UNIVERSITY OF THE
AEGEAN

SCHOOL OF ENGINEERING

DEPARTMENT OF INFORMATION
AND COMMUNICATION
SYSTEMS ENGINEERING

"Analysis of decentralised third generation blockchain technology for implementation and security purposes"

Diploma Thesis for the MSc Programme

"INFORMATION AND COMMUNICATION SYSTEMS SECURITY"

GKOGKOTSIS TSERKEZOPOULOS ANTONIOS

10/01/2024

SAMOS

GKOGKOTSIS TSERKEZOPOULOS ANTONIOS

Analysis of decentralised third generation blockchain technology for implementation and security purposes

10/01/2024

Diploma Thesis for the MSc Programme

"INFORMATION AND COMMUNICATION SYSTEMS SECURITY"

Department of Information and Communication Systems Engineering

School of Engineering

Submitted by: Gkogkotsis Tserkezopoulos Antonios

Supervisor: Dr. Yiorgos Stergiopoulos

Members of Supervising Committee:

.....

……

SAMOS

Affirmation

I am the author of this MSc Thesis and any help I have had in its preparation is fully acknowledged and referenced in the Thesis.

I have also cited all sources I used data or ideas, either directly or indirectly.

I certify that this work was prepared by me personally, specifically for this postgraduate Thesis.

Acknowledgements

I would like to thank my supervisor, Dr. Yiorgos Stergiopoulos, for his support, understanding and guidance throughout the preparation of this Thesis.

I would also like to thank my wife, Fenia Deligianni. Without her support, I would not have been able to finalize this Thesis.

Abstract

Since its inception, Blockchain technology has caused extensive debates on multiple areas of human interaction. It sparked sociopolitical, philosophical, economic, and technological evolution similar to the creation of the internet in 1983, yet one of its core aspects, cybersecurity threats and attack vectors has been academically under-explored.

This Thesis entitled "Cybersecurity in 3rd Generation Blockchains" addresses the critical question of how secure third-generation blockchains are against sophisticated cyberattacks, specifically attacks that target the underlying consensus mechanisms and/or their smart contracts' writing capabilities. Utilizing a **systematic literature review methodology**, this study assesses peer-reviewed articles, whitepapers, incident reports and other relevant media to identify and analyze **ten prominent attack vectors**. The findings reveal inherent vulnerabilities that have been and / or could be exploited to siphon millions in USD, highlighting both the sophisticated methods employed by hackers and mostly Advanced Persistent Threats (APTs) as well as the logical mistakes and / or bugs developers create within their code that make exploiting smart contracts and consensus mechanisms easy targets even for non-technical hackers. While the study provides **comprehensive coverage** of documented and potential **attacks**, it acknowledges the limitation of rapidly evolving attack methodologies that may mostly outpace current literature, which is fairly significant for the most part.

The results offer valuable insights for developers, security professionals, and stakeholders in the blockchain domain, presenting comparisons between Ethereum and Cardano, two of the most known and well-developed blockchain ecosystems currently.

Through this analysis, the Thesis contributes to the foundational knowledge necessary for developing robust defense mechanisms in the face of evolving cybersecurity threats.

Finally, this Thesis will present actionable recommendations to enhance the security frameworks of emerging blockchain technologies.

Keywords: *Blockchain, Cybersecurity, Ethereum, Cardano, Proof of Stake attacks, Smart contract attacks, DeFI Attack, Re-entracy Attack.*

Executive Summary

What have we done:

To understand cybersecurity in third-generation blockchains, this Thesis embarked on a journey exploring possible cybersecurity threats. The research began by analysing all **core components of a blockchain**, addressing the initial challenges **distributed networks** need to solve and highlighting the crucial role of **cryptography** in securing blockchain ecosystems. Subsequently, the analysis was extended to examine the evolution of technology across its three generations.

Thesis Research:

This research endeavour thoroughly explored the **landscape of cybersecurity in third-generation blockchains**. We started by establishing a foundation for each security threat, enabling us to grasp the intricacies of each Attack separately. Afterwards, we studied its subject, analysing multiple research papers and other relevant materials. Finally, it is worth noting that in the spectrum of this Thesis, Ethereum has already finalised its transition from Proof of Work to Proof of Stake, promoting it into a 3rd generation blockchain ecosystem.

Findings:

Through our research, we shed light on the most known attack vectors and vulnerabilities third-generation blockchains face today. We divided the research into two segments, Proof of Stake attacks and Smart Contract attacks and expanded our research into two of the most well-known blockchain ecosystems currently available, Ethereum and Cardano.

Proof of Stake attacks includes the following:

- Majority Attack (also known as the 51% attack)
- "Nothing at stake" problem, which leads to the "Double spend" attack
- "Long Range" attack and the
- Stake Grinding attack

The following smart contract vulnerabilities have been explored:

- Integer Overflow attack
- Transaction Order Dependence

- Gas exhaustion Denial of Service and the
- Re-entrance attack

The following attacks have been briefly reviewed occurring in the Decentralised Finance space:

- The Flash Loan attack and the
- Liquidity Pool attack

Discussion:

A comparative analysis related to Ethereum and Cardano and their cybersecurity resilience against the attacks explored during this Thesis. This analysis helps us understand the approaches the two major third generation blockchain ecosystems take and how they mitigate the vulnerabilities we examined. Our goal through this analysis was to offer a better view of the ecosystems and the attacks in this emerging technology.

Who does this Thesis interest?

The insights gained from this research hold significance for various audiences, including researchers, businesses, and public entities. For researchers in particular, our findings provide a resource for exploration and analysis in the constantly evolving field of blockchain cybersecurity. Businesses can leverage our conclusions to strengthen their security protocols and safeguard their assets within the blockchain ecosystem. Public entities may find value in our recommendations as they formulate regulations and policies to ensure the sustainable development of third generation blockchains.

We aspire that the information shared in this Thesis adds value to the discussion on security and promotes a safer and stronger blockchain environment.

Table of Contents

Table of Figures

Abbreviations

ADA	Cardano’s Native coin
AMM	Automated Market Maker
BNB	Binance Native Token
BSC	Binance Smart Chain
BTC	Bitcoin
BTF	Byzantine Fault Tolerance
CA	Certificate Authority
CEM	Constraint Emitting Machines
CFG	Control Flow Graphs
DAO	Decentralized Autonomous Organization
DApps	Decentralized Applications
DBFT	Delegated Byzantine Fault Tolerance
DDoS	Distributed Denial-of-Service
DeFi	Decentralized Finance
DEX	Decentralized Exchange
DoS	Denial-of-Service
DPoS	Delegated Proof of Stake
EIP	Ethereum Improvement Proposal
ESCs	Ethereum Smart Contracts
ETC	Ethereum Classic
ETH	Ethereum
EUTXO	Extended Unspent Transaction Output

EVM	Ethereum Virtual Machine
FBA	Federated Byzantine Agreement
ICO	Initial Coin Offering
IoC	Indicators of Compromise
IOHK	Input Output Hong Kong
JIT	Just in Time
LRA	Long Range Attack
MEV	Maximum Extractable Value (previously called Miner Extractable Value)
NFT	Non-Fungible Token
PBFT	Practical Byzantine Fault Tolerance
PGA	Priority Gas Auctions
PKI	Public Key Infrastructure
PoA	Proof of Authority
PoAc	Proof of Activity
PoB	Proof of Burn
PoC	Proof of Capacity
PoET	Proof of Elapsed Time
PoI	Proof of Importance
PoS	Proof of Stake
PoSpace	Proof of Space
PoW	Proof of Work
RA	Reentrancy Analyzer
TEE	Trusted Execution Environment

UTXO	Unspent Transaction Output

Chapter 1 Introduction

1.1 General Framework

1.1.1 Introduction to Blockchain Technology

At its core, blockchain technology is a digital, decentralized, trustless, immutable ledger. This terminology means that any time a transaction is being made within a blockchain network, this transaction is final. It cannot be altered in any way or form. Once written in a blockchain ledger, the recorded data remains immutable forever. Since a blockchain is a financial instrument, it has captured the attention of hackers and other malicious groups that want to extract value from this new form of money. As per the Chainanalysis Crypto Crime Report, “Despite the market downturn, illicit transaction volume rose for the second consecutive year, hitting an all-time high of $20.6 billion”.(Chainanalysis, 2023) This underscores the criticality of cybersecurity in blockchain operations as the financial stakes of such illicit activities continue to escalate. In order to deep dive into the cybersecurity attack vectors analyzed, in Chapter 2, we provide a short introduction to the characteristics of a blockchain network, the mathematical problems they had to solve, the key cryptographic structures blockchains were built upon and a summary of the three generations of blockchain technology.

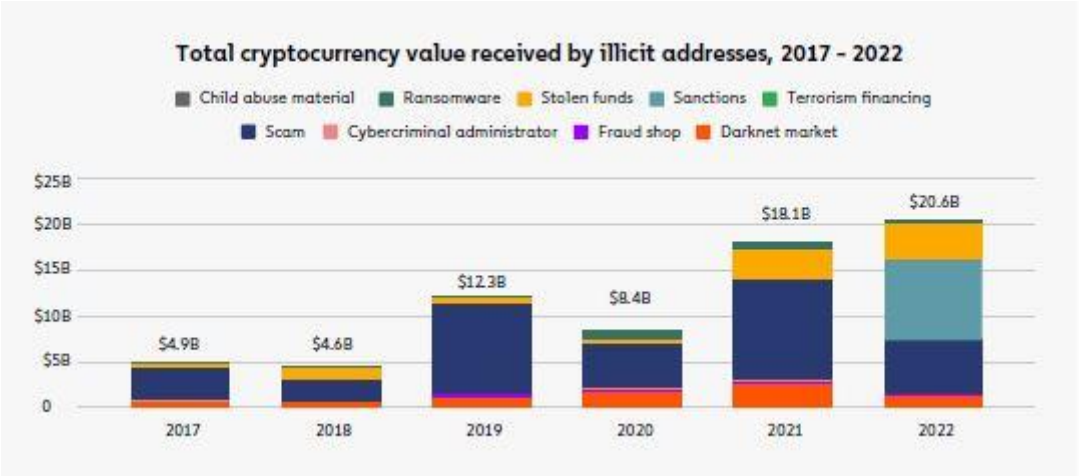


Figure 1: Chain analysis Crypto Crime Report 2023 (Chain analysis, 2023)

1.1.2 Deciphering Blockchain's Core

Decentralized means that “*the recording of the transactions in the distributed ledger is not stored in a single location or controlled by a single authority, but instead, it is distributed across a network of nodes/computers”*.

Trustless means that "*a blockchain is a system for electronic transactions without relying on trust*"(Nakamoto, 2008). Transactions do not require trust between involved parties; trust is placed in the cryptographic algorithms and the network protocol.

Immutability indicates that "*once a transaction is confirmed and recorded on the blockchain, it is considered final and cannot be altered or deleted*".

1.1.3 Critical Challenges in Distributed Networks

Multiple attempts for a native digital currency were made prior to the invention of Bitcoin(Satochi Nakamoto, 2008). Some of the most known attempts were the Digicash (Frankenfield Jake, 2023), E-gold ("E-Gold," n.d.), Liberty Reserve (Kimmell Matthew, 2020) and Hashcash ("Hashcash," n.d.). None of these attempts were able to adequately address the two major mathematical and logical problems that Bitcoin was able to solve. These two problems are the Byzantine Generals Problem (Lewis-Pye and Roughgarden, 2023)and the Double Spending (Karame et al., 2012)problem.

The **Byzantine Generals problem** is a well-known problem in distributed computing and system reliability, where some components of the distributed system may malfunction and give conflicting information. In the context of Bitcoin, this translates to achieving an agreement of the distributed ledger where trust is not centralized but distributed across multiple parties.(Lamport et al., 1982)

The **Double Spending Problem** is another well-known problem of the digital era, where a single digital token can be spent more than once as digital files are easily copied and distributed. The system's credibility is compromised by this risk because, unlike physical money, digital tokens need strong measures to stop fraudulent transactions to ensure the currency's value and prevent inflationary pressures. Bitcoin introduced a solution to this problem by using a peer-to-peer distributed timestamp system that proves the order in which transactions occur. This method, known as Proof of Work (Pow), "effectively prevents double spending in bitcoin's blockchain".(Nakamoto, 2008)

1.1.4 Fundamentals of Blockchain cryptography

This subsection briefly analyses the three cryptographic concepts of Blockchain Technology: the Public Key Infrastructure (PKI) and the Digital Signatures and the Hash Functions.

PKI ensures electronic information transfer for network activities such as e-commerce and internet banking. It uses an encryption model that combines the efficiency of encryption with the security of encryption. (Diffie and Hellman, 2022) The asymmetric part involves generating

a pair of keys. The public key is used openly for encrypting messages, while the private key remains confidential and is used for decryption. (Rivest et al., 1978) To verify the identity of certificate holders and their association with a key, a trusted entity known as the Certificate Authority (CA) issues certificates. PKI also includes a Registration Authority (RA) to verify user certificate requests. Additionally, PKI encompasses roles, policies, hardware/software requirements, and procedures involved in creating, managing, distributing, using, storing, and revoking digital certificates (Housley and Polk, 2001).

Digital signatures play a role in communication protocols, acting as the digital equivalent of handwritten signatures. They provide authentication nonrepudiation and ensure the integrity of digital messages. The concept of signatures (Fang et al., 2020) is based on the use of key-paired cryptography. In this cryptography method, a user signs their message with a key, and anyone can verify the signature by using the corresponding public key (Rivest et al., 1978). This essential pairing process verifies that the key's owner created any message and that it has not been tampered with during transmission. This process thereby guarantees the message's proof of integrity and origin.

Hash functions are widely used tools in cryptography as they generate a fixed-size string that appears to be random. These functions are deterministic, meaning that for a given input, they always produce the same hash value (not the same hash). While being computationally efficient for applications is essential for hash functions, they also should be resistant to preimage attacks (meaning they need to be practically impossible to reverse engineer an input from its hash) and collision attacks (where it is highly improbable to find two different inputs producing the same hash).

1.1.5 A Brief Review of the Blockchain Generations

Upon the inception of blockchain technology, a consensus mechanism called **Proof of Work** (PoW) was introduced. This mechanism serves as the foundation for bitcoin's operations. In PoW, miners compete to solve a puzzle by guessing a value that, when hashed with the SHA 256 algorithm, produces a number equal to or less than the target hash. This process, known as **mining**, requires both effort and energy. Its purpose is to make block creation resource-intensive and statistically improbable for anyone attempting to alter the blockchain or double-spend coins (Lee and Kim, 2023; Nakamoto, 2008). This mechanism allows transactions without relying on any authority, such as a controlling bank. In order to ensure and maintain the system's stability and security while simultaneously managing the rate of block creation for the blockchain, the complexity of this puzzle is periodically adjusted. This difficulty is measured through hash rate, which is how much computing power is needed to solve the puzzle. This adjustment to the hash rate power ensures a critical component of the Bitcoin Network that a block will be mined every precisely ten minutes. If a block is mined in less than

ten minutes, the difficulty increases, and if the block is mined in more than ten minutes, the difficulty is reduced. Miners who successfully solve this mathematical puzzle validate the incoming block of transactions, so they are rewarded for their efforts with generated bitcoins - commonly referred to as receiving the block reward. This incentivization plan encourages miners to contribute their resources towards maintaining the integrity of the network. As we approach Bitcoin's supply limit of 21 million, the block reward gradually decreases over time if it occurs every four years, known as the halving. (Chiu Jonathan and Koeppl Thorsten, 2019)All future block rewards are cut in half during this event, which will continue until every bitcoin is mined. It has been noted that miners will increasingly rely on transaction fees as their motivation instead of the newly minted bitcoins. Although PoW has demonstrated its effectiveness in ensuring a blockchain's core system security, it does have some drawbacks. One concern is the **computational work** required for mining, which raises questions about **energy consumption** and the **environmental impact** of cryptocurrencies. Furthermore, as the network expands and mathematical puzzles become more complex, new miners may encounter difficulties when attempting to join the network, leading to a concentration of mining power among those already possessing such computational resources.

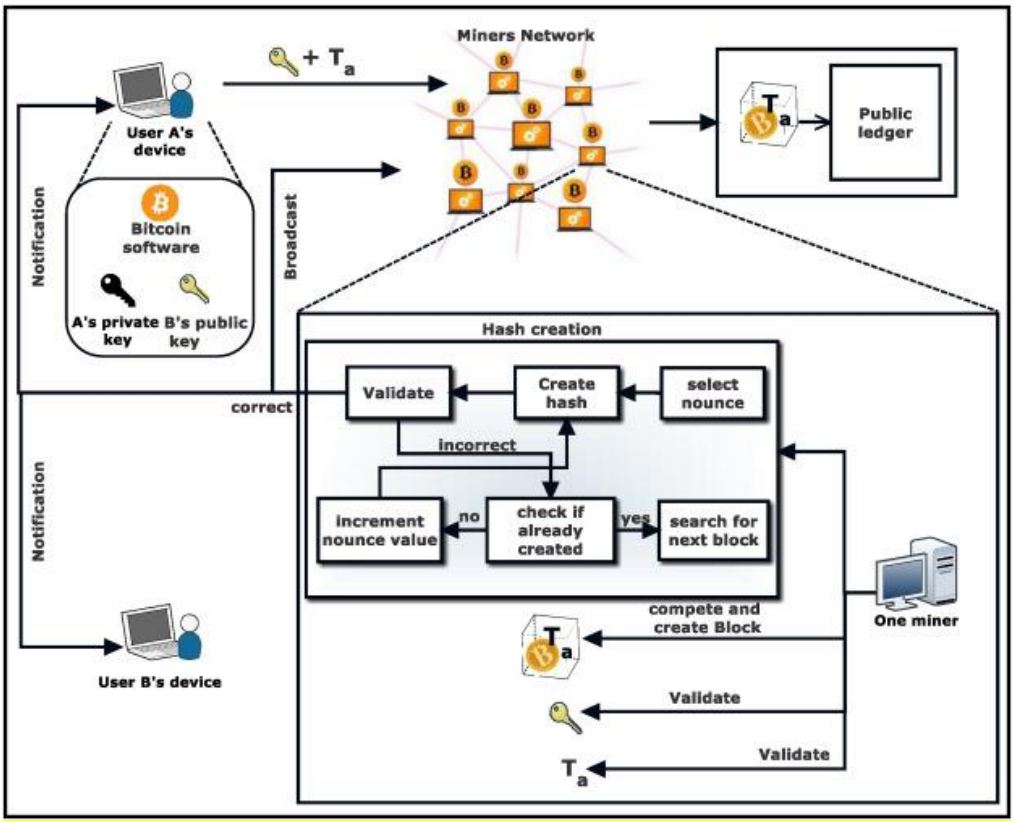


Figure 2:Transaction Lifecycle of the Bitcoin Network (Ghosh et al., 2020)

The second iteration of this technology, also known as **the 2nd generation**, further develops the potential and momentum created by **Bitcoin**. The most significant development in this generation is the adoption of **smart contracts**. These contracts are unique because they are self-executing agreements written directly into lines of code. By introducing transactions and operating based on determined conditions, smart contracts enhance the capabilities of first-generation blockchains. Smart contracts remove mediators' need to carry out their agreements once the specified conditions are fulfilled. This automation improves transaction efficiency and locks a world of possibilities for decentralized applications (Antonopoulos and Wood, 2018)(DApps) built on the blockchain(Buterin, 2014; Christidis and Devetsikiotis, 2016). Currently, the **leading 2nd generation blockchain is Ethereum** (Tikhomirov, 2018). Solidity is the programming language for Ethereum's contracts, allowing developers to write the code that runs on the Ethereum Virtual Machine (EVM). The EVM acts as a computing resource, ensuring that contracts function precisely as intended. This reliability is crucial for establishing trust in the system as it guarantees that deployed contracts will operate without interruptions, censorship or interference from third - parties (Wood Gavin, 2014). Smart contracts have expanded their applications beyond their use cases and now facilitate many innovations in blockchain ecosystems. These include platforms for **Decentralized Finance** (DeFi), Non-Fungible Tokens (NFTs) and **Decentralized Autonomous Organizations** (DAOs). These groundbreaking applications are revolutionizing industries by offering disruptive solutions to traditional centralized systems. However, this wave of evolution faces challenges such as scalability issues, limitations in transaction speed and high gas fees – especially with the network's expansion and increased demand. To overcome these obstacles, ongoing efforts include Ethereum's transition towards a Proof of Stake consensus mechanism and the introduction of layer 2 scaling solutions.

Figure 3: How Smart Contracts Work (Empiricinfotech, 2023)

The **3rd generation of blockchain technology** seeks to address the limitations of scalability, energy efficiency, and complexity that have constrained previous generations. It explores innovative consensus mechanisms beyond the PoW protocols employed by Bitcoin and its predecessors while it aims to reconcile the trade-offs between scalability, security, and decentralization. The most prominent example of a 3rd generation blockchain is **Cardano**.(Shrimali and Patel, 2022) Cardano is a leading blockchain ecosystem that operates on a consensus mechanism called **Proof of Stake (**PoS). In PoS consensus mechanisms, validators are selected based on their stake in cryptocurrency holdings rather than competing for block creation through computationally intensive mining processes. This approach enables Cardano to achieve significantly improved energy efficiency while enhancing scalability. The network is divided into epochs, assigning block creation responsibilities to designated slot leaders. Also, **Cardano streamlines transaction processing, further reducing transaction fees and enabling broader adoption of its blockchain technology.**

Additionally, Cardano takes a **research-oriented approach** by subjecting its protocols to peer review processes, ensuring that any changes made to the network undergo substantial peer review examination before implementation, significantly contributing to maintaining a secure blockchain. Like Ethereum's second-generation blockchain, Cardano has a Turing complete language for creating smart contracts. Their programming language based on Haskell is called Plutus.(Cardano Development Team, 2023)

While there are advantages to transitioning to PoS and implementing technologies, challenges must also be addressed. Some of these security challenges will be analyzed in greater depth in Chapter 2.

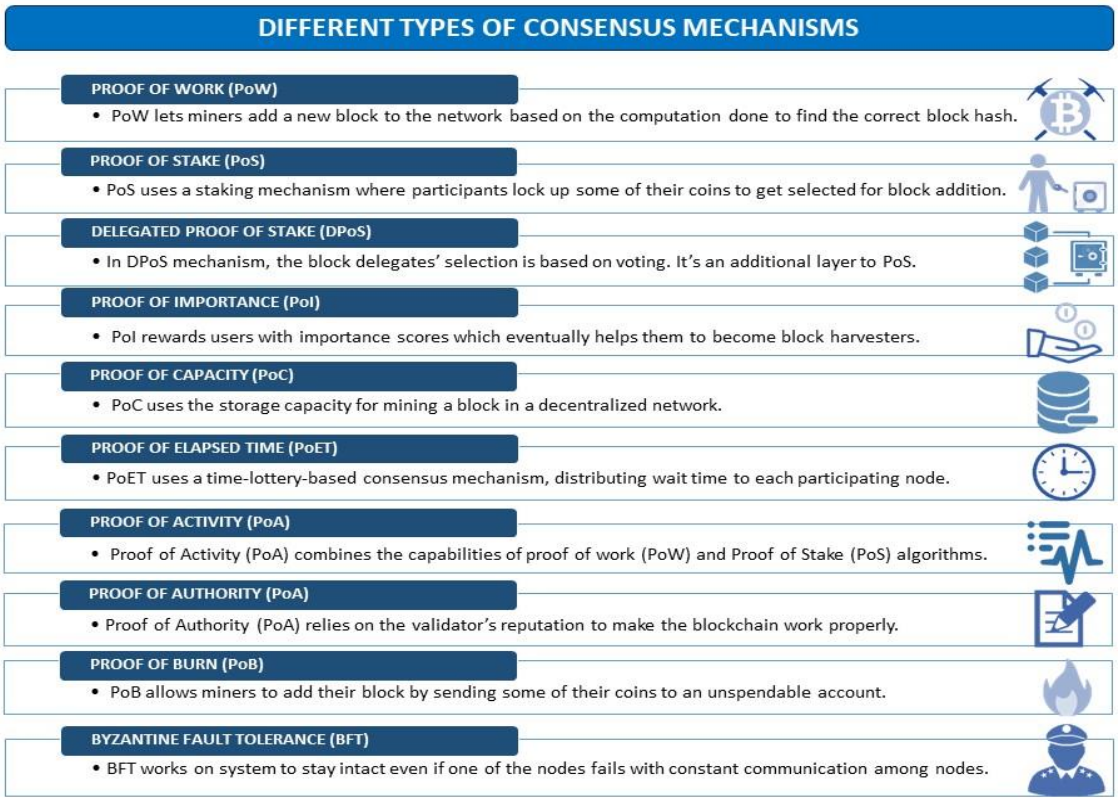


Figure 4: All Consensus mechanisms(Shiksha, 2023)

1.1.6 Alternative Consensus Mechanisms in 3rd-Generation Blockchains

Although PoS is currently the consensus mechanism showcasing the least comptonization of its scalability, security, and decentralization, as this is a new field of technology, many blockchains are trying different approaches. This subsection will briefly analyze the remaining prominent consensus mechanisms and their strengths and weaknesses. The main mechanisms, as shown in Figure 4, include Delegated Proof of Stake (DPoS), Proof of Importance (PoI), Proof of Space (PoSpace) / Proof of Capacity (PoC), Proof of Elapsed Time (PoET), Proof of Activity (PoAc), Proof of Authority (PoA), Proof of Burn (PoB) and Byzantine Fault Tolerance (BFT) Mechanisms.

