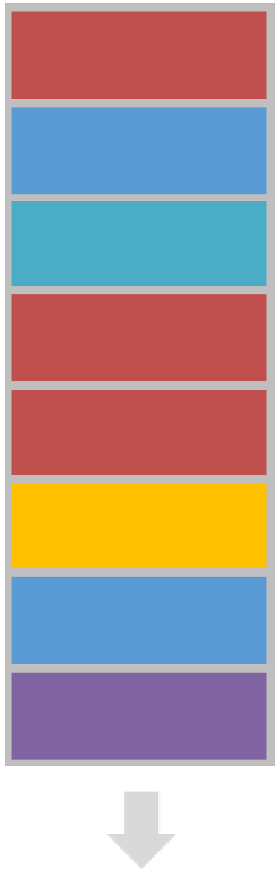
Blockchain Technology

timestamped
append-only log

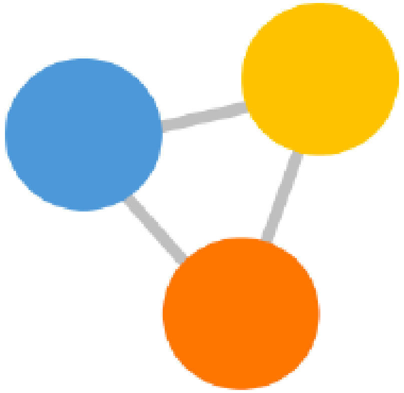


auditable database

Secured via cryptography
- Hash functions for **tamper resistance** and **integrity**
- Digital signatures for **consent**

Consensus for **agreement**

network consensus protocol



Addresses '**cost of trust**'
(Byzantine Generals problem)
- Permissioned
- Permissionless

Source : MIT OpenCourseWare, https://ocw.mit.edu/ 15.S12 Blockchain and Money Fall 2018

Bitcoin – Technical Features

- Cryptographic Hash Functions
- Timestamped Append-only Logs (Blocks)
- Block Headers & Merkle Trees
- Asymmetric Cryptography & Digital Signatures
- Addresses

- Consensus through Proof of Work
- Network of Nodes
- Native Currency

- Transaction Inputs & Outputs
- Unspent Transaction Output (UTXO)
- Scripting language

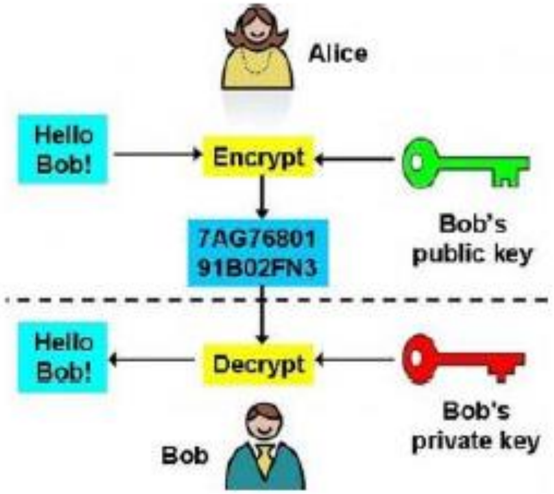
Source : MIT OpenCourseWare, https://ocw.mit.edu/ 15.S12 Blockchain and Money Fall 2018

Cryptography:

Communications in the presence of adversaries

Scytale Cipher
Ancient Times

Enigma Machine
1920s - WWII



Asymmetric Cryptography
1976 to today

Source : MIT OpenCourseWare, https://ocw.mit.edu/ 15.S12 Blockchain and Money Fall 2018

Cryptographic Hash Functions

Digital Fingerprints for Data

- General Properties
  - Maps Input **x** of any size to an Output of fixed size – called a ‘Hash’
  - Deterministic: Always the same Hash for the same **x**
  - Efficiently computed

- Cryptographic Properties
  - Preimage resistant (One way): infeasible to determine **x** from Hash(x)
  - Collision resistant: infeasible to find and **x** and **y** where Hash(**x**) = Hash(**y**)
  - Avalanche effect: Change **x** slightly and Hash(**x**) changes significantly
  - Puzzle friendliness: knowing Hash(**x**) and part of **x** it is still very hard to find rest of **x**

Source : MIT OpenCourseWare, https://ocw.mit.edu/ 15.S12 Blockchain and Money Fall 2018

Cryptographic Hash Functions

Digital Fingerprints for Data

- Uses as
  - Names
  - References
  - Pointers
  - Commitments

- Bitcoin Hash Functions
  - Headers & Merkle Trees – SHA 256
  - Bitcoin Addresses – SHA 256 and RIPEMD160

Source : MIT OpenCourseWare, https://ocw.mit.edu/ 15.S12 Blockchain and Money Fall 2018

Timestamped Append-only Log - Blockchain

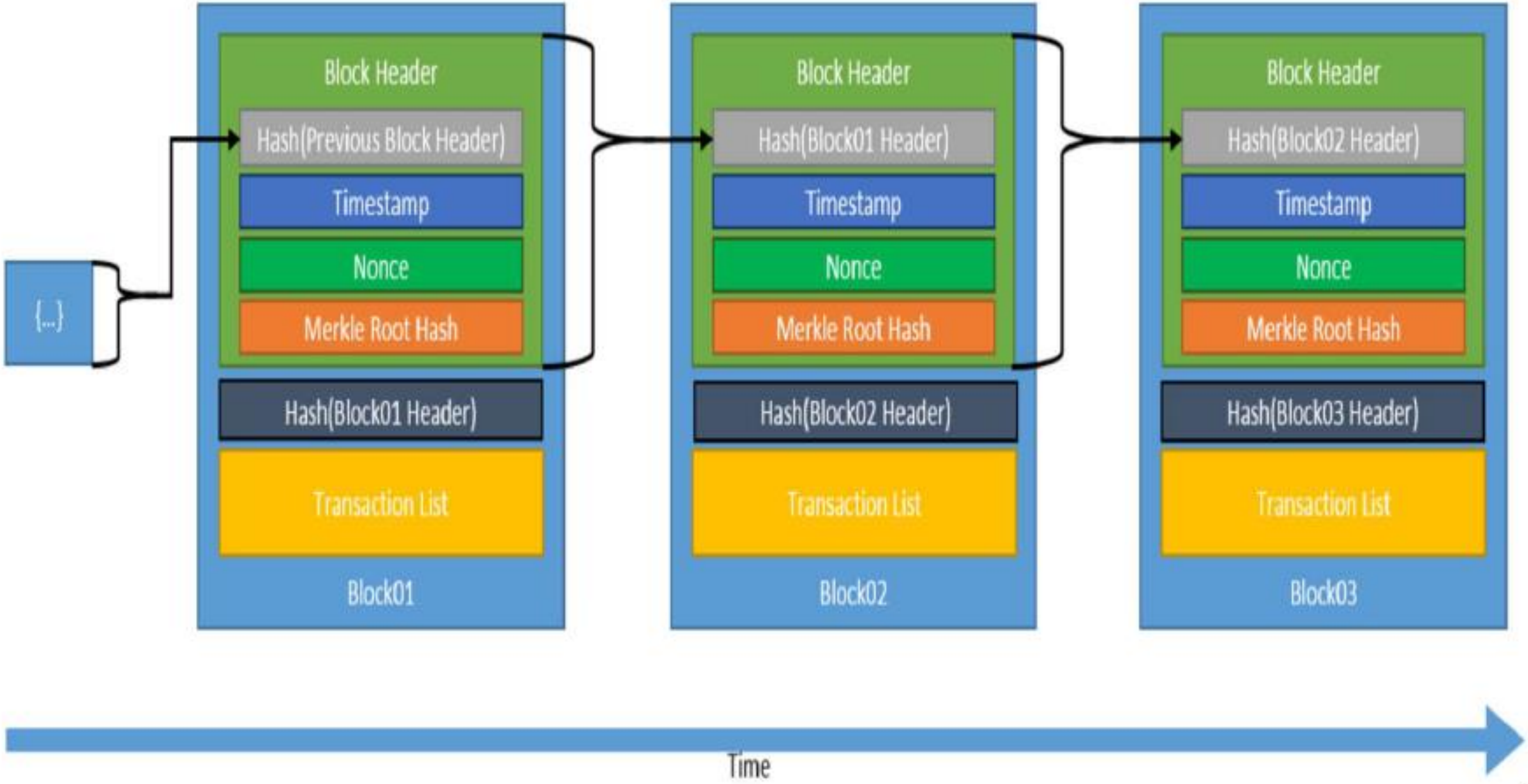


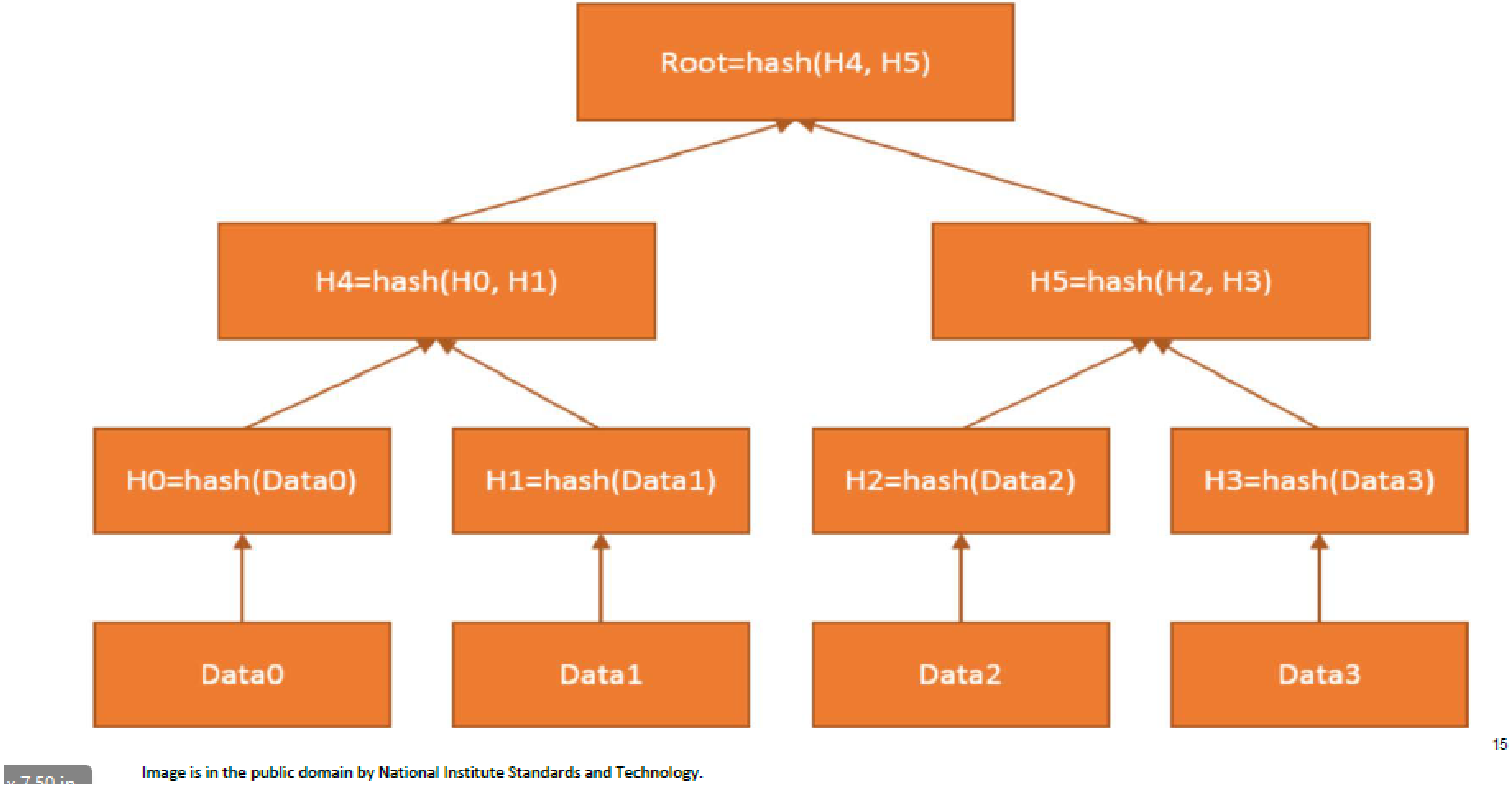
Image is in the public domain by National Institute of Standards and Technology.

