Blockchain economy

Ilgiz Gimaltdinov

Транзакции: input, ouput

Transaction 7957a35fe64f80d234d76d83a2a8f1a0d8149a41d81de548f0a65a8a999f6f18

INPUTS From		OUTPUTS To	
From (previous transactions Joe has received):		Output #0 Alice's Address	0.1000 BTC (spent)
Joe	0.1005 BTC	Transaction Fees:	0.0005 BTC

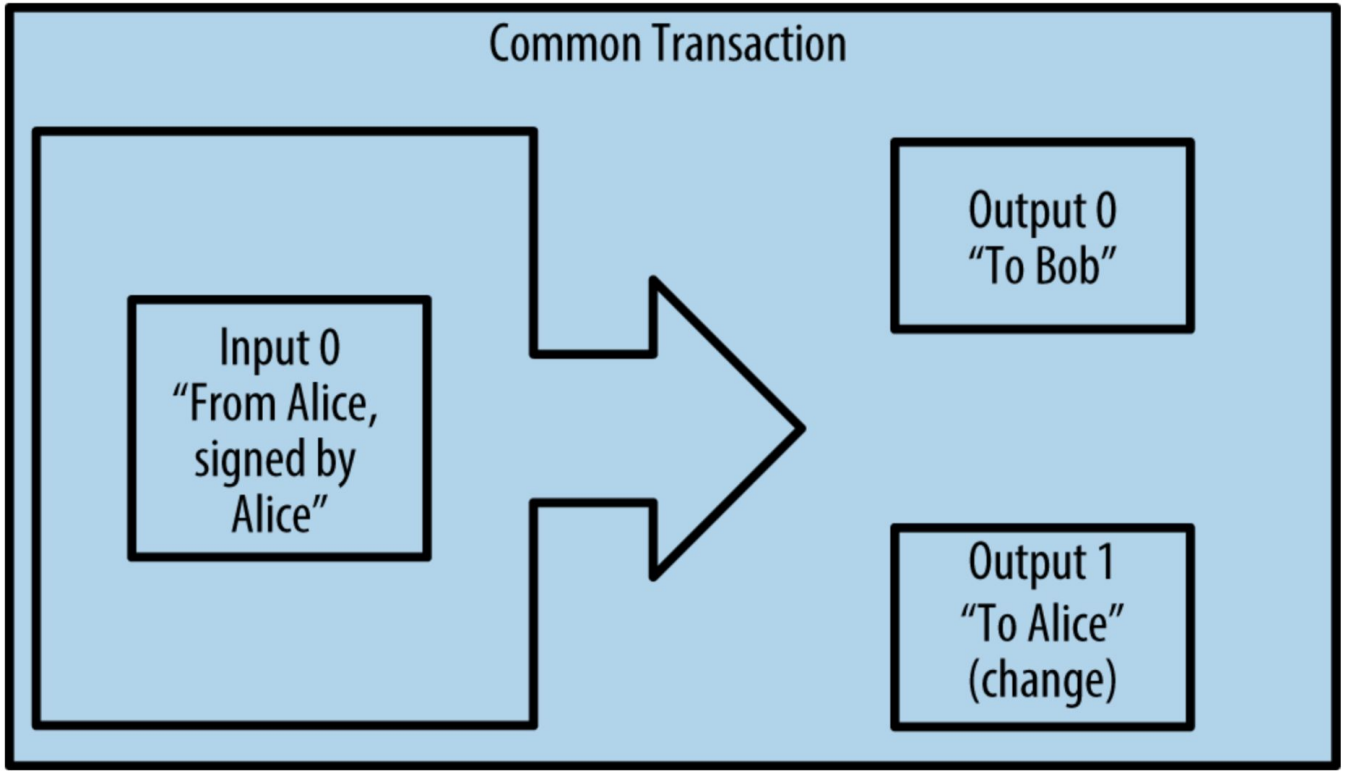
Transaction 0627052b6f28912f2703066a912ea577f2ce4da4caa5a5fbd8a57286c345c2f2

INPUTS From		OUTPUTS To	
7957a35fe64f80d234d76d83a2a8f1a0d8149a41d81de548f0a65a8a999f6f18 : 0		Output #0 Bob's Address	0.0150 BTC (spent)
Alice	0.1000 BTC	Output #1 Alice's Address (change)	0.0845 BTC (unspent)
		Transaction Fees:	0.0005 BTC

