A Hundred Attacks in Distributed Systems

ARASH VAEZI*, SARA AZARNOUSH*, and PARSA MOHAMMADIAN*, Sharif University of Technology,

The objective of any security system is the capacity to keep a secret. It is vital to keep the data secret when it is stored as well as when it is sent over a network. Nowadays, many people utilize the internet to access various resources, and several businesses employ a dispersed environment to give services to their users. As a result, a more secure distributed environment is required, in which all transactions and processes can be effectively completed safely. It is critical in a distributed system environment to deliver reliable services to users at any time and from any place. As an example of a distributed system, Blockchain is a unique distributed system that has confronted lots of attacks despite its security mechanism. Security is a top priority in a distributed setting. This paper organizes many attacks that byzantine users may apply to take advantage of the loyal users of a system. A wide range of previous articles dealt considered diverse types of attacks. However, we could not find a well-organized document that helps scientists consider different attacking aspects while designing a new distributed system. A hundred various kinds of most essential attacks are categorized and summarized in this article.

Additional Key Words and Phrases: Decentralized Systems, Attacks, Distributed Systems, Blockchain

*All authors contributed equally to this research.

Authors' address: Arash Vaezi, avaezi@ce.sharif.edu; Sara Azarnoush, sa.azarnoush@sharif.edu; Parsa Mohammadian, pmohammadian@ce.sharif.edu, Sharif University of Technology,

Contents

1 INTRODUCTION

In today's networked world, computers collaborate with each other for the purposes of communication, processing, data transfer, data storage, etc. Researchers in the literature have used a variety of definitions to describe what a distributed system is. A distributed system, according to Coulouris et al., is one in which the hardware and software components are installed on geographically dispersed computers that coordinate and collaborate on their actions by passing messages between them [30]. A distributed system, according to Tanenbaum and Van Steen, is a collection of systems that appear to users as a single system [8]. According to Tanenbaum's definition, a distributed system is a transparent system that tries to hide the complexities from its users. Combining these definitions, a distributed system is a middleware which connects with a variety of distributed hardware and software to coordinate the activities of several processes running on various platforms of computers over a communication network, so that all components cooperate to perform a set of related tasks aimed at a common goal [54]. The Blockchain is a well-known distributed system. A blockchain is a distributed data storage that is shared by computer network nodes. A blockchain, like a database, stores information electronically in digital format. Blockchains are best known for playing an important role in the cryptocurrency system.

Due to the wide range of applicability of Blockchain technology (BT), it has been gaining popularity in many domains. Bitcoin was the first cryptocurrency to employ blockchain technology, and it has now been utilized in a variety of different applications, including e-commerce, trade and business, manufacturing and production, finance, and gaming. BT employs a peer-to-peer system, which is a more decentralized approach to storing transactions and data records. Since there is not any single point of failure or third-party centralized control of transactions, BT distinguishes itself from other emerging technologies [96]. It employs a chain of blocks in which each block is locked using the hash of the preceding block to which it is linked, resulting in an immutable database of all transactions maintained as a digital ledger that cannot be altered without impacting all the blocks linked together in the chain [13]. A block is, in fact, a permanent store of records that, once written, cannot be altered or removed. Figure 1 illustrates a block structure [34].

Ethereum is a BT platform that can be used to write algorithms expressed in a general-purpose programming language, enabling developers to create tons of applications, from basic wallet apps to complicated financial systems for the banking sector. These programs, which are called Smart Contracts, are written in a Turing-complete byte-code language, called EVM byte-code[1].

EOSIO is a blockchain intended to support the running of a new kind of software known as a decentralized application (dapp). Its technology tries to address prior challenges with utilizing blockchains to host dapps, since popular applications have even blocked capacity on bigger, more mature blockchains like Ethereum (ETH), resulting in performance concerns for all users[2]. The EOSIO blockchain, one of the most prominent Delegated Proof-of-Stake implementations[3] (DPoS) blockchain platforms have grown rapidly recently. Meanwhile, a number of high-profile vulnerabilities and attacks targeting top EOSIO DApps and their smart contracts have been discovered and observed in the wild, resulting in significant financial losses. Because the majority of EOSIO smart contracts are not open-source and are often compiled to WebAssembly (Wasm) bytecode, it is difficult to examine and identify the existence of potential vulnerabilities.

Remark. We expect the reviewers of this article to be familiar with the overall concept of distributed and blockchain-based systems. Most of the attacks mentioned in this paper consider a blockchain system as a target.

[1] "Ethereum home page," Accessed on 01-10-2019. [Online]. Available:https://www.ethereum.org/
[2] https://www.kraken.com/learn/what-is-eosio
[3] Proof-of-Stake is a cryptocurrency consensus mechanism for processing transactions and creating new blocks in a blockchain. See https://www.investopedia.com/terms/p/proof-stake-pos.asp for more details.

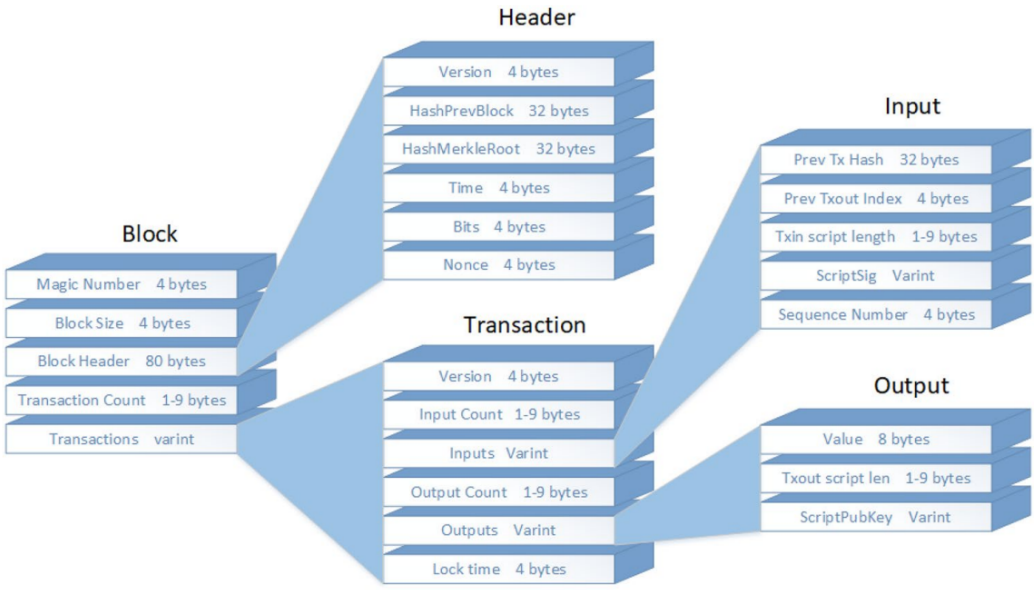

Fig. 1. The structure of a block in a blockchain system is illustrated. The figure is designed by [34].

2 ATTACKS AND VULNERABILITIES

Because of the substantial economic potential, decentralized systems have become primary targets for attackers. This section covers a summary of the concepts of a large variety of different types of attacks.

2.1 Service

2.1.1 Attack 1: Double-Spending. The double-spending problem refers to a single money unit being spent more than once. This results in a discrepancy between spending history and accessible currency[4].

In other words, double spending happens when a particular node announces the same transaction with a different destination to two different nodes. Since these two nodes are not necessarily in sync, they both add the transaction to the ledger.

In practice, networks overcome this attack by introducing a consensus algorithm that decides which transaction to be included in the main ledger (for example, by accepting the longest chain). In this case, users should wait for confirmation to be confident about a transaction [34].

The former introduced method can only detect double-spending; in addition, we can use tamper-resistant hardware to prevent double-spending rather than just detect it. These devices are usually called wallets and keep track of the balance in a tamper-free manner [81].

Another way to avoid double-spending is using green addresses. Green addresses are some known third parties that provide financial services [81].

[4] https://www.sofi.com/learn/content/double-spending/

Articles related to this attack [28, 31, 34, 37, 59, 61, 79, 90].

2.1.2 Attack 2: Race. The race attack allows for a situation in which an attacker creates two transactions, one genuine and one fraudulent. The target is any node that accepts transactions with 0-unconfirmed status, meaning the transaction is visible but not included in any block yet. The attacker connects directly to the target as a network peer. The attacker also tries to connect closely or directly to a mining pool. By sending the fraudulent transaction to the target and the legitimate transaction to the mining pool, the attack may succeed if the target accepts the fraudulent transaction and provides some goods or services before seeing the legitimate transaction. This is why it is advised to wait for a minimum number of confirmations before considering a transaction valid [34].

Articles related to this attack [34, 88].

2.1.3 Attack 3: Finney. An attacker pre-mined one transaction into a block and spent the same coins before releasing the block to the public network to invalidate that transaction. This is called a Finney attack. The Finney attack is a fraudulent double-spend that requires the participation of a miner once a block has been mined. An adversary can only perform a double-spending (see Section 2.1.1) in the presence of one-confirmation vendors [88].

Articles related to this attack [5, 16, 34, 88].

2.1.4 Attack 4: Vector76. Vector76 is a variation of double-spending (see Section 2.1.1). The name came from the username that introduced this attack in the Bitcoin talk forum[5]. The targets of this attack are usually exchanging websites. In this attack, the attacker first announces a relatively significant transaction to the network; after transaction validation in a block, the attacker withdraws the currency from the destination account. Then the attacker takes advantage of well-positioned nodes (which he must identify before the attack) and sends them a relatively small transaction, which will be validated and invalidate the first transaction [60].

Articles related to this attack [60].

2.1.5 Attack 5: Nothing-At-Stake. The nothing-at-stake issue is a security hole in proof-of-stake systems. The problem can arise whenever there is a fork in the blockchain, which means two honest validators propose blocks simultaneously, whether maliciously or accidently[6].

When a fork arises, it is in everyone's self-interest to continue mining both chains. This is for two reasons:

(1) Because mining is without cost, mining both chains does not affect the miner's bottom line.
(2) If miners continue mining only one fork while one of the other forks becomes longer, the miners will receive no benefit from the time spent mining the shorter chain. In other words, mining all forks ensures that the miner receives their payment regardless of which fork is successful.

This may make double-spending (see Section 2.1.1) more plausible. An attacker wishing to perform a double-spending attack might do so by forking the blockchain one block before spending the coins. If the attacker mines just their fork while the other miners mine both forks, the attacker's fork will ultimately become the longest chain, even if the attacker has a small stake in the network.

Articles related to this attack [16].

2.1.6 Attack 6: Goldfinger. In a market setting where shorting is brutal, 51% (see Section 2.15.2) attacks on the blockchain are often a negative approach, as the miner is also attempting to undermine the value of the asset he is attempting to

[5] https://bitcointalk.org/index.php?topic=36788.msg463391#msg463391
[6] ethereum/wiki. GitHub. https://github.com/ethereum/wiki/wiki/Problems

purchase or double-spend. It is more profitable to operate ethically and merely race to verify blocks. This alignment of incentives between miners, holders, and spenders is one of the factors that contribute to Bitcoin functioning better in practice than in theory. Users may now take a position in the derivatives and futures markets that rewards them for attacking the blockchain. This is referred to as the Goldfinger attackfootnote[7].

To see more information on this attack see the web address in the footnote.

2.1.7 Attack 7: Refusal To Sign. A malicious agent may choose not to sign a transaction that is not advantageous to him. While avoiding this attack is impossible, punitive actions against the refusal agents are available [44].

Articles related to this attack [44].

2.2 Network

2.2.1 Attack 8: Routing. Routing attacks include traffic route diversion, hijacking, and denial-of-service (DoS) attacks. (see section 2.4.2). Besides simple data eavesdropping or modification, these attacks may lead to network partitioning, which in turn raises the risks of 51% attacks (see section 2.6.2) or selfish mining attacks(see section 2.15.2). Countermeasures include multi-homing nodes (or the use of VPNs) for route variety, as well as selecting additional peers whose connections do not run via the same autonomous systems (ASes), preference of peers hosted on the same AS within the same /24 prefix (to reduce risk of partitions), and fetching the same block from multiple peers [12]. Another mitigation is SABRE [11], a secure relay network that runs alongside the Bitcoin network. The BGPsec [68] is a security extension for BGP used between neighboring ASes, and it provides assurance of route origin and propagation by cryptographic verification [58].

Articles related to this attack [11, 12, 58, 68].

2.2.2 Attack 9: BGP Hijack. BGP hijacking is sending traffic to a different destination than the real intended one to intercept the packets. Again, the purpose is to steal user credential by phishing. It can also be used to show host unwanted ads in a webpage.

Articles related to this attack [1].

2.2.3 Attack 10: Module-Enabling. Go-version Ethereum client (geth)[8] does not provide APIs to dynamically enable modules while providing the Remote Procedure Call (RPC) / Web Socket (WS) service. Modules can be enabled by stopping and then restarting RPC/WS, on the condition that the geth provides RPC/WS service with the admin module enabled. Without identity authentication, anyone, including the adversary, can enable any module once the geth node provides RPC/WS service and enables the admin module [118].

Articles related to this attack [118].

2.2.4 Attack 11: Piracy. Piracy attack refers to the attack in, the adversary signs data or sends transactions through Remote Procedure Call (RPC) or Web Socket (WS) services in the name of the target accounts after the adversary remotely unlocks accounts or discovers unlocked accounts. This attack can be fulfilled with three conditions, i.e., the adversaries have access to unlocked accounts; the geth (see subsection 2.2.3) node provides RPC/WS service and enables the eth module; geth node has the target user's key file.

Articles related to this attack [118].

