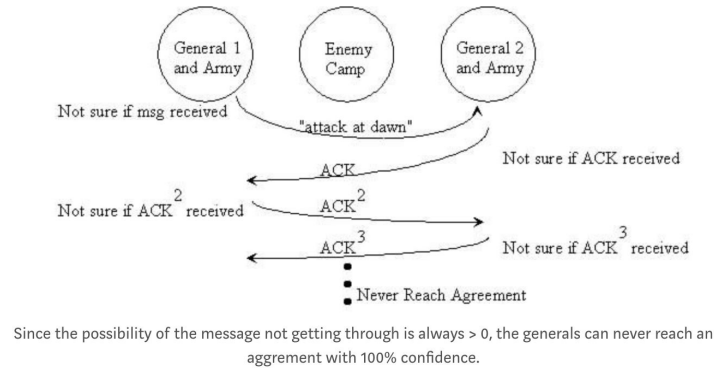


Bitcoin and Consensus

Srikar Varadaraj

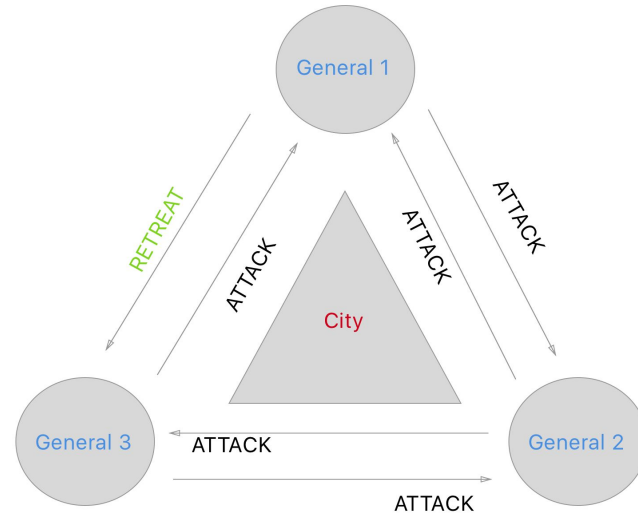
Two Generals Problem



- **Result:** Simple proof that there is no deterministic algorithm or non-deterministic algorithm that solves the Two Generals Problem

Byzantine Generals Problem

- A version of the Two Generals Problem, **where one or more generals can be faulty (traitor)**
- **1982: Leslie Lamport** et. al gives an algorithm that achieves consensus (all good actors agree on a single action) **as long as $> 2/3$ of generals are good**
- Byzantine Fault Tolerance algorithm: consensus should be achieved even when there are Byzantine failures (i.e. nodes act maliciously)



Byzantine Fault Tolerance

- BFT is a general specification
- There are **multiple consensus algorithms** that can tackle BFT
- **Practical applications:** Boeing 777, SpaceX specs



General bound proved by Leslie Lamport: $< \frac{1}{3}$ of the processors can be faulty to reach consensus with probability 1

What is



- The consensus problem is to get a group of n processes in a distributed system to agree on a value.
- A **consensus protocol** is an algorithm that produces such an agreement.
- Each process in a consensus protocol has, as part of its initial state, an input from some specified range, and must eventually decide on some output from the same range.
- All consensus algorithms must have the following qualities:
 1. **Agreement.** All processes that decide choose the same value.
 2. **Termination.** All non-faulty processes eventually decide.
 3. **Validity.** The common output value is an input value of some process.

Consensus: Historical Background

- **1980:** FLP Impossibility for asynchronous systems
 - **Fischer-Lynch-Paterson (FLP)** impossibility result demonstrates that there is **no deterministic protocol** that satisfies the agreement, termination, and non-triviality conditions for an asynchronous message-passing system in which any single process can fail undetectably
- One way to deal with it: Randomized algorithms
 - Given a current state, have the processor make a randomized decision
- First randomized algorithm to deal with this: **Ben-Or** ($<n/5$ faulty)
- Other papers improved upon this: **Bracha** ($<n/3$ faulty)
- Flurry of research with varying assumptions for randomized algorithms (shared-coin, private channels, etc.)

Further Developments

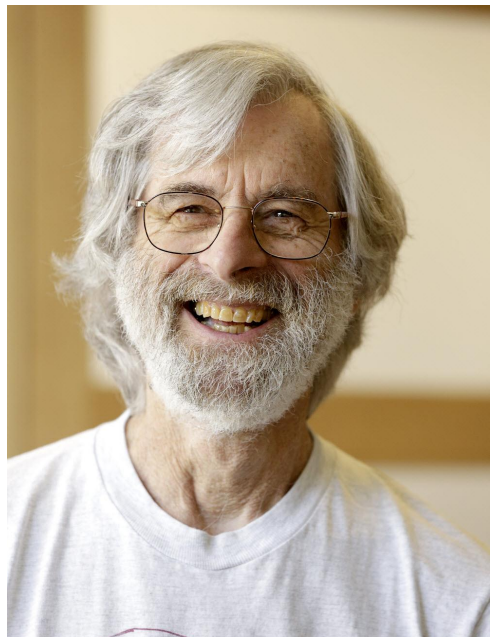
1989 -1995: Paxos & Byzantine Paxos (Leslie Lamport)