Source : MIT OpenCourseWare, https://ocw.mit.edu/ 15.S12 Blockchain and Money Fall 2018

Block Header

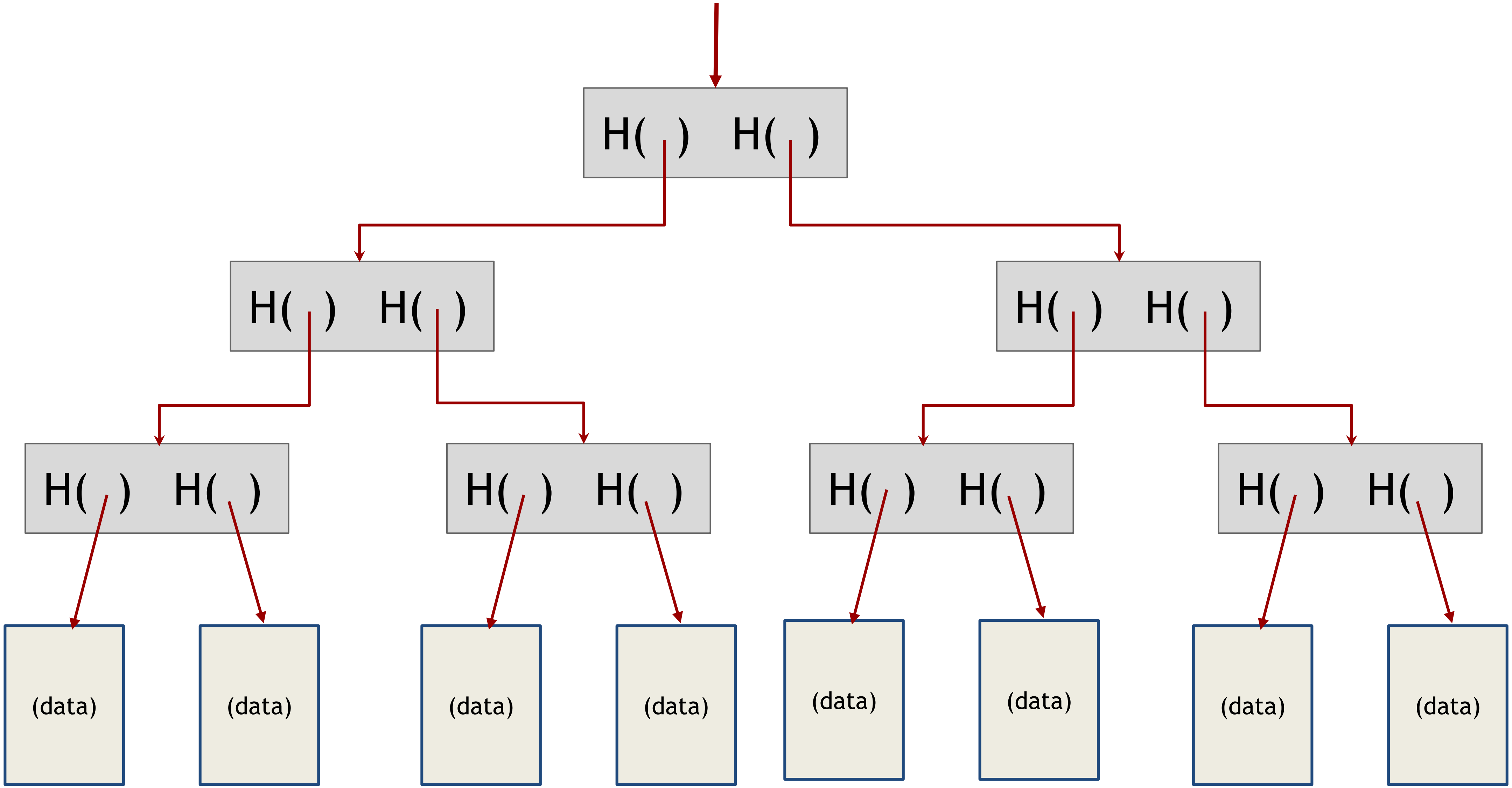
- Version
- Previous Block hash
- Merkle Root hash
- Timestamp
- Difficulty target
- Nonce

Merkle Tree – Binary Data Tree with Hashes



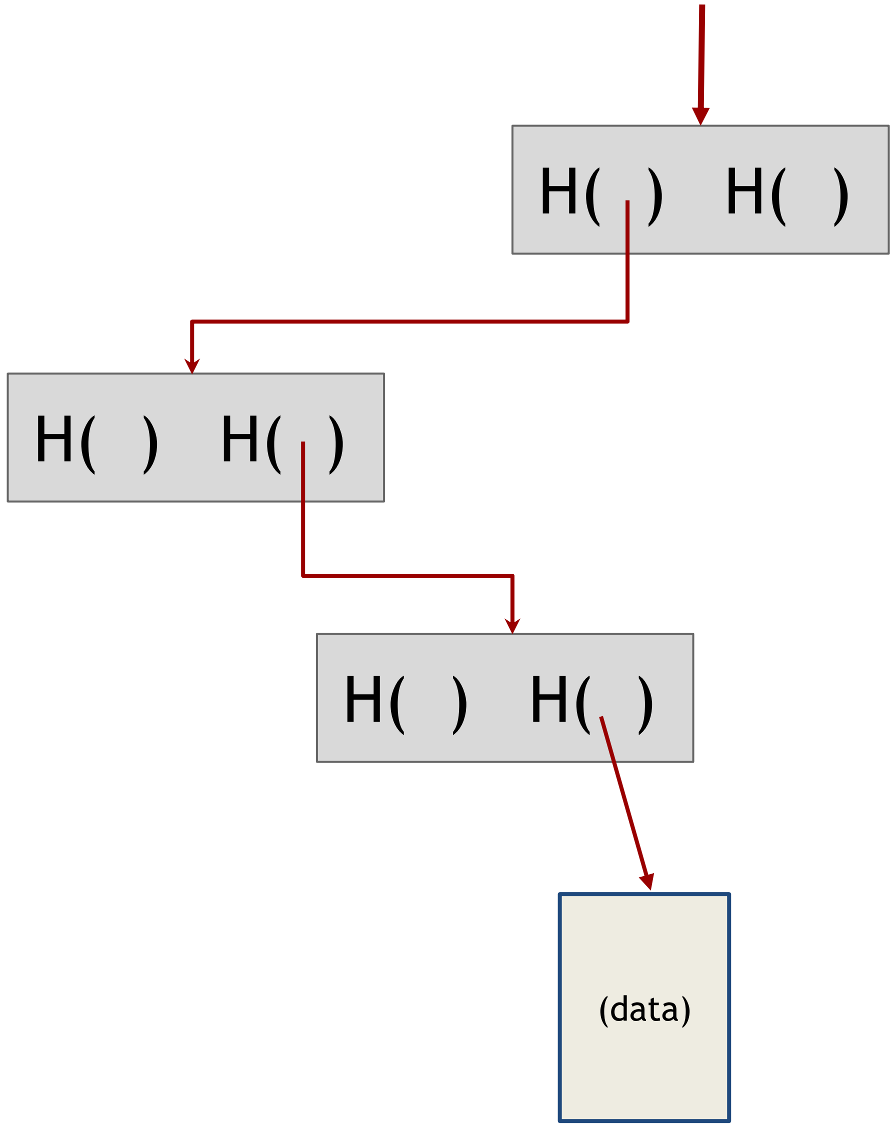
Source : MIT OpenCourseWare, https://ocw.mit.edu/ 15.S12 Blockchain and Money Fall 2018

binary tree with hash pointers = "Merkle tree"



proving membership in a Merkle tree

show O(log n) items



Advantages of Merkle trees

Tree holds many items

but just need to remember the root hash

Can verify membership in O(log n) time/space

Variant: sorted Merkle tree

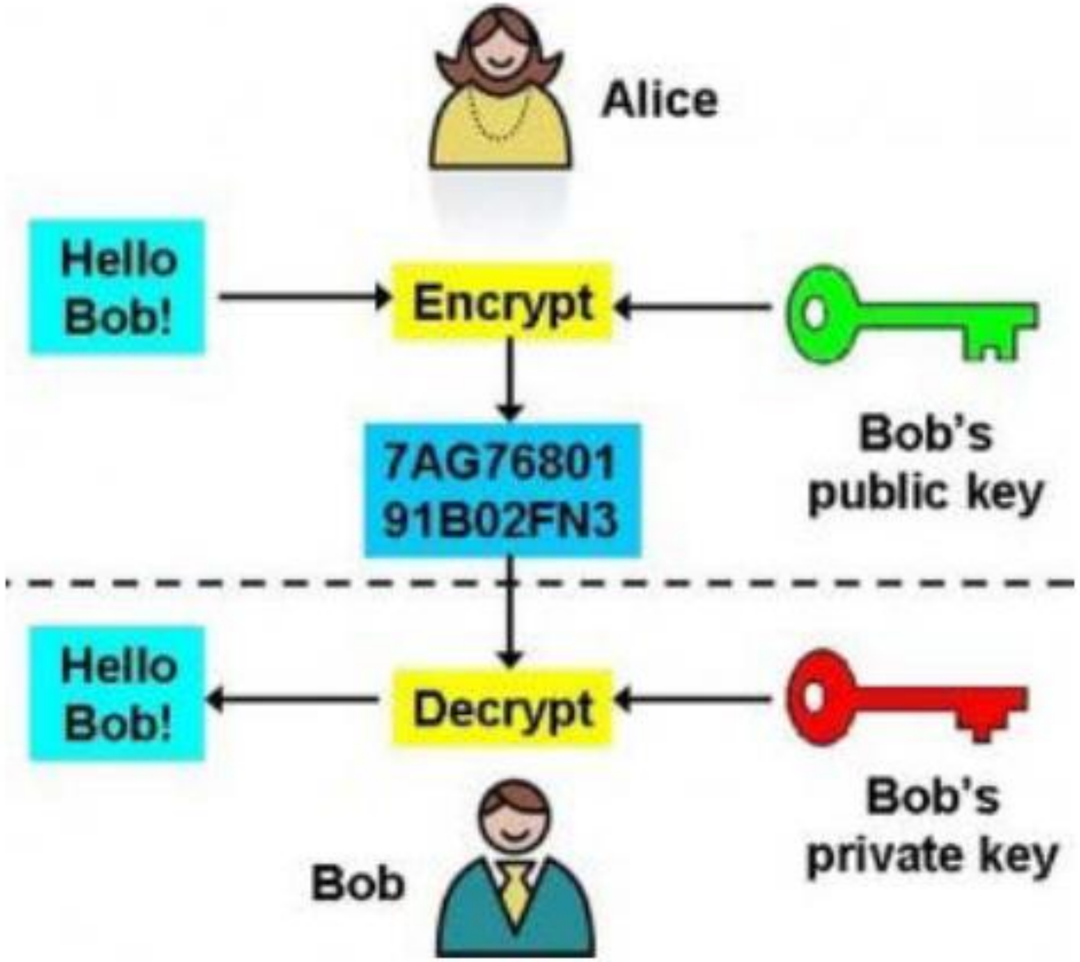
can verify non-membership in O(log n)

(show items before, after the missing one)

More generally ...

can use hash pointers in any pointer-based
data structure that has no cycles

Asymmetric Cryptography & Digital Signatures

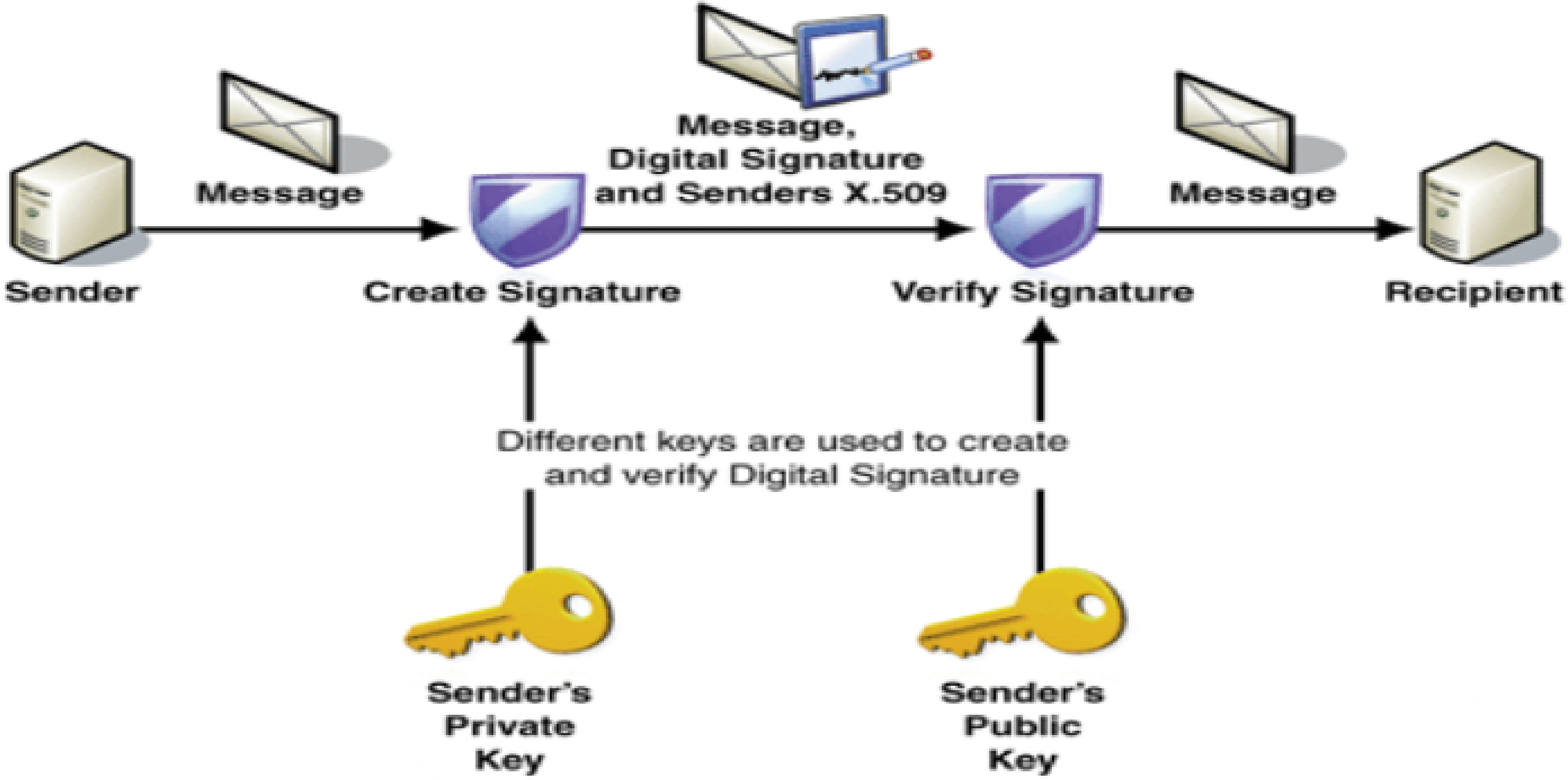


Source : MIT OpenCourseWare, https://ocw.mit.edu/ 15.S12 Blockchain and Money Fall 2018

