How to proof authorization to spend UTXO

Outline

1. UTXO?
2. The Anatomy of a Bitcoin Transaction
3. Script

1. Unspent transaction outputs (UTXO) -- What are they?

Unspent transaction outputs are various amounts of bitcoin that can still be transferred (spent). They haven't been used as transaction inputs yet.

A transaction takes an UTXO as input and creates new UTXO as output(s).

UTXOs are typically locked to a specific owner.

2. The Anatomy of a Bitcoin Transaction

Table 5-1. The structure of a transaction

Size	Field	Description
4 bytes	Version	Specifies which rules this transaction follows
1–9 bytes (VarInt)	Input Counter	How many inputs are included
Variable	Inputs	One or more transaction inputs
1–9 bytes (VarInt)	Output Counter	How many outputs are included
Variable	Outputs	One or more transaction outputs
4 bytes	Locktime	A Unix timestamp or block number

2. The Anatomy of a Bitcoin Transaction

Table 5-2. The structure of a transaction output

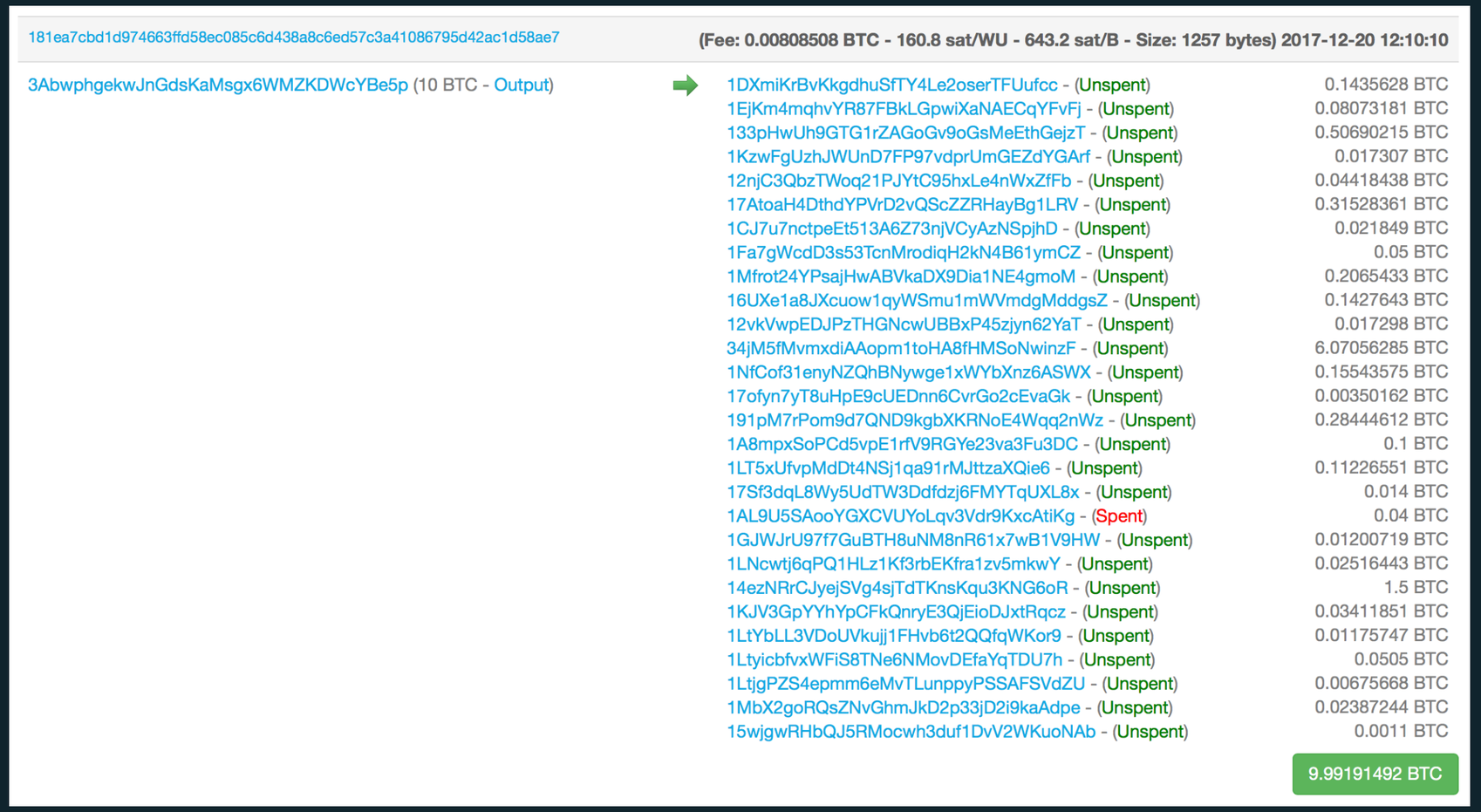
Size	Field	Description
8 bytes	Amount	Bitcoin value in satoshis (10^{-8} bitcoin)
1-9 bytes (VarInt)	Locking-Script Size	Locking-Script length in bytes, to follow
Variable	Locking-Script	A script defining the conditions needed to spend the output

