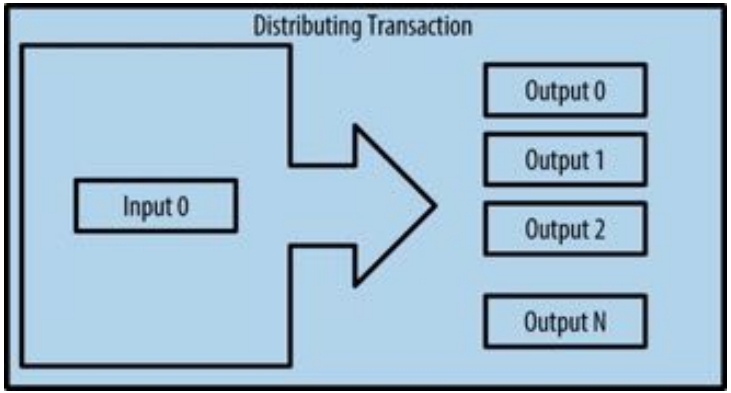
- Tells the network that the owner (has private key) of number of bitcoins authorized (by digitally signing) their transfer to another owner
- Outputs = tell how many bitcoins should belong to the new owner (specified by the Bitcoin address)
- Inputs = references to previously unspent outputs (UTXO)
- Tx contains 0+ inputs & 1+ outputs
- Tx mining fee: Sum(inputs) - Sum(outputs) > 0
- Coinbase tx has no inputs (creates “free Bitcoins”)

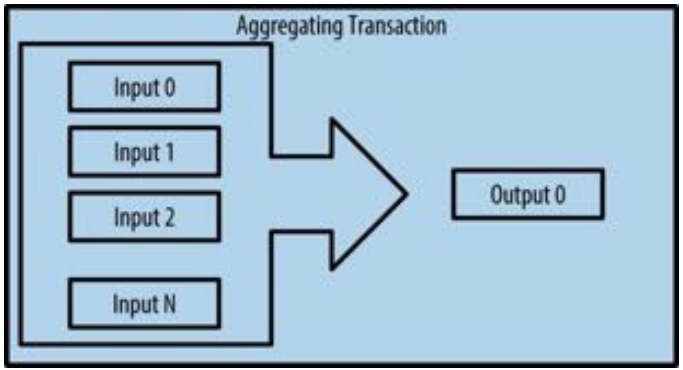
Girls in Tech GIBRALTAR

Thank you Wikipedia: https://de.wikipedia.org/wiki/Bitcoin

Bitcoin Common Transactions





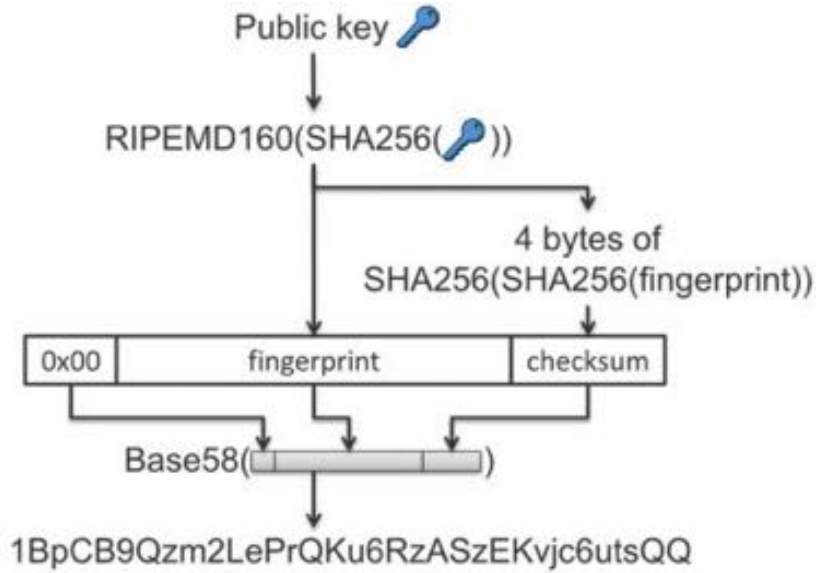


- Multi signature (M-of-N)
- Null data / Metadata
- Lock times
- Arbitrary script

