Motivation
Scope
Problem
Solution

Hardening Bitcoin against off-topic data inclusion attacks
Master’s Thesis Conclusion

Andreas Seeg (andreas.seeg@posteo.de)

Advised by: Prof. Dr. Dmitrienko
Pseudonymously published paper in 2008
- Decentralized payment system
- Stores transactions in a blockchain, a distributed and immutable data structure
- "Secure" if attackers control less than 50% of utilized computing power
- >$3,000,000,000 transferred daily
Motivation Scope Problem Solution

Bitcoin’s fundamental principles

- 21 million coins
- No censorship: Nobody should be able to prevent valid txs from being confirmed
- Open source
- **Permissionless:** No arbitrary gatekeepers
- Pseudonymous: No ID required
- Fungible: All coins are equal and should be equally spendable.
- Irreversible Transactions: Blockchain History should be immutable.
Proposal PIBA

- **Problem**
  - Illegal/unneeded data on blockchain hurts Bitcoin

- **Idea**
  - Impede users’ ability to put arbitrary data on the blockchain via proof system

- **Benefit**
  - Attackers can’t store (illegal) data on the blockchain to hurt Bitcoin

- **Action**
  - Create proof system and polymorphic engine for Bitcoin core
The Problem

Once on the blockchain, data can not be removed and is needed for block verification.

Legitimate content on the blockchain:
- financial transactions
- conditional transactions / contracts through scripts
- arbitrary data (ex.: proof of ownership)

Why is this a problem?
Possible Scenario

[Ateniese et al., EuroS&P, 2017]:

- Store illegal content on the blockchain
- Wait for lawsuit
- Transmission of that part of the blockchain becomes illegal
- Validation of blocks and/or transactions on the blockchain fails

$\rightarrow$ Bitcoin is shut down / looses many participants.
**Motivation**

Scope

**Problem**

**Solution**

---

**Bitcoin’s Blockchain**

Andreas Seeg (andreas.seeg@posteo.de) — Hardening Bitcoin against off-topic data inclusion attacks 7/27
Example for P2PKH payment type

TODO CHANGE THIS

- Sent 0.01 BTC to 75 addresses
- Addresses are actually 20 Bytes of ASCII
Two-part solution

(1) Simple Bitcoin payments

(2) Bitcoin payment scripts
Implementation of P2SH² idea (2013)

TODO CHANGE THIS
Old proposal by Gregory Maxwell is similar to own idea

Implemented in most popular software (Bitcoin Core, C++)

Expanded idea to include P2PK payments

Bitcoin nodes running this code only accept transactions where proof can be provided

Useful as a proof of concept

Useful because it uncovered details about Bitcoin not apparent from the documentation
Recent research (2018)

- Even better (theoretical) solution by R. Matzutt et al ("Thwarting unwanted blockchain content insertion")
- Hard to include data in transactions because only "cryptographically non-manipulable" values are stored
- **Takeaway**: Various ways to make **simple** Bitcoin payments free of bad data
Two-part solution

(1) Simple Bitcoin payments

(2) Bitcoin payment scripts
Complicated Bitcoin payment scripts

- Arbitrary code in special programming language
- Not turing complete, but potent
- Only hash of script is provided on payment
- Script (arbitrary data) is shown when received Bitcoin are to be spent
- Problem: What if script encodes illegal data?
Example for P2SH payment type

TODO CHANGE THIS DATA NOW IN INPUTS
Script has to be read and run by most Bitcoin participants

- It is user-definable by definition

⇒ Inclusion prevention not possible
Transaction deletion breaks verifiability:

1. Transaction hash cannot be calculated
   ⇒ Hash of merkle tree cannot be calculated
   ⇒ Block hash verification fails
   ⇒ Bitcoin block is rejected

2. Unclear which bitcoin were spent and who received them
   ⇒ Breaks verifiability: Who has how many bitcoin?
   ⇒ Breaks trustlessness: What if someone else’s Bitcoin were used?
Deletion

**Motivation**

**Scope**

**Problem**

**Solution**

- **Blockheader #2**
  - Hash of previous block
  - Timestamp
  - Hash of all transactions
  - Nonce

- **Hash 12**
  - Hash #1
  - Hash #2
  - Tx #1
  - Tx #2

- **Hash 34**
  - Hash #3
  - Hash #4
  - Tx #3
  - Tx #4

- **Hash 78**
  - Hash #7
  - Hash #8
  - Tx #7
  - Tx #8

- **Blockheader #3**
  - Hash of previous block
  - Timestamp
  - Hash of all transactions
  - Nonce

- **Hash 56**
  - Hash #5
  - Hash #6
  - Tx #5
  - Tx #6

- **Tx #5 references Tx #1 with bad data**

- **Tx #7 references clean Tx #4**
Deletion: Idea

- Idea: Make only Script deletable
- Replace script with a reference (hash) within the transaction
- Put Script outside of the block, but still require it
  ⇒ Hash of transaction can be calculated
  ⇒ Hash of merkle tree can be verified, block verification ok
  ⇒ Verifiable ledger: It is clear who has how many bitcoin
- Still not trustless: What if everybody in the past is lying?
Pay for deletion

- Inputs prove that bitcoins may be used/sent with a transaction
- Input script to be deleted? Pay the same amount of bitcoin for deletion
- No monetary gain from lying, no trust necessary
- Payment via clean, simple transaction (solution 1)
- Caveat: Small chance of abuse
Implementation is complicated

- All transaction types have to be reworked
- New block layout (externalized scripts)
- Some transaction types have to be disabled (alternatives exist)
- Verification logic change: New blocks become important sooner
Problem
- Allowing the inclusion of illegal or unneeded data on the blockchain hurts Bitcoin

Idea
- Impede users’ ability to put arbitrary data on the blockchain via proof system. Allow deletion in special cases.

Benefit
- Attackers can’t store (illegal) data on the blockchain to hurt Bitcoin, or it can be deleted for a price.

Action
- Creation of a proof system for simple payments and proposal of a potential deletion mechanism that becomes trustless by requiring payment via the simple payment proof system.
Motivation Scope Problem Solution

■ Proof of concept usage
■ Discussion of deletion solutions for different software systems
Further research

- Convoluted content insertion via transaction ids.
- Software implementation of second part of the thesis.
- Remaining question on what to do with censored outputs and how to sell the idea to the Bitcoin userbase
[Ateniese et al., EuroS&P, 2017]
Redactable Blockchain -- or -- Rewriting History in Bitcoin and Friends, Giuseppe Ateniese and Bernardo Magri and Daniele Venturi and Ewerton Andrade, Cryptology ePrint Archive, Report 2016/757

[R. Matzutt et al., 2018]
Thwarting unwanted blockchain content insertion, M. Henze, J. H. Ziegeldorf, J. Hiller, K. Wehrle, IC2E

[G. Maxwell, 2018]
P2SH², sourceforge.net/o/bitcoin/mailman/message/30705609, 2013
Questions
What to do with freed outputs?

- During long 51 percent attack, anyones bitcoin can be spent and the input censored
- (1) If output remains used, someone just lost money
- (2) If output becomes unused, someone can gain money by putting bad data on the blockchain
- : But: Making someone lose money (1) requires a sustained 51 percent attack, making Bitcoin broken anyway
- (2) can be made unappealing by not censoring all inputs and introduction of liability for block creators