Compiled Analysis on Blockchain Security

Mustafa Aljadery mustafa@mustafaaljadery.com
Independent Researcher

Abstract

Participants of a distributed ledger receive confirmation that the cryptographic chain in their funds are stored is secure and invulnerable to attacks. While the infrastructure of a blockchain is specifically designed for security and decentralization, all systems are prone to attacks. In this paper, I have compiled most of the literature on large-scale attacks on the blockchain that may be able to compromise security. Attacks on the blockchain are broken into two parts: 1) Fundamental vulnerabilities in the blockchain in which the entire blockchain may be compromised.

2) Vulnerabilities in smart contract code on the usage of the blockchain. These errors are made by developers utilizing the infrastructure of the blockchain.

Blockchain Attacks are many times more common on the surface level at which developers produce code than attacks on the fundamental infrastructure of the blockchain. Blockchains are specifically designed so that democracy is fundamental in all instances, this secure design is a necessity for any financial asset. However, largest-scale exploits are still possible. I present most of the known attacks on blockchain infrastructure and attacks on surface-level execution on the blockchain.

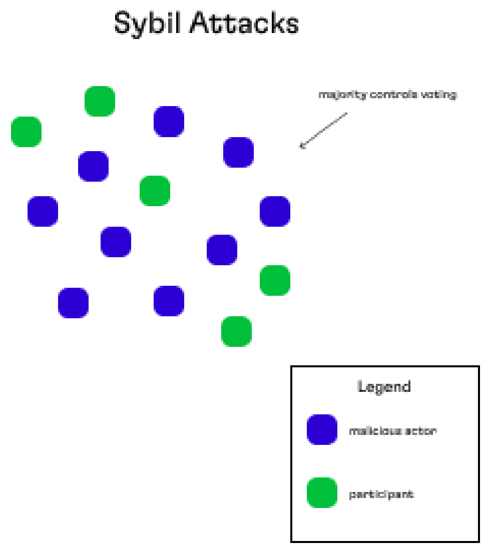
1 Fundamentals

To understand vulnerability in blockchains, we first must understand how a blockchain works. A blockchain by itself is just a database, similar to SQL. This database is shared among all of the nodes in the blockchain. Different from a SQL database, data in the blockchain is organized into compartments called blocks. Each block has a certain capacity in terms of memory. Each block, once capacity is reached, is closed and all new information will be put in the subsequent block. Since data is stored in blocks and once the blocks are full they are closed, the data is irreversible in the blockchain. This design is chosen for a distributed ledger because of its security. Blockchains guarantee trust in the availability of the data, and with the help of consensus and other mechanisms, decentralization and security.
Cryptocurrencies such as Ethereum and Bitcoin rely on a consensus mechanism to further secure the protocol from malicious actors. The consensus mechanism saw a big revolution in the Bitcoin whitepaper where the concept of Nakamoto Consensus was introduced. Fundamentally, Nakamoto consensus is a set of rules to ensure all participants of a blockchain are behaving correctly. However, the interesting feature in Nakamoto Consensus that was not present in contemporary Byzantine Fault Tolerant systems, is consensus without a leader. In Nakamoto Consensus, no leader initiates consensus, every participant has an equal say. Decentralization is the key optimization here, you would not need to rely on a central authority to initiate consensus. Nakamoto consensus does have vulnerabilities and more efficient consensus mechanisms have been proposed, these are going to be explained in this paper.
A smart contract is a contract that runs on the blockchain. It is similar to any other contract in which a settlement is decided between two parties. Important features of smart contracts include rapid execution of an agreement on a blockchain and immediate certainty of an outcome. The three major benefits of a smart contract are the accuracy a smart contract entails, singularity of a decision (a smart contract can't give you two outcomes in the same dispute, centralized parties can), and cost-efficiency of execution. Although smart contracts are secured by the fundamental infrastructure of the blockchain, smart contracts allow developers to write the contracts. Since code is written and then executed on the blockchain, numerous exploits may be present in the form of bugs that the developer did not intend. In this paper, I am going to identify common mistakes in writing smart contracts.

In addition to attacks in the fundamental infrastructure of the blockchain and attacks on poorly written smart contract code, in this paper, I am going to identify social engineering attacks that participants of the blockchain are prone to.

