Aula 010 - Bitcoin

Uma Visão Geral

Prof. Rogério Aparecido Gonçalves[1]
rogerioag@utfpr.edu.br

[1]Universidade Tecnológica Federal do Paraná (UTFPR)
Departamento de Computação (DACOM)
Campo Mourão - Paraná - Brasil

Programa de Pós Graduação em Ciência da Computação
Mestrado em Ciência da Computação
PPGCC17 - Tópicos em Redes de Computadores e Cibersegurança

Agenda i

Introdução

Objetivos

- Apresentação de uma Visão Geral sobre **Bitcoin**.

Bitcoin

Bitcoin na perspectiva de usuário i

- Passos de como enviar e receber pagamentos:
  - A transação começa com um remetente assinando a transação com sua chave privada.
  - A transação é serializada para que possa ser transmitida pela rede.
  - A transação é transmitida para a rede.
  - Mineradores que escutam transações pegam a transação.
  - A transação é verificada quanto à sua legitimidade pelos mineradores.
  - A transação é adicionada ao bloco candidato/proposto para mineração.
  - Uma vez minerado, o resultado é transmitido para todos os nós da rede *Bitcoin*.
  - Normalmente, neste momento, os usuários aguardam até seis confirmações para serem recebidas antes que uma transação seja considerada final; no entanto, uma transação pode ser considerada final na etapa anterior.

Bitcoin na perspectiva de usuário ii

- As confirmações servem como um mecanismo adicional para garantir que haja probabilidade muito baixa de uma transação ser revertida, mas, caso contrário, uma vez que um bloco minerado seja finalizado e anunciado, as transações dentro desse bloco serão finais nesse ponto.

Chaves Criptográficas i

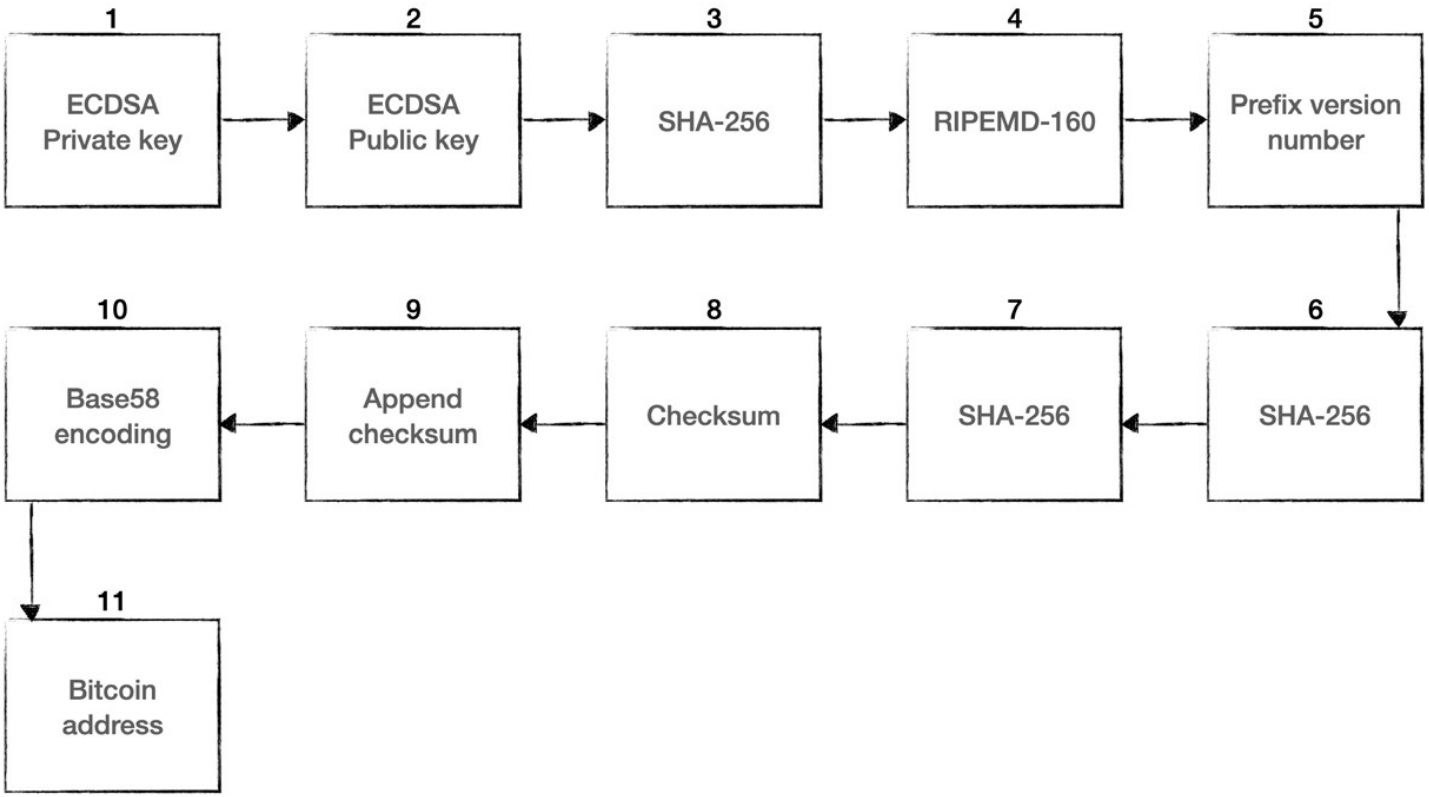
- Private keys in Bitcoin
  - Private keys are used to digitally sign the transactions, proving ownership of the bitcoins.
- Public keys in Bitcoin
  - Public keys are used by nodes to verify that the transaction has indeed been signed with the corresponding private key.
- Addresses in Bitcoin
  - A Bitcoin address is created by taking the corresponding public key of a private key and hashing it twice, first with the SHA256 algorithm and then with RIPEMD160.

QR code of the Bitcoin address
1ANAguGG8bikEv2fYsTBnRUmx7QUcK58wt

From *bitaddress.org*,
a private key and Bitcoin address
in a paper wallet

Geração de Endereços no Bitcoin i

- Para gerar um endereço no **Bitcoin**, é usado um processo de 11 etapas:



Transações i

- A user/sender sends a transaction using wallet software or some other interface.
- The transaction is signed using the sender's private key.
- The transaction is broadcasted to the Bitcoin network using a flooding algorithm.
- Mining nodes (miners) who are listening for the transactions verify and include this transaction in the next lock to be mined.
- Next, the mining starts.
- Finally, the confirmations start to appear in the receiver's wallet.

Estrutura de dados de uma Transação i

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.

Estrutura de dados de uma Transação – entradas e saídas i

Transaction input data structure

Field	Size	Description
Transaction hash	32 bytes	The hash of the previous transaction with UTXO
Output index	4 bytes	This is the previous transaction's output index, such as UTXO to be spent
Script length	1-9 bytes	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`

Transaction output data structure

Field	Size	Description
Value	8 bytes	The 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`)

Script i

- Simple stack-based language used to describe how bitcoins can be spent and transferred
- Evaluated from left to right using a Last in, First Out (LIFO) stack
- Composed of two components: elements and operations.
- Scripts use various operations (opcodes) to define their operations.