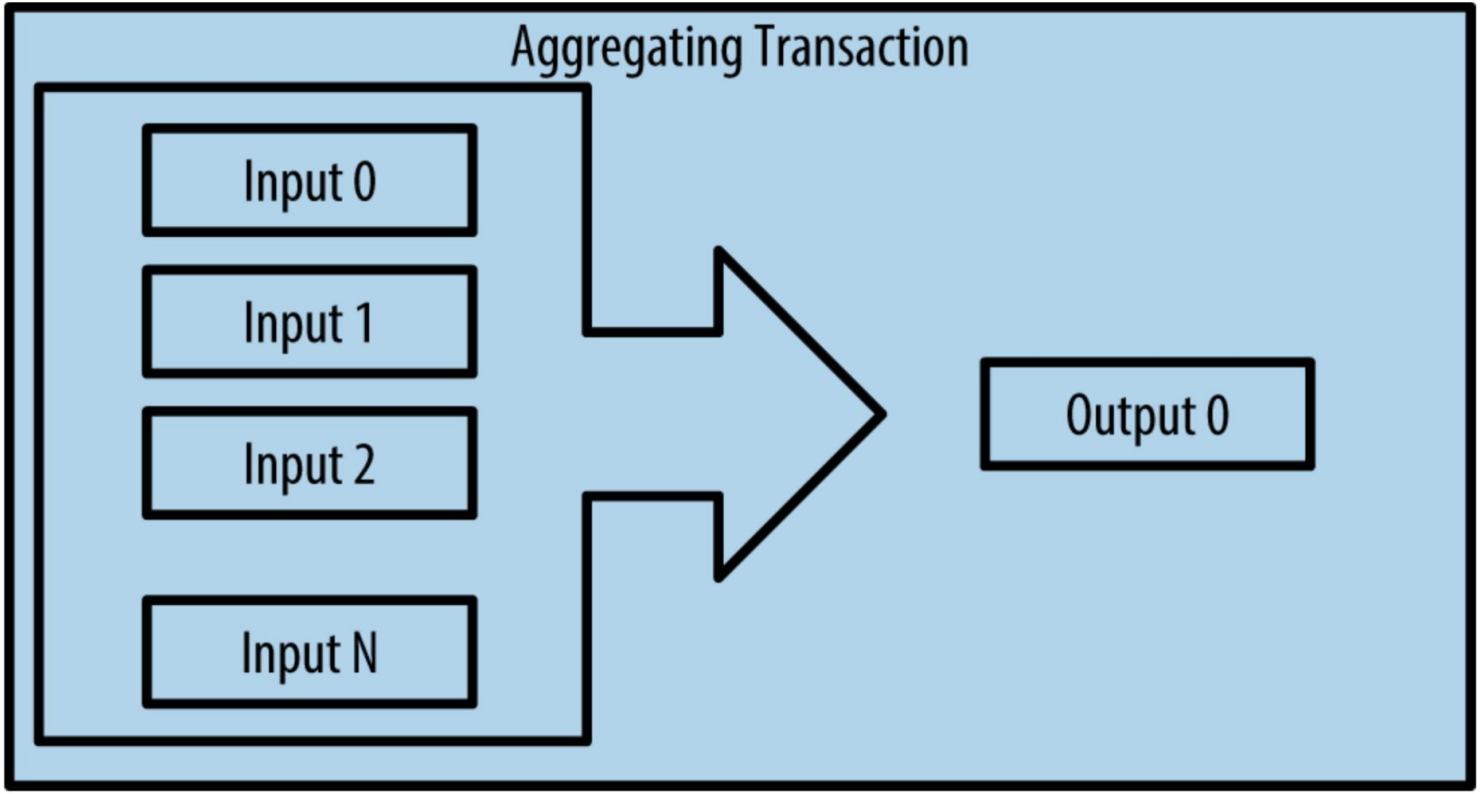
Transaction 2bbac8bb3a57a2363407ac8c16a67015ed2e88a4388af58cf90299e0744d3de4

INPUTS From		OUTPUTS To	
0627052b6f28912f2703066a912ea577f2ce4da4caa5a5fbd8a57286c345c2f2 : 0		Output #0 Gopesh's Address	0.0100 BTC (unspent)
Bob	0.0150 BTC	Output #1 Bob''s Address (change)	0.0845 BTC (unspent)
		Transaction Fees:	0.0005 BTC

Транзакции


Common Transaction
Input 0
"From Alice,
signed by
Alice"
Output 0
"To Bob"
Output 1
"To Alice"
(change)

Транзакции


Aggregating Transaction
Input 0
Input 1
Input 2
Input N
Output 0

Жизненный цикл транзакции

0627052b6f28912f2703066a912ea577f2ce4da4caa5a5fbd8a57286c345c2f2

1Cdid9KFAaatwczBwBttQcwXYCpvK8h7FK (0.1 BTC - Output)

1GdK9UzpHBzqzX2A9JFP3Di4weBwqgmoQA - (Unspent) 0.015 BTC

1Cdid9KFAaatwczBwBttQcwXYCpvK8h7FK - (Unspent) 0.0845 BTC

97 Confirmations | 0.0995 BTC

Summary	
Size	258 (bytes)
Received Time	2013-12-27 23:03:05
Included In Blocks	277316 (2013-12-27 23:11:54 +9 minutes)

Inputs and Outputs	
Total Input	0.1 BTC
Total Output	0.0995 BTC
Fees	0.0005 BTC
Estimated BTC Transacted	0.015 BTC

Жизненный цикл транзакции:

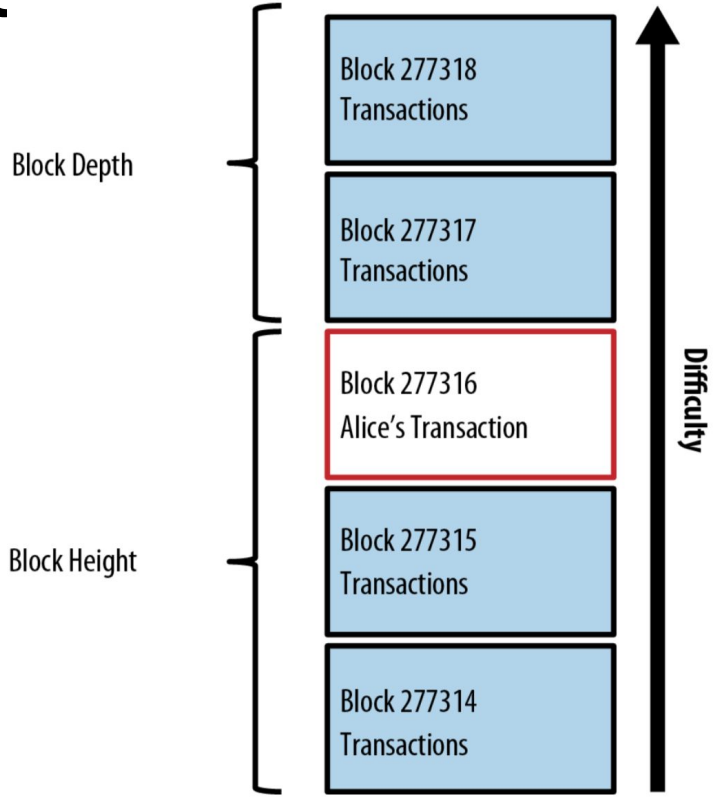
1. Соседние ноды с Alice получают информацию о транзакции
2. Распространение по сети неподтвержденной транзакции
3. Подтверждение транзакции при создании блока
4. Распространение блока по сети
5. Боб получает биткоин

Когда Боб может тратить деньги?

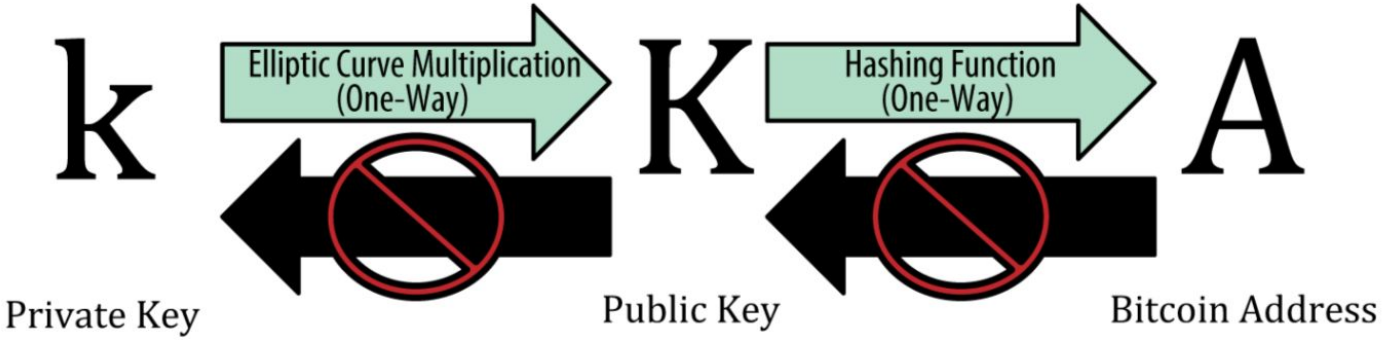
Понятие высоты и глубины блока

- Высота блока – порядковый номер выбранного блока начиная с блока генезиса

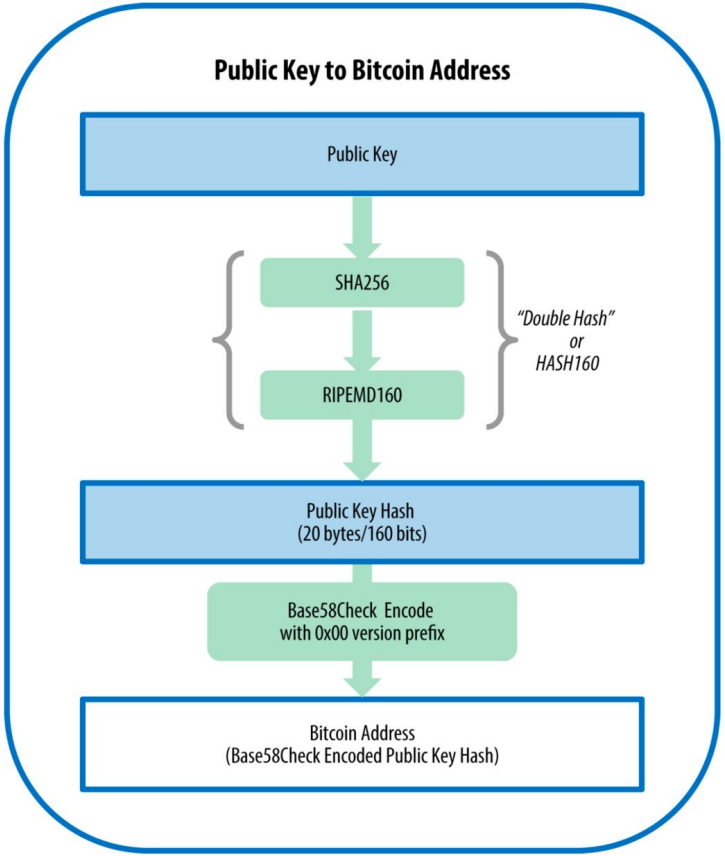
- Глубина блока – количество блоков в цепочке блокчейна после выбранного блока



