```
BIP: 113
Layer: Consensus (soft fork)
Title: Median time-past as endpoint for lock-time calculations
Author: Thomas Kerin <me@thomaskerin.io>
        Mark Friedenbach <mark@friedenbach.org>
Comments-Summary: No comments yet.
Comments-URI: https://github.com/bitcoin/bips/wiki/Comments:BIP-0113
Status: Final
Type: Standards Track
Created: 2015-08-10
License: PD
```

Abstract

This BIP is a proposal to redefine the semantics used in determining a time-locked transaction's eligibility for inclusion in a block. The median of the last 11 blocks is used instead of the block's timestamp, ensuring that it increases monotonically with each block.

Motivation

At present, transactions are excluded from inclusion in a block if the present time or block height is less than or equal to that specified in the locktime. Since the consensus rules do not mandate strict ordering of block timestamps, this has the unfortunate outcome of creating a perverse incentive for miners to lie about the time of their blocks in order to collect more fees by including transactions that by wall clock determination have not yet matured.

This BIP proposes comparing the locktime against the median of the past 11 block's timestamps, rather than the timestamp of the block including the transaction. Existing consensus rules guarantee this value to monotonically advance, thereby removing the capability for miners to claim more transaction fees by lying about the timestamps of their block.

This proposal seeks to ensure reliable behaviour in locktime calculations as required by BIP65 (CHECKLOCKTIMEVERIFY) and matching the behavior of BIP68 (sequence numbers) and BIP112 (CHECKSEQUENCEVERIFY).

Specification

The values for transaction locktime remain unchanged. The difference is only in the calculation determining whether a transaction can be included. Instead of an unreliable timestamp, the following function is used to determine the current block time for the purpose of checking lock-time constraints:

```
enum { nMedianTimeSpan=11 };
```

```
int64_t GetMedianTimePast(const CBlockIndex* pindex)
{
int64_t pmedian[nMedianTimeSpan];
int64_t* pbegin = &pmedian[nMedianTimeSpan];
int64_t* pend = &pmedian[nMedianTimeSpan];
for (int i = 0; i < nMedianTimeSpan && pindex; i++, pindex = pindex->pprev)
*(--pbegin) = pindex->GetBlockTime();
std::sort(pbegin, pend);
return pbegin[(pend - pbegin)/2];
}
```

Lock-time constraints are checked by the consensus method IsFinalTx(). This method takes the block time as one parameter. This BIP proposes that after activation calls to IsFinalTx() within consensus code use the return value of 'GetMedianTimePast(pindexPrev)' instead.

The new rule applies to all transactions, including the coinbase transaction.

A reference implementation of this proposal is provided by the following pull request:

https://github.com/bitcoin/bitcoin/pull/6566

Deployment

This BIP is to be deployed by "versionbits" BIP9 using bit 0.

For Bitcoin **mainnet**, the BIP9 **starttime** will be midnight 1st May 2016 UTC (Epoch timestamp 1462060800) and BIP9 **timeout** will be midnight 1st May 2017 UTC (Epoch timestamp 1493596800).

For Bitcoin **testnet**, the BIP9 **starttime** will be midnight 1st March 2016 UTC (Epoch timestamp 1456790400) and BIP9 **timeout** will be midnight 1st May 2017 UTC (Epoch timestamp 1493596800).

This BIP must be deployed simultaneously with BIP68 and BIP112 using the same deployment mechanism.

Acknowledgements

Mark Friedenbach for designing and authoring the reference implementation of this BIP.

Thanks go to Gregory Maxwell who came up with the original idea, in #bitcoin-wizards on 2013-07-16.

Thomas Kerin authored this BIP document.

Compatibility

Transactions generated using time-based lock-time will take approximately an hour longer to confirm than would be expected under the old rules. This is not known to introduce any compatibility concerns with existing protocols.

References

BIP9: Versionbits

BIP65: OP_CHECKLOCKTIMEVERIFY

BIP68: Consensus-enforced transaction replacement signaled via sequence numbers

BIP112: CHECKSEQUENCEVERIFY

Softfork deployment considerations

Version bits

Copyright