```
BIP: 11
Layer: Applications
Title: M-of-N Standard Transactions
Author: Gavin Andresen <gavinandresen@gmail.com>
Comments-Summary: No comments yet.
Comments-URI: https://github.com/bitcoin/bips/wiki/Comments:BIP-0011
Status: Final
Type: Standards Track
Created: 2011-10-18
Post-History: 2011-10-02
```

Abstract

This BIP proposes M-of-N-signatures required transactions as a new 'standard' transaction type.

Motivation

Enable secured wallets, escrow transactions, and other use cases where redeeming funds requires more than a single signature.

A couple of motivating use cases:

- A wallet secured by a "wallet protection service" (WPS). 2-of-2 signatures required transactions will be used, with one signature coming from the (possibly compromised) computer with the wallet and the second signature coming from the WPS. When sending protected bitcoins, the user's bitcoin client will contact the WPS with the proposed transaction and it can then contact the user for confirmation that they initiated the transaction and that the transaction details are correct. Details for how clients and WPS's communicate are outside the scope of this BIP. Side note: customers should insist that their wallet protection service provide them with copies of the private key(s) used to secure their wallets that they can safely store off-line, so that their coins can be spent even if the WPS goes out of business.
- Three-party escrow (buyer, seller, and trusted dispute agent). 2-of-3 signatures required transactions will be used. The buyer and seller and agent will each provide a public key, and the buyer will then send coins into a 2-of-3 CHECKMULTISIG transaction and send the seller and the agent the transaction id. The seller will fulfill their obligation and then ask the buyer to co-sign a transaction (already signed by seller) that sends the tied-up coins to him (seller).
  If the buyer and seller cannot agree, then the agent can, with the cooperation of either buyer or seller, decide what happens to the tied-up coins. Details of how buyer, seller, and agent communicate to gather signatures or public keys are outside the scope of this BIP.

Specification

A new standard transaction type (scriptPubKey) that is relayed by clients and included in mined blocks:

```
m {pubkey}...{pubkey} n OP_CHECKMULTISIG
```

But only for n less than or equal to 3.

OP_CHECKMULTISIG transactions are redeemed using a standard scriptSig:

```
OP_0 ...signatures...
```

(OP_0 is required because of a bug in OP_CHECKMULTISIG; it pops one too many items off the execution stack, so a dummy value must be placed on the stack).

The current Satoshi bitcoin client does not relay or mine transactions with scriptSigs larger than 200 bytes; to accommodate 3-signature transactions, this will be increased to 500 bytes.

Rationale

OP_CHECKMULTISIG is already an enabled opcode, and is the most straightforward way to support several important use cases.

One argument against using OP_CHECKMULTISIG is that old clients and miners count it as "20 sigops" for purposes of computing how many signature operations are in a block, and there is a hard limit of 20,000 sigops per block--meaning a maximum of 1,000 multisig transactions per block. Creating multisig transactions using multiple OP_CHECKSIG operations allows more of them per block.

The counter-argument is that these new multi-signature transactions will be used in combination with OP_EVAL (see the OP_EVAL BIP), and **will** be counted accurately. And in any case, as transaction volume rises the hard-coded maximum block size will have to be addressed, and the rules for counting number-of-signature-operations-in-a-block can be addressed at that time.

A weaker argument is OP_CHECKMULTISIG should not be used because it pops one too many items off the stack during validation. Adding an extra OP_0 placeholder to the scriptSig adds only 1 byte to the transaction, and any alternative that avoids OP_CHECKMULTISIG adds at least several bytes of opcodes.

Implementation

OP_CHECKMULTISIG is already supported by old clients and miners as a non-standard transaction type.

https://github.com/gavinandresen/bitcoin-git/tree/77f21f1583deb89bf3fffe80fe9b181fedb1dd60

Post History

- OP_EVAL proposal