Opcodes i

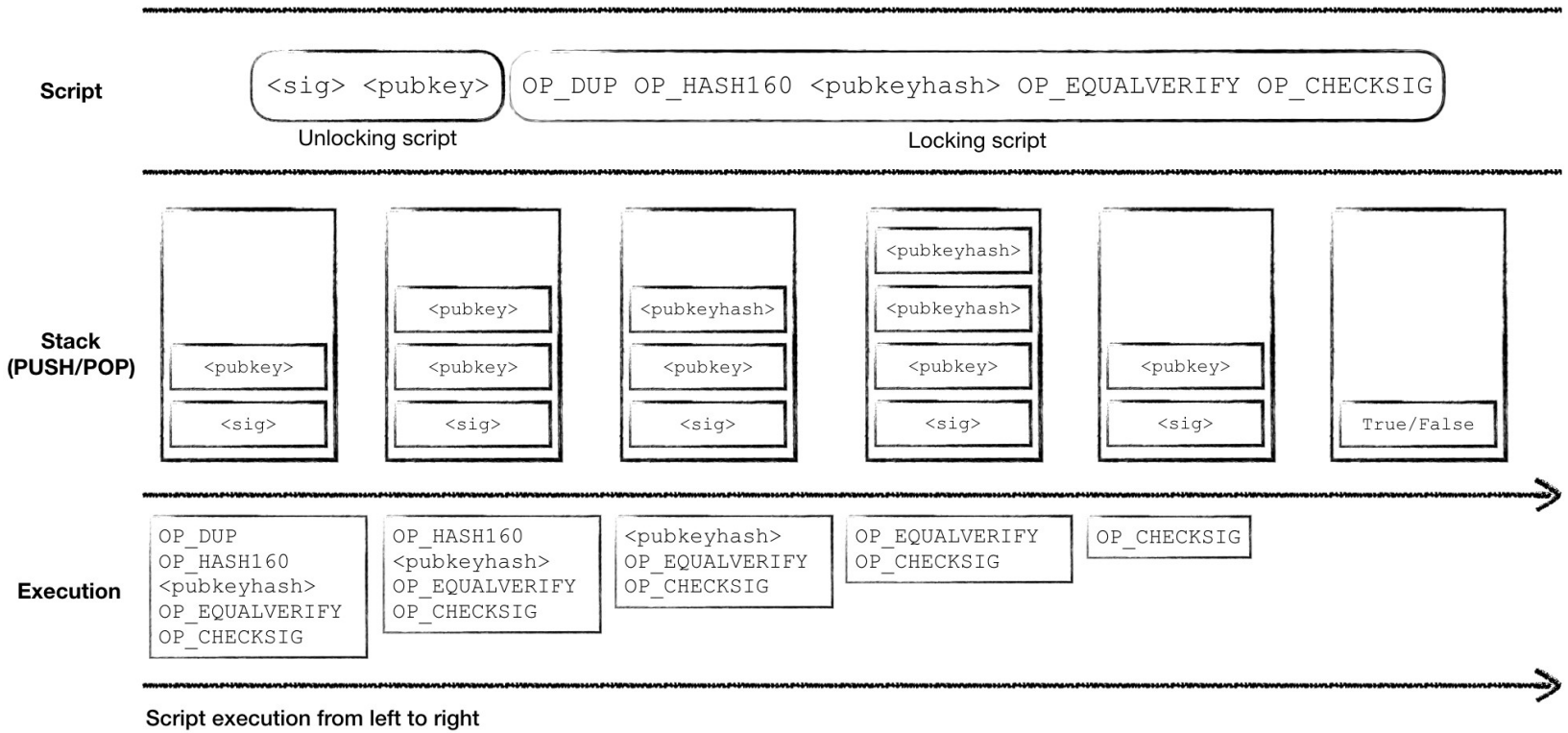
- Here are some examples of a few useful opcodes used in the Script language on the Bitcoin blockchain.

Opcodes ii

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` afterward.
`OP_CHECKMULTISIG`	This instruction 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.
`OP_HASH256`	The input is hashed twice with SHA-256.
`OP_MAX`	This returns the larger value of two inputs.

P2PKH script execution i

- P2PKH is the most commonly used transaction type and is used to send transactions to Bitcoin addresses.



Validação de Transações i

During validation, the following are checked:

- That transaction inputs are previously unspent. This validation step prevents double-spending by verifying that the transaction inputs have not already been spent by someone else.
- That the sum of the transaction outputs is not more than the total sum of the transaction inputs. However, both input and output sums can be the same, or the sum of the input (total value) could be more than the total value of the outputs. This check ensures that no new bitcoins are created out of thin air.
- That that the digital signatures are valid, which ensures that the script is valid.

Blocos i

- A estrutura de um Bloco Bitcoin é mostrado na tabela:

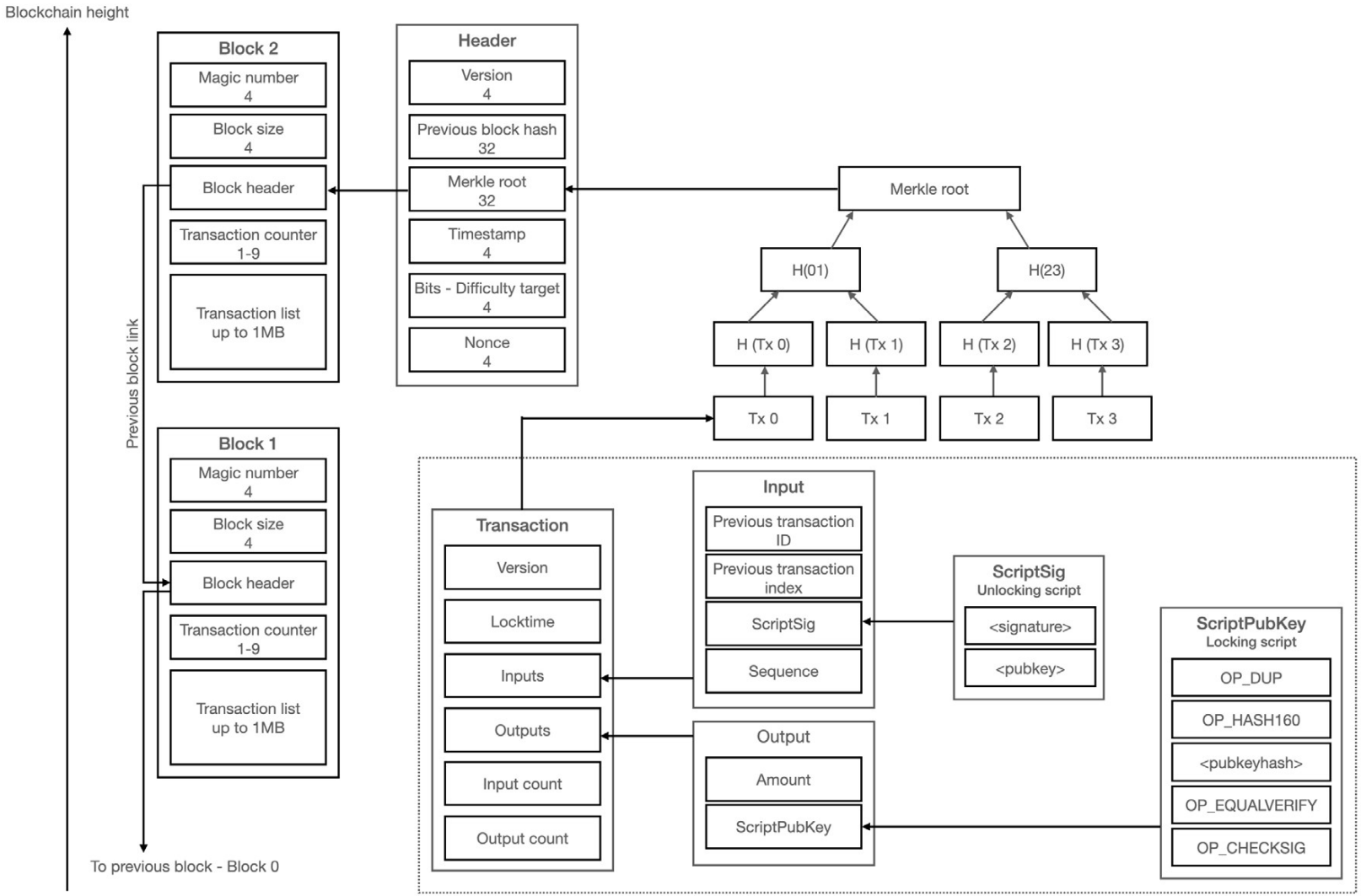
Field	Size	Description
Block size	4 bytes	The size of the block.
Block header	80 bytes	This includes fields from the block header described in the next section.
Transaction counter	Variable	The field contains the total number of transactions in the block, including the coinbase transaction. Size ranges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

Blocos ii

- A estrutura do cabeçalho de um bloco:

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 SHA256 hash of the previous block's header.
Merkle root hash	32 bytes	This is a double SHA256 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 Unix-epoch time format. More precisely, this is the time when the miner started hashing the header (the time from the miner's location).
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.

Uma Visualização da Blockchain do Bitcoin i



Bloco Genesis i

Bloco Genesis

O Bloco Genesis ou bloco $\#0$ foi *hardcoded* (codificado) por suas características especiais: ele é o único que não aponta para nenhum bloco anterior. No seu *hash* foi encriptado o bloco junto com a mensagem *"The Times 03/Jan/2009 Chancellor on brink of second bailout for banks"*, manchete do jornal naquele dia. Além de servir como prova datada, a manchete escolhida representa justamente uma crítica ao sistema bancário.

Bloco Genesis ii

THE TIMES

Max 5C, min -5C | Saturday January 3 2009 timesonline.co.uk No 69523 | £1.50

Eat Out from £5

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

Start collecting tokens today Pullout Inside

Israel prepares to send tanks and troops into Gaza

Israel allowed foreigners to flee the Gaza Strip as it prepared for a ground offensive. At least 430 Palestinians were killed in a week of airstrikes News, page 3

Chancellor on brink of second bailout for banks

Billions may be needed as lending squeeze tightens

Francis Elliott Deputy Political Editor
Gary Duncan Economics Editor

Alistair Darling has been forced to consider a second bailout for banks as the lending drought worsens.

The Chancellor will decide within weeks whether to pump billions more into the economy as evidence mounts that the £37billion 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.

The Bank of England revealed yesterday that, despite intense pressure, the banks curbed lending in the final quarter of last year and plan even tighter restrictions in the coming months. Its findings will alarm the Treasury.

