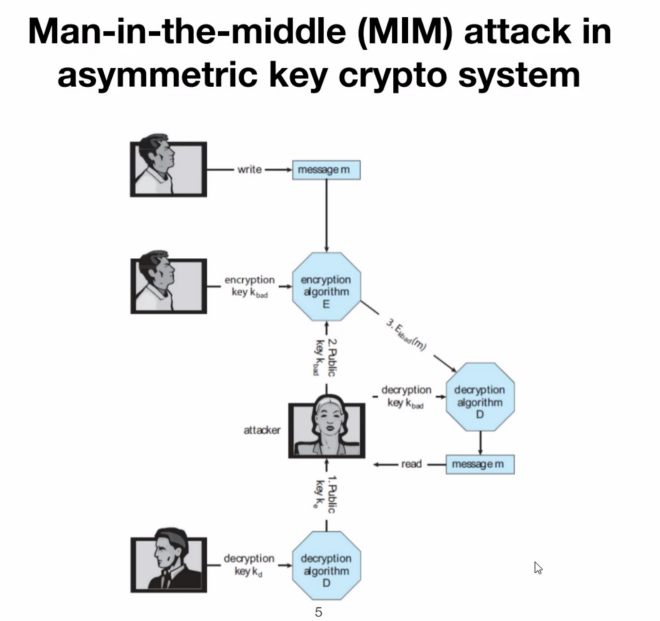
# Week 6

17 กุมภาพันธ์ 2565 16:05




Somebody could advertise their own public key as yours, so people will think using public key of that guy as communicating to you. People will send msg to that guy instead of you.

How to solve: Let's see piece by piece

Secure hash function

Security Properties
Collision-free
Hiding - given that you know about the digest it's going to be difficult to encrypt the input, the only way to do is brute force.
Puzzle-friendly

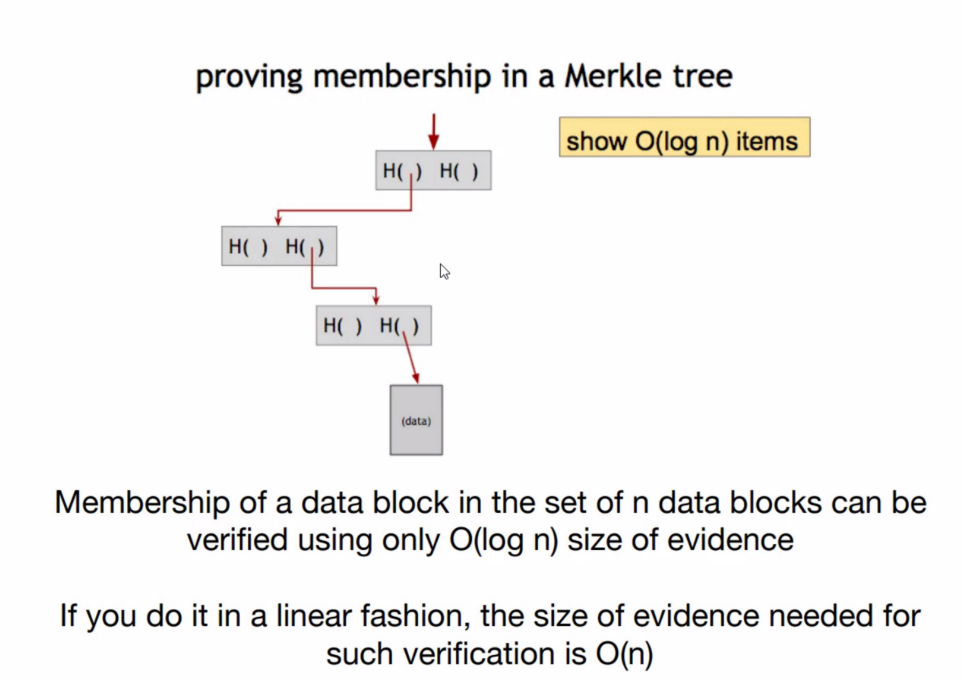
Secure download
Set up server so people download file that you certified but not the raw file but the digest than redirected to mirror site to download the file from there, In order to check that the file is the true file compare the digest.

