Understanding Bitcoin and Novel Results

Heung-No Lee

March 27th 2019

고려대학교 강의

Bitcoin, what is it?

Bitcoin P2P e-cash paper

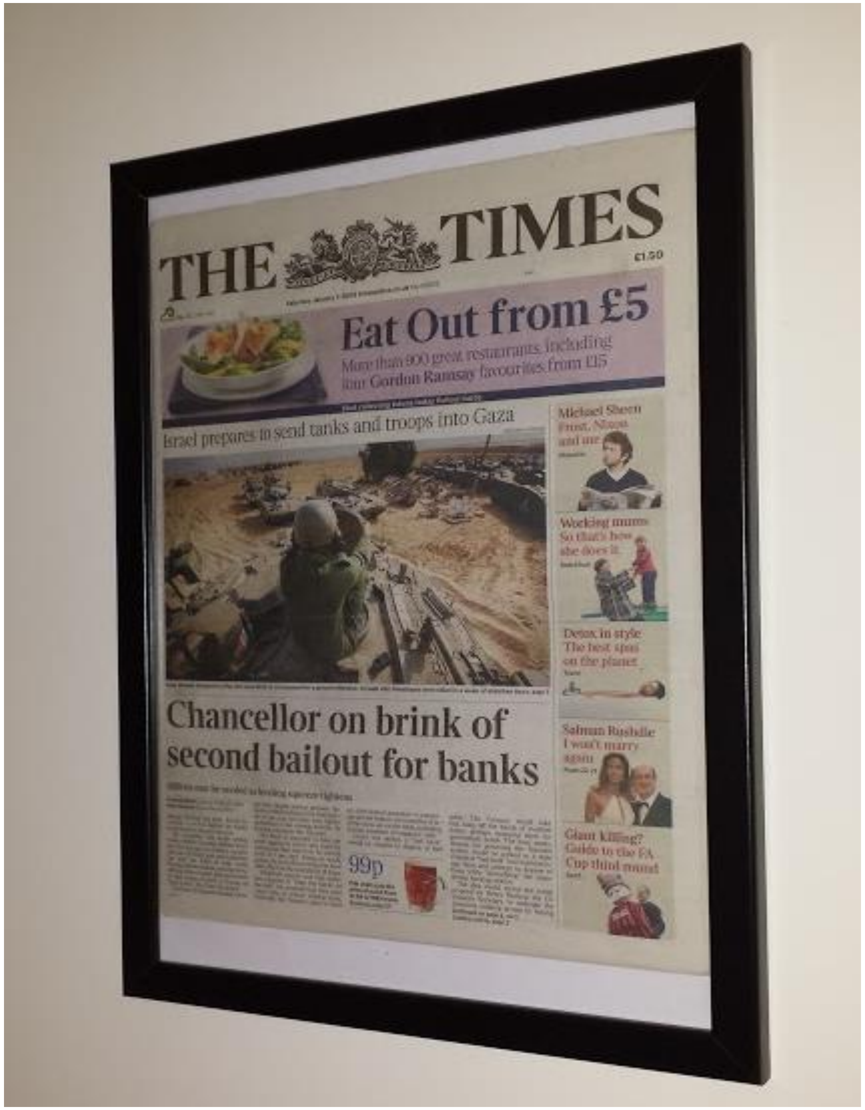
- NOVEMBER 1, 2008 SATOSHI NAKAMOTO CRYPTOGRAPHY MAILING LIST
- I've been working on a new electronic cash system that's fully peer-to-peer, with no trusted third party.
- The paper is available at:
  http://www.bitcoin.org/bitcoin.pdf
- The main properties:
  Double-spending is prevented with a peer-to-peer network.
  No mint or other trusted parties.
  Participants can be anonymous.
  New coins are made from Hashcash style proof-of-work.
  The proof-of-work for new coin generation also powers the network to prevent double-spending.
- Bitcoin: A Peer-to-Peer Electronic Cash System
- Abstract. A purely peer-to-peer version of electronic cash would allow online payments to be sent directly from one party to another without the burdens of going through a financial institution. Digital signatures provide part of the solution, but the main benefits are lost if a trusted party is still required to prevent double-spending. We propose a solution to the double-spending problem using a peer-to-peer network. The network timestamps transactions by hashing them into an ongoing chain of hash-based proof-of-work, forming a record that cannot be changed without redoing the proof-of-work. The longest chain not only serves as proof of the sequence of events witnessed, but proof that it came from the largest pool of CPU power. As long as honest nodes control the most CPU power on the network, they can generate the longest chain and outpace any attackers. The network itself requires minimal structure. Messages are broadcasted on a best effort basis, and nodes can leave and rejoin the network at will, accepting the longest proof-of-work chain as proof of what happened while they were gone.
- Satoshi Nakamoto

Bitcoin v0.1 released

- January 9, 2009 Satoshi Nakamoto Cryptography Mailing List
- Announcing the first release of Bitcoin, a new electronic cash system that uses a peer-to-peer network to prevent double-spending. It's completely decentralized with no server or central authority.
- See bitcoin.org for screenshots.
- Download link: http://downloads.sourceforge.net/bitcoin/bitcoin-0.1.0.rar
- Windows only for now. Open source C++ code is included.
- – Unpack the files into a directory
  – Run BITCOIN.EXE
  – It automatically connects to other nodes
- If you can keep a node running that accepts incoming connections,
  you'll really be helping the network a lot. Port 8333 on your firewall needs to be open to receive incoming connections.

Genesis Block

- The Times 03/Jan/2009 Chancellor on brink of second bailout for banks



THE TIMES

£1.50

Eat Out from £5

More than 900 great restaurants, including four Gordon Ramsay favourites from £15

Israel prepares to send tanks and troops into Gaza

Michael Sheen Frost, Nixon and me

Working mum So that's how she does it

Detox in style The best spas on the planet

Salman Rushdie I won't marry again

Glam killing? Guide to the FA Cup third round

Chancellor on brink of second bailout for banks

99p

https://en.bitcoin.it/wiki/Genesis_block

- The Chancellor will decide within weeks whether to pump billions more into the economy as evidence mounts that the £37 billion part-nationalisation last year has failed to keep credit flowing.
- Options include cash injections, offering banks cheaper state guarantees to raise money privately or buying up "toxic assets", *The Times* has learnt.

At the birth of Bitcoin, there were many issues which made us to think!

- Today, currency is not money.
  - USD does not have any internal value (No more gold standard).
  - Currency is created by banks when someone takes out a loan or government issues bonds (I.O.U.) to banks, or by increasing an electronic balance to the commercial banks at the whim of FED.

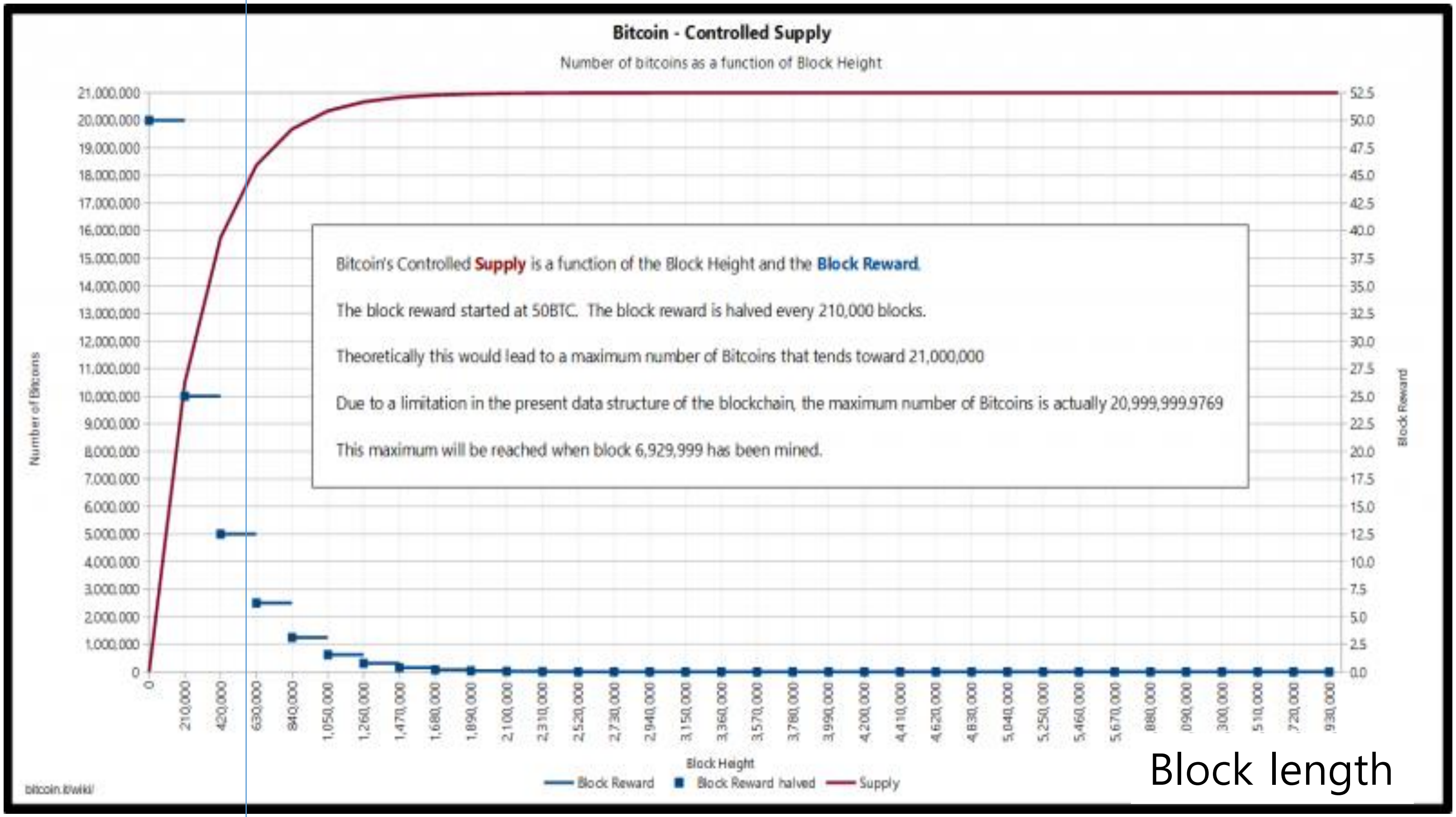
- With frequent financial crises, trust to gov. has been greatly tarnished.
  - People are grown wary of budget deficit and currency expansion.

- Issues around bitcoin are
  - Decentralization
  - Reforming Wall Street
  - Unbundling big corporations
  - Reducing inequality

Bitcoin Issuance Schedule

Coin reward
Per block

Total coins



Bitcoin's Controlled **Supply** is a function of the Block Height and the **Block Reward**.

The block reward started at 50BTC. The block reward is halved every 210,000 blocks.

Theoretically this would lead to a maximum number of Bitcoins that tends toward 21,000,000

Due to a limitation in the present data structure of the blockchain, the maximum number of Bitcoins is actually 20,999,999.9769

This maximum will be reached when block 6,929,999 has been mined.

Block length

2019

How does it work?

Public and Private Keys
Secure Hash Functions
Proof-of-Work

Bitcoin

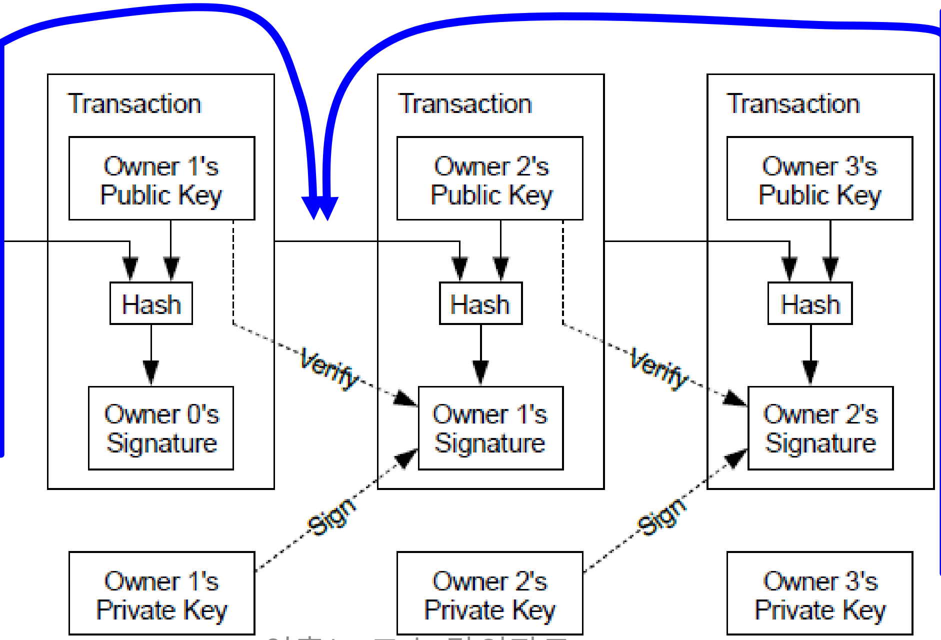
- Bitcoin is a chain of signatures.
  - Digital money with the effect of in-person transfer of money

An e-coin is a chain of signatures.

We define an electronic coin as a chain of digital signatures. Each owner transfers the coin to the next by digitally signing a hash of the previous transaction and the public key of the next owner and adding these to the end of the coin. A payee can verify the signatures to verify the chain of ownership.

2nd Tx box

1. The first TX shows that O1 owns the coin.
2. O1 can transfer it to anybody, say O2.
3. O1 writes TX 2.
4. O1 asks Owner2, the new owner, for his public key.
5. Anyone can hash the received public key and TX1, and calculates the hash value.



6. To show his ownership status, O1 signs the hash value and leaves the signature in TX2.
7. Now, anybody can verify O1's signature with O1's pubic key written in TX 1.

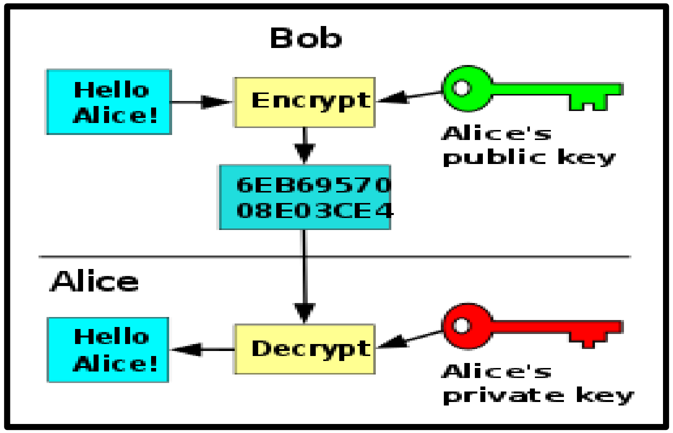
Once TX 2 recorded and published, anybody can easily see TX 2 and knows that O1 has transferred his coin ownership to O2.

Digital signatures provide part of the solution! **(RSA example)**

- A pair of *private and public key generated to each individual is given.*
- *Bob* wants to send a private message *m* to *Alice.*
- *Bob encrypts m* with *Alice's public key Pub_a.*

  $y = ENC(m, Pub\_a)$
- *Alice receives y and decrypts it using its private key.*

  $m = DEC(y, Pri\_a)$
- *ENC* and *DEC* are given and known functions.



Google pictures

Digital Signing with RSA

Q.1. Let *e*, *m* and *n* be *known* positive integers.

Is it easy to find *d?*

$$\left(m^{e}\right)^{d} = m \bmod n \quad \text{-- (1)}$$

Once d known, it is easy to check

$$\left(m^{d}\right)^{e} = m \bmod n \quad \text{-- (2)}$$

Let d be pri-key and e public-key.

Private-key

Public-key

Ex 1) *Bob wants to send a private message m to Alice.*

Bob uses public key e of Alice, send c = m^e to Alice.

Only Alice can recover original message m, using d in (1).

Ex 2) *Bob can append his signature $h(m)^d$ to his message m sent to Alice.*

Bob uses his pri-key d to generate $h(m)^d$.

Using Bob's pub-key e, Alice recovers h(m) via (2).

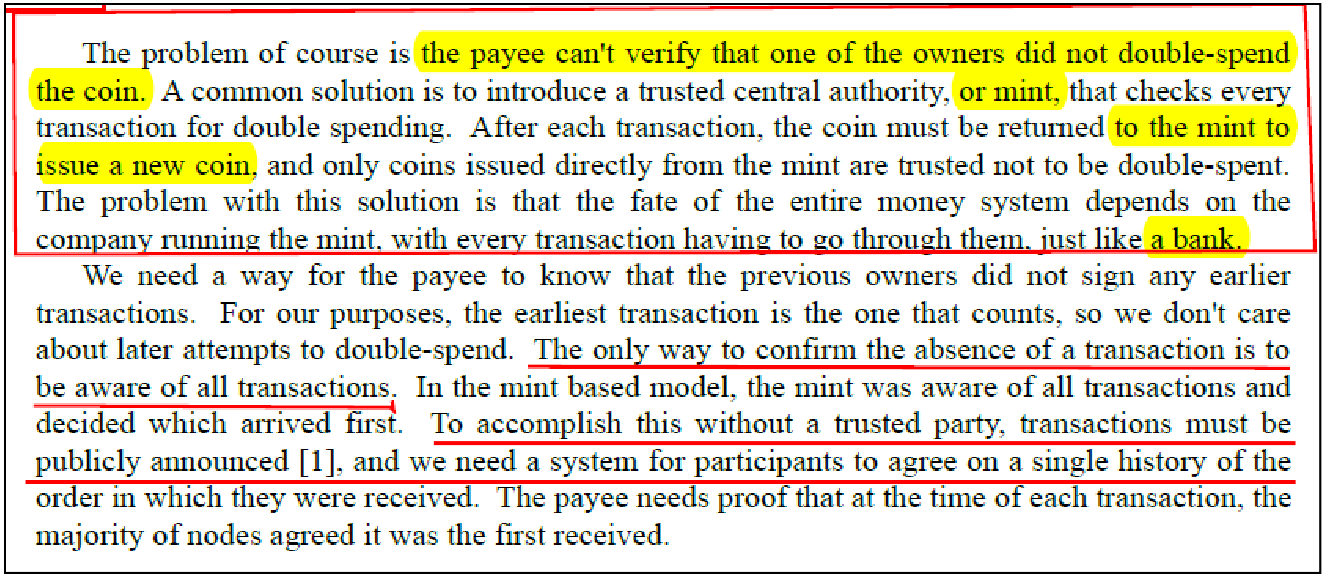
Using Bob's message m recovered from Ex1), Alice generates h(m).

Alice checks if the two hash values match.

Note: bitcoin does not use RSA but use secure hash functions and Elliptic Curve Digital Signature Algorithms.

But for today, we use RSA because it is more familiar to us.

Double Spending Problem



The problem of course is the payee can't verify that one of the owners did not double-spend the coin. A common solution is to introduce a trusted central authority, or mint, that checks every transaction for double spending. After each transaction, the coin must be returned to the mint to issue a new coin, and only coins issued directly from the mint are trusted not to be double-spent. The problem with this solution is that the fate of the entire money system depends on the company running the mint, with every transaction having to go through them, just like a bank.

We need a way for the payee to know that the previous owners did not sign any earlier transactions. For our purposes, the earliest transaction is the one that counts, so we don't care about later attempts to double-spend. The only way to confirm the absence of a transaction is to be aware of all transactions. In the mint based model, the mint was aware of all transactions and decided which arrived first. To accomplish this without a trusted party, transactions must be publicly announced [1], and we need a system for participants to agree on a single history of the order in which they were received. The payee needs proof that at the time of each transaction, the majority of nodes agreed it was the first received.

Bitcoin uses the internet.

Bitcoin attracts P2P nodes.

P2P nodes share a blockchain.

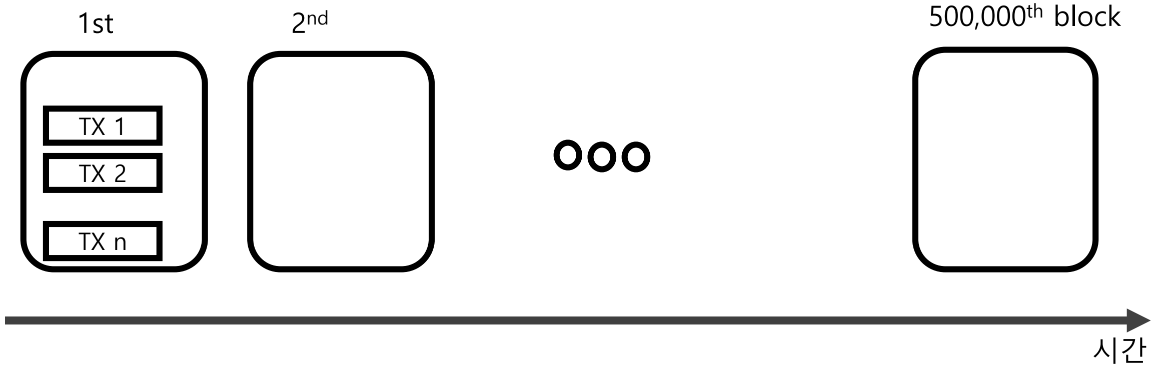
- Blockchain is to mean a digital leger:

  Blockchain is a chain of blocks.

  Each block is time stamped.

  Each block stores TXs.

- Blockchain also implies the technology itself.