[7] https://soundcrypto.com/blog/2017/11/16/crypto-options-and-goldfinger-attacks
[8] https://github.com/ethereum/go-ethereum

2.2.5 Attack 12: Steal Mining Reward. In Ethereum, a reward is given to the coinbase specified by the block miner, which can be an arbitrary Ethereum account. This attack can be fulfilled in the case that the geth (see subsection 2.2.3) provides Remote Procedure Call (RPC) or Web Socket (WS) service and enables the miner module. Adversaries call the miner setEtherbase to change the coinbase to a designated address. The designated address is used as the receiver of the mining reward and does not need the private key. After setting the coinbase, adversaries can start the block mining by calling the miner start function on the target node to earn rewards [118].

Articles related to this attack [118].

2.3 DNS

2.3.1 Attack 13: DNS. DNS attacks are frequently the result of cache poisoning [102] (see section 2.3.4), which affects not only nodes that use DNS bootstrapping [9] to connect to online peers but also users of online blockchain explorers.One countermeasure is DNSSEC, a DNS security extension that offers authentication and data integrity. In addition to standard DNS, name resolution can also be made using alternate DNS servers [58]. This attack can block new nodes from participating in blockchain by tampering with DNS. A nice gathering of information about the DNS attacks are mentioned in the nxlog.co[10]. Security was not even a consideration when the Domain Name System was designed. Nefarious actors are now utilizing DNS for data theft, distributed denial-of-service assaults, command-and-control, and other malicious activities. According to an EfficientIP survey conducted in 2018, the average cost of a DNS assault in 2018[11] was $715,000 (57 percent more than in 2017), and 77% of firms experienced a DNS attack.

Articles related to DNS attacks [58, 102].

2.3.2 Attack 14: DNS Hijacking. DNS hijacking can be accomplished by influencing user workstations or DNS servers. In the first scenario, malware is used to change the name servers configured on a workstation, causing DNS requests to be sent to hostile servers instead. On the other hand, a DNS server may be hacked and modified to send out erroneous responses. Users are routed to an attacker's site in either case, which allows the attacker to steal user credentials (phishing), generate traffic (pharming), transmit malware, or publish a defaced version of the website.

In early 2019, FireEye[12] discovered an "unprecedented scale" DNS hijacking effort that had a "high degree of success," targeting victims worldwide. Soon after, the US Cybersecurity and Infrastructure Security Agency issued Emergency Directive 19-01[13], which details steps to prevent DNS hijacking, including DNS audits and monitoring. The recent Sea Turtle[14] campaign is a state-sponsored DNS hijacking attack that used man-in-the-middle attacks to capture user credentials from at least 40 different companies in 13 different countries[15].

To see more information on this attack, see the web address in the footnote[16].

2.3.3 Attack 15: DNS tunneling. DNS queries and responses can include data payloads capable of carrying malware, unauthorized access, Command and control information, or bidirectional protocols like SSH. DNS traffic is frequently regarded as safe and unmonitored. This means that an attacker could use DNS tunneling in order to communicate secretly.

[9] https://bitcoinj.github.io/-model
[10] https://nxlog.co/whitepapers/dns-logging
[11] https://www.efficientip.com/resources/dns-threat-report-2018/, EfficientIP, retrieved 2019-05-14
[12] https://www.mandiant.com/resources/global-dns-hijacking-campaign-dns-record-manipulation-at-scale, FireEye, Threat Research, retrieved 2019-05-14
[13] https://cyber.dhs.gov/ed/19-01/
[14] https://blog.talosintelligence.com/2019/04/seaturtle.html, Cisco Talos Intelligence Group, Talos Blog, retrieved 2019-05-14
[15] https://nxlog.co/whitepapers/dns-logging
[16] https://encyclopedi.kaspersky.com/glossary/dns-hijacking/

Point of sale (PoS) malware such as MULTIGRAIN is among the many types of client malware that use DNS tunneling[17], remote backdoors such as DNSMessenger[18] and DNSpionage [19]. Heyoka, dnscat2, and iodine are several implemented examples of open-source DNS tunnels.

To see more information on this attack, see the web addresses in the footnote.

2.3.4 Attack 16: DNS Cache Poisoning. When a DNS resolver accepts an invalid resource record as a result of a vulnerability, this is referred to as cache poisoning (or spoofing). An intruder could use a long TTL (time-to-live) number, which causes the data to be retained in the resolver's cache and the resolver to be regarded "poisoned." The effect is comparable to DNS hijacking. Cryptographic signatures, like those introduced by the Domain Name System Security Extensions (DNSSEC), prevent DNS cache poisoning. However DNSSEC is not yet widely deployed.

To see more information on this attack, see the web addresses in the footnote.

2.4 DoS

2.4.1 Attack 17: DDoS. DDoS attacks are not unique to blockchain networks, but they can be used to target blockchain and asset exchange networks with certain additional modifications. In this type of attack, an attacker uses a network of hijacked devices to flood a network with a large number of requests, causing the network's capacity to handle legitimate traffic to be harmed. At its inception, Bitcoin Gold, one of the forks of Bitcoin, was subjected to a massive DDoS attack, receiving 10 million false requests per minute[20] [34].

To see more information on this attack see the web address in the footnote. Articles related to this attack [16, 34, 117].

2.4.2 Attack 18: DoS. Adversaries can leverage the private key protection mechanism to perform a Denial-of-Service (DoS) attack by widening the brute force attack. The scrypt algorithm, which is employed to safeguard the private key, requires a lot of memory and processing power. Adversaries can send unlocking requests continuously and concurrently to delay or stop other attempts from benign users. The geth (see subsection 2.2.3) node must provide the RPC/WS service and enable the individual module for this attack to work. It takes a long time to unlock the account. With the number of unlocking requests, the time consumption grows linearly. A single unlocking request takes around a second on the MacBook Pro. The result of a DoS attack is that the attacker calls the personal unlock account function 40 times in a row. The geth instance barely used roughly 65MB of memory and 0.1 percent of the CPU before the DoS attack. The geth uses roughly 7.8GB of memory and 213 CPU cycles during the DoS attack (more than two cores). The time it takes to unlock an account has been considerably increased, from seconds to minutes or longer [118].

Articles related to this attack [118].

2.4.3 Attack 19: DoS Attacks On Connectivity. DoS attacks on the connectivity of nodes may lead to loss of consensus power, thus preventing consensus nodes from being rewarded[21]. For validating nodes, this attack leads to a disruption of some blockchain-dependent services[22]. Countermeasures: One mitigation is to peer only with white-listed nodes. Methods to prevent volumetric DDoS, include on-premise filtering (i.e., with an extra network device), cloud filtering (i.e., redirection of traffic through a cloud when DDoS is detected or through a cloud DDoS mitigation service), or hybrid filtering [57]. Articles related to this attack [57].

[17] https://www.fireeye.com/blog/threat-research/2016/04/multigrain_pointo.html, FireEye, Threat Research,retrieved 2019-05-14
[18] https://trust.titanhq.com/acton/media/31047/webtitan-web-filter-up-21-01-2022-form
[19] https://blog.talosintelligence.com/2018/11/dnspionage-campaign-targets-middle-east.html
[20] https://assignmenthelp4me.com/article-advantages-and-disadvantages-of-cybersecurity-342.html
[21] https://www.coindesk.com/markets/2015/03/12/bitcoin-mining-pools-targeted-in-wave-of-ddos-attacks/
[22] https://news.bitcoin.com/ddos-attacks-bitcoin-com-uncensored-information/

2.4.4 Attack 20: DoS Attacks On Local Resources. DoS attacks on local resources, including memory and storage, have the potential to degrade nodes peering and consensus capabilities[23]. An example attack is flooding the network with low fee transactions (a.k.a., penny-flooding), which may cause memory pool depletion, resulting in a system crash. A possible mitigation is raising the minimum transaction fee and rate limit to the number of transactions. Several mitigating techniques are applied to Bitcoin[24] nodes including scoring DoS attacks and banning misbehaving peers[25] [58].

For more information related to these attacks see [58], and the web addresses in the footnote.

2.4.5 Attack 21: Smart Contract DoS. A Denial of Service (DoS) attack occurs when an attempt is made to interfere with a service so that it becomes less or unavailable. Simply put, normal service requests that the system cannot process a user needs. For example, it is a DoS when a computer system is unable to provide regular service because of a lack of bandwidth or a full hard drive.

On the Internet, DoS attacks can be roughly divided into three categories: the use of software implementation defects Exploiting loopholes in the protocol; Use of resources to suppress. In the blockchain, DoS attacks disrupt, suspend, or freeze the execution of a normal contract, or even the logic of the contract itself[26].

A DoS vulnerability can be understood as "unrecoverable malicious manipulation, or uncontrolled, unlimited resource consumption," i.e., a DoS attack on an Ethereum contract that could result in massive consumption of Ether and Gas and, worse, render the original contract code logic unworkable.

Ethereum Smart Contracts are programs deployed as decentralized applications, using the building blocks of the blockchain consensus protocol. Thanks to this technology, consumers can reach agreements in a transparent and conflict-free environment. Security flaws in these smart contracts pose a risk to applications and their users, as has been demonstrated in the past, and they have the potential for huge losses. Denial of Service vulnerabilities in Ethereum Smart Contracts can be detected using a framework that is a combination of static and dynamic analysis. Noama Fatima Samreen proposed this framework, Manar H. Alalf [94].

Articles related to this attack [94].

2.4.6 Attack 22: Resource Exhaustion. Computer security exploits that cause a program or system to crash, freeze, or otherwise interfere with the intended target are known as resource exhaustion attacks. Even though they are denial-of service attacks, they are not the same as distributed denial of service attacks, which aim to take down an entire network host, for example, by flooding it with requests from various places simultaneously[71].

Articles related to this attack [71].

2.5 Gas

2.5.1 Attack 23: Gas Limit Block Stuffing. Block stuffing is a blockchain attack in which an attacker sends transactions that purposefully exceed the block's gas limit, causing other transactions to halt. The attacker might pay more outstanding transaction fees to guarantee that their transactions are included by miners. The attacker can impact the number of transactions included in the block by influencing how much gas is used by their transactions. [27].

A Block Stuffing attack can be used against any contract that requires a specific action to be done within a specified duration. However, like with any attack, it is only profitable if the predicted benefit outweighs the cost. The cost of this

[23] https://bitcointalk.org/index.php?
[24] https://en.bitcoin.it/wiki/Weaknesses#
[25] https://blog.radware.com/20security/2018/04/choosing-the-right-ddos-solution-hybrid-protection/
[26] https://medium.com/@Knownsec_Blockchain_Lab/in-depth-understanding-of-denial-of-service-vulnerabilities-dd437b1d7a1c
[27] https://solmaz.io/2018/10/18/anatomy-block-stuffing/

assault is related to the number of blocks that must be filled. If a substantial payoff can be acquired by stopping other parties from acting, the contract will almost certainly be attacked[28].

To see more information on this attack see the web address in the footnote.

2.5.2 Attack 24: Forcible Balance Transfer. Without a fallback method, coercive balance transfers to the contract can occur in insecure smart contract scripts. This can be used to deplete the gas supply and prevent the final transaction from taking place [93].

Articles related to this attack [93].

2.6 Adversarial Centralization of Consensus Power

A design premise about the decentralized distribution of consensus power is violated in these attacks. 51% attack 2.6.2 for PoR and PoS protocols, as well as $\frac{1}{3}$ of Byzantine nodes 2.6.3 for BFT protocols, are examples of this category (and their combinations). Whenever an attacker has a majority of the consensus power, they influence the protocol's outcome [57].

2.6.1 Attack 25: Byzantine. Byzantine attackers submit "fake reports" in order to disrupt the system's usual report aggregation process. For example, a Byzantine attacker could report a very high passing time for a particular road section, causing the aggregated passing time to be much higher than it is, forcing other users to utilize alternative routes while they speed through the road. A Byzantine attacker, on the other hand, can report a a meager passing time for a road section, leading other users to utilize a road segment that is already congested, inflating traffic congestion [116].

A quorum of 13 adversarial consensus nodes could cause the protocol to be degraded or even terminated in Byzantine attacks. It is critical to foster decentralization through incentive systems that reward honest participation while discouraging [75] or punishing protocol [21, 33] violations as a design-oriented countermeasure [57].

Articles related to this attack [21, 33, 57, 75, 116].

2.6.2 Attack 26: Collusion 51%. A 51% assault is a type of consensus algorithm attack carried out by a group of miners with more than 50% of the network's mining hash rate or computer power. New transactions can be blocked, and transactions between certain users or all users can be halted. They'd also be able to undo previously completed transactions that hadn't been confirmed, allowing them to double-spend(see 2.1.1). The attacker is unable to create new coins or change existing blocks. So, even if a 51% attack proved severely devastating, it is unlikely that Bitcoin or another blockchain-based money would be wrecked.

In August 2016, two Ethereum-based blockchains, Krypton and Shift, were subjected to a 51% attack. Bitcoin Gold, the26^{th}-largest cryptocurrency at the moment, was hit by a 51% attack in May of 2018. The attackers had sufficient control over Bitcoin Gold's hash power that they could double-spend for several days despite Bitcoin Gold's repeated attempts to raise the exchange thresholds, finally stealing more than eighteen million dollars in Bitcoin Gold. In the year 2020, Bitcoin Gold was hit once more. The Bitcoin SV (BSV) network was recently attacked in August 2021[29].

The article [97] outlined a solution for countering the 51% attack. They talked about the five most advanced defense tactics for preventing this attack, as well as their significant drawbacks. They also found that, in most circumstances, security methods fail to provide effective protection against the 51% assault since the flaws are passed down from

[28] https://bui-duc-huy.github.io/posts/smart-contract-best-practice-3/
[29] https://www.investopedia.com/terms/1/51-attack.asp

consensus protocols. Similarly, n confirmation and selfish mining 2.15.2 are two attack approaches similar to the 51 percent assault tactic.