Girls in Tech GIBRALTAR

Thank you Andreas M. Antonopoulos: https://github.com/bitcoinbook/bitcoinbook/blob/second_edition/ch02.asciidoc

Bitcoin Address



- Bitcoin transactions recipients (in Outputs) are represented by Bitcoin addresses
- Public key is transformed to address
- Checksum
- 58 characters (62 alphanumeric minus confusing 4: “O0Il”)
- Quantum computing resistant

Girls in Tech GIBRALTAR

Thank you Junbang Huang: https://github.com/JunbangHuang/blockchain/blob/master/doc/Keys-and-Addresses.md

Double-Spending Problem

- Transaction that spends some UTXO is effectively a file, can be copied and duplicated “creating” money that did not previously exist (”printing worthless money”)
- Bitcoin is the first distributed protocol that solves this problem without relaying on trusted third party to validate transactions

Girls in Tech
GIBRALTAR

Mining

- Process of bundling transactions together
- Valid block contains only valid transactions:
  - No double-spend transactions
  - Valid signatures of all spends
  - Proof of work
- To incentivize creation of valid blocks miners are allowed to claim “free Bitcoins” (now 12.5 BTC per block and diminishes with time) and collect all of the transactions mining fees (txs inputs minus outputs value)
- Invalid blocks are discarded by other miners and validating nodes

Girls in Tech
GIBRALTAR

Mining / Proof of Work

- Method of choosing (consensus) which of the nodes/miners will be privileged to claim “free Bitcoins” on average every 10 minutes
- CPU intensive (electricity costs are transformed into decentralized trust)
- Goal: generate SHA-256 hash from some txs-related data that would be prefixed with sufficient number of 0-s, e.g.:
  Block #506407 hash:
  0000000000000000003afdd7dbf2fb22e36d33f8b14ed4d0847f469bc62c26c1
- Difficulty (number of prefixing 0-s) increases/decreases as more electricity/CPU/ miners join/drop competition to
- If a solution is found to the Proof of Work problem, the new block is added to the local blockchain and broadcast to the rest of peer-to-peer network

Girls in Tech
GIBRALTAR

Ethereum vs Bitcoin

- Bitcoin: limited low-level non-Turing complete scripting language (no loops = guarantee it will stop)
- Ethereum: Turing complete smart contract language (calculations stop enforced by gas costs)
- Both public blockchains
- Both use Proof of Work (though different algorithms)
- Accounts (two types) vs UTXOs
- Ethereum transaction comprise of:
- Value (ETH)
- Data (new smart contract bytecode or invocation data for existing contract)
- Ethereum: state machine & state transitions

Girls in Tech
GIBRALTAR

Ethereum vs Bitcoin

Subcurrency Example

```
contract Coin {
    address minter;
    mapping (address => uint) balances;
    function Coin() {
        minter = msg.sender;
    }
    function mint(address owner, uint amount) {
        if (msg.sender != minter) return;
        balances[owner] += amount;
    }
    function send(address receiver, uint amount) {
        if (balances[msg.sender] < amount) return;
        balances[msg.sender] -= amount;
        balances[receiver] += amount;
    }
    function queryBalance(address addr) constant returns (uint balance) {
        return balances[addr];
    }
}
```

Standard Transaction to Bitcoin address (pay-to-pubkey-hash)

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

Bitcoin assembly script

Ethereum/Solidity

Girls in Tech GIBRALTAR

R3 Corda vs Bitcoin

- Corda: private/permissioned ledger (can have different data for each peer)
- Cluster of Notaries vs Proof of Work
- Corda: point-to-point direct communication between nodes (data shared only on a need-to-know basis) vs Bitcoin: global broadcast
- Corda: Contracts are written in a JVM programming language (e.g. Java or Kotlin)
- Corda: Contract execution is deterministic and its acceptance of a transaction is based on the transaction's contents alone (no access to outside world)
- Corda: Flows automate the process of agreeing ledger updates (and can be integrated with any IT system/database)
- Both uses UTXOs (in Corda UTXO = state/fact)

