BLOCKCHAIN TECHNOLOGY

Introducing Bitcoin

Introduction

- Bitcoin
  - Is the first application of blockchain technology
  - has started a revolution with the introduction of the very first fully decentralized digital currency
  - has proven to be extremely secure and stable from a network and protocol point of view
- The identity of Bitcoin inventor is unknown
  - Satoshi Nakamoto is believed to be a pseudonym

A brief history

- In 1982 a scheme was proposed that
    - used blind signatures to build untraceable digital currency
    - a bank would issue digital money
        - by signing a blind and random serial number
        - presented to it by the user
    - The user could then use the digital token signed by the bank as currency.
    - The limitation was that the bank had to keep track of all used serial numbers.
    - This was
        - a central system by design
        - required to be trusted by the users

A brief history

- In 1988 a refined version named e-cash was proposed that
  - allowed the detection of double spending
    - but did not prevent it.
  - If the same token was used at two different locations
    - the identity of the double spender would be revealed

A brief history

- hashcash was introduced in 1997
  - was originally proposed to thwart email spam
  - The idea was to solve a computational puzzle that
    - was easy to verify
    - but comparatively difficult to compute
  - The idea was
    - for a single user and a single email
      - the extra computational effort was negligible
    - but someone sending many spam emails
      - the needed time and resources would increase substantially

A brief history

- In 1998, B-money was proposed
  - introduced the idea of using Proof of Work (PoW) to create money.
    - providing a solution to a previously unsolved computational problem.
  - Major weakness
    - an adversary with higher computational power could generate money
      - without allowing the network to adjust the difficulty level.

A brief history

- In 1999, an e-cash scheme was proposed
  - for the first time, used
    - Merkle trees to represent coins
    - Zero-Knowledge Proofs to prove the possession of coins
  - a central bank was required
    - kept a record of all used serial numbers.
  - allowed users to be fully anonymous.
  - was not practical to implement due to inefficient proof mechanisms

A brief history

- In 2008, Bitcoin
  - The paper named: “Bitcoin: A Peer-to-Peer Electronic Cash System”
  - is built on decades of cryptographic research
    - Merkle trees, hash functions, public key cryptography, and digital signatures.
    - ideas such as BitGold, B-money, hashcash
  - All technologies are cleverly combined
    - to create the world's first decentralized currency.
  - The key bitcoin innovations
    - an elegant solution to the Byzantine Generals' Problem
    - a practical solution of the double-spend problem

A brief history

- For libertarian people
  - Bitcoin is a platform which can be used instead of banks
  - If regulations require Know Your Customer (KYC) checks
    - then it might be too much information to share
    - as a result, Bitcoin may not be attractive anymore

A brief history

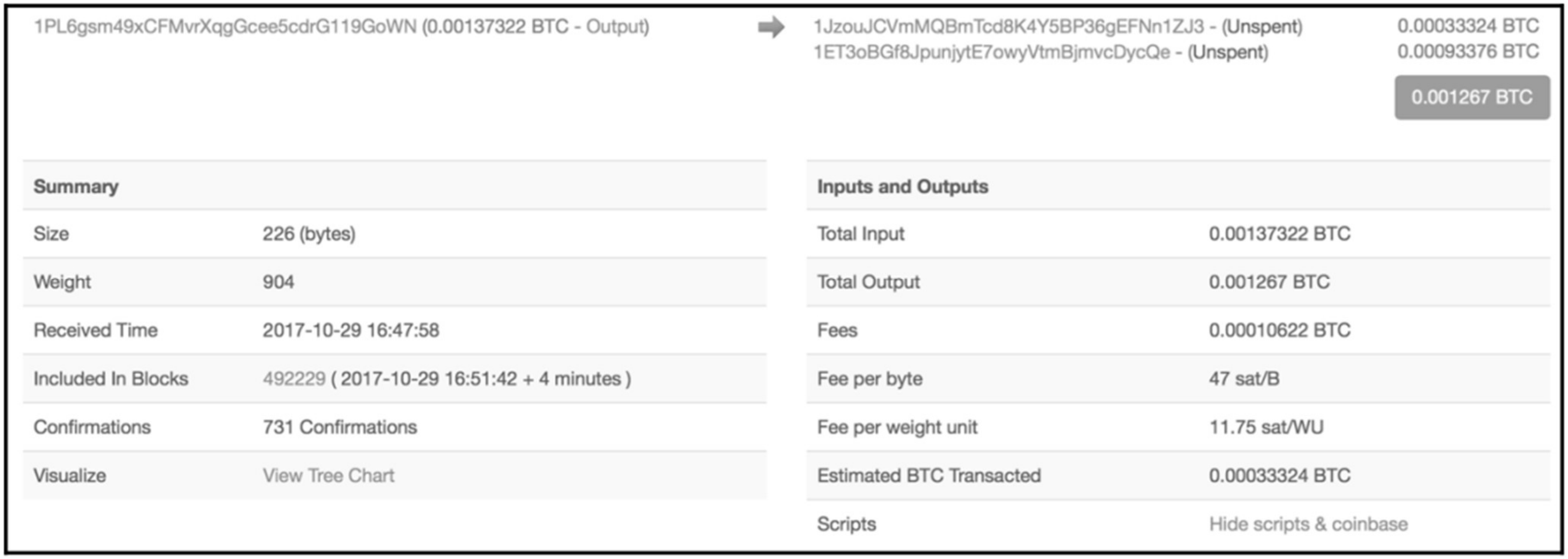
- The growth of Bitcoin is also due to network effect
    - a concept that means more users who use the network, the more valuable it becomes
    - Reasons
        - largely financial gain driven
        - built-in inflation control mechanism
            - there are only 21 million bitcoins that can ever be mined
            - miner reward halves every four years.

Bitcoin from a user's point of view

- Sending a payment to someone
  1. the receiver sends his Bitcoin address to the sender
     - via any appropriate communication mechanism
  2. the sender
     - constructs the transaction by following some rules
       - digitally signed using the private key of the sender
     - Broadcasts it to the Bitcoin network
  3. The transaction will be picked up by miners
     - to be verified and included in the block

Bitcoin from a user's point of view

- Sending a payment to someone



1PL6gsm49xCFMvrXqgGcee5cdrG119GoWN (0.00137322 BTC - Output) → 1JzouJCVmMQBmTcd8K4Y5BP36gEFNn1ZJ3 - (Unspent) 0.00033324 BTC
1ET3oBGf8JpunjytE7owyVtmBjmvcDycQe - (Unspent) 0.00093376 BTC

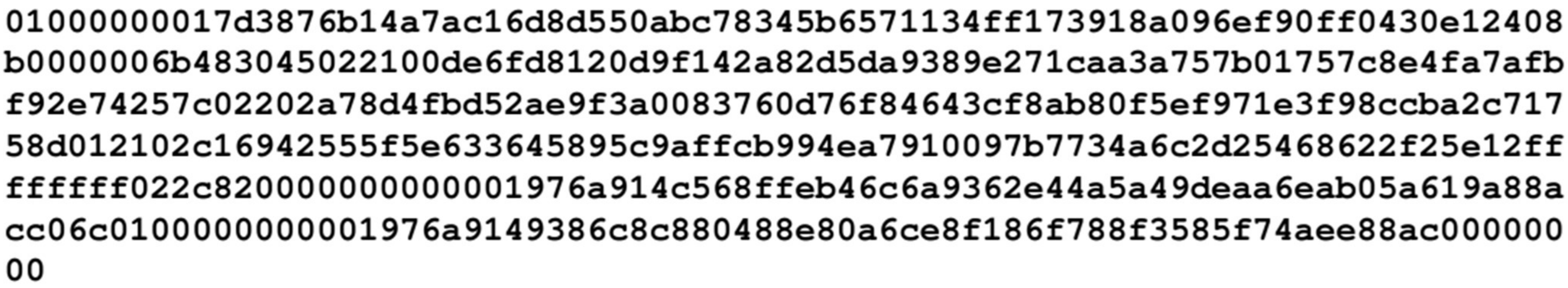
0.001267 BTC

Summary	
Size	226 (bytes)
Weight	904
Received Time	2017-10-29 16:47:58
Included In Blocks	492229 (2017-10-29 16:51:42 + 4 minutes)
Confirmations	731 Confirmations
Visualize	View Tree Chart

Inputs and Outputs	
Total Input	0.00137322 BTC
Total Output	0.001267 BTC
Fees	0.00010622 BTC
Fee per byte	47 sat/B
Fee per weight unit	11.75 sat/WU
Estimated BTC Transacted	0.00033324 BTC
Scripts	Hide scripts & coinbase

Bitcoin from a user's point of view

- Sending a payment to someone
  - transactions
    - are serialized for transmission over the network
    - encoded in hexadecimal format