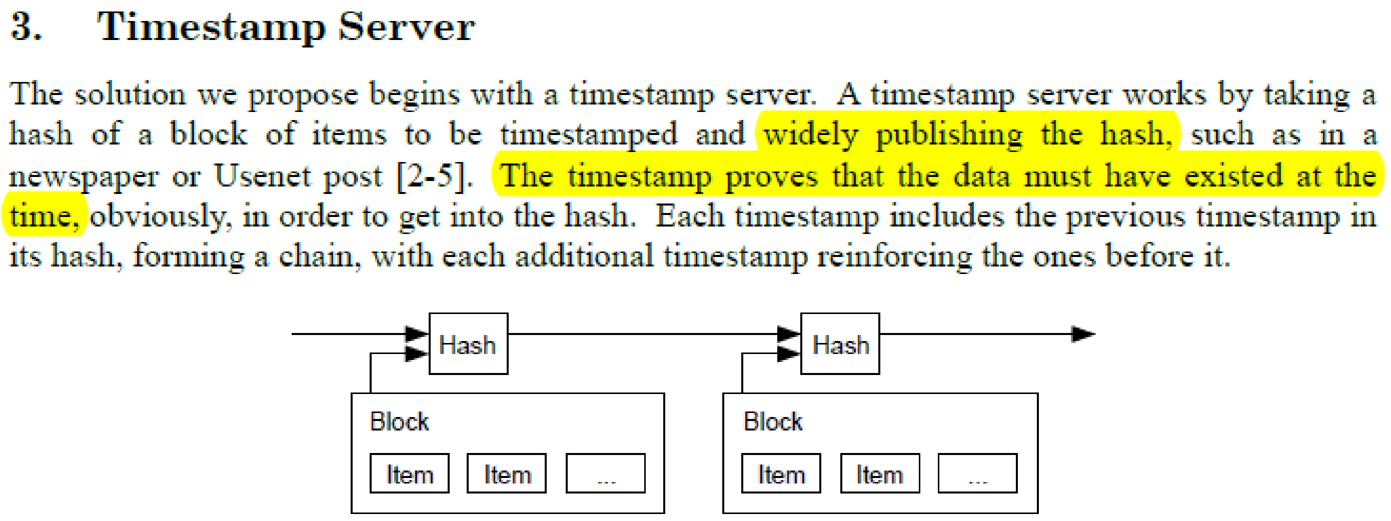
The blockchain is left open for viewing.

- The digital ledger is left open.
- Anyone can talk to a node and view the ledger. (Public Blockchain)

These ledgers are the same except the most recent blocks.

A ledger in America

A leger in Korea

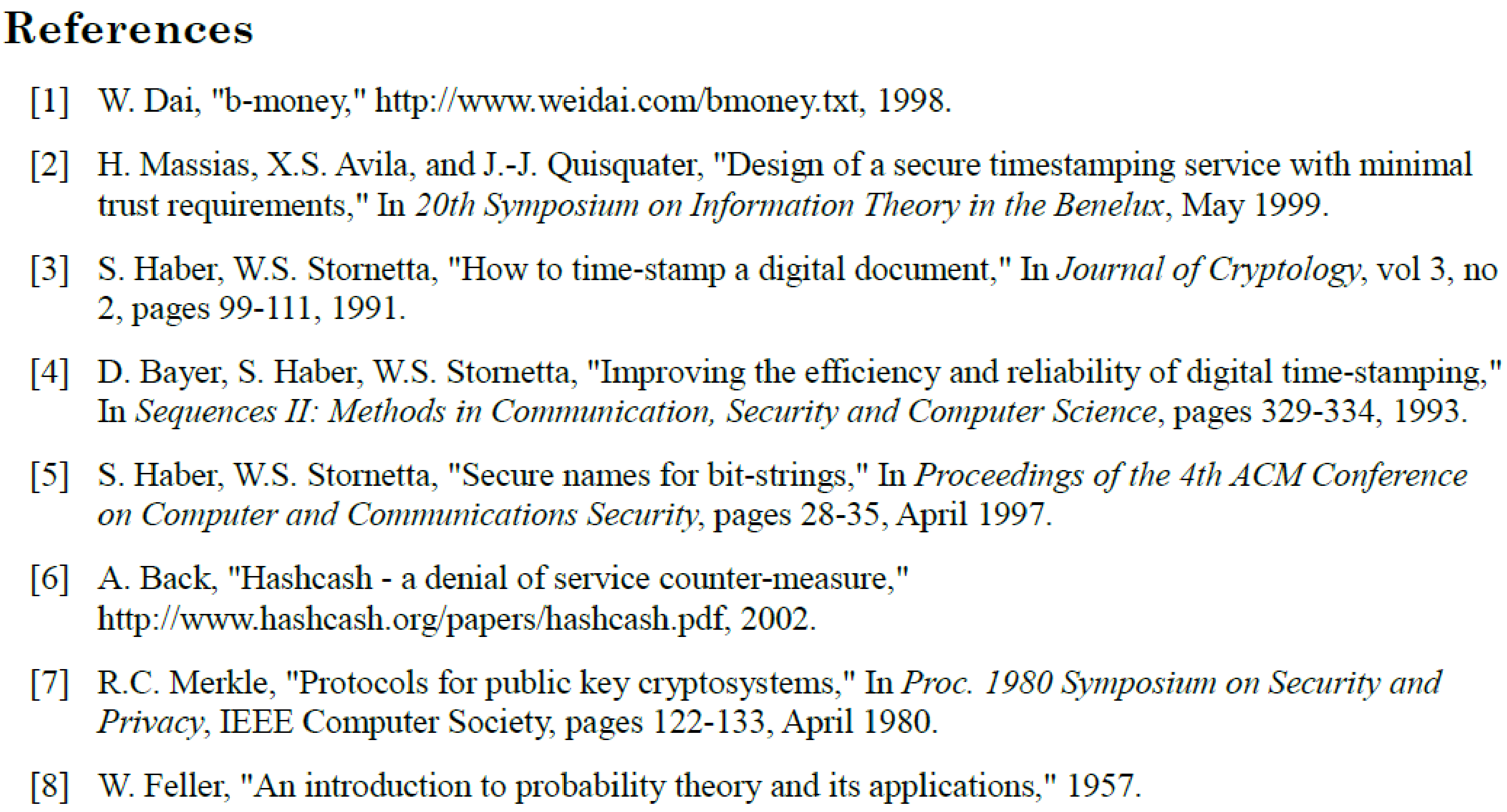


3. Timestamp Server

The solution we propose begins with a timestamp server. A timestamp server works by taking a hash of a block of items to be timestamped and widely publishing the hash, such as in a newspaper or Usenet post [2-5]. The timestamp proves that the data must have existed at the time, obviously, in order to get into the hash. Each timestamp includes the previous timestamp in its hash, forming a chain, with each additional timestamp reinforcing the ones before it.

- If a timestamp server indicates the existence of hash value at a certain time point, then a legitimate ledger can indeed be made?
  - If hash values only are published while no block contents are published, there will be no issue of scalability, and privacy can be kept since no one other than the parties involved in the transactions can see the content of transactions!
  - But how can one verify for coin ownership and double spending transactions.
- The problem is to decide who should run the timestamp server?
- If a government runs it, it becomes a private blockchain (social terms it is a public chain)!
- What possible problems are there if it is run by government?

Reference of Bitcoin White Paper



References

[1] W. Dai, "b-money," http://www.weidai.com/bmoney.txt, 1998.

[2] H. Massias, X.S. Avila, and J.-J. Quisquater, "Design of a secure timestamping service with minimal trust requirements," In *20th Symposium on Information Theory in the Benelux*, May 1999.

[3] S. Haber, W.S. Stornetta, "How to time-stamp a digital document," In *Journal of Cryptology*, vol 3, no 2, pages 99-111, 1991.

[4] D. Bayer, S. Haber, W.S. Stornetta, "Improving the efficiency and reliability of digital time-stamping," In *Sequences II: Methods in Communication, Security and Computer Science*, pages 329-334, 1993.

[5] S. Haber, W.S. Stornetta, "Secure names for bit-strings," In *Proceedings of the 4th ACM Conference on Computer and Communications Security*, pages 28-35, April 1997.

[6] A. Back, "Hashcash - a denial of service counter-measure," http://www.hashcash.org/papers/hashcash.pdf, 2002.

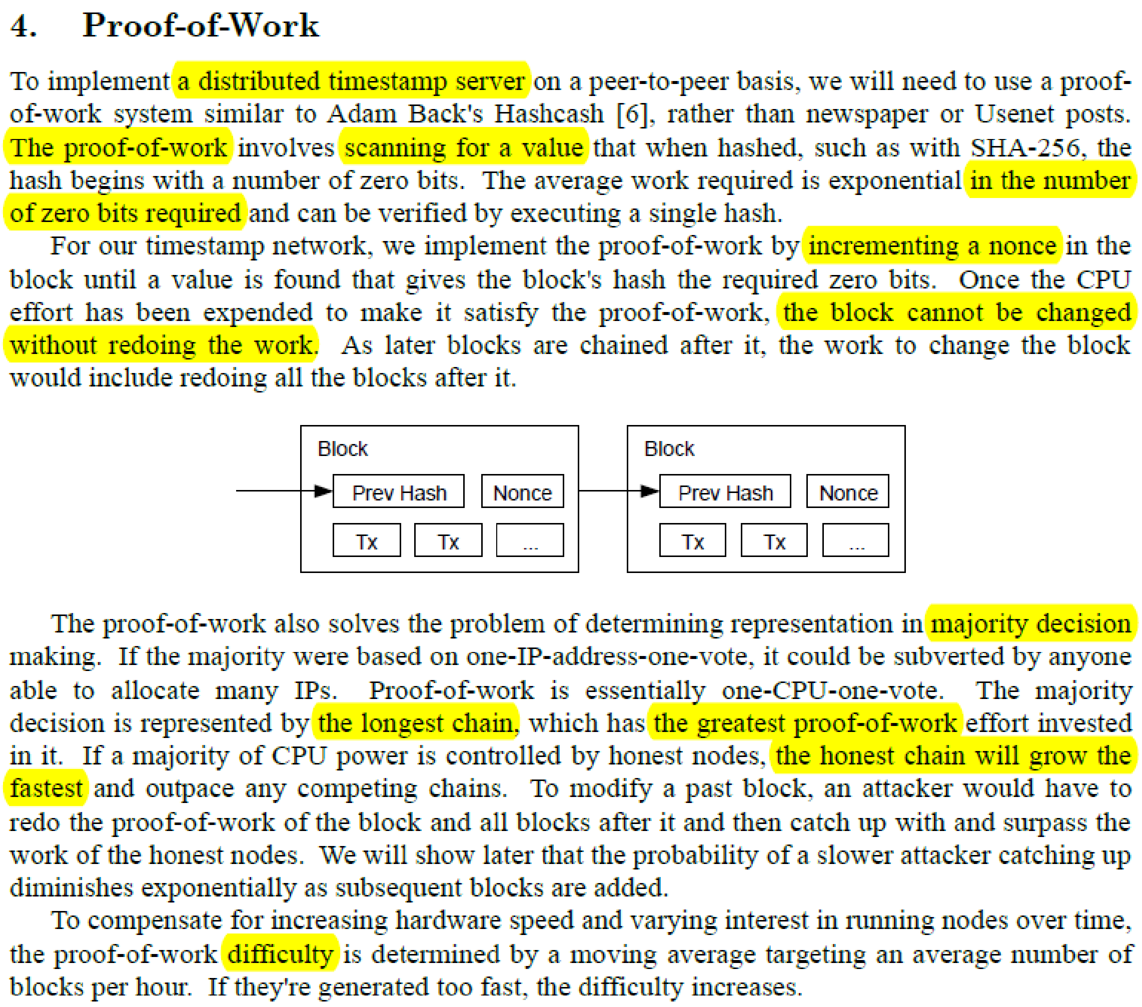
[7] R.C. Merkle, "Protocols for public key cryptosystems," In *Proc. 1980 Symposium on Security and Privacy*, IEEE Computer Society, pages 122-133, April 1980.

[8] W. Feller, "An introduction to probability theory and its applications," 1957.

Blockchain & Proof-of-Work

- Aim to make a timestamp Server in a P2P network.
  - Why?
  - Not to rely on any central authority.
  - Central authority such as banks and states
  - Within a nation, the state government may run the timestamp server.
  - But for trades overseas, P2P across different nations is needed.

- Solution?
  - Distributed timestamp P2P server network
  - Distributed, thus, it is difficult to maintain the integrity of data.
  - To keep the integrity of data, PoW system is proposed!



4. Proof-of-Work

To implement a distributed timestamp server on a peer-to-peer basis, we will need to use a proof-of-work system similar to Adam Back's Hashcash [6], rather than newspaper or Usenet posts. The proof-of-work involves scanning for a value that when hashed, such as with SHA-256, the hash begins with a number of zero bits. The average work required is exponential in the number of zero bits required and can be verified by executing a single hash.

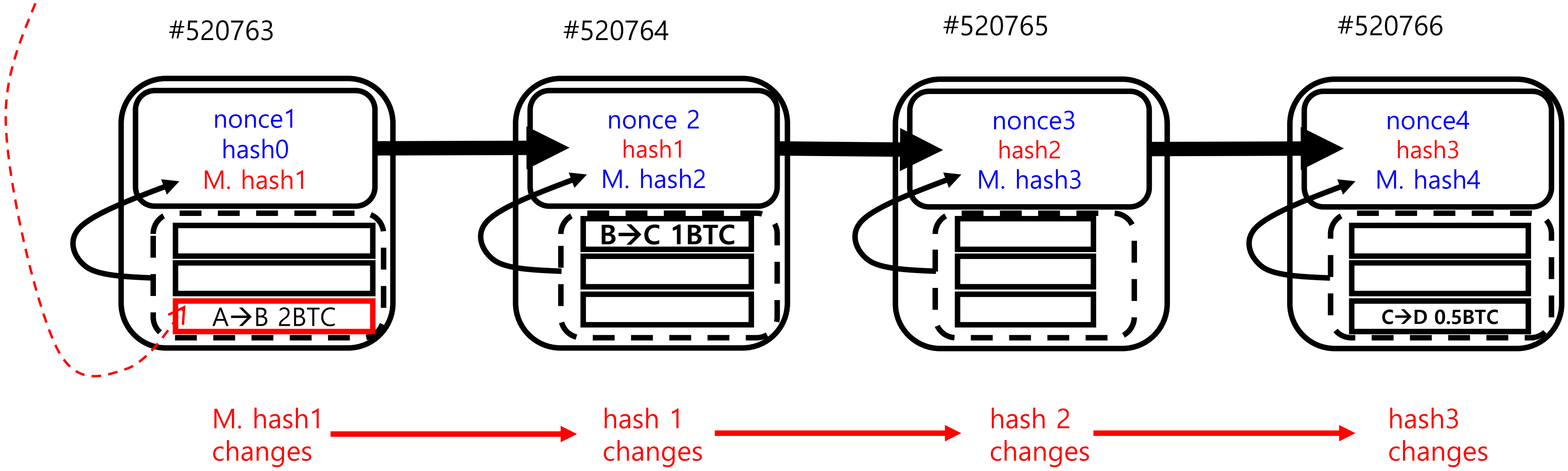
For our timestamp network, we implement the proof-of-work by incrementing a nonce in the block until a value is found that gives the block's hash the required zero bits. Once the CPU effort has been expended to make it satisfy the proof-of-work, the block cannot be changed without redoing the work. As later blocks are chained after it, the work to change the block would include redoing all the blocks after it.

The proof-of-work also solves the problem of determining representation in majority decision making. If the majority were based on one-IP-address-one-vote, it could be subverted by anyone able to allocate many IPs. Proof-of-work is essentially one-CPU-one-vote. The majority decision is represented by the longest chain, which has the greatest proof-of-work effort invested in it. If a majority of CPU power is controlled by honest nodes, the honest chain will grow the fastest and outpace any competing chains. To modify a past block, an attacker would have to redo the proof-of-work of the block and all blocks after it and then catch up with and surpass the work of the honest nodes. We will show later that the probability of a slower attacker catching up diminishes exponentially as subsequent blocks are added.

To compensate for increasing hardware speed and varying interest in running nodes over time, the proof-of-work difficulty is determined by a moving average targeting an average number of blocks per hour. If they're generated too fast, the difficulty increases.

Content in the blockchain cannot be changed (easily).

- What happens when any alteration is made?
- A small alteration is easily noticeable!
- If an unnoticeable change is wanted, a whole alteration is needed.
- The whole job is to redo all the hashes of the following blocks.
- Proof-of-Work (PoW) is imposed in each block and thus the whole job cannot be made easily.



#520763
nonce1
hash0
M. hash1
A→B 2BTC

#520764
nonce 2
hash1
M. hash2
B→C 1BTC

#520765
nonce3
hash2
M. hash3

#520766
nonce4
hash3
M. hash4
C→D 0.5BTC

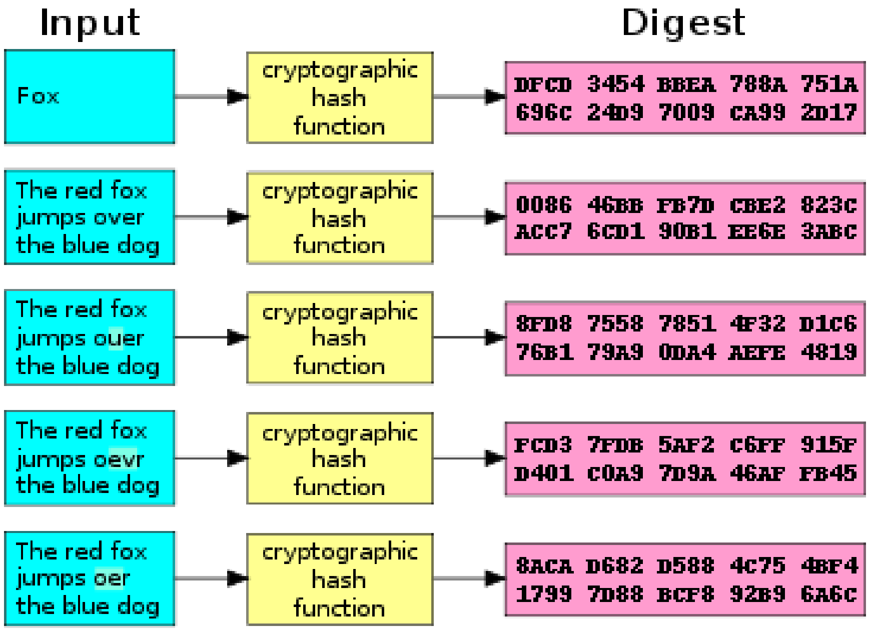
M. hash1 changes → hash 1 changes → hash 2 changes → hash3 changes

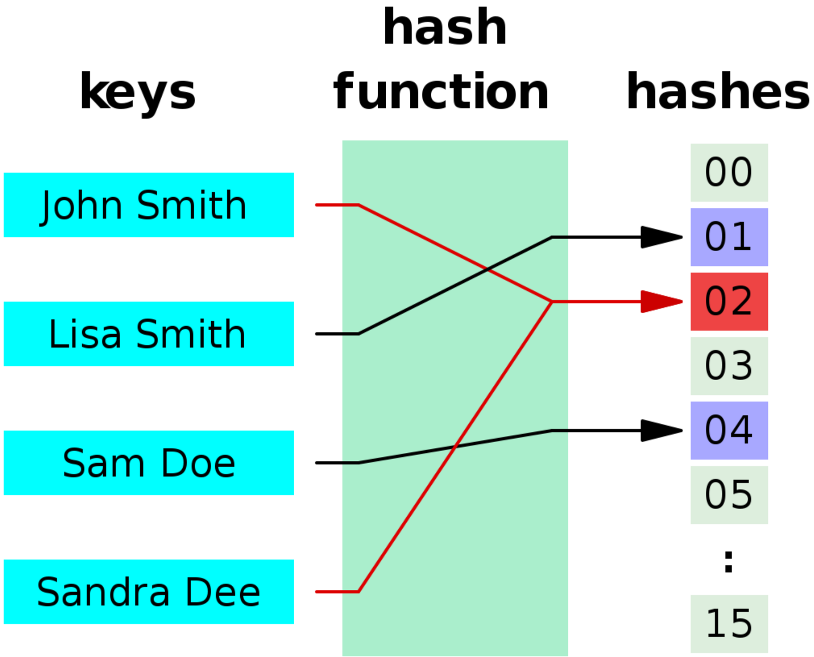
What is Hash Function?

- Bitcoin uses SHA256.
- The input to the hash function is a text message or a file.
- The output of the hash function is 256 bit string.
- Conditions for Good Hash Function
  - (One way) With a little change in the input, the output is completely different.
    - Input distance has no relation to output distance.
  - (Collision free) Given y = H(x), finding x1 such that H(x1) = y shall be almost impossible!
  - (Collision free stronger) Finding an input pair x and x1 which leads to H(x) = H(x1) shall be almost impossible!

- See examples in MIT blockchain Demo, http://blockchain.mit.edu/how-blockchain-works/

Secure Hash Function I/O


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

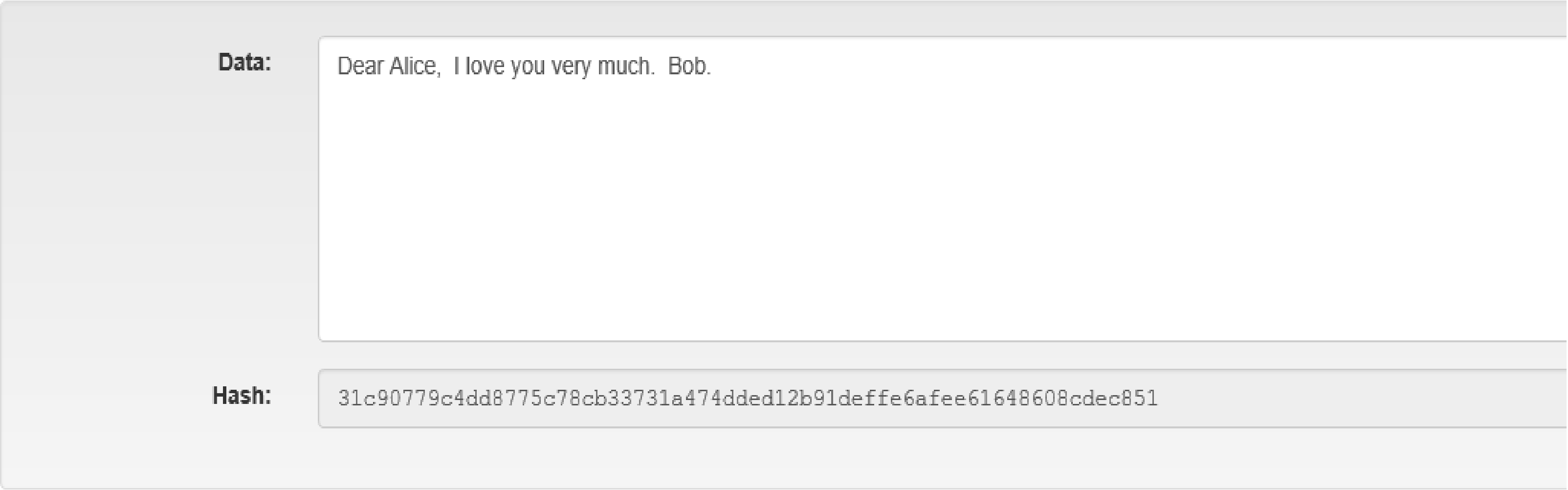


Pictures from google image

http://blockchain.mit.edu/hash

Use the Hash function at MIT

2. HASH



Data: Dear Alice, I love you very much. Bob.