2 Sybil Attacks

Sybil attacks occur when malicious actors create multiple identities to bypass verification. Current proof-of-work protocols, which include Bitcoin and Ethereum, require all nodes on the protocol to provide computational power to prevent malicious actors. Without a proof mechanism, participants of the network are vulnerable to attacks. The proof-of-work protocol iterators a hashing algorithm until the round ends. The randomness of the miner that is chosen in the next round is proportional to the work done by that miner.
Most blockchain protocols are prone to the majority-of-network attack. For example, if an entity in the proof-of-work protocol has the majority of the computational power, it can bypass consensus. Consensus is just voting, and if you have the majority of the voting, you can vote on what benefits you, leaving other participants of the blockchain vulnerable. This is known as the 51 percent attack. The 51 percent attack is not only found in proof-of-work protocols but proof-of-stake protocols. In a proof-of-stake protocol, the underlying financial asset of the blockchain is reserved and if there are any malicious actors, the protocol takes their reserved (staked) assets. Proof-of-stake protocols are also vulnerable to 51 percent attacks, if an entity controls 51 percent of the underlying protocol financial assets, they take consensus into their own hands and effectively control the execution mechanism.
An adaptation of the proof-of-stake protocol with an entirely new solution to the Byzantine Fault Tolerance problem is Snow Consensus. Snow consensus was first proposed by an anonymous team called Team Rocket in 2018, which detailed a consensus mechanism that was both more secure and reached faster consensus than contemporary proof-of-stake and proof-of-work protocols. Snow consensus enables the fastest finality in blocks as of current technology. The concept of snow consensus lies in the idea of metastability. Decisions in snow consensus are not stable, the more information that is added, the more decisions can change. This is different from for example Bitcoin which has a single decision. This type of consensus is of higher security as you need to gain 81 percent of the network in order to overthrow the consensus protocol. Although this is a lot more secure for the end-users of the blockchain, it is a lot less decentralized. Viewers of the mempool need to stake orders of magnitude larger in order to view the mempool.
Prevention of a Sybil attack is impossible as this is the fundamental structure of a consensus mechanism, however, as more participants are available in the blockchain, the more decentralized it becomes and harder for a malicious attacker to gain control of the chain. In the proof-of-work example, the more miners there are in the network, the harder it is for one individual to gain control of the majority of the mining power. Such high hash power is an expensive task. In proof-of-stake, the more participants there are in a blockchain, the harder it is for one individual to own the majority of the financial assets.



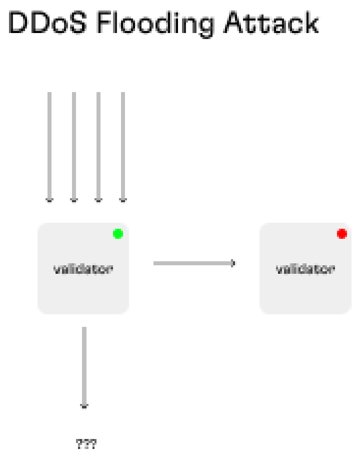
3 Denial of Service Attack

A distributed denial-of-service attack (DDoS Attack) is an attack in which a malicious attacker sends so much traffic to a server that the server goes offline. Specifically, the network resources of a machine go offline. Although the architecture of a blockchain coupled with a consensus mechanism makes it difficult for a blockchain to go offline, certain DDoS attacks can still impact the blockchain. The biggest DDoS attack that blockchains are prone to is transaction flooding. A transaction flood is when a malicious attacker sends so many transactions to the blockchain that numerous blocks get filled. Since many of the blocks are filled before they are executed, other transactions in the blockchain take many orders of magnitude more time to confirm, slowing down the experience of all other users.

An example of a large DDoS attack is the Solana network DDoS Attack. The Solana network experienced traffic of approximately 400,000 transactions per second at the time of the launch of a new project on the network. Since no block can handle 400,000 transactions, transactions were sent to subsequent blocks. Sending transactions to subsequent blocks was not the problem, but since the architecture of Solana is that blocks are massive compared to other decentralized blockchains, validators were not able to handle processing such large blocks. Many of the validators ran out of memory. This resulted in a hard fork of the blockchain, as the network went down from lack of memory in the validators.

Another attack on the blockchain included a layer 2 attack. Arbitrum is a layer 2 scaling solution which is similar to the architecture of validators, send all transactions that don't fit in the unit of execution to the next one. A DDoS flood attack was executed in 2021 that the system was overflooded with transactions that the protocol could not accept any more transactions. The system went offline for over 45 minutes.

