Do You Really Need a Blockchain?

An Operational Risk Assessment

June 2022

Prepared by:

Evan Sultanik
evan.sultanik@trailofbits.com

Mike Myers
mike.myers@trailofbits.com

About Trail of Bits

Founded in 2012 and headquartered in New York, Trail of Bits provides technical security assessment and advisory services to some of the world's most targeted organizations. We combine high-end security research with a real-world attacker mentality to reduce risk and fortify code. With 80+ employees around the globe, we've helped secure critical software elements that support billions of end users, including Kubernetes and the Linux kernel.

We maintain an exhaustive list of publications at https://github.com/trailofbits/publications, with links to papers, presentations, public audit reports, and podcast appearances.

In recent years, Trail of Bits consultants have showcased cutting-edge research through presentations at CanSecWest, HCSS, Devcon, Empire Hacking, GrrCon, LangSec, NorthSec, the O'Reilly Security Conference, PyCon, REcon, Security BSides, and SummerCon.

We specialize in software testing and code review projects, supporting client organizations in the technology, defense, and finance industries, as well as government entities. Notable clients include HashiCorp, Google, Microsoft, Western Digital, and Zoom.

Trail of Bits also operates a center of excellence with regard to blockchain security. Notable projects include audits of Algorand, Bitcoin SV, Chainlink, Compound, Ethereum 2.0, MakerDAO, Matic, Uniswap, Web3, and Zcash.

To keep up to date with our latest news and announcements, please follow @trailofbits on Twitter and explore our public repositories at https://github.com/trailofbits. To engage us directly, visit our "Contact" page at https://www.trailofbits.com/contact, or email us at info@trailofbits.com.

Trail of Bits, Inc.
228 Park Ave S #80688
New York, NY 10003
https://www.trailofbits.com
info@trailofbits.com

Preface

This report is intended to provide decision-makers with the context necessary to assess the operational risks of using blockchains and distributed ledger technology. It includes a brief introduction to blockchain technology and covers the current state of the technology and its use cases and deficiencies. This report also surveys the common pitfalls, failures, and vulnerabilities that Trail of Bits has observed as a leader in the field of blockchain assessment, security tooling, and formal verification. This report is written such that neither a technical background nor prior knowledge of blockchain is a prerequisite; however, technical resources are cited for domain experts or deeper review, as necessary.

Table of Contents

What Is a Blockchain?

Blockchains, or more generally, *distributed ledger technology* (DLT), provide a means to maintain a tamper-resistant, auditable ledger without any centralized control. New data—or *transactions*—can be appended to the ledger, but under normal operation, previous data can never be altered or removed, making it immutable. Anyone can inspect the ledger and observe every piece of data ever recorded. Blockchains are, therefore, completely open. Nothing is private on a blockchain, and it is impossible to permanently and securely store sensitive information on a blockchain.

Cryptocurrency

The advent of a distributed, tamper-resistant ledger is what enabled the development of the original use case for blockchains: cryptocurrency. Alice offers to pay Bob ₿20 for a product he is selling. By inspecting the public ledger, Bob can see that, in fact, Alice has only ₿10 to her name and can reject her offer. The blockchain's public record, combined with its distributed nature and resistance to manipulation, prevents so-called "double-spending" (i.e., spending the same currency twice) and enables cryptocurrency exchanges to function.

Anonymity

The provenance of every unit of cryptocurrency is reconstructable from the transaction ledger. For example, for any user's *address* (which can be thought of as an "account"), anyone can easily enumerate every transaction sent from or received by that address. The primary source of potential anonymity in a blockchain is the way that humans connect to it: It might be known that a specific address was used to make illegal transactions, but there is no easy way to associate that address with the specific human or even the IP address that controls it. However, if that illicit address ever transacts with an address with a known human association (e.g., a federally regulated cryptocurrency exchange that can convert cryptocurrency to a fiat currency), it is typically possible to de-anonymize the owner of the illicit address and to reveal the owner's entire transaction history. For example, the IRS recently identified a Russian-Swedish administrator accused of running a Bitcoin anonymizing system[1] by tracking the provenance of the very fiat transactions that his system was supposed to obfuscate[2].

[1] For a fee, these systems, often called *tumblers*, will mix identifiable cryptocurrency with legitimate cryptocurrency in an attempt to obscure their provenance.
[2] Andy Greenberg, "Feds Arrest an Alleged $336M Bitcoin-Laundering Kingpin," *WIRED*, April 27, 2021.

Immutability

A blockchain can store any data, not just financial transactions. For example, in a tribute to the late Len Sassaman[3], Travis Goodspeed and Dan Kaminsky added a picture of Ben Bernanke (who, at the time, was Chair of the United States Federal Reserve) to the Bitcoin blockchain[4]. As we noted, under normal operation, blockchains like Bitcoin are immutable; the correctness of every new piece of data depends on the correctness of every piece of data previously added. Therefore, it is amusing to think that the cryptographic validity of the current state of the Bitcoin blockchain is conditioned on the existence of a picture of one of its biggest naysayers. But this immutability also adds operational risk for using or running a blockchain: What if unwanted data is added? It will be practically impossible to remove. Copyrighted and even illicit data such as child pornography have been discovered on public blockchains[5]. This exposes blockchain service providers to significant liability, which we will discuss in detail later in this report.

Data storage on blockchains is very expensive. This is by design, since data will remain on the blockchain forever and will be replicated across all blockchain nodes. Therefore, the digital art associated with non-fungible tokens (NFTs) is almost never stored on the blockchain itself, as it would be prohibitively expensive to do so. Instead, the art is hosted off-chain on a web server, and the on-chain NFT is simply a receipt linking to the image's URL. However, since the image itself is not stored on the blockchain, nothing prevents the website hosting the image from changing its content or deleting it.

Smart Contracts and DeFi

Because a blockchain can store any data, it can also store code. Ethereum is by far the most popular blockchain that is specifically designed to store code: Users can deploy *smart contracts*, or specialized computer programs stored on the blockchain. Other users can interact with these programs, and the results of these interactions are stored on the blockchain. For example, Alice can deploy a smart contract that holds some cryptocurrency in escrow until Bob instructs the contract to transfer the money to Eve. This automated and permissioned logic enabled the recent boom in *decentralized finance* (DeFi) systems, in which smart contracts automatically perform the functions of more traditional financial instruments. However, like any code, smart contracts can and often do have bugs, and the irreversible nature of blockchain transactions only heightens the stakes. Hundreds of millions of dollars have been permanently lost because of smart contract bugs. In some cases, these losses were reverted through software changes. For example, when $60 million were stolen from an organization called the DAO in 2017, the Ethereum community

[3] A strong case can be made that Len Sassaman was in fact Satoshi Nakamoto, the pseudonymous creator of Bitcoin: evanhatch.eth, Len Sassaman and Satoshi: A Cypherpunk History," *Medium*, February 21, 2021.
[4] "Block 138725," *Bitcoin Explorer, Blockchain.com*, last updated March 7, 2022.
[5] "Child Abuse Images Hidden in Crypto-Currency Blockchain," *BBC*, February 6, 2019.

voted to modify the blockchain software to undo the theft. This is one example of how blockchains are not completely immutable.

Types of Blockchains

There are dozens of different blockchain implementations, many of which are only subtly distinct, if at all. This section covers their primary characteristics.

Blockchain Permissions

Permissionless

Permissionless or *public* blockchains are open and decentralized. Almost all cryptocurrencies, including Bitcoin and Ethereum, fall into this category. They do not have a central entity to manage membership or use, so any user can connect and interact with them. These properties make it possible for a malicious actor to unduly influence the state of a permissionless blockchain. However, this is typically prevented through a combination of cryptography that renders attacks computationally intractable and transaction fees that simultaneously make attacks expensive and incentivize honest participation.

Private

A *private* blockchain typically has a network of distributed nodes that collaborate to agree on the current state of the system, just like a public blockchain. But unlike public blockchains, which at least strive for decentralization, private blockchains are effectively centralized; a central authority or coalition controls the access rights of all users. For example, only a subset of users might be allowed to add new data to the blockchain. Examples include Hyperledger and R3 Corda. However, the primary difference is that arbitrary public users cannot independently verify private blockchains. Moreover, the controller of a private blockchain can revoke any user's access to the blockchain at any time. Therefore, a private blockchain is more like a traditional managed database, with the addition of a cryptographically verifiable audit log. However, the audit log is not necessarily public; in fact, it cannot be if the contents of the private blockchain are sensitive. Moreover, whoever controls the private blockchain can revert the database state. Private blockchains are not as resistant to tampering as public blockchains.

Permissioned

A *permissioned* blockchain is like a public blockchain, but it allows for fine-grained control over which users can perform which operations. Like a private blockchain, a permissioned blockchain is typically centrally managed; however, like a public blockchain, it allows arbitrary public users to inspect its validity. This auditability comes at the expense of an increased attack surface, since permissioned blockchains must typically be run on the public internet rather than a private or air-gapped network.

Consensus

Blockchains typically have two types of network participants: *nodes* and *miners*. Nodes accept new transactions to be considered for inclusion on the blockchain. Miners compete to process those pending transactions, producing new “blocks” that can be appended to the blockchain. Miners receive a reward for their effort. Nodes then distributedly agree on whether a new block should be added.

The order in which transactions are processed is paramount. For example, say Alice currently owns ₿1, and she submits two transactions at the same time: one transaction transferring her ₿1 to Bob and another transaction transferring her ₿1 to Eve. Whichever transaction is processed first will succeed, and whichever is processed second will fail because of Alice’s lack of funds. It is up to the blockchain nodes and miners to distributedly reach consensus on which transaction is recorded. Different blockchains use different mechanisms to reach consensus.

