Thwarting Unwanted Blockchain Content Insertion

Roman Matzutt, Martin Henze, Jan Henrik Ziegeldorf, Jens Hiller, Klaus Wehrle
Communication and Distributed Systems
RWTH Aachen University
Aachen, Germany
Email: lastname@comsys.rwth-aachen.de

***Abstract*—Since the introduction of Bitcoin in 2008, blockchain systems have seen an enormous increase in adoption. By providing a persistent, distributed, and append-only ledger, blockchains enable numerous applications such as distributed consensus, robustness against equivocation, and smart contracts. However, recent studies show that blockchain systems such as Bitcoin can be (mis)used to store arbitrary content. This has already been used to store arguably objectionable content on Bitcoin's blockchain. Already single instances of clearly objectionable or even illegal content can put the whole system at risk by making its node operators culpable. To overcome this imminent risk, we survey and discuss the design space of countermeasures against the insertion of such objectionable content. Our analysis shows a wide spectrum of potential countermeasures, which are often combinable for increased efficiency. First, we investigate special-purpose *content detectors* as an ad hoc mitigation. As they turn out to be easily evadable, we also investigate content-agnostic countermeasures. We find that *mandatory minimum fees* as well as mitigation of transaction manipulability via *identifier commitments* significantly raise the bar for inserting harmful content into a blockchain.**

***Keywords*-Bitcoin, blockchain, security, objectionable content, countermeasure**

I. Introduction

Blockchain-based cryptocurrencies such as Bitcoin enjoy unbroken popularity, averaging at over 280 000 daily confirmed transactions in 2017 [1]. This popularity is also reflected by the size of the cryptocurrencies' underlying peer-to-peer networks and their user base: Bitcoin's network size has doubled since 2015 [2], while its number of users is peaking in the millions [3]. Cryptocurrencies, especially Bitcoin, have thus become a well-accepted trading medium due to their security, timeliness, and decentralization.

Besides offering a platform for financial transactions, recent work [4]–[6] shows that Bitcoin's blockchain can also be used as an anonymous and irrevocable content store. By inserting short, non-financial messages, Bitcoin can be extended to realize additional services, e.g., digital notary services [7], secure releases of cryptographic commitments [8], or non-equivocation schemes [9].

While this initially unintended extensibility appears promising, and in fact 1.4 % of Bitcoin transactions hold non-financial data [6], it can severely compromise blockchain systems: A recent study [6] reveals that over 1600 files have been irrevocably engraved into Bitcoin's blockchain, e.g., to be shared in a censorship-resistant manner. These files range from simple text to over 155 images, source codes, and PDF files. Any objectionable content, e.g., illegal pornography, in such files is then inevitably distributed to all nodes of the cryptocurrency's underlying peer-to-peer network. It is expected that court rulings in major jurisdictions such as Germany and the USA will then find node operators culpable of possessing objectionable content [6]. As a consequence, the node operators must delete affected parts of the blockchain, thereby breaking the blockchain's integrity and verifiability. The insertion of objectionable content has thus the potential of jeopardizing cryptocurrencies, as all users ultimately depend on this verification. Indeed, recent research [6] finds that, while most content is likely harmless, Bitcoin's blockchain already today contains content that is objectionable in many jurisdictions, e.g., an image of a nude young woman or hundreds of links to child pornography.

Reacting to the evident threats, in this work we explore potential *countermeasures* to prevent insertion of objectionable content w.r.t. blockchain-based cryptocurrencies, using Bitcoin as a real-world working example. We first analyze the harmfulness of different types of blockchain content and find that short, token-like messages enable beneficial use cases while insertion of arbitrarily-sized content must be prevented and we acknowledge that full prevention of content insertion is impossible. We hence focus on countermeasures that either heuristically hinder the insertion of *large* chunks of arbitrary data or financially disincentivize content insertion to a freely tunable degree. Our results are two-fold. One the one hand, we find that the naïve approach of targeted content detection constitutes a first ad hoc solution against objectionable content, but it is easily evadable. On the other hand, simple adaptions to Bitcoin, such as introducing mandatory minimum fees or replacing manipulable blockchain identifiers, can effectively mitigate objectionable blockchain content with moderate overheads.

II. Background

In this section, we provide the technical background on Bitcoin required for the remainder of this paper.

Bitcoin Primer. Bitcoin [10] was the first digital currency to rely on the *blockchain*. The blockchain is a persistent, distributed, and append-only ledger of *events*, serving in

Bitcoin as distributed, immutable record of financial transactions between pseudonymous *Bitcoin addresses*. A Bitcoin address is controlled by a cryptographic key pair, which is used to access and transfer the associated bitcoins. A *transaction* collects funds from one or more addresses, the *inputs*, and reassigns them to one or more other addresses, the *outputs*. To prevent double spending of bitcoins, transactions are only considered valid if they are immutably recorded in the blockchain. Transaction inputs and outputs are realized using a script language that allows authenticating payments via addresses' public keys or hash values thereof.

