Analysis of the SegWit adoption in Bitcoin

C. Pérez-Solà, S. Delgado-Segura, J. Herrera-Joancomartí, G. Navarro-Arribas
Dep. of Information and Communications Engineering,
Universitat Autònoma de Barcelona

***Abstract*—Segregated witnesses (SegWit) is the most controversial modification performed in the Bitcoin protocol so far. Although it was a soft fork modification with backward compatibility, some miners reluctance to accept the improvement let finally to a fork in the Bitcoin blockchain that produced the new Bitcoin Cash cryptocurrency. In this paper, we describe the segregated witness upgrade providing information about its motivation and improvements in terms of security (malleability) and scalability (payment channels). Furthermore, we provide some data on its real adoption on the Bitcoin ecosystem.**

***Index Terms*—Bitcoin, cryptocurrency, segregated witness**

I. Introduction

Recently, the increase of both the popularity and the use of Bitcoin has shown its bounds regarding the ability of the system to scale with the number of users. It is obvious that a system with a unique (although replicated) register containing all system transactions (i.e. the blockchain) may present a bottleneck.

There are several ways of measuring scalability, as pointed out in [1]. From latency (the time for a transaction to be confirmed) to bootstrap time (the time it takes a new node to download and process the history necessary to validate payments) through cost per confirmed transactions, different measures can be used to evaluate the efficiency of a payment system. Croman et al. [1] gave approximations of all of these metrics for the Bitcoin network. However, probably the easiest metric to compare Bitcoin with existing global payment systems and the one that has a direct impact on scalability is the system transaction throughput. The transaction throughput can be measured by the maximum number of transactions per second that a system can deal with. Throughput is often chosen to evaluate systems because it is objective, easy to compute, and gives a metric to easily compare different payment systems. For instance, Visa reported to allow around 2,000 transactions per second in normal situation [1] while reaching a peak of 56,000 in a stress test [2]. Paypal manages lower values, providing 136 transactions per second on average on his payment network.[1]

Regarding Bitcoin, its throughput can be measured considering different parameters, from network communication latency to processing power of the nodes performing transaction validation. However, what limits the throughput of Bitcoin the most is the block size limit, which is currently fixed at 1MB.[2]

Limiting the block size to 1MB implies a maximum throughput of 7 transactions per second [1]. Such limit is an approximation obtained by dividing the maximum block size by the average size of Bitcoin transactions (250 bytes) and the inter-block time (10 minutes). Therefore, a block may contain, at most, $1{,}000{,}000/250 = 100$ average sized transactions, thus giving a throughput of $100/600 = 6.6$ transactions per second. Notice that such value is far behind the throughput that other payment systems, like Visa or PayPal, can deal with.

The rest of the paper is organized as follows. Section II reviews the solutions that were proposed to solve the Bitcoin scalability problem. Then, Section III presents SegWit, the solution that was finally deployed. After that, Section IV analyses the adoption of SegWit since its deployment up until today. Finally, Section V presents the conclusions.

II. Bitcoin scaling proposals

An increase on the adoption and use of the system made Bitcoin face a state in which blocks became full. Full blocks implied a longer confirmation time, since the network produced more transactions than blocks were able to handle. The lack of space in blocks also affected transaction fees, making them increase, and therefore making the currency less usable. In order to deal with the scaling problem several solutions where proposed towards dealing with the main bottleneck, the block size limit.

Some proposals suggested to increase the limit following different strategies or even propose to remove the limit. Jeff Garzik's BIP 100 [4] proposed to change the 1MB fixed limit to a new floating block size limit, where miners may increase the block size by consensus. Gavin Andresen BIP 101 [5] proposal (currently withdrawn) consisted in initially increasing block size to 8MB and doubling the size every two years for 20 years, after which the block size remains fixed. Jeff Garzik's BIP 102 [6] proposed to simply increase block size to 2MB. Pieter Wuille's BIP 103 [7] proposed to increase the maximum block size by 4.4% every 97 days until 2063, implying a 17.7% block size increase per year. Gavin Andresen's BIP 109 [8] proposed a fixed block size increase to 2MB with a new limit on the amount of data that can be hashed to compute signature hashes and a change on how the maximum number of signatures is counted. However, increasing the block size implies a change on the consensus rules of the currency, that have to be deployed via a hard fork of the protocol.