Delegated Proof of Stake (DPoS): Delegated Proof of Stake (DPoS) is a consensus mechanism that enables high throughput and significantly improves transactions per second compared to the Proof of Work (PoW) systems utilized by Bitcoin and Ethereum(Francisco, 2018). In DPoS, a more democratic approach is being presented. Stakeholders of the native tokens vote for a limited and specific number of Delegators, who are responsible for validating the transactions of the blockchain as well as maintaining the blockchain's integrity. A fundamental weakness in the Delegated Proof of Stake (DPoS) mechanism is in its governance model. This system relies heavily on a few validators and an active community to

properly elect these “Leaders” each epoch. Bad actors can take over the blockchain if a community is not actively participating in the election process. Similarly, few validators mean the blockchain's decentralization aspect is sacrificed to provide faster throughput (Grigg, 2017).

Proof of Importance (PoI): Proof of Importance (PoI) serves as a consensus algorithm utilized by the NEM blockchain to determine the significance of each account within the network. In PoI, every account is assigned an 'importance score' based on its contribution to the NEM economy. This importance score plays a role in determining the probability of an account 'harvesting' a block. This approach ensures that the importance score reflects not an account's wealth but its transaction frequency and overall involvement in the network. PoI measures participants’ activity within NEM, not just their balance (NEM Developer Team, 2018). A fundamental weakness of this mechanism is the “nothing-at-stake” attack. As the cost creation of a block in this mechanism is negligible (compared to Proof of Work and Proof of Stake algorithms), this Attack is highly likely to occur. A more detailed analysis of this attack will be provided in Chapter 2.

Proof of Space (PoSpace) / Proof of Capacity (PoC): These interchangeable names of the exact consensus mechanism describe an innovative mechanism that leverages unused space storage of the participant’s devices. The most well-known blockchain that leverages this technology is “**CHIA Network**”. This approach aims to achieve a more energy-efficient and environmentally friendly alternative to traditional blockchain mechanisms, aligning with Chia's goal of sustainable blockchain practices (Cohen and Pietrzak, 2019). It was mentioned that this approach solves the “Long-range” and “Grinding” attacks. More details will be provided in Chapter 2.

Proof of Elapsed Time (PoET): The PoET Blockchain primarily uses the Sawtooth mechanism. Sawtooth is a blockchain that can be operated as a public and permissioned (private) network. PoET is a *“Nakamoto-style consensus algorithm designed to be a production-grade protocol capable of supporting large network populations”.*(“Hyperledger,” 2024) PoET relies on secure instruction execution to achieve the scaling benefits of a Nakamoto-style consensus algorithm without the power consumption drawbacks of the Proof of Work algorithm. Sawtooth's core ledger architecture differs from application-specific settings, resulting in enhanced security and speed optimization. The primary component of Sawtooth consists of a decentralised ledger that documents transactions and smart contract operations, guaranteeing uniformity and dependability among all network nodes. In addition, the platform incorporates the "Sawtooth Lake" smart contract engine to facilitate quick deployment and execution. It is complemented by a RESTful API that enables streamlined

interfaces with the ledger. Sawtooth is designed to be scalable, meaning it can handle large networks with thousands of nodes and execute millions of transactions per second. This makes it a perfect choice for a diverse range of applications, from supply chain management to digital asset tracking and voting systems. In addition, Sawtooth's design philosophy promotes simplicity in application development, enabling developers to create and manage apps using their preferred programming languages without requiring an extensive understanding of the underlying blockchain infrastructure.(Sawtooth Developer Team, 2022)

Proof of Activity (PoAc): PoAc is a mechanism that combines the PoS and PoW mechanisms. During the process of creating new blocks in PoAc, participants are needed to fulfil challenges that are comparable to those that are found in PoW blockchains. Once a new block has been mined, validators comparable to those found in PoS blockchains are called to validate and sign the block. The incentives obtained during the verification process are distributed equally between the miners and the validators. (Bada et al., 2018)

Proof of Authority (PoA): The PoA consensus mechanism is a system where nodes must be authorized before participating in the blockchain. Once authorized, all nodes can generate new blocks and earn rewards, ensuring that wealthier nodes do not gain an unfair advantage. PoA is incredibly efficient in network utilization due to its quick selection of block producers, allowing more time for transaction data transmission (Web3 for Better Whitepaper 3.0 2023). One limitation, however, in PoA is the absence of a mechanism to prevent node misconduct. While any misconduct can be later used as evidence against a node, PoA does not inherently deter actions from happening in the first place.

Similar to Nakamoto consensus algorithms, PoA only offers probabilistic security assurance for transactions, meaning there is a slight but non-negligible possibility that an attacker could disrupt the network. This risk becomes more significant in scenarios like large-scale network partitioning where PoA might not effectively mitigate it in highly asynchronous situations requiring high security. (Web3, for Better Whitepaper 3.0 2023) The most known drawback of this technology is the increased complexity of the consensus algorithm. Similarly to DPoS, the validators need to be chosen wisely to avoid centralization of power.

Proof of Burn (PoB): PoB is a mechanism that suggests the irreversible destruction of native tokens of their blockchain. The process is initiated when a miner sends coins to an address known as the “burn address”. This mechanism is cost-effective since the only action is to ensure the blockchain's security of the miner sending his tokens to the burn address (Karantias et al., 2012). In 2014, Slimcoin used this technique, but it has subsequently been terminated.(Slimcoin Development Team, n.d.)

Byzantine Fault Tolerance (BFT) Mechanisms: The BFT mechanism aims to reduce the number of bad actors and malicious nodes within its ecosystem so that the system will not crash while the honest nodes remain on consensus. The three most widely used mechanisms in blockchains are the Practical Byzantine Fault Tolerance (PBFT), the Federated Byzantine Agreement (FBA), and the Delegated Byzantine Fault Tolerance (DBFT).(Bada et al., 2018) PBFT is a protocol that implements the notion of a byzantine in an asynchronous environment. FBA is a protocol that uses groups of nodes known as "**Quorum Slices**" to achieve consensus through "**Quorum**". Quorums are agreements that cannot change with the change of time. This concept is slicing the number of nodes into groups the member can trust and use to achieve faster and more trustworthy consensus. Stellar is the most known blockchain using this consensus method(L. Matthews., 2021). DBFT is a consensus mechanism in which token holders have the right to vote on the delegators regardless of the coin amount they possess. One of the delegators randomly becomes the "speaker". The speaker creates the new block and then presents it to the validators for proof and validation(C. Comben, 2019). NEO is the most known blockchain that uses this technology.("Neo Project," 2021) Unfortunately, as it is clear, all these mechanisms mentioned above sacrifice the decentralized aspect of their blockchain technology to achieve faster and more trustworthy blocks.

1.2 Research Questions

The following are the **Research Questions** of this Thesis, which will be analyzed in the following chapters.

Research Question 1: What are the fundamental elements of blockchain technology, and how do they contribute to the cybersecurity challenges in third-generation blockchains?

Research Question 2: What are the primary cybersecurity risks that are unique to third-generation blockchains, and how do they appear in prominent blockchain ecosystems like as Ethereum and Cardano?

Research Question 3: How can businesses, researchers, and public entities apply the findings of this thesis to improve cybersecurity practices in third-generation blockchain ecosystems?

Chapter 2 Attacks and Vulnerabilities Overview

Within the constantly evolving realm of third generation blockchain technology, the research of this Thesis reveals **three primary categories of vulnerabilities** that pose a substantial risk to their functional reliability.

- The first category includes **attacks that specifically target the consensus mechanisms** that form the foundation of these blockchain systems. The processes responsible for preserving the decentralised integrity and security of the blockchain are vulnerable to a range of sophisticated cyber assaults. In Proof of Stake (PoS) systems, vulnerabilities such as the 'Nothing at Stake' problem and the potential for majority attacks present substantial concerns.
- The second category of vulnerabilities **relates to the smart contracts** essential to these ecosystems. Smart contracts, despite their automation and speed, are frequently susceptible to security breaches as a result of vulnerabilities in their code or logic. These vulnerabilities can potentially cause significant financial losses and erode confidence in the integrity of the blockchain infrastructure. Instances of such vulnerabilities encompass re-entrancy attacks, gas limit concerns, and complications emerging from transaction order dependencies.
- The third category, which is are **attacks in the Decentralized Finance sector**. This vector leverages the psychological facets of human behaviour to persuade individuals to compromise the security of the blockchain network. These attacks often involve techniques such as phishing, pretexting, and baiting. This focus on attacking humans highlights the significance of using technological measures to protect against it and implementing extensive programmes to educate and raise awareness among users.

2.1 Proof of Stake Attack Vectors

These types of attacks, also named the "**black swan attacks**", (Caldarelli and Ellul, 2021) are rarely recorded. Such attacks will result in catastrophic damage to the confidentiality, integrity and availability of data within the blockchain and the reputation of the ecosystem and all its developers and projects.

2.1.1 Analyzing the 51% (Majority) Attack

Description: The **Majority Attack** is widely recognized as one of the fundamental threats associated with Blockchain Technology. In the case of Bitcoin's PoW blockchain "*a 51% attack occurs when an individual or a group gains control over than half of the mining power. This level of control grants them the ability to monopolize block creation and collect all the rewards. Additionally, they can hinder users' transactions by excluding them from blocks, potentially*

even reversing transactions to spend coins twice" (Nakamoto, 2008). On a PoS blockchain, a 51% attack occurs when a single entity or group acquires over 50% of the staking power. This enables them to manipulate the network by altering consensus mechanisms or engaging in double spending activities. Unlike Proof of Work (PoW),(Eyal and Sirer, 2018) where such an attack necessitates majority power, PoS attackers would need to own a majority share of cryptocurrency – an expensive endeavor that risks devaluing their holdings.(Neuder et al., 2020). The Majority Attack in blockchain technology is mainly defined by the attacker's capacity to modify the transaction history of the blockchain. In Proof of Work (PoW) systems, this entails obtaining a majority of the processing power within the network. Subsequently, the attacker can generate blocks at a higher rate than the remaining network, enabling them to establish the longest blockchain that other users in the network will deem legitimate. In PoS systems, the attack occurs when an individual possesses over 50% of the cryptocurrency's overall staking power, granting them an imbalanced level of control over the validation of new blocks. This feature allows users to choose exclude or alter the sequence of transactions, hence aiding the act of double-spending. The success of the attack depends on the idea that the longest or most validated chain is regarded as the legitimate blockchain according to the consensus rules of the network. The fundamental idea behind the attack is to subvert the decentralization of the blockchain, transforming it into a system that is governed by a sole entity or a collective. This consolidation undermines the fundamental principles of blockchain as a distributed ledger system. The Majority Attack reveals the weaknesses in the consensus processes that control blockchain networks. It is crucial to rectify these weaknesses in order to sustain the security and reliability of blockchain systems. (Sapirshtein et al., 2015)A **detailed analysis of this attack vector and the challenges of multiple consensus mechanism** is provided (Sayeed and Marco-Gisbert, 2019). While PoW is highly more susceptible to attacks due to the ease of acquiring a majority share of power, PoS appears less susceptible because obtaining a majority stake is typically more expensive. This does not exclude the possibility of a 51% Attack in a PoS ecosystem, but it indicates that an attacker would be required to spend a lot of economic resources of his own to attack and devalue his tokens, reducing his benefit as well.

Step by step explanation of the Attack: A Majority Attack on a blockchain network is carried out by a predetermined sequence of events. At first, the attacker endeavors to amass more than 50% of the network's mining or staking power, a task that can be more or less intricate and expensive depending on the network's scale and configuration. After surpassing this level, the assailant initiates the process of mining or validating blocks in a covert manner, so

generating a clandestine alternative version of the blockchain. This confidential blockchain remains concealed from the general public network, while the latter continues to function in its regular manner. As the attacker's private chain expands, it surpasses the length of the public blockchain as a result of the attacker's dominant control. Once the attacker chooses to make their private chain available to the network, the rules of the blockchain protocol determine that the longest chain is regarded as the legitimate one. As a result, the network substitutes the previously acknowledged blocks with the chain created by the attacker. This substitution has the potential to nullify previously verified transactions, allowing the assailant to duplicate the use of coins. The success of the attack depends on the attacker's capacity to retain control of the bulk of the network's computational power and the network's adherence to the rule that the longest chain is considered valid. The Majority Attack poses a direct threat to the security of the blockchain and necessitates substantial resources and strategic planning for its execution.

Methodologies used in the Attacks: (Gervais et al., 2016)To carry out a Majority Attack, assailants utilize distinct techniques, necessitating meticulous planning and deployment of resources. Initially, it is necessary to evaluate the target blockchain network to ascertain its susceptibility to a 51% attack. This entails examining the hash rate distribution in Proof of Work (PoW) blockchains or the distribution of staking power in Proof of Stake (PoS) systems. Adversaries can employ simulation tools to simulate the network and forecast the consequences of their actions. In Proof of Work (PoW) systems, obtaining the required computational capacity usually entails establishing a substantial quantity of mining rigs or, more frequently, leasing hash power from mining pools. In PoS blockchains, assailants must amass a substantial quantity of the digital currency, either by acquiring it from the market, exchanges, or alternative methods. The process also encompasses strategizing the time and implementation of the assault, taking into account variables such as network traffic and transaction volume to optimize the impact. Attackers must also anticipate the consequences of their attack, such as any defensive actions taken by the network and the community. The economic impact of the attack is a critical factor to consider, as the necessary expenditure needs to be balanced against the possible benefits of double-spending or other harmful actions. The tactics employed in Majority Attacks are intricate and necessitate substantial technical proficiency and resources.

Exploited vulnerabilities:(Rosenfeld, 2011) A Majority of Attacks capitalize on particular serious vulnerabilities in blockchain networks. The decentralised character of blockchain has both advantages and disadvantages. On one hand, it offers protection against centralized control. On the other hand, it lacks a central authority to prevent the concentration of power in

the hands of a single institution. In Proof of Work (PoW) blockchains, the assault is made possible by obtaining or leasing a significant amount of processing power to dominate the majority of the network's hash rate. This can be accomplished by implementing a comprehensive mining infrastructure or utilizing mining pools' capabilities. In PoS systems, the attack entails amassing a significant portion of the network's staking power. This can be expensive, but it is achievable if the attacker possesses considerable financial resources. The assault exploits the blockchain consensus mechanism, requiring the longest or most validated chain to be deemed the authoritative account of events. This rule is utilized to deliberately alter the historical records on the blockchain, facilitating the reversal of transactions. The feasibility of the attack is also contingent upon the scale and concentration of mining or staking power inside the network; smaller or more centralized networks are more susceptible to such attacks. The economic ramifications of such attacks are substantial, as they can result in the erosion of confidence and financial instability within the network. Gaining a comprehensive understanding of these vulnerabilities is of utmost importance in devising effective countermeasures to safeguard blockchain networks against Majority Attacks. However, in a PoS ecosystem like Ethereum, instead of the 51% Attack, many bad actors could delay and halt transactions and the blockchain creation itself with much lower resources. At 33% of the majority, attackers can cause finality delays. At 34% of the majority, attackers cause double finality, and finally, at 50% of the majority, they could maintain two forks and prevent finality. (Corwin Smith, 2023; Deirmentzoglou et al., 2019)

Potential impacts of Majority Attacks:(Bonneau et al., 2015)(Bonneau et al., 2015) The potential impacts of Majority Attacks on blockchain networks are significant and complex. The primary consequence is the potential occurrence of double-spending, when the attacker utilizes the same digital currency twice, resulting in financial detriment for both businesses and consumers. These attacks have the potential to significantly undermine the confidence and dependability of the blockchain, as they reveal inherent vulnerabilities in its security. The network's reputation and the worth of its affiliated cryptocurrency can endure substantial harm, potentially resulting in a decrease in user acceptance and financial commitment. Majority Attacks not only result in cash losses but also interrupt the regular operation of the blockchain, causing a halt in transaction confirmations and a slowdown in network operations. This interruption can have wider ramifications for apps and services that depend on the blockchain, impacting many sectors such as financial services and supply chain management. The apprehension of possible assaults might result in heightened centralization as networks may opt to combine mining or staking power in order to avert such incidents, hence contradicting

the decentralised principles of blockchain technology. Achieving a successful Majority Attack could have enduring consequences, such as increased regulatory scrutiny and demands for more rigorous security measures. To ensure the continued viability of cryptocurrencies and blockchain networks, it is crucial to counter the threat posed by Majority Attacks. The imperative for perpetual innovation in blockchain security is emphasized by the dynamic nature of these assaults and the advanced capabilities of attackers.

Examples of Majority Attacks: (Liao and Katz, n.d.)Various prominent case studies exemplify the practical effects of Majority Attacks on blockchain networks. An extensively documented case is the assault on Bitcoin Gold in May 2018, during which attackers successfully gained control of the network's hash power and engaged in double-spending activities, resulting in the loss of around $18 million. This incident exposed the weaknesses of smaller Proof of Work (PoW) blockchains in terms of centralization of hash rate. In January 2019, Ethereum Classic saw a notable incident where attackers rearranged more than 100 blocks, resulting in significant financial damages and prompting concerns regarding the security of smaller Proof of Stake (PoS) networks. In April 2018, the Verge blockchain had a 51% attack, during which malicious actors used a coding flaw to mine blocks faster, leading to significant illicit profits. These instances highlight the tangible dangers linked to Majority Attacks and emphasize the necessity for robust security mechanisms in both Proof of Work (PoW) and Proof of Stake (PoS) blockchain systems.

Furthermore, they emphasize the significance of ongoing surveillance and revision of blockchain protocols to tackle developing weaknesses. These occurrences function as a warning for blockchain developers and users, underscoring the importance of being watchful and implementing proactive security measures. Unfortunately, while a substantial body of research addresses the vulnerabilities and defense mechanisms related to the 51% attack in PoW blockchains, exploring this topic is constrained by the availability of open research. This highlights the need for more accessible scholarly communication that could foster wider discussions surface in the field of blockchain security.

Practical Aspects of Majority Attacks: (König et al., 2020; Saad et al., 2019)The practical aspects of majority attacks in blockchain technology involve various factors, including technical viability and economic and strategic consequences. A considerable amount of computational resources is necessary to carry out a Majority Attack in Proof of Work (PoW) systems. This typically involves obtaining or renting a significant amount of mining power, which can be financially burdensome and logistically complex. In Proof of Stake (PoS) blockchains, an assault requires possessing a significant share of the cryptocurrency, which entails a considerable amount of capital and the potential danger of depreciating the attacker's

investment. The timing of the attack is critical; assailants frequently bide their time for favorable instances when the network is susceptible, such as during periods of diminished hash rate in Proof of Work networks. The attack's success relies on retaining dominance over the bulk of the network's power while avoiding early notice, as prompt detection can result in countermeasures. The consequences of a Majority Attack are of comparable significance since they can result in an erosion of confidence in the blockchain, potentially leading to a decrease in the cryptocurrency's worth and user acceptance. Attackers should also consider the possibility of retaliatory measures by the network community, such as implementing hard forks that can render the attack ineffective. Such attacks' moral and legal consequences are substantial, potentially resulting in judicial proceedings against the individuals responsible. Blockchain networks must employ stringent security measures and ongoing surveillance to identify and avert Majority Attacks, which pose a significant threat. Furthermore, the feasibility of these assaults renders them a crucial issue for blockchain developers, users, and investors, emphasizing the necessity for continuous vigilance and advancement in blockchain security. An alternative safeguard, analyzed in Chapter 1.6, is the alternative consensus mechanism focusing on preventing a 51% attack.(Shifferaw and Lemma, 2021) By requiring transactions to include a reference to a recent block and only allowing coin-days destroyed to be counted in blockchains that build off of that block, it would be impractical to maintain secret alternative blockchains. (Larimer Daniel, 2013)

2.1.2 The “Nothing at stake” vulnerability and the Double Spend Attack

Description: This attack vector is another attack that can have potentially catastrophic results for any blockchain ecosystem, as a Nothing at stake vulnerability could result in a Double Spending issue, negating one of the two fundamental problems a blockchain needs to be able to solve in order to work. This Attack can only occur in the Proof-of-Stake consensus mechanism and other hybrid interpretations of this mechanism (DPoS,etc). This occurs because in contract to a Proof of Work ecosystem, in Proof of Stake validators have the ability to create multiple blocks for the same time slot without incurring significant computational costs. This vulnerability occurs when forks of a blockchain are created. As shown in Figure 5, with those new forks validators can produce blocks on branches, creating challenges and competing with the honest nodes on how can reach a consensus first, due to an excessive number of valid blocks flooding the network.

Main concepts: (Gazi et al., 2018)The fundamental idea behind the "Nothing at Stake" attack is centered on the effortless ability of validators to mine on several blockchain forks without incurring substantial expenses, a situation that is not feasible in Proof of Work (PoW) systems. This capability results in a scenario where validators may endorse many forks concurrently,

hence augmenting the probability of double-spending assaults. The attack compromises the core principle of the blockchain, which is to guarantee the unchangeability of transactions and agreement on the current state of the ledger.

Step by step explanation:(König et al., 2020) In a "Nothing at Stake" attack, validators are motivated to mine on every fork of a blockchain, as it is economically advantageous for them to do so. As a result, there are numerous iterations of the blockchain being expanded at the same time. Validators engage in this practice to guarantee their receipt of block rewards, irrespective of the current fork. As an increasing number of validators embrace this approach, the network encounters difficulties in achieving consensus, resulting in confusion and lack of uniformity. The attack reaches its peak when this state of confusion is manipulated to carry out double-spending, which involves spending the same digital asset on multiple forks. This fraudulent activity is facilitated by the network's failure to reach a consensus on a singular, authoritative version of blockchain history.

Methodologies used: In order to carry out a "Nothing at Stake" assault, attackers often wait for a specific point when the blockchain is prone to bifurcation. Subsequently, they verify and generate blocks on every iteration of the diverged blockchain. In PoS systems, the computational work required is modest compared to the energy-intensive mining needed in PoW. The methodology capitalizes on the inexpensive nature of block creation in PoS to undermine the consensus mechanism and enable the act of double-spending.

Exploited vulnerabilities: The "Nothing at Stake" attack draws on the inherent design of Proof of Stake (PoS) by taking advantage of the low cost associated with block creation. Attackers exploit this situation by engaging in mining on several forks, a tactic that would be economically impractical under PoW due to the exorbitant energy expenses. This attack is especially potent in times of network instability, such as during system upgrades or when transaction volumes are high, as these circumstances increase the likelihood of forks.

Potential impacts: The possible consequences of a "Nothing at Stake" attack are substantial. The potential consequences encompass compromising the integrity of the blockchain through facilitating double-spending, diminishing user confidence in the system's security, and potentially resulting in financial detriment. The attack can also induce perplexity and incongruity in the blockchain's history, rendering it arduous to ascertain the veritable condition of the ledger. Over time, these assaults have the potential to discourage the acceptance and utilization of PoS-based blockchains.