Blockchain Maintenance. The blockchain is maintained by the *Bitcoin network* in a distributed manner to reach consensus about valid transactions among honest nodes [11]: Users propose their transactions to the network, which are then added to the blockchain by special nodes, the *miners*, via a proof-of-work scheme. To ensure correctness of the blockchain and its transactions, the *full nodes* of the Bitcoin network *independently* verify all transactions and blocks they receive and reject incorrect information. Furthermore, full nodes maintain a *full copy* of the whole blockchain to serve newly joining nodes. Assuming an honest majority (among full nodes), the longest blockchain constitutes the Bitcoin network's consensus.

Incentives and Fees. Miners are incentivized to perform the proof of work via block rewards. Each miner includes a *coinbase transaction* in her blocks, which rewards her with a prescribed number of freshly minted bitcoins. As the block reward is exponentially decreasing to limit the total supply of bitcoins, *transaction fees* are a second tier of miner rewards. Miners may collect excess bitcoins from all transactions in their blocks as a form of tip, which is paid as a byte-wise fee. As overpaying fees incentivizes miners to consider a transaction faster, recommendations on transaction fees emerged [12]. For instance, to get a transaction included into the blockchain during December 2017 within an hour, it is recommended to pay on average 423 satoshi per Byte (B), i.e., 16.14 USD for an average transaction of 250 B size [12].

Content Insertion. A comprehensive study on how Bitcoin transactions can be augmented with arbitrary content is given in [6]. While short messages of up to 100 B can be added via intended mechanisms (coinbase transactions and `OP_RETURN`), transactions can be *manipulated* to hold arbitrarily-sized content such as images or archives. Predominantly, content inserters arbitrarily replace the *blockchain identifiers*, usually 20 B-long cryptographic hash values of public keys, of multiple outputs with their content, potentially making the output unspendable [6]. Using encodings such as Apertus [13] allows spreading content over multiple transactions while retaining efficient decoding [6].

III. Scenario and Problem Statement

In this section, we define the underlying scenario for blockchain content insertion with the goal of designing countermeasures against such practices. We first outline that different classes of arbitrary blockchain content can be harmful or beneficial to cryptocurrencies (Section III-A). We then discuss related work (Section III-B) and show that it is impossible to prevent insertion of *all* unintended content into the blockchain (Section III-C). From this analysis, we distill the problem statement for this paper (Section III-D).

A. Harmfulness of Arbitrary Blockchain Content

Current blockchain designs allow augmenting user-generated transactions with short chunks of *arbitrary* content as described in Section II. Notably, manipulating transactions allows inserting *arbitrary amounts* of *unintended* data even into special-purpose blockchains, e.g., storing non-financial data on cryptocurrency blockchains. In this paper, we refer to Bitcoin as our real-work working example.

If a miner includes content-holding transactions into her blocks, the content is irrevocably distributed to all full nodes. Recent research [6] shows that this puts full node operators at risk: Although court rulings are yet to come, it is expected that major jurisdictions could find that maintaining a blockchain containing objectionable content, e.g., illegal pornography, constitutes possession of illegal documents [6]. Full node operators hence face a dilemma: If they keep maintaining the blockchain, they may become culpable. Yet, if they delete content-holding transactions from their local blockchain, they break its integrity and thus its verifiability.

Deleting blockchain content locally to comply with legal obligations severely impedes the health of the Bitcoin network. It is critical that newly joining full nodes obtain an intact blockchain copy in order to successfully synchronize with the Bitcoin network. Furthermore, also the users of lightweight solutions such as online wallets ultimately depend on full nodes performing the verification process on their behalf. Hence, we argue that objectionable, i.e., illegal-to-possess, blockchain content must be proactively prevented from entering the blockchain to the largest extent possible.

However, only certain content can jeopardize blockchain systems. In fact, the outlined culpability only holds if content is objectionable and *easily extractable* from the blockchain. For instance, full pictures of illegal pornography, as one the most ubiquitously objected content type, can be stored on the blockchain using tens of kilobytes [14]. The most imminent risk therefore stems from *arbitrary-length* and *easy-to-read* content, especially if it is easily accessible via standard software after extracting it from the local blockchain copy.

Contrarily, *short* pieces of blockchain content are less likely to be harmful as they cannot hold objectionable content directly. Even short *links* to objectionable content do *not* put full node operators at risk of possessing said content: As the content is not stored directly on the blockchain, operators do not own a physical copy of it. Furthermore, they could even cooperate with local authorities to take down the target server without impeding the blockchain integrity.

As a consequence, we deem short-sized content ($\leq 1\,\mathrm{KiB}$) to have a lower harm potential and consider very short contents ($\leq 100\,\mathrm{B}$) harmless. These thresholds may, however, require adaption in the future, e.g., due to court rulings.