Приватный, публичный ключ, адрес кошелька



Base58

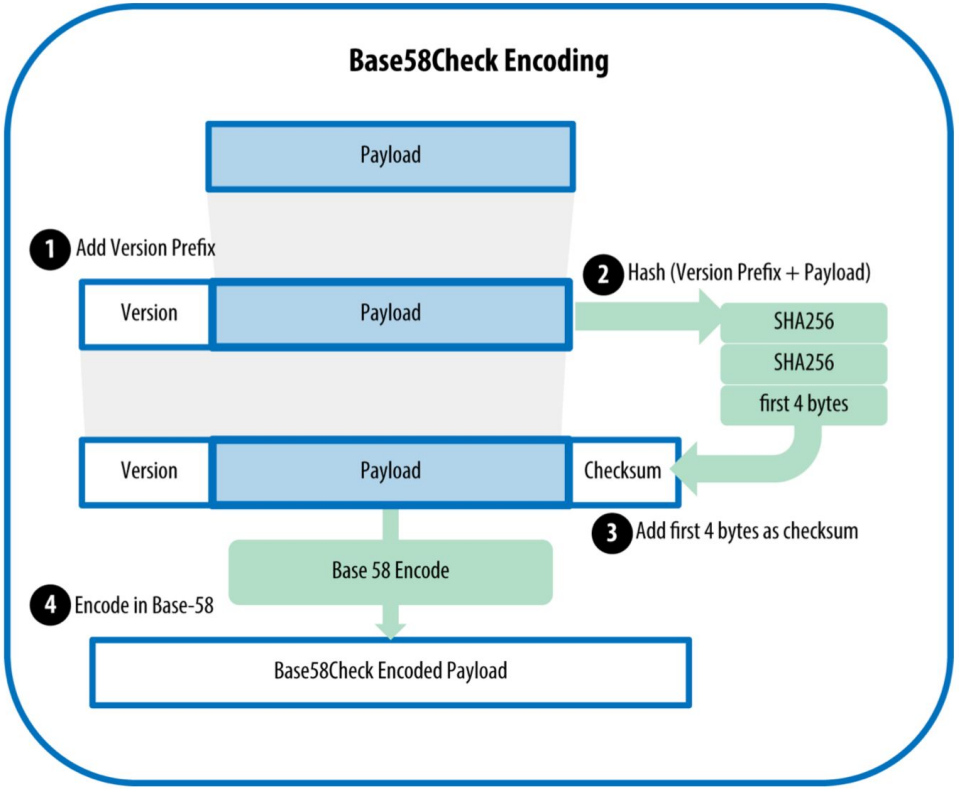


приватный ключ может быть любым числом между 1 и n - 1, где n константа (n = 1.158 * 10^77, немного меньше, чем 2^256)

Алгоритм генерации приватного ключа:

1. Выбираем случайное число
2. Применяем алгоритм SHA256
3. Публичный ключ вычисляется из приватного ключа с помощью необратимого умножения на эллиптических кривых (алгоритм secp256k1, принятий NIST)

Base58Check Encoding Key



Type	Version prefix (hex)	Base58 result prefix
Биткоин-адрес	0x00	1
Pay-to-Script-Hash адрес	0x05	3
Testnet-адрес	0x6F	m или n
Приватный ключ WIF	0x80	5, K или L
BIP38 зашифрованный приватный ключ	0x0142	6P
BIP32 Расширенный публичный ключ	0x0488B21E	xpub

Кошелек

- Кошелек — это контейнер для приватных ключей

Кошелек

- Кошелек — это контейнер для приватных ключей

- Простейший способ генерации ключей: детерминистическая генерация ключей

Недетерминированные кошельки

- Кошелек — это контейнер для приватных ключей

- Генерация ключей:
  - Детерминистическая: каждый новый приватный ключ получается использованием односторонней хэш-функции от предыдущего закрытого ключа, связывая их в последовательности
  - Недетерминистическая: случайная генерация ключей

Детерминированные (с зерном) кошельки

- приватные ключи происходят от общего зерна через использование односторонней хэш-функции.
- Зерно — это случайное число, которое в сочетании с другими данными, такими как номер индекса, позволяет получить приватные ключи