A. *Protocol forks*

Modification proposals in the Bitcoin protocol, even those of utter importance like the ones affecting the scalability of the system, are often difficult to tackle since they have to be deployed with extreme caution and maximum consensus. Furthermore, if changes affect the consensus mechanisms of the protocol, their implications may cause a blockchain fork and that could have a big impact in the cryptocurrency. Moreover, the collateral implications of changes need to be also

[1] PayPal Q1 2016 Results [3] reported handling 1.41B payment transactions, which leads to an estimated $1.41B/4/30/24/60/60 = 136$ transactions per second.

[2] https://github.com/bitcoin/bitcoin/blob/a6a860796a44a2805a58391a009ba22752f64e32/src/consensus/consensus.h#L9

considered beforehand to prevent unexpected consequences, specially those related to security and decentralization.

Changes in the Bitcoin consensus rules may be introduced by soft (protocol) forks or hard (protocol) forks. A soft fork is produced when the protocol rules are changed but they remain backward-compatible with the old rules, making the old nodes able to accept blocks produced by new nodes. On the contrary, hard forks make a change in the protocol that changes its behavior in some sense, and make old nodes not able to follow the protocol anymore. When a soft fork occurs, only the majority of miners have to upgrade to the new version, since old nodes can accept newly generated blocks. However, when a hard fork occurs, all nodes that want to follow the new protocol rules are forced to upgrade to the new version.

As already pointed out, increasing the block size will imply a hard fork. However, it is unclear whether an increase in the block size by its own will solve future scalability problems. In some sense arbitrarily increasing the block size is like making hard drives twice as big to deal with storage problems, instead of designing smarter hard drives able to store more information within the same physical space.

B. *Off-chain Payment Channels*

An alternative solution to deal with the Bitcoin scaling problem are off-chain payment channels, that intend to move most of the transactions performed in the system outside of the blockchain, reducing the number of transactions to be included in blocks and therefore indirectly increasing the throughput.

The first proposal of such a mechanism targeted unidirectional payments (payments from a payer to a payee). Its main goal was to avoid the fees that transactions in the blockchain imply and that are not affordable for micropayment transactions [9]. To set up the payment channel, a transaction is included in the blockchain as a deposit of the money that will be used in the payment channel. A refund transaction is also created, allowing the payer to recover the deposited funds if the payee does not cooperate. The refund transaction can not be included in the blockchain until a certain point in the future, and thus the channel may remain open until that moment is reached. Between the set up and the closing of the payment channel, the payer can perform multiple payments to the payee through transactions that, although build following the Bitcoin transaction format, would be transferred privately between A and B without using the Bitcoin network. Furthermore, the individual payment transactions will not appear in the blockchain: only the set up transaction that opens the channel and the last transaction that closes the channel will be broadcast through the Bitcoin P2P network and therefore included in a block.

A more recent proposal, which allows bidirectional payments between parties, was proposed by Poon and Dryja [10]: the Lightning network[3].

However, both the basic off-chain protocol and the Lightning network faced an issue with the Bitcoin version at the time, the transaction malleability problem.

[3]The Lightning network is an extensive topic to talk about, and will not fit within the paper page limit. Interested readers should refer to the original paper.

C. *Transaction Malleability Problem*

Bitcoin transactions are identified by their transaction id, a value computed using a double SHA256 function over the raw data that defines the transaction. Each identifier should uniquely identify a transaction, and each transaction should have a unique identifier. However, due to how Bitcoin transactions are formatted the previous statement does not always hold. Some modifications can be performed over a signed transaction that will keep the origin and destination of payment while completely modifying the transaction id. Therefore, a transaction id is not final until the transaction can not be modified anymore, that is, once is has been included in a block. In scenarios like off-chain payment channels, were some transactions are created referring to transactions that are not final, transaction malleability poses a risk in the system. It is important to mention that malleability does not affect the ability of an attacker to spend/steal the bitcoins from an already created transaction, since he can not forge the digital signature of the owner. For that reason, although transaction malleability is known back from 2011, it has never been considered as a security issue.

III. Segregated Witness

Segregated Witness (SegWit) [11] was proposed as a solution for the malleability problem, by redesigning the transaction structure to compute transaction identifiers without counting the signatures. Such modification was proposed as a soft fork of the protocol, that in addition, virtually increased the block size up to four times the limit at the time (4 MB). Moreover, by deploying SegWit, off-chain payment channels could build on top of Bitcoin, increasing the throughput virtually to no limit.

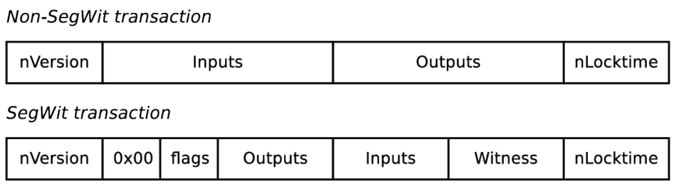

Fig. 1. Simplified non-SegWit and SegWit transaction.