Asymmetric Cryptography & Digital Signatures

Guarding against Tampering & Impersonation

Digital Signature without Hash

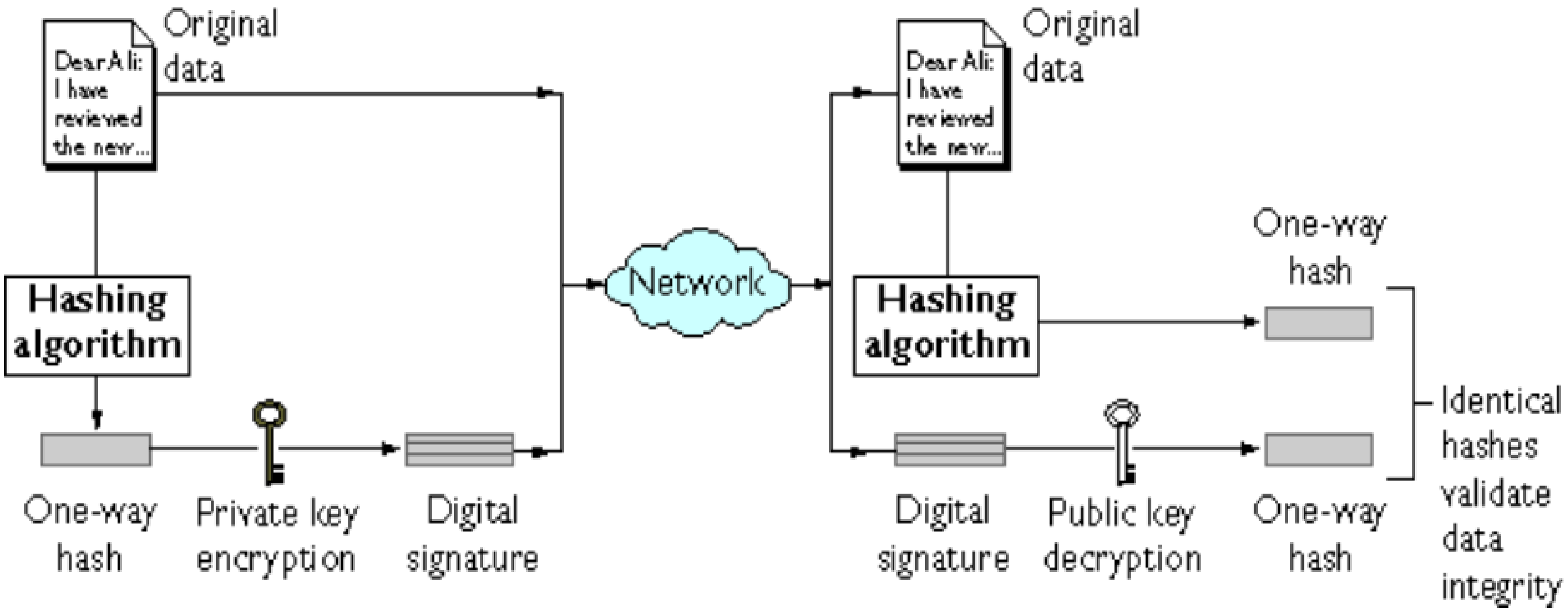


Source : MIT OpenCourseWare, https://ocw.mit.edu/ 15.S12 Blockchain and Money Fall 2018

Asymmetric Cryptography & Digital Signatures

Guarding against Tampering & Impersonation

Digital Signature with Hash



Shyam Nandan Kumar et al. Review on Network Security and Cryptography.

12

Source : MIT OpenCourseWare, https://ocw.mit.edu/ 15.S12 Blockchain and Money Fall 2018

Asymmetric Cryptography & Digital Signatures

- Digital Signature Algorithms
  - Generate Key Pair - Public Key (**PK**) & Private Key (**sk**) - from random number
  - Signature – Creates Digital Signature (**Sig**) from message (**m**) and Private Key (**sk**)
  - Verification – Verifies if a signature (**Sig**) is valid for a message (**m**) and a Public Key (**PK**)

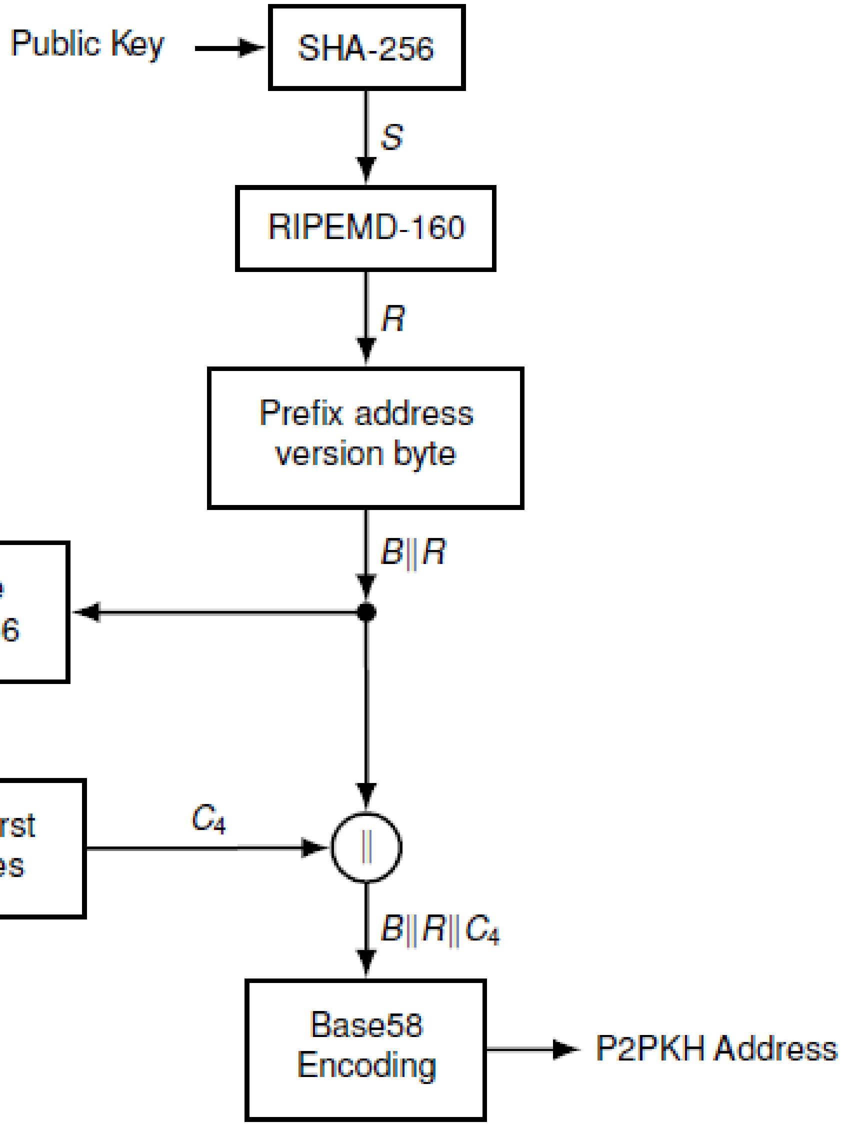
- Properties
  - Infeasible to find Private Key (**sk)** from Public Key (**PK**)
  - All valid signatures verify
  - Signatures infeasible to forge

- Bitcoin Digital Signature Function
  - Elliptic Curve Digital Signature Algorithm (EDCSA) … y2 = x3 + 7

Source : MIT OpenCourseWare, https://ocw.mit.edu/ 15.S12 Blockchain and Money Fall 2018

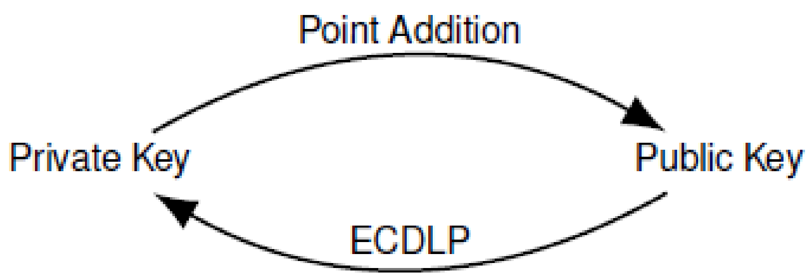
Pay to Public Key Hash

Pay to Public Key Hash Address

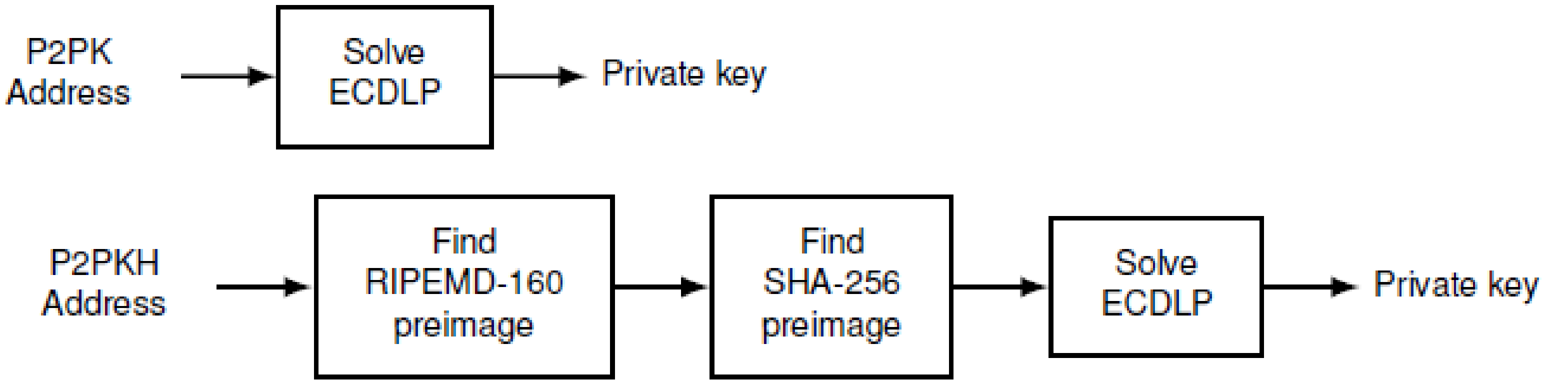


Source : Slides from 'An Introduction to Bitcoin' by Prof. Saravanan Vijayakumaran, IIT Madras

Why Hash the Public Key?



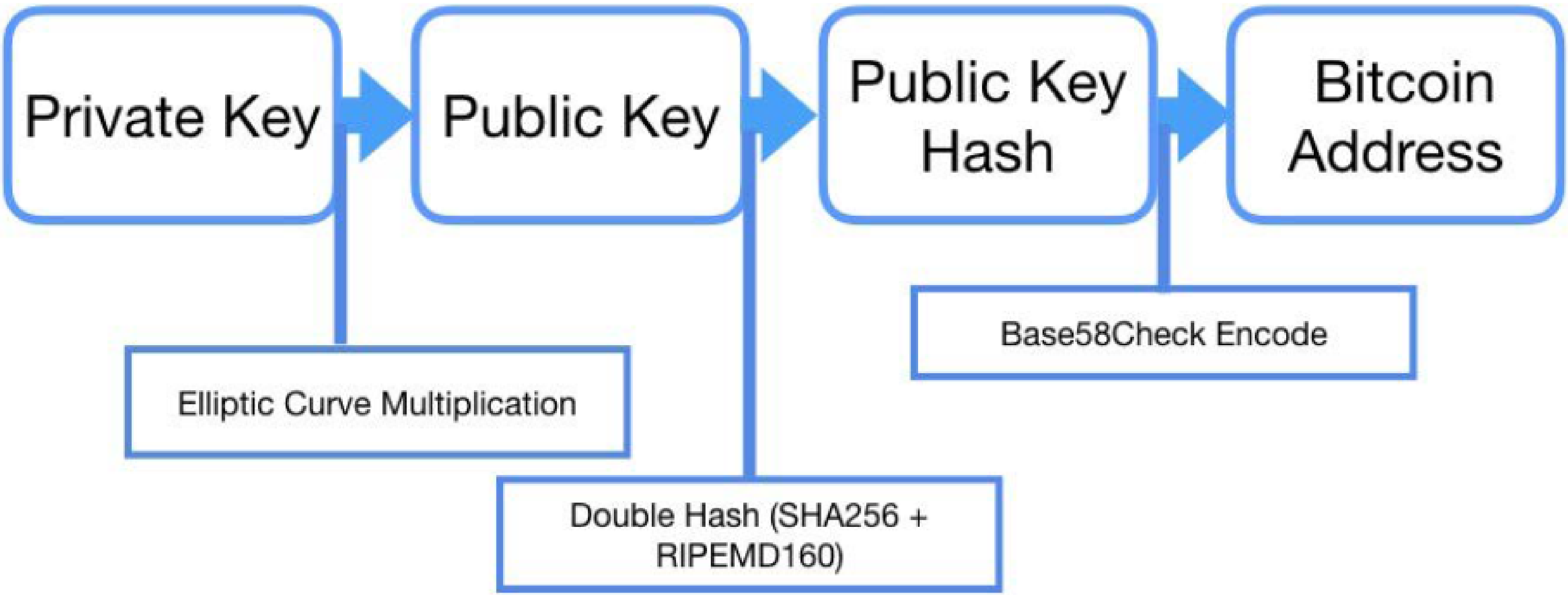
- ECDLP = Elliptic Curve Discrete Logarithm Problem
- ECDLP currently hard but no future guarantees
- Hashing the public key gives extra protection



Source : Slides from 'An Introduction to Bitcoin' by Prof. Saravanan Vijayakumaran, IIT Madras

