

Bitcoin Core version 0.13.1 is now available from:

<https://bitcoin.org/bin/bitcoin-core-0.13.1/>

This is a new minor version release, including activation parameters for the segwit softfork, various bugfixes and performance improvements, as well as updated translations.

Please report bugs using the issue tracker at github:

<https://github.com/bitcoin/bitcoin/issues>

To receive security and update notifications, please subscribe to:

<https://bitcoincore.org/en/list/announcements/join/>

## Compatibility

Microsoft ended support for Windows XP on April 8th, 2014, an OS initially released in 2001. This means that not even critical security updates will be released anymore. Without security updates, using a bitcoin wallet on a XP machine is irresponsible at least.

In addition to that, with 0.12.x there have been varied reports of Bitcoin Core randomly crashing on Windows XP. It is not clear what the source of these crashes is, but it is likely that upstream libraries such as Qt are no longer being tested on XP.

We do not have time nor resources to provide support for an OS that is end-of-life. From 0.13.0 on, Windows XP is no longer supported. Users are suggested to upgrade to a newer version of Windows, or install an alternative OS that is supported.

No attempt is made to prevent installing or running the software on Windows XP, you can still do so at your own risk, but do not expect it to work: do not report issues about Windows XP to the issue tracker.

From 0.13.1 onwards OS X 10.7 is no longer supported. 0.13.0 was intended to work on 10.7+, but severe issues with the libc++ version on 10.7.x keep it from running reliably. 0.13.1 now requires 10.8+, and will communicate that to 10.7 users, rather than crashing unexpectedly.

## Notable changes

### Segregated witness soft fork

Segregated witness (segwit) is a soft fork that, if activated, will allow transaction-producing software to separate (segregate) transaction signatures (witnesses) from the part of the data in a transaction that is covered by the txid. This provides several immediate benefits:

- **Elimination of unwanted transaction malleability:** Segregating the witness allows both existing and upgraded software to calculate the transaction identifier (txid) of transactions without referencing the witness, which can sometimes be changed by third-parties (such as miners) or by co-signers in a multisig spend. This solves all known cases of unwanted transaction malleability, which is a problem that makes programming Bitcoin wallet software more difficult and which seriously complicates the design of smart contracts for Bitcoin.
- **Capacity increase:** Segwit transactions contain new fields that are not part of the data currently used to calculate the size of a block, which allows a block containing segwit transactions to hold more data than allowed by the current maximum block size. Estimates based on the transactions currently found in blocks indicate that if all wallets switch to using segwit, the network will be able to support about 70% more transactions. The network will also be able to support more of the advanced-style payments (such as multisig) than it can support now because of the different weighting given to different parts of a transaction after segwit activates (see the following section for details).
- **Weighting data based on how it affects node performance:** Some parts of each Bitcoin block need to be stored by nodes in order to validate future blocks; other parts of a block can be immediately forgotten (pruned) or used only for helping other nodes sync their copy of the block chain. One large part of the immediately prunable data are transaction signatures (witnesses), and segwit makes it possible to give a different “weight” to segregated witnesses to correspond with the lower demands they place on node resources. Specifically, each byte of a segregated witness is given a weight of 1, each other byte in a block is given a weight of 4, and the maximum allowed weight of a block is 4 million. Weighting the data this way better aligns the most profitable strategy for creating blocks with the long-term costs of block validation.
- **Signature covers value:** A simple improvement in the way signatures are generated in segwit simplifies the design of secure signature generators (such as hardware wallets), reduces the amount of data the signature generator needs to download, and allows the signature generator to operate more quickly. This is made possible by having the generator sign the amount of bitcoins they think they are spending, and by having full nodes refuse to accept those signatures unless the amount of bitcoins being spent is exactly the same as was signed. For non-segwit transactions, wallets instead had to download the complete previous transactions being spent for every payment they made, which could be a slow operation on hardware wallets and in other situations where bandwidth or computation speed was constrained.
- **Linear scaling of sighash operations:** In 2015 a block was produced that required about 25 seconds to validate on modern hardware because of

the way transaction signature hashes are performed. Other similar blocks, or blocks that could take even longer to validate, can still be produced today. The problem that caused this can't be fixed in a soft fork without unwanted side-effects, but transactions that opt-in to using segwit will now use a different signature method that doesn't suffer from this problem and doesn't have any unwanted side-effects.