Examples: Although there is limited documentation on specific occurrences of "Nothing at Stake" attacks, there have been theoretical assessments and simulations undertaken. Additional mitigation actions can be achieved by improving the suffix length required for consistency, a fundamental design parameter in PoS blockchain ecosystems. By trimming the suffix length of this parameter, the consensus mechanisms can be improved and guarded by attacks such as the Nothing-at-stake. In more detail, “the quadratic dependence on the suffix length in current analyses, affecting the time required for transaction settlement and questioning the PoS systems' intrinsic limitations is one challenge of significant importance for blockchain ecosystems”. (Blum et al., 2019) As this Attack highly affects all PoS ecosystems, I was also studied and mitigated in the Cardano ecosystem. Stake-bleeding(Gazi et al., 2018), an attack similar to the Nothing-at-Stake Attack is analyzed. In that Attack, the attacker could simulate an honest blockchain while secretly accumulating rewards and fees on a parallel chain. By secretly building a parallel copy of the honest blockchain, the attacker could create a situation where double spending occurs, if she/he decides to reveal the hidden blockchain and become the main chain due to its length. This attack methodology highlights sophisticated attack vectors that need to be under consideration for all PoS ecosystems that use the longest chain method to validate their blockchain, without any frequent checkpoints. Lastly, Cardano was able to safeguard against this Attack and similar by introducing Ouroboros. (Badertscher et al., 2019a, 2019b; Kiayias et al., 2019; Kiayias and Russell, 2018) Ouroboros provides security properties similar to that of the Bitcoin blockchain and proposes a novel reward mechanism to incentivize PoS validators to behave. Ouroboros also discourages stakers from forming new forks of the blockchain thus adding an additional layer of security for the blockchain.

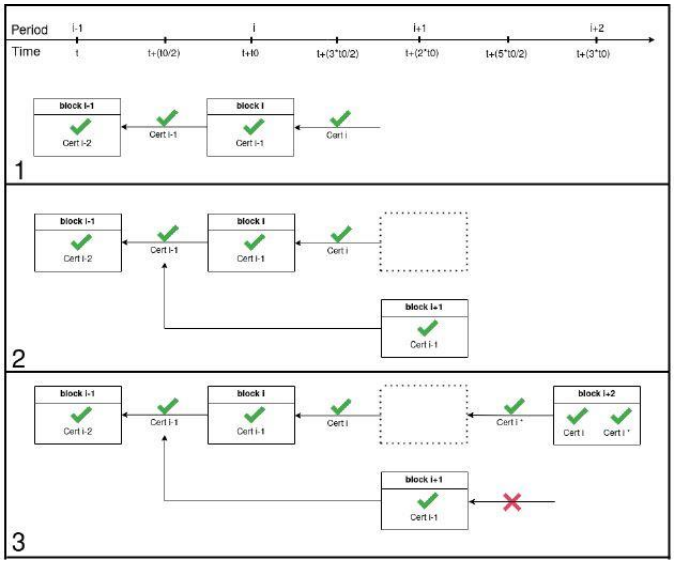


Figure 5: Nothing at stake exploitation leading to a double spend. (Lys et al., 2023)

As shown in Figure 6, researchers (Gaži et al., 2022) were able to verify that both Long-range as well as Nothing-at-stake attacks are more likely to occur in a PoS rather than a PoW because the adversary's capacity to insert multiple blocks in a single slot enables the double spending attack, a feat unachievable in PoW systems.

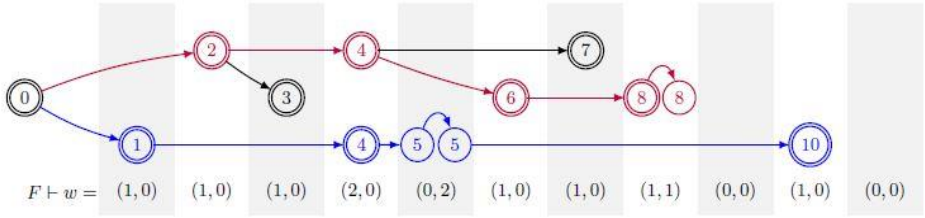


Fig. 2. A PoW tree F for the characteristic string w with $\Delta = 1$. Honest vertices are shown with double-struck boundaries, while adversarial vertices are simple circles. Vertices are labeled with $\mathsf{l}_{\#}(\cdot)$. The tree indicates a k-consistency violation for $k = 4$ —given by the red and blue chains—in a circumstance where the simple private-chain attack does not succeed: in particular, the tree constructs two alternate chains with disjoint suffixes of length 5, while only three adversarial proofs of work are discovered over this period. We remark that $F = F_{\lceil 1}$, since the last symbol of w is $(0,0)$, and that $\overline{F_{\lceil 1}}$ is obtained by removing the adversarial vertex with label 8. Thus $\mathsf{len}(\overline{F_{\lceil 1}}) = 5$, this maximum length achieved by the blue chain. Note, then, that the two chains indicated in red and blue each have advantage equal to zero, and both are dominant. Considering that these chains share no vertices after the root, they witness $\beta_1(F) \geq 0$ for the tree F and hence $\beta_1(w) \geq 0$ for the characteristic string w.

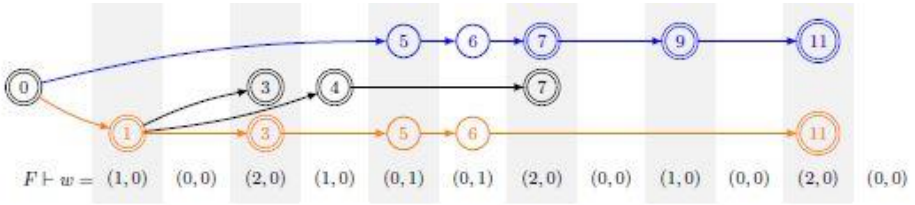


Fig. 3. A public PoS tree F for the characteristic string w with $\Delta = 1$, using the same graphical representation as Fig 2. The tree indicates a successful double spend attack given by the orange and blue chains and highlights a notable feature of the proof-of-stake setting: the adversary's ability to play multiple blocks in slots 5 and 6 permits a double spend attack in circumstances where there would be no attack in the proof-of-work case. We remark that $F = \overline{F} = F_{\lceil 1} = \overline{F_{\lceil 1}}$, since all leaves of F are honest and the last symbol of w is $(0,0)$. Clearly $\mathsf{len}(\overline{F}_{\lceil 1}) = 5$. The two chains indicated in red and blue each have advantage equal to zero, and both are dominant. Considering that these chains share no vertices after the root, they witness $\mu_1(F) \geq 0$ for the tree F and hence for the characteristic string w.

Figure 6: Comparison of PoW and PoS trees during a long rage attack. (Gazi et al., 2018)

Although this Attack has never actually occurred until now in any blockchain ecosystem as it can be partially mitigated by the penalization called “slashing” of fraudulent transactions and the validators that approve them (Lys et al., 2023), it cannot be mitigated in blockchains that allow for parallel block production, such as sharding will be for Ethereum’s blockchain ecosystem, (Shin Laura, 2023) and Hydra will be for Cardano’s ecosystem.(Chakravarty et al., 2021) To tackle this problem, a protection strategy is proposed to combine Byzantine Fault Tolerance (BFT) and Nakamoto consensus principles, thus enabling block production while maintaining network integrity. Another mitigation action for this Attack is the creation of exact probability formulas for such attacks instead of relying on commonly used estimates found in existing literature. This methodology will provide vendors, validators, and other interested parties with a method to determine how many confirmation blocks are required before accepting transactions. (Karpinski et al., 2021a)

Practical aspects: (Karpinski et al., 2021b)The practical implications of "Nothing at Stake" assaults in Proof of Stake (PoS) blockchains revolve around the simplicity of carrying out such attacks, facilitated by the inexpensive validation of many blockchain forks. The easiness of this situation poses a substantial security obstacle for PoS networks, particularly during instances of network instability or when forks are probable. To effectively defend against such assaults, it is necessary to develop consensus methods that actively discourage or penalize validators that attempt to work on several chains. This approach is implemented in more recent Proof of Stake (PoS) protocols like Ouroboros. The attack highlights the importance of ongoing innovation in blockchain consensus processes to guarantee the integrity and security of the network.

2.1.3 The threat of “Long Range” Attack

Description: Long-range attacks similar to the Nothing-at-stake attacks that we analyzed above, are a significant threat to any PoS blockchain. Even more prominent of a threat than in proof of work (PoW) protocols due to the issues of Costless Simulation and Weak Subjectivity. This attack vector occurs by creating a fork from the genesis block or any subsequent block that significantly diverges from the main chain (Figure 7). The Attack is successful if this alternative branch becomes longer and overtakes the main chain.

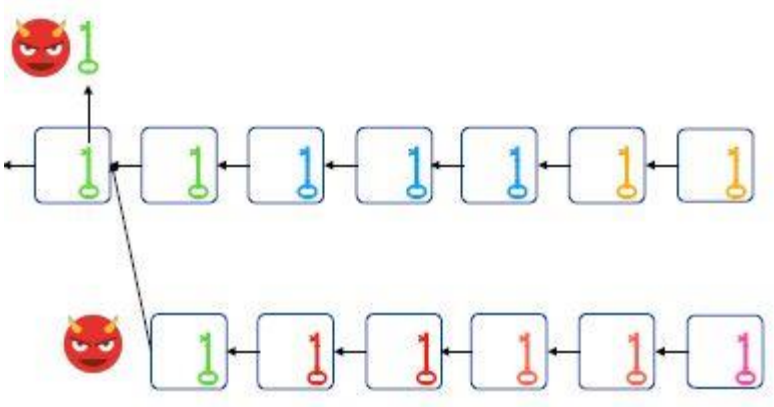


Figure 7: Long-Range Attack (Azouvi and Vukolić, 2022)

Long-range attacks can be categorized into three categories: Simple Attacks, Posterior Corruption Attacks and Stake Bleeding Attacks. In the table below, we see a summary of its methodology and its proposed mitigation patterns. (Deirmentzoglou et al., 2019)

Name	**Simple**	**Posterior Corruption**	**Stake Bleeding**
Description	Attackers do not need to consider block timestamps, creating branches that may contain different transactions	Utilizes the private keys of past validators to forge blocks and increase the attacker's chances of overtaking the main chain	Involves stalling the main chain while the attacker's branch accumulates more blocks and stake, eventually outpacing the main chain
Mitigation Actions	**Longest Chain Rule** The most straightforward technique is used by default in conjunction with other techniques in PoS protocols. It defines the main chain as the branch with the most blocks.	**Key-Evolving Cryptography** This technique involves using key-evolving signatures where the key's lifetime is divided into epochs, and a different private key is used for each epoch while the public key remains the same.	**Plenitude Rule** It detects changes in the density of blocks within competing branches. A branch where the density of blocks suddenly increases can indicate an attack.

	Moving Checkpoints This technique limits how far back in the chain the reorganization can occur, allowing only the latest blocks to be reorganized.	**Context-Aware Transactions** It includes a reference to a specific block within a transaction, binding it to a particular branch and making it difficult for an adversary to move transactions to a different branch.	**Economic Finality** This measure introduces financial punishments for validators who engage in misconduct, which could lead to the loss of staked coins.
			Trusted Execution Environment (TEE) This solution involves all signing within a TEE, protecting private keys from leakage and preventing them from being used to sign blocks on a malicious branch.

Main concept: (Li et al., 2020)The primary principle underlying long-range attacks is using the PoS method, wherein the expense of generating blocks is significantly lower compared to PoW. Adversaries use this opportunity by constructing an alternate sequence from an initial stage in the blockchain's past. The objective is to extend this alternate chain to surpass the present primary chain, capitalizing on the principle of the blockchain that the longest chain is typically deemed the legitimate one. This assault undermines the core security principles of blockchain technology, specifically the unchangeability and conclusiveness of the recorded transactions.

Step by step explanation:(Gazi et al., 2018) During a long-range attack, the attacker initiates the process by selecting an initial block within the blockchain, typically the genesis block. Subsequently, they clandestinely initiate the construction of an alternative branch of the blockchain from this point on. The existence of this other branch extends over a period of time, unbeknownst to the participants of the honest network. Given the inexpensive nature of forging blocks in PoS, the attacker can persist in this activity until their concealed chain surpasses the

length of the primary chain. After the attacker's chain exceeds the length of the main chain, they distribute it to the network. The network's protocol, which usually regards the longest chain as the legitimate one, can consequently approve the chain created by the attacker. This acceptance leads to a profound restructuring of the blockchain, involving the rewriting of its whole history. The success of the attack depends upon the attacker's capacity to maintain clandestinely until their chain exceeds the length of the main chain.

Methods used: (Buterin Vitalik, n.d.)The strategy for carrying out a long-range attack entails carefully choosing an initial block to diverge from and systematically constructing a covert alternate blockchain. The attacker must guarantee that their blockchain remains concealed from the sincere network until it exceeds the primary blockchain in terms of length. Strategic planning and patience are necessary for this task, as disclosing an alternative chain too early can result in failure. In order to sustain the alternative chain for an extended period, the attacker must effectively allocate their resources, as this can need a significant amount of resources, even in Proof of Stake (PoS) systems. The process also encompasses comprehending the network's consensus rules in order to exploit them efficiently, namely the rule that deems the longest chain as the genuine one. Attackers may employ advanced software solutions to automate the procedure of constructing and managing the alternate chain.

Exploited vulnerabilities: Exploiting the vulnerability of Costless Simulation, implementing a long-range attack in PoS blockchains involves taking advantage of the fact that generating and maintaining a fork is relatively inexpensive. The attack exploits the concept of Weak Subjectivity, which relies on the premise that the network's security is upheld by a majority of honest validators. The assailant establishes a covert alternative chain, progressively growing its length until it surpasses that of the primary chain. The minimal resource requirements of PoS facilitate the process of block formation. Once the network acknowledges and approves the attacker's chain, it has the ability to modify the historical records of the blockchain, which includes transactions and account balances. This form of attack is most potent in networks lacking safeguards against extensive reorganizations or methods to verify the legitimacy of lengthy chain extensions.

Potential Impacts: The potential implications of a long-range strike are substantial. They encompass the process of modifying the historical records of a blockchain, hence changing the account balances and rendering previously verified transactions null and void. Such occurrences can result in monetary setbacks for users and erode confidence in the integrity of the blockchain. The attack can also induce perplexity and ambiguity within the network, as members grapple with discerning the authentic chain. Over time, effective long-distance

attacks may discourage the acceptance and financial commitment to Proof of Stake (PoS) blockchains. The attack underscores the necessity of implementing strong security protocols in PoS systems to thwart deep chain reorganizations and guarantee the network's enduring stability and dependability.

Examples: In Figure 8, we observe a novel **security protocol, "Pikachu**" (Azouvi and Vukolić, 2022), that creates checkpoints of the state of the PoS deployed directly into Bitcoin's Proof-of-Work blockchain. This action leverages the security and decentralization of the Bitcoin Network by the use of its Schnorr signatures and the Taproot upgrade to create efficient, constant-size transactions. The protocol uses an asynchronous key pairing to indicate the validator configuration in the PoS blockchain and update its public keys if any changes occur. This key is updated on the Bitcoin blockchain, with each transaction consolidating the state of the PoS blockchain and changing the validator set. This way, the protocol ensures that all signatures for updating the blockchain state are aggregated into a single constant-size signature, thus maintaining efficiency.

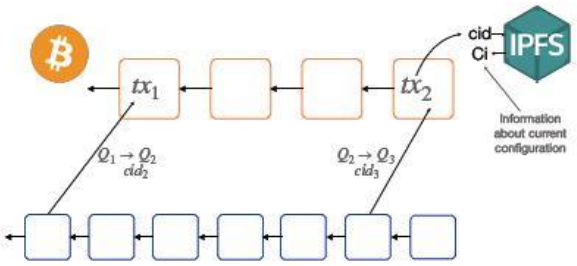


Figure 2: High-level visualization of the Pikachu protocol. Checkpoints from the PoS chain (in blue) are periodically pushed to the Bitcoin blockchain (in orange) by the PoS Validators. The checkpoints contain the Taproot address Q (which itself contains the aggregated public key of the configuration and commitment to the PoS chain ckpt) as well as a content identifier *cid* that can be used with any content-addressable storage to retrieve information about the configuration (IPFS pictured).

Figure 8: The Pikachu Protocol (Azouvi and Vukolić, 2022)

In the Cardano ecosystem, Ouroboros - Chronos (Badertscher et al., 2019a) protocol is responsible for permissionless clock synchronization within its ecosystem. Ouroboros-Chronos bolsters its defenses against potential long-range manipulations that rely on the divergence of the honest blockchain and their forks by ensuring accurate clock synchronization and stake distribution recognition. Additionally, Ouroboros–Praos (Badertscher et al., 2019) is the protocol that establishes the ecosystem's dynamic availability, a crucial component of a blockchain where node behaviour is often unpredictable. Ouroboros-Praos can adapt to the dynamic availability of participants, maintaining the ledger's properties

across multiple epochs under specified conditions. This way, the Ouroboros-Praos strengthens Cardano’s defenses against potential long-range alterations to the blockchain state.

Practical aspects: The practical aspects of long-range attacks in Proof of Stake (PoS) blockchains are intricate and pose substantial obstacles to network security. These attacks entail the creation of a fork from either the genesis block or another initial block, which then deviates substantially from the main chain. The possibility of carrying out such attacks is increased in PoS systems because of the 'Costless Simulation' problem, which allows for creating blocks on a fork with minimal resources, in contrast to Proof of Work (PoW) systems. This is reinforced by the 'Weak Subjectivity,' wherein the network's dependence on validators' subjective evaluation of the chain's history can be manipulated. Practically, malicious individuals can discreetly construct an alternative sequence of blocks over a prolonged duration and, after that, introduce it into the network, intending to supplant the existing sequence. This approach can be incredibly impactful if the perpetrator gains a substantial ownership share in the first stages of the blockchain's development. The effectiveness of the attack depends on persuading a significant proportion of the network to recognize the alternate chain as the authentic one. To address these threats, it is necessary to implement measures such as checkpointing and social consensus. Checkpointing involves agreeing upon specific past states of the blockchain that cannot be altered, establishing a solid foundation for the chain's history.

2.1.4 Stake Grinding Attacks

Description: In a Stake Grinding attack, a validator tries to manipulate the blockchain system in order to increase their chances of being chosen as the validator. This will affect the influence that validators have in the blockchain’s governance model and the economic incentives from minting the new block. This Attack can be done by predicting when the consensus model of a blockchain ecosystem will choose its validators so that the attacker can first use a flash loan attack to obtain all the tokens needed to guarantee his selection as a validator. This manipulation can lead to unfair advantages and compromise the blockchain's security. This type of Attack only works in blockchains other than Proof of Work as in PoW validators, and never chosen, they are rewarded for energy consumption.

Main concepts:(Kiayias and Panagiotakos, 2016) Stake Grinding attacks involve manipulating the process of selecting validators in Proof of Stake (PoS) and similar blockchain systems. Attackers strive to enhance their likelihood of being selected as validators, therefore

acquiring an imbalanced influence on the blockchain's governance and incentive systems. This is accomplished by artificially increasing their ownership in the network, typically through temporary methods such as flash loans. The attack subverts the ideas of equity and decentralisation in blockchain governance, as it enables assailants to acquire disproportionate authority and incentives. Stake Grinding differs fundamentally from attacks in PoW systems, as it utilises the unique mechanics of stake-based validator selection rather than processing power.

Step-by-step explanation: During a Stake Grinding attack, the perpetrator initially examines the consensus algorithm of the blockchain in order to comprehend the process of selecting validators. Subsequently, they make predictions regarding their likelihood of being selected as a validator, using this procedure. Prior to the selection process, the assailant obtains a substantial quantity of tokens, frequently through the utilization of a flash loan, in order to augment their ownership in the network. This increased stake enhances their likelihood of getting chosen as a validator. Upon selection, the assailant gains the ability to manipulate the governance of the blockchain and obtain greater pecuniary benefits from block creation. Upon the conclusion of the validation period, the assailant can relinquish the borrowed tokens, having cunningly exploited the system for their benefit. The attacker can repeatedly engage in this cycle, continuously exploiting the system's weaknesses for personal benefit.

Methods u**sed**: (Gazi et al., 2018)The tactics utilized in Stake Grinding assaults incorporate predictive analysis and financial manipulation. Adversaries scrutinize the validator selection algorithm of the blockchain in order to anticipate their potential selection. Subsequently, they employ financial mechanisms like flash loans to momentarily get a significant ownership interest in the network. Proficiency in both the technical intricacies of the blockchain and the financial systems that facilitate token acquisition is essential for this procedure. To optimize their impact while minimizing potential dangers and expenses, the assailant must strategically coordinate the timing of obtaining and returning tokens. This form of assault necessitates meticulous organization and implementation, merging advanced knowledge of blockchain technology with astute financial tactics.

Exploited vulnerabilities: Stake-grinding attacks happen by taking advantage of the known nature of the validator selection process in Proof of Stake (PoS) systems. The attacker aims to influence this procedure by deliberately augmenting their ownership interest. Typically, this is accomplished by temporarily obtaining tokens through financial processes such as flash loans. The attack takes advantage of the vulnerability of the PoS system, which relies on the assumption of a majority stake being honest. The attacker manages to bypass this assumption. By increasing their ownership percentage, the assailant acquires excessive

control over the network, which can result in partial decision-making and unjust financial benefits. To carry out this attack, one must possess expertise in the technical aspects of the blockchain system and have significant financial resources at their disposal.

Potential impacts: The potential effects of stakeholder grinding attacks are substantial. Attackers have the potential to consolidate power, which undermines the decentralised and democratic governance paradigm of the blockchain. The process of centralization might lead to partial validation of transactions and blocks, which may result in the possibility of censorship or favoritism. From an economic perspective, these attacks can potentially disrupt the fair allocation of rewards, granting attackers an unjust edge and diminishing the motivation of honest participants. Over time, these attacks can potentially undermine confidence in the blockchain system, which might result in decreased engagement and investment. The integrity and security of the blockchain have been compromised, which might have significant repercussions for its users and applications.

Examples: A mitigation methodology for this Attack is to use machine-learning algorithms to classify nodes, thus ensuring a fair and secure consensus process. (Sanda et al., 2023) Researchers of this mitigation methodology propose a novel dataset and a deep learning-based method to classify nodes in a Proof-of-Stake (PoS) blockchain as malicious or non-malicious, with demonstrated effectiveness. A secondary mitigation methodology is used in Cardano's Ouroboros. (Kiayias et al., 2019) As analyzed above, Ouroboros provides lackluster protection to the Cardano ecosystem through cryptographic randomness in Leader selection. This random process makes it difficult for a malicious attacker to predict and manipulate the exact timing of the validator selection, thus making this Attack currently challenging. Ouroboros also uses adaptive security, changing its behaviour and network conditions if and when potentially malicious actors control any significant portion of the total tokens staked. Finally, through its peer review methodology, the Cardano ecosystem remains vigilant at any time of new attack vectors and implements security features accordingly.

Practical aspects: Stake-grinding attacks in blockchain systems, specifically in those utilizing Proof of Stake (PoS) or its variations, pose a notable security risk since they can interrupt the process of selecting validators. During these assaults, a validator exploits the system by manipulating it to enhance its likelihood of being selected for block validation. This is typically achieved by accurately forecasting when they will likely be chosen and obtaining more tokens, perhaps using flash loans, to guarantee their selection. Such manipulation can result in an unjust advantage, distorting the blockchain's governance structure and economic incentives. Essentially, this kind of assault weakens the core idea of equal chance in PoS systems and potentially jeopardizes the integrity and security of the blockchain. To effectively combat Stake

Grinding assaults, it is necessary to implement solid and unpredictable methods for selecting validators and systems to identify and punish these manipulative behaviours.