Girls in Tech
GIBRALTAR

R3 Corda – IOU Example

```
data class IOUState(val value: Int, val lender: Party, val borrower: Party):
        ContractState {
    override val participants: List<AbstractParty> get() = listOf(lender, borrower)
}

open class IOUContract : Contract {
    override fun verify(tx: LedgerTransaction) {
        val command = tx.commands.requireSingleCommand<Commands.Create>()
        requireThat { this: Requirements
            // Generic constraints around the IOU transaction.
            "No inputs should be consumed when issuing an IOU." using (tx.inputs.isEmpty())
            "Only one output state should be created." using (tx.outputs.size == 1)
            val out = tx.outputsOfType<IOUState>().single()
            "The lender and the borrower cannot be the same entity." using (out.lender != out.borrower)
            "All of the participants must be signers." using (command.signers.containsAll(out.participants.map { it.owningKey }))

            // IOU-specific constraints.
            "The IOU's value must be non-negative." using (out.value > 0)
        }
    }
    interface Commands : CommandData { class Create : Commands }
}
```

Girls in Tech
GIBRALTAR

R3 Corda – IOU Example

```
object IOUFlow {
    @InitiatingFlow
    @StartableByRPC
    class Initiator(val iouValue: Int, val otherParty: Party) : FlowLogic<SignedTransaction>() {
        @Suspendable
        override fun call(): SignedTransaction {
            // Obtain a reference to the notary we want to use.
            val notary = serviceHub.networkMapCache.notaryIdentities[0]

            // Generate an unsigned transaction.
            val iouState = IOUState(iouValue, serviceHub.myInfo.legalIdentities.first(), otherParty)
            val txCommand = Command(IOUContract.Commands.Create(), iouState.participants.map { it.owningKey })
            val txBuilder = TransactionBuilder(notary).withItems(StateAndContract(iouState, contract: "contracts.IOUContract"), txCommand)

            // Verify that the transaction is valid.
            txBuilder.verify(serviceHub)

            // Sign the transaction.
            val partSignedTx = serviceHub.signInitialTransaction(txBuilder)

            val otherPartyFlow = initiateFlow(otherParty)
            // Send the state to the counterparty, and receive it back with their signature.
            val fullySignedTx = subFlow(CollectSignaturesFlow(partSignedTx, setOf(otherPartyFlow)))

            // Notarise and record the transaction in both parties' vaults.
            return subFlow(FinalityFlow(fullySignedTx))
        }
    }
}
```

Girls in Tech
GIBRALTAR

Future Directions

- Privacy preserving ledgers
  - Zero-knowledge proofs
  - Data encryption and/or anonymization
- Scalability
  - Bitcoin: 7 tx/s, Ethereum: 20 tx/s, Visa: 56’000 tx/s
  - Lightning Network
  - Raiden / hashlocks
  - State Channels
- Provable correctness of the code, formal verifications
  - The DAO hack? Parity wallet hack?

Girls in Tech
GIBRALTAR

Further Materials

- “Mastering Bitcoin” free book: https://github.com/bitcoinbook/bitcoinbook
- “Introduction to Digital Currencies” free MOOC: https://digitalcurrency.unic.ac.cy/free-introductory-mooc/
- “Bitcoin and Cryptocurrency Technologies”: http://bitcoinbook.cs.princeton.edu
- Learn to Code Ethereum DApps By Building Your Own Game: https://cryptozombies.io
- High quality technical courses (Ethereum, Hyperledger, JP Morgan Quorum): https://academy.b9lab.com/courses
- Cryptography course (hard): https://www.coursera.org/learn/crypto
- Thoughtful technical & economical papers: https://www.r3.com/research/

Girls in Tech
GIBRALTAR

Thank You

marcin@zduniak.com

WhatsApp: +350 540 36 529

www.espeo.eu

tontinetrust.com

Girls in Tech
GIBRALTAR

Proudly Sponsored By

PLATINUM SPONSOR

GiG
GAMING INNOVATION GROUP

SILVER SPONSOR

playtech
SOURCE OF SUCCESS

EVENT SPONSOR

colorworks

VENUE PARTNER

sunborn
GIBRALTAR
5-STAR YACHT HOTEL

Contact Girls in Tech

Web: gibraltar.girlsintech.org

Instagram: girlsintech.gibraltar

Twitter: @GirlsinTech_GIB

Find us on **Facebook** and **LinkedIn**: Girls in Tech Gibraltar

Girls in Tech
GIBRALTAR