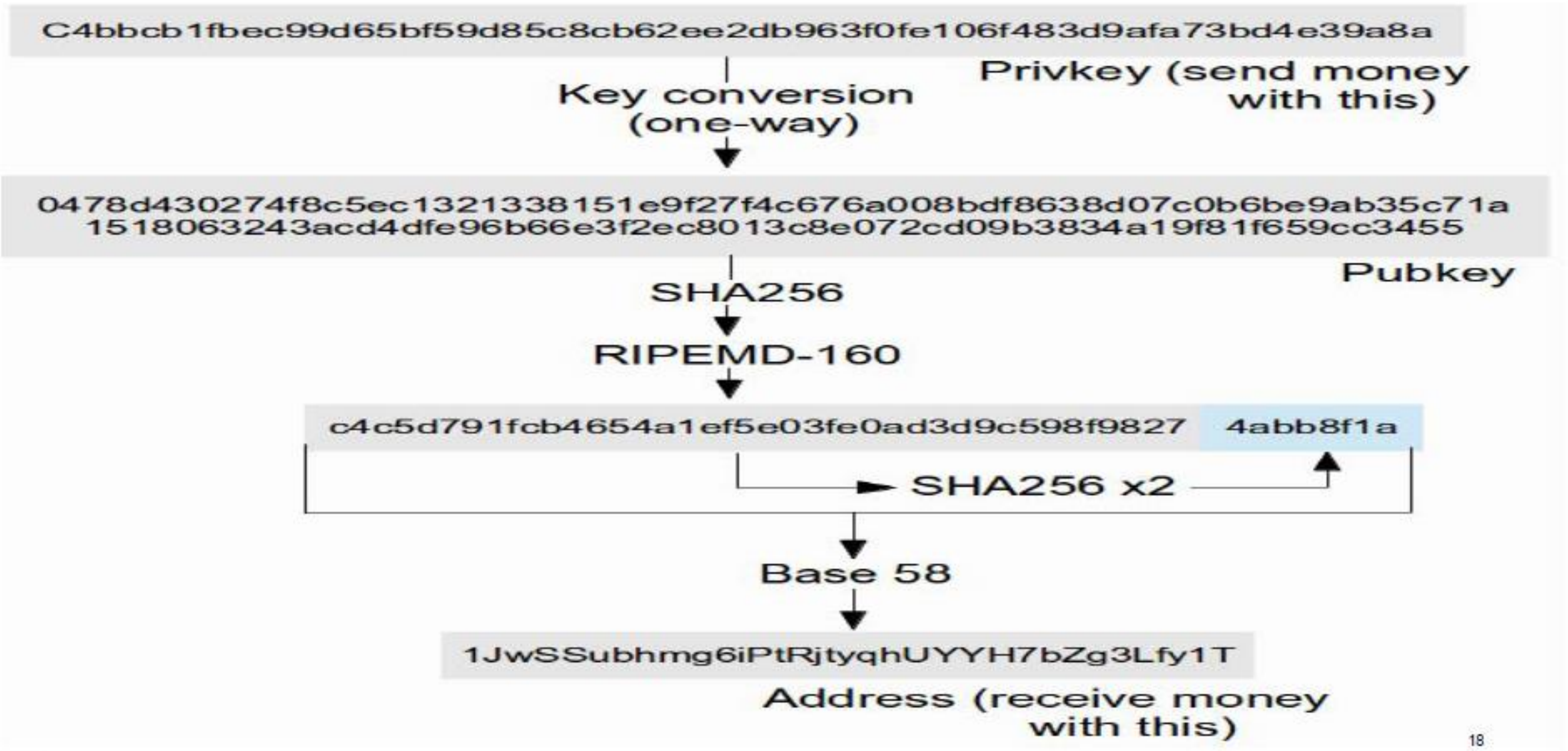
Bitcoin Address

Determined by – but not identical to - Public Key



Source : MIT OpenCourseWare, https://ocw.mit.edu/ 15.S12 Blockchain and Money Fall 2018

Bitcoin Addresses



Source : MIT OpenCourseWare, https://ocw.mit.edu/ 15.S12 Blockchain and Money Fall 2018

Base58 Encoding

1EHNa6Q4Jz2uvNExL497mE43ikXhwF6kZm

$$\updownarrow$$

0091B24BF9F5288532960AC687ABB035127B1D28A50074FFE0

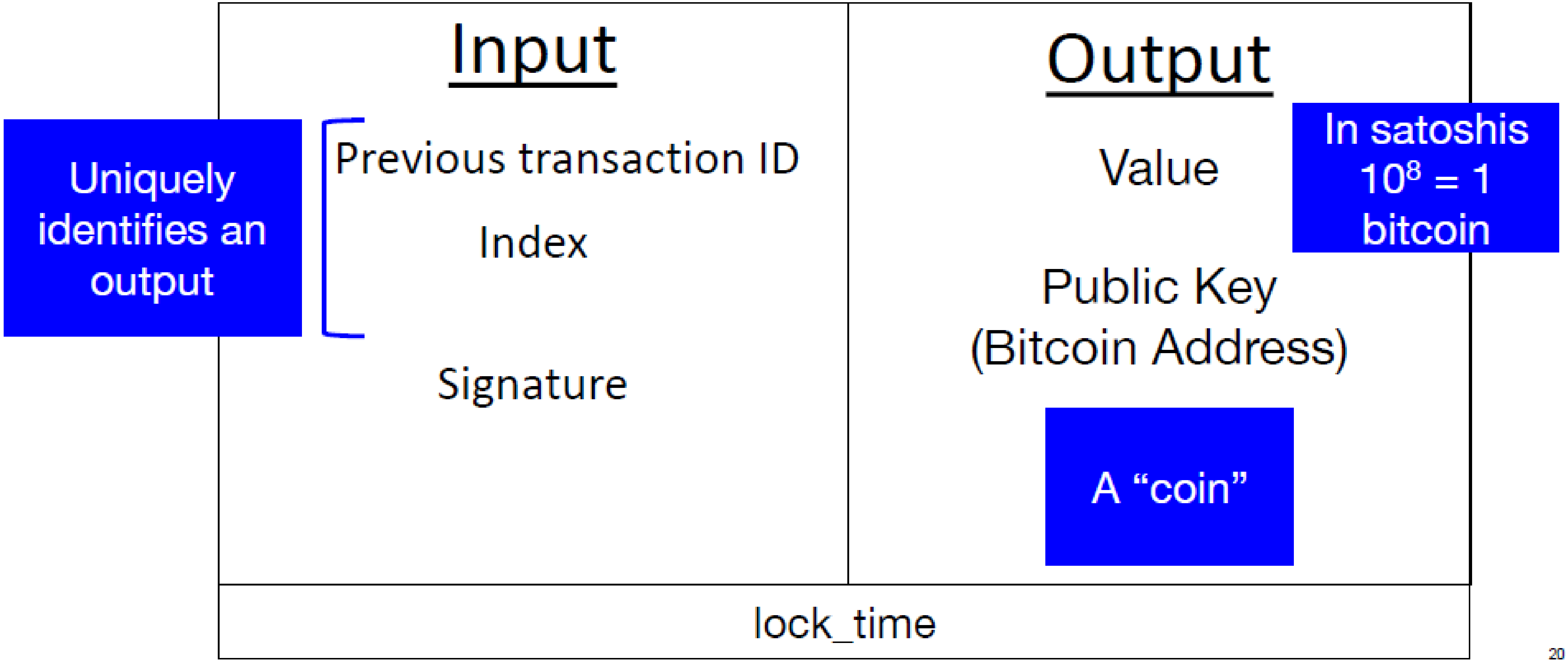
- Alphanumeric representation of bytestrings
- From 62 alphanumeric characters 0, O, I, l are excluded

Ch	Int	Ch	Int	Ch	Int	Ch	Int	Ch	Int	Ch	Int	Ch	Int
1	0	A	9	K	18	U	27	d	36	n	45	w	54
2	1	B	10	L	19	V	28	e	37	o	46	x	55
3	2	C	11	M	20	W	29	f	38	p	47	y	56
4	3	D	12	N	21	X	30	g	39	q	48	z	57
5	4	E	13	P	22	Y	31	h	40	r	49		
6	5	F	14	Q	23	Z	32	i	41	s	50		
7	6	G	15	R	24	a	33	j	42	t	51		
8	7	H	16	S	25	b	34	k	43	u	53		
9	8	J	17	T	26	c	35	m	44	v	53		

- Given a bytestring $b_n b_{n-1} \cdots b_0$
  - Encode each leading zero byte as a 1
  - Get integer $N = \sum_{i=0}^{n-m} b_i 256^i$
  - Get $a_k a_{k-1} \cdots a_0$ where $N = \sum_{i=0}^{k} a_i 58^i$
  - Map each integer a_i to a Base58 character

Source : Slides from ‘An Introduction to Bitcoin’ by Prof. Saravanan Vijayakumaran, IIT Madras

Transaction format

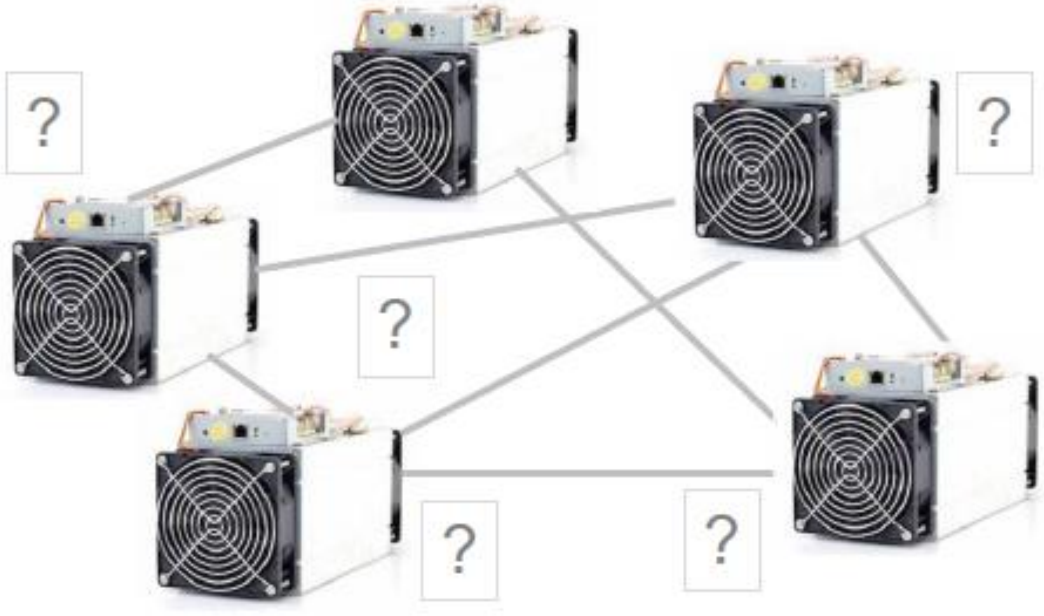


Source : MIT OpenCourseWare, https://ocw.mit.edu/ 15.S12 Blockchain and Money Fall 2018

Decentralized Networks

Byzantine Generals Problem

Permissionless Blockchains - **Unknown** participants



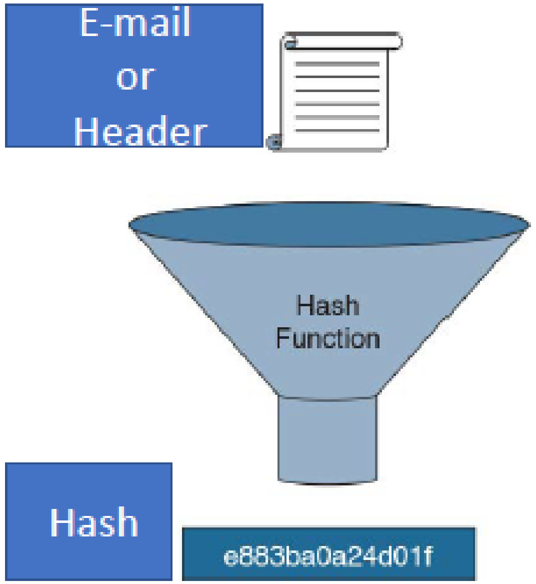
Security based on:

- Consensus protocol &
- Native currency

Hashcash – Proof of Work (Adam Back, 1997)

Proposed to address E-mail Spam and Denial of Service attacks

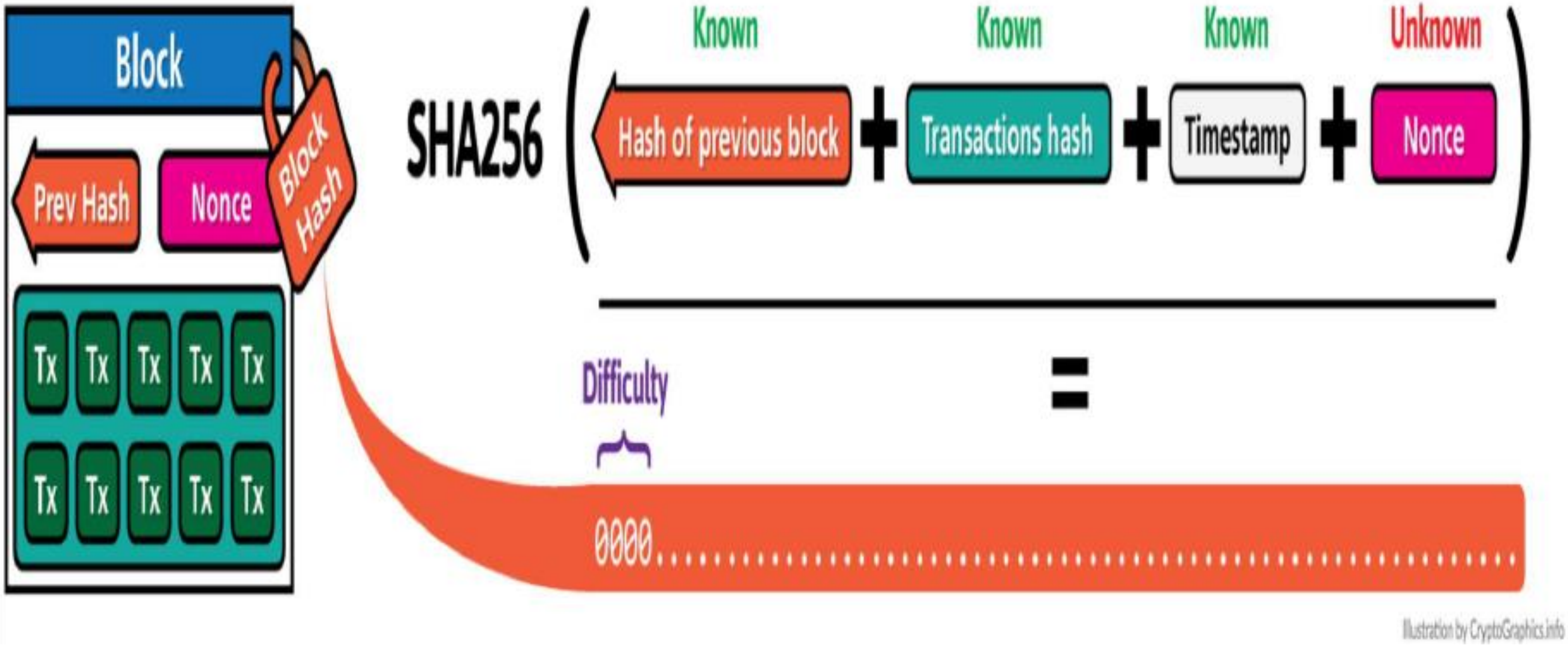
- Requires computational work to find a hash within predetermined range