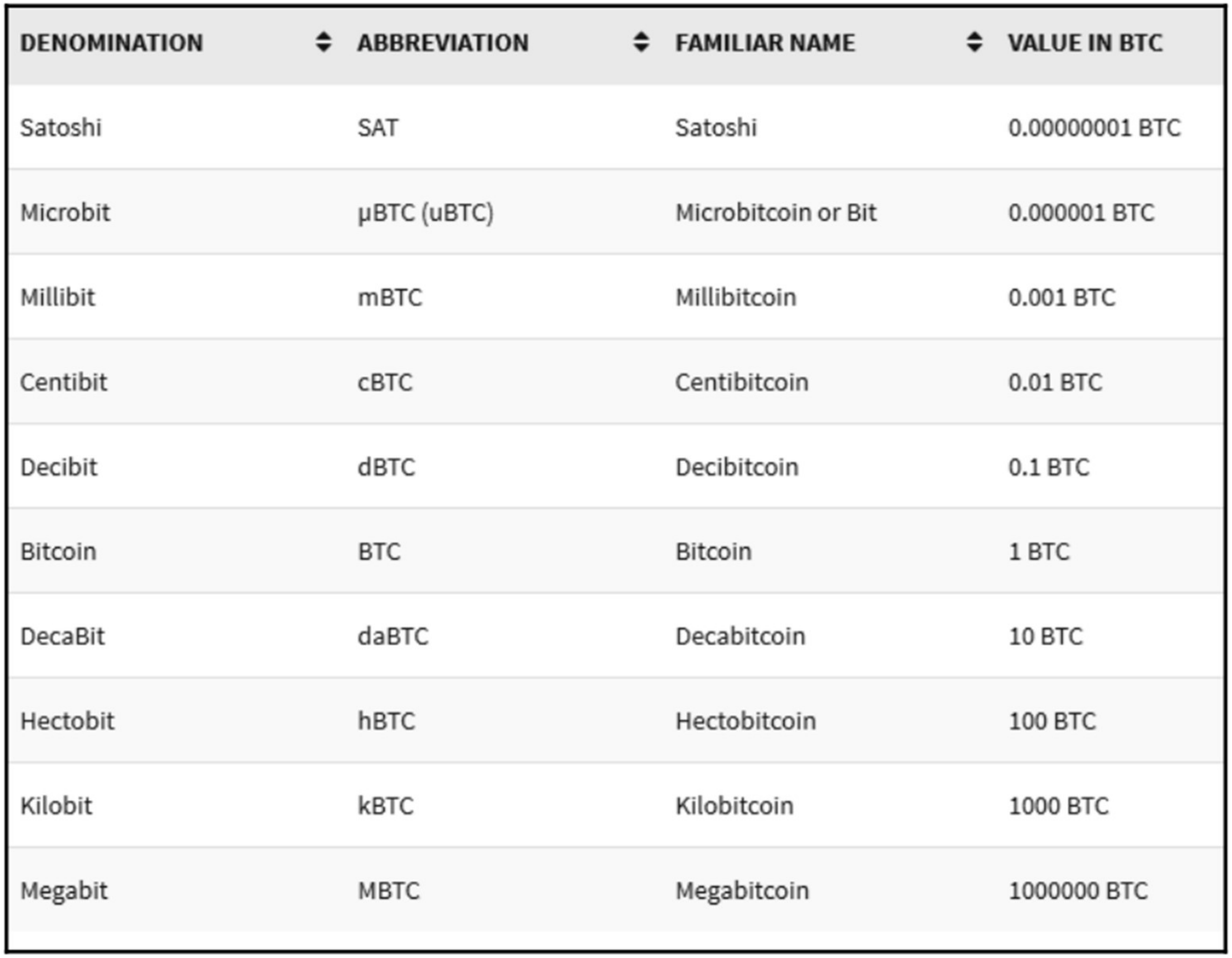
```
01000000017d3876b14a7ac16d8d550abc78345b6571134ff173918a096ef90ff0430e12408
b0000006b483045022100de6fd8120d9f142a82d5da9389e271caa3a757b01757c8e4fa7afb
f92e74257c02202a78d4fbd52ae9f3a0083760d76f84643cf8ab80f5ef971e3f98ccba2c717
58d012102c16942555f5e633645895c9affcb994ea7910097b7734a6c2d25468622f25e12ff
ffffff022c820000000000001976a914c568ffeb46c6a9362e44a5a49deaa6eab05a619a88a
cc06c0100000000000001976a9149386c8c880488e80a6ce8f186f788f3585f74aee88ac000000
00
```

Bitcoin from a user's point of view

- Sending a payment to someone
  - Summary
    1. Transaction starts with a sender signing the transaction with their private key
    2. Transaction is serialized so that it can be transmitted over the network
    3. Transaction is broadcasted to the network
    4. Miners listening for the transactions picks up the transaction
    5. Transaction are verified for their validity by the miners
    6. Transaction are added to the candidate/proposed block for mining
    7. Once mined, the result is broadcasted to all nodes on the Bitcoin network

Bitcoin from a user's point of view

- various denominations of bitcoin



DENOMINATION	ABBREVIATION	FAMILIAR NAME	VALUE IN BTC
Satoshi	SAT	Satoshi	0.00000001 BTC
Microbit	µBTC (uBTC)	Microbitcoin or Bit	0.000001 BTC
Millibit	mBTC	Millibitcoin	0.001 BTC
Centibit	cBTC	Centibitcoin	0.01 BTC
Decibit	dBTC	Decibitcoin	0.1 BTC
Bitcoin	BTC	Bitcoin	1 BTC
DecaBit	daBTC	Decabitcoin	10 BTC
Hectobit	hBTC	Hectobitcoin	100 BTC
Kilobit	kBTC	Kilobitcoin	1000 BTC
Megabit	MBTC	Megabitcoin	1000000 BTC

Bitcoin main components

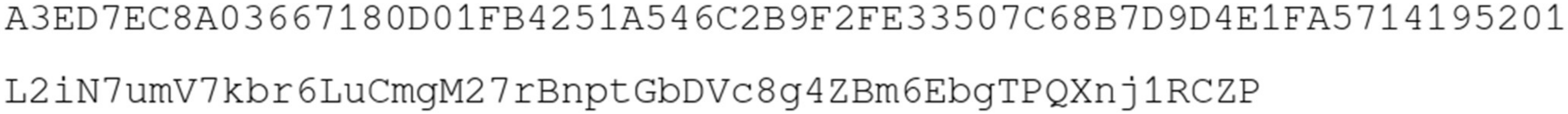
- Digital keys
- Addresses
- Transactions
- Blockchain
- Miners
- The Bitcoin network
- Wallets (client software)

Bitcoin main components

- Digital keys and addresses
  - possession and transfer of bitcoins upon
    - private keys
    - public keys
    - addresses
  - Elliptic Curve Cryptography (ECC) is used to generate key pairs

Bitcoin main components

- Private keys
  - 256-bit numbers
    - randomly chosen in the range
      - from 0x1 to 0xFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFEBAAEDCE6AF48A03BBFD25E8CD0364140
  - are usually encoded using Wallet Import Format (WIF)
  - an example:



```
A3ED7EC8A03667180D01FB4251A546C2B9F2FE33507C68B7D9D4E1FA5714195201
L2iN7umV7kbr6LuCmgM27rBnptGbDVc8g4ZBm6EbgTPQXnj1RCZP
```

Bitcoin main components

- Private keys
  - mini private key format (minikey)
    - private key with a maximum of up to 30 characters allow storage where physical space is limited
      - damage-resistant QR codes
        - more dots can be used for error correction
        - less for encoding the private key
    - can be converted into a normal size private key
      - Not vice versa

Bitcoin main components

- Public keys
  - exist on the blockchain
  - all network participants can see it
  - are derived from private keys
  - are used to verify that the transaction has been signed
  - are 256-bits in length

Bitcoin main components

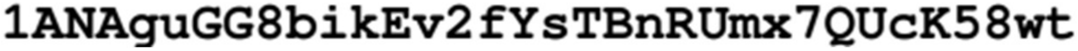
- Public keys
  - are x and y coordinates on an elliptic curve
    - both 32-byte in length
  - can be represented in
    - uncompressed format
      - are presented with a prefix of 0x4
      - 65-bytes long
    - compressed format.
      - 33-bytes long
      - includes only the x part
        - the y part can be derived from it

Bitcoin main components

- Keys are identified by prefixes
    - Uncompressed public keys use 0x04 as the prefix
    - Compressed public key starts with
        - 0x03 if the y is odd
        - 0x02 if the y is even

Bitcoin main components

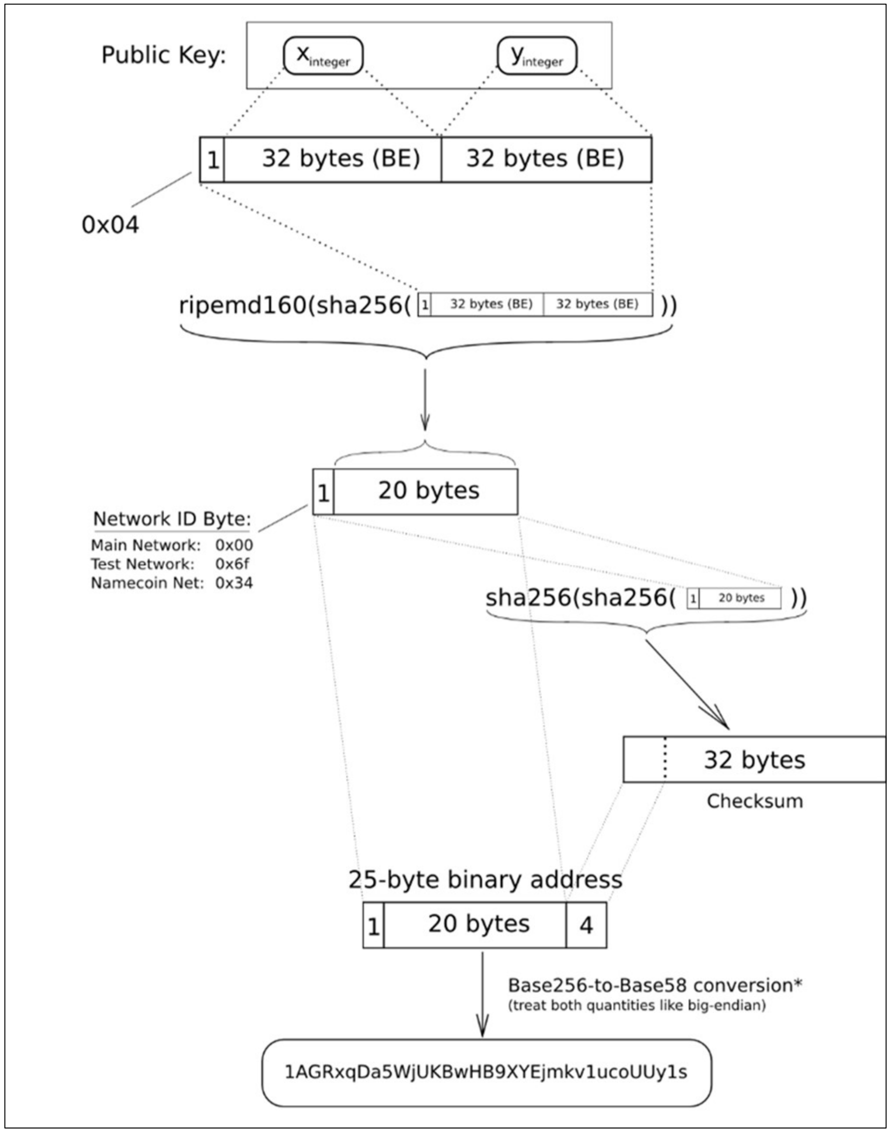
- Addresses
  - created by
    - taking the corresponding public key hashing it twice
      - first with the SHA-256 algorithm
      - then with RIPEMD-160.
    - The 160-bit hash is then
      - prefixed with a version number
      - finally encoded with a Base58Check encoding scheme.
  - are 26-35 characters long and begin with digit 1 or 3



