```
BIP: 60
Layer: Peer Services
Title: Fixed Length "version" Message (Relay-Transactions Field)
Author: Amir Taaki <genjix@riseup.net>
Comments-Summary: Discouraged for implementation (one person)
Comments-URI: https://github.com/bitcoin/bips/wiki/Comments:BIP-0060
Status: Draft
Type: Standards Track
Created: 2013-06-16
License: PD
```

Abstract

BIP 0037 introduced a new flag to version messages which says whether to relay new transaction messages to that node.

The protocol version was upgraded to 70001, and the (now accepted) BIP 0037 became implemented.

The implementation is problematic because the RelayTransactions flag is an optional part of the version message stream.

Motivation

One property of Bitcoin messages is their fixed number of fields. This keeps the format simple and easily understood. Adding optional fields to messages will cause deserialisation issues when other fields come after the optional one.

As an example, the length of version messages might be checked to ensure the byte stream is consistent. With optional fields, this checking is no longer possible. This is desirable to check for consistency inside internal deserialization code, and proper formatting of version messages originating from other nodes. In the future with diversification of the Bitcoin network, it will become desirable to enforce this kind of strict adherance to standard messages with field length compliance with every protocol version.

Another property of fixed-length field messages is the ability to pass stream operators around for deserialization. This property is also lost, as now the deserialisation code must know the remaining length of bytes to parse. The parser now requires an additional piece of information (remaining size of the stream) for parsing instead of being a dumb reader.

Specification

version

When a node creates an outgoing connection, it will immediately advertise its version. The remote node will respond with its version. No futher communication is possible until both peers have exchanged their version.

Payload:

Field Size	Description	Data type	Comments
4	version	int32_t	Identifies protocol version being used by the node
8	services	uint64_t	bitfield of features to be enabled for this connection
8	timestamp	int64_t	standard UNIX timestamp in seconds
26	addr_recv	net_addr	The network address of the node receiving this message
version >= 106			
26	addr_from	net_addr	The network address of the node emitting this message
8	nonce	uint64_t	Node random nonce, randomly generated every time a version packet
?	user_agent	var_str	User Agent (0x00 if string is 0 bytes long)
4	start_height	int32_t	The last block received by the emitting node
1	relay	bool	Whether the remote peer should announce relayed transactions or no

A "verack" packet shall be sent if the version packet was accepted.

The following services are currently assigned:

Value	Name	Description
1	NODE_NETWORK	This node can be asked for full blocks instead of just headers.

Code Updates

fRelayTx is added to the PushMessage() call inside PushVersion() (net.cpp)

```
void CNode::PushVersion()
{
    /// when NTP implemented, change to just nTime = GetAdjustedTime()
    int64 nTime = (fInbound ? GetAdjustedTime() : GetTime());
    CAddress addrYou = (addr.IsRoutable() && !IsProxy(addr) ? addr : CAddress(CService("0.0.
    CAddress addrMe = GetLocalAddress(&addr);
    RAND_bytes((unsigned char*)&nLocalHostNonce, sizeof(nLocalHostNonce));
    printf("send version message: version %d, blocks=%d, us=%s, them=%s, peer=%s\n", PROTOCO
    PushMessage("version", PROTOCOL_VERSION, nLocalServices, nTime, addrYou, addrMe,
                nLocalHostNonce, FormatSubVersion(CLIENT_NAME, CLIENT_VERSION, std::vector<s
                nBestHeight, true);
}
```

Additionally the protocol version is increased from 70001 to 70002.

Copyright

This document is placed in the public domain.