The main idea behind SegWit is to detach from the transaction the information needed to validate its correctness while keeping the information related to its effects. This information comprises scripts and signatures, which are then placed in a new structure called *witness* (see Figure 1). The transaction signature includes only the fixed information making the id unique. Figure 2 show the (simplified) fields used to compute the transaction id (*tx_id*) in SegWit and non-SegWit transactions.

The witness has to be included in the block where the transaction belongs. To do that, an `OP_RETURN` output of the generation transaction of the block includes the root of a Merkle tree of the witnesses of the transactions.

As we will see in the next section, there are several types SegWit scripts. Most notably SegWit can be used *natively* or encapsulated in P2SH outputs (also called *nested* SegWit). We thus have encapsulated Pay-to-Witness-Public-Key-Hash (P2SH-P2WPKH), native Pay-to-Witness-Public-

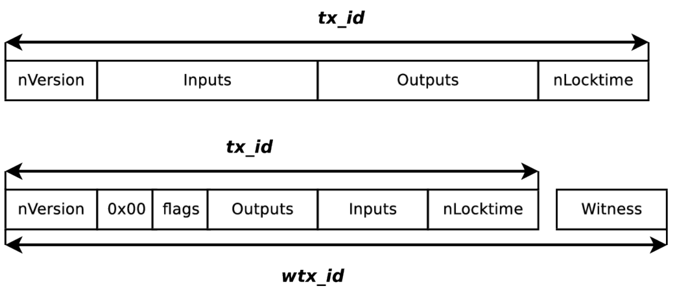

Fig. 2. Simplified SegWit transaction id vs. non-SegWit transaction id.

Key-Hash (P2WPKH), or native Pay-to-Witness-Script-Hash (P2WSH).

As a result of the use of SegWit, the size of a transaction is reduced, increasing the number of transaction that fit in a 1MB block. Fees for transaction are also reduced and computed based on the transaction *virtual size*.

SegWit also introduces a script versioning system, which allows to expand the scripting language through soft-forks.

It is true that in overall, SegWit introduces some complexity to Bitcoin, but this complexity was required in order to maintain backward compatibility and allow its deployment as a soft-work.

IV. Analysis

In this section, an analysis of the usage of segregated witness in the Bitcoin blockchain is presented. The analysis is made using BlockSci [12], an open-source software for blockchain analysis.

Segregated Witness was activated on the main Bitcoin network at block height 481,824 (on the 24th of August, 2017). Previously, it was locked-in at height 479,707, thus providing almost a two-week grace period for clients and wallets to upgrade to the new rules. Once activated, the new consensus rules started to be enforced, allowing to create transactions whose signature data was placed outside the old transaction format, with all the consequences on scalability and malleability that this implied.

As explained in the previous section, SegWit scripts may be deployed either with their native definition or encapsulated within standard BIP16 P2SH outputs. Native SegWit outputs may be identified directly by looking at their output scripts, because they follow a fixed structure. On the contrary, SegWit scripts encapsulated in P2SH outputs can not be identified until the output has been spent (and thus the redeem script has been revealed). This is because a P2SH output only contains the hash of the redeem script and, therefore, the actual redeem script is not known until the output is spent. Consequently, our analysis of SegWit usage is focused in studying, on one hand, redeem scripts found in P2SH inputs (for nested SegWit) and, on the other hand, output scripts (for native SegWit). Finally, we study the effects of deploying SegWit (both native and nested scripts) in the scalability of the system, both directly by effectively allowing bigger block sizes and indirectly by solving the malleability problem.

A. *P2SH nested SegWit scripts*

The firsts SegWit transactions in the Bitcoin main network correspond to P2WPKH nested within P2SH inputs and are found in block 481,824. This very same block has five transactions with this kind of inputs.[4] Three of these transactions include references to the activation of segregated witness: two of them have `OP_RETURN` outputs with messages saying hello to SegWit and a third one spends an output with a vanity address starting with `35SegWitPieWKVH...`. The first P2SH input with a P2WSH redeem script is found a little latter, in block 481,831.[5] Figure 3 shows the (2000-block moving average) percentage of inputs in each block that correspond to P2SH scripts with either P2WPKH or P2WSH redeem scripts. P2WPKH shows an still increasing tendency (accounting for almost 8% of today's inputs). Contrarily, P2WSH shows an increase during the first months (up until block 490,000) but seems to stabilize from that block onward. Altogether, nested SegWit scripts account for up to 11% of the current Bitcoin inputs in blocks.

