Ruben Recabarren* and Bogdan Carbunar

Toward Uncensorable, Anonymous and Private Access Over Satoshi Blockchains

Abstract: Providing unrestricted access to sensitive content such as news and software is difficult in the presence of adaptive and resourceful surveillance and censoring adversaries. In this paper we leverage the distributed and resilient nature of commercial Satoshi blockchains to develop the first provably secure, censorship resistant, cost-efficient storage system with anonymous and private access, built on top of commercial cryptocurrency transactions. We introduce max-rate transactions, a practical construct to persist data of arbitrary size entirely in a Satoshi blockchain. We leverage max-rate transactions to develop UWeb, a blockchain-based storage system that charges publishers to self-sustain its decentralized infrastructure. UWeb organizes blockchain-stored content for easy retrieval, and enables clients to store and access content with provable anonymity, privacy and censorship resistance properties.
We present results from UWeb experiments with writing 268.21 MB of data into the live Litecoin blockchain, including 4.5 months of live-feed BBC articles, and 41 censorship resistant tools. The max-rate writing throughput (183 KB/s) and blockchain utilization (88%) exceed those of state-of-the-art solutions by 2-3 orders of magnitude and broke Litecoin's record of the daily average block size. Our simulations with up to 3,000 concurrent UWeb writers confirm that UWeb does not impact the confirmation delays of financial transactions.

Keywords: blockchain, censorship resistance, privacy

DOI 10.2478/popets-2022-0011
Received 2021-05-31; revised 2021-09-15; accepted 2021-09-16.

1 Introduction

Basic human rights continue to be eroded around the world through fine-grained monitoring of Internet access [1, 2], and restricted access to information [3, 4] that includes select news, software artifacts [5, 6] and even research [7, 8]. Traditional solutions such as private access services, e.g., VPNs, centralize user access and were shown to keep access records [9] and share them with censors [10–12]. Tor and its onion routing and hidden services have well known limitations that simplify deanonymization not only for governments but also for some corporations [13].

In this paper we leverage the distributed nature and substantial collateral damage inflicted by blocking cryptocurrency blockchains, to develop a censorship-resistant storage system that provides private and secure data access, by embedding data into cryptocurrency transactions.

Blockchains have been advertised as a platform for distributed services [14–33], including for censorship resistance [34–37]. For instance, early efforts have used blockchains to post sensitive articles, e.g., on vaccine-related offenses of a biotechnology company, and bypass government censorship [34, 35]. The ideal blockchain-storage solution should however (1) support more frequent writing needs and larger content, and (2) rely on a blockchain that is completely distributed, and sufficiently popular to inflict unpalatable collateral damage to censors and to have a mining hashrate high enough to make it difficult to launch majority attacks.

In this paper we develop techniques to significantly extend the amount of data that can be stored on Satoshi blockchains, i.e., Bitcoin and its variants. At the time of writing, Satoshi cryptocurrencies have the highest market capitalization thus can inflict the highest collateral damage to would-be censors: the total market cap of the top 50 Satoshi cryptocurrencies exceeds $1 trillion [38].

Our quest introduces however a new set of requirements that include (1) practicality in terms of storage throughput, goodput and cost, (2) efficient retrieval and access of data stored among millions of transactions, e.g., 615 million transactions in Bitcoin [39], 60 million in Litecoin [40], and (3) satisfaction of constrains imposed by Satoshi bockchains, e.g., on the transaction size, the number of input and output counts, the transaction fees, the number of unconfirmed transactions and confirmation times. The ideal solution should also provide fully on-chain storage, to avoid the censorship and

***Corresponding Author: Ruben Recabarren:** Florida International University, Miami, FL 33199, E-mail: recabarren@gmail.com
Bogdan Carbunar: Florida International University, Miami, FL 33199, E-mail: carbunar@gmail.com

surveillance vulnerabilities introduced by hybrid storage solutions [23, 24].

In addition, non-segwit Satoshi transactions are vulnerable to *integrity attacks* (Appendix B), also known as malleability attacks [41]. Of particular concern are integrity attacks where transactions are modified within data portions unprotected by cryptographic signatures.

To simultaneously address the blockchain-writing constraints, in this paper we introduce *max-rate transactions*, data-storage constructs that use cryptocurrency transactions to prevent integrity attacks, and further optimize the writable space within the building blocks of a single transaction and the chaining of data-embedding transactions to maximize the amount of data written per time unit, thus minimize data access latency.

To enable efficient search and access of blockchain-stored data we build on max-rate transactions to introduce UWeb, the first practical, entirely on-chain storage system that efficiently organizes blockchain-stored data in directory-inspired structures. UWeb's main use case is the one-to-many distribution of popular but sensitive content, e.g., news and software.

We prove that an all-powerful monitor cannot distinguish UWeb users from regular cryptocurrency users and cannot determine what data they access. Further, blocking access to UWeb would deprive the economy of the censored region from access to a financial market whose capitalization exceeds $1 trillion.

We use max-rate transactions to write a total of 268.21 MB of content considered sensitive by many censors, i.e., BBC articles and censorship evading software, into the Litecoin blockchain. Our experiments reveal the practicality of the proposed solutions, that achieved an aggregate throughput of 183 KB/s. They also increased the average Litecoin daily block size to 206KB, breaking Litecoin's lifetime record.

Our writing experiments had no effects on the confirmation times of regular financial transactions. We further confirmed this through simulations with up to 3,000 concurrent UWeb writers (1.14GB issued in a 4 hour interval) and up to 10 times more financial transactions.

All the data written in our experiments is available for free public access in the Litecoin blockchain by inspecting the spending of the Litecoin address LZAhHQjxf6dQaTxTAK7g1wTz3hZRXX5MkG.

In summary, our contributions are the following:

– **Max-rate transactions**. We develop the first blockchain-writing constructs that, for the lowest price, and with a single input address, provably achieve a storage throughput asymptotic to the available bandwidth, and a goodput that approaches the theoretic limit. We prove that max-rate transactions are standard and prevent integrity attacks.
– **UWeb**. We introduce the first practical, entirely on-chain, secure and private storage system that leverages max-rate transactions to efficiently discover, recover and reconstruct content of interest embedded among hundreds of millions of transactions. We prove that UWeb provides users with anonymous and private access, and censorship resistance.
– **Litecoin mainnet experiments**. We show through experiments with writing more than 268MB of data in the live Litecoin mainnet (not testnet) that UWeb achieved 2-3 orders of magnitude improvements in throughput and block utilization, and wrote 4-5 orders of magnitude more data from a single funding address, when compared to state-of-the-art solutions.

2 Background and Related Work

Cryptocurrency Transactions. We model a cryptocurrency transaction as a tuple $\tau = (v, f, I_T, O_T, w, l_o)$, where v is the version number, f is a flag that indicates the presence of witness data, I_τ is a list of inputs, O_τ is a list of outputs, w an optional block of witness data, and l_o is the transaction lock time. $|\tau|$, used to denote the size of τ, is the total size of its components. The *transaction id* of τ is the double SHA256 hash of τ's concatenated components.

The list of inputs is $I_\tau = \{I_x | 0 \leq x \leq c\}$, where c is the total count of inputs used to prefix the list. An input is a tuple $I_x = (p_h, p_i, s_i, z)$. p_h is the id of the previous transaction that contains the funding output for I_τ, p_i is the output's index in the transaction with id p_h, s_i is a *script* (called the *scriptSig* script) used to verify that a user is authorized to spend the balance from transaction p_h and index p_i, and z is a sequence number related to the transaction lock time.

Similarly, the count-prefixed output of transaction τ is defined as $O_\tau = \{O_x : 0 \leq x \leq c\}$, where c is the number of outputs used as prefix for list, and an output O_x is a tuple $O_x = (s_o, o_v)$. s_o is the *scriptPubKey* script, and o_v is the output value to be transferred from the sum of values specified in the transaction τ list of inputs. A transaction τ is invalid if the sum of the values from the inputs is smaller than or equal to the sum of the outputs. The balance after subtracting the output values is considered to be the *miner fee*. We note that a zero-fee transaction will not be mined by modern pools [42] and it may not be broadcast by default configured Satoshi-compliant nodes [43].

A transaction that has not yet been mined into a block is said to be *unconfirmed*. Further, a pair of transactions where one transaction spends the other's value are said to be *chained*. Using this basic knowledge of Satoshi networks, previous efforts have designed sub-optimal or insecure data insertion solutions that we summarize next.

Apertus. Kaminsky [44] proposed the first blockchain-writing solutions, that use the output address bytes in the *scriptPubKey* to store data: the Pay-to-PubkeyHash (p2pkh) and Pay-to-Script-Hash (p2sh) techniques. Apertus [45] uses p2pkh writing that overwrites the 20 bytes of a destination address to store arbitrary data.

Catena and Blockstack. Catena [25] and Blockstack [23] introduce inexpensive solutions for small payloads. These systems use an OP_RETURN based writing, to mark a transaction as invalid, and output unspendable transactions that are immediately prunable from the un-spent transaction set. This contract allows for writing 80 bytes after the OP_RET opcode. Since standard transactions can only carry one OP_RET output [46], it is impossible to improve the efficiency of this construct, thus limiting Catena and Blockstack to sub-optimal blockchain utilization.

Blockchain-Based Censorship Resistance. Early attempts have written a few sensitive articles (e.g., on vaccination misbehaviors of a biotech company) on the Ethereum blockchain to avoid censorship [34, 35]. More rigorous, academic efforts include Tithonus [36] and MoneyMorph [37]. Tithonus [47] provides solutions that allow censored clients to surreptitiously embed paid requests for sensitive content into Bitcoin transactions and also for uncensored services to redeem the payment when they embed the requested content into transactions. MoneyMorph [37] further designs rendezvous protocols over the blockchains of commercial cryptocurrencies to bootstrap censorship resistant communications. When compared against Bitcoin, Monero and Ethereum, Zcash provides MoneyMorph with the best bandwidth per transaction and the lowest cost.

Table 2 in § 5.5 provides details of the comparison of the max-rate transactions that we develop in this paper against state-of-the-art blockchain-writing solutions (Apertus, Catena, MoneyMorph and Tithonus). In addition, our solutions embed 4.6MB in the blockchain in a single mining event, improving significantly over MoneyMorph that embedded 20 bytes into a Bitcoin transaction, 20 bytes in Ethereum, 640 in Monero and 1,148 in a Zcash transaction, and also Tithonus that was able to embed 1,635 bytes into a Bitcoin transaction. Max-rate transactions thus achieve a storage throughput that improves on existing solutions by 3-4 orders of magnitude, and a blockchain utilization that improves on existing solutions by 2-4 orders of magnitude. In addition, max-rate transactions address the Tithonus vulnerability to integrity attacks.

Further, UWeb improves on existing blockchain-writing solutions by providing novel, on-chain-only techniques to organize and update content stored in the blockchain for efficient retrieval, and to access content with provable privacy and censorship resilience.

Staged Transactions. Unlike previous solutions that overload portions of the output scripts, *staged transactions*, documented by Todd [48], use the inputs section, the largest field in typical transactions. Unlike output writing where only one transaction is needed, input overloading requires the use of a pair of transactions: A *funding* transaction with a p2sh output that specifies the hash of its redeeming script, and a *spending* transaction, whose input script provides a *redeemScript* that satisfies the funding transaction's conditions, and stores the actual arbitrary data. Tithonus [47], a censorship resilient system, introduces variations of this technique, that are however vulnerable to output and input script modification attacks, see § 3.2. In § 5.5 we show that our proposed blockchain-writing techniques are significantly more efficient than state-of-the-art solutions.

Blockchain Based Services. Blockchains have been used to store arbitrary user data [17–19, 31, 33], sensitive data [21], including medical records [22], to provide a decentralized PKI system [23, 24], data provenance for clouds [26], a privacy preserving domain name system [32], data integrity assurances for cloud-based IoT [27, 28], to implement content sharing and access control for social networks [29], and to secure the BGP and DNS infrastructure [30].