- Downloading securely from untrusted mirror sites
  - Mirror sites are to reduce the burden on the main secured server
- Before downloading from a mirror site, contact the main secured server to obtain the hash of the file to download
- With collision-free property, it is impossible to find another file with a different content that can generate the same hash output

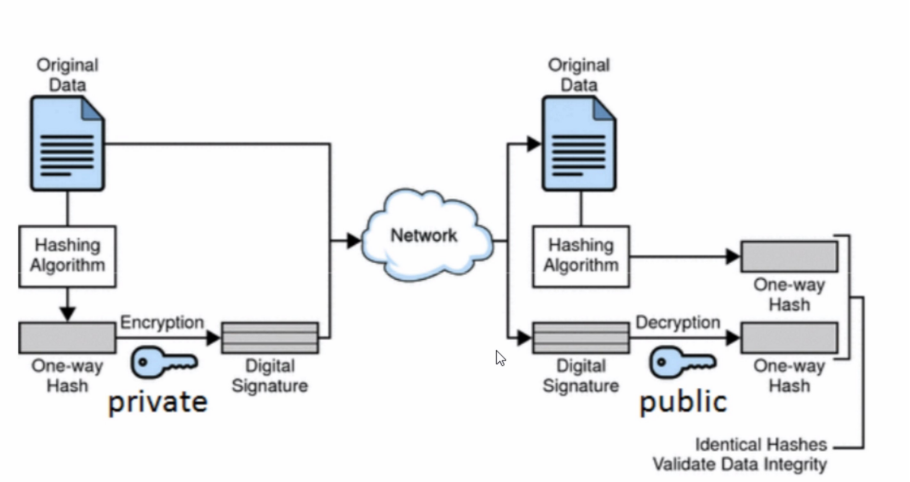
Commitment

Hash pointer
Pointer that ref a block of data that also have the digest, if the block is modified the digest gonna be

different so you will know. Have root hash separated from the rest.




Digital signature




You have a doc, a statement of transactions. In order to produce a digital signature that confirm that it is from you. Hash data to produce digest -> encrypt -> got digest -> digital signature -> send out to the network along with og data -> people can verified that the data is true not modified by -> get og data -> produce digest -> check if this digest match the decrypted msg from the digital signature.

Non-Repudiation
If digital signature is produce you cannot reject that you did not produce it.

- If the private key is never revealed to anyone beside X, the digital signature of X has non-repudiation property:
  - X cannot deny his/her associations/actions with $X_{private}$
- This non-repudiation property can not be had in a symmetric-key crypto system
  - If Alice sends a message to Bob, Alice can refuse that she ever sent that message and claim that Bob was sending the message to himself

What about the man in the middle problem?

# Certificate authorities

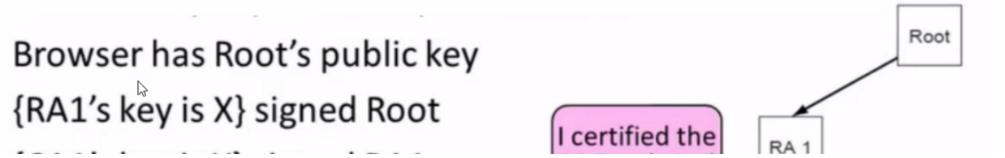
Institutions that people relay on to certified that key belong to you.

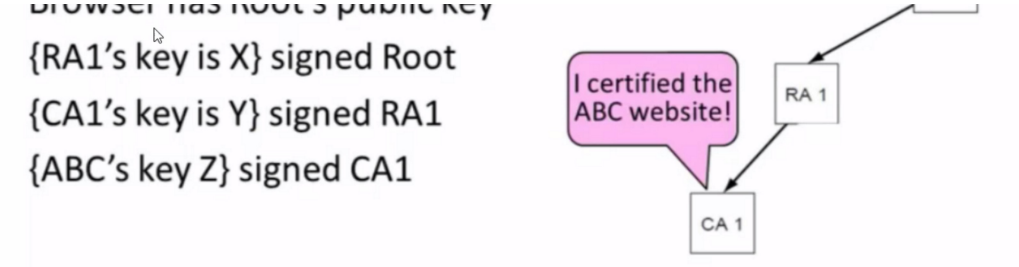
- How do we know that $X_{public}$ really belongs to X and we are not facing a man-in-the-middle attack
  - The answer: trust a certificate authority (CA) such as Verisign or Entrust
  - X goes to a CA to presents the necessary evidence to authenticate himself
    - CA signs $X_{public}$ with the CA's digital signature to ascertain the $X_{public}$ does belong to X: **[ $X_{public}$, $H(X_{public})^{CAprivate}$] - this is a certificate**
  - Before we use $X_{public}$ in a communication session, ask X for the certificate of $X_{public}$
  - Then, verify the authenticity of the certificate with **$CA_{public}$**
  - Where do we obtain all these **$CA_{public}$** ?
    - Browser vendors compile these public keys of various CAs into browser programs

Those public key they where pre compile to your browser executable. Like when you download chrome it already have public key of well known website like amazon.