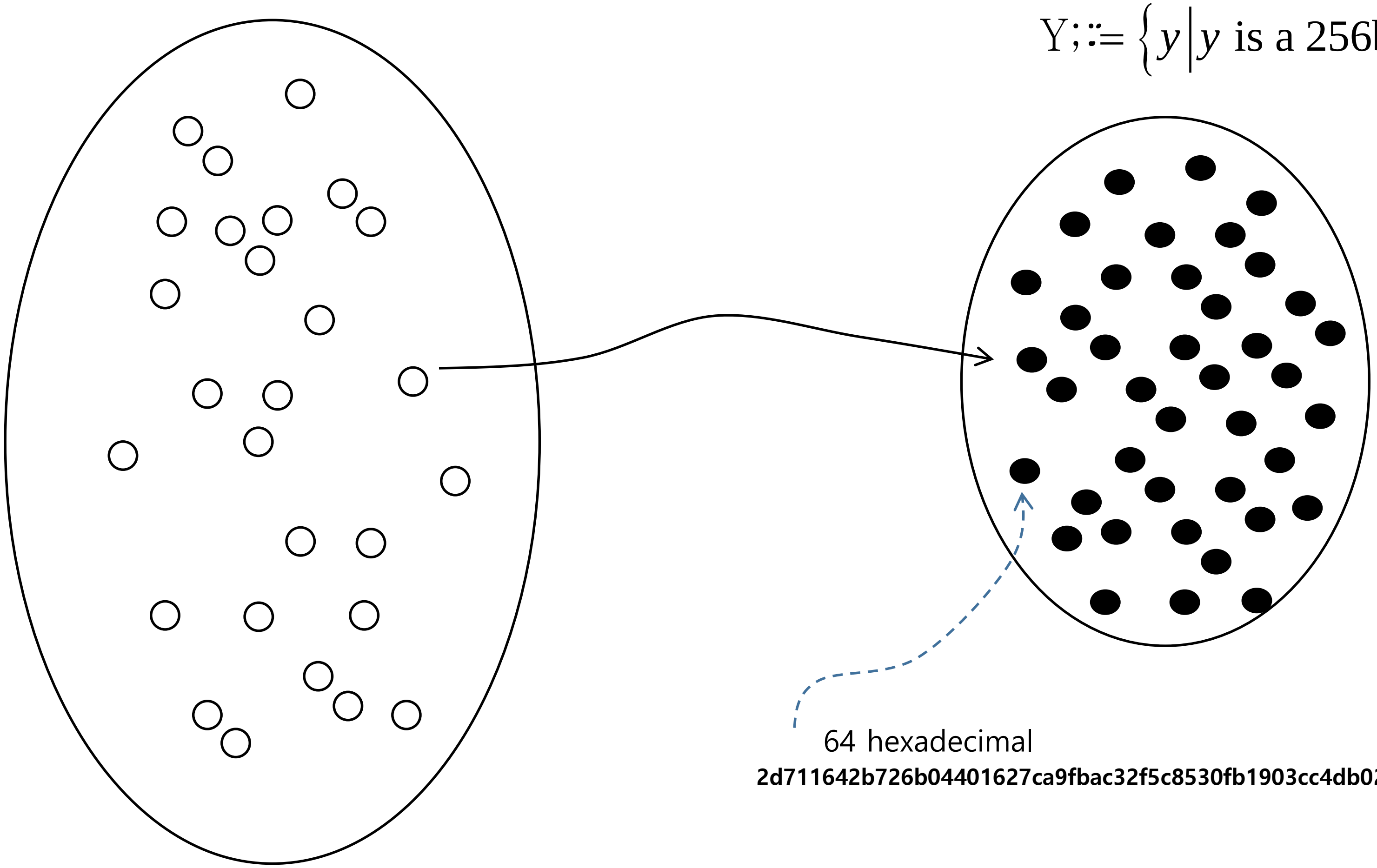
Hash: 31c90779c4dd8775c78cb33731a474dded12b91deffe6afee61648608cdec851

31c90779c4dd8775c78cb33731a474dded12b91deffe6afee61648608cdec851

SHA256, F(x) = y

$$\mathrm{X} ::= \left\{ x \,\middle|\, x \text{ is a message up to 1 Mbyte in size} \right\}$$

$$\mathrm{Y} ::= \left\{ y \,\middle|\, y \text{ is a 256bit string} \right\}$$



64 hexadecimal

2d711642b726b04401627ca9fbac32f5c8530fb1903cc4db02258717921a4881

What is PoW?

Finding Good Block Summary

- Let H(*) be the Hash Function

- Function F takes an input x and gives output y

  y = F(x)

- F(block) = block summary (hash value)

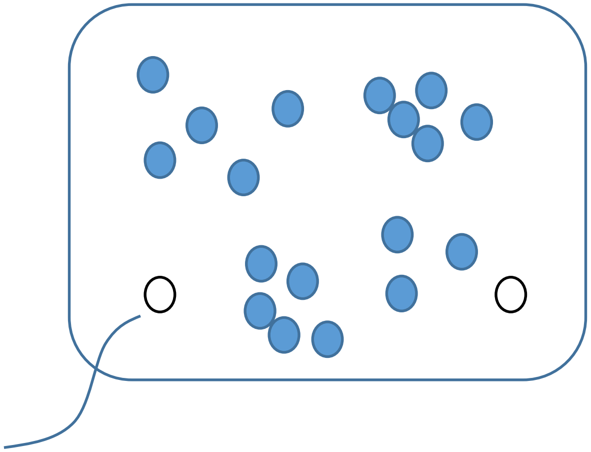
- Finding good block summary can be written as.

  F(block, *nonce*) < a certain value (PoW)

- Given a block, find nonce which satisfies the above inequality.

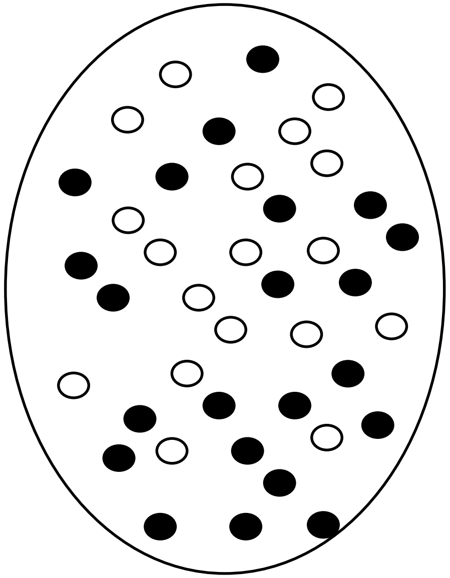
- Once *nonce* found, record it in the block header.

What is the probability to select a white ball?



The probability a cpu solves (PoW) in a single cycle, given the first four strings are zeros?

$$Y;\!:= \left\{ y \,\middle|\, y \text{ is a 256bit string} \right\}$$



256/4 = 2^8/2^2 = 2^6 = 64

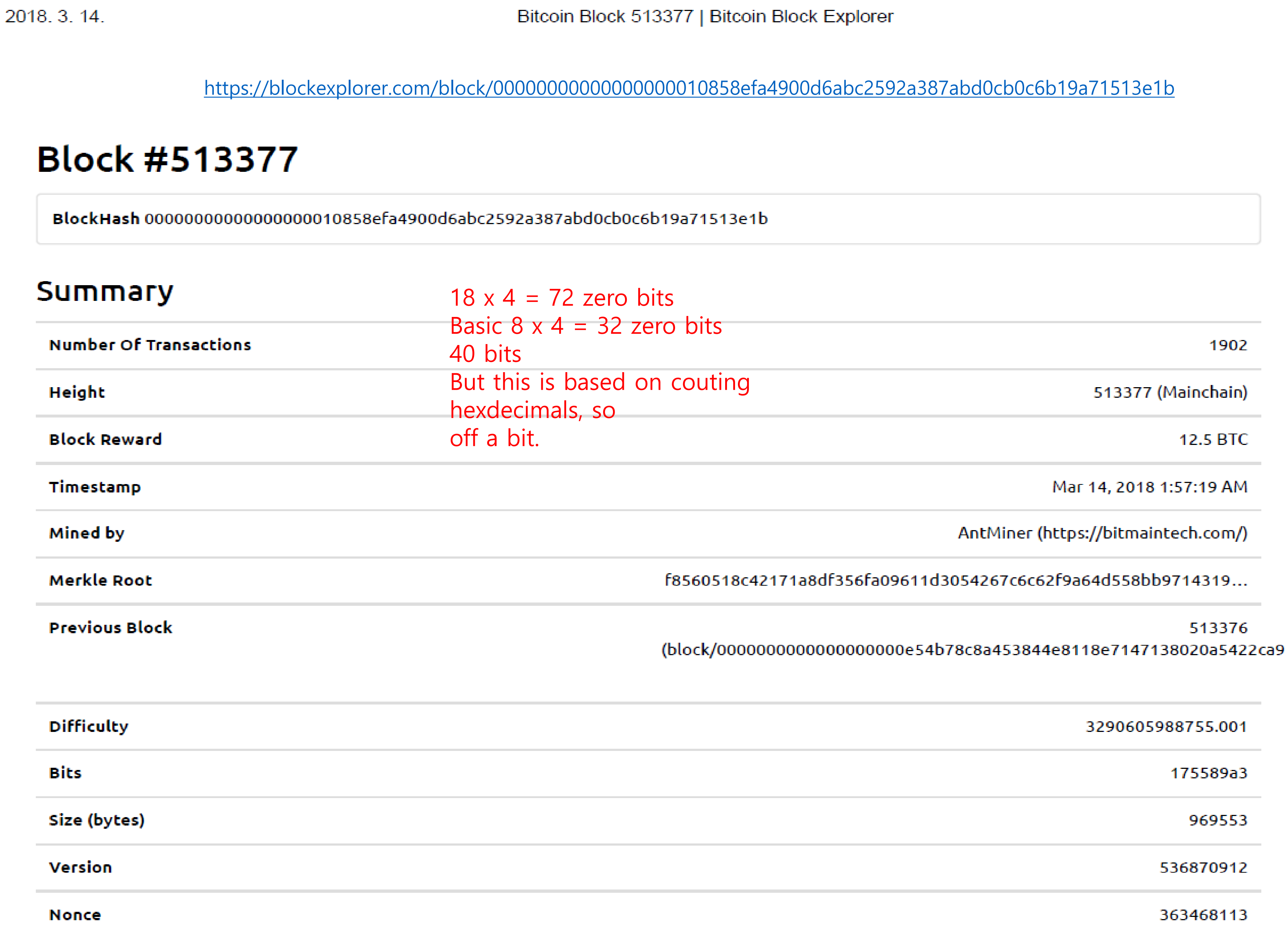
256 bit is 64 hexadecimal string

A hash value
2d711642b726b04401627ca9fbac32f5c8530fb1903cc4db02258717921a4881

A good hash value which passes the condition that the first four digits are 0s.
0000f727854b50bb95c054b39c1fe5c92e5ebcfa4bcb5dc279f56aa96a365e5a

c = the set of any hash values = 2^256
a = the set of wanted hash values= 2^(256 – 16) = 2^240

P1 = a/c = 2^-16 = 1/(2^16) ~ 1/64000



2018. 3. 14. Bitcoin Block 513377 | Bitcoin Block Explorer

https://blockexplorer.com/block/000000000000000000010858efa4900d6abc2592a387abd0cb0c6b19a71513e1b

Block #513377

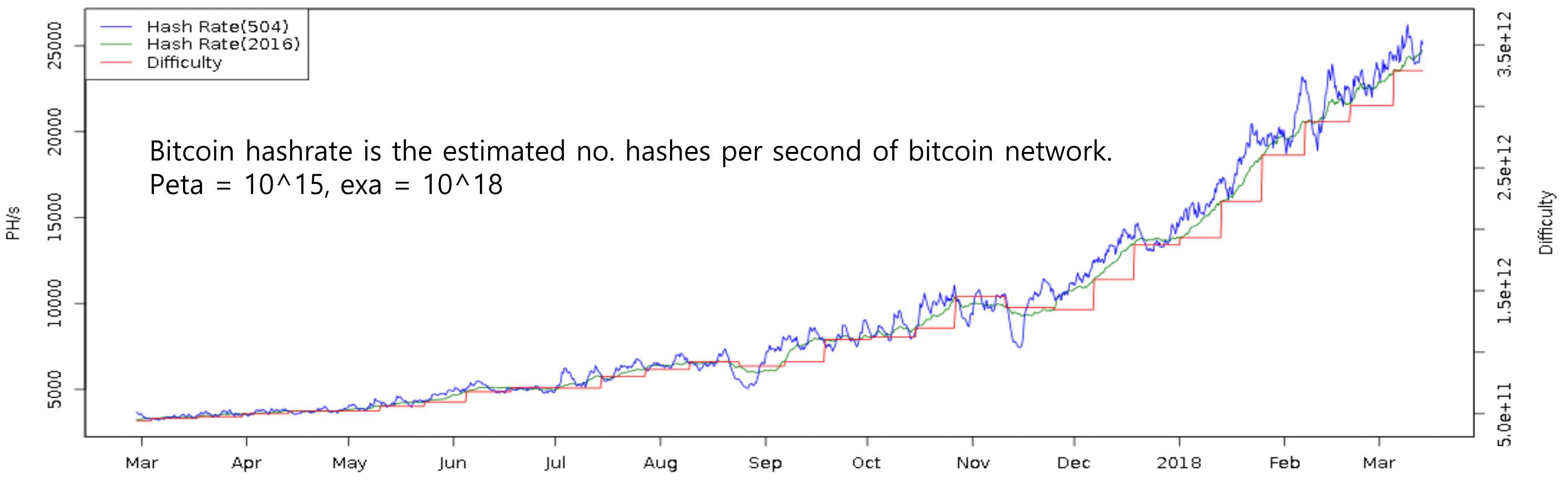
BlockHash 000000000000000000010858efa4900d6abc2592a387abd0cb0c6b19a71513e1b

Summary

18 x 4 = 72 zero bits
Basic 8 x 4 = 32 zero bits
40 bits
But this is based on couting hexdecimals, so
off a bit.

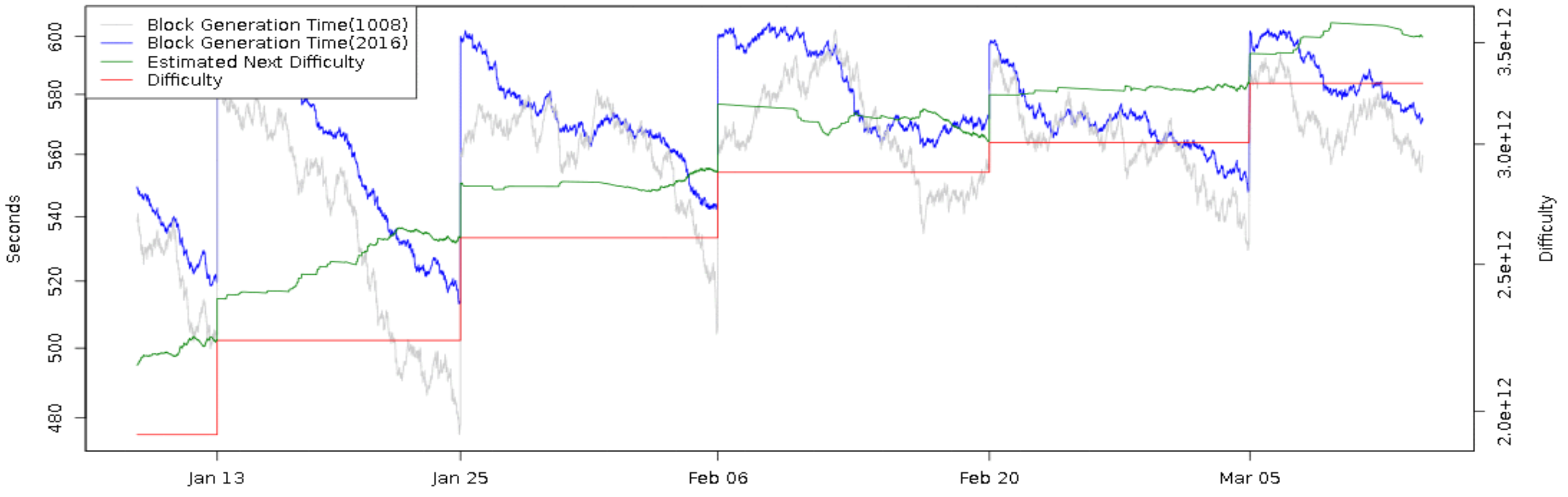
Number Of Transactions	1902
Height	513377 (Mainchain)
Block Reward	12.5 BTC
Timestamp	Mar 14, 2018 1:57:19 AM
Mined by	AntMiner (https://bitmaintech.com/)
Merkle Root	f8560518c42171a8df356fa09611d3054267c6c62f9a64d558bb9714319...
Previous Block	513376 (block/000000000000000000e54b78c8a453844e8118e7147138020a5422ca9
Difficulty	3290605988755.001
Bits	175589a3
Size (bytes)	969553
Version	536870912
Nonce	363468113

Bitcoin Hash Rate vs Difficulty (9 Months)

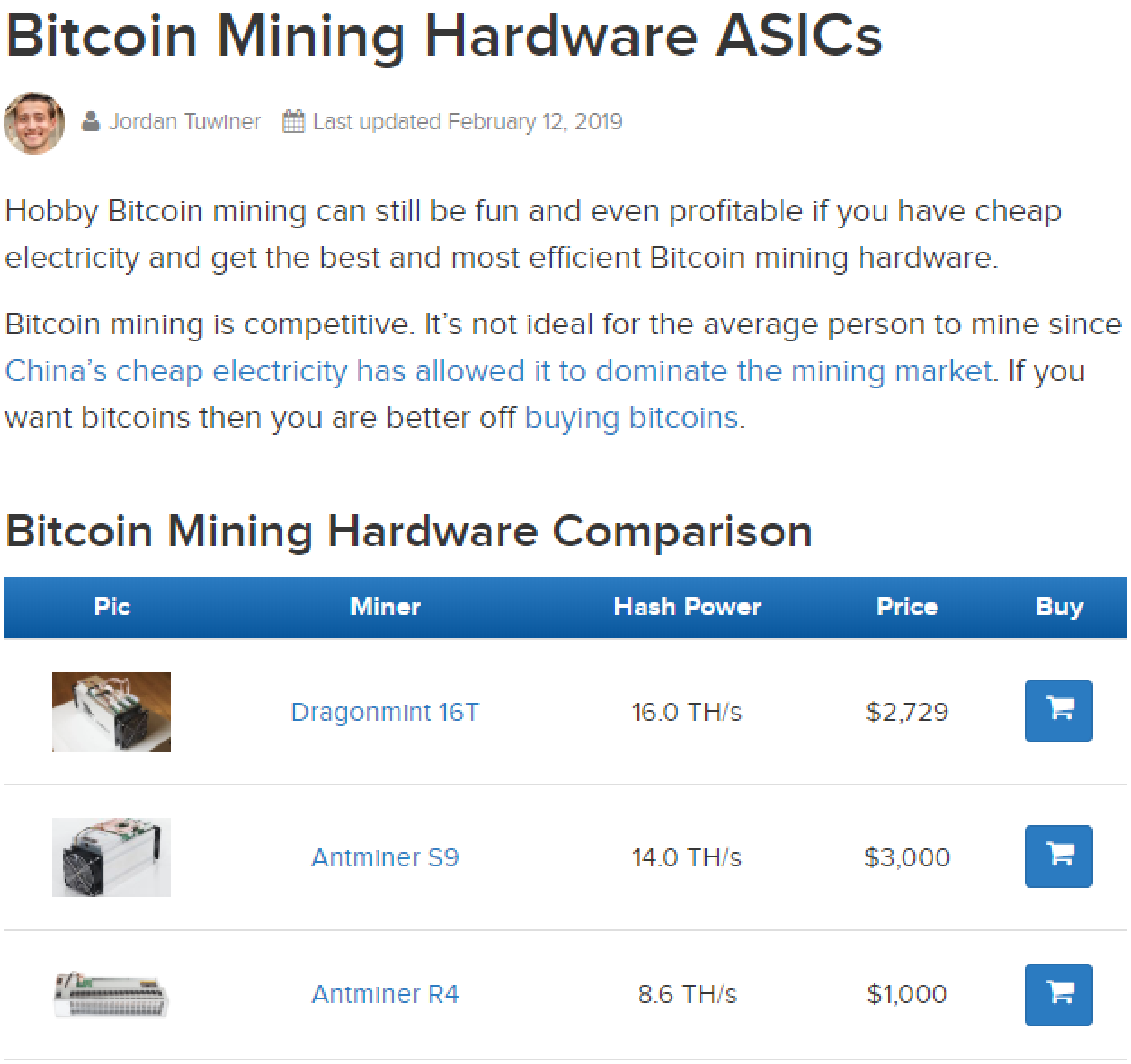


Bitcoin hashrate is the estimated no. hashes per second of bitcoin network.
Peta = 10^15, exa = 10^18

Bitcoin Block Generation Time vs Difficulty



https://bitcoinwisdom.com/bitcoin/difficulty



Bitcoin Mining Hardware ASICs

Jordan Tuwiner   Last updated February 12, 2019

Hobby Bitcoin mining can still be fun and even profitable if you have cheap electricity and get the best and most efficient Bitcoin mining hardware.

Bitcoin mining is competitive. It's not ideal for the average person to mine since China's cheap electricity has allowed it to dominate the mining market. If you want bitcoins then you are better off buying bitcoins.

Bitcoin Mining Hardware Comparison

Pic	Miner	Hash Power	Price	Buy
	Dragonmint 16T	16.0 TH/s	$2,729	🛒
	Antminer S9	14.0 TH/s	$3,000	🛒
	Antminer R4	8.6 TH/s	$1,000	🛒

Any new node can join.