1ANAguGG8bikEv2fYsTBnRUmx7QUcK58wt

Bitcoin main components

- Addresses



Bitcoin main components

- Transactions
  - composed of at least one input and output
    - Inputs: coins being spent
      - have been created in a previous transaction
    - Outputs: coins being created.
  - are not encrypted
  - are publicly visible in the blockchain
  - If a transaction is minting new coins
    - there is no input
    - no signature is needed
  - If a transaction is to send coins to some other address
    - it needs to be signed with private key
    - a reference is also required to the previous transaction
      - in order to show the origin of the coins
  - Coins are unspent transaction outputs represented in Satoshis

Bitcoin main components

- Transaction lifecycle
  1. A sender sends a transaction using wallet software or some other interface.
  2. The wallet software signs the transaction using the sender's private key.
  3. The transaction is broadcasted to the Bitcoin network using a flooding algorithm.
  4. Miners verify and include this transaction in the next block to be mined.
     - Before placing in the block,
       - transactions are placed in transaction pool
         - A memory buffer in local memory of nodes
  5. Mining starts
  6. A miner solves the PoW problem
     - it broadcasts the newly mined block to the network
  7. The nodes verify the block and propagate the block further
  8. confirmations start to appear in the receiver's wallet
     - After approximately three confirmations
       - the transaction is considered finalized
       - The probability of double spending is virtually eliminated

Bitcoin main components

- Transaction fee
  - are charged by the miners
  - are dependent upon the size and weight of the transaction
  - are calculated by

$$fee = sum(inputs) - sum(outputs)$$

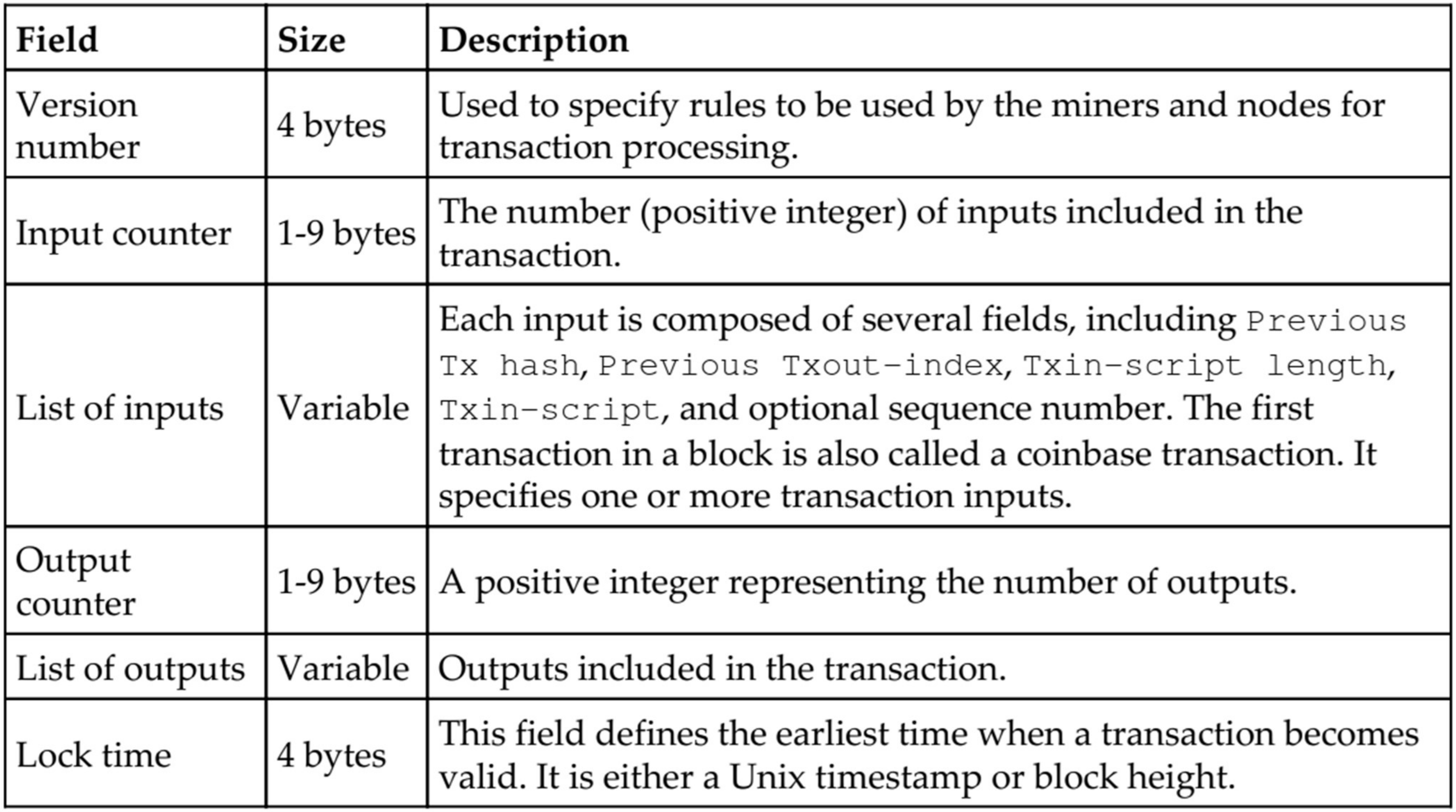
  - are used as an incentive for miners
    - miners pick up transactions based on their priority
      - a transaction with a higher fee will be picked up sooner

Bitcoin main components

- Transaction fee
  - are not fixed by the Bitcoin protocol
    - are not mandatory
      - a transaction with no fee will be processed
        - but may take a very long time.
        - no longer practical
          - due to the high volume of transactions and competing investors
  - time for transaction confirmation usually ranges from 10 minutes to over 12 hours in some cases.
    - dependent on transaction fees and network activity
    - If the network is very busy,
      - transactions will take longer to process
    - if you pay a higher fee
      - your transaction is more likely to be picked by miners first

Bitcoin main components

- The transaction data structure



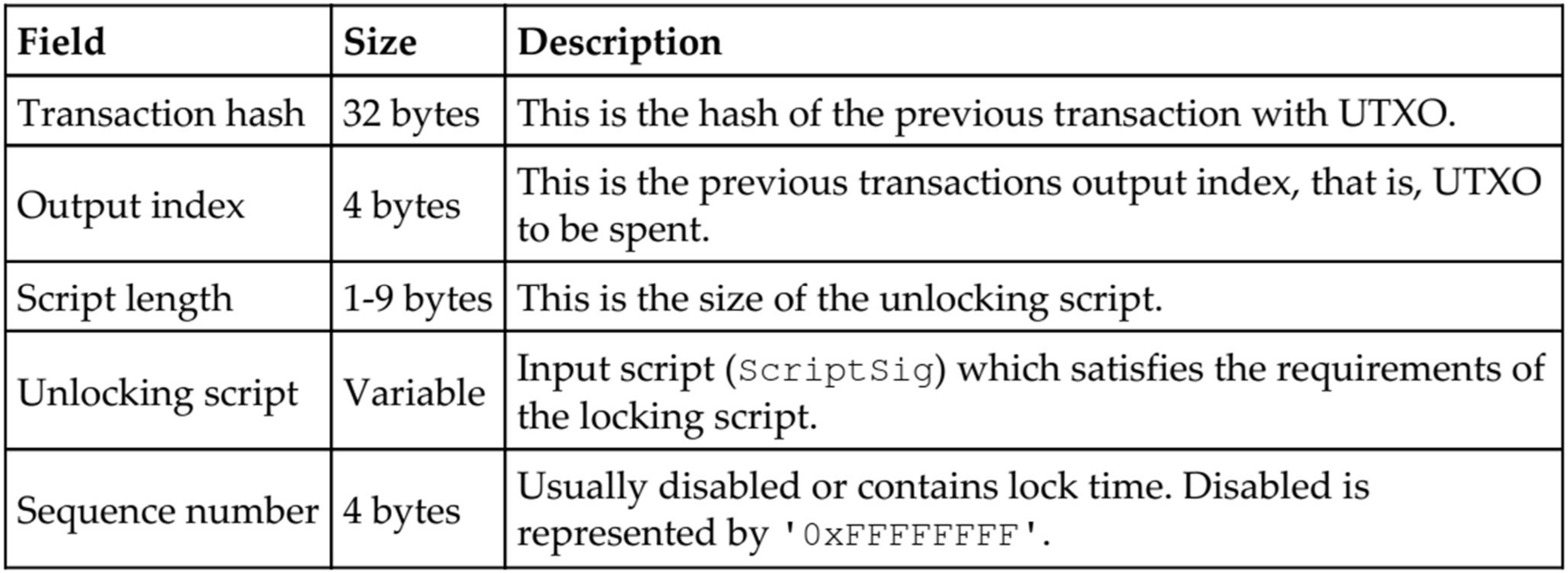
Field	Size	Description
Version number	4 bytes	Used to specify rules to be used by the miners and nodes for transaction processing.
Input counter	1-9 bytes	The number (positive integer) of inputs included in the transaction.
List of inputs	Variable	Each input is composed of several fields, including `Previous Tx hash`, `Previous Txout-index`, `Txin-script length`, `Txin-script`, and optional sequence number. The first transaction in a block is also called a coinbase transaction. It specifies one or more transaction inputs.
Output counter	1-9 bytes	A positive integer representing the number of outputs.
List of outputs	Variable	Outputs included in the transaction.
Lock time	4 bytes	This field defines the earliest time when a transaction becomes valid. It is either a Unix timestamp or block height.