Submitting a transaction to a node does not guarantee that it will be recorded on the blockchain. In fact, nodes have every right to simply ignore transactions. Blockchain users who are being disenfranchised by a recalcitrant node can simply choose to submit their transactions to another node or to run their own nodes. Likewise, miners have the relative freedom to choose which transactions they include in a block and in which order. A cabal in control of a significant portion of the consensus nodes, or *hashrate* (i.e., mining power), could deny service to selected addresses, refusing to process any transactions associated with them. This threat is typically mitigated through the financial incentives rewarded to well-behaved miners. But adversaries for whom financial gain is not an object can, and have been able to, manipulate cryptocurrencies. Moreover, if a single entity controls a significant portion of the hashrate (typically more than half), then the immutability guarantees of the blockchain are violated. This risk is discussed in detail later in this report.

Different blockchains employ different protocols and algorithms for the nodes to reach consensus faster. There are also various approaches across blockchains to deter Sybil attacks that would provide malicious users with undue influence over the hashrate. The most common approach to deter Sybil attacks is *proof of work*, also called the *Nakamoto Consensus* after the pseudonymous creator of Bitcoin, who employed it. Proof of work associates the expenditure of “work” (typically, computation) with value. Miners work on mathematical problems that are computationally hard to solve but easy to verify; one result of such work is newly minted cryptocurrency. This provides the artificial scarcity (and, therefore, value) of the cryptocurrency, since the value of the resulting rewards is proportional to the computational effort expended. However, the significant amount of computing resources to achieve proof of work largely contributes to both the electrical and temporal overhead of most blockchains. There are several alternative Sybil deterrence schemes, including *proof of stake*, *proof of personhood*, and *proof of space*, each with different properties and tradeoffs.

On-Chain Computation

Some blockchains, like Ethereum, are designed to provide computational primitives for running smart contracts, whose code and associated states are stored on-chain. Private blockchains can have trusted code and a trusted execution environment and need to store only the resulting states on-chain. There are also several efforts to implement smart contract capabilities on top of blockchains that were not originally designed for them (e.g., RSK and Bitcoin Computer). Blockchains that support smart contracts have additional mining overhead due to the added computation necessary to execute and verify code. Moreover, the attack surface is significantly increased by virtue of the additional programming and the compiler infrastructure necessary for translating human-readable programs into the low-level instructions on which the blockchain operates.

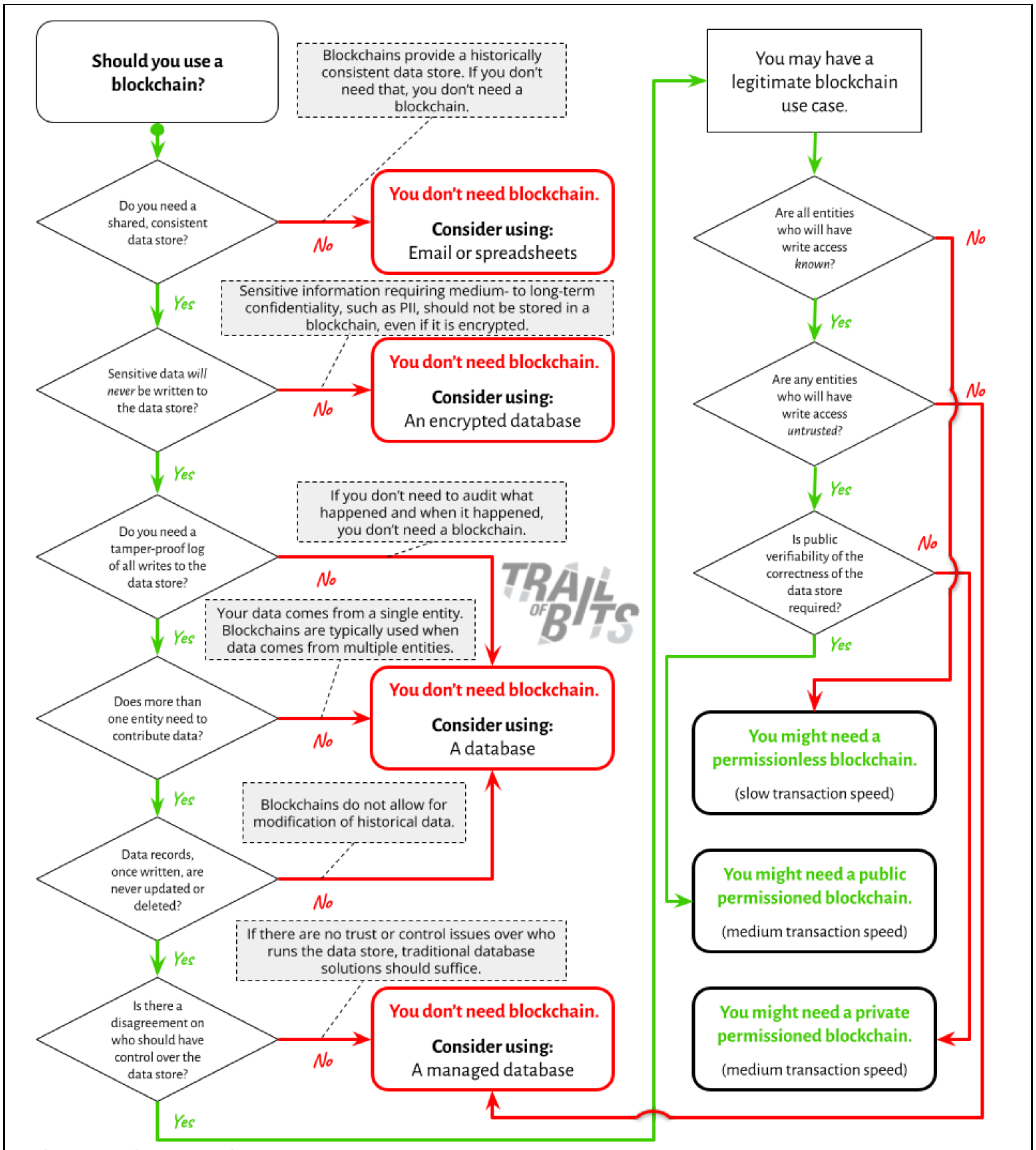
Do You Need a Blockchain? If so, What Kind?

Due to the diversity of blockchain types and features, it can be challenging to decide whether a blockchain is an appropriate technical solution for a given problem and, if so, which type of blockchain to use. Blockchains have significantly different constraints, security properties, and resource requirements than traditional data storage alternatives.

	Traditional Database	**Public Permissionless Blockchain**	**Public Permissioned Blockchain**	**Private Permissioned Blockchain**
Software Maturity	~50 years	~13 years	~6 years	~13 years
Transaction Speed	Fast (>100M/second)	Very Slow (~10/second)	Slow (~1000/second)	Slow (~1000/second)
Computational Resources (CPU/RAM/ Storage)	Low	High	Medium	Medium
Can Securely Store Sensitive Information	Yes	No	No	Yes†
Immutable	No	Yes	Yes‡	Yes‡
Public Attack Surface	No	Yes	Yes	No

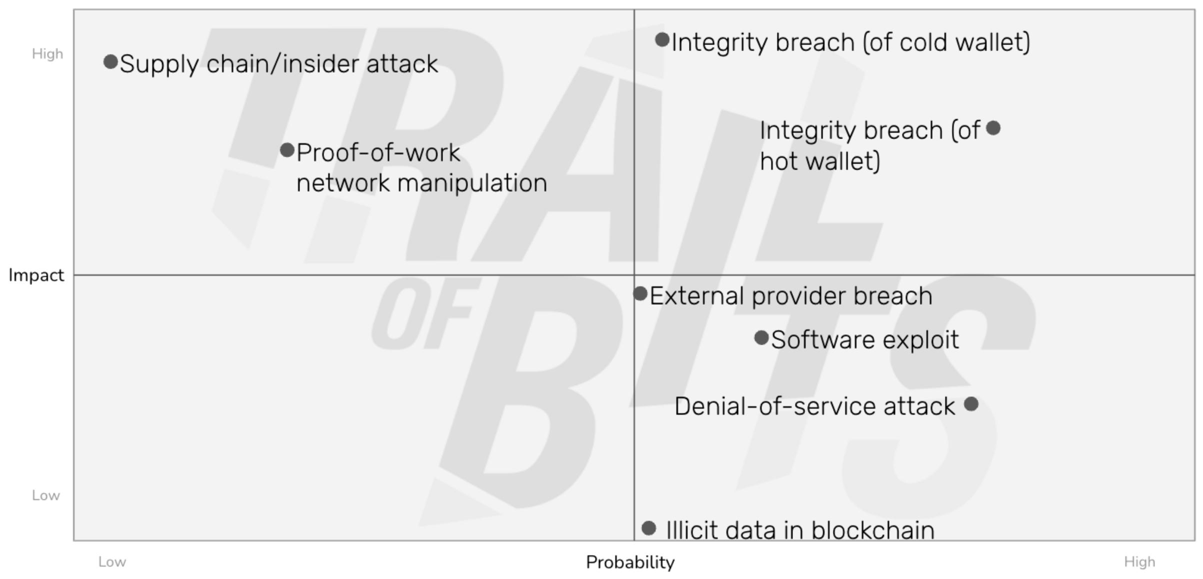
† As long as all users are trusted
‡ Administrators are able to delete, but not necessarily modify, historical data.

Before selecting a blockchain as a storage or consensus mechanism, it is vital to weigh its resource requirements against its potential benefits. The following decision tree can help you determine which type of blockchain to use for your problem, if any.



