BIP 13 - LINK

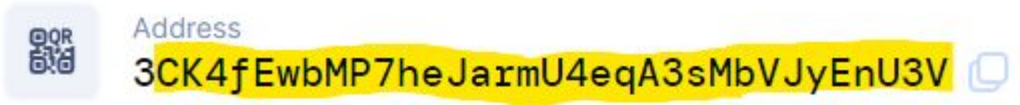
Address Format for pay-to-script-hash (P2SH)

How Does BIP 13 Improve Bitcoin?

This BIP is very straightforward; BIP 13 defines an address format (type) for pay to script hash (P2SH) transactions. Defined as below. These addresses start with a ‘3’

```
    base58-encode: [one-byte version][20-byte hash][4-byte checksum]
```

P2SH Addresses use a hex ‘0x05’ for the first character which becomes an ASCII ‘3’



Address
3CK4fEwbMP7heJarmU4eqA3sMbVJyEnU3V

The highlighted yellow portion of this example address is the input script hashed twice: first with SHA-256 and then with RIPEMD-160. A 4 byte checksum is included. This acts as error detection for the address.

Why are P2SH addresses even needed?

P2SH addresses extend the isStandard() function. This function was introduced in 2010 into Bitcoin Core by Gavin Andresen. This function creates restrictions in regard to which types of transactions get relayed across the network. *A.K.A if your transaction isn't normal it doesn't propagate across the network.*

After the implementation of isStandard() there was no longer a practical way to utilize the bitcoin scripting language on the network. Today this isn't completely true. The implementation of P2SH as a standard transaction allowed users to hash their scripts and then publish these hashes to the timechain. A spender could then provide the needed script and required inputs to spend those coins!

P2SH and Multisig ❤️Well Almost ...

Remember from the last lesson that pay to multisig (P2MS) was limited to a maximum **‘M of 3’** multisig though OP_CHECKMULTISIG could handle up to **‘M of 20’**? Well this limitation gets removed because suddenly you aren’t storing all the public keys on the timechain, instead only the 160 bit hash and a few opcodes are stored. You can wrap P2MS in P2SH

The well almost is because of the fact that P2SH transactions utilizing multisig must maintain not just the private keys but all **‘N’** public keys as well. This is because the entire original script must be recreated to spend those bitcoins.

What do the ‘P2SH’ Addresses represent?

A P2SH Address represent bitcoin that are unlocked by the encoded hash of a script and the successful execution of the script.

This is compared to the traditional ECDSA methods to unlock bitcoins in P2PKH transactions.

My analogy to this is similar to a treasure chest with keyhole that has a very specific shape and doesn’t contain any locking mechanism (gears, pins, etc …) and corresponding keys. To open the chest and spend the bitcoin the user must bring their own locking mechanism and keys. Only once this is complete funds can be spent. Only exact locking mechanism configuration will work.

Rationale - From BIP 13

“This is one piece of the simplest path to a more secure bitcoin infrastructure. It is not intended to solve all of bitcoin's usability or security issues, but to be an incremental improvement over what exists today. A future BIP or BIPs should propose more user-friendly mechanisms for making payments, or for verifying that you're sending a payment to the Free Software Foundation and not Joe Random Hacker.

Assuming that typing in bitcoin addresses manually will become increasingly rare in the future, and given that the existing checksum method for bitcoin addresses seems to work "well enough" in practice and has already been implemented multiple times, the Author believes no change to the checksum algorithm is necessary.

The leading version bytes are chosen so that, after base58 encoding, the leading character is consistent: for the main network, byte 5 becomes the character '3'. For the testnet, byte 196 is encoded into '2'.”

BIP 13

LINKS

1. BIP-0013 Source
2. P2SH Explained Simply
3. Base58 Encoding Walkthrough
4. isStandard First Implementation