Bitcoin main components

- The transaction Inputs
  - each input spends a previous output
  - Each output is considered as Unspent Transaction Output (UTXO)
    - until an input consumes it

Bitcoin main components

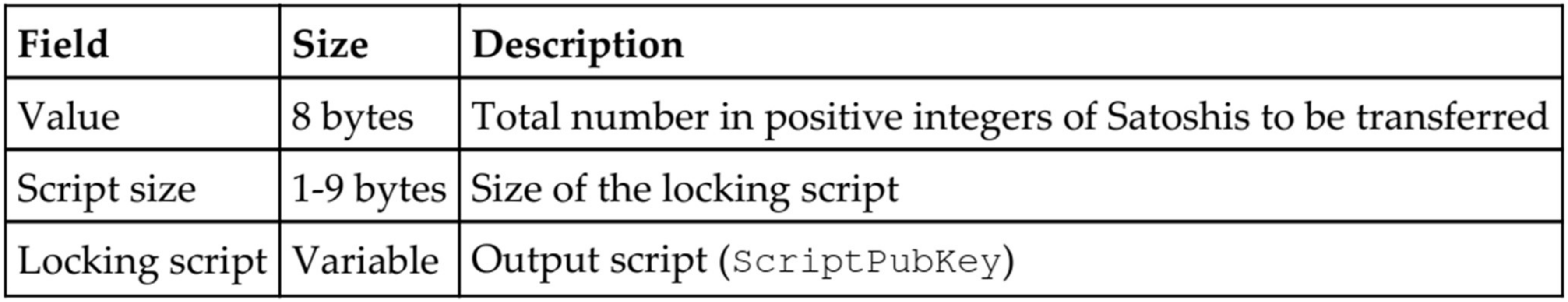
- The transaction Inputs



Field	Size	Description
Transaction hash	32 bytes	This is the hash of the previous transaction with UTXO.
Output index	4 bytes	This is the previous transactions output index, that is, UTXO to be spent.
Script length	1-9 bytes	This is the size of the unlocking script.
Unlocking script	Variable	Input script (`ScriptSig`) which satisfies the requirements of the locking script.
Sequence number	4 bytes	Usually disabled or contains lock time. Disabled is represented by `'0xFFFFFFFF'`.

Bitcoin main components

- The transaction Outputs



Field	Size	Description
Value	8 bytes	Total number in positive integers of Satoshis to be transferred
Script size	1-9 bytes	Size of the locking script
Locking script	Variable	Output script (`ScriptPubKey`)

Bitcoin main components

- Transaction Verification
  - performed using Bitcoin's scripting language

Bitcoin main components

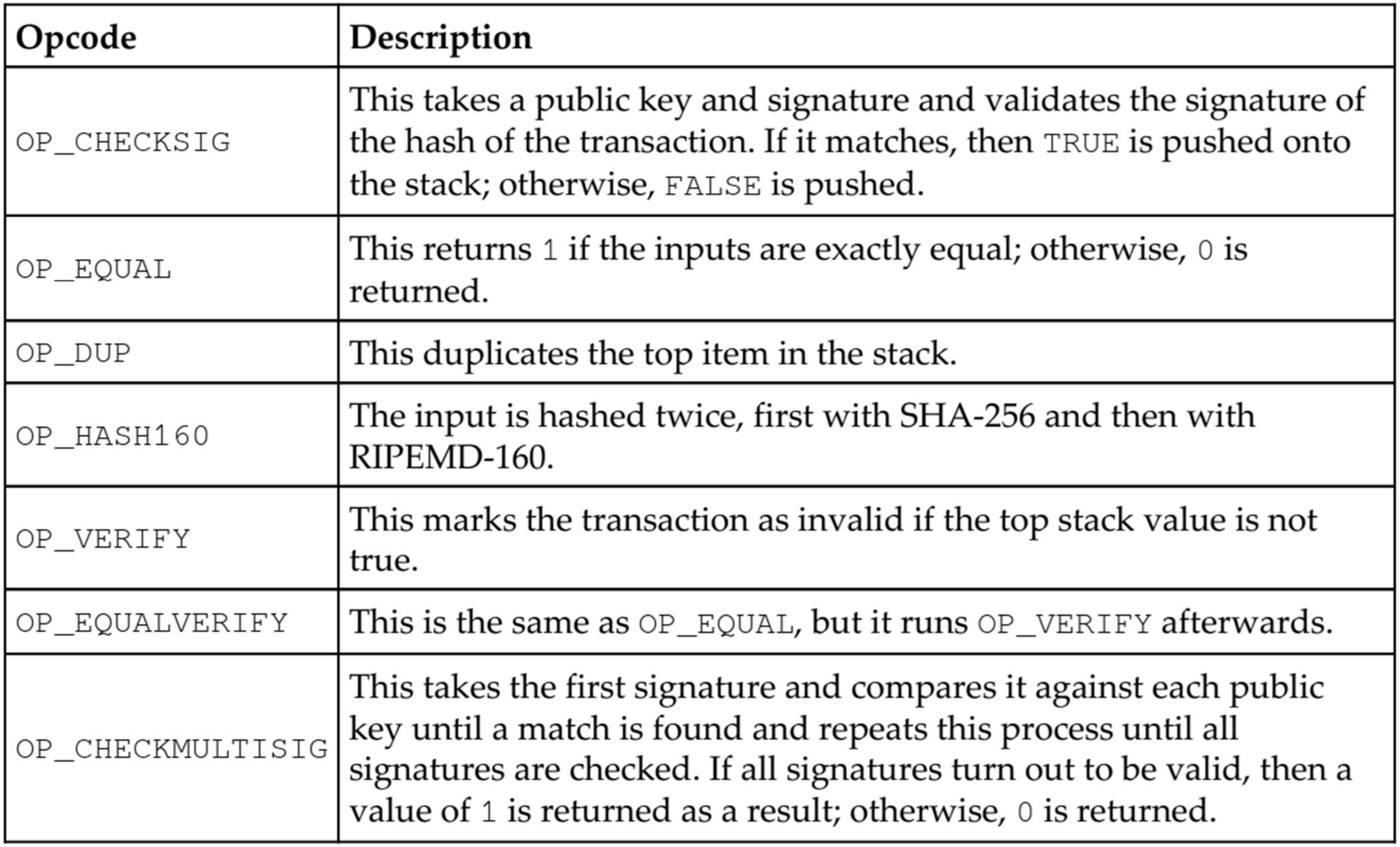
- The script language
  - simple stack-based language called script
  - describes how bitcoins can be spent and transferred.
  - is not Turing complete
  - has no loops
    - to avoid any undesirable effects of long-running/hung
  - is evaluated from the left to the right
  - use various opcodes
    - also known as words, commands, or functions
  - a few opcodes are no longer used
    - due to bugs discovered in their design

Bitcoin main components

- The script language
  - A transaction script is evaluated by combining ScriptSig and ScriptPubKey
    - ScriptSig is the unlocking script
    - ScriptPubKey is the locking script
  - transaction evaluation steps:
    1. it is unlocked
       - ScriptPubkey specifies the spending conditions
       - ScriptSig is provided by the user who wishes to unlock the transaction
         - Must fulfill spending conditions
    2. then it is spent

Bitcoin main components

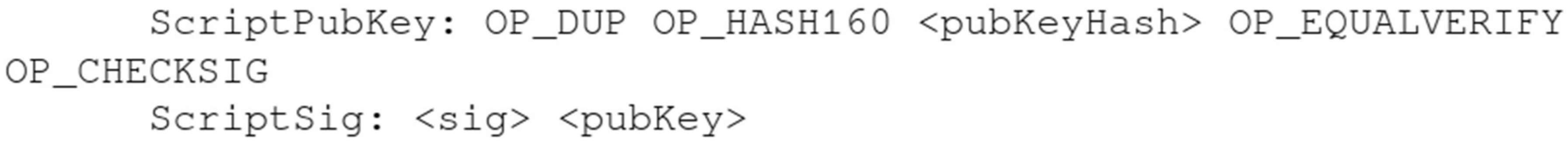
- The script language commonly used opcodes



Opcode	Description
OP_CHECKSIG	This takes a public key and signature and validates the signature of the hash of the transaction. If it matches, then TRUE is pushed onto the stack; otherwise, FALSE is pushed.
OP_EQUAL	This returns 1 if the inputs are exactly equal; otherwise, 0 is returned.
OP_DUP	This duplicates the top item in the stack.
OP_HASH160	The input is hashed twice, first with SHA-256 and then with RIPEMD-160.
OP_VERIFY	This marks the transaction as invalid if the top stack value is not true.
OP_EQUALVERIFY	This is the same as OP_EQUAL, but it runs OP_VERIFY afterwards.
OP_CHECKMULTISIG	This takes the first signature and compares it against each public key until a match is found and repeats this process until all signatures are checked. If all signatures turn out to be valid, then a value of 1 is returned as a result; otherwise, 0 is returned.