Should you use a blockchain?
Blockchains provide a historically consistent data store. If you don't need that, you don't need a blockchain.
Do you need a shared, consistent data store?
No
You don't need blockchain. Consider using: Email or spreadsheets
Yes
Sensitive information requiring medium- to long-term confidentiality, such as PII, should not be stored in a blockchain, even if it is encrypted.
Sensitive data will never be written to the data store?
No
You don't need blockchain. Consider using: An encrypted database
Yes
If you don't need to audit what happened and when it happened, you don't need a blockchain.
Do you need a tamper-proof log of all writes to the data store?
No
Yes
Your data comes from a single entity. Blockchains are typically used when data comes from multiple entities.
Does more than one entity need to contribute data?
No
You don't need blockchain. Consider using: A database
Yes
Blockchains do not allow for modification of historical data.
Data records, once written, are never updated or deleted?
No
Yes
If there are no trust or control issues over who runs the data store, traditional database solutions should suffice.
Is there a disagreement on who should have control over the data store?
No
You don't need blockchain. Consider using: A managed database
Yes
You may have a legitimate blockchain use case.
Are all entities who will have write access known?
No
Yes
Are any entities who will have write access untrusted?
No
Yes
Is public verifiability of the correctness of the data store required?
No
Yes
You might need a permissionless blockchain. (slow transaction speed)
You might need a public permissioned blockchain. (medium transaction speed)
You might need a private permissioned blockchain. (medium transaction speed)
TRAIL OF BITS

Source: Trail of Bits; Adapted from:
Dylan Yaga (NIST), Peter Mell (NIST), Nik Roby (G2), and Karen Scarfone (Scarfone Cybersecurity), "NISTIR 8282 Blockchain Technology Overview", *U.S. Department of Commerce*, *National Institute of Standards and Technology*, October, 2018.
Karl Wüst (ETH Zurich) and Arthur Gervais (Imperial College London), "Do You Need a Blockchain?", *2018 Crypto Valley Conference on Blockchain Technology*, Accessed from Cryptology ePrint Archive, Report 2017/375.



Once you have determined that a blockchain or DLT is the appropriate technology for your use case, you need to understand the operational risks it will incur. The remainder of this report outlines these risks and offers suggestions for mitigating them.

Operational Risk in Blockchains and DLTs

The previous section outlined the criteria for deciding whether blockchains and DLTs are an appropriate technical solution for a problem. In the remainder of this report, we outline the operational risks of utilizing this technology and the associated best practices and mitigations.

Key Insights

- **Proof-of-work technology and its risks are relatively well understood** compared to newer consensus mechanisms like proof of stake, proof of authority, and proof of burn.
- **The foremost risk is "the storage problem."** It is not the storage of cryptocurrency, but rather the storage of the cryptographic private keys that control the ownership of an address (account). Disclosure of, or even momentary loss of control over, the keys can result in the complete and immediate loss of that address's funds.
    - Specialized key-storing hardware, either a hardware security module (HSM) or hardware wallet, is an effective security control when designed and used properly, but current hardware solutions are less than perfect.
    - Compartmentalization of funds and multisignature wallets are also effective security controls and complement the use of HSMs.
- **Security breaches or outages at third-party API providers represent a secondary risk**, which is best mitigated by contingency planning.
- **Centralization of mining power is a systemic risk whose impact is less clear** but important to monitor; it represents a potential for blockchain manipulation and, therefore, currency manipulation.
- **Most blockchain software, though open source, has not been formally assessed by reputable application-security teams.** Commission regular security reviews to assess blockchain software for traditional vulnerabilities. Use network segmentation to prevent blockchain software from being exposed to potential exploitable vulnerabilities.

The following are the risk areas for users of blockchain technology, ordered by priority:

1. **Managing Wallets and Keys**
    - Key Management
    - Wallet Creation/Seed Generation

- Avoiding Mistakes with Multisignature Wallets
    - The Secure Use of HSMs
2. **Relying on External API Providers**
    - Mitigating Denial-of-Service Attacks
    - Identifying API Design Failures
    - Avoiding Weak 2FA on Accounts
3. **Operating a Blockchain Service Organization**
    - High-Integrity Software Development Practices
    - High-Assurance System Configurations
    - Managing Personnel Access
4. **Blockchains Are Distributed Networks**
    - Planning for Transaction Congestion Effects
    - Detecting Price Manipulation
    - Evaluating Illicit Data Stored on the Blockchain
    - Managing Blockchain Network Forks
5. **Potential Vulnerabilities in Blockchain Client Software**
    - Blockchain Client Software
    - Mining Pool Software
    - Denial of Service via the Consensus Protocol

Examples of non-technical risk areas that are not fully covered in this report include the following:

- Market dynamics (e.g., changing investor risk appetite)
- Market competition/saturation
- Fractional reserve exchanges and the risk of liquidity crises
- Governmental regulation
- Legal liability or legal recourse
- Evaluations of the legality of financial instruments or transactions
- Adherence to know your customer (KYC) standards and to anti-money laundering (AML) and securities laws

Managing Wallets and Keys

Cryptographic keys and key operations (signing, verifying, encrypting, and decrypting) play an essential role in any blockchain. It is critical that blockchains have a mechanism to securely create, store, and use keys within cryptographic operations. This is the core element of trust in a blockchain.

In certain industries and use cases, compliance with industry-standard certification and validation (e.g., FIPS 140-2) may also be required.

Even in a best-case scenario, in which you have rendered the theft of private keys impossible, any integrity breach of the overall transaction-signing system could allow an attacker to co-opt it to sign unauthorized transactions and to steal holdings. To further mitigate this risk, create a cryptocurrency wallet that requires every transaction to be signed by more than one system (multisigning), with each signing system requiring compartmentalized access. One implementation of this scheme is the use of multisignature wallets.

Recommended Security Controls

Hardware for storing cryptographic keys	Use tamper-resistant cryptographic hardware device peripherals designed to store and perform operations with private keys without ever disclosing the keys to a host computer. HSMs and cryptocurrency hardware wallets are two types of this hardware solution.
Key compromise protocol (KCP)	Create a contingency plan in preparation for a key disclosure incident, even a suspected incident. A proper KCP describes a series of steps to transition to a new secure private key without losing access to or control of protected data, and with minimal impact to the organization's service availability.
Cold storage	Restrict access to the majority of assets in a wallet on an offline (air-gapped, physically access-controlled) system. Transactions can be signed there and manually taken to an online system for publication to the blockchain.
Multisignature wallets	Use wallets in which the private keys are split across separate systems and 2-of-3 consensus is required to spend from the wallet. Though 2-of-3 is the most common, other configurations (3-of-5, etc.) are also possible.

Key Management

Any large organization using a blockchain should use HSMs to generate, store, and use its blockchain-related cryptographic keys. These are the same devices trusted to protect code-signing keys and to control ownership of the top-level domains of the internet and are common throughout the banking industry. HSMs protect private keys from disclosure and provide secure crypto-processing capabilities (e.g., on-board key generation and asymmetric cryptography operations). HSM security is evaluated against the standards in the NIST FIPS 140 publication series. Requirements for HSMs include secure random number generation, controlled execution timing, and a tamper-resistant, tamper-evident design (at Level 2 and higher).

Given the cost and difficulty for a small organization to use business-grade HSMs, the Ledger Nano models S and X are the most popular options today for protecting cryptocurrency private keys. A hardware cryptocurrency wallet like the Ledger Nano is basically an HSM in portable device form. That said, the hardware wallet's firmware security has never been sufficiently reviewed. About a dozen locally exploitable vulnerabilities in the Ledger Nano S were reported during the first year of its release by individual researchers[6], and our experience leads us to assume that more will be found.

Under no circumstance should private keys be generated, stored, or used in a "software wallet." The risk of wallet key theft on a general-purpose computing device is unacceptably high for only a small increase in convenience. Even so-called "paper" wallets—in which the private key is stored on a physical piece of paper—have been compromised through backdoors in their software generators, leading to the theft of over $6 million in cryptocurrency in February of 2021 alone.

Wallet Creation/Seed Generation

Cryptocurrency wallets are initialized to an address on the blockchain using a pseudo-random number-generation routine that must be "seeded" with true/secure random values. A seed is a series of numbers, but it can also be represented as a human-friendly phrase of dictionary words via standards like BIP39. These "recovery words" can be used to initialize another wallet to the same address on the blockchain. The use of seed values chosen by an attacker or those that lack randomness may allow an attacker to withdraw funds from a wallet and deposit them into his own.

In other words, if a user accepts a preconfigured hardware wallet or a preselected set of recovery words, she is putting her assets into a wallet controlled by an attacker. Anyone with access to the recovery phrase (in this case, the attacker) has complete control over the wallet, the ability to watch it for activity, and the ability to extract all coins from it.

[6] Such as Riscure, Saleem Rashid, and an anonymous researcher

It is essential that users obtain their solutions for generating their wallets from a trusted source (e.g., directly from the vendor) and initialize their wallets themselves by generating new recovery words. Paper backups of the BIP39 recovery passphrase words must be handwritten, never entered into or stored on another device.

Unlike hardware wallets, each HSM is initialized with a set of smart cards called the administrator card set (ACS). After initialization, these cards should never be needed again. Users who want to avoid guarding the ACS indefinitely can destroy them.[7] The ACS cards can be used to recover the private key material on the HSMs, so they function like the BIP39 recovery passphrase described above. Refer to the "The Secure Use of HSMs" section for more recommendations.

Do not store BIP39 recovery words alongside a hardware wallet. Seal the recovery words sheet in a tamper-evident bag, and keep them in a separate secure location. The recovery words should be needed only in a recovery scenario. They are not needed for signing transactions (spending) from the hardware wallet.

Let us assume that at least one keyholder stores his BIP39 recovery words alongside his HSM or hardware wallet, even if they are both in a bank's safe deposit box. This "puts all his eggs in one basket" in the event of a natural disaster and defeats the protection provided by the device's PIN and anti-hardware-tampering defenses.

In a theft scenario, anyone who gains access to the device and its list also gains access to the private key, because the key can be reconstituted from the recovery words. Although a 2-of-3 multisignature wallet can survive the loss of one private key, the loss of one key is still a risky step toward the loss of all of the funds.