- **Increased security for multisig:** Bitcoin addresses (both P2PKH addresses that start with a '1' and P2SH addresses that start with a '3') use a hash function known as RIPEMD-160. For P2PKH addresses, this provides about 160 bits of security—which is beyond what cryptographers believe can be broken today. But because P2SH is more flexible, only about 80 bits of security is provided per address. Although 80 bits is very strong security, it is within the realm of possibility that it can be broken by a powerful adversary. Segwit allows advanced transactions to use the SHA256 hash function instead, which provides about 128 bits of security (that is 281 trillion times as much security as 80 bits and is equivalent to the maximum bits of security believed to be provided by Bitcoin's choice of parameters for its Elliptic Curve Digital Security Algorithm [ECDSA].)
- **More efficient almost-full-node security** Satoshi Nakamoto's original Bitcoin paper describes a method for allowing newly-started full nodes to skip downloading and validating some data from historic blocks that are protected by large amounts of proof of work. Unfortunately, Nakamoto's method can't guarantee that a newly-started node using this method will produce an accurate copy of Bitcoin's current ledger (called the UTXO set), making the node vulnerable to falling out of consensus with other nodes. Although the problems with Nakamoto's method can't be fixed in a soft fork, Segwit accomplishes something similar to his original proposal: it makes it possible for a node to optionally skip downloading some blockchain data (specifically, the segregated witnesses) while still ensuring that the node can build an accurate copy of the UTXO set for the block chain with the most proof of work. Segwit enables this capability at the consensus layer, but note that Bitcoin Core does not provide an option to use this capability as of this 0.13.1 release.
- **Script versioning:** Segwit makes it easy for future soft forks to allow Bitcoin users to individually opt-in to almost any change in the Bitcoin Script language when those users receive new transactions. Features currently being researched by Bitcoin Core contributors that may use this capability include support for Schnorr signatures, which can improve the privacy and efficiency of multisig transactions (or transactions with multiple inputs), and Merklized Abstract Syntax Trees (MAST), which can improve the privacy and efficiency of scripts with two or more conditions. Other Bitcoin community members are studying several other improvements that can be made using script versioning.

Activation for the segwit soft fork is being managed using BIP9 versionbits.

Segwit's version bit is bit 1, and nodes will begin tracking which blocks signal support for segwit at the beginning of the first retarget period after segwit's start date of 15 November 2016. If 95% of blocks within a 2,016-block retarget period (about two weeks) signal support for segwit, the soft fork will be locked in. After another 2,016 blocks, segwit will activate.

For more information about segwit, please see the segwit FAQ, the segwit wallet developers guide or BIPs 141, 143, 144, and 145. If you're a miner or mining pool operator, please see the versionbits FAQ for information about signaling support for a soft fork.

## Null dummy soft fork

Combined with the segwit soft fork is an additional change that turns a long-existing network relay policy into a consensus rule. The `OP_CHECKMULTISIG` and `OP_CHECKMULTISIGVERIFY` opcodes consume an extra stack element ("dummy element") after signature validation. The dummy element is not inspected in any manner, and could be replaced by any value without invalidating the script.

Because any value can be used for this dummy element, it's possible for a third-party to insert data into other people's transactions, changing the transaction's txid (called transaction malleability) and possibly causing other problems.

Since Bitcoin Core 0.10.0, nodes have defaulted to only relaying and mining transactions whose dummy element was a null value (0x00, also called `OP_0`). The null dummy soft fork turns this relay rule into a consensus rule both for non-segwit transactions and segwit transactions, so that this method of mutating transactions is permanently eliminated from the network.