2. The Anatomy of a Bitcoin Transaction

Table 5-3. The structure of a transaction input

Size	Field	Description
32 bytes	Transaction Hash	Pointer to the transaction containing the UTXO to be spent
4 bytes	Output Index	The index number of the UTXO to be spent; first one is 0
1-9 bytes (VarInt)	Unlocking-Script Size	Unlocking-Script length in bytes, to follow
Variable	Unlocking-Script	A script that fulfills the conditions of the UTXO locking script.
4 bytes	Sequence Number	Currently disabled Tx-replacement feature, set to oxFFFFFFFF

2. The Anatomy of a Bitcoin Transaction



181ea7cbd1d974663ffd58ec085c6d438a8c6ed57c3a41086795d42ac1d58ae7 (Fee: 0.00808508 BTC - 160.8 sat/WU - 643.2 sat/B - Size: 1257 bytes) 2017-12-20 12:10:10

3AbwphgekwJnGdsKaMsgx6WMZKDWcYBe5p (10 BTC - Output)	1DXmiKrBvKkgdhuSfTY4Le2oserTFUufcc - (Unspent)	0.1435628 BTC
	1EjKm4mqhvYR87FBkLGpwiXaNAECqYFvFj - (Unspent)	0.08073181 BTC
	133pHwUh9GTG1rZAGoGv9oGsMeEthGejzT - (Unspent)	0.50690215 BTC
	1KzwFgUzhJWUnD7FP97vdprUmGEZdYGArf - (Unspent)	0.017307 BTC
	12njC3QbzTWoq21PJYtC95hxLe4nWxZfFb - (Unspent)	0.04418438 BTC
	17AtoaH4DthdYPVrD2vQScZZRHayBg1LRV - (Unspent)	0.31528361 BTC
	1CJ7u7nctpeEt513A6Z73njVCyAzNSpjhD - (Unspent)	0.021849 BTC
	1Fa7gWcdD3s53TcnMrodiqH2kN4B61ymCZ - (Unspent)	0.05 BTC
	1Mfrot24YPsajHwABVkaDX9Dia1NE4gmoM - (Unspent)	0.2065433 BTC
	16UXe1a8JXcuow1qyWSmu1mWVmdgMddgsZ - (Unspent)	0.1427643 BTC
	12vkVwpEDJPzTHGNcwUBBxP45zjyn62YaT - (Unspent)	0.017298 BTC
	34jM5fMvmxdiAAopm1toHA8fHMSoNwinzF - (Unspent)	6.07056285 BTC
	1NfCof31enyNZQhBNywge1xWYbXnz6ASWX - (Unspent)	0.15543575 BTC
	17ofyn7yT8uHpE9cUEDnn6CvrGo2cEvaGk - (Unspent)	0.00350162 BTC
	191pM7rPom9d7QND9kgbXKRNoE4Wqq2nWz - (Unspent)	0.28444612 BTC
	1A8mpxSoPCd5vpE1rfV9RGYe23va3Fu3DC - (Unspent)	0.1 BTC
	1LT5xUfvpMdDt4NSj1qa91rMJttzaXQie6 - (Unspent)	0.11226551 BTC
	17Sf3dqL8Wy5UdTW3Ddfdzj6FMYTqUXL8x - (Unspent)	0.014 BTC
	1AL9U5SAooYGXCVUYoLqv3Vdr9KxcAtiKg - (Spent)	0.04 BTC
	1GJWJrU97f7GuBTH8uNM8nR61x7wB1V9HW - (Unspent)	0.01200719 BTC
	1LNcwtj6qPQ1HLz1Kf3rbEKfra1zv5mkwY - (Unspent)	0.02516443 BTC
	14ezNRrCJyejSVg4sjTdTKnsKqu3KNG6oR - (Unspent)	1.5 BTC
	1KJV3GpYYhYpCFkQnryE3QjEioDJxtRqcz - (Unspent)	0.03411851 BTC
	1LtYbLL3VDoUVkujj1FHvb6t2QQfqWKor9 - (Unspent)	0.01175747 BTC
	1LtyicbfvxWFiS8TNe6NMovDEfaYqTDU7h - (Unspent)	0.0505 BTC
	1LtjgPZS4epmm6eMvTLunppyPSSAFSVdZU - (Unspent)	0.00675668 BTC
	1MbX2goRQsZNvGhmJkD2p33jD2i9kaAdpe - (Unspent)	0.02387244 BTC
	15wjgwRHbQJ5RMocwh3duf1DvV2WKuoNAb - (Unspent)	0.0011 BTC
		9.99191492 BTC