The Bank is expected to take yet more aggressive action this week by cutting the base rate from its current level of 2 per cent. Doing so would reduce the cost of borrowing but have little effect on the availability of loans.

Whitehall sources said that ministers planned to "keep the banks on the boil" but accepted that they need more help to restore lending levels. Formally, the Treasury plans to focus on state-backed gurantees to encourage private finance, but a number of interventions are on the table, including further injections of taxpayers' cash.

Under one option, a "bad bank" would be created to dispose of bad debts. The Treasury would take bad loans off the hands of troubled banks, perhaps swapping them for government bonds. The toxic assets, blamed for poisoning the financial system, would be parked in a state vehicle or "bad bank" that would manage them and attempt to dispose of them while "detoxifying" the mainstream banking system.

The idea would mirror the initial proposal by Henry Paulson, the US Treasury Secretary, to underpin the American banking system by buying

Continued on page 6, col 1
Leading article, page 2

99p

Pub chain cuts the price of a pint from £1.69 to 1989 levels

Business, page 47

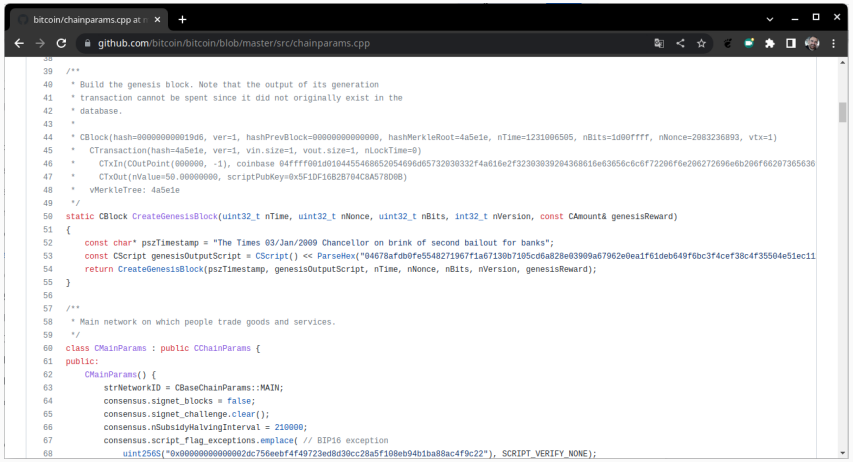
Michael Sheen Frost, Nixon and me
Magazine

Working mums So that's how she does it
Body&Soul

Detox in style The best spas on the planet
Travel

Salmon Rushdie I Won't Marry Again
Pages 22, 23

Giant Killing? Guide to the FA Cup Third Round
Sport



bitcoin/chainparams.cpp at m

github.com/bitcoin/bitcoin/blob/master/src/chainparams.cpp

```
/**
 * Build the genesis block. Note that the output of its generation
 * transaction cannot be spent since it did not originally exist in the
 * database.
 *
 * CBlock(hash=000000000019d6, ver=1, hashPrevBlock=0000000000000, hashMerkleRoot=4a5e1e, nTime=1231006505, nBits=1d00ffff, nNonce=2083236893, vtx=1)
 *   CTransaction(hash=4a5e1e, ver=1, vin.size=1, vout.size=1, nLockTime=0)
 *     CTxIn(COutPoint(000000, -1), coinbase 04ffff001d0104554686520546966d65732030332f4a616e2f32303039204368616e63656c6c6f72206f6e206272696e6b206f662073656636
 *     CTxOut(nValue=50.00000000, scriptPubKey=0x5F1DF16B2B704C8A578D0B)
 *   vMerkleTree: 4a5e1e
 */
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("04678afdb0fe5548271967f1a67130b7105cd6a828e03909a67962e0ea1f61deb649f6bc3f4cef38c4f35504e51ec11
    return CreateGenesisBlock(pszTimestamp, genesisOutputScript, nTime, nNonce, nBits, nVersion, genesisReward);
}

/**
 * Main network on which people trade goods and services.
 */
class CMainParams : public CChainParams {
public:
    CMainParams() {
        strNetworkID = CBaseChainParams::MAIN;
        consensus.signet_blocks = false;
        consensus.signet_challenge.clear();
        consensus.nSubsidyHalvingInterval = 210000;
        consensus.script_flag_exceptions.emplace( // BIP16 exception
            uint256S("0x00000000000002dc756eebf4f49723ed8d30cc28a5f108eb94b1ba88ac4f9c22"), SCRIPT_VERIFY_NONE);
```

Fonte: https://github.com/bitcoin/bitcoin/blob/master/src/chainparams.cpp

Bloco Genesis iii

```
/**
 * Build the genesis block. Note that the output of its generation
 * transaction cannot be spent since it did not originally exist in the
 * database.
 *
 * CBlock(hash=000000000019d6, ver=1, hashPrevBlock=00000000000000, hashMerkleRoot=4a5e1e, nTime=1231006505, nB
 *   CTransaction(hash=4a5e1e, ver=1, vin.size=1, vout.size=1, nLockTime=0)
 *     CTxIn(COutPoint(000000, -1), coinbase 04ffff001d0104455468652054696d65732030332f4a616e2f3230303920436861
 *     CTxOut(nValue=50.00000000, scriptPubKey=0x5F1DF16B2B704C8A578D0B)
 *   vMerkleTree: 4a5e1e
 */
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("04678afdb0fe5548271967f1a67130b7105cd6a828e03909a6
  return CreateGenesisBlock(pszTimestamp, genesisOutputScript, nTime, nNonce, nBits, nVersion, genesisReward);
}
```

A carteira de Satoshi i

- Carteira: `1A1zP1eP5QGefi2DMPTfTL5SLmv7DivfNa`

A carteira de Satoshi ii

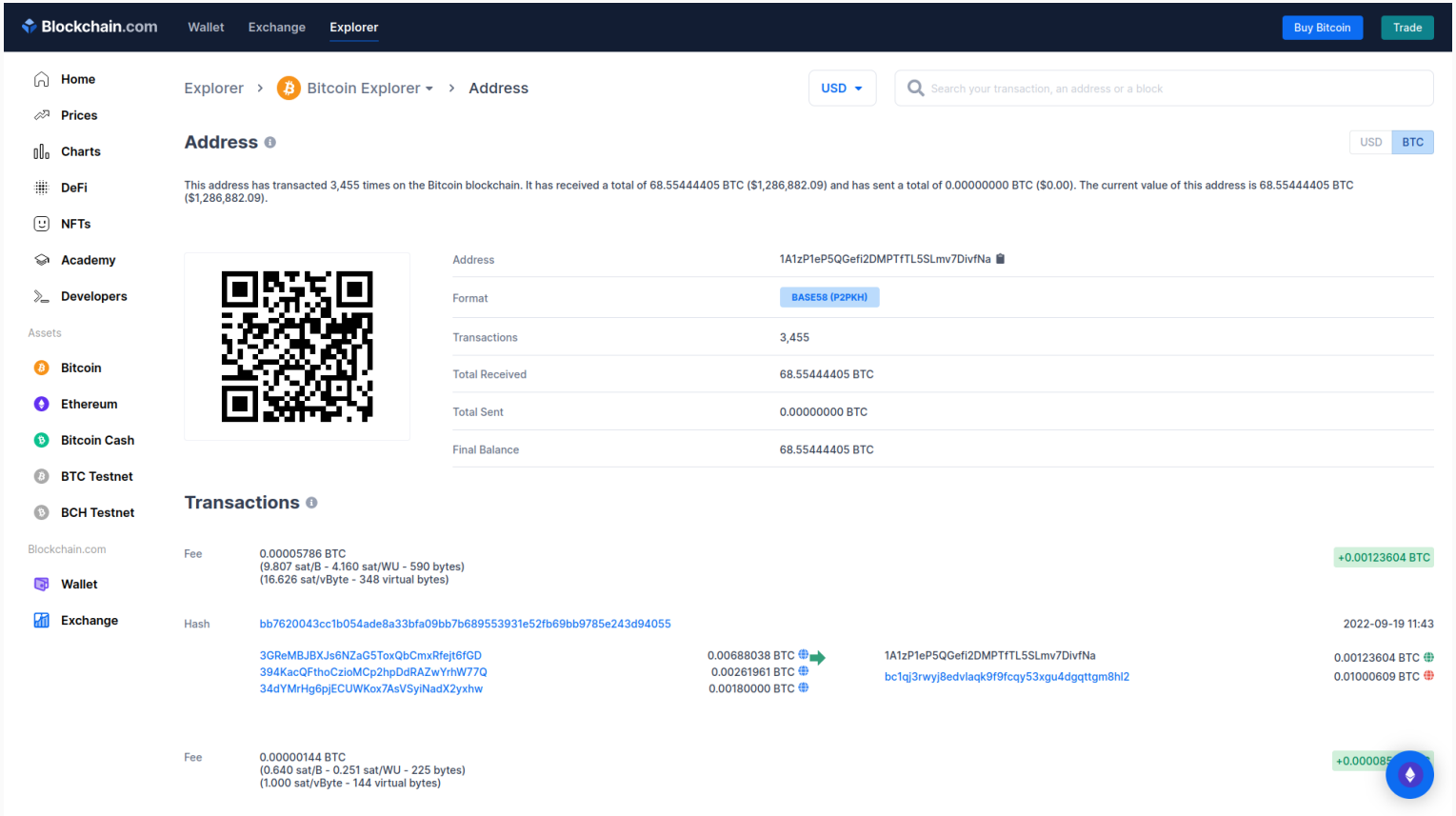


Figura 1: 1A1zP1eP5QGefi2DMPTfTL5SLmv7DivfNa

A carteira de Satoshi iii

- Essa primeira transação foi incluída no **bloco #0**, sob o *hash* 4a5e1e4baab89f3a32518a88c31bc87f618f76673e2cc77ab2127b7afdeda33b.