Signaling for the null dummy soft fork is done by signaling support for segwit, and the null dummy soft fork will activate at the same time as segwit.

For more information, please see BIP147.

## Low-level RPC changes

- `importprunedfunds` only accepts two required arguments. Some versions accept an optional third arg, which was always ignored. Make sure to never pass more than two arguments.

## Linux ARM builds

With the 0.13.0 release, pre-built Linux ARM binaries were added to the set of uploaded executables. Additional detail on the ARM architecture targeted by each is provided below.

The following extra files can be found in the download directory or torrent:

- `bitcoin-${VERSION}-arm-linux-gnueabi.tar.gz`: Linux binaries targeting the 32-bit ARMv7-A architecture.

- `bitcoin-${VERSION}-aarch64-linux-gnu.tar.gz`: Linux binaries targeting the 64-bit ARMv8-A architecture.

ARM builds are still experimental. If you have problems on a certain device or Linux distribution combination please report them on the bug tracker, it may be possible to resolve them. Note that the device you use must be (backward) compatible with the architecture targeted by the binary that you use. For example, a Raspberry Pi 2 Model B or Raspberry Pi 3 Model B (in its 32-bit execution state) device, can run the 32-bit ARMv7-A targeted binary. However, no model of Raspberry Pi 1 device can run either binary because they are all ARMv6 architecture devices that are not compatible with ARMv7-A or ARMv8-A.

Note that Android is not considered ARM Linux in this context. The executables are not expected to work out of the box on Android.

### 0.13.1 Change log

Detailed release notes follow. This overview includes changes that affect behavior, not code moves, refactors and string updates. For convenience in locating the code changes and accompanying discussion, both the pull request and git merge commit are mentioned.

#### Consensus

- #8636 9dfa0c8 Implement NULLDUMMY softfork (BIP147) (jl2012)
- #8848 7a34a46 Add NULLDUMMY verify flag in bitcoinconsensus.h (jl2012)
- #8937 8b66659 Define start and end time for segwit deployment (sipa)

#### RPC and other APIs

- #8581 526d2b0 Drop misleading option in importprunedfunds (MarcoFalke)
- #8699 a5ec248 Remove createwitnessaddress RPC command (jl2012)
- #8780 794b007 Deprecate getinfo (MarcoFalke)
- #8832 83ad563 Throw JSONRPCError when utxo set can not be read (MarcoFalke)
- #8884 b987348 getblockchaininfo help: pruneheight is the lowest, not highest, block (luke-jr)
- #8858 3f508ed rpc: Generate auth cookie in hex instead of base64 (laanwj)
- #8951 7c2bf4b RPC/Mining: getblocktemplate: Update and fix formatting of help (luke-jr)

#### Block and transaction handling

- #8611 a9429ca Reduce default number of blocks to check at startup (sipa)

- #8634 3e80ab7 Add policy: null signature for failed CHECK(MULTI)SIG (jl2012)
- #8525 1672225 Do not store witness txn in rejection cache (sipa)
- #8499 9777fe1 Add several policy limits and disable uncompressed keys for segwit scripts (jl2012)
- #8526 0027672 Make non-minimal OP\_IF/NOTIF argument non-standard for P2WSH (jl2012)
- #8524 b8c79a0 Precompute sighashes (sipa)
- #8651 b8c79a0 Predeclare PrecomputedTransactionData as struct (sipa)