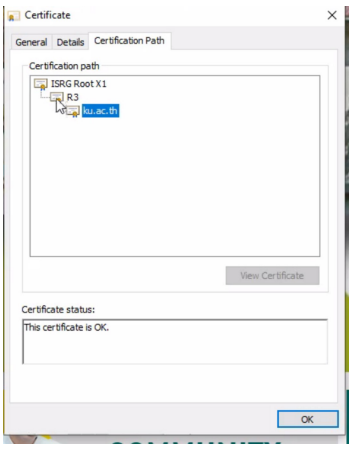
# PKI (Public Key Infrastructure)

- The server must house its own certificate together with certificates of other CAs at higher levels, all the way up to the root CA






The server will contain its own certificate along with the higher level certificate all the way up to the root.

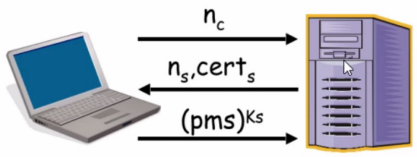



Your browser prolly have key for ISRG Root X1, so it will verified the first and then it will know that ISRG certified R3.

# Asymmetric vs symmetric key crypto system

- Symmetric-key crypto system is about 1000 times faster that its asymmetric-key counterpart
- Therefore, in a communication session, the initial stage is to use asymmetric key system to exchange a shared-key
- After the initial stage, this shared-key is used to encrypt/ decrypt all communicating data thereafter

# SSL/TLS HTTPS




- SSL Web Protocol
  - Port 443: secure http (https)
  - Use public-key encryption to distribute a shared-key
- Server has a certificate containing digital signature of the certificate authority
- Want to construct a 48-byte master secret to represent a shared symmetric key
  - Client sends a random data of 28 bytes nc to the server
  - Server sends a random data of 28 bytes ns and certificate certs to the client
  - Client verifies the certificate, making sure it is talking the right server
  - Client constructs a pre-master secret (pms) of 46 bytes encrypted using the server's public key
  - At this point, the server and the client share nc , ns ,and pms, both use this shared information to construct a master secret of 48 bytes
  - A shared symmetric key to be used in subsequent communication session after this initial setup stage can be generated from this master secret

# Overview of Blockchain, Cryptocurrency, and Smart Contract

View as database

View as application

Despite its name, Bitcoin is not a coin per se but a giant (and slow) accounting system

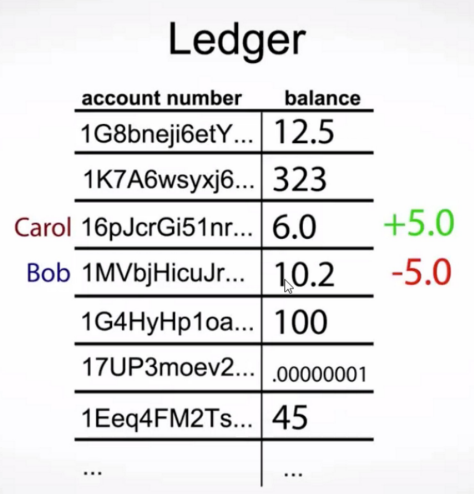
Ledger

| | account number | balance |
|---|---|---|
| | 1G8bneji6etY... | 12.5 |
| | 1K7A6wsyxj6... | 323 |
| Carol | 16pJcrGi51nr... | 1 |
| Bob | 1MVbjHicuJr... | 15.2 |
| | 1G4HyHp1oa... | 100 |
| | 17UP3moev2... | .00000001 |
| | 1Eeq4FM2Ts... | 45 |
| | ... | ... |

Bitcoin is like ledger

How can it be modified?