- In the public blockchain network, anyone can join and become a guard (miner).

Miners are everywhere.

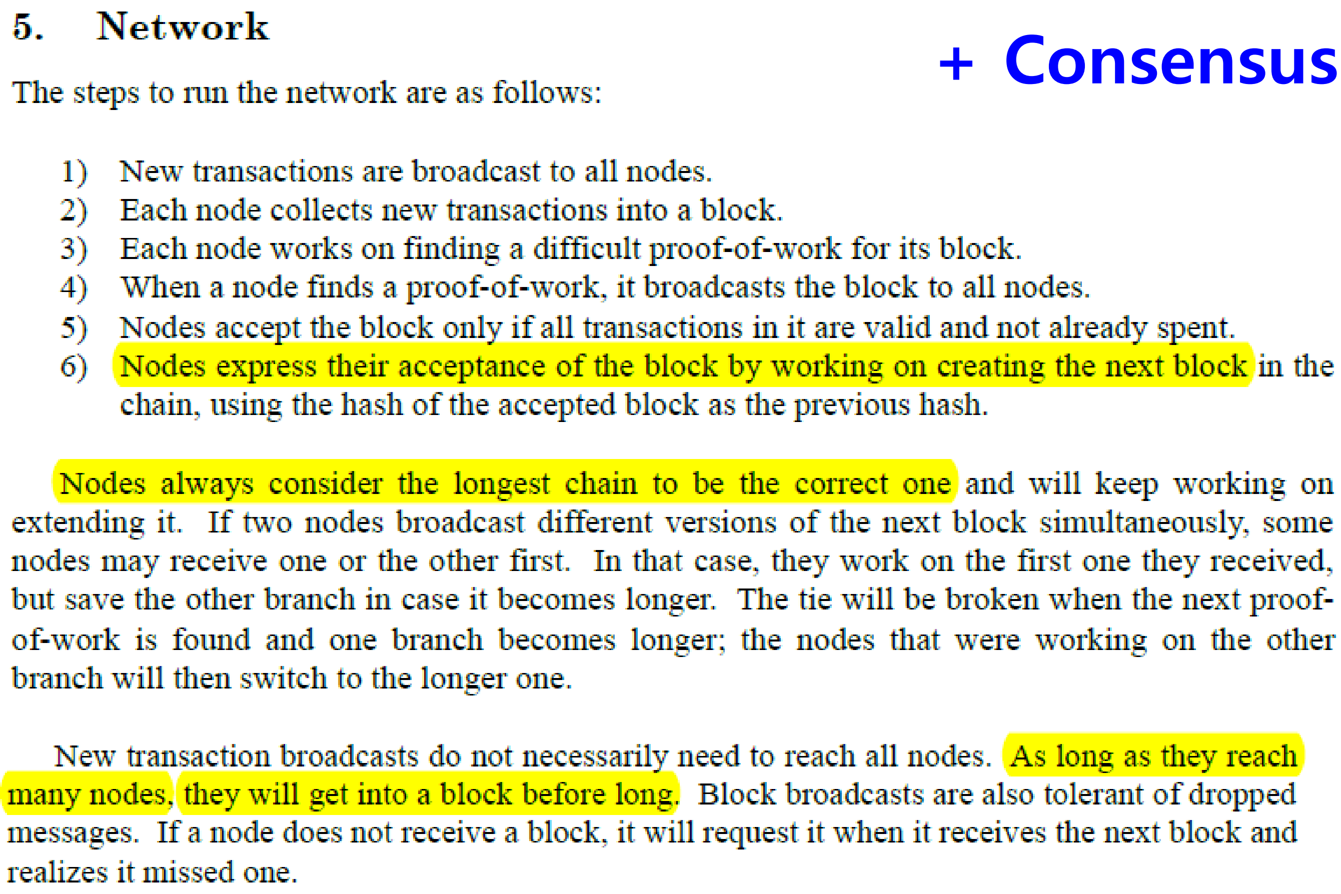
- Each block is formed by a node.
- A node gathers TXs, validates them, forms a block.
- As a reward, the node which formed a block is given a block mining reward (e.g. 12.5 BTC).
- Thus, they are called miners.

500th block
Formed in
America

502th block
Formed in
China

501th block
Formed in
Korea

+ Consensus Rule



5. Network

The steps to run the network are as follows:

1) New transactions are broadcast to all nodes.
2) Each node collects new transactions into a block.
3) Each node works on finding a difficult proof-of-work for its block.
4) When a node finds a proof-of-work, it broadcasts the block to all nodes.
5) Nodes accept the block only if all transactions in it are valid and not already spent.
6) Nodes express their acceptance of the block by working on creating the next block in the chain, using the hash of the accepted block as the previous hash.

Nodes always consider the longest chain to be the correct one and will keep working on extending it. If two nodes broadcast different versions of the next block simultaneously, some nodes may receive one or the other first. In that case, they work on the first one they received, but save the other branch in case it becomes longer. The tie will be broken when the next proof-of-work is found and one branch becomes longer; the nodes that were working on the other branch will then switch to the longer one.

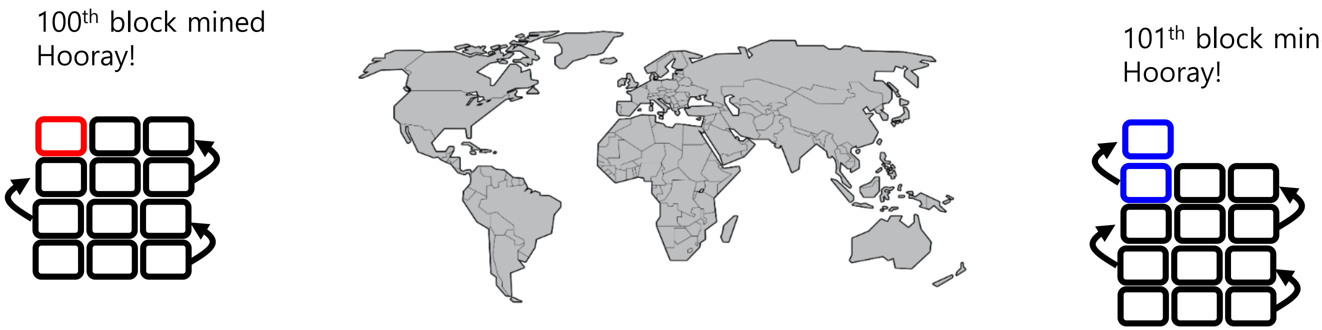
New transaction broadcasts do not necessarily need to reach all nodes. As long as they reach many nodes, they will get into a block before long. Block broadcasts are also tolerant of dropped messages. If a node does not receive a block, it will request it when it receives the next block and realizes it missed one.

- Are there any guarantee for transactions to be included into blocks?
- With a large incentive(tx fee), a tx can be put on high priority, but if the production rate of txs is higher than the service rate, then there must be some transactions not to end up in the blockchain.

Consensus mechanism plays the key role in blockchain.

- Multiple different chains are possible, as miners work independently.
- When any two chains are available, miners choose the longer one!

Which one wins when there are two chains announced?



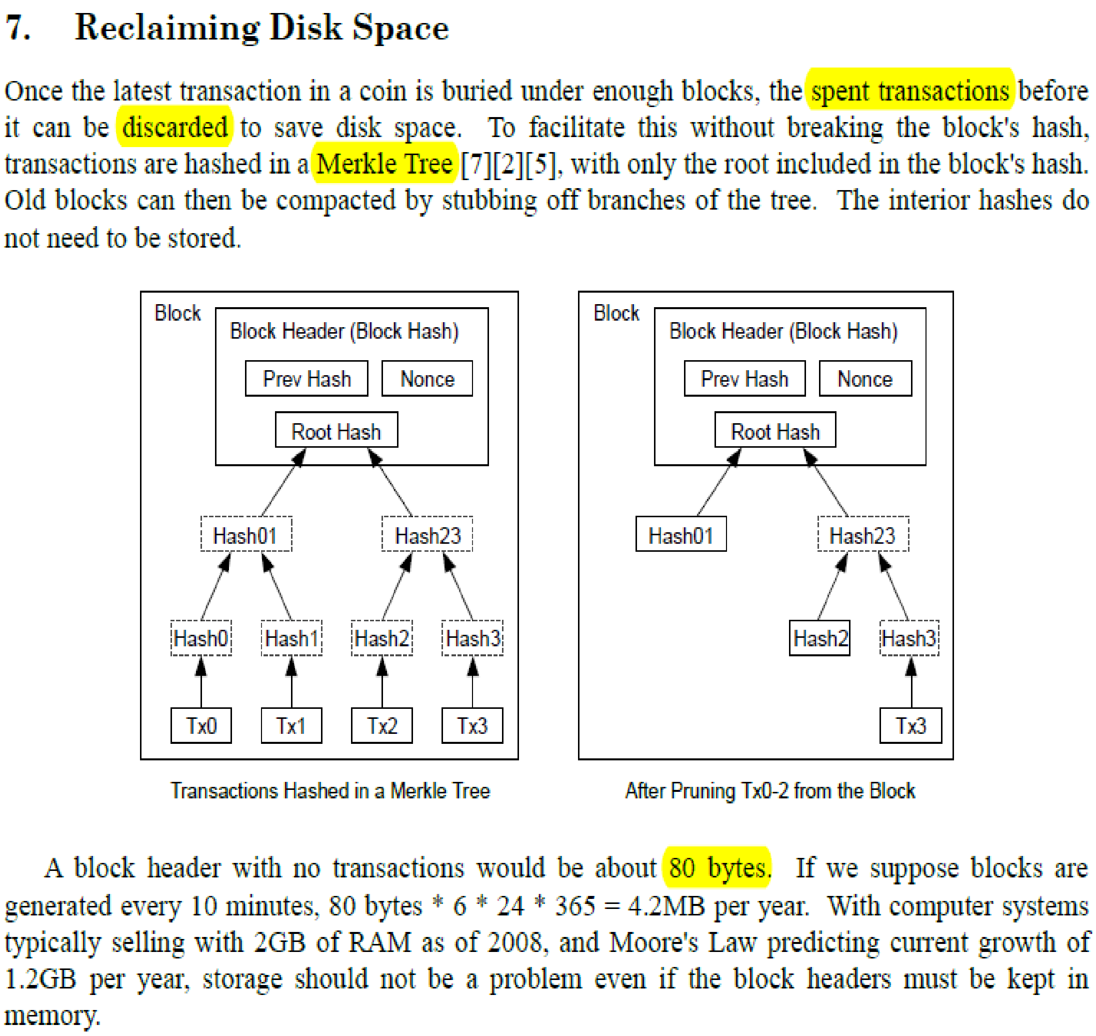
Longer chain wins!

Blockchain Scalability

- Use Merkle tree and save disk space
- Save the blockhash in the header.
- Those tree branches recording past transactions are erased but the hash values are kept.
- 80 byte Blockheader

.

1. Prev hash: 256 bit = 2^8 = 2^5*(2^3) = 2^5 Bytes = 32 Bytes
2. Roothash = 32 Bytes
3. Nonce = 4 Bytes = 32 bit
4. Time
5. Difficulty
6. version



7. Reclaiming Disk Space

Once the latest transaction in a coin is buried under enough blocks, the spent transactions before it can be discarded to save disk space. To facilitate this without breaking the block's hash, transactions are hashed in a Merkle Tree [7][2][5], with only the root included in the block's hash. Old blocks can then be compacted by stubbing off branches of the tree. The interior hashes do not need to be stored.

Transactions Hashed in a Merkle Tree

After Pruning Tx0-2 from the Block

A block header with no transactions would be about 80 bytes. If we suppose blocks are generated every 10 minutes, 80 bytes * 6 * 24 * 365 = 4.2MB per year. With computer systems typically selling with 2GB of RAM as of 2008, and Moore's Law predicting current growth of 1.2GB per year, storage should not be a problem even if the block headers must be kept in memory.

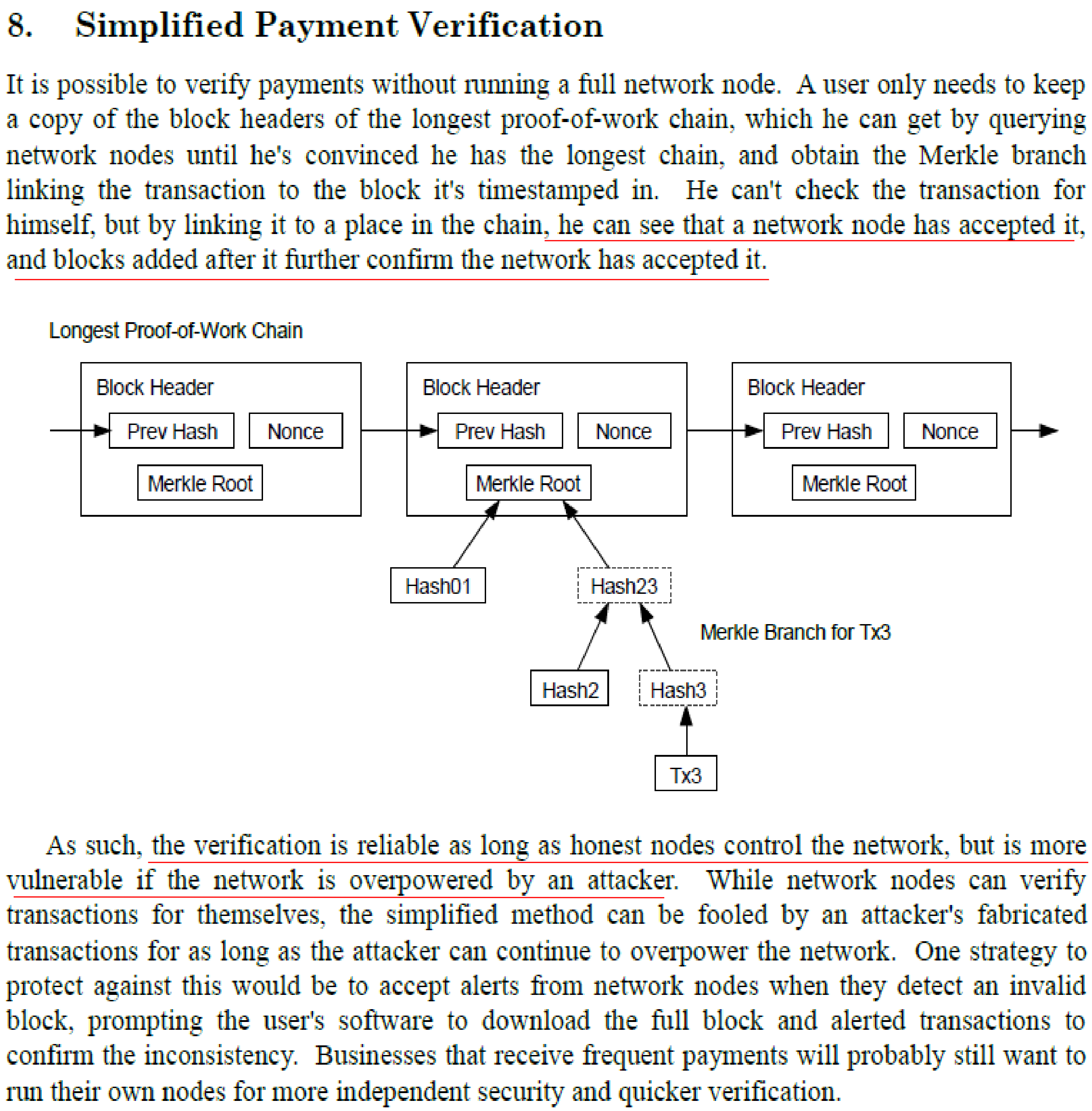
80 Byte Block Header

Bytes	Name	Data Type	Description
4	version	int32_t	The block version number indicates which set of block validation rules to follow. See the list of block versions below.
32	previous block header hash	char[32]	A SHA256(SHA256()) hash in internal byte order of the previous block's header. This ensures no previous block can be changed without also changing this block's header.
32	merkle roothash	char[32]	A SHA256(SHA256()) hash in internal byte order. The merkle root is derived from the hashes of all transactions included in this block, ensuring that none of those transactions can be modified without modifying the header. See the merkle trees section below.
4	time	uint32_t	The block time is a Unix epoch time when the miner started hashing the header (according to the miner). Must be strictly greater than the median time of the previous 11 blocks. Full nodeswill not accept blocks with headers more than two hours in the future according to their clock.
4	nBits	uint32_t	An encoded version of the target threshold this block's header hash must be less than or equal to. See the nBits format described below.
4	nonce	uint32_t	An arbitrary number miners change to modify the header hash in order to produce a hash less than or equal to the target threshold. If all 32-bit values are tested, the time can be updated or the coinbase transaction can be changed and the merkle root updated.

Source : https://bitcoin.org/en/developer-reference#block-headers

Longest chain is trusted, why?

- A headers-only chain can be used for simplification!
- For full verification, one can download the full chain with full transaction record.
- But there is no guarantee with regard to chain's validity even for the full chains are used, as attacks are possible at any time and thus the network is vulnerable whenever network is overpowered by attackers.
- There is no guarantee that one obtains the longest chain by querying either.
- But when one has been around for sufficiently long time, then it shall not be difficult for one to obtain the longest chain.
- Things work as long as honest nodes control the network.
- But when there are nodes complaining inconsistencies and discontinuities, it becomes the time to stop believing the integrity of even the longest status-quo chain.



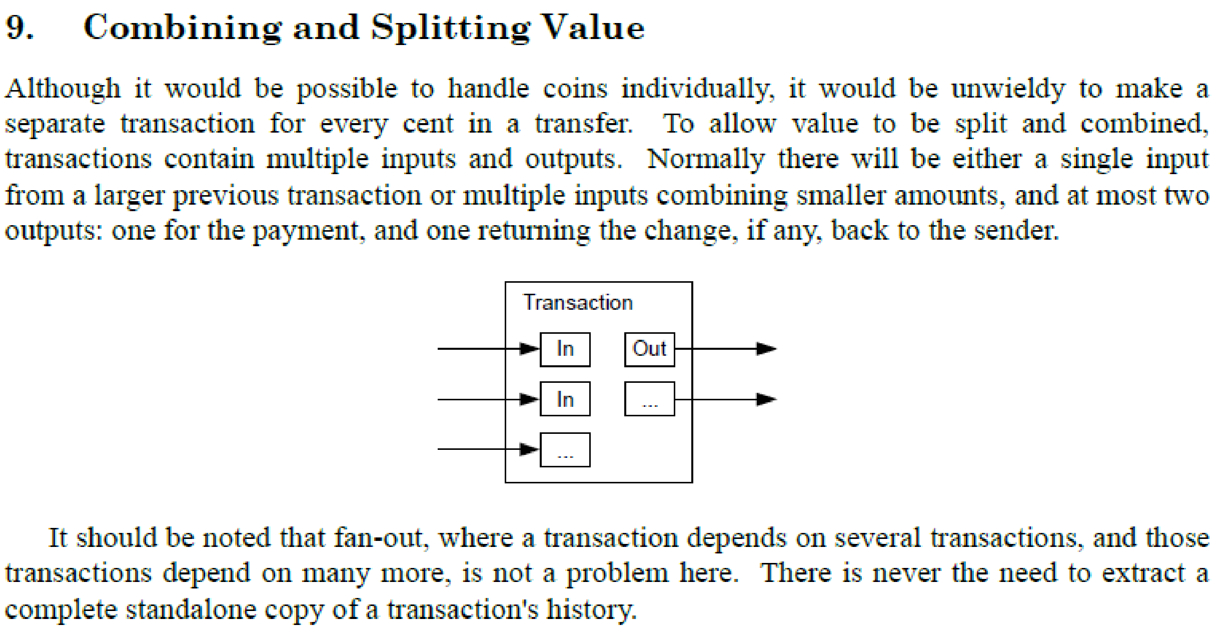
8. Simplified Payment Verification

It is possible to verify payments without running a full network node. A user only needs to keep a copy of the block headers of the longest proof-of-work chain, which he can get by querying network nodes until he's convinced he has the longest chain, and obtain the Merkle branch linking the transaction to the block it's timestamped in. He can't check the transaction for himself, but by linking it to a place in the chain, he can see that a network node has accepted it, and blocks added after it further confirm the network has accepted it.

As such, the verification is reliable as long as honest nodes control the network, but is more vulnerable if the network is overpowered by an attacker. While network nodes can verify transactions for themselves, the simplified method can be fooled by an attacker's fabricated transactions for as long as the attacker can continue to overpower the network. One strategy to protect against this would be to accept alerts from network nodes when they detect an invalid block, prompting the user's software to download the full block and alerted transactions to confirm the inconsistency. Businesses that receive frequent payments will probably still want to run their own nodes for more independent security and quicker verification.

Payment and changes

- How to get the change?