## P2P protocol and network code

- #8740 42ea51a No longer send local address in addrMe (laanwj)
- #8427 69d1cd2 Ignore notfound P2P messages (laanwj)
- #8573 4f84082 Set jonasschnellis dns-seeder filter flag (jonasschnelli)
- #8712 23feab1 Remove maxuploadtargets recommended minimum (jonasschnelli)
- #8862 7ae6242 Fix a few cases where messages were sent after requested disconnect (theuni)
- #8393 fe1975a Support for compact blocks together with segwit (sipa)
- #8282 2611ad7 Feeler connections to increase online addrs in the tried table (EthanHeilman)
- #8612 2215c22 Check for compatibility with download in FindNextBlocksToDownload (sipa)
- #8606 bbf379b Fix some locks (sipa)
- #8594 ab295bb Do not add random inbound peers to addrman (gmaxwell)
- #8940 5b4192b Add x9 service bit support to dnsseed.bluematt.me, seed.bitcoinstats.com (TheBlueMatt, cdecker)
- #8944 685e4c7 Remove bogus assert on number of outbound connections. (TheBlueMatt)
- #8949 0dbc48a Be more aggressive in getting connections to peers with relevant services (gmaxwell)

## Build system

- #8293 fa5b249 Allow building libbitcoinconsensus without any univalue (luke-jr)
- #8492 8b0bdd3 Allow building bench\_bitcoin by itself (luke-jr)
- #8563 147003c Add configure check for -latomic (ajtowns)
- #8626 ea51b0f Berkeley DB v6 compatibility fix (netsafe)
- #8520 75f2065 Remove check for openssl/ec.h (laanwj)

## GUI

- #8481 d9f0d4e Fix minimize and close bugs (adlawren)
- #8487 a37cec5 Persist the datadir after option reset (achow101)
- #8697 41fd852 Fix op order to append first alert (rodasmith)

- #8678 8e03382 Fix UI bug that could result in paying unexpected fee (jonasschnelli)
- #8911 7634d8e Translate all files, even if wallet disabled (laanwj)
- #8540 1db3352 Fix random segfault when closing “Choose data directory” dialog (laanwj)
- #7579 f1c0d78 Show network/chain errors in the GUI (jonasschnelli)

## Wallet

- #8443 464dedd Trivial cleanup of HD wallet changes (jonasschnelli)
- #8539 cb07f19 CDB: fix debug output (crowning-)
- #8664 091cdeb Fix segwit-related wallet bug (sdaftuar)
- #8693 c6a6291 Add witness address to address book (instagibbs)
- #8765 6288659 Remove “unused” ThreadFlushWalletDB from removeprunedfunds (jonasschnelli)

## Tests and QA

- #8713 ae8c7df create\_cache: Delete temp dir when done (MarcoFalke)
- #8716 e34374e Check legacy wallet as well (MarcoFalke)
- #8750 d6ebe13 Refactor RPCTestHandler to prevent TimeoutExpired (MarcoFalke)
- #8652 63462c2 remove root test directory for RPC tests (yurizhykin)
- #8724 da94272 walletbackup: Sync blocks inside the loop (MarcoFalke)
- #8400 bea02dc enable rpcbind\_test (yurizhykin)
- #8417 f70be14 Add walletdump RPC test (including HD- & encryption-tests) (jonasschnelli)
- #8419 a7aa3cc Enable size accounting in mining unit tests (sdaftuar)
- #8442 8bb1efd Rework hd wallet dump test (MarcoFalke)
- #8528 3606b6b Update p2p-segwit.py to reflect correct behavior (instagibbs)
- #8531 a27cdd8 abandonconflict: Use assert\_equal (MarcoFalke)
- #8667 6b07362 Fix SIGHASH\_SINGLE bug in test\_framework SignatureHash (jl2012)
- #8673 03b0196 Fix obvious assignment/equality error in test (JeremyRubin)
- #8739 cef633c Fix broken sendcmpct test in p2p-compactblocks.py (sdaftuar)
- #8418 ff893aa Add tests for compact blocks (sdaftuar)
- #8803 375437c Ping regularly in p2p-segwit.py to keep connection alive (jl2012)
- #8827 9bbe66e Split up slow RPC calls to avoid pruning test timeouts (sdaftuar)
- #8829 2a8bca4 Add bitcoin-tx JSON tests (jnewbery)
- #8834 1dd1783 blockstore: Switch to dumb dbm (MarcoFalke)
- #8835 d87227d nulldummy.py: Don't run unused code (MarcoFalke)