Bitcoin main components

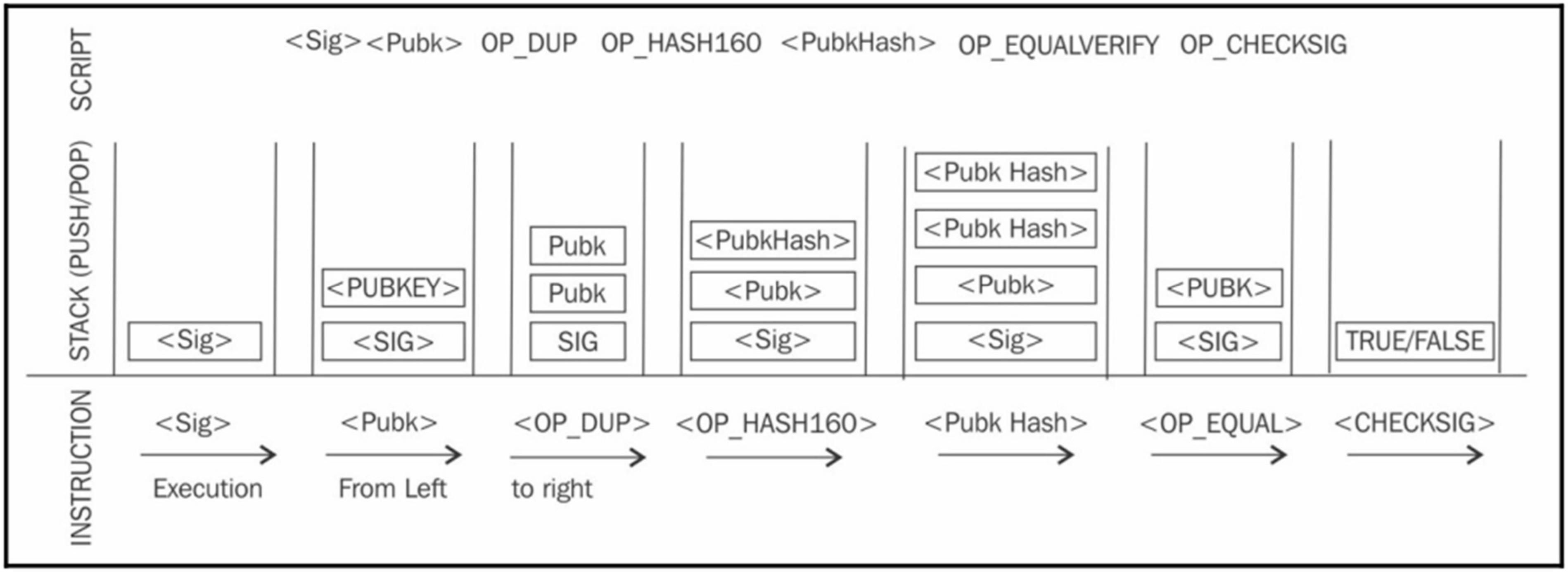
- Types of transactions
  - Pay to Public Key Hash (P2PKH)
    - most commonly used transaction type
    - is used to send transactions to the bitcoin addresses
    - The ScriptPubKey and ScriptSig parameters are concatenated together and executed.
    - Format:



```
      ScriptPubKey: OP_DUP OP_HASH160 <pubKeyHash> OP_EQUALVERIFY
OP_CHECKSIG
      ScriptSig: <sig> <pubKey>
```

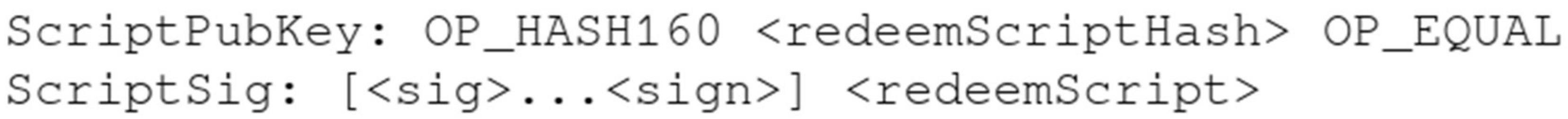
Bitcoin main components

- Types of transactions
  - Pay to Public Key Hash (P2PKH)



Bitcoin main components

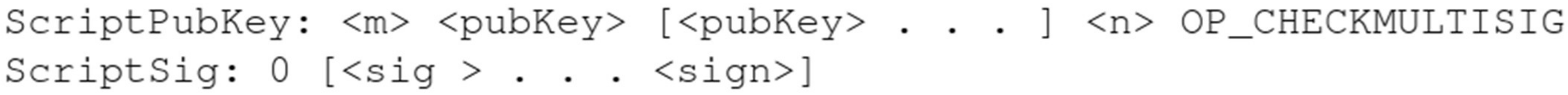
- Types of transactions
  - Pay to Script Hash (P2SH)
    - is used in order to send transactions to a script hash
      - that is, the addresses starting with 3
    - the redeem script is also evaluated and must be valid



```
ScriptPubKey: OP_HASH160 <redeemScriptHash> OP_EQUAL
ScriptSig: [<sig>...<sign>] <redeemScript>
```

Bitcoin main components

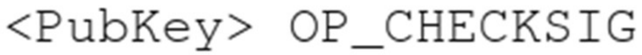
- Types of transactions
  - MultiSig (Pay to MultiSig)
    - M-of-N MultiSig transaction script
      - a complex type of script
      - it is possible to construct a script that required multiple valid signatures
    - Raw multisig is obsolete
      - multisig is usually part of the P2SH redeem script



```
ScriptPubKey: <m> <pubKey> [<pubKey> . . . ] <n> OP_CHECKMULTISIG
ScriptSig: 0 [<sig > . . . <sign>]
```

Bitcoin main components

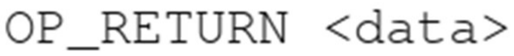
- Types of transactions
  - Pay to Pubkey
    - a very simple script
    - commonly used in coinbase transactions.
    - It is now obsolete
      - was used in an old version of bitcoin.



```
<PubKey> OP_CHECKSIG
```

Bitcoin main components

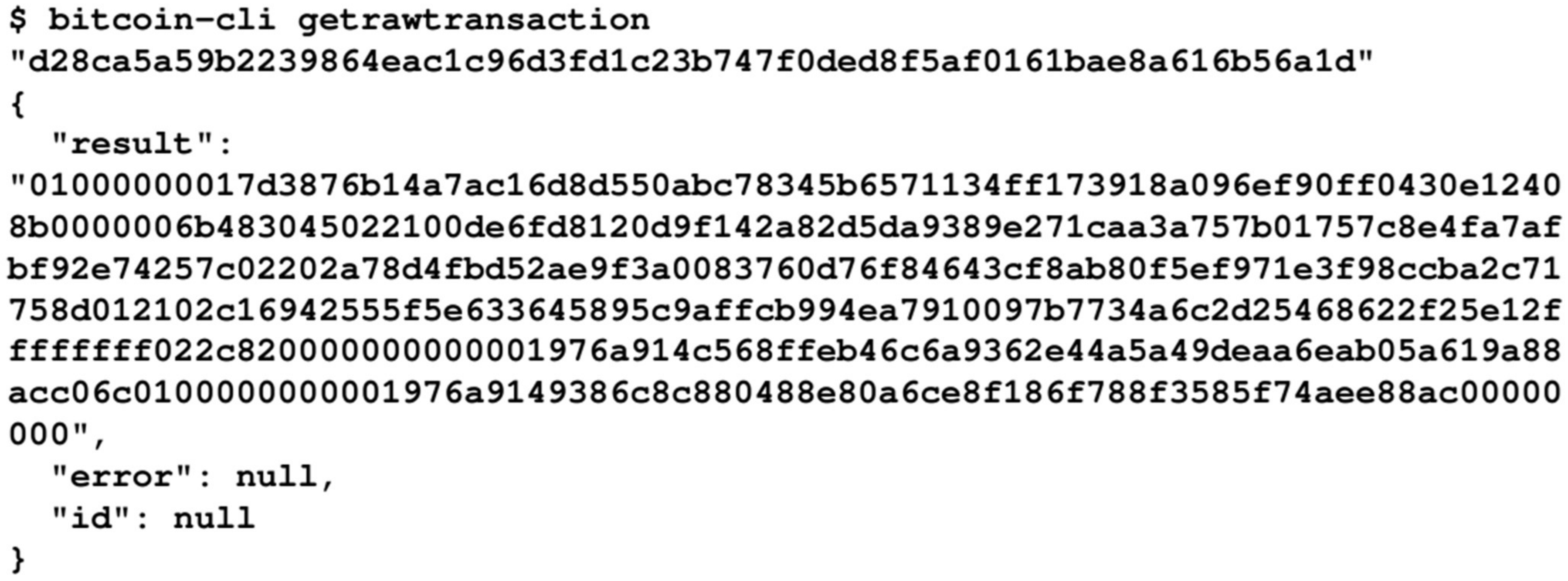
- Types of transactions
  - Null data/OP_RETURN
    - is used to store arbitrary data on the blockchain for a fee.
    - The limit of the message is 40 bytes
    - The output of this script is unredeemable
    - ScriptSig is not required in this case.



```
OP_RETURN <data>
```