In this paper we develop a censorship-resistant blockchain-based storage service that uses decentralized blockchains to avoid single points of censorship [14–16]. Further, to avoid majority attacks [49], e.g., through opportunistic renting of mining equipment using sites like nicehash [50], we build UWeb on Satoshi blockchains, thus we improve on solutions based on custom-made blockchains [33] or blockchains that have small gossip networks and mining infrastructure [17–19].

Solutions like Blockstack [23, 24] implement a hybrid blockchain/cloud approach, where the data is stored in traditional cloud storage, e.g. Amazon S3, and only a cryptographic pointer to this data is stored in the blockchain. UWeb instead stores all data on-chain, including metadata, e.g., directory structure, public key certificates. This endows UWeb with resistance to censorship and privacy compromise, since an adversary can no longer obtain cooperation from the cloud provider or correlate blockchain and cloud accesses.

3 Model and Problem Definition

We first describe the UWeb ecosystem then define the adversary model and the problems that we seek to solve.

3.1 System Model

We consider the model illustrated in Figure 1, where clients that are censored and under surveillance need to access sensitive content posted by publishers who are outside the censored area. For this, we define a storage system UWeb $= (\mathcal{B}, \mathit{ClientSetup}, \mathit{Store}, \mathit{Access})$ built over a blockchain $\mathcal{B}$ that consists of functions to setup client functionality, and store and access content.

Content publishers use the *Store* function to embed data in the fields of cryptocurrency transactions that are persisted in the blockchain $\mathcal{B}$ at the cost of mining them. Consumers use the *Access* function to access stored content for free. While UWeb's reliance on cryptocurrency transactions can be leveraged to enable content publishers, content consumers and combinations thereof to pay for the storage of data, the specific payment arrangements are outside the scope of this paper.

Data can be stored in a single transaction or in multiple transactions. We view the blockchain $\mathcal{B}$ to be an ordered set of transactions of the simplified form (i, d, u), where i is the index of the transaction, d is the transaction content, and u is a boolean that specifies whether this is a financial transaction $(u = 0)$ or a data-storing transaction $(u = 1)$.

We assume that any publisher and consumer has control over her computer and can install arbitrary software, including a cryptocurrency node, the cryptocurrency reference software, and the UWeb client that we develop. We assume that developed software can only communicate through the gossip network and the blockchain. While other communication media would simplify the solution design, they often leak sensitive information to the provider [51, 52], and can be vulnerable to denial of service and insider attacks. While we assume that software cannot access off-chain storage services, we assume that it can access basic services, e.g., routing, DNS, certification authorities, and other cryptocurrency nodes.

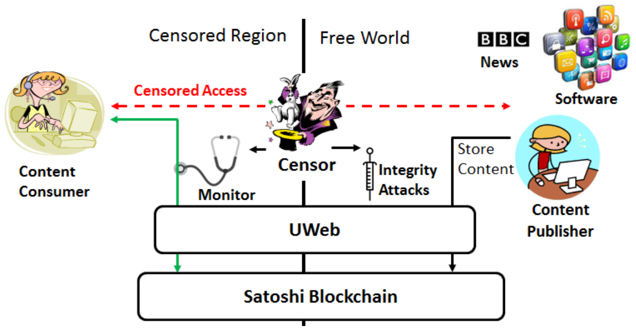

Fig. 1. System and adversary model. A client in the censored region cannot directly access sensitive services and news. Instead, content publishers embed content on commercial blockchains; clients access data with privacy and anonymity through standard Satoshi clients. The adversary can censor select client communications, monitor communications and perform integrity attacks.

Standard Transactions: Blockchain-Writing Constraints. Transactions used by blockchain-writing solutions need to be *standard* in order to be relayed through the gossip network, and are eventually mined into the blockchain. For this, they need to satisfy several restrictions imposed by Satoshi blockchains [53]. We now document the most relevant restrictions.

Restriction 3.1. (Size restrictions) The maximum size of a transaction is 100KB [54]. The maximum size of a block is 1MB. The individual scripts of an input (output) of a standard transaction cannot exceed a total size of 1,650 KB [55]. Script primitives to operate on data are limited to 512 bytes [56].

Restriction 3.2. (Input/output count) Standard transactions can have a theoretical maximum of 2^8 inputs and outputs. For P2SH, the most common transaction type, the size of a transaction input is 108B and the size of an output is 34B. Assuming one input and given the above maximum transaction size, the maximum number of outputs in a P2SH transaction is 2,937.

Restriction 3.3. (Confirmation time) Issued transactions need to wait to be confirmed to be mined into a block. Thus, the data-writing process needs to wait one *mining epoch* (on the order of minutes) for a segment of data to be confirmed on a block.

Restriction 3.4. (Unconfirmed chains) The length of a chain of unconfirmed transactions (§ 2) cannot exceed 25 transactions. The total size of the chain of unconfirmed transactions cannot exceed 101KB [57].

Restriction 3.5. (Minimum transaction fees) The current minimum relay fee rate is 10^{-8} LTC per byte (resp. BTC) for the Litecoin (resp Bitcoin) systems.

3.2 Adversary Model

We consider an adversary who seeks to monitor the access to content of users in a certain region, and even to prevent accesses to content considered sensitive, see Figure 1. In the following we first detail the monitoring adversary, then the censoring one.

In the following, we say that a network communication Γ is possible when accessing transaction $T_i = (i, d_i, u_i)$ if $P(\Gamma|T_i) > 0$, i.e., the conditional probability that UWeb performed communication Γ with the cryptocurrency network to access data d is positive.

We consider adversaries that are able to monitor the communications of UWeb users and other Satoshi nodes, and arbitrarily inject, delete, and reorder messages. We assume however that the adversary cannot decrypt encrypted content or forge signatures, without knowledge of the decryption and signature generation keys, respectively. We assume that the adversary can run any number of clients and services, including of UWeb, and can publish content or access content published by others.

Private Network Access (PNA) Game. We say that a communication system that uses blockchain $\mathcal{B}$ provides *private access* in terms of network communication, if given a pair of transactions T_{i_0}, T_{i_1} in $\mathcal{B}$, the adversary can not identify the transaction id choice of the UWeb system with probability significantly greater than that of a random guess (1/2). Formally, we want to demonstrate that any probabilistic polynomial time (PPT) adversary $\mathcal{A}$ has only a negligible advantage over random guessing on the following security game against a challenger $\mathcal{C}$ that accesses data stored on $\mathcal{B}$.

- $\mathcal{C}$ installs required software (e.g., Satoshi client) and sets up functionality as described in § 4.3. $\mathcal{A}$ stores the blockchain $\mathcal{B}$ to serve its content over the cryptocurrency p2p network, upon request.
- The adversary may perform a polynomially bounded number of operations to select 2 indices i_0, i_1 of transactions in blockchain $\mathcal{B}$.
- At a desired time, $\mathcal{A}$ sends to $\mathcal{C}$ the indices i_0 and i_1, where $T_{i_0} = (i_0, d_0, u_0) \in \mathcal{B}$, $T_{i_1} = (i_1, d_1, u_1) \in \mathcal{B}\}$.
- $\mathcal{C}$ picks a bit $b \in_R \{0, 1\}$ uniformly at random. $\mathcal{C}$ performs communication Γ to access transaction T_{i_b} from $\mathcal{A}$, then signals completion.
- $\mathcal{A}$ performs additional computations and outputs bit b', his prediction of the bit b chosen by $\mathcal{C}$.

The advantage of $\mathcal{A}$ in this game is:

$$\mathbf{Adv}_{\text{UWeb}}^{\text{PNA}}(\mathcal{A}) = |P(b' = b) - P(b' \neq b)|$$

We then introduce the following definition:

Definition 3.1. (Private Network Access) A blockchain-based censorship resistant system provides private network access if there exists no PPT adversary with a non-negligible advantage in the PNA game.

Anonymous Network Access (ANA) Game. We say that a communication system that uses blockchain $\mathcal{B}$ provides *anonymous access* in terms of network communication, if given a set of transaction ids $[i] = \{i \mid T_i = (i, d_i, u_i) \in B\}$, the adversary can not determine if the consumer is using UWeb. Formally, we want to demonstrate that any probabilistic polynomial time (PPT) adversary $\mathcal{A}$ has only a negligible advantage over random guessing on the following security game against a challenger $\mathcal{C}$ that chooses whether to use UWeb to access data stored on blockchain $\mathcal{B}$.

- $\mathcal{C}$ installs required software (e.g., Satoshi client) and sets up functionality as described in § 4.3. $\mathcal{A}$ stores the blockchain $\mathcal{B}$ to serve its content over the cryptocurrency p2p network protocol upon request.
- The adversary may perform a polynomially bounded number of operations to select $m + n > 0$ indices $i \in \{1 \dots m + n\}$ of transactions $T_i = (i, d_i, u_i)$ in blockchain $\mathcal{B}$ such that m transactions have bit $u = 1$ and n transactions have bit $u = 0$.
- At a desired time, $\mathcal{A}$ sends to $\mathcal{C}$ the set of $m + n$ indices: $[i] = \{i \mid i \in 1..m + n, T_i = (i, d_i, u_i) \in \mathcal{B}\}$.
- $\mathcal{C}$ selects a bit $b \in_R \{0, 1\}$ uniformly at random. $\mathcal{C}$ then performs communication Γ to collect all transactions $T_i = (i, d_i, u_i = b)$ from $\mathcal{A}$ (either m or n of them) and signals completion.
- $\mathcal{A}$ performs additional computations and eventually outputs b', his prediction of the bit b chosen by $\mathcal{C}$.

We define the advantage of adversary $\mathcal{A}$ in the anonymous network access game to be:

$$\mathbf{Adv}_{\text{UWeb}}^{\text{ANA}}(\mathcal{A}) = |P(b' = b) - P(b' \neq b)|$$

We introduce then the following definition:

Definition 3.2. (Anonymous Network Access) A blockchain-based censorship resistant system provides anonymous network access if there exists no PPT adversary who has a non-negligible advantage in the above ANA game.

Claim 1. *A blockchain-based censorship resistant system that provides private network access also provides anonymous network access.*

Proof. Let us assume that there exists a PPT adversary $\mathcal{A}$ with advantage ϵ in the anonymous network access security game (Definition 3.2). Then, we use that adversary against the challenger in the private network access security game (Definition 3.1) to achieve the same advantage ϵ against that challenger. Formally, we create a new adversary $\mathcal{A}'$ that acts as a challenger with $\mathcal{A}$ in an anonymous access game, and acts as adversary against a challenger $\mathcal{C}$ in a private access game:

- $\mathcal{A}$ sets $m = 1$ and $n = 1$, and chooses transactions T_{i_0}, T_{i_1} such that $u_{i_0} = 0$ and $u_{i_1} = 1$. $\mathcal{A}$ sends i_0 and i_1 to $\mathcal{A}'$.
- $\mathcal{A}'$ forwards them to $\mathcal{C}$, then acts as a proxy between $\mathcal{A}$ and $\mathcal{C}$, enabling $\mathcal{C}$ to fetch blockchain $\mathcal{B}$ from $\mathcal{A}$.
- $\mathcal{C}$ picks a bit b randomly then accesses TX T_{i_b}.
- $\mathcal{A}$ sends to $\mathcal{A}'$ his guess bit b'. $\mathcal{A}'$ forwards b' to $\mathcal{C}$.

The advantage of $\mathcal{A}'$ in the private network access game is $\mathbf{Adv}_{\mathrm{UWeb}}^{\mathrm{PNA}}(\mathcal{A}') = |P(b' = b) - P(b' \neq b)|$. The advantage of $\mathcal{A}$ in the anonymous network access game with $\mathcal{A}'$ is also $\mathbf{Adv}_{\mathrm{UWeb}}^{\mathrm{ANA}}(\mathcal{A}) = |P(b' = b) - P(b' \neq b)|$, which by definition is ϵ. Thus, we have build an adversary $\mathcal{A}'$ that has advantage ϵ in the private network access game. □

Censoring Adversary. We further consider a censoring adversary who attempts to prevent data retrieval from the blockchain for a set of victim participants, e.g., located within a certain geographic region governed by the censor. For this, the adversary will specifically target the communications required to store and retrieve blockchain data.

However, we assume that the censor is not willing to block or significantly hinder cryptocurrency use inside the censored area, due to the associated collateral damage [58, 59]. For instance, China ranks first in the world on the "activity of non-professional, individual cryptocurrency users, based on how much cryptocurrency they are transacting compared to the wealth of the average person" [60], while Russia ranks second in the world in trading volume in Bitcoin [61].

We also assume that the censor controls gossip network nodes, mining nodes, and even mining pools. Such a censor can then further launch output and input script modification attacks, a.k.a., sniping and integrity attacks [62] (see Appendix B), which effectively corrupt data embedded in transactions that have not yet been mined. These attacks duplicate victim transactions, modify the duplicates at will, then rush the fabricated transactions in the p2p network in an effort to have them mined before the original transactions.

However, since a majority attack would damage the trust in the financial aspect of any cryptocurrency, we assume that the censor does not control more than half of the mining power of the network. Recent research [63] has shown that no country or mining pool controls 51% of the mining power in Bitcoin. Further, we assume that the censor cannot block the access of honest clients, to all the nodes with access to an honest pool. We now introduce the following definition:

Definition 3.3. (p-Cap Collateral Damage) We say that a blockchain-based publishing system imposes a p-cap collateral damage if the above defined censor who seeks to prevent well-behaved publishers from posting desired content and consumers from accessing published content, will deprive the economy of the censored region of access to a financial market with a capitalization value of p.

3.3 Problem Definitions

We consider the problem of providing efficient storage of information such that access to the data is hard to monitor or censor. Developed solutions need to satisfy blockchain-imposed restrictions (§ 3.1) and optimize several relevant metrics, which we now define. We use the term *construct* to refer to the solution's basic unit of storing data, e.g., transaction or group of transactions.
Metrics. We call the *throughput* of the solution to be the size of data made available on the blockchain (gossip) network by the solution per time unit, and *goodput* to be the ratio of size of payload, i.e., data embedded in a construct, to the total size of the construct. Finally, we define the *cost* of the solution to be the price per byte of stored data, i.e., the total size of the data-storing construct multiplied by the transaction fee, divided by the payload size.

We now define the data storage problem:

Definition 3.4. (Optimal Blockchain Storage) Given content C of size N and a single funding address F, develop a transaction construct and an optimal organization of such constructs that embed content C and are funded from F. The constructs should be standard (§ 3.1), maximize storage throughput and goodput, minimize the storage cost, and be secure against integrity attacks (Appendix B).

Further, we define the data access problem:

Definition 3.5. (Private, Censorship Resistant, Efficient Access) A censored client needs to efficiently access content stored among hundreds of millions of transactions, with private and anonymous network access (Def. 3.1 and 3.2) and censorship resilience (Def. 3.3).

4 The UWeb System

In this section we first introduce blockchain storage solutions that satisfy Definition 3.4: techniques to insert data into standard Satoshi transactions, and indexing methods that ensure the scalability of writing large content. Second, we leverage these solutions to build the UWeb storage system defined in § 3.1 that satisfies Definition 3.5.

Our system model (§ 3) assumes publishers on the censor-free region. This enables us to develop constructs that tradeoff detectability for a significant increase over state-of-the-art in the amount of data that can be stored in a single transaction and between mining events.

4.1 Max-Size Data Storing Script

To satisfy Definition 3.4 we first develop input script-writing solutions that achieve optimal script utilization, i.e., maximize input goodput, and simultaneously prevent the input/output modification attacks of § 3.2 and satisfy the blockchain restrictions for standard transactions. We observe that the blockchain restrictions for standard transactions imply that optimal script utilization allows for a maximum of 3 large push operations (1,650 B / 520 B) and a few extra bytes per input script.

Algorithm 1 shows the proposed smart contract that achieves optimal utilization of the available storage space on an input script under these restrictions and also prevents the transaction integrity attacks described in Appendix B. Similar to Todd's technique [48], we use hash lock constructs (lines 7-9) to protect the data pushed (lines 2-4). However, unlike Todd's technique, we do not sacrifice one of the data push operations (lines 2-4) to include a public key and a signature verification on the redeeming script (lines 5-9). We do not need this verification because our outputs are simple OP_RET outpoints (right of Figure 2). Thus, the maximum storage capacity per input script is 1,568 bytes and after accounting for the overhead, the typical size of a funding/spending transaction pair is 1,703 bytes. We analyze the security that this technique provides against input and output modification attacks, in § 4.4.

Algorithm 1 Smart contract that maximizes data storage using 1,650 B of space available to transaction scripts. The contract pushes 3 chunks of 512 bytes, validated in the redeeming script via hash comparisons. Only the last hash validation pushes a TRUE value to the stack as mandated by the latest standard transaction rules [56].

```
1.P2SH.scriptSig(data)
2.    OP_PUSHDATA2 data.getNext(520)
3.    OP_PUSHDATA2 data.getNext(520)
4.    OP_PUSHDATA2 data.getNext(520)
5.    OP_PUSHDATA1 % push next 4 lines, 79 bytes
6.    OP_PUSH data.getNext(8) OP_DROP
7.    OP_HASH160 OP_PUSH data.getNext(20)
          OP_EQUALVERIFY
8.    OP_HASH160 OP_PUSH data.getNext(20)
          OP_EQUALVERIFY
9.    OP_HASH160 OP_PUSH data.getNext(20)
          OP_EQUAL
```

4.2 Data Storage Solutions

We now leverage the above data storing script, to introduce *max-rate transactions*, the first practical, transaction-based data-storing construct that satisfied Definition 3.4). Given a single input address with sufficient funds, max-rate transactions achieve a throughput asymptotic to the available bandwidth, entirely through a Satoshi blockchain (gossip) network, at a minimum cost rate, satisfy the blockchain restrictions of § 3.1, and prevent the transaction integrity attacks of Appendix B.

To achieve this, we observe that factoring out the transaction overhead, a basic data-storing transaction can handle 59 data storing, spending input scripts (i.e. 100KB / 1,650 B, since the maximum size of a transaction is 100KB) while funding transactions can handle up to 2,937 outputs, with one output reserved for balance change (according to restriction 3.2, § 3.1). Figure 2 shows the optimal max-rate transaction construct that allows funding outputs to create spending inputs in a web of transactions that minimizes storage overhead. To optimize storage, we group the maximum amount of inputs into one transaction, thus reduce the overhead of using several transactions for this purpose.

The first issued, *max-rate funding transaction* (left of Figure 2) bundles funding outputs Out Addr 1,2,..., n, where the maximum n is 2,936 + 1 change output. Each output is constructed with an address that references the redeeming script of the corresponding payload-storing input (Fat In Script 1,2,..., n) inside a *max-rate spending transaction* (right of Figure 2). As described above, each spending transaction can have a maximum of 59 such payload-storing inputs. A max-

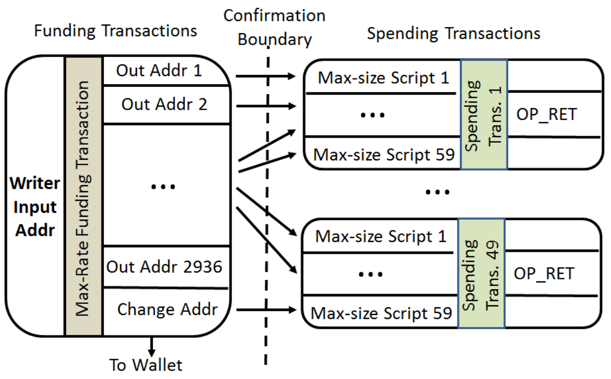

Fig. 2. Optimal data storage construction creates a web of funding and spending transaction that minimizes storage overhead. This basic construct allows for storing maximum file sizes of 4.6 MB in one confirmation epoch.

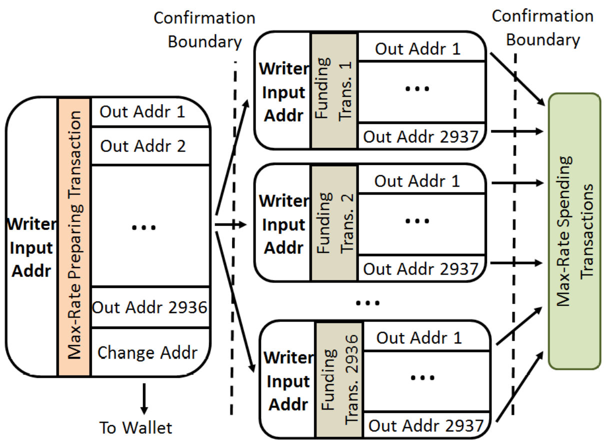

Fig. 3. Indexing technique that allows the linking of related funding transactions. We store metadata as regular transaction input/outputs in different levels of indirection. The last level consists of the spending transactions that store the actual data (see right side of Figure 2). Transactions between consecutive confirmation boundaries, can be sent simultaneously up to the maximum block size, since all their inputs were already confirmed in the previous epoch.

rate funding transaction can thus fund 49 (= 2,937 / 59) completely full, fat spending transactions and 1 more spending transaction with 46 inputs. Therefore, the largest amount of data that can be written with this construct (i.e., 1 funding and 50 spending transactions) is 4.6MB (2936*1568).

The max-rate funding transaction and subsequent spending transactions are separated by a confirmation boundary that represents a new block mining event and its respective waiting time. That is, the funding transaction needs to be mined and confirmed before we issue the spending transactions. This is imposed by the blockchain requirement that chains of unconfirmed transactions do not exceed 101KB in size (restriction 3.4, § 3.1). Thus, every funding output is already confirmed before any spending input enters the network. This ensures minimum wait time for payload storing transactions, since they are no longer constrained by the maximum chain of unconfirmed transactions. We further discuss this in § 4.4.

Storing Large Data. Providing access to large amounts of stored data requires a method for linking together funding transactions that point to related data-storing inputs. The left side of Figure 3 shows the *preparing transactions*, a tree-like construct that recursively uses funding transactions to fund other funding transactions. Each level of confirmation boundary multiplies the maximum file size storage by a factor of 2,936: the maximum degree of each *root* node in this tree is 2,936. Thus, approach provides an exponential increase on stored data size, for a linear increase in waiting time.

4.3 UWeb

We now leverage max-rate transactions to build the UWeb storage system defined in § 3.1 and provide private, censorship resistant and efficient access to data stored in a blockchain among millions of transactions. To organize stored content, UWeb defines several content-storing transaction-based entry types, illustrated in Figure 4: *DATA entries* are transactions that store compressed content organized in files, and *DIR entries* organize information about directories and/or files. DIR entries point to either DATA entries or recursively to other DIR entries. DIR entries also store content meta-data, e.g., whether they point to a file or a directory, and a signature generated by the publisher over the entry's previous fields, to provide integrity. In the following we define the UWeb functions of § 3.1.

ClientSetup Function. To set up a content publisher, the *ClientSetup* function creates a root directory (denoted rootDIR), that contains public identifying information (INIT entries), including, e.g., public key certificates of the publisher, sub-directories (DIR entries) for other content that include file pointers, publisher metadata and data (DATA entries), e.g., web content.

Content Publishing. The *Store* function takes as input a directory name *Dir*, a file name *Fname* and content *data*, and uses the DIR and DATA entries defined above to store the content. Specifically, *Store* first uses the max-rate transaction constructs of § 4.1 to write *data* to the blockchain, see the DATA-labeled blocks in Figure 4. Let *Tid* be the Id of the first transaction storing *data*. Then, if an OP entry exists for directory *Dir* UWeb follows the chain depicted in Figure 4 until it finds an un-spent output. Otherwise, it creates a new

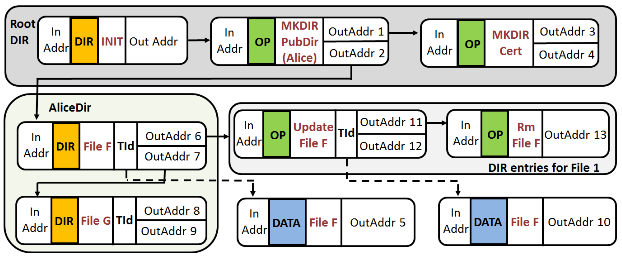

Fig. 4. UWeb directory organization for publisher Alice. RootDIR points to AliceDIR and pubkey certificate (Cert) subdirs. DIR entries point to either the first DATA transaction that stores content, or to other DIR entry transactions. AliceDir contains two files (F and G). OP entries are chained to record all operations performed on a file (e.g., Update File F, Remove File F).

OP entry for directory *Dir* and uses one of its outputs. Once such an un-spent output is chosen, the *Store* function creates a DIR entry with the FILE directive and writes in it the *Fname* parameter provided as metadata and the above *Tid*, see Figure 4.

All UWeb transaction entries may make use of multiple input addresses in order to ensure the ability to chain these entries even when an output address runs out of enough funding. However, output addresses consist of 1) one OP_RET output with directives and associated metadata and 2) one or more P2SH outputs for chaining. Because OP_RET outputs are limited to 80 bytes only, metadata larger than 80 bytes can be stored chaining more DIR INIT entries using the remaining P2SH outputs. In order to mark the end of a metadata chained in this way, we use the same variable length integer format used in Bitcoin [64]. All DIR entries should reserve one P2SH un-spent output in order to continue the chain of directives.

Update: Content Logs. The append-only nature of the blockchain makes it challenging to update this directory structure, e.g., to update or remove content. Inspired by log-structured file systems [65–67], the *Store* function defines *OP entries*, to record information about operations performed on files and directories, and appends them to create *OP chains*, using Catena [25] chaining. Figure 4 illustrates OP chains, where the first output address of the DIR transaction of a directory/file initiates a chain of all the operations performed on the directory/file. To add to or modify File F (see Figure 4), the *Store* function first writes the new content (the whole file or only deltas) to a new set of DATA entries, and stores them on the blockchain (see § 4.2). It then generates an OP entry that embeds the Id of the first transaction that stores this content (i.e., a funding or preparing transaction, see § 4.2), and funds this transaction from an output address of the transaction that stores the previous OP entry in the chain (i.e., OutAddr 6, or OutAddr 11 in Figure 4).

UWeb Access. The *Access* function takes as input either a UWeb directory and file name, or a transaction ID. It uses a Satoshi reference client to access the raw underlying cryptocurrency blockchain, for instance [68]. UWeb has shared access to the internal data storage of the reference client so that its operation is effectively local. As a consequence, the network fingerprint of UWeb content consumers is indistinguishable from a vanilla cryptocurrency user.

To discover content the first *Access* function call needs to download the entire blockchain. Once the entire blockchain history is available, UWeb scans all transactions looking for DIR entries of different publishers and traverses the structure depicted in Figure 4. In order to do this, the client looks for OP_RET outputs that start with the DIR INIT tag ($0x44495220494E4954$ in hex) and reads off origin metadata including a public key certificate of the publisher. After validating this certificate, downstream content in this data structure can be verified with strong cryptographic assurances.

As UWeb discovers new content, either by traversing the entire blockchain or processing only new transactions, it builds an external database for content random access. When a user searches for content of interest, data is searched in this external database. Thus, accessing content is completely decoupled from the Satoshi reference client operation.

4.4 Analysis

Claim 2. *The throughput of max-rate transactions is asymptotic to the available network bandwidth.*

Proof. The expected throughput $R(N)$ corresponding to a given size of payload N is given by $R(N) = \frac{N}{w \times E + N/B}$, where E is the number of confirmation epochs (i.e., mining events) required to generate the max-rate transaction, w is the expected wait time for a confirmation event (e.g., $w = 150$ secs for the Litecoin network) and B is the upload bandwidth available to connect to the gossip network. Further, $E = 1 + \frac{N}{p \times F}$, where p is the maximum payload size in a script and F is the maximum number of funding outputs that fit in one block. For our max-rate transaction, $p = 1,568$ and $F \approx 29,370$. For a 46MB payload and a 1Gbs upload bandwidth in our lab, the expected throughput is 154KB/s. Further, the throughput is hyperbolic with respect to the payload size. Thus, when $N \to \infty$, $R(N) \to B$, the available upload bandwidth. □

Claim 3. *The goodput of max-rate transactions approaches theoretical limit.*

Proof. For content of size N, the number of spending transactions is given by $Q(N) = \frac{N}{p \times m}$, where m is the maximum number of max-size inputs (§ 4.1) that fit in a transaction. For Bitcoin/Litecoin, $m = 100\text{KB}/1,720 \approx 59.5$. The number of funding transactions is given by $L(N) = \frac{N}{p \times f}$, where f is the maximum number of funding outputs that fit in a transaction. For Bitcoin and Litecoin, $f = 2,937$. Thus, the total size of max-rate transactions is $S(N) = ts \times \left(\frac{N}{p \times f} + \frac{N}{p \times m}\right)$, where ts is the maximum size of a transaction. For Litecoin, $ts = 100\text{KB}$. Then, the goodput of a max-rate transaction is $G(N) = \frac{N}{S(N)}$. □

Claim 4. *Max-rate transactions are standard.*

We include the proof in Appendix A. Further, in Appendix A we also show that max-rate transactions prevent input and output modification attacks. Then, given that our blockchain-writing constructs are standard, maximize throughput and goodput and are secure against input and output modification attacks, we conclude that max-rate transactions satisfy the optimal blockchain storage problem (Def. 3.4).

Claim 5. *UWeb provides private network access.*

Proof. Following the private network access game (Definition 3.1), UWeb first downloads the entire blockchain before accessing all data-containing transaction T_i (see § 4.3). Thus, the only possible communication Γ is the entire log of transactions in blockchain $\mathcal{B}$ until all T_i have been observed. Therefore, $P(\Gamma = \mathcal{B} \mid b) = P(\Gamma = \mathcal{B}) = 1$ for all of $\mathcal{C}$'s choices of b, i.e., no matter the choice of b, $\mathcal{A}$ always observes the same communication. This implies that knowledge of Γ does not provide $\mathcal{A}$ with an advantage in the choice of b. Formally, for any PPT $\mathcal{A}$, the conditional probability of guessing $\mathcal{C}$'s b value given communication Γ is, according to Bayes' theorem, $P(b' = b \mid \Gamma = \mathcal{B}) = P(\Gamma = \mathcal{B} \mid b) \times P(b)/P(\Gamma = \mathcal{B})$. Since the choice of b is random (see the PNA game in § 3.2), $P(b) = 1/2$, thus $P(b' = b \mid \Gamma = \mathcal{B}) = 1 \times (1/2)/1 = 1/2$. Thus, we have that for any adversary $\mathcal{A}$:

$$\begin{aligned}\mathbf{Adv}^{PNA}_{UWeb}(\mathcal{A}) &= |P(b' = b) - P(b' \neq b)| \\ &= |P(b' = b) - (1 - P(b' = b))| \\ &= |2 \times P(b' = b \mid \Gamma = \mathcal{B}) - 1| = 0\end{aligned}$$

□

UWeb also provides anonymous network access: due to Claim 1, since UWeb provides private network access it also provides anonymous network access.

UWeb Provides p-Censorship Resilience. First, we observe that since max-rate transactions are standard (see Claim 4), nodes following the protocol will broadcast them in the p2p network and the consensus protocol will eventually add them permanently to the blockchain. Further, a censor cannot use input and output attacks to prevent a publisher outside the censored area from using UWeb to distribute information and persist it in the blockchain, see above discussion.

A censor who seeks to block access to the whole blockchain will trivially deprive the entire censored area of access to the market of the corresponding cryptocurrency. A censor who seeks to selectively filter certain blocks, e.g., containing transactions that embed UWeb content, would generate a split universe with global impact on the consensus protocol, thus will eventually similarly deprive the entire censored area of access to the market of the corresponding cryptocurrency. Given that UWeb can be ported to all Satoshi compliant cryptocurrencies, a censor that wants to block UWeb needs to block all Satoshi compliant cryptocurrencies, thus depriving its economy from access to a market capitalization of p = \$1 trillion (see Definition 3.3).

5 Evaluation

In this section we experimentally evaluate max-rate transactions. We first detail our experimental setup, then reveal details of a longitudinal storage of BBC news and a stress-test of the Litecoin blockchain. We performed our experiments on the Litecoin blockchain, that required thus real money. All script constructs proposed in this paper were accepted by their networks without any modification, as expected for standard transactions.

In the following we use two more metrics, in addition to the ones introduced in § 3.1. First, the *block space utilization* of a blockchain-writing solution, defined to be the ratio of the payload size in a block, to the total size of the block. Second, the *block transaction utilization* of a blockchain-writing solution, defined to be the ratio of the number of payload carrying transactions to the total number of transactions in a block.

5.1 Experimental Setup

The Max-Rate Toolset. We implemented a suite of python scripts (1,765 loc) for creating, sending, reading, and monitoring max-rate transactions, that avoided using Litecoin library dependencies as we required the creation of highly customized smart contract constructs. We created a different set of python and bash scripts (118 loc) for monitoring and analyzing mempool statistics, and for scraping and processing journalistic data for its eventual publishing.

Instrumenting the Litecoin Network. In order to evaluate the impact of max-rate transactions over normal blockchain operations, we instrumented the Litecoin network to measure the size of the Litecoin mempool and the expected time to confirmation for new unconfirmed transactions. Specifically, we created a script that queried the mempool of a node in our lab every 5 seconds. After each query, we recorded timestamps, size, fee and block height for newly received transactions. Further, every 10 minutes, another script scanned the collected list looking for transactions with 1 confirmation, and calculated the time required for them to make it to the blockchain.

Equipment. For our experiments we used two virtual machines with 8 core Intel(R) Xeon(R) Gold 6126 CPU @ 2.60GHz, 8 GB of RAM and 500 GB of hard drive each. We also deployed our scripts on a Raspberry PI 3B+ with an ARM Cortex-A53 CPU @ 1.4GHz, 1 GB of RAM and 32 GB SDHC card, and the Raspbian OS. We downloaded the entire Litecoin blockchain on the PI and validated the execution of the max-rate toolsuite for payloads up to 40 MBs.

5.2 Ethical Considerations

UWeb Impact on Full Nodes. UWeb provides privacy access assurance for arbitrary data inserted in the blockchain. This may suggest that this will inevitably lead to an excessive bloat of the blockchain, and will act as a counterincentive for full nodes to continue to participate. We note however that UWeb imposes costs on writing efforts, thus max-rate transactions are equivalent to a surge in popularity of financial transactions with a minimum transaction fee rate. Dealing with such an event was considered by the Satoshi design; the following experiments show that it works as expected. Thus, this problem is within the scope of the natural evolution of cryptocurrency systems.

UWeb Impact on Financial Transactions. Our experiments in the Litecoin realnet (§ 5.4) and simulations (§ 5.6) reveal that UWeb imposes no delays on the confirmation times of financial transactions.

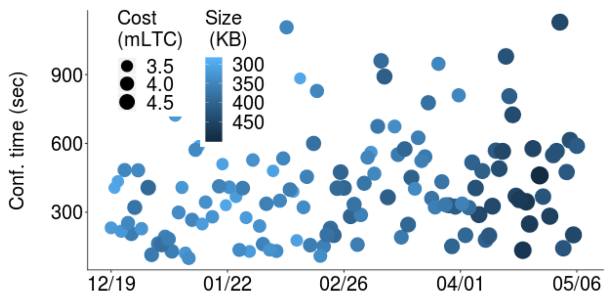

Fig. 5. Writing BBC articles daily on the Litecoin blockchain. The cost of writing does not impact the time to first confirmation.

Objectionable Content. UWeb may seem to enable the distribution of objectionable content in an unbounded fashion throughout the blockchain. We note however that UWeb is not the ideal tool for posting objectionable content since it does not provide writers with privacy or anonymity (UWeb provides access privacy/anonymity only for readers). Objectionable content already exists in blockchains, and was written with other, much less efficient but more private writing solutions, e.g., overwriting the transaction value field [69].

We note that promising research on *redactable blockchains* has been conducted by Deuber et al. [70], and also [71–73]. We envision a system where it is cheaper for a majority to delete undesirable content than for a minority to keep re-introducing it. Such a system would self-regulate the impact of arbitrary data insertions on the overall blockchain ecosystem. We acknowledge however that (1) content could be written encrypted, for valid confidentiality reasons, and (2) mining pools and regular nodes need incentives to correctly implement such blockchain redactions. Further, we observe that un-checked blockchain redactions are a threat to UWeb's censorship-resistance.

5.3 Longitudinal News Feed Writing

We obtained permission from the British Broadcasting Corporation (BBC) to reproduce their web articles (only text) into the Litecoin blockchain. We created a crawler that de-fanged articles (also removed the ads) from all public BBC's RSS feeds every day for 134 consecutive days. We daily bundled the text-only articles (average of 402 daily articles, at an average of 951 characters and 555 words per article) into categorized directories, compressed them into a gzipped file and wrote the archive using max-rate transactions (§4.2).

In total, we performed 134 writing operations (1 per day) to store 51,208,863 Bytes for 53,823 articles with an accumulated cost of 0.575132 LTC ($137.6865 at the

	Min	1st Qu.	Median	Mean	3rd Qu.	Max.
Daily articles	340.0	380.0	394.0	401.6	421.8	483.0
Size (Kbytes)	302.5	341.9	361.4	370.7	397.5	495.8
TX time (sec)	31.28	78.31	142.97	190.48	259.38	951.28
Conf. time (sec)	99.82	241.34	368.06	412.21	532.84	1128.39
Cost (mLTC)	3.478	3.931	4.155	4.263	4.573	5.700

Table 1. Statistics from UWeb writing of BBC articles.

time of writing). Thus, the average cost per article is 1100 Litoshi ($0.0026 at the time of writing), and the average daily cost is 0.0041 LTC ($0.9815). Assuming a daily average of 400 articles, at an average of 1000 characters per article, we expect the daily storage size to be around 400 KB. At this publication rate and current LTC price, we expect a monthly publication cost of $6 USD, which is below the average monthly news digital subscription price in the US of $9.24 [74].

Each daily storage operation required one funding transaction and four spending transactions. Each storing operation achieved an average transmission throughput of 3,469.20 Bytes/sec and an average confirmation throughput of 1,245.45 Bytes/sec. Figure 5 shows the daily time required for all transactions to obtain at least one confirmation on the blockchain. This time corresponds to the confirmation time row in Table 1. The color intensity of each dot represents the size in bytes of the bundle of daily articles, whose statistics are shown in the size row of Table 1. Further, the dot radius in Figure 5 represents the cost in mLTC, which corresponds to the cost row in Table 1. These results suggest that the transaction cost and size do not have a consistent impact over the expected confirmation time. On the contrary, confirmation time may be more sensitive to block utilization and network congestion due to large mempool sizes, as we shall see in the following.

Impact on Litecoin Network. Figure 6 shows the network mempool size evolution during a subset of 18 days of the BBC writing experiment, between March 20, 2019 and April 7, 2019. We observe that the BBC writing operations generate substantial, periodic spikes in terms of mempool size (in KB). However, these spikes quickly subside. We attribute this to the efficient blockchain utilization of max-rate transactions: all BBC writing transactions are likely embedded in the same block, which, when mined, frees mempool space. We note however that since each BBC writing operation fits in only 5 transactions, its impact on the number of unconfirmed transactions in the mempool is negligible. In addition, we observe that blue (transaction count) spikes are not correlated to the BBC writing red (total size) spikes. This suggests that max-rate transactions

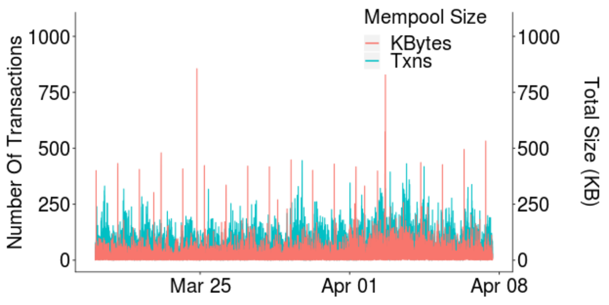

Fig. 6. Timeline of mempool size, with 1 min granularity, of (1) total size in KB, y axis on the right, and (2) number of unconfirmed transactions, y axis on the left. BBC article writing activity is responsible for the regular red spikes in total mempool size. Blue spikes are un-correlated to the red spikes, suggesting that temporary increases in activity do not pose a risk to the current network operation.

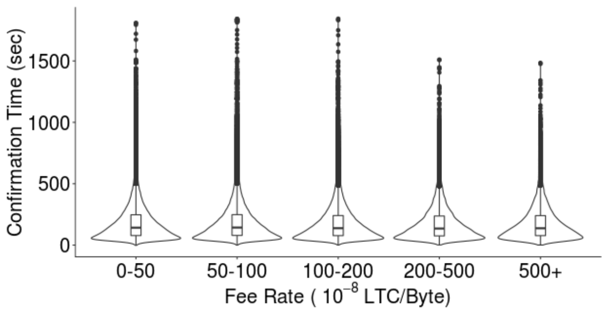

Fig. 7. Violin plots of new transactions confirmation time versus employed fee rate. Small bursts of arbitrary (BBC) writing do not have a measurable impact over transaction confirmation times.

achieve efficient block utilization, by occupying large portions of the block size while using a few transactions.

Further, we evaluated the impact of the BBC writing operations, on the time it takes Litecoin transactions of other users to be confirmed in the blockchain. Figure 7 shows the time required to achieve one confirmation for different fee rate levels during the March 20 - April 7 interval, over all the transactions posted on Litecoin during that interval. The time required to confirm new transactions is consistent with the expected 2.5 min epoch time. Moreover, for these levels of block utilization, the fee rate does not appear to have any significant impact over the network throughput.

5.4 Blockchain Stress-Test

To evaluate UWeb and max-rate transactions under large writing loads, we have collected a set of 41 censorship resistant (CR) tools from [75] that include Tor[76], Stegotorus[77], Uproxy-client[78], Bit-smuggler[79] and Shadowsocks. We excluded projects that did not publish an explicit re-distribution license. Table 3 in Appendix D provides the full list. The total compressed size of the 41 tools is 217 MB.

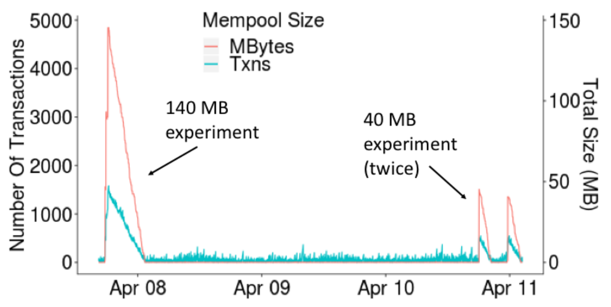

Fig. 8. Mempool size in total number of bytes and number of unconfirmed transactions, during our experiments.

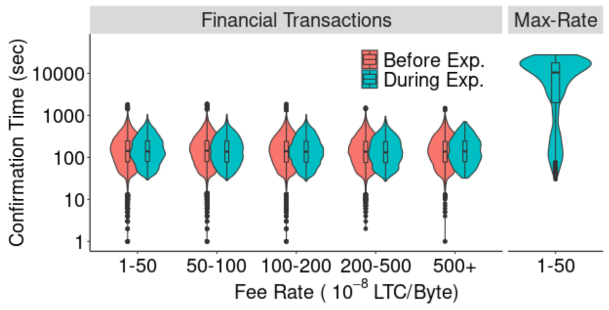

Fig. 9. Violin plots of confirmation times during and before our experiment. Financial transactions on the left, max-rate transactions on the right. Large-scale writing in the blockchain has no impact on financial transactions of any fee rate. Max-rate transactions however experience significant delays, up to 27,500s.

We designed our experiment in two phases. First, we almost-concurrently sent the first 3 largest size projects as individual archives, for a total of 140 MB. Second, we sent the rest of the tools compressed, in one split archive, for a total of 77 MB. This strategy allowed us to send files of size at most 45MB each, compatible with our virtual machine RAM memory resources. The total combined cost for this experiment was 2.51315448 LTC ($601.65 at the time of writing).

Mempool and Confirmation Time Impact. Figure 8 shows the timeline of the mempool size in terms of number of transactions (y axis left side) and total byte size (y axis right side), during the 3 writing operations. We observe that our writing experiments generate substantial spikes in terms of both number of transactions and their total size, which last for 7.65, 2.62 and 2.36 hours for the 140MB, 41.9MB and 38.5MB writing experiments, respectively. Further, the mempool is vacated linearly, which implies that higher fee transactions get prioritized correctly and our actions do not produce a degradation at current network usage levels.

We then measured the time each new unconfirmed transaction spent in the mempool before obtaining one confirmation on the Litecoin blockchain. Figure 9 compares the distribution of the confirmation time (y axis, log scale) over transactions employing different fee rates (x axis), during (1) our 140 MB writing experiment that lasted 7.6 hours from the first unconfirmed max-rate transaction observed in the mempool, until the last one received at least one confirmation (blue violins), and (2) before our experiment: since the start of the mempool instrumentation measurements on March 20, 2019, until the first unconfirmed transaction of the 140 MB experiment arrived to the mempool on April 7, 2019 (red violins). Interval (2) is the same 18 days when we monitored the mempool during the BBC experiment.

We observe that our significant writing experiment did not impact the confirmation times of financial transactions, including those employing the lowest fee rates (1- 50 lit/B). However, the maximum max-rate transaction confirmation time was 7.63 hours.

Financial transactions are unaffected because mining pools greedily select the next block to mine based on the fee rate of the transactions in the mempool [80]. Max-rate transactions have the lowest fee rate (1 litoshi per byte) thus have the lowest priority for selection. Only four financial transactions (of sizes up to 1,994B) had a fee rate of 1 litoshi/B. Their delays ranged from 69s to 527s (8.8 mins). This suggests that an overwhelming majority of financial transactions use fee rates that exceed the UWeb fee rate, thus are not impacted. While it is theoretically possible for minimum fee-rate transactions to be impacted by UWeb, this did not occur in our experiments.

Goodput and Throughput. We evaluate the goodput and throughput achieved during the first large-scale writing experiment (140+ MB data, over 7.6 hours) on the Litecoin blockchain. We split the aggregated data into 3 chunks of around 45 MBs each and sent them sequentially near in time. We took this precaution to have more control over deciding whether to call off the experiment in case the network was unable to handle the load. The average goodput measured over these 3 combined experiments was 90.8%, consistent with our theoretical analysis in § 4.4. The average throughput measured over these experiments, assuming 3 available funding inputs, was 183 KB/s. This exceeds the theoretical throughput of § 4.4, that considers a single funding input.

Block Utilization. Figure 11 compares the block size distribution over 200 blocks mined during the first large-scale writing experiment (blue bars) and over 1 million blocks mined before the experiment. We observe that most blocks mined during our experiment are close to the 1MB block-size limit, in contrast to the small size of blocks mined before.

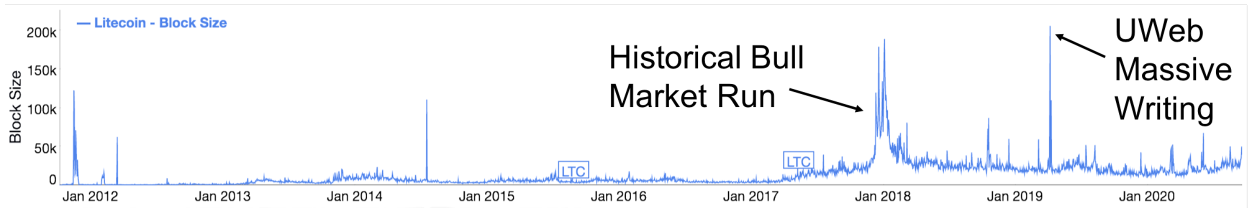

Fig. 10. Bulk writing experiment achieved the largest daily average block size (206.02KB) ever recorded on the history of Litecoin, exceeding the former record of the bull market run of 2017-2018 (189.1KB).

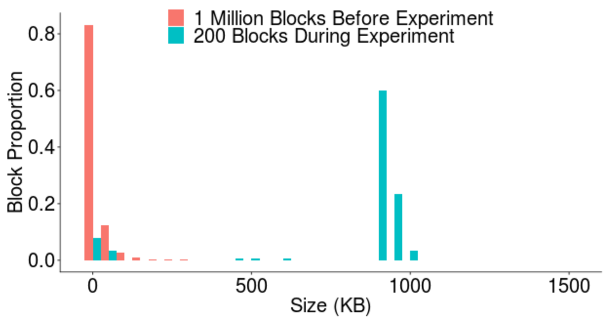

Fig. 11. Block utilization comparison before and during writing experiment. The distribution of block utilization is preserved even for very large periods of historical data.

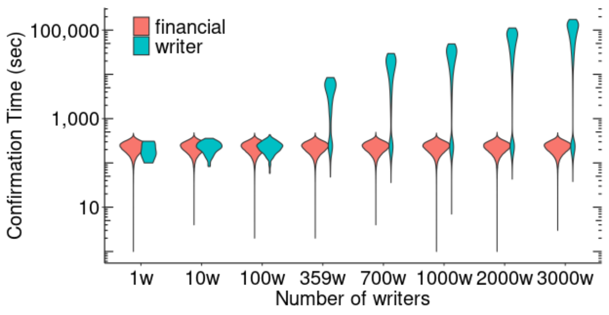

Fig. 12. Violin plots of confirmation times (y axis log scale) of financial and max-rate transactions when up to 3,000 BBC writers are active concurrently. Financial transactions are not affected by UWeb writing. This confirms our realnet experiments. Once the 1MB block size is reached, the confirmation time of max-rate transactions increases linearly with the number of writers.

Our large-scale writing experiments achieved a lifetime record high daily average block size as recorded by bitinfocharts [81]. Figure 10 shows the daily average block size of the Litecoin network for its entire lifetime. During the day of our experiments, the average daily block size reached 206KB, the largest ever recorded in the history of Litecoin.

In Appendix C we zoom-in into block utilization results when using UWeb to write the Tor source code [82] compressed archive (6.4 MB).

5.5 Max-Rate Transactions vs. The World

Table 2 compares max-rate transactions against state-of-the-art blockchain-writing solutions, on the metrics included in Definition 3.4. In our experiments, max-rate transactions achieve 2-3 orders of magnitude improvements in throughput and block utilizations and are the only solution that provides private access, censorship resistance, and resilience to integrity attacks. Catena [25] and Apertus [45] embed tens of bytes per input in a transaction, and have confirmation time requirements. This, coupled with the required transaction fees and the fact that often these transactions are non-redeemable, also leads to impractical costs for writers.

While Tithonus [47] has the same cost and slightly lower goodput to max-rate transactions, UWeb significantly improves on its throughput. Since UWeb does not attempt to publish content from within the censored region, it significantly improves on the throughput, goodput and cost efficiency of MoneyMorph [37].

Further, we notice that the goodput of max-rate transactions is consistent with the theoretical value derived in the proof of Claim 3. The throughput achieved by max-rate transactions is also consistent with the theoretical value derived in the proof of Claim 2. In contrast, the constructs used by Apertus [45], Catena [25] and Blockstack [23] have a constant relationship between payload size and throughput, thus cannot reduce the ratio between the maximum and their observed empirical throughput.

5.6 Simulation Stress-Test

We performed simulations to understand the impact on the Litecoin blockchain of multiple concurrent UWeb writers and higher levels of financial transactions. The simulator generates the next block to be mined by prioritizing the transactions from the mempool that have the highest fee rate [80], and breaks the tie based on arrival times.

Concurrent Writers. To evaluate UWeb and the Litecoin ecosystem under various loads of concurrent writers, we performed a simulation using the trace of finan-

Insertion Technique	Cost/Byte (Sat-Lit)	Max payload/ Construct	Construct Goodput (%)	Construct Throughput	Block Space Utilization (%)	Block Txn Utilization (%)	Safe to output mod. attack	Safe to input mod. attack	Private access	Censorship Resilience
Apertus	787.3	20 B	13.1	0.3 B/min	0.016	0.052	✓	✓	✗	✓
Catena	205.62	80 B	34.0	1.3 B/min	0.026	0.052	✓	✓	✗	✓
MoneyMorph(Bitcoin)	359.63	20 B	13.1	0.3 B/min	0.016	0.052	✓	✓	✓	✓
Tithonus	1.12	1635 B	88.8	545 B/s	0.20	0.052	✗	✗	✓	✓
Max Trans.	1.12	46.3 MB	90.8	115.1 KB/s	88	30	✓	✓	✓	✓

Table 2. Comparison of blockchain-writing techniques, considering the availability of a single spending address. In our experiments, the max-rate transaction approach (last row) achieves 2-3 orders of magnitude improvements in throughput and block utilization compared to state-of-the-art solutions, and a goodput improvement of 7 percentage points, while also providing private access, censorship resilience, and resilience to transaction integrity attacks.

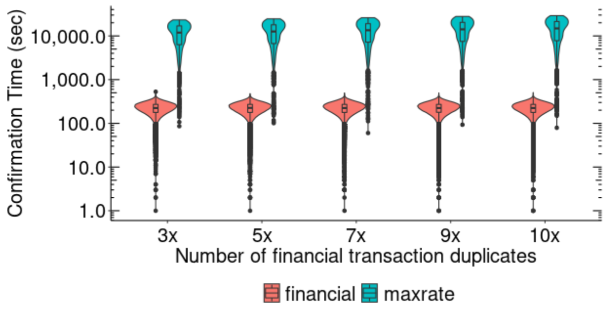

Fig. 13. Violin plots of confirmation times (y axis log scale) for the experiment depicted in Figure 9, when the number of financial transactions is up to 10 times the level in the experiment. Financial transactions are not impacted, while max-rate transactions experience a linear increase.

cial transactions recorded in the Litecoin blockchain between April 7 and April 10. This interval includes the interval of the realnet experiment shown in Figure 9.

We have simulated between 1 and 3,000 concurrent writers, each writing 400KB in the same 4 hour interval. Thus, each writer is equivalent to the daily writing load from the BBC experiment (§ 5.3). We have added max-rate transactions from each writer (four of size 99,931B one of 10,876B) at times randomly distributed in the same four-hour interval at the beginning the financial transaction trace. The case of 359 writers is roughly equivalent to the realnet experiment whose results are shown in Figure 9, where we wrote 140MB in the Litecoin blockchain. The 3,000 writers post 1.14GB of data-storing transactions.

Figure 12 shows the distributions of confirmation times of financial and max-rate transactions. We observe that financial transactions (red violins) are not impacted by the number of writers. This is because all financial transactions have a fee rate that exceeds UWeb's 1 litoshi/B rate. The maxrate transactions of UWeb writers have similar confirmation times to financial transactions, up to a number of writers that do not consume the available space in a block. After that point the confirmation time for maxrate transaction increases linearly with the number of writers.

Higher Levels of Financial Transactions. We have further simulated the impact of higher levels of financial transactions on the 140MB realnet-writing experiment depicted in Figure 9. For this, we have generated synthetic transaction traces where each financial transaction in a 36 hour interval following the start of the data writing experiment, contributes between 3 to 10 copies. Each copy of a financial transaction arrives at a random time during the 36h experiment. Figure 13 shows the distributions of the confirmation times. We observe that financial transactions are not impacted by this increased load, experiencing a mean of around 224s and maximum of 527s in all experiments. However, the confirmation times of maxrate transactions experience a linear increase with the density of financial transactions. For instance, at 3x financial transactions, the mean confirmation delay for max-rate transactions is 11,646s (SD = 6508, max = 23,253), while at 10x the mean delay is 14,546 (SD = 8,213, max = 29,101).

Blockchain Writing Throughput. The per-writer blockchain throughput is upper bounded by $A/150n$, where A is the available block space (1MB minus the financial transactions from the mempool), 150s is the average epoch length, and n is the number of writers. We note however that UWeb transactions will reach all the online UWeb readers in a few seconds over the Litecoin p2p gossip network [36]. Offline UWeb readers are not impacted by high confirmation delays: once a reader is back online they can either access newly written transactions in the mempool or in the blockchain: during the 3,000 concurrent writers experiment, the mempool had at its peak 11,379 max-rate transactions with a total of 1.05GB, and all these transactions received at least one confirmation within 52.27 hours.

6 Discussion

What If China Censors Blockchains? China has not blocked cryptocurrency-carrying traffic for almost

13 years despite earlier reports of their use to evade its firewall [34, 35, 83]. Further, UWeb is compatible and can be ported to all Satoshi blockchains. Thus, a censor that would block only a subset of Satoshi blockchains would still be unable to prevent UWeb use. A censor that blocks all Satoshi cryptocurrencies will deprive the economy of the censored region of access to a financial instrument with a cap that exceeds $1 trillion at the time of writing. We also note that UWeb provides value for all countries with restricted freedom of speech [84] that do not block Satoshi cryptocurrencies.

Why Not Other Cryptocurrencies? Other cryptocurrencies can also provide the underlying censorship-resistant storage medium for UWeb, particularly Monero and Zcash that offer additional privacy guarantees. MoneyMorph [37] compared Monero, Zcash, Bitcoin, and Ethereum, and revealed that Zcash shielded addresses offer the highest censorship-resistant bandwidth by allowing the embedding of up to 1,148B. Other efforts have also used the Ethereum blockchain to avoid censorship resistance [34, 35].

Our choice of Satoshi cryptocurrencies is based on their huge share of the cryptocurrency market cap (62%) [38], a network of 10,000 P2P nodes [85] and a large and growing hashrate (150m Th/s) [86].

In this paper we also developed solutions that embed up to 1,650B in a Satoshi address and up to 46.3MB of data in a staging and spending transaction construct. Our solutions achieve a writing throughput that exceeds those of state-of-the-art solutions by 2-3 orders of magnitude. We hope that our work will encourage further research in the use of Monero, Zcash and Ethereum as a medium for censorship resistance.

We also note that our main reason for conducting realnet experiments on Litecoin instead of Bitcoin is that Litecoin allowed us to perform more experiments for the same money. While Litecoin is identical to Bitcoin in terms of most technical aspects, including priority policy to choose transactions for mining, we acknowledge that Bitcoin's longer confirmation interval would impose longer confirmation times on all transactions, including max-rate transactions.

Blocking Max-Rate Transactions. Cryptocurrency ecosystems could modify their mining software to block max-rate transactions. We note that funding transactions are indistinguishable from any other P2SH transaction and thus can not be profiled or blocked by non-conforming nodes. We also note that modifications to mining software that attempt to block the corresponding spending transactions would make the blockchain open to cluttering from unspent transactions, that would excessively pollute the UTXO. Further, the financial motivations to obtain the associated mining fees are also incompatible with such a modification, which has not been observed so far in the wild.

Participation Incentives. None of the Satoshi blockchains provide incentives for full node participation. However, UWeb actually provides an incentive for its users to host full nodes, in order to protect their access privacy. Thus, by providing incentives for hosting full nodes, UWeb has the potential to strengthen Satoshi ecosystems.

Can PIR Reduce Bandwidth Use? While private information retrieval (PIR) solutions could reduce the communication costs imposed on UWeb clients, they would also cue the censor that the nodes running them use UWeb. This is because the reduced communications of PIR solutions can be profiled by the censor, and will look quite distinguishable from the communications of a standard Satoshi node. Second, the censor will see that the blocks retrieved by a node contain UWeb content, whereas those not retrieved do not contain such content. For instance, the use of Bloom filters specific for UWeb transactions will be distinguishable in terms of their UWeb content from the Bloom filters used by regular financial clients.

7 Conclusions

We have shown that arbitrary data insertion in commercial Satoshi blockchains can be practical when leveraging max-rate transactions, the customized smart contract constructs that we introduced. We have designed UWeb, a persistent data storage solution that builds on max-rate transactions to provide private and anonymous access to content and censorship resistance. We have implemented and evaluated UWeb on experiments with writing 268.21 MB of data in the Litecoin mainnet, and have shown that it achieves a 2-3 order of magnitude improvement on the storage throughput and blockchain utilization of state-of-the-art solutions. While our experiments broke Litecoin's daily block size lifetime record, they had only short-term effects on its ecosystem.

8 Acknowledgments

This research was supported by NSF grants CNS-2013671 and CNS-2114911, and CRDF grant G-202105-67826. This publication is based on work supported by a grant from the U.S. Civilian Research & Development Foundation (CRDF Global). Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of CRDF Global.

References

[1] Margot Williams, Henrik Moltke, Micah Lee, and Ryan Gallagher. Meltdown Showed Extent of NSA Surveillance. The Intercept, https://theintercept.com/2019/05/29/nsa-sidtoday-surveillance-intelligence/, 2019.

[2] Government Monitoring of Social Media: Legal and Policy Challenges. https://www.brennancenter.org/our-work/research-reports/government-monitoring-social-media-legal-and-policy-challenges.

[3] The List of Blocked Websites in China. https://www.saporedicina.com/english/list-of-blocked-websites-in-china/.

[4] List of Websites Blocked in Russia. https://en.wikipedia.org/wiki/List_of_websites_blocked_in_Russia.

[5] Devin Coldewey. China moves to ban foreign software and hardware from state offices. TechCrunch, https://techcrunch.com/2019/12/09/china-moves-to-ban-foreign-software-and-hardware-from-state-offices/, December 2019.

[6] U.S. bans WeChat, TikTok as China becomes major focus of election. The Washington Post, https://www.washingtonpost.com/technology/2020/09/18/tiktok-wechat-ban-trump/, September 2020.

[7] Stephanie Kirchgaessner, Emma Graham-Harrison, and Lily Kuo. China clamping down on coronavirus research, deleted pages suggest. The Guardian, https://www.theguardian.com/world/2020/apr/11/china-clamping-down-on-coronavirus-research-deleted-pages-suggest, 2020.

[8] George Cooper. Chinese state censorship of COVID-19 research represents a looming crisis for academic publishers. https://blogs.lse.ac.uk/impactofsocialsciences/2020/04/24/chinese-state-censorship-of-covid-19-research-represents-a-looming-crisis-for-academic-publishers/, 2020.

[9] Does your VPN Keep Logs? 120 VPN Logging Policies Revealed. https://www.comparitech.com/vpn/vpn-logging-policies/.

[10] Hundreds of Millions Have Downloaded Suspicious VPN Apps With Serious Privacy Flaws. Apple and Google Haven't Taken Action. Entrepreneur, https://www.entrepreneur.com/article/337885.

[11] Study finds half of most popular VPN apps linked to China. FinancialTimes,https://www.ft.com/content/e5567d8a-ee65-11e8-89c8-d36339d835c0.

[12] Jiang Peng seized and warned for watching YouToube videos and shopping on Amazon. Turbo VPN records used as evidence. https://twitter.com/SpeechFreedomCN/status/1211095986908516352.

[13] Tor 0day: Finding IP Addresses. https://www.hackerfactor.com/blog/index.php?/archives/896-Tor-0day-Finding-IP-Addresses.html.

[14] IBM Blockchain. Now delivering value around the world. https://www.ibm.com/blockchain.

[15] AWS Blockchain Partners: Accelerating your distributed ledger journey. Amazon, https://aws.amazon.com/partners/blockchain/.

[16] Microsoft Azure Blockchain. Develop, test, and deploy secure blockchain apps. https://azure.microsoft.com/en-us/solutions/blockchain/.

[17] MaidSafe. https://maidsafe.net/.

[18] FileCoin. https://filecoin.io/.

[19] Sia: Fully Decentralized Cloud. https://sia.tech/.

[20] LTI: Let's Solve. https://www.lntinfotech.com/services/consulting/blockchain/.

[21] Eleftherios Kokoris-Kogias, Enis Ceyhun Alp, Sandra Deepthy Siby, Nicolas Gailly, Philipp Jovanovic, Linus Gasser, and Bryan Ford. Hidden in plain sight: Storing and managing secrets on a public ledger. *IACR Cryptology ePrint Archive*, 2018.

[22] Alevtina Dubovitskaya, Zhigang Xu, Samuel Ryu, Michael Schumacher, and Fusheng Wang. Secure and trustable electronic medical records sharing using blockchain. *CoRR*, abs/1709.06528, 2017.

[23] Muneeb Ali, Jude Nelson, Ryan Shea, and Michael J. Freedman. Blockstack: A Global Naming and Storage System Secured by Blockchains. In *Proceedings of the Usenix Annual Technical Conference*, pages 181–194, 2016.

[24] Muneeb Ali, Ryan Shea, Jude Nelson, and Michael J Freedman. Blockstack: A new decentralized internet. *Whitepaper*, 2017.

[25] Alin Tomescu and Srinivas Devadas. Catena: Efficient Non-equivocation via Bitcoin. In *Proceedings of IEEE Symposium on Security and Privacy*, pages 393–409, 2017.

[26] Xueping Liang, Sachin Shetty, Deepak Tosh, Charles Kamhoua, Kevin Kwiat, and Laurent Njilla. Provchain: A blockchain-based data provenance architecture in cloud environment with enhanced privacy and availability. In *Proceedings of the 17th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing*, pages 468–477, 2017.

[27] Bin Liu, Xiao Liang Yu, Shiping Chen, Xiwei Xu, and Liming Zhu. Blockchain based data integrity service framework for iot data. In *Web Services (ICWS), 2017 IEEE International Conference on*, pages 468–475. IEEE, 2017.

[28] Hossein Shafagh, Lukas Burkhalter, Anwar Hithnawi, and Simon Duquennoy. Towards blockchain-based auditable storage and sharing of iot data. In *Proceedings of the 2017 on Cloud Computing Security Workshop*, pages 45–50, 2017.

[29] Antorweep Chakravorty and Chunming Rong. Ushare: User controlled social media based on blockchain. In *Proceedings of the 11th International Conference on Ubiquitous Information Management and Communication*, pages 99:1–99:6, 2017.

[30] Adiseshu Hari and T. V. Lakshman. The Internet Blockchain: A Distributed, Tamper-Resistant Transaction Framework for the Internet. In *Proceedings of the 15th ACM Workshop on Hot Topics in Networks*, pages 204–210, 2016.

[31] Binanda Sengupta, Samiran Bag, Sushmita Ruj, and Kouichi Sakurai. Retricoin: Bitcoin based on compact proofs of retrievability. In *Proceedings of the 17th International Conference on Distributed Computing and Networking*, 2016.

[32] Matthias Wachs, Martin Schanzenbach, and Christian Grothoff. A censorship-resistant, privacy-enhancing and

fully decentralized name system. In *International Conference on Cryptology and Network Security*, pages 127–142. Springer, 2014.

[33] Andrew Miller, Ari Juels, Elaine Shi, Bryan Parno, and Jonathan Katz. Permacoin: Repurposing bitcoin work for data preservation. In *IEEE Symposium on Security and Privacy (SP)*, pages 475–490, 2014.

[34] Nir Kshetri. Chinese internet users turn to the blockchain to fight against government censorship. The Conversation, https://theconversation.com/chinese-internet-users-turn-to-the-blockchain-to-fight-against-government-censorship-111795, 2019.

[35] Wolfie Zhao. Pharma Scandal Prompts Calls to Put Vaccine Data on a Blockchain. CoinDesk, https://www.coindesk.com/pharma-scandal-prompts-calls-to-put-vaccine-data-on-a-blockchain, 2018.

[36] Ruben Recabarren and Bogdan Carbunar. Tithonus: A Bitcoin Based Censorship Resilient System. *PoPETS*, 2019(1), 2019.

[37] Mohsen Minaei, Pedro Moreno-Sanchez, and Aniket Kate. Moneymorph: Censorship resistant rendezvous using permissionless cryptocurrencies. *Proceedings on Privacy Enhancing Technologies*, 2020(3):404–424, 2020.

[38] CoinMarketCap. https://coinmarketcap.com/.

[39] Total Number of (Bitcoin) Transactions. https://www.blockchain.com/charts/n-transactions-total.

[40] Total Number of (Litecoin) Transactions. https://blockchair.com/litecoin.

[41] Transaction malleability. https://en.bitcoin.it/wiki/Transaction_malleability.

[42] Miner Fees - Historic rules for fee transactions. https://en.bitcoin.it/wiki/Miner_fees#Historic_rules_for_free_transactions.

[43] Miner Fees - Relaying. https://en.bitcoin.it/wiki/Miner_fees#Relaying.

[44] Len Sassman tribute by Dan Kaminsky. http://www.slideshare.net/dakami/black-ops-of-tcpip-2011-black-hat-usa-2011.

[45] Apertus 0.3.17-beta. Archive data on your favorite blockchains. http://apertus.io/.

[46] Number of OP_RET operations check. https://github.com/bitcoin/bitcoin/blob/0.18/src/policy/policy.cpp#L135.

[47] Ruben Recabarren and Bogdan Carbunar. Tithonus: A bitcoin based censorship resilient system. *PoPETs*, 2019(1):68–86, 2019.

[48] Peter Todd. Input Writing Example. https://github.com/petertodd/python-bitcoinlib/blob/master/examples/publish-text.py.

[49] Majority Attack. https://en.bitcoin.it/wiki/Majority_attack.

[50] Largest Crypto-Mining Marketplace. https://www.nicehash.com/.

[51] ISP Tracking – 4 Ways to Stop it and Protect Your Privacy. https://digital.com/blog/isp-tracking/.

[52] Is My Internet Service Provider Spying on Me? https://securethoughts.com/internet-service-provider-spying/.

[53] IsStandard() Function. https://github.com/bitcoin/bitcoin/blob/0.18/src/policy/policy.cpp#L57.

[54] Maximum Transaction Size. https://github.com/bitcoin/bitcoin/blob/0.16/src/policy/policy.h#L24.

[55] Effective ScriptSig Size. https://github.com/bitcoin/bitcoin/blob/0.16/src/policy/policy.cpp#L108.

[56] Standard Transaction. https://bitcoin.org/en/glossary/standard-transaction.

[57] Unconfirmed chains. https://github.com/bitcoin/bitcoin/blob/57b34599b2deb179ff1bd97ffeab91ec9f904d85/doc/release-notes/release-notes-0.12.0.md.

[58] John Holowczak and Amir Houmansadr. Cachebrowser: Bypassing chinese censorship without proxies using cached content. In *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*, pages 70–83. ACM, 2015.

[59] David Fifield, Chang Lan, Rod Hynes, Percy Wegmann, and Vern Paxson. Blocking-resistant communication through domain fronting. *Proceedings on Privacy Enhancing Technologies*, 2015(2):46–64, 2015.

[60] Bitcoin Adoption in the world. https://blog.chainalysis.com/reports/2020-global-cryptocurrency-adoption-index-2020.

[61] Bitcoin trading volume. https://www.statista.com/statistics/1195753/bitcoin-trading-selected-countries/.

[62] Andrew Sward, Vecna OP, and Forrest Stonedahl. Data Insertion in Bitcoin's Blockchain. Augustana Digital Commons, 2017.

[63] Natkamon Tovanich, Nicolas Soulié, and Petra Isenberg. Visual analytics of bitcoin mining pool evolution: on the road toward stability? In *3rd International Workshop on Blockchains and Smart Contracts (BSC 2020-2021), held in conjunction with the 11th IFIP International Conference on New Technologies, Mobility and Security (IFIP NTMS 2021)*, 2021.

[64] Bitcoin Wiki. Variable length integer format. https://en.bitcoin.it/wiki/Protocol_documentation#Variable_length_integer.

[65] Mendel Rosenblum and John K. Ousterhout. The design and implementation of a log-structured file system. *ACM Trans. Comput. Syst.*, 10(1):26–52, February 1992.

[66] John H. Hartman and John K. Ousterhout. The zebra striped network file system. *ACM Trans. Comput. Syst.*, 13(3):274–310, August 1995.

[67] Athicha Muthitacharoen, Robert Morris, Thomer M. Gil, and Benjie Chen. Ivy: A read/write peer-to-peer file system. *SIGOPS Oper. Syst. Rev.*, 36(SI):31–44, December 2002.

[68] Litecoin - Download. https://litecoin.org/.

[69] Saias" "Evyatar. BITCOINS, BLOCKCHAINS, AND BOTNETS. https://blogs.akamai.com/sitr/2021/02/bitcoins-blockchains-and-botnets.html.

[70] Dominic Deuber, Bernardo Magri, and Sri Aravinda Krishnan Thyagarajan. Redactable blockchain in the permissionless setting. *CoRR*, abs/1901.03206, 2019.

[71] Eugenia Politou, Fran Casino, Efthimios Alepis, and Constantinos Patsakis. Blockchain mutability: Challenges and proposed solutions. *arXiv preprint arXiv:1907.07099*, 2019.

[72] Martin Florian, Sebastian Henningsen, Sophie Beaucamp, and Björn Scheuermann. Erasing data from blockchain nodes. In *2019 IEEE European Symposium on Security and Privacy Workshops (EuroS&PW)*, pages 367–376. IEEE, 2019.

[73] David Derler, Kai Samelin, Daniel Slamanig, and Christoph Striecks. Fine-grained and controlled rewriting in blockchains: Chameleon-hashing gone attribute-based. *IACR*

Cryptology ePrint Archive, 2019:406, 2019.
[74] Tracy M. Cook. How much U.S. newspapers charge for digital subscriptions. https://www.americanpressinstitute.org/publications/reports/digital-subscription-pricing/.
[75] Curated list of open-source anti-censorship tools. https://github.com/danoctavian/awesome-anti-censorship.
[76] Alternative methods to download Tor Browser. https://www.torproject.org/projects/gettor.
[77] Stegotorus. https://sri-csl.github.io/stegotorus/.
[78] UProxy-client. https://sourceforge.net/projects/unr-proxy/files/UProxy-Client/.
[79] BitSmuggler. https://github.com/danoctavian/bit-smuggler.
[80] Miner Fees: Including in Blocks. https://en.bitcoin.it/wiki/Miner_fees#Including_in_Blocks.
[81] Litecoin Block Size Historical Chart. https://bitinfocharts.com/comparison/size-ltc.html.
[82] Download Tor. https://www.torproject.org/download/download.
[83] Gregory Rocco. Public Blockchains as a Means to Resist Information Censorship. CUNY Academic Works, https://academicworks.cuny.edu/gc_etds/2995, 2019.
[84] 2020 World Press Freedom Index. https://rsf.org/en/ranking.
[85] Global Bitcoin Nodes Distribution. https://bitnodes.io/.
[86] Total Bitcoin Hashrate. https://www.blockchain.com/charts/hash-rate.

A UWeb Analysis and Proofs

We prove Claim 4, i.e., that max-rate transactions are standard.

Proof. Our max-rate transaction $\tau{=}(v, f, I_T, O_T, w, l_o)$ (§ 4.2) is standard since it satisfies the requirements of the reference client code [53]. Specifically, first *transaction weight check*: we choose $c = 59$, so that the total size of τ is less than 100KB, see § 4.2. Second, *transaction inputs check*: the maximum size of our data storing script (see § 4.1) is 1,568 bytes, thus smaller than 1,650B. Third, *number of OP_RET operations check*: satisfied since our output list consists of only one OP_RET output, see § 4.1. Fourth, *transaction outputs check*: we do not use bare multi-sig outputs and since we use one OP_RET output the dust check is ignored. Fifth, we satisfy the following conditions for the `isStandard()` function [53]: (1) *Solver function*: Our construct is of type `TX_NULL_DATA` as it uses one `OP_RET` output (§ 4.1), (2) *Multi-Sig txn check*: Trivially satisfied, since we do not use multi-sig outputs, (3) *Null-transaction check*: We use a 0 length `OP_RET` output, which is below 83B. We will include the full proof in the technical report. □

Max-Rate Transactions Prevent Input Modification Attacks. Transaction validation for spending outputs guarantees the *integrity* of the data contained in our input script. The hash lock constructs (lines 7-9 of Algorithm 1) protect the data pushed (lines 2-4). Only the hash of the redeeming script can unlock the output while only the data pushed in the stack allows for an error-less script execution. This prevents the input modification attack of our spending transactions as only the intended data pushes will satisfy the redeeming script that spends our funding transaction output. Since the rules for standard transactions require that the stack is left with only one TRUE value, spurious data pushes preceding our data are not a concern.
Max-Rate Transactions Prevent Output Modification (Rebind) Attacks. Max-rate transactions make the output modification attacks impossible, by using an OP_RET output and a minimum fee rate in order to effectively prevent the adversary from *stealing* dust value or decreasing the effective transaction fee rate. The attacker can not replace our OP_RET output with even a dust value output as this would increase the size of the transaction (thus violating the transaction weight check [53]) and any output value larger than zero would further reduce the fee rate to smaller than the minimum accepted for relaying (thus violating the transaction output check [53] and restriction 3.5, see § 3.1).

B Integrity Attacks

Output Script Modification Attack. When receiving (over the gossip network) a transaction that does not include the operations OP_CHECKSIG or OP_CHECKSIGVERIFY in its input scripts, the adversary replaces the transaction's output addresses with his own. The lack of a signature verification involving the output scripts, implies that the result is a valid transaction. The adversary rushes to broadcast the modified transaction, in an attempt to get it mined ahead of the original transaction. If the modified transaction is mined before the original, the attacker succeeds and effectively steals the corresponding output values. Further, if data is stored in transaction outputs, the adversary can also modify this data, thus also achieves a data integrity attack.
Input Script Modification Attack. An attacker that receives a transaction with redeeming scripts that do not perform stack validation operations, e.g. via the

sequences: OP_HASH160 <hash160> OP_EQUAL or OP_HASH160 <hash160> OP_EQUALVERIFY, can replace any data preceding the redeeming script and still obtain a valid input script, effectively achieving a data corruption attack. The attacker rushes to broadcast the modified transaction over the gossip network, in an effort to have it mined before the original.

C Block Utilization in the Tor-Writing Experiment

We zoom-in into block utilization results when using UWeb to write the Tor source code [82] compressed archive (6.4 MB). Making Tor available to the world in the Litecoin blockchain required 17 funding and 73 spending transactions with a total cost of 0.0735 LTC ($17.5959 at the time of writing). Figure 14 shows the ability of UWeb and max-rate transactions to harness the blockchain capacity to the limit. The height of the bars represent the total transaction size per block. Our writing starts on block 1525546 and proceeds to block 1525556. We observe that blocks that do not contain our transactions (blocks smaller than 1525546 and bigger than 1525556) are mostly empty.

We note that block 1525550 consists of 10 max-rate transactions, each of 99,931B. Thus, our data is responsible for 90% of the transactions in this block and takes up 99.96% of the total block space. However, this usage of the Litecoin block did not result in a backup of financial transactions: for instance, block 1525553 that was mined in the middle of the experiment (Figure 14), only contains financial transactions, and their total size is smaller than in blocks before our experiment, e.g., blocks 1525544 and 1525546. Further, our above experiments in the realnet (e.g., 140MB) and simulation (up to 1.14GB in Section 5.6) show that writing at a much larger scale than the Tor writing experiment imposed no delays on financial transactions.

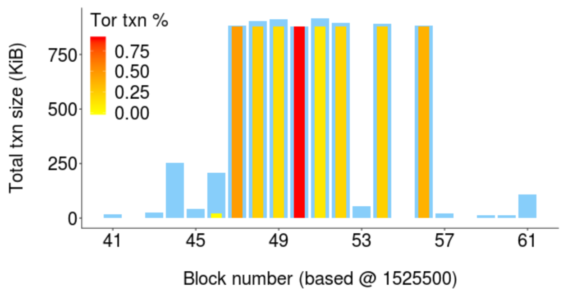

Fig. 14. Impact on Litecoin blockchain of UWeb, Tor source code writing. Writing starts on block 46 and finishes on block 56. The block size increase is due to max-rate transactions, however, except for blocks 47 and 50, most Tor containing blocks contain many other small-size transactions.

D UWeb-Written Censorship Resistant Systems & Licenses

Table 3. CRS source code permanently stored in the Litecoin Blockchain and their licenses that allow for free redistribution (Ordered by size).

Project Name	Redistribution License	Zip Size (KB)
Stegotorus	3-clause BSD	52000
Uproxy-client	Apache v2	45000
Bit-smuggler	GPLv2	42000
shadowsocks	3-clause BSD	11000
lantern	Apache v2	10000
tribler	GPLv3	9700
Tor	Custom: Ok to redistribute	6400
telex	Custom: Ok to redistribute and Apache v2	6000
twister-core	Custom: Ok to redistribute	4500
infranet	BSD	2600
i2p.i2p	Public Domain	2200
obfuscated-openssh	BSD or more free than BSD	1900
ZeroNet	GPL v2	1900
Dust	Custom: Ok to redistribute	1600
streisand	GPLv3	1400
ipfs	MIT	978
firefly-proxy	Custom: 2-Clause BSD-like	930
Vuvuzela	GPL v3	490
obfsproxy	Custom: Ok to redistribute	465
rubberhose	Custom: Ok to redistribute	392
WireGuard	GPL v2	314
Whonix	GPLv3	285
facebook-tunnel	Apache v2	264
cachebrowser	MIT	230
go-packetflagon	Custom: 2-clause BSD-like	179
codetalkertunnel	GPLv3	169
gfw_whitelist	MIT	108
iodine	Custom: all permissions granted	78
gfwlist	GPL v2.1	73
govpn	GPLv3	67
obfs4	Custom: Ok to redistribute	67
GoodbyeDPI	Apache v2	42
marionette	Apache v2	38
ChinaDNS	GPL v3	37
Dat	Custom: 3-Clause BSD-like	37
gohop	GPLv3	37
fteproxy	Apache v2	28
antizapret	Creative Commons Legal Code	21
blockcheck	MIT	17
uproxy-server	Apache v2	6.9
fwlite	GPL v3	1.3