Articles related to this attack [5, 59, 97].

2.6.3 Attack 27: Consensus 34%. The BFT network was targeted for a 34% Consensus Majority Attack, which is a type of Consensus Majority Attack[30]. It is a type of potential attack on a blockchain that employs the "Tangle" Consensus method, in which an individual or organization gains control of at least 34% of the overall network mining power (hash rate) and then manipulates the general ledger to approve or disapprove selected transactions via majority approval[31].

To see more information on this attack see the web address in the footnote.

2.6.4 Attack 28: Punitive And Feather Forking. The purpose of punitive forking is to blacklist or censor Bitcoin addresses held by specific persons (e.g., those who do not pay a fee) so that they cannot spend any of their bitcoins.

When the attacker possesses the majority of the hash power, this works. The attacker declares that any chain containing blacklisted transactions will not be extended and if blocks containing such transactions do emerge, the attacker forks and establishes a longer chain. In feather forking, the attacker declares a similar purpose but also says that after a while, they will give up trying to fork (say falling k confirmations behind the main chain).

Other miners are nonetheless motivated to block blacklisted transactions because doing so increases the chances of losing their reward. An attacker who performs feather forking can also use it to blackmail a customer by threatening to ban all of her transactions unless she pays the ransom coins demanded [29].

Articles related to this attack [29, 57]. To see more information on this attack see the web address in the footnote.[32]

2.6.5 Attack 29: Bribery. One of the most basic and widespread concerns is double-spending. Even if they don't have a lot of hash power, attackers can bribe other miners to break the consensus agreement and raise the chances of double-spending. Bribery is the term for this type of attack. The lack of systematic quantitative methodologies makes evaluating and comparing bribery attack models difficult. The costs and advantages of attackers, in particular, are rarely evaluated and are modified by a variety of factors [105]. Bribery attacks, in contrast to feather forking attack(), include the provision of direct rewards to miners, whereas feather forking involves adversaries attempting to influence miners' behavior by threatening to harm their profits [57].

Articles related to this attack [19, 39, 40, 57, 105].

2.6.6 Attack 30: Consensus Delay. A consensus delay attack is another approach to extending the transaction's accepting time [50]. While a block's latency is increased by the memory pool due to the transactions queue, consensus delay is caused by the majority of users' propagation time. When a new block is available, it should be confirmed by the most of blockchain participants [42].

Articles related to this attack [42, 50].

2.7 Time

2.7.1 Attack 31: Timejacking. Computers and machinery separated by great distances must usually be synced to work together. If the time and date are not synced, problems with security, usability, and overall reaction time can arise.

[30] https://cloudsecurityalliance.org/blog/2020/10/26/blockchain-attacks-vulnerabilities-and-weaknesses
[31] https://www.coitok.com/34-attack-407
[32] https://bitcointalk.org/index.php?topic=312668.0

Time should continue to advance even if users switch from one computer to another if they have communicative programs operating on various systems. If one system is forward of the others, the rest are behind it. Switching between these systems would cause the time to leap forward and back, which would be undesirable from the standpoint of an outside observer. As a result, isolated networks may run on schedule, but the consequences will be obvious as soon as they connect to the Internet. When time goes backward, even on a single machine, some apps experience issues.

An attacker can manipulate a node's network time counter and trick it into accepting another blockchain by broadcasting incorrect timestamps while connecting to it. This could raise the likelihood of a successful double-spend, deplete a node's processing resources, or reduce transaction confirmation rates.

The Bitcoin network's nodes keep an internal counter that indicates network time. During the bootstrapping phase, the node requests network time from surrounding nodes calculate the median, and stores it. If the median is longer than 70 minutes, the system time will be used. An eclipse attack, for example, would continue to allow surrounding nodes to submit incorrect timestamps. Because their timestamp exceeds the network timestamp by 120 minutes, such an attack could cause the node to reject the blocks. Such a targeted node will eventually be disconnected from the network [46, 62].

Articles related to this attack [16, 46, 62].

2.7.2 Attack 32: Time-Validation. In PoW and PoS, nodes keep network time, which is computed as the median value of the time collected from peers. Time-Validation attacks, also known as time de-Synchronization attacks, usually keep network time in addition to system time. This time is frequently included in the block header, and when nodes receive a block, they check to see if it meets the freshness requirements. An attacker can take advantage of this method by joining a large number of nodes and propagating inaccurate timestamps, which can cause the target node's network time to be slowed or sped up[33]. Due to freshness limits, when such a desynchronized node creates a block, the network may discard it. A node can construct a reputation list of trusted peers or use a timestamping authority to avoid de-synchronization attacks [58, 108].

Articles related to this attack [58, 108].

2.7.3 Attack 33: Time-Spoofing. Time-spoofing attacks aim to reduce the complexity of the problem and hence the effort required to earn the same reward by targeting a time-based difficulty computation mechanism in a PoR protocol. The attacker is a consensus node that mines blocks with delayed timestamps, indicating that the puzzle is too complex to satisfy the block generation rate and that the difficulty should be reduced. Countermeasures: To improve the accuracy of the timestamps, a solution that incorporates partial solutions obtained by all nodes into an averaged timestamp computation may be used [109]. It's worth noting that a time spoofing attack might have a significant impact on the application layer, particularly in use cases that rely on timestamp accuracy [57].

Articles related to this attack [57, 109].

2.8 Iterative False Generation

2.8.1 Attack 34: Sybil. The attacker subverts the service's recommendation system by generating a large number of pseudonymous identities and leveraging them to gain disproportionately large influence. Sybil, the subject of the book Sybil, a case study of a woman with dissociative identity disorder, inspired this attack [34]. In other words, a Sybil attack is described as a small number of individuals impersonating several peer identities to compromise a disproportionately

[33] http://culubas.blogspot.com/2011/05/timejacking-bitcoin_802.html
[34] https://www.npr.org/2011/10/20/141514464/real-sybil-admits-multiple-personalities-were-fake

large amount of the system. In other words, an adversary tries to make a large number of nodeIds appear and function as distinct nodes, which may or may not be generated randomly. The adversary can get closer to a specific object or a group of objects in the P2P overlay by using numerous identities. It improves the ability to intercept message routing and operate an overlay network. It's even possible for a bad actor to take over the P2P overlay network [20].

Articles related to this attack [16, 20].

2.8.2 Attack 35: Spam. By slowing the network and delaying block formation, a spam attack affects a committed transaction [16].

Articles related to this attack [16, 80, 91].

2.8.3 Attack 36: Peer Flooding. An attacker can cause actual nodes to slow down or become unresponsive as they seek to connect to the freshly announced false peers by generating a large number of fake peers in a network (peer to peer or otherwise)[35]

To see more information on this attack see the web address in the footnote.

2.8.4 Attack 37: Peer Flooding Attack Slowloris Variant. An attacker might cause actual nodes to slow down or become unresponsive when they attempt to connect to the newly announced peers by generating a large number of sluggish peers (natural systems that reply very slowly to network requests) in a network. Slowloris peers, unlike fake peers, are actual but communicate slowly enough to keep sockets and resources available for minutes or hours[36]. To see more information on this attack see the web address in the footnote.

2.9 Disconnecting Node

2.9.1 Attack 38: Eclipse. Eclipse attacks are a form of attack in which the attacker monopolizes all of the victim's incoming and outgoing connections to isolate the target from the rest of the network. This allows the attacker to tamper with the target's blockchain view, force it to waste compute power or use the target's compute power for malicious purposes [34, 56].

Articles related to this attack [16, 34, 56]

2.9.2 Attack 39: Triangle. This attack occurs when three nodes are in a triangular position, and only one can reach out to other nodes in a distributed network such as blockchain. Figure 2 illustrates this attack with three nodes denoted by A, B, and C and rectangle as a schema of a distributed network. In such a situation, C has complete control over transactions between A and B.

Articles related to this attack [119].

2.9.3 Attack 40: Partitioning. A partition attack aims to isolate a group of nodes from the network entirely. The attacker must reroute and cut all connections between the set of nodes and the rest of the network to do this [12].

Articles related to this attack [12].

2.9.4 Attack 41: Balance. It enables a low-mining-power attacker to interrupt communications between subgroups with equivalent mining power for a short period. They represent blockchain as a DAG (Directed Acyclic Graph) tree, with DAG = B, P >. B are the nodes that represent the information in the blocks, and they are connected by directed edges P. The attacker issues transactions in one subgroup (called "transaction subgroup") and mines blocks in another

[35] https://cloudsecurityalliance.org/blog/2020/10/26/blockchain-attacks-vulnerabilities-and-weaknesses/
[36] https://cloudsecurityalliance.org/blog/2020/10/26/blockchain-attacks-vulnerabilities-and-weaknesses/

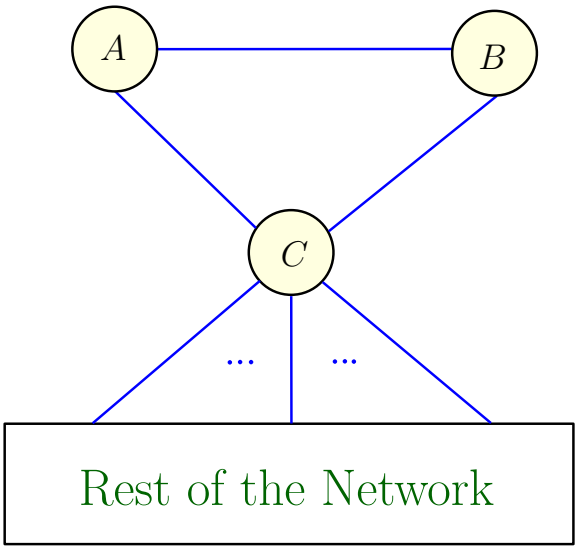

Fig. 2. Triangle Attack on Blockchain.

subgroup (called "block subgroup") after introducing a delay between correct subgroups of equivalent mining power. This ensures that the tree of block subgroup outweighs the tree of transaction subgroup. Even though the transactions have been committed, the attacker can outweigh the transaction tree and alter blocks with a high probability.

The balanced attack breaks the main branch prefix's persistence and allows duplicate spending. The attacker must first identify the merchants who are active in the subgroup and then create transactions to purchase goods from them. The attacker then sends transactions to this subgroup and propagates the mined blocks to the group's remaining nodes. The attacker stops delaying messages as long as the merchant ships items. The attacker could successfully reprint another transaction using the same coins if the DAG tree seen by the merchant is outweighed by another tree.

The balanced attack demonstrates that a PoW-based blockchain is blocked agnostic. When a transaction is written to the main chain, there is a chance that the attacker will be able to override or remove the block that contains the transaction. In a separate experiment, the authors set up an Ethereum private chain with R3 consortium-like characteristics[37] and demonstrated that they could successfully carry out a balancing attack that requires just approximately 5% of total computer power [70].

Articles related to this attack [70].

2.10 Change Block

2.10.1 Attack 42: Tampering. An unintentional but unlawful act that results in the change of a system, its components, its intended behavior, or its data[38]. Attackers can interfere with block delivery [50], in a blockchain network, but the ledger itself is impenetrable.

Articles related to this attack [50].

[37] https://www.r3.com/
[38] https://csrc.nist.gov/glossary/term/tampering

2.10.2 Attack 43: Modification. Modification attacks may be classified as integrity attacks; however, they might equally be called availability attacks. They have ruined the integrity of the data contained in a file if they get unauthorized access to it and alter the data it contains. However, if the file in question is a configuration file that controls how a particular service behaves, such as a Web server, changing the contents of the file may impact the service's availability. Continuing with this approach and the configuration changed in the file for the Web server affects how the server handles encrypted connections. This could even be considered a confidentiality attack [7].

Articles related to this attack [2, 7, 66, 122].

2.10.3 Attack 44: Transaction Malleability. The transaction ID is modified after it is made but before it is mined in the blockchain network in a transaction malleability attack. The source and destination addresses, and the transaction amount, cannot be changed. Still, other aspects of the transaction can result in a transaction ID (TXID) different from the original. Typically, the goal is to duplicate the original transaction that could not be mined first using this actual malleable transaction [10]. Double-spending can occur as a result of a successful transaction malleability attack [34, 63].

Articles related to this attack [10, 34, 63].

2.11 History

2.11.1 Attack 45: Grinding. If the leader or committee responsible for producing a block is known before the round begins, the attacker can manipulate the process to improve her chances of being chosen in the future. For instance, if a PoS protocol uses the hash of the previous block to determine the risk. If an attacker gathers enough keys with sufficient stake in the past, he can restart the consensus protocol and alter the blockchain's history. [57].

Articles related to this attack [57].

2.11.2 Attack 46: Keeping Secrets Exploit. Fields in contracts can be public (i.e., everyone can see them) or private (i.e., only other users/contracts can see them). Even yet, designating a private field does not ensure its privacy. Users must transmit a valid transaction to miners, who will subsequently broadcast it on the blockchain to set the value of a field. Because the blockchain is open to the public, anyone may examine the contents of the transaction and infer the field's new value. As a result, an attacker can readily exploit confidential data.

In such circumstances, the contract can use appropriate cryptographic techniques, such as timed commitments, to ensure that a field remains secret until a specific event occurs [13].

Articles related to this attack [9, 13, 18].

2.11.3 Attack 47: Front-running. Front-running is an attack where a compromised node sees a transaction after it has been broadcast but before it has been finished and tries to have its own transaction verified before or instead of the observed transaction [41]. Any type of distributed network can be used for front-running.

Articles related to this attack [41].

