```
BIP: 13
Layer: Applications
Title: Address Format for pay-to-script-hash
Author: Gavin Andresen <gavinandresen@gmail.com>
Comments-Summary: No comments yet.
Comments-URI: https://github.com/bitcoin/bips/wiki/Comments:BIP-0013
Status: Final
Type: Standards Track
Created: 2011-10-18
```

Abstract

This BIP describes a new type of Bitcoin address to support arbitrarily complex transactions. Complexity in this context is defined as what information is needed by the recipient to respend the received coins, in contrast to needing a single ECDSA private key as in current implementations of Bitcoin.

In essence, an address encoded under this proposal represents the encoded hash of a script, rather than the encoded hash of an ECDSA public key.

Motivation

Enable "end-to-end" secure wallets and payments to fund escrow transactions or other complex transactions. Enable third-party wallet security services.

Specification

The new bitcoin address type is constructed in the same manner as existing bitcoin addresses (see Base58Check encoding):

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

Version byte is 5 for a main-network address, 196 for a testnet address. The 20-byte hash is the hash of the script that will be used to redeem the coins. And the 4-byte checksum is the first four bytes of the double SHA256 hash of the version and hash.

Rationale

One criticism is that bitcoin addresses should be deprecated in favor of a more user-friendly mechanism for payments, and that this will just encourage continued use of a poorly designed mechanism.

Another criticism is that bitcoin addresses are inherently insecure because there is no identity information tied to them; if you only have a bitcoin address, how can you be certain that you're paying who or what you think you're paying?

Furthermore, truncating SHA256 is not an optimal checksum; there are much better error-detecting algorithms. If we are introducing a new form of Bitcoin address, then perhaps a better algorithm should be used.

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'.

Backwards Compatibility

This proposal is not backwards compatible, but it fails gracefully-- if an older implementation is given one of these new bitcoin addresses, it will report the address as invalid and will refuse to create a transaction.

Reference Implementation

See base58.cpp/base58.h at https://github.com/bitcoin/bitcoin/tree/master/src

See Also

- BIP 12: OP_EVAL, the original P2SH design
- BIP 16: Pay to Script Hash (aka "/P2SH/")
- BIP 17: OP_CHECKHASHVERIFY, another P2SH design