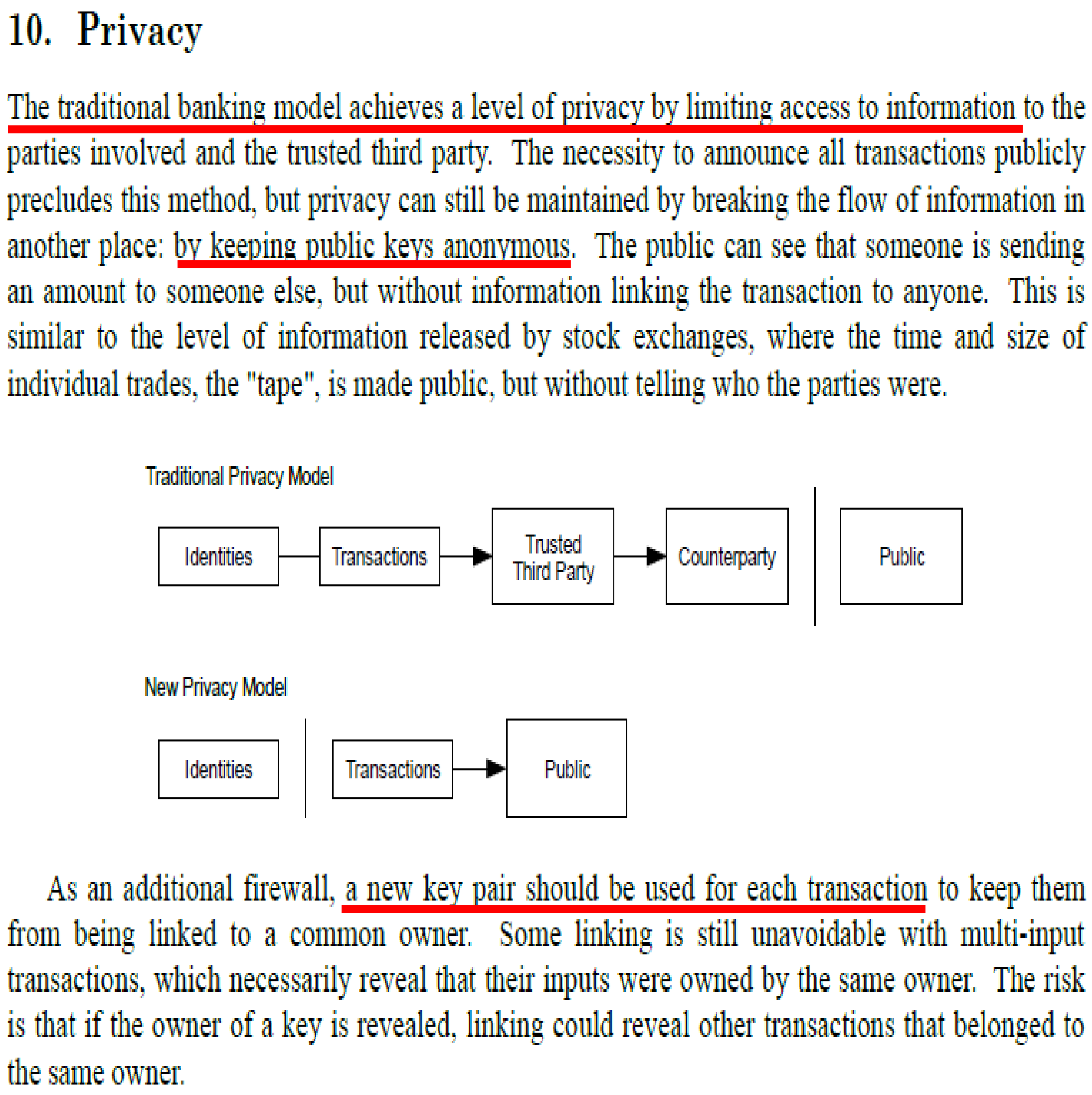
9. Combining and Splitting Value

Although it would be possible to handle coins individually, it would be unwieldy to make a separate transaction for every cent in a transfer. To allow value to be split and combined, transactions contain multiple inputs and outputs. Normally there will be either a single input from a larger previous transaction or multiple inputs combining smaller amounts, and at most two outputs: one for the payment, and one returning the change, if any, back to the sender.

It should be noted that fan-out, where a transaction depends on several transactions, and those transactions depend on many more, is not a problem here. There is never the need to extract a complete standalone copy of a transaction's history.

Privacy, by Anonymous Pub Key

- Blockchain is published.
- Privacy is maintained by keeping public key anonymous!
- Additional privacy by using new public key per transaction!



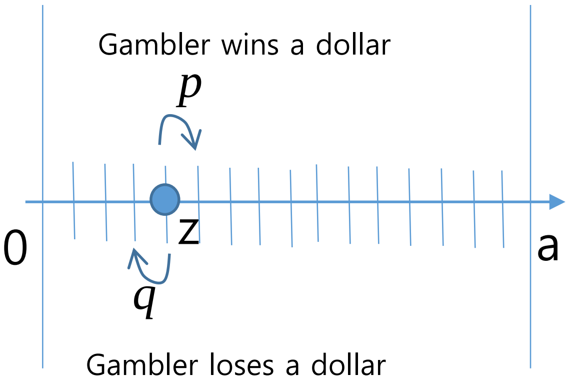
10. Privacy

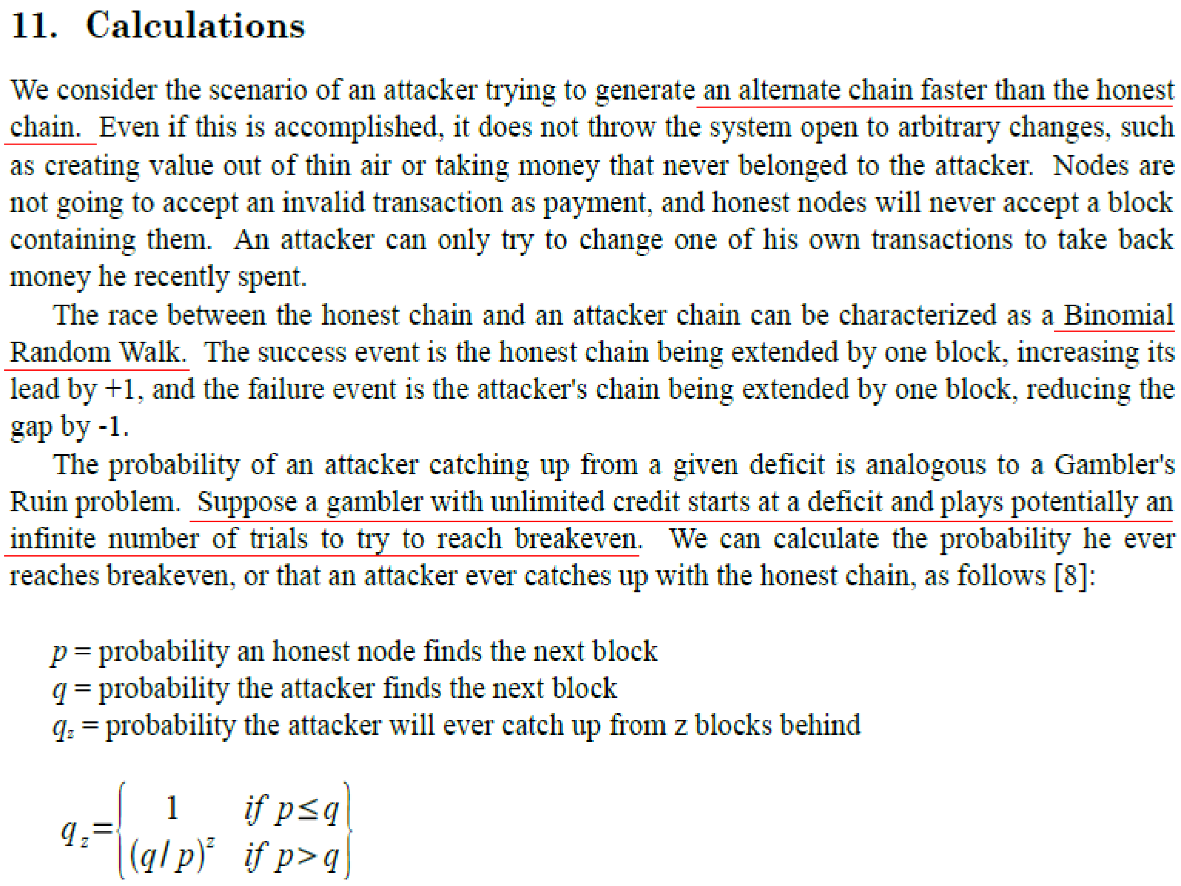
The traditional banking model achieves a level of privacy by limiting access to information to the parties involved and the trusted third party. The necessity to announce all transactions publicly precludes this method, but privacy can still be maintained by breaking the flow of information in another place: by keeping public keys anonymous. The public can see that someone is sending an amount to someone else, but without information linking the transaction to anyone. This is similar to the level of information released by stock exchanges, where the time and size of individual trades, the "tape", is made public, but without telling who the parties were.

As an additional firewall, a new key pair should be used for each transaction to keep them from being linked to a common owner. Some linking is still unavoidable with multi-input transactions, which necessarily reveal that their inputs were owned by the same owner. The risk is that if the owner of a key is revealed, linking could reveal other transactions that belonged to the same owner.

How Difficulty to Attack?

- What happens when the attacker's chain dominates the honest chain?
- The best attack that can be made is to alter its own transaction.
- Namely, reclaim what he has paid.





11. Calculations

We consider the scenario of an attacker trying to generate an alternate chain faster than the honest chain. Even if this is accomplished, it does not throw the system open to arbitrary changes, such as creating value out of thin air or taking money that never belonged to the attacker. Nodes are not going to accept an invalid transaction as payment, and honest nodes will never accept a block containing them. An attacker can only try to change one of his own transactions to take back money he recently spent.

The race between the honest chain and an attacker chain can be characterized as a Binomial Random Walk. The success event is the honest chain being extended by one block, increasing its lead by +1, and the failure event is the attacker's chain being extended by one block, reducing the gap by -1.

The probability of an attacker catching up from a given deficit is analogous to a Gambler's Ruin problem. Suppose a gambler with unlimited credit starts at a deficit and plays potentially an infinite number of trials to try to reach breakeven. We can calculate the probability he ever reaches breakeven, or that an attacker ever catches up with the honest chain, as follows [8]:

p = probability an honest node finds the next block
q = probability the attacker finds the next block
q_z = probability the attacker will ever catch up from z blocks behind

$$q_z = \left\{ \begin{matrix} 1 & \text{if } p \le q \\ (q/p)^z & \text{if } p > q \end{matrix} \right\}$$

Double Spend Race Attack

A announces a TX showing A sends B 1 BTC at the end of time t0.

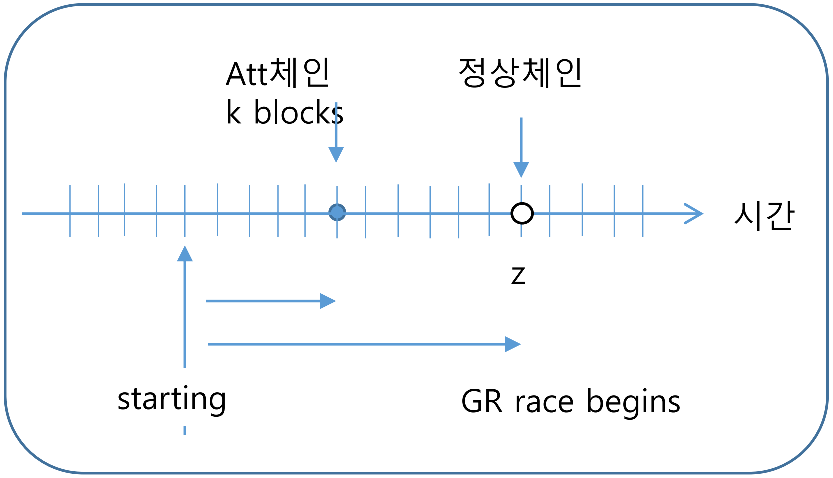
This TX gets into a block (1 confirmation) at t1.

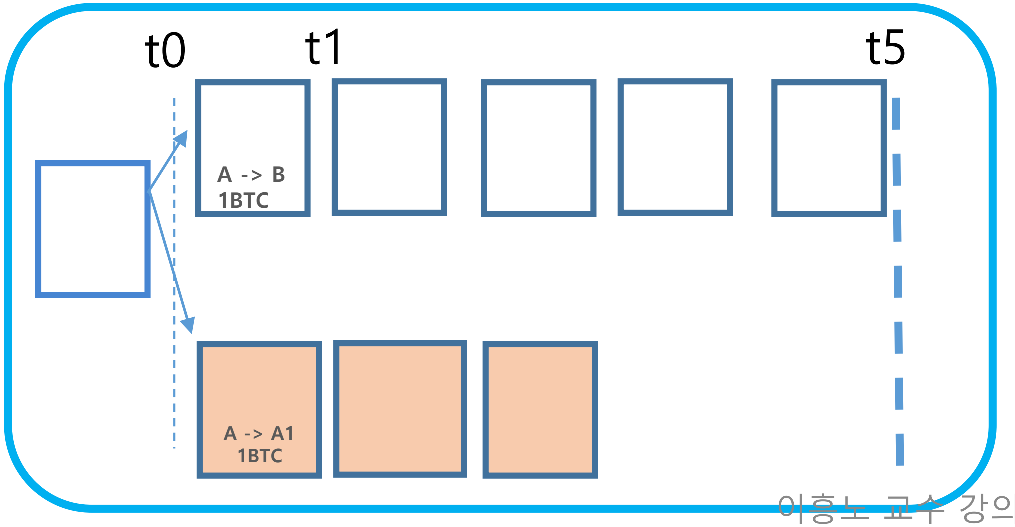
B waits until he gets the 5th confirmation at t5.

A is the attacker.

A starts preparing a double spend attack at t0. Namely, A grows its own chain. His chain has replaced the TX A->B 1BTC with a TX, A -> A1 1BTC. A1 is another public key of A.

At t5, A has mined 3 blocks and needs to decide if he continues to grow his own chain or not.





Attack Success Probability(q,z)

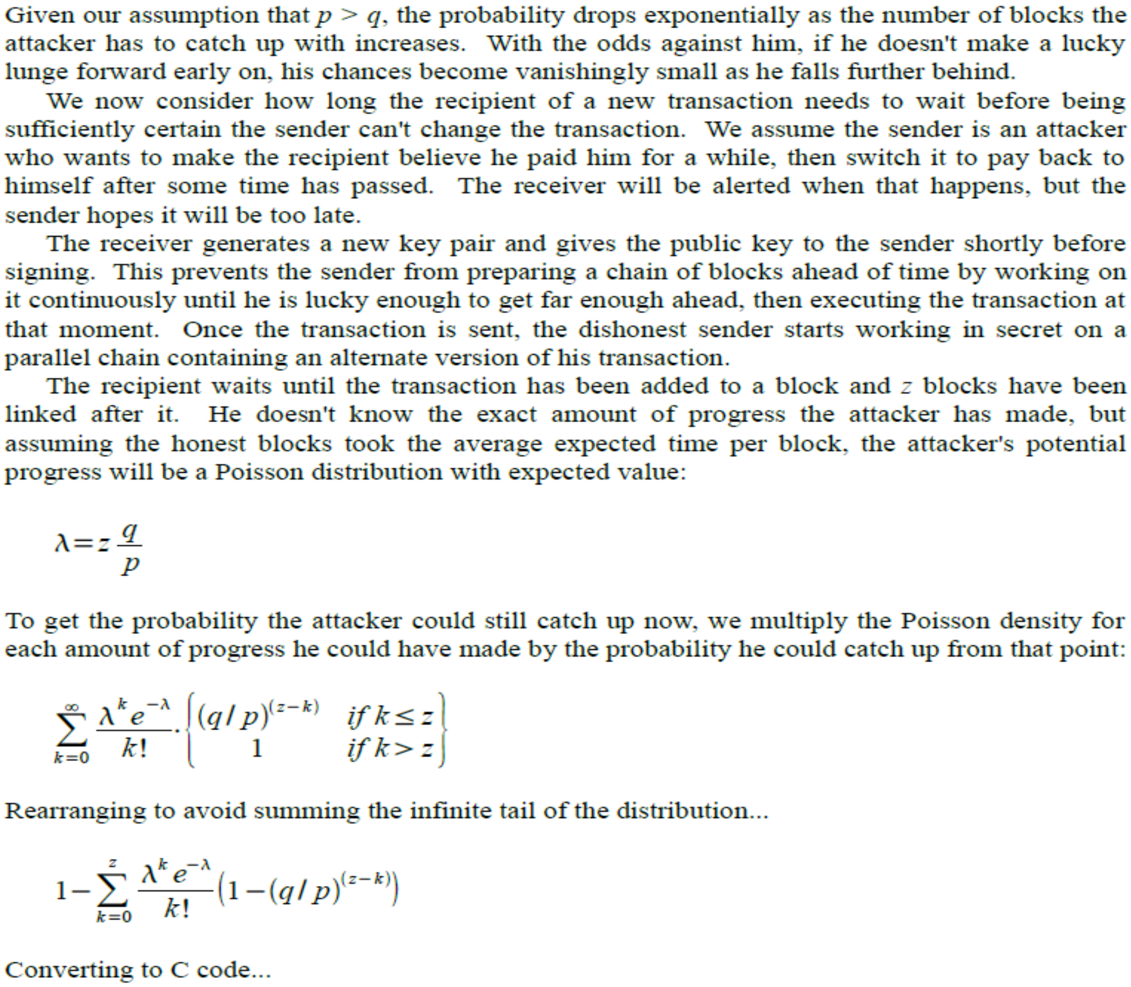
$$\sim\sum_{k=0}^{\infty}\begin{Bmatrix}(q/p)^{z-k} & k<z\\ 1 & k\geq z\end{Bmatrix}\text{Poisson}(\lambda=zq/p)$$

λ is the average number of blocks that the attacker mines in z unit of time

$$=\sum_{k=0}^{\infty}\begin{Bmatrix}(q/p)^{z-k} & k<z\\ 1 & k\geq z\end{Bmatrix}\frac{(zq/p)^{k}\,e^{-zq/p}}{k!}$$

Attacker is the payer, fooling the payee!

- Given z blocks added. Assumed average time took by the honest nodes.



Given our assumption that $p > q$, the probability drops exponentially as the number of blocks the attacker has to catch up with increases. With the odds against him, if he doesn't make a lucky lunge forward early on, his chances become vanishingly small as he falls further behind.

We now consider how long the recipient of a new transaction needs to wait before being sufficiently certain the sender can't change the transaction. We assume the sender is an attacker who wants to make the recipient believe he paid him for a while, then switch it to pay back to himself after some time has passed. The receiver will be alerted when that happens, but the sender hopes it will be too late.

The receiver generates a new key pair and gives the public key to the sender shortly before signing. This prevents the sender from preparing a chain of blocks ahead of time by working on it continuously until he is lucky enough to get far enough ahead, then executing the transaction at that moment. Once the transaction is sent, the dishonest sender starts working in secret on a parallel chain containing an alternate version of his transaction.

The recipient waits until the transaction has been added to a block and z blocks have been linked after it. He doesn't know the exact amount of progress the attacker has made, but assuming the honest blocks took the average expected time per block, the attacker's potential progress will be a Poisson distribution with expected value:

$$\lambda = z\frac{q}{p}$$

To get the probability the attacker could still catch up now, we multiply the Poisson density for each amount of progress he could have made by the probability he could catch up from that point:

$$\sum_{k=0}^{\infty} \frac{\lambda^k e^{-\lambda}}{k!} \cdot \left\{ \begin{array}{ll} (q/p)^{(z-k)} & if\ k \le z \\ 1 & if\ k > z \end{array} \right\}$$

Rearranging to avoid summing the infinite tail of the distribution...

$$1 - \sum_{k=0}^{z} \frac{\lambda^k e^{-\lambda}}{k!} \left(1 - (q/p)^{(z-k)}\right)$$

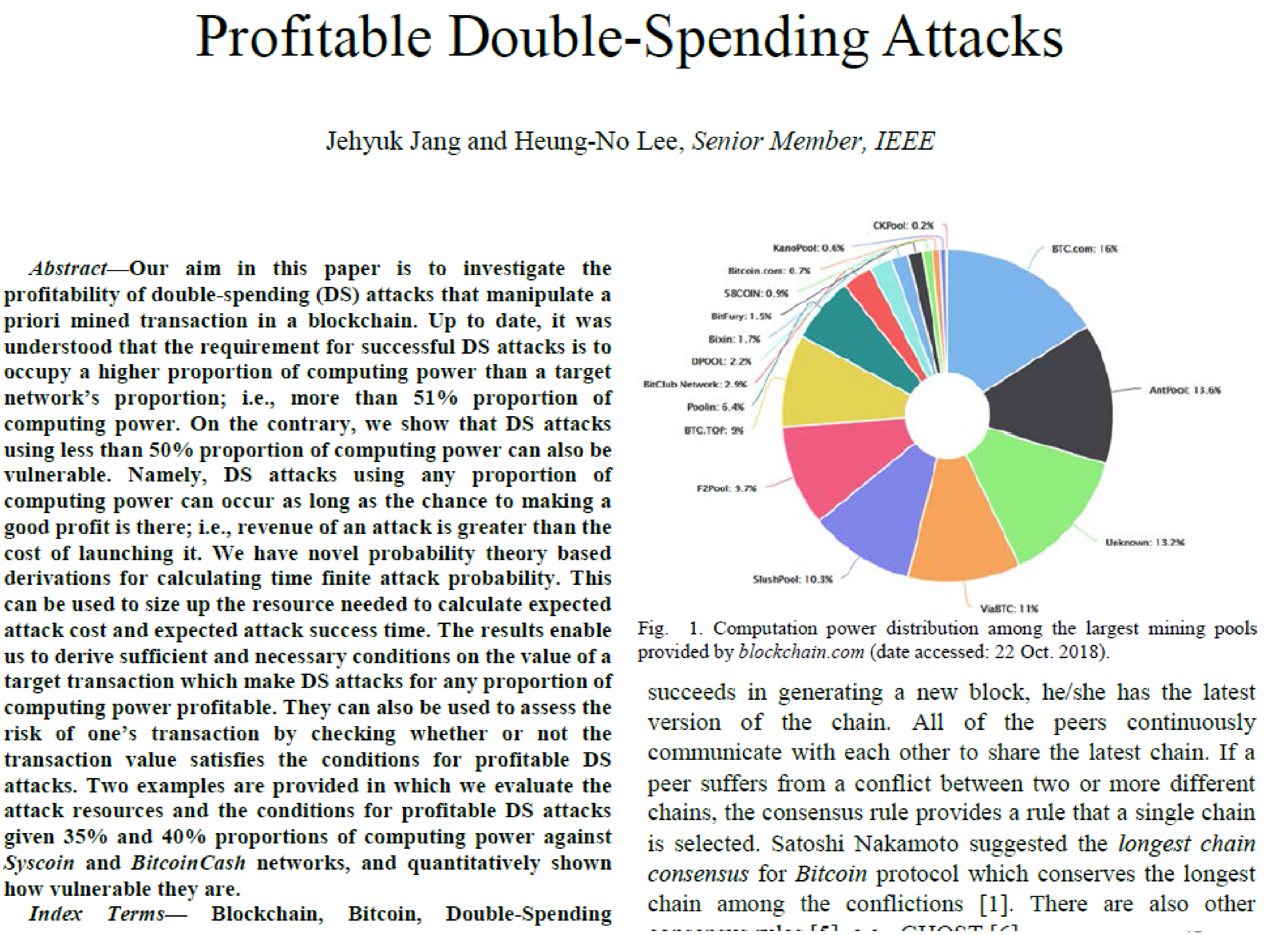
Converting to C code...