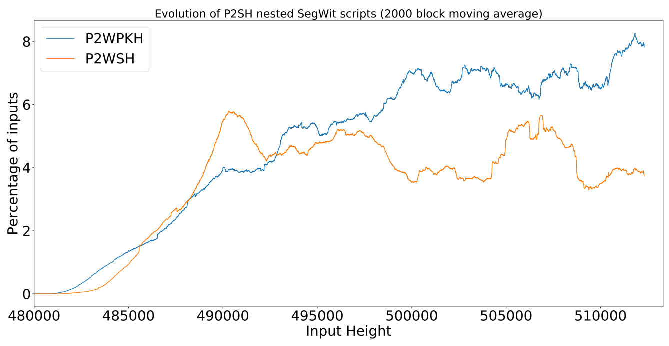

Fig. 3. Evolution of the number of P2SH nested SegWit inputs.

The average number of blocks between a P2SH SegWit input and the output it spends is 1,032 blocks, with small differences between nested P2WPKH and P2WSH inputs.

B. *Native SegWit scripts*

The first native SegWit outputs can be also found in block 481,824, that contains one transaction with a P2WSH[6] and two with P2WPKH outputs[7]. One of these two transactions is also spending a P2WPKH nested in P2SH input, and the other contains an `OP_RETURN` output announcing that a Japanese exchange supports SegWit. Figure 4 shows the (2000-block moving average) percentage of outputs in each block that correspond to native SegWit scripts, either P2WPKH or P2WSH witness programs. In contrast with encapsulated SegWit scripts, the adoption of native SegWit scripts is still very small, with an average of less than 0.5% of the outputs using these kinds of scripts. However, its adoption is still increasing. Also in contrast with nested SegWit scripts, native P2WSH outputs are seen more often than native P2WPKH outputs.

[4]Transactions with ids:
8f907925d2ebe48765103e6845c06f1f2bb77c6adc1cc002865865eb5cfd5c1c
b6e7c5365351ec7f8d29725afcdace8d76cf63a83bc9b58b9ff7cba61670b2be
c586389e5e4b3acb9d6c8be1c19ae8ab2795397633176f5a6442a261bbdefc3a
d09e2a5edbb6a0ac390a52a1b5292d88667f5445eb8e507441737a7bdd7157ee
dfcec48bb8491856c353306ab5febeb7e99e4d783eedf3de98f3ee0812b92bad

[5] fb5e59e65fa92719b3e24f17f20e5a2e19cf0a3643403d512d0dd806a84d510d

[6] 461e8a4aa0a0e75c06602c505bd7aa06e7116ba5cd98fd6e046e8cbeb00379d6

[7] dfcec48bb8491856c353306ab5febeb7e99e4d783eedf3de98f3ee0812b92bad
f91d0a8a78462bc59398f2c5d7a84fcff491c26ba54c4833478b202796c8aafd

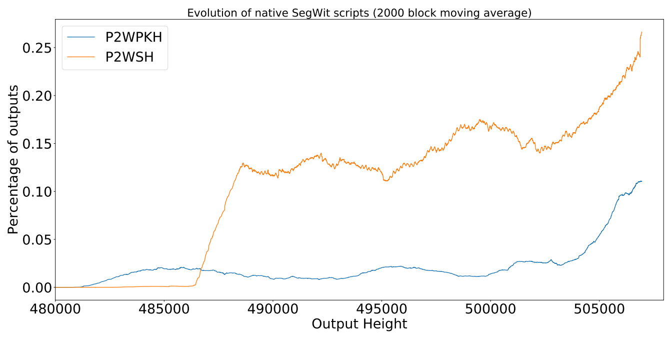

Fig. 4. Evolution of the number of native SegWit outputs.

Regarding the percentage of native SegWit scripts that were spent when this analysis was performed (block height 507,952), native P2WSH and P2WPKH present notable differences: 96.92% of P2WSH outputs have been already spent, whereas this percentage goes down to 74.21% for P2WPKH outputs.

Taking into account already spent (native SegWit) outputs only, the average number of blocks between an output and the input that spends it is 310 (the average for P2WPKH outputs is 529, whereas for P2WSH outputs is 287).

C. The impact of SegWit in the scalability of Bitcoin

SegWit inputs allow to move part or the totality (nested or native SegWit inputs, respectively) of their signature data to the witness structure. Since witness data is discounted when accounting for the total block size, SegWit allows to effectively create bigger block sizes.