A carteira de Satoshi iv

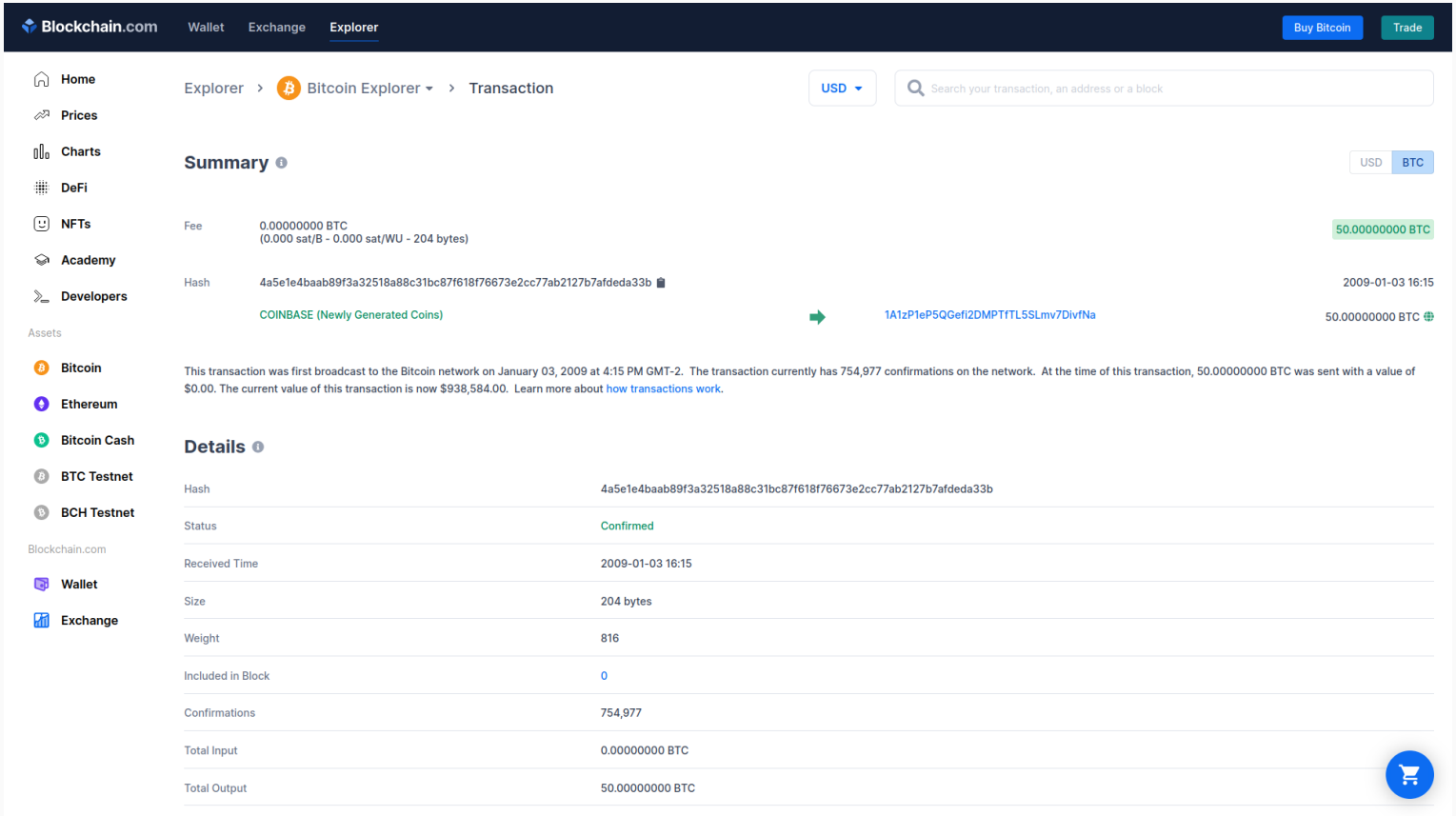


Figura 2: 4a5e1e4baab89f3a32518a88c31bc87f618f76673e2cc77ab2127b7afdeda33b

A carteira de Satoshi v

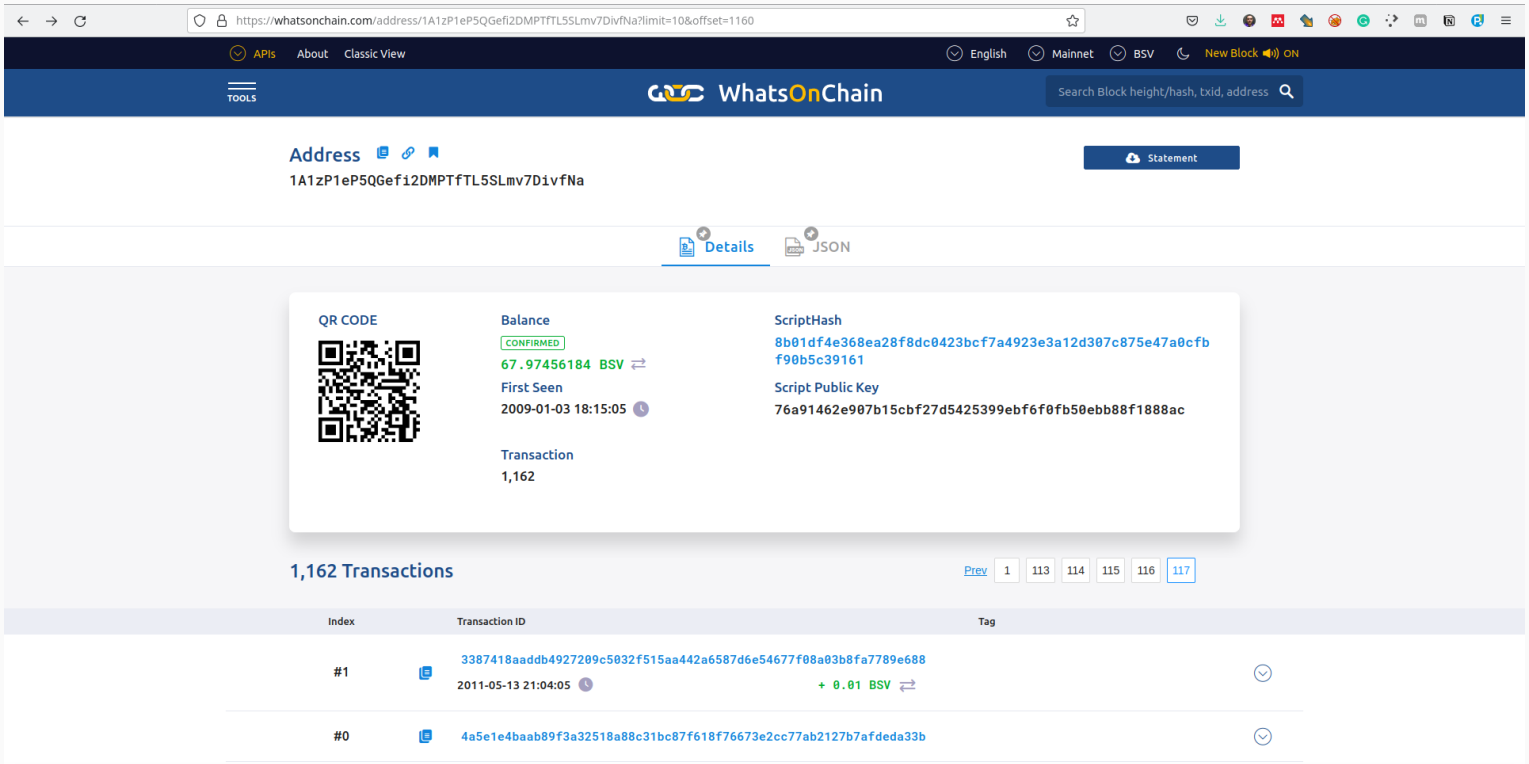


Figura 3: 1A1zP1eP5QGefi2DMPTfTL5SLmv7DivfNa

A carteira de Satoshi vi

- Detalhes da Transação:

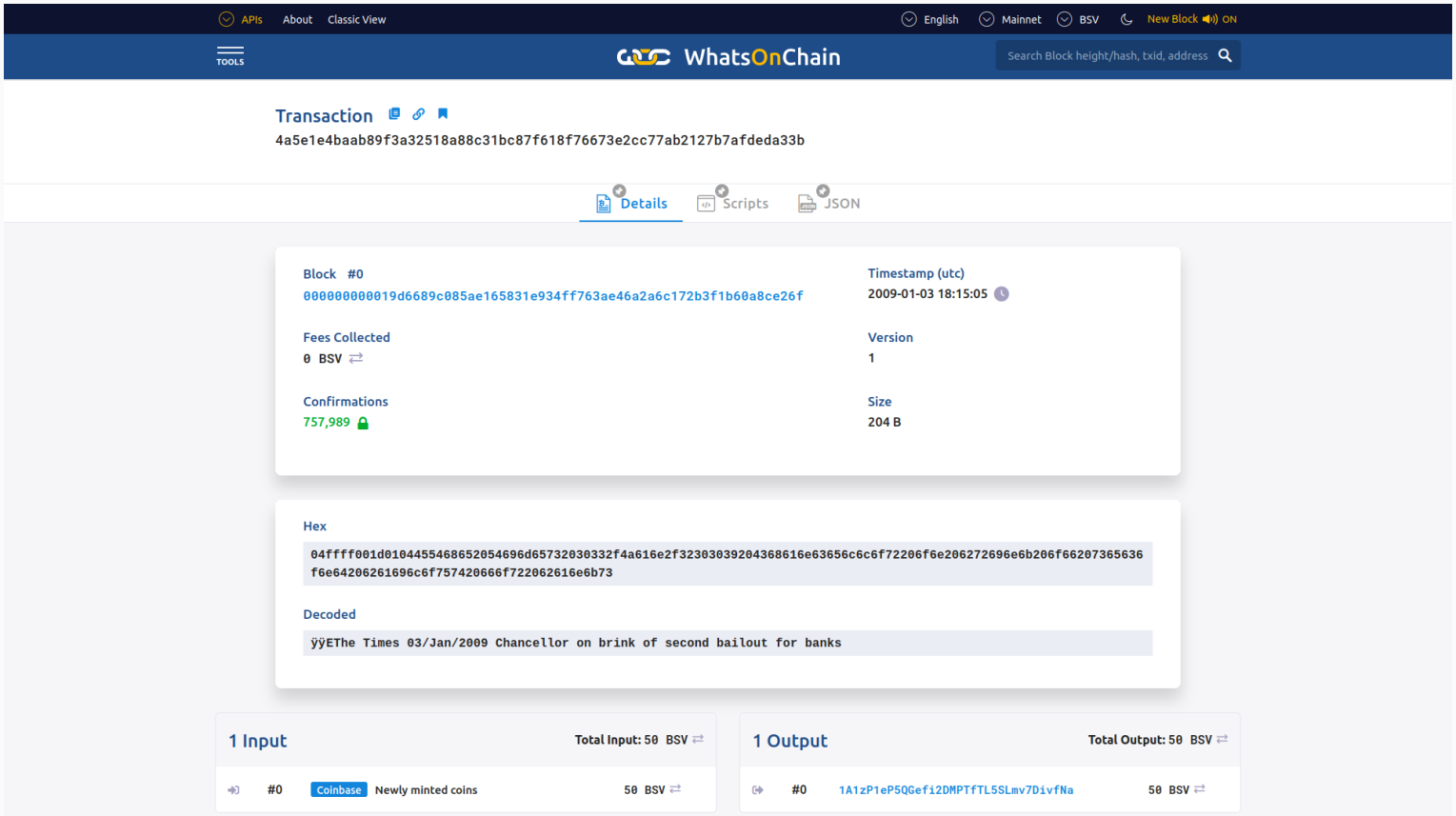
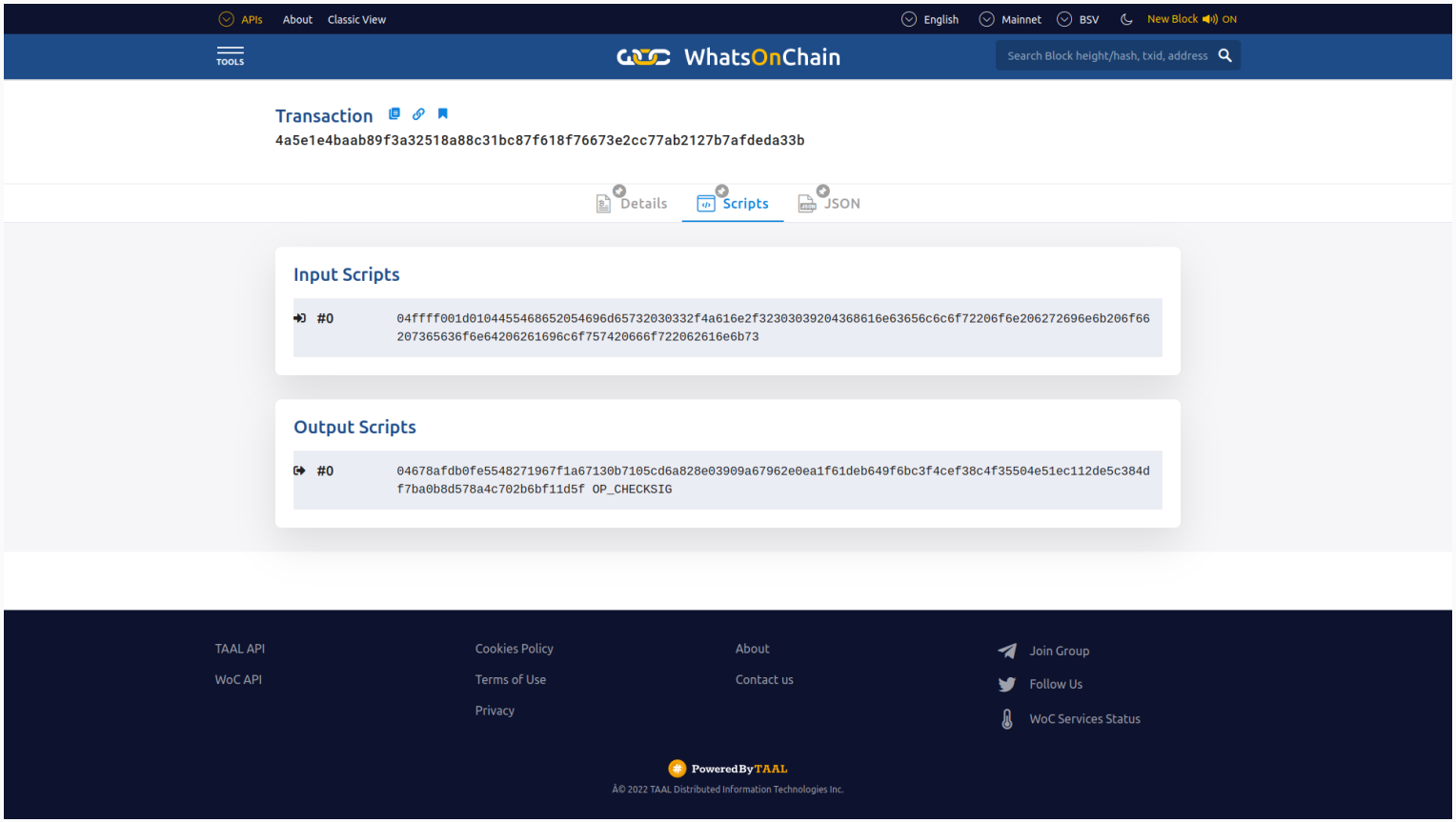