1995: Raft

1996: Chandra - Toueg

1999: pBFT

General idea: Safety (Agreement, Non-triviality) and Liveness (Termination, eventual end of consensus round) properties must be achieved to a certain degree by a consensus algorithm



Satoshi Nakamoto and Proof of Work (PoW)

- **1997: Hashcash** proposed by Adam Beck in “Pricing via Processing or Combatting Junk Mail”
- **The basic idea of POW is that: Work should be hard to do, but easy to verify**
- Text encoding of hashcash stamp added to header of e-mail
- **SHA-256** applied to the header should give a string with a certain number of 0's in the front (20 in the case below)



X-Hashcash:

1:20:1303030600:adam@cypherspace.org::McMybZIhxKXu57jd:ckvi

Bitcoin

In 2008, Satoshi suggested using PoW algorithm to solve the BFT problem in a probabilistic manner

Re: Bitcoin P2P e-cash paper

Satoshi Nakamoto | Thu, 13 Nov 2008 19:34:25 -0800

James A. Donald wrote:

```
> It is not sufficient that everyone knows X. We also
> need everyone to know that everyone knows X, and that
> everyone knows that everyone knows that everyone knows X
> - which, as in the Byzantine Generals problem, is the
> classic hard problem of distributed data processing.
```

The proof-of-work chain is a solution to the Byzantine Generals' Problem. I'll try to rephrase it in that context.

A number of Byzantine Generals each have a computer and want to attack the King's wi-fi by brute forcing the password, which they've learned is a certain number of characters in length. Once they stimulate the network to generate a packet, they must crack the password within a limited time to break in and erase the logs, otherwise they will be discovered and get in trouble. They only have enough CPU power to crack it fast enough if a majority of them attack at the same time.

They don't particularly care when the attack will be, just that they all agree. It has been decided that anyone who feels like it will announce a time, and whatever time is heard first will be the official attack time. The problem is that the network is not instantaneous, and if two generals announce different attack times at close to the same time, some may hear one first and others hear the other first.

They use a proof-of-work chain to solve the problem. Once each general receives whatever attack time he hears first, he sets his computer to solve an extremely difficult proof-of-work problem that includes the attack time in its hash. The proof-of-work is so difficult, it's expected to take 10 minutes of them all working at once before one of them finds a solution. Once one of the generals finds a proof-of-work, he broadcasts it to the network, and everyone changes their current proof-of-work computation to include that proof-of-work in the hash they're working on. If anyone was working on a different attack time, they switch to this one, because its proof-of-work chain is now longer.

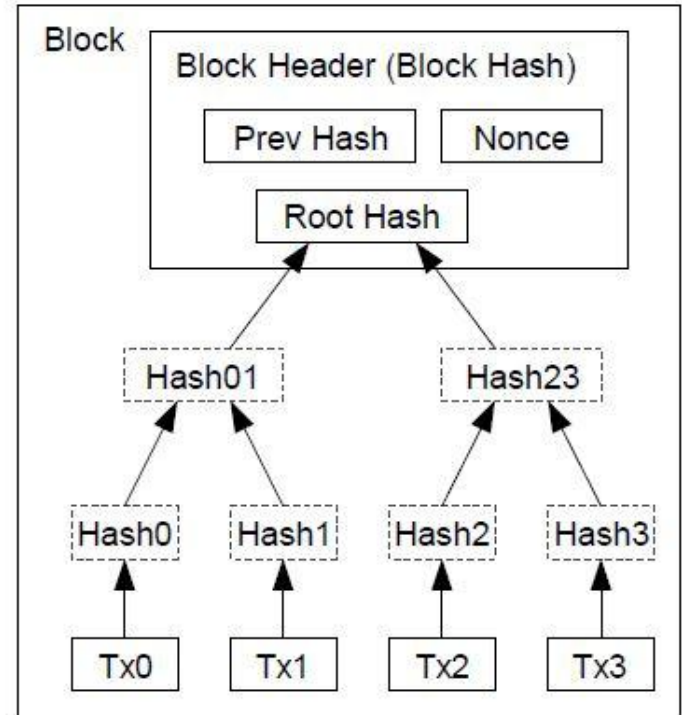
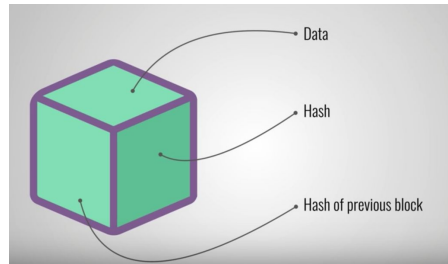
Key Components of Bitcoin