2.11.4 Attack 48: Long Range. A Long-Range attack is one in which the attacker returns to the genesis block and forks the network. The new branch has a history that is either partially or entirely separate from the main chain. The attack succeeds when the adversary's manufactured branch is longer than the main chain, and therefore it overtakes it [35].

Articles related to this attack [16, 35, 57].

2.11.5 Attack 49: Stake Bleeding. When using the proof of stake consensus algorithm, an adversary in a forking chain can accumulate the rewards associated with the creation of new blocks (which are initially created by honest users in the main chain) in order to inflate its stake until it reaches a sufficient level to take over the network [14].

Articles related to this attack [49].

2.11.6 Attack 50: Alternative History. An alternate history attack, also known as a blockchain reorganization attack, can occur even with numerous confirmations. However, it needs a significant amount of computing power on the part of the hacker. A malevolent user in this example sends a transaction to a recipient while simultaneously mining an alternative fork with another transaction that yields the same currency. Even if the recipient accepts the transaction after n confirmation and delivers a product, the recipient may lose a lot of money if the attacker publishes a longer chain and recovers the coins. In August 2020, one of the most recent blockchain rearrangement attacks occurred when a miner employed obsolete software and lost internet access for some time while mining Ethereum Classic. When two versions of the blockchain contended for legitimacy from network nodes, a restructuring occurred, resulting in around a 3000-block insertion[39].

To see more information on this attack see the web address in the footnote.

2.12 Identity

2.12.1 Attack 51: Replay. A replay attack happens when an intruder takes a packet from the network and sends it to a service or application as though the intruder was the user who provided the packet in the first place. When the packet is an authentication packet, the intruder can use the replay attack to authenticate on another person's behalf and thereby access that person's resources or data [3].

A Replay Attack does not imply that someone else has access to money. Only a Replay Attack may replicate an existing transaction from the new split blockchain and duplicate it on the old blockchain (or the other way around)[40][41].

Articles related to this attack [3, 24].

2.12.2 Attack 52: Impersonation. An attacker attempts to impersonate a genuine user to carry out unlawful actions.

Articles related to this attack [34, 44, 115].

2.12.3 Attack 53: Identity Revealing. Identity-revealing attacks are carried out by associating a node's IP address with the identity conveyed in transactions[42] [17, 76]. The use of Sybil listeners to analyze traffic can show the relationship between node IP addresses and transactions[43]. VPNs and anonymization services, such as Tor, are examples of countermeasures [58].

Articles related to this attack [58].

2.12.4 Attack 54: Deanonymization. An adversary tries to deduce the identity of a specific individual from a set of mobility traces in a de-anonymization attack. More exactly, assume that throughout the training phase, the opponent watched the actions of select individuals for a significant period (e.g., many days or weeks). Later, the adversary has access to a second geolocated dataset that contains the mobility traces of some of the people seen during the training phase, as well as maybe some unknown people. The adversary's goal is to de-anonymize this dataset (dubbed the testing

[39] https://www.apriorit.com/dev-blog/578-blockchain-attack-vectors
[40] https://support.exodus.com/article/168-what-is-a-replay-attack
[41] https://www.itprotoday.com/security/understanding-how-kerberos-authentication-protects-against-replay-attacks
[42] https://www.linkedin.com/pulse/deep-dive-security-reference-architecture-blockchains-kumar-giri-/
[43] https://www.coindesk.com/markets/2015/03/14/chainalysis-ceo-denies-sybil-attack-on-bitcoins-network/

dataset) by associating it with the appropriate persons in the training dataset. It's worth noting that just changing people's real names to pseudonyms before publishing a dataset isn't always enough to keep their identities anonymous. The mobility traces contain information that may be uniquely traced back to a person. Furthermore, while a dataset can be cleaned before being released by adding geographical and temporal noise, there is still a danger of re-identification through a de-anonymization attack. As a result, developing a way to quantify this risk is critical to determine its relevance [47].

S. Gambs et al. [47] created a mobility model called Mobility Markov Chain for implementing this attack(MMC). The mobility traces obtained during the training phase are utilized to create an MMC, then used to execute the attack during the testing phase.

Facebook, LinkedIn, and Xing, among other social networking sites, have reported exponential growth rates and millions of registered members. G. Wondracek et al. [120] presented a unique de-anonymization technique that takes advantage of group membership data accessible on social networking sites.

Articles related to this attack [38, 47, 83, 89, 120].

2.13 Key Attack

2.13.1 Attack 55: Hashing Operation Vulnerability. SHA-256 is widely used in the blockchain Hashing function, though other hashing algorithms, such as Ripemd160 and sCrypt, are also utilized in the operation and construction of blockchain Hashing functions. The SHA-256 algorithm is currently thought to be unbreakable. It is, nevertheless, vulnerable to the length extension attack. Without knowing the shared secret, a hash of a signed message can be modified by appending some attacker-controlled data to the original message. To avoid the length extension attack, Ferguson and Schneier recommend using double SHA-256 [43]. Birthday attacks, which are probabilistic attacks that break collision resistance via repeated evaluations, are likewise vulnerable to these hash algorithms. This type of attack's effectiveness in the real world was recently proved for the SHA-1 algorithm [104]. As a result, algorithms such as MD5 and SHA-1 are effectively flawed and should never be used for cryptographic operations [34].

Articles related to this attack [34].

2.13.2 Attack 56: Cryptography Key Vulnerability. ECDSA, the elliptic curve variation of DSA, is widely utilized in blockchain implementations because it has various advantages over other discrete logarithm-based algorithms and factoring modulus techniques, including smaller key size and faster processing. Specific domain parameters define which elliptic curves are acceptable for cryptographic operations. Many of these standardized elliptic curves have flaws in theory or were created with problematic parameters. Some cryptographers, for example, are wary about the NIST P-256 curve since the derivation of the curve parameters is not correctly described and leaves open the possibility of manipulation, resulting in the curve including intentional flaws or "backdoors." This type of deliberate flaw has been seen before, as NIST previously issued Dual-EC-DRBG, a standard for a cryptographically secure random number generator based on elliptic curve operations. The backdoor was suspected even before the standard was published, and it was later disclosed in a Reuters piece that RSA Security was paid $10 million to use this method as the default random number generator in the RSA BSAFE library, despite RSA's denial[44]. The backdoor enables anyone with access to a secret set of numbers to decrypt any message with only 32 bits of ciphertext[45] [34].

Articles related to this attack [34].

[44] https://www.reuters.com/article/us-usa-security-rsa-idUSBRE9BJ1C220131220
[45] https://www.schneier.com/essays/archives/2007/11/did_nsa_put_a_secret.html

2.13.3 Attack 57: Certificate Authority. A certificate authority, also called as a certification authority (CA), is a cryptographic body that issues digital certificates. A digital certificate verifies that the named subject of the certificate owns a public key. Others (relying parties) can trust signatures and assertions about the private key that matches the certified public key. A CA serves as a trusted third party, trusted by both the certificate's subject (owner) and the party relying on the certificate[46][47].

The security and utility of the Internet public-key infrastructure (PKI) have been jeopardized by recent attacks on certification authorities (CAs) and fraudulently issued certificates. Such attacks are anticipated to recur on a regular basis, necessitating the development, implementation, and implementation of appropriate responses [82].

Articles related to this attack [51, 82].

2.14 Reputation-based

An agent manipulates his reputation by rebranding himself as a good guy [44].

2.14.1 Attack 58: Hiding Blocks. Under this attack, an agent only In this assault, an agent selectively displays transactions that have a positive influence on his reputation while concealing transactions that have a negative impact[44].

Articles related to this attack [44, 62].

2.14.2 Attack 59: Whitewashing. To clear history, create a new account. The defense is simple: while executing the allocation policy, give new identity agents lesser priority [22].

Articles related to this attack [22, 44].

2.15 Mining Pool

2.15.1 Attack 60: Pool Hopping. The pool hopping attack reduces the mining pool's and honest miners' expected profits in Blockchain. The mainstream countermeasures, namely PPS (pay-per-share) and PPLNS (pay-per-last-N-share), can hedge pool hopping but need to charge miners some fees when they join in a pool. The higher fee charged the higher cost of joining the pool, the less motivation of a miner to mine in the pool. In an article written by Hongwei Shi et al. [99], they applied the zero-determinant (ZD) theory to design a novel pooled mining that offers an incentive mechanism for motivating miners not to switch in pools strategically by economic means without a fee charged.

A pool hopping attack occurs when miners leave a pool when the financial rewards are lower and rejoin when the financial rewards of mining are higher in blockchain networks. By leaving and rejoining the pool only during profitable periods, the miner earns more than the computational power they contribute. Miners leaving the pool deprive the pool's collective hash power, yielding the pool incapable of successfully mining the block. As a result, its competitors begin mining the block before they can complete mining. Existing research shows pool hopping resistant measures and detection strategies; however, they do not offer any robust preventive solution to discourage miners from leaving the mining pool. To prevent pool hopping attacks, a smart contract-based pool hopping attack prevention model is proposed by [101].

Articles related to this attack [57, 98, 99, 101].

2.15.2 Attack 61: Selfish Mining. Even if the fair majority assumption holds, the selfish mining (SM) attack gives a miner more than her fair share by straying from real mining (i.e., following the publicly acknowledged correct method).

[46] https://en.wikipedia.org/wiki/Certificate_authority
[47] https://datahack4fi.org/what-is-certificate-authority-in-network-security

The attack's fundamental idea is to keep the mined blocks secret so that they can be extended individually and then release them later to eliminate other blocks from the main chain. Because honest miners always choose the chain with the most blocks, and the difficulty adjusts over time, the attack succeeds. In truth, none of these causes appear to be avoidable, as the former is a preventative mechanism against network partitions and propagation delays, while the latter is an early design decision to ensure the main chain's projected inter-block duration remains constant [23].

Articles related to this attack [23, 57].

2.15.3 Attack 62: Baseline. The baseline attack approach is a selfish miner 2.15.2 making two blocks, BS1 and BS2, then forking the main blockchain to invalidate the block BH of an honest miner. For ten minutes, the attacker rents half of NiceHash's Bitcoin hash power. The attack process is divided into two rounds. In the first round, the attacker uses his own hashing power to compute the first block, BS1. It then withholds the block and waits for the network to accept the honest miner's block BH. The attacker utilizes the hired hash power in the second round to compute the next block BS2 before anybody else on the network. After computing the block, the attacker forks the main blockchain with his or her own private chain. As a result, the network shifts to the selfish miner who has branched private chains and discards the honest miner's block. The selfish miner succeeds in his attack and receives more rewards than the attack cost [92].

Articles related to this attack [92].

2.15.4 Attack 63: Fork-After-Withhold (FAW). A fork after withholding (FAW) attack is not just a different form of attack. The reward for a FAW attacker is always equal to or greater than the reward for a BWH attacker, and it can be used up to four times more frequently per pool than the reward for a BWH attacker. When several pools are taken into account, the current state of the Bitcoin network, the additional reward for a FAW attack is approximately 56% greater than that for a BWH attack. Additionally, when two pools launch FAW attacks against one another, the miner's dilemma may not hold: under some conditions, the larger pool can continuously win. More crucially, unlike selfish mining, while using intentional forks, a FAW attack does not suffer from practical concerns 2.15.2. FAW attacks are expected to be seen among mining pools [67].

Articles related to this attack [67].

2.15.5 Attack 64: Liveness. The liveness attack, proposed by Aggelos et al. [64], can delay the confirmation time of a target transaction as much as feasible. They also demonstrate two variants of the attack on Bitcoin and Ethereum. The attack preparation phase, transaction denial phase, and blockchain retarder phase are the three steps of an aliveness attack:

- Attack preparation phase: An attacker gains an edge over honest miners in some way before the target transaction TX is broadcasted to the public chain, similar to a selfish mining attack(see 2.15.2). The attacker creates a longer private chain than the public chain.
- Transaction denial phase: In order to prevent TX from being published into the public chain, the attacker keeps the block containing TX privately.
- Blockchain retarder phase: At some point during the public chain's expansion, TX will no longer be able to be privately kept. In this situation, the attacker will make the block containing TX public. Whenever the depth of the block that contains TX is larger than a constant in various blockchain systems, TX is considered genuine. As a result, in order to gain an advantage over the public chain, the attacker will continue to establish private chains. The attacker will then post her privately owned blocks into the public chain at the appropriate time to

slow down the rate of growth of the public chain. When TX is verified as legitimate in the public chain, the liveness attack will be over[48] [70].

Articles related to this attack [56, 64, 69, 70, 100, 125, 126].

2.15.6 Attack 65: Block Withholding. A block mining attack was constructed by a few pool components in the general block withholding assault, although they did not express any blocks [16]. In this example, the attacker constructs a legal block but does not broadcast it; instead, transaction X is broadcast as a payment for goods or services. When a retailer notices transaction X, he or she may accept this 0-confirmation transaction. The attacker will then begin broadcasting the previously created block containing transaction Y, which is in conflict with transaction X, and the Bitcoin network will accept his block and invalidate transaction X[49] [113].

Participants in the Bitcoin system are rewarded for solving cryptographic riddles. Some users form mining pools and distribute the pool's benefits according to each participant's input in order to get more constant payouts over time. Several attacks, however, put the ability to participate in pools in jeopardy. By allowing malevolent players to get unjust pay while merely seeming to provide work, the block withholding (BWH) attack makes the pool reward structure unfair. When two pools conduct BWH assaults against one other, the miner's dilemma arises: in a Nash equilibrium, both pools' income is reduced[67]. An attacker can unjustly gain more rewards by purposely constructing forks in another attack known as selfish mining 2.15.2.

Articles related to this attack [16, 57, 67, 111, 113].