Мнемонические кодовые слова

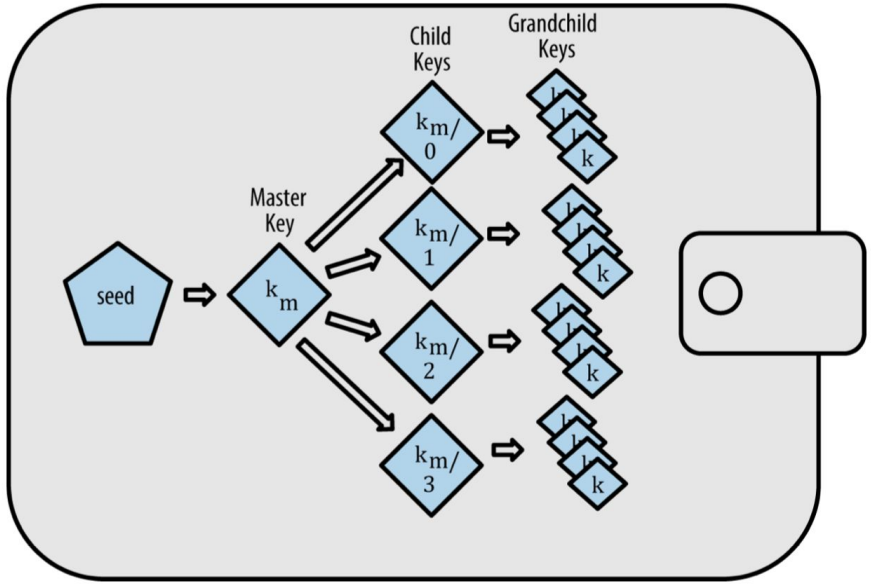
- Мнемонические коды — это последовательности английских слов, которые представляют собой (кодируют) случайное число, которое используется в качестве зерна для детерминированного кошелька

BIP0039 (создание мнемонического кода и зерна):

- Создать случайную последовательность (энтропию) длиной от 128 до 256 бит.
- Создать контрольную сумму случайной последовательности, взяв первые несколько битов из ее SHA256 хэша.
- Добавить контрольную сумму в конец случайной последовательности.
- Разделить последовательность на части длиной в 11 бит, и использовать их в качестве индекса по словарю из 2048 предопределенных слов.
- Получить от 12 до 24 слов, представляющих собой мнемонический код.

Иерархические детерминированные кошельки (BIP0032/BIP0044)

Иерархические детерминистические кошельки содержат ключи в виде древовидной структуры, так что из родительского ключа можно вывести последовательность производных ключей, от каждого из которых, в свою очередь, также получается последовательность производных ключей и так далее без ограничения глубины вложенности



Структура транзакции

Table 1. Структура транзакции

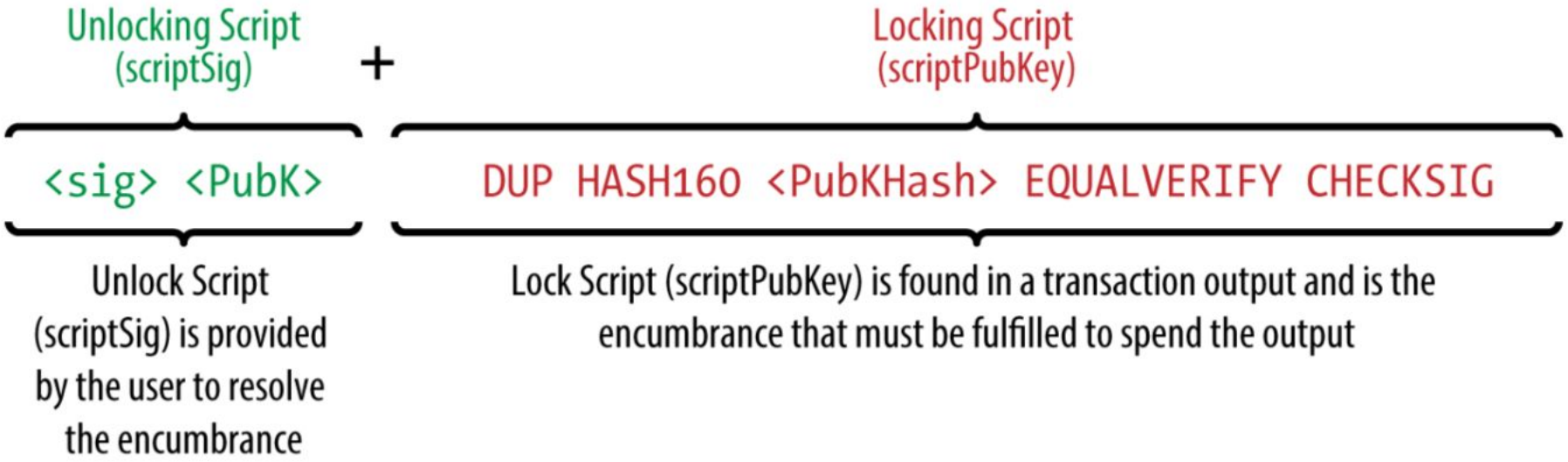
Размер	Поле	Описание
4 байта	Версия	Описывает каким правилам подчиняется данная транзакция
1-9 байтов (VarInt)	Счетчик входов	Число входов транзакции
Переменный	Входы	Один или несколько входов транзакции
1-9 байтов (VarInt)	Счетчик выходов	Число выходов транзакции
Переменный	Выходы	Один или несколько выходов транзакции
4 байта	Locktime	UNIX-время или номер блока