On the contrary, short-sized blockchain content has proven to fuel innovation and create new applications. A wide range of applications now rely on engraving short tokens into the blockchain to leverage its security model for off-blockchain services. By adding hash values of files, it becomes publicly verifiable that a given document existed by the time the transaction was added to the blockchain [7]. Similarly, the blockchain can become a general-purpose event ledger, e.g., for non-equivocation logs [9]. Allowing short text messages, e.g., meta information, on the blockchain furthermore enables services such as distributed management of assets [15] or key-value pairs [16] and the execution of smart contracts [17]. Although arbitrary-length blockchain content can also be beneficial, e.g., for whistleblowers to unveil misconducts in a censorship-resistant manner, a single piece of objectionable content can jeopardize the whole system. Hence, the risks introduced by arbitrary-length blockchain content by far outweigh the potential benefits.

It is thus crucial to face the currently disregarded risks of arbitrary-length, easy-to-read blockchain content and to design countermeasures. However, the benefits of short blockchain messages require such designs to trade off security against innovation by gauging which content is harmless.

B. Related Work

Monitoring incoming data as well as recognizing and filtering unwanted content is a classical application of firewalls, intrusion detection systems, and spam filters [18]–[21], which have drastically improved security and quality of service within their respective domains. However, their often high adaptability requirements are commonly tackled via supervised or automated learning of what content should be filtered. This is challenging w.r.t. blockchain systems as it must be guaranteed that learning is deterministic and all overhead from computation-intense local checks multiply over the whole network and should be avoided. However, we deem a further investigation promising future work.

A new line of blockchain-based systems promises to avoid the problems of objectionable blockchain content altogether by persistently maintaining account balances instead of the whole transaction ledger [22]–[24]. As transaction outputs are separated during the balance update, it is hard to link individual chunks of blockchain content to each other to reveal the full content. Yet, forfeiting the event history considerably limits potential applications. For instance, notary services cannot be realized via such blockchains. The risks of content insertion must thus be tackled for all blockchains.

Similarly, redactable blockchains [25] emerged, which enable after-the-fact alternation and deletion of transactions. These blockchains use chameleon hash functions [26] to link blocks such that trusted entities or quorums can alter them. The arising trust issues can be tackled by issuing decentralized votes on blockchain alternations [27], but again, this enhancement is incompatible to existing systems.

C. Impossibility of Rigorous Blockchain Content Filtering

We have shown that objectionable content puts blockchain systems at risk. However, as we argue in this section, unintended data cannot entirely be prevented from entering public blockchains. This circumstance stems from the fact that public blockchains are pseudonymous. Full nodes do not verify whether an alleged receiver in a proposed transaction does indeed exist. As of now, content can thus be easily inserted by manipulating the receiver's identifiers. E.g., Bitcoin allows inserting tens of kilobytes of arbitrary data per transaction using multiple manipulated outputs [6].

Unfortunately, such manipulation cannot be comprehensively detected by full nodes: Users can, and are even encouraged to [10], refresh their Bitcoin addresses arbitrarily often. Hence, a user willing to insert blockchain content can continuously create new Bitcoin addresses for herself by brute-forcing private keys such that the content can be encoded via, e.g., the first few bytes of each output used. The user can subsequently craft a transaction that sends arbitrary amounts of bitcoins to each of her chosen outputs and publish it. This transaction is valid and indistinguishable from non-manipulated transactions. Furthermore, full nodes cannot link the transaction outputs to the single user during transaction validation due to the lack of centralized user-identity associations. Hence, it is impossible for full nodes to detect and reject all transactions holding unintended content.

In the following, we thus explore *heuristics* that reject potentially harmful content, either by analyzing transactions to reveal content, e.g., plain text or image files, or by disincentivizing insertion of arbitrarily-sized content.

D. Problem Statement

We argued that persistently storing arbitrarily-sized content puts blockchain systems at risk while short data pieces have proven beneficial. Furthermore, we have shown the impossibility of preventing all unintended blockchain content. The goal of this paper is therefore to explore the design space of *countermeasure heuristics* that (i) prevent harmful content from entering the blockchain, (ii) are easily deployable even for established systems such as Bitcoin, and (iii) are *adaptable* in case that also short pieces of blockchain content reveal unforeseen risks. To this end, content pieces shall be (i) allowed if very short ($\leq 100\,\mathrm{B}$), (ii) tolerated if medium-sized ($\leq 1\,\mathrm{KiB}$), and (iii) effectively prevented if arbitrarily-sized. We note that these thresholds can be freely adapted if deemed necessary, e.g., by court rulings. As complete prevention of content insertion is impossible, our countermeasures must render the insertion of arbitrarily-sized content either computationally or financially infeasible.