Two Novel Results

1. Profitable Double-Spending Attacks
2. Scalable DeSecure Blockchains

Profitable DS Attacks

https://arxiv.org/abs/1903.01711



Profitable Double-Spending Attacks

Jehyuk Jang and Heung-No Lee, *Senior Member, IEEE*

Abstract—Our aim in this paper is to investigate the profitability of double-spending (DS) attacks that manipulate a priori mined transaction in a blockchain. Up to date, it was understood that the requirement for successful DS attacks is to occupy a higher proportion of computing power than a target network's proportion; i.e., more than 51% proportion of computing power. On the contrary, we show that DS attacks using less than 50% proportion of computing power can also be vulnerable. Namely, DS attacks using any proportion of computing power can occur as long as the chance to making a good profit is there; i.e., revenue of an attack is greater than the cost of launching it. We have novel probability theory based derivations for calculating time finite attack probability. This can be used to size up the resource needed to calculate expected attack cost and expected attack success time. The results enable us to derive sufficient and necessary conditions on the value of a target transaction which make DS attacks for any proportion of computing power profitable. They can also be used to assess the risk of one's transaction by checking whether or not the transaction value satisfies the conditions for profitable DS attacks. Two examples are provided in which we evaluate the attack resources and the conditions for profitable DS attacks given 35% and 40% proportions of computing power against *Syscoin* and *BitcoinCash* networks, and quantitatively shown how vulnerable they are.

Index Terms— Blockchain, Bitcoin, Double-Spending

Fig. 1. Computation power distribution among the largest mining pools provided by *blockchain.com* (date accessed: 22 Oct. 2018).

succeeds in generating a new block, he/she has the latest version of the chain. All of the peers continuously communicate with each other to share the latest chain. If a peer suffers from a conflict between two or more different chains, the consensus rule provides a rule that a single chain is selected. Satoshi Nakamoto suggested the *longest chain consensus* for *Bitcoin* protocol which conserves the longest chain among the conflictions [1]. There are also other

Definitions (Mary the attacker)

Parameter	Description
p_A	Mary's portion of computing power (0~100%)
p_H	Network's portion of computing power (p_A+p_H=100%)
λ_H	Miners' average block generation speed [blocks/sec]
N_{BC}	Block confirmation number
t_{cut}	Attack cut time for cut loss
T_{AS}	Attack success time (random variable)

※ Mary's average block generation speed $\lambda_A = \lambda_H \, p_A/(1 - p_A)$.

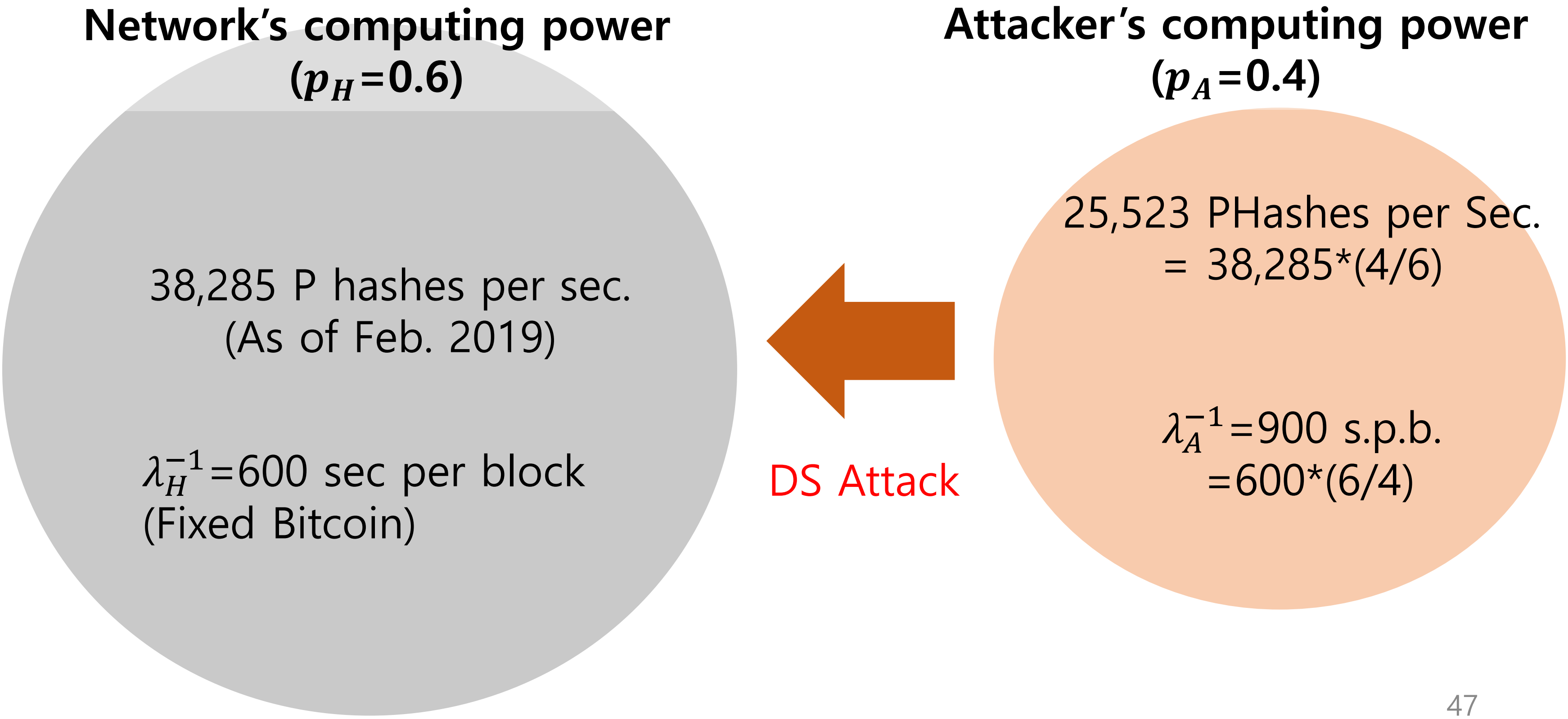
Definition. A DS attack succeeds if

- Target transaction has got N_{BC} blocks confirmed ,
- Mary's chain has grown longer than the public chain, and
- **the above two conditions have been satisfied within a cut time** t_{cut}.

Meaning of Computing Powers

- p_H = 0.6 vs. p_A = 0.4

  (Network 100% vs. Attacker 66.6%)



Network's computing power
(p_H=0.6)

38,285 P hashes per sec.
(As of Feb. 2019)

λ_H^{-1}=600 sec per block
(Fixed Bitcoin)

DS Attack

Attacker's computing power
(p_A=0.4)

25,523 PHashes per Sec.
= 38,285*(4/6)

λ_A^{-1}=900 s.p.b.
=600*(6/4)

Our results (Main)

Definition. A DS attack is *profitable* if and only if the expected revenue is greater than the expected cost.

※ Revenue: cheating value of target transaction
※ Cost: operating expense for computing hash functions

Theorem. For all attacker's fractions of computing power p_A (1%~99%), DS attacks are profitable if the value V of target transaction is greater than

$$V_{Suf.}(p_A; N_{BC}) = \gamma'(p_A)\frac{\lambda_H p_A \mathrm{E}[T_{AS}]}{(1-p_A)\mathbb{P}_{AS}(p_A, t_{cut}; N_{BC})}.$$

"Even though p_A **is less than** 50% ,
Mary can make a profitable DS attack."

Our results (2)

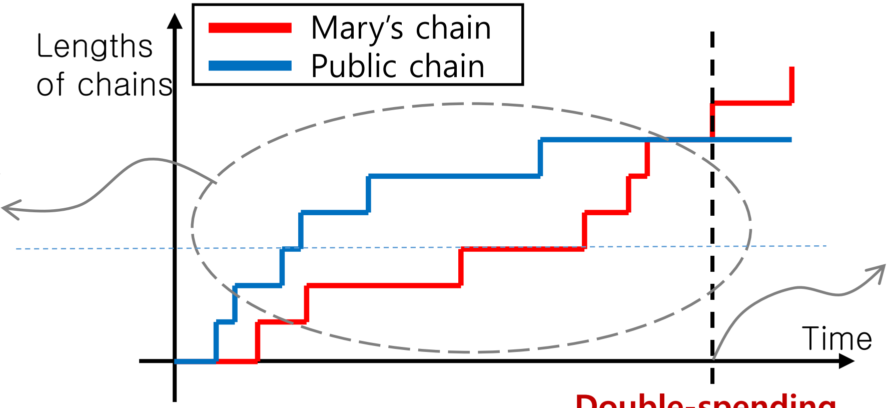
Theorem. A DS attack using p_A less than 50% are profitable only if a finite cut time $t_{cut} < \infty$ is given.

- The odd that Attacker's chain NEVER catch up w/ the network chain is nonzero.
- If there is no time limit and fake chain does not catch up with the public chain, the operating expense is infinity.
- To avoid this, attacker should give up her attack at a certain **cut time** to cut loss.

Our results (3)

“We provide the probability density function of attack success time.”

- DS attack is modeled as a competition of two Poisson processes.
- There are infinitely many combinations for the two Poisson processes to compete which give a DS attack success at the end.
- There are infinite possibilities Mary’s chain can catch up with the public chain.
- We came up with a novel way of calculating probabilities, using combinatorics and generating functions.

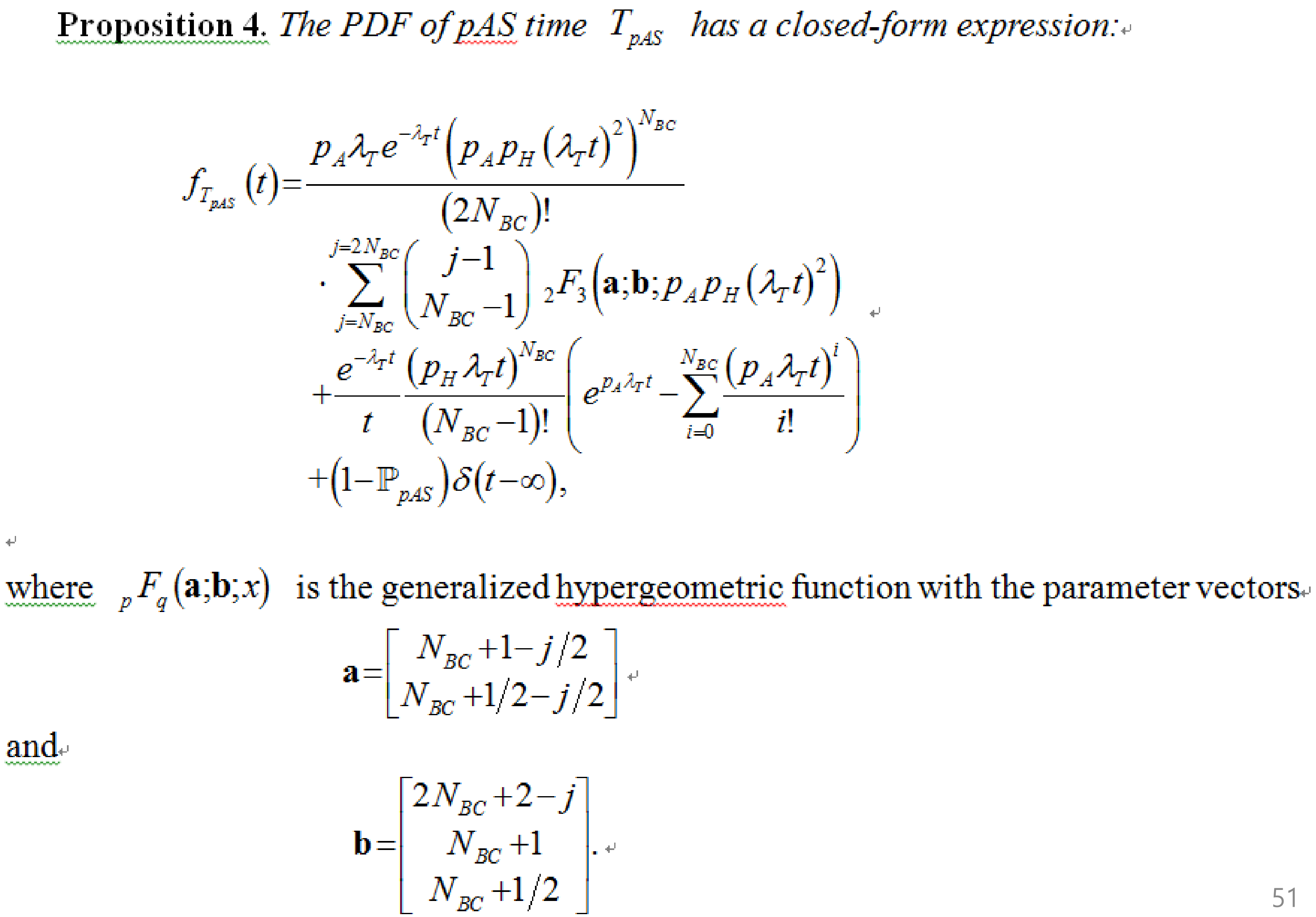


Infinitely many combinations

Attack success time is a random variable.

Double-spending Attack Success!

Our results (3)



Proposition 4. *The PDF of pAS time* T_{pAS} *has a closed-form expression:*

$$
\begin{aligned}
f_{T_{pAS}}(t) = {} & \frac{p_A \lambda_T e^{-\lambda_T t}\left(p_A p_H (\lambda_T t)^2\right)^{N_{BC}}}{(2N_{BC})!} \\
& \cdot \sum_{j=N_{BC}}^{j=2N_{BC}} \binom{j-1}{N_{BC}-1} {}_2F_3\left(\mathbf{a};\mathbf{b};p_A p_H (\lambda_T t)^2\right) \\
& + \frac{e^{-\lambda_T t}}{t} \frac{(p_H \lambda_T t)^{N_{BC}}}{(N_{BC}-1)!} \left(e^{p_A \lambda_T t} - \sum_{i=0}^{N_{BC}} \frac{(p_A \lambda_T t)^i}{i!} \right) \\
& + \left(1 - \mathbb{P}_{pAS}\right) \delta(t - \infty),
\end{aligned}
$$

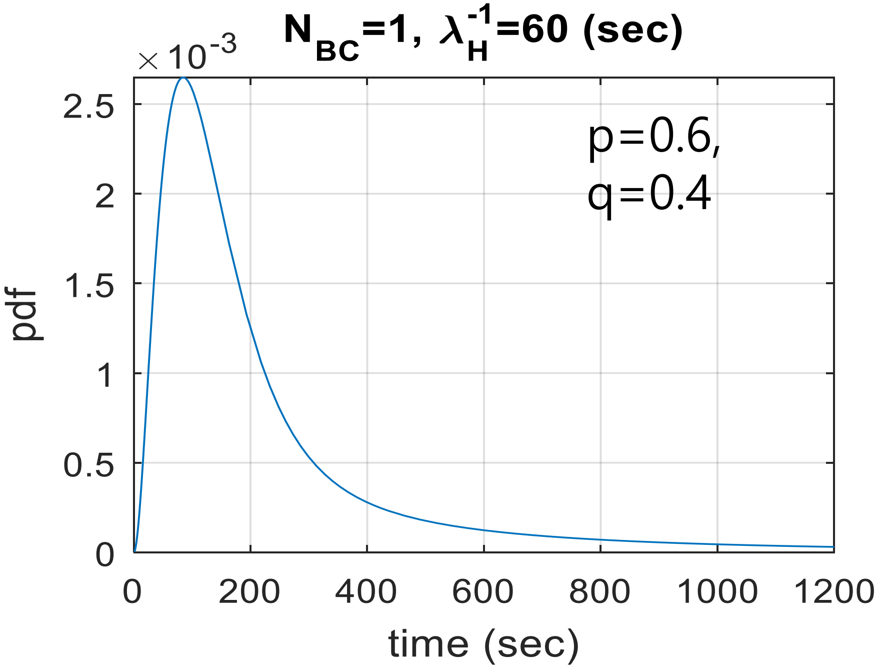
where ${}_pF_q(\mathbf{a};\mathbf{b};x)$ is the generalized hypergeometric function with the parameter vectors

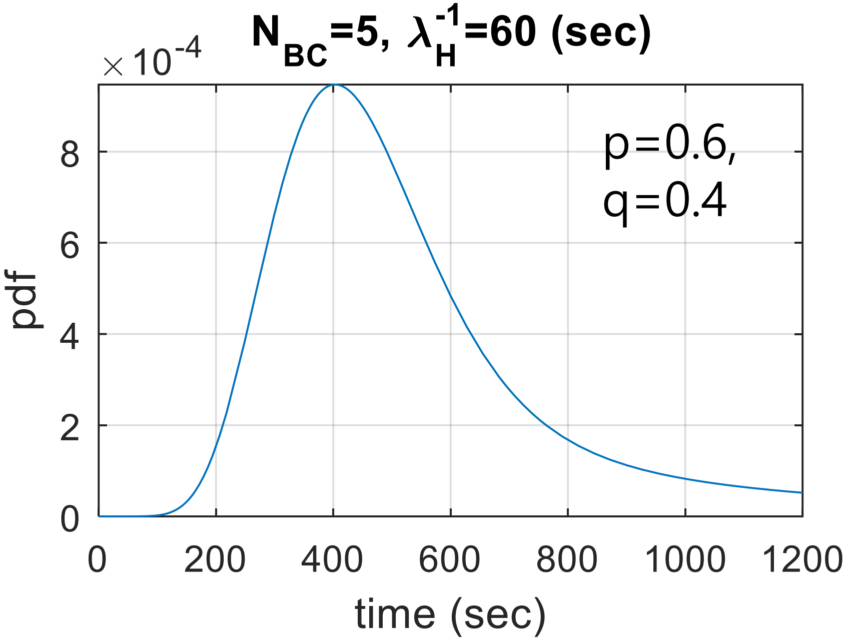
$$
\mathbf{a} = \begin{bmatrix} N_{BC} + 1 - j/2 \\ N_{BC} + 1/2 - j/2 \end{bmatrix}
$$

and

$$
\mathbf{b} = \begin{bmatrix} 2N_{BC} + 2 - j \\ N_{BC} + 1 \\ N_{BC} + 1/2 \end{bmatrix}.
$$

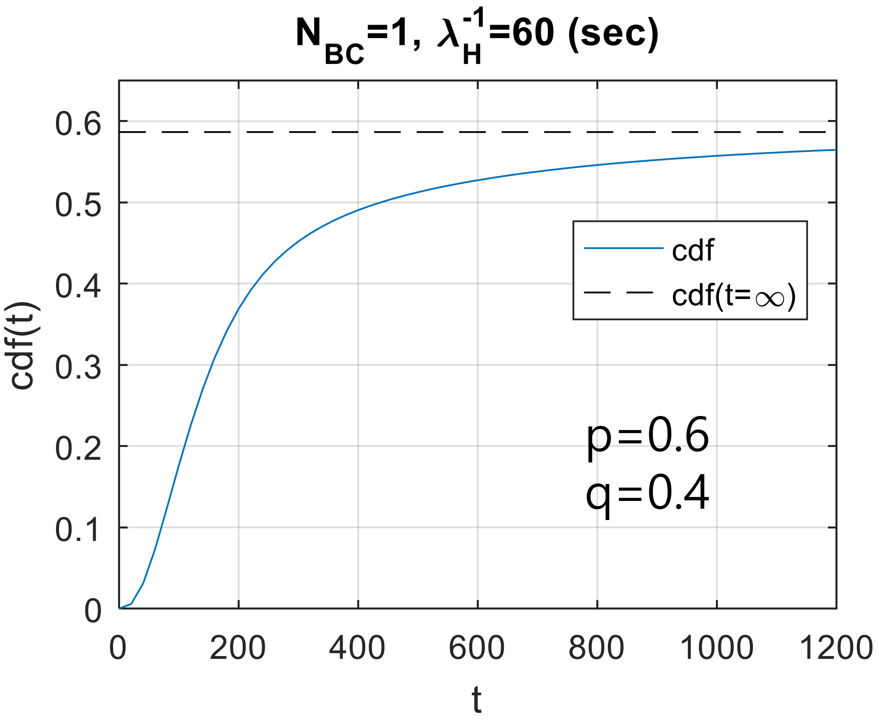
New Result on PDF over time

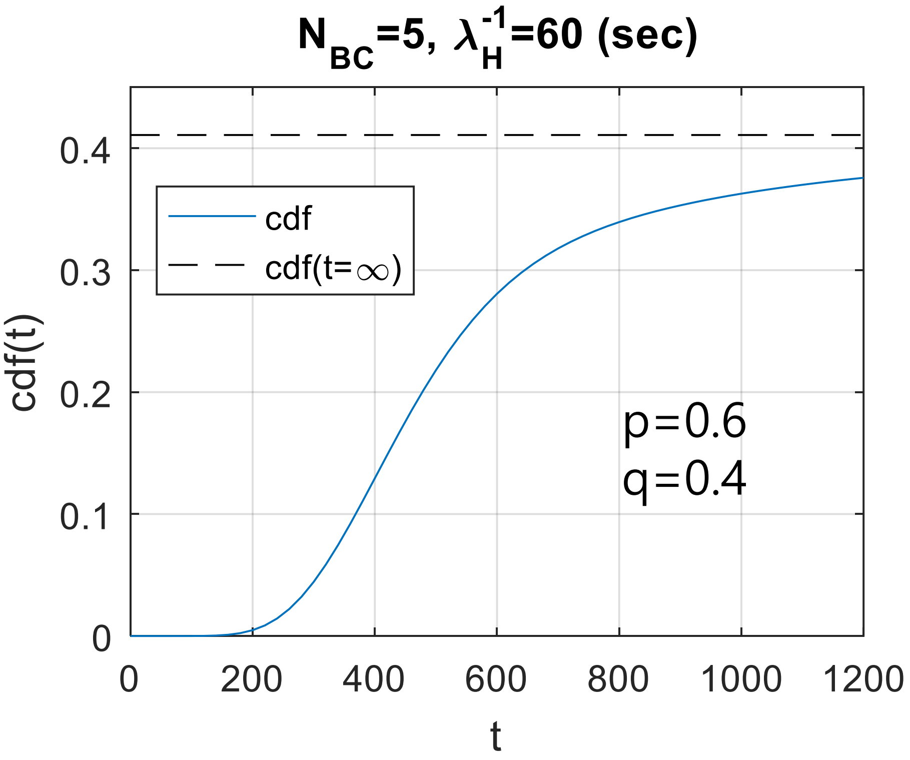




New Result on PDF over time

(Dashed black is Rosenfield Result.)