Выход транзакции

Table 2. Структура выхода транзакции

Размер	**Поле**	**Описание**
8 байтов	Сумма	Количество в сатошах (10^{-8} bitcoin)
1-9 байтов (VarInt)	Размер запирающего сценария	Длина в байтах
Переменная	Запирающий сценарий	Сценарий, определяющий условия, удовлетворение которых требуется для того, чтобы можно было потратить выход

Подпись транзакции



Язык сценариев

2+7-3+1 == 7

2 7 OP_ADD 3 OP_SUB 1 OP_ADD 7 OP_EQUAL

<u>Пример</u>:

Скрипт проверки: 3 OP_ADD 5 OP_EQUAL

Сценарий со входом 2 – разблокирует скрипт

Полнота/неполнота по Тьюрингу

- Язык Bitcoin:
  - неполный по Тьюрингу
  - stateless
- Язык Ethereum – Тьюринг-полный

Тьюринг-полнота: означает возможность реализовать на нём любую вычислимую функцию (т. е. можно написать алгоритм, вычисляющий значение функции)

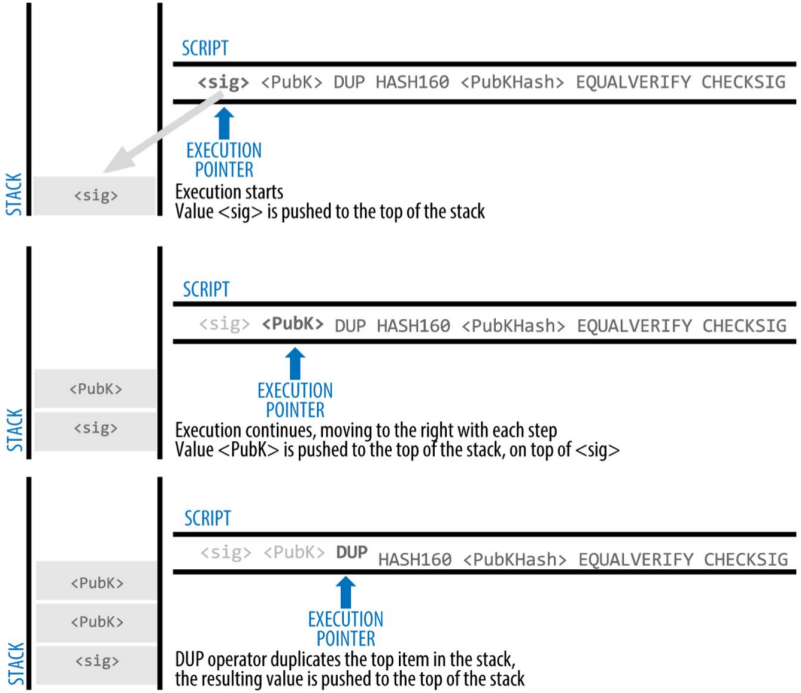
Виды отпирающих скриптов

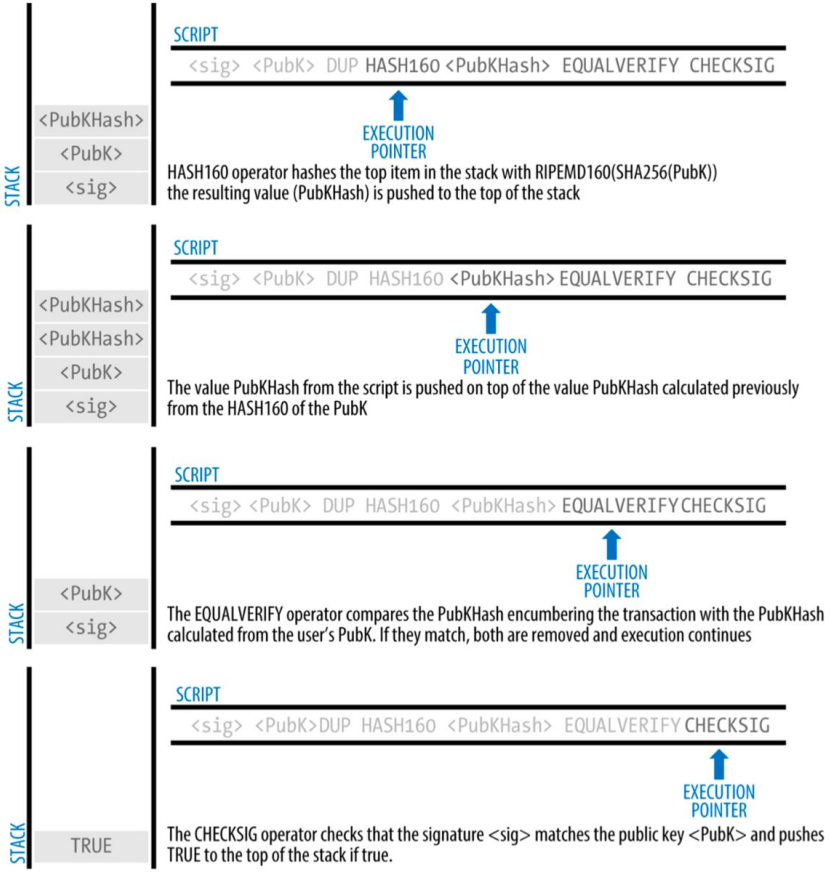
- Pay-to-Public-Key-Hash (P2PKH)
- Pay-to-Public-Key
- multi- signature
- pay-to-script-hash (P2SH)
- выход данных (OP_RETURN)