2.15.7 Attack 66: Block Reordering. Specific cryptographic techniques (such as erroneously employing CBC or ECB) allow blocks to be re-ordered while correctly decrypting[50]. For example, in such algorithms, an attacker can swap two blocks and know that the second block will be decrypted to the same plaintext as the original.

To see more information on this attack see the web address in the footnote.

2.15.8 Attack 67: Block Discarding Attack and Difficulty Raising. According to a widely held security claim made in the original Bitcoin white paper, the Bitcoin system is secure as long as no attacker controls half or more of the total computational power used to maintain the system. However, this claim is supported by theoretically erroneous assumptions. Lear Bahack [15] analyzes two types of attacks based on two theoretical weaknesses: Block Discarding and Difficulty Raising. They argue that the theoretical limit on an attacker's fraction of total computational power required for system security is currently not $\frac{1}{2}$ but slightly less than $\frac{1}{4}$, and outline protocol changes that can raise this limit to as close to $\frac{1}{2}$ as possible.

Articles related to this attack [15].

2.16 Wallet

Since private keys can be stored in lots of fashions, many attacks can be used to obtain access to them.

2.16.1 Attack 68: Wallet Theft. Wallets are mainly used to store private keys. Wallets of many forms, including paper wallets, mobile wallets, desktop wallets, hardware wallets, and web wallets, have all been used. It is critical to protect private keys from loss or theft. These wallets, however, are vulnerable to failure or attacks and, in some cases, are not irreversible. Damage resistance may be possible with introducing multi-signature addresses and cloud storage

[48] https://www.sciencedirect.com/topics/engineering/liveness
[49] https://jheusser.github.io/2013/02/03/satcoin.html
[50] https://cloudsecurityalliance.org/blog/2020/10/26/blockchain-attacks-vulnerabilities-and-weaknesses/

systems unless cooperation occurs[78]. A hardware wallet is a collection of physical devices linked to a private key. It is impervious to digital attacks since it lacks complicated components (such as an operating system). A paper wallet is a written piece of paper with a private key. This method is deemed secure because the private key is kept away from the internet. Although it is impervious to digital threats, the paper wallet is vulnerable to lose, misreading, or damage. As a result, it is no more widely utilized[51][52] [45].

Articles related to this attack [45, 78].

2.16.2 Attack 69: Wallet Malware. Malware is software that infects, examines, steals, or performs nearly any behavior an attacker desires. It is often sent through a network. Malware has the ability to steal the private key from a victim's wallet[53].

To see more information on this attack see the web address in the footnote.

2.16.3 Attack 70: Wallet Phishing. Phishing attacks trick users into providing personal information by sending them legitimate-looking but phony emails and web pages [121]. Wallet phishing entails revealing secret information about a crypto wallet, such as the private key.

Articles related to this attack [121].

2.16.4 Attack 71: Etherdelta. EtherDelta is a decentralized exchange that lists almost every Ethereum-based coin currently available. Although it does not have the same volume as more extensive exchanges, it is a vital first step for traders after a new token is created in an ICO (initial coin offering) The smart contracts that regulate EtherDelta's behavior were apparently unaffected by the hack. Instead, the attacker was able to take control of EtherDelta's DNS server and redirect visitors to a phony version of the site. This is significantly more harmful than the typical phishing attempt, in which a fake website creates a domain name that looks similar to the actual one (such as etherrddeltta.com). Users who visited the official EtherDelta website on Wednesday afternoon (ET time) were given a partially working but convincing version of the site. The attack appears to have been stopped within a few hours, and the legitimate EtherDelta site has been restored, although anyone connected with the false site may have sent ether or other tokens to the hacker[54].

Articles related to this attack [51].

2.16.5 Attack 72: Attacks On Cold Wallets. Hardware wallets, also known as cold wallets, can be hacked as well. For example, researchers used flaws in the Nano S Ledger wallet to launch an Evil Maid attack. Researchers received the victims' private keys, as well as their PINs, recovery seeds, and passwords, as a result of this incident[55].

To see more information on this attack see the web address in the footnote.

2.16.6 Attack 73: Attacks On Hot Wallets. Hot wallets are web-based applications that store private cryptographic keys. Though cryptocurrency exchange owners maintain that their users' data is kept in wallets that aren't connected to the internet, a $500 million attack on Coincheck in 2018 demonstrated that this isn't always the case[56].

To see more information on this attack see the web address in the footnote.

[51] https://crypto-current.co/what-is-paper-wallet/
[52] https://www.investopedia.com/terms/p/paper-wallet.asp
[53] https://www.paloaltonetworks.com/cyberpedia/what-is-malware
[54] https://mashable.com/article/etherdelta-hacked
[55] https://www.apriorit.com/dev-blog/578-blockchain-attack-vectors
[56] https://www.apriorit.com/dev-blog/578-blockchain-attack-vectors

2.16.7 Attack 74: Passphrase Extraction. A passphrase [57] protects the private key; adversaries with the passphrase can use the private key on the account owner's behalf, such as signing data and sending transactions. Two different methods are present to extract a passphrase [118]: Passphrase Sniffing is mentioned in subsection 2.16.8, and Brute Force is discussed in subsection 2.16.10.

Articles related to this attack [118].

2.16.8 Attack 75: Passphrase Sniffing. A user can remotely call the personal unlock-Account function to unlock its account. However, the function and the parameters, including the passphrase, are transmitted in plaintext, which provides adversaries opportunities to sniffer the passphrase. This attack requires two conditions: the adversary has the ability to capture the traffic between the geth (see subsection 2.2.3) node and the remote user; and the user calls the personal unlock-Account [118].

Articles related to this attack [118].

2.16.9 Attack 76: Packet Sniffing. Packet Sniffing can be seen as a more general version of the Passphrase Sniffing. Sniffing attacks refer to the theft or interception of data by capturing the network traffic using a packet sniffer. Sniffing attacks are also called "packet sniffing" or "network sniffing" attacks because cybercriminals sniff data packets within a network. A data packet is a logical unit of data transmitted and received over a network.

The sniffing attack is the act of intercepting or capturing data as it traverses a network. This perception is similar to law enforcers wiretapping a suspect's phone line to gather essential information[58].

Sniffer is a network traffic monitoring and analysis program. It is intended to identify network bottlenecks and problems. A network administrator can use this information to maintain traffic flow efficiently. A sniffer can also be used to capture data being transmitted on a network. A sniffing attack is when a sniffer is used to capture the data in transit [1].

Sniffing attacks enable common network threat types such as man-in-the-middle attacks, insider threats etc[59].

For more information related to these attacks see [1], and footnote.

2.16.10 Attack 77: Brute Force. There is no limitation on retrying the unlock account calls because the RPC and WS services are stateless[60]. The personal unlock-Account function is realized in the source code of"/internal/ethapi/api.go". The code does not record the refused times nor the last failure time. As a result, the adversaries can keep on trying passphrases to find the correct one. This attack requires that the geth node provides the RPC/WS service and enables the personal module

Kerberos is vulnerable to password guessing since it cannot detect a dictionary attack [114]. Brute-force password-cracking attacks target the encrypted timestamp that is embedded in the Kerberos pre-authentication data [36]. The user's password is used to encrypt the timestamp. Because timestamps are relatively easy to recognize, it is possible to mount brute-force attacks against the encrypted pre-authentication data and derive the user's password. There are ways to protect against this attack; for example, using Windows smartcard login with Kerberos extension or encrypting the network traffic between the client and the KDC using IP Security (IPSec) [3].

Articles related to this attack [3, 24, 24, 114, 118].

[57] A passphrase is a string of characters that is used to control access to the system, application, or data. It is similar to a password, but it is often lengthier for increased protection.
[58] https://www.techslang.com/definition/what-is-a-sniffing-attack/
[59] https://cisomag.eccouncil.org/what-are-sniffing-attacks-and-how-to-defend-against-them/
[60] https://github.com/ethereum/wiki/wiki/JSON-RPC

2.16.11 Attack 78: Dictionary. When an attacker uses a dictionary attack to guess a user's password, they attempt each word in the dictionary (a list of likely passwords) in the hopes that one of those guesses is the user's real password. This is known as a brute-force attack [4].

Attackers use common words and phrases, such as those from a dictionary, to try to guess passwords in a dictionary attack. Dictionary attacks can be successful because people frequently use simple, easy-to-remember passwords for multiple accounts.

A dictionary attack, as opposed to a traditional brute-force attack, uses a much smaller set of pre-selected words and phrases to bypass authentication controls. While a dictionary attack takes less time and resources to perform, it reduces the chances that a difficult password will be guessed correctly[61].

Articles related to this attack [4].

2.16.12 Attack 79: Private key Recovery. A key-recovery attack is an attempt by an adversary to recover an encryption scheme's cryptographic key. This typically indicates that the attacker possesses a pair, or multiple pairs, of plaintext messages and their associated ciphertext. Historically, cryptanalysis of block ciphers has concentrated on key recovery. Still, security against these types of attacks is extremely weak, as it may not be essential to recover the key in order to extract partial information about the message or to decrypt the message entirely[62]. This attack is also possible to occur with the advancement of quantum computing so that the cryptographic keys can be cracked quickly [32].

Articles related to this attack [32].

2.16.13 Attack 80: Dusting. A dusting attack is a comparatively recent kind of malicious activity in which hackers and fraudsters attempt to compromise the privacy of cryptocurrency users by sending them insignificant amounts of money to their wallets. The attackers then monitor the wallets' transactional behavior, doing a combined analysis of many addresses in order to de-anonymize the individual or entity behind each wallet[63].

In other words, a dusting assault is one in which a negligible quantity of cryptocurrency, referred to as dust, is transmitted to thousands—and perhaps hundreds of thousands—of wallet addresses. This assault is being launched with the goal of unmasking or de-anonymizing these addresses. Dust is found on the majority of public blockchains, including but not limited to Bitcoin, Litecoin, Bitcoin Cash, and Dogecoin[64].

Numerous groups conduct dusting assaults. Dusting attacks have been used to de-anonymize people with significant cryptocurrency holdings. Individuals with substantial assets may be targeted in a variety of ways, including phishing attacks and cyber-extortion. Users with significant bitcoin holdings in high-risk locations may also face physical attacks or have a family member abducted for a cryptocurrency ransom.

Articles related to this attack [87].

2.16.14 Attack 81: Refund. BIP70 is a widely adopted Payment Protocol standard that defines how merchants and customers handle Bitcoin transactions. This standard is supported by the majority of significant wallets and the two largest payment processors, Coinbase and BitPay, which together provide the infrastructure for more than 100,000 merchants to accept Bitcoin as a form of payment. P. McCorry et al. [74] describe new Payment Protocol attacks that affect all BIP70 merchants. The Silkroad Trader attack demonstrates a vulnerability in the Payment Protocol's authentication mechanism, whereas the Marketplace Trader attack takes advantage of current Payment Processors'

[61] https://www.csoonline.com/article/3568794/what-is-a-dictionary-attack-and-how-you-can-easily-stop-them.html
[62] https://en.wikipedia.org/wiki/Key-recovery_attack
[63] https://academy.binance.com/en/articles/what-is-a-dusting-attack
[64] https://www.gemini.com/cryptopedia/crypto-dusting-attack-bitcoin

refund policies. Both attacks have been experimentally verified on real-world merchants with the aid of a customized Bitcoin wallet. Both Coinbase and Bitpay have recognized the attacks and implemented interim mitigation measures.

Articles related to this attack [51, 52, 74].

2.17 Splitting

2.17.1 Attack 82: Orphaned Blocks. An orphaned block is one in which the hash field of its parent block points to an unauthentic block that has been removed from the Blockchain [93]. Inconsistencies in the Blockchain are caused by orphan blocks. It can be caused by race conditions in the miners' work or introduced by an attacker.

Articles related to this attack [93].

2.18 Design Flaw

Because of flaws in system design, some systems are targets of specific attacks. The DAO attack is a well-known example of a design flaw-related disaster. DAO is the acronym for *Decentralized Autonomous Organization*. The DAO is a crowd-funding platform implemented using a smart contract released on Ethereum on May 28, 2016. Only after the DAO contract had been deployed for 20 days was it attacked. DAO had raised $150 million before the hack, making it the largest crowd-funding project ever. The thief made off with around $60 million in cash. In this situation, the attacker made use of the Re-entrancy flaw. The following subsection defines the re-entrancy attack. First, the attacker creates a malicious smart contract containing a call to DAO's withdraw() function in its callback function. The callee, also called, will receive Ether from the withdraw() function. As a result, it will call the malicious smart contract's callback function once more. In this technique, the attacker can take all of DAO's Ether [70].

Articles related to this attack [70]. To see more information on this attack see the web address [65].

2.18.1 Attack 83: Re-entrancy. When an attacker recursively calls the target's withdraw function, he or she is committing a re-entrancy attack. The attacker can repeatedly execute the withdraw function to drain the contract's funds if the contract fails to update its status, such as a victim's balance, before sending payments. The DAO hack, which resulted in a $60 million loss, is a well-known real-world Re-entrancy attack [95].

Articles related to this attack [13, 16, 34, 70, 95]. To see more information on this attack see the web address [66].

2.19 Code Flaw

Due to coding issues, other wallet applications have also been attacked. Due to a fault in the Parity wallet software, an individual using the handle devops199 "accidentally" destroyed 513,743 ether, worth around 355 million at the time of writing. This was achieved because Parity encapsulated all contracts in a library, another Ethereum smart contract. Unfortunately, this library fawned so that it was easy for someone to use the library to create a wallet, giving them ownership of the library. Unfortunately, devops199 called this initialization code, then called a kill function built into this library, either by accident or on purpose. Because all multi-sig wallets in the Parity system relied on this library contract, terminating it effectively rendered the monies lost unrecoverable. One reviewer compares the occurrence to strolling into a bank, finding the vault door is open, entering the vault, and burning all of the money [34].