Bitcoin main components

- All transactions are eventually encoded into the hexadecimal format
  - before transmitting over the Bitcoin network



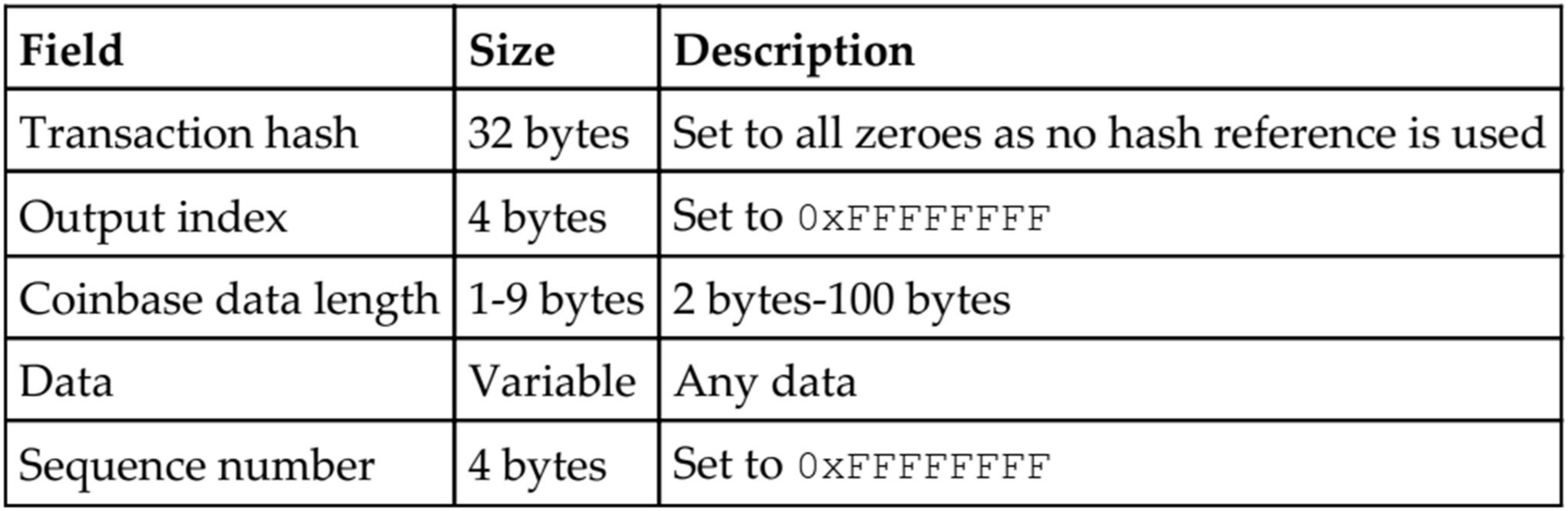
```
$ bitcoin-cli getrawtransaction
"d28ca5a59b2239864eac1c96d3fd1c23b747f0ded8f5af0161bae8a616b56a1d"
{
  "result":
"0100000001 7d3876b14a7ac16d8d550abc78345b6571134ff173918a096ef90ff0430e1240
8b0000006b483045022100de6fd8120d9f142a82d5da9389e271caa3a757b01757c8e4fa7af
bf92e74257c02202a78d4fbd52ae9f3a0083760d76f84643cf8ab80f5ef971e3f98ccba2c71
758d012102c16942555f5e633645895c9affcb994ea7910097b7734a6c2d25468622f25e12f
fffffff022c820000000000001976a914c568ffeb46c6a9362e44a5a49deaa6eab05a619a88
acc06c010000000000001976a9149386c8c880488e80a6ce8f186f788f3585f74aee88ac00000
000",
  "error": null,
  "id": null
}
```

Bitcoin main components

- Coinbase transactions (generation transactions)
  - always created by a miner
  - is the first transaction in a block.
  - is used to create new coins.
  - includes a special field called coinbase
    - acts as an input
  - allows to store up to 100 bytes of arbitrary data
  - has the same number of fields as usual transaction input
    - but the structure contains coinbase data size and coinbase data fields
      - instead of unlocking script size and unlocking script fields.
    - Also, it does not have a reference pointer to the previous transaction.
  - Has special condition
    - prevent them from being spent until at least 100 blocks

Bitcoin main components

- Coinbase transactions (generation transactions)



Field	Size	Description
Transaction hash	32 bytes	Set to all zeroes as no hash reference is used
Output index	4 bytes	Set to `0xFFFFFFFF`
Coinbase data length	1-9 bytes	2 bytes-100 bytes
Data	Variable	Any data
Sequence number	4 bytes	Set to `0xFFFFFFFF`

Bitcoin main components

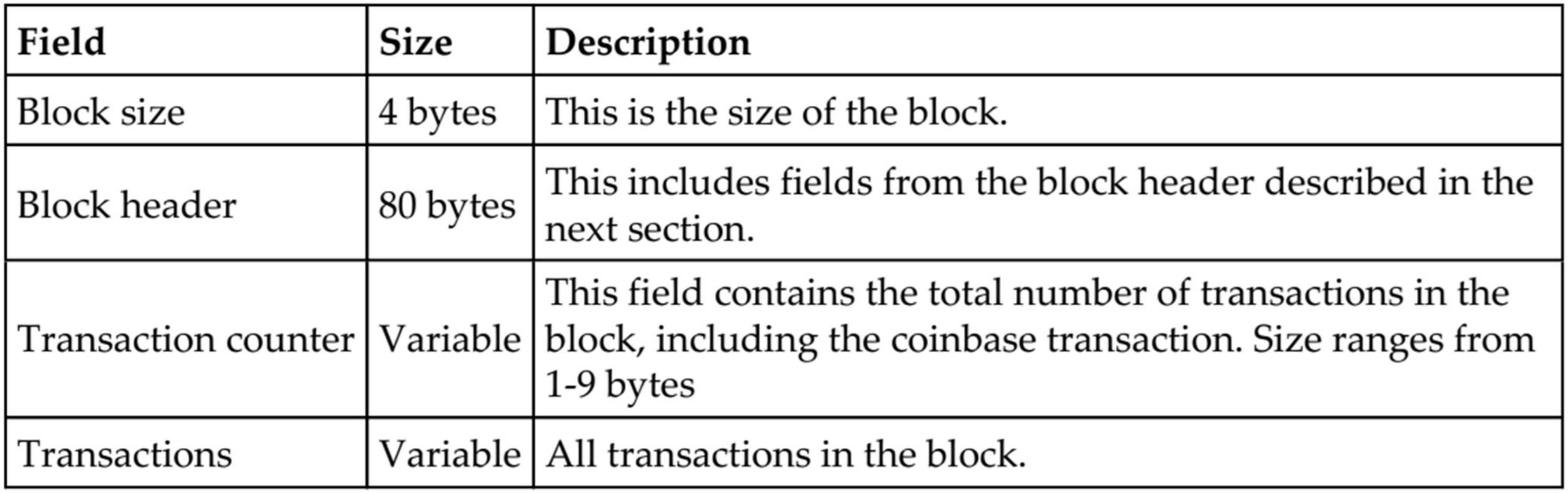
- Contracts
  - transactions that use the Bitcoin system to enforce a financial agreement.
  - can be built using the Bitcoin scripting language
  - E.g.,
    - the release of funds only if multiple parties sign the transaction
      - can be realized using multisig

Bitcoin main components

- Blockchain
  - a public ledger of a timestamped, ordered, and immutable list of all transactions
  - Each block
    - is identified by a hash in the chain
    - Is linked to its previous block by referencing the previous block's hash