1. Hash Functions
2. Mining
3. Blocks
4. Transactions



Hash Functions

- A **hash function** is a one-way function with a **deterministic** output of **fixed length**
- $|x1-x2|$ and $|h(x1)-h(x2)|$ should not be too correlated (random)
- In Bitcoin, cryptographic hash functions are used to write new transactions (bundled into **blocks**) onto the blockchain (in a process called **mining**)



Miners

- Miners bundle incoming transactions to add to a block
- Miners solve PoW puzzle for current block (search for nonce)
- The winning miner publishes a block and receives a **mining reward**



Transactions (tx's)

Transaction View information about a bitcoin transaction

[e9a5b4dca428a33692941387df702014bde6fb76ecddd4c10d354a416a8c637c](#)

18xYhWn6k2GdckH9tKoe7SKZE7esu3fzyz
12KKFwn85Gw8fGdzh9v3WV9RsEkCAhawkg



1zpFor8kAFuiSVfSFGmnNxBrCP2NkFVcj
1CYhHfprxtJGkA5d7dzYCXcg6zxyhKrNu5

0.0109092 BTC
0.00775229 BTC

1 Confirmations

0.01866149 BTC

Summary

Size 373 (bytes)

Weight 1492

Received Time 2018-04-06 09:23:48

Lock Time Block: 516875

Included In Blocks [516876](#) (2018-04-06 09:24:13 + 0 minutes)

Confirmations 1 Confirmations

Visualize [View Tree Chart](#)

Inputs and Outputs

Total Input 0.02090923 BTC

Total Output 0.01866149 BTC

Fees 0.00224774 BTC

Fee per byte 602.611 sat/B

Fee per weight unit 150.653 sat/WU

Estimated BTC Transacted 0.0109092 BTC

Scripts [Show scripts & coinbase](#)


Source: <https://blockchain.info/>

Field	Purpose	Updated when...	Size (Bytes)
Version	Block version number	You upgrade the software and it specifies a new version	4
hashPrevBlock	256-bit hash of the previous block header	A new block comes in	32
hashMerkleRoot	256-bit hash based on all of the transactions in the block	A transaction is accepted	32
Time	Current timestamp as seconds since 1970-01-01T00:00 UTC	Every few seconds	4
Bits	Current target in compact format	The difficulty is adjusted	4
Nonce	32-bit number (starts at 0)	A hash is tried (increments)	4

Block #516872

BlockHash 00000000000000000003bb7e54ea9e0d0182c2e2aea0a8e2f6eb4329c24259e79 

Summary

Number Of Transactions	2382	Difficulty	3511060552899.7197
Height	516872 (Mainchain)	Bits	17502ab7
Block Reward	12.5 BTC	Size (bytes)	954799
Timestamp	Apr 6, 2018 5:01:34 AM	Version	536870912
Mined by	SlushPool	Nonce	3240983331
Merkle Root	 2db49367ee5df973def06e44aed2d...		
Previous Block	516871		

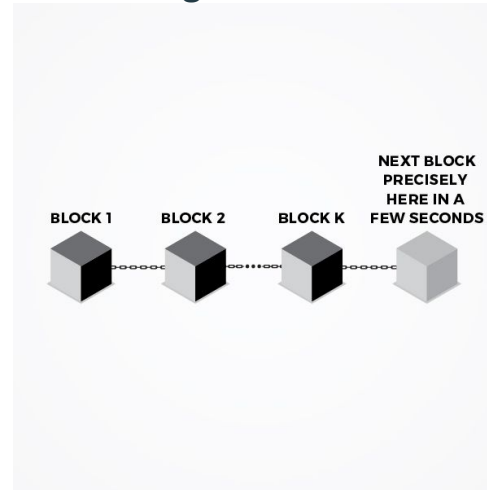
Consensus Algorithm/Structure	Proof of Work	Proof of Stake	Delegated Proof of Stake	Directed Acyclic Graph
Example Coins	Bitcoin, Litecoin, Monero, ZCash	Nxt, Waves, Decred (hybrid)	Lisk, Ark, EoS, Bitshares	IOTA, Nano
One Sentence Description	Race to solve computational puzzles in return for reward	Put up “bond” or “stake” of coins for right to mine	Proof of Stake except with elected “delegators” as miners	Not a blockchain, but a different distributed consensus system altogether
“Pros”	Have been working for a longer time;	Helps blockchains scale;	Nothing at stake somewhat solved	If it works, very efficient scaling solution
“Cons”	Heavy capital investment in mining hardware; centralized pools have majority hashpower	Nothing at Stake; Rich Get Richer	Trust in delegates required; less decentralized	Not proven to work, multiple flaws found

Ongoing Research - Consensus Algorithms

Ethereum



Algorand



Contact Me!

srikar@chainintelligence.io

srikar.varadaraj@dunyalabs.io