- Difficulty defined by Hash outputs' # of leading zeros
- Proof of Work can be Efficiently Verified

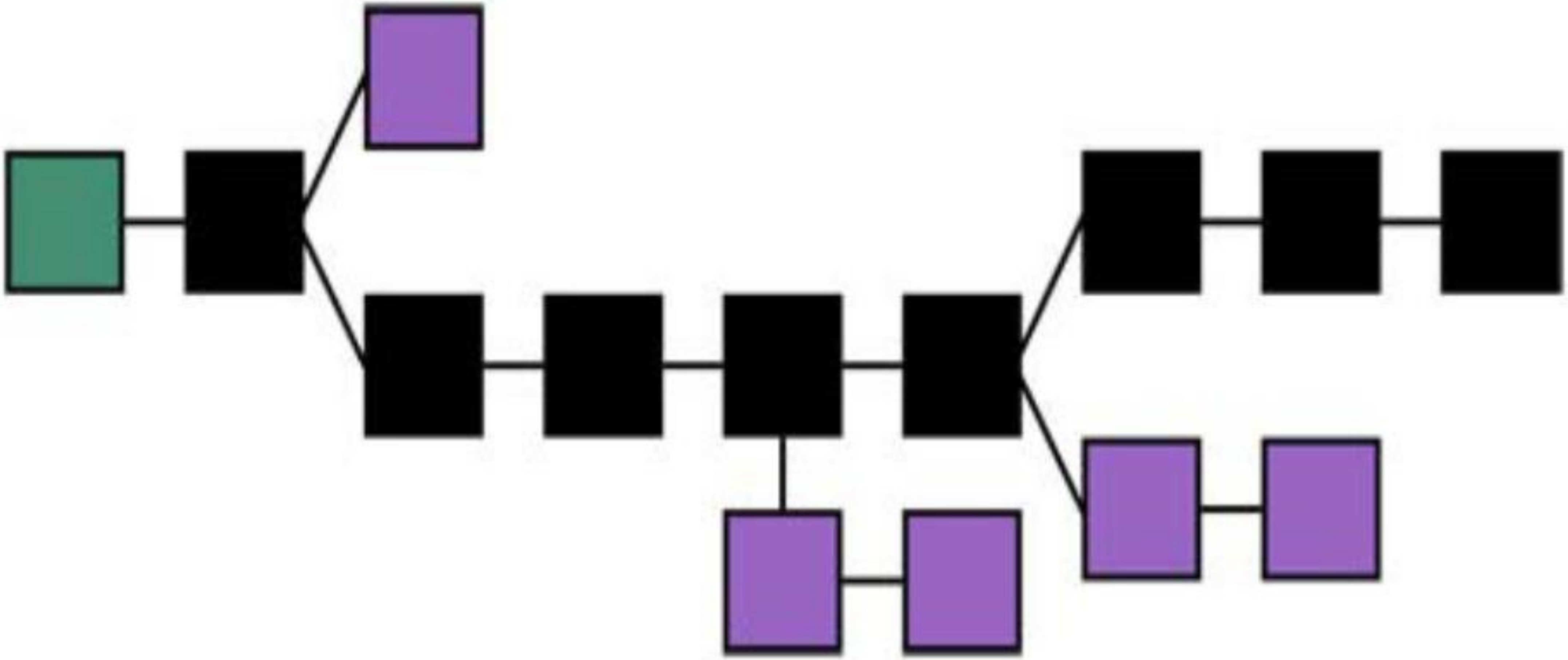
Blockchain – Proof of Work

Innovation – Chained Proof of Work for Distributed Network Consensus & Timestamping



Source : MIT OpenCourseWare, https://ocw.mit.edu/ 15.S12 Blockchain and Money Fall 2018

Blockchain – Consensus supports Longest Chain

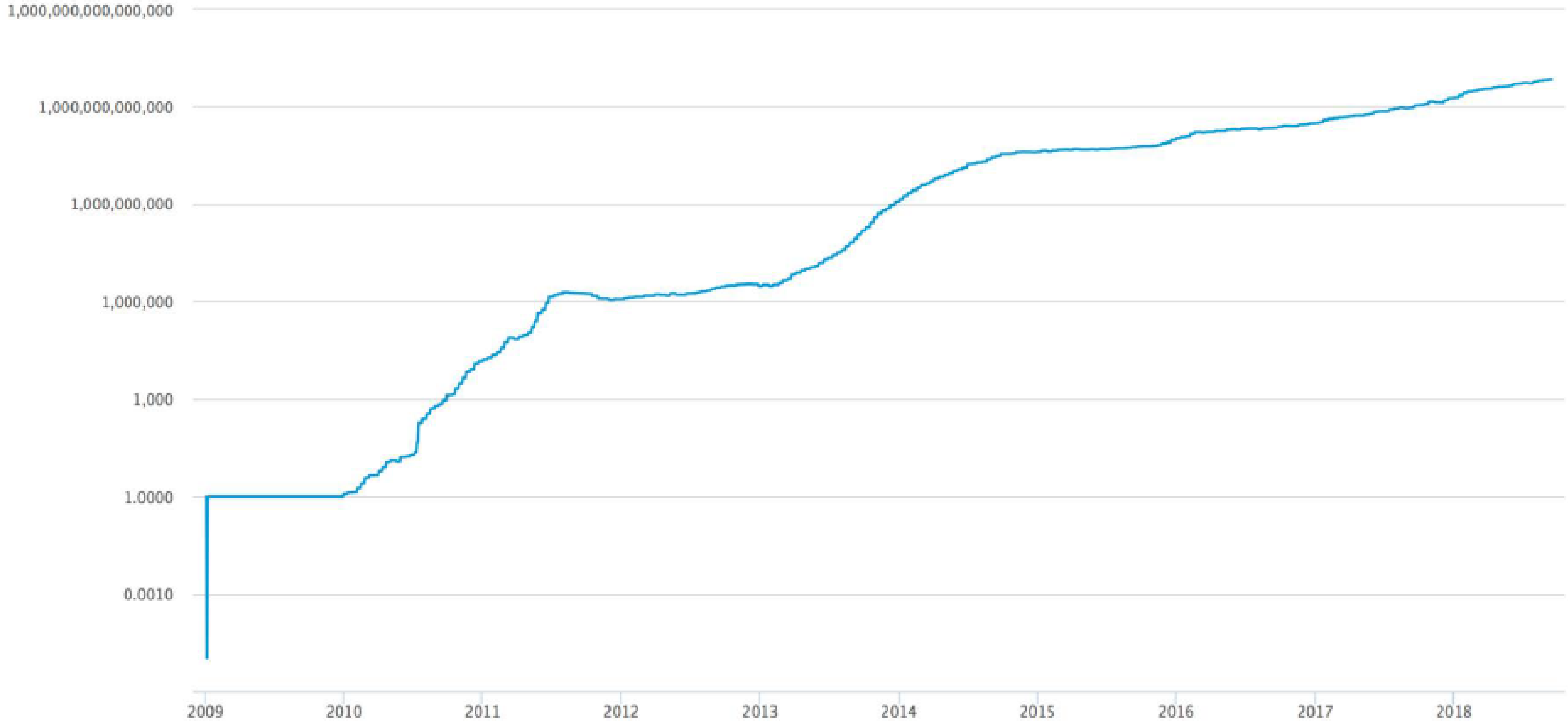


Source : MIT OpenCourseWare, https://ocw.mit.edu/ 15.S12 Blockchain and Money Fall 2018

Bitcoin Proof of Work Difficulty

- Targets 10 minute average block generation time
- Defined by the # of leading zeros Hash output requires to solve proof of work
- Adjusts every 2016 blocks - about every two weeks
- Currently, $\geq$ 18 leading zeros (out of 64 hexadecimal characters)
- Block 541974 (9/18/18)- 18 leading zeros
  0000000000000000001104a863046dfbad1a2941128815669623ff93c2a3945f
- Genesis Block (1/3/09) – 10 leading zeros, though only required 8
  000000000019d6689c085ae165831e934ff763ae46a2a6c172b3f1b60a8ce26f

Bitcoin Mining Difficulty



Source : MIT OpenCourseWare, https://ocw.mit.edu/ 15.S12 Blockchain and Money Fall 2018

Bitcoin Mining Evolution

Central Processing Units
(CPUs) 2009 – 2010
2 - 20 MH/S

Graphics Processing Units
(GPUs) 2010 – 2013
20 - 300 MH/S

Image is in the public domain.

Application Specific Integrated Circuit
(ASICs) 2013 – 2018
4 – 16 TH/S

Image by InstagramFOTOGRAFIN on Pixabay.

Modern Mining Factory

Source : MIT OpenCourseWare, https://ocw.mit.edu/ 15.S12 Blockchain and Money Fall 2018

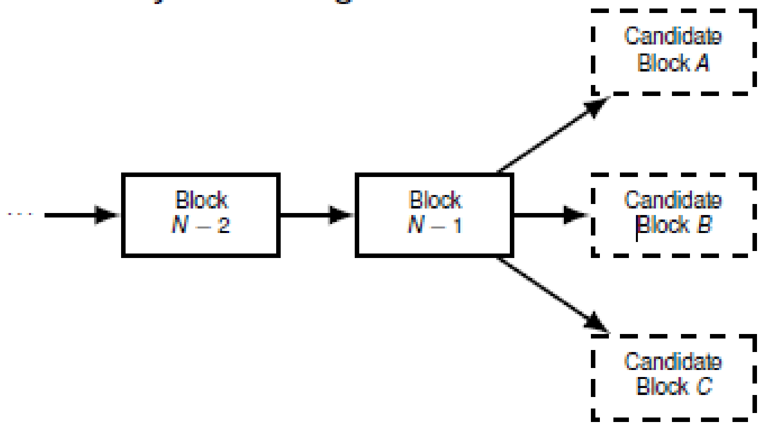
Why should anyone mine blocks?

- Successful miner gets rewarded in bitcoins
- Every block contains a **coinbase transaction** which creates 12.5 bitcoins
- Each miner specifies his own address as the destination of the new coins
- Every miner is competing to solve their own PoW puzzle
- Miners also collect the transaction fees in the block

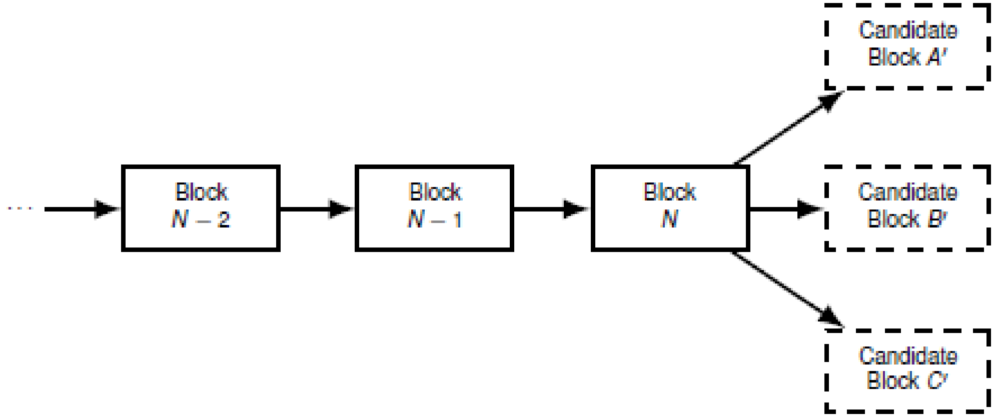
Source : Slides from ‘An Introduction to Bitcoin’ by Prof. Saravanan Vijayakumaran, IIT Madras

Block Addition Workflow

- Nodes broadcast transactions
- Miners accept valid transactions and reject invalid ones (solves double spending)
- Miners try extending the latest block

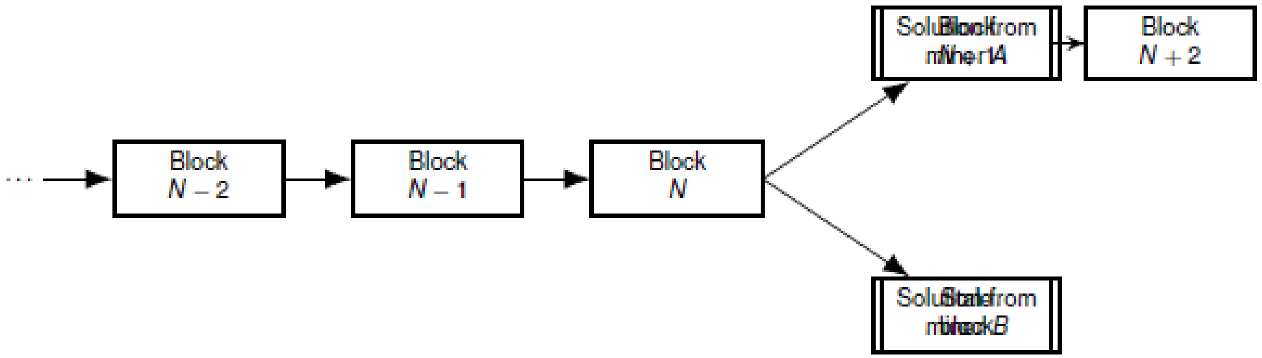


- Miners compete to solve the search puzzle and broadcast solutions
- Unsuccessful miners abandon their current candidate blocks and start work on new ones

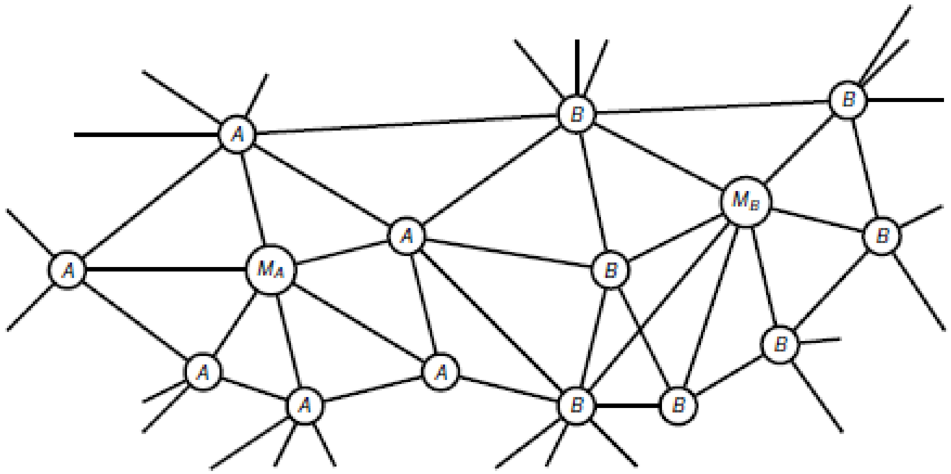


Source : Slides from 'An Introduction to Bitcoin' by Prof. Saravanan Vijayakumaran, IIT Madras

What if two miners solve the puzzle at the same time?