2.2 Smart Contract Attack Vectors

These attacks typically have less impact than those analyzed above as they target protocols built with smart contracts and not their entire ecosystem infrastructure. Hence, they typically only have financial repercussions and damage the project and / or developers that created them. Since a blockchain is an immutable ledger, so are the smart contracts written on top of it. This irreversibility causes this type of logical programming errors to evolve into actual attacks with economic ramifications. Unfortunately, given the economic incentives and the lack of proper mitigation mechanisms, as we will analyze later, these attacks are the most used in blockchain.

2.2.1 Integer Overflow / Underflow Attack

Description: An integer Overflow or Underflow is an attack that targets intelligent contracts with no memory restriction into their variables. So, when an attacker attempts to store inside a value greater or smaller than maximum and / or minimum value it can hold, the variable behaves unexpectedly. In Ethereum the Integer Overflow / Underflow attacks are categorized into multiplication, addition, and subtraction overflows. (Duan et al., 2022) Ethereum had multiple instances of attacks occurring through this vector due to its flawed original design which is susceptible to it.

Main concepts: Integer Overflow/Underflow attacks exploit the limited storage capacity of variables in smart contracts. When a variable reaches its maximum storage capacity and an operation attempts to increase its value beyond this limit (overflow), or when it is at its minimum and an operation drops it further (underflow), the value loops back to the opposite extreme. Attackers can exploit this behaviour to influence the logic of smart contracts, frequently resulting in unauthorized activities or access. The attack is enabled by the absence of enough safeguards in the smart contract code to address such exceptional scenarios. These weaknesses are frequently disregarded throughout the construction of smart contracts, rendering them a prevalent target for attackers.

Step by step explanation: During an Integer Overflow/Underflow attack, the attacker initially selects a susceptible smart contract in which variables are not adequately checked for their range. Subsequently, they perform a transaction that initiates an arithmetic operation (such as addition, subtraction, or multiplication) on this variable. An overflow or underflow occurs when the outcome of an operation exceeds the maximum or minimum capacity of the variable. The unforeseen conduct can modify the state or logic of the contract in a manner that favors the

assailant, for instance, by artificially augmenting their balance. The assailant capitalizes on this modified condition to withdraw funds or obtain unauthorized entry, resulting in financial losses for other users or the proprietor of the contract.

Methods u**sed**:(Brent et al., 2018) In order to mitigate the risks of Integer Overflow/Underflow attacks, it is imperative for smart contract developers to utilize thorough testing and auditing procedures. This involves incorporating verification mechanisms into the contract code to validate arithmetic operations and guarantee that values stay inside the permissible range. Static analysis tools are capable of identifying probable instances of overflow or underflow vulnerabilities in the code. Automated testing frameworks provide the capability to mimic different inputs in order to verify that the contract functions as anticipated even in challenging circumstances. Moreover, the utilization of formal verification methods can logically demonstrate the accuracy of contract logic, hence reducing the likelihood of such assaults.

Exploited vulnerabilities: Attackers execute Integer Overflow/Underflow attacks by creating transactions that modify susceptible variables within smart contracts. They take advantage of the absence of boundary checks in the contract's arithmetic operations. Attackers can manipulate transaction inputs to surpass storage restrictions of variables, resulting in unforeseen behaviour. This can lead to modifying crucial contractual conditions, such as account balances or access limitations. The exploited vulnerabilities usually arise from insufficient input validation and error handling in the contract code.

Potential impacts: The potential effects of Integer Overflow/Underflow attacks are substantial. These vulnerabilities can result in significant financial losses for users and contract owners by allowing unauthorized transfers of funds or manipulation of assets. These assaults can potentially erode confidence in the impacted smart contract and, consequently, the broader blockchain platform. Furthermore, they have the potential to cause enduring harm to the developers and organizations implicated, affecting their reputation in the long run. Severe vulnerabilities can result in the total failure of the impacted smart contract, requiring it to be redeployed and even leading to legal ramifications.

Examples: EASYFLOW is a **mitigation tool** designed to detect and prevent these attacks. EASY FLOW operates at the EVM level and utilizes a taint analysis tracking technique to categorize safe, well-protected, and potentially vulnerable contracts. EASY FLOW can also generate transactions so it can trigger potential overflows, identifying vulnerabilities that may not be observed in regular contract execution.(Gao et al., 2018) Safemath is a **mathematical library** which acts as an effective countermeasure for this type of Attack. The library provides the user with transactions to trigger potential overflows, identifying vulnerabilities that may not

be evident in regular contract execution. Safemath is the most preferred mitigation tool for this Attack in the Ethereum ecosystem.(OpenZeppelin Docs, 2023).Another **novel mechanism** introduced by (Sun et al., 2022) was to specialized mutation operators aimed at detecting sufficiency in Ethereum Smart Contracts. It is worth noting that although this Attack is a well-known attack vector, an empirical study of 40 open-source smart contracts showed that all 179 known integer overflow vulnerabilities could be reproduced. A feature also able to mitigate this Attack is **SolType**. This feature solves scenarios where providing annotations for every arithmetic operation in a smart contract can be impractical. Additionally, SolType can extend its capability to handle complex nested data structures, thus allowing for the creation of sophisticated relationships between integer values within these contracts.(Tan et al., 2022) Lastly, this type of Attack was made impossible in Cardanodue to the EUTXO model (Chakravarty et al., 2020a). EUTXO incorporates additional data in transaction outputs—the datum—that allow the smart contracts to carry the contract-specific (local) states of information without changing the contract's code. By ensuring that transaction outputs carry both value and state information, alongside enforcing strict validation rules through both datum and context, the EUTXO model provides a structured and secure framework for smart contract execution.

Practical aspects: Integer Overflow/Underflow attacks exploit frequent coding oversights in smart contract development, focusing on the practical aspects. These attacks can be easily carried out on susceptible contracts, which makes them a favored option for attackers. To address these vulnerabilities, a mix of meticulous code development practices, comprehensive testing, and the utilization of automated analysis techniques is necessary. The consequences of these attacks can be significant, resulting in substantial financial losses and a decrease in trust within the blockchain ecosystem. Hence, it is crucial to have a thorough understanding and take proactive steps in the development and implementation of smart contracts to mitigate these risks.

2.2.2 Transaction Order Dependence, MEV Attack

Description: Transaction Order Dependence, also referred to as Miner Extractable Value (MEV), is a phenomenon observed in blockchain ecosystems like as Ethereum. It involves miners or validators gaining extra value by rearranging the order of transactions. This approach is strategically rearranging the sequence of transactions within a block to optimize profits, beyond the conventional block rewards and gas fees. MEV can result in front-running, when miners prioritize their transactions over others, or sandwich attacks, where user transactions are purposefully placed between others to get a financial advantage. The transparent nature of pending transactions in the blockchain's mempool enables the extraction

of MEV. The unique and intricate nature of MEV in blockchain ecosystems necessitates a thorough comprehension and the implementation of solutions to either capitalize on or alleviate its effects.

Main concepts: Miners or validators, who own the power to determine the sequence of transactions in a block, might strategically influence this sequence to benefit themselves. This manipulation can manifest in different ways, such as strategically inserting their transactions to exploit price fluctuations or rearranging user transactions to capitalize on arbitrage opportunities. The notion relies on the transparency and predictability of transactions in the mempool, enabling miners to foresee and capitalize on lucrative chances. MEV expresses apprehensions regarding equity and safety within blockchain ecosystems, since it has the potential to result in biassed treatment and potential manipulation of transaction results.

Step by step explanation: (Zhou et al., 2020)During a typical Miner-Extractable Value (MEV) scenario, a miner or validator initially watches the mempool for transactions that have the potential to generate reward. They recognize potential opportunities, such as significant transactions on decentralised exchanges, which have the potential to impact the pricing of assets. Subsequently, the miner proceeds to create a block, deliberately arranging transactions to maximize the benefits derived from these opportunities. For example, they may prioritize their transaction before a significant trade (front-running) or strategically position a user's transaction between their own trades to take advantage of price fluctuations. The miner concludes the block by arranging the transactions in a customized order and publicly adds it to the blockchain, thereby gaining a financial advantage from the manipulated order while normal users bear the cost.

Methods u**sed**: (Qin et al., 2021)The methodologies used to take advantage of MEV require advanced monitoring and analysis of the blockchain mempool in order to uncover lucrative possibilities for rearranging transactions. Automated bots and algorithms are frequently employed to identify and carry out these possibilities instantaneously. Miners can also cooperate with traders or operate their own trading bots to optimize the extraction of maximum MEV. Regarding mitigation, techniques involve implementing protocols that introduce randomization in transaction ordering or conceal transaction data until they are mined. In addition, DeFi platforms are investigating the use of time-lock measures and enhanced trade execution algorithms to mitigate vulnerabilities related to MEV.

Exploited vulnerabilities: Miners or validators who possess authority over the arrangement of transactions within a block carry out MEV attacks. They take use of the visibility of pending transactions in the mempool and the predictable nature of blockchain consensus procedures. Through the examination of the mempool, assailants can anticipate the influence of specific transactions on the value of assets or other states within the blockchain. Subsequently, they strategically create blocks to capitalize on this information, either by executing their transactions in advance of others or by altering the order of transactions to generate advantageous market conditions. This exploitation results in unjust benefits and the possibility of market manipulation.

Potential impacts: The possible effects of MEV are substantial, encompassing diminished equity and confidence in blockchain ecosystems and financial setbacks for ordinary users. MEV can result in market manipulation, wherein miners or validators unjustly gain profits to the detriment of other participants. Such manipulation can potentially alter market dynamics, resulting in inefficient and unforeseen consequences. Over time, MEV can potentially compromise the reliability of blockchain systems, discouraging users from getting involved and making investments. In addition, the pursuit of Maximum Extractable Value (MEV) might incentivize miners or validators to centralize, undermining the decentralised nature of blockchain networks.

Examples: MEV occurs in DeFI projects when no proper protective measures are taken from either dAPPs or the user themselves and because in Ethereum, the order of transactions significantly affects the final state of the blockchain and, by extension, the distribution of tokens among its participants (Figure 9).(Daian et al., 2020) A **security simulation toolbox** is introduced in “Simulation of Front-Running Attacks and Privacy Mitigations in Ethereum Blockchain” (Stucke et al., 2022). In this simulation software it is possible to review an MEV attack and their mitigations using the MEV-geth protocol. Flashbots Auction is the only currently known working countermeasure for this type of leeching Attack in the Ethereum network. Flashbots operate a dark pool-like system which seeks to provide anonymity, privacy and reduce MEV costs. In the Cardano ecosystem, Ouroboros – Proas addresses these concerns and minimizes the risk of this Attack through adaptively-secure, semi-synchronous proof-of-stake. (Badertscher et al., 2019b)

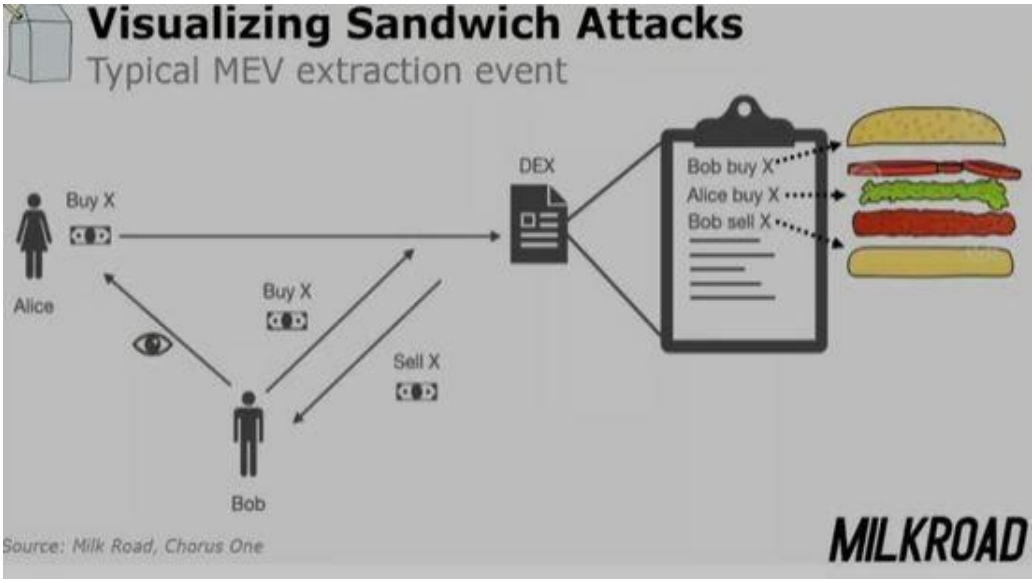


Figure 9: Mev Sandwich Attack ("Ethereum Proposer-Builder Separation: Past, Present, and Future," 2023)

Practical aspects: (Daian et al., 2020)MEV poses practical difficulties within the Ethereum blockchain ecosystem. MEV is derived from the capacity of miners or validators to rearrange transactions within a block in order to optimize their earnings, surpassing the conventional block rewards and gas costs. These behaviours can result in front-running when miners strategically prioritize their transactions over others to obtain an advantage or sandwich attacks, where the order of user transactions is manipulated to benefit the miner. The practical consequences encompass diminished equity and clarity in transaction processing and an elevated susceptibility to market manipulation in decentralised finance (DeFi) applications. To tackle MEV, it is necessary to implement advanced procedures that guarantee fairness in the ordering of transactions. This can be achieved by implementing protocols that randomize the sequence of transactions or hide their information until they are confirmed.

2.2.3 Gas exhaustion & Blockchain DDOS

Description: Gas is a fundamental mechanism for all blockchain ecosystems, designed to measure and limit the consumption of computational resources during transactions. In several ecosystems, an exhaustion attack is possible by exploiting preexisting block gas limitations set by the blockchain consensus mechanism. These attacks are done on a smart contract level but, if done correctly, can halt a blockchain's block creation. DETER (DDOS Attack on Ethereum clients) can be done with minimal to no cost and turn off essential services like mining and transaction propagation. Furthermore, evidence shows these attacks can be enlarged enough to target vital centralized services like mining pools and transaction relay services. (Li et al., 2021)

Main concepts: (Chen et al., 2020)The primary principle underlying gas exhaustion attacks is to intentionally generate transactions or smart contract interactions that utilize the maximum permissible quantity of gas within a block. By doing this, attackers can hinder the inclusion of other transactions in the block by exploiting the gas limit. This attack leverages the system responsible for measuring and restricting the usage of computational resources in blockchain transactions. The assault primarily focuses on impairing the network's capacity to execute transactions effectively, perhaps resulting in denial of service. It exploits the inherent constraints in the blockchain's architecture about the allocation and restriction of computational resources.

Step by step explanation: During a gas exhaustion attack, the assailant initiates the process by developing a smart contract specifically engineered to use a substantial quantity of gas. Subsequently, they commence transactions or function calls to this contract with the intention of consuming the entire gas limit allocated to a block. As these transactions consume the block's gas limit, additional legal transactions cannot be executed, causing to a backlog in the network. This can significantly impede or decelerate the block building process, as miners are unable to incorporate supplementary transactions into the block. The attack persists as long as the assailant is prepared to cover the cost of the fuel needed to carry out these resource-intensive transactions, thus maintaining the disruption of the network.

Methods u**sed**: (Brent et al., 2018)In order to mitigate gas exhaustion attacks, blockchain networks must use approaches that systematically monitor and limit gas consumption. This involves establishing flexible petrol limitations that adapt according to network congestion and transaction intricacy. Sophisticated surveillance systems can be utilized to identify unusual surges in petrol consumption that may indicate an assault. Smart contract developers should additionally incorporate gas-efficient code and establish sensible gas restrictions for the execution of contracts. Furthermore, network enhancements can be introduced to optimize the processing of transactions and reduce the impact of such assaults.

Exploited vulnerabilities: Gas exhaustion attacks are executed by taking advantage of the predetermined gas limit per block in blockchain networks. Adversaries generate smart contracts or transactions with the intention of consuming a significant quantity of petrol. Attackers can obstruct the inclusion of other users' transactions by repeatedly performing these gas-intensive tasks, resulting in the block being filled with their own transactions. Such attacks exploit the vulnerability in the blockchain's resource allocation scheme, where malevolent actors can monopolize the set limit on computational resources each block. The attack is made easier by the relatively inexpensive nature of initiating high-gas transactions, considering the significant influence they can have on the network.

Potential impacts: Gas exhaustion attacks are executed by taking advantage of the predetermined gas limit per block in blockchain networks. Attackers create smart contracts or transactions that purposely consume a huge quantity of gas. Attackers can disrupt the inclusion of other users' transactions by repeatedly executing high-gas-consuming activities, which fill up blocks with their own transactions. This attack leverages the weakness in the blockchain's resource allocation scheme, where bad actors can seize control of the set limit on computational resources each block. The attack is made easier by the relatively inexpensive nature of initiating high-gas transactions, considering the significant influence they can have on the network.

Examples: **Lazy contracts** is a protocol designed to reduce the high gas costs associated with the on-chain execution of smart contracts in the Ethereum blockchain. The protocol achieves this cost reduction by moving most computations off-chain, ensuring minimal on-chain gas usage.(Farokhnia, 2023) For Ethereum, an additional countermeasure is **the V-GAS system**. (Ma et al., 2019) V-GAS is designed to handle the well-known out-of-gas error in Ethereum’s smart contracts. V-Gas deploys a feedback-directed mutational fuzz testing system, which builds a gas-weighted control flow graph (CFG) for a smart contract’s functions. Subsequently, it uses gas consumption-guided selection and mutation strategies to generate inputs that maximize gas consumption, reducing the underestimation cases. V-GAS was able to identify up to 25 new out-of-gas vulnerabilities, receiving in the process 5 CVEs from the US National Vulnerability Database, while also in a real world scenario was able to improve gas consumption by 44.02%.

Practical aspects: Gas exhaustion attacks in blockchain ecosystems, such as Ethereum, aim to disrupt the core function of gas, which is utilized to quantify and restrict the utilization of computing resources in transactions. These attacks manipulate the block gas restrictions established by the blockchain's consensus mechanism, with the goal of exhausting the gas supply in a block by executing intricate or many transactions. The practical consequences of such attacks encompass the ability to cease the creation of blocks, interrupt mining activities, and hinder the propagation of transactions. Gas depletion can be achieved with minimal expenditure, but it can have a substantial influence, especially on centralized services such as mining pools and transaction relay networks. To tackle these risks, it is necessary to provide strong network monitoring, optimize petrol pricing algorithms, and establish measures to avoid malicious transaction flooding.

2.3 Decentralized Finance (DeFI) Attack Vectors

In these types of attacks no system is compromised but many financial on a personal level occur.

2.3.1 Flash Loans attacks

Description: Flash loans attacks rely on obtaining a temporary high liquidity from what are called “flash loans” in order to manipulate the price of a cryptocurrency, exploit other vulnerabilities (such the as the re-entrancy) and / or steal funds from Decentralized Applications. (Palamarchuk Roman, 2023) Flash loans attacks as shown in Figure 10 need to be execute these three steps: borrow, manipulation, repayment within the same transaction. Hence the title Flash.

Main concepts: Flash loan attacks involve exploiting DeFi protocols by utilizing substantial, uncollateralized loans that be acquired and repaid inside a single transaction. These loans are utilized by attackers to manipulate market conditions, such as artificially inflating asset values or causing liquidity shortages. This manipulation allows companies to generate profits by exploiting opportunities such as arbitrage, price manipulation, or by exploiting flaws such as re-entrancy in smart contracts. The attack exploits the distinctive characteristic of flash loans, which is their absence of collateral and immediate settlement, to carry out intricate financial maneuvers that would normally necessitate significant resources.

Step by step explanation: During a flash loan physical assault, the assailant initially selects a specific DeFi protocol as their target and identifies a vulnerability that they may manipulate. Subsequently, they acquire a flash loan, a financial instrument that grants them a substantial sum of funds without requiring any initial security. With this capital, they exert control over the market or the target protocol, for instance, by deliberately increasing the price of a cryptocurrency. Upon reaching the targeted market condition, the assailant promptly carries out trades or other operations in order to actualize a profit. Ultimately, the flash loan is repaid in the same transaction, thereby concluding the assault cycle.

Methods u**sed**: The carrying out of flash loan attacks involves the utilization of smart contract programming, market intelligence, and precise timing. Adversaries frequently employ automated scripts or bots to surveil DeFi platforms for weaknesses and advantageous market situations. In order to guarantee loan repayment, it is imperative to accurately compute the necessary capital and possible earnings. The attack transaction is often intricate, encompassing many DeFi platforms and financial instruments. In addition, attackers must take

into account fuel fees and transaction timing to ensure the successful execution of the attack within a single block.

Exploited vulnerabilities: Flash loan attacks are carried out through the creation and execution of smart contracts that engage with DeFi protocols that provide flash lending services. Assailants take use of weaknesses, such as the manipulation of pricing oracles, the occurrence of re-entrancy in smart contracts, or problems related to liquidity in decentralised exchanges. The attack contract is specifically crafted to carry out a sequence of actions, including obtaining cash, influencing the market or protocol, and then benefiting from this manipulation, all within a single transaction block, without repaying the loan.

Potential impacts: Flash loan attacks have the potential to cause significant losses in money for both DeFi systems and its users. These attacks have the potential to erode the reliability and faith in the DeFi ecosystem, resulting in market volatility and a decline in investor trust. Furthermore, they reveal inherent weaknesses in DeFi protocols, necessitating immediate evaluations and enhancements of smart contract code and security mechanisms. Flash loan attacks underscore the hazards linked to intricate financial instruments and swift advancements in the DeFi sector, within a wider framework.

Examples: Prominent instances of flash loan attacks encompass the bZx protocol attack that occurred in February 2020. In this incident, malevolent actors capitalized on price oracles and loan mechanisms to drain funds. An other instance occurred in October 2020, known as the Harvest Finance attack, which led to the significant financial loss of millions of dollars due to the manipulation of prices. In November 2020, the Value DeFi protocol experienced a flash loan attack, which exposed weaknesses in its vaults and governance procedures. During **Borrow the Attacker** accesses high liquidity from multiple reserves in a single flashLoan transaction. During manipulation the borrowed funds are used from the attacker to manipulate the price of its victim token within a DeFI smart contract. Lastly, during the repaying step the attacker repays the flash loan.(AAVE Developer Team, 2023) This is not a vulnerability that can be exploited on its own but rather a well known attack after an initial vulnerability has been found which would allow for the manipulation of a Decentralized Application protocol first. Cardano ecosystem is protected from these types of attacks because of its EUTXO model. Each transaction in the EUTXO model is treated as a separate object and is entirely independent of its surroundings (local). Without a global state of contracts, the flash loan can’t occur because it can’t create 2 different actions in the same transaction.(Chakravarty et al., 2020a)

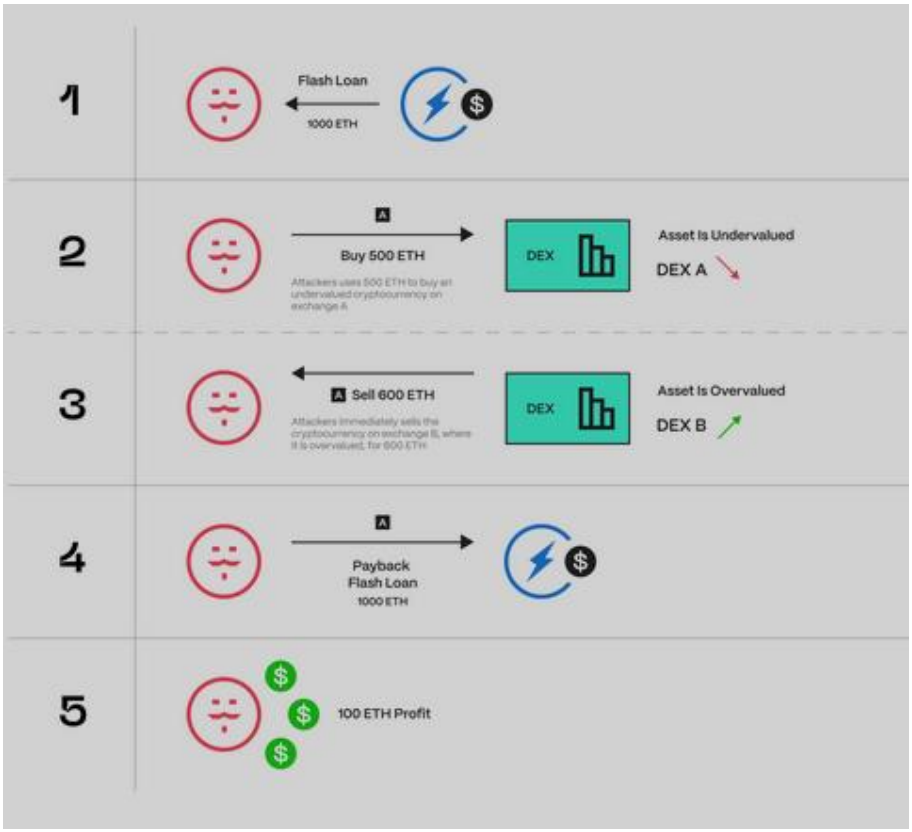


Figure 10: Example of a Simple Flash Loan

Practical aspects:(Qin et al., 2020) Flash loan attacks are executed within the DeFi ecosystem, utilizing the distinctive characteristics of flash loans to address practical concerns. These assaults necessitate a high level of comprehension and precise execution of intelligent contract interactions across various DeFi platforms. They emphasize the necessity for enhanced security measures, such as improved oracle designs and more rigorous intelligent contract auditing. Flash loan attacks also stimulate debates over ethical and legal considerations in decentralised finance (DeFi). Essentially, they function as a reminder for DeFi platforms to improve their security protocols and for users to be mindful of the inherent dangers associated with these nascent financial technologies.