Pay-to-Public-Key-Hash (P2PKH)

- запирающий сценарий, который обременяет выход хешем публичного ключа (адрес Bitcoin)
- OP_DUP OP_HASH160 <Public Key Hash> OP_EQUAL OP_CHECKSIG
- Отпирающий скрипт: <Signature> <Public Key>

Pay-to-Public-Key-Hash (P2PKH)





Pay-to-Public-Key

- запирающий сценарий - публичный ключ
- <Public Key A> OP_CHECKSIG
- Отпирающий скрипт: <Signature from Private Key A>

Зачем нужен P2KH, если есть P2PK?

Multisig

- запирающий сценарий - публичный ключ M-из-N
- M <Public Key 1> <Public Key 2> ... <Public Key N> N OP_CHECKMULTISIG

Пример:

- 2 <Public Key A> <Public Key B> <Public Key C> 3 OP_CHECKMULTISIG
- Отпирающий скрипт: OP_0 <Signature B> <Signature C>

Вывод данных (OP_RETURN)

- Добавляется 80 байт к данным транзакции
- OP_RETURN <data>
- OP_RETURN явно создает *доказуемо неспособный быть потраченным* выход

Для чего это нужно:

- Цифровой нотариус

Pay-to-Script-Hash (P2SH)

Table 4. Сложный скрипт без P2SH

Запирающий скрипт	2 PubKey1 PubKey2 PubKey3 PubKey4 PubKey5 5 OP_CHECKMULTISIG
Отпирающий скрипт	Sig1 Sig2

Table 5. Комплексный сценарий в качестве P2SH

Погашающий скрипт	2 PubKey1 PubKey2 PubKey3 PubKey4 PubKey5 5 OP_CHECKMULTISIG
Запирающий скрипт	OP_HASH160 <20-байтный хэш погашающего скрипта> OP_EQUAL
Отпирающий скрипт	Sig1 Sig2 погашающий скрипт

Преимущества P2SH

- Сложные сценарии заменяются на более хеши на выходе транзакции, что делает размер транзакций меньше.
- Сценарии могут быть закодированы в виде адреса, так что отправителю и кошельку отправителя не требуется сложной реализации P2SH.
- P2SH перекладывает бремя построения сценария на получателя, а не отправителя.
- P2SH перекладывает бремя хранения данных для длинного сценария с выхода (который находится во множестве UTXO) на вход (который хранится в blockchain).
- P2SH перекладывает бремя хранения данных для длинного сценария с настоящего времени (момента платежа) на будущее (момент распоряжения средствами).
- P2SH переносит оплату комисионных за длинный сценарий с отправителя на

Сеть биткоин

Децентрализованная Сетевая Архитектура

Типы и роли узлов

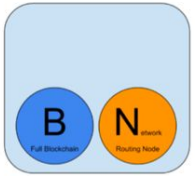
Функции биткоин узла:

- маршрутизация
- базы данных blockchain
- майнинг
- кошелек

Типы и роли узлов

Reference Client (Bitcoin Core)

Contains a Wallet, Miner, full Blockchain database, and Network routing node on the bitcoin P2P network.

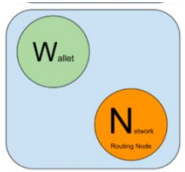


Full Block Chain Node

Contains a full Blockchain database, and Network routing node on the bitcoin P2P network.

Solo Miner

Contains a mining function with a full copy of the blockchain and a bitcoin P2P network routing node.



Lightweight (SPV) wallet

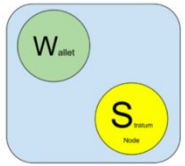
Contains a Wallet and a Network node on the bitcoin P2P protocol, without a blockchain.

Pool Protocol Servers

Gateway routers connecting the bitcoin P2P network to nodes running other protocols such as pool mining nodes or Stratum nodes.