Ex: *BitcoinCash* network

BitcoinCash Public Info.

- The amount of TXs over 24 hours is about 10 billion dollars.
- Miners' average block generation time λ_H^{-1} is fixed to 600 secs.
- Block confirmation number (N) of target transaction: 5

Mary's Hidden Info.

- Average block generation time: 1143secs (p_A =35%)
- Attack cut time (t_{cut}): 3hours 36mins
- Operating cost per time: γ

Analysis Result

- The value of target transaction: V
- Attack success probability within the cut time: 22%
- Expected attack success time(T_{AS})(if attack succeeded): 1hour 42mins

Profit Info.

- Expected revenue: $0.22 * V$
- Expected cost: 0.22*(1hour 42mins)*γ+0.78*(3hours 36mins)*γ
- Profit=(expected revenue)-(expected cost)

Ex with *BitcoinCash* network

Profit Info.

- Expected revenue: $0.22 * V$
- Expected cost: 0.22*(1hour 42mins)*γ+0.78*(3hours 36mins)*γ
- Profit=(expected revenue)-(expected cost)

- How to make attack profitable?

"Make a target TX value V i.e. Profit>0."

- The operating expense per time (γ) is given in internet.
- For example, *nicehash.com* provides a rental service of hash power.
- According to nicehash.com, the **expected cost is 2.909 BTC.**

"If $V > 13.225$ BTC, this attack is profitable."

Profitable DS Attack Contribution

New theorems and propositions are developed such as

1. probabilistic behaviors of attack success time and
2. conditions for profitable DS attacks.

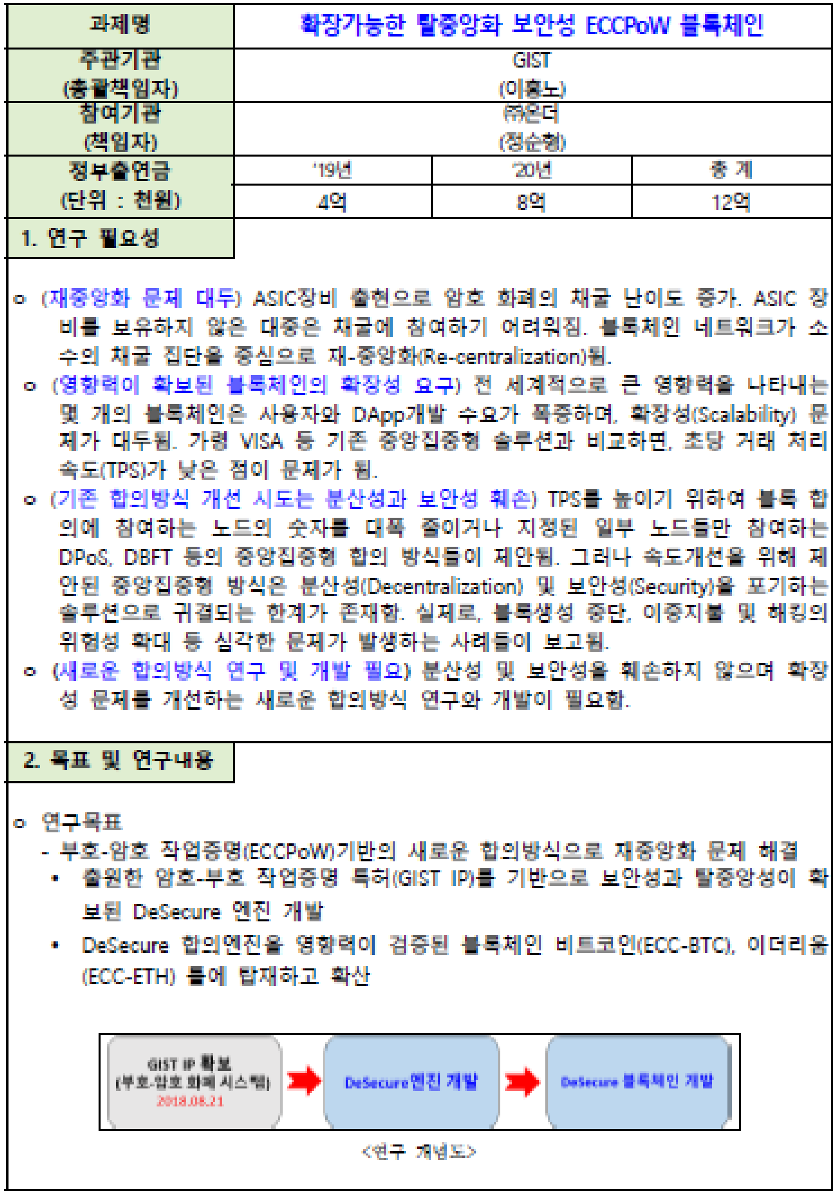
These tools enabled our analyses such as

1. riskiness of DS attacks even with an attack less than 50% of hash power and
2. guidances to prevent profitable DS attacks.

Scalable **DeSecure Blockchains**

PoW is fundamental.
But there is problems.
Let us fix its problem and use it.

IITP 과제 선정



과제명	확장가능한 탈중앙화 보안성 ECCPoW 블록체인		
주관기관 (총괄책임자)	GIST (이흥노)		
참여기관 (책임자)	㈜온더 (정순형)		
정부출연금 (단위 : 천원)	'19년	'20년	총 계
	4억	8억	12억

1. 연구 필요성

ㅇ (재중앙화 문제 대두) ASIC장비 출현으로 암호 화폐의 채굴 난이도 증가. ASIC 장비를 보유하지 않은 대중은 채굴에 참여하기 어려워짐. 블록체인 네트워크가 소수의 채굴 집단을 중심으로 재-중앙화(Re-centralization)됨.

ㅇ (영향력이 확보된 블록체인의 확장성 요구) 전 세계적으로 큰 영향력을 나타내는 몇 개의 블록체인은 사용자와 DApp개발 수요가 폭증하며, 확장성(Scalability) 문제가 대두됨. 가령 VISA 등 기존 중앙집중형 솔루션과 비교하면, 초당 거래 처리속도(TPS)가 낮은 점이 문제가 됨.

ㅇ (기존 합의방식 개선 시도는 분산성과 보안성 훼손) TPS를 높이기 위하여 블록 합의에 참여하는 노드의 숫자를 대폭 줄이거나 지정된 일부 노드들만 참여하는 DPoS, DBFT 등의 중앙집중형 합의 방식들이 제안됨. 그러나 속도개선을 위해 제안된 중앙집중형 방식은 분산성(Decentralization) 및 보안성(Security)을 포기하는 솔루션으로 귀결되는 한계가 존재함. 실제로, 블록생성 중단, 이중지불 및 해킹의 위험성 확대 등 심각한 문제가 발생하는 사례들이 보고됨.

ㅇ (새로운 합의방식 연구 및 개발 필요) 분산성 및 보안성을 훼손하지 않으며 확장성 문제를 개선하는 새로운 합의방식 연구와 개발이 필요함.

2. 목표 및 연구내용

ㅇ 연구목표

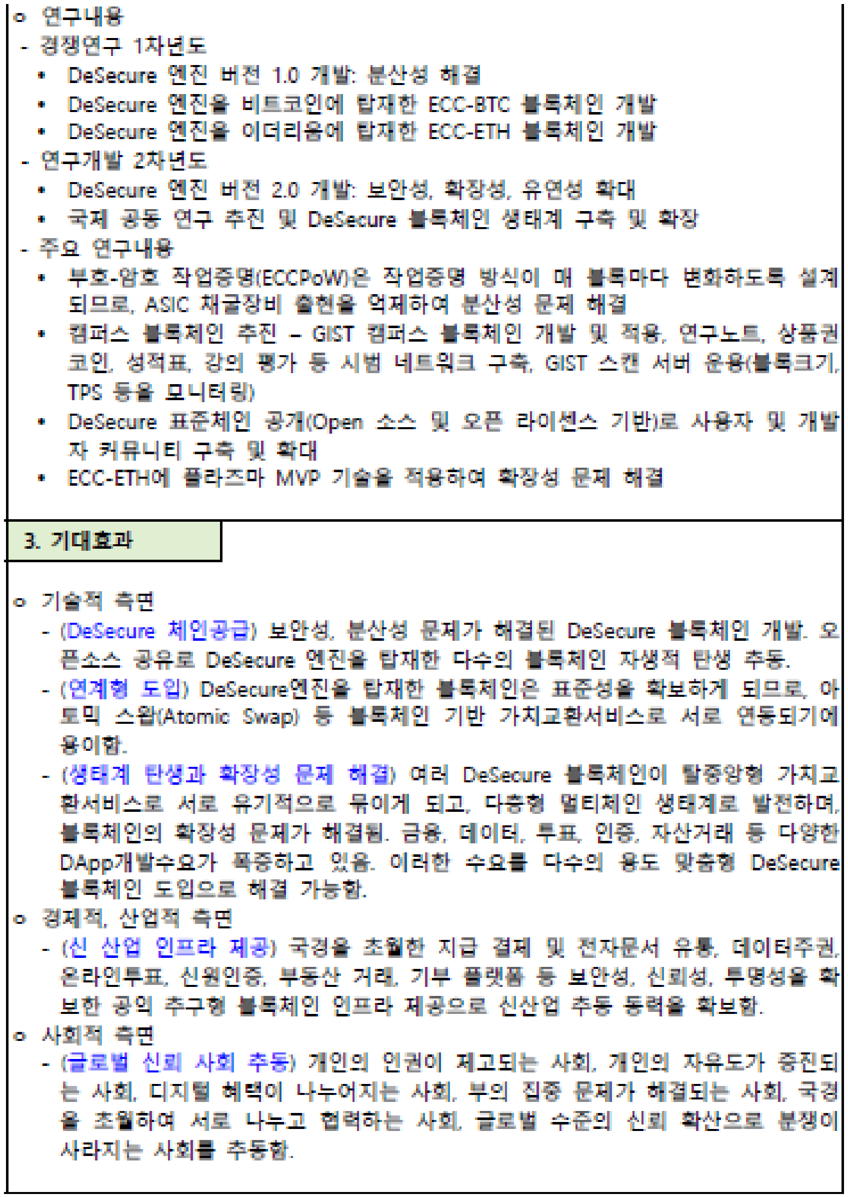
- 부호-암호 작업증명(ECCPoW)기반의 새로운 합의방식으로 재중앙화 문제 해결
  - 출원한 암호-부호 작업증명 특허(GIST IP)를 기반으로 보안성과 탈중앙성이 확보된 DeSecure 엔진 개발
  - DeSecure 합의엔진을 영향력이 검증된 블록체인 비트코인(ECC-BTC), 이더리움(ECC-ETH) 등에 탑재하고 확산

GIST IP 확보
(부호-암호 화폐 시스템)
2018.08.21

DeSecure엔진 개발

DeSecure 블록체인 개발

<연구 개념도>



ㅇ 연구내용

- 경쟁연구 1차년도
  - DeSecure 엔진 버전 1.0 개발: 분산성 해결
  - DeSecure 엔진을 비트코인에 탑재한 ECC-BTC 블록체인 개발
  - DeSecure 엔진을 이더리움에 탑재한 ECC-ETH 블록체인 개발
- 연구개발 2차년도
  - DeSecure 엔진 버전 2.0 개발: 보안성, 확장성, 유연성 확대
  - 국제 공동 연구 추진 및 DeSecure 블록체인 생태계 구축 및 확장
- 주요 연구내용
  - 부호-암호 작업증명(ECCPoW)은 작업증명 방식이 매 블록마다 변화하도록 설계되므로, ASIC 채굴장비 출현을 억제하여 분산성 문제 해결
  - 캠퍼스 블록체인 추진 – GIST 캠퍼스 블록체인 개발 및 적용, 연구노트, 상품권 코인, 성적표, 강의 평가 등 시범 네트워크 구축, GIST 스캔 서버 운용(블록크기, TPS 등을 모니터링)
  - DeSecure 표준체인 공개(Open 소스 및 오픈 라이센스 기반)로 사용자 및 개발자 커뮤니티 구축 및 확대
  - ECC-ETH에 플라즈마 MVP 기술을 적용하여 확장성 문제 해결

3. 기대효과

ㅇ 기술적 측면

- (DeSecure 체인공급) 보안성, 분산성 문제가 해결된 DeSecure 블록체인 개발. 오픈소스 공유로 DeSecure 엔진을 탑재한 다수의 블록체인 자생적 탄생 추동.
- (연계형 도입) DeSecure엔진을 탑재한 블록체인은 표준성을 확보하게 되므로, 아토믹 스왑(Atomic Swap) 등 블록체인 기반 가치교환서비스로 서로 연동되기에 용이함.
- (생태계 탄생과 확장성 문제 해결) 여러 DeSecure 블록체인이 탈중앙형 가치교환서비스로 서로 유기적으로 묶이게 되고, 다층형 멀티체인 생태계로 발전하며, 블록체인의 확장성 문제가 해결됨. 금융, 데이터, 투표, 인증, 자산거래 등 다양한 DApp개발수요가 폭증하고 있음. 이러한 수요를 다수의 용도 맞춤형 DeSecure 블록체인 도입으로 해결 가능함.

ㅇ 경제적, 산업적 측면

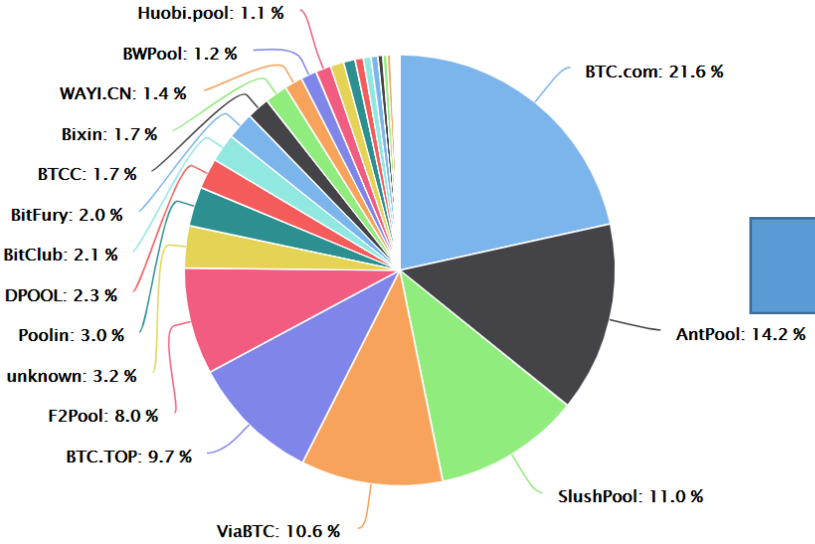
- (신 산업 인프라 제공) 국경을 초월한 지급 결제 및 전자문서 유통, 데이터주권, 온라인투표, 신원인증, 부동산 거래, 기부 플랫폼 등 보안성, 신뢰성, 투명성을 확보한 공익 추구형 블록체인 인프라 제공으로 신산업 추동 동력을 확보함.

ㅇ 사회적 측면

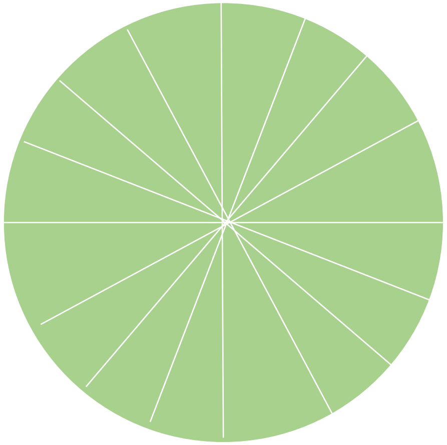
- (글로벌 신뢰 사회 추동) 개인의 인권이 제고되는 사회, 개인의 자유도가 증진되는 사회, 디지털 혜택이 나누어지는 사회, 부의 집중 문제가 해결되는 사회, 국경을 초월하여 서로 나누고 협력하는 사회, 글로벌 수준의 신뢰 확산으로 분쟁이 사라지는 사회를 추동함.

PoW 문제점을 해결, 제안기술의 도전성, 혁신성

- PoW 재 중앙화, 전력소비 문제를 해결하는 새로운 작업증명 기술 개발 필요
  - ASIC 채굴기의 출현, 소수의 채굴 점유, 재 중앙화 문제 발생
    - 소수 채굴업자 블록체인 보상 독점
    - 채굴업자 담합, 블록 위·변조 위험 대두



중앙화된 블록체인 네트워크



탈 중앙화된 블록체인 네트워크

1. ASIC 저항성 증가
2. 블록체인 위변조 위험 감소

Proof-of-XXX, Many alternatives to PoW

	Pros	Cons	Coins within top 50 rank
PoW (Proof-of-Work)	• Strong security - Difficult to produce - Easy to verify	• Extreme computing power • 51% attacks • Transaction speed / Transaction throughput	Bitcoin Ethereum
PoS (Proof-of-Stake)	• Energy & hardware efficiency • Much more expensive 51% attacks	• Recentralization • The rich-get-richer • "Noting at stake" problem	Qtum Stratis
DPoS (Delegated PoS)	• Scalability and speed • Energy & hardware efficiency • Encouraging good behavior by real-time voting	• Recentralization • DDoS attacks	EOS NEO smart economy
PoA (Proof-of-Activity)	• Much more expensive 51% attacks • Decentralization - Validators are randomly selected.	• Extreme computing power • Recentralization • The rich-get-richer	decred

ECCPoW와 다른 확장성 방법과 비교

DeSecure Blockchain aims to resolve the re-centralization problem without sacrificing the secureness!

체인 이름	DeSecure	Bitcoin		Ethereum	
방법	다층멀티체인 ECCPoW	세그윗	라이트닝 네트워크	플라즈마	샤딩
구현	ECCPoW 기반 독립체인들을 여러 계층으로 묶음	블록 데이터 구조를 변경하여 구현	오프체인 거래 진행 최종 결과값을 메인 블록체인 기록	하부 체인 생성 거래 진행 후 최소한의 기록만 메인 블록체인 기록	블록체인의 DB에 해당하는 스테이트를 여러 샤드로 분할, 트랜잭션 별 분리 처리
장점	서로 다른 블록체인 연결해 다양한 기능과 역할 구현	쉽게 구현이 가능함	결제 속도 제고 즉각적인 완결성 수수료 절감	수수료 절감	트랜잭션 처리 속도 증가
단점	No single chain solution/생태계필요	트랜잭션 처리속도 증가 효과 미비	오프체인 거래기록 없음	Full노드 만 플라즈마 사용 가능	S/W 복잡도 상승

트릴레마, Single 블록체인으로 해결 가능?

Blockchain Trilemma?