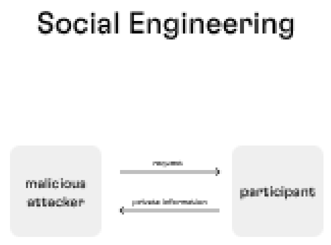
One of the best ways to deal with DDoS attacks is stress-testing the infrastructure before mainnet launch. If the infrastructure is tested for processing potential memory issues, then the risk of a validator going offline is a lot less. Furthermore, a filtering system may be useful in such attacks as well.



4 Social Engineering Attacks

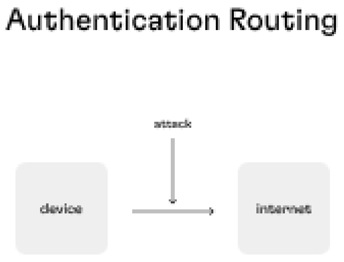
Social engineering attacks come in many different forms and they can be characterized as attacks on the surface layer of the blockchain. These types of attacks are not fundamental errors in the blockchain but rather errors in the way participants in the blockchain use the blockchain. In simple terms, social engineering is the method of manipulating other participants in the blockchain to release private information or perform unintended actions. Malicious attackers trick people into doing something at is unintended. To a certain extent, these types of exploits are unavoidable as many of the participants in the blockchain are not educated about the infrastructure. The most common social engineering attack is the release of blockchain private keys. A private key is used to secure your assets on the blockchain. Each account on the blockchain has a private key and anyone with access to that private key has access to all of the funds in the account. A common way that social engineering hackers attack other individuals is they pose as a trusted identity and manipulate the supporters/users of that trusted identity to send them their private keys. Once unsuspecting supports send private keys, all the money in their account is sent to the account of the malicious attacker and the funds can not be retrieved.

Attacks like these result in billions of dollars in losses in both centralized and decentralized entities. The most common way to defend against such attacks is to understand the different types of social engineering attacks and to understand the fundamentals of the blockchain. Before entering any website, always check that the link is from a trusted entity. Furthermore, never input your private key no matter where it is asked. The idea of a private key is there to protect your funds, all transactions can be done without the input of a private key. Always make sure you have the right information when dealing with cryptocurrency projects.



5 Authentication Routing

Authentication routing or more commonly known as wifi packet injection is a middleman attack. The blockchain itself is connected to the internet, and to interact with the blockchain, users must use the internet. Authentication attacks work when a malicious attacker modifies the behavior of the intended action from your device and the internet. For example, on your device you may intend to send a certain number of financial assets to one account, however, the malicious attacker will manipulate the request to send the assets to his account. This type of attack can be taken to a much bigger scale than it can disrupt the entire blockchain.
The key aspect here is that many of the nodes that run the biggest blockchains today are housed in centralized entities. The biggest centralized entities that the large majority of the blockchain runs on are ISPs. A routing attack or more commonly known as IP hijacking is when a group of IP addresses is corrupted in the Border Gateway Protocol. Through this type of routing attack, you can partition the chain into multiple components. Such an attack will invalidate a huge number of transactions. Furthermore, it can slow down the entire network. The block delivery can be manipulated to take many times more time than it usually does.
The solution to such an attack is protecting connections. If the connections are protected from the ISP's then this attack is less frequent. Such a problem is less about the participants of the blockchain, how blockchains are built, but what they are built upon. Moreover, with all of the attacks outlined, many could be coupled into one to create a massive breach in the blockchain. For example, a routing attack that exploits nodes may make it easier for a 51 percent attack as fewer nodes are participating in consensus. One single attack is too expensive to perform, but many coupled together are dangerous to a blockchain.



6 Smart Contract Attacks

Most of the attacks on smart contracts are labeled as surface-level attacks. Surface-level attacks are attacks that are present because of bad practices within the participants of the blockchain rather than the infrastructure of the blockchain. Very commonly, surface-level attacks are done on smart contracts. The biggest was THE DAO reentrancy attack which caused a hard fork in the Ethereum blockchain. I am going to identify the most common smart contract attacks.