Figura 4: 4a5e1e4baab89f3a32518a88c31bc87f618f76673e2cc77ab2127b7afdeda33b

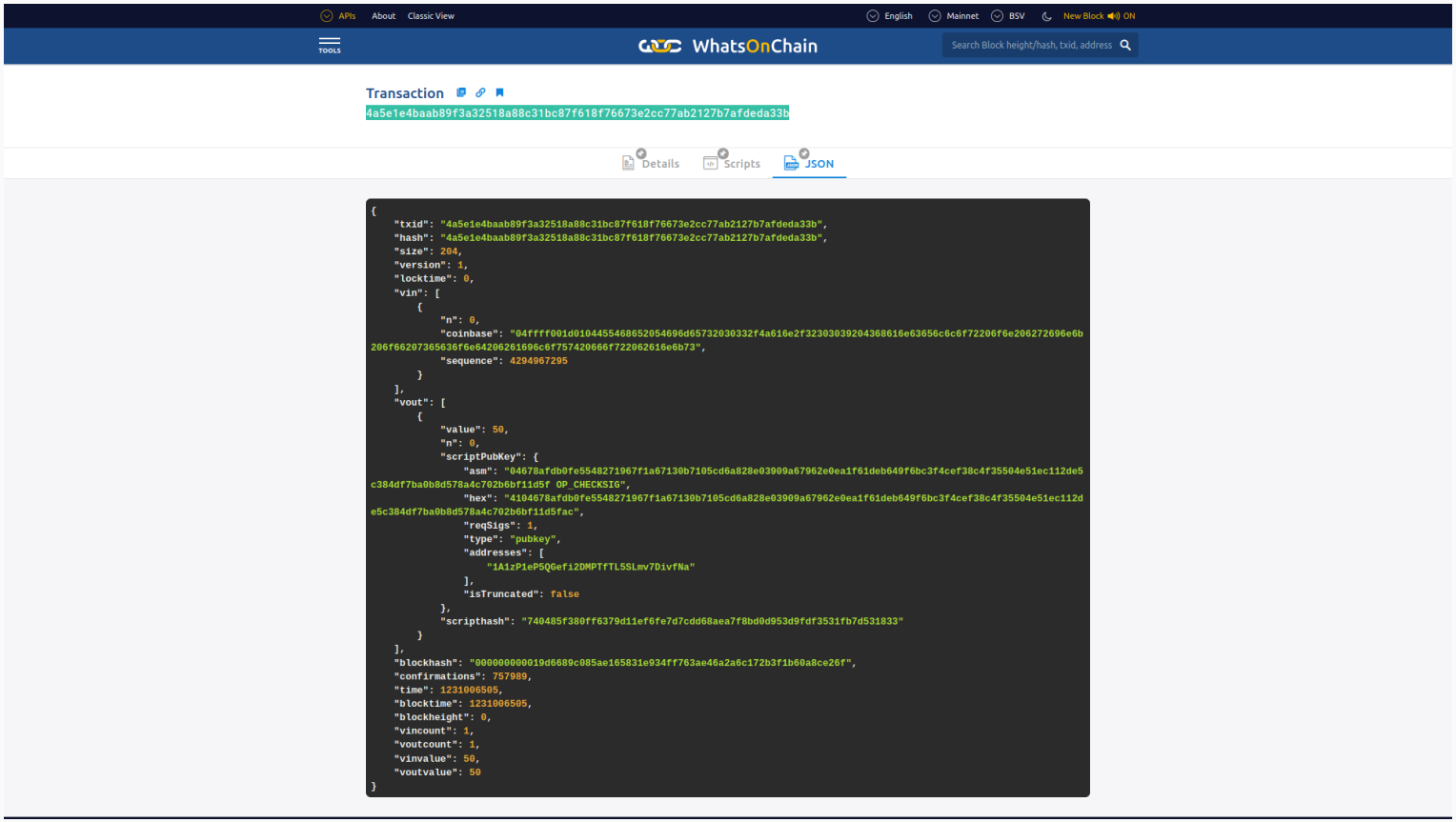
A carteira de Satoshi vii

- Scripts



A carteira de Satoshi viii

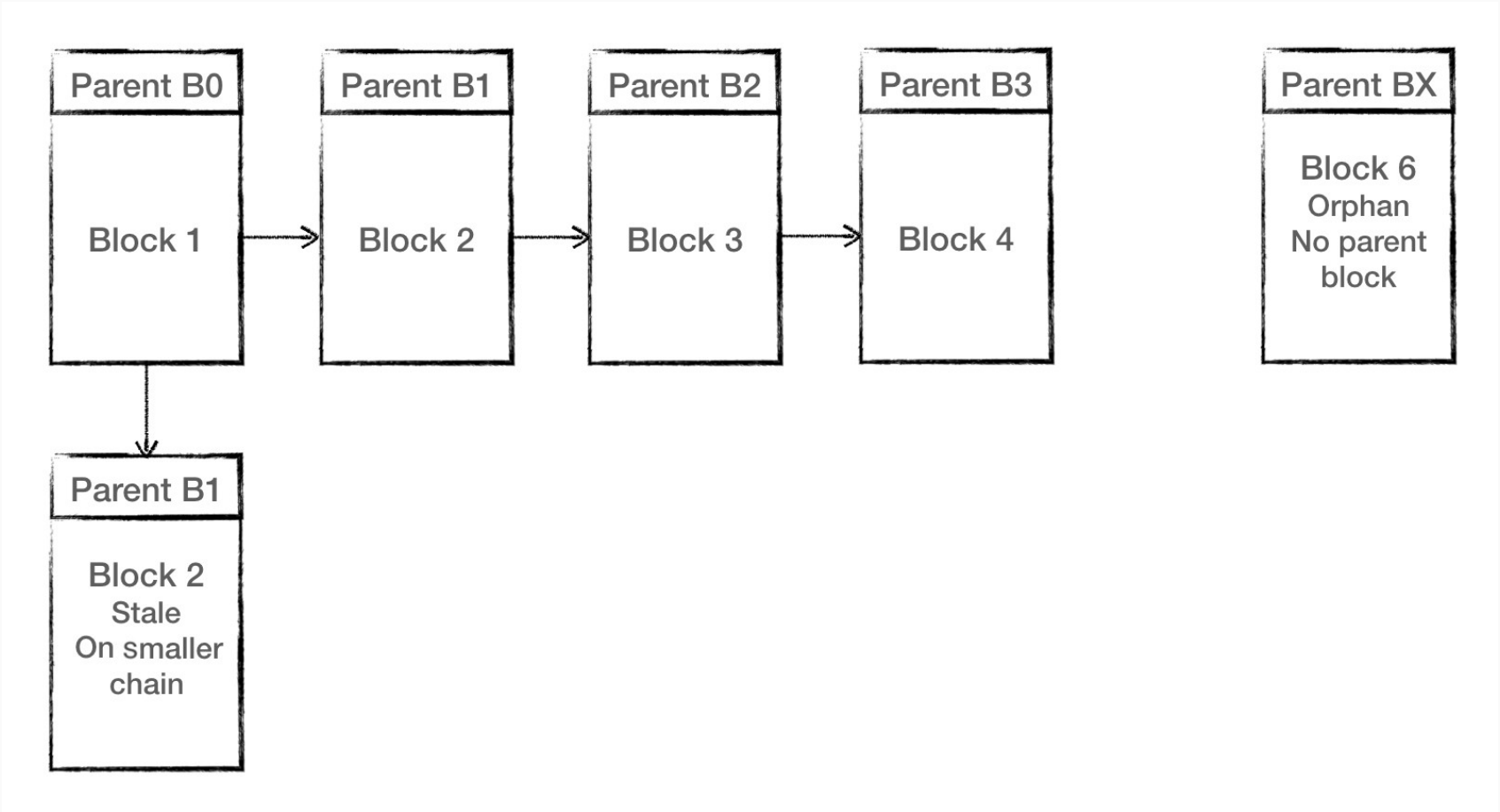
- JSON



Blocos Obsoletos e Orfãos i

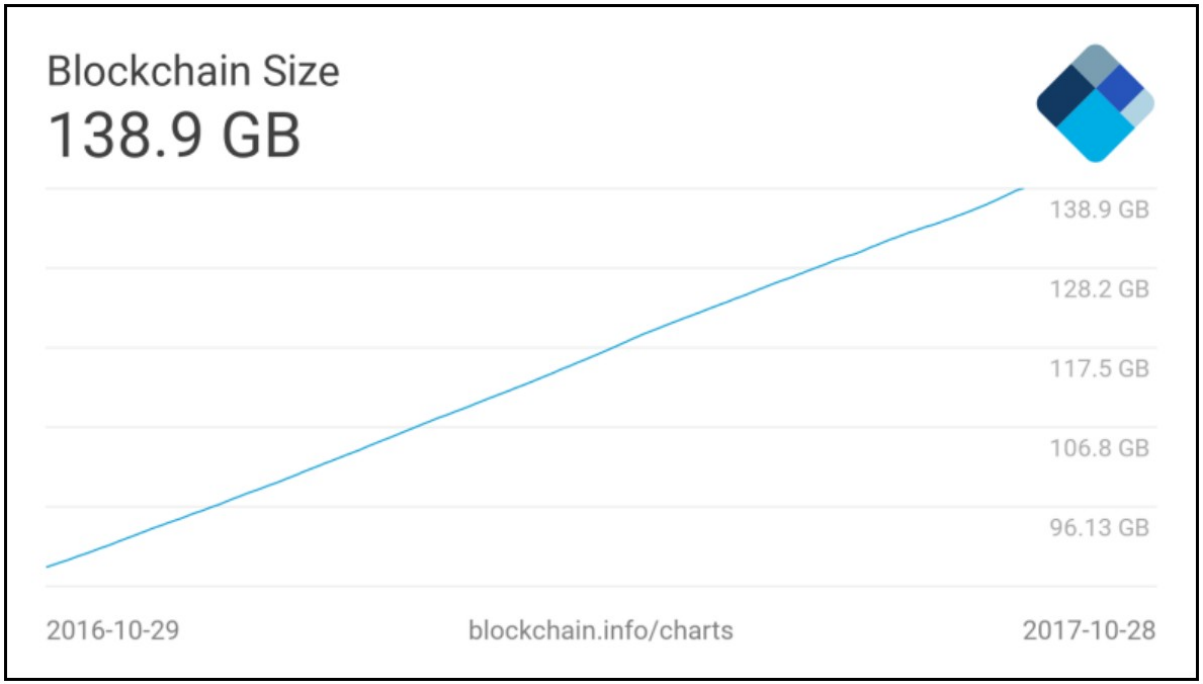
- Blocos obsoletos existem em uma cadeia mais curta, a partir da qual a cadeia principal progrediu.
- Os blocos pais de blocos órfãos são desconhecidos.

