SoK: Research Perspectives and Challenges for Bitcoin and Cryptocurrencies

Joseph Bonneau*†‡, Andrew Miller§, Jeremy Clark¶, Arvind Narayanan*, Joshua A. Kroll*, Edward W. Felten*
*Princeton University, †Stanford University, ‡Electronic Frontier Foundation, §University of Maryland, ¶Concordia University

***Abstract*—Bitcoin has emerged as the most successful cryptographic currency in history. Within two years of its quiet launch in 2009, Bitcoin grew to comprise billions of dollars of economic value despite only cursory analysis of the system's design. Since then a growing literature has identified hidden-but-important properties of the system, discovered attacks, proposed promising alternatives, and singled out difficult future challenges. Meanwhile a large and vibrant open-source community has proposed and deployed numerous modifications and extensions.**

We provide the first systematic exposition Bitcoin and the many related cryptocurrencies or 'altcoins.' Drawing from a scattered body of knowledge, we identify three key components of Bitcoin's design that can be decoupled. This enables a more insightful analysis of Bitcoin's properties and future stability. We map the design space for numerous proposed modifications, providing comparative analyses for alternative consensus mechanisms, currency allocation mechanisms, computational puzzles, and key management tools. We survey anonymity issues in Bitcoin and provide an evaluation framework for analyzing a variety of privacy-enhancing proposals. Finally we provide new insights on what we term disintermediation protocols, which absolve the need for trusted intermediaries in an interesting set of applications. We identify three general disintermediation strategies and provide a detailed comparison.

I. Why Bitcoin is worthy of research

比特币实际行得通，但还不能给出理论支撑

Consider two opposing viewpoints on Bitcoin in strawman form. The first is that "Bitcoin works in practice, but not in theory." At times devoted members of the Bitcoin community espouse this philosophy and criticize the security research community for failing to discover Bitcoin, not immediately recognizing its novelty, and still today dismissing it due to the lack of a rigorous theoretical foundation.

专业安全员避免研究区块链，因为没有精确的安全模型

A second viewpoint is that Bitcoin's stability relies on an unknown combination of socioeconomic factors which is hopelessly intractable to model with sufficient precision, failing to yield a convincing argument for the system's soundness. Given these difficulties, experienced security researchers may avoid Bitcoin as a topic of study, considering it prudent security engineering to only design systems with precise threat models that admit formal security proofs.

We intend to show where each of these simplistic viewpoints fail. To the first, we contend that while Bitcoin has worked surprisingly well in practice so far, there is an important role for research to play in identifying precisely *why* this has been possible, moving beyond a blind acceptance of the informal arguments presented with the system's initial proposal. Furthermore, it is crucial to understand whether Bitcoin will still "work in practice" as practices change. We expect external political and economic factors to evolve, the system must change if and when transaction volume scales, and the nature of the monetary rewards for Bitcoin miners will change over time as part of the system design. It is not enough to argue that Bitcoin has worked from 2009–2014 and will therefore continue likewise. We do not yet have sufficient understanding to conclude with confidence that Bitcoin will continue to work well in practice, which is a crucial research challenge that requires insight from computer science theory.

To the second viewpoint, we contend that Bitcoin is filling an important niche by providing a virtual currency system *without any trusted parties* and *without pre-assumed identities among the participants*. Within these constraints, the general problem of consensus in a distributed system is impossible [7], [93] without further assumptions like Bitcoin's premise that rational (greedy) behavior can be modeled and incentives can be aligned to ensure secure operation of the consensus algorithm. Yet these constraints matter in practice, both philosophically and technically, and Bitcoin's approach to consensus within this model is deeply surprising and a fundamental contribution. Bitcoin's core consensus protocol also has profound implications for many other computer security problems beyond currency[1] such as distributed naming, secure timestamping and commitment, generation of public randomness, as well as many financial problems such as self-enforcing ("smart") contracts, decentralized markets and order books, and distributed autonomous agents. In short, even though Bitcoin is not easy to model, it is worthy of considerable research attention as it may form the basis for practical solutions to exceedingly difficult and important problems.

With this dichotomy in mind, we set out to synthesize the collective knowledge from the first six years of Bitcoin's operation and development, as well as from its many derived cryptocurrencies. Our goal is both to highlight the many areas where significant innovation has already occurred, ranging from novel payment protocols to user-friendly key management, and also highlight the most important open research challenges for Bitcoin and future cryptocurrencies.

[1] As we shall see, it may not be possible to remove the currency functionality and still have a working consensus system.

II. Overview of Bitcoin

A. A Contextualized History

We refer the interested reader to existing surveys on the "first wave" of cryptocurrency research [15], [95]. In short, cryptographic currencies date back to Chaum's proposal for "untraceable payments" in 1983 [28], a system involving *bank-issued* cash in the form of blindly signed coins. Unblinded coins are transferred between users and merchants, and redeemable after the bank verifies they have not been previously redeemed. Blind signatures prevent the bank from linking users to coins, providing unlinkability akin to cash.

Throughout the 1990s, many variations and extensions of this scheme were proposed. Significant contributions include removing the need for the bank to be online at purchase time [29], allowing coins to be divided into smaller units [92] and improving efficiency [27]. Several startup companies including DigiCash [107] and Peppercoin [99] attempted to bring electronic cash protocols into practice but ultimately failed in the market. No schemes from this "first wave" of cryptocurrency research achieved significant deployment.

A key building block of Bitcoin, moderately hard "proof-of-work" puzzles, was proposed in the early 1990s for combating email spam [42] although it was never widely deployed for this purpose [71]. Many other applications followed, including proposals for a fair lottery [51], minting coins for micropayments [100], and preventing various forms of denial-of-service and abuse in anonymous networks [10]. The latter, Hashcash, was an alternative to using digital micropayments (e.g., NetBill [110] and Karma [121]). Proof-of-work was also used to detect sybil nodes in distributed peer-to-peer consensus protocols [7], similar to its current use in Bitcoin consensus.

Another essential element of Bitcoin is the public ledger, which makes double-spending detectable. In auditable e-cash [105], [106], proposed in the late 1990s, the bank maintains a public database to detect double-spending and ensure the validity of coins, however the notion of publishing the entire set of valid coins was dismissed as impractical (only a Merkle root was published instead). B-money [36], proposed in 1998, appears to be the first system where all transactions are publicly (though anonymously) broadcast. Proposed on the Cypherpunks mailing list, b-money received minimal attention from the academic research community.

Smart contracts [114], proposed in the early 1990s, enable parties to formally specify a cryptographically enforceable agreement, portending Bitcoin's scripting capabilities.

In 2008, Bitcoin was announced and a white paper penned under the pseudonym Satoshi Nakamoto was posted to the Cypherpunks mailing list [90], followed quickly by the source code of the original reference client. Bitcoin's *genesis block* was mined on or around January 3, 2009.[2] The first use of Bitcoin as a currency is thought to be a transaction in May 2010, where one user ordered pizza delivery for another in exchange for 10,000 bitcoins. Since then, an increasing number of merchants and services have adopted Bitcoin and the price has generally risen, reaching a peak of approximately US$1200 per bitcoin in late 2013.

Bitcoin's history has also been colored by its association with crime. The popular black market website Silk Road [30] operated from Feb. 2011 until Oct. 2013 when it was seized and shut down by the FBI. Botnets have found Bitcoin mining to be a supplemental source of income [57]. A current US federal court case involves a large Bitcoin-based Ponzi scheme [109]. In 2014, a computer virus called CryptoLocker extorted millions of dollars from victims by encrypting their files and demanding a Bitcoin ransom to release the decryption key [47]. Many users' Bitcoins have been lost due to theft [41] and collapsed exchanges [86].

B. A Technical Overview

We present Bitcoin's three main technical components: transactions (including scripts), the consensus protocol, and the communication network. Bitcoin is exceedingly complex—our goal is to present the system with sufficient technical depth that the literature on Bitcoin and be reviewed and evaluated in later sections of this paper. In particular, a key benefit of our three-component breakdown is that it makes evaluating and systematizing proposed changes (Sections VI & VIII) insightful by "decoupling" concepts that may be changed independently.

Sources of information on Bitcoin. Bitcoin can be difficult to define as there is no authoritative formal specification. The original Bitcoin white paper [90] provides a good overview of Bitcoin's design philosophy but many important technical details are omitted or outdated. The reference implementation `bitcoind` is considered a de facto specification, with further knowledge scattered across a series of "Bitcoin Improvement Proposals" (BIPs), forum postings, online wiki articles, the developer mailing list, and logged IRC discussions.[3] We systematize these sources into a precise technical introduction, putting forward the components of the system we consider to be independent design decisions.

1) Transactions & Scripts: The state of the world in Bitcoin is represented by a series of messages called *transactions*. Among other possibilities, transactions are foremost published to transfer currency from one user to another. It is important to note that the large (and growing) list of transactions is the only state in Bitcoin. There is no built-in notion of higher-level concepts such as users, account

[2] Famously, the first block contains the string "The Times 03/Jan/2009 Chancellor on brink of second bailout for banks."

[3] Which can be found, respectively, at: https://github.com/bitcoin/bitcoin/bips, https://bitcointalk.org/, https://bitcoin.it/, bitcoin-development@lists.sourceforge.net, irc://freenode.net/#bitcoin-dev, and irc://freenode.net/#bitcoin-wizards

balances or identities—these all exist only to the extent that they can be imputed from the list of published transactions.

Transaction format. A transaction contains an array of *inputs* and an array of *outputs*. The entire transaction is hashed using SHA-256[4] and this hash eventually[5] serves as its globally unique transaction ID. Transactions are represented using an ad hoc binary format; this is an early example of an important detail for which `bitcoind` is the de facto specification.

Each output contains an integer value representing a quantity of the Bitcoin currency. The precision of this value limits the extent to which units of the currency can be subdivided; the smallest unit is called a *satoshi*. By convention, 10^8 satoshis is considered the primary unit of currency, called one "bitcoin"[6] and denoted ₿, BTC or XBT.

Each output also has a short code snippet (in a special scripting language) called the *scriptPubKey* representing the conditions under which that transaction output can be redeemed, that is, included as an input in a later transaction.

Transaction scripts. Typically, the scriptPubKey specifies the hash of an ECDSA public key and a signature validation routine. This is called a "pay-to-pub-key-hash" transaction and the entire redeeming transaction must be signed using a key with the the specified hash. The vast majority of Bitcoin transactions are pay-to-pub-key-hash and the system is often described with this being the only possibility, although other transaction types are possible. The scripting language is an ad hoc, non-Turing-complete stack language with fewer than 200 commands called opcodes. They include support for cryptographic operations—*e.g.,* hashing data and verifying signatures. Like the transaction format, the scripting language is only specified by its implementation in `bitcoind`.

Transaction inputs refer to previous transactions by their transaction hash and the index of the output within that transaction's output array. They must also contain a code snippet which "redeems" that transaction output called the *scriptSig*. To successfully redeem a previous transaction, the scriptSig and scriptPubKey, must both execute successfully, one after the other, using the same stack. For pay-to-pub-key-hash transactions, the scriptSig is simply a complete public key (with the correct hash) and a signature.

Conservation of value. In addition to the requirements that each transaction input matches a previous transaction output and that the two scripts execute successfully, transactions are only valid if they satisfy the fundamental constraint that the sum of the values of all transaction outputs is less than or equal to the sum of the values of all inputs. We discuss the one exception in Section II-B2: *coinbase* transactions used to create new units of currency.