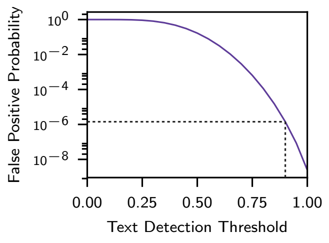


Figure 1: Expected false-positive rate for text detection in a $20\,\mathrm{B}$-long identifier

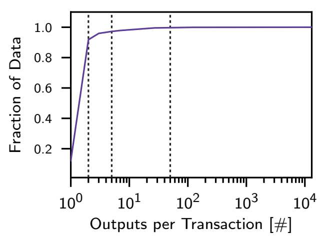


Figure 2: Cumulative distribution of numbers of outputs per transaction

IV. Countermeasures Against Content Insertion

To effectively mitigate risks of blockchain content, we identify countermeasures that limit the amount of insertable, unintended content or make such insertion financially infeasible in accordance with our problem statement in Section III-D. As a representative working example, we consider countermeasures that are easily applicable to Bitcoin. After proposing evaluation criteria for the countermeasure quality (Section IV-A), we propose to (i) introduce naïve content filtering (Section IV-B), to (ii) adapt the fee model (Section IV-C), and to (iii) include proofs of key authenticity within the transactions themselves (Section IV-D).

A. Evaluation Criteria

We evaluate our countermeasures' efficiency against insertion of harmful blockchain content w.r.t. *filtering quality*, *usability*, *network burden*, and *deployability*. High *filtering quality* means that insertion of harmful content (as in Section III-D) becomes either computationally or financially infeasible. A countermeasure is *usable* if it does not impede normal system use. A low *network burden* is achieved if neither the blockchain's nor the Bitcoin network's performance is decreased significantly by a countermeasure. Finally, a countermeasure should be *deployable* via only *minor* changes to already-established blockchain systems (changing transaction acceptance *always* requires an update).

B. Filtering Content-holding Transactions

An intuitive countermeasure against unintended blockchain content is the systematic analysis of proposed transactions and subsequent rejection of content-holding transactions by full nodes and miners. As discussed in Section III-A, the most imminent threat stems from arbitrary-sized content that is objectionable and easily accessible, e.g., common files such as images. We thus first explore the naïve approach of adding *content detectors* to Bitcoin's transaction verification to detect and reject content-holding transactions.

To detect easily accessible content in transactions, we propose (i) a *text detector* to identify transactions carrying text or ASCII-based files and (ii) a *known-file detector* to identify binary files such as images or archives.

Detecting large fractions of printable text within a transaction prevents custom text as well as text-based files, e.g., HTML pages or Python scripts, from entering the blockchain. We hence propose a *text detection threshold* $t \in [0, 1]$ to check whether individual transaction outputs consist of large fractions of printable ASCII characters.

To choose t, we consider the detector's expected false-positive rate (FPR). False positives can occur as blockchain identifiers may contain printable ASCII characters by chance. Figure 1 shows the expected FPR for random blockchain identifiers ($20\,\mathrm{B}$ length, 95 of 256 printable characters) and varying thresholds t. We observe that only high thresholds lead to negligible FPRs: While $t = 0.75$ still yields an expected FPR of $0.064\,\%$, $t = 0.9$ yields an expected FPR of $1.42 \times 10^{-4}\,\%$. As this confirms the intuition from previous works [6], [28], we suggest $t = 0.9$.

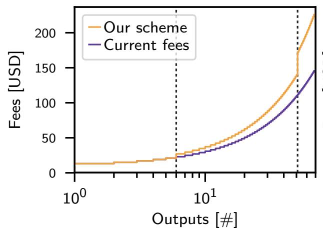
Unfortunately, reusing Bitcoin addresses, e.g., to collect donations, can cause the text detector to reject valid payments if the corresponding blockchain identifier is a false positive. Hence, also small expected FPRs can impede the usability seriously as such identifiers are potentially used heavily. This is illustrated, e.g., by one Bitcoin address[1], which is a false positive w.r.t. our text detector but received bitcoins from over 370 transactions as of January 8th, 2018.

We thus propose to further restrict the text detector to reject only transactions with more than 5 distinct text-holding outputs ($100\,\mathrm{B}$ worth of content). This way, only short and thus harmless texts can be inserted. Notably, most of these texts can already be inserted using an `OP_RETURN` output (at lower costs). Hence, the text detector mitigates harmful text insertion without discriminating honest users.

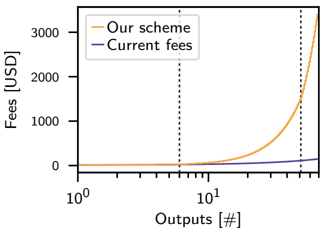
Contrarily, preventing insertion of binary files is not feasible via this approach. While file types can be determined using magic numbers, i.e., byte sequences that are unique to the respective file type, these sequences are often only few bytes long [29]. Hence, the expected FPR increases drastically. Although the accuracy can be improved by considering more characteristic features, e.g., default headers, content inserters can evade this overly specific detector more easily. They can, e.g., introduce easily revertible modifications to the content such as a deterministic padding.