2.3.2 Liquidity Pool Attack

Description: A Just-In-Time (JIT) attack is strategic flash manipulation of liquidity tokens in order to extract value from the arbitrage that will occur when the pool is trying to self-balance its liquidity.(Wan Xin and Adams Austin, 2023)

Main concepts: JIT attacks mostly involve exploiting the automated liquidity balancing methods of DeFi pools. Attackers strategically manipulate the liquidity by either infusing a substantial quantity of tokens or withdrawing them at precisely the right moment to generate

advantageous arbitrage situations. The manipulation results in transient price disparities inside the pool, which the attackers take advantage of by engaging in opportunistic buying and selling to maximize their profits. The attack exploits the anticipated reaction of AMM protocols to alterations in liquidity, enabling the attacker to gain profit from the resulting price slippage. JIT attacks refer to a type of market manipulation that takes advantage of the intrinsic structure of liquidity pools on DeFi platforms.

Step-by-step explanation: (Daian et al., 2020)During a Just-In-Time (JIT) attack, the assailant initially identifies a DeFi pool that possesses a substantial amount of liquidity. Subsequently, they either acquire a substantial flash loan or utilize their personal cash to carry out a substantial transaction that dramatically modifies the liquidity balance of the pool. The abrupt shift in liquidity triggers the pool's automatic systems to modify pricing, resulting in the emergence of an arbitrage opportunity. The assailant promptly seizes this chance, carrying out transactions that exploit the momentary disparity in prices. Ultimately, the assailant concludes their position, typically reimbursing the flash loan if utilized, and ensuring a profit on the arbitrage.

Methods u**sed**: The execution of JIT attacks requires meticulous investigation of DeFi liquidity pools and a comprehensive comprehension of their AMM protocols. Attackers employ various tools and scripts to monitor pools for ideal circumstances, such as notable price discrepancies or instances of low liquidity. Flash loans might be utilized to get the required funds for the assault without the need for initial collateral. Optimal implementation of the assault necessitates meticulous timing in order to fully exploit the arbitrage opportunity resulting from the manipulation of liquidity. After an attack, the approach comprises promptly leaving the location to achieve gains and reduce risk.

Exploited vulnerabilities: JIT attacks are carried out by conducting substantial transactions that cause significant changes in the liquidity of a DeFi pool. These transactions take advantage of the automatic methods of automatic Market Makers (AMMs), which are created to ensure a balanced liquidity but can be susceptible to significant and abrupt fluctuations. The attack commonly entails purchasing a significant quantity of a token to artificially increase its price, followed by selling it at the inflated price, or vice versa. This method capitalizes on the anticipated behaviour of AMM protocols when faced with fluctuations in liquidity, enabling the attacker to generate and take advantage of arbitrage opportunities.

Potential impacts: JIT attacks have the ability to cause financial losses for other members in the DeFi pool. This is because the manipulated pricing may compel them to purchase at high prices and sell at low prices. These assaults have the potential to erode confidence in DeFi

platforms and associated liquidity mechanisms, resulting in decreased engagement and liquidity in the impacted pools. JIT attacks also expose inherent weaknesses in AMM protocols, requiring thorough evaluations and enhancements in their structure. Within a wider framework, these attacks can result in regulatory examination and demands for improved security measures in the DeFi ecosystem.

Examples: In order for the Attack to work, as shown in Figure 11, 3 steps are required: find a potential target, initiate a large trade, simultaneously add a really large amount of the token you are withdrawing to the Liquidity pool, immediately after the large transfer occurs remove your remaining fund from the pool. Hence draining the pool with minimal risk exposure. (Wan Xin and Adams Austin, 2023) A Just-In-Time (JIT) attack is not possible to occur in Cardano as the EUTXO model of Cardano will not allow for transactions to occur within the same state.(Chakravarty et al., 2020a) To our knowledge no detection tool and or mitigation plan exists yet.(Zhou et al., 2022)

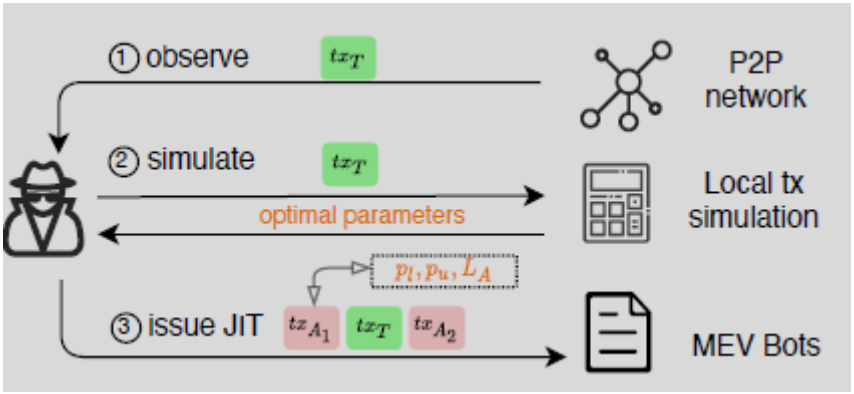


Figure 11: Overview of a JIT Liquidity Hack (Wan Xin and Adams Austin, 2023)

Practical aspects: JIT attacks in DeFi pools include exploiting the liquidity dynamics for practical purposes. These attacks necessitate a significant amount of capital, which can be either held or obtained through methods such as flash loans. Proficiency in AMM procedures and the capacity to anticipate and exploit market responses to significant liquidity fluctuations are essential. Efficient execution of JIT attacks necessitates precise scheduling and swift implementation in order to optimise revenues. Essentially, these attacks emphasise the necessity for enhanced liquidity management and market monitoring systems in DeFi platforms to deter such manipulative activities.

Chapter 3 Technical Analysis of the Re-entrancy Attack

In this chapter of the Thesis, we will analyze this attack in depth, reviewing code examples and showcasing etherscan results of actual attacks. Finally, we will be discussing potential countermeasures for Ethereum's and Cardano's smart contracts.

3.1 Definition of the Re-entrancy Attack

Re-entracy is considered one of the most prominent, severe, and effective attacks in smart contracts. Re-entracy of contracts happens when a contract calls a second external contract which re-calls the initial contract. All these actions take place in a single transaction. This leads to unexpected contract states which allow the attacker to call the initial contract multiple times in a recursive manner without the first invocation of the contract even to occur. It is worth nothing that legitimate re-entracy often happens during contract executions as it is a common part of Ethereum's programming language Solidity as we see in Figure 5. The ability to re-entry unexpectedly is what causes these actions to become malicious in nature.(Rodler et al., 2018). Given the immutable nature of the smart contracts these vulnerabilities that later become attacks can be utilized permanently once they are known.(Chinen et al., 2020)

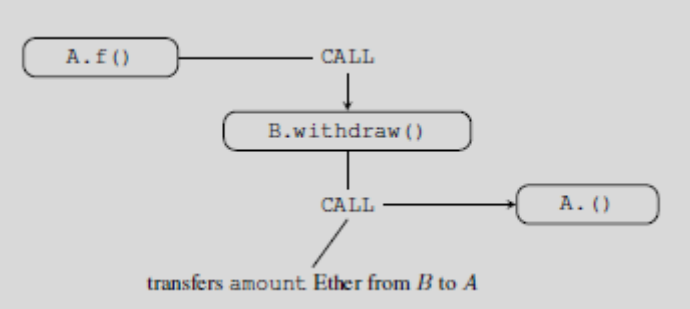


Figure 12: Common Withdrawal Pattern(Rodler et al., 2018)

In order to showcase the malicious attack, we will be naming a contract as "Victim" and the secondary contract as the "Attacker". In Figure 6 we observe a contract Victim that suffers from the re-entracy vulnerability, since it tracks the amount of token it possesses after a transfer has occurred and has the "withdraw" function enabled. (Rodler et al., 2018) For the attacker to act these are the two main conditions that need to co-exist within a contract's invocation.

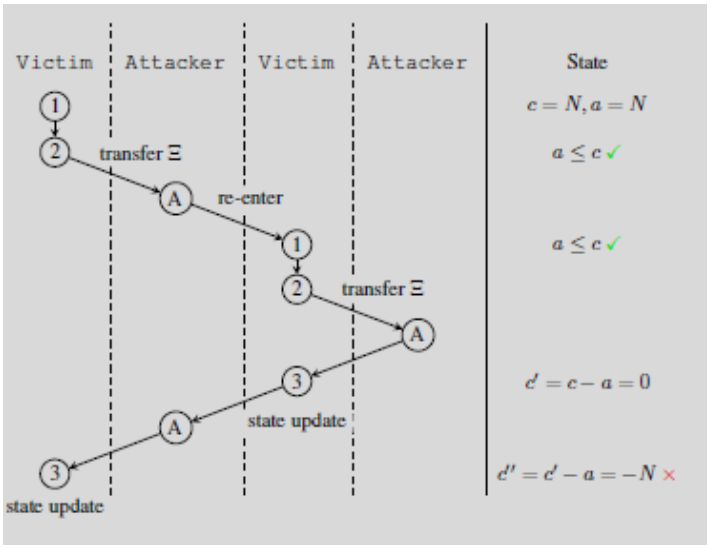


Figure 13: Contract vulnerable to re-entrancy attacks(Rodler et al., 2018)

The attackers' steps are:

1. Send to the Victim any amount of native tokens using the Deposit function.
2. Observe that the "update" function is called after the native tokens were received.
3. Send again any amount of native tokens using the Deposit function.
4. While performing step 3, request the withdraw function from the Victim pointing to the attacker's address (Figure 7).
5. Repeat step 4 multiple times until the Victim is completely drained of its native tokens. (kamilpolak, 2022)

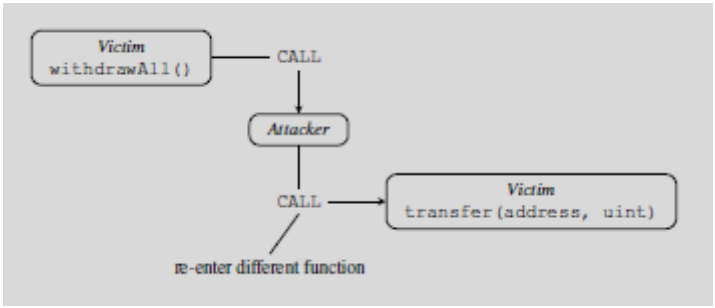


Figure 14: Call the Withdraw function instead of Deposit during the Re-entracy Attack. (Rodler et al., 2018)

3.2 Smart contracts of a re-entrancy Attack

In Figure 12 we can view a contract that is vulnerable to re-entracy attack. The contract "DepositFunds" does initialize both the "deposit" and the "withdraw" functions while also calling refreshing the balance after these calls have been made.

In Figure 13 we can view the Attacker's contract. This contract calls for the deposit function of its Victim, afterwards the Attacker calls the withdraw function, when a token is indeed received from the Victim the fallback function is used to re-call the withdraw function. Since the contract has not yet checked internally its balance, the token is send to the Attacker, which creates a loop calling the fallback function again. This continues until the balance of the Victim is 0.

Lastly, in Figure 14 we can see that the prevention of this fallback attack is as easy as ensuring through the contract that all state changes happen before calling any additional external code.

```
contract DepositFunds {
    mapping(address => uint) public balances;

    function deposit() public payable {
        balances[msg.sender] += msg.value;
    }

    function withdraw() public {
        uint bal = balances[msg.sender];
        require(bal > 0);

        (bool sent, ) = msg.sender.call{value: bal}("");
        require(sent, "Failed to send Ether");

        balances[msg.sender] = 0;
    }


}
```

Figure 15: The Victim's source code in solidity

```
contract Attack {
    DepositFunds public depositFunds;

    constructor(address _depositFundsAddress) {
        depositFunds = DepositFunds(_depositFundsAddress);
    }

    // Fallback is called when DepositFunds sends Ether to this contract.
    fallback() external payable {
```

```
        if (address(depositFunds).balance >= 1 ether) {
            depositFunds.withdraw();
        }
    }

    function attack() external payable {
        require(msg.value >= 1 ether);
        depositFunds.deposit{value: 1 ether}();
        depositFunds.withdraw();
    }


}
```

Figure 16: The Attacker's source code in Remix code in solidity

```
contract ReEntrancyGuard {
    bool internal locked;

    modifier noReentrant() {
        require(!locked, "No re-entrancy");
        locked = true;
        _;
        locked = false;
    }
}
```

Figure 17: Prevention of the Re-entracy Attack in solidity

3.3 Known Attacks

In the table below, we can see the impact of this trivial yet highly effective attack in smart contracts and how it persists throughout the years. We will be analyzing each of them separately to understand how to explain this attacks occurred.

Contracts Attacked	Funds stolen	Year
DAO Attack	$60 million	2016
Lendf.ME	$25 million	2020
BurgerSwap	$7.2 million	2021
XSurge	$4 million	2021
Siren Protocol	$3.5 million	2021
Orion Protocol	$3 million	2023
dForce	$3.7 million	2023
Sturdy Finance	$0.6 million	2023
Stars Arena	$3 million	2023

3.3.1 The DAO Attack

On April of 2016 a group of programmers using the Ethereum Blockchain launched a crowd-funding effort for a project known as "**The DAO**". Unfortunately, this fund raising method would prove to alter the course of Ethereum and create multiple discussions between developers, investors and entrepreneurs.(Shier et al., 2017) The DAO was created contained the now well-known re-entrancy bug produced by the programming logical errors of these programmers. The mission of the DAO, as all Autonomous Organizations strive to be, was to become a self-directed venture capital fund with its contributors able to vote its future projects and aspects. (Dupont, 2017)This way of crowd-funding through the ICO would eventually raise $168 million USD worth of Ether, making it the most successful crowd funding of its era. After the crowd funding was finished the unknown yet attacker used a mechanism intended to create "child DAO's" to syphon to his child DAO's almost 40% of the total amount the DAO possessed in Ether. Because these child DAO's were a copy of the original DAO, it was hardcoded to them the lockout period of 28 days (as this was the original time window for the initial ICO). As the DAO was by far the largest project in Ethereum at that time any action taken or non-taken would have large repercussions. Ethereum was 10 months old project at that time.

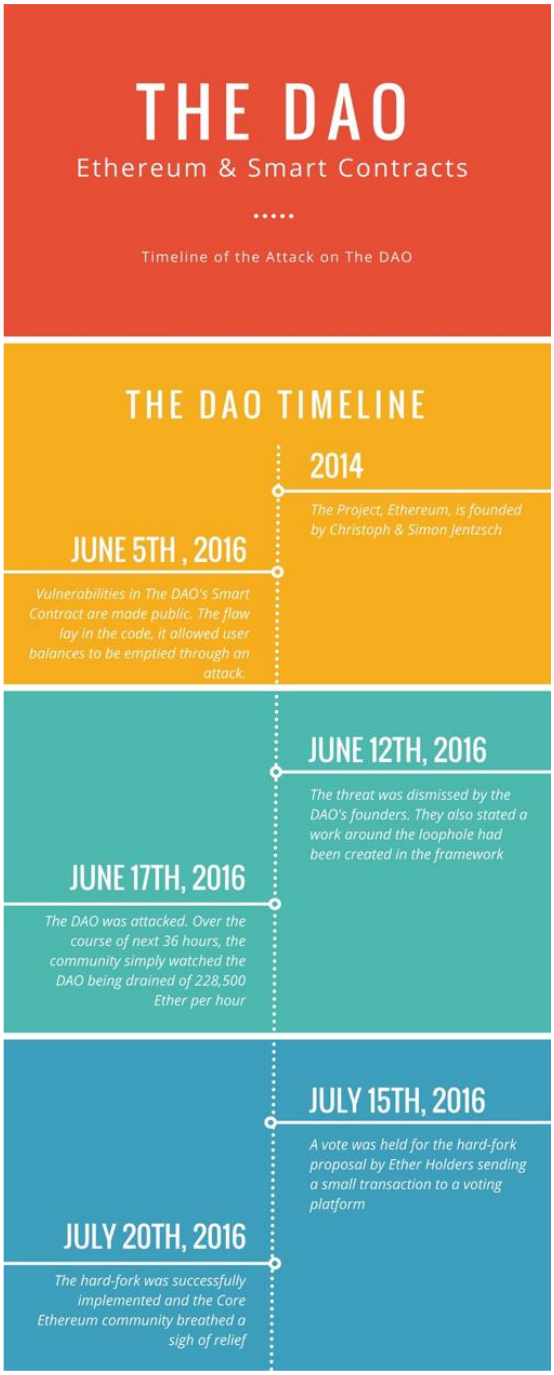


Figure 18: Timeline of the DAO Attack (Shier et al., 2017)

Some attempts were made to stop the cryptocurrency from being taken, but the required consensus of votes could not be obtained from the collective in such a short time. (Morrison et al., 2020) After careful consideration the **Ethereum Foundation proposed 3 options**:

1. **Do nothing and allow the hacker** to use the stolen fund after the 28day lock-up period
2. **Build a blacklist** into the Ethereum code in order to freeze these specific funds (soft fork option)

3. **Unwind the hack entirely** returning all stolen Ether to the DAO and its investors.

If the Foundation decided to do nothing, they could face legal battle against the investors of the DAO. If the Foundation created a hard fork, that would shackle the trust of the decentralized nature of Ethereum. (Okereke Innocent, 2021)This fight created the well-known catchphrase “Code is Law” from the users that sided with the Attacker since this was not a malicious action from a user but a use of a valid, legal function existing within the smart contract. In the end, the Foundation performed a hard fork of their blockchain and the funds were returned to the DAO investors. Those who disagreed with the decision, created another hard fork of the Ethereum blockchain and named it Ethereum Classic (ETC). Although the Attack was erased from the blockchain the debate and more importantly the evidence of this DAO still exist within the Ethereum blockchain and can be seen in this Ethereum Address: 0x15DEF77337168d707E47E68aB9f7F6c17126b562. In Figure 16 we see a screenshot of Ethereum’s Blockchain Explorer where it is clearly shown that the transactions of the DAO.

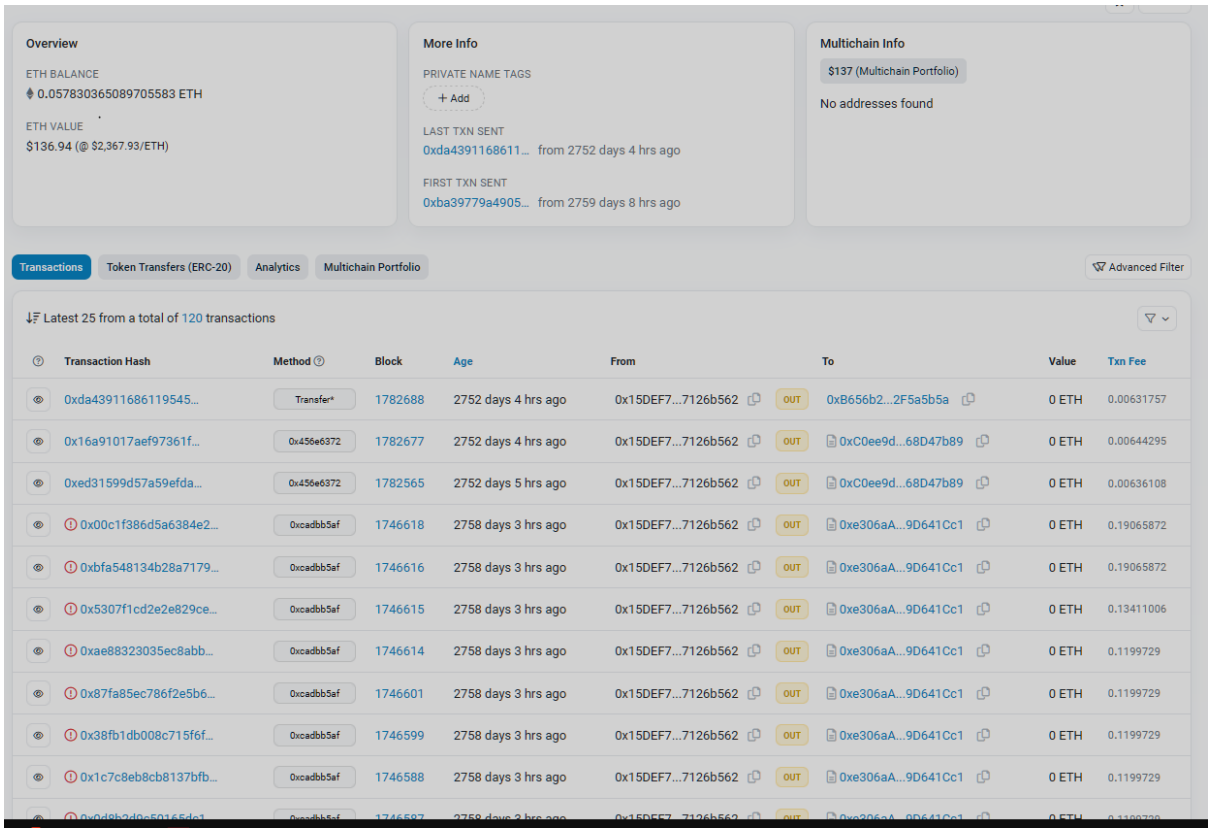
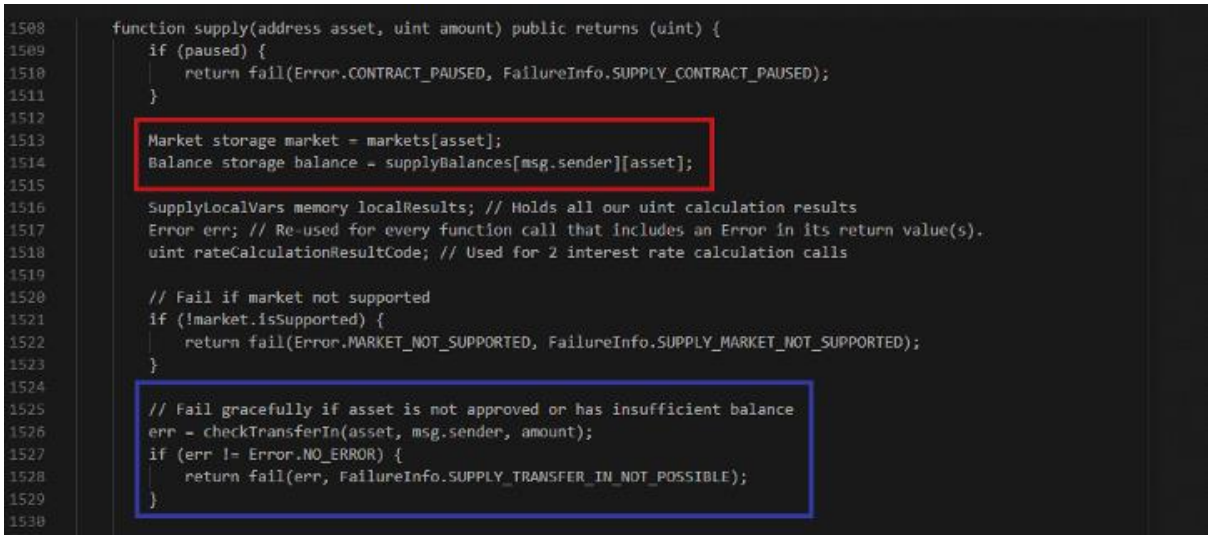