Avoid counterfeit or tampered-with hardware. Hardware tampering that predetermines the seed secret is theoretically possible. HSM or hardware wallet users must put ultimate trust in the supply chain of that device. That is why it is important to ensure that the devices were purchased from a trusted source, preferably directly from the vendor, new and unopened. While resetting a maliciously preconfigured device ought to be sufficient, there is still a risk that a counterfeit backdoored hardware device cannot be reset safely.

Any suitable HSM or hardware wallet should have the ability to verify its provenance, authenticity, or hardware integrity after purchase, using cryptographic attestation performed inside its internal secure element. Look for hardware that supports this feature, and consider exercising it per the vendor's instructions.

[7] Matthew Green, "Is Apple's Cloud Key Vault a Crypto Backdoor?" *A Few Thoughts on Cryptographic Engineering* (blog), August 13, 2016.

Avoiding Mistakes with Multisignature Wallets

The concept of N-of-M multisignature Bitcoin transactions is defined in the BIP45 standard for hierarchical deterministic multiparty multisignature (HDPM) wallets. Most commonly, three individuals hold the wallet's three private keys, and 2-of-3 consensus is required to spend from the wallet. This is an excellent mitigation against theft. The loss of one key should not result in the loss of the entire wallet.

Follow a multisignature wallet use policy. Consider the following example policy for how an organization uses its 2-of-3 multisignature wallet.

Transactions that spend from the wallet are coordinated by Peter, after which the decision is made together with other key holders. Peter does not hold any of the keys. The transaction is signed by one of the keyholders, transmitted to another of the signatories for cosigning, and then broadcast to the Bitcoin network. Also, the signatory who is requested to cosign the transaction may verify that the transaction is valid with a voice call to Peter. Whether the signatory should call Peter is not defined by the policy. Rather, this is an optional step to be taken at the signatory's own discretion.

Hypothetical example of a policy for an organization's use of its multisignature wallet

A policy such as the above could be a good starting point to ensure the safe and proper use of a multisignature wallet.

Establish a KCP. In any scenario in which one of the private keys in a 2-of-3 multisignature wallet is lost, the funds in the wallet are still available but are now at an increased risk of loss. A multisignature wallet cannot revoke a key in the manner of a typical public key infrastructure (PKI) system or publish a replacement key using a revocation certificate, as Pretty Good Privacy (PGP) can do.

Blockchain addresses are valid forever, as are 2-of-3 wallets once they are formed. Keyholding members of a 2-of-3 wallet cannot be removed or replaced with other members without changing the wallet address. Instead, members must *migrate* wallets by forming a new 2-of-3 wallet with a new third master public key (MPK) and then transferring the assets from the old wallet to the new wallet.

The following describes example steps for a 2-of-3 wallet recovery procedure (for a small organization that uses Ledger Nano S hardware wallets):

1. Keep the two remaining Ledger Nano S devices in their current state. Changing them is not necessary. In fact, it is preferable to retain control of the original 2-of-3 wallet, in case someone accidentally transfers funds to that address in the future. In most proof-of-work blockchains, there is no way to prevent an address on the blockchain from receiving cryptocurrency.

2. Acquire a new Ledger Nano S and initialize it to create a new third MPK.

3. Use software on the host system to form a new 2-of-3 Bitcoin wallet and a new address, uniting the two surviving MPKs from the original wallet plus the MPK that was just created with the new Ledger Nano S.

4. Using small amounts of cryptocurrency, test that this new address can both receive and send transactions. It is essential to test the address's ability to send. If cryptocurrency fails to send, then it is permanently lost. For the sending test, use the new Ledger Nano S to create one of the two transaction signatures needed.

5. Using the two surviving Ledger Nano S devices from the original multisignature wallet, send the remaining funds from the original wallet to the new wallet address.

The Secure Use of HSMs

Successful software-only attacks against HSMs are rare but have occasionally been discovered, such as weak key-derivation schemes discovered in Gemalto HSMs, disclosed in 2015.[8] Even then, a successful attack had to compromise the security of the host system first. A side-channel analysis attack requires prolonged physical access during operation. That risk is mitigated by the physical access control to the facility holding the HSM.

Follow these recommendations to securely use HSMs:

- Physically control access to, or consider destroying, the ACS after programming the HSMs. If storing the ACS, the cards that make up the set should be stored separately, with a consensus of individuals required to access them.

- Before destroying the ACS cards, determine whether they are needed to update the HSM firmware in the event that another key-recovery vulnerability like CVE-2015-5464 is published during the expected operational life of the HSM.

- Assume that the HSMs are capable of being configured in a "persistent" mode, in which the operator card set (OCS) cards can be removed during operation. Physically control access to the OCS and require that the cards be reinserted if the system must be rebooted. Keeping the OCS physically secured will prevent an attacker capable of stealing the HSM from being able to boot it after power loss.

- Follow all available guidance from the HSM vendor during configuration and initialization.

[8] "CVE-2015-5464," *CVE List, MITRE Corporation*, last updated March 8, 2022.

- Complete a manual security audit of the software, following guidance provided by CryptoSense's PKCS#11 application programming interface (API). Consider using CryptoSense to check the software you develop for the HSMs, if applicable.

Relying on External API Providers

As with many internet-based business operations, a blockchain service organization will probably depend (to some degree) on a digital service from an external provider. These third-party providers implement APIs to automate the use of their services. The API design security, correct usage, and general availability of the third-party service all contribute to the risk of relying on an external provider.

Recommended Security Controls

Contingency planning	Plan for inevitable incidents at third-party partners and document the steps to restore service or ensure the continuity of service. Document the points of contact on both ends of the relationship with your partner organization.
Control of wallet private keys	Use multisignature wallets; do not use custodial wallets (in which a third party holds and manages your private keys). It should never be necessary to exchange private keys in order to join a multisignature wallet, only public keys.
Operation of a "full node" on the blockchain	Do not rely on a third party for a view of the blockchain that you will use for any critical decision-making processes, such as transaction verification.
Deployment of secure two-factor authentication (2FA) solutions	Access to your third-party partner service APIs, or at least the ability to make administrative changes, should require 2FA.
Review of API design security	Review the design of the third-party API with a focus on how it implements access controls, how it prevents message spoofing, and how it handles credential-reset functionality.

Mitigating Denial-of-Service Attacks

Recognize how your organization's dependence on any single third-party service can be exploited by an attacker who can perform a denial-of-service attack against that service. When establishing the contingency plan, either identify an alternate provider of the necessary services or outline a plan to bring the service in-house.

Identifying API Design Failures

Guidance on secure API design is beyond the scope of this document. Best practice dictates that a thorough evaluation of a third party's API should focus on access controls, specifically the concept of separation of privilege. Seek services with fine-grained or role-based access controls over their APIs. Ideally, the access controls will separate the credentials needed for standard actions from the credentials needed for administrative actions. Also, evaluate the mechanisms that the APIs provide for ensuring and maintaining control of the account, like the ability to require 2FA.

Avoiding Weak 2FA on Accounts

Single-factor authentication typically involves passwords, which are notoriously difficult to manage and protect. Require that third-party services support strong 2FA or multifactor authentication (MFA) through hardware Universal 2nd Factor (U2F) keys or time-based one-time password (TOTP) authenticators, never through cell phone short message service (SMS) messages. SMS-based 2FA is susceptible to SIM hijacking attacks, which are still rampant. Blockchain-related account logins remain a common target for theft.

Operating a Blockchain Service Organization

A blockchain service organization is commonly trusted with the blockchain storage problem, and thus is always a high-value target for hackers. It must function as a high-assurance operation, developing software that prioritizes fine-grained access controls, system-integrity monitoring, and code-integrity assurances.

Recommended Security Controls

The *two-man rule* or *four-eyes principle*	Design access controls to require two-person approval in order to implement changes, alter system configurations, or perform sensitive administrative functions.
Dependency monitoring and upstream alerts	Identify and adopt an automated (i.e., sustainable, practical) way to monitor all of the upstream code dependencies that expose you to software supply-chain risk.
Commit signing (Git) and required code reviews	Require approval from a separate reviewer for developer commits to source control. Ensure stronger authentication of developer write access to the code repository by requiring digital signatures on commits and tags.
Monitoring to ensure executable integrity	Consider adopting an application allowlisting solution with file integrity monitoring. Signed-code enforcement achieves a similar result, if working on a platform that supports it well.
Immutable system configurations	Endeavor to deploy, update, and redeploy systems as immutable, meaning that all operational systems are in one of a finite set of “known good [configuration] states.” Immutable system configurations also discourage practices like routine access to operational systems by privileged account holders.
Separation of duties and role-based access controls	Reduce each person’s access privileges to only what he or she needs to perform his or her role. Separate roles so that the same person does not have privileged access over multiple high-value assets or controls.

High-Integrity Software Development Practices

In this context, *high-integrity software development practices* refers to those practices whose goal is to mitigate (but not completely eliminate) the risk of a malicious actor in the development and build process.

A team attempting to achieve high-integrity development should use cryptographic features to increase trust in the code’s authenticity (e.g., Git workflows with signed tags and commits[9]) and code review requirements to ensure that an independent reviewer examines each proposed code change. Going further, require 2FA protection on accounts that can access the master version control repository. Enumerate any code dependencies with a solution such as Sonatype OSS Index, a free service used by developers to automatically identify open-source dependencies and to determine whether they contain any known (publicly disclosed) vulnerabilities.