Blocos Obsoletos e Orfãos ii



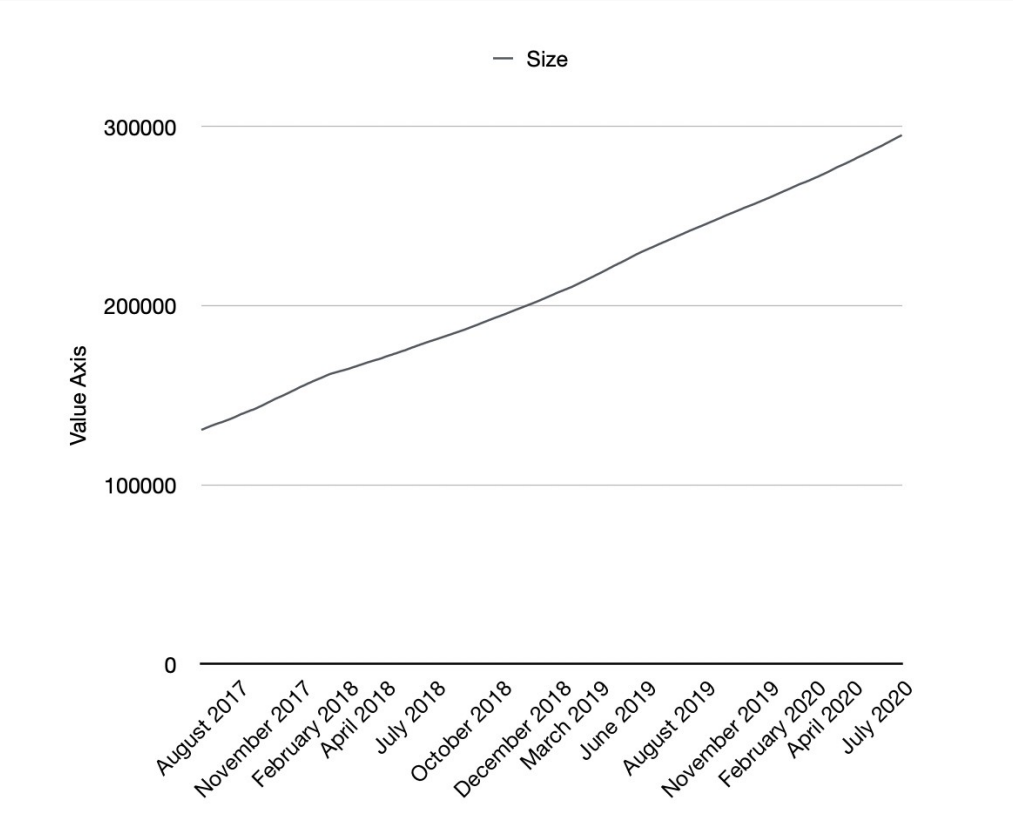
Tamanho do Blockchain do Bitcoin i

- O *Blockchain* do *Bitcoin* tinha em **October 29, 2017**, aproximadamente: $139GB$



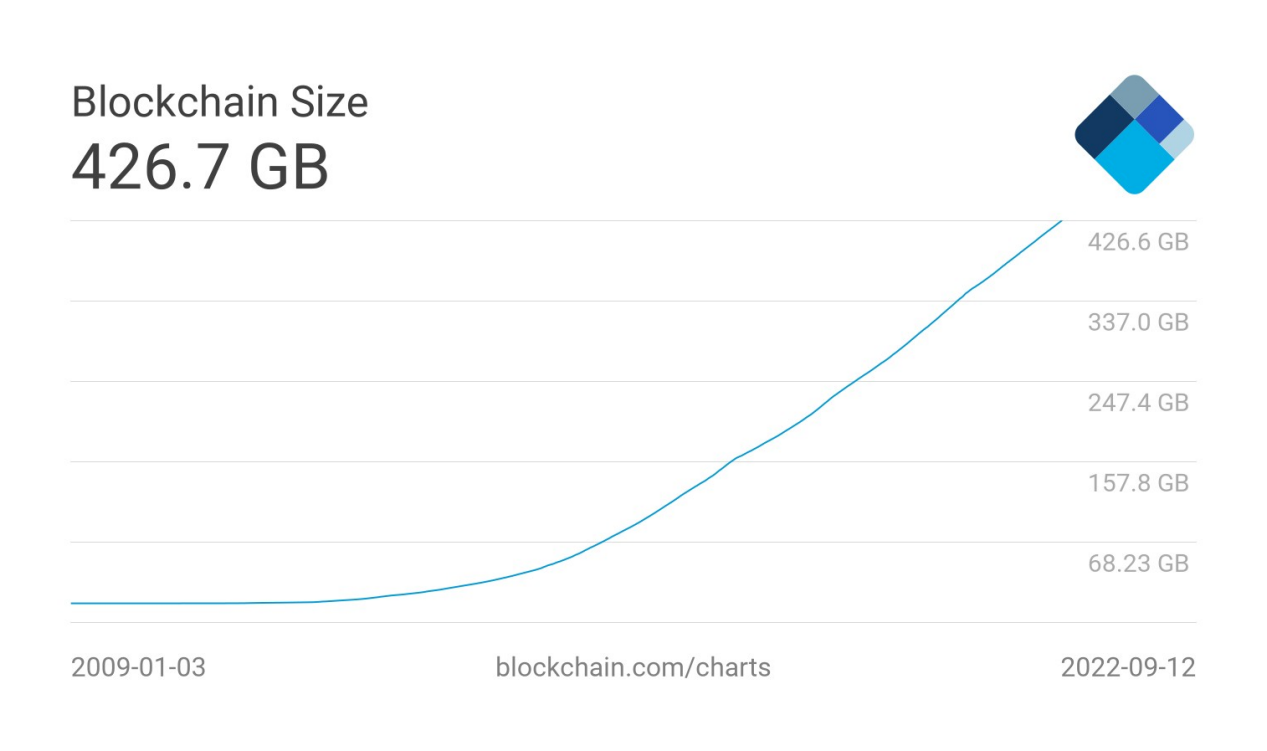
Tamanho do Blockchain do Bitcoin ii

- A figura mostra a evolução de **Aug 2017** para **Jul 2020**. Aproximandamente, $286GB$.



Tamanho do Blockchain do Bitcoin iii

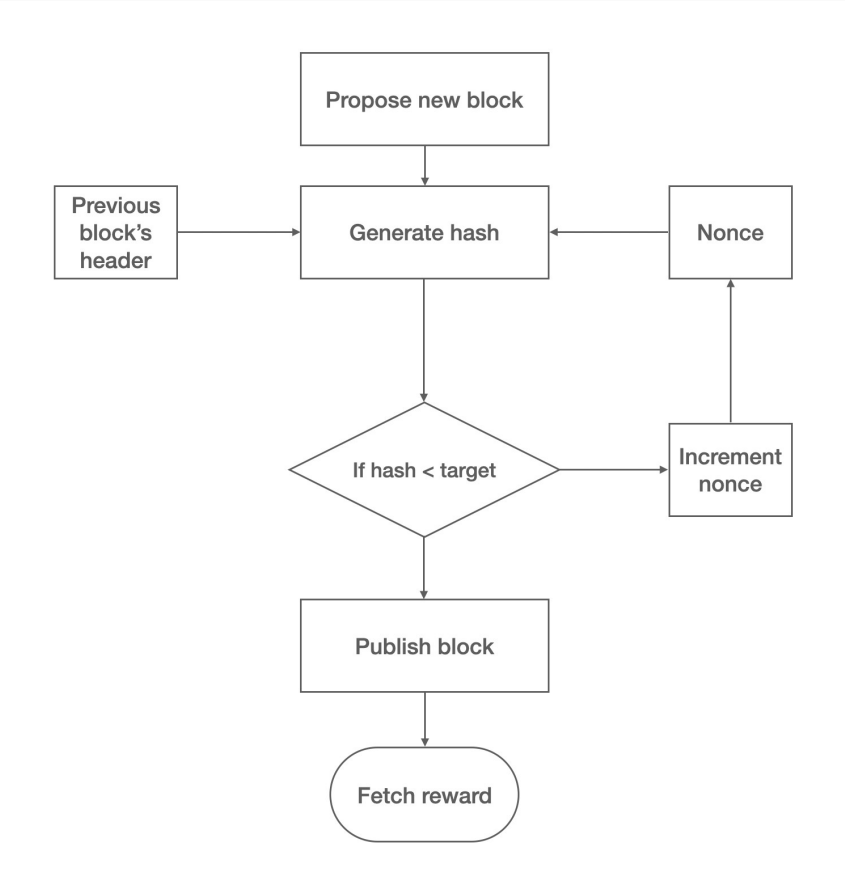
- A figura mostra a evolução de **Jan 2009** para **Set 2022**. Aproximandamente, $426.7GB$.