Figure 19: DAO’s Etherscan Tx’s

3.3.2 The Lendf.me Protocol

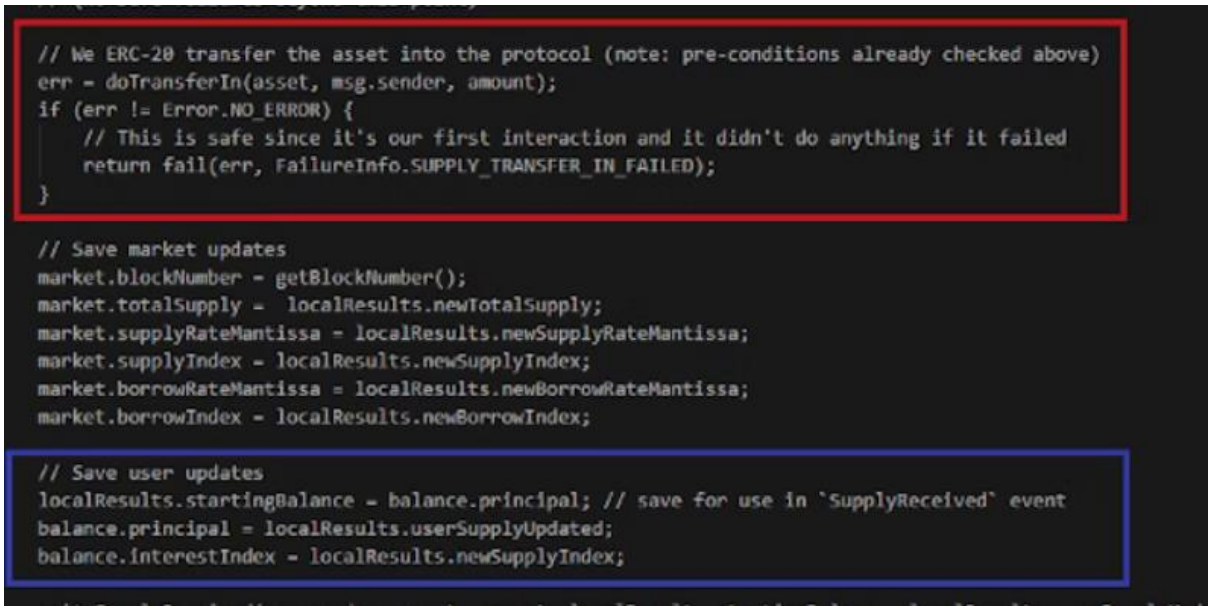
Lendf.me was a DeFI lending dAPP that was attacked on April 19,2020 and caused a loss of $25 million. Lendf.me was attacked through a smart contract named “MoneyMarket” that utilized the core logic of the platform.(Valid.Network, 2020) In Figures 17,18 and 19 we can

observe the logic of this smart contract. The logical bug of the contract is that the *MoneyMarket.supply() function* is updating the user's asset balance after the external call to asset.transferFrom() (lines 1599–1600), but based on a value that was read before the external call (line 1514), ignoring any updates that were made from that external call.



```
1508    function supply(address asset, uint amount) public returns (uint) {
1509        if (paused) {
1510            return fail(Error.CONTRACT_PAUSED, FailureInfo.SUPPLY_CONTRACT_PAUSED);
1511        }
1512
1513        Market storage market = markets[asset];
1514        Balance storage balance = supplyBalances[msg.sender][asset];
1515
1516        SupplyLocalVars memory localResults; // Holds all our uint calculation results
1517        Error err; // Re-used for every function call that includes an Error in its return value(s).
1518        uint rateCalculationResultCode; // Used for 2 interest rate calculation calls
1519
1520        // Fail if market not supported
1521        if (!market.isSupported) {
1522            return fail(Error.MARKET_NOT_SUPPORTED, FailureInfo.SUPPLY_MARKET_NOT_SUPPORTED);
1523        }
1524
1525        // Fail gracefully if asset is not approved or has insufficient balance
1526        err = checkTransferIn(asset, msg.sender, amount);
1527        if (err != Error.NO_ERROR) {
1528            return fail(err, FailureInfo.SUPPLY_TRANSFER_IN_NOT_POSSIBLE);
1529        }
1530
```

Figure 20: The MoneyMarket code 1/3 (Valid.Network, 2020)



```
// We ERC-20 transfer the asset into the protocol (note: pre-conditions already checked above)
err = doTransferIn(asset, msg.sender, amount);
if (err != Error.NO_ERROR) {
    // This is safe since it's our first interaction and it didn't do anything if it failed
    return fail(err, FailureInfo.SUPPLY_TRANSFER_IN_FAILED);
}

// Save market updates
market.blockNumber = getBlockNumber();
market.totalSupply =  localResults.newTotalSupply;
market.supplyRateMantissa = localResults.newSupplyRateMantissa;
market.supplyIndex = localResults.newSupplyIndex;
market.borrowRateMantissa = localResults.newBorrowRateMantissa;
market.borrowIndex = localResults.newBorrowIndex;

// Save user updates
localResults.startingBalance = balance.principal; // save for use in `SupplyReceived` event
balance.principal = localResults.userSupplyUpdated;
balance.interestIndex = localResults.newSupplyIndex;
```

Figure 21: The MoneyMarket code 2/3(Valid.Network, 2020)

```
404
405        token.transferFrom(from, address(this), amount);
406
```

Figure 22: The MoneyMarket code 3/3(Valid.Network, 2020)

3.3.3 BurgerSwap Attack

Burgerswap is a DEX on the Binance Smart Chain (BSC) that was hacked for $7.2 million dollars during a flash loan attack. Burgerwap was a clone of Uniswap v 2.0 with the exception that *a x*y >= k check was not missing*. This missing statement within the "*swap ()"* function allowed anyone to remove any amounts from a pool on the protocol. (Figure 20) (QuillHash Team, 2021)

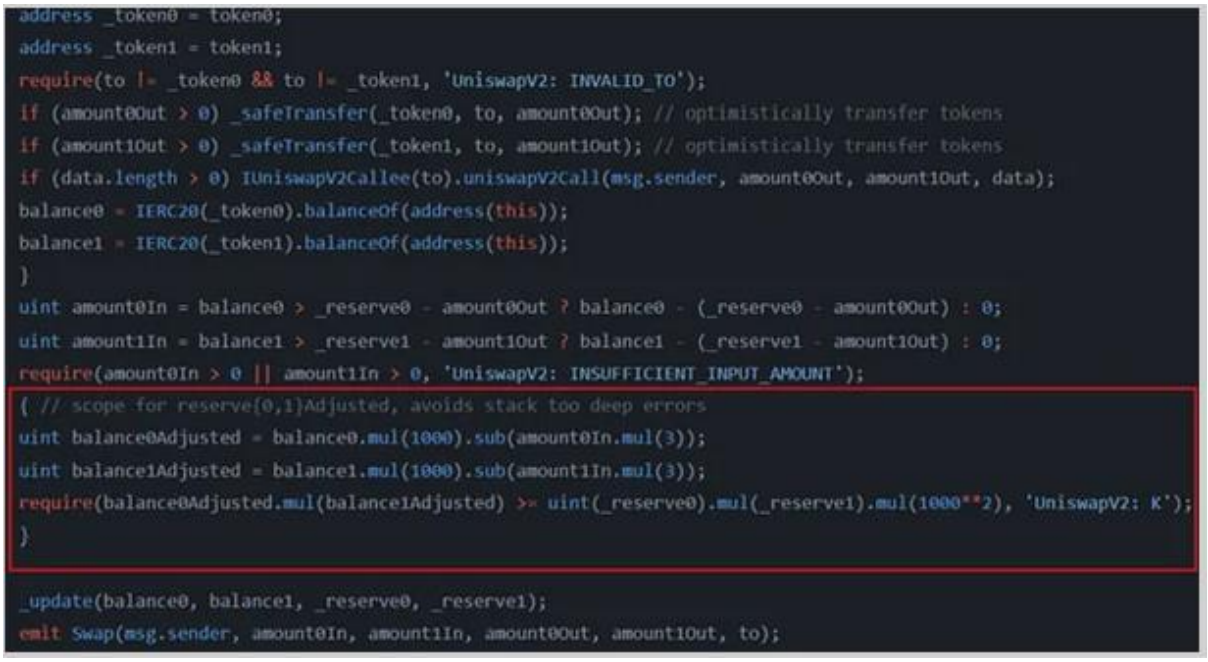
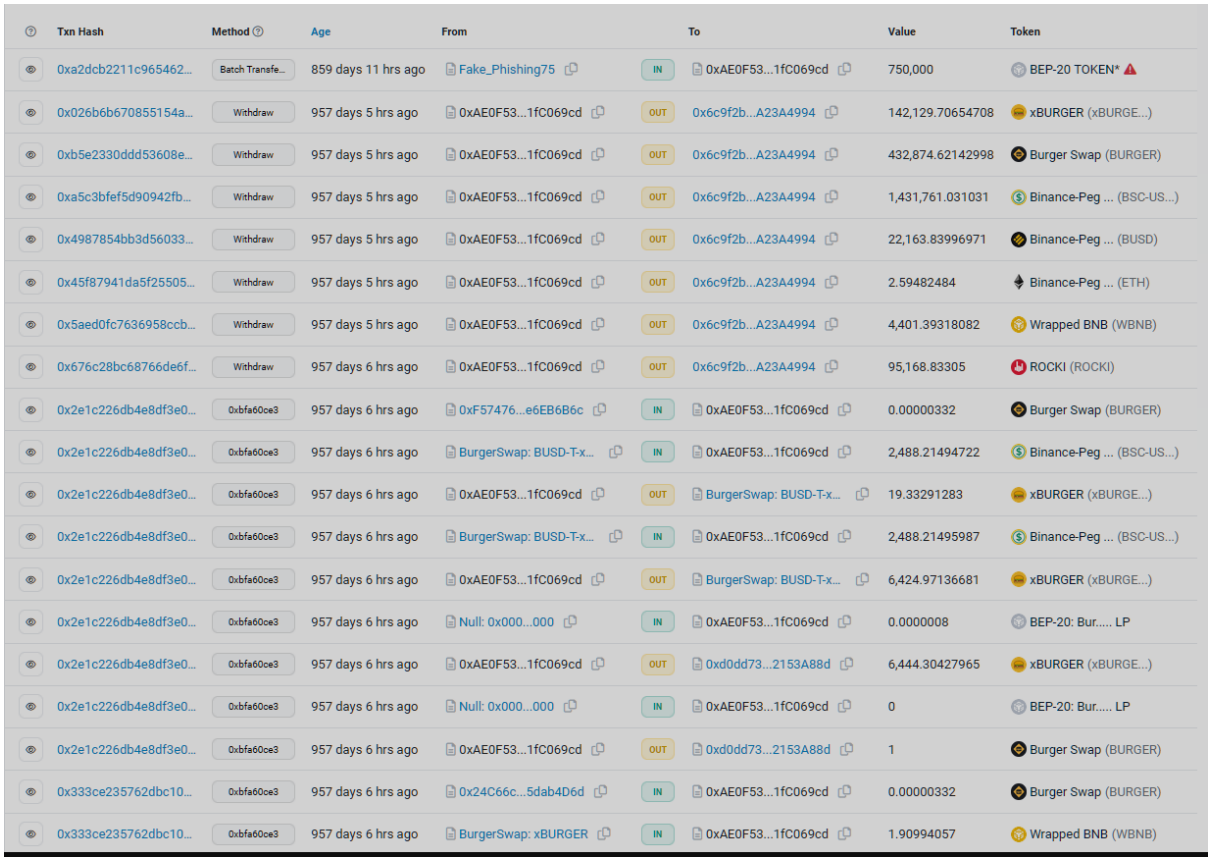


Figure 23: Burgerswap swap function(QuillHash Team, 2021)

The Attacker created a pair of newly created tokens and Burger, the native token of the DEX. During the call of the swap function the Attacker switched the newly created token with the token he wanted to steal withdrawing that way any amount he wanted to. In Figure 21 we see the transactions as they finalized in Binance's BNB Chain Explorer. The Attacker's contract was: xAE0F538409063e66ff0E382113cb1a051fC069cd.

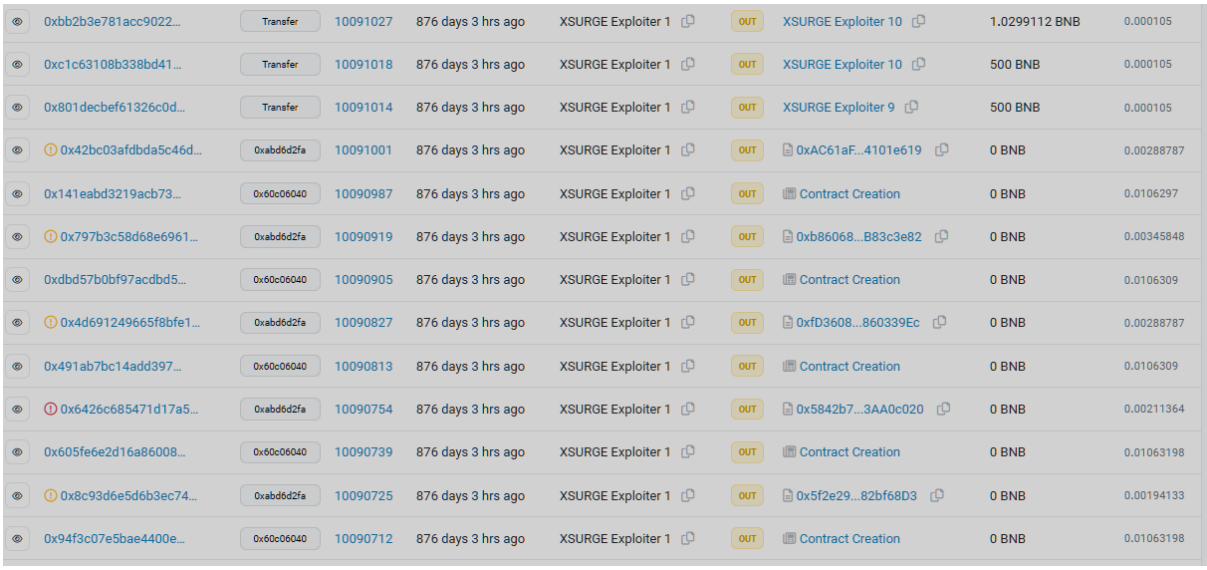


Txn Hash	Method	Age	From		To	Value	Token
0xa2dcb2211c965462...	Batch Transfe...	859 days 11 hrs ago	Fake_Phishing75	IN	0xAE0F53...1fC069cd	750,000	BEP-20 TOKEN*
0x026b6b670855154a...	Withdraw	957 days 5 hrs ago	0xAE0F53...1fC069cd	OUT	0x6c9f2b...A23A4994	142,129.70654708	xBURGER (xBURGE...)
0xb5e2330ddd53608e...	Withdraw	957 days 5 hrs ago	0xAE0F53...1fC069cd	OUT	0x6c9f2b...A23A4994	432,874.62142998	Burger Swap (BURGER)
0xa5c3bfef5d90942fb...	Withdraw	957 days 5 hrs ago	0xAE0F53...1fC069cd	OUT	0x6c9f2b...A23A4994	1,431,761.031031	Binance-Peg ... (BSC-US...)
0x4987854bb3d56033...	Withdraw	957 days 5 hrs ago	0xAE0F53...1fC069cd	OUT	0x6c9f2b...A23A4994	22,163.83996971	Binance-Peg ... (BUSD)
0x45f87941da5f25505...	Withdraw	957 days 5 hrs ago	0xAE0F53...1fC069cd	OUT	0x6c9f2b...A23A4994	2.59482484	Binance-Peg ... (ETH)
0x5aed0fc7636958ccb...	Withdraw	957 days 5 hrs ago	0xAE0F53...1fC069cd	OUT	0x6c9f2b...A23A4994	4,401.39318082	Wrapped BNB (WBNB)
0x676c28bc68766de6f...	Withdraw	957 days 6 hrs ago	0xAE0F53...1fC069cd	OUT	0x6c9f2b...A23A4994	95,168.83305	ROCKI (ROCKI)
0x2e1c226db4e8df3e0...	0xbfa60ce3	957 days 6 hrs ago	0xF57476...e6EB6B6c	IN	0xAE0F53...1fC069cd	0.00000332	Burger Swap (BURGER)
0x2e1c226db4e8df3e0...	0xbfa60ce3	957 days 6 hrs ago	BurgerSwap: BUSD-T-x...	IN	0xAE0F53...1fC069cd	2,488.21494722	Binance-Peg ... (BSC-US...)
0x2e1c226db4e8df3e0...	0xbfa60ce3	957 days 6 hrs ago	0xAE0F53...1fC069cd	OUT	BurgerSwap: BUSD-T-x...	19.33291283	xBURGER (xBURGE...)
0x2e1c226db4e8df3e0...	0xbfa60ce3	957 days 6 hrs ago	BurgerSwap: BUSD-T-x...	IN	0xAE0F53...1fC069cd	2,488.21495987	Binance-Peg ... (BSC-US...)
0x2e1c226db4e8df3e0...	0xbfa60ce3	957 days 6 hrs ago	0xAE0F53...1fC069cd	OUT	BurgerSwap: BUSD-T-x...	6,424.97136681	xBURGER (xBURGE...)
0x2e1c226db4e8df3e0...	0xbfa60ce3	957 days 6 hrs ago	Null: 0x000...000	IN	0xAE0F53...1fC069cd	0.0000008	BEP-20: Bur..... LP
0x2e1c226db4e8df3e0...	0xbfa60ce3	957 days 6 hrs ago	0xAE0F53...1fC069cd	OUT	0xd0dd73...2153A88d	6,444.30427965	xBURGER (xBURGE...)
0x2e1c226db4e8df3e0...	0xbfa60ce3	957 days 6 hrs ago	Null: 0x000...000	IN	0xAE0F53...1fC069cd	0	BEP-20: Bur..... LP
0x2e1c226db4e8df3e0...	0xbfa60ce3	957 days 6 hrs ago	0xAE0F53...1fC069cd	OUT	0xd0dd73...2153A88d	1	Burger Swap (BURGER)
0x333ce235762dbc10...	0xbfa60ce3	957 days 6 hrs ago	0x24C66c...5dab4D6d	IN	0xAE0F53...1fC069cd	0.00000332	Burger Swap (BURGER)
0x333ce235762dbc10...	0xbfa60ce3	957 days 6 hrs ago	BurgerSwap: xBURGER	IN	0xAE0F53...1fC069cd	1.90994057	Wrapped BNB (WBNB)

Figure 24: BurgerSwap Attack Tx's

3.3.4 XSurge Attack

SurgeBNB was the native token of the XSurge Platform, a Decentralized stablecoin provider platform. SurgeBNB token was used as loot in the 2021 of the XSurge Platform resulting in only $4 million losses in USD due to the swift response from the developers that destroyed the contract with the bug that caused this re-entry attack to occur. The attack occurred as the sell function of this protocol modifies the data after the transfer.(Beosin Developers, 2021) In Figure 22 we can see the transactions the attacker made in the BSChain as well as the final withdrawal of 1022 BNB coins. The Attacker's contract was: 0x59c686272e6f11dC8701A162F938fb085D940ad3



0xbb2b3e781acc9022...	Transfer	10091027	876 days 3 hrs ago	XSURGE Exploiter 1	OUT	XSURGE Exploiter 10	1.0299112 BNB	0.000105
0xc1c63108b338bd41...	Transfer	10091018	876 days 3 hrs ago	XSURGE Exploiter 1	OUT	XSURGE Exploiter 10	500 BNB	0.000105
0x801decbef61326c0d...	Transfer	10091014	876 days 3 hrs ago	XSURGE Exploiter 1	OUT	XSURGE Exploiter 9	500 BNB	0.000105
0x42bc03afdbda5c46d...	0xabd6d2fa	10091001	876 days 3 hrs ago	XSURGE Exploiter 1	OUT	0xAC61aF...4101e619	0 BNB	0.00288787
0x141eabd3219acb73...	0x60c06040	10090987	876 days 3 hrs ago	XSURGE Exploiter 1	OUT	Contract Creation	0 BNB	0.0106297
0x797b3c58d68e6961...	0xabd6d2fa	10090919	876 days 3 hrs ago	XSURGE Exploiter 1	OUT	0xb86068...B83c3e82	0 BNB	0.00345848
0xdbd57b0bf97acdbd5...	0x60c06040	10090905	876 days 3 hrs ago	XSURGE Exploiter 1	OUT	Contract Creation	0 BNB	0.0106309
0x4d691249665f8bfe1...	0xabd6d2fa	10090827	876 days 3 hrs ago	XSURGE Exploiter 1	OUT	0xfD3608...860339Ec	0 BNB	0.00288787
0x491ab7bc14add397...	0x60c06040	10090813	876 days 3 hrs ago	XSURGE Exploiter 1	OUT	Contract Creation	0 BNB	0.0106309
0x6426c685471d17a5...	0xabd6d2fa	10090754	876 days 3 hrs ago	XSURGE Exploiter 1	OUT	0x5842b7...3AA0c020	0 BNB	0.00211364
0x605fe6e2d16a86008...	0x60c06040	10090739	876 days 3 hrs ago	XSURGE Exploiter 1	OUT	Contract Creation	0 BNB	0.01063198
0x8c93d6e5d6b3ec74...	0xabd6d2fa	10090725	876 days 3 hrs ago	XSURGE Exploiter 1	OUT	0x5f2e29...82bf68D3	0 BNB	0.00194133
0x94f3c07e5bae4400e...	0x60c06040	10090712	876 days 3 hrs ago	XSURGE Exploiter 1	OUT	Contract Creation	0 BNB	0.01063198

Figure 25: Surge Attack Tx's

3.3.5 Siren Protocol Attack

Siren Protocol is a non-custodial DeFi trading platform. An attacker analyzed and was able to exploit a logical flaw in the core contracts operations escaping with $3.5 million USD in 2021. The flaw as with all re-entracy attacks what that the update of the internal state of the contract was performed after the "_sellOrWithdrawActive" and the "safeTransfer" functions were called. (Behnke Rob, 2021; Siren Protocol, 2021),. The Attacker used 4 different addresses for his attack. These are:

- 0x07Ba7e8947f8Fb4d33f3C7E25c2CB35B858F02Eb
- 0xfAc4088BbA1fA090FD3F1F52fd691a45C30AC053
- 0xf834eFE5B959E52E3b78cB28c4BC501b52CE41da
- 0x99DA8fB52f74B7a3E38d9C75c634f6386F1649C7

3.3.6 Orion Protocol Attack

Orion Protocol is a multi-purpose DeFi system that finds the lowest fee routers for traders. The platform fell victim to a re-entracy attack in 2023 due to a faulty logical third-party smart contract. The attacker after utilizing an Attack similar to XSurge, created a new token, created a pair and started draining multiple pools before sending his loot to the mixer "Tornado Cash". (Nelson Danny, 2023) In this attack, it is clear that the sophistication of the attackers has evolved over the years, as instead of stealing and holding the tokens in a flagged wallet, they now deploy laundering mechanisms in order to be able to use said tokens. Similarly detecting

mechanisms have also evolved as is evident in Figure 23. The Orion Team has identified the attacking wallet as: 0x3DabF5e36DF28F6064a7c5638D0c4e01539E35F1

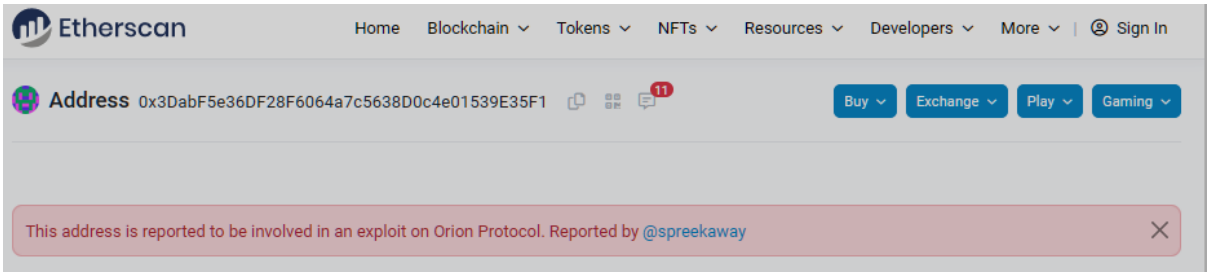


Figure 26: Orion Protocol Tx's

3.3.7 dForce Protocol Attack

DForce is a DeFi stablecoin protocol that was hacked for $3.6 million USD in 2023. The platform fell victim to a re-entracy attack on the function "get_virtual_price" of their smart contract used to calculate oracle prices when connected to Curve Finance for Optimism and Arbitrum blockchain ecosystems. The DeFi protocol took swift actions and paused all protocols in order to stop the attack. (Chawla Vishal, 2023)

3.3.8 Sturdy Finance Attack

DeFi protocol Sturdy Finance was hacked for $0.6 million USD throught a re-entracy attack. The attack exploited similarly to dForce a smart contract that was communicating with an oracle. The protocols teams, swift move froze all contracts, but the attacker was able to siphon and launder through Tornado cash 442 Ether. (Reguerra Ezra, 2023)The Attacker's Address was identified as: 0xeb87ebc0a18aca7d2a9ffcabf61aa69c9e8d3c6efade9e2303f8857717fb9eb7

3.3.9 Stars Arena Attack

Stars Arena is a blockchain-based social token platform. On its platform, creators sell their tickets to private chat rooms. Stars Arena was hacked through a re-entracy attack in 20023 for $2.9 million in $AVAX. The vulnerability existed in the code that managed "shares" to the users. These shares should be sold at a certain amount, but due to the bug the attacker was able to inflate the weight of its share, thereby increasing the value of their share. After the incident the Stars Arena Team froze the platform and began fixing its codebase.(Behnke Rob, 2023)

3.3.10 summary of the **potential impact** of each attack

- The DAO attack: The DAO hack significantly influenced the Ethereum community, resulting in a hard fork that divided the blockchain into two separate entities: Ethereum

and Ethereum Classic. It prompted significant inquiries over the security of smart contracts and the unchangeability of blockchain technology.