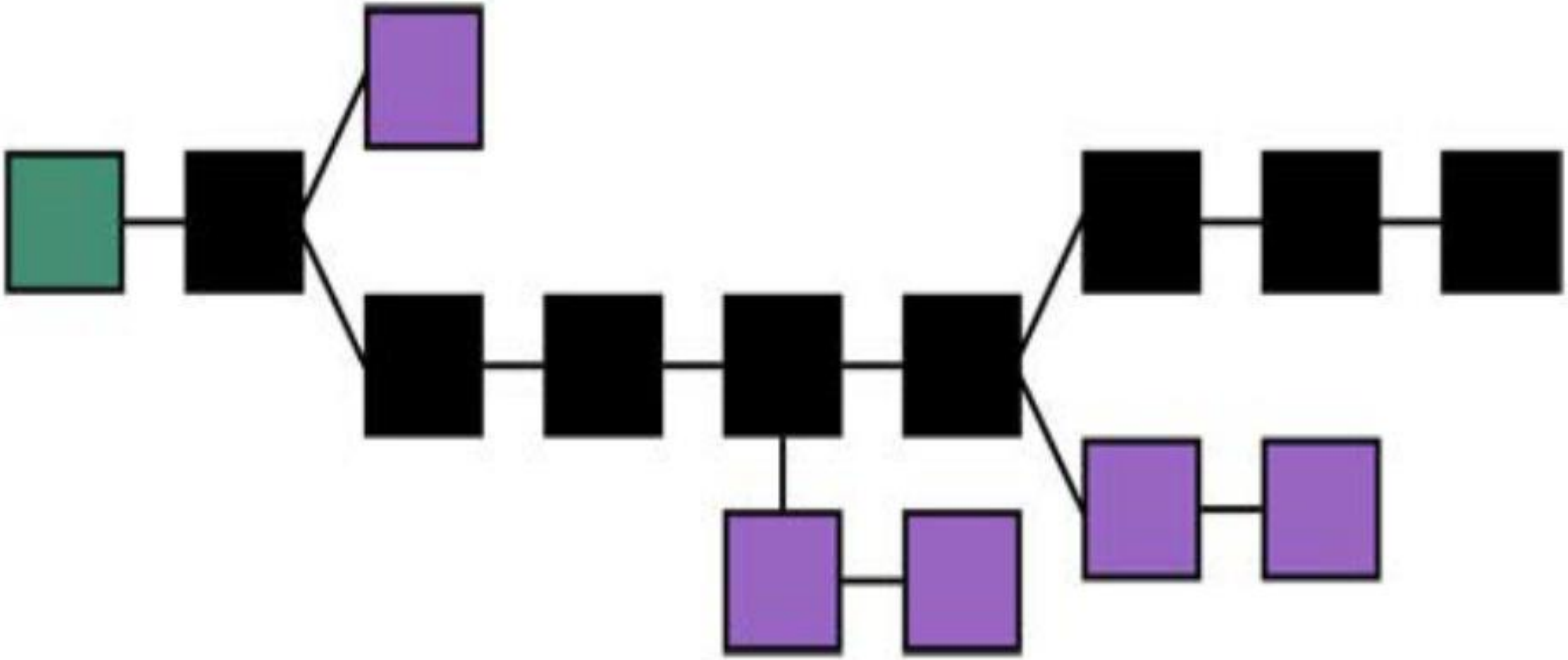
- Both miners will broadcast their solution on the network
- Nodes will accept the first solution they hear and reject others



- Nodes always switch to the longest chain they hear
- Eventually the network will converge and achieve consensus

Source : Slides from ‘An Introduction to Bitcoin’ by Prof. Saravanan Vijayakumaran, IIT Madras

Blockchain – Consensus supports Longest Chain



Source : MIT OpenCourseWare, https://ocw.mit.edu/ 15.S12 Blockchain and Money Fall 2018

How often are new blocks created?

- Once every 10 minutes

nVersion
hashPrevBlock
hashMerkleRoot
nTime
nBits
nNonce

- Every 2016 blocks, the target T is recalculated
- Let t_{sum} = Number of seconds taken to mine last 2016 blocks

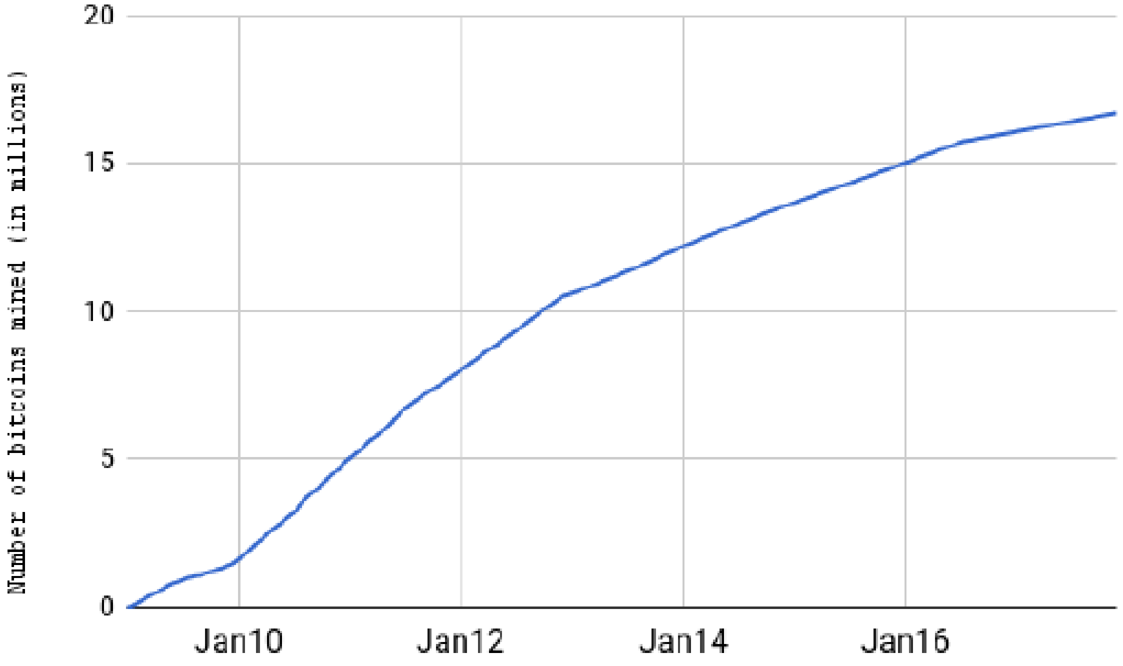
$$T_{\text{new}} = \frac{t_{\text{sum}}}{14 \times 24 \times 60 \times 60} \times T$$

- Recall that probability of success in single trial is $\frac{T+1}{2^{256}}$
- If $t_{\text{sum}} = 2016 \times 8 \times 60$, then $T_{\text{new}} = \frac{4}{5}T$
- If $t_{\text{sum}} = 2016 \times 12 \times 60$, then $T_{\text{new}} = \frac{6}{5}T$

Source : Slides from 'An Introduction to Bitcoin' by Prof. Saravanan Vijayakumaran, IIT Madras

Bitcoin Supply

- The block subsidy was initially 50 BTC per block
- Halves every 210,000 blocks $\approx$ 4 years
- Became 25 BTC in Nov 2012 and 12.5 BTC in July 2016
- Total Bitcoin supply is 21 million



- The last bitcoin will be mined in 2140

Source : Slides from ‘An Introduction to Bitcoin’ by Prof. Saravanan Vijayakumaran, IIT Madras

Bitcoin Payment Workflow

- Merchant shares address out of band (not using Bitcoin P2P)
- Customer broadcasts transaction t which pays the address
- Miners collect broadcasted transactions into a candidate block

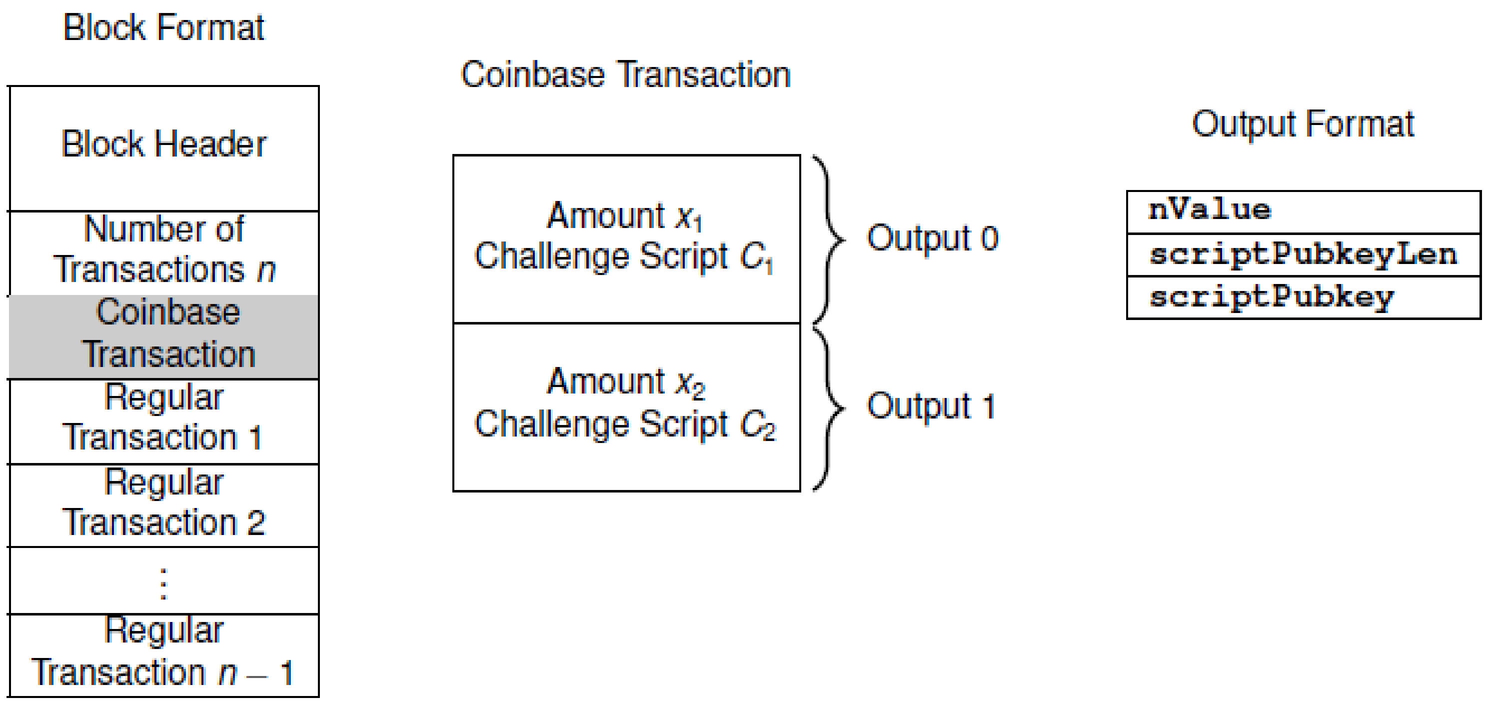
Block Header
Number of Transactions n
Coinbase Transaction
Regular Transaction 1
Regular Transaction 2
$\vdots$
Regular Transaction $n-1$

- One of the candidate blocks containing t is mined
- Merchant waits for confirmations on t before providing goods

Source : Slides from ‘An Introduction to Bitcoin’ by Prof. Saravanan Vijayakumaran, IIT Madras

Coinbase Transaction Format

Pre-SegWit

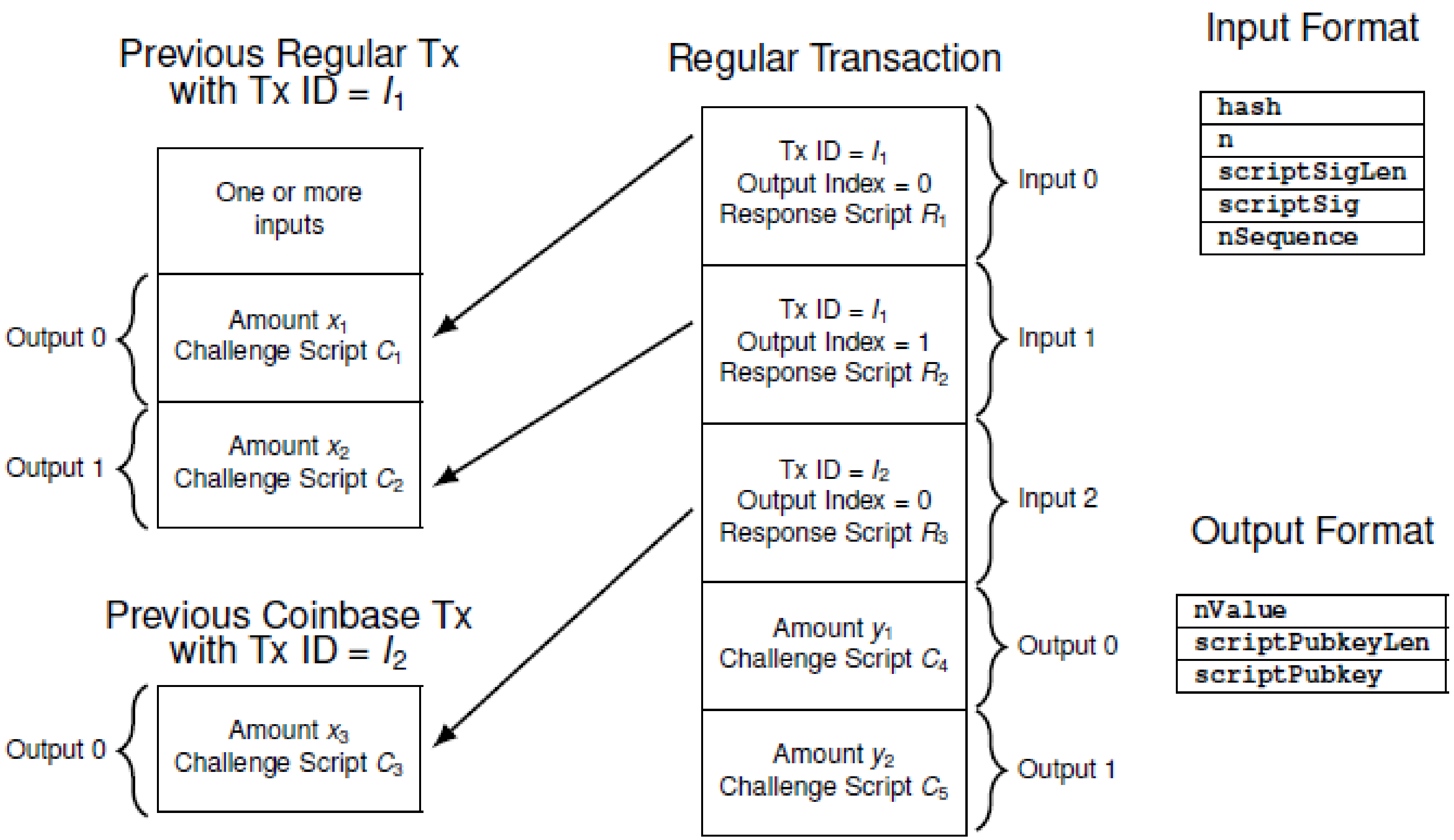


- `nValue` contains number of satoshis locked in output
  - 1 Bitcoin = 10^8 satoshis
- `scriptPubkey` contains the challenge script
- `scriptPubkeyLen` contains byte length of challenge script