From transactions to ownership. By themselves, this format of transaction implies several interesting properties. There is no inherent notion of identities or individual accounts which "own" bitcoins. Ownership simply means knowing a private key which is able to make a signature that redeems certain outputs—an individual owns as many bitcoins as they can redeem. Public key hashes, as specified in pay-to-pub-key-hash transactions, effectively function as pseudonymous identities within the system and are referred to as *addresses*. No real-world name or identifying information are required.

Arguably, there is little that is deeply innovative about Bitcoin's transaction format. However, the use of a scripting language to specify redemption criteria and the realization that transactions can specify the entire state of the system are non-obvious design choices given prior cryptocurrency systems, both of which have been standard in essentially all subsequent designs. Some proposals extend the semantics of Bitcoin transactions (often by enhancing the scripting language) without changes to any other components.

2) Consensus and Mining: A transaction-based currency system would be insecure if transactions were sent directly between users to transfer funds. While the signatures would limit only the valid recipient of a previous transaction from referencing it in valid follow-up transactions, there is nothing in the transactions themselves to limit Alice from redeeming some transaction input twice in separate transactions sent to Bob and Carol, both of which would appear valid in isolation. Bitcoin takes a simple approach to solving this *double spending* attack: all transactions must be published in a global, permanent transaction log and any individual transaction output may only be redeemed in one subsequent transaction. Verifying a transaction now requires verifying the transaction's scripts as well as ensuring that it is successfully published to the log. In Bitcoin, the log is implemented as a series of *blocks* of transactions, each containing the hash of the previous block, committing this block as its sole antecedent. It is referred to as the *blockchain*.

Note that this design still requires global consensus on the contents of the blockchain. If Bob and Carol see two divergent blockchains, they will be vulnerable to double-spending attacks. One solution is to use a trusted central authority to collect transactions and publish them in signed blocks. However, this is undesirable as this authority might refuse to publish certain transactions (effectively freezing a user's assets), might go offline completely, or might intentionally *fork* the blockchain to double-spend coins.

Nakamoto consensus. Bitcoin instead establishes consensus on the blockchain through a decentralized, pseudonymous protocol dubbed *Nakamoto consensus*. This can be considered Bitcoin's core innovation and perhaps the most crucial ingredient to its success. Any party can attempt to add to the chain by collecting a set of valid pending transac-

[4] In fact, whenever Bitcoin uses SHA-256, the hash function is actually applied twice. This could be denoted SHA-256^2, but we omit this notation.

[5] Prior to publication in a block, a transaction's hash is not a unique ID due to transaction malleability [6].

[6] When capitalized "Bitcoin" refers to the entire system whereas lowercase "bitcoin" refers to one unit of currency.

tions and forming them into a block. The core ingredient is the use of a challenging computational puzzle (usually given the slight misnomer *proof of work*[7]) to determine which party's block will be considered the next block in the chain.

The process for choosing a new block is simple: the first announced valid block containing a solution to the computational puzzle is considered correct. Upon hearing of it, other participants are meant to begin working to find a followup block. If an announced block contains invalid transactions or is otherwise malformed, all other participants are meant to reject it and continue working until they have found a solution for a valid block. At any given time, the consensus blockchain is the "longest" version. Typically this is simply the branch with the most blocks, but because the mining difficulty can vary between long forks the longest chain must be defined as the one with the greatest expected difficulty to produce.[8]

It is also possible for two valid solutions to be found at approximately the same time (depending on network latency), which leads to a temporary fork during which there are two equal-length chains. Miners can choose either fork in this scenario. Due to the random nature of the computational puzzle, one blockchain will eventually be extended further than the other at which point all miners should adopt it.

While Bitcoin's original specification provided only an informal argument that eventual consensus would emerge [90], followup work has proved that, assuming an effective and timely broadcast channel and that miners controlling a majority of computational power follow the protocol faithfully, the protocol is robust and the network gradually reaches consensus [46], [84]. We will discuss this further in Section III.

Block confirmation. The gradual nature of this consensus mechanism implies that users must wait for blocks to be found in order to gain high confidence that a transaction is permanently included in the blockchain. During a fork, one of the branches will eventually be discarded after miners converge on the other. Although both branches typically include mostly the same transactions, if *conflicting* transactions are included in competing branches then one may be apparently included in the longest chain but be revoked if the other branch surpasses it. In the worst case, this can effectively enable a double spending attack [12], [60].

In theory, users can never be completely sure that a transaction won't eventually be removed by a very deep fork [13], [70]. However, if a majority of miners follow the default protocol then users can infer that a transaction is exponentially increasingly likely (see Section III-A) to end up on the eventual longest chain as more confirming blocks are found. In practice, most Bitcoin clients require 6 "confirmation" blocks before accepting that a transaction is published. The choice of 6 blocks is arbitrary, it originates from the reference client and is not based on any analysis of the probability of deep forks.

Deep forks are also prevented in an ad-hoc manner by including hard-coded blockchain prefixes (*checkpoints*) with the default Bitcoin client which clients require in any valid blockchain. Laurie [70] argues that these checkpoints demonstrate that Bitcoin is not a true decentralized consensus protocol, as they are chosen in a centralized manner.

Incentivizing correct behavior. A critical component of the protocol is that a participant who finds a block can insert a coinbase transaction minting a specified amount of currency and transferring it to an address of their choosing. Because participants are working (indeed, racing) to solve this computational puzzle in exchange for monetary rewards, they are called *miners*. This new currency, called the *block reward*, incentivizes miners to only work on valid blocks, as invalid ones will be rejected by the network and their mining rewards will then not exist in the eventually-longest blockchain. Note that from the miner's point of view, "valid" blocks are simply those which they believe the majority of other miners will accept and build upon, trumping any other specification of validity (of which there is none beyond the `bitcoind` implementation).

Because this consensus algorithm relies on monetary rewards for miners it cannot easily be used in systems with no notion of transferable value. In Bitcoin, miners receive all new currency initially and there is no other allowed mechanism for money creation. This is not strictly essential, but the consensus protocol does require some reward is issued to miners or else they have no incentive to find valid blocks and solve the difficult computational puzzle.

Mining details. The computational puzzle itself requires finding a partial pre-image for SHA-256, a cryptographic hash function. Specifically, the puzzle is to find a block (consisting of a list of transactions, the hash of the previous block, a timestamp and version number, plus an arbitrary nonce value) whose SHA-256 hash is less than a target value. The puzzle is often described approximately as finding a hash that starts with d consecutive zero bits.[9] The standard strategy is simply to try random nonces[10] until a solution is found (though this may not be the only strategy [34]).

The randomized nature of this puzzle is important; with a non-randomized puzzle (true proof-of-work) the most powerful individual miner could be expected to find every block first. With a randomized puzzle each miner will have a probability of finding the next block proportional to their

[7] Bitcoin's mining puzzle is not a true *proof-of-work* scheme but a probabilistic one. Finding a solution is computationally challenging on expectation, but it is possible to get lucky and find a solution with very little work.

[8] Specifically, this prevents an attacker from forking the blockchain, modifying timestamps on their fork to produce a lower difficulty, and using this lower difficulty to more easily overtake the previous longest chain.

[9] At the time of this writing $d \approx 68$.

[10] The puzzle is slightly more complicated in that the randomness is split into a 32-bit nonce in the block header and an arbitrary "extra nonce" in the coinbase transaction. Most miners proceed by choosing a random coinbase nonce and then exhausting all 2^{32} values for the header nonce.

share of the competing computational power.

The difficulty of the puzzle is calibrated so that a new block is found, on average, once every 10 minutes. To maintain this, the difficulty is adjusted once every 2016 blocks, or approximately every two weeks, by a deterministic function of the timestamps included in the previous 2016 blocks.[11]

Mining rewards and fees. The size of the block reward is determined by a fixed schedule. Initially, each block created Ƀ50. This has since halved to Ƀ25, and is scheduled to halve roughly every four years until roughly 2140 at which point no new bitcoins will be created.

To enable this wind-down of currency creation, miners do not only profit from block rewards: they are also allowed to claim the net difference in value between all input and all output transactions in this block. For users, a block with greater input value than output value thus includes a *transaction fee* paid to the miners.

To date, transaction fees have primarily been used to discourage overuse of the network with many small transactions (called *penny flooding*) and have never provided more than 1–2% of mining revenue [87]. Fee values have primarily been determined by defaults configured in the reference client [87], with a small number of users opting to pay higher fees to have their transactions published more quickly.

Mining pools. In practice, miners often collaborate in mining *pools* [102] to lower the variance of their revenue by sharing rewards with a group of other miners. Mining pools are typically administered by a manager who, for a small fee, collects mining rewards from valid blocks found by all participating members and allocates the funds to members in proportion to the amount of work they have performed on behalf of the pool. Participating miners prove (probabilistically) the amount of work they have performed by sending *shares* which are "near-blocks" whose hash starts with a large number of zeros (say $d' = 40$) but are not valid Bitcoin blocks. Pool members receive lower variance in rewards due to risk sharing, in exchange for a small drop in expected earnings to cover the manager's fee.

Although pools were not described in the original protocol and may have been unanticipated, since 2013 the majority of mining power has been organized into pools. A number of formulas have been used to divide revenue between pool members in order to encourage loyalty and minimize "pool-hopping" while still being friendly to new members [102]. There are also several standard protocols for low-latency communication from pool operators to members [94] and between the operators of different pools [32], [74]. While the most popular pools are centrally administered, there are also *ad hoc* pools using the p2pool protocol [122].

3) Peer-to-Peer Communication Network: The final core component of Bitcoin is its communication network. Essentially, it is a decentralized, ad hoc peer-to-peer broadcast network used to announce new transactions and proposed blocks. Generally, this is the least innovative of the three components and few altcoins have made substantial changes.

Impact on consensus. The performance and stability of the network has an important impact on the consensus protocol for two reasons. First, any latency between the discovery of a block and its receipt by all other nodes increases the possibility of a temporary fork. Fear of frequent forks motivated the choice of 10 minutes as the block creation time in the original design. Second, a malicious miner who is able to control a substantial portion of the network may attempt to favor the broadcast of their own blocks, increasing the likelihood of their blocks "winning" a fork and thus increasing their expected mining rewards. Similarly, any party able to censor the network can selectively block transmissions and freeze assets. Thus it is important for Bitcoin to have a broadcast network which is decentralized (fitting with its overall design), low latency, and where it is difficult to censor or delay messages.

Network topology and discovery. Any node can join the network by connecting to a random sample of other nodes. By default, each node attempts to make 8 outgoing connections and is prepared to receive up to 125 incoming connections. Nodes behind a NAT, such as mobile clients, are unable to receive incoming connections. Peers who join the network initially need a way to find out about other peers. Like many other peer-to-peer networks, Bitcoin achieves this through the use of dedicated directory servers or "seed nodes," the identities of whom are hard coded into the reference client; thereafter, each node maintains a list of peer addresses it knows about.

Peers also propagate information about each other through two other mechanisms: first, when a node establishes a new outgoing connection, it triggers a cascade of relay messages containing its connection information; second, upon receiving an incoming connection, a node asks its peer for a sample from its list of known addresses. This mechanism establishes a well-connected random network, with low degree yet low diameter, suitable for rapid broadcast of information through diffusion [38], [61].

Communication protocol. New blocks and pending transactions are broadcast to the entire network by *flooding*. Nodes send `INV` messages to all of their peers containing the hashes of new blocks or pending transactions whenever they first hear of them. Peers can respond by requesting the full contents of these blocks or transactions if they have not yet seen them (via a `GETDATA` message). By default nodes will only forward new data once, preventing infinite propagation; only relay transactions and blocks that are valid; only relay the first block they hear of when two blocks are found in a temporary fork; and will not broadcast pending transactions which conflict (double-spend) with pending transactions they have sent. These limits are performance optimizations

[11] Sanity checks are in place to prevent manipulated timestamps from dramatically altering the difficulty. Blocks with implausible timestamps will be rejected by the network.

designed to limit data on the network—a non-compliant node may relay invalid or conflicting data, requiring all nodes to independently validate all data they receive.

Relay policy. By default, Bitcoin nodes only relay transactions and blocks which satisfy *stricter* validation rules than what is permitted by the general transaction validity rules. The goal is to prevent various denial of service attacks—an application of the classic robustness principle "be conservative in what you send, be liberal in what you accept." For example, default nodes only relay transactions containing scripts from a very narrow whitelist of *standard transaction* types. The implication of this policy is that users of the system wishing to have non-standard transactions included in the blockchain cannot use the normal Bitcoin network, but will need to contact an agreeable miner directly.[12] Another example is that default nodes refuse to relay more than a few thousand transactions below ₿0.001 per minute as a penny-flooding defense.

III. Stability of Bitcoin

Stability for Bitcoin has been defined in many vague and sometimes conflicting ways, but it is broadly taken to mean that the system will continue to behave in a way that facilitates a functional currency as it grows and participants attempt novel attacks. We will consider notions of stability for each component of Bitcoin in turn. It remains an open question under which exact conditions Bitcoin is stable, though stability results exist under strong assumptions.

A. Stability of transaction validity rules

It is under-analyzed how participants in the Bitcoin ecosystem achieve consensus about transaction validity rules. The baseline philosophy is that the rules were set in stone by Satoshi, which we can call *canonicalism*. This has mediated some disagreements about the specified rules, such as a benign bug in the original `OP_CHECKMULTISIG` opcode which has been preserved as canonical.

However, canonicalism cannot fully explain the current rules of Bitcoin as changes have already been implemented to add new features (*e.g.,* pay-to-script-hash [2]). Rules have also been modified to fix bugs, with the best-known example occurring in March 2013 when a bug limiting the size of valid blocks was removed. This caused a fork as new, larger blocks were rejected by unpatched clients. To resolve this, the updated clients abandoned a 24-block fork and temporarily ceased including larger blocks during a two-month window for older clients to upgrade [1]. Eventually however, the bug fix won out and unpatched clients were eventually excluded despite arguably implementing the canonical rules.

Within Bitcoin itself, no process is specified for updating transaction validation rules. Without unanimity among miners, any change may permanently fork the system, with different populations considering the longest blockchain reflecting their interpretation of the rules to be authentic, regardless of its length relative to other blockchains. At this point, it would no longer be clear which version is "Bitcoin." Thus despite the popular conception of Bitcoin as a fully decentralized system, the need for rule changes (or disambiguation) means some level of governance is inherently required to maintain real-world consensus about which blockchain is considered Bitcoin [48], [64].

Currently, de facto governance is provided by the core Bitcoin developers who maintain `bitcoind`, with the Bitcoin Foundation providing a basic organizational structure and raising a small amount of funding through donations to support the development team. As with many early Internet protocols, there is as of yet no formal process for making decisions beyond rough consensus.

B. Stability of the consensus protocol

Assuming consensus on transaction validity rules, various attempts have been made to describe the properties of the consensus protocol which must hold for the blockchain to be considered stable. We systematize properties proposed by various analyses [46], [64], [84], [90] into five basic stability properties. Note that these have been given different names and different technical definitions by different authors, we only give an informal overview here.

- **Eventual consensus.** At any time, all compliant nodes will agree upon a prefix of what will become the eventual valid blockchain. We cannot require that the longest chain at any moment is entirely a prefix of the eventual blockchain, as blocks may be discarded (become "stale") due to temporary forks.
- **Exponential convergence.** The probability of a fork of depth n is $O(2^{-n})$. This gives users high confidence that a simple "k confirmations" rule will ensure their transactions are permanently included with high confidence.
- **Liveness.** New blocks will continue to be added and valid transactions with appropriate fees will be included in the blockchain within a reasonable amount of time.
- **Correctness.** All blocks in the longest chain will only include valid transactions.
- **Fairness.** On expectation, a miner with a proportion α of the total computational power will mine a proportion $\sim \alpha$ of blocks (assuming they choose valid blocks).

If all of these properties hold we can say that the system is *stable*, but it isn't clear that all are necessarily required. Users of the currency may be indifferent to the fairness property, but this property is often assumed to hold and in its absence many miners might cease to participate, which could eventually threaten other stability properties.

Liveness is perhaps the hardest property to define and to our knowledge there is no compelling formal definition. Clearly, we would like anybody willing to pay to be able to use the network, but it is not clear what exact requirements

[12] For example, Andrychowicz et al. [5] reported needing to submit their complex multiparty lottery scripts directly to the Eligius mining pool.

in terms of transaction cost and inclusion time are reasonable. Strict liveness also implies an anti-censorship property which may not be required or even desirable to some, though this is also often assumed to be a core property of Bitcoin.

Surprisingly, correctness is not actually required for a functioning currency, as participants could simply disregard any invalid transactions in the longest chain. However, correctness enables an important performance benefit in the form of SPV clients which validate only proof-of-work and not transactions (see Section IV-A).

Incentive compatibility and game theory. Nakamoto originally argued that Bitcoin will remain stable as long as all miners follow their own economic incentives [90], a property called *incentive compatibility*. Incentive compatibility has never been formally defined in the context of Bitcoin or cryptocurrencies; its prevalence as a term likely stems from its intuitive appeal and marketing value. We can consider *compliant*[13] miners whose strategy is following the default mining rules (see Section II-B2). In game-theoretic terms, if universal compliance were shown to be a Nash equilibrium, this would imply incentive compatibility for Bitcoin as no miner would have any incentive to unilaterally change strategy. This would imply a notion of weak stability if other equilibria exist and strong stability if universal compliance were the sole equilibrium. If on the other hand non-compliant strategies dominate compliance, we must ask whether the resulting strategy equilibrium leads to stability for the consensus protocol.

1) Stability with bitcoin-denominated utility: We discuss known results on Bitcoin stability, assuming that miners' objective is purely obtaining nominal bitcoins.

Simple majority compliance may not ensure fairness. An interesting non-compliant mining strategy is *temporary block withholding* [11], [45], [46],[14] in which a miner initially keeps blocks secret after finding them. If the miner finds itself two blocks ahead of the longest publicly-known chain, it can then effectively mine unopposed until the remainder of the network has caught up to within one block at which point the withheld blocks can be published. For a miner controlling at least $\alpha > 1/3$ of the mining power, this strategy dominates compliance because, when employed against compliant miners, it results in a higher expected share of the mining rewards. It may also be advantageous for an attacker with lower levels of mining power depending on how miners choose between near-simultaneously announced blocks. An attacker with a privileged network position may be able to announce their withheld blocks faster than rival blocks, demonstrating that stability does inherently rely on assumptions about the communication network.

While these results show that universal compliance is not a Nash equilibrium for many distributions of mining power, including several that have been observed in practice, there has been no evidence of a selfish mining attack occurring and it remains unknown what equilibria exist given the available strategy of temporary block withholding. If temporary withholding were performed, this would undermine fairness.

Majority compliance is an equilibrium with perfect information. Kroll et al. [64] analyzed a simplified model in which miners have perfect information about all discovered blocks (precluding any withholding). In this model, universal compliance is a Nash Equilibrium (although not unique), implying that Bitcoin is (weakly) stable.

Majority compliance implies convergence, consensus, and liveness. It can be shown that with a majority of miners behaving compliantly, a single longest (correct) chain will rapidly emerge. The original Bitcoin paper [90] models a malicious miner trying to reverse a transaction by "trying to generate an alternate chain faster than the honest chain." as a binomial random walk and shows that the attacker will eventually lose the "race" with the rest of the network. Miller and LaViola [84] and Garay et al. [46] provide more detailed formal proofs that if a majority of miners follow the compliant strategy and communication latency is small compared to the expected time to discover a block, miners will eventually agree on an ever-growing prefix of the transaction history regardless of the strategy of non-compliant miners. This is sufficient to ensure all stability properties except fairness (due to potential temporary withholding), with the exact size of the majority required depending slightly on network and other assumptions.

With a majority miner, stability is not guaranteed. It is well known that a single non-compliant miner which controls a majority of computational power could undermine fairness by collecting *all* of the mining rewards, simply by ignoring blocks found by others and building their own chain which by assumption will grow to become the longest chain. The majority miner could separately choose to undermine liveness by arbitrarily censoring transactions by refusing to include them and forking if they appear in any other block. Finally, the majority miner could undermine both convergence and eventual consensus by introducing arbitrarily long forks in the block chain, potentially to reverse and double-spend transactions for profit. All of these strategies would result in nominal profits, but since these behaviors are detectable, they may not be in a rational miner's long-term interest. We will return to this point in the next section.

If miners can collude, stability is not known. Even in the absence of a majority miner, smaller miners could potentially collude to form a cartel controlling a majority of mining power and emulating any strategy available to a single majority miner. It is not known whether such a cartel would be internally stable or whether members might be tempted to defect or if excluded miners could break it up by offering to form an alternate cartel on more favorable terms.

[13] This is sometimes called "honest" mining but we eschew this as non-compliant strategies might also reasonably be considered honest.

[14] This attack strategy was called *selfish mining* by Eyal and Sirer [45] who were among the first to analyze it.

Mining pools could possibly be a technical mechanism for cartel formation; the dynamics of miners' choice of pools and migration between pools have not been studied. It also appears no rigorous analysis has been attempted of whether and how miners might encourage others to participate in a cartel through side-payments.

Stability is not known as mining rewards decline. All of these results have used a simplified model in which each block carries a constant, fixed reward fee. The planned transition of miner revenue from block rewards to transaction fees will negate this assumption and require more complex models which take into account the distribution of available transaction fees. To our knowledge there has been no thorough analysis of how stability will be affected either in the end state of no mining rewards or in intermediate states as transaction fees become a non-negligible source of revenue.

2) Stability with externally-denominated utility: Results in the bitcoin-denominated utility model do not provide convincing justification of Bitcoin's observed stability in practice (let alone assurance of its continued stability in the future), due to the lack of observed attacks despite the existence of large mining pools potentially in the position to profit by non-compliant behavior. In reality, miners are clearly not solely interested in obtaining nominal bitcoins but in obtaining real-world profits. Modeling this requires developing a utility function for miners which incorporates not only how many bitcoins they earn, but also how effectively they can convert their bitcoins into real-world value or other currencies. Miners' strategies might affect their ability to convert bitcoin-denominated wealth into real-world value due to three related factors:

Liquidity limits. Currently, exchanges which trade Bitcoin for external currencies typically have low liquidity. Thus, an attacker may obtain a large number of bitcoins but be unable to convert them all into external value, or can only do so at a greatly reduced exchange rate.

Exchange rates in the face of attack. Some non-compliant strategies, particularly those that would affect stability in a visible way, might undermine public confidence and hence weaken demand for bitcoins in the short run. Indeed, in practice the exchange rate has been found to dip in the face of technical glitches with the system [72]. A strategy which quickly earns many nominal bitcoins but is likely to crash the exchange rate once discovered may thus be difficult to cash out before the exchange rate can react, particularly given the liquidity limits mentioned above.

Long-term stake in bitcoin-denominated mining rewards. Most large miners have an additional interest in maintaining Bitcoin's exchange rate over time because they have significant capital tied up in non-liquid mining hardware which will lose value if the exchange rate declines. If miners expect they will maintain their share of mining power far into the future with low marginal costs (*e.g.,* if a substantial portion of their operational costs are paid upfront to buy equipment), then they may avoid strategies which earn them more bitcoins but decrease the expected value of their future mining rewards. Note that this is a limiting factor even if a miner might otherwise be able to cash out stolen bitcoins more quickly than the public can react, as long as there is no effective market in which miners can sell expected future mining power.

Nakamoto outlined a version of this argument [90] to downplay the likelihood of majority-miner attacks, arguing that they would permanently damage the system (and exchange rate) and "playing by the rules" (following a compliant strategy) would be more profitable over time. In practice, the GHash.IO mining pool exceeded 50% of the network's computational capacity for an extended period in July 2014 and publicly promised to limit their capacity in the future in order to avoid damaging confidence in the system.

Unfortunately, exchange rates are difficult to capture in a tractable game-theoretic model as it inherently depends on human judgment and market confidence. Modeling the effects of exchange rates and real-world utility functions more formally is a significant open problem.

3) Stability with incentives other than mining income: At least two strategies have been analyzed which may be advantageous for a miner whose utility is not purely derived from mining rewards.

Goldfinger attacks. If a majority miner's goal is explicitly to destroy Bitcoin's stability and hence its utility as a currency, they can easily do so. Kroll et al. [64] introduced this model and named it a *Goldfinger attack*. For example, a state wishing to damage Bitcoin to avoid competition with its own currency, or an individual heavily invested in a competing currency, may be motivated to attempt such an attack. Arguably, these attacks have already been observed through *altcoin infanticide*, in which deep-forking attacks against new competing currencies with low mining capacity have been successfully mounted by Bitcoin miners.[15] If a mature futures market arises in which a miner can take a significant short position on Bitcoin's exchange rate, then Goldfinger-style attacks may be directly profitable.

Feather-forking. Miller [82] proposed the strategy of *feather-forking*, in which a miner attempts to censor a blacklist of transactions by publicly promising that if a blacklisted transaction is included in the block chain, the attacker will retaliate by ignoring the block containing the targeted transaction and attempting to fork the block chain. The attacker's fork will continue until it either outraces the main branch and wins, or falls behind by k blocks at which point the attacker will concede publication of the targeted transaction. An attacker with $\alpha < 50\%$ of the mining power will, on expectation, lose money, but will succeed in blocking a blacklisted transaction with positive probability.

[15] For example, CoiledCoin was an altcoin that was destroyed by a significant attack from Eligius, a Bitcoin mining pool [77].

However, if the attacker can convincingly show that they are serious about retaliatory forking, other miners will be motivated to shun the targeted transactions as they also lose on expectation if the attacker retaliates. Thus, an attacker may be able to enforce their blacklist with no actual cost as long as all other miners believe the attacker will perform a costly feather-forking retaliation if tested.

C. *Stability of mining pools*

Mining pools rely on participants to submit valid blocks when they are found and are vulnerable to participants submitting partial shares in exchange for compensation but withholding valid blocks to lower the pool's profitability. Though this attack has long been known, it appears self-destructive as the participant withholding a block is lowering their own earnings in addition to other pool members. However, it has been shown [33] that a large miner (or a pool) can actually profit from using some of its mining power to *infiltrate* another pool by submitting partial shares but withholding valid blocks. The benefit is that the capacity used to infiltrate will not contribute to increasing the difficulty of the mining puzzle (as blocks are not published) but can still earn profits. This strategy is advantageous to a large miner or pool across a range of mining capacities for the attacker and the infiltrated pool.

Eyal [44] provides an extended treatment of this attack and shows that, between any two pools, the resulting game is an iterated prisoner's dilemma, with a Nash equilibrium of both pools attacking but a Pareto equilibrium of neither attacking. This attack can be detected statistically if done on a large scale, which has happened at least once in the wild against the Eligius pool in June 2014 [124]. However, a clever attacker can easily obfuscate the attack using many participant addresses. Further countermeasures have been proposed but not seriously studied or deployed. As an iterated prisoner's dilemma, it is possible pools will avoid attacking each other through out-of-channel communication and the threat of retaliation.

D. *Stability of the peer-to-peer layer*

Almost all analysis of Bitcoin assumes that the peer-to-peer layer functions as specified and that, in general, a majority of participants will learn nearly all of the available protocol state information within reasonable time scales. However, Babaioff et al. [8] demonstrated that information propagation at the peer-to-peer layer is not always incentive compatible. It remains unknown whether participants internalize sufficient value from the peer-to-peer network as a public good to justify the opportunity costs of propagating information Babaioff et al. identified, or whether the information propagation equilibrium observed in the wild (in which people willingly participate in the peer-to-peer protocol) is unstable and might break down eventually.

Johnson et al. [59], [68] study whether and when participants in the peer-to-peer protocol are incentivized to engage in network-level denial-of-service attacks against others. They conclude that mining pools have an incentive to engage in attacks, that larger pools are better to attack than smaller pools and that larger pools have a greater incentive than smaller pools to attack at all. Denial-of-service attacks against pools are regularly observed in the wild, so this theoretical analysis can be backed up by observed phenomenology [120]. Others have performed measurement and simulation studies to determine the dynamics and time scale of information propagation [38], [40].

IV. Client-side Security

Bitcoin's popularity has made usable and secure key management important to a large new group of users. Unlike many other applications of cryptography, users will suffer immediate and irrevocable monetary losses if keys are lost or compromised. Hence it is an exciting and important area of research in usable security.

A. *Simplified Payment Verification (SPV) Security*

Although the reference Bitcoin client maintains a validated copy of the entire blockchain, this would impose a prohibitive burden on mobile devices. A simple observation leads to a lightweight alternative: assuming that a majority of nodes only mine on valid chains (the *correctness* property of Section III-B), then clients need validate only the proofs of work and can trust that the longest chain only contains valid transactions. Such SPV proofs [90] enable untrusted nodes to efficiently prove to lightweight clients that a transaction has been included in the agreed-upon history.

SPV is implemented in the BitcoinJ library which underlies most mobile Bitcoin clients. SPV verification requires processing an ever-growing chain of proof-of-work solutions, although optimizations are possible such as starting from hard-coded checkpoints. SPV also carries privacy concerns as it requires disclosing the set of addresses the client is interested in to third parties (see Section VII and [49]).

B. *Key Management*

Bitcoin relies on public key cryptography for user authentication while nearly all other forms of online commerce today rely on passwords or confidential credit card information. Developers of Bitcoin software have attempted a variety of approaches solve, or at least mask, longstanding usability issues with key storage and management. Eskandari et al. [43] propose a set of evaluation criteria for the usability of Bitcoin key management interfaces and conclude that current tools employ complex metaphors which do not fully capture the implications of key management actions.

Keys stored on device. Storing a pool of keys on disk directly is the simplest model, but keys may be stolen by specifically-crafted malware [75]. Some clients send change to newly created Bitcoin addresses, requiring a new backup each time the current key pool is depleted (generally without any user-interface indication when it happens), while others send change to the originating address or derive all keys from a single random seed.

Split control. To avoid a single point of failure and enhance security, bitcoins can be stored using a k-of-n *multi-signature* script which specifies n public keys. For the script to be redeemed, valid signatures must be provided from k of these n keys. A simple example is a wallet which requires both a user's laptop and mobile phone to sign before sending funds. Alternatively, funds can be stored under a single public key, but *shares* of this key can be split among n parties using threshold cryptography [50]. Threshold signatures achieve the same k-of-n security, but look like normal pay-to-pub-key-hash transactions on the blockchain and keep the parameters k and n private.

Password-protected wallets. A Bitcoin client may allow a stored key pool file (called a wallet) to be encrypted with a key derived from a user-chosen password. Password-protected wallets deter certain types of theft, additionally requiring password guessing or keystroke capture if the file is physically or digitally stolen. Password-protected wallets may mislead the user to believe that the password itself provides access to their funds on a new device.

Password-derived wallet. Key pools can be deterministically derived from a single user-chosen secret, enabling cross-device use if the secret is committed to memory (this approach is often called a *brain wallet*). Unthrottled exhaustive search of common/weak passwords is possible—rainbow tables have uncovered inadequately protected Bitcoin addresses on the blockchain. Additionally, a forgotten password will render all associated funds irrecoverable.

Offline storage. Wallets stored offline in passive portable media, such as paper or a USB thumb drive, enhance theft-protection from malware-based threats and provide a familiar mental model of physical security. However they must be updated as the key pool is depleted. For paper wallets, private keys printed in scannable form (*e.g.,* QR codes) can be stolen by passive observation of the wallet (*e.g.,* on live television [101]). Finally, offline wallets must eventually loaded keys into a device to be used, becoming susceptible to malware at that point.

Air-gapped and hardware storage. Air-gapped storage is a special case of offline storage, where the device holding the keys can perform computations, such as signing transactions for the keys it holds. Air-gapped devices can thwart certain types of thefts by never exposing keys directly to an internet-connected device. That said, unauthorized access to a transaction-signing oracle is not much different from accessing keys themselves—both allow theft. Hardware security modules (HSMs) emulate the properties of an air gap by isolating the key material from the host device and only exposing the ability to sign transactions.

Hosted wallet. Third party web services offer key storage, management, and transaction functions through standard web authentication mechanisms, such as a password or two-factor authentication. This provides the closest experience to traditional online banking, however it requires trusting the host. Many incidents of theft [41] or bankruptcy [86] by hosted wallets have been documented including over 40 events involving losses greater than ฿1000.

V. Modifying Bitcoin

We now turn our attention to proposed changes and extensions to Bitcoin. In the remainder of the paper we will evaluate and compare proposed changes, in this section we discuss available mechanisms for implementing changes.

A. Upgrading Bitcoin itself

We can distinguish changes on the following levels:

- **Hard forks.** A protocol change requires a hard fork if it enables transactions or blocks which would be considered invalid under the previous rules, such as increasing the block reward, changing the fixed block size limit, or adding a new opcode. If miners update to the new protocol, they may produce blocks that are rejected by other nodes leading to a permanent (and thus "hard") fork. Changes involving a hard fork therefore require near-unanimity to be attempted in practice.
- **Soft forks.** In contrast to a hard fork, a soft-fork change is one that's backward compatible with existing clients; generally this involves a restriction of which blocks or transactions are considered valid. Such a change requires only the support of a majority of miners to upgrade, since older clients will continue to consider their blocks valid. A miner that doesn't upgrade may waste computational work by generating blocks that the rest of the network considers invalid and ignores, but will always rejoin the longest chain found by the majority of the miners. This makes soft-forking changes much safer to introduce than hard forks. In some cases, a soft fork can be used to introduce new opcodes to the scripting language. This is possible because there are currently several *unused* opcodes that are interpreted as no-ops; including these in a transaction output may make it spendable by *anyone*, and hence they are typically avoided. However, any one of these op-codes can be given new semantics if miners decide to reject transactions that fail some condition indicated by this opcode. This is a strict narrowing of the set of acceptable transactions, and hence requires only a soft fork.In retrospect, it would have been wise to define all unused opcodes initially as no-ops, providing maximum flexibility to introduce new changes by soft-forks.
- **Relay policy updates.** Recall from Section II-B3 that nodes enforce a stricter policy in what they will relay than what they will actually accept as valid. Changing this policy or most other aspects of the communication network

require the least coordination as they can typically be done in a backwards-compatible fashion with nodes advertising their protocol version number. The default relay policy has already changed several times to add new standard transaction types such as multi-signature transactions.

B. Altcoins

Due to the limits on what can be changed about Bitcoin without a hard fork, hundreds of derivative systems, referred to as *altcoins*, have arisen with alternate design approaches. Many of these systems have forked Bitcoin's code base and maintained most of its features, although some systems (such as Ripple) are completely independent designs. Altcoins must bootstrap the initial allocation of currency to entice users to participate, which can be achieved in several ways:

- **New genesis block.** Altcoins may simply start a new blockchain from scratch, allocating funds to initial miners as Bitcoin did in its early days. This approach is now viewed warily by the cryptocurrency community due to a wave of altcoins launched by founders hoping to cash in through early mining.[16]
- **Forking Bitcoin.** To avoid privileging its founders, an altcoin might intentionally choose to fork Bitcoin at a certain point, accepting the prior transaction history and ownership of funds. Bitcoin owners would continue to have bitcoins in the original system, plus an equal amount of the new currency at the time of its founding. Technically this would function exactly like a hard fork, only without the claim that the fork is the legitimate Bitcoin blockchain. Interestingly, this approach seems not to have been attempted seriously.
- **Proof-of-burn.** A more popular approach to inheriting Bitcoin's allocation is *proof-of-burn* [113], in which users must provably destroy a quantity of bitcoins, typically by transferring funds in Bitcoin to a special address whose private key cannot be found such as the key with a hash of all zeroes. This approach has the downside of permanently lowering the quantity of bitcoins in circulation.
- **Pegged sidechains.** Most recently, a number of influential Bitcoin developers [9] proposed *sidechains*, to which bitcoins can be transferred and eventually redeemed. Adding validation rules to redeem currency from a sidechain would require at least a soft fork of Bitcoin.

Altcoins also must compete with Bitcoin for miners (and avoid Goldfinger attacks by Bitcoin miners), which can be difficult prior to the currency achieving a non-zero exchange rate. A popular approach is *merged mining*, whereby an altcoin accepts blocks if their root is included in a valid Bitcoin block, thus enabling Bitcoin miners to mine blocks in the altcoin without performing any additional work. This can quickly provide an altcoin the full mining power of Bitcoin, as many Bitcoin miners now merge mine a large number of altcoins to earn extra rewards. However, it precludes the altcoin from deviating from Bitcoin's computational puzzle.

VI. Alternative consensus protocols

Bitcoin's consensus protocol has been its most heavily debated component, due to the open questions about stability (see Section III-B), concerns about the performance and scalability of the protocol [112], and concerns that its computational puzzle wastes resources. In this section we evaluate alternative proposals for consensus, noting that in each case the stability implications of the proposed changes are unknown and alternative proposals rarely define any specific stability properties they claim to provide.

Typically, alternate consensus schemes aim to fix some specific perceived problem with Bitcoin and hope that stability arguments for Bitcoin will carry over, although given the lack of a solid model guaranteeing stability for Bitcoin itself this may be a shaky assumption.

A. Parameter changes

Bitcoin's consensus protocol incorporates many "magic constants" which were hard-coded based on initial guesswork. Nearly every altcoin has varied at least some of these parameters, yet the modifications are often controversial and we still have only a few clear guidelines on how these should be chosen and how they may affect stability.

Inter-block time and difficulty adjustment window. Bitcoin automatically adjusts the difficulty of its computational puzzle so that solutions are found (on average) ten minutes apart. This setting is constrained primarily by network latency; if the rate of solutions is too high then miners will frequently find redundant blocks before they can be propagated. On the other hand, a slower block rate directly increases the amount of time users need to wait for transaction confirmations. Bitcoin's setting is by all accounts *conservative*; all altcoins we know of have the same rate or faster (Litecoin, the second most popular system, is four times faster). There are many proposals to modify aspects of the communication network to reduce latency, allowing this parameter to be safely reduced [38], [73], [112].

Limits on block and transaction size. One of the most controversial proposed changes is to increase the 1 MB limit on the size of a block [3]. As transaction volume continues to steadily increase, this limit may soon be regularly reached. The upper bound on transaction volume is currently only 7 per second, approximately 1,000 times smaller than the peak capacity of the Visa network [53]. Once this limit is reached, transactions will effectively need to use their fees to bid for a scarce resource. This may raise the cost of using Bitcoin, potentially slowing adoption, yet increasing the revenue for miners. It may also lead users to rely on intermediaries who aggregate and settle transactions off-chain. The limit is artificial and the network's bandwidth could likely sustain

[16] For Bitcoin itself, Satoshi Nakamoto was the only miner at first and amassed over ₿1 million by 2011, most of which remains unspent.

an increase; on the other hand, increased transaction volume may exclude some participants who are bandwidth-limited. Several altcoins have raised this limit in their specification, though to our knowledge none has come close to actually utilizing this capacity so it remains unknown how it would affect operation of the system.

Monetary Policy. Bitcoin's consensus protocol effectively mandates a monetary policy through the rate at which new currency is minted and the schedule by which this rate changes. By mandating a capped amount of currency, Bitcoin effectively has a *deflationary* monetary policy which has caused multiple economists to predict the system will eventually be destabilized by a *deflationary spiral* in which nobody is willing to spend bitcoins as hoarding them is considered more profitable [52], [65]. Issuance of coins is one of the most widely varied parameters: for example, in Dogecoin inflation will continue indefinitely but at a harmonically-diminishing rate while in Freicoin [116], the inflation rate stays constant forever.

B. Alternative computational puzzles

Miller et al. [85] present a formalism for Bitcoin-compatible proof-of-work schemes called scratch-off puzzles, which essentially must be decomposable into individual attempts. This property is often referred to as the puzzle being "progress-free." This guarantees that the creator of each block is chosen by a weighted random sample of computational power, even small participants are able to receive (proportional) rewards for their contribution, and the time between consecutive puzzle solutions is sufficiently large that puzzle solutions propagate. Progress-freeness is necessary but not sufficient for the resulting consensus protocol to achieve fairness. Bitcoin's SHA-256 puzzle is progress-free, but many other constructions are possible.

ASIC-resistant puzzles. While Bitcoin mining was originally performed using general-purpose processors, the competitive nature of mining has led to a steady movement towards more powerful and energy-efficient customized hardware. Today, ASICs account for most of Bitcoin's computational power. Taylor provides an excellent survey of the technical challenges in computing SHA-256 efficiently at scale and estimates that today's ASICs are already within an order of magnitude of theoretical efficiency limits [115].

This is often perceived negatively as it moves Bitcoin mining away from its core democratic value (i.e., "one-CPU-one-vote" [90]) since most participants in the system do not own ASICs and hence perform no mining at all. Many proposals have been made for ASIC-resistant mining puzzles. Ideally, an ASIC-resistant puzzle could be effectively solved using commodity hardware, with only minor performance gains for customized hardware. The primary approach taken so far has been to design "memory-hard" puzzles which are designed to require efficient access to a large memory. The most popular memory-hard puzzle so far (used in Litecoin and Dogecoin, among others) has been the scrypt hash function [96] originally designed for cracking-resistant password hashing. Until 2014 it was unknown if it is possible to design a puzzle which is memory-hard to compute but memory-easy to verify. Tromp's cuckoo-cycle puzzle [117] appears to answer this question affirmatively.

It remains an important open problem if ASIC-resistance is possible.[17] ASICs that mine scrypt, for example, have already been released in the market and offer performance improvements comparable to SHA-256 ASICs. It is also not clear that ASIC-resistance is desirable. ASICs mean that botnets which steal cycles from commodity equipment are no longer competitive against modern mining rigs [57]. Large miners who are dependent on future Bitcoin-denominated mining rewards to recoup their investment in special-purpose ASICs with no other value [23] may also have stronger disincentives to attack, as discussed in Section III-B2.

Useful puzzles. Achieving consensus through computational puzzles appears to be wasteful both in terms of the energy consumed in computation and the energy and resources used to manufacture mining equipment. If it is possible to obtain the same level of security while utilizing the work for some additional purpose, then some of this waste can be recovered. Becker et al. [14] also posit that Bitcoin might eventually be dominated by real-world entities with control of the world's energy supplies.

A common suggestion is to use a search function with applications to scientific research, such as the popular Folding@Home [67] project. A challenge for useful puzzles is that they must be automatically generated and verified with no trusted parties, otherwise this party could choose puzzles on which they already had a head start. Kroll et al. [64] further argue that any useful puzzle must produce a pure public good, or else it might increase the amount mining by the amount it recovers, canceling out any recycling effect.

Primecoin [62] introduced the first useful puzzle in a successful altcoin. Its puzzle requires finding sequences of large prime numbers of mathematical interest and which may be used as parameters for cryptographic protocols. Miller et al. [83] proposed a puzzle incorporating proof-of-retrievability, so that mining requires storing a portion of a large public data set. In particular, if the public data set is of use to the Bitcoin network itself (e.g., the blockchain history), this approach provides additional incentives to contribute resources to the network.

Nonoutsourceable puzzles. The growth of large mining pools [78] and their potential to facilitate collusion and cartel formation has motivated the design of puzzles which cannot be easily outsourced. Members of a pool do not inherently trust each other; instead, these coalitions succeed because members can easily prove that they are performing

[17] This problem has applications in other applications including password hashing and password-based encryption, towards which the current Password Hashing Competition is attempting to identify a new standard.

mining work that, if successful, would pay the reward to the pool manager. Miller et al. [85] as well as Sirer and Eyal [111] have proposed "nonoutsourceable" puzzles that ensure whoever performs the mining work can claim the reward for themselves when a block is found, thus thwarting pools' enforcement mechanisms and making the formation of large pools between anonymous participants unlikely.

C. Virtual Mining and Proof-of-Stake

At a high level, proof-of-work puzzles exist to require expenditure of resources to perform mining. Instead of having participants "mine" by exchanging their wealth for computational resources (which are then exchanged for mining rewards), it may be possible to simply have them exchange wealth directly for the ability to choose blocks. Rather than advancing the global history by a random sample of participants weighted by computational power, the random sample is weighted by the current allocation of wealth. We can call this approach *virtual mining*. It is also sometimes called "proof-of-stake" [98].

Virtual mining offers two main benefits: first, it may be more difficult for an attacker to acquire a sufficiently large amount of digital currency than to acquire sufficiently powerful computing equipment. Second, by avoiding the consumption of *real* resources (i.e., compute cycles), no real-world resources are wasted. There have been several variations of virtual mining proposed to date, which vary mainly on the criteria by which possession of a quantity of currency makes one eligible to choose the next block:

- **Proof-of-coin-age.** Peercoin [63] proposed mining by demonstrating possession of a quantity of currency by posting a transaction (potentially to oneself, in which case the coins are not lost). Each quantity of currency is weighted by its "coin-age", the time since the coins were last moved.
- **Proof-of-deposit.** In Tendermint [66], participation in mining requires depositing coins in a time-locked bond account, during which they cannot be moved.
- **Proof-of-burn.** Stewart [113] proposed mining by *destroying* coins (sending them to an unspendable address).
- **Proof-of-activity.** Bentov et al [20] proposed having every coin owner implicitly entered into a mining lottery by default; periodically, random values from a beacon (e.g., generated from transactions occurring on the network) are used to select randomly among all the coins in the system; the current owner of the winning coin must respond with a signed message within some time interval.

There has yet to be any formalization of the model assumptions that may allow virtual mining systems to achieve security, or to compare virtual mining systems to computational puzzles in a common setting. Poelstra [97] presents a survey of the folklore arguments suggesting that consuming external resources (i.e., burning energy) is necessary for blockchain security and hence virtual mining schemes are inherently infeasible. The central argument – deemed the *nothing-at-stake* problem – is that virtual mining is susceptible to costless simulation attacks; it costs nothing to construct an alternate view of history in which the allocation of currency evolves differently. Providing a rigorous argument for or against stability of virtual mining remains an open problem.

D. Designated Authorities.

Although Bitcoin's decentralized nature has proved an effective selling point and is a fiercely-defended principle among many in the community, consensus would be drastically simpler if we could rely on a (small) number of designated authorities to receive, sequentially order, and sign transactions. This would make stability assumptions much easier to reason about and remove concerns about wasteful computation all at once. Laurie [69] first proposed using a designated list of authorities and a standard Byzantine agreement protocol.

Similar to the argument that large Bitcoin miners are not incentivized to attack due to their stake in the future exchange rate, if the authorities earn a small income by behaving honestly they would have no incentive to misbehave. Similar options are available for allocating new funds as exist for proof-of-stake solutions (Laurie's original proposal [69] suggested a lottery among the authorities). Trust in these authorities might further be limited by using a mutually untrusted set of authorities [69], using social networks to choose which authorities to trust [108] or empowering coin owners to choose their trusted authorities every time they spend coins [24]. Ripple [108] is one of the few altcoins deployed with this model; however, its stability argument remains essentially unproven.

VII. ANONYMITY & PRIVACY

Bitcoin provides a limited form of unlinkability: users may trivially create new pseudonyms (addresses) at any time. This was argued in the original specification to provide strong privacy [90], however it quickly became clear that due to the public nature of the blockchain it is sometimes possible to trace the flow of money between pseudonyms and conclude that they are likely controlled by the same individual. [56] In this section we discuss threats to privacy for Bitcoin users and proposed privacy-enhancing designs.

A. Deanonymization

The actual level of unlinkability depends heavily on implementation details that we term *idioms of use*, following [80]. For example, merchants that generate a fresh payment address for each sale ensure that received payments are not automatically linkable on the blockchain. By contrast, the customer may need to assemble the payment amount from multiple addresses she owns,[18] linking these addresses (and

[18] An alternative payment approach is to use multiple distinct merchant addresses to avoid merges [54], but this is not yet standardized or adopted.

their accompanying transactional history) together on the blockchain, given that different users rarely contribute inputs to a single, joint transaction.[19] Other idioms such as "every non-change output is controlled by a single entity" [4] and "an address is used at most once as change" [80] can also be utilized by an adversary to link together different addresses controlled by the same entity.

Linking can be applied transitively to yield clusters of addresses; this is an instance of *transaction graph analysis*. A major challenge for the adversary is that these idioms are fragile: they may yield false positives and lose accuracy over time as implementations evolve. New linking techniques may also arrive. For example, multi-signature addresses have an unintended negative effect on privacy since the multi-sig structure in a change address can be matched to the sending address even if the keys involved change [50].

To de-anonymize, the adversary must take the further step of linking address clusters to real-world identities. Meiklejohn et al. [80] were successful at identifying clusters belonging to online wallets, merchants, and other service providers since it is easy to learn at least one address associated with such entities by interacting with them. As for identifying regular users, the authors suggest that this may be easy for authorities with subpoena power since most flows pass through these centralized service providers (who typically require customer identity and keep records). Without such access, however, the adversary is limited precisely due to the centrality of flows—online wallets and other such services mix users' coins together.

Network de-anonymization. The other major target of de-anonymization efforts is the peer-to-peer network. Nodes leak their IP address when broadcasting transactions. Using an anonymity network is therefore crucial for privacy. However, Biryukov et al. [21] point out a DoS attack to disconnect Tor exit nodes from the Bitcoin network. It remains to be seen if Bitcoin's P2P layer will evolve to better utilize Tor or if a dedicated anonymity network will be developed. Finally, current SPV implementations provide little anonymity due to the difficulty of privately retrieving the list of transactions that the client is interested in [49].

B. Proposals for improving anonymity

There are three main classes of anonymity proposals. A comparison is provided in Table VII-A with respect to five security and deployment properties (with ● meaning a scheme has a property and ◐ indicating it partially does).

Peer-to-peer. In P2P mixing protocols, a set of Bitcoin holders jointly create a series of transactions which (privately) permute ownership of their coins, making each participant anonymous within this set. This process may be repeated between different users to grow the anonymity set.

Proposal	*Class*	*Security*			*Deploy.*	
		Internal Unlinkability	Theft Resistance	DoS Resistance	Bitcoin-compatible	# Transactions
CoinJoin [79]	P2P		●		●	1
Shuffle Net [35]	P2P		●		●	1
Fair Exchange [13]	P2P		●		●	4
CoinShuffle [104]	P2P	●	●	◐	●	1
Mixcoin [26]	distr.	◐	◐	●	●	2
Blindcoin [118]	distr.	●	◐	●	●	4
CryptoNote [119]	altcoin	●	●	●		0
Zerocoin [81]	altcoin	●	●	●		2
Zerocash [16]	altcoin	●	●	●		0

Table I
COMPARATIVE EVALUATION OF ANONYMITY TECHNIQUES.

A straightforward mechanism for achieving this is CoinJoin [79], where a set of users form a single standard Bitcoin transaction with one input from each user and a fresh output address controlled by each user such that no external party examining the transaction knows which input corresponds to which output (providing external unlinkability). Any user can refuse to sign the transaction if their desired output address is not included, preventing theft but making it vulnerable to DoS by any individual. In vanilla CoinJoin, users announce their output address to the other users (not providing internal unlinkability). This can be addressed through toggling a new Tor circuit or other ad hoc methods. For robust internal unlinkability, CoinShuffle [104] is an overlay protocol for forming CoinJoin transactions through a cryptographic mixing protocol. It also partially (◐) prevents DoS by identifying which parties abort.

Two earlier proposals offer similar properties to CoinJoin, one based on a shuffling network [35] and one based on fair exchange [13]. However, both are limited to two-party mixing making internal unlinkability impossible. To address the difficulty of finding partners for two party mixing protocols, Xim [22] is a decentralized protocol for finding mixing partners using three stages of fees paid to miners to discourage denial of service attacks.

Distributed mix network. In Mixcoin [26], users send standard-sized transactions to a third-party mix and receive back the same amount from coins submitted by other users of the same mix. This provides anonymity toward external entities and partial internal anonymity (◐), as the mix will know the linking between users and outputs but other users will not. Other users also cannot disrupt the protocol. While mixes may steal Bitcoins at any time, cheating mixes can be identified using signed *warrants* (providing partial ◐ theft resistance). While Mixcoin's warranties and other features have not been deployed, this is the closest proposal to third-party mixes which are most commonly used in practice [88].

Blindcoin [118] extends Mixcoin using blinded tokens

[19] One exception is CoinJoin in Section VII-B, which explicitly uses multi-input transactions to *increase* anonymity.

similar to Chaum's original e-cash proposal [28]. This prevents an honest-but-curious mix from learning the mapping between inputs and outputs and upgrade to full internal unlinkability, at a cost of two additional transactions to publish and redeem the blinded tokens.

Altcoins with integrated unlinkability. Zerocoin [81] is a proposed altcoin with integrated unlinkability, using a Bitcoin-like base currency and an anonymous shadow currency called zerocoins. Users transact solely in the base currency, but can cycle the base currency into and out of zerocoins with anonymity relative to the set of all zerocoins (a much larger anonymity set than the other techniques above). This provides strong unlinkability with no theft or DoS concerns and without relying on any entities other than miners. However, it is not compatible with Bitcoin and must be implemented as an altcoin (or hard fork). PinnochioCoin [37] is a similar proposal using a different cryptographic construction.

Zerocash [16] is an even stronger proposal for an anonymous altcoin. Zerocash transactions are a special type of zero-knowledge proofs called SNARKs [17] which reveal no information at all about the amount or recipients (except a possible public transaction fee), enabling a completely untraceable ledger in which no information is revealed publicly. SNARKs are a new cryptographic primitive without any real-world deployment to date and require an initial generation of secret parameters by a trusted party; however, recent work has shown this initial setup can be distributed among a set of mutually untrusted parties [18].

CryptoNote [119] is a cryptographic mixing protocol using ring signatures which has already been used as the basis for several privacy-focused altcoins. Users can send one coin by providing a *one-time ring signature* on a set of k (possibly unspent) coins of their choice, which function as an anonymity set. The one-time property ensures that double-spend attempts can be linked to each other, resulting in an invalid transaction. Transaction sizes are linear in k, the size of the anonymity set of a single transaction. This scheme has better performance but weaker anonymity compared to Zerocoin or Zerocash.

VIII. Extending Bitcoin's functionality

While Bitcoin can be described simply as a digital currency, the power of the scripting language with enforcement by miners makes many other types of interaction possible between two or more mutually distrusting parties that would otherwise require a trusted intermediary. We use the term *disintermediation* to refer to the general process of designing transactions that remove the need for a trusted intermediary.

A. Disintermediation with Bitcoin today

The extent to which Bitcoin is an extensible platform is often overstated. The scripting language remains highly constrained. However, many protocols have been designed for disintermediation which can be realized with Bitcoin's current transaction semantics. We identify three general disintermediation strategies:

Atomicity. In many cases, a desired security property can be enforced directly using functionality provided by the blockchain and the fact that transactions can be *atomic*, being invalid until multiple parties sign. CoinJoin [79] is a simple example, with no participant's coins swapped until all parties sign. Another example is Hearn's "serial micropayments" protocol [55], which makes efficient use of an out-of-band channel to allow one party to authorize a nearly-continuous slow release of funds (e.g., a fraction of a penny per second) in exchange for some metered service such as Internet access. The payer can end the protocol at any time by ceasing to sign any more transactions, at which point only one transaction needs posting to the blockchain. Another clever protocol is Nolan's atomic cross-chain exchange protocol, which allows users to swap currency between two altcoins with two linked transactions and atomic security [91].

Collateral. In other cases, when a desired security property cannot be enforced directly, Bitcoin can provide an acceptable remedy by posting a deposit or bond which is only refunded in the case of correct behavior. This approach is exemplified by the multi-player lottery protocol of Andrychowicz et al. [5]. Each of N parties places a ฿k bet, and one party (chosen at random) walks away with ฿kN. In order to guarantee that a cheating player doesn't spoil the game by learning the outcome first and selectively aborting the protocol, every player must deposit ฿kN^2. If any participant aborts the protocol they forfeit their deposit, which is used to compensate the others to the maximum amount they could have won. This approach is not limited to lotteries, but in fact can provide a notion of fairness for arbitrary multiparty computations [19].

Auditability. Even if Bitcoin is not used to apply an immediate remedy against a dishonest party, it can still play a crucial role in providing evidence that incriminates the dishonest party. One example is green addresses [58] in which a payment processor with a well-known public key pledges never to sign an invalid or conflicting transaction. A user who receives a transaction from a green address may accept it (i.e., make an irrevocable decision) before waiting for it to be included in blocks. If at some point the transaction is preempted by a conflicting transaction published in the blockchain, the user obtains easily checkable evidence that the server cheated. A similar technique is used in Mixcoin [26], in which semi-trusted parties provide signed *warranties* which, along with the blockchain, will provide irrefutable evidence of misbehavior.

B. Bitcoin as a data store

An alternate approach to extending Bitcoin is to use it only as global append-only log to which anybody can write.

Secure timestamping. Because the blockchain is (modulo forks) append-only, it can be used immediately as a secure timestamping service [31], which is useful in a variety of security protocols. Arbitrary data can be written into the blockchain through several mechanisms—the community prefers the use of a small provably unspendable script which includes data in an unused variable.[20] Multiple services collect data from users and publish a Merkle root to the blockchain, allowing anybody to timestamp arbitrary data.

Digital tokens: Colored Coins. Because data can be written into individual transactions, it is possible to mark certain transactions with a "color." This enables a protocol called Colored Coins [103] which defines a set of rules (not enforced by miners) to transfer color from input transactions to output transactions. Coins may initially be colored by including a special signature from any authority trusted to issue color for some application. This allows the creation of arbitrary tokens which can be traded for each other or for ordinary uncolored bitcoins. Colored coins have been proposed for many applications, such as trading stocks or property rights. Because Bitcoin miners do not enforce the rules of the colored coins protocol, validating a transaction's color requires scanning the blockchain for all ancestor transactions (precluding SPV proofs).

Colored coins use the history-tracking functionality of the blockchain as a feature. In general, it has been observed that every transaction output has a unique history of ancestors which may be meaningful to different users, meaning that in the long run bitcoins are not guaranteed to be fungible [89].

Overlay protocols: Mastercoin. A more flexible approach is to use Bitcoin's consensus mechanism but define completely different transaction syntax (with arbitrary validity rules) to be written as arbitrary data on the blockchain. Note that this design removes correctness property that Bitcoin's consensus mechanism normally provides, as Bitcoin miners will not know about the new transaction types. Thus invalid overlay transactions may be published and need to be ignored by participants in the overlay system. SPV proofs are also impossible as users must validate the entire overlay transaction history. Two prominent such systems are Counterparty [39] and Mastercoin [123], which define a large number of additional transaction types for trading digital assets and contracts.

C. Extending Bitcoin's transaction semantics.

The Bitcoin scripting language is deliberately restrictive; in fact, the original source contains the makings of a much more versatile language, but most of the opcodes are marked as unusable. In the full online version of our paper [25] we explain and evaluate a variety of proposals such as Namecoin [76] or Ethereum [125] to extend Bitcoin's functionality to provide a more versatile platform for disintermediation.

IX. Concluding remarks

Our extensive analysis of Bitcoin based on both the academic and (vast, fragmented) online literature shows a renaissance of new ideas in designing a practical cryptocurrency, a longstanding challenge for the computer security community. Innovation has not been limited to new cryptocurrency protocol designs but has touched many areas of computer security, distributed systems, hardware design and economics. This is a rich and deep space and it should not be overlooked simply because many ideas did not originate from traditional computer science research institutes.

Yet while our knowledge has grown considerably, our understanding is often still lacking. A simple fact demonstrates this: given the chance to design a currency system from scratch, it is unclear what significant deviations from Bitcoin would be desirable or what effects they would have in practice. This is not to say Bitcoin is flawless, as its many design quirks show. There are also several areas, such as anonymity, in which clearly superior designs have been proposed. Yet for basic stability and efficiency, it remains unclear if it is possible to design an alternate decentralized consensus system which can improve on Bitcoin. The literature does not even provide adequate tools to assess under which economic and social assumptions Bitcoin itself will remain stable. Similarly, for designing disintermediated protocols with new features, it is not clear how to expand Bitcoin's functionality without upsetting its observed stability.

On the whole, we simply don't have a scientific model with sufficient predictive power to answer questions about how Bitcoin or related systems might fare with different parameters or in different circumstances. Despite occasional misgivings about academic computer science research in the Bitcoin community, however, we advocate an important role for research in place of simply "letting the market decide." It is difficult today to assess the extent to which Bitcoin's success compared to altcoins is due to its specific design choices as opposed to its first-mover advantage.

Bitcoin is a rare case where practice seems to be ahead of theory. We consider that a tremendous opportunity for the research community to tackle the many open questions about Bitcoin which we have laid out.

Acknowledgments

The authors would like to thank the following colleagues for feedback on drafts of this paper: Sergio Demian Lerner, Luke Valenta, Albert Szmigielski, Gus Gutoski, Ben Laurie, Ittay Eyal as well as the anonymous reviewers at IEEE Security & Privacy and members of the Bitcoin community. Joseph Bonneau is supported by a Secure Usability Fellowship. Jeremy Clark is supported by an NSERC Discovery Grant. Joshua A. Kroll is supported by an NSF Graduate Research Fellowship under Grant No. DGE-1148900. Arvind Narayanan is supported by NSF Grant CNS-1421689.

[20] Proof-of-burn is also a solution, but this is not provably unspendable and so it is discouraged by miners.

References

[1] G. Andresen. March 2013 Chain Fork Post-Mortem. BIP 50.

[2] G. Andresen. Pay to Script Hash. BIP 16, 1 2012.

[3] G. Andresen. Blocksize Economics. bitcoinfoundation.org, October 2014.

[4] E. Androulaki, G. O. Karame, M. Roeschlin, T. Scherer, and S. Capkun. Evaluating User Privacy in Bitcoin. In *Financial Cryptography*, 2013.

[5] M. Andrychowicz, S. Dziembowski, D. Malinowski, and L. Mazurek. Secure Multiparty Computations on Bitcoin. In *IEEE Symposium on Security and Privacy*, 2014.

[6] M. Andrychowicz, S. Dziembowski, D. Malinowski, and L. Mazurek. On the Malleability of Bitcoin Transactions. In *Workshop on Bitcoin Research*, 2015.

[7] J. Aspnes, C. Jackson, and A. Krishnamurthy. Exposing computationally-challenged Byzantine impostors. Technical report, Yale, 2005.

[8] M. Babaioff, S. Dobzinski, S. Oren, and A. Zohar. On Bitcoin and Red Balloons. In *SIGecom Exchanges*, pages 56–73. ACM, 2012.

[9] A. Back, M. Corallo, L. Dashjr, M. Friedenbach, G. Maxwell, A. Miller, A. Poelstra, J. Timón, and P. Wuille. Enabling blockchain innovations with pegged sidechains, 2014.

[10] A. Back et al. Hashcash-a denial of service counter-measure, 2002.

[11] L. Bahack. Theoretical Bitcoin Attacks with less than Half of the Computational Power (draft). Technical Report abs/1312.7013, CoRR, 2013.

[12] T. Bamert, C. Decker, L. Elsen, R. Wattenhofer, and S. Welten. Have a snack, pay with Bitcoins. In *IEEE P2P*, 2013.

[13] S. Barber, X. Boyen, E. Shi, , and E. Uzun. Bitter to Better—How to Make Bitcoin a Better Currency. In *Financial Cryptography*, 2012.

[14] J. Becker, D. Breuker, T. Heide, J. Holler, H. Rauer, and R. Böhme. Can We Afford Integrity by Proof-of-Work? Scenarios Inspired by the Bitcoin Currency. In *WEIS*, 2012.

[15] M. Belenkiy. E-Cash. In *Handbook of Financial Cryptography and Security*. CRC, 2011.

[16] E. Ben-Sasson, A. Chiesa, C. Garman, M. Green, I. Miers, E. Tromer, and M. Virza. Zerocash: Decentralized anonymous payments from Bitcoin. In *IEEE Symposium on Security and Privacy*, 2014.

[17] E. Ben-Sasson, A. Chiesa, D. Genkin, E. Tromer, and M. Virza. SNARKs for C: Verifying program executions succinctly and in zero knowledge. In *CRYPTO*, 2013.

[18] E. Ben-Sasson, A. Chiesa, M. Green, E. Tromer, and M. Virza. Secure Sampling of Public Parameters for Succinct Zero Knowledge Proofs. In *IEEE Symposium on Security and Privacy*, 2015.

[19] I. Bentov and R. Kumaresan. How to Use Bitcoin to Design Fair Protocols. In *CRYPTO*, 2014.

[20] I. Bentov, C. Lee, A. Mizrahi, and M. Rosenfeld. Proof of Activity: Extending Bitcoin's Proof of Work via Proof of Stake. Cryptology ePrint Archive, Report 2014/452, 2014.

[21] A. Biryukov and I. Pustogarov. Bitcoin over Tor isn't a good idea. In *IEEE Symposium on Security and Privacy*, 2015.

[22] G. Bissias, A. P. Ozisik, B. N. Levine, and M. Liberatore. Sybil-Resistant Mixing for Bitcoin. In *WPES'14: Workshop on Privacy in the Electronic Society*, 2014.

[23] J. Bonneau. Why ASICs may be good for Bitcoin. https://freedom-to-tinker.com/blog/jbonneau/why-asics-may-be-good-for-bitcoin/, December 2014.

[24] J. Bonneau and P. Eckersley. Agile Tokens, 2014.

[25] J. Bonneau, A. Miller, J. Clark, A. Narayanan, J. A. Kroll, and E. W. Felten. Research Perspectives and Challenges for Bitcoin and Cryptocurrencies (Extended Version). Cryptology ePrint Archive, Report 2015/261, 2015.

[26] J. Bonneau, A. Narayanan, A. Miller, J. Clark, J. A. Kroll, and E. W. Felten. Mixcoin: Anonymity for Bitcoin with accountable mixes. In *Financial Cryptography*, 2014.

[27] J. Camenisch, S. Hohenberger, and A. Lysyanskaya. Compact e-cash. In *EUROCRYPT*, 2005.

[28] D. Chaum. Blind signatures for untraceable payments. In *CRYPTO*, 1982.

[29] D. Chaum, A. Fiat, and M. Naor. Untraceable electronic cash. In *CRYPTO*, 1990.

[30] N. Christin. Traveling the Silk Road: A measurement analysis of a large anonymous online marketplace. In *WWW*, 2013.

[31] J. Clark and A. Essex. CommitCoin: carbon dating commitments with Bitcoin. In *Financial Cryptography*, 2012.

[32] M. Corallo. High-speed Bitcoin Relay Network, November 2013.

[33] N. T. Courtois. On the longest chain rule and programmed self-destruction of crypto currencies. *arXiv preprint arXiv:1405.0534*, 2014.

[34] N. T. Courtois, M. Grajek, and R. Naik. Optimizing sha256 in bitcoin mining. In *Cryptography and Security Systems*, 2014.

[35] O. Coutu. Decentralized Mixers in Bitcoin. *Bitcoin Conference*, 2013.

[36] W. Dai. b-money. www.weidai.com/bmoney.txt, 1998.

[37] G. Danezis, C. Fournet, M. Kohlweiss, and B. Parno. Pinocchio Coin: building Zerocoin from a succinct pairing-based proof system. In *PETShop*, 2013.

[38] C. Decker and R. Wattenhofer. Information propagation in the bitcoin network. In *IEEE P2P*, 2013.

[39] A. K. R. Dermody and O. Slama. Counterparty announcement. https://bitcointalk.org/index.php?topic=395761.0, January 2014.

[40] J. A. D. Donet, C. Pérez-Sola, and J. Herrera-Joancomartı. The Bitcoin P2P network. In *Workshop on Bitcoin Research*, Jan. 2014.

[41] dree12. List of Major Bitcoin Heists, Thefts, Hacks, Scams, and Losses. bitcointalk.org, August 2014.

[42] C. Dwork and M. Naor. Pricing via processing or combatting junk mail. In *CRYPTO*, 1992.

[43] S. Eskandari, D. Barrera, E. Stobert, and J. Clark. A first look at the usability of bitcoin key management. *Workshop on Usable Security (USEC)*, 2015.

[44] I. Eyal. The Miner's Dilemma. In *IEEE Symposium on Security and Privacy*, 2015.

[45] I. Eyal and E. G. Sirer. Majority is not enough: Bitcoin mining is vulnerable. In *Financial Cryptography*, 2014.

[46] J. Garay, A. Kiayias, and N. Leonardos. The Bitcoin Backbone Protocol: Analysis and Applications. Cryptology ePrint Archive, Report 2014/765, 2014.

[47] L. Garber. Government Officials Disrupt Two Major Cyberattack Systems. *IEEE Computer*, July 2014.

[48] A. Gervais, G. O. Karame, V. Capkun, and S. Capkun. Is bitcoin a decentralized currency? *IEEE Security & Privacy*, 12(3):54–60, 2014.

[49] A. Gervais, G. O. Karame, D. Gruber, and S. Capkun. On the Privacy Provisions of Bloom Filters in Lightweight Bitcoin clients. In *ACSAC*, 2015.

[50] S. Goldfeder, R. Gennaro, H. Kalodner, J. Bonneau, E. W. Felten, J. A. Kroll, and A. Narayanan. Securing bitcoin wallets via a new DSA/ECDSA threshold signature scheme, 2014.

[51] D. M. Goldschlag and S. G. Stubblebine. Publicly Verifiable Lotteries: Applications of Delaying Functions. In *Financial Cryptography*, 1998.

[52] R. Grinberg. Bitcoin: An Innovative Alternative Digital Currency, November 2011.

[53] M. Hearn. Dan Kaminsky's thoughts on scalability. bitcointalk.org, 2011.

[54] M. Hearn. Merge-Avoidance: a note on privacy-enhancing techniques in the Bitcoin protocol. medium.com, 2013.

[55] M. Hearn. Rapidly-adjusted (micro)payments to a pre-determined party. bitcointalk.org, 2013.

[56] J. Herrera-Joancomart. Research and Challenges on Bitcoin Anonymity. Keynote Talk: 9th International Workshop on Data Privacy Management, 2014.

[57] D. Y. Huang, H. Dharmdasani, S. Meiklejohn, V. Dave, C. Grier, D. McCoy, S. Savage, N. Weaver, A. C. Snoeren, and K. Levchenko. Botcoin: monetizing stolen cycles. In *NDSS*, 2014.

[58] jav. Instawallet introduces new approach to instant payment: Green address technique. bitcointalk.org, July 2011.

[59] B. Johnson, A. Laszka, J. Grossklags, M. Vasek, and T. Moore. Game-theoretic analysis of DDoS attacks against Bitcoin mining pools. In *Workshop on Bitcoin Research*, 2014.

[60] G. O. Karame, E. Androulaki, and S. Capkun. Double-spending fast payments in Bitcoin. In *ACM CCS*, 2012.

[61] R. Karp, C. Schindelhauer, S. Shenker, and B. Vocking. Randomized rumor spreading. In *Foundations of Computer Science*, 2000.

[62] S. King. Primecoin: Cryptocurrency with prime number proof-of-work, 2013.

[63] S. King and S. Nadal. PPCoin: Peer-to-Peer Crypto-Currency with Proof-of-Stake, August 2012.

[64] 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.

[65] P. Krugman. Bitcoin is Evil. *The New York Times*, Dec 2013.

[66] J. Kwon. TenderMint: Consensus without Mining, August 2014.

[67] S. M. Larson, C. D. Snow, M. Shirts, et al. Folding@ Home and Genome@ Home: Using distributed computing to tackle previously intractable problems in computational biology. Technical report, arXiv preprint, 2002.

[68] A. Laszka, B. Johnson, and J. Grossklags. When Bitcoin Mining Pools Run Dry: A Game-Theoretic Analysis of the Long-Term Impact of Attacks Between Mining Pools. In *Workshop on Bitcoin Research*, 2015.

[69] B. Laurie. An Efficient Distributed Currency, 2011.

[70] B. Laurie. Decentralised currencies are probably impossible (but let's at least make them efficient), 2011.

[71] B. Laurie and R. Clayton. Proof-of-work proves not to work. In *WEIS*, 2004.

[72] T. B. Lee. Major glitch in Bitcoin network sparks sell-off; price temporarily falls 23%. Ars Technica, March 2013.

[73] S. D. Lerner. Even faster block-chains with the DECOR protocol. https://bitslog.wordpress.com/2014/05/02/decor/, May 2014.

[74] S. D. Lerner. The Private Automatic Miner Backbone Protocol (PAMBA), April 2014.

[75] P. Litke and J. Stewart. Cryptocurrency-stealing malware landscape. Technical report, Dell SecureWorks Counter Threat Unit, 2014.

[76] A. Loibl. Namecoin. namecoin.info, 2014.

[77] makomk. [DEAD] Coiledcoin - yet another cryptocurrency, but with OP_EVAL! bitcointalk.org.

[78] J. Matonis. The Bitcoin Mining Arms Race: GHash.io and the 51% Issue, July 2014.

[79] G. Maxwell. CoinJoin: Bitcoin privacy for the real world. bitcointalk.org, August 2013.

[80] S. Meiklejohn, M. Pomarole, G. Jordan, K. Levchenko, D. McCoy, G. M. Voelker, and S. Savage. A Fistful of Bitcoins: Characterizing Payments Among Men with No Names. In *IMC*, 2013.

[81] I. Miers, C. Garman, M. Green, and A. D. Rubin. Zerocoin: Anonymous Distributed E-Cash from Bitcoin. In *IEEE Symposium on Security and Privacy*, 2013.

[82] A. Miller. Feather-forks: enforcing a blacklist with sub-50% hash power. bitcointalk.org, October 2013.

[83] A. Miller, A. Juels, E. Shi, B. Parno, and J. Katz. Permacoin: Repurposing Bitcoin Work for Data Preservation. In *IEEE Symposium on Security and Privacy*, May 2014.

[84] A. Miller and J. J. LaViola Jr. Anonymous Byzantine Consensus from Moderately-Hard Puzzles: A Model for Bitcoin, 2014.

[85] A. Miller, E. Shi, A. Kosba, and J. Katz. Nonoutsourceable Scratch-Off Puzzles to Discourage Bitcoin Mining Coalitions (preprint), 2014.

[86] T. Moore and N. Christin. Beware the Middleman: Empirical Analysis of Bitcoin-Exchange Risk. In *Financial Cryptography*, 2013.

[87] M. Möser and R. Böhme. Trends, Tips, Tolls: A Longitudinal Study of Bitcoin Transaction Fees. In *Workshop on Bitcoin Research*, 2015.

[88] M. Möser, R. Böhme, and D. Breuker. An inquiry into money laundering tools in the Bitcoin ecosystem. In *IEEE eCrime Researchers Summit (eCRS)*, 2013.

[89] M. Möser, R. Böhme, and D. Breuker. Towards Risk Scoring of Bitcoin Transactions. In *Workshop on Bitcoin Research*, 2014.

[90] S. Nakamoto. Bitcoin: A Peer-to-Peer Electronic Cash System. http://bitcoin.org/bitcoin.pdf, 2008.

[91] T. Nolan. Alt chains and atomic transfers. bitcointalk.org, May 2013.

[92] T. Okamoto and K. Ohta. Universal electronic cash. In *CRYPTO*, 1992.

[93] M. Okun. Agreement among unacquainted Byzantine generals. In *Distributed Computing*. 2005.

[94] M. Palatinus. Stratum mining protocol - asic ready. https://mining.bitcoin.cz/stratum-mining, September 2012.

[95] R. Parhonyi. Micropayment Systems. In *Handbook of Financial Cryptography and Security*. CRC, 2011.

[96] C. Percival and S. Josefsson. The scrypt Password-Based Key Derivation Function, 2012.

[97] A. Poelstra. Distributed Consensus from Proof of Stake is Impossible, May 2014.

[98] QuantumMechanic. Proof of stake instead of proof of work. bitcointalk.org, July 2011.

[99] R. L. Rivest. Peppercoin micropayments. In *Financial Cryptography*, 2004.

[100] R. L. Rivest and A. Shamir. PayWord and MicroMint: Two simple micropayment schemes. In *Security Protocols Workshop*, 1997.

[101] S. Ro. A Bloomberg TV Host Gifted Bitcoin On Air And It Immediately Got Stolen. *Business Insider*, December 2013.

[102] M. Rosenfeld. Analysis of Bitcoin Pooled Mining Reward Systems. Technical report, CoRR, 2011.

[103] M. Rosenfeld. Overview of Colored Coins, 2012.

[104] T. Ruffing, P. Moreno-Sanchez, and A. Kate. CoinShuffle: Practical decentralized coin mixing for Bitcoin. In *ESORICS*, 2014.

[105] T. Sander and A. Ta-Shma. Auditable, anonymous electronic cash. In *CRYPTO*, 1999.

[106] T. Sander, A. Ta-Shma, and M. Yung. Blind, auditable membership proofs. In *Financial Cryptography*, 2001.

[107] B. Schoenmakers. Security aspects of the EcashTM payment system. *State of the Art in Applied Cryptography*, 1998.

[108] D. Schwartz, N. Youngs, and A. Britto. The Ripple Protocol Consensus Algorithm. https://ripple.com/consensus-whitepaper/, September 2014.

[109] SEC vs Shavers. https://www.sec.gov/litigation/complaints/2013/comp-pr2013-132.pdf, 2013.

[110] M. Sirbu and J. D. Tygar. NetBill: An internet commerce system optimized for network-delivered services. *IEEE Personal Communications*, 2(4):34–39, 1995.

[111] E. G. Sirer and I. Eyal. How to Disincentivize Large Bitcoin Mining Pools, June 2014.

[112] Y. Sompolinsky and A. Zohar. Accelerating bitcoin's transaction processing fast money grows on trees. *Not Chains*, 2013.

[113] I. Stewart. Proof of burn. bitcoin.it, December 2012.

[114] N. Szabo. Formalizing and securing relationships on public networks. *First Monday*, 2(9), 1997.

[115] M. B. Taylor. Bitcoin and the age of bespoke Silicon. In *CASES*, 2013.

[116] J. Timón and M. Friedenbach. Freicoin - easy-to-use demurrage currency. http://freico.in/.

[117] J. Tromp. Cuckoo Cycle: a memory-hard proof-of-work system. In *Workshop on Bitcoin Research*, 2015.

[118] L. Valenta and B. Rowan. Blindcoin: Blinded, Accountable Mixes for Bitcoin. In *Workshop on Bitcoin Research*, 2015.

[119] N. van Saberhagen. Cryptonote v 2.0. *https://cryptonote.org/whitepaper.pdf*, 2013.

[120] M. Vasek, M. Thornton, and T. Moore. Empirical analysis of denial-of-service attacks in the Bitcoin ecosystem. In *Workshop on Bitcoin Research*, 2014.

[121] V. Vishnumurthy, S. Chandrakumar, and E. G. Sirer. Karma: A secure economic framework for peer-to-peer resource sharing. In *Workshop on Economics of Peer-to-Peer Systems*, 2003.

[122] F. Voight. p2pool: Decentralized, dos-resistant, hop-proof pool. https://bitcointalk.org/index.php?topic=18313.0, June 2011.

[123] J. R. Willett. MasterCoin Complete Specification, v1.1, 2013.

[124] wizkid057. Re: [6600Th] Eligius: 0% Fee BTC, 105% PPS NMC, No registration, CPPSRB (New Thread). bitcointalk.org, June 2014.

[125] G. Wood. Ethereum: A secure decentralized transaction ledger. http://gavwood.com/paper.pdf, 2014.