Articles related to this attack [34].

[65] https://hackernoon.com/smart-contract-attacks-part-1-3-attacks-we-should-all-learn-from-the-dao-909ae4483f0a
[66] https://quantstamp.com/blog/what-is-a-re-entrancy-attack

2.19.1 Attack 84: Coindash. During the initial coin offering[67], the Ethereum address to which investments were supposed to be delivered was moved to the hackers' wallet. The perpetrators who hacked the Coindash website carried out this attack. The victims were compensated, but the perpetrators were not identified, which is frequent in blockchain-related crimes. The robbery was thought to have been started by an insider. This assault may have been stopped if the website had adequate security, such as network-based threat detection and mitigation68. Furthermore, keeping an eye out for malicious insiders may have reduced the chances of an insider breach[123].

Articles related to this attack [123]. To see more information on this attack see the web address in the footnote.

2.19.2 Attack 85: Overflow. When more value is submitted than the maximum value, an overflow error attack on a smart contract happens. When this occurs, the value returns to zero, and this feature can be taken advantage of by continuously activating the feature that increases the value[68].

To see more information on this attack see the web address in the footnote.

2.19.3 Attack 86: Underflow. This mistaken attack works in the opposite direction as the overflow error. An underflow mistake happens when going below the minimum amount rather than exceeding the maximum value. Instead of reverting to zero, this causes the system to bring right back up to maximum value[69].

To see more information on this attack see the web address in the footnote.

2.19.4 Attack 87: Cryptojacking. This attack uses web and cloud-based services to mine a block without permission. In other words, the attacker steals the web server's computing power to mine the blocks without the owner's knowledge. Turn it into a mining pool in other circumstances [62].

Articles related to this attack [62].

2.19.5 Attack 88: False Data Injection. False data injection attacks were recently identified as a significant class of cyberattacks against large area measuring and monitoring systems of smart grids. These cyberattacks are designed to manipulate the readings from several power grid sensors and phasor measuring units in order to deceive the operation and control centers. Recent studies have shown that if an adversary has complete knowledge on the power grid topology and transmission-line admittance values, he/she can adjust the false data injection attack vector such that the attack remains undetected and successfully passes the residue-based bad data detection tests that are commonly used in power system state estimation [86].

According to M. Ahmed and S. Kh. Pathan [6] , the concept of false data injection attack (FDIA) originated in the smart grid domain. The term may sound comprehensive; however, it refers to the situation in which an attacker manipulates sensor readings in such a way that undetected errors are introduced into state variable and value calculations. As the Internet and accompanying complex adaptive systems continue to increase in popularity, cyber attackers are interested in attempting similar attacks in additional application fields such as healthcare, finance, defense, and governance. FDIA has become a primary priority in today's more dangerous cyber environment of sophisticated adaptive systems. Today, it is essential to have a stronger understanding of cyberattacks and a better system for defending against them.

Identifying a false data injection attack (FDIA) in power systems is difficult due to the complexity of the data and the low detection accuracy. In view of the correlation and redundancy of the attack data's main properties, a method for detecting the FDIA in smart grids based on cyber-physical genes is proposed in [85].

[67] An initial currency offering or initial coin offering(ICO) is a sort of cryptocurrency-based fundraising. https://en.wikipedia.org/wiki/Initial_coin_offering
[68] https://cryptoadventure.com/understanding-overflow-and-underflow-attacks-on-smart-contracts/
[69] https://cryptoadventure.com/understanding-overflow-and-underflow-attacks-on-smart-contracts/

Articles related to this attack [6, 85, 86].

2.19.6 Attack 89: Man In The Middle. To accept payments in a cryptocurrency system like Bitcoin, users employ string-hashes obtained from public keys. These string-hashes are similar to random strings. Users' identities cannot be verified because there is no authority to do so. For the most part, users can't show that an address is linked to their actual identity. It is possible that a victim could be provided with a fictitious address and be instructed to send money to this fake location [48]. It is worth mentioning that with man in the middle attack, the attacker can not tamper the transaction because it is signed by private key.

Articles related to this attack [48].

2.19.7 Attack 90: SQL-Injection. SQL injection is, in fact, a type of cyber attack in which a hacker manipulates a database using SQL (Structured Query Language) code to obtain access to potentially important data.

SQL injections are also considered a type of code injection attack that targets applications and their data. They are most commonly associated with website hacking, although ome may use SQL injections to attack a SQL database. An attacker must first locate weak user inputs within the web page or application to initiate a SQL Injection attack. Malicious SQL instructions are performed in the database when the attacker creates and sends input content. In this way, an attacker can get complete control of the database. SQL injections have been around for quite some time and most of them are automated[70].

Allowing attackers to spoof identity, interfere with existing data, cause repudiation issues such as making changes to the balances or canceling transactions, allow the complete disclosure of all data on the system, destroy or make it unavailable, and even becoming administrators of the database server are all possible outcomes of SQL injection attacks[71].

To truly bring security to the masses, R. Chandrashekhar et al. [25] propose a classification that not only enumerates but also categorizes the various attack methodologies and also the testing frameworks and prevention mechanisms.

Articles related to this attack [25, 84].

2.20 Smart Contracts And Ethereum Virtual Machine (EVM)

Obviously, if the source code of a smart contract contains weaknesses, the parties that sign the contract are at risk. Bugs in source code, a network's virtual machine, the runtime environment for smart contracts, and the blockchain itself are among the most common blockchain security vulnerabilities related to smart contracts. In the following, we discuss a few smart contracts and EVM-related issues[72].

To see more information on this attack see the web address in the footnote.

2.20.1 Attack 91: Tx.Origin. If a smart contract's source code contains flaws, the parties who sign the contract are at risk. The most prevalent blockchain security vulnerabilities connected to smart contracts are bugs in source code, a network's virtual machine, the runtime environment for smart contracts, and also the blockchain itself[73] [77].

Articles related to this attack [77].

2.20.2 Attack 92: Fake Receipt. The key feature of this attack is that a phony notice of receiving tokens deceives a vulnerable smart contract, while the actual token transfer takes place between two attacker accounts [55].

[70] https://www.neuralegion.com/protect-your-application-against-sql-injection/
[71] https://en.wikipedia.org/wiki/SQL_injection
[72] https://www.apriorit.com/dev-blog/578-blockchain-attack-vectors
[73] https://www.apriorit.com/dev-blog/578-blockchain-attack-vectors

Articles related to this attack [51, 55].

2.20.3 Attack 93: Fake EOS. The most essential aspect of this attack is to defraud the susceptible smart contract of official EOS tokens by employing fake EOS tokens, which can be identified by transaction records carrying token issuer information. Initially, all token transfer transactions will be filtered out with the token symbol "EOS" based on the observation. Then, these transactions will be grouped according to the following definitions:

- Fake-sending transactions that send fake EOS tokens.
- True-sending transactions that send true EOS tokens.
- True-receiving transactions that receive true EOS tokens.

As [55] mentions, a possible attack is defined as a series of fake-sending transactions followed by true-receiving transactions. A fake-sending transaction A can be connected with a true-receiving transaction B if and only if they both occur during the same period, with A preceding B. In all of these potential transactions, focus on those who have gained more actual EOS tokens than they have spent. For this purpose, compare the attacker's and vulnerable contracts' input-output ratios to identify unusual attacks. Finally, depending on the suspicious attacks, see if the vulnerable smart contracts will continue normal execution (for example, running a lottery for a real player) after receiving the phony EOS tokens. If that's the case, classify the transaction as a fake EOS attack.

Articles related to this attack [55].

2.20.4 Attack 94: Selfdestruction. Ethereum smart contracts provide a self-destruct mechanism that allows them to terminate a contract on the blockchain system. It is, however, a two-edged sword for developers. On the one hand, the self-destruct function allows developers to delete smart contracts (SC) from Ethereum and transfer Ethers in the event of an emergency, such as an assault. On the other side, this function might add to the development's complexity and provide an attack channel for attackers [26].

Articles related to this attack [26, 27].

2.20.5 Attack 95: Immutable Defects. Because blockchain blocks are immutable by nature, a smart contract can't be modified once it's been formed. However, if a smart contract's code includes any defects, they are likewise impossible to correct. As with the DAO assault, thieves may uncover and exploit code weaknesses in order to steal Ether or build a new fork [74].

Articles related to this attack [13]. To see more information on this attack see the web address in the footnote.

2.20.6 Attack 96: Cryptocurrency Lost In Transfer. This is conceivable if Ether is sent to an orphaned address with no owner or contract[75], and the address is lost permanently. Because lost Ether cannot be retrieved, programmers must manually verify that recipient addresses are valid [13].

Articles related to this attack [13].

2.20.7 Attack 97: Bugs.

Bugs in Access Control. In Ethereum smart contracts, there is a missed modifier bug (missing or wrongly utilized) that allows a hacker to gain access to sensitive functionality[76].

[74] https://www.apriorit.com/dev-blog/578-blockchain-attack-vectors
[75] https://www.apriorit.com/dev-blog/578-blockchain-attack-vectors
[76] https://www.apriorit.com/dev-blog/578-blockchain-attack-vectors

Other Smart Contract Bugs. Note that some kinds of bugs are caused by specific Solidity[77] versions. Since it is possible to use different versions of Solidity to develop smart contracts, we mention the Solidity version that cause these kinds of bugs.

(1) Integer Sign
Converting *int* type to *uint* type may produce incorrect results [110].
(2) Wrong Operator[78]
Users can use =+ and =- operators in the integer operation without compiling errors (up to and including version 0.4.26).
(3) Uninitialized Storage Variables[79]
The uninitialized storage variable serves as a reference to the first state variable in a contract, which may cause the state variable to be inadvertently modified (up to and including version 0.4.26).
(4) Transaction Order Dependence
Miners can decide which transactions are packaged into the blocks and the order in which transactions are packaged. The current main impact of this kind of bugs is the approve function in the ERC20 token standard [112].
(5) Signature With Wrong Parameter[80]
When the parameters of the ecrecover()[81] function are wrong, the ecrecover() function will return 0x0[82].
(6) Re-entrancy Vulnerability: See subsection 2.18.1.
When the call-statement is used to call other contracts, the callee can call back the caller and enter the caller again [72].
(7) Short Address Attack: See subsection 2.20.8.
When Ethereum packs transaction data if the data contains the address type and the length of the address type is less than 20 bits, subsequent data will be used to make up the length of the address type[83].

For more information related to these attacks see [72, 110, 124], and footnote.

2.20.8 Attack 98: Short Address. There is a known flaw in the Ethereum Virtual Machine (EVM) that might make transactions vulnerable to the "Short Address Attack" (SAA). Since the EVM CALL function pads arguments with 0s up to 32 bytes in length, it was determined that Smart Contracts that transfer ERC20 tokens might be vulnerable to the SAA. An SAA occurs when a contract gets less data than expected and there is insufficient or no validation of the length of the transaction payload – this may have been exploited to "fool" smart contracts into conducting harmful transactions. To reduce the danger of such attacks, the creator of a TKN ERC20 Contract implements protection that verifies the legitimacy of transfers by examining the transaction duration[84].

To see more information on this attack, see the web address in the footnote.

[77] Solidity is an object-oriented, high-level language for implementing smart contracts.
https://docs.soliditylang.org/en/v0.8.11/
[78] https://swcregistry.io/
[79] https://swcregistry.io/
[80] https://github.com/sec-bit/awesome-buggy-erc20-tokens/blob/master/ERC20_token_issue_list.md
[81] ecrecover() is a very useful Solidity function that allows the smart contract to validate that incoming data is properly signed by an expected party
https://docs.kaleido.io/faqs/why-ecrecover-fails/.
[82] When the attacker gives the wrong parameters and the value of the specified parameter id is 0x0, the attacker can pass the identity verification, which eventually leads to the ethers in the contract being destroyed [124]
[83] https://github.com/doingblock/smart-contract-security
[84] https://medium.com/monolith/tkn-and-short-address-attack-mitigation-88cc895734ba

2.21 Attacks on Shards

Sharding means that consensus nodes are distributed among subgroups (i.e., shards) such that each node only validates the transactions in its group. Shards operate in parallel and can achieve higher scalability and throughput since each shard has a throughput similar to an entire non-sharded blockchain. On the other hand, sharding has the potential to harm security because each shard has a lower number of participating nodes than the entire blockchain, which means that it may be easier for the attacker to compromise a single shard than the entire blockchain[85,86]. The main mitigation technique is to achieve a truly random distribution of nodes among the shards, thus minimizing the potential for adversaries to bias the randomness used for shard distribution. For example, Elastico [73] uses PoW to distribute nodes among shards, whereas Omniledger [65] uses a bias-resistant distributed randomness protocol (e.g., RandHound [107]). Poor design of sharding protocols may also lead to vulnerabilities such as replay attacks [57, 103].

Articles related to this attack [57, 65, 73, 103].

2.21.1 Attack 99: Single Shard Takeover (aka 1% Attack). The adversary can gather its corrupted nodes to the victim shard and compromise the shard's consensus. Since voting power is distributed among shards, compromising a single shard is easier than compromising a non-sharded blockchain [53].

Articles related to this attack [53].

2.21.2 Attack 100: Transaction Forging. The adversary creates a fake cross-shard transaction for a shard, then convinces other shards that this fake transaction has been included on that shard. Without cross-shard communication and the knowledge of the shard's ledger, other shards cannot determine whether this the cross-shard transaction is valid or not [53].

Articles related to this attack [53].

2.22 Other Attacks

There happen to be quite a few attacks in the current world of distributed and decentralized systems. Here, we list other types of attacks mentioned in several articles.

