5-2018

Security Threats Classification in Blockchains

Jamal Hayat Mosakheil
St. Cloud State University, jamalkhanhayat@gmail.com

Security Threats Classification in Blockchains

by

Jamal Hayat Mosakheil

A Starred Paper

Submitted to the Graduate Faculty of

St. Cloud State University

in Partial Fulfillment of the Requirements

for the Degree

Master of Science

in Information Assurance

May, 2018

Abstract

Blockchain, the foundation of Bitcoin, has become one of the most popular technologies to create and manage digital transactions recently. It serves as an immutable ledger which allows transactions take place in a decentralized manner. This expeditiously evolving technology has the potential to lead to a shift in thinking about digital transactions in multiple sectors including, Internet of Things, healthcare, energy, supply chain, manufacturing, cybersecurity and principally financial services. However, this emerging technology is still in its infancy. Despite the huge opportunities blockchain offers, it suffers from challenges and limitation such as scalability, security, and privacy, compliance, and governance issues that have not yet been thoroughly explored and addressed. Although there are some studies on the security and privacy issues of the blockchain, they lack a systematic examination of the security of blockchain systems. This research conducted a systematic survey of the security threats to the blockchain systems and reviewed the existing vulnerabilities in the Blockchain. These vulnerabilities lead to the execution of the various security threats to the normal functionality of the Blockchain platforms. Moreover, the study provides a case-study for each attack by examining the popular blockchain systems and also reviews possible countermeasures which could be used in the development of various blockchain systems. Furthermore, this study developed taxonomies that classified the security threats and attacks based on the blockchain abstract layers, blockchain primary processes and primary business users. This would assist the developers and businesses to be attentive of the existing threats in different areas of the blockchain-based platforms and plan accordingly to mitigate risk. Finally, summarized the critical open challenges, and suggest future research directions.

Keywords: Blockchain, Digital Ledger, Security and Privacy, Scalability, Security Threats

Acknowledgements

“Research is what I’m doing when I don’t know what I’m doing.” - Braun

I would like to express my sincere gratitude to my advisor Dr. Abdullah Abu Hussein for the continuous support of my Starred paper related research, for his patience, motivation, and immense knowledge. His guidance helped me in all the time of research and writing of this research paper. I could not have imagined having a better advisor who helped me not only in this research study but also mentor and guides me in my personal life.

Besides my advisor, I would like to thank the rest of my Starred paper committee: Dr. Susantha Herath, Dr. Tirthankar Ghosh, for their insightful comments and encouragement, but also for the hard question which incented me to expand my research from various perspective. Thank you, Professor Herath, for your academic advising, support and guidance during my master’s degree at St. Cloud State University. I would like to thank you again, Professor Ghosh, for offering me the practical knowledge about the Information Assurance.

I would like to thank the Fulbright program to sponsor my study at St. Cloud State University. Last but not the least, I would like to thank my family: my parents, and my brothers and sister for supporting me spiritually, mentally throughput writing this research paper and in my life in general.

Table of Contents

List of Tables

List of Figures

Chapter I: Introduction

A Blockchain is a distributed, decentralized ledger or database that facilitate the process of recording transaction (digital events) in the business network (Swan, 2015). In other words, A Blockchain is a distributed, transactional database that is shared across all the nodes participating in the network. Each transaction in the public ledger is verified by consensus of a majority of the participants in the network. Once the transaction is verified in the block and added to the blockchain, it is nearly impossible to erase or mutate the records. Bitcoin is the first implementation of Blockchain, introduced in 2009 (Nakamoto, n.d.). Bitcoin is a cryptographically secure electronic payment system, or cryptocurrency, that uses peer-to-peer (P2P) technology, and it operates without any trusted third-party authority such as a bank, or any other centralized institutes. The owner of Bitcoin can use it anywhere, at any time without involving any centralized authority. Since the introduction of Bitcoin, Blockchain has shown promising application prospects and attracted a lot of attention from both academia and industry. The reason for interest in the Blockchain is its features that provide security, anonymity, and data integrity, without any third-party involvement in the transaction control.

Blockchain technique has been applied in many other disciplines, including Internet of Things, healthcare, supply chain, software engineering, cybersecurity and so on. With the introduction of smart contracts, blockchain marks the start of blockchain 2.0 era. A smart contract is a computer program that directly controls the transfer of assets between parties under the specific conditions. A smart contract not only defines rules and penalties around an agreement in the same way as a traditional contract does, but it can also automatically enforce those obligations. Ethereum is a next generation smart contract and decentralized application platform, leveraging the blockchain, that introduced the smart contract (Buterin, 2014). Ethereum is now the most widely used smart contract platform.

Since blockchain technology has already attracted many industries, specifically the FinTech (Financial Technology) industry, but there are some core challenges still creates huge concerns such as security, privacy, scalability, compliance, and governance.

According to CoinCapMarket site, there are currently 1560 Blockchain based projects operating and providing various services. These platforms offer services including, financial services, smart economy, IoT, green energy, online education, social media, audit, cloud computing, engineering, healthcare, cybersecurity and many others. With the market capitalization of 117 billion dollars (April 7, 2018) Bitcoin is the top cryptocurrency, which is a distinct target of adversaries. Ethereum is the second largest used blockchain platform which provides smart contracts. Most of the existing blockchains platforms are either based on the Bitcoin or Ethereum blockchain protocols.

Why are hackers increasingly focusing on blockchain based platforms? From the business perspective, there are four attack surfaces in the blockchain platforms. (1) Most the existing blockchain platforms provide digital currencies (cryptocurrencies) for monetary purposes to charge their services. For instance, Sia-blockchain based decentralize storage platform, requires individuals to pay in Sia coin to rent the storage. Furthermore, to buy digital currency (suppose Sia coins) digital currency exchanges are used. (2) Cryptocurrency exchanges are businesses that allow customers to trade cryptocurrencies for other assets, such as conventional fiat money, or other digital currencies. (3) Moreover, some of the proof-of-work based-blockchain platforms such as Bitcoin, Ethereum and others, need mining to authenticate transactions and release new coins, as there is no central authority like Bank to take care of these functions. (4) To access coins, users need their private keys. Private keys are stored in wallets, and the wallet provides the services of holding users' private keys and also create transactions (on user's behalf) in the blockchain systems. As we can see, the above four areas of blockchain platforms (cryptocurrency, exchanges, mining, and wallet) involve interaction with money, and it encourages adversaries to perform various high-profit hacks and theft. Although blockchain is considered as a tamper-proof ledger, still, different security and privacy threats involve in blockchain platforms, which makes financial services industry wondering whether blockchain technology can be made secure enough from criminals.

Problem Statement

Since the launch of Bitcoin in 2009, there are almost 1560 blockchain-based platforms on the market. These blockchain-based platforms serve either directly as an electronic payment system (Bitcoin) or use digital currency for financial purposes to charge their services (Ethereum). The use of digital currency (cryptocurrency) by almost every public blockchain platforms has grown the economy of cryptocurrencies at an enormous rate, and it's now worth about $259 billions of dollars. The market capitalization for Bitcoin itself now worth about $117 billion dollars. This exponential growth in the market value of cryptocurrencies motivates adversaries to exploit the weakness in these platforms for profit, and researchers to discover new vulnerabilities in the systems, propose countermeasures, and predict upcoming trends.

Various attacks including double-spending, transaction malleability, Sybil attack, attacks on Blockchain network, attacks on mining pools, bugs in smart contracts and in wallets software create huge security and privacy concerns in the market. The research (Luu & Velner, n.d.), discovered that among 19,366 exiting Ethereum smart contracts, 8,833 of them are vulnerable, including the DAO bug which led to a 60 million US dollar loss in June 60. Hacker exploits a recursive calling vulnerability in DAO.

Mt. Gox, a Bitcoin exchange based in Tokyo, Japan, reported losses of USD 4.6 million due to hack (transaction mutability in Bitcoin) in April 2013 (Park & Park, 2017). Mt. Gox, which had already suffered losses due to hacking, again reported losses of 470 million dollars' worth Bitcoin due to hacking in February 2014 and subsequently filed for bankruptcy. Bitfinex a Hong-Kong-based bitcoin exchange reporting losses of USD 65 million due to hacking in August 2015 (Park & Park, 2017). Recently on January 26, 2018, Japan-based Coincheck, one of the largest cryptocurrency exchange in Japan, was hacked and loss of USD 534 million worth XEM, the native cryptocurrency of NEM (Asia, 2018). According to the Imperva Incapsula, a cloud-based service provider, report (Q3 2017 Global DDoS Threat Landscape), more than 73% of all bitcoin sites using the Imperva services were attacked in the third quarter of 2017 ("Global DDoS Threat Landscape | Q3 2017 | Incapsula," n.d.).

Nature and Significance of the Problem

Although there are some studies on the security and privacy issues of the blockchain, they lack a systematic examination of the security of blockchain systems. This research conducted a systematic survey of the security threats to the blockchain systems and surveyed the related attacks. It also provides a case-study for each attack by examining the popular blockchain systems and also reviews possible countermeasures which could be used in the development of various blockchain systems. Finally, this study classifies each of the security threats and its attack vectors based on the blockchain abstract layers, blockchain primary processes and primary targets (business perspective). This systematic study would help the blockchain developers and researchers, and businesses in understanding the nature of the various security threats. For instance, the security threats that could affect the different layers of the blockchain systems or blockchain processes, and finally the targeted business users.

Objective of the Study

The primary objective of this study is to review the existing vulnerabilities in the Blockchain and how the current vulnerabilities lead to the execution of various security threats to the normal functionality of the Blockchain-based platforms.

- In order to understand the problem significantly, this study discussed in details the various components of the blockchain, essential characteristics, types of blockchains, applications, architecture, abstract layers, primary processes, and consensus mechanisms.
- The study discussed the existing technical challenges and limitations in the blockchain systems, particularly the scalabilities challenges and reviewed some advancements.
- Studying scalability issue helped in analyzing the tradeoff between security and scalability.
- Furthermore, conducted a systematic survey that covered the security aspects of the blockchain. It included to study the various security attacks and provided case-study and
- Developed taxonomies that classify the security threats and attacks in corresponding to affected blockchain layers, processes, and business users,

- Finally, summarized the critical open challenges, and suggest future research directions.

Study Questions

1. What are the existing vulnerabilities in the Blockchain?
2. How the existing vulnerabilities lead to the execution of various security threats?
3. What are the attack vectors for the security threats?
4. Which abstract layer(s) of the blockchain are targeted or affected by certain attacks?
5. Which blockchain primary process(es) are targeted or affected by certain attacks?
6. From business perspective who are the primary target of the certain attacks?

Limitations of the Study

This study mainly focuses on the security aspects of both Blockchain 1.0, and Blockchain 2.0. It did not provide insights on the privacy and anonymity aspect of the blockchain systems. There are various privacy-related threats to the existing blockchain systems notably Bitcoin

Definition of Terms

Blockchain terminologies ("Comprehensive Blockchain Glossary," n.d.).

- **Blockchain:** A blockchain is a shared ledger where transactions are permanently recorded by appending blocks.
- **Block:** Blocks are packages of data that carry permanently recorded data on the blockchain. A collection of transactions gathered into a block that can then be hashed and added to the blockchain.
- **Bitcoin:** Bitcoin is the first decentralized, open source cryptocurrency that runs on a global peer to peer network. It is the first application of blockchain.

- **Cryptocurrency.** Also called digital currency, or tokens, are representations of digital assets.
- **Central Ledger.** A ledger maintained by a central agency.
- **Distributed Ledger:** Ledgers in which data is stored across a network of decentralized nodes. A distributed ledger does not have to have its own currency and may be permissioned and private.
- **Address:** Cryptocurrency addresses are used to send or receive transactions on the network.
- **Mining:** Mining is the act of validating blockchain transactions. The necessity of validation warrants an incentive for the miners, usually in the form of coins. It is also called block reward.
- **Difficulty:** This refers to how easily a data block of transaction information can be mined successfully.
- **Fork:** Forks create an alternative version of the blockchain, leaving two blockchains to run simultaneously on different parts of the network.
- **Node:** A copy of the ledger operated by a participant of the blockchain network.
- **Consensus:** Consensus achieved when all participants of the network agree on the validity of the transactions, ensuring that the ledger are exact copies of each other.

Principles of Information Security, 5th edition, defined following terminologies as:

- **Vulnerability:** Weakness or fault that can lead to an exposure.
- **Threat:** Generic term for objects, people, other entity that represent a constant danger to an asset (via attacks).

- **Threat agent:** Specific object, person who pose a potential danger (by carrying out an attack). DDoS attack is a threat; if a hacker carries out a DDoS attack, he is a threat agent.
- **Vector:** How the attack was carried out, e.g., malicious email attachment.
- **Exposure:** A successful attack.
- **Incident:** Any attack, or all attacks using vulnerability X, etc.
- **Risk:** Probability that "something bad" happens times expected damage to the organization.

Summary

In this chapter, the introduction to the blockchain has been described, including the problem statement which is the reviewing the existing vulnerabilities in the blockchain and how they lead to various security threats. Furthermore, the significant of the problem, objective of the study, research questions and how this study is designed and conducted are discussed.

Chapter II: Background and Review of Literature

This chapter discussed the concept of the blockchain including the key characteristics, applications, types of blockchain, how blockchain works, and consensus mechanisms. Furthermore, the chapter discussed the efforts that have been put so far various researchers study and addressed the security and privacy aspects of the blockchain

Blockchain Overview

Blockchain definitions: Different perspectives. Blockchain is a public electronic ledger, similar to the relational database, that can be openly shared among the different users and that creates an unchangeable record of their transactions, each one time-stamped and linked to the previous one (Mearian, 2018). Each digital record or transaction in the thread is called a block, and it allows either an open or specific set of users to participate in the digital ledger. Blockchain can only be updated by consensus between the participants in the network, and when new data is entered, it can never be changed or erased which provides high data integrity in the blockchain. The blockchain contains a verifiable record of each and every transaction ever made in the system.

Bitcoin is the first application of Blockchain and the Bitcoin based Blockchain is a public ledger system that maintain the integrity of transaction. Satoshi Nakamoto the founder of Bitcoin defines Bitcoin as a peer-to-peer electronic cash that allows online payments to be sent directly from one party to another without going through a financial institute (Nakamoto, n.d.). According to Don and Alex Tapscott "The blockchain is an incorruptible digital ledger of economic transactions that can be programmed to record not just financial transactions but virtually everything of value." Virtually anything of value can be tangible assets like cash, car, house or intangible such as copyrights, patents, or branding.

From network perspective, Blockchain is distributed file system where participants keep copies of the file and agree on the changes by consensus. The file is composed of blocks and each block includes a set of transactions plus main data that includes, timestamp and a cryptographic signature (hash) of the previous block, hash of the current block, and some other information. The hash of the previous block ties the current block to the previous block and also the subsequent blocks will require the hash of the current block, so these all block are chained together. If anything in the block is modified, one could compute its hash and will find a different value as the stated one and will not accept the block. Thus, including the previous block's hash in the current block integrates the entire system history into the current block.