Figure 5 shows the evolution of block sizes in Bitcoin. The orange dots denote base block size, that is, the size of the block without witness data. Base block size is limited to 1MB. The evolution of base block size over time shows how some blocks started to became full around height 325,000. Blue dots denote total block sizes, that is, block sizes including witness data. Since segregated witness was activated, many blocks total sizes have surpassed the 1MB limit, with block 505,253 setting a maximum of 2,217,342 bytes. However, all these blocks have a virtual size still less than 1MB, allowing them to be valid with the existing consensus rules. For instance, block 505,253 has a virtual size of 998,206 bytes. Currently (20th of April, 2018), up to 15,121 blocks have total sizes bigger than 1MB.

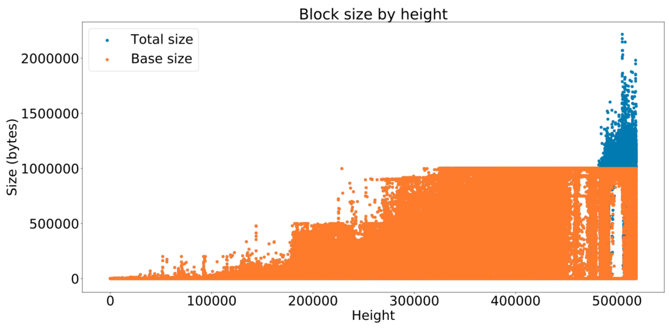

Fig. 5. Evolution of the size of blocks in Bitcoin.

Apart from the increase in block sizes, SegWit also solved the malleability problem, and thus allowed the deployment of payment channel networks such as the Lightning Network. Currently, the Lightning Network has almost 2,000 nodes and more than 5,000 channels[8].

V. Conclusions

After a very controversial scaling debate, Segregated Witness was deployed in the Bitcoin main network as a solution to deal with the scaling problems of Bitcoin while collateraly offering many other features (such as solving the malleability problem or introducing a new way to make upgrades to script).

As we have seen in this paper, since its activation SegWit has actively been used, mainly in its nested form where it is encapsulated within a classical P2SH script. The usage of native SegWit outputs is increasing, but it is still very residual. In any case, the usage of SegWit transactions has allowed, on one hand, to surpass the 1MB block limit and, on the other hand, to effectively deploy second layer solutions such as the Lightning Network.

Acknowledgements

This work is partially supported by the Spanish ministry under grant number TIN2014-55243-P.

References

[1] K. Croman, C. Decker, I. Eyal, A. E. Gencer, A. Juels, A. Kosba, A. Miller, P. Saxena, E. Shi, and E. Gün, "On scaling decentralized blockchains," in *Proc. 3rd Workshop on Bitcoin and Blockchain Research*, 2016.
[2] Visa, "56,582 transaction messages per second!" http://visatechmatters.tumblr.com/post/108952718025/56582-transaction-messages-per-second, July 2014, last accessed: June 2016.
[3] Paypal, "Paypal q1 2016 fast facts," June 2016, last accessed: June 2016.
[4] J. Garzik, "BIP 100: Making decentralized economic policy," 2015, last accessed: June 2016.
[5] G. Andresen, "BIP 101: Increase maximum block size," 2015, last accessed: June 2016.
[6] J. Garzik, "BIP 102: Block size increase to 2mb," 2015, last accessed: June 2016.
[7] P. Wuille, "BIP 103: Block size following technological growth," 2015, last accessed: June 2016.
[8] G. Andresen, "BIP 109: Two million byte size limit with sigop and sighash limits," 2016, last accessed: June 2016.
[9] M. Hearn and J. Spilman, "Bitcoin contracts," https://en.bitcoin.it/wiki/Contract, last accessed: June 2016.
[10] J. Poon and T. Dryja, "The bitcoin lightning network: Scalable off-chain instant payments," Technical Report (draft). https://lightning. network, Tech. Rep., 2015.
[11] E. Lombrozo, J. Lau, and P. Wuille, "BIP 141: Segregated witness (consensus layer)," 2015, last accessed: June 2016.
[12] H. Kalodner, S. Goldfeder, A. Chator, M. Möser, and A. Narayanan, "Blocksci: Design and applications of a blockchain analysis platform," *arXiv preprint arXiv:1709.02489*, 2017.

[8]There are many graphical explorers of the Lightning Network. A list can be found in https://gist.github.com/bretton/798ec38165ffabc719d91e0f4f67552d