- #8836 eb18cc1 bitcoin-util-test.py should fail if the output file is empty (jnewbery)
- #8839 31ab2f8 Avoid ConnectionResetErrors during RPC tests (laanwj)
- #8840 cbc3fe5 Explicitly set encoding to utf8 when opening text files (laanwj)
- #8841 3e4abb5 Fix nulldummy test (jl2012)
- #8854 624a007 Fix race condition in p2p-compactblocks test (sdaftuar)
- #8857 1f60d45 mininode: Only allow named args in wait\_until (MarcoFalke)
- #8860 0bee740 util: Move wait\_bitcoins() into stop\_nodes() (MarcoFalke)
- #8882 b73f065 Fix race conditions in p2p-compactblocks.py and send\_headers.py (sdaftuar)
- #8904 cc6f551 Fix compact block shortids for a test case (dagurval)

## Documentation

- #8754 0e2c6bd Target protobuf 2.6 in OS X build notes. (fanquake)
- #8461 b17a3f9 Document return value of networkhashps for getmininginfo RPC endpoint (jlopp)
- #8512 156e305 Corrected JSON typo on setban of net.cpp (sevastos)
- #8683 8a7d7ff Fix incorrect file name bitcoin.qrc (bitcoinsSG)
- #8891 5e0dd9e Update bips.md for Segregated Witness (fanquake)
- #8545 863ae74 Update git-subtree-check.sh README (MarcoFalke)
- #8607 486650a Fix doxygen off-by-one comments, fix typos (MarcoFalke)
- #8560 c493f43 Fix two VarInt examples in serialize.h (cbarcenas)
- #8737 084cae9 UndoReadFromDisk works on undo files (rev), not on block files (paveljanik)
- #8625 0a35573 Clarify statement about parallel jobs in rpc-tests.py (isle2983)
- #8624 0e6d753 build: Mention curl (MarcoFalke)
- #8604 b09e13c build,doc: Update for 0.13.0+ and OpenBSD 5.9 (laanwj)
- #8939 06d15fb Update implemented bips for 0.13.1 (sipa)

## Miscellaneous

- #8742 d31ac72 Specify Protobuf version 2 in paymentrequest.proto (fanquake)
- #8414,#8558,#8676,#8700,#8701,#8702 Add missing copyright headers (isle2983, kazcw)
- #8899 4ed2627 Fix wake from sleep issue with Boost 1.59.0 (fanquake)
- #8817 bcf3806 update bitcoin-tx to output witness data (jnewbery)
- #8513 4e5fc31 Fix a type error that would not compile on OSX. (JeremyRubin)
- #8392 30eac2d Fix several node initialization issues (sipa)
- #8548 305d8ac Use \_\_func\_\_ to get function name for output printing



(MarcoFalke)

- #8291 a987431 [util] CopyrightHolders: Check for untranslated substitution (MarcoFalke)

## Credits

Thanks to everyone who directly contributed to this release:

- adlawren
- Alexey Vesnin
- Anders Øyvind Urke-Sætre
- Andrew Chow
- Anthony Towns
- BtcDrak
- Chris Stewart
- Christian Barcenás
- Christian Decker
- Cory Fields
- crowning-
- Dagur Valberg Johannsson
- David A. Harding
- Eric Lombrozo
- Ethan Heilman
- fanquake
- Gaurav Rana
- Gregory Maxwell
- instagibbs
- isle2983
- Jameson Lopp
- Jeremy Rubin
- jnewbery
- Johnson Lau
- Jonas Schnelli
- jonnynewbs
- Justin Camarena
- Kaz Wesley
- leijurv
- Luke Dashjr
- MarcoFalke
- Marty Jones
- Matt Corallo
- Micha
- Michael Ford
- mruddy
- Pavel Janík

- Pieter Wuille
- rodasmith
- Sev
- Suhas Daftuar
- whythat
- Wladimir J. van der Laan

As well as everyone that helped translating on Transifex.