Evaluation. Content detectors can be fine-tuned specifically to reject unwanted content with a low overhead for full nodes. Furthermore, usability is guaranteed since we detect false positives with high probability. Thus, honest users are not affected by the detector. However, the detector's filtering quality is insufficient for binary files and we expect a wide range of evasion schemes to emerge. This would imply a poor deployability, as novel evasion schemes result in *frequent* mandatory updates for *all* honest full nodes. In conclusion, explicit content detection constitutes a first line of defense against unwanted blockchain content and works for text-based content. Mitigating the insertion of unwanted binary files, however, requires more general approaches.

[1]Bitcoin address: 154QWLN3Uz43nHMAM7ioYUx8tkYXdNKDtQ



(a) Piecewise constant growth



(b) Piecewise linear growth

Figure 3: Proposed minimum fees for different growths $\beta(n)$

C. *Mandatory Minimal Transaction Fees*

Bitcoin transaction fees are usually paid per byte, with a current recommendation of $423\,\mathrm{satoshi/B}$ ($16.14\,\mathrm{USD}$ for the average $250\,\mathrm{B}$-large transaction) as of December 2017 [12]. Although such fees seem high, new content is actively being added [6]. We thus propose to adapt Bitcoin's underlying fee model to hinder content insertion.

As Figure 2 shows, the vast majority of all nearly 255 million transactions (as of August 2017) has at most 50 outputs ($99.73\,\%$). Of all transactions, $97.22\,\%$ even have 5 outputs or fewer and $91.77\,\%$ have at most 2 outputs.

Thus, we propose to enforce *mandatory minimum fees* to penalize large transactions and thus disincentivize content insertion. Given a proposed transaction t with size s_t and number of outputs n_t, we propose the simple fee function:

$$F(t) = \alpha \cdot (s_t + \beta(n_t) \cdot n_t).$$

Here, α is the byte-wise base fee and $\beta(n)$ is a piecewise-defined function depending on thresholds T_s and T_m that distinguish small, medium-sized, and large output numbers. As an example, Figure 3 shows the resulting fees for $\alpha = 423\,\mathrm{satoshi/B}$, thresholds $T_s = 6, T_m = 51$ and

$$\beta_C(n) = \begin{cases} 0 & n < T_{s,m} \\ 10 & n \in [T_s, T_m] \\ 20 & n > T_m \end{cases}$$

in Figure 3a and $\beta_L(n) = \sum_0^n \beta_C(n)$ in Figure 3b (the dashed lines denote T_s and T_m). While we leave the exact parametrization open, our choices for $\beta(n)$ to be piecewise constant ($\beta_C(n)$) or piecewise linear ($\beta_L(n)$) showcase the design space for mandatory minimum fees. Neither approach impedes small transactions (over $97\,\%$ of all transactions), but both penalize larger transactions in varying degrees.

For instance, using a piecewise constant fee growth per output, a borderline-large transaction (50 outputs) would inflict additional 29 USD of fees, while the linear fee growth yields roughly $1310\,\mathrm{USD}$ in additional fees, i.e., legitimate creators of medium-sized transactions are penalized more. However, the increase in penalty fees grows gradually for medium-sized transactions. Content inserters, contrarily, are especially punished by linear-growth fee penalties: E.g., storing a small JPEG image of $20\,\mathrm{KiB}$ (1024 outputs), would cost the content inserter roughly $1.19 \times 10^6\,\mathrm{USD}$ in fees.

Evaluation. Mandatory minimum fees are promising to disincentivize content insertion. They are easily deployable (a single check during transaction verification) and have negligible overhead as full nodes only must check whether the transaction pays at least the required fees. However, mandatory fees have an inherent trade-off between usability and filtering quality: If the fee model is tuned towards rejecting even small amounts of content, honest users currently relying on large transactions, e.g., exchange services, are potentially penalized. Hence, the fee model's parametrization must be thoroughly evaluated prior to its deployment.

D. *Self-Verifying Account Identifiers*

We propose a simple adaption of standard Bitcoin transactions to make content insertion computationally infeasible. We approximate the best case of infeasible content insertion outlined in Section III-C by only recording *cryptographically non-manipulable* values on the blockchain.