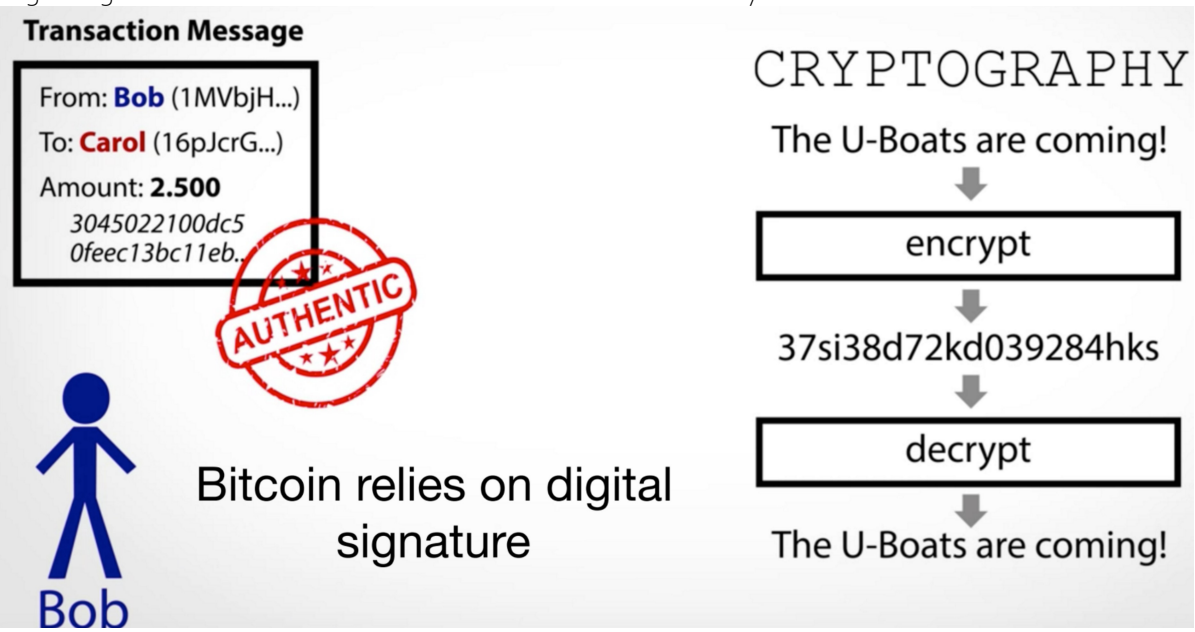
In contrast to the normal bank account, the ledger will change state when there are transaction from one address to another.

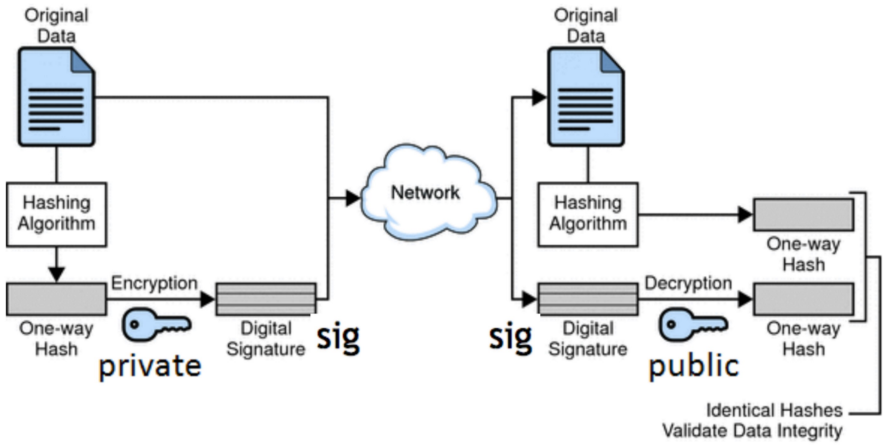



If Bob transfer 5 to Carol the ledger must reflect that state.

Unlike conventional banking systems, Bitcoin ledger is not centrally controlled. Bitcoin ledger lives in a distributed database that no single authority controls it. Such a distributed database used in Bitcoin is called blockchain.

Digital signature = technic for certified that documents are from you.