Source : Slides from 'An Introduction to Bitcoin' by Prof. Saravanan Vijayakumaran, IIT Madras

Regular Transaction Format

Pre-SegWit



- `hash` and `n` identify output being unlocked
- `scriptSig` contains the response script

Source : Slides from 'An Introduction to Bitcoin' by Prof. Saravanan Vijayakumaran, IIT Madras

Transaction format

Uniquely identifies an output

Input	Output
Previous transaction ID	Value
Index	Public Key (Bitcoin Address)
Signature	

In satoshis $10^8 = 1$ bitcoin

A “coin”

lock_time

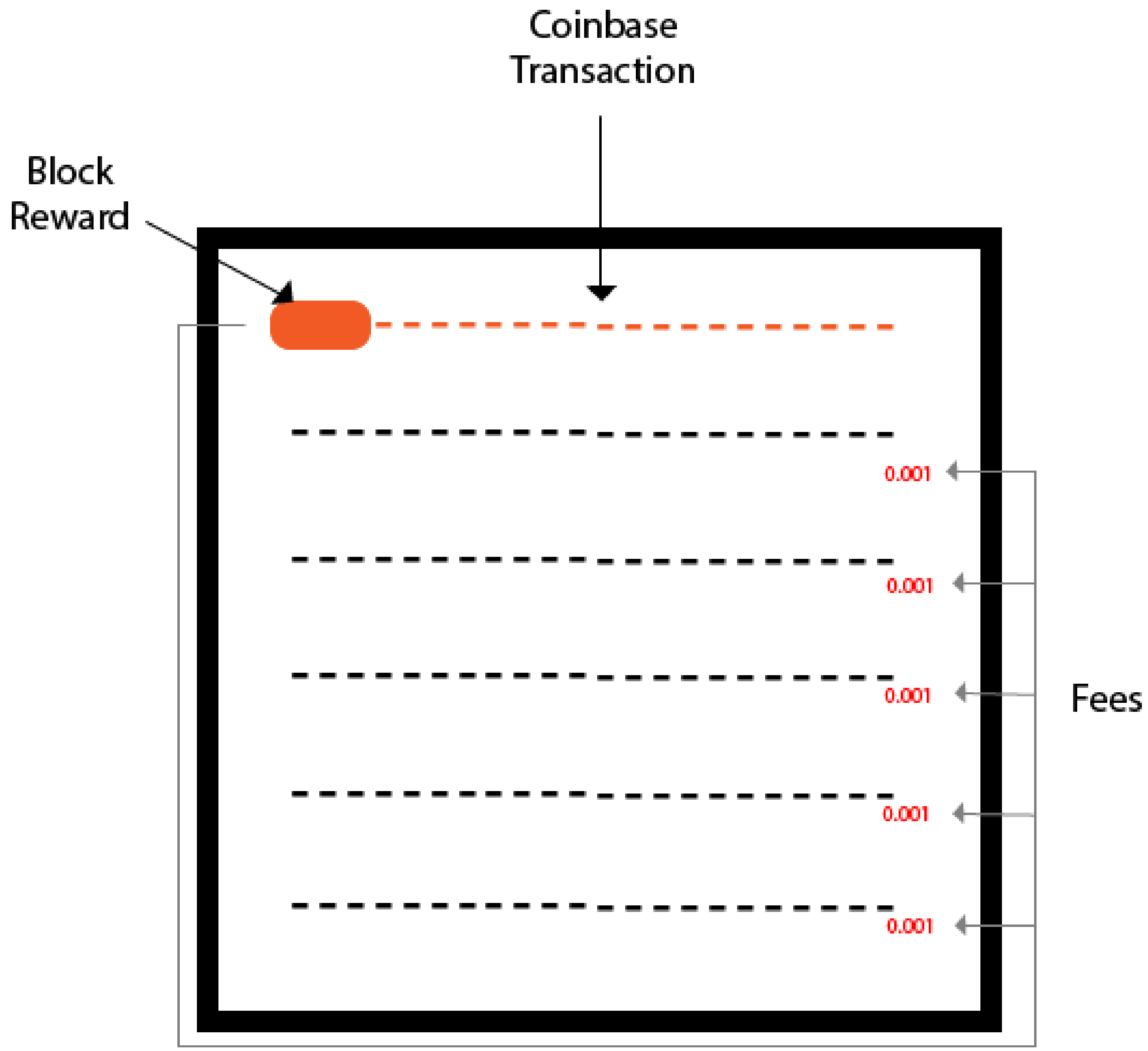
Source : MIT OpenCourseWare, https://ocw.mit.edu/ 15.S12 Blockchain and Money Fall 2018

Coinbase Transaction:

- A coinbase transaction is the **first** transaction in a block.
- It is a **unique** type of bitcoin transaction that can be created by a miner.
- The miners use it to collect the block reward for their work and any other transaction fees collected by the miner are also sent in this transaction.
- Each transaction executed on the bitcoin network are combined together to form a block.

Coinbase Transaction:

- When a block is formed, immediately, it will be added in the blockchain.
- Now, these blocks are immutable and tamper-proof for all transactions that are made on the bitcoin network.
- Each block must contain one or more transactions, and the first transaction in the block is called the coinbase transaction.


Coinbase
Transaction
Block
Reward
0.001
0.001
0.001
0.001
0.001
Fees

Coinbase Transaction:

- The miners are always responsible for creating a block. When a block is successfully created, he will be rewarded from bitcoin for their work.
- The bitcoin block reward is always dependent on the number of blocks from the genesis block and the number of fees included in the transactions of the block.
- The total amount of rewards that a miner will collect is the sum of the block reward and the transaction fees taken from all the transactions that have been included in the block.
- In the start of the bitcoin, the block reward is 50 bitcoin per block.
- The block reward is reduced by half after every 210, 000 blocks, i.e. approximately in every four years.
- The current reward for successfully creating a block is 12.5 bitcoin.
- It will be going to get reduced 6.25 bitcoin per block in the year 2020.
- There is one important feature of a coinbase transaction is that bitcoins involved in the transaction cannot be spent until they have received at least 100 block confirmations in the blockchain.

Native Currency

Economic Incentive System
'Monetary Policies' vary widely

- Bitcoin - BTC
  - Created through Coinbase Transaction in each block
  - 'Monetary Policy' preset in Bitcoin Core
  - Creation originally 50 Bitcoin per block
  - Reward halves (1/2s) every 210,000 blocks
  - Currently 12.5 BTCs created per block – thus 'inflation' 4.1%
  - Currently 17.3 million BTC; capping at 21 million BTC in 2040
  - Market based transaction fee mechanism also provided for in Bitcoin Core
- Ethereum
  - Currently 3 ETH per block – thus 'inflation' 7.4%
  - Recent proposal to decline to 2 ETH per block in 11/18
  - Fees paid in Gas (10^9 Gas per ETH) for computation are credited to miners

Source : MIT OpenCourseWare, https://ocw.mit.edu/ 15.S12 Blockchain and Money Fall 2018

Network

- Full Nodes – Store full Blockchain & able to Validate all Transactions
- Pruning Nodes – Prune transactions after validation and aging
- Lightweight Nodes - Simplified Payment Verification (SPV) nodes – Store Blockchain Headers only
- Miners – Performs Proof of Work & Create new Blocks - Do not need to be a Full Node
- Mining Pool Operators
- Wallets – Store, View, Send and Receive Transactions & Create Key Pairs
- Mempool – Pool of unconfirmed (yet validated) Transactions

Source : MIT OpenCourseWare, https://ocw.mit.edu/ 15.S12 Blockchain and Money Fall 2018

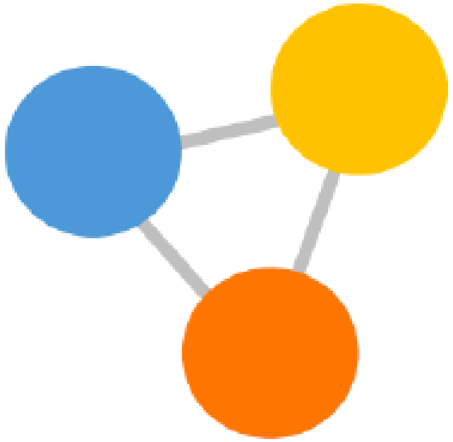
Alternative Consensus Protocols

Generally Randomized or Delegated Selection of Nodes to Validate next Block

- May have added mechanism to confirm Block Validators' Work

Randomized Selection May be Based upon:

- Proof of Stake – Stake in Native Currency
- Proof of Activity - Hybrid of POW and POS
- Proof of Burn – Validation comes with Burning of Coins
- Proof of Capacity (Storage or Space) – Based upon Hardware Space



Delegated Selection May be Based upon Tiered System of Nodes

Major Permissionless Blockchain Applications still use Proof of Work – though:

- DASH is a hybrid of POW with a tiered system of 'Masternodes'
- NEO uses a Delegated protocol of 'Professional Nodes'

UTXO model

Inputs ≥ Outputs

Inputs – Outputs = Fees

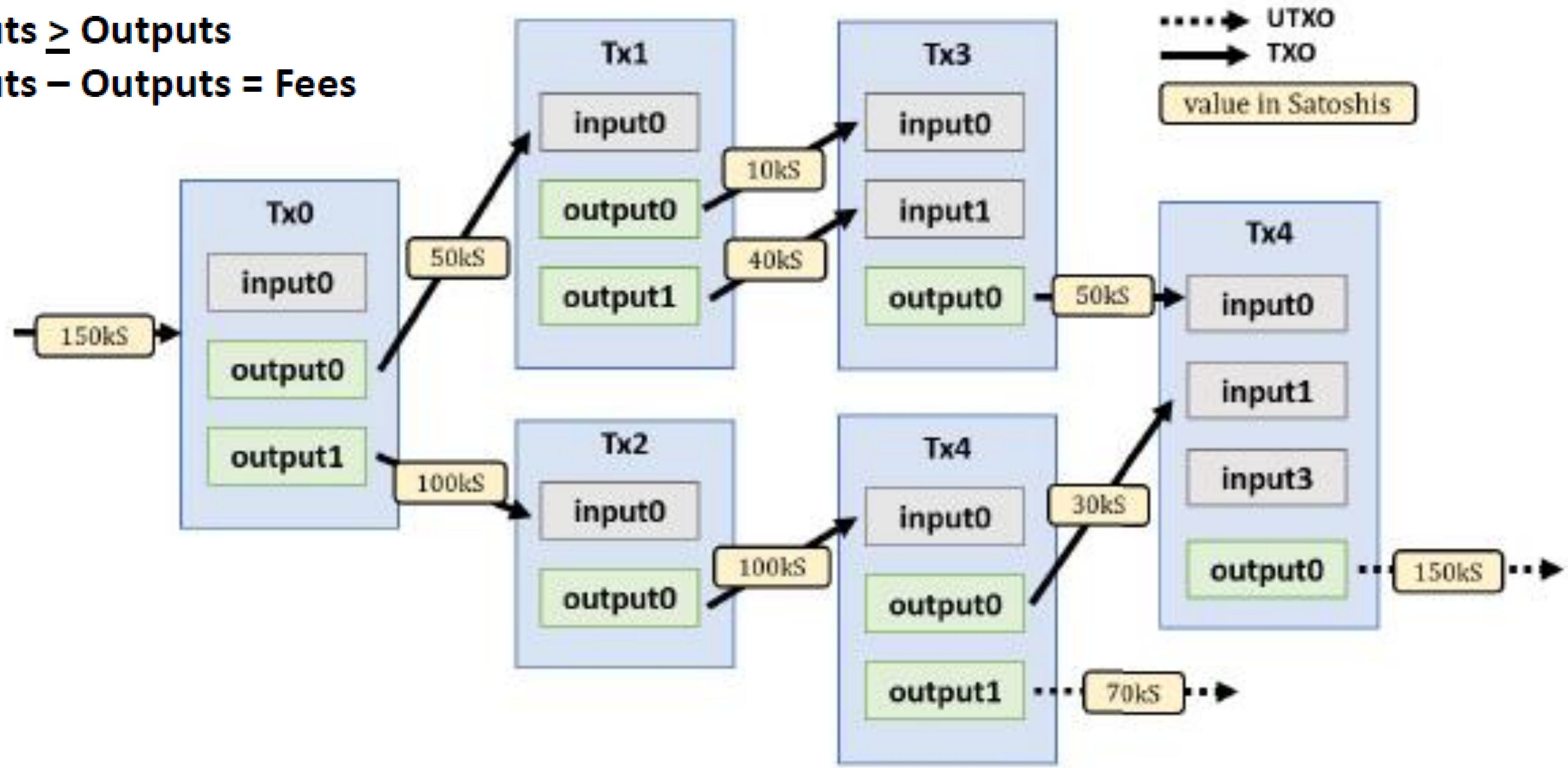


Fig. 9. An example of UTXO-based transfers in Bitcoin.

[Source – Belotti, Marianna, et al. "A vademecum on blockchain technologies: When, which, and how." *IEEE Communications Surveys & Tutorials* 21.4 (2019): 3796-3838.]

Unspent Transaction Output (UTXO) Set

Bitcoin transaction outputs that have not been spent at a given time

- Contains All Currently Unspent Transaction Outputs
- Speeds up Transaction Validation Process
- Stored using a LevelDB database in Bitcoin Core called ‘chainstate’

Bitcoin Script

Programing Code used for Transactions

- Stack-based Code, with no Loops (not Turing-complete)
- Provides a Flexible Set of Instructions for Transaction Validation and Signature Authentication
- Most Common Script Types in UTXO:
  - Transaction sent to Hash of Bitcoin Address – 'Pay-to-PubkeyHash' (81%)
  - Transaction sent to Hash of Conditional Script – 'Pay-to-ScriptHash' (18%)
  - Transaction subject to Multiple Signatures – 'M of N Multisig' (0.7%)
  - Transaction sent to Bitcoin Address – 'Pay-to-Pubkey' (0.1%)

    (Source: Perez-Sola, Delgado-Segura, et al.)

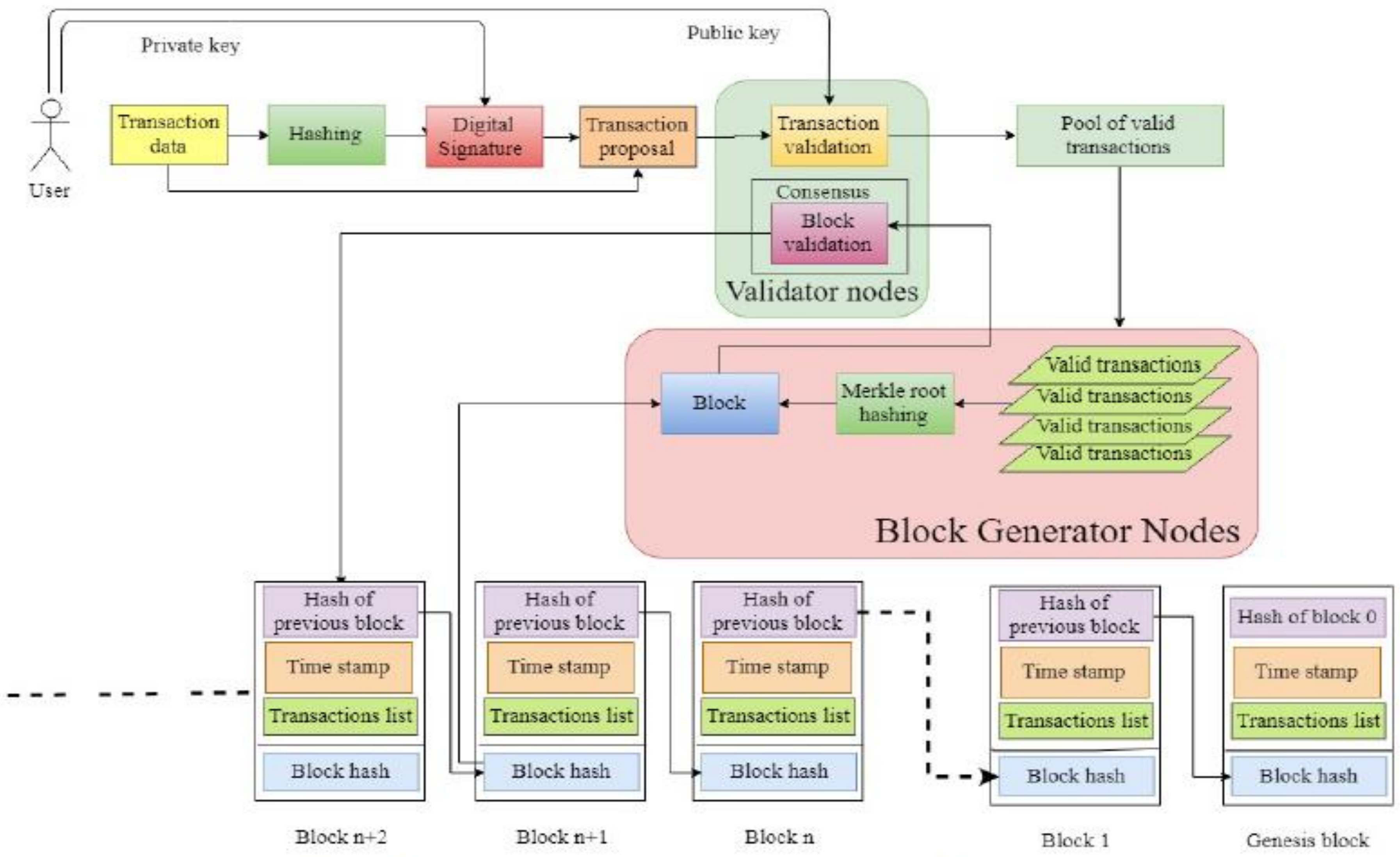
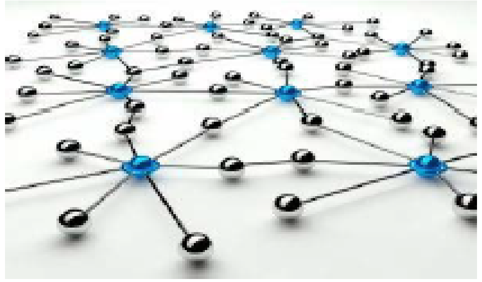


Figure 2. Overview of Transaction Execution Flow in Blockchain.

[Source - Ismail, Leila, and Huned Materwala. "A review of blockchain architecture and consensus protocols: Use cases, challenges, and solutions." *Symmetry* 11.10 (2019): 1198]

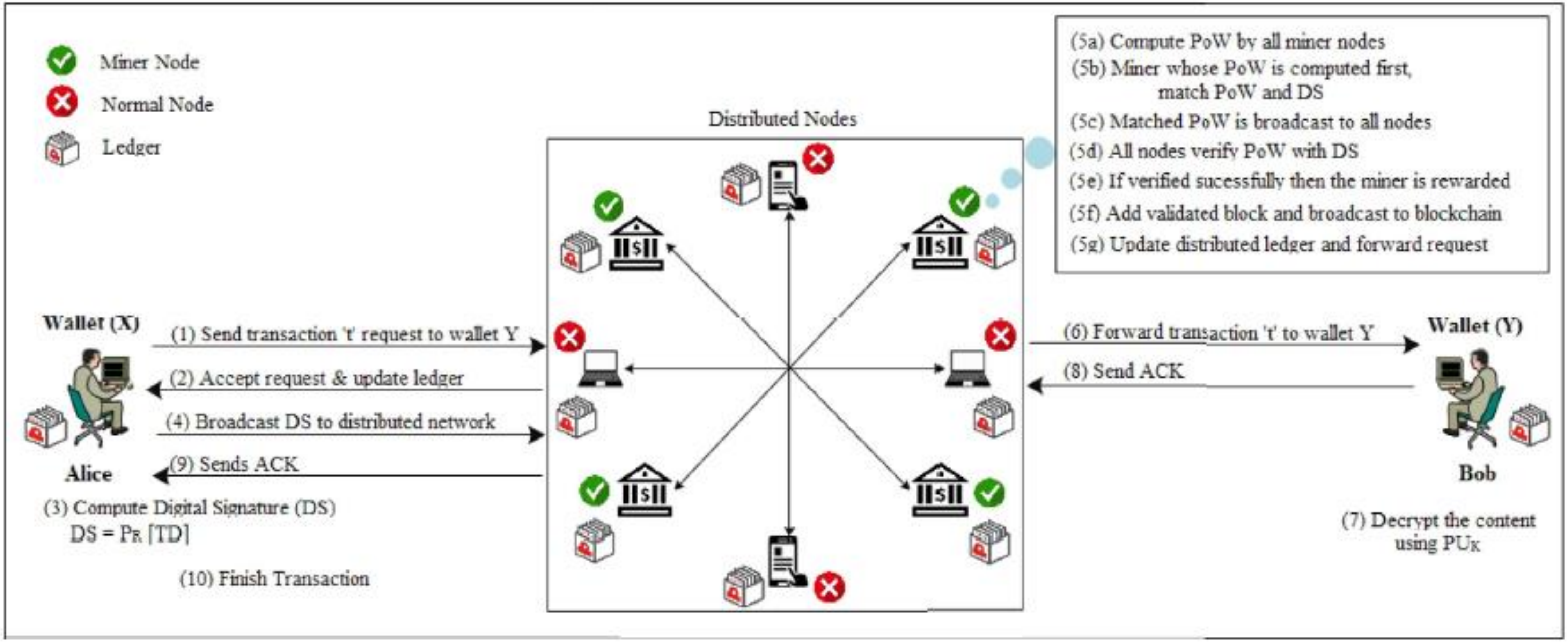
Network



- Full Nodes – Store full Blockchain & able to Validate all Transactions
- Pruning Nodes – Prune transactions after validation and aging
- Lightweight Nodes - Simplified Payment Verification (SPV) nodes – Store Blockchain Headers only
- Miners – Performs Proof of Work & Create new Blocks - Do not need to be a Full Node
- Mining Pool Operators
- Wallets – Store, View, Send and Receive Transactions & Create Key Pairs
- Mempool – Pool of unconfirmed (yet validated) Transactions

Source : MIT OpenCourseWare, https://ocw.mit.edu/ 15.S12 Blockchain and Money Fall 2018

Steps of bitcoin transaction



[Source : Aggarwal, Shubhani, et al. "Blockchain for smart communities: Applications, challenges and opportunities." *Journal of Network and Computer Applications* 144 (2019): 13-48.]

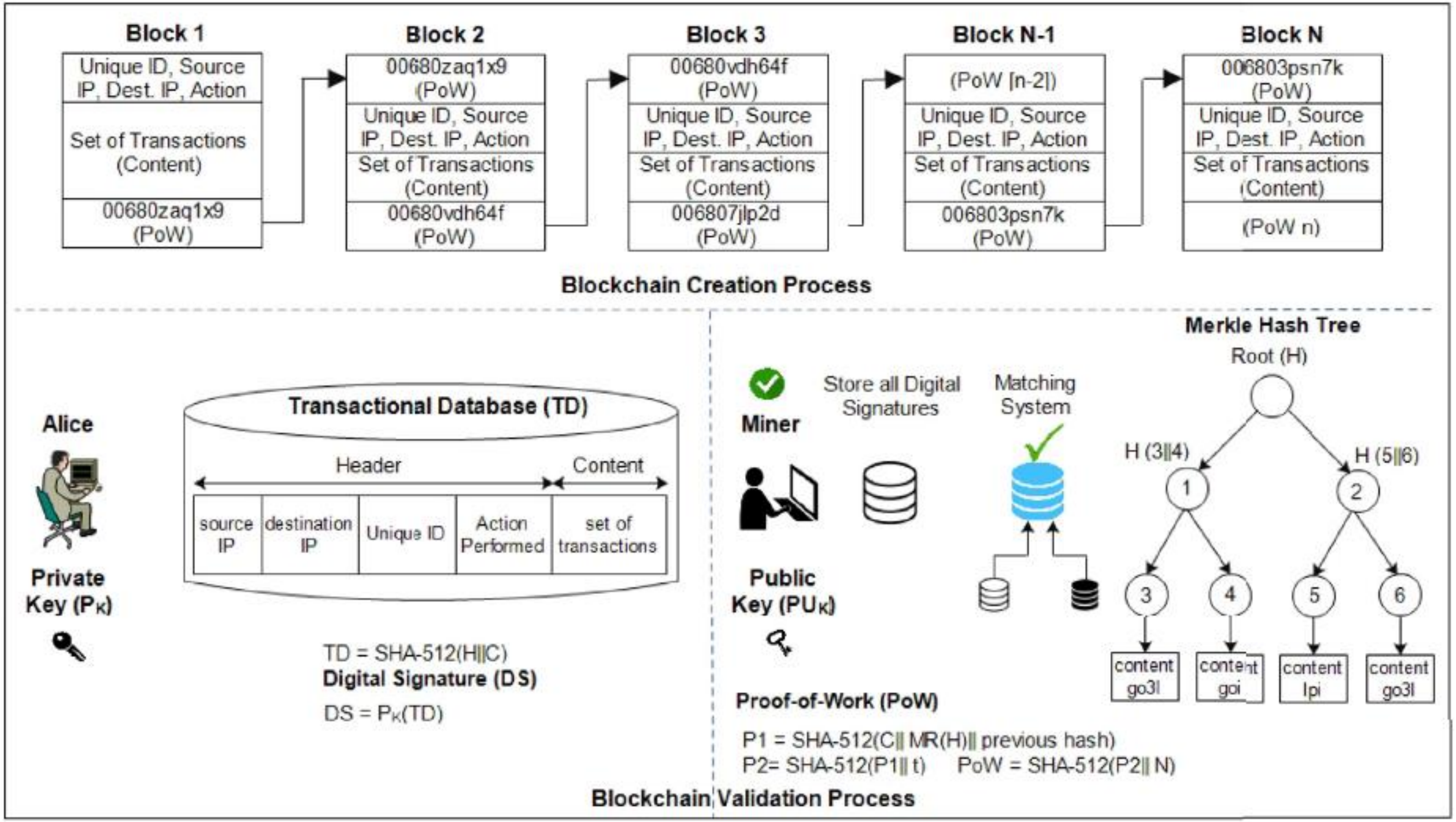


Fig. 5. Block creation and block validation process.

[Source - Ismail, Leila, and Huned Materwala. "A review of blockchain architecture and consensus protocols: Use cases, challenges, and solutions." *Symmetry* 11.10 (2019): 1198]