(1) Flawed Key Generation Exploit
(2) Unbounded Operations Exploit
(3) Trade-Off Attack
(4) Stack Size Exploit
(5) Opcode Exploit
(6) Delegate Function Exploit
(7) Exception Disorder Vulnerability Exploit
(8) Fake Deposit Exploit
(9) Fake Tokens Attack
(10) Gasless Send Exploit
(11) Multiple Withdrawal Attack
(12) Permission Check Exploit
(13) Rollback Vulnerability Exploit

[85] https://www.mangoresearch.co/1-shard-attack-explained-ethereum-shardingcontd/
[86] https://ethresear.ch/t/on-the-probability-of-1-attack/4009/6

(14) Blockchain Ingestion Attack
(15) Ransomware Network Attack
(16) Changing Systems Parameters Attack
(17) Timestamp Dependence Exploit
(18) Incorrect Transaction Attack
(19) Overlay Network DoS
(20) Breaking Network Assumptions
(21) Erebus Attack

3 DISCUSSION

Distributed consensus and anonymity are two essential characteristics of a distributed system such as the blockchain. The digital world has been revolutionized by blockchain technology, which enables a distributed consensus system in which all online transactions involving digital assets, both past, and present, can be confirmed at any time in the future. However, we live in risk in the digital world because we rely on a third party to secure and protect our digital assets. However, these third-party sources are susceptible to being hacked, altered, or infiltrated.

As defined by Neeraj Suri [106], a distributed system is typically a collection of geographically dispersed resources (computing and communication) that collectively (a) provides services that connect dispersed data producers, and consumers, (b) provides on-demand, highly reliable, highly available, and consistent resource access, frequently through the use of replication schemas to handle resource failures, and (c) enables a collective aggregated capability (computational or service-based) from the distributed resources. Distributed system security addresses the dangers posed by exploiting vulnerabilities in the attack surfaces produced by the distributed system's resource structure and functionality. Vulnerabilities are design or operational flaws that allow an attacker to potentially compromise a system. A threat is a measure of an attacker's potential or likelihood of causing damage or compromising the system. A threat is any behavior that has the potential to cause harm, whether foreseen or unexpected. The may or may not be resolved.

We focused on attack mechanisms in distributed systems. A distributed system may encounter various types of attacks and threats. It would be nice if a system designer or a scientist could predict harmful behaviors and think of a solution before using or designing a system. This article lists a wide range of types of attacks that may occur in a distributed system. This information aggregation would surely help anyone who wants to research or design a distributed system.

REFERENCES

[1] Chapter 4 - web-based mail issues. In *E-Mail Virus Protection Handbook*, Syngress, Ed. Syngress, Burlington, 2000, pp. 119–145.
[2] Chapter 3 - understanding threats. In *Host Integrity Monitoring Using Osiris and Samhain*, B. Wotring and B. Potter, Eds. Syngress, Burlington, 2005, pp. 79–100.
[3] Abdullah, N., Hakansson, A., and Moradian, E. Blockchain based approach to enhance big data authentication in distributed environment. In *2017 Ninth International Conference on Ubiquitous and Future Networks (ICUFN)* (2017), pp. 887–892.
[4] Adams, C. *Dictionary Attack*. Springer US, Boston, MA, 2011, pp. 332–332.
[5] Aggarwal, S., and Kumar, N. Chapter twenty - attacks on blockchain. In *The Blockchain Technology for Secure and Smart Applications across Industry Verticals*, S. Aggarwal, N. Kumar, and P. Raj, Eds., vol. 121 of *Advances in Computers*. Elsevier, 2021, pp. 399–410.
[6] Ahmed, M., and Pathan, A.-S. K. False data injection attack (fdia): an overview and new metrics for fair evaluation of its countermeasure. *Complex Adaptive Systems Modeling* (April 2020).
[7] Andress, J. *The Basics of Information Security: Understanding the Fundamentals of InfoSec in Theory and Practice*, 2nd ed. Syngress Publishing, 2014.
[8] Andrew, T. S., and Steen, M. V. *Distributed Systems: Principles and Paradigm.*, 2nd ed. Upper Saddle River, NJ, USA: Pearson Higher Education, 2007.

[9] Andrychowicz, M., Dziembowski, S., Malinowski, D., and Mazurek, L. Secure multiparty computations on bitcoin. In *2014 IEEE Symposium on Security and Privacy* (2014), pp. 443–458.

[10] Andrychowicz, M., Dziembowski, S., Malinowski, D., and Łukasz Mazurek. On the malleability of bitcoin transactions. *Financial Cryptography and Data Securityg* (2015), 1—-18.

[11] Apostolaki, M., Marti, G., Müller, J., and Vanbever, L. Sabre: Protecting bitcoin against routing attacks.

[12] Apostolaki, M., Zohar, A., and Vanbever., L. Hijacking bitcoin: Routing attacks on cryptocurrencies. *IEEE Symposium on Security and Privacy* (2017), 375–392.

[13] Atzei, N., Bartoletti, M., , and Cimoli, T. A survey of attacks on ethereum smart contracts sok. *Springer-Verlag New York, Inc. New York, NY, USA* (2017).

[14] Azouvi, S., Danezis, G., and Nikolaenko, V. Winkle: Foiling long-range attacks in proof-of-stake systems. pp. 189–201.

[15] Bahack, L. Theoretical bitcoin attacks with less than half of the computational power. *arXiv:1312.7013* (December 2013).

[16] Begum, A., Tareq, A. H., Sultana, M., Sohel, M. K., Rahman, T., and Sarwar, A. H. Blockchain attacks, analysis and a model to solve double spending attack. *International Journal of Machine Learning and Computing 10*, 2 (February 2020), 352—-357.

[17] Biryukov, A., and Pustogarov, I. Bitcoin over tor is not a good idea., 02 2015.

[18] Boneh, D., and Naor, M. Timed commitments. In *Advances in Cryptology - CRYPTO 2000, 20th Annual International Cryptology Conference, Santa Barbara, California, USA, August 20-24, 2000, Proceedings* (2000), vol. 1880 of *Lecture Notes in Computer Science*, Springer, pp. 236–254.

[19] Bonneau, J. Why buy when you can rent? bribery attacks on bitcoin-style consensus. In *Financial Cryptography and Data Security - International Workshops, FC 2016, BITCOIN, VOTING, and WAHC, Revised Selected Papers* (2016), K. Rohloff, J. Clark, S. Meiklejohn, D. Wallach, M. Brenner, and P. Ryan, Eds., Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), Springer Verlag, pp. 19–26. International Workshops on Financial Cryptography and Data Security, FC 2016 and 3rd Workshop on Bitcoin and Blockchain Research, BITCOIN 2016, 1st Workshop on Advances in Secure Electronic Voting Schemes, VOTING 2016, and 4th Workshop on Encrypted Computing and Applied Homomorphic Cryptography, WAHC 2016 ; Conference date: 26-02-2016 Through 26-02-2016.

[20] Buford, J. F., Yu, H., and Lua, E. K. *P2P Networking and Applications*. Morgan Kaufmann, Boston, 2009.

[21] Buterin, V., and Griffith, V. Casper the friendly finality gadget.

[22] Cai, Y., and Zhu, D. Fraud detections for online businesses: a perspective from blockchain technology. *Financial Innovation 2*, 20 (2016), 1–10.

[23] cer, O. B., and Kupcu., A. Fortis: Selfish mining mitigation by (for)geable (ti)me(s)tamps.

[24] Chalaemwongwan, N., and Kurutach, W. A practical national digital id framework on blockchain (nidbc). In *2018 15th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology (ECTI-CON)* (2018), pp. 497–500.

[25] Chandrashekhar, R., Mardithaya, M., Thilagam, S., and Saha, D. Sql injection attack mechanisms and prevention techniques. In *Advanced Computing, Networking and Security* (Berlin, Heidelberg, 2012), P. S. Thilagam, A. R. Pais, K. Chandrasekaran, and N. Balakrishnan, Eds., Springer Berlin Heidelberg, pp. 524–533.

[26] Chen, J., Xia, X., Lo, D., and Grundy, J. Why do smart contracts self-destruct? investigating the selfdestruct function on ethereum., 2021.

[27] Chen, J., Xia, X., Lo, D., and Grundy, J. Why do smart contracts self-destruct? investigating the selfdestruct function on ethereum. *ACM Trans. Softw. Eng. Methodol. 31*, 2 (December 2021).

[28] Chohan, U. W., and Chohan, U. W. The double spending problem and cryptocurrencies.

[29] Conti, M., E, S. K., Lal, C., and Ruj, S. A survey on security and privacy issues of bitcoin. *CoRR abs/1706.00916* (2017).

[30] Coulouris, G. F., Dollimoreand, J., and Kindberg, T. *Distributed Systems-Concepts and Design*, 4th ed. London, England: Addison, 2005.

[31] Courtois, N. T. Double-spending fast payments in bitcoin.

[32] Crosby, M., Pattanayak, P., Verma, S., and Kalyanaraman, V. Applied innovation review. *Applied Innovation Review 2* (2016), 5–20.

[33] Daian, P., Pass, R., and Shi, E. Snow white: Robustly reconfigurable consensus and applications to provably secure proof of stake. In *Financial Cryptography* (2019).

[34] Dasgupta, D., Shrein, J., and Gupta, K. D. A survey of blockchain from security perspective. *Journal of Banking and Financial Technology 3* (January 2019).

[35] Deirmentzoglou, E., Papakyriakopoulos, G., and Patsakis, C. A survey on long-range attacks for proof of stake protocols. *IEEE Access 7* (2019), 28712–28725.

[36] Dell'Amico, M., Michiardi, P., and Roudier, Y. Limitations of the kerberos authentication system. *SIGCOMM Comput Commun Rev 20*, 5 (10 1990), 119—-132.

[37] Dey, S. Securing majority-attack in blockchain using machine learning and algorithmic game theory: A proof of work.

[38] Ding, X., Zhang, L., Wan, Z., and Gu, M. A brief survey on de-anonymization attacks in online social networks. In *2010 International Conference on Computational Aspects of Social Networks* (2010), pp. 611–615.

[39] Ebrahimpour, G., and Haghighi, M. S. Analysis of bitcoin vulnerability to bribery attacks launched through large transactions. *CoRR abs/2105.07501* (2021).

[40] Ebrahimpour, G., and Haghighi, M. S. Analysis of bitcoin vulnerability to bribery attacks launched through large transactions, 2021.

[41] Eskandari, S., Moosavi, M., and Clark, J. Sok: Transparent dishonesty: Front-running attacks on blockchain. pp. 170–189.

[42] Fedotov, I., and Khritankov, A. Statistical model checking of common attack scenarios on blockchain. *Electronic Proceedings in Theoretical Computer Science 342* (September 2021), 65—-77.

[43] Ferguson, N., and Schneier, B. *Practical cryptography*. New York : Wiley, ©2003., 2003.

[44] Ferrag, M. A., Derdour, M., Mukherjee, M., Derhab, A., and Janicke, L. M. H. Blockchain technologies for the internet of things: Research issues and challenges.

[45] Frankenfield, J. Paper wallet., August 2011.

[46] Fu, X., Wang, H., and Shi, P. A survey of blockchain consensus algorithms: mechanism, design and applications. *Sci. China Inf. Sci. 64* (2021).

[47] Gambs, S., Killijian, M.-O., and Núñez del Prado Cortez, M. De-anonymization attack on geolocated data. *Journal of Computer and System Sciences 80*, 8 (2014), 1597–1614. Special Issue on Theory and Applications in Parallel and Distributed Computing Systems.

[48] Garba, A., Guan, Z., Li, A., and Chen, Z. Analysis of man-in-the-middle of attack on bitcoin address. In *ICETE* (2018).

[49] Gazi, P., Kiayias, A., and Russell, A. Stake-bleeding attacks on proof-of-stake blockchains. pp. 85–92.

[50] Gervais, A., Ritzdorf, H., Karame, G. O., and Capkun, S. Tampering with the delivery of blocks and transactions in bitcoin. CCS '15, Association for Computing Machinery, pp. 692—-705.

[51] Guggenberger, T., Schlatt, V., Schmid, J., and Urbach, N. A structured overview of attacks on blockchain systems. *Twenty-fifth Pacific Asia Conference on Information System* (2021).

[52] Gupta, B., and Sheng, Q. *Machine Learning for Computer and Cyber Security: Principle, Algorithms, and Practices*. Cyber Ecosystem and Security. CRC Press, 2019.

[53] Han, R., Yu, J., Lin, H., Chen, S., and Esteves-Veríssimo, P. On the security and performance of blockchain sharding. Cryptology ePrint Archive, Report 2021/1276, 2021. https://ia.cr/2021/1276.

[54] Harinath, D., Satyanarayana, P., and Murthy, M. V. R. A review on security issues and attacks in distributed systems. 1–9.

[55] He, N., Zhang, R., Wang, H., Wu, L., Luo, X., Guo, Y., Yu, T., and and, X. J. Eosafe: Security analysis of eosio smart contracts. *30th USENIX Security Symposium* (2021), 1271–1288.

[56] Heilman, E., Kendler, A., Zohar, A., and Goldberg., S. Eclipse attacks on bitcoin's peer-to-peer network. *24th USENIX conference Security Symposium* (08 2015), 129—-144.

[57] Homoliak, I., Venugopalan, S., Hum, Q., Reijsbergen, D., Schumi, R., and Szalachowski, P. The security reference architecture for blockchains: Towards a standardized model for studying vulnerabilities, threats, and defenses.

[58] Homoliak, I., Venugopalan, S., Hum, Q., and Szalachowski., P. A security reference architecture for blockchains.

[59] Iuon-Chang Lin, T.-C. L. A survey of blockchain security issues and challenges. *International Journal of Network Security, 19*, 5 (2017), 653–659.