2. The Anatomy of a Bitcoin Transaction

What is an address?

`private key > public key > SHA256 > RIPEMD160 > Base58`

Base58 aka Bitcoin Address:
- balance between compact representation, readability, and error detection and prevention
- uses `123456789ABCDEFGHJKLMNPQRSTUVWXYZabcdefghijkmnopqrstuvwxyz` as it's alphabet without the 0 (number zero), O (capital o), l (lower L), I (capital i), and the symbols "+" and "/"

3. Script

Script is a stack based scripting language used to write locking and unlocking scripts for bitcoin transactions.

```
1 19 OP_ADD 12 OP_SUB 4 OP_ADD 13 OP_EQUAL
```

Question for the Audience: Does the script evaluate to TRUE?

3. Script

A valid unlocking script to a locking script:

```
unlocking script | locking script
2                | 19 OP_ADD 12 OP_SUB 4 OP_ADD 13 OP_EQUAL
```

3. Script

Actual valid Pay-To-Public-Key (P2PK) script:

```
unlocking script | locking script
<signature>      | <publicKey> OP_CHECKSIG

OP_CHECKSIG:
1. Hash the entire transaction's outputs, inputs, and script
2. Check if signature is valid for that hash and the public key
```

3. Script

The five valid script types:

— Pay-To-Public-Key-Hash (P2PKH)
— Pay-To-Public-Key (P2PK)
— Multi-Signature
— Data Output
— Pay-To-Script-Hash (P2SH)