- **The Lendf. me Protocol Attack** resulted in a substantial financial loss of $25 million, eroding confidence in DeFi platforms and exposing weaknesses in the design of smart contracts. The text underscored the necessity for implementing more stringent security protocols in DeFi applications.
- **The BurgerSwap** attack led to a financial loss of $7.2 million, revealing significant weaknesses in decentralised exchange protocols and emphasizing the necessity for stronger security measures in smart contract functions.
- **XSurge attack**: The XSurge attack resulted in a financial loss of $4 million, highlighting the dangers linked to re-entrance vulnerabilities in DeFi platforms. Developers promptly responded to address the issue and prevent any potential future assaults.
- **Siren Protocol Attack**: This incident led to the unauthorized acquisition of $3.5 million, underscoring the vulnerabilities associated with re-entrance attacks in decentralised finance (DeFi) trading platforms. It emphasized the significance of ensuring secure contract operations and effective internal state management.
- **Orion Protocol attack**: The Orion Protocol assault resulted in substantial monetary damages and demonstrated the increasing complexity of attackers in exploiting DeFi systems and employing money laundering techniques to conceal illegal profits.
- **The dForce Protocol Attack** resulted in a financial loss of $3.6 million and revealed weaknesses in calculating oracle prices in DeFi protocols. The incident triggered a quick response to stop the attack and review security standards.
- **Sturdy Finance attack**: The Sturdy Finance attack incurred a financial loss of $0.6 million and highlighted the vulnerabilities of re-entrance attacks in DeFi intelligent contracts that rely on oracles. As a precautionary measure, all contracts were frozen to prevent further breaches.
- **Stars Arena Attack**: The Stars Arena attack resulted in a $2.9 million loss in AVAX, exposing weaknesses in the administration of user shares on social token systems based on blockchain technology. The platform was compelled to halt its operations and rectify security vulnerabilities in its coding.

3.4 Mitigation Actions

In the first half of 2023, out of 24 major attacks, reentrancy vulnerabilities were implicated in four incidents, underscoring their ongoing relevance and risk. In the last section of our in-depth analysis of this attack vector, we will look at mitigation actions that can occur in order to protect an intelligent contract from such an attack.

It is evident that the root of the attack is the faulty logic in smart contract development. It is also evident that this misconfiguration can be proactively fixed before a smart contract is deployed into the EVM or a similar computing cloud. To mitigate this attack as much as possible we looked into the most recent academic papers to locate tools, methodologies and other solutions.

Name of Tool / Features	OYENTE [13]	SmartCheck [10]	Securify [14]	GASPER [11]	ZEUS [16]	F* Framework [17]	MAIAN [4]
Type of Analysis	Static	Static	Static	Static	Static	Formal Verification	Dynamic
Code Transformation method	Disassembly, Control flow graph	Abstract Syntax Tree Analysis	Disassembly, De-compilation	Disassembly, Control flow graph	Abstract Syntax Tree Analysis	Formal Methods – translation to formal programming language	Disassembly, Control flow graph
Methodology for analysis	Symbolic Execution and constraint solving for Bytecode	Performs analysis on Solidity code	Abstract interpretation and Datalog (Horn Logic) for Bytecode	Symbolic Execution and constraint solving for Bytecode	Constraint Solving on Solidity code	Uses theorem provers and constructs program logics	Symbolic Execution and constraint solving for Bytecode
Usability & setup	CLI and Web Interface both. Provides docker image to deploy the app.	Web Interface.	Web Interface online.	N/A	N/A	CLI Initial setup takes significant time	Written in python, CLI tool.
Performance metrics	This tool provides high accuracy but detects only 4 vulnerabilities.	This tool fails to detect bugs that can be affected by user input, i.e. taint analysis	This tool generated a positive rate of true warnings. However, it fails to comprehend numerical analysis	This tool detects only 1 vulnerability – that is Gas costly patterns	This tool generated zero false negatives and attributes involving mathematical operations are not validated.	Only focuses on functional correctness and run-time safety. This tool is unable to detect bugs.	This tool traced through an invocation method and generated a high true positive rate.

Figure 27: A comparison of Tools for Analyzing Security Vulnerabilities in Ethereum Smart Contracts(Moona and Mathew, 2021)

A plethora of modern tools that mitigate this attack are discussed (Prasad and Ramachandram, 2022). **Remix** is the first tool that we will be analyzing. Remix is an online IDE used to program smart contracts in Solidity. Remix provides formal verification to detect bugs in solidity and Vyper programming languages. (Bhargavan et al., 2016; Mense and Flatscher, 2018) **Mythril** is another tool released by ConsenSYS that is based on symbolic representation and analyzes contracts depending on their EVM bytecode.(He et al., 2020; Prechtel et al., 2019) **F* Frames** is a tool released by Microsoft based on formal verification with functional programming language. Smart contract code or EVM byte code converted into F* language for detecting vulnerabilities like exception handling and re-entrancy.

(Praitheeshan et al., 2019) **VANDAL** (Brent et al., 2018) appears to be the tool providing the fastest results when compared to all other existing tools. VANDAL takes input solidity code and converts it into semantic relations to detect bugs within the code. **Oyente** is a symbolic analysis tool used to identify security vulnerabilities in the EVM bytecode. (Luu et al., 2016; Praitheeshan et al., 2019). Lastly, **ZEUS** is a static analysis tool that converts Solidity contracts into an XACML- styled format. (Kalra et al., 2018)

RA (Re-entrancy Analyzer) is suggested as a mitigation tool(Chinen et al., 2021). The tool uses the satisfiability modulo theories (SMT) solver to evaluate Ethereum smart contracts for vulnerabilities to re-entrancy attacks. RA's key advantage is its ability to analyze EVM bytecodes without prior knowledge of attack patterns or the need to spend Ether. This is a significant improvement from the other suggested tools as they require both knowledge and gas in order to operate. RA also supports the analysis of inter-contract behaviours, an area where current tools also lack acting capabilities. RA's approach is to simulate the behaviour of smart contracts and several attacks symbolically, generate control flow graphs (CFGs), and check the equivalence of program states with and without re-entrancy. This methodology allows RA to precisely determine the vulnerability of contracts to re-entrancy attacks in a local environment and not in the EVM chain itself. It is noted that RA also possesses inter-contract analysis capabilities, in order to mitigate attacks in cases of both a new contract is created or an existing one calls a different function. Given that third-party smart contracts, smart-contract copying, and usage are both highly used from developers. This tool would be able to analyze a chain of contracts and their relationship potentially proactively mitigating such attacks. (Chinen et al., 2021). **SmartCheck** is a static analysis tool similar to RA although with fewer capabilities, as it cannot detect all vulnerabilities. SmartCheck mainly focuses on identifying issues such as fallback functions, mismatches in compiler versions, and violations of style guides. (Tikhomirov et al., 2018). Building upon SmartCheck (Fei et al., 2023), a new advanced tool is presented **MSmart**. MSmart analyzes smart contracts by converting the source code into an intermediate representation and using XPath rules to identify vulnerabilities. This tool demonstrated improved efficiency, accuracy and false positive ratio when used on a 6.000 real-world intelligent contracts dataset. **Slither** is also a static analysis framework for smart contracts like SmartCheck. (Feist et al., 2019) Slither converts the intelligent contract code into an Abstract Syntax Tree (AST) as its input format for analysis. Like SmartCheck, Slither cannot identify all critical bugs and vulnerabilities associated with the re-entry attacks.

A solution (Alkhalifah et al., 2021) suggests creating a **balance keep tracker** of a smart contract so that It can compare the overall balance of all participants at any time, whitelist

raising a flag if any alterations occur. This would fix the faulty programming logic of updating the internal balance state after all external contracts are called Victim contract would be calling this solution as an external contracts. The advantage of this approach is that it could potentially patch intelligent contracts that have already been deployed in the EVM (making them immutable). The solutions researchers emphasize the necessity of such a mechanism as smart contracts' popularity and financial significance continue to grow, making them attractive targets for adversaries. **Sereum** is a new technology (Rodler et al., 2018) that operates based on runtime monitoring and validation with the capacity to detect and prevent re-entrance attacks without modifying any pre-existing smart contracts. Sereum utilizes dynamic taint-tracking to monitor the flow of potentially attackable data through any contract's stored variables. If a stored variable is found to be potentially attackable through its control-flow decision, Sereum could detect and detain that contract to present exploitation. Lastly, Sereum builds a dynamic call tree during execution, thus allowing the understanding and monitoring of complex call relationships within transactions.

It is clear, given the magnitude of this attack that has occurred in Ethereum's EVM and similar blockchains with EVM compatible programming languages. This has resulted in the **multitude of mitigation tools and different security approaches** we analyzed above. For the last mitigation methodology will be delving into the Cardano blockchain ecosystem. Cardano uses the **"Extended Unspent Transaction Output (EUTXO) model"** (Chakravarty et al., 2020a, 2020b) This transaction model is the core of Cardano's Blockchain ecosystem and introduces Constraint Emitting Machines (CEMs) as well as demonstrates a weak bisimulation between the machine model and the ledger model. In the EUTXO model, every transaction can only access the state of contracts connected at the time. This local contract approach greatly diminishes the attack surface of the re-entry attack, which typically uses the global blockchain state. Additionally, the EUTXO model avoids using shared mutable states. Once a state is read, it cannot be changed by another transaction until the current transaction is completed. This serial logic again prevents the re-entracy attack from occurring. Lastly EUTXO is based on data flow instead of flow control, meaning that the outputs (including state changes) are determined before execution, making it impossible for a malicious actor to alter the path of a contract after execution.

The figure below presents the categorization of the tools mentioned above and their main functionalities.

Figure 28: categorization of the tools

Chapter 4 Comparative Analysis

In this chapter, there is an evaluation of all attacks mentioned in Chapters 2 and 3 while also comparing Cardano's and Ethereum's mitigation techniques. Please note that although Ethereum has suffered more breaches and possesses more open vulnerabilities as Ethereum was the first blockchain to implement smart contracts in 2015 (Alkhalifah et al., 2019), whereas Cardano was founded in 2015, and its smart contracts functionality was implemented in 2021. (Vladislav Sopov, 2021)

Attack Name	**Attack Vector**	**Cardano Analysis**	**Ethereum Analysis**
51% Attack	consensus mechanism	Never occurred	Never occurred
		Ouroboros mitigates this attack via Cryptographic Randomness in Leader Selection	33% of attacks can cause finality delays
Nothing at Stake	consensus mechanism	Never occurred	Never occurred
		Ouroboros mitigates this attack through incentive alignment	Ethereum penalizes malicious validators
Long Range	consensus mechanism	Never occurred	Never occurred
		Ouroboros mitigates this attack via Bootstrapping from Genesis and Incentives Alignment.	Ethereum 2.0 mitigates this attack through weak subjectivity, validator deposits, and slashing.
Stake Grinding	consensus mechanism	Never occurred	Never occurred
		Ouroboros mitigates this attack via Cryptographic Randomness in Leader Selection.	Ethereum uses random functions for validator selection.
Integer Overflow	smart contract	Never occurred	Multiple attacks occurred
		EUTXO model is designed to mitigate this attack	Multiple tools have been deployed to stop these attacks: EASYFLOW, SafeMath, Smartcheck and more.
MEV (Miner	smart contract	Never occurred	Multiple attacks occurred and are still ongoing

Extractable Value)		Ouroboros - Praos mitigates this attack by providing limited influence of Transaction orders to validators.	Flashbot, a research Team, is investigating how to mitigate this attack properly.
Gas exhaustion & DDOS	smart contract	Never occurred	2016 DoS attack
		Hydra will increase the performance and reduce the likelihood of this attack even more	the network performance is expected to be increased with sharding
Re-entry attack	smart contract	Never occurred	2016 DAO Attack
		EUTXO model is designed to mitigate this attack	The most common attack on Ethereum
Flash Loan	DeFi	Never occurred	Multiple attacks occurred and are still ongoing
		EUTXO model is designed to mitigate this attack	Ethereum global state is enabling this attack
Liquidity Attack	DeFi	Never occurred	Multiple Events have occurred
		EUTXO model is designed to mitigate this attack	Ethereum global state is enabling this attack

4.1 Synopsis of the Attacks

As briefly summarized in the table above, the first 4 Attacks, 51% Attack, Nothing at Stake, Long Range and Stake Grinding, occur at a consensus level. These attacks were analyzed in Chapter 2, but the incentivization for such an attack to occur would only be from external factors, not the blockchains' users. Any of these four attacks would have catastrophic consequences for a specific ecosystem and the trust in the Blockchain. Given the low incentivization but high destruction rates and the academic interest in their analysis, it would be safe to assume that these attacks will likely never occur in the two most prominent 3rd Generation Blockchain ecosystems. A notable exclusion would be the 51% attack, as we ought to monitor the ecosystems and try to minimize any centralization effort. The similarities between these attack vectors are also evident from the analysis of these attacks. A long-range

attack can eventually perform a 51% attack, and a Stake-grinding attack can be used to perform a long-range attack, etc.

Integer Overflow / Underflow, Transaction order Dependence, Gas exhaustion and re-entrance attacks occur at the smart contract level. This attack level is a much more narrowly defined spectrum than the previous, only affecting individual users and applications built upon an ecosystem. Indirectly, it can harm the reputation of a blockchain, but the main target is economic incentives from malicious actors. As shown in the detailed analysis of Chapter 3, even if an attacker is successful, unless he siphons the fund fast through a tumbler or a mixer, he will be unable to them for anything of substance. These attacks are automatically flagged in the Centralized Exchanges (CEX) and the stablecoin issuing companies, making it impossible to use the funds without revealing oneself to the authorities. Given the analysis of the re-entry attack, an attack that was first exploited in 2016 and continues to be exploited in 2023, it is safe to say that due to the fast growth of the technology, the third-party applications and reusability of the code, it is a logical programming bug that will persist. We can conclude Through the same analysis that even if the funds of these attacks are highly illiquid, that will not prevent many malicious actors from constantly attacking and looting protocols with weak security policies. Security consulting, security auditing and security guidelines are highly needed so the developers of the blockchain ecosystems can focus on innovation alone.

Lastly, the attacks named “Flash Loans” and “Just in Time” exist only in EVM-compatible blockchains because only in the Ethereum Virtual Machine can a malicious user manipulate the smart contracts' global state to bundle and execute multiple requests on the same block. An observation of this Thesis is that this is designed to promote parallel programming, and indeed, Decentralized Finance, currently the flagship of all 3rd generation blockchains, would not exist if it were not for the Ethereum Virtual Machine.

4.2 Ethereum Security Approach

Ethereum is at the forefront of many of the above-analyzed attacks, with multiple hacks, chain halts, one eventual hard fork and billions of USD stolen and contained in addresses that no one is actively using. Ethereum has helped shape many of the beliefs in the entire blockchain technology. More specifically, because of the hard fork and the centralized decision it was called to make at such an early stage, it has sparked philosophical discussions on what malicious behaviour is and whether a backdoor should exist in blockchains or “code is always law”. While Ethereum’s mindset of “move fast, break things and fix them later” does seem to be currently working as it is the undisputed king of the arena of code development, this approach can only work so far until better, safer, and more robust solutions arrive at the arena. Of the ten attacks we used in this Thesis Ethereum has suffered hacks via 6. Multiple attacks

this Thesis did not have the room to analyze exist, and almost all are focused on Ethereum. Ethereum is a honeypot for users and developers but also malicious hackers and scammers looking to obtain as much capital as possible from honest users.

Plans of Ethereum known as the Surge, the Verge, the Purge and the Splurge have smart contract improvement proposals that will enhance Ethereum's security and scalability, making it on par with the roadmap of other blockchains. Currently, Ethereum is also the second ecosystem (alongside Bitcoin) with an active Layer 2 ecosystem, assisting in mitigating many of the attacks that are active in Ethereum Layer 1 through their multiple competing blockchains.

4.3 Cardano Security Approach

Cardano has taken a completely different approach to cybersecurity from many other blockchain ecosystems competing with Ethereum. Cardano opted for a methodical, peer-reviewed process that initially prioritizes asset and blockchain security. The insight provided from the literature review of Chapters 2 and 3, alongside the evident lack of attacks on its ecosystem, is stellar proof that Cardano has taken its security infrastructure seriously. Ouroboros and EUTXO are the flagships of this blockchain, mitigating 9 out of the 10 Attacks this Thesis showcased. However, this approach does have its drawbacks. Cardano, until recently, did not have a functional smart contract programming language; therefore, it had no DeFi ecosystem to foster. Cardano is highly criticized for its peer review process, as it is a rather lengthy and meticulous process, making Cardano appear to be a slower evolving/moving ecosystem than its competitors.

Cardano is in the 4th step (out of 5) of its roadmap with upgrades such as the scaling solution Hydra and the Cardano Store, will even further advance this blockchain and make it the number one in terms of security and potential throughput.

Chapter 5 Conclusion

As this thesis draws to a close, it is clear that blockchain technology is a fast-paced, rapidly evolving, complex technology. Although blockchain can become the bedrock of all future infrastructures as the Internet currently is, it is clear that all these fast-faced actions open the door to multiple vulnerabilities and attack vectors. Due to the immutability of blockchain technology, these bugs are permanent without the capacity to be fixed. **Both Cardano and Ethereum are making impressive strides ahead, but they operate in a hostile environment where malicious actors constantly seek to exploit possible shortcomings**. The importance of robust and adaptive defenses and the pursuit of new defenses to bolster ever-evolving attacking methodologies cannot be overstated.

Answering the "Research Questions" of these Thesis:

Research Question 1: What are the fundamental elements of blockchain technology and how do they contribute to the cybersecurity challenges in third-generation blockchains?

Security in third generation blockchains is heavily impacted by the cryptographic principles, consensus procedures, distributed ledgers, and other underlying components of blockchain technology. Unique security issues, such as the 'Nothing at Stake' dilemma and vulnerability to other attack vectors like majority and long-range attacks, are introduced when platforms like Ethereum go from Proof of Work (PoW) to Proof of Stake (PoS). Even though its efficacy against complex cyber threats is still an issue, cryptography plays an essential role in protecting transactions and data. In order to establish the strengths and weaknesses of third-generation blockchains, this thesis seeks to give a thorough comprehension of these components and how they affect security.

Research Question 2: What are the primary cybersecurity risks that are unique to third generation blockchains, and how do they appear in prominent blockchain ecosystems like as Ethereum and Cardano?

Key cybersecurity threats specific to blockchains of the third generation are the subject of this research. This requires a thorough investigation of potential entry points for attacks on PoS consensus mechanisms and weaknesses in smart contracts. Integer Overflow, Transaction Order Dependence, Gas Exhaustion, and Majority Attacks are among of the smart contract vulnerabilities covered in the report. Other attacks include 'Nothing at Stake,' 'Long Range,' and Stake Grinding. Attacks on Flash Loans and Liquidity Pools, two new forms of decentralised financial infrastructure, are also part of the study's scope. An important portion

of this research is devoted to figuring out how these dangers show up in prominent blockchain ecosystems, like as Ethereum and Cardano, so we can learn about their resistance to and reactions to these problems.

Research Question 3: How can businesses, researchers, and public entities apply the findings of this thesis to improve cybersecurity practices in third-generation blockchain ecosystems?

Finally, the research discusses how third generation blockchain ecosystems' cybersecurity procedures might be improved by using the findings by enterprises, researchers, and public entities. Businesses can use the findings to strengthen their security policies and strategies for managing risk. The results can be used by researchers to delve deeper into blockchain cybersecurity, which could result in new security solutions. If these findings can help public bodies shape regulatory frameworks to promote the long-term viability of blockchain technology, that would be great. By bringing together theoretical considerations and real-world applications, this study hopes to aid in the development of better cybersecurity protocols for the dynamic blockchain environment.

Chapter 6 Future Work

The work that will be done in the future should concentrate not only on discovering and understanding new vulnerabilities, but also on building novel techniques and tools for defense and mitigation to address these vulnerabilities. Because of the dynamic nature of blockchain technology and the inventiveness of opposed actors, it is necessary to take a proactive and continuously evolving approach to its security. The reliability and integrity of blockchain technology as a fundamental technology of the future will be ensured by doing research of this nature, which will be essential in the process of strengthening blockchain infrastructure against a wide variety of cyber attacks. The end goal of this ongoing research is to improve the security architecture of blockchain technology, making it more resistant to sophisticated attacks and adaptable to the quickly changing digital landscape. This is the ultimate goal of this research.

.

Bibliography

AAVE Developer Team, 2023. Flash Loans [WWW Document]. AAVE. URL https://docs.aave.com/developers/guides/flash-loans (accessed 1.10.24).

Alkhalifah, A., Ng, A., Chowdhury, M.J.M., Kayes, A.S.M., Watters, P.A., 2019. An Empirical Analysis of Blockchain Cybersecurity Incidents, in: 2019 IEEE Asia-Pacific Conference on Computer Science and Data Engineering (CSDE). IEEE, pp. 1–8. https://doi.org/10.1109/CSDE48274.2019.9162381

Alkhalifah, A., Ng, A., Watters, P.A., Kayes, A.S.M., 2021. A Mechanism to Detect and Prevent Ethereum Blockchain Smart Contract Reentrancy Attacks. Front Comput Sci 3. https://doi.org/10.3389/fcomp.2021.598780

Antonopoulos, A.M., Wood, G., 2018. Mastering Ethereum: Building Smart Contracts and DApps. . O'Reilly Media.