[60] Joshi, J., and Mathew, R. A survey on attacks of bitcoin. In *Proceeding of the International Conference on Computer Networks, Big Data and IoT (ICCBI - 2018)*. (Cham, 2020), A. Pandian, T. Senjyu, S. M. S. Islam, and H. Wang, Eds., Springer International Publishing, pp. 953–959.

[61] Karame, G. O., Androulaki, E., and Capkun, S. Double-spending fast payments in bitcoin. In *Proceedings of the 2012 ACM Conference on Computer and Communications Security* (New York, NY, USA, 2012), CCS '12, Association for Computing Machinery, pp. 906—-917.

[62] Kausar, F., Senan, F. M., Asif, H. M., and Raahemifar, K. 6g technology and taxonomy of attacks on blockchain technology. *Alexandria Engineering Journal* (2021).

[63] Khana, K. M., Arshadb, J., and Khanc, M. M. Simulation of transaction malleability attack for blockchain-based e-voting. *Preprint submitted to Computers & Electrical Engineering* (November 2019).

[64] Kiayias, A., and Panagiotakos, G. On trees, chains and fast transactions in the blockchain.

[65] Kokoris-Kogias, E., Jovanovic, P., Gasser, L., Gailly, N., Syta, E., and Ford, B. Omniledger: A secure, scale-out, decentralized ledger via sharding. In *2018 IEEE Symposium on Security and Privacy (SP)* (2018), pp. 583–598.

[66] Kraus, R., Barber, B., Borkin, M., and Alpern, N. J. Chapter 2 - active directory – escalation of privilege. In *Seven Deadliest Microsoft Attacks*, R. Kraus, B. Barber, M. Borkin, and N. J. Alpern, Eds. Syngress, Boston, 2010, pp. 25–48.

[67] Kwon, Y., Kim, D., Son, Y., Vasserman, E., and Kim, Y. Be selfish and avoid dilemmas: Fork after withholding (faw) attacks on bitcoin. *CCS '17: Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security* (October 2017), 195––209.

[68] Lepinski, M., and Sriram, K. Bgpsec protocol specification. *RFC 8205* (2017), 1–45.

[69] Li, X., Zhao, M., Zeng, M., Mumtaz, S., Menon, V. G., Ding, Z., and Dobre, O. A. Hardware impaired ambient backscatter noma systems: Reliability and security. *IEEE Trans. Commun.* (2021), 2723–2736.

[70] Lia, X., Jianga, P., Chenb, T., and Wenc, X. L. Q. A survey on the security of blockchain systems.

[71] Lindqvist, U., and Jonsson, E. How to systematically classify computer security intrusions. In *Proceedings. 1997 IEEE Symposium on Security and Privacy (Cat. No.97CB36097)* (1997), pp. 154–163.

[72] Liu, C., Liu, H., Cao, Z., Chen, Z., Chen, B., and Roscoe, B. Reguard: Finding reentrancy bugs in smart contracts. In *2018 IEEE/ACM 40th International Conference on Software Engineering: Companion (ICSE-Companion)* (2018), pp. 65–68.

[73] Luu, L., Narayanan, V., Zheng, C., Baweja, K., Gilbert, S., and Saxena, P. A secure sharding protocol for open blockchains. CCS '16, Association for Computing Machinery, p. 17–30.

[74] McCorry, P., Shahandashti, S. F., and Hao, F. Refund attacks on bitcoin's payment protocol. In *Financial Cryptography and Data Security* (Berlin, Heidelberg, 2017), J. Grossklags and B. Preneel, Eds., Springer Berlin Heidelberg, pp. 581–599.

[75] Miller, A., Kosba, A., Katz, J., and Shi, E. Nonoutsourceable scratch-off puzzles to discourage bitcoin mining coalitions. In *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security* (New York, NY, USA, 2015), CCS '15, Association for Computing Machinery, p. 680–691.

[76] Miller, A. K., Litton, J., Pachulski, A., Gupta, N., Levin, D., Spring, N., and Bhattacharjee, B. Discovering bitcoin ' s public topology and influential nodes.

[77] Min, T., and Cai., W. A security case study for blockchain games .

[78] Moubarak, J., Filiol, E., and Chamoun, M. On blockchain security and relevant attacks. In *2018 IEEE Middle East and North Africa Communications Conference (MENACOMM)*. (2018), pp. 1–6.

[79] Nakamoto., S. Bitcoin: A peer-to-peer electronic cash system.

[80] Nakayama, K., Moriyama, Y., and Oshima, C. An algorithm that prevents spam attacks using blockchain. *International Journal of Advanced Computer Science and Applications 9* (01 2018).

[81] Narayanan, A., Bonneau, J., Felten, E., Miller, A., and Goldfeder, S. *Bitcoin and Cryptocurrency Technologies: A Comprehensive Introduction*. Princeton University Press, USA, 2016.

[82] Oppliger, R. Certification authorities under attack: A plea for certificate legitimation. *IEEE Internet Computing 18*, 01 (January 2014), 40–47.

[83] Peng, W., Li, F., Zou, X., and Wu, J. A two-stage deanonymization attack against anonymized social networks. *IEEE Transactions on Computers 63*, 2 (2014), 290–303.

[84] Ping-Chen, X. Sql injection attack and guard technical research. *Procedia Engineering 15* (2011), 4131–4135. CEIS 2011.

[85] Qu, Z., Dong, Y., Qu, N., Li, H., Cui, M., Bo, X., Wu, Y., and Mugemanyi, S. False data injection attack detection in power systems based on cyber-physical attack genes. *Frontiers in Energy Research 9* (2021), 57.

[86] Rahman, M. A., and Mohsenian-Rad, H. False data injection attacks with incomplete information against smart power grids. In *IEEE Global Communications Conference (GLOBECOM)* (2012), pp. 3153–3158.

[87] Ramos, S., Pianese, F., Leach, T., and Oliveras, E. A great disturbance in the crypto: Understanding cryptocurrency returns under attacks. *Blockchain: Research and Applications* (2021), 100021.

[88] Rathod, N., and Motwani, P. D. Security threats on blockchain and its countermeasures. *IRJET Journal 5* (November 2018), 1637–1642.

[89] Ravindra, V., and Grama, A. De-anonymization attacks on neuroimaging datasets. In *Proceedings of the 2021 International Conference on Management of Data* (New York, NY, USA, 2021), SIGMOD/PODS '21, Association for Computing Machinery, pp. 2394––2398.

[90] Rosenfeld, M. Analysis of hashrate-based double-spending.

[91] Saad, M., Kim, J., Nyang, D., and Mohaisen, D. Contra-*: Mechanisms for countering spam attacks on blockchain memory pools, 05 2020.

[92] Saad, M., Njilla, L., Kamhoua, C., and Mohaisen, A. Countering selfish mining in blockchains.

[93] Saad, M., Spaulding, J., Njilla, L., Kamhoua, C., Shetty, S., Nyang, D., and Mohaisen, D. Exploring the attack surface of blockchain: A comprehensive survey. *IEEE Communications Surveys Tutorials 22*, 3 (2020), 1977–2008.

[94] Samreen, N. F., and Alalf, M. H. Smartscan: An approach to detect denial of service vulnerability in ethereum smart contracts.

[95] Samreen, N. F., and Alalfi, M. H. Reentrancy vulnerability identification in ethereum smart contracts. In *2020 IEEE International Workshop on Blockchain Oriented Software Engineering (IWBOSE)* (2020), IEEE, pp. 22–29.

[96] Samreen, N. F., and Alalfi, M. H. A survey of security vulnerabilities in ethereum smart contracts. *ArXiv abs/2105.06974* (2021).

[97] Sayeed, S., and Marco-Gisbert, H. Assessing blockchain consensus and security mechanisms against the 51 *Applied Sciences 9*, 9 (2019).

[98] Schrijvers, O., Bonneau, J., Boneh, D., and Roughgarden, T. Incentive compatibility of bitcoin mining pool reward functions. pp. 477–498.

[99] Shi, H., Wang, S., Hu, Q., Cheng, X., Zhang, J., and Yu, J. Fee-free pooled mining for countering pool-hopping attack in blockchain. *IEEE Transactions on Dependable and Secure Computing 18*, 04 (July 2021), 1580–1590.

[100] Singh, S., Hosen, A., and Yoon, B. Blockchain security attacks, challenges, and solutions for the future distributed iot network. *IEEE Access* (2021), 13938–13959.

[101] Singh, S. K., Salim, M. M., Cho, M., Cha, J., Pan, Y., and Park, J. H. Smart contract-based pool hopping attack prevention for blockchain networks. *Symmetry 11*, 7 (2019).

[102] Son, S., and Shmatikov, V. The hitchhiker's guide to dns cache poisoning. In *SecureComm* (2010).

[103] Sonnino, A., Bano, S., Al-Bassam, M., and Danezis, G. Replay attacks and defenses against cross-shard consensus in sharded distributed ledgers. In *2020 IEEE European Symposium on Security and Privacy (EuroS P)* (2020), pp. 294–308.

[104] Stevens, M., Bursztein, E., Karpman, P., Albertini, A., and Markov, Y. The first collision for full sha-1. In *CRYPTO* (2017).

[105] Sun, H., Ruan, N., and Su, C. How to model the bribery attack: A practical quantification method in blockchain. In *Computer Security – ESORICS 2020* (Cham, 2020), L. Chen, N. Li, K. Liang, and S. Schneider, Eds., Springer International Publishing, pp. 569–589.

[106] Suri, N. Distributed systems security knowledge area issue 1 . 0.

[107] Syta, E., Jovanovic, P., Kogias, E. K., Gailly, N., Gasser, L., Khoffi, I., Fischer, M. J., and Ford, B. Scalable bias-resistant distributed randomness. In *2017 IEEE Symposium on Security and Privacy (SP)* (2017), pp. 444–460.

[108] Szalachowski, P. Towards more reliable bitcoin timestamps. *ArXiv abs/1803.09028* (2018).

[109] Szalachowski, P., Reijsbergen, D., Homoliak, I., and Sun, S. StrongChain: Transparent and collaborative Proof-of-Work consensus. In *28th USENIX Security Symposium (USENIX Security 19)* (Santa Clara, CA, Aug. 2019), USENIX Association, pp. 819–836.

[110] Torres, C., Schütte, J., and State, R. Osiris: Hunting for integer bugs in ethereum smart contracts, 12 2018.

[111] Tosh, D., Shetty, S., Liang, X., Kamhoua, C., Kwiat, K., and Njilla, L. Security implications of blockchain cloud with analysis of block withholding attack.

[112] Tsankov, P., Dan, A. M., Drachsler-Cohen, D., Gervais, A., Buenzli, F., and Vechev, M. T. Securify: Practical security analysis of smart contracts. *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security* (2018).

[113] Vokerla, R. R., Shanmugam, B., Azam, S., Karim, A., Boer, F. D., Jonkman, M., and Faisal, F. An overview of blockchain applications and attacks. In *2019 International Conference on Vision Towards Emerging Trends in Communication and Networking (ViTECoN)*. (2019), pp. 1–6.

[114] Wang, D., and Wang., P. Offline dictionary attack on password authentication schemes using smart cards. *16th International Conference on Information Security 7807* (January 2015), 221—-237.

[115] WANG, I., LI, M., andHONG LI, Y. H., XIAO, K., and WANG, C. A blockchain based privacy-preserving incentive mechanism in crowdsensing applications. *IEEE Access 6* (April 2018), 17545–17556.

[116] Wang, Q., Ji, T., Guo, Y., Yu, L., Chen, X., and Li, P. Trafficchain: A blockchain-based secure and privacy-preserving traffic map. *IEEE Access 8* (2020), 60598–60612.

[117] Wang, X., Chellappan, S., Boyer, P., and Xuan, D. On the effectiveness of secure overlay forwarding systems under intelligent distributed dos attacks. *IEEE Transactions on Parallel and Distributed Systems 17* (July 2006), 619—-632.

[118] Wang, X., Zha, X., Yu, G., and Ni., W. Attack and defence of ethereum remote apis.

[119] Wang, Y., and Li, G. Detect triangle attack on blockchain by trace analysis. In *2019 IEEE 19th International Conference on Software Quality, Reliability and Security Companion (QRS-C)* (2019), pp. 316–321.

[120] Wondracek, G., Holz, T., Kirda, E., and Kruegel, C. A practical attack to de-anonymize social network users. In *2010 IEEE Symposium on Security and Privacy* (2010), pp. 223–238.

[121] Wu, M., Miller, R. C., and Little, G. Web wallet: Preventing phishing attacks by revealing user intentions. In *Proceedings of the Second Symposium on Usable Privacy and Security* (New York, NY, USA, 2006), SOUPS '06, Association for Computing Machinery, pp. 102—-113.

[122] York, D. Chapter 3 - eavesdropping and modification. In *Seven Deadliest Unified Communications Attacks*, D. York, Ed. Syngress, Boston, 2010, pp. 41–69.

[123] Zamani, E., He, Y., and Phillips, M. On the security risks of the blockchain. *Journal of Computer Information Systems* (2018).

[124] Zhang, P., Xiao, F., and Luo, X. A framework and dataset for bugs in ethereum smart contracts, 09 2020.

[125] Zhang, S., and Lee, J. H. Mitigations on sybil-based double-spend attacks in bitcoin. *IEEE Consumer Electronics Magazine 10*, 5 (September 2021), 23–28.

[126] Zhou, Z., Zhang, C., Wang, J., Gu, B., Mumtaz, S., Rodriguez, J., and Zhao, X. Energy-efficient resource allocation for energy harvesting-based cognitive machine-to-machine communications. *IEEE Trans. Cognit. Commun. Networking* (2019), 595–607.