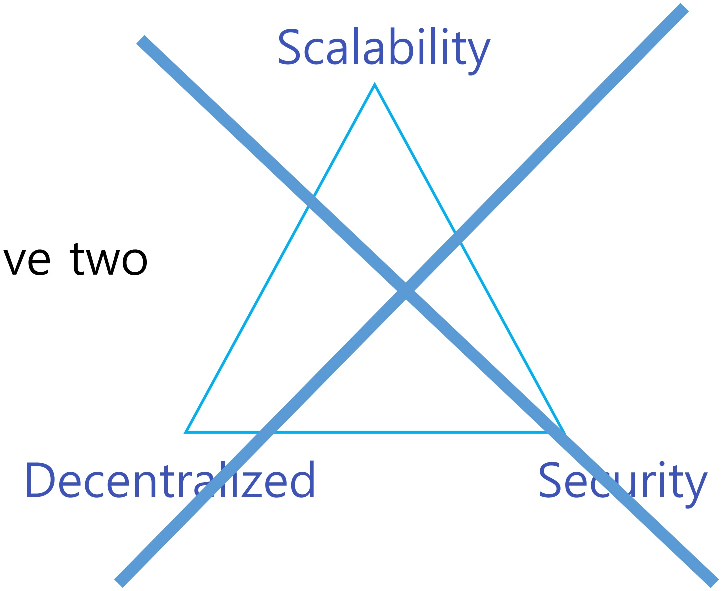
"

blockchain systems can only at most have two of the following three properties

- Vitalik Buterin, Sharding FAQ

https://github.com/ethereum/wiki/wiki/Sharding-FAQ

"



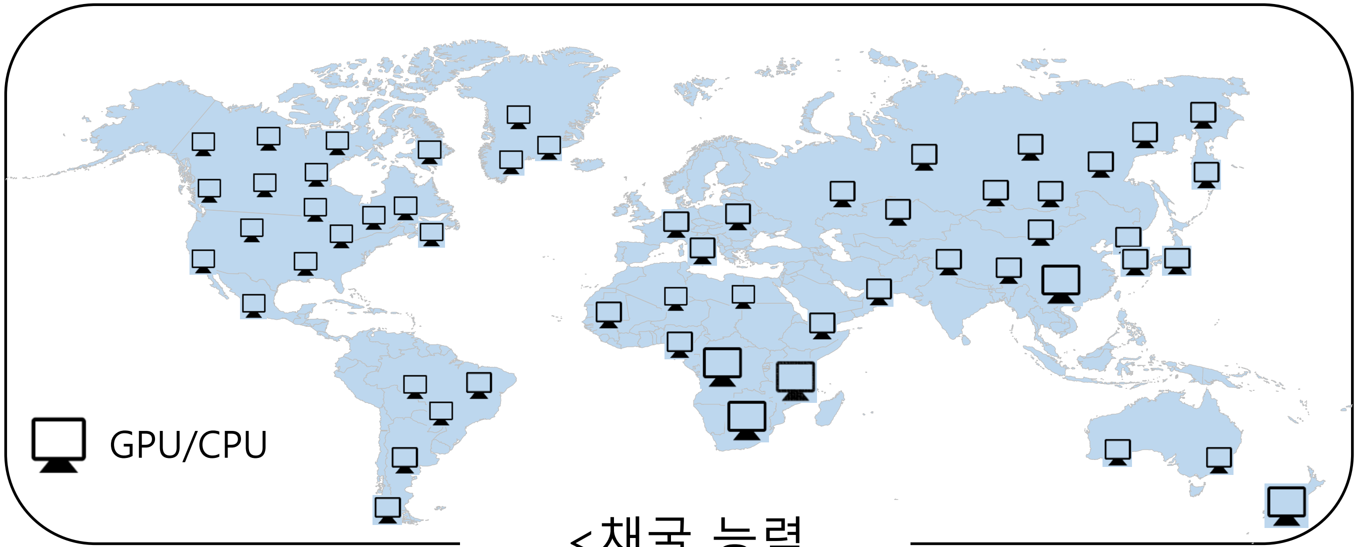
- Wrong approach!
- Not in a single blockchain, can it be achieved!
- *We shall promote many decentralized secure (DeSecure) blockchains to achieve scalability!*

Concept of ECCPoW

▣ 부호-암호 작업증명 (ECCPoW) 목표

- 부호-암호 작업증명은 다음 요건들을 만족시켜야 함
  1. 암호 퍼즐은 풀기 어려우나, *증명과 검증은 쉬워야 함*
  2. 암호 퍼즐은 외부 *공격*으로부터 견고한 *저항성*을 가져야 함
  3. 풀린 암호 퍼즐의 정답은 재사용되지 않아야 함
  4. 암호 퍼즐의 난이도는 조절 가능해야 함
  5. CPU를 갖고 있으면 *누구든지 작업 증명에 참여* 할 수 있어야 함
  6. 작업증명에 쓰이는 함수는 *매 블록마다 변화*하여 바꿀 수 있어야 함

▣ ECCPoW는 ASIC 채굴장비 출현 억제, 채굴능력 평준화도 개선

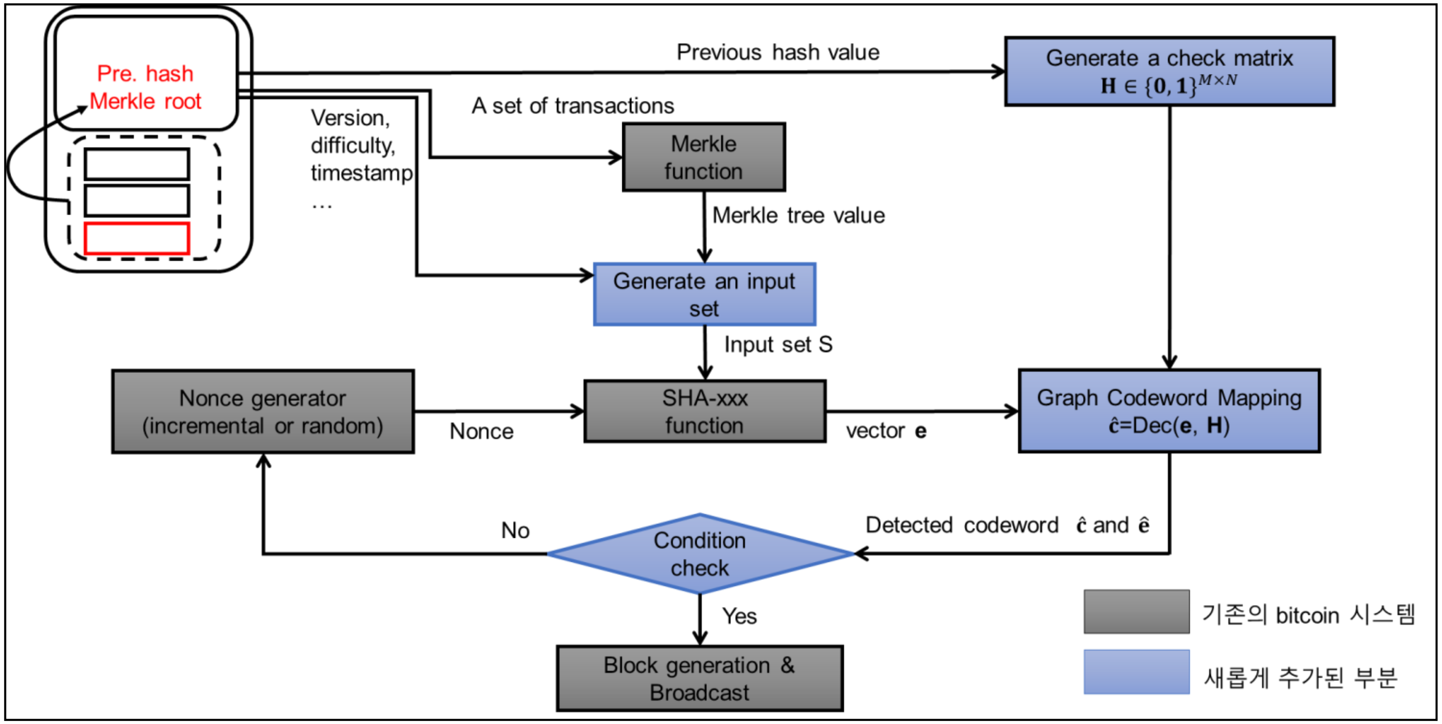


<채굴 능력 평준화>

ECCPoW의 원리 및 구조

▣ ECCPoW의 원리: 매 블록마다 변화하는 암호 퍼즐 디코더

- 기존 Bitcoin의 PoW: SHA함수 대입문제를 반복적으로 풀어 원하는 답 찾기
- 새로운 ECCPoW: LDPC 부호의 역문제를 풀어 원하는 답 찾기 (작업증명 *난이도 조절능력*)
- 블록에 기록된 해쉬 값을 활용하여 *매블록마다 새로운 LDPC 행렬 생성 (ASIC Resistance)*
- LDPC행렬을 매 블록마다 바꾸어 *역문제를 매 블록마다 바꿈 (ASIC Resistance)*
- *SHA함수 Output을 여러 개 묶어 LDPC 부호의 길이를 크게 할 수 있음 (ASIC Resistance)*



<부호-암호 작업증명>

ECCPoW 하드포크의 의미는 합의엔진 교체

블록체인 프로그램 코어

블록체인 구성 요소 3가지

1. Web server interface networking of peers

- Node registration, get-address, give-address
- Full node or light node
- Communication among the wallets and the miners

2. Wallet for TX generations

- Make private and public keys, address, store UTXOs, make TX, put signature, announce it to the neighbor, check to see if the TX is supported by the blockchain.

3. Consensus Mechanism

합의엔진

- Data: Genesis block + regular blocks, one block every 10 min, block-size 1Mbyte
- Protocol: consensus, block header, difficulty level adjustment, …
- Mining: Get the longest chain, validate it and all transactions within it, get transactions from mempool and form a block, run SHA repeatedly until you hit a good hash, put the proof into the block header, and attach the proofed block to the longest chain, and make announcement ASAP.

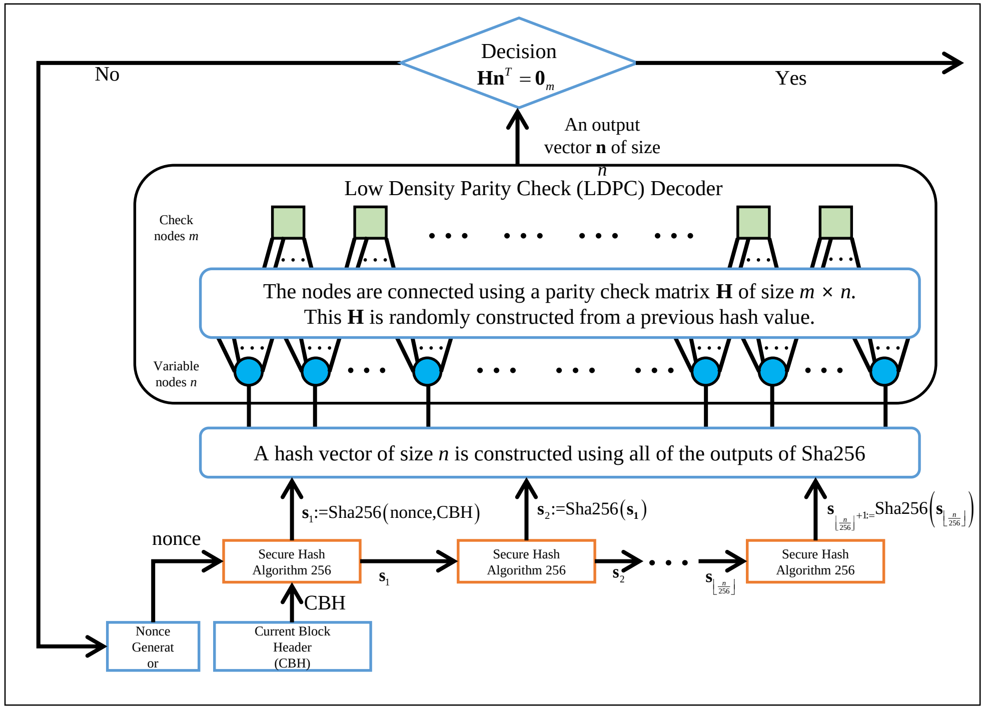
Program Suite

- **C++, Python, Go, Java, Flask, http**
- **Download and run, then you have a blockchain server.**

ECCPoW 합의 엔진 개념도

▣ ECCPoW 합의 엔진

- SHA와 LDPC Decoder의 합성함수 개발
- Parity Check Matrix (PCM) 크기 변경, 필요한 자원(mem, comp)량 변경
- PCM은 이전 블록의 hash 값을 입력하여 블록마다 변경 가능



Decision

$\mathbf{Hn}^T = \mathbf{0}_m$

No

Yes

An output vector **n** of size *n*

Low Density Parity Check (LDPC) Decoder

Check nodes *m*

The nodes are connected using a parity check matrix **H** of size *m* × *n*.
This **H** is randomly constructed from a previous hash value.

Variable nodes *n*

A hash vector of size *n* is constructed using all of the outputs of Sha256

$\mathbf{s}_1 := \mathrm{Sha256}(\mathrm{nonce}, \mathrm{CBH})$

$\mathbf{s}_2 := \mathrm{Sha256}(\mathbf{s}_1)$

$\mathbf{s}_{\lfloor \frac{n}{256} \rfloor + 1} := \mathrm{Sha256}\left(\mathbf{s}_{\lfloor \frac{n}{256} \rfloor}\right)$

nonce

Secure Hash Algorithm 256

$\mathbf{s}_1$

Secure Hash Algorithm 256

$\mathbf{s}_2$

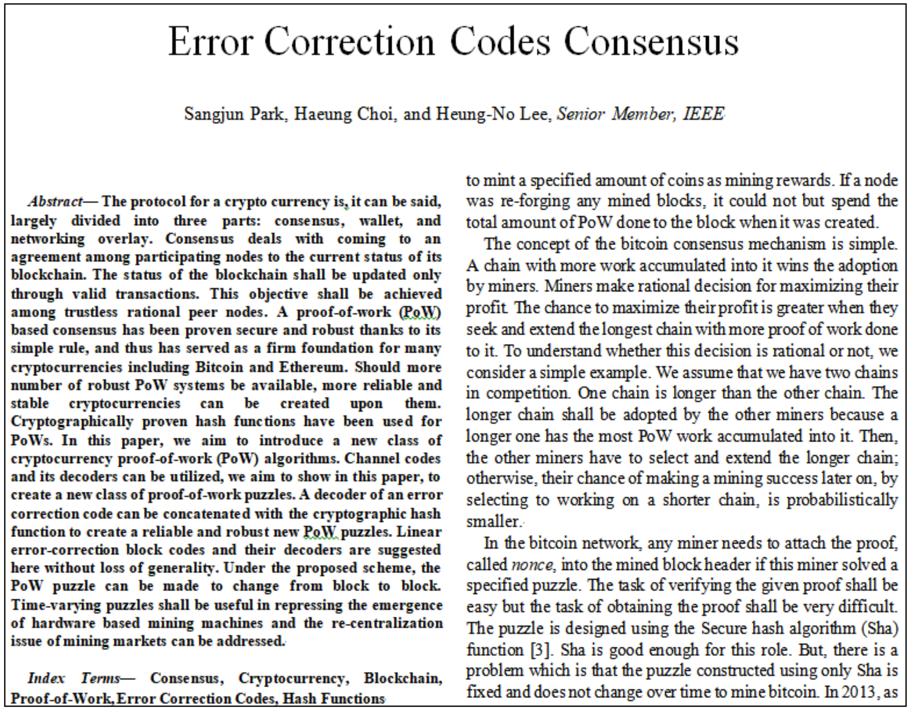
$\mathbf{s}_{\lfloor \frac{n}{256} \rfloor}$

Secure Hash Algorithm 256

CBH

Nonce Generator

Current Block Header (CBH)



Error Correction Codes Consensus

Sangjun Park, Haeung Choi, and Heung-No Lee, *Senior Member, IEEE*

Abstract— The protocol for a crypto currency is, it can be said, largely divided into three parts: consensus, wallet, and networking overlay. Consensus deals with coming to an agreement among participating nodes to the current status of its blockchain. The status of the blockchain shall be updated only through valid transactions. This objective shall be achieved among trustless rational peer nodes. A proof-of-work (PoW) based consensus has been proven secure and robust thanks to its simple rule, and thus has served as a firm foundation for many cryptocurrencies including Bitcoin and Ethereum. Should more number of robust PoW systems be available, more reliable and stable cryptocurrencies can be created upon them. Cryptographically proven hash functions have been used for PoWs. In this paper, we aim to introduce a new class of cryptocurrency proof-of-work (PoW) algorithms. Channel codes and its decoders can be utilized, we aim to show in this paper, to create a new class of proof-of-work puzzles. A decoder of an error correction code can be concatenated with the cryptographic hash function to create a reliable and robust new PoW puzzles. Linear error-correction block codes and their decoders are suggested here without loss of generality. Under the proposed scheme, the PoW puzzle can be made to change from block to block. Time-varying puzzles shall be useful in repressing the emergence of hardware based mining machines and the re-centralization issue of mining markets can be addressed.

Index Terms— Consensus, Cryptocurrency, Blockchain, Proof-of-Work, Error Correction Codes, Hash Functions

to mint a specified amount of coins as mining rewards. If a node was re-forging any mined blocks, it could not but spend the total amount of PoW done to the block when it was created.

The concept of the bitcoin consensus mechanism is simple. A chain with more work accumulated into it wins the adoption by miners. Miners make rational decision for maximizing their profit. The chance to maximize their profit is greater when they seek and extend the longest chain with more proof of work done to it. To understand whether this decision is rational or not, we consider a simple example. We assume that we have two chains in competition. One chain is longer than the other chain. The longer chain shall be adopted by the other miners because a longer one has the most PoW work accumulated into it. Then, the other miners have to select and extend the longer chain; otherwise, their chance of making a mining success later on, by selecting to working on a shorter chain, is probabilistically smaller.

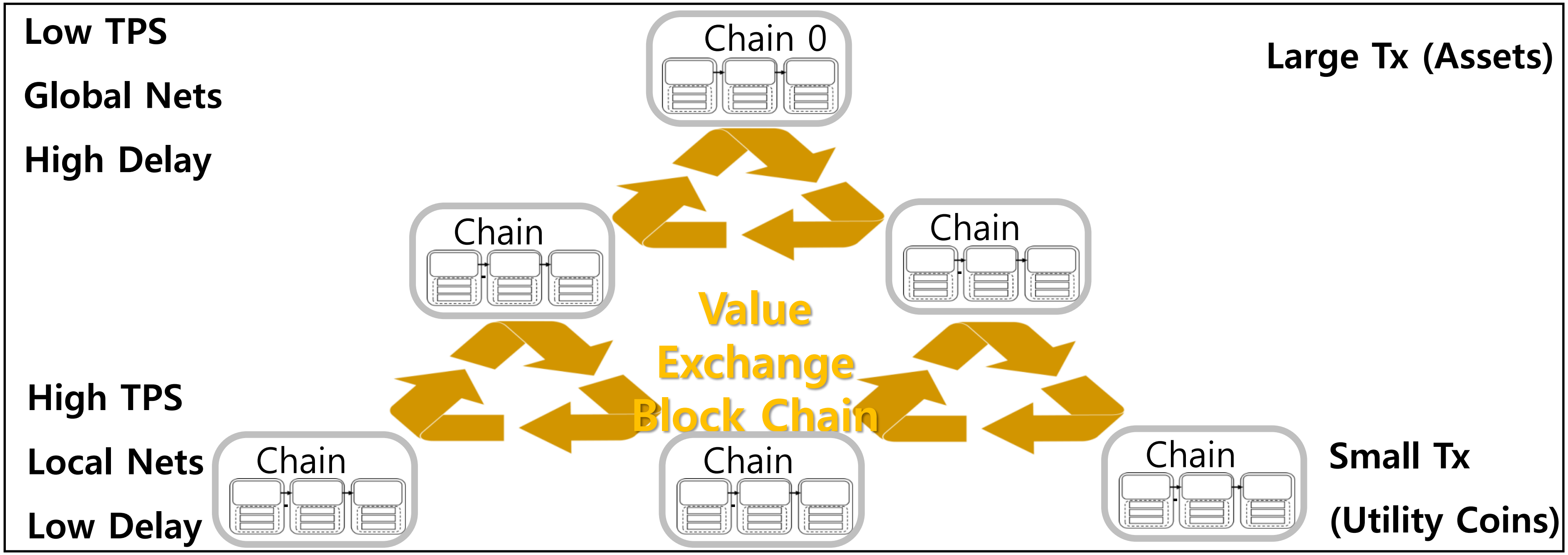
In the bitcoin network, any miner needs to attach the proof, called *nonce*, into the mined block header if this miner solved a specified puzzle. The task of verifying the given proof shall be easy but the task of obtaining the proof shall be very difficult. The puzzle is designed using the Secure hash algorithm (Sha) function [3]. Sha is good enough for this role. But, there is a problem which is that the puzzle constructed using only Sha is fixed and does not change over time to mine bitcoin. In 2013, as

※ 국제 학술지 IEEE trans. Information Forensics and Security에 제출예정

DeSecure 표준체인 사용 확대, 확장성 해결

▣ 다층 복합구조 블록체인, 코인가치교환 블록체인

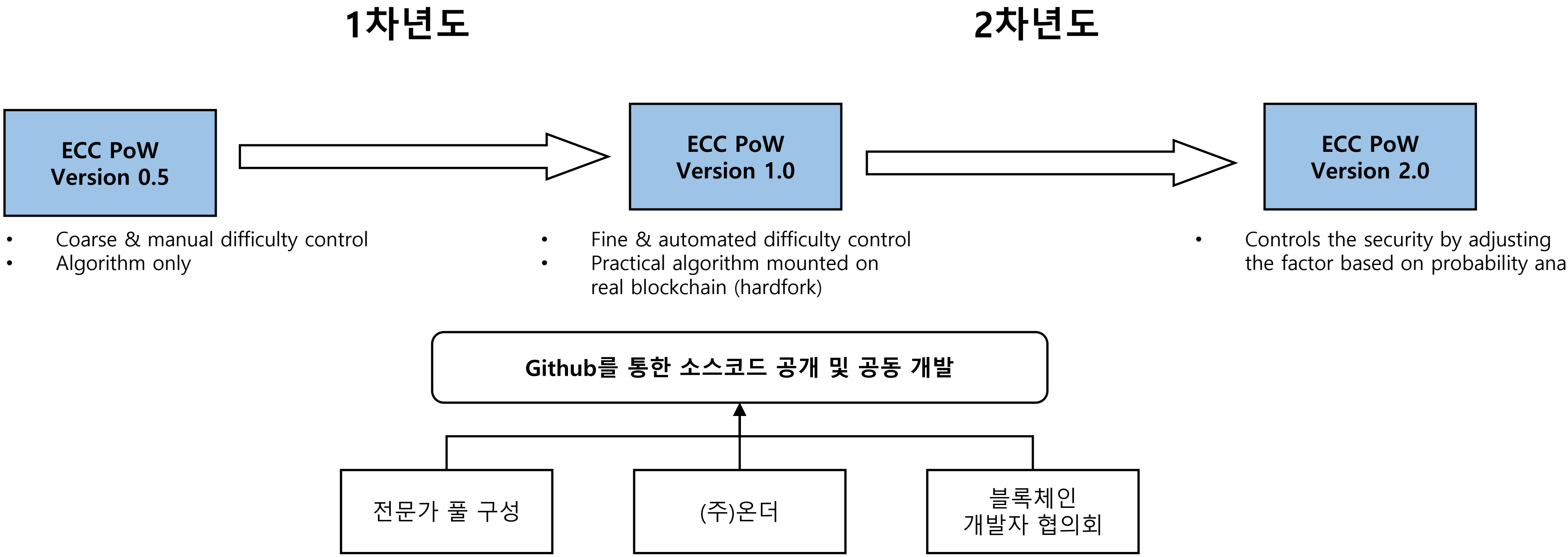
- 현재 비트코인, 이더리움의 초당 처리속도는 수십 건 내외임
- ECCPoW 비트코인에서 확장성 개선을 위해 다층 복합구조 블록체인 제안
- 다 계층 블록체인 간 가교 역할을 담당할 아토믹스왑 기술 적용 가능



<다층 복합구조 블록체인>

연구 추진 전략

▣ DeSecure 블록체인 개발 추진 전략 및 일정



1차년도

2차년도

ECC PoW Version 0.5

- Coarse & manual difficulty control
- Algorithm only

ECC PoW Version 1.0

- Fine & automated difficulty control
- Practical algorithm mounted on real blockchain (hardfork)

ECC PoW Version 2.0

- Controls the security by adjusting the factor based on probability ana

Github를 통한 소스코드 공개 및 공동 개발

전문가 풀 구성

(주)온더

블록체인 개발자 협의회

Summary

- Bitcoin white paper was born in the time of financial crisis.
- It has problems such as re-centralization, burning of electricity.
- But it is fundamental to blockchain security.
- Preventing DS attacks is necessary for any blockchain.
- Scalability can be resolved with a community build up with many DeSecure blockchains deployed.

Q&A

References

- 이흥노 교수 랩 블록체인페이지 https://infonet.gist.ac.kr/?page_id=6370.
- Blockchain.net
- Bitcoin.org
- Coursera course on Cryptocurrencies
- MIT Blockchain center
- Blockchain A beginner's guide, Blockchain Hub
- Satoshi Nakamoto's Bitcoin white paper.
- 그 외