3. Script

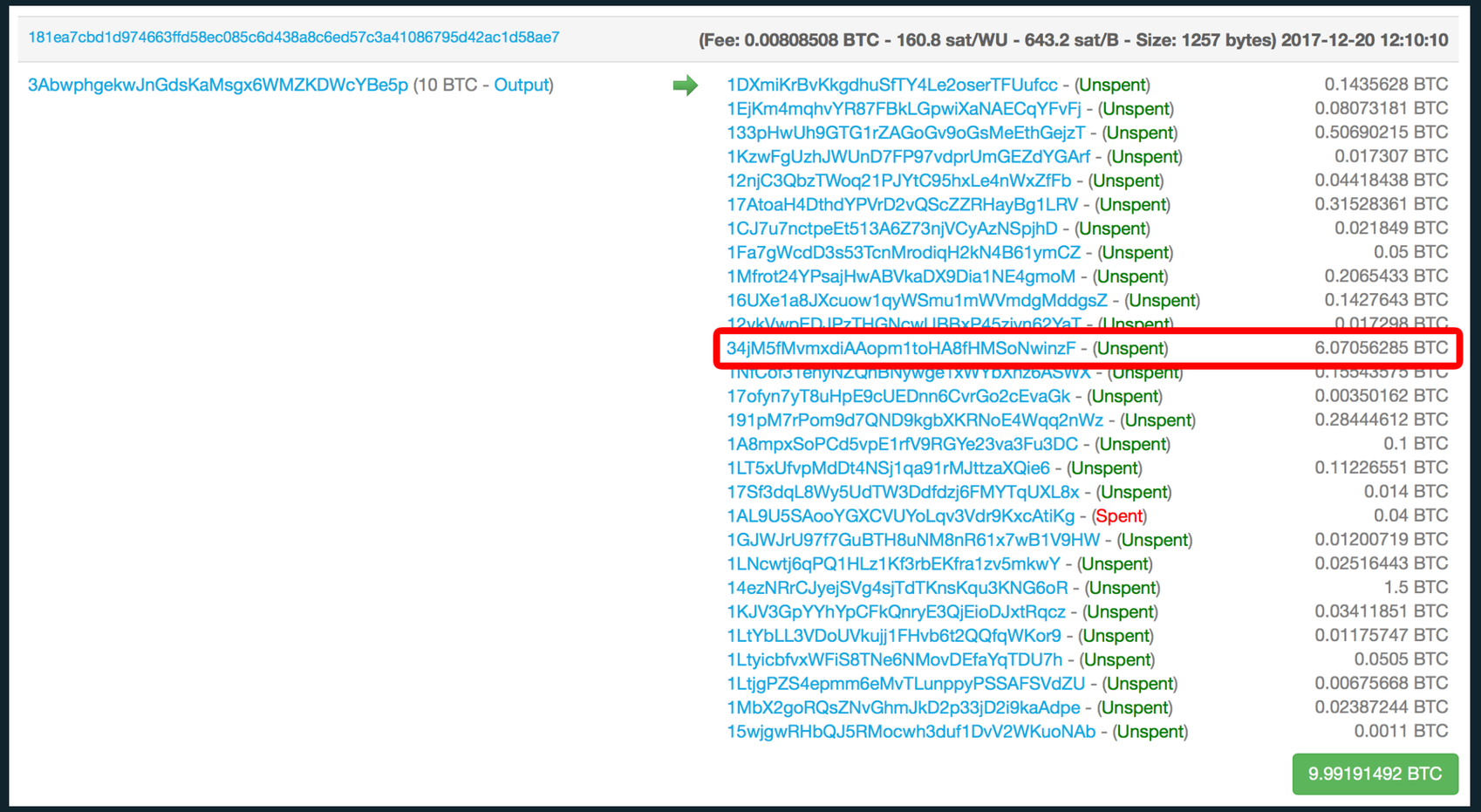
Script is limited in a way that it doesn't allow for loops, meaning that execution time is predictable and there can't be any DoS attacks on the network by creating long running scripts.

The scripts don't have a previous state, meaning all the state they need is contained in the script itself. They should execute the same way on any system.

Summary

1. UTXOs are spendable bitcoin currency (~64mio as of Dec. 20th)
2. A UTXO is usually locked to a double hashed public key
3. A unlocking script (providing signature and public key) has to be provided to spend the UTXO

Bonus Slide: Pay-to-Script-Hash (P2SH)



Benefits of Pay-to-Script-Hash (P2SH)

- Complex scripts are replaced by shorter fingerprints in the transaction output, making the transaction smaller
- Scripts can be coded as an address, so the sender and the sender's wallet don't need complex engineering to implement P2SH
- P2SH shifts the burden of constructing the script to the recipient, not the sender
- P2SH shifts the burden in data storage for the long script from the output (which is in the UTXO set) to the input (stored on the blockchain)
- P2SH shifts the burden in data storage for the long script from the present time (payment) to a future time (when it is spent)
- P2SH shifts the transaction fee cost of a long script from the sender to the recipient, who has to include the long redeem script to spend it

Thank you