Mining Nodes

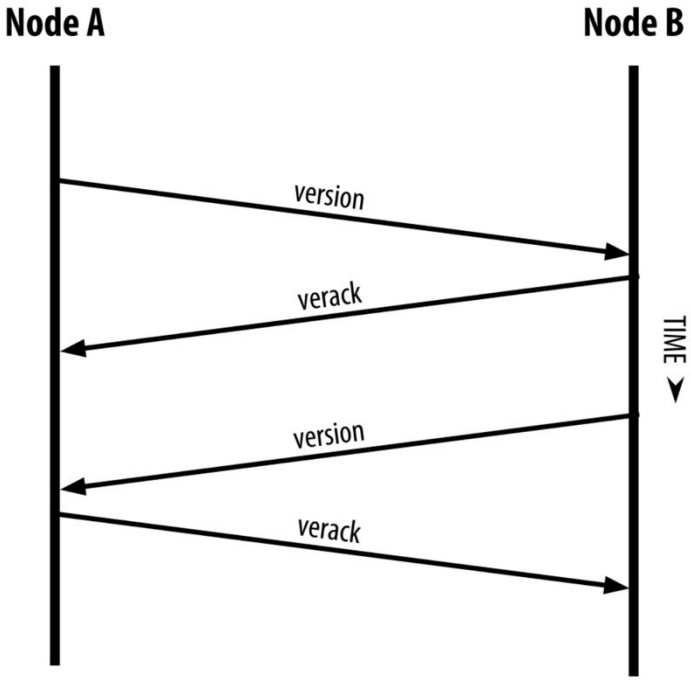
Contain a mining function, without a blockchain, with the Stratum protocol node (S) or other pool (P) mining protocol node.



Lightweight (SPV) Stratum wallet

Contains a Wallet and a Network node on the Stratum protocol, without a blockchain.

Подключение к сети Bitcoin



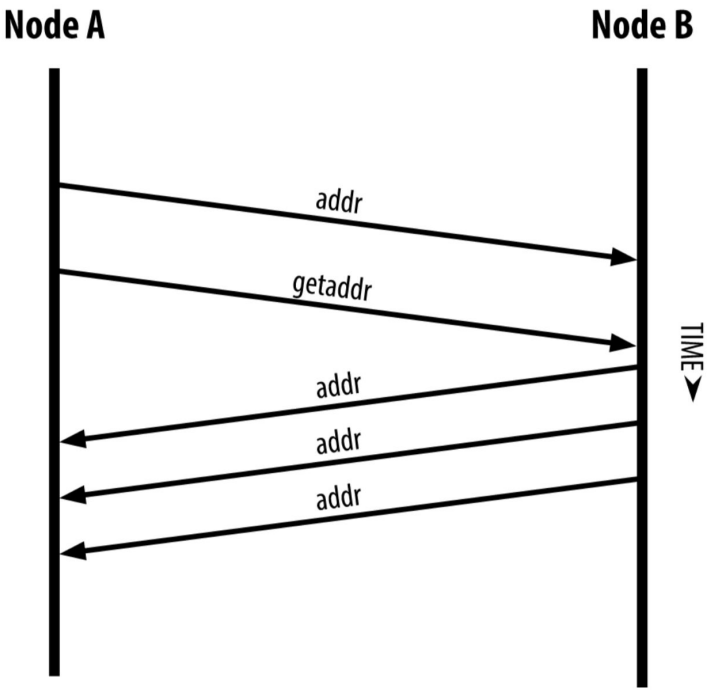
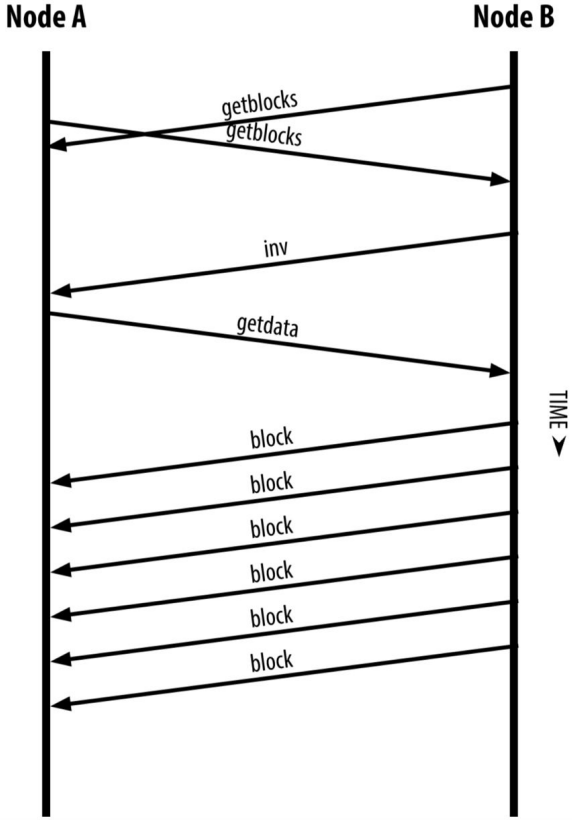


Figure 4. Начальное рукопожатие между пирами

Узлы упрощенного подтверждения оплаты (SPV)

SPV-узлы загружают только заголовки блоков и не загружают транзакции, включенные в каждый блок.

Полный узел будет полностью проверяет всю цепочку блоков и транзакций вниз по blockchain (во времени) вплоть до блока генезиса
SPV-узел будет проверять цепь всех блоков и связывать эту цепь с интересующей транзакцией (с помощью Merkle root)



Конфиденциальность и SPV

- SPV-узлы получают определенные транзакции для их выборочной проверкти
- **Фильтр Блума**: вероятностный фильтр поиска транзакции без точного ее описания