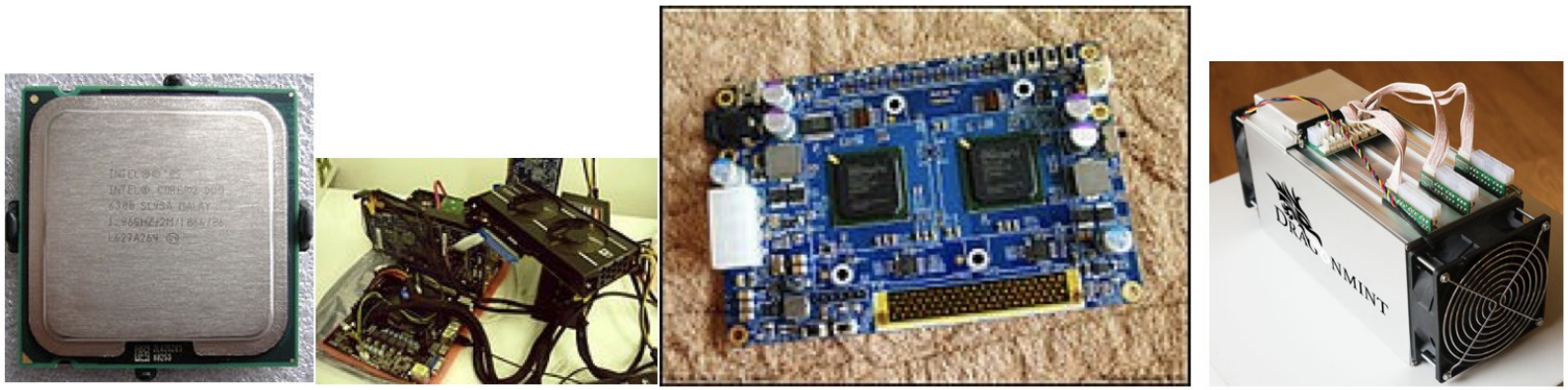
- Fonte: https://www.blockchain.com/charts/blocks-size

Mineração i

- Synchronizing with the network
- Transaction validation
- Block validation
- Create a new block
- Perform PoW
- Fetch reward



Mining systems

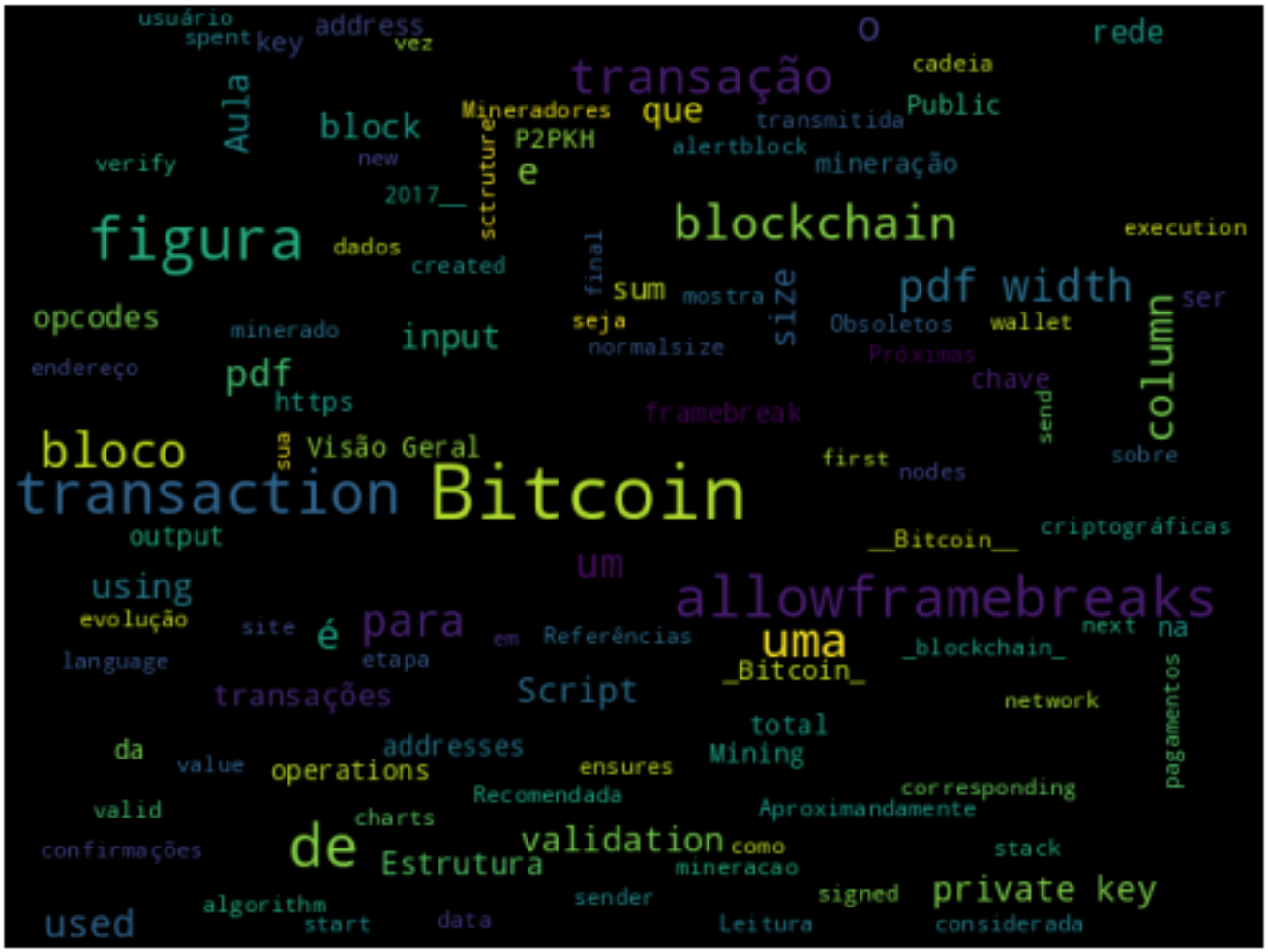


Four types of mining hardware. From left to right: a CPU, GPU, FPGA, and an ASIC

Atividade

- Instalar o Bitcoin client e executar experimentos nele.

Word Cloud


usuário
spent
key
address
vez
o
rede
transação
cadeia
Aula
Mineradores
que
Public
block
P2PKH
transmitida
verify
new
alertblock
e
mineração
2017__
sctruture
figura
blockchain
execution
dados
created
final
sum
mostra
size
pdf width
ser
opcodes
minerado
input
seja
Obsoletos
wallet
normalsize
Próximas
endereço
pdf
chave
column
https
framebreak
send
bloco
sua
Visão Geral
first
sobre
transaction
Bitcoin
nodes
output
__Bitcoin__
criptográficas
um
using
allowframebreaks
evolução
site
é
para
em
Referências
uma
next
na
language
etapa
blockchain
Bitcoin
transações
Script
network
total
da
addresses
Mining
pagamentos
value
operations
ensures
corresponding
Recomendada
Aproximandamente
valid
charts
de
validation
como
stack
confirmações
Estrutura
mineracao
private key
algorithm
sender
signed
used
start
data
Leitura
considerada

Leitura Recomendada

Leitura Recomendada

Capítulo 5/6: Introduction Bitcoin

Livro: IMRAN BASHIR. Mastering Blockchain : Distributed Ledger Technology, Decentralization, and Smart Contracts Explained, 2nd Edition.

Próximas Aulas

Próximas Aulas

- Pagamentos com *Bitcoin*.

Referências

Referências i

Imran, Bashir. 2018. *Mastering Blockchain : Distributed Ledger Technology, Decentralization, and Smart Contracts Explained, 2nd Edition.* Packt Publishing. https://search.ebscohost.com/login.aspx?direct=true&db=e000xww&AN=1789486&lang=pt-br&site=eds-live&scope=site.