Challenges in Updating Distributed Database with No Central Control

- Who is going to be the coordinator to lead the consensus so as to make sure that the ledger is consistent
  - There is simply no single trusted authority
- Some nodes in the Bitcoin peer-to-peer network are malicious and want to destroy it
  - Again, there is no single authority to block the membership of a decentralized peer-to-peer network

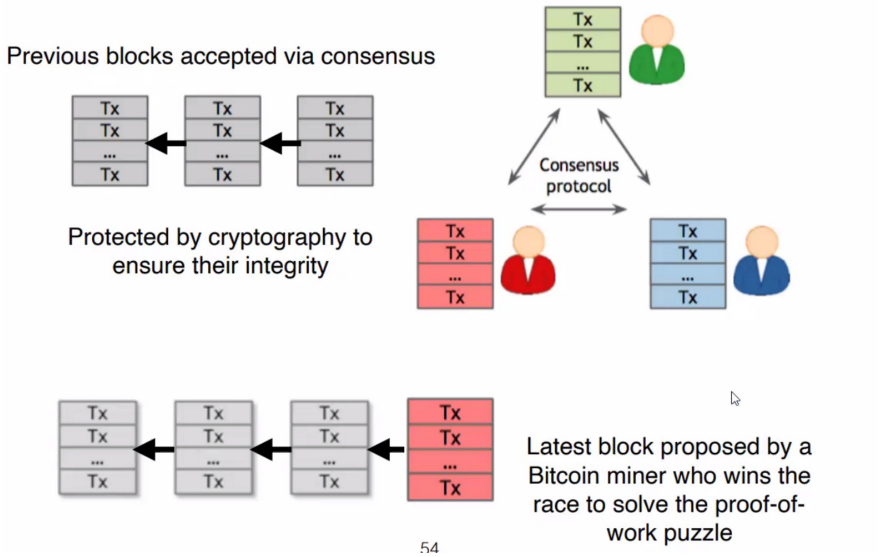
# Bitcoin Consensus

## Nakamoto Consensus

- Each node competes to find a solution to an extremely difficult math problem (Proof-of-Work)
- The first node to obtain such a solution wins and is recognized to be a co-ordinator to lead the update of the ledger
- The update is broadcast to other peer nodes who gladly accept it and update their database accordingly

A node participating in the consensus is called a **Bitcoin miner**

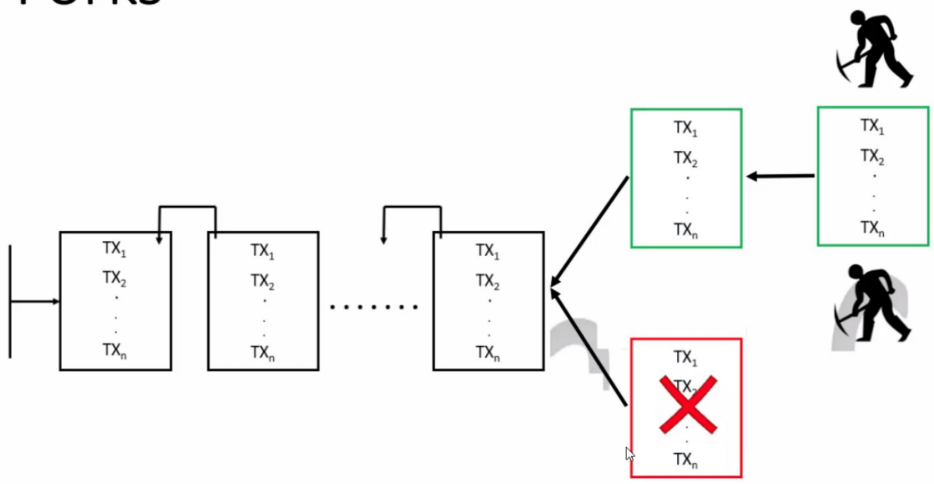
# Updating of the Bitcoin Blockchain




54

# Longest chain wins

## Forks




If there is more than one PoW winner:
Then there will be a fork and the longest chain win (more people join in)

Nodes that try to destroy bitcoin ecosystem (like upload garbage)