Bitcoin main components

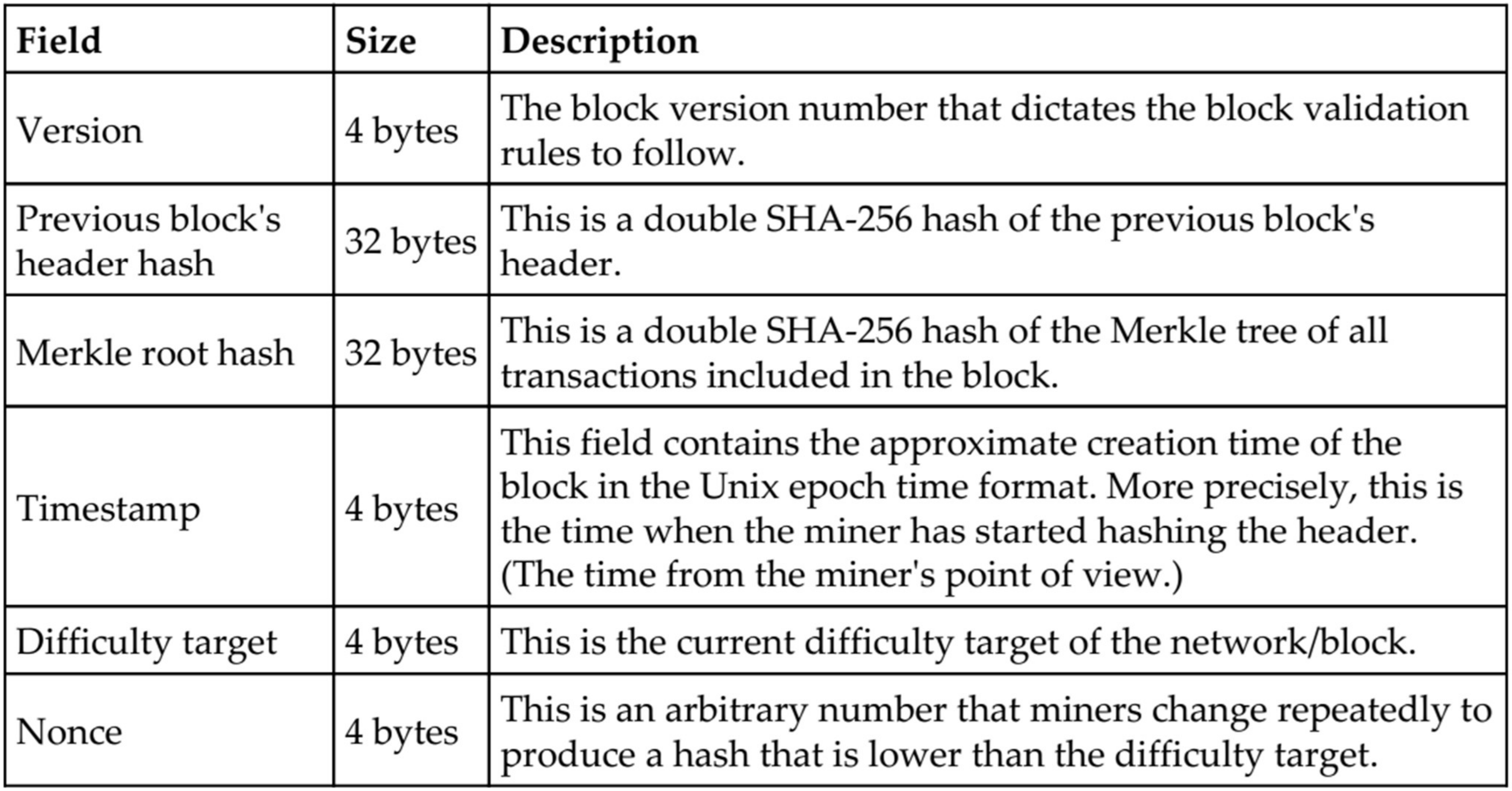
- Blockchain
  - The structure of a block



Field	Size	Description
Block size	4 bytes	This is the size of the block.
Block header	80 bytes	This includes fields from the block header described in the next section.
Transaction counter	Variable	This field contains the total number of transactions in the block, including the coinbase transaction. Size ranges from 1-9 bytes
Transactions	Variable	All transactions in the block.

Bitcoin main components

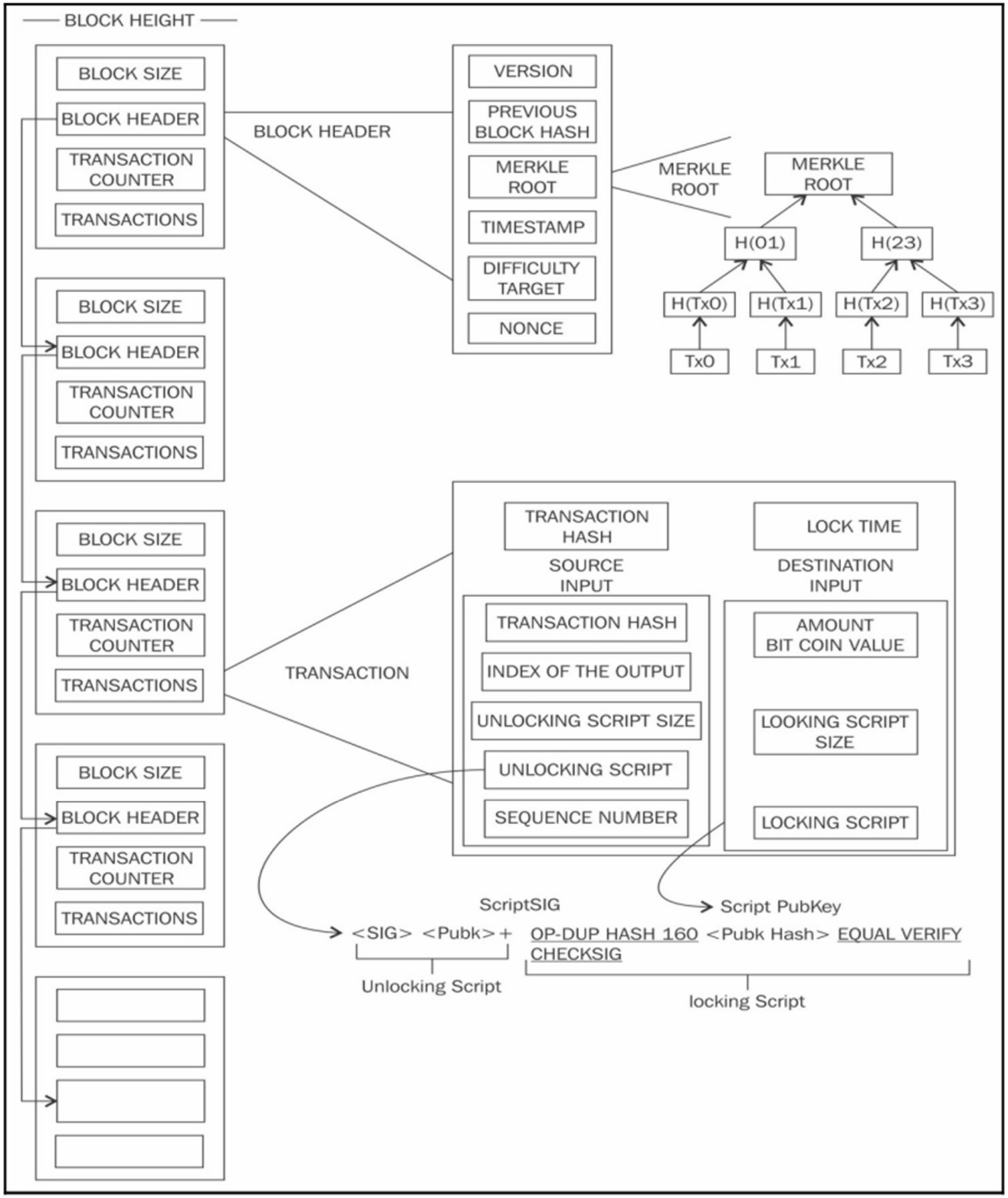
- Blockchain
  - The structure of block header



Field	Size	Description
Version	4 bytes	The block version number that dictates the block validation rules to follow.
Previous block's header hash	32 bytes	This is a double SHA-256 hash of the previous block's header.
Merkle root hash	32 bytes	This is a double SHA-256 hash of the Merkle tree of all transactions included in the block.
Timestamp	4 bytes	This field contains the approximate creation time of the block in the Unix epoch time format. More precisely, this is the time when the miner has started hashing the header. (The time from the miner's point of view.)
Difficulty target	4 bytes	This is the current difficulty target of the network/block.
Nonce	4 bytes	This is an arbitrary number that miners change repeatedly to produce a hash that is lower than the difficulty target.

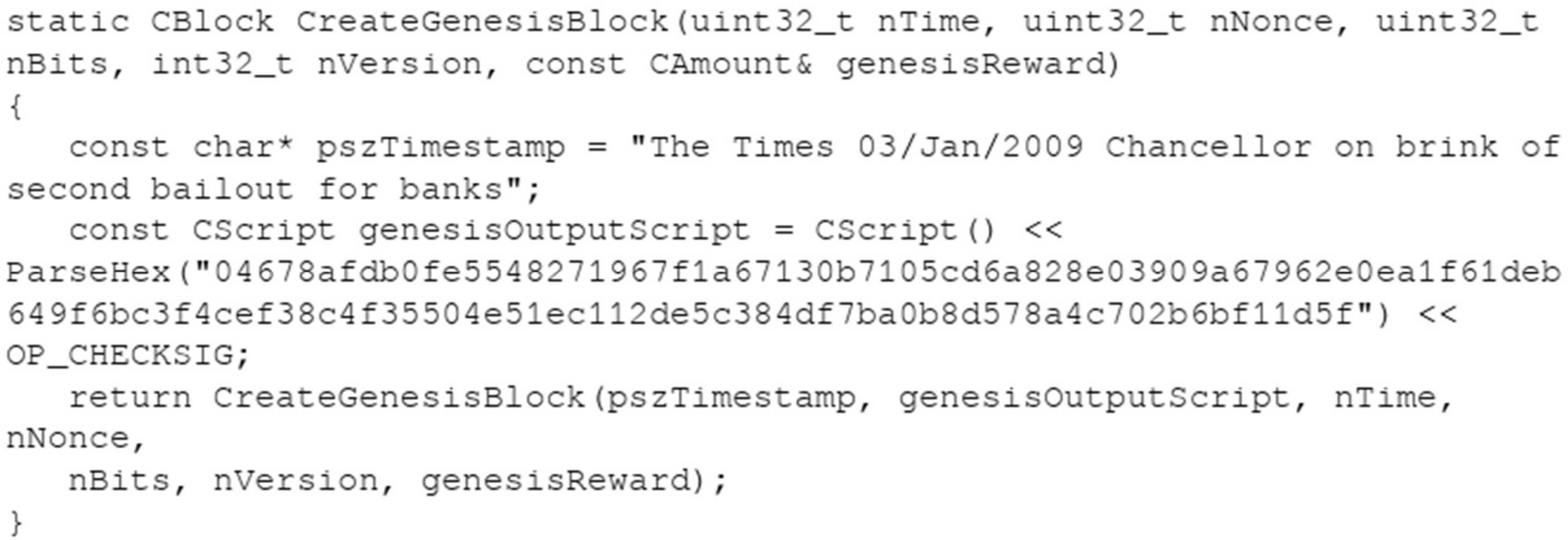
Bitcoin main components

- Blockchain
    - is a chain of blocks
    - each block is linked to its previous block
        - by referencing the previous block header's hash.
    - no transaction can be modified without modifying
        - the block that records it
        - All previous blocks


BLOCK HEIGHT
BLOCK SIZE
BLOCK HEADER
TRANSACTION COUNTER
TRANSACTIONS
BLOCK SIZE
BLOCK HEADER
TRANSACTION COUNTER
TRANSACTIONS
BLOCK SIZE
BLOCK HEADER
TRANSACTION COUNTER
TRANSACTIONS
BLOCK SIZE
BLOCK HEADER
TRANSACTION COUNTER
TRANSACTIONS
BLOCK HEADER
VERSION
PREVIOUS BLOCK HASH
MERKLE ROOT
TIMESTAMP
DIFFICULTY TARGET
NONCE
MERKLE ROOT
MERKLE ROOT
H(01)
H(23)
H(Tx0)
H(Tx1)
H(Tx2)
H(Tx3)
Tx0
Tx1
Tx2
Tx3
TRANSACTION
TRANSACTION HASH
LOCK TIME
SOURCE INPUT
DESTINATION INPUT
TRANSACTION HASH
INDEX OF THE OUTPUT
UNLOCKING SCRIPT SIZE
UNLOCKING SCRIPT
SEQUENCE NUMBER
AMOUNT BIT COIN VALUE
LOOKING SCRIPT SIZE
LOCKING SCRIPT
ScriptSIG
Script PubKey
<SIG> <Pubk>+
OP-DUP HASH 160 <Pubk Hash> EQUAL VERIFY CHECKSIG
Unlocking Script
locking Script

Bitcoin main components

- The genesis block
  - hardcoded in the bitcoin core software



```
static CBlock CreateGenesisBlock(uint32_t nTime, uint32_t nNonce, uint32_t
nBits, int32_t nVersion, const CAmount& genesisReward)
{
   const char* pszTimestamp = "The Times 03/Jan/2009 Chancellor on brink of
second bailout for banks";
   const CScript genesisOutputScript = CScript() <<
ParseHex("04678afdb0fe5548271967f1a67130b7105cd6a828e03909a67962e0ea1f61deb
649f6bc3f4cef38c4f35504e51ec112de5c384df7ba0b8d578a4c702b6bf11d5f") <<
OP_CHECKSIG;
   return CreateGenesisBlock(pszTimestamp, genesisOutputScript, nTime,
nNonce,
   nBits, nVersion, genesisReward);
}
```

Bitcoin main components

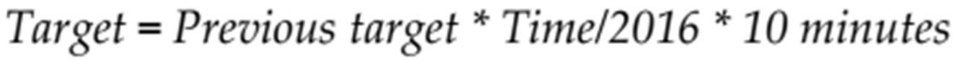
- Network forks
  - a situation where there are two blockchains with different transactions
  - can occur naturally
    - Because of the distributed nature of bitcoin
    - This is an undesirable situation but
      - can be addressed by only accepting the longest chain.
      - the smaller chain will be considered orphaned

Bitcoin main components

- Network forks
  - can also occur with changes in the Bitcoin protocol
    - Soft fork
      - only miners are required to upgrade to the new client software
      - a client not upgrading to the latest version will still be able to work and operate normally
      - New transaction types are sometimes added as a soft fork
    - Hard fork
      - invalidates previously valid blocks
      - requires all users to upgrade
      - any major changes results in a hard fork
        - block structure change
        - major protocol changes

Bitcoin main components

- Network difficulty
  - means how hard it is for miners to find a new block
  - is adjusted dynamically every 2016 blocks
    - to maintain a steady addition of new blocks
    - New blocks are added approximately every 10 minutes
  - is calculated using the following equation



*Target = Previous target * Time/2016 * 10 minutes*

-
  -
    - Time is the time spent to generate previous 2016 blocks

Bitcoin main components

- Mining
    - a process by which new blocks are added to the blockchain.
    - is resource-intensive
        - including computing power and electricity
    - miners compete to solve PoW problem
        - also known as partial hash inversion problem
        - finding a number less than the difficulty target
        - The only way is the brute force method
    - Miners
        - are rewarded with new coins
        - are paid with transaction fees
    - secures the system against frauds and double spending attacks

Bitcoin main components

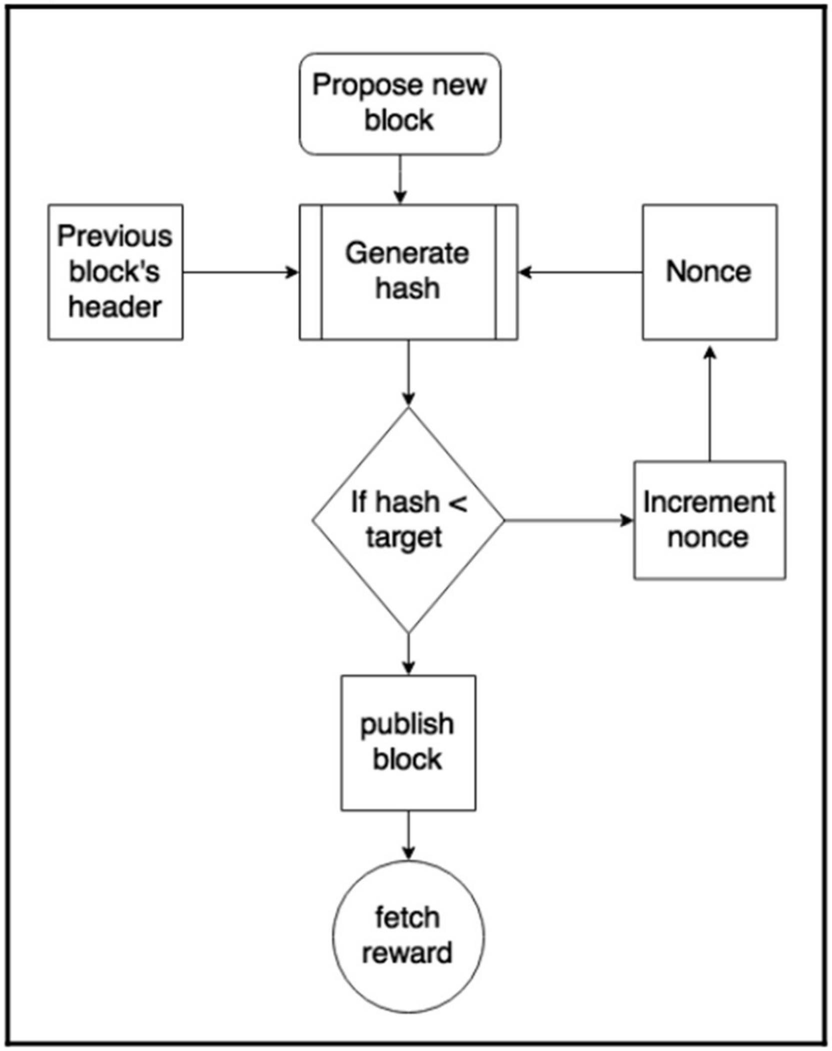
- Mining
  - every 210,000 blocks (roughly every 4 years)
    - The rate of creation of new bitcoins decreases by 50%
  - Block reward
    - was originally 50 bitcoins
    - is currently 6.5 bitcoins
  - Bitcoin supply is limited
    - Only 21 million bitcoins exist
    - in year 2140, the last bitcoin will be finally created
      - no new bitcoins can be created after that
      - miners will be able to profit only by charging transaction fees.

Bitcoin main components

- Tasks of the miners
  1. Synching up with the network
     - a new miner downloads the blockchain from other nodes
  2. Transaction validation
     - verifying and validating signatures and outputs
  3. Block validation
     - evaluating blocks against certain rules including
       - verification of each transaction in the block
       - verification of the nonce value
  4. Perform Proof of Work
     - Creating a new block
       - by combining transactions after validating them
     - Mining the new block
       - solving a computational puzzle
     - Broadcasting the mined block
  5. Fetch reward
     - other nodes verify the minted block
       - There is a slight chance that block will not be accepted
         - due to a clash with another block found at roughly the same time
     - If the minted block is accepted
       - the miner is rewarded with minted bitcoins and any associated transaction fees

Bitcoin main components

- The mining process



Bitcoin main components

- The hash rate
  - the rate of calculating hashes per second
  - the speed at which miners are calculating hashes
  - Currently **238.71 EH/s**
    - 238,710,000,000,000,000,000 hashes per second
- Mining systems
  - CPU
  - GPU
  - FPGA
  - ASICs

Bitcoin main components

- Mining pools
  - group of miners work together to mine a block
  - pool manager
    - receives the coinbase transaction if the block is successfully mined
    - responsible for distributing the reward to the group of miners
  - is profitable compared to solo mining
    - Because the reward is paid to each member regardless of whether they solved the puzzle or not.