Currently, content inserters can easily replace mutable identifiers in their transaction outputs with arbitrary values. Full nodes are unable to validate the correctness of these identifiers until they receive a future transaction attempting to spend a particular output. Hence, full nodes are forced to accept manipulated transaction outputs into the blockchain. We hence propose to replace manipulable identifiers of transaction outputs with *identifier commitments (ICs)*. Namely, the IC $\mathcal{C}(x)$ is obtained by interpreting an identifier x as a private key over Bitcoin's elliptic curve secp256k1 and signing the corresponding public value $x{\cdot}G$ (G the generator of secp256k1) together with a salting nonce r via ECDSA:

$$\mathcal{C}(x) \quad := \quad (x{\cdot}G,\ r,\ \mathsf{sig}(x{\cdot}G \| r, x))$$

Replacing x with $\mathcal{C}(x)$ ensures that (a possibly content-hiding) x never appears on the blockchain. This hiding of x is sufficient for outputs, i.e., inputs are not manipulable as they can only be created by proving possession of a private key. To hinder content insertion, x must not be efficiently computable from $\mathcal{C}(x)$ and $\mathcal{C}(x)$ must not enable content insertion by other means than brute-forcing it. To this end, we rely on ICs to be *one-way*, *fresh*, and *self-verifying*.

The *one-way* property of $x \cdot G$ guarantees that x cannot be computed efficiently from $x \cdot G$. Thus, $x \cdot G$ can safely be added to the blockchain. Furthermore, x is unknown for manipulated $x \cdot G$ and hence the signature $\mathsf{sig}(x \cdot G \| r, x)$ cannot be computed. Thus, it is computationally infeasible to compute a valid $\mathcal{C}(x)$ hiding content in $x{\cdot}G$.

Freshness of $\mathcal{C}(x)$ is required to hamper the creation of rainbow tables for identifiers x that yield ICs well-suited for content insertion, e.g., a reusable file header. We ensure freshness via adding the salt $r = \mathrm{CRC32}(t_1 \| ... \| t_n)$, where t_i is the hash value identifying the previous, already mined transaction referenced by the i-th input. As r depends on predecessor transactions, it changes for almost all transactions, which makes creating rainbow tables unprofitable. Moreover,

```
                P2UC:                P2PKH:
Input Script:   [signature]          [signature]
                [public key]         [public key]
Output Script:  OP_DUP               OP_DUP
                OP_HASH256           OP_HASH160
                OP_COMMIT
                [commitment x*G]     [public key hash]
                [salt r]
                [sig(x*G||r, x)]
                OP_DROP
                OP_DROP
                OP_EQUALVERIFY       OP_EQUALVERIFY
                OP_CHECKSIG          OP_CHECKSIG
```

Figure 4: P2UC script; P2SC replaces P2SH analogously.

it prevents content storage in r itself. The $4\,\mathrm{B}$ short salt ensures that the blockchain is not bloated unnecessarily.

Finally, the *self-verifying* property ties $\mathcal{C}(x)$ to x such that only possession of the private value x enables the spending of funds. Furthermore, it ensures that $x \cdot G$, r, and the signature $\mathrm{sig}(x \cdot G \| r, x)$ cannot be manipulated to contain content without invalidating the signature.

Implementation. ICs can be easily integrated into Bitcoin by adding only a new `OP_COMMIT` operation to it's stack-based scripting language. This way, we can replace the most common transaction scripts, P2PKH and P2SH, with non-manipulable alternatives: *Pay-to-User-Commitment (P2UC)* and *Pay-to-Script-Commitment (P2SC)*.

Figure 4 exemplarily shows the transition from P2PKH to P2UC (P2SC analogously replaces P2SH). We replace the manipulable identifier x with its IC $\mathcal{C}(x)$. However, full nodes must verify the correctness of $\mathcal{C}(x)$ using the salt r and the signature $\mathrm{sig}(x \cdot G \| r, x)$ *before* the transaction is added to the blockchain. Hence, this check is independent from actually executing the script once the transaction output should be spent. Instead, full nodes only need to verify the correctness of $x \cdot G$ to authenticate payments. To this end, we introduce `OP_COMMIT`, which replaces the current top stack element x with $x{\cdot}G$. We ignore the signature and salt during payment verification via the `OP_DROP` operations.

Performance. We evaluate our scheme w.r.t. validation and payment verification times and transaction sizes.

Validating a IC requires one additional signature verification, which takes $0.4\,\mathrm{ms}$ on average on a commodity PC (Intel Core 2 Quad Q9400 CPU running at $2.66\,\mathrm{GHz}$ with $4\,\mathrm{GiB}$ RAM). Hence, full nodes need only $3.8\,\mathrm{s}$ to validate a proposed block consisting of roughly 1900 transactions (average over all blocks of 2017 [1]) with at most 5 outputs each (cf. Section IV-C). As new blocks are only published on average every $10\,\mathrm{min}$ [11], this additional check is clearly feasible for full nodes even for exceptionally large transactions with up to 50 outputs each ($38\,\mathrm{s}$).

Introducing ICs slightly changes payment verification: A full node must execute `OP_COMMIT` and compute `OP_HASH256` instead of `OP_HASH160`. Computing one IC takes $0.2\,\mathrm{ms}$ on average and, notably, `OP_HASH256` outperforms `OP_HASH160` for small input sizes. Furthermore, computing the salt as a CRC32 checksum is negligible. Hence, ICs do not impede payment verification.

Replacing blockchain identifiers with ICs increases the overall transaction size. An IC is $112\,\mathrm{B}$ long ($x{\cdot}G$: $33\,\mathrm{B}$; r: $4\,\mathrm{B}$; $\mathrm{sig}(x{\cdot}G\|r, x)$: $77\,\mathrm{B}$; $4\,\mathrm{B}$ for additional operations), in contrast to the $20\,\mathrm{B}$ size of the unprotected identifier. Hence, a standard (P2PKH) transaction consisting of one input and two outputs grows from $225\,\mathrm{B}$ to $409\,\mathrm{B}$. Blocks can therefore hold up to 2445 transactions, which comfortably sustains the current average of 1900 transactions per block.

Evaluation. ICs have a high filtering quality as they reduce insertable content to the theoretic minimum (cf. Section III-C). Furthermore, usability is not impeded: Users only need to additionally compute the IC, which is clearly feasible for frequencies of transaction creations expected for individual users. Even though we need to introduce `OP_COMMIT` and extend Bitcoin's transaction validation process, we argue that the required changes are only minor and thus ICs are well-deployable. ICs inevitably increase transaction sizes. However, the Bitcoin network can still sustain its transaction throughput. Thus, the overhead is worth the IC-based protection against content insertion.

V. Conclusion

The threat of inserting arbitrary content into blockchains was only recently recognized: Objectionable content can be anonymously and irrevocably inserted and thus distributed to the nodes of a blockchain-based system, whose operators can then be culpable for possessing the content.

We proposed conceptual countermeasures to empower the nodes of a blockchain-based system to heuristically reject transactions holding unintended content with high probability. Namely, we propose a content detector to identify and reject content-holding transactions, mandatory minimum transaction fees to make content insertion economically infeasible, and a computationally non-manipulatable, commitment-based replacement for easily manipulable identifiers in transaction outputs.

Until our countermeasures fully complement each other, they can be deployed gradually, e.g., the content detector is immediately deployable. Nevertheless, content inserters can quickly adapt to its strategy and evade detection. It is thus only an effective ad hoc defense that should soon be accompanied by a fee model that disincentivizes large transactions which are suited to hold objectionable content. Furthermore, using non-manipulable blockchain identifiers limits content insertion to the theoretical minimum at moderate costs.

Acknowledgments

This work has been funded by the German Federal Ministry of Education and Research (BMBF) under funding reference number 16KIS0443. The responsibility for the content of this publication lies with the authors, who would also like to thank the German Research Foundation DFG for the kind support within the Cluster of Excellence “Integrative Production Technology for High-Wage Countries”.

References

[1] Blockchain.info. (2011) Bitcoin Charts & Graphs. Accessed 02/04/2018. [Online]. Available: https://blockchain.info/charts

[2] A. Yeow, "Bitnodes: Global Bitcoin Nodes Distribution," 2018, accessed 02/04/2018. [Online]. Available: https://bitnodes.earn.com/dashboard/?days=730

[3] D. D. F. Maesa, A. Marino, and L. Ricci, "Data-driven analysis of Bitcoin properties: exploiting the users graph," *International Journal of Data Science and Analytics*, pp. 1–18, Sep. 2017.

[4] K. Shirriff. (2014) Hidden surprises in the Bitcoin blockchain and how they are stored: Nelson Mandela, Wikileaks, photos, and Python software. Accessed 02/04/2018. [Online]. Available: http://www.righto.com/2014/02/ascii-bernanke-wikileaks-photographs.html

[5] R. Matzutt, O. Hohlfeld, M. Henze, R. Rawiel, J. H. Ziegeldorf, and K. Wehrle, "POSTER: I Don't Want That Content! On the Risks of Exploiting Bitcoin's Blockchain as a Content Store," in *Proceedings of the 23rd ACM Conference on Computer and Communications Security (CCS)*. ACM, 2016.

[6] R. Matzutt, J. Hiller, M. Henze, J. H. Ziegeldorf, D. Müllmann, O. Hohlfeld, and K. Wehrle, "A Quantitative Analysis of the Impact of Arbitrary Blockchain Content on Bitcoin," in *Proceedings of the 22nd International Conference on Financial Cryptography and Data Security (FC)*. Springer, 2018.

[7] PoEx Co., Ltd. (2015) Proof of Existence. Accessed 02/04/2018. [Online]. Available: https://poex.io

[8] J. Clark and A. Essex, "CommitCoin: Carbon Dating Commitments with Bitcoin," in *Proceedings of the 16th International Conference on Financial Cryptography and Data Security (FC)*. Springer, 2012, pp. 390–398.

[9] A. Tomescu and S. Devadas, "Catena: Efficient non-equivocation via bitcoin," in *IEEE Symposium on Security and Privacy (S&P)*. IEEE, 2017, pp. 393–409.

[10] S. Nakamoto, "Bitcoin: A Peer-to-Peer Electronic Cash System," Tech. Rep., 2008, accessed 02/04/2018. [Online]. Available: https://bitcoin.org/bitcoin.pdf

[11] J. Bonneau, A. Miller, J. Clark, A. Narayanan, J. A. Kroll, and E. W. Felten, "SoK: Research Perspectives and Challenges for Bitcoin and Cryptocurrencies," in *IEEE Symposium on Security and Privacy (S&P)*. IEEE, 2015, pp. 104–121.

[12] (2016) Bitcoin Transaction Fees. Accessed 02/04/2018. [Online]. Available: https://bitcoinfees.info

[13] HugPuddle. (2013) Apertus – Archive data on your favorite blockchains. Accessed 02/04/2018. [Online]. Available: http://apertus.io

[14] A. Russell, P. Norby, and S. Bakhshi. (2015) Perceptual Image Compression at Flickr. Accessed 02/04/2018. [Online]. Available: https://code.flickr.net/2015/09/25/perceptual-image-compression-at-flickr

[15] M. Bartoletti and L. Pompianu, "An analysis of Bitcoin OP_RETURN metadata," in *Proceedings of the 4th Workshop on Bitcoin and Blockchain Research (BITCOIN)*, 2017.

[16] Namecoin. Accessed 02/04/2018. [Online]. Available: https://namecoin.org

[17] G. Wood, "Ethereum: A Secure Decentralised Generalised Transaction Ledger," 2016, accessed 02/04/2018. [Online]. Available: http://gavwood.com/Paper.pdf

[18] M. Roesch, "Snort - Lightweight Intrusion Detection for Networks," in *Proceedings of the 13th USENIX Conference on System Administration (LISA)*. USENIX Association, 1999, pp. 229–238.

[19] S. Ioannidis, A. D. Keromytis, S. M. Bellovin, and J. M. Smith, "Implementing a Distributed Firewall," in *Proceedings of the 7th ACM Conference on Computer and Communications Security (CCS)*. ACM, 2000, pp. 190–199.

[20] H. Hu, W. Han, G.-J. Ahn, and Z. Zhao, "FLOWGUARD: Building Robust Firewalls for Software-defined Networks," in *Proceedings of the Third Workshop on Hot Topics in Software Defined Networking (HotSDN)*. ACM, 2014, pp. 97–102.

[21] U. Hanani, B. Shapira, and P. Shoval, "Information filtering: Overview of issues, research and systems," *User Modeling and User-Adapted Interaction*, vol. 11, no. 3, pp. 203–259, Aug. 2001.

[22] J. D. Bruce, "The Mini-Blockchain Scheme," White paper, 2014, accessed 02/04/2018. [Online]. Available: http://cryptonite.info/files/mbc-scheme-rev3.pdf

[23] A. Chepurnoy, M. Larangeira, and A. Ojiganov, "Rollerchain, a Blockchain With Safely Pruneable Full Blocks," White paper, 2016, accessed 02/04/2018. [Online]. Available: https://arxiv.org/pdf/1603.07926

[24] A. Molina and H. Schoenfeld, "PascalCoin Version 2," White paper, 2017, accessed 02/04/2018. [Online]. Available: https://www.pascalcoin.org/wp-content/uploads/2017/07/PascalCoinWhitePaperV2.pdf

[25] G. Ateniese, B. Magri, D. Venturi, and E. Andrade, "Redactable Blockchain – or – Rewriting History in Bitcoin and Friends," in *IEEE European Symposium on Security and Privacy (EuroS&P)*. IEEE, 2017, pp. 111–126.

[26] J. Camenisch, D. Derler, S. Krenn, H. C. Pöhls, K. Samelin, and D. Slamanig, "Chameleon-Hashes with Ephemeral Trapdoors," in *Proceedings of Public-Key Cryptography (PKC)*, S. Fehr, Ed. Springer, 2017, pp. 152–182.

[27] I. Puddu, A. Dmitrienko, and S. Capkun, "μchain: How to forget without hard forks," *IACR Cryptology ePrint Archive*, vol. 2017:106, 2017, accessed 02/04/2018. [Online]. Available: http://eprint.iacr.org/2017/106

[28] "Hyena". Cryptograffiti.info. Accessed 02/04/2018. [Online]. Available: http://cryptograffiti.info

[29] G. Kessler. (2002) File Signature Table. Accessed 02/04/2018. [Online]. Available: https://www.garykessler.net/library/file_sigs.html