The blockchain is commonly used for data storage. The data could be financial transactions, which are recorded in the digital ledger and it is replicated across a number of systems in almost real-time. The blockchain (digital ledger) usually exists over a peer-to-peer network and uses cryptography and digital signatures to prove identity, authenticity and enforce read/write access rights. The digital ledger can be written by the certain participants and can be read by a broader audience. Each block in the blockchain is cryptographically linked with the previous block which makes it hard to change the historical records or at least make it easy to detect when someone is trying to tamper (Zheng, Xie, Dai, Chen, & Wang, 2017).

The key characteristics of Blockchain technology are, decentralization, immutability, anonymity, persistency and auditability. Blockchain technology is an integrated multi-field infrastructure, that contains cryptography, mathematics, Algorithm and economical model, peer-to-peer networks and using distributed consensus algorithms to solve the traditional database synchronization problem (Zheng et al., 2017).

Key Characteristics of Blockchain

Digital. All the information on blockchain is digitized, eliminating the need of manual documentation.

Distributed. Transactions are grouped into blocks for processing and standard network protocol ensures every node (participant) receives every transaction in near real-time and applies the same rules. Blockchain is designed to be distribute and synchronized across the networks, which makes it ideal for multi-organizational business networks such as supply chain or financial associations. It also encourages organizations to come out from behind their firewalls and share data (Pattison, 2017).

Decentralization. All participants (nodes) have own copy of all data in the system and no need for a central authority. This helps to obtain no single point of vulnerability or failure. In conventional centralized transaction system, each transaction needs to be validated through a central authority (e.g., bank) which requires some service fees, time and performance bottlenecks at the central servers. However, there is no central authority in the blockchain network, and no middle man/authority service fees is required, and also make the transaction faster. Consensus algorithms used to maintain data consistency in decentralized, distributed network (Zheng et al., 2017).

Immutability. Data is immutable in the blockchain. Once the participants agreed on a transaction and recorded, it is nearly impossible to delete or rollback transactions once they are included in the blockchain. If someone try to subsequently record another transaction about that asset to change its state, but s/he cannot hide the original transaction. This gives the idea of provenance of assets, which means that for any asset you can tell where it is, where it has been and what has happened throughout its life (Pattison, 2017).

Consensus. There are standard algorithm/mechanism used to ensure all nodes agree on the integrity of transaction data in the system, replacing the need for a trusted third party. Before one can execute a transaction, there must be an agreement between all the participants that the transaction is valid. This process is known as “consensus” and it helps keep inaccurate or fraudulent transactions out of the blockchain. Blocks that includes invalid transactions could be revealed immediately.

Anonymity. Each user can interact with the blockchain with a generated address, which does not reveal the real identity of the user, but participants can see the transaction (Zheng et al., 2017). It is arguable the bitcoin blockchain cannot guarantee the perfect privacy preservation due to its intrinsic constraints but there are some other alternative blockchain protocols that claims for providing highest privacy.

Traceable. Every transaction added to a public or private blockchain is digitally signed and timestamped, which means that organization can trace back to a specific time for each transaction and identify the corresponding party (through their public address) on the blockchain (Swan, 2015). So, every block is immutably and verifiably linked to the previous block. A full history can always be reconstructed right back to the beginning (the genesis block). For example, Bitcoin blockchain stores data about user balances based on the Unspent Transaction Output (UTX-O) model. If any transaction has to refer to some previous unspent transaction, once the current unspent transaction is recorded into the blockchain, the state of that unspent transaction will be switched to spent (Vyas & Lunagaria, 2014). Therefore, it is easy to verify and trace the transaction.

Smart Contracts. Blockchain provides the functionality of smart contracts, or scripts that automatically execute when certain conditions are met. For instance, users of Ethereum–Ether (alt-cryptocurrency) exchange must meet the pre-defined conditions that prove someone owns the cryptocurrency and have authority to send the money they claim to own. It is also possible to develop smart contracts that require more than one set of inputs to trigger a transaction (Mearian, 2018). For example, real estate transactions require sign offs between buyers, sellers and their financial institutions.

Application of Blockchain

Besides cryptocurrencies blockchain has many other applications. In financial services, blockchain can be used for asset management, insurance, cross-border payments. Blockchain can be used to facilitate distributed cloud storage. Blockchain has compelling use cases in the Internet of Things (IoT); encrypting the smart appliances on blockchain protects individual ownership and enables transferability. Blockchain can provide a robust solution for supply chain sensors. Blockchain can store, manages, protects, identify and transfers smart information for supply chain sensors.

Smart Contracts are digital code which is embedded with an if-this-then-that (IFTTT) code, which gives them self-execution (Buterin, 2014). Smart contract is one of the profound features of the blockchain. Blockchain not only waives the need for third parties but also ensures that all ledger participants know the contract details and the contractual terms implement automatically once conditions are met. Smart Contract can be used in various areas, such as financial derivation, insurance premiums, property law, crowdfunding agreements and others (Ream, Chu, & Schatsky, 2016). Healthcare can use blockchain smart contract to store and encrypt the personal health records with a private key which grant access only to specific individuals. The same strategy could be used to ensure that research is conducted through HIPAA laws securely and confidentially (Srinivasan, 2017). Besides there, blockchain can be used for other general healthcare management, such as supervising drugs, testing results, managing healthcare supplies, and regulation compliance.

Blockchain provides solutions to the critical problems of ownership rights, royalty distribution, and transparency in the music industry. Blockchain can be used for digital voting, digital identity such as passport, and other certificates and many others.

Blockchain in Cybersecurity.

Protecting data integrity. Preventing data manipulation and maintaining data consistency, and integrity is vital in information systems. Many techniques used for to maintain data integrity are data encryption, data digesting (hash comparison), and digital signature. However, the blockchain built-in features, immutability and traceability, provide organizations with a mean to ensure data integrity (Swan, 2015). The combination of sequential hashing and cryptography with the decentralized structure offers blockchain its most robust features called immutability. It helps protect data integrity and identify any data tampering with it in contrast to a standard database.

Keyless Signature Structure (KSI), a blockchain project led by a startup called GuardTime, aims to replace key-based (PKI) data authentication (Buldas, Kroonmaa, & Laanoja, 2013). KSI stores hashes of original data and files on the blockchain and verify other copies by running hashing algorithms and comparing the results with the stored copy in the blockchain (Dickson, 2017). If any manipulation may happen to the data, it will be quickly discovered because the original hash exists on many of nodes where blockchain is running.

Non-repudiation. Since every transaction added to a blockchain is digitally signed and timestamped, which means that organization can trace back to a specific time for each transaction and identify the corresponding party (through their public address) on the blockchain. This feature relates to an important information security property called non-repudiation, which is the assurance that someone cannot duplicate the authenticity of their signature on a file or the authorship a transaction that they originated [1]. So, this feature help organization to trace and detect any tamper attempts or fraudulent transactions.

Protecting identities. Public Key Infrastructure (PKI) is the most commonly used identity protection technique used to secure emails, websites and other forms of communication. But most of PKI implementations rely on centralized, trusted third party Certificate Authorities (CA) to issue, revoke, and store key pairs for every participant, and hackers can compromise them to spoof user identities and crack encrypted communications. Publishing keys on blockchain eliminates the risk relating to false key propagation and identity theft. Moreover, there are various projects are ongoing to eliminate that need for issuing certificates by central authorities. These advances made blockchain more robust in terms of authentication and elimination of any possibility of being a single point of failure. For example, the CertCoin (Fromknecht & Yakoubov, n.d.) project developed at MIT, is one of the first implementations of blockchain-based PKI. CertCoin aims to removes the Central Authorities altogether and uses the blockchain as a distributed ledger of domains and their associated public keys (Dickson, 2017). CertCoin provides a public and auditable PKI that also does not have a single point of failure.

Preventing distributed denial of service attacks. Distributed Denial of Service Attacks (DDoS) is one of the most common types of attacks, can also cause the most disruption to internet services and hence blockchain enabled solutions. The traditional Domain Name Services (DNS) system relies on DNS caching and due to this the 21st October 2016 DDoS attack on DNS servers which provides DNS services for major websites such as Twitter, Netflix, PayPal, and other services (Woolf, 2016). The October 2016 DDoS attack on DNS cut off access to these websites for several hours which shows another failure of centralized infrastructure. The decentralization and peer-to-peer characteristics of the technology make it harder to disrupt than the conventional distributed application architecture (such as client-server), yet they are also subject to DDoS attack and adequate protection are required both at the network level and application level.

Blockchain has a solution for this problem. Blockchain-based DNS approach improves security by removing the single target that attackers can attack to compromise the entire system. Nebulis is a project that is exploring the concept of a distributed DNS system that is near to impossible to fail under an access of requests. Nebulis uses the Ethereum blockchain and Interplanetary Filesystem (IPFS), a distributed alternative to HTTP, to register and resolve domain names (Dickson, 2017). As the current DNS system highly relies on caching and DNS cache is always the high target of the hackers. So, by using the blockchain approach, it can provide a solution to the single point of failure.

Besides there, blockchain can be used for identity protection or identity security, traditional Endpoint protection, transaction and communication infrastructure security, protection from compromised nodes or server failure and security from malicious insiders.

Eliminating human factor from authentication. Businesses can authenticate devices and users without the need for a password or other multi-factor authentications with the help of blockchain. This eliminates human intervention from the process of authentication, thereby avoiding it from becoming a potential attack vector. The current password-based authentication in the centralized architecture is still a huge problem. Usually, organization invest in security and standard but still if employees and customers use passwords that are easy to steal or crack, then these all efforts are ineffective. Blockchain offers strong authentication and resolving single point of attack at the same time (Eskandari, Clark, Barrera, & Stobert, 2015). Blockchain can be used in an organization to provide a distributed public key infrastructure for authenticating devices and users. This security system can provide each device with a specific SSL certificate instead of a password. Also, management of certificate data is carried out on the blockchain, and this makes it virtually impossible for attackers to utilize face certificates. With the help of blockchain, a security system used in an organization can leverage a distributed public key infrastructure for authenticating devices and users. This security system provides each device with a specific SSL certificate instead of a password. Management of certificate data is carried out on the blockchain and this makes it virtually impossible for attackers to utilize fake certificates. REMME a blockchain based project, authenticate users and devices without the need for a password, and provides SSL certificate to each device instead of password ("REMME Technical Paper," 2017).

Secured decentralized storage. Blockchain can be used to build a decentralized, distributed storage and blockchain users can maintain their data on their computer in their network. This will assure that chain will not collapse. For example, if a hacker attempts to tamper with a block, the entire system (digital ledger) examines every data block to locate the one that differs from the test, it easily can reject the block from the chain and recognize it as fraudulent. As blockchain is decentralized, everyone in the blockchain network is responsible for verifying the data that is shared and maintained to ensure data cannot be altered. This approach facilitates more robust and secure storage as compare the current centralized storage. Some of the blockchain based decentralized projects are Sia (Vorick & Champine, n.d.), and Storj (Wilkinson et al., n.d.).

How blockchain works? The blockchain is a database or a ledger that provides a way for information to be recorded and shared by a community. In this community, each member keeps his or her copy of the information, and all members must validate any updates collectively. It's distributed in nature, meaning that there is no central server holding the entire database or chain, but instead, the participating nodes have a copy of the ledger. The new records are appended to the ledger.

Typically, from record perspective, blockchain consists of two types of elements (Banafa, 2017). Transactions are the actions created by participants in the blockchain network. Blocks record these transactions and make sure they are in right sequence and have not been tampered. Blocks also record the timestamp when the transactions were added. Each block has one or more than one transactions.

Let us suppose A wants to send money to B. First, a block is created online and represents the transaction. Then this block is broadcasted to every participant in the blockchain network and set of participants approves the transaction and validates it. Once the block is validated, it is added to the chain which provides a permanent, non-reputable and transparent record of the transaction. Finally, B receives the money from A. The above steps are shown clarified in Figure 1.

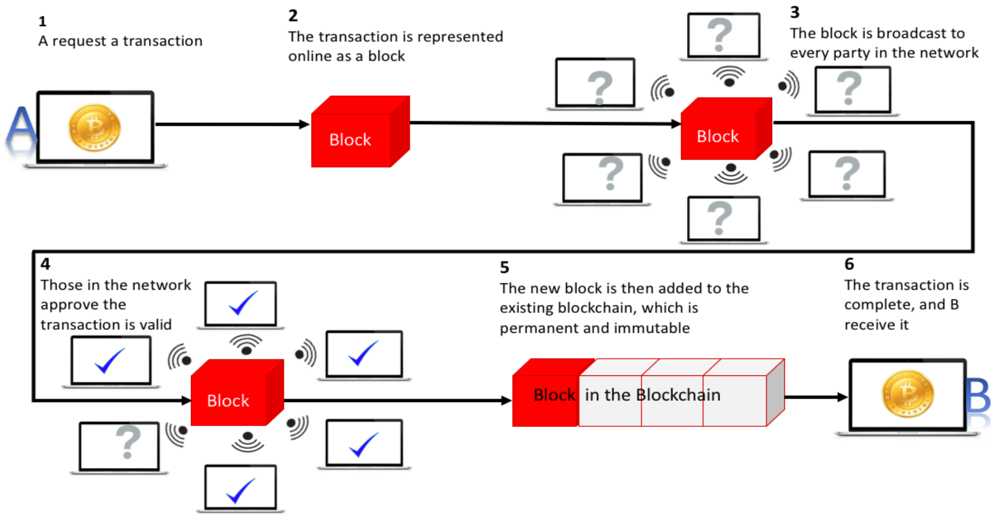
1
A request a transaction
2
The transaction is represented online as a block
3
The block is broadcast to every party in the network
A
Block
Block
4
Those in the network approve the transaction is valid
Block
5
The new block is then added to the existing blockchain, which is permanent and immutable
Block in the Blockchain
6
The transaction is complete, and B receive it
B

Figure 1: How blockchain works simplified.

Following are the detailed processes involved in the blockchain network (Piscini, Guastella, Rozman, & Nassim, 2016).

Following are the detailed processes involved in the blockchain network (Piscini et al., 2016).

Transaction. Two parties want to exchange data, and this could be money, ship a product, contracts, medical records, customer details, or any other asset (tangible and intangible) that can be described in digital form. This transaction (record) is sent to everyone in the system, or in other words, all the participants get notification of the new transaction (Piscini et al., 2016).

There are two types of transactions in the Bitcoin. One is the standard transaction, created by the users which include coin. Another special type of transaction is called Coinbase transaction, or Generation transaction, which is created by the miner to award himself. In fact, it is the reward that miner gets for successful mining a block. The current block reward is 12.5 Bitcoins per block, and the reward halves every 210,000 blocks, or roughly every four years ("Bitcoin Block Reward Halving Countdown," n.d.). In case of Bitcoin, only the UXTO (Unspent Transaction Output Set) is stored in the memory and the remaining data is stored on the disk ("Weaknesses–Bitcoin Wiki," n.d.). Once a client wants to process a transaction, before fetching transaction inputs from disk to memory, the client checks that all the inputs are unspent.

Verification. Depending on the blockchain network's parameters, the transaction is either verified immediately or placed in a queue of pending transactions (Piscini et al., 2016). In this case, the nodes (computers) in the network determine if the transaction is valid or not based on a set of rules of the network, upon which participants in the network were agreed.

Structure. Block contains the transaction or set of transactions, and each block is identified by a 256-bit hash valued, created by message digest algorithm agree upon by the network. In most cases, SHA256 message digest algorithm is used. Typically, a block contains a header, a pointer to previous block hash, and a group of transactions.

Validation. Blocks must be validated before added to the blockchain. The most popular consensus algorithm based on which blocks are validated is Proof-of-Work (PoW), in which the solution to a mathematical puzzle derived from the block's header by miners.

Mining. Miners use Proof-of-Work or other consensus algorithms to solve the mathematical puzzle, and once the problem is solved, the block is validated. The process of mining consists of finding an input to a cryptograhic hash function which hashes below or equal to a fixed target value. It is brute force because at every iteration the content to be hashed is slightly changed in the hope to find a valid hash. In case of Bitcoin, a mining program essentially performs the following simplified code:

```
While(nonce <max):
    If sha256(sha256(block+nonce)) < target:
    return nonce
    nonce +=1
```

The task is to find a nonce which is included in the bitcoin block header, and hashes below a certain value.

In case of Ethereum a simplified example could be:

```
nonce=random int
While hashimoto(block,nonce)*difficulty> threshold
increment nonce
return nonce
```

Where difficulty is a dynamically adusts parameter defined originally in gensis block. If the above code, find the nonce and return it - “return nonce” that is the solution for the puzzle.

The chain. When a block is validated, the miner who solved the mathematical puzzle is rewarded, and then the validated block is broadcasted to the network (Piscini et al., 2016). After validation, the block is added to the majority chain, and by this, the action is completed (if it is money, then it is received to person B from A, as shown in the Figure 4 example).

If a malicious miner tries to submit an altered block to the chain, the hash function of that block and the hash function of all the following blocks will change, and other nodes would detect the changes and reject the block from the majority chain.

Blockchain Architecture

The blockchain is a sequence of blocks, which holds a complete list of transactions records or a log of all transactions like conventional public ledger (Zheng et al., 2017). Figure 2 below illustrates an example of a blockchain. Each block contains block header, parent block hash (hash of the previous block), and list of transactions logs. The first block in blockchain is called genesis block which has no parent block.

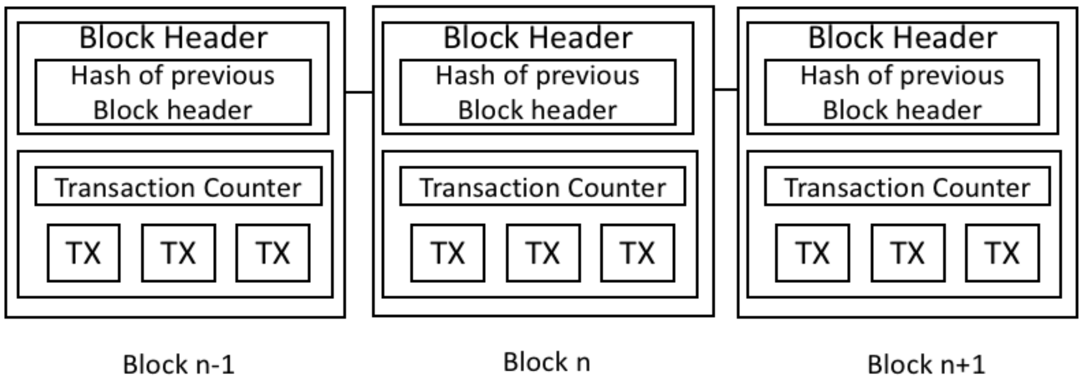

Figure 2: Bitcoin blockchain architecture

Transactions data is permanently recorded in files called blocks. Each block contains block header and a record of some or all recent transactions, and a reference to the previous block. Each block consists of the block header and block body as shown in Figure 3 (Zheng et al., 2017).

Block header. There are six elements in a block header.

Block Version. It is 4 bytes in size that indicates the block version number, that is, it indicates which set of block validation rules to follow. For instance, when Bitcoin core software upgraded then it specifies a new version. Parent block hash. It's a 256-bit hash of the previous block header, and it is 32 bytes in size, and it's updated when a new block comes into the blockchain.

Merkle tree root hash. A 256-bit hash bashed on all the transactions in the block, and it is updated when a transaction is accepted, and it's 32 bytes in size.

nBits. It is 4 bytes in size, and it's a target threshold of a valid block hash, in other words, it is the answer to a difficult-to-solve mathematical puzzle that solved by miners to validate the block and the answer to the puzzle is unique to each block it is the answer to a difficult-to-solve mathematical puzzle that solved by miners to validate the block and the answer to the puzzle is unique to each block.

Timestamp. Current time as seconds in the universal time since January 1, 1970. Timestamp updated every few seconds, and it takes 4 bytes in size of block header.

Nonce. It is a 4 bytes field, and it starts with 0 and increment for each hash calculations.

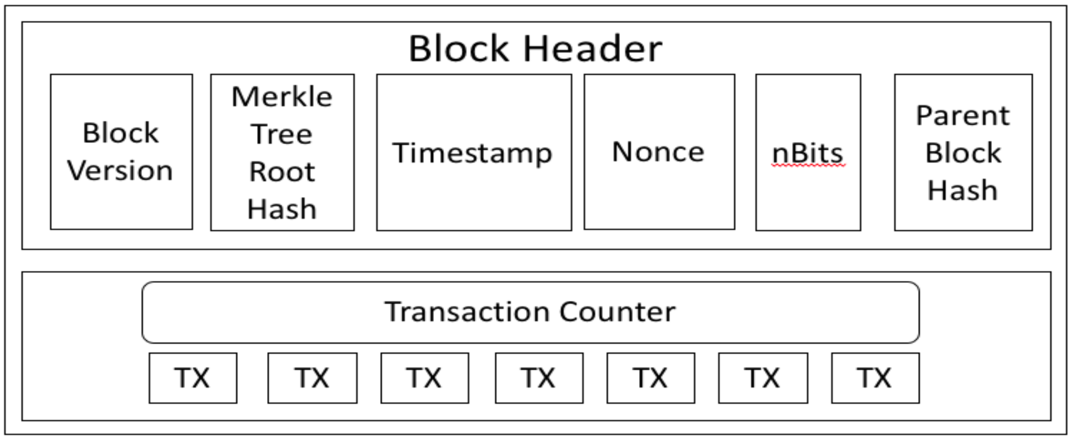

Figure 3: Block structure.

Cryptographic components.

Digital signature. The typical digital signature algorithm used in Blockchain is the elliptic curve digital signature algorithm (ECDSA) (Zheng et al., 2017). Each user who participate in Blockchain network owns a pair of public and private key. The private key is kept confidential and used to sign the transactions and the signed transactions are broadcasted to the whole Blockchain network. The digital signature process involves two steps: signing phase and verification phase (Zheng et al., 2017). For instance, a user Alice wants to send a transaction or message to another user Bob. (1) In the signing phase Alice encrypts her data with her private key and sends to Bob the encrypted result and original data. (2) In the verification phase, Bob verify the value with Alice's public key and check if the data has been tempered or not.

Cryptographic hash function. A hash function is a mathematical process that takes input data of any size and preform an operation and returns output of a fixed size. Blockchain uses SHA256 message digest algorithm for generating Hash values. Data on the Blockchain is hashed in each block and if the block is changed or tempered, the hashed valued would be different and everyone in the network could detect that something had changed. Because the hashed value of the previous block is used to calculate the hashed value of the current block creating this link between the blocks and this makes Blockchain tamper-resilience linked-list.

Merkle tree. Merkle tree is a data structure and it is used in Blockchain to contain a summary of all the transactions in the block, which maintain the integrity of the transactions in the block. In fact, Merkle tree contains cryptographic hashes of the transactions. The most common and simple form of Merkle tree is the binary Merkle tree, where a bucket always consists of two hashes of two adjacent transaction hashes. Merkle tree produces an overall digital fingerprint of the entire set of transactions, and provides a very efficient process to verify whether a transaction is included in a block or not (Antonopoulos, 2015). A Merkle tree is created by recursively hashing pairs of nodes until there is only one hash, called the root, or Merkle root. A Merkle tree is constructed bottom-up fashion. For instance, in the below Figure 4, there are four transactions, T0, T1, T2, and T3 and the hashes of each transactions forms the leaves of the Merkle tree called Hash0, Hash1, Hash2, and Hash3. First, hash of two adjacent transactions for example, Hash0 and Hash1 is calculated and stored in the parent node Hash01, and the same is done for the Hash2 and Hash3 and their hash is stored in their parent nodes Hash23. This process continues until there is only one node left at the top which is called root (Antonopoulos, 2015). The final hash is 32-byte hash that is stored in the block header which summarizes all the data in the four transactions.

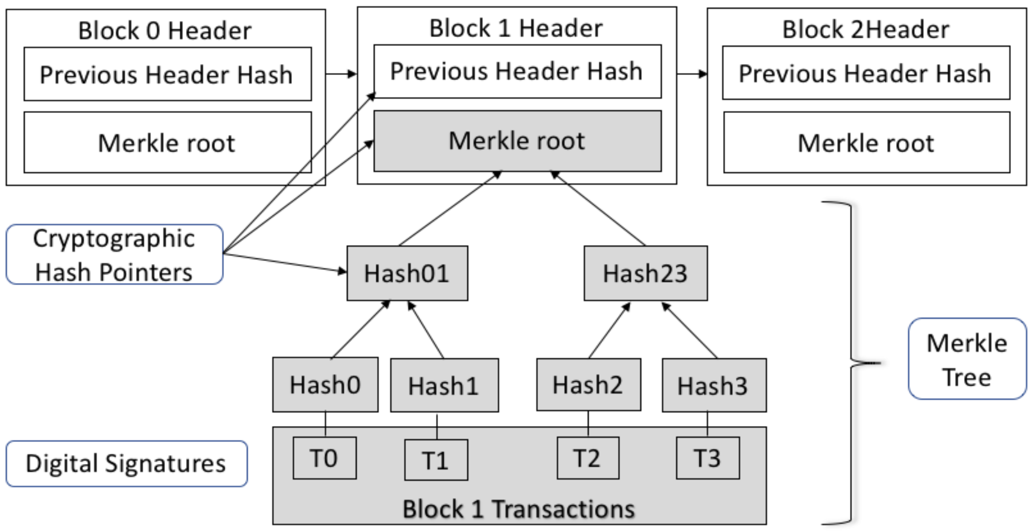

Figure 4. Merkel tree in blockchain's block.

Types of Blockchain

Typically, there are two types of blockchain public and private.

Public blockchain. A public blockchain as its name suggest is the blockchain of public, i.e., anyone can participate in reading, writing and auditing the blockchain without permission. Public blockchain is open and transparent hence anyone can review the transaction at a given point of time. As no one is in charge in public blockchain so the consensus mechanisms such as Proof-of-Work (PoW) and Proof-of-Stack(PoS), and many others are used for decisions making.

Bitcoin, Ethereum, Litecoin and many others are example of public blockchain. Anyone can download the code and start running a public node on their local device, validating transactions in the network, thus participating in the consensus process- the process for determining what blocks get added to the chain and what is the current state ("Blockchains & Distributed Ledger Technologies," n.d.). Anyone can send transactions across the world and expect to see them included in the blockchain once verified and validated. Transactions are transparent in public blockchain, but anonymous or pseudonymous.

Private blockchain. In private blockchain the write permissions are kept centralized to one organization. Read permissions may be public or restricted to an arbitrary extent. Private blockchains are a way of taking advantage of blockchain by setting up groups and participants who can verify transactions internally. This create the risk of security breaches like a centralized system, as opposed to public blockchain secured by game theoretic incentive mechanisms ("Blockchains & Distributed Ledger Technologies," n.d.). Private blockchain has its own use case, particularly, when it comes to scalability and state compliance of data privacy rule and other regulatory issues. MONAX, MultiChain are the examples of private blockchain.

Consortium or federated blockchain. Consortium is sometime considered as third type of blockchain platform, but typically it a special type of private blockchain. This type of blockchain removes the individual autonomy which is responsible for bringing changes in the blockchain as in private blockchain. In consortium or federated blockchains operate under the control of a group of institutions. As opposed to public blockchains, consortium blockchain does not allow everyone to participate in the process of verifying transactions. A pre-selected set of nodes controls the consensus process; for instance, a consortium of ten financial institutions, each of which operates a node and of which seven must sign every block for the block to be valid (Seco, n.d.). The right to read the blockchain may be public or restricted to the participants. Banks usually use consortium blockchain and are faster and scalable and provides more transaction privacy. R3 (banks), EWF (Energy), and B3i (Insurance) are examples of the federated blockchain.

Consensus Algorithms

The purpose of a consensus algorithm is that everyone accepts and supports the decision, understand the reasons for making it, and shares collective responsibility for its consequences. The consensus is one of the fundamental problems of distributed computing. In blockchain, how to reach consensus among the untrusted nodes is a transformation of the Byzantine Generals (BG) problem. In BG problem, a group of generals who command Byzantine army and circle the city. The communication between the generals was by messenger. The problem was some generals prefer to attack while other refuse. To make the attack successful, the generals must agree upon a common battle plan, and all generals have to strike at the same time otherwise the attack may be failed. There is also a possibility that some generals may be traitors who will try to distract the loyal generals. The problem is to find a mechanism to ensures that all generals will reach an agreement and even if the communication is only verbal messages, this problem is solvable if more than two-thirds of the generals are loyal and agree upon the deal (Lamport, Shostak, & Pease, 1982). In blockchain, as there is no central node to control or monitor the distributed digital ledger, so the same problem raised but there must be some consensus mechanisms to solve this problem. Various consensus algorithms are used to bring consensus in the blockchain systems.

PoW - Proof of Work. Bitcoin network uses Proof of Work (PoW) consensus mechanism (Nakamoto, n.d.). Bitcoin blockchain is a decentralized network, and it does not require authorization from any trusted third party to process the transactions. The nodes work with each other in a collaborative environment and develop the blockchain without relying on any central authority. However, some nodes may behave maliciously and act against the common goal. Thus, Bitcoin blockchain runs a fault tolerance consensus mechanism called PoW to assures that all nodes agree on the new entries (blocks) added to the bitcoin blockchain. PoW happens through miners trying to solve extremely difficult mathematic puzzles and finding a solution is basically a guessing game but checking if a solution is correct is easy. In PoW, each node of the network is calculating a hash value of the block header (Gervais et al., 2016). The block header contains a nonce and miners would change the nonce frequently to get a different hash value, and the consensus requires that the calculated value must be equal to or smaller than a particular given value. When one node finds a given value, it will broadcast the block to other nodes, and all other nodes in the network must mutually confirm the correctness of the hash value. Once the block is validated, other miners would append this new block to their blockchains. Nodes that calculate the hash values are called miners, and the PoW procedure is called mining in the Bitcoin blockchain (Gervais et al., 2016).

In Bitcoin blockchain, once the new transactions happen, they are broadcasted to all nodes in the network. Each node collects new transactions into a block and starts works on finding a problematic proof-of-work for its block. Once a node detects a proof-of-work, it broadcasts the block to all nodes. Other nodes in the network accept this new block only if all transactions in it are valid and not already spent. Nodes express their acceptance of the new block by working on creating the next block in the chain, using the hash of the accepted block as the previous has. Nodes in the network always consider the most extended chain to be the valid one and will keep working on extending it (Xu et al., 2017).

In the decentralized network, the valid blocks might be generated simultaneously when multiple nodes find the suitable nonce nearly at the same time. Due to which a new branch will be created. But it is unlikely that two competing forks will generate next block simultaneously (Gervais et al., 2016). In PoW based consensus algorithm, the participants require no authentication to join the network, which makes the Bitcoin consensus mechanism makes it extremely scalable regarding supporting thousands of network nodes. However, PoW based consensus is vulnerable to 51% attacks, in which an adversary has control over 51% of the hash-rate in the network. Hence it can write its blocks or fork the blockchain that at a later point converges with the main blockchain. Fifty-one percent of attacks can further leverage the miner to carried out other types of attacks such as double-spending, eclipse, and denial of service (Xu et al., 2017). The PoW based consensus algorithm wastes a lot of energy in hash computations during the mining process, but, it also facilitates high scalability regarding nodes participating in the network and operates entirely in a decentralized manner. To mitigate the waste of resources, some PoW protocols in which works could have some side-applications have been designed. For example, the Primecoin is a new type of PoW based on searching for prime numbers (King, 2013).

PoS–Proof of Stack. Proof of Stack (PoS) is the most common energy-saving alternative to Proof of Work algorithm. In PoS, instead of investing in computing power in a race to mine blocks, a 'validator' invests in the coins of the system, i.e., the validators in PoS have to prove the ownership of the amount of the currency. It is believed that people with more coins will less likely attack the network. There is no coin creation (mining) exists in the PoS. Instead, all the coins created from day one, and validators (stakeholders) are paid strictly in transaction fees. In PoS, the validator is chosen to develop next block based on the number of coins (stake) has in the system. Once the validator creates a block, that block still needs to be committed to the blockchain, and various PoS systems handle this differently. In Tendermint, every node in the network has to sign off on a block until a majority vote is reached, while in other PoS systems, a random group of participants is chosen to sign the block to commit the blockchain (Vasin, 2014).

There are some problems existed with PoS. Suppose if a validator creates two blocks for the same set of transactions and claim two sets of transactions fees, and also a signer signed both of those blocks. This is called the 'nothing-at-stake' problem, i.e., a participant with nothing to lose has no reason not to misbehave (Vasin, 2014). One proposed solution to this problem is to require a validator to lock their currency in a type of virtual vault, and those coins will severe if the validator tries to double sign or fork the network. The selection of validator based on the account balance is considered unfair because the single richest person is bound to be dominant in the network. Different solutions are proposed to tackle this problem. Peercoin chooses validator based on the coin's age and more extensive set of coins (King & Nadal, 2012). Blackcoin uses randomization to predict the next generator and uses a formula that looks for the lowest hash value in combination with the size of the stake (Vasin, 2014). Compare to PoW, PoS saves more energy and is more efficient. However, as the mining cost is nearly zero, attacks might come as a consequence.

PoA–Proof-of-Authority. In PoA consensus mechanism, transaction and blocks are validated by approved accounts, known as validators. The process of the validation is automated by running software and does not require validators to be continually monitoring their nodes. However, the validator nodes have to keep running and uncompromised. Any one of the validators can approve the non-consecutive block in the PoA-based system, which minimized the risk. Three main rules must be fulfilled for a validator ("Proof-of-authority," 2018). (1) The identity of the validator must be verified correctly, (2) Eligibility must be challenging to obtain, (3) There must be a complete unity in the checks and procedures for establishing authority.

The PoW-based system uses a mining mechanism to generate and validate a block. The PoS-based system uses an algorithm that selects participants with the highest stakes as validators. The PoA-based system uses identity as the only verification of the authority to validate, and there is no mining process to verify the work.

Literature Review Related to Problem

This section briefly reviews the related studies about the problem. Some studies provide a survey about the security and privacy issues on the Blockchain. The authors Conti et al. (Conti, E, Lal, & Ruj, 2017) provided a comprehensive survey on security and privacy issues in Bitcoin. This paper studied the security and privacy works in details but it only focusses the Bitcoin not the blockchain in general. Lin et al. (Lin & Liao, 2017) provide a survey of blockchain security issues and challenges. This paper presents a general overview about blockchain, how blockchain works, its characteristics, some of the Blockchain's application and briefly studies some security problems in Blockchain such as 51% attack, and Fork problems. "Introduction to Security and Privacy on the Blockchain" (Halpin & Piekarska, 2017) paper is based on the IEEE Privacy and Security on the blockchain Workshop (IEEE S&B). They presented peer-reviewed papers that are collected from academia and industry to analyze problems ranging from deploying newer cryptographic primitives on Bitcoin to enabling use-cases like privacy-preserving file storage. This paper mainly focuses on the issues that discussed in the workshop.

Vyas and Lunagaria (2014) state the security concerns and issues for Bitcoin. This paper focuses on Bitcoin protocol and discusses some of the attacks on Bitcoin such as majority attack (51 % attack), Double-spending attack, Timejacking attack, and selfish mining attack. Wüst (2016) completed a master's thesis on Security of Blockchain. In fact, this thesis provides an in-depth study of Proof of Work (PoW) consensus algorithm of Bitcoin. This study provides a quantitative framework based on Markov Decision Processes (MDP) to analyze the security of different PoW Blockchain instances with various parameters against selfish mining and double-spending attacks.

Yli-Huumo, Ko, Choi, Park, and Smolande (2016) state a systematic review on finding out "Where is current research on Blockchain technology?". The objective of this study was to understand the current research topics, challenges and future directions regarding blockchain technology from the technical perspective. This research report studied different papers that discuss various technical challenges of Blockchain Technology. For instance, Blockchain's scalability issues wasted resources and also presents some papers that were dedicated to security and privacy issues of Blockchain. Though this research report discusses the security and privacy aspect of the blockchain more in general and provides the report based on the 14 papers that were covered in the scope of this report. However, this research provides a roadmap for technical challenges of Blockchain that should be researched.

Literature Reviews Related to Methodology

To best of our knowledge, this is the first state-of-arts that discussed the security issues in the both Blockchain 1.0 and Blockchain 2.0. The paper (Atzei, Bartoletti, & Cimoli, 2016) provides a survey on the Ethereum smart contracts and discusses 12 types of vulnerabilities on the Ethereum smart contract. Xu et al. (2017) state a taxonomy of Blockchain-based systems for architecture design. This paper classifies and compares blockchains and blockchain-based systems to assist with design and assessment of their impact on software architecture. This taxonomy discusses the architectural characteristics of blockchain and the effect of their principle design decision, and it does not provide insight on security and privacy issues of Blockchain. Lloyd's London presents a report called “Emerging Risk Report 2015” (Beecroft, 2015) and this report discussed different risk factors specifically in Bitcoin. Lloyd's report studies risk in various domain of Bitcoin such as operational risks, technological risks, market risks and a minor report on security risks in Bitcoin. Cambridge Centre for Alternative Finance conducted a global blockchain benchmarking Study (Hileman & Rauchs, 2017). This benchmarking study discusses the state of the blockchain ecosystem from the finance perspective and very slight attention to the privacy factors of Blockchain.

The Gervais et al. (2016) paper introduced a novel quantitative framework to analyze the security and performance implications of various consensus and network parameters of Proof of

Work (PoW) blockchains. This paper formulates adversarial strategies for double-spending and selfish mining while taking into account real-world constraints such as network propagation, different block sizes, block generation intervals, information propagation mechanism, and the impact of eclipse attack.

Apostolaki, Zohar, and Vanbever (2017) discuss the Bitcoin's Hijacking. This paper provides a taxonomy of routing attacks and their impact on Bitcoin, considering both small-scale attacks, targeting individual nodes, and large-scale attacks, targeting the network as a whole. The paper discusses two general network attacks, partitioning attack and delay the attack.

Wan, Lo, Xia, and Cai (2017) discuss bug characteristics in blockchain systems. This paper performs an empirical study on bug characteristics in eight representatives open source blockchain systems. First, the article manually examines 1,108 bug reports to understand the nature of the reported bugs and then leverage card sorting to label the bug reports and get ten bug categories in blockchain systems. This paper further investigates the frequency distribution of bug categories across projects and programming languages, and finally, study the relationship between bug categories and bug fixing time.

Summary

This chapter summaries the basic concept of the blockchain, its features, applications, types of blockchain, consensus mechanism. Finally, the chapter included the literatures reviews based on the problem and methodology.

Chapter III: Methodology

This study followed the qualitative research methods which involves surveys and case studies. In this chapter, the classification methods and its underlying technologies and concepts required for the taxonomies are presented.

Design of the Study

The study used qualitative research method, because it concentrates on collecting and analyzing data and gain greater insight and knowledge by reproducing or recognizing the problem. The qualitative research method involves survey and case study. To better under understand the security threats problem in the blockchain (both blockchain 1.0, 2.0), the study presented a systematic survey. The systematic survey reviews the existing vulnerabilities in the blockchain, and examines how these vulnerabilities lead to the execution of various security threats. Moreover, the study provided a case-study for each attack by examining the popular blockchain systems and also reviews possible countermeasures which could be used in the development of various blockchain systems. In order to recognize specifically the targets for each threat, taxonomies are developed. The taxonomies classified the each of the examined security threats in term of affected abstract layers of blockchain, affected primary processes of the blockchain and finally the business users.

Since in the qualitative research, the data are analyzed by themes from descriptions by informants and keep that in mind, the study developed taxonomies and classify the threats (Mcleod, 2008).

Moreover, this work also provided a comprehensive study about the blockchain concept, its characteristics, the underlying technologies to better recognize the problem. It also surveyed the existing technical challenges in the blockchain such as scalabilities, usability and some government and compliance concerns. Though the government and compliance was not in the scope of this study, but provided a brief insight about the current standards and government issues.

Data Collection

The data collected and analyzed in this study was from various relevant papers that are extracted from scientific databases. The relevant research included journal papers, conference papers, technical and white papers and used some academic blogs for obtaining up to date statistics to evaluate the objective of the problem discussed in this study.

Tools and Techniques

There is no specific tool used in addressing the problem, but the study used the taxonomy as a technique to evaluate and classify the security threats and attacks in the different domains of the blockchain. The three taxonomies as first based on the abstract layers of the blockchain, the primary technical processes in the blockchain and finally the primary targets (users) involves in the blockchain platforms from the business perspective.

Taxonomy #1. This taxonomy is based on the abstract layers of the blockchain. The abstract layer considers in this taxonomy are: Network layer, Consensus layer, Data model layer, Execution layer. The taxonomy based on the abstract layers is formed by studying the nature of each of the existing vulnerabilities that lead to the security threats and attacks, and which abstract layer can be affect by each attack.

Taxonomy #2. This taxonomy is designed in considering the primary processes of the blockchain platforms. By examining each of the security threats and how it can affect the four primary processes. The four common processes in the blockchain platforms are: Network discovery, Transaction creations, Mining or Block generation and Block validation.

Taxonomy #3. The taxonomy considered the primary target of the business users corresponding to each attack. The actors involved in the blockchain platforms are: User, Merchant, Miner, Mining pool, Cryptocurrency Exchanges, and Blockchain network.

Summary

In this chapter, the approach to the study has been presented. In particular, the study follows a systematic survey the existing vulnerabilities in the blockchain and the related security threats and attacks. Furthermore, using the finding of the systematic survey, the security threats and attacks are classified in term of affected abstract layers, processes and business users.

Chapter IV: Data Presentation and Analysis

This chapter discussed the systematic survey conducted by this study. In particular the systematic survey addressed various area related to the security problem of the blockchain: (1) Reviewed the existing technical challenges in the blockchain such as scalability and usability and studied the solutions. (2) Presented the detail study of the existing vulnerabilities and analyses how the vulnerabilities lead to the execution of the security threats. In particular the systematic survey discussed the security *threats to double spending, threats to mining or mining pool, threats to blockchain network, threats to wallets (client software),* and finally *threats to smart contracts*. (3) Studied each of the security threats and their corresponding attack vectors or attacks by providing a case-study or attack scenario and reviewed the possible countermeasures. Finally, developed three taxonomies and classified each of the security threats in term of *affected abstract layers of the blockchain*, *affected blockchain process,* and finally the *affected business users*.

Blockchain Technical Challenges and Advances

Scalability. Almost all existing Blockchain systems including the Bitcoin, Ethereum, Ripple and their associated consensus protocols have a scalability limitation. The challenging restriction is due to the decentralized nature of the blockchain system-every node on the network processes every transaction and maintains a copy of the entire state of the ledger. Though a decentralization consensus mechanism offers some critical benefits, such as fault tolerance, strong security especially data integrity, political neutrality, and authenticity. However, it comes at the cost of scalability. Bitcoin and Ethereum are two more dominant blockchain protocols that many other existing blockchain systems leverages the components of Bitcoin or Ethereum or both of them. In fact, Ethereum used the Bitcoin protocol and added smart contract functionality. The main scalability problem is the time take to put a transaction in a block, and the time taken to reach a consensus. More preciously following three functionalities or components of the blockchain networks need to be addressed.

Throughput. Bitcoin manages around 7 transactions per second ("Bitcoin Charts & Graphs–Blockchain," n.d.), Ethereum does about 20 transactions per second ("Ethereum Charts and Statistics," n.d.). Other transaction processing network such as VISA controls 1667 transactions per second and PayPal does 193 transactions per second. So, for the Bitcoin and Ethereum to compete with the more mainstream system like VISA and PayPal, they need to increase their throughput. In general, when the frequency of transactions in Blockchain rises to a similar level of VISA, the throughput of the blockchain networks need to be improved.

Latency. It takes currently roughly 10 minutes in Bitcoin network to create or mine a block which contains transaction, for Ethereum it's around 14 seconds ("Bitcoin, Litecoin, Namecoin, Dogecoin, Peercoin, Ethereum stats," n.d.). To achieve efficiency in the security, more time has to be spent on a block creation and validation, to ensure that the inputs for the transactions have not been previously used, which lead to double-spending attacks. Existing blockchain systems need to improve the block creation and validation time, to complete a transaction while maintaining the security.

Size and bandwidth. The current size of the Bitcoin blockchain is 190.65 GB, and Ethereum blockchain size is 330.61 GB (March 23, 2018) ("Bitcoin, Litecoin, Namecoin, Dogecoin, Peercoin, Ethereum stats," n.d.). When the throughput increases to the level of VISA network, bitcoin blockchain could multiply. The current average block size of Bitcoin is 1 MB.

Ethereum uses gas limit mechanism rather than the block size. The time to create a Bitcoin 1 MB block which contains on average 500 transactions takes on average 10 minutes (Yli-Huumo et al., 2016). If the Bitcoin blockchain needs to control more transactions, the size and bandwidth issues have to be resolved.

The current Bitcoin blockchain is globally distributed in 11,875 full nodes ("Global Bitcoin nodes distribution," n.d.), and Ethereum blockchain is globally distributed in 17,263 full nodes ("ethernodes.org–The ethereum node explorer," n.d.) (March 23, 2018). That is, Ethereum is more distributed than Bitcoin blockchain. However, blockchain scalability issue increases as more nodes are added to its blockchain because of the inter-node latency that logarithmically increases with every additional node. Contrary to the traditional database system, in which the performance improved by adding more computer power (servers). But in the decentralized blockchain systems, as every node needs to process and verify transactions and maintain the updated copy of the blockchain the inter-node latency increases. The current existing blockchain systems tradeoff between the low transaction throughput and a high degree of centralization. Thus, to scale the current blockchain protocols, it's important to limit the number of participants needed to validate each transaction, without leading to the risk of centralization and losing the network's trust that each transaction is valid.

Proposed solutions. There are a number of solutions proposed to address the scalability issue of the blockchain, which could be categorized into two types.

Storage optimization of the blockchain. It is harder for the nodes to maintain the full copy of the ledger, Bruce (Bruce, n.d.) proposed a scheme that contains three core components called, mini-blockchain, account tree and proof of chain, to enhance the scalability. The mini-blockchain is a variant of the Bitcoin protocol which aims to eliminate the need for storing the full blockchain and overcome the “blockchain bloat” problem. In this scheme, the old transaction records are removed (or forgotten) by the network, that is, old blocks can be trimmed from the chain. The block headers are kept as a Proof of Work record, but all other transactions can be discarded. The address balances are managed separately in a hash tree structure called the “account tree” which is a self-contained balance sheet designed to keep track of all non-empty addresses. New blocks act upon the entries in the account tree to perform transactions, and the master hash of the account tree is embedded into the block headers to ensure consistency and agreement between the nodes. To secure the whole system from the malicious activities, a chain of interlocking PoW solutions called the “proof chain” is used. The proof chain is merely a chain of block headers which encapsulate all the energy expended by the network on a given chain, and it secures the mini-blockchain and account tree against the attackers. The paper (van den Hooff, Kaashoek, & Zeldovich, 2014) propose a novel scheme called VerSum. Versum allows lightweight clients to outsource expensive computations over extensive and frequently changing data structure, such as the Bitcoin, Namecoin blockchain, or a Certificate Transparency log. VerSum assumes that at least one server is honest in the network, and when servers disagree, VerSum uses an effective conflict resolution protocol to determine which server(s) made a mistake and get the correct output by comparing the outputs from multiple servers.

Redesigning blockchain. The paper (Eyal, Emin, & Sirer, n.d.) proposed a new blockchain protocol called Bitcoin-NG (Next Generation) to scale. The Bitcoin-NG is a Byzantine-fault-tolerant blockchain protocol that is robust to extreme churn and also shares the same trust model as the Bitcoin protocol. The main idea of the Bitcoin-NG is to split the conventional block into two parts: Key blocks for leader election and microblocks that contain the ledger entries. Each block has a header that includes, among the other fields, the unique reference of its predecessors. Bitcoin-NG divides time into epochs or period, and in each epoch, a single leader is in charge of serializing state machine transactions. Miners are competing to become a leader, and once a miner becomes a leader he/she is responsible for microblock generation until a new leader appears. Bitcoin-NG also extended the longest (heaviest) chain strategy where only key blocks count and microblocks carry no weight. In this way, blockchain is redesigned, and the tradeoff between the block size and the network security has been discussed.

Sharding is a concept that has been used by distributed systems for a long time, to improve scalability, performance, and input/output bandwidth. The existing blockchain systems including Ethereum face the big problem of scalability, that is, the speed of transaction verification. Each full node in the network has to maintain the copy of the entire blockchain. With sharding, it can be breaks down a transaction into shards and spreads it among the network. The nodes can work on individual shards in parallel. This in turn decrease the overall time taken by the transaction. Zilliqa (“whitepaper.pdf,” n.d.) is the first public blockchain platform that implement sharding. It automatically split the nodes in the network into parallel chains called “shards”, where each shard processes a small portion of all transaction in conjunction with other shards, resulting in a microblock from each shard. These microblocks are then merged into one complete block that is then added to the blockchain. Zilliqa claims that it is linearly scalable. Linear scalability means that as the number of the participating nodes in the network increases, the transaction throughput also increases at an almost linear rate. However, it is opposite in blockchain, as the number of participating nodes increases, each transaction has to be broadcasted to a greater number of nodes before being validated and added to the blockchain, thereby limiting transaction throughput. Zilliqa uses PoW only to establish miners' identities and it's not used as a consensus protocol, which significantly reduces the energy consumption. Instead, Zilliqa uses an optimized Practical Byzantine Fault Tolerant (PBFT) consensus protocol which give finality to transactions. Unlike the PoW, where multiple confirmations are required, PBFT does not allow temporary forks due to the consensus protocols, that is, once a block added to the blockchain no other block can share the same parent as the committed block, and as a result, no confirmations are required. Because of finality, the entire transaction history is not required to be stored on the blockchain, instead it is sufficient to store only the latest state.

Plasma (Poon & Buterin, n.d.) is another proposed framework for scalability, which is a series of smart contracts that run on the top of the root blockchain (i.e., the Ethereum blockchain). The root blockchain enforces the validity of the state in the plasma chains using a mechanism called "fraud proofs." Fraud-proof is a mechanism by which nodes can determine if a block is invalid using a mathematical proof. The Plasma framework forms the blockchain in a tree hierarchy, and each branch is treated as a blockchain that has its own blockchain history and computations that are MapReduce. The "plasma blockchains" could be considered as the child chains, each of which is a chain within a blockchain. The Plasma blockchain does not reveal the contents of the blockchain on the root chain (e.g., Ethereum). Instead, only the block header hashes are submitted on the root chain, which is enough to determine the validity of the block. The data is sent to the root chain in Byzantine conditions. As a result, the root blockchain process only a small amount of commitments from child blockchains, which in turn decreases the amount of data passed onto the root blockchain and allows for a much more significant number of computations.

Wasted resources, fork, and usability issues.

Wasted resources. Many other existing blockchain systems, Bitcoin and Ethereum are based on the Proof-of-Work consensus mechanism, which requires mining (computational power) to do the proof of work or solve the puzzle. The Bitcoin's current annual electricity consumption is 57.17 TWh ("Bitcoin Energy Consumption Index," n.d.), which is estimated above $15 million/day. Electricity consumed per Bitcoin's transactions is 847 KWh, which is more than a household's energy used in a week. Moreover, a single Bitcoin transaction takes thousands of times more energy than a credit card swap. Ethereum is the second largest electricity consumer blockchain system. Thus, immense computer power, and massive electricity consumption still an enormous challenge to the blockchain system. Besides that, the annual carbon footprint (kt of CO2) of Bitcoin network is 28,015 ("Bitcoin Energy Consumption Index," n.d.) which is an unsustainable and not eco-system friendly. Some of the alternative proposed are Proof-of-Stake, Proof-of-Authority, and many mores, which does not require mining for creating and validating blocks.

Usability. There is no standard available for developing the Application Programming Interface (API), and different blockchains follow different programming languages for development. It makes it difficult to use in term of developing services, notably the Bitcoin's API. There is a need to create a more developer-friendly API for the blockchains that should use some standards such as REST (Representational State Transfer) APIs (Yli-Huumo et al., 2016).

***Versioning, hard forks, multiple chains*.** Usually, updates in the main protocols lead to versioning or hard forks, if all the blockchain community does not have consensus of the new update. This makes the blockchains split and usually happened for administrative or versioning purposes. For instance, after increasing the Bitcoin protocol block size from 1MB to 2MB makes a hard fork (called segwit) in the Bitcoin network and creates a fork or blockchain named Bitcoin Cash, and the blockchain core developers split into two teams. One is supporting the original Bitcoin protocol (BTC), and other advocates the new fork, Bitcoin Cash.

Security Threats to Blockchain Systems

This section provides a details study of each of the security threats to the blockchain systems. It includes the double-spending security threats, mining or mining pool security threats, blockchain network security threats and finally smart contract security threats. Furthermore, this section also discussed the attack scenario for each attack and provided available countermeasures. The list of security threats and attacks are presented in the table 1, and explained in detail as follows.

Table 1

Taxonomy of Blockchain Security Threats

Security Threats	**Attack Vectors**	**Cause**
Double-Spending Threats	Race Attack	Transaction Verification mechanism
	Finney Attack	Transaction Verification mechanism
	Vector76 Attack	Transaction Verification mechanism
	Alternative History Attack	Transaction Verification mechanism
	51% Attack	Consensus mechanism
Mining/Pool Threats	Selfish Mining/Block-discard Attack	Consensus mechanism
	Block-Withholding Attack (BWH)	Consensus mechanism
	Fork-After-withhold Attack (FAW)	Consensus mechanism
	Bribery Attack	Consensus mechanism
	Pool Hopping Attack	Consensus mechanism
Wallet Threats	Vulnerable signature	ECDSA flaws - Poor randomness
	Lack of control in address creation	Public nature of the blockchain
	Collison & Pre-Image Attack	Flaws in ECDSA, SHA256 & RIPEMD 160
	Flawed key generation	Flaws in implementing ECDSA
	Bugs & Malware	Client design flaws
Network Threats	DDoS Attack	External resources, contracts underpriced operations
	Transaction Malleability Attack	Flaws in blockchain protocols -Transaction ID
	Timejacking Attack	Flaws in blochain protocols - timestamp handling
	Partition Routing Attack	Flaws in Internet routing - routing manipulations
	Delay Routing Attack	Flaws in Internet routing - routing manipulations
	Sybil Attack	Structured P2P network limitation - forge identities
	Eclipse Attack	Flaws in blockchain protocols - outgoing connections
	Refund Attack	Flaws in BIP70 payment protocol -Bitcoin refund policy
	Balance Attack	Consensus mechanism
	Punitive and Feather forking Attack	Consensus mechanism - blacklisting transactions
Smart Contracts Threats	Vulnerabilities in contracts source code	Program design flaws
	Vulnerabilities in EVM Bytecode	EVM design flaws
	Vulnerabilities in Blockchain	Program design flaws
	Eclipse Attack on Smart contract blockchain	EVM design flaws
	Low-level attacks	underprice operations

Double-spending Security Threats

A double-spending attack is an attack where a consumer uses the same cryptocurrency multiple times for transactions, i.e., the given set of coins is spent in more than one transaction. For instance, Bob sends money to Alice (merchant) to get some product, Alice then ships the product to Bob, now since nodes always adopt the longer tail as the confirmed transactions, if Bob cloud generate a longer tail that contains a reverse transaction with the same input reference, Alice would be out of her money and her product.

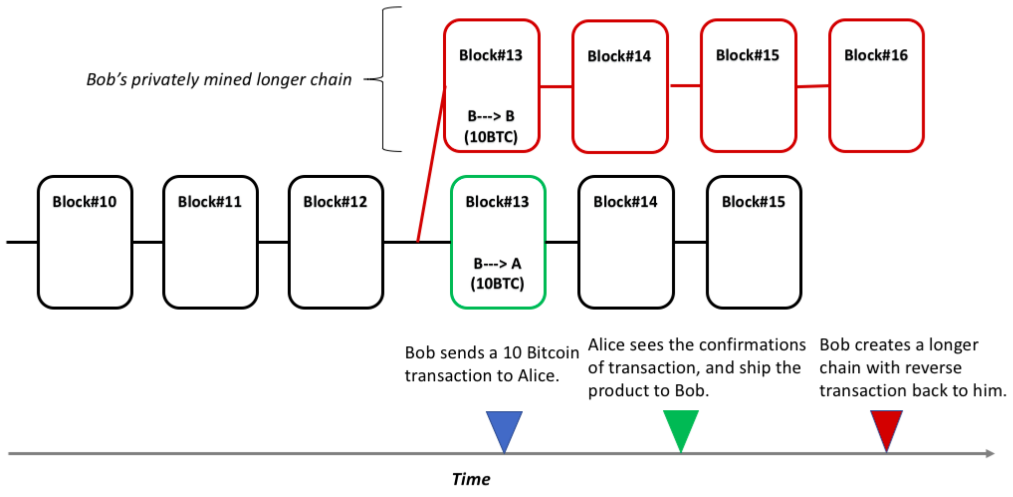
Bob's privately mined longer chain
Block#13
B---> B
(10BTC)
Block#14
Block#15
Block#16
Block#10
Block#11
Block#12
Block#13
B---> A
(10BTC)
Block#14
Block#15
Bob sends a 10 Bitcoin transaction to Alice.
Alice sees the confirmations of transaction, and ship the product to Bob.
Bob creates a longer chain with reverse transaction back to him.
Time

Figure 5. Double-spending attack simplified.

There are various double-spending attack vectors or various ways to perform a double-spending attack, such as Race attack, Finney attack, Vector76 attack, Alternative history attack, 51% attack.

Race attack. Race attack happened when an attacker sends two conflicting transactions in rapid succession into the Bitcoin network. This type of attack is relatively easy to implement in PoW-based blockchains. Merchants who accepts a payment immediately with

"0/unconfirmed" are exposed to the transaction being reversed ("Irreversible Transactions - Bitcoin Wiki," n.d.).

Attack scenario. In Race attack, the attacker in order to do double-spending, sends a transaction paying the merchant directly, and as the merchant accept the payment immediately without waiting for confirmation and ship the product to attacker. Meanwhile, the attacker sends a conflicting transaction (that has the same coins previously paid to merchant) to himself to the rest of blockchain network. It is likely that the second conflicting transaction will be mined into block and accepted by the Bitcoin nodes as genuine, and the coins included in the transaction sent to merchant will be considered invalid. Here, the attacker can exploit the intermediate time between two transactions' initiation and confirmation to launch a double-spending attack quickly.

As the merchants are willing to accept payment (Bitcoin) on 0/unconfirmed, can take precautions such as disable incoming connections or only connect to well-connected nodes. But, still impossible to prevent such type of attack if there is no confirmation mechanism available in the network.

Possible countermeasures. There are a couple of studies and proposals that could be used to detect and used as a precaution to double-spending attack, particularly on the Bitcoin network. Ghassan et al. (Karame, Androulaki, & Capkun, 2012) discusses the double-spending attacks on fast payments in Bitcoin, and propose an attack model that enables the detection of double-spending attacks in fast transactions. The three detection techniques the paper presented are: (a) listening period, (b) inserting observers, and (c) forwarding double spending attempts. The Bitcoin daemon locally generates an error if it receives a transaction whose inputs have already been spent, but this error is not displayed to the Bitcoin user. In the “listening period”, the vendor associates a listening period with each received transaction, and it monitors all the receiving transactions during this period. The vendor only delivers the product, or provide the service, if he does not see any attempt of double-spending during the listening period. Another technique is called “inserting observers”, in which the vendor inserts a node or couple of nodes that it controls within the Bitcoin network called “observer” that would directly relay all the transactions that it receives to the vendor. This helps the vendor to detect a double-spending attempt within seconds by himself or by its observers. The third technique is “communicating double-spending alerts among peers”. This technique is considered an efficient countermeasure to combat double-spending on fast Bitcoin payment. In this technique, the Bitcoin network peers propagate alerts whenever they receive two more transactions that share common inputs and different outputs (double-spending attempt).

Another countermeasure called “Forwarding Double-Spending Attempts in the Network” present in Karame, Androulaki, Roeschlin, Gervais, and Čapkun (2015), states nearby peers should notify the merchant about the attempt that double-spending the same coins in the Bitcoin network. Specifically, whenever a peer receives a new transaction, it checks whether the transaction uses coins that have not been spent in any other transaction that exists in the blockchain and their memory pool. If the transaction does not have already spent coins in it, peers add the transaction to their memory pool and forward it in the network. On the other hand, if peers detect that there is another transaction in the memory pool that spends that same coins to different recipients (address), then peers forward this double-spending transaction to their neighbors (without adding it to their memory pool). The primary intention behind this technique is not to prevent the double-spending attack, but to detect it and inform the vendor so s/he can identify the double-spending transaction before sending or providing service to the attacker.

Podolanko, Ming, and Wright (n.d.) provide a countermeasure against double-spending attacks on Bitcoin Fast-Pay transactions. They proposed a solution called Enhanced Observers (ENHOBS), a hybrid of observers and the peer alert system. In this scheme, the ENHOBS will do more in-depth inspections of all transactions received and compares their outputs and inputs. Once a double-spending attack is detected, an alert message that also contains the double-spending transactions as evidence through the P2P network. Once the alert is received and verified, any transactions that match the same inputs are dropped from the memory pool immediately.

Bamert, Decker, Elsen, Wattenhofer, and Welten (2013) propose couple of countermeasures against double-spending. For instance, the merchant should connect to a sufficiently large random number of nodes in the Bitcoin network. This will make harder for the attacker to inject incorrect transaction information or double-spending transaction because the attacker does not know over which nodes the merchant communicates. Furthermore, the merchant should not accept direct incoming connections. Thus, the attacker cannot directly send a transaction to the merchant, and forcing him to broadcast it over the network. The nodes that will forward that transaction will review it and detected if the transaction has a double-spending attempt. The subsequent transactions using the same input (address) from the attacker would be ignored by those nodes and would make it harder for the attacker to do double-spending.

Finney attack. An attacker, pre-mined one transaction into a block and spend the same coins before releasing the block to public network to invalidate that transaction. This is called a

Finney attack (Sompolinsky & Zohar, n.d.). 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 in the presence of *one-confirmation* vendors.

Attack scenario. To launch a Finney attack, (1) an attacker (A) privately pre-mined a block which contains the double-spending transaction (T_{Aadd}) that he sent to his address (attacker address - A_{add}). (2) The attacker creates another transaction (T_{Madd}) which uses the same coins (suppose same Bitcoins) and sends it to merchant (M_{add}) address. Transactions in the Blockchain network are group by node into a block and broadcast the block to the network, and transactions which are not yet in a block are considered unconfirmed. (3) The attacker waits until the transaction (that he sent to merchant) is accepted by the merchant, and merchant only accepts the transaction, when it receives confirmation from the miners indicating that this transaction is valid and included in the blockchain. (4) Once merchant receives confirmation, he sends the product to the attacker. (5) After receiving the product, the attacker releases the privately pre-mind block into the main (public) blockchain, thus creates a blockchain fork (suppose B_f). Now, if the next mined block in the network extends the fork blockchain (B_f) instead of the main blockchain, then all the miners will create new blocks on the top of the fork blockchain (B_f). As the fork blockchain is the longest chain in the network, the miners will ignore the previous blockchain, and the fork blockchain will be considered the valid blockchain onward. Thus, the block which has transaction (T_{Madd}) which was sent to the merchant and confirmed in the previous blockchain by miners will become invalid and resulting the merchant loses his product and attacker receives the product and his coins back.

Possible countermeasure. Since a Finney attack can only be performed against a one-confirmation vendor. In order to avoid the Finney attack, the vendor should wait for multiple confirmations before releasing the product or providing a service to the client. The waiting for multiple confirmation, will not prevent the double-spending attack, but will mitigate the risk and makes it harder for the attacker to spend the same coins more than once. The countermeasures presented for Race attacks (Karame et al., 2012, 2015) can also be used to mitigate the risk of Finney attack.

Vector76 attack. Vector76 is also called a one-confirmation attack, in which attacker uses the privately mined block to perform a double-spending attack on the exchanges (Sompolinsky & Zohar, n.d.). It is a combination of the race attack and the Finney attack such that a transaction that even has one confirmation can still be reversed ("Irreversible Transactions–Bitcoin Wiki," n.d.). A vector76 attack is possible when a wallet service such as cryptocurrency exchange runs a node that accepts direct (incoming) connections. Assuming that this node is using a static IP address, which will not be difficult for the attacker to find the IP address.

Attack scenario. The attack would be carried out as follows. Now, let's assume that exchange (a digital market where traders can buy, sell or exchange cryptocurrencies) requires only one confirmation and then enables you to withdraw those deposited funds after that deposit transaction gets the one confirmation. To launch the vector76 attack, the dishonest miner (attacker) needs to maintain two nodes (say- Node A and B). The Node A is only connected to the exchange (e-wallet service) node, and the Node B is connected to one or more well-connected peers in the blockchain network. Then, the dishonest miner creates two transactions that spend the same coins. The first transaction T1 (suppose 30BTC) is sent to the attacker Bitcoin address in the exchange. The second transaction T2 (suppose 0.5 BTC) is sent to his own wallet. Both of the transactions are not yet broadcasted to the Bitcoin network. The dishonest miner tries to solve (mine) a block which includes the first transaction T1 (30BTC). Once the block is solved, then instead of releasing that solved block to the public Bitcoin network, the attacker keeps that solved block and instead carry out the T1 (30BTC) on Node A and T2 (0.5 BTC) on Node B simultaneously. Now, the only nodes that know about the 30BTC transaction are the own attacker node (Node A) and the exchange node. The exchange node will propagate the 30BTC transaction to its peer. But the second transaction (0.5BTC) was done by Node B which was connected to well-connected peers in the Bitcoin network, likely has reached to most of all Bitcoin nodes in the network and verified by peers. The nodes in the network that know the 0.5 BTC transaction will reject the 30BTC transaction as it is a double-spend of the same coin. When the attacker see this solved block on Node B, he then broadcast the previously withheld pre-mined block from Node A (which is connected to the exchange). As the exchange provide services based on the one-confirmation and that one-confirmation has been already done by the own attacker node (Node A), the exchange will deposit the 30BTC transaction into attacker account. The attacker would then immediately withdraw the 30BTC from his account. Eventually, the Bitcoin mining network will solve yet another block and that blockchain most likely knows of the 0.5BTC transaction and thus rejects the 50BTC transaction. When the exchange node sees the next block, it will discard the previous block obtained from the dishonest miner (attacker) which includes the 30BTC transaction as that block's chain end at a block height that is below what is now the most extended chain. Thus, the exchange allowed a with

(“Irreversible Transactions–Bitcoin Wiki,” n.d.) draw of the 30BTC even the deposit transaction for that was later reversed. So, in this scenario, the attacker got the 30BTC (Bitcoins) withdraw from his exchange's account and also keeps the 0.5BTC (Bitcoin) that was sent to attacker own wallet, and the exchange lost 30 Bitcoins in the attack.

Possible countermeasure. The protective actions could be, waiting for multi-confirmation, no incoming connections (Bamert et al., 2013), explicit outgoing connections to a well-connected nodes, inserting observers in the network, notify the merchant about the on-going double-spend (Karame et al., 2012, 2015; Podolanko et al., n.d.).

Alternative history attack. The alternative history attack is still possible in case of multiple confirmations but requires high hash-rate and risk of significant expense in wasted electricity to the attacking miner (Lei, n.d.).

Attack scenario. To launch the alternative history attack to carry out the double-spending, the dishonest miner (attacker), submits to the merchant or network a transaction which pays the merchant. Meanwhile, the dishonest miner privately mines an alternative blockchain fork in which a fraudulent double-spending transaction is included. After waiting for the n confirmations, the merchant sends the product to the attacker. At this time, if the attacker finds more than n blocks, he releases his fork and regains his coins; otherwise, he keeps trying to continue extending his fork with the hope of being able to catch up with the network. If the attacker never able to extend his fork compare to the main blockchain fork, then the attack is failed, and he wasted his resource and also lose the reward for the mined blocks. The probability of success for such attack is based on the attacker’s hash-rate as a proportion of the total network hash-rate, and for the number of the confirmations, the merchant waits. Based on the bitcoin network difficulty, it is hard to launch such attack on the Bitcoin network, but it is possible on the blockchain. The Reddcoin blockchain system was the victim of such attack in May 2014 ("[Case Study] 51% attack and double spending," 2014). The Reddcoin team considered it as 51% attac, but eventually the attacker tried to do a multi-confirmation double-spending attack, in which the initial confirmation was 6 and after the attack. The Reddcoin blockchain increases the number of confirmations from 6 to 60. Later on, the Reddcoin blockchain system moved from Proof-of-Work to a Proof-of-Stake variety called Proof-of-Stake-Velocity (PoSV) to mitigate the risk of such attack.

Possible countermeasure. The possible protective measures could be, no incoming connections (Bamert et al., 2013), explicit outgoing connections to a well-connected nodes, inserting observers in the network, notify the merchant about the on-going double-spend (Karame et al., 2012, 2015; Podolanko et al., n.d.).

Fifty-one percent or >50% attack or majority hash rate attack. The blockchain relies on the distributed consensus mechanisms to maintain mutual trust in the network. However, the consensus mechanisms themselves have 51% vulnerability which can be exploited by the attackers to control the entire blockchain network. Though, the blockchain is designed with the assumption that honest nodes control the network. But when a user or group of users (miners) able to take control of more than 50% of the hash power in Proof-of-Work, then the 51% attack may be launched. The 51% attack or >50% is considered the most threatening attack on the blockchain network. It gives power to the attacker to destroy the stability of the whole network including actions such as double spending attack, exclude, modify, and self-reverse transactions and prevent some or all mining of valid blocks for their benefits. If the attacker controls more than half of the network hash-rate, the success rate of Alternative history attack is 100% ("Irreversible Transactions–Bitcoin Wiki," n.d.). Since, the attacker can generate blocks faster than the rest of the network, and maintain his private fork until it becomes longer than the main Bitcoin fork built by the honest network miners. Mainly in Bitcoin network, the 51% attack can lead to the various attacks. For instance, reverse transactions while the attacker is in control and initiates a double-spending attack. Prevent some or all transactions from to get confirmed. Prevent miners from mining any valid blocks. But the attacker cannot prevent transactions from being sent at all. Reverse other user's transactions without their cooperation, or send coins that never belonged to him, or create coins out of thin air**.**

Attack scenario. To launch the 51% attack to carry out the double-spending attack (the same coins are spent twice) the attacker performs the following:

- Attacker submits to the merchant/network a transaction which pays the merchant.
- Meanwhile, create a private blockchain starting from block number N (which includes the double-spending transaction) and do not broadcast it to the network.
- On the public blockchain, exchange coins for goods or services (for example, sell coins for BTC on an exchange).
- Attacker waits for M blocks to be generated on the public blockchain to confirm the transaction (usually in Bitcoin network, merchant waits for 6 confirmation).
- After m confirmation (at block N+M), the provider of goods or services deliver the product to attacker.
- At this time, the private blockchain has generated more than m blocks, therefore longer than the public blockchain.

- Publish the privately mined blocks (private blockchain) to the network and replace the public blockchain, effectively removing the record of the previous transactions for goods or services.
- Attacker received its money and assets back, and successfully launched the double-spending.

The majority attack has never been successfully executed on the Bitcoin network. However, according to the Blockchain.info, in January 2014 the “ghash.io” mining pool reached 42% of the total Bitcoin computing power, and in June 2014 the same mining pool hashrate fluctuated between 40% and 50% of the network’s total power over a week. This force some miners to voluntarily leave the ghash.io mining pool to drop the computing power and eliminate the fear of 51% attack. The Verge (XVG) a privacy blockchain experienced a 51% attack on April 4th, 2018, in which the attacker exploits a bug in the Verge code to exploit the timestamp (Sedgwick, 2018). The attacker stolen around 250,000 verge coins. The attack relented after three hours by initiating an immediate hard fork on the blockchain by the Verge team.

Possible countermeasures. The 51% attack is considered the most worst-case scenario as the adversary can do anything with the network. No amount of confirmation can prevent such attack; however, waiting for the confirmations does increase the aggregated resource cost of performing the attack. As the Bitcoin’s security model relies not on a single coalition of miners controlling more than half of the network hash-rate. So, a miner or a mining pool with more than 50% hash power is an incentive to reduce their mining power and reframe from attacking. Therefore, the primary precaution is that no single miner or mining pool should have more than half of network hash-rate.

Other possible solutions are inserting observers into the network, communicating double-spending alerts among peers (Karame et al., 2012, 2015). The paper (Eyal & Sirer, n.d.), introduced two-phase proof-of-work (2P-PoW), to disincentivize large mining pools. The paper (Luu & Velner, n.d.), proposes a novel approach for particle decentralized pooled mining using smart contracts. This paper claims that 95% of the Bitcoin's and 80% of the Ethereum mining power resides with less than 10 and six mining pools, respectively. This makes the mining more centralized and pose the risk of transaction censorship from pool operators, and open up possibilities for cooperation between mining pools for executing severe attacks, including 51%.

Mining Pools Security Threats

Mining pools are created by a group of miners to work together, pool their resources, and contribute to the generation of a block, and then share the block reward according to the added processing power. In case of the Bitcoin, when a pool solves a block, the 12.5 BTC (Bitcoin) generated by that block's solution reward is split and distributed in between the miners ("Pool vs. solo mining–Bitcoin Wiki," n.d.). The Bitcoin block mining reward halves every 210,000 blocks, so the coin reward will decrease from 12.5 to 6.25 Bitcoins ("Bitcoin Block Reward Halving Countdown," n.d.). There is also solo mining when a miner performs the mining operations alone without joining a pool, and for each discovered block, 12.5 BTC and the transaction fees are paid to the miner. But in case of pools, the transaction fee is kept by the mining pool and not distributed among pool participants. However, the current difficulty level of Bitcoin (3,290,605,988,755) March 16, 2018 ("Bitcoin Block Reward Halving Countdown," n.d.), makes it practically impossible for soloists to make a profit mining unless the solo miner has a vast amount of computer resources. The pool operator or manager controlled the mining pools. The pool operator determines the amount of work done by individual pool members, by using the number of shares, a member find and submit while trying to discover a new block.

Different attack vectors exploit the vulnerabilities in the pool based mining approach. These attack vectors lead to both internal and external attacks on a mining pool. The internal attacks are carried out by some dishonest miners within the pool to collect more than their fair share of collective reward or interrupt the functionality of the honest miner in the pool and distant it from the successful mining attempts (Conti et al., 2017). In external attacks on pools, dishonest miners in the pool could use their higher hash power and perform some double-spending attack. Distributed Denial of Service (DDoS) is another significant external threat to the mining pool.

Most of the attacks on mining or mining pools uses the strategic objective to assure some of the dishonest miners a certain unfair advantage. For instance, to obtain an expected revenue from mining higher their fair share based on the contributed computer power. This strategic objective of dishonest miners leads to launch various types of attacks on the solo miners or pools.

Following chart presents most of the mining pools across the world has Bitcoin network hashrate till March 10, 2018.

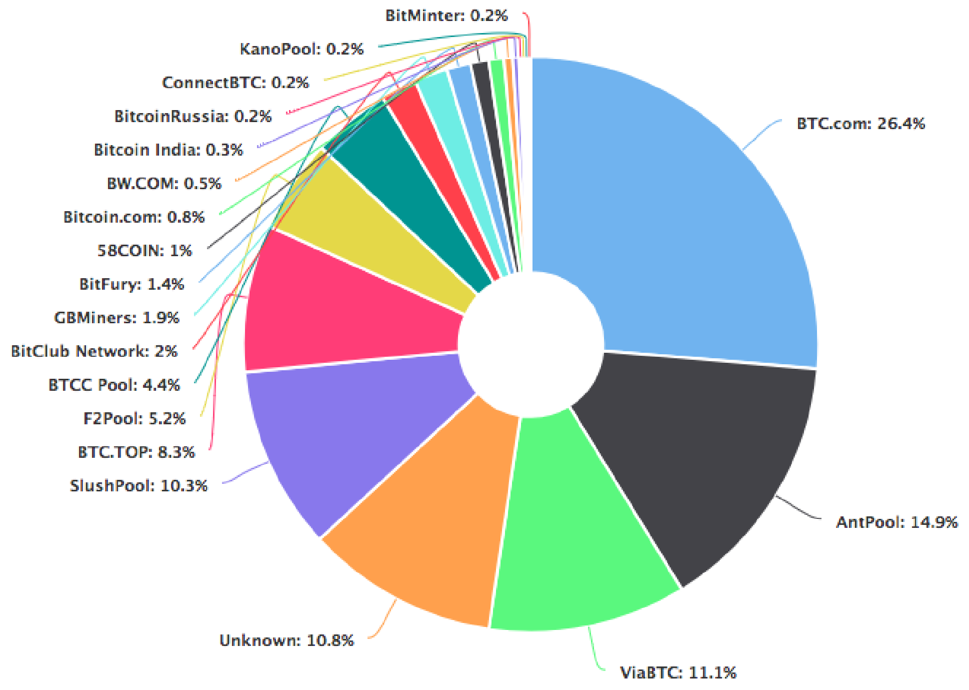

Figure 6. Bitcoin network hashrate distribution across the world, til March 10, 2018 (“Hashrate Distribution,” n.d.).

Selfish Mining Attack. The selfish mining attack was first introduced by by Eyal and Sirer in the paper “Majority is not Enough” (Eyal & Sirer, 2014). There is another other paper that called the selfish mining attack as block-discarding attack (Bahack, 2013). In selfish mining attack, a dishonest miner(s), does not publish and distribute a valid solution to the rest of the network. In this attack, the dishonest miner(s) rather than acting like a regular miner and publishing blocks to the network immediately upon finding them, the attacker selectively releases blocks, or publishing many blocks all at once and thus forcing the rest of the network to discard their blocks and lose revenue (Eyal & Sirer, 2014). The primary motives of selfish mining are to obtain an unfair reward which is bigger than their share of computer power spent, and confuse other honest miners and lead them to waste their resources in a wrong direction.

Attack scenario. In paper “Majority is not Enough” (Eyal & Sirer, 2014) by Eyal and Sirer, states the following strategies for a selfish mining attack. In case a new block is discovered by honest miners, (1) if the size of the public chain (honest branch) is longer than then selfish branch (private chain created by attacker), then the selfish miners tries to set its private branch equal to the public branch. (2) if the selfish branch is one block is longer than the public branch, then selfish miners publish their private chain completely. (3) If the selfish branch is more than one block longer than the public branch, then the selfish miners publish only the head of their private branch. This scenario is simplified in the Figure 7. In case a new block is discovered by selfish miners, they keep this new block private and in case of competition with the honest miners, they publish their private branch to win the race. This scenario is simplified in Figure 8.

Followings are the states to carrying out the selfish attack presented by (Eyal & Sirer, 2014).

Event: The honest miners discover a block.

State 0: Only one single chain. That is, if the public chain is longer than the private chain of the attacker, set the private chain equal to the public chain.

State 1: Dishonest miner (attacker) manages to mine a block, and kept the block private.

State 2: Dishonest miner manages to mine another block and kept it private.

State 2: Honest miners also find a block.

In this situation, dishonest miner(s) published the private chain, and the honest miners lose their block. Because the private chain is longer than the public chain.

State 0: After releasing the private chain, the private chain became the public or main chain, and honest miners will start mining on top of it. The previous block mined by honest miners became invalid.

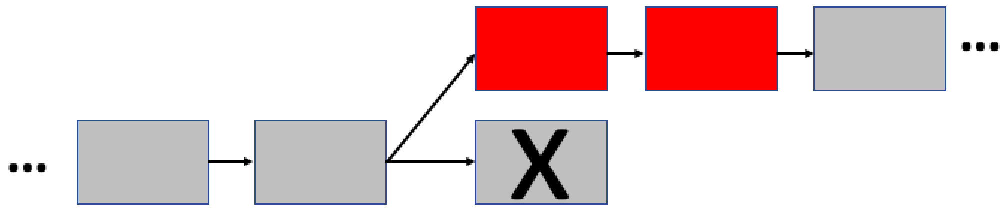

Figure 7. Selfish mining attack with private chain simplified.

Event: The dishonest miner discovers a block.

State 1: A dishonest miner manages to mine a block, and kept it private.

State 0': Honest miners and dishonest miner chain are competing.

Once, the honest miners find a new block as in state 0, the dishonest miner release the private block and hope the honest miners will mine on top of it.

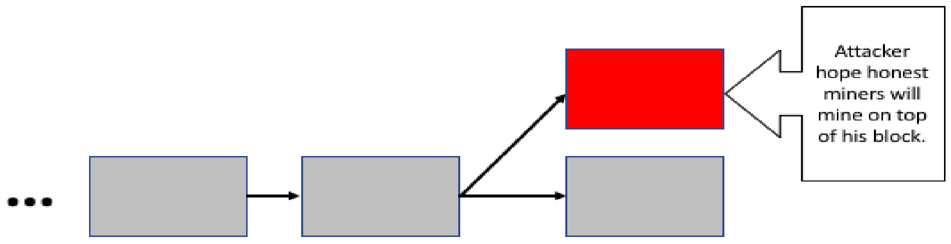

Figure 8. Selfish mining with one private block simplified.

Possible countermeasures. There are different solutions proposed for such type of attack. Bitcoin improvement proposals (BIPS), suggested to randomly assign miners to various branches of pools, to lower the probability of the selfish mining attack. The paper (Heilman, 2014), proposed a solution called Freshness Preferred (FP). In which the block with recent timestamp should be chosen. The FP uses unforgeable timestamps in blocks beside the timestamp that is already existed in each block header. The unforgeable timestamp can prove that a block was mined recently. As in case of the selfish mining, the attacker withholds a block, so using this approach it will decrease the motives for selfish mining because the withheld blocks will lose block race against the newly minted or fresh blocks. Thus, if dishonest miner releases a long list of blocks (private chain) then the rest of honest miners will weight their validity against the timestamp they were hashed and the timestamp they were reported to the network.

The paper proposes an algorithm called ZeroBlock, which uses timestamp-free prevention of block withholding attack (Solat & Potop-Butucaru, 2016). The key idea of this solution is that each block must be generated and received by the network within a maximum acceptable time (mat). If the miners could not mine a new block within mat interval, then miner itself need to generate a dummy block, called ZeroBlock. As in case of block withholding attack, the selfish miners hold the block and publish it later to the public chain, so that private blocks mined by selfish miners could have more value than mat. The dummy block is generated locally by honest miners. It conveys the solved blocks to prove, that the network witnesses the block and that a competing block is absent before miners can work on it. The dummy block includes the index of mat interval and the hash of the previous block, and the computation of each mat interval is done locally by each miner, which is based on the expected delay for a block mining and the information propagation time in the Bitcoin network.

Block-withholding attack (BWH). The paper “The Miner’s Dilemma” (Ittay, n.d.), and the paper by Bag, Ruj, and Sakurai (2017), presented another attack on the mining pool called the block-withholding (BWH) attack. In block withholding attacks, blocks are discarded, and dishonest miners never publish a mined block to sabotage the pool revenue. However, in selfish mining, dishonest miners just kept the mined block secret until the right time to release them. Block withholding attack is usually made by infiltrating another pool.

Attack scenario. The mining pools have two types of users, one is pool manager, and other are regular miners. The pool manager forwards unsolved work units to pool members or miners. Miners in the pool generate partial Proof-of-Work (PPoW) and full Proof-of-Work (FPoW) and submit them to the pool's manager as shares. Once a miner creates a block, it is submitted to the manager with FPoW, and then the manager broadcast the newly generated block to the Bitcoin network to receive the mining reward. Once the manager gets the rewards, it is distributed among the miners based on their PPoW. The paper (Ittay, n.d.) described the block-withholding attack strategy as follows.

As most of the pools nowadays are open pools and an attacker can register with a pool and participate as a regular miner. In block-withholding attack, the pool manager uses some of its miners for its own mining. The manager infiltrates a victim pool (another pool) by registering as a regular miner. He receives some of the work from the victim pool and then sends them to some of its loyal miners in his pool. The mining powers the attacker (manager) redirects towards the victim's task is called the infiltration rate, and the miners are called infiltrating miners. Once the attacker receives the PPoW from his infiltrating miners, he sends them to the victim pool, which estimates their true mining power. But, when the attacker gets FPoW from his infiltrating miners, it withholds and discards them, and therefore it does not contribute to the victim's revenue. The victim pool was thinking that the infiltrating workforce was doing useful mining and shares its revenue with the victim pool, but it tricked. The attacker then distributes its revenue from its own mining and from its infiltration among all its loyal miners. In this scenario, the attacker's mining power is reduced, since he used some of his miners for the block withholding, but he earns additional revenue through his infiltration of the other pool. Also, the total effective mining power in the system is reduced, causing the Bitcoin protocol to minimize the difficulty (Ittay, n.d.).

Possible countermeasure. Since pool registration typically requires only a Bitcoin address for revenue collection and sometimes an email address, so it's challenging for the victim pool to find out who is the attacker. The paper by Courtois, Bahack & Lear (2014) suggests a solution for block withholding attack, that pool manager should only allow trusted miners to register who are personally known to him or her. Also, if the pool revenue goes down than expected from its computational effort the pool should be closed. The paper (Bag et al., 2017) presented a cryptographic commitment scheme to counter BWH attack. This scheme can be implemented by making a small change to the existing Bitcoin protocol which will protect a pool from rogue miners and also from rogue pool managers. This scheme makes it impossible for the miners to distinguish between a PPoW and FPoW, and it is designed that the administrator cannot cheat on the entire pool.

Fork-After-Withholding attack (FAW). The paper by Kwon, Kim, Son, Vasserman, and Kim (2017) introduced the fork after-withholding attack. FAW is another variant of BWH attack. In case of the FAW attack, the attacker's reward is always equal to or greater than that for a BWH attacker, and it is four times more practical per pool than the BWH attack. In fact, the FAW attack combines components of selfish mining and BWH attack.

Attack scenario. To launch a FAW attack, an attacker (individual or pool) do as follows. (1) The attacker divides his computing power between the innocent mining and infiltration mining. The infiltration mining is the mining as part of a target or victim pool. (2) When the attacker finds an FPoW as part of infiltration mining in a pool, he holds that block privately and does not publish it. (3) Depending on what happens next, the privately owned FPoW block can be either release to the target pool manager hoping to create a fork (as in the selfish mining) or discard it (as in block withholding). The attacker will immediately release the privately held FPoW if he notices that other miners who are not part of the target pool and propagating a valid block. The attack does this to create a fork. But the attacker will drop his privately held FPoW in two cases. (a) The attacker notices that another miner in the target pool finds an FPoW, (attacker will still share in the reward from FPoW publish by another miner in the target pool). (b) If the attacker themselves finds another FPoW through their innocent mining and the attacker will receive a greater share of the block reward for the block they located outside of the target pool (Kwon et al., 2017).

Possible countermeasures. There is no efficient solution so far reported and finding a cheap and efficient countermeasure remains an open problem. The same paper (Kwon et al., 2017) suggest a partial solution to reward higher the miner who finds an FPoW in a pool, as compare to other miners who only submit PPoW. For instance, rewards 0.1 BTC to the miner who finds an FPoW in a pool and 0.9 BTC shared among the other miners in the pool according to their PPoW submissions. But, because of the high reward difference, miners may be hesitated to join pools using this reward system.

There are some other attacks that directly or indirectly disrupt the miner or mining pools activities. The paper by Rosenfeld (2011) presented the Pool Hopping attack. Pool-hopping is the exploitation of the circumstance by mining only when the attractiveness is high and living when the attractiveness is low. The attractiveness is the expected earnings. The main idea of this attack is that if a miner mines in a pool in which a lot of shares have already been submitted and no block has yet been found, he will gain less in expectation because the reward will be shared with the miners who have contributed to this pool. So, at a particular moment, it may be profitable to stop mining in this pool and participate in some other pools or mine individually. Usually, it is considered a good practice if the miner will not frequently change the pool based on the attractiveness. The paper by Bonneau (n.d.) discusses a theoretical attack called, Bribery attacks, in which an attacker might purchase mining power for a short period via bribery. The bribery can be performed in-band with the system itself enforcing the bribe. That is, the attacker attempts to bribe using Bitcoin itself by creating a fork which contains the bribe money and available for the miner adopting the fork. Besides in-band payment, out-of-band payment, in which attacker will pay directly to rent their computing power, or by Negative-Fee mining pool, in which the attacker forms a pool by paying a higher return. This attack could lead to Majority attack, or Distributed Denial of Service attack (DDoS) to disrupt the whole Bitcoin network functionality.

Blockchain Network Security Threats

The blockchain peer-to-peer nature of the network, which includes all the nodes who maintain and run the blockchain protocols and provides services come under the blockchain network. In case of the Bitcoin there are two types of nodes: those that accept incoming TCP connection, and generate blocks in the blockchain (miners) and other nodes (users) who only create transaction and submit it into the Bitcoin network. There are various attacks can happen at the Blockchain network layer.

Distributed Denial of Service attacks (DDoS). The distributed, decentralized natures of the blockchain makes it harder for the attacker to launch an effective DDoS attack on the blockchain system than the conventional client-server model. However, blockchain ecosystem is still subject to DDoS attacks. A DDoS is an effort to make target system resources unavailable by overwhelming it with service requests. DDoS attacks can be either network layer or at application layer attacks. The DDoS at network layer cause network saturation by consuming much of the available network bandwidth. The DDoS at application layer attacks, intend to bring down a server by consuming much of its processing resources (e.g., memory or CPU) with a high number of requests. The application layer DDoS attacks usually facilitated by DDoS botnets. According to the Bitcoin forum, there are 23 built-in, prevention mechanisms in the Bitcoin client version 0.8.0, against the denial of service attacks and 7 Bitcoin protocol rules to prevent the denial of service attack in Bitcoin (“Weaknesses–Bitcoin Wiki,” n.d.). Nevertheless, Bitcoin and other blockchain base platform are still vulnerable to sophisticated DDoS, and it becomes the most common attack on the blockchain ecosystem. DDoS main target is crypto exchanges, mining pools, e-wallets, smart contracts and other financial services in the blockchain system. According to the Imperva Incapsula, a cloud-based service provider, report (Q3 2017 Global DDoS Threat Landscape), more than 73% of all bitcoin sites using the Imperva services were attacked in the third quarter of 2017 (“Global DDoS Threat Landscape | Q3 2017 | Incapsula,” n.d.), which means the cryptocurrency exchanges and services are the relatively high target of DDoS attacks. The paper (Vasek, Thornton, & Moore, 2014) performed an empirical study of DDoS attacks, and documented 142 unique DDoS attacks on the Bitcoin services between May 2011 and October 2013. The study found that 7% of all popular operators have been affected by DDoS attacks, but the currency exchanges, mining pools, gambling operators, eWallets, and other financial services are much more likely to be attacked. Between these two studies (2013 and 2017) (“Global DDoS Threat Landscape | Q3 2017 | Incapsula,” n.d.; Vasek et al., 2014), we can conclude that still the DDoS attacks high intension is currency exchages and mining pools and definetely most of the exchanges are using some sort of wallet services.

Possible countermeasures. Various anti- DDoS services are available that could protections for the blockchain network and applications in the market such as CloudFlare, Incapsula, or Cloud Services (Amazon) and many other, if used by the currency exchanges and mining pools can mitigate the risk of DDoS on their services. At the consensus protocol level the paper by Bentov, Lee, Mizrahi, and Rosenfeld (2014) proposed a Proof-of-Authority (PoA) consensus mechanism that combines the PoW components with the PoS. The authors claim that PoA protocol offers good security against the possible attacks on the Bitcoin, and has relatively low penalty in terms of network performance and storage usage.

Transaction malleability attack. The transaction malleability considered as a flaw in the original Bitcoin protocol. The paper by Andrychowicz, Dziembowski, Malinowski, and Mazurek (2015) proposes the transaction malleability attack, and also the paper by Decker and Wattenhofer (2014) did a study to traces the Bitcoin network for over a year and showed that the transaction malleability problem is real and it was not well-known before the closure of MtGox hack (transaction malleability hack). In transaction malleability attacks, attacker tricks his target into believing that a transaction has failed. The attacker then asks for the transaction to be repeated. In this way, the attacker who was already owned X Bitcoins could fraudulently obtain twice the amount. This could be happened by changing the transaction hash (transaction ID) of a Bitcoin transaction before it is confirmed on the Bitcoin network.

Each user in the Bitcoin network may create an arbitrary number of addresses that can be used to send and receive Bitcoins. An address is derived from an ECDSA (Elliptic Curve Digital Signature Algorithm) key pair that is later used to prove the ownership of the Bitcoins associated with that address. The ECDSA is a cryptographic algorithm used by the Bitcoin to ensure that funds can only be spent by their owners. The only operation allowed to modify the address balances is the transaction. A transaction is identified by the SHA256 hash of their serialized representation, which consists of one or more inputs and outputs. The input is used to specify which bitcoins will be transferred, and the output specifies the address that should be credited to the Bitcoin being transferred (Decker & Wattenhofer, 2014). The input, output, number of Bitcoins to send and some other data are cryptographically signed, into a unique piece of data called hash. The hash is essentially the transaction ID. Once the miner confirms the transaction, the transaction ID is included in a block and stored in the blockchain. Now, as any modification to the original data should generate a different hash. Thus, if the attacker can modify somehow the same transaction data, it will produce a different hash. So, two transactions can exist with the same inputs and outputs, valid signature but the different hashes. This is called transaction malleability. The main reason for the success of this attack is that in Bitcoin each transaction is uniquely identified by its transaction ID (hash).

A malleability attack is considered as a variant of the double-spending attack but with some difference. In transaction malleability attack, the attacker is no longer the person who is issuing the transaction (as in double-spending case). Instead, it is the receiving party (Decker & Wattenhofer, 2014). In fact, the attacker would cause the victim to create a transaction that transfers some Bitcoins to an address controlled by the attacker. Once the attacker receives the transaction, the same transaction is then modified by altering the signature of it, without invalidating it. The modified transaction then has a different transaction identification hash. The changed transaction is then broadcasted to the network, and either of the two transactions may later be confirmed. The malleability attack is considered successful if the modified version of the transaction is later confirmed (Decker & Wattenhofer, 2014). Now, if the victim relies only on the transaction identity hash to track and verify its account balance, and the victim will see that transaction it issued has not been confirmed. It will credit the amount to the attacker or attempting to send another transaction at a later time. The attacker would have doubled efficiently the funds the victim sent it.

The current reference Bitcoin core client is not vulnerable to this attack as it tracks the unspent transaction output set by applying all confirmed transactions to it, rather than concluding only from transaction it issued (Decker & Wattenhofer, 2014).

Attack scenario. MtGox a popular Bitcoin exchange suffered from transaction malleability attack that eventually led to the filing for bankruptcy of MtGox. The attacker tried to exploit the vulnerability in the MtGox client that identified both transactions as different, and this leads to the transaction malleability. The attack on the MtGox proceeds as follows: (1) Attacker deposits n Bitcoins in his MtGox account. (2) Then the attacker sends a transaction T to the MtGox to transfer his n Bitcoins back. (3) MtGox issues a transaction T' which transfer the n Bitcoins back to the attacker. (4) The attacker performed the transaction malleability by tweaking the signature data and get the transaction T' which is semantically equivalent to T but has a different transaction identification hash or transaction id. Now assumes that T' gets included into the Bitcoin blockchain instead of T. (5) Once the attacker receives the T', he complains to MtGox that the transaction T was not successful. (6) MtGox did an internal check, and it will not found a successful transaction of T. Thus, MtGox credited the attacker's account again. The problem is that the MtGox should not only search the transaction based on the transaction id but also any semantically equivalent of T (Conti et al., 2017).

Possible countermeasures. The BIP 62 (Bitcoin Improvement Proposal 62) that intended to deal with malleability and listed the sources of malleability ("bitcoincore development–How much of BIP 62 [('Dealing with malleability'] has been implemented?," n.d.). The BIP 62 was supposed to be implemented as soft fork to fix this issue and enforced the new transaction validity rule which consider multiple metrics for transaction verification.

Timejacking attack. Timejacking attack is an attack on the Bitcoin network that is due to a theoretical vulnerability in Bitcoin timestamp handling. In Timejacking attack, attacker try to announce inaccurate timestamps when connecting to a node. Once the network time counter of node is altered by the attacker, the deceived node may accept an alternate blockchain. This could significantly increase the chances of a successful double-spending, drain a node's computational resources, or simply slow down the transaction confirmation rate (Boverman, 2011).

Attack scenario. Each node in the network internally maintains a time counter that represents the network time, which is based on the median time of a node's peers which is sent in the version message when peers connect. But, if the median time differs by more than 70 minutes from the system time, the network time counter reverts to the system time. An attacker could plant multiple fake peers in the network and all these fake peers will report inaccurate timestamps, which could potentially slow down or speed up a node’s network time (Boverman, 2011).

Possible countermeasures. The Boverman (2011) study mentioned a few solutions to overcome this dilemma. (1) Use the node’s system time instead of the network time to determine the upper limit of the block timestamps and when creating blocks. (2) Restricted the acceptance time ranges. That is, the node’s network time could be restricted to a value within 30 minutes. This changes the maximum initial attack window to between 30 and 60 minutes instead of 70 and 140. However, it would not prevent splits entirely, and nodes with incorrect daylight savings handling might be left behind. (3) Use only trusted peers.

Routing attacks. The paper (Apostolaki et al., 2017) introduced the routing attacks on the Bitcoin network. This paper studied the impact of that Internet routing attacks (such as BGP hijacks), and malicious Internet Service Providers (ISP) can have on the Bitcoin network. The routing attacks can impact individual nodes and also target the whole network. This paper states that two key properties make routing attacks practically: (a) the efficiency of the routing manipulation, and (b) the significant centralization of Bitcoin concerning mining and routing. More specifically, the key observation suggests that any attacker with few (<100) hijacked BGP prefixes could partition nearly 50% of the mining power.

A BGP (Border Gateway Protocol) hijack is a routing attack in which an ISP diverts internet traffic by advertising a fake announcement in the internet routing system (“BGP hijacking,” 2018). This research shows that most of the Bitcoin nodes are hosted in few ISPs; 13 ISPs host 30% of the entire Bitcoin network and most of the traffic exchanged between Bitcoin nodes traverse few ISPs. Particularly, 60% of the all possible Bitcoin connections cross 3 ISPs, in other words, 3 ISPs can see 60% of all Bitcoin traffic. These two characteristics make it relatively easy for a dishonest ISP to intercept a lot of Bitcoin traffic (Apostolaki et al., 2017). Thus, because of the extreme efficiency of the Internet routing attacks and the centralization of the Bitcoin network in few networks worldwide (mining pools), the paper shows following two attacks that are practically possible.

Partition attack. Any ISP can partition the Bitcoin network by hijacking few IP prefixes. The goal of a partition attack is to disconnect as set of nodes from the network entirely. This requires the attacker to divert and cut all the connection between the set of the nodes and the rest of the network, and partition the network into disjoint components. By preventing nodes within a partition to communicate with outside nodes, the attacker forces the creation the parallel blockchains. To perform partition attack, the attacker first diverts the traffic and intercepts the Bitcoin traffic (e.g., based on the TCP ports) and identifies whether the corresponding connections cross the partition he tries to create. If So, the attacker drops the packets, if not meaning that the connection is inside the partition P. The attacker keeps monitors the Bitcoin traffic to detect the “leakage points.” Leakage points are nodes currently within the partition P, which maintains connections with nodes outside of P, that the attacker cannot intercept which is called “stealth” connection.

- Attack scenario. The following figure shows the partition attack scenario (Apostolaki et al., 2017).

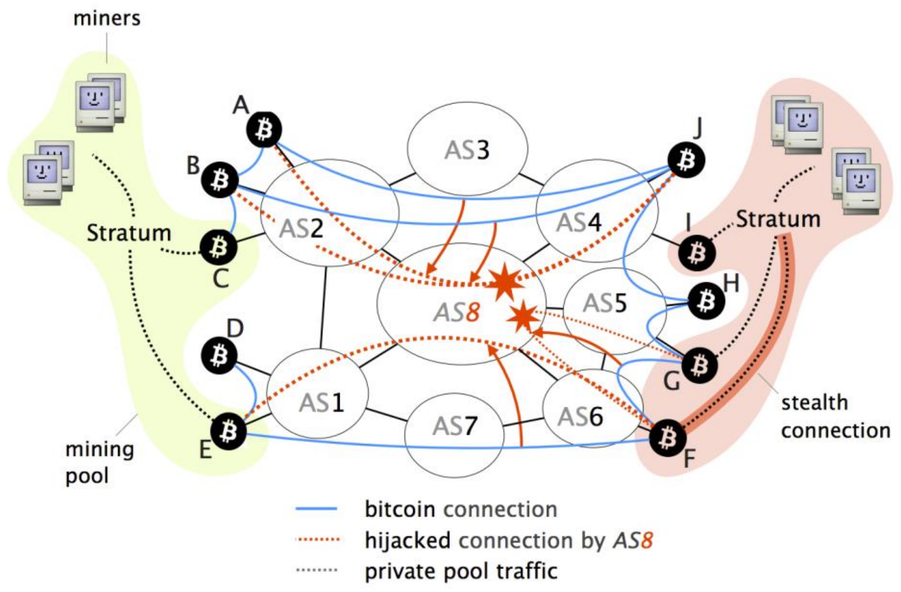

Figure 9: Partition routing attack simplified. How ISP adversary can intercept Bitcoin traffic by hijacking prefixes to isolate the set of nodes P = (A,B,C,D,E,F) (Apostolaki et al., 2017). Here AS8 is the Autonomous Systems or ISP number 8.

The attack is as follows: (1) Nodes of the left and right side of the network communicate through a Bitcoin connections denoted by blue lines. (2) The attacker attracts the traffic destined to the left nodes by performing a BGP hijack. (3) The attacker cuts these connections, effectively partitioning the network into two parts. (4) During the attack, nodes within each side continue communicating with nodes of the same side. After the attack, the attacker can discard all the blocks mined within the smaller partitions including all the transactions and the miner’s revenue.

Delay attack. The goal of this attack is to slow down the propagation of blocks towards or from a given set of nodes. Here an attacker can use routing attacks to the delivery of a block to a victim node by 20 minutes while staying completely undetected. During this period, the victim is unaware of the most recently mined blocks. So, it the victim is a merchant, it is susceptible to double-spending attacks, and if it is a miner, the miner wastes its computation power, and if the victim is a regular node, then it is unable to contribute to the network by propagating the last version of the blockchains.

- Attack scenario. Following figure shows the partition attack scenario(Apostolaki et al., 2017).

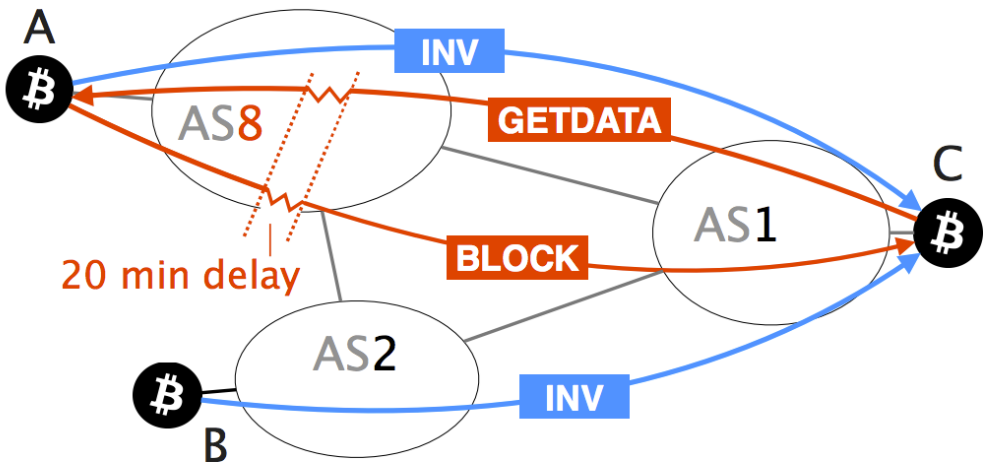

Figure 10: Delay Routing attack simplified. Example of delay attