Blockstack: A Global Naming and Storage System Secured by Blockchains

Muneeb Ali and Jude Nelson, *Princeton University and Blockstack Labs;*
Ryan Shea, *Blockstack Labs;* Michael J. Freedman, *Princeton University*

https://www.usenix.org/conference/atc16/technical-sessions/presentation/ali

This paper is included in the Proceedings of the
2016 USENIX Annual Technical Conference (USENIX ATC '16).

June 22–24, 2016 • Denver, CO, USA

978-1-931971-30-0

Open access to the Proceedings of the
2016 USENIX Annual Technical Conference
(USENIX ATC '16) is sponsored by USENIX.

Blockstack: A Global Naming and Storage System Secured by Blockchains

Muneeb Ali[⋆†], Jude Nelson[⋆†], Ryan Shea[†], Michael J. Freedman[⋆]
[⋆]*Princeton University,* [†]*Blockstack Labs*

Abstract

Blockchains like Bitcoin and Namecoin and their respective P2P networks have seen significant adoption in the past few years and show promise as naming systems with no trusted parties. Users can register human meaningful names and securely associate data with them, and only the owner of the particular private keys that registered them can write or update the name-value pair. In theory, many decentralized systems can be built using these blockchain networks, such as new, decentralized versions of DNS and PKI. As the technology is relatively new and evolving rapidly, however, little production data or experience is available to guide design tradeoffs.

In this paper, we describe our experiences operating a large deployment of a decentralized PKI service built on top of the Namecoin blockchain. We present various challenges pertaining to network reliability, throughput, and security that we needed to overcome while registering and updating over 33,000 entries and 200,000 transactions on the Namecoin blockchain. Further, we discuss how our experience informed the design of a new blockchain-based naming and storage system called Blockstack. We detail why we switched from the Namecoin network to the Bitcoin network for the new system, and present operational lessons from this migration. Blockstack is released as open source software and currently powers a production PKI system for 55,000 users.

1 Introduction

Cryptocurrency blockchains and their respective P2P networks are useful beyond exchanging money. They provide cryptographically auditable, append-only ledgers that are already being used to build new, *decentralized* versions of DNS [41] and public-key infrastructure (PKI) [43], along with other applications like file storage [23] and document timestamping [15]. Because blockchains have no central points of trust or failure, they enable a new class of decentralized applications and services that minimize the degree to which users need to put trust in a single party, like a DNS root server or a root certificate authority.

Blockchain networks have attracted a lot of interest from enthusiasts, engineers, and investors. In fact, 1.1 billion USD has been invested in blockchain startups over the last several years [19]. With the rapid capital infusion, infrastructure for blockchains is getting quickly deployed [18] and blockchains are emerging as publicly available common infrastructure for building decentralized systems and applications. However, blockchain networks are at a very early stage and there is very little production data available to guide design trade-offs.

Many non-financial applications of blockchains imply the need for a naming system that securely binds *names*, which can be human-readable, to arbitrary *values*. The blockchain gives consensus on the global state of the naming system and provides an append-only global log for state changes. Writes to name-value pairs can only be announced in new blocks, as appends to the global log. The global log is logically centralized (all nodes on the network see the same state), but organizationally decentralized (no central party controls the log).

The decentralized nature of blockchain-based naming introduces meaningful security benefits, but certain aspects of contemporary blockchains present technical limitations. Individual blockchain records are typically on the order of kilobytes [49] and cannot hold much data. Latency of creating and updating records is capped by the blockchain's write propagation and leader election protocol, and it is typically on the order of 10-40 minutes [14]. The total new operations in each round are limited by average bandwidth of nodes participating in the network (for Bitcoin the current average is $\sim$1500 new operations per new round [2]). Further, new nodes need to independently audit the global log from the beginning: as the system makes forward progress, the time to bootstrap new nodes increases linearly.

We believe that in spite of these scalability and performance challenges, blockchains provide important infrastructure for building secure, decentralized services. The cost of tampering with blockchains grows with their adoption: today, it would require hundreds of millions of dollars to attack a large blockchain like Bitcoin [1].

These benefits motivated us to use blockchains to build a new decentralized PKI system. Our system enables users to register unique, human-readable usernames and associate public-keys, like PGP [53], along with additional data to these usernames. There is no need for any central or trusted party in our PKI system. This paper presents our experiences from operating this PKI system on the Namecoin network, which is one of the largest services built on top of a blockchain to date. We outline the challenges that we had to overcome for registering and updating over 33,000 user entries and for sending over 200,000 transactions on the Namecoin network.

Our production deployment led to many interesting experiences where we observed and analyzed network anomalies and security problems that were not discovered or documented before. **We discovered a critical security problem where a single miner consistently had more than 51% of the total compute power on the Namecoin network** (see [35] for details on the 51% attack and compute power of miners). A 51% attack is one of the most serious attacks on a blockchain and impacts its security and decentralization properties.

Moreover, we also encountered chronic networking issues with broadcasting transactions on the Namecoin network. Reliability of the network generally depends on how actively a blockchain network is monitored and maintained, as well as the financial incentives for operating the network. Therefore, for both security and reliability reasons, blockchain-based services should use the largest and most secure blockchain, which at the time of writing is the Bitcoin blockchain.

Our experience with Namecoin informed the design and implementation of a new blockchain-based naming and storage system, called Blockstack, that uses the Bitcoin blockchain. Unlike previous blockchain-based systems, Blockstack *separates its control and data plane considerations*: it keeps only minimal metadata (namely, data hashes and state transitions) in the blockchain and uses external datastores for actual bulk storage. Blockstack enables fast bootstrapping of new nodes by using checkpointing and *skip lists* to limit the set of blocks that a new node must audit to get started. We have released Blockstack as open source [13].

Modifying production decentralized systems like Bitcoin (and introducing new functionality for which it was not designed) is quite difficult, particularly that the system still needs to reach "consensus." With Blockstack, we extend the single state machine model of blockchains to allow for arbitrary state machines without requiring consensus breaking changes in the underlying blockchain. This design was non-intuitive before our work; indeed, the standard approach for the past three years was to fork the main Bitcoin blockchain to add new and different functionality. Our experience with the Namecoin blockchain shows that starting new, smaller blockchains leads to security problems (like reduced computational power needed to attack the network) and should be avoided when possible.

This paper makes the following contributions:

- We present the first analysis of security and network reliability of a blockchain other than Bitcoin and report a critical security problem where a major alternate blockchain, Namecoin, had a single miner with well over 51% of the compute power for months.
- We report that merged mining, a popular method to secure smaller blockchains, is currently failing in practice. The total compute power dedicated to blockchains is currently insufficient to support multiple secure blockchains.
- We present the design of Blockstack's logically separate layer, *virtualchain*, which introduces novel new functionality to production blockchains without requiring any consensus-breaking changes from the underlying blockchain.
- We present a migration framework for migrating from one blockchain to another under a failure of the underlying blockchain, and present lessons from a successful migration of our production system from Namecoin to Bitcoin. This was the first cross-chain migration of a production system running on blockchains.

2 Motivation and Background

In this section, we describe the motivation for building naming systems that have no central point of trust and provide the relevant background on blockchains. In this paper, we use the term *naming system* to mean (a) names are **human-readable** and can be picked by humans, (b) name-value pairs have **a strong sense of ownership**—that is, they can be owned by cryptographic keypairs, and c) there is **no central trusted party** or point of failure. Building a naming system with these three properties was considered impossible according to Zooko's Triangle [32] and most traditional naming systems provide two out of these three properties [31]. Namecoin [41] used a blockchain-based approach to provide the first naming system that offered all three properties: human-readability, strong ownership, and decentralization.

2.1 Background on Blockchains

Blockchains provide a global append-only log that is publicly writeable. Writes to the global log, called *transactions*, are organized as *blocks* and each block packages multiple transactions into a single atomic write. Writing to the global log requires a payment in the form of a *transaction fee*. Nodes participating in a blockchain network follow a leader election protocol for deciding which node gets to write the next block and collect the respective transaction fees. Not all nodes in the network participate in leader election. Nodes actively competing to become the leader of the next round are called *miners*. At the start of each round, all miners start working on a new computation problem, derived from the last block, and the miner that is the first to solve the problem gets to write the next block. In Bitcoin, the difficulty of these computation problems is automatically adjusted by the protocol so that 1 new block is produced roughly every 10 minutes. See [14] for further details on how blockchains work and how they reach consensus.

2.2 Namecoin's Naming System

Namecoin is one of the first forks of Bitcoin and is the oldest blockchain other than Bitcoin that is still operational, with a cryptocurrency market capitalization of 5 million USD as of May 2016 [6] (the market capitalization of a cryptocurrency is the exchange-traded value of its coins multiplied by its number of coins in existence). The main motivation for starting Namecoin was to create an alternate DNS-like system that replaces DNS root servers with a blockchain for mapping domain names to DNS records [41]. Given that blockchains don't have central points of trust, a blockchain-based DNS is much harder to censor and registered names cannot be seized from owners without getting access to their respective private keys [31]. Altering name registrations stored in a blockchain requires prohibitively high computing resources because re-writing blockchain data requires proof-of-work [8]. Before our work, it was common practice to start new blockchains (by forking them from Bitcoin) to introduce new functionality and make modifications required by the respective service/application, which is the precise approach taken by Namecoin.

Just like DNS, there is a cost associated with registering a new name. The name registration fee discourages people from registering a lot of names that they don't actually intend to use. In Namecoin, the recipient of registration fees is a "black hole" cryptographic address from which money cannot be retrieved [31]. Namecoin defines a pricing function for how the cost of name registrations changes over time. Namecoin supports multiple namespaces (like TLDs in DNS), and the same rules for pricing and name expiration apply to all namespaces. By convention, the $d/$ namespace is used for domain names.

In Namecoin, name registration uses a two-phase commit method where a user first **pre-orders** a name hash and then **registers** the name-value pair by revealing the actual *name* and the associated *value*. This is done to avoid front-running of unconfirmed name registrations [31]. Name registrations expire after a fixed amount of time, measured in new blocks written (currently 36,000 blocks, which translates to roughly 8 months). Namecoin also supports updating the value associated with a name, as well as ownership transfers.

2.3 Blockchain-based PKI System

We used Namecoin to build a PKI and identity system, called *Blockstack ID*, by starting a new namespace $u/$ on it. We defined the format for publishing public keys, like PGP [53], along with other profile data in the blockchain [3]. This is similar to defining the format of DNS records. Namecoin already had support for human-readable names and registering name-value pairs. Namecoin provided limited storage per name-value pair and we extended the storage capacity by using linked lists of name-value pairs. We also improved the read performance of Namecoin for our production system.

We launched a web service [43] in March 2014 that enabled people to easily register names on the $u/$ namespace of Namecoin and associate profile data with them. In our web service, we first register the name on the user's behalf (and also pay the registration fee) and then transfer the name to a cryptocurrency address owned by the user. Our implementation is one of the first production PKI systems that binds user identities to public keys using a blockchain (see Section 6 for other systems). All registered names have an ECDSA public key [28] binding by default, and a subset of users have added their PGP keys as well. According to a study by Harry et al. [31], our system has the second largest namespace on Namecoin by volume and the largest by number of active users.

3 Lessons from Namecoin Deployment

In this section, we describe our experience with running a year-long production system on Namecoin and the challenges we faced. We present lessons we learned for securing blockchains (§3.1, §3.3, and §3.5), improving network reliability (§3.2), and for deploying consensus breaking changing (§3.4). These lessons directly influenced the design of our new system, Blockstack (§4).

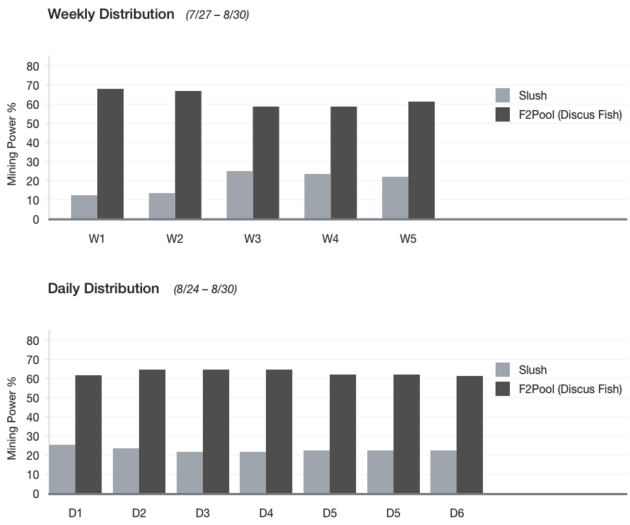


Figure 1: Weekly and daily mining distribution.

3.1 Blockchain Security

The security of name ownership is tied to the security of both the underlying blockchain and the software powering it. The most important factor in the security of a blockchain is the total cost of attacking the blockchain and tampering with recently written data. Miners often pool their resources to form a *mining pool*, which is essentially a super node on the network (a lot of computational power behind a single miner node). If the amount of computational power under the control of a single miner (or pool) is more than the rest of the network, called a *51% attack*, then that miner has the ability to attack the network and rewrite recent blockchain history, censor transactions (e.g., for name registrations), and steal cryptocurrency using *double spend* attacks [49]. This is because it will win the leader election for a majority of the time, and produce a blockchain history with more proof-of-work than any disagreeing miner. The more expensive it is to control a majority of the compute power on a blockchain, the more secure the blockchain.

We noticed in late 2014 that a single mining pool consistently had more than 51% of the compute power on Namecoin. Recently, the situation has been even worse, with a single mining pool controlling over 60% of Namecoin's compute power. Figure 1 shows the weekly and daily distribution of mining power for the month of August 2015, right before we migrated our system away from Namecoin. In fact, we have observed F2Pool (also known as Discus Fish) control up to 75% of compute power in a particular week. At such concentration, Namecoin is effectively controlled by a single party; F2Pool gets to write most of the new blocks and can undermine the security of the blockchain at will.

Other than raw hashing power, software bugs can also introduce security problems, e.g., a Namecoin bug allowed people to steal names from anyone [26]. Denial-of-service attacks are another attack vector; the more peers a cryptocurrency network has, the more resilient the network is to denial-of-service attacks.

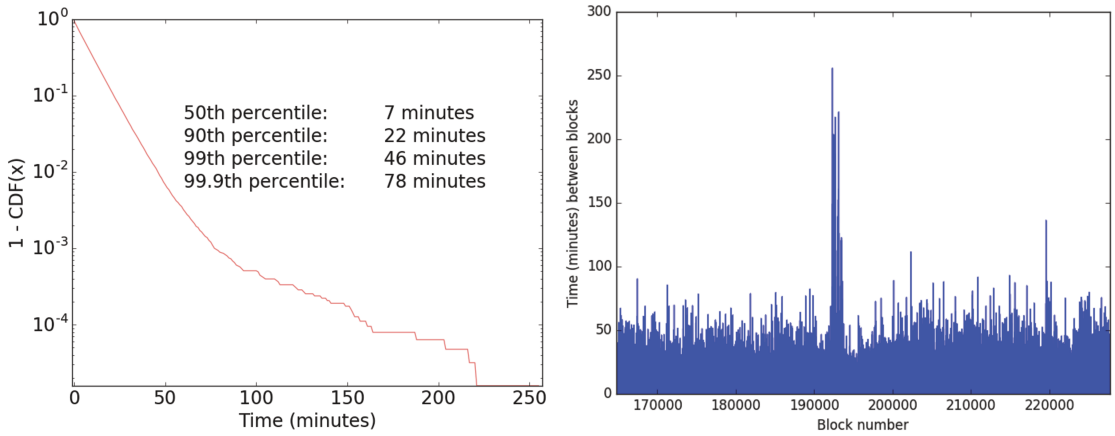
Bitcoin currently has the largest amount of computational power securing the blockchain data. Bitcoin's codebase is more actively developed with more bug bounties than other blockchains. Namecoin has many fewer peer nodes than Bitcoin (170 vs. 4,600 in Jan 2016 [4]), which makes it more vulnerable to DDoS attacks as well. The Bitcoin blockchain is currently by far the most secure blockchain. However, it's extremely hard to introduce new functionality to Bitcoin because that requires consensus-breaking changes (Section 3.4).

Lesson #1: There is a fundamental tradeoff between blockchain security and introducing new functionality to blockchains. Starting a new blockchain network is how developers typically introduce new functionality not provided by Bitcoin, e.g., a naming system that is of interest to many emerging applications. However, new blockchains are significantly less secure than Bitcoin. In Section 4, we introduce Blockstack to overcome this tradeoff by creating *virtualchains* that introduce new functionality as a layer on top of Bitcoin.

3.2 Network Reliability and Throughput

The throughput of our PKI system (number of entries we can register/update) is directly dependent on the throughput of the underlying blockchain. The number of new register/update operations that can be performed per hour is limited by the number of transactions that can be sent (and confirmed) on the underlying blockchain per hour. Similarly, reliability of our PKI system is impacted if the underlying blockchain cannot perform operations reliably and consistently.

Network Latency Spike: As a fork of Bitcoin, Namecoin shares many protocol properties with Bitcoin, including a 10 minute average leader election time (the "latency target") and a 1MB bandwidth limit on block size (giving throughput of $\sim$1000 transactions per block). Figure 2(a) shows that since we launched our PKI system in March 2014, Namecoin on average performed well on the network latency target. As expected, most new blocks were written within 10 and 40 minutes (similar times have also been observed on Bitcoin [14]). Figure 2(b) shows an incident in late August 2014 (at block number 192000), where network latency skyrocketed for a couple of weeks ($\sim$1000 blocks are roughly a week). After investigating the issue and having discussions with Namecoin developers, we discovered that the latency spike was caused by software issues in Namecoin. Someone on the network was sending transactions with a large



(a) CCDF of network latency (03/14 – 04/15)

(b) Network latency per new block (03/14 – 04/15)

Figure 2: Network latency spike in the Namecoin network.

number of data fields per transaction. This was causing severe performance problems for the miners and their Namecoin daemons kept crashing. Without stable miner nodes, blocks were not getting appended in a timely fashion. This shows that unexpected protocol/software issues can trigger network latency problems. During this period, we noticed a slow down in rate of new registrations of our PKI system along with a spike in user complaints.

Network Throughput Drop: In early September 2014, right after the latency spike incident we noticed that our transactions were not getting accepted for many consecutive blocks and, after a while, will get accepted in bulk in a single block that packaged a lot of transactions. We noticed that a lot of new blocks had no transactions in them. This issue persisted for over a week and Figure 3 plots the number of transactions that we were trying to send (shown as "tx target") vs. the number of transactions that were getting accepted by the network. Network latency was completely normal (shown at top of Figure 3), but network throughput went down because of no transactions in new blocks. We tried upgrading our software and rebroadcasting transactions, but the issue persisted. We concluded that there is a large mining pool that is either intentionally refusing or is unable to package transactions in the blocks it is writing. Our transactions will get packaged only when some other miner was elected to write the new block. We discuss this issue in more detail in the next section.

Lesson #2: There is currently a significant difference between the network reliability of the largest public blockchain network (Bitcoin) and network reliability of the long tail of alternate blockchains. Problems with the Bitcoin network impact a lot more users and businesses than Namecoin and other smaller blockchains. Our work is the first analysis of the network reliability of a blockchain other than Bitcoin.

3.3 Potential Selfish Mining

The signs that we noticed in the incident where miners were not accepting our transactions (Section 3.2) looked similar to a selfish mining attack [22]. In a selfish mining attack, (a) a miner needs to have a large amount of mining power (more than 33%), (b) people would notice long delay in blocks followed by blocks in very quick succession, and (c) there will be a lot of rejected blocks. We noticed all these signs, and believe that the unusually high computing power of a single miner led to conditions similar to selfish mining. That is, the miner was able to work on new blocks faster than the others and append them in rapid succession.

Lesson #3: Selfish-mining is not just a theoretical attack, but selfish-mining like behavior can already be observed in production blockchains. This is the first time that data collected from a production network shows signs of selfish-mining like behavior, regardless of if the miner was intentionally attacking the network or not.

3.4 Consensus-breaking Changes

For major updates, like changes to name pricing, Namecoin requires a "hard fork" in which everyone on the network must upgrade their software, and nodes on previous versions can no longer participate in the network. Anecdotal evidence suggests that it's hard to get miners to upgrade their software because they don't have enough incentive to spend engineering hours on maintaining a small cryptocurrency like Namecoin, which is not their main reason for operating a mining pool. Our experience monitoring the Namecoin network showed that whenever

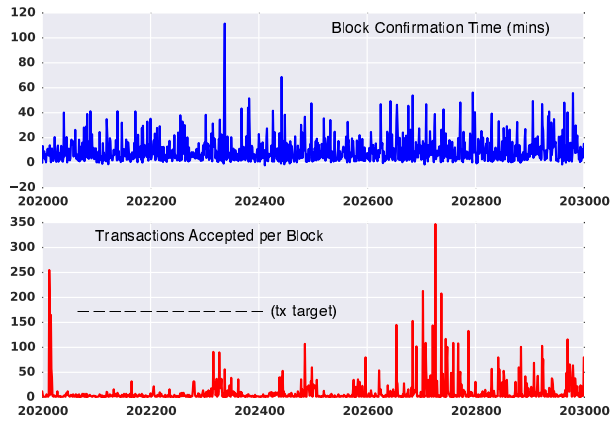


Figure 3: *Throughput drop in the Namecoin network. The number of transactions we were trying to send is shown as "tx target".*

software updates were issued on Namecoin, there was a considerable fluctuation of computing power. In fact, we noticed that after the recent upgrade to Namecoin Core [42], a major upgrade to the Namecoin daemon, many miners dropped out and never came back online.

Lesson #4: Other than the engineering problems, consensus-breaking changes are complicated because of fundamental incentive structures of the parties involved. System designers have never dealt with consensus-breaking changes before cryptocurrencies; it's a novel challenge. For software upgrades to cryptocurrency networks we should: (a) separate consensus-breaking upgrades from other upgrades as a software engineering rule (Bitcoin recently started doing this in their codebase [11]) and (b) try to align miner incentives given their cost (engineering time) of software upgrades. This resistance to upgrades is present in Bitcoin as well, but is exaggerated for the long tail of smaller blockchains. In Section 4, we describe how Blockstack accounts for these incentives and introduces new features without requiring miners to upgrade software.

3.5 Failure of Merged Mining

The security of a blockchain depends on the relative compute power of miners and the cost for a single party to employ more computing power than the rest of the network. New, smaller blockchains have a *bootstrapping problem*, however: in the initial days of a new blockchain, it would be relatively easy for a single party to take it over, since the total compute power on the blockchain is not yet large enough to prevent this. To address this problem, Satoshi Nakamoto (author of Bitcoin) introduced "merged mining" [39], where an alternate blockchain can allow Bitcoin miners to participate in the new network without requiring them to spend extra compute cycles. The miners can make extra profits on the new blockchain without adding computational overhead. With a merge-mined cryptocurrency, the security of the blockchain is typically a subset of the "main blockchain," because in practice not all miners of the main blockchain go through the trouble of setting up merged mining.

Namecoin switched to merged mining with Bitcoin to increase security of its blockchain [31]. Namecoin is the oldest and largest merged-mined cryptocurrency and inspired other cryptocurrencies to consider it as well. One of our key findings is that merged mining is currently failing in practice: the leading merged-mined blockchain, Namecoin, is vulnerable to the 51% attack (Section 3.1). Moreover, merged-mining provided a false sense of security. F2Pool controls 30-35% computing power of Bitcoin, but over 60% of Namecoin's computing power through merged mining, leaving Namecoin vulnerable to a 51% attack. Unless the merged mined cryptocurrency can consistently attract a very high ratio of main blockchain miners to support their software, merged mining will not keep it safe from 51% attacks.

Lesson #5: At the current stage in the evolution of blockchains, there are not enough compute cycles dedicated to mining to support multiple secure blockchains. The respective financial capital attached to blockchains relative to Bitcoin supports this argument: as of Feb 2016, Bitcoin has a 5.9 billion USD market cap, which accounts for 89% of the market cap of all 500+ blockchains combined, while the second and third largest market caps are 3.2% and 2.6% of Bitcoin, respectively [6]. While multiple secure blockchains may be possible after the technology matures and enjoys wider adoption, in the near future, Bitcoin's blockchain is the only one that is prohibitively expensive to attack.

3.6 Summary

Namecoin deserves full credit for originally solving naming on a blockchain. But after considering all of the above factors, it was an easy decision to move our PKI system from Namecoin to Bitcoin. In general, after our experience, we strongly believe that decentralized applications and services need to be on the largest, most secure, and most actively maintained blockchain. Currently, no other blockchain even comes close to Bitcoin in terms of these security requirements.

4 Design of Blockstack

Blockstack is designed to implement a naming system with human-readable names in a layer above the blockchain. In this section, we describe how Blockstack

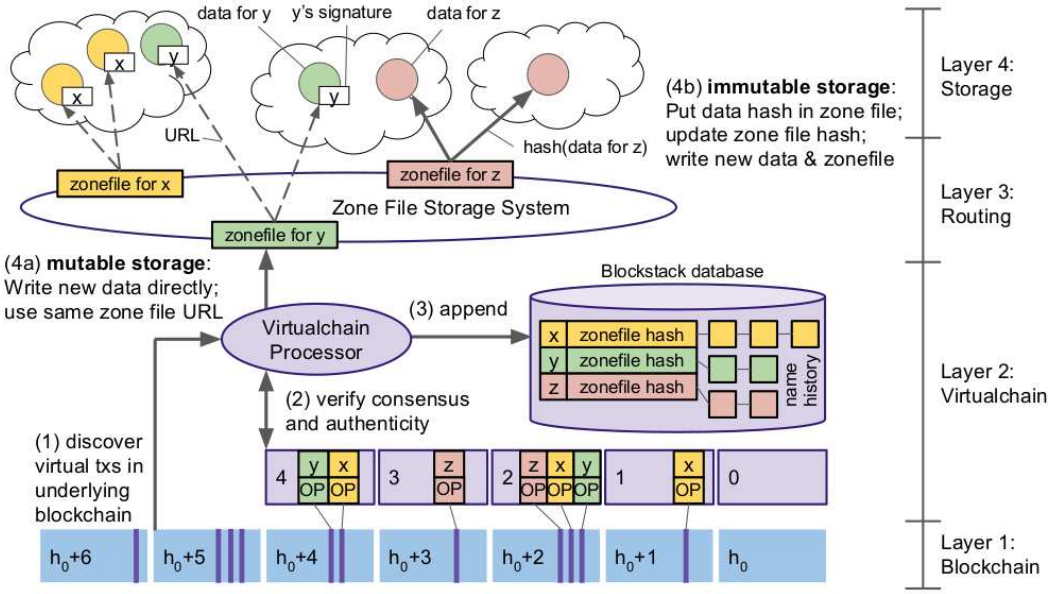


Figure 4: *Overview of Blockstack's architecture. Blockchain records give (name, hash) mappings. Hashes are looked up in routing layer to discover routes to data. Data, signed by name owner's public-key, is stored in cloud storage.*

uses the underlying blockchain, and present how it copes with technical limitations of contemporary blockchains.

4.1 Challenges

Building systems with blockchains presents challenges:

• **Limits on Data Storage:** Individual blockchain records are typically on the order of kilobytes [49] and cannot hold much data. Moreover, the blockchain's log structure implies that *all* state changes are recorded in the blockchain. All nodes participating in the network need to maintain a full copy of the blockchain, limiting the total size of blockchains to what current commodity hardware can support. As of May 2016, Bitcoin nodes need to dedicate 69GB total disk space to blockchain data for staying synchronized with the network.

• **Slow Writes:** The transaction processing rate is capped by the blockchain's write propagation and leader election protocol, and it is pegged to the rate at which new blocks are announced by leader nodes, called *miners* in many blockchain networks [14]. New transactions can take several minutes to a few hours to be accepted.

• **Limited Bandwidth:** The total number of transactions per block is limited by the *block size* of blockchains. To maintain fairness and to give all nodes a chance to become leader in the next round, all nodes should receive a newly announced block at roughly the same time. Therefore, the *block size* is typically limited by average uplink bandwidth of nodes [14]. For Bitcoin the current bandwidth is 1MB (∼1000 transactions) per new block.

• **Endless Ledger:** The integrity of blockchains depends on the ability for anyone to audit them back to the first block. As the system makes forward progress and issues new blocks, the cost of an audit grows linearly with time, which makes booting up new nodes progressively more time consuming. We call this the *endless ledger problem*. Bitcoin's blockchain currently has ∼413,000 blocks and new nodes take 1-3 days to download the blockchain from Bitcoin peers, verify it, and boot up.

4.2 Architecture Overview

Blockstack maintains a naming system as a separate logical layer on top of the underlying blockchain on which it operates. Blockstack uses the underlying blockchain to achieve consensus on the state of this naming system and bind names to data records. Specifically, it uses the underlying blockchain as a communication channel for announcing state changes, as any changes to the state of name-value pairs can only be announced in new blockchain blocks. Relying on the consensus protocol of the underlying blockchain, Blockstack can provide a total ordering for all operations supported by the naming system, like name registrations, updates and transfers.

Separation of the Control and Data Plane: Blockstack decouples the security of name registration and name ownership from the availability of data associated with names by separating the control and data planes.

The control plane defines the protocol for registering human-readable *names*, creating (*name*,*hash*) bindings, and creating bindings to owning cryptographic keypairs.

The control plane consists of a blockchain and a logically separate layer on top, called a "virtualchain".

The data plane is responsible for data storage and availability. It consists of (a) zone files for discovering data by hash or URL, and (b) external storage systems for storing data (such as S3, IPFS [29], and Syndicate [30]). Data values are signed by the public keys of the respective name owners. Clients read data values from the data plane and verify their authenticity by checking that either the data's hash is in the zone file, or the data includes a signature with the name owner's public key.

We believe this separation is a significant improvement over Namecoin, which implements both the control and the data plane at the blockchain level. Our design not only significantly increases the data storage capacity of the system, but also allows each layer to evolve and improve independently of the other.

Agnostic of the Underlying Blockchain: The design of Blockstack does not put any limitations on which blockchain can be used with it. Any blockchain can be used, but the security and reliability properties are directly dependent on the underlying blockchain. We believe that the ability to *migrate* from one blockchain to another is an important design choice as it allows for the larger system to survive, even when the underlying blockchain is compromised. Currently, Blockstack core developers decide which underlying blockchain(s) to support in which version of the software. Individual applications can decide to run the software version of their choice and keep their namespace on a particular blockchain, if they prefer not to migrate. Section 5 gives more details on the migration process.

Ability to Construct State Machines: A key contribution of Blockstack is the introduction of a logically separate layer on top of a blockchain that can construct an arbitrary *state machine* after processing information from the underlying blockchain. We call this layer a *virtualchain* (Section 4.3.2). A *virtualchain* treats transactions from the underlying blockchain as inputs to the state machine and valid inputs trigger state changes. At any given time, where time is defined by the block number, the state machine can be in exactly one global state. Time moves forward as new blocks are written in the underlying blockchain and the global state is updated. **A virtualchain can introduce new types of state machines without requiring any changes from the underlying blockchain.** Introducing new state machines directly in a blockchain requires peers to upgrade. Upgrades potentially break consensus and cause forks. In practice, they are difficult to orchestrate [14]. Currently, Blockstack introduces a state machine that represents the global state of a naming system, including who owns a particular name and what data is associated with a name. Further, it's possible to use the *virtualchain* concept to define other types of state machines as well.

4.3 Blockstack Layers

Blockstack introduces new functionality on top of blockchains by defining a set of new operations that are otherwise not supported by the blockchain. Blockstack has four layers, with two layers (blockchain layer and *virtualchain* layer) in the control plane and two layers (routing layer and data storage layer) in the data plane.

4.3.1 Layer 1: Blockchain Layer

The blockchain occupies the lowest tier, and serves two purposes: it stores the sequence of Blockstack operations and it provides consensus on the order in which the operations were written. Blockstack operations are encoded in transactions on the underlying blockchain.

4.3.2 Layer 2: Virtualchain Layer

Above the blockchain is a *virtualchain*, which defines new operations without requiring changes to the underlying blockchain. Only Blockstack nodes are aware of this layer and underlying blockchain nodes are agnostic to it. Blockstack operations are defined in the *virtualchain* layer and are encoded in valid blockchain transactions as additional metadata. Blockchain nodes do see the raw transactions, but the logic to process Blockstack operations only exists at the *virtualchain* level.

The rules for accepting or rejecting Blockstack operations are also defined in the *virtualchain*. Accepted operations are processed by the *virtualchain* to construct a database that stores information on the global state of the system along with state changes at any given blockchain block. Virtualchains can be used to build a variety of state machines. Currently, Blockstack defines only a single state machine - a global naming and storage system.

4.3.3 Layer 3: Routing Layer

Blockstack separates the task of *routing* requests (i.e., how to discover data) from the actual storage of data. This avoids the need for the system to adopt any particular storage service from the onset, and instead allows multiple storage providers to coexist, including both commercial cloud storage and peer-to-peer systems.

Blockstack uses *zone files* for storing routing information, which are identical to DNS zone files in their format. The *virtualchain* binds *names* to respective $hash(zonefile)$ and stores these bindings in the control plane, whereas the *zone files* themselves are stored in the routing layer. **Users do not need to trust the routing layer** because the integrity of *zone files* can be verified by checking the $hash(zonefile)$ in the control plane.

In Blockstack's current implementation, nodes form a DHT-based peer network [36] for storing *zone files*. The DHT only stores *zone files* if $hash(zonefile)$ was previously announced in the blockchain. This effectively whitelists the data that can be stored in the DHT. Due to space constraints, we omit most details of our DHT storage from this paper; the key aspect relevant to the design of Blockstack is that routes (irrespective of where they are fetched from) can be verified and therefore cannot be tampered with. Further, most production servers maintain a full copy of all *zone files* since the size of *zone files* is relatively small (4KB per file). Keeping a full copy of routing data introduces only a marginal storage cost on top of storing the blockchain data.

4.3.4 Layer 4: Storage Layer

The top-most layer is the storage layer, which hosts the actual data values of name-value pairs. All stored data values are signed by the key of the respective owner of a *name*. By storing data values outside of the blockchain, Blockstack allows values of arbitrary size and allows for a variety of storage backends. **Users do not need to trust the storage layer** because they can verify the integrity of the data values in the control plane.

There are two modes of using the storage layer and they differ in how the integrity of data values is verified; Blockstack supports both storage modes simultaneously.

(a) Mutable Storage is the default mode of operation for the storage layer. The user's zone file contains a URI record that points to the data, and the data is constructed to include a signature from the user's private key. Writing the data involves signing and replicating the data (but not the zone file), and reading the data involves fetching the zone file and data, verifying that $hash(zonefile)$ matches the hash in Blockstack, and verifying the data's signature with the user's public key. This allows for writes to be as fast as the signature algorithm and underlying storage system allows, since updating the data does not alter the zone file and thus does not require any blockchain transactions. However, readers and writers must employ a data versioning scheme to avoid consuming stale data.

(b) Immutable Storage is similar to mutable storage, but additionally puts a TXT record in the zone file that contains $hash(data)$. Readers verify data integrity by fetching the data and checking that $hash(data)$ is in the zone file, in addition to verifying the data's signature and the zone file's authenticity. This mode is suitable for data values that don't change often and where it's important to verify that readers see the latest version of the data value. For immutable storage, updates to data values require a new transaction on the underlying blockchain (since the zone file must be modified to include the new hash), making data updates much slower than mutable storage.

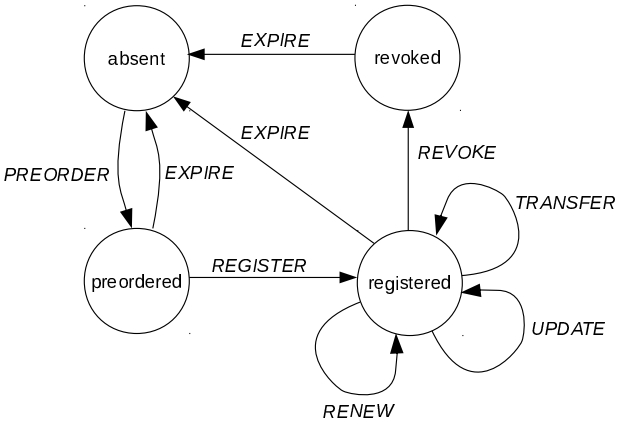


Figure 5: *States and transitions for a name.*

4.4 Naming System

Blockstack uses its four tiers to implement a complete naming system. Names are owned by cryptographic addresses of the underlying blockchain and their associated private keys (e.g. ECDSA-based private keys used in Bitcoin [14]). As with Namecoin, a user *preorders* and then *registers* a name in two steps in order to claim a name without revealing it to the world first and allowing an attacker to race the user in claiming the name. The first user to successfully write both a preorder and a register transaction is granted ownership of the name. Further, any previous preorders become invalid when a name is registered. Once a name is registered, a user can *update* the name-value pair by sending an update transaction and uploading the new value to the storage layer, changing the name-value binding. Name *transfer* operations simply change the address that is allowed to sign subsequent transactions, while *revoke* operations disable any further operations for names.

The naming system is implemented by defining a state machine and rules for state transitions in the *virtualchain*. Figure 5 shows the different states a *name* can be in and how state transitions work. Names are organized into *namespaces*, which are the functional equivalent of top-level domains in DNS—they define the costs and renewal rates of names. Like names, namespaces must be preordered and then registered. Expired names can be re-registered and names can be *revoked* such that they cannot be re-registered for a certain period of time.

4.4.1 Pricing Functions for Namespaces

Anyone can create a namespace or register names in a namespace, as there is no central party to stop someone from doing so. *Pricing functions* define how expensive it

is to create a namespace or to register names in a namespace. Defining intelligent pricing functions is a way to prevent "land grabs" and stop people from registering a lot of namespaces or names that they don't intend to actually use. Blockstack enables people to create namespaces with sophisticated pricing functions. For example, we use the *.id* namespace for our PKI system and created the *.id* namespace with a pricing function where (a) the price of a name drops with an increase in name length and (b) introducing non-alphabetic characters in names also drops the price. With this pricing function, the price of *john.id* > *johnadam.id* > *john0001.id*. The function is generally inspired by the observation that short names with alphabetics only are considered more desirable on namespaces like the one for Twitter usernames. It's possible to create namespaces where name registrations are free as well. Further, we expect that in the future there will be a reseller market for names, just as there is for DNS. A detailed discussion of pricing functions is out of the scope of this paper, and the reader is encouraged to see [31] for more details on pricing functions.

Like names, namespaces also have a *pricing function* [13]. **To start the first namespace on Blockstack, the *.id* namespace, we paid $10,000 in bitcoins to the network. This shows that even the developers of this decentralized system have to follow Blockstack rules and pay appropriate fees.**

4.5 Simple Name Verification

Blockstack nodes can independently calculate a *consensus hash* at any blockchain block. Consensus hashes help Blockstack nodes figure out if they have the same view of the global state at any given block. Each consensus hash $CH(h)$ is constructed from block h's sequence of virtualchain operations V_h, as well a geometric series of prior consensus hashes P_h defined by:

$$CH(h) = hash(V_h + P_h)$$

where

$$P_h = \{CH(h-2^i) | i \in \mathbb{N}, h-2^i >= h_0\}$$

and h_0 is the first block. Other than detecting that two Blockstack nodes have the same global view, consensus hashes also address the *endless ledger problem* (defined in Section 4.1). As the underlying blockchain grows in size, new Blockstack nodes need to process more and more blocks before they boot up.

A new Blockstack node can bootstrap by using an untrusted database of state information at a given block number, combined with a trusted consensus hash $CH(h)$ of the same block number. The block number is also termed the *block height* in the literature, and it increases with each new block. A new Blockstack node can reconstruct the virtualchain from the untrusted database and reprocess virtualchain operations at each blockchain block, recalculating each $CH(h)$ along the way. If the final consensus hash matches the trusted consensus hash at h_n, then the database associated with h_n is trustworthy and the node can start processing blocks after h_n. This is much faster than the traditional approach of starting from the first block h_0 and fetching all transactions, even though most of them will be discarded.

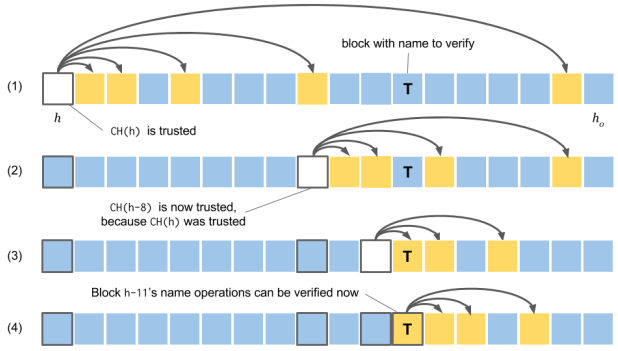


Figure 6: *Overview of SNV. Example SNV query of a record in block* ***T****.*

The process of verifying the authenticity of a prior name operation with a later trusted consensus hash is called *Simplified Name Verification (SNV)*. SNV enables support for "thin clients," which can query the past state of the system without running Blockstack nodes or having access to the full blockchain history. Support for thin clients is important for users on mobile devices.

As such, if a user trusts that $CH(h)$ is authentic, then she can query and verify the *virtualchain* operations V_h and previous consensus hash P_h for block h. The construction of $CH(h)$ allows a user to verify the authenticity of any *virtualchain* operation from a block with height $h_{prior} < h$, using only a logarithmic number of queries. Figure 6 shows an example SNV query. Each row represents the blockchain, in decreasing block height order from left to right ($h > h_0$). Here, the user is able to verify the authenticity of a name operation in a target block (marked with a T). In each step, the user recursively trusts the consensus hash for the white outlined blocks.

On current commodity hardware, booting new Blockstack nodes can take 1-2 hours with SNV, compared to 2-4 days without SNV. Further engineering improvements in our Python implementation are currently possible.

4.6 Performance of Reads and Writes

We evaluated the performance of reads and writes through Blockstack to demonstrate that it reads and

Figure 7: *Performance overhead of Blockstack.*

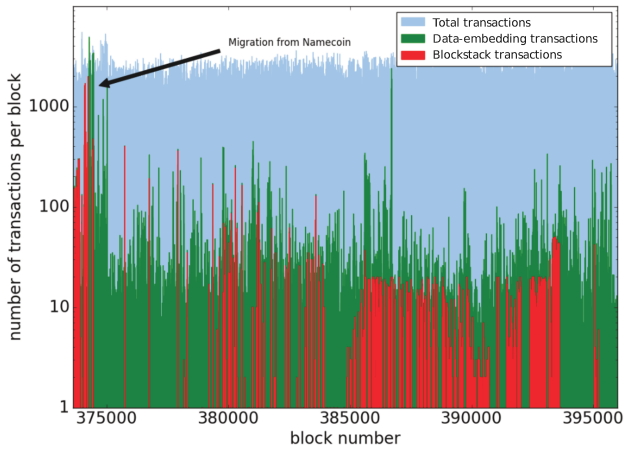


Figure 8: *All data-embedding transactions on Bitcoin; it's already becoming a frequent use case.*

writes files at competitive rates with the underlying storage. Blockstack adds a negligible constant storage space overhead per file (roughly 5% larger files with compression). There is CPU overhead for encryption and compression, but since the file size difference is very small, the network performance for reads and writes is similar to directly accessing the underlying storage service.

The write performance and overheads associated with uploading 1, 10, and 100 megabyte files to Amazon S3 is shown in Figure 7 (each trial was performed 25 times). We see that the CPU-bound overhead is in the order of 2 seconds for large (100MB) files. Many low-hanging performance optimizations still remain in our implementation. Similarly, reading encrypted files from Blockstack with S3 as storage backend is competitive with a direct read from S3 (Figure 7). We omitted the file download time to emphasize the overhead in the graph. The sources of overhead, verifying the signature and decrypting the data, are CPU-bound while in practice performance will largely be network-bound for wide-area usage.

5 Lessons from Migration to Bitcoin

We implemented Blockstack in 40,344 lines of Python code [13] and the current implementation uses Bitcoin as the underlying blockchain. In September 2015, we completed migration of 33,000 users of our production PKI system [43], from Namecoin to Blockstack/Bitcoin. These users were migrated from the *u/* namepsace on Namecoin to the *.id* namespace on Blockstack.

Blockstack embeds additional data in Bitcoin transactions using special fields dedicated for including arbitrary data [12]. Embedding additional data in Bitcoin transactions is already a popular way of defining higher-level protocols on top of Bitcoin, like Counterparty [20], Open Assets [44], etc. Figure 8 shows recent bandwidth usage of data-embedding protocols on the Bitcoin blockchain. The spike of 10,000+ transactions, near block 375000, was during our migration to Bitcoin. Our production system [43] currently accounts for most of Blockstack transactions, which is currently 26.9% of all data-embedding transactions ever made on Bitcoin [45]. Below are some observations we made while working with the Bitcoin network:

Network Throughput: Bitcoin currently supports between 3 and 7 transactions per second with a 1MB block size. Even after a year of heated debate [52] amongst the Bitcoin developers and the broader community, the block size has not been increased. We noticed these limitations first hand when we throttled our transactions so that our transactions wouldn't exceed 20-30% of Bitcoin blocks, which in turn significantly increased the amount of time it took for completing registrations. When scaling to millions of users, as opposed to thousands, even 8MB blocks will not suffice and the community needs to look into performing registrations across multiple chains [10] and novel methods for packing multiple name operations in a single transaction (an area of future work for us).

Network Attacks: During our migration to Bitcoin, a UK-based company called CoinWallet was performing a stress test on the Bitcoin network [17]. The stress test included a high volume of small transactions which had transaction amounts that were too low for miners to package in a block (due to protocol rules designed to prevent spam). This resulted in an extremely high number of unconfirmed transactions on the network and we ended up paying 2-3 times higher transaction fees to get our transactions packaged by miners. This experience shows how

a single actor can force high mining fees on the rest of the network (although in this case there was a cost factor attached to the attack). We believe that networking attacks on blockchains, like the one we experienced or other DDoS attacks [37], are likely to become more frequent. Protections against such attacks is an important area of future research.

6 Related Work

Binding names to values in naming systems is a well-explored problem space. UIA [24] gives a great overview of global naming systems and their importance. We encourage the reader to UIA [24] for a detailed background on naming systems. Unlike Namecoin [41] or Blockstack, UIA doesn't try to provide globally unique names. In authentication systems like InCommon [27], OpenID [47], and the Web's certificate authorities, a federation of authorities attests to bindings. Blockstack, however, does not require a federation.

Other than Namecoin, blockchains like Ethereum [7] and BitShares [5] also have support for human-readable names. Further, sidechains [10] enable implementation of naming systems as an alternate blockchain that is linked to the main Bitcoin blockchain. All these designs involve smaller, alternate blockchains and Blockstack directly uses the most secure blockchain (Bitcoin). Non-blockchain based PKI systems, like Keybase [34] and CONIKS [38], achieve some of the same goals as Blockstack for automating key management. The main difference is that Blockstack both provides users with direct access and control over their data, without placing trust in any specific principals while giving global state.

In networked systems it's hard to get global state without involving central trusted parties [33], Blockstack is able to provide global state (and not just approximate global state). Our system is open ("permissionless"), whereas existing wide-area systems like OceanStore [21] and Bonafide [16] have a closed ("permissioned") set of peers that use BFT agreement to make progress for the whole system. Blockstack differs from decentralized storage systems which allow open membership but offer stronger-than-eventual data consistency (like Shark [9], Pond [48], and Scatter [25]) by focusing on decentralization while supporting a wide variety of external datastores that give strong consistency.

Storage-oriented cryptocurrency blockchains like Filecoin [23], Permacoin [40], and Storj [50] seek to replace cloud storage by distributing files as sets of transactions within a blockchain, and rewarding miners for proof-of-storage (instead of proof-of-work). Blockstack differs from these systems by decoupling hosting data from operations of the underlying blockchain, allowing developers to use storage systems appropriate for their problem domains. Blockstack currently uses a simple Kademlia [36] based DHT as discovery layer, but other protocols like Chord [51] or caching optimizations like Beehive [46] are possible.

7 Conclusion

Our experience with running a production network on Namecoin, one of the oldest and largest cryptocurrency blockchains other than Bitcoin, shows how a single miner consistently had more than 51% hashing power and how network reliability was far inferior to Bitcoin. Our data shows that out of the hundreds of blockchains currently in use [6], even the more stable and more popular blockchains like Namecoin are not suitable for production use. Currently, the security of Bitcoin far outweighs other blockchains.

We have presented Blockstack, a blockchain-based naming and storage system. Blockstack introduces separate control and data planes, and by doing so, it enables the introduction of new functionality without modifying the underlying blockchain. The design of Blockstack was informed by a year of production experience from one of the largest blockchain-based production systems to date. We have made several novel improvements (like introducing the ability to do cross-chain migrations, faster bootstrapping of new nodes, and keeping data updates off the slow blockchain network) that make it easier to build decentralized services using publicly-available infrastructure. Our performance results show that Blockstack can give comparable performance to the underlying storage service and only introduces a small CPU overhead. We've released Blockstack as open-source [13].

Acknowledgments

We thank Bitcoin developers Gavin Andresen for first pushing us to build a Namecoin-like system on top of Bitcoin, Matt Corallo for feedback on protocol design, and Peter Todd for discussion around delaying rewards to miners and associated security implications. Namecoin developer Daniel Kraft helped us with debugging Namecoin issues, Ryan Castellucci with getting mining stats, Guy Lepage with generating performance graphs, and Riccardo Casatta with collecting stats on data-embedding transactions. We also thank Larry Peterson, Lakshmi Subramanian, Andrea LaPaugh, our shepherd Mohit Aron, Jon Howell, and anonymous USENIX ATC reviewers for helpful discussions and feedback. Finally, we'd like to thank the entire Blockstack open-source community (`http://blockstack.org`) for various discussions, bug reports, and Github pull requests during the production-ready release of Blockstack.

References

[1] Bitcoin hashrate. https://blockchain.info/charts/hash-rate.

[2] Bitcoin transactions per blocks. https://blockchain.info/charts/n-transactions-per-block.

[3] Blockstack ID format, version 2. https://blockstack.org/docs/blockstack-profiles.

[4] Crypto-currencies stats – active nodes. https://bitinfocharts.com.

[5] Bitshares namespaces, 2016. http://docs.bitshares.eu/namespaces/index.html.

[6] Cryptocurrency market cap, Jan. 2016. http://www.coincap.io.

[7] A next-generation smart contract and decentralized application platform, 2016. https://github.com/ethereum/wiki/wiki/White-Paper.

[8] Adam Back. Hashcash - A Denial of Service Counter-Measure. Tech report, 2002. http://www.hashcash.org/papers/hashcash.pdf.

[9] S. Annapureddy, M. J. Freedman, and D. Mazieres. Shark: Scaling file servers via cooperative caching. In *Proc. 2nd NSDI*, Boston, MA, 2005.

[10] A. Back, M. Corallo, L. Dashjr, M. Friedenbach, G. Maxwell, A. Miller, A. Poelstra, J. Timon, and P. Wuille. Enabling Blockchain Innovations with Pegged Sidechains. White paper, Blockstream, 2014. https://blockstream.com/sidechains.pdf.

[11] Bitcoin Core Developers. Bitcoin core version 0.10.0 release notes: Consensus library, Feb. 2015. https://bitcoin.org/en/release/v0.10.0.

[12] Bitcoin.org: Bitcoin developer guide, 2015. http://bitcoin.org/en/developer-guide.

[13] Blockstack source code release v0.10, 2016. http://github.com/blockstack/blockstack-server.

[14] J. Bonneau, A. Miller, J. Clark, A. Narayanan, J. A. Kroll, and E. W. Felten. Sok: Research perspectives and challenges for bitcoin and cryptocurrencies. In *2015 IEEE Symposium on Security and Privacy, SP 2015, San Jose, CA, USA, May 17-21, 2015*, pages 104–121, 2015.

[15] Chainpoint white paper. https://tierion.com/chainpoint.

[16] B.-G. Chun, P. Maniatis, S. Shenker, and J. Kubiatowicz. Tiered fault tolerance for long-term integrity. In *FAST*, pages 267–282, 2009.

[17] Coindesk. Bitcoin network stress test could occur next week, Sep 2015. http://coinde.sk/1Ku5oWc.

[18] Coindesk. State of bitcoin 2015: Ecosystem grows despite price decline, 2015. http://coinde.sk/1tJDDvv.

[19] Coindesk. State of blockchain q1 2016: Blockchain funding overtakes bitcoin, May 2016. http://www.coindesk.com/state-of-blockchain-q1-2016/.

[20] Counterparty protocol specifications. http://counterparty.io/docs/protocol_specification/.

[21] P. Eaton, H. Weatherspoon, and J. Kubiatowicz. Efficiently binding data to owners in distributed content-addressable storage systems. In *Security in Storage Workshop, 2005. SISW'05. Third IEEE International*, pages 12–pp. IEEE, 2005.

[22] I. Eyal and E. G. Sirer. Majority is not enough: Bitcoin mining is vulnerable. *CoRR*, abs/1311.0243, 2013.

[23] Filecoin: A Cryptocurrency Operated File Network. Tech report, 2014. http://filecoin.io/filecoin.pdf.

[24] B. Ford, J. Strauss, C. Lesniewski-Laas, S. Rhea, F. Kaashoek, and R. Morris. Persistent personal names for globally connected mobile devices. In *Proceedings of the 7th USENIX Symposium on Operating Systems Design and Implementation (OSDI '06)*, Seattle, Washington, Nov. 2006.

[25] L. Glendenning, I. Beschastnikh, A. Krishnamurthy, and T. Anderson. Scalable consistency in scatter. In *Proceedings of the Twenty-Third ACM Symposium on Operating Systems Principles*, pages 15–28. ACM, 2011.

[26] M. Gronager. Namecoin was stillborn, i had to switch off life-support, Oct 2013. https://bitcointalk.org/index.php?topic=310954.

[27] Incommon federation. https://www.incommon.org/federation/.

[28] D. Johnson, A. Menezes, and S. A. Vanstone. The elliptic curve digital signature algorithm (ecdsa). *Int. J. Inf. Sec.*, 1(1):36–63, 2001.

[29] Juan Benet. IPFS - Content Addressed, Versioned, P2P File System. Draft, ipfs.io, 2015. https://github.com/ipfs/papers.

[30] Jude Nelson and Larry Peterson. Syndicate: Virtual cloud storage through provider composition. In *ACM BigSystem*, 2014.

[31] H. Kalodner, M. Carlsten, P. Ellenbogen, J. Bonneau, and A. Narayanan. An empirical study of Namecoin and lessons for decentralized namespace design. *WEIS '15: Proceedings of the 14th Workshop on the Economics of Information Security*, June 2015.

[32] D. Kaminsky. Spelunking the triangle: Exploring aaron swartzs take on zookos triangle, Jan 2011. http://dankaminsky.com/2011/01/13/spelunk-tri/.

[33] S. Keshav. Efficient and decentralized computation of approximate global state. *SIGCOMM Comput. Commun. Rev.*, 36(1):69–74, Jan. 2006.

[34] Keybase. `https://keybase.io`.

[35] J. A. Kroll, I. C. Davey, and E. W. Felten. The economics of bitcoin mining, or bitcoin in the presence of adversaries. In *WEIS 2013*.

[36] P. Maymounkov and D. Mazières. Kademlia: A peer-to-peer information system based on the xor metric. In *Revised Papers from the First International Workshop on Peer-to-Peer Systems*, IPTPS '01, pages 53–65, London, UK, UK, 2002. Springer-Verlag.

[37] J. McGovern. Official statement on the last weeks ddos-attack against ghash.io mining pool, 2015. `http://bit.ly/1nu49vR`.

[38] M. S. Melara, A. Blankstein, J. Bonneau, E. W. Felten, and M. J. Freedman. CONIKS: bringing key transparency to end users. In *24th USENIX Security Symposium (USENIX Security 15)*, pages 383–398, 2015.

[39] Merge mining specification. `https://en.bitcoin.it/wiki/Merged_mining_specification`.

[40] A. Miller, A. Juels, E. Shi, B. Parno, and J. Katz. Permacoin: Repurposing bitcoin work for data preservation. In *Security and Privacy (SP), 2014 IEEE Symposium on*, pages 475–490. IEEE, 2014.

[41] Namecoin. `https://namecoin.info`.

[42] Namecoin core, required software for miners, 2015. `https://github.com/namecoin/namecoin-core`.

[43] Onename. `https://onename.com`.

[44] Open assets protocol. `http://www.openassets.org`.

[45] Statistics of usage for bitcoin op_return. `http://opreturn.org`.

[46] V. Ramasubramanian and E. G. Sirer. Beehive: O(1)lookup performance for power-law query distributions in peer-to-peer overlays. In *Proceedings of the 1st Conference on Symposium on Networked Systems Design and Implementation - Volume 1*, NSDI'04, pages 8–8, Berkeley, CA, USA, 2004. USENIX Association.

[47] D. Recordon and D. Reed. Openid 2.0: A platform for user-centric identity management. In *Proceedings of the Second ACM Workshop on Digital Identity Management*, DIM '06, pages 11–16, New York, NY, USA, 2006. ACM.

[48] S. C. Rhea, P. R. Eaton, D. Geels, H. Weatherspoon, B. Y. Zhao, and J. Kubiatowicz. Pond: The oceanstore prototype. In *FAST*, volume 3, pages 1–14, 2003.

[49] Satoshi Nakamoto. Bitcoin: A peer-to-peer electronic cash system. Tech report, 2009. `https://bitcoin.org/bitcoin.pdf`.

[50] Shawn Wilkinson and Tome Boshevski and Josh Brandof and Vitalik Buterin. Storj: A peer-to-peer cloud storage network. Tech report, storj.io, 2014. `http://storj.io/storj.pdf`.

[51] I. Stoica, R. Morris, D. Liben-Nowell, D. Karger, M. F. Kaashoek, F. Dabek, and H. Balakrishnan. Chord: A scalable peer-to-peer lookup service for internet applications. *IEEE Transactions on Networking*, 11, Feb. 2003.

[52] A. V. Wirdum. A closer look at bip100: The block size proposal bitcoin miners are rallying behind, Aug. 2015. `http://bit.ly/1VbyoXP`.

[53] P. R. Zimmermann. *The Official PGP User's Guide*. MIT Press, Cambridge, MA, USA, 1995.