Azouvi, S., Vukolić, M., 2022. Pikachu: Securing PoS Blockchains from Long-Range Attacks by Checkpointing into Bitcoin PoWusing Taproot, in: ConsensusDay 2022 - Proceedings of the 2022 ACM Workshop on Developments in Consensus, Co-Located with CCS 2022. Association for Computing Machinery, Inc, pp. 63–65. https://doi.org/10.1145/3560829.3563563

Bada, A.O., Damianou, A., Angelopoulos, C.M., Katos, V., 2018. Towards a Green Blockchain: A Review of Consensus Mechanisms and their Energy Consumption.

Badertscher, C., Gaži, P., Kiayias, A., Russell, A., Zikas, V., 2019a. Ouroboros Chronos: Permissionless Clock Synchronization via Proof-of-Stake.

Badertscher, C., Gaži, P., Kiayias, A., Russell, A., Zikas, V., 2019b. Ouroboros Genesis: Composable Proof-of-Stake Blockchains with Dynamic Availability.

Behnke Rob, 2023. Explained: The Stars Arena Hack (October 2023) [WWW Document]. Halborn. URL https://www.halborn.com/blog/post/explained-the-stars-arena-hack-october-2023 (accessed 1.10.24).

Behnke Rob, 2021. Explained: The SIREN Protocol Hack (September 2021) [WWW Document]. Halborn. URL https://www.halborn.com/blog/post/explained-the-siren-protocol-hack-september-2021 (accessed 1.10.24).

Beosin Developers, 2021. XSURGE on the BSC Chain was Attacked by Lightning Loans — A Full Analysis [WWW Document]. Medium.com. URL https://beosin.medium.com/a-sweet-blow-fb0a5e08657d (accessed 1.10.24).

Bhargavan, K., Delignat-Lavaud, A., Fournet, C., Gollamudi, A., Gonthier, G., Kobeissi, N., Kulatova, N., Rastogi, A., Sibut-Pinote, T., Swamy, N., Zanella-Béguelin, S., 2016. Formal Verification of Smart Contracts, in: Proceedings of the 2016 ACM Workshop on Programming Languages and Analysis for Security. ACM, New York, NY, USA, pp. 91–96. https://doi.org/10.1145/2993600.2993611

Blum, E., Kiayias, A., Moore, C., Quader, S., Russell, A., 2019. The combinatorics of the longest-chain rule: Linear consistency for proof-of-stake blockchains.

Bonneau, J., Miller, A., Clark, J., Narayanan, A., Kroll, J.A., Felten, E.W., 2015. SoK: Research Perspectives and Challenges for Bitcoin and Cryptocurrencies, in: 2015 IEEE Symposium on Security and Privacy. pp. 104–121. https://doi.org/10.1109/SP.2015.14

Brent, L., Jurisevic, A., Kong, M., Liu, E., Gauthier, F., Gramoli, V., Holz, R., Scholz, B., 2018. Vandal: A Scalable Security Analysis Framework for Smart Contracts.

Buterin, V., 2014. Ethereum: A Next-Generation Smart Contract and Decentralized Application Platform.

Buterin Vitalik, n.d. Long-Range Attacks: The Serious Problem With Adaptive Proof of Work [WWW Document]. 2014. URL https://blog.ethereum.org/2014/05/15/long-range-attacks-the-serious-problem-with-adaptive-proof-of-work (accessed 1.10.24).

C. Comben, 2019. Delegated byzantine fault tolerance (dbft) explained [WWW Document].

Caldarelli, G., Ellul, J., 2021. The Blockchain Oracle Problem in Decentralized Finance—A Multivocal Approach. Applied Sciences 11, 7572. https://doi.org/10.3390/app11167572

Cardano Development Team, 2023. Plutus [WWW Document]. IOHK. URL Cardano Development Team. (2023). Plutus. IOHK. (accessed 1.10.24).

Chainanalysis, 2023. The 2023 Crypto Crime Report.

Chakravarty, M.M.T., Chapman, J., Mackenzie, K., Melkonian, O., Jones, M.P., Wadler, P., 2020a. The Extended UTXO Model.

Chakravarty, M.M.T., Chapman, J., Mackenzie, K., Melkonian, O., Müller, J., Jones, M.P., Vinogradova, P., Wadler, P., 2020b. Native Custom Tokens in the Extended UTXO Model.

Chakravarty, M.M.T., Coretti, S., Fitzi, M., Gaži, P., Kant, P., Kiayias, A., Russell, A., 2021. Hydra: Fast Isomorphic State Channels.

Chawla Vishal, 2023. DForce protocol drained of $3.6 million in reentrancy attack [WWW Document]. The Block. URL https://www.theblock.co/post/210518/dforce-protocol-drained-of-3-6-million-in-reentrancy-attack (accessed 1.10.24).

Chen, T., Li, Z., Zhu, Y., Chen, J., Luo, X., Lui, J.C. s, Lin, X., Zhang, X., 2020. Understanding Ethereum via Graph Analysis. ACM Trans Internet Technol 20, 1–32. https://doi.org/10.1145/3381036

Chinen, Y., Yanai, N., Cruz, J.P., Okamura, S., 2021. Ra: A static analysis tool for analyzing re-entrancy attacks in ethereum smart contracts. Journal of Information Processing 29, 537–547. https://doi.org/10.2197/IPSJJIP.29.537

Chinen, Y., Yanai, N., Cruz, J.P., Okamura, S., 2020. Hunting for Re-Entrancy Attacks in Ethereum Smart Contracts via Static Analysis.

Chiu Jonathan, Koeppl Thorsten, 2019. The Economics of Cryptocurrencies—Bitcoin and Beyond [WWW Document]. ideas.repec.org. URL https://ideas.repec.org/p/bca/bocawp/19-40.html (accessed 1.10.24).

Christidis, K., Devetsikiotis, M., 2016. Blockchains and Smart Contracts for the Internet of Things. IEEE Access 4, 2292–2303. https://doi.org/10.1109/ACCESS.2016.2566339

Cohen, B., Pietrzak, K., 2019. The Chia Network Blockchain.

Corwin Smith, 2023. Ethereum proof-of-stake attack and defense [WWW Document]. Ethereum Foundation.

Daian, P., Goldfeder, S., Kell, T., Li, Y., Zhao, X., Bentov, I., Breidenbach, L., Juels, A., 2020. Flash boys 2.0: Frontrunning in decentralized exchanges, miner extractable value, and consensus instability, in: Proceedings - IEEE Symposium on Security and Privacy. Institute of Electrical and Electronics Engineers Inc., pp. 1106–1120. https://doi.org/10.1109/SP40000.2020.00040

Deirmentzoglou, E., Papakyriakopoulos, G., Patsakis, C., 2019. A survey on long-range attacks for proof of stake protocols. IEEE Access 7, 28712–28725. https://doi.org/10.1109/ACCESS.2019.2901858

Diffie, W., Hellman, M.E., 2022. New Directions in Cryptography, in: Democratizing Cryptography. ACM, New York, NY, USA, pp. 365–390. https://doi.org/10.1145/3549993.3550007

Duan, L., Sun, Y., Zhang, K., Ding, Y., 2022. Multiple-Layer Security Threats on the Ethereum Blockchain and Their Countermeasures. Security and Communication Networks 2022. https://doi.org/10.1155/2022/5307697

Dupont, Q., 2017. Experiments in Algorithmic Governance: A history and ethnography of " The DAO, " a failed Decentralized Autonomous Organization.

E-Gold [WWW Document], n.d. . stanford.edu. URL https://cs.stanford.edu/people/eroberts/cs201/projects/2010-11/Bitcoins/e-gold.html (accessed 1.10.24).

Empiricinfotech, 2023. Medium - Understanding Smart Contracts: A Beginner's Guide [WWW Document]. URL https://medium.com/@empiricinfotech/understanding-smart-contracts-a-beginners-guide-202202e7a2ca (accessed 1.9.24).

Ethereum Proposer-Builder Separation: Past, Present, and Future [WWW Document], 2023. . GateLearn. URL https://www.gate.io/learn/articles/ethereum-proposer-builder-separation/921 (accessed 1.10.24).

Eyal, I., Sirer, E.G., 2018. Majority is Not Enough: Bitcoin Mining is Vulnerable. Commun. ACM 61, 95–102. https://doi.org/10.1145/3212998

Fang, W., Chen, W., Zhang, W., Pei, J., Gao, W., Wang, G., 2020. Digital signature scheme for information non-repudiation in blockchain: a state of the art review. EURASIP J Wirel Commun Netw 2020, 56. https://doi.org/10.1186/s13638-020-01665-w

Farokhnia, S., 2023. Lazy Contracts: Alleviating High Gas Costs by Secure and Trustless Off-chain Execution of Smart Contracts.

Fei, J., Chen, X., Zhao, X., 2023. MSmart: Smart Contract Vulnerability Analysis and Improved Strategies Based on Smartcheck. Applied Sciences (Switzerland) 13. https://doi.org/10.3390/app13031733

Feist, J., Grieco, G., Groce, A., 2019. Slither: A Static Analysis Framework For Smart Contracts. https://doi.org/10.1109/WETSEB.2019.00008

Francisco, S., 2018. Advanced Decentralized Blockchain Platform Whitepaper Version: 2.0.

Frankenfield Jake, 2023. DigiCash: Meaning, History, Implications [WWW Document]. Investopedia. URL https://www.investopedia.com/terms/d/digicash.asp (accessed 1.10.24).

Gao, J., Liu, H., Liu, C., Li, Q., Guan, Z., Chen, Z., 2018. EASYFLOW: Keep Ethereum Away From Overflow. https://doi.org/10.1109/ICSE-Companion.2019.00029

Gazi, P., Kiayias, A., Russell, A., 2018. Stake-Bleeding Attacks on Proof-of-Stake Blockchains, in: 2018 Crypto Valley Conference on Blockchain Technology (CVCBT). IEEE, pp. 85–92. https://doi.org/10.1109/CVCBT.2018.00015

Gaži, P., Ren, L., Russell, A., 2022. Practical Settlement Bounds for Longest-Chain Consensus.

Gervais, A., Karame, G.O., Wüst, K., Glykantzis, V., Ritzdorf, H., Capkun, S., 2016. On the Security and Performance of Proof of Work Blockchains, in: Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security. ACM, New York, NY, USA, pp. 3–16. https://doi.org/10.1145/2976749.2978341

Ghosh, A., Gupta, S., Dua, A., Kumar, N., 2020. Security of Cryptocurrencies in blockchain technology: State-of-art, challenges and future prospects. Journal of Network and Computer Applications 163, 102635. https://doi.org/https://doi.org/10.1016/j.jnca.2020.102635

Grigg, I., 2017. EOS-An Introduction.

Hashcash [WWW Document], n.d. . 2022. URL https://en.bitcoin.it/wiki/Hashcash (accessed 1.10.24).

He, D., Deng, Z., Zhang, Y., Chan, S., Cheng, Y., Guizani, N., 2020. Smart Contract Vulnerability Analysis and Security Audit. IEEE Netw 34, 276–282. https://doi.org/10.1109/MNET.001.1900656

Housley, R., Polk, W.T., 2001. Planning for PKI: Best practices guide for deploying public key infrastructure. John Wiley & Sons.

Hyperledger [WWW Document], 2024. URL https://www.hyperledger.org/about (accessed 1.10.24).

Kalra, S., Goel, S., Dhawan, M., Sharma, S., 2018. ZEUS: Analyzing Safety of Smart Contracts, in: Proceedings 2018 Network and Distributed System Security Symposium. Internet Society, Reston, VA. https://doi.org/10.14722/ndss.2018.23082

kamilpolak, 2022. Reentracy Attack Scenario [WWW Document]. hackernoon.com. URL https://hackernoon.com/hack-solidity-reentrancy-attack (accessed 1.10.24).

Karame, G.O., Androulaki, E., Capkun, S., 2012. Double-spending fast payments in bitcoin, in: Proceedings of the 2012 ACM Conference on Computer and Communications Security. ACM, New York, NY, USA, pp. 906–917. https://doi.org/10.1145/2382196.2382292

Karantias, K., Kiayias, A., Zindros, D., 2012. Proof-of-Burn.

Karpinski, M., Kovalchuk, L., Kochan, R., Oliynykov, R., Rodinko, M., Wieclaw, L., 2021a. Blockchain technologies: Probability of double-spend attack on a proof-of-stake consensus. Sensors 21. https://doi.org/10.3390/s21196408

Karpinski, M., Kovalchuk, L., Kochan, R., Oliynykov, R., Rodinko, M., Wieclaw, L., 2021b. Blockchain Technologies: Probability of Double-Spend Attack on a Proof-of-Stake Consensus. Sensors 21, 6408. https://doi.org/10.3390/s21196408

Kiayias, A., Panagiotakos, G., 2016. Speed-Security Tradeos in Blockchain Protocols.

Kiayias, A., Russell, A., 2018. Ouroboros-BFT: A Simple Byzantine Fault Tolerant Consensus Protocol.

Kiayias, A., Russell, A., David, B., Oliynykov, R., 2019. Ouroboros: A Provably Secure Proof-of-Stake Blockchain Protocol.

Kimmell Matthew, 2020. Liberty Reserve [WWW Document]. URL https://www.coindesk.com/company/liberty-reserve/ (accessed 1.10.24).

König, L., Unger, S., Kieseberg, P., Tjoa, S., 2020. The risks of the blockchain a review on current vulnerabilities and attacks. Journal of Internet Services and Information Security 10, 110–127. https://doi.org/10.22667/JISIS.2020.08.31.110

L. Matthews., 2021. The 15 most sustainable cryptocurrencies for 2021. [WWW Document]. URL https://www.leafscore.com/blog/the-9-mostsustainable- cryptocurrencies-for-2021/ (accessed 1.9.24).

Lamport, L., Shostak, R., Pease, M., 1982. The Byzantine Generals Problem.

Larimer Daniel, 2013. Transactions as Proof-of-Stake.

Lee, K., Kim, H., 2023. Two-Round Multi-Signatures from Okamoto Signatures. Mathematics 11, 3223. https://doi.org/10.3390/math11143223

Lewis-Pye, A., Roughgarden, T., 2023. Byzantine Generals in the Permissionless Setting.

Li, K., Wang, Y., Tang, Y., 2021. DETER: Denial of Ethereum Txpool sERvices, in: Proceedings of the ACM Conference on Computer and Communications Security. Association for Computing Machinery, pp. 1645–1667. https://doi.org/10.1145/3460120.3485369

Li, X., Jiang, P., Chen, T., Luo, X., Wen, Q., 2020. A survey on the security of blockchain systems. Future Generation Computer Systems 107, 841–853. https://doi.org/10.1016/j.future.2017.08.020

Liao, K., Katz, J., n.d. Incentivizing Blockchain Forks via Whale Transactions.

Luu, L., Chu, D.-H., Olickel, H., Saxena, P., Hobor, A., 2016. Making Smart Contracts Smarter, in: Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security. ACM, New York, NY, USA, pp. 254–269. https://doi.org/10.1145/2976749.2978309

Lys, L., Forestier, S., Vodenicarevic, D., Laversanne-Finot, A., 2023. Defending against the nothing-at-stake problem in multi-threaded blockchains.

Ma, F., Fu, Y., Ren, M., Sun, W., Song, H., Jiang, Y., Sun, Jun, Sun, Jiaguang, 2019. V-Gas: Generating High Gas Consumption Inputs to Avoid Out-of-Gas Vulnerability.

Mense, A., Flatscher, M., 2018. Security Vulnerabilities in Ethereum Smart Contracts, in: Proceedings of the 20th International Conference on Information Integration and Web-Based Applications & Services. ACM, New York, NY, USA, pp. 375–380. https://doi.org/10.1145/3282373.3282419

Moona, A., Mathew, R., 2021. Review of Tools for Analyzing Security Vulnerabilities in Ethereum based Smart Contracts. SSRN Electronic Journal. https://doi.org/10.2139/ssrn.3769774

Morrison, R., Mazey, N.C.H.L., Wingreen, S.C., 2020. The DAO Controversy: The Case for a New Species of Corporate Governance? Frontiers in Blockchain 3. https://doi.org/10.3389/fbloc.2020.00025

Nakamoto, S., 2008. Bitcoin: A Peer-to-Peer Electronic Cash System.

Nelson Danny, 2023. Orion Protocol Loses $3M of Crypto in Trading Pool Exploit [WWW Document]. Coindesk. URL https://www.coindesk.com/business/2023/02/02/orion-protocol-loses-3m-of-crypto-in-trading-pool-exploit/ (accessed 1.10.24).

NEM Developer Team, 2018. NEM technical Reference Whitepaper.

Neo Project [WWW Document], 2021. URL https://docs.neo.org/docs/enus/index.html (accessed 1.9.24).

Neuder, M., Moroz, D.J., Rao, R., Parkes, D.C., 2020. Defending Against Malicious Reorgs in Tezos Proof-of-Stake, in: Proceedings of the 2nd ACM Conference on Advances in Financial Technologies. ACM, New York, NY, USA, pp. 46–58. https://doi.org/10.1145/3419614.3423265

Okereke Innocent, 2021. History Of Ethereum Hard Forks [WWW Document]. Medium.com. URL https://medium.com/xord/history-of-ethereum-hard-forks-3f3429fd1ba3 (accessed 1.10.24).

OpenZeppelin Docs, 2023. SafeMath Library [WWW Document]. OpenZeppelin Docs. URL https://docs.openzeppelin.com/contracts/2.x/api/math (accessed 1.10.24).

Palamarchuk Roman, 2023. Flash Loan Attacks: Risks & Prevention [WWW Document]. Hacken.io. URL https://hacken.io/discover/flash-loan-attacks/ (accessed 1.10.24).

Praitheeshan, P., Pan, L., Yu, J., Liu, J., Doss, R., 2019. Security Analysis Methods on Ethereum Smart Contract Vulnerabilities: A Survey.

Prasad, B., Ramachandram, S., 2022. Prevention and Detection Mechanisms for Re-Entrancy Attack and King of Ether Throne Attack for Ethereum Smart Contracts. Ingenierie des Systemes d'Information 27, 725–735. https://doi.org/10.18280/isi.270505

Prechtel, D., Gros, T., Muller, T., 2019. Evaluating Spread of ‘Gasless Send’ in Ethereum Smart Contracts, in: 2019 10th IFIP International Conference on New Technologies, Mobility and Security (NTMS). IEEE, pp. 1–6. https://doi.org/10.1109/NTMS.2019.8763848

Qin, K., Zhou, L., Gamito, P., Jovanovic, P., Gervais, A., 2021. An empirical study of DeFi liquidations: incentives, risks, and instabilities. https://doi.org/10.1145/3487552.3487811

Qin, K., Zhou, L., Livshits, B., Gervais, A., 2020. Attacking the DeFi Ecosystem with Flash Loans for Fun and Profit.

QuillHash Team, 2021. BurgerSwap Flash Loan Attack | Analysis [WWW Document]. Medium.com. URL https://quillhashteam.medium.com/burgerswap-flash-loan-attack-analysis-888b1911daef (accessed 1.10.24).

Reguerra Ezra, 2023. Attacker drains $800K from DeFi protocol Sturdy Finance [WWW Document]. Cointelegraph.

Rivest, R.L., Shamir, A., Adleman, L., 1978. A method for obtaining digital signatures and public-key cryptosystems. Commun ACM 21, 120–126. https://doi.org/10.1145/359340.359342

Rodler, M., Li, W., Karame, G.O., Davi, L., 2018. Sereum: Protecting Existing Smart Contracts Against Re-Entrancy Attacks.

Rosenfeld, M., 2011. Analysis of Bitcoin Pooled Mining Reward Systems.

Saad, M., Spaulding, J., Njilla, L., Kamhoua, C., Shetty, S., Nyang, D., Mohaisen, A., 2019. Exploring the Attack Surface of Blockchain: A Systematic Overview.

Sanda, O., Pavlidis, M., Seraj, S., Polatidis, N., 2023. Long-Range attack detection on permissionless blockchains using Deep Learning. Expert Syst Appl 218. https://doi.org/10.1016/j.eswa.2023.119606

Sapirshtein, A., Sompolinsky, Y., Zohar, A., 2015. Optimal Selfish Mining Strategies in Bitcoin.

Satochi Nakamoto, 2008. Bitcoin - Open source P2P money [WWW Document]. Bitcoin.org.

Sawtooth Developer Team, 2022. Sawtooth Dynamic Consensus [WWW Document]. URL https://sawtooth.hyperledger.org/docs/1.2/ (accessed 1.8.24).

Sayeed, S., Marco-Gisbert, H., 2019. Assessing Blockchain Consensus and Security Mechanisms against the 51% Attack. Applied Sciences 9, 1788. https://doi.org/10.3390/app9091788

Shier, C., Mehar, M.I., Giambattista, A., Gong, E., Fletcher, G., Sanayhie, R., Laskowski, M., Kim, H.M., 2017. Understanding a Revolutionary and Flawed Grand Experiment in Blockchain: The DAO Attack. SSRN Electronic Journal. https://doi.org/10.2139/ssrn.3014782

Shifferaw, Y., Lemma, S., 2021. LIMITATIONS OF PROOF OF STAKE ALGORITHM IN BLOCKCHAIN: A REVIEW, Journal of EEA.

Shiksha, 2023. Consensus Mechanisms in Blockchain [WWW Document].

Shin Laura, 2023. What Is Ethereum Sharding? A Beginner's Guide [WWW Document]. Coindesk.com. URL https://www.coindesk.com/learn/what-is-ethereum-sharding-a-beginners-guide/ (accessed 1.10.24).

Shrimali, B., Patel, H.B., 2022. Blockchain state-of-the-art: architecture, use cases, consensus, challenges and opportunities. Journal of King Saud University - Computer and Information Sciences 34, 6793–6807. https://doi.org/10.1016/j.jksuci.2021.08.005

Siren Protocol, 2021. SIREN Incident Report [WWW Document]. Siren. URL https://blog.siren.xyz/siren-incident-report-264e57f16d7 (accessed 1.10.24).

Slimcoin Development Team, n.d. The slimcoin project. [WWW Document]. 2014. URL https://github.com/slimcoin-project/Slimcoin (accessed 1.9.24).

Stucke, Z., Constantinides, T., Cartlidge, J., 2022. Simulation of Front-Running Attacks and Privacy Mitigations in Ethereum Blockchain, in: European Modeling and Simulation Symposium, EMSS. https://doi.org/10.46354/i3m.2022.emss.041

Sun, J., Huang, S., Zheng, C., Wang, T., Zong, C., Hui, Z., 2022. Mutation Testing for Integer Overflow in Ethereum Smart Contracts.

Tan, B., Mariano, B., Lahiri, S.K., Dillig, I., Feng, Y., 2022. SolType: Refinement types for arithmetic overflow in solidity. Proceedings of the ACM on Programming Languages 6. https://doi.org/10.1145/3498665

Tikhomirov, S., 2018. Ethereum: State of Knowledge and Research Perspectives. pp. 206–221. https://doi.org/10.1007/978-3-319-75650-9_14

Tikhomirov, S., Voskresenskaya, E., Ivanitskiy, I., Takhaviev, R., Marchenko, E., Alexandrov, Y., 2018. SmartCheck: Static Analysis of Ethereum Smart Contracts, in: 2018 IEEE/ACM 1st International Workshop on Emerging Trends in Software Engineering for Blockchain (WETSEB). pp. 9–16.

Valid.Network, 2020. How Did Lendf.Me Lose $25 Million to A Reentrancy Attack? [An Analysis] [WWW Document]. Hackernoon. URL https://hackernoon.com/how-did-lendfme-lose-dollar25-million-to-a-reentrancy-attack-an-analysis-091iy32s7 (accessed 1.10.24).

Vladislav Sopov, 2021. Cardano's (ADA) Smart Contract Environment Plutus Goes Live in Devnet [WWW Document]. U today.

Wan Xin, Adams Austin, 2023. Just-in-Time Liquidity on the Uniswap Protocol [WWW Document]. URL https://blog.uniswap.org/jit-liquidity (accessed 1.10.24).

Zhou, L., Qin, K., Torres, C.F., Le, D. V, Gervais, A., 2020. High-Frequency Trading on Decentralized On-Chain Exchanges.

Zhou, L., Xiong, X., Ernstberger, J., Chaliasos, S., Wang, Z., Wang, Y., Qin, K., Wattenhofer, R., Song, D., Gervais, A., 2022. SoK: Decentralized Finance (DeFi) Attacks.