High-Assurance System Configurations

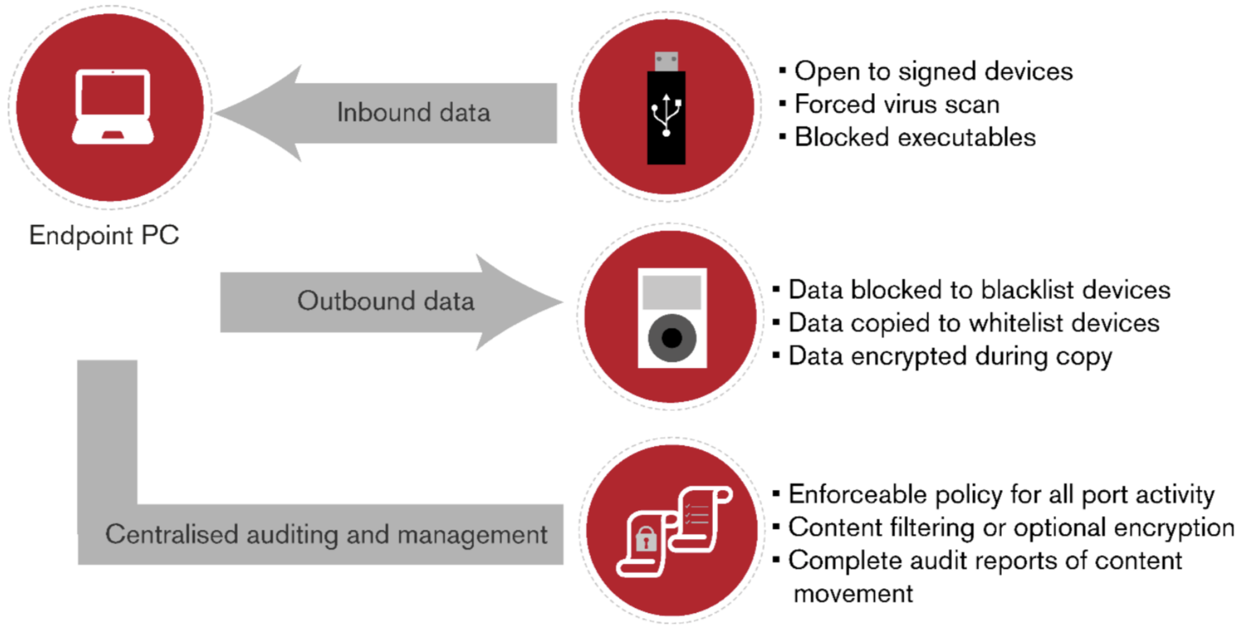
A blockchain service organization must use *high-assurance system configurations* on systems dedicated to the single task of operating the wallet. Each system can be locked down to a greater degree because it is not used for other day-to-day tasks or exposed to as many potential sources of compromise.

Microsoft has deployed this practice internally and named its high-assurance system the Privileged Access Workstation (PAW), a Microsoft-specific implementation of the general secure admin workstation (SAW). PAW is a dual-use system with strong compartmentalization provided by Hyper-V, enabling security features like Credential Guard, and a virtual machine (VM) that sandboxes internet activity.

- Microsoft: Protecting High-Value Assets with SAWs
- Microsoft: How Microsoft IT Builds PAWs

A cold storage wallet system is offline, by definition. To transfer funds from a cold storage tier to a warmer tier, the most common procedure is to hand-carry the half-signed transaction file on a USB flash drive. Removable media introduces its own risks, but effective security controls can mitigate them: enforce a removable device policy with centralized auditing and management.

[9] "Trusted Team Communication: Trusting Git Commits," *Useful IT Policies, The Linux Foundation IT,* August 13, 2015.


Inbound data
Endpoint PC
Outbound data
Centralised auditing and management
▪ Open to signed devices
▪ Forced virus scan
▪ Blocked executables
▪ Data blocked to blacklist devices
▪ Data copied to whitelist devices
▪ Data encrypted during copy
▪ Enforceable policy for all port activity
▪ Content filtering or optional encryption
▪ Complete audit reports of content movement

The ideal security configuration for removable media use on a controlled endpoint system (source: NCC's "Endpoint Connectivity" white paper)

Managing Personnel Access

An access control is the selective restriction of access to a place or other resource. Access controls can be physical, such as controls over access to a data center, or logical, such as firewalls or system user accounts. A blockchain service organization should administer its systems using role-based access and least-privilege principles.

Follow these recommendations to implement secure access controls:

- A user access matrix (below) should be used to track user privileges and role-based access. Ideally, this document should be checked and/or generated by an audit procedure. Establish a policy to update the matrix as soon as changes in access or personnel occur.

		Access Type and Role						
User	Team	VPN	Firewall	Backup	Build	GitHub	Hot	Cold
Lisa	Management	User			User			Admin
Bob	Compliance	User	Admin	Admin	User	Admin		
Teddy	Engineering	User		Admin	User	User		
Alice	Engineering	Admin			Admin	User	Admin	

Example user access matrix (source: Chris McNab, AlphaSOC)

- Segment the admin roles to reflect the segmentation of the blockchain service organization’s system design (i.e., hot and cold administrator access should belong to different individuals).
- Consider which roles can be performed with read-only access (such as backup, logging, or auditing) and provide only the access level required to perform the role.
- Only ever grant access to infrastructure via trusted communication channels. Identify those channels in advance.

Blockchains Are Distributed Networks

Proof-of-work blockchains are large, global networks with a distributed community of users. For them to function securely, their design requires a high degree of sustained decentralization. A majority of nodes must be presumed honest. In reality, one must monitor the network continuously for behaviors that diverge from the design ideals and plan accordingly to mitigate risk.

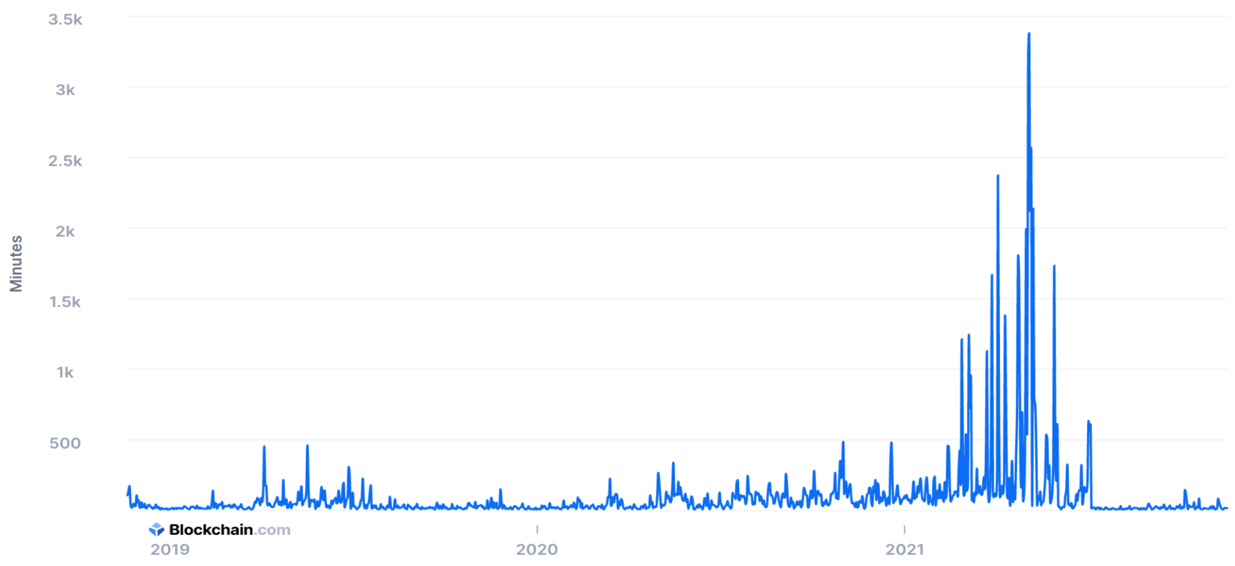
Recommended Security Controls

Contingency planning	Plan for variability in service levels due to events at the blockchain's network level. Document the steps necessary to ensure the continuity of service, or to provide partial service, during predictable network-level events like forks and periods of congestion resulting from a sudden drop in mining pool availability.
Blockchain network health monitoring	Because the blockchain is public, it is possible to monitor the block-mining process and the transactions appended to the ledger.[10] Various statistics like centralization indicate the "health" of the network. Monitoring these statistics will improve the detection and forecasting of potential problems.
Discounting the price effects of manipulations	Periodically evaluate the price effects of large actors' purchasing activities versus the organic market demand at the individual investor level, in order to assess true asset value and investment risk.

Planning for Transaction Congestion Effects

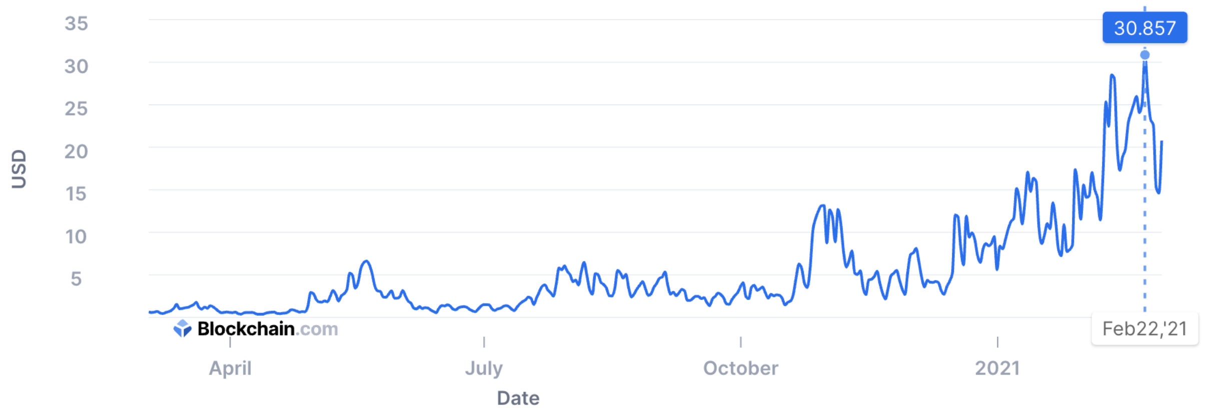
A common criticism of proof-of-work blockchains is that transaction verification time is unpredictable and worsens in times of high transaction volume. A view of transaction times for various blockchains over a selected historic period is available through blockchain.com.

[10] Bitcoin Stats is one example of a third-party API for observing blockchain statistics.



Verification time for a Bitcoin transaction over the last few years, peaking at over 50 hours

Most proof-of-work blockchains like Bitcoin are designed so that transaction fees increase as verification time increases, providing more reward incentives to verify transactions (a computation performed by the miners). Fees-per-transaction have also risen dramatically since mid-2020.



Fees in USD for a single Bitcoin transaction over the last year

Because they coincide, transaction delay and transaction fee variability effects threaten to create a liquidity bottleneck, a risk for both investors and businesses operating on proof-of-work blockchains. A liquidity bottleneck could become a crisis during a sudden market rally or crash, as it would prevent buyers and sellers from completing transactions in an acceptable time frame. Both fees and verification time can rise and fall by orders of magnitude over the span of weeks. This volatility will persist as long as market transaction volume grows faster than the growth of the overall network mining power.

During times of network congestion, the cost of denial-of-service attacks decreases, and the risk of such attacks increases. When the network is already at or near capacity, an attacker needs only to add some amount of additional transaction traffic to completely clog the blockchain network with high fees and high transaction times.

Detecting Price Manipulation

Two major concerns for blockchain networks—for Bitcoin, in particular—at the moment are (a) inter-exchange trading of unsecured "stablecoin" digital currencies to prop up the price of Bitcoin on the market, and (b) the dangerous centralization of a transaction-validating authority as represented in the concentration of Bitcoin mining power. We explain each concern in the next sections.

Unsecured Stablecoin Schemes

Starting in the cryptocurrency market capitalization peak in late 2017 and early 2018, whistleblowers like the pseudonymous "Bitfinexed", academics like Griffin and Shams of the University of Texas, and, more recently, an anonymous startup founder have been sounding the alarm about cryptocurrency exchanges like Bitfinex, Bybit, and Binance and their use of Tether. Tether is a "stablecoin," a cryptocurrency that is pegged to the US dollar (USD), providing price stability. Specifically, there are concerns that Tether, a cryptocurrency purportedly asset-backed by USD collateral reserves, is being generated from nothing (meaning that it is *not* actually backed by anything) and then used to buy Bitcoin. Suspicious transaction patterns indicate that purchases of Bitcoin with minted Tether always correlate with downward Bitcoin price shocks beneath round number thresholds, and that approximately 50% of Bitcoin's rise in price is attributable to the Tether-backed purchase events. One could argue that Tether has had an even greater upward influence on Bitcoin's price, considering the indirect psychological effects on potential investors of an apparent market rally.

As one would expect in any such scenario, a money supply that can be printed at will drives up the price of goods (in this case, Bitcoin)[11]. Neither the creators of Tether nor the suspicious exchanges disclose which bank holds their supposed USD assets that back the value of the Tether currency. In fact, both the creators and the suspicious exchanges have recently admitted that the assets backing Tether are not, in fact, held in USD and have never been audited by a third party. A reasonable assumption is that the Tether currency is <u>un</u>secured and that the assets backing Tether have <u>not</u> been audited. Some forecast a 30–80% reduction in the price of Bitcoin if Tether were to suddenly disappear.

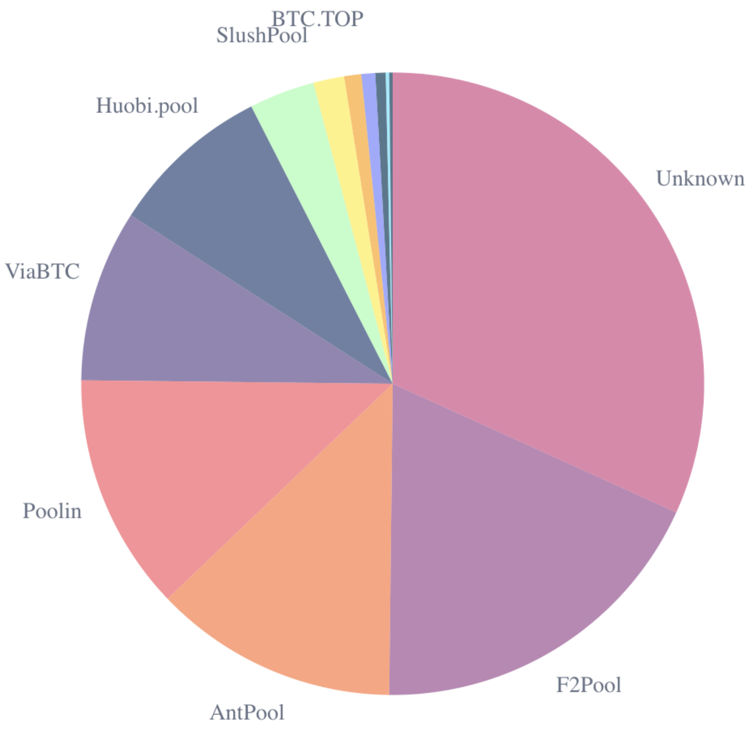
A reasonable question is, "why do some exchanges even accept Tether in exchange for Bitcoin?" Their willingness to accept Tether represents the strong underlying need for a stable cryptocurrency, however imperfect. There may be little incentive for exchanges to avoid Tether, since exchanges benefit even from a potentially fraudulent stablecoin. A

[11] For a historical analogy, recall the era of wildcat banking, specifically the Panic of 1837.

stablecoin allows them to sidestep the regulatory burdens of transacting in real USD. If exchanges do delist Tether, though, it will likely cause a Bitcoin price crash.

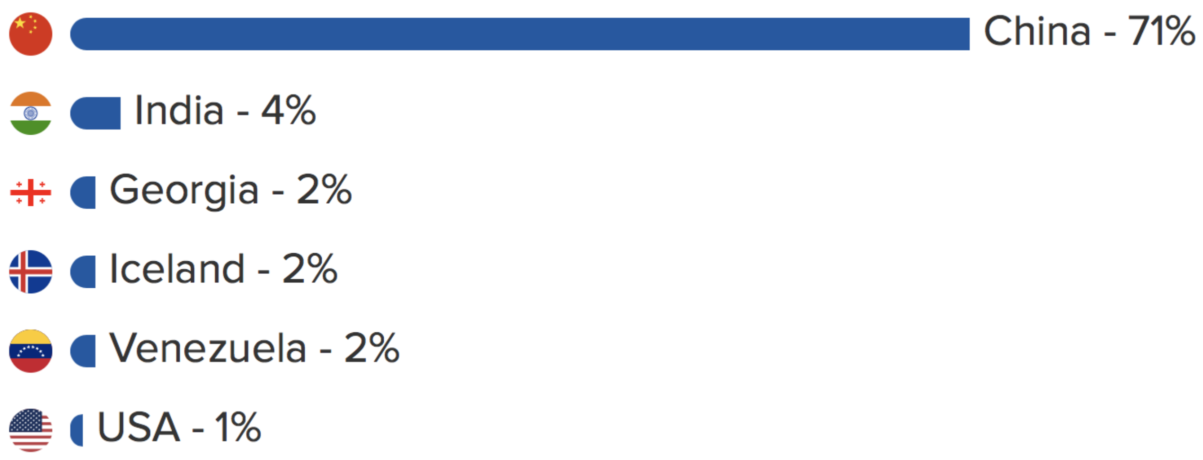
Geographic Advantages

Any lottery-based reward system for determining consensus (as is currently used by Bitcoin and Ethereum) will naturally result in the formation of pools in order to share the rewards in relation to the amount of work that each node may perform. This is a logical response to the rules of the system and a centralizing influence on the network. In fact, Bitcoin participants have centralized to such a degree that nearly the entirety of the network is controlled by fewer than 10 mining pools, almost all of which are situated in China (historically, well over 51% of the overall hashrate). Even after China's ban on cryptocurrency mining toward the end of 2021, experts estimate that China still accounts for a fifth of the world's miners.



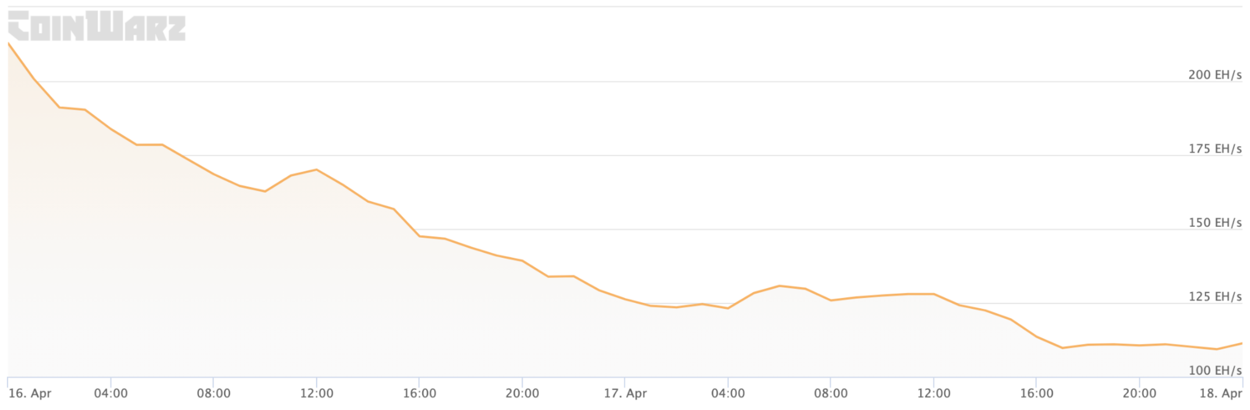
Bitcoin hashrate distribution by mining pool, circa March of 2021 (source: blockchain.com)

China has always been an attractive location for cryptocurrency mining activity because of market advantages on computing equipment and low-cost electricity. With this much of the mining power situated in one country, the risk of government regulation of Bitcoin increases. Estimating the geographic distribution of a blockchain's hashrate is challenging, since it is almost impossible to associate IP addresses with miners. Before China's ban, statistics could be collected from mining pools to identify locations. However, since the ban, there has been a significant increase in the use of Tor to obfuscate node locations.



Chinese mining pools' control of the Bitcoin hashrate as of June of 2018 (source: "Bitcoin Mining in China")[12]

For example, on April 10, 2021, a coal mine in Xinjiang, China experienced a gas explosion and flood[13]. Six days later, on the morning of April 16, authorities closed all cryptocurrency mining operations in the region for associated safety inspections[14]. As a result, the Bitcoin hashrate dropped by 50%, from a historical high of 218 EH/s down to 109 EH/s. The value of Bitcoin also dropped by about 15% that day, which some attribute to the loss of hashrate.



Bitcoin hashrate cut in half due to emergency mining farm closures in China (source: CoinWarz)

[12] Estimating the geographic location of miners is inherently difficult, since most blockchain protocols do not explicitly reveal miners on the network. Therefore, whether a node is mining or acting on behalf of a miner must be inferred. This figure is from the best academic study on the subject, which relied on surveying willing participants in mining pools.
[13] "Xinjiang Coalmine Accident Traps 21 - China State Media," *Reuters*, April 10, 2021.
[14] Thomas Heller (@thomasheller_), "Mining farms in Xinjiang closed this morning for inspections," Twitter, April 15, 2021.

A similar regulatory closure of Xinjiang mining farms in 2018 caused a 20% drop in Bitcoin's hashrate[15].

Potential for 51% Attacks

Any group that controls more than 50% of the Bitcoin network's mining power (hashrate) has control over whether to include or exclude any particular transaction. This amount of control over a network is known as a 51% attack. In many cases, the effective hashrate necessary to execute a 51% attack can be much lower than 51%; see our accompanying report for more information[16]. The ability to decide on the validity of transactions allows attackers to "double-spend" (the ability to spend Bitcoin, receive goods, and then get back the spent Bitcoin), censor transactions, and perform general denial-of-service attacks. With as little as 25% of the mining power, an attacker could perform "selfish mining" tactics that unfairly increase the likelihood of earning a profit.[17] However, this is a lesser impact, of concern mostly to other Bitcoin mining organizations.

Minor changes to the Bitcoin protocol, if the community could ever agree on them, would still not fully mitigate the risk posed by the current extent of mining power centralization. Currently, the best mitigation strategy is contingency planning combined with continued monitoring of the Bitcoin network. One form of monitoring is to observe where control is centralized; a small number of groups in one country indicates a risk of a 51% attack. Some suspect that 51% attacks are already a routine occurrence.

In the case of an active 51% attack affecting a wide swath of the network, honest participants in a proof-of-work blockchain would be incentivized to force a network fork (refer to Managing Blockchain Network Forks). But a network fork will drive down the volume (and, therefore, the price) of any blockchain. Participants are monetarily incentivized to avoid forks. So we can expect these attacks, when they happen, to be targeted against individual blockchain addresses (censorship, denial-of-service attacks), while the rest of the blockchain network stands idly by, doing nothing. It is difficult to detect a 51% attack if it is used to censor transactions, preventing them from appearing on the ledger; while it is possible to monitor for broadcast transactions, some will naturally be missed. Others will not be appended to the ledger on their first attempt for benign reasons, like insufficient fees. New research and new monitoring tools to detect censorship attacks are still needed.

[15] Nano Bank (@NanoBank), "During a tax audit in China, #miningfarms in the provinces of Xinjiang and Guizhou were disconnected from electricity," Twitter, November 15, 2018.
[16] Evan Sultanik et al., "Are Blockchains Decentralized? Unintended Centralities in Distributed Ledgers," June 21, 2022.
[17] Ittay Eyal and Emin Gün Sirer, "How a Mining Monopoly Can Attack Bitcoin," *Hacking Distributed* (blog), June 16, 2014.

When establishing a contingency plan, consider how your organization will respond if it is individually targeted by someone who has the money to wield 51% of mining resources.

Evaluating Illicit Data Stored on the Blockchain

As we discussed above, blockchains can store any data, not just financial transactions. This has been clear since the early days of Bitcoin. Multiple times, anonymous individuals have uploaded illicit data (including copyright violations and child abuse imagery) as part of blockchain transactions. Once appended to the ledger, such transactions can never be removed.

At least one company provides a service that will automatically record Tweets to the Ethereum blockchain for a small fee and provide "ownership" of the tokenized Tweet to the requestor. Tweets are copyrighted, exposing the company to the liability of copyright infringement. The company does provide a manual mechanism for copyright dispute, but there is technically no way for the copyrighted data to ever be expunged, even if the original Tweet is deleted from Twitter.

There is currently no way to prevent this from happening in blockchains because, by design, there is no way to roll back a transaction or prune any part of the append-only ledger. Illicit data is a nuisance, similar to graffiti. To the best of our knowledge, law enforcement has not yet pursued any node operators for distributing such data through a blockchain, but laws are "being considered" in multiple countries, including the United States.

The U.S. Department of the Treasury has started denylisting Bitcoin addresses suspected to be associated with sanctioned foreigners. Receiving a transaction from such an address—or even mining a block containing that transaction—could be construed as a violation of the Foreign Investment and National Security Act (FINSA), money laundering, or various other crimes, and prosecuted as a federal offense.

Managing Blockchain Network Forks

Blockchain protocols can be changed at any time that a majority of the active network decides to change it. This event is known as a fork, since it (either temporarily or permanently) results in two networks: one made of the nodes operating the new way and the other made of the nodes operating the old way. Temporary forks are a common occurrence and resolve themselves through the consensus protocol. We discuss permanent forks below.

A "majority" in a proof-of-work blockchain is not a democratic plurality of the network nodes, but rather the majority as represented by the hashrate/mining power (refer to Potential for 51% Attacks). The centralization of mining power into a handful of coordinating pools has greatly increased the risk of network forks.

Bitcoin forks have happened at least four times in the past, resulting in Bitcoin Cash (August 2017), Bitcoin Gold (October 2017), Bitcoin Private (February 2018), and Bitcoin SV (November 2018). If you held 10 Bitcoin prior to one of these forks, then you now hold 10 Bitcoin in the new fork as well and can spend these two currencies independently of one another. Some users move their funds on the newly forked blockchain to a different address to avoid confusion, an action referred to as “splitting”; this is strictly optional.

A blockchain network fork does not require immediate action. Expect additional price volatility on both the original blockchain and the newly forked blockchain. Transaction volume may increase, spiking transaction fees and reducing liquidity (refer to Planning for Transaction Congestion Effects). It may take time for blockchain software providers, exchanges, wallets, and other third parties to add support for the newly forked blockchain. During this time of change and uncertainty, there will be an increased risk of frauds and scams. Beware phony websites, backdoored wallet applications, phishing attacks, and other attempts to exploit less sophisticated users. The first few days of a network fork are also a risky time to transact on the newly forked blockchain, as it may still be unstable and more vulnerable to 51% attacks. If you operate a custodial service, your organization should decide on whether and how to support your users’ access to the newly forked blockchain.

Potential Vulnerabilities in Blockchain Client Software

Unless they rely on a third-party service, participants in a cryptocurrency must operate a software client that acts as their node on the distributed network. An effort to track the disclosed vulnerabilities in Bitcoin clients lists about 50 known issues discovered between 2010 and 2021. This is a relatively low number for such a codebase, which suggests either a high level of code quality or a lack of in-depth security reviews.

In terms of network-exposed attack surface, Bitcoin clients must support an API with various commands, served via a JSON-RPC interface, which in turn runs over unencrypted HTTP on a TCP socket. Because a cryptocurrency distributed network is "trustless" (or rather, the trust is placed in the cryptography as opposed to other participants or an intermediary), data is exchanged on the network in cleartext. This makes man-in-the-middle style attacks especially feasible, though perhaps unnecessary since the exposed interface already allows an incoming connection from any host.[18]

Recommended Security Controls

Continuous security monitoring	Equip systems running Bitcoin client software with security solutions for host-integrity monitoring. Consider adapting an application-layer, content-filtering HTTP firewall for the Bitcoin protocol to filter and detect attacks.[19]
Commissioning security assessments	Hire an experienced software security assessment team to perform a clear box (full-knowledge) code assessment of the relevant software, looking for exploitable vulnerability conditions in Bitcoin software using manual and automated analysis tools.
Separating "Bitcoin wallet control" systems from "full blockchain node" systems	Separate the system that controls access to funds in a Bitcoin wallet and can sign "spend" transactions from the system used to publish those transactions. Only the latter system needs to be exposed to the Bitcoin network.

Blockchain Client Software

Participants that only want to view balances and create transactions can use a Simplified Payment Verification (SPV) client (also known as a "thin client"), while those attempting to mine blocks must operate a "full node." Although this full node software is still called a

[18] Alex Biryukov and Ivan Pustogarov, "Bitcoin over Tor Isn't a Good Idea," *2015 IEEE Symposium on Security and Privacy*: 122–134.
[19] Marc Jansen, "Increasing Security of Nodes of a Blockchain by Simple Web Application Firewalls," *ICIW 2017: The Twelfth International Conference on Internet and Web Applications and Services* (2017): 48–51.

“client,” it implements a peer-to-peer network node with both client-like and server-like behaviors.

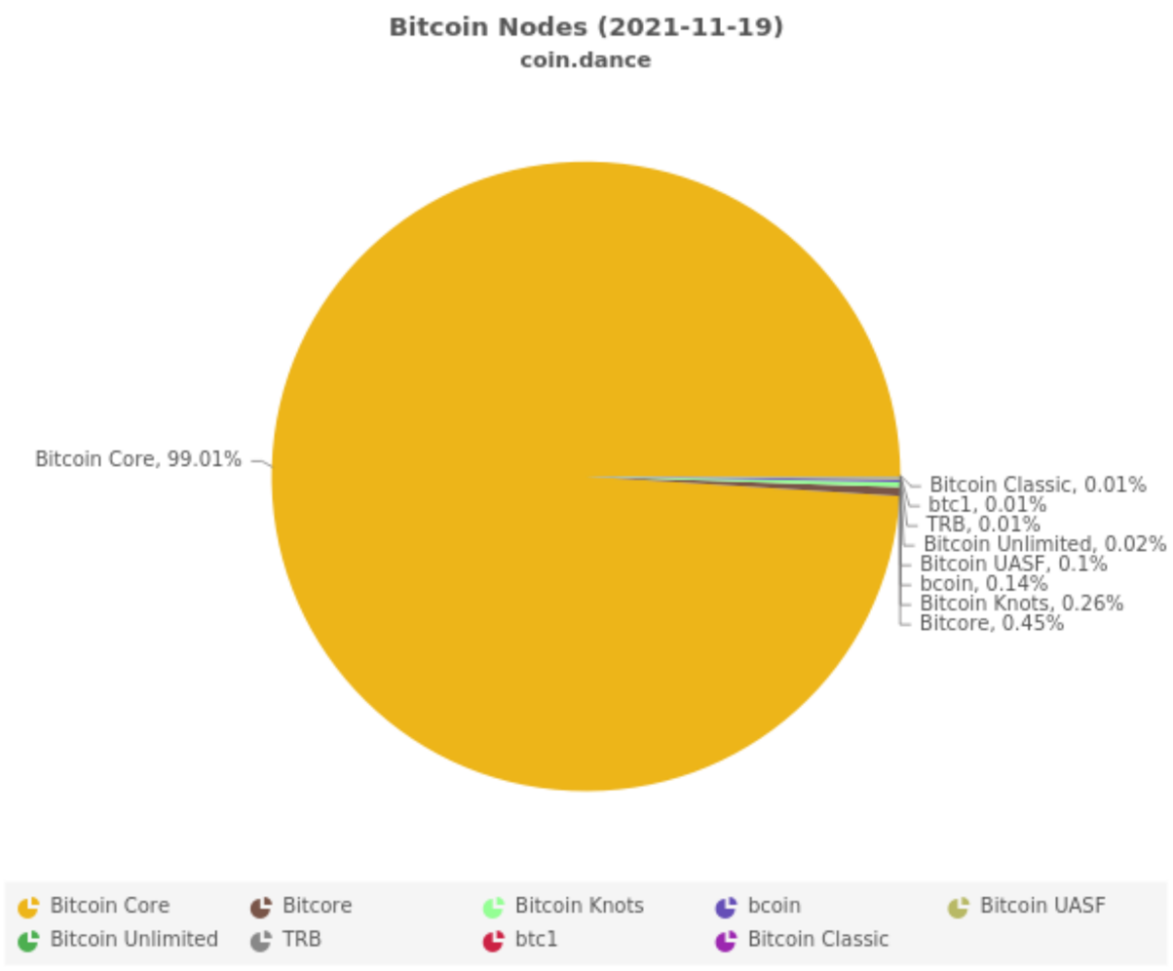
Client	**Type**	**Project Programming Language**
`bitcoind` (bundled with the GUI, Bitcoin-Qt), also known as Bitcoin Core	Full node	C and C++
`btcd` (daemon-only) and btcwallet	Full node	Go
`libbitcoin-server` (libbitcoin)	Full node	C++
`bitcoinj`	SPV	Java
`picocoin` (libccoin)	SPV	C
Electrum	SPV	Python

A selection of Bitcoin clients (all open source)

Because SPV clients do not download the entire blockchain, they *need to trust full nodes to access the blockchain on their behalf*. This is a different trust model than full nodes use and introduces the risk of attacks by dishonest full nodes relaying incorrect information. SPV nodes attempt to mitigate this risk either probabilistically by using a random selection of multiple full nodes or by limiting trust to a preset list of nodes (as in the case of Electrum).

Although SPV clients are less vulnerable to direct attacks than full nodes, all clients are part of an open distributed network and necessarily receive untrusted network input. With about half of cryptocurrency clients implemented in native code, the potential for a memory corruption bug (and resulting code execution vulnerability) is very real and quite likely. On August 25, 2018, security researcher Guido Vranken disclosed a remotely triggerable memory corruption crash in `btcd` (a Bitcoin client written in Go). Moreover, any discovered code execution vulnerability in a distributed network node is inherently wormable and could spread to a significant portion of the network before there is time to patch it.

Cryptocurrency creators encourage the existence of multiple client implementations in order to prevent common-mode failures, but the reality is that users overwhelmingly prefer a particular client implementation, as shown in the figure below.



Bitcoin node software homogenization, as of November of 2021 (source: Coin Dance)

On September 17, 2018, a bug in the popular Bitcoin Core client was discovered that could be exploited to produce unlimited Bitcoin. This bug became known as the "inflation vulnerability." It was quickly patched by developers before it could be exploited in Bitcoin, but not before attackers were able to abscond with large amounts of cryptocurrency from less popular (but still well-monetized) Bitcoin forks that used the same vulnerable client code. Today, three years later, 5% of all Bitcoin nodes are still running the unpatched, vulnerable software.

Mining Pool Software

Mining cryptocurrency produces erratic rewards unless one joins a mining pool to combine work and share profits. Additionally, because proof-of-work algorithms are more efficient to perform on special-purpose field-programmable gate arrays (FPGAs) or application-specific integrated circuits (ASICs) than on general-purpose CPUs or GPUs, miners also need software to manage a peripheral (the FPGA- or ASIC-based mining hardware). For participating in pools and managing mining peripherals, so-called mining software can be used in place of the standard cryptocurrency client. While mining software is not required to support mining in a pool, all of the software we consider here does.

Mining pools use a client-server architecture, in which the miner running the mining software is the client and the pool operator running the pool software is the server. The pool software needs to communicate with the cryptocurrency network using the appropriate core client software, mentioned earlier. For work allocation and tracking, multiple competing protocols exist, but the leading two are `getblocktemplate` (the successor to getwork) and Stratum-MP.

Software	Type	Project Programming Language
CGMiner	Mining (client)	C
BFGMiner	Mining (client) and pool (server)	C
`libblkmaker`	Mining (client library)	C
`ckpool`	Pool (server)	C
Eloipool	Pool (server)	Python
Stratum-Mining	Pool (server)	Python

A selection of Bitcoin mining and mining pool software (all open source)

Membership in a mining pool typically requires registration on a website, after which the mining software just needs a server URL, a username, and, sometimes, a password.

The protocol used by mining software to request work allocations from mining pool software is plaintext JSON-RPC, optionally over HTTP. For instance, the `getwork` and `getblocktemplate` protocols are JSON-RPC additions to the original HTTP-based Bitcoin protocol. Each request–response communication is driven by the mining software in client–server communication style. Communication is in plaintext. Authentication is an afterthought or missing entirely. Work is often submitted by username alone. With the Stratum-MP protocol, the pool server initiates a communication with the mining software, using JSON-RPC messages over a plain TCP socket (no HTTP). The mining software options written in C use the Jansson library, a JSON parser written in pure C (a language not known for its propensity to produce safe parsers). The attack surface of mining software on either end is similar to that of the cryptocurrency clients themselves.

Denial of Service via the Consensus Protocol

At least some blockchain clients have incorporated denial-of-service mitigations into their design at the network- and command-handling layers. Blockchain protocols like Bitcoin are generally designed with conservative limits on block size and on the amount of work that could be issued in a command, in recognition of the denial-of-service threat.

That said, there remains a broad potential for denial-of-service attacks against clients. Besides the obvious risk of resource-exhaustion attacks, other nodes could try to force a victim node into various failure states, to desync it from the network, or to introduce network delays in the transmission of transactions.

A denial-of-service attack alone could seem to be just an annoying disruption, but against an economic system, it may be a means to another end. Attackers wishing to exploit the distributed consensus protocol would be particularly incentivized to attempt an ongoing denial-of-service attack: Targeting and eliminating subsets of nodes from participation in consensus decisions would effectively allow them to steal control of the distributed network.

Cryptocurrencies like Bitcoin use transaction fees as a throttle to prevent spam transactions (sending money to and from oneself to fill the public ledger and waste other participants' time). However, a sufficiently motivated participant can still pay her way to total network congestion, as was seen in 2015 and 2016. The important point is that when the network is congested with legitimate traffic, the effect of an intentional denial-of-service attack is amplified. An adversary could congest the network with legitimate transactions as he launches a denial-of-service attack. He may seek to drive down the price of the currency by degrading the network in order to make a profit by "shorting" the currency. A cryptocurrency denial of service, then, could be a very lucrative attack.

We distinguish denial-of-service attacks from common network congestion, although they could appear similar in effect. Note that in this section we have focused on potential denial-of-service opportunities at the protocol-design level. Cryptocurrency *services* (e.g., exchanges and online wallets) have been attacked with traditional volumetric denial-of-service or distributed-denial-of-service tactics.

Conclusions

There are many varieties of DLT and blockchain technology, varying primarily in their permissions and consensus mechanisms. These variations can have a profound impact on the security properties of the system. This report has provided a simple decision procedure for determining whether blockchains are the correct solution to a problem and, if so, which variant. Once you have determined that a blockchain is the correct approach, it is vital to understand the operational risks it will impose on your system and organization. This report has also provided an assessment of these risks.

Acknowledgements

Goldman Sachs partnered with Trail of Bits in 2018 to create a Cryptocurrency Risk Framework. This report applies and updates some of the results from that study.

This research was conducted by Trail of Bits based upon work supported by DARPA under Contract No. HR001120C0084 (Distribution Statement A, Approved for Public Release: Distribution Unlimited). Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the United States Government or DARPA.