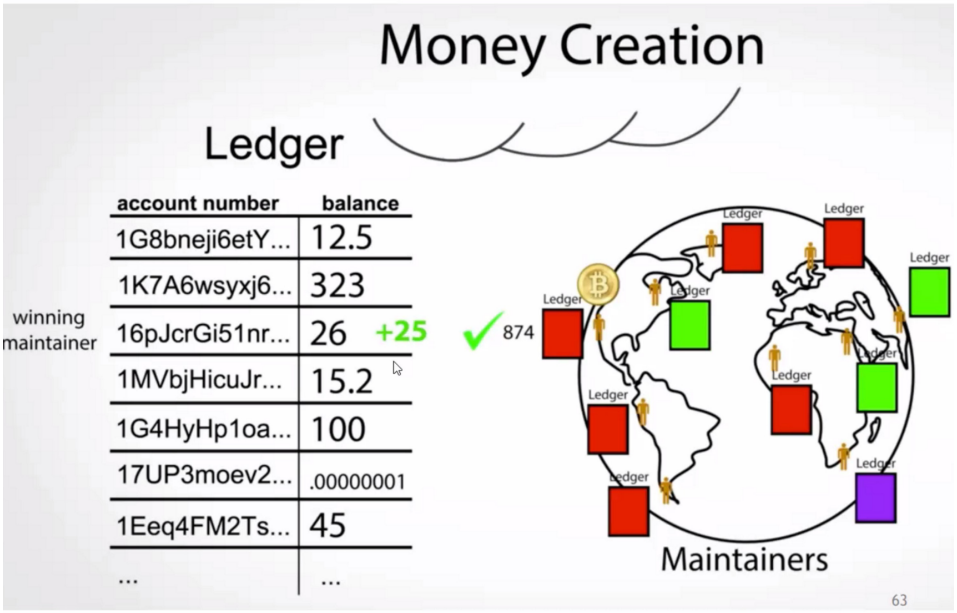
# Nakamoto Consensus and Malicious Nodes

- To update the ledger, every node must perform the proof-of-work in the competition
- Hence, if there are more than 50% of the honest nodes in the network, it is unlikely that the malicious nodes win the competition
- So, the malicious nodes is unlikely to update the ledger and destroy it

## Nakamoto Consensus and Malicious Nodes

- To update the ledger, every node must perform the proof-of-work in the competition
- Hence, if there are more than 50% of the honest nodes in the network, it is unlikely that the malicious nodes win the competition
- So, the malicious nodes is unlikely to update the ledger and destroy it
  - In case it does win the race and update garbage to the ledger, honest nodes will revert this state later

## Why Would Anyone Want to Become a Miner




It's mean you print new money

Bitcoin vs Thai baht

Desirable properties of money:

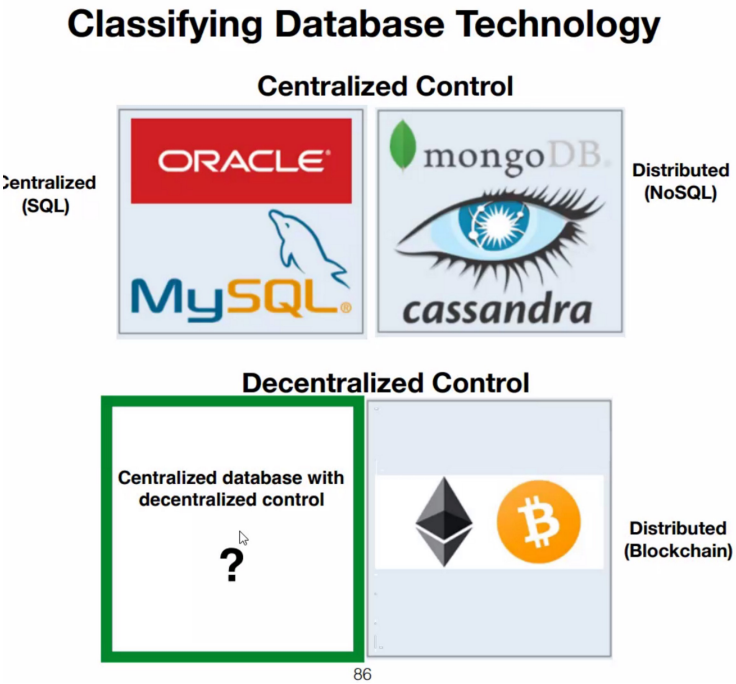
- Store of value
- Medium of exchange
- Unit of account

Currently Baht can satisfy all 3 value but Bitcoin still lack medium of exchange.

# Usage of Bitcoin: Past to Present

- Transactions on businesses operated on the Darknet and paying ransom demanded by Ransomware
- On-line gambling
- Transactions on everyday life
- As a store-of-value asset
  - Buy/sell via crypto exchanges for long term investment or short term profit taking

Can we build decentralize database so that the management and control can be done by anyone?
It's blockchain




The left quadrant still doesn't exist.

# Bitcoin in Summary

- Bitcoin brought us a new type of system that does not rely on the control of some central authorities
- Bitcoin introduced us to blockchain, a new type of distributed database with decentralized control
- Bitcoin blockchain is a database that primarily stores Bitcoin transactions
- **What if we could have blockchain that not only stores transactions but also computer programs**
  - **Coming to you in the next section: the Etheruem blockchain**

Ethereum and smart contracts

# Smart Contract

- Is a type of software stored on the Ethereum blockchain
- Is a representation of real world contracts, specifications, and rules
- In the real world, there are centralized trusted authorities that handle such things
  - Insurance companies, constitutional courts, etc.
- **Ethereum smart contracts** are written in a **high-level programming language** called **Solidity**

# What Is Ethereum?

- Decentralized platform for smart contracts
  - Afford extremely high degree of fault-tolerance
  - Censorship resistant
- Act like decentralized computers
  - However, it is not the same as cloud computing
- The is a currency called Ether being rewarded to miners who are selected to update the Ethereum blockchain

## Types of Accounts on Ethereum

- Externally Owned Account (EOA)
  - There is an owner of the EOA
  - Has an address
  - Has an Ether balance
  - Can transact with the contract code
- Contract Account (Contract)
  - Has an address
  - Has an Ether balance
  - Store smart contracts
  - Execute the stored smart contracts upon calls originated from transactions from EOA or other contract accounts

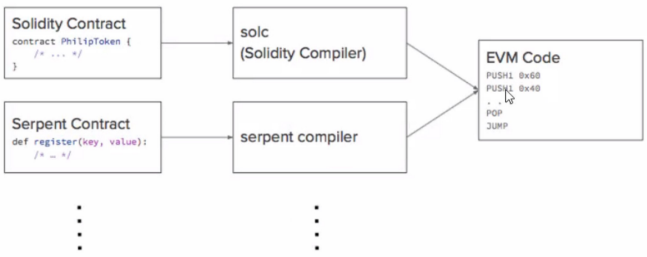


Ethereum Virtual Machine (EVM)

- EVM runs contract code
- EVM code = low-level stack-based bytecode language
- Every node in the Ethereum network runs EVM to verify blocks to be updated

- Simplicity - small number of low-level opcodes for ease of correctness and security proofs
- EVM assembly must be small and does not require a large amount of storage
- Deterministic execution; state transition can be predicted

# EVM Code Compilation




Gas and Fee

What if contract code contains infinite loop? Could it generate DoS of the system?

```
function foo()
{
    while (true) {
        /* Loop forever! */
    }
}
```

- If we can tell whether some contract code will not terminate, this would be ideal
- But, we cannot as we cannot solve the Halting problem
- Ethereum's solution
  - For a contract to run, it needs some gas
  - Each transaction must specify startgas (max gas to utilize) and gasprice (fee in Ether the originator of the transaction is willing to pay)
- To execute a transaction, the amount of Ether = startgas * gasprice will be deducted from the account of the transaction sender
- If the transaction executes successfully, the remaining Ether will be returned to the sender
- If the transaction runs out of gas before it can execute successfully, the sender will loose all the Ether and gets nothing in return as if the transaction never gets executed

## Running Smart Contracts

- Does not emphasize utmost efficiency like what cloud computing does
- Contracts run **redundantly in parallel** to achieve consensus of the network state without having a central authority to control

# Ethereum in Summary

- Decentralized platform for smart contracts
- Use blockchain to store smart contracts
- Nodes execute transactions to activate smart contract code that generate state changes
- A selected node leads the consensus to update the Ethereum blockchain

| | Ethereum | Bitcoin |
|---|---|---|
| Target time between blocks | 14.5 seconds | 10 minutes |
| Proof of work | Equihash | SHA-256[2] |
| Stale block rewards | Uncle rewards | none |
| Hard forks | regularly planned | avoided |
| State | **explicit** (committed in each block) | implicit |

Traditional contracts vs. smart contracts

| | Traditional | Smart |
|---|---|---|
| specification | Natural language + “legalese” | Code |
| identity & consent | Signatures | Digital signatures |
| dispute resolution | Judges, arbitrators | Decentralized platform |
| nullification | By judges | ???? |
| payment | As specified | built-in |
| escrow | Trusted third party | built-in |