6.1 Reentrancy Attack

The reentrancy attack is to be thought of as a false state attack. In poorly written smart contracts, a function can be called continuously without the first call ending. Such an attack would be a large problem as you are still using the previous state. The state only updates when the function ends, the function has not ended yet meaning you can exploit the state. Furthermore, the reentrancy attack is not only bound to multiple executions of a single function but rather multiple functions may be vulnerable to an invalid state.
The biggest attack on the Ethereum blockchain was a simple reentrancy attack. The attack was so big that the community decided to initiate a hard fork in order to retrieve the funds of the victims. The DAO was hacked because of an error in the smart contract in which Ethereum's fallback function was exploited. Recursive calls were performed on the function, and all funds were sent to the attacker's wallet.
The only solution to such an attack is that all state changes must first be done inside of a smart contract before external capabilities are called. Then there would be no invalid state and a reentrancy attack would not occur. In the example of the DAO, if all internal states would have updated before the transfer of the ether then the attack would not have been valid.

6.2 Frontrunning Attacks

Frontrunning is the term used for when members of a network gain an advantage in terms of execution time. All transactions in the blockchain network can be seen before they are included in a block. Since all transactions can be seen beforehand, in a decentralized exchange, if a buy order is placed and before it is put in a block another participant places a buy order with a higher gas price, it is run before the original participant. Thus, the participant deals with

higher fees in terms of a spread. The main concept is about reaction time, if you can react faster than other opponents in the market, you have an information advantage and with a speed advantage, you can get better deals on your orders than others.
This type of attack is not only present in the blockchain but presenting in centralized financial markets as well. For example, in centralized markets, market makers or high-frequency trading firms may attack other participants in the exchange because of their speed advantage.
The primary solution to such an attack is to remove incentives from specific ordering. Have all the orders in a batch and thus the attacker would gain no advantage when executing their transaction before another participant.

6.3 Integer Overflow

Integers and other data types all have to be stored in memory. However, memory in a system is finite and numbers theoretically can be infinite. Integer overflow is when an integer does not fit the memory space that it is assigned to. Integers in solidity are 256 bits. Numbers exceeding this limit are subject to integer overflow. Numbers less than this limit are subject to integer underflow. In normal systems, integer overflow may lead to buffer overflow to which an attack can gain shell access. Furthermore, a negative balance may become a positive balance depending on the implementation of the system.
In the Ethereum protocol once the limit is reached, the smart contract restarts and goes back to 0. Developers should be aware of this and build systems that complement solidity implementation. Protecting yourself from such an attack is just the matter of writing code that is not subject to such attacks by placing limits and understanding how the language you are writing in handles such errors.

6.4 Reliance on External Data

Although the decentralization of a blockchain is one of the most important elements that attract participants, some applications rely on external data. Most external data is stored in centralized solutions, making it very exploitable when routed to the decentralized blockchain. The data itself may be incorrect or the centralized entity may be compromised or of ill intention. If one of the cases is true, the data once routed to the blockchain is malicious and participants are not aware of this.
One of the best solutions to such a problem is using a decentralized routing mechanism. A decentralized routing mechanism statistically downplays the risk of an attack from a centralized entity. Since the data from the centralized entity is not decentralized, the decentralized blockchain will have no problem accepting it and ensuring the information is correct. Chainlink is the lead provider in decentralized routing.

7 Conclusion

In this paper, we explored exploits that are common on the infrastructure level of the blockchain and the smart contract level of the blockchain. The infrastructure level can be viewed as layer 0 and the smart contract level as layer 1. Common layer 0 attacks include Sybil attacks and denial of service attacks. Sybil attacks take advantage of the consensus mechanism in the blockchain while denial of service attacks take advantage of poorly written validator code. Furthermore, a subset of surface-level attacks is social engineering attacks in which the participants of the blockchain can have their funds stolen by malicious attackers that promise financial returns. Finally, I explored smart contract attacks which are errors made by developers that build on top of the blockchain (layer 1). Smart contract exploits usually exploit poorly written functions and because of the ability of the smart contract for multiple calls, smart contract exploits can be very large.

References

[1] What is blockchain security?

[2] Vitalik Buterin. Ethereum: A next-generation smart contract and decentralized application platform., 2014.

[3] Andrew Lewis-Pye and Tim Roughgarden. How does blockchain security dictate blockchain implementation?, Nov 2021.

[4] Satoshi Nakamoto. Bitcoin: A peer-to-peer electronic cash system, Nov 2008.

[5] Gavin Wood. Ethereum: A secure decentralized generalised transaction ledger, 2022.

[6] Anatoly Yakovenko. Solana: A new architecture for a high performance blockchain, 2018.

[7] Rui Zhang, Rui Xue, and Ling Liu. Security and privacy on blockchain, Aug 2019.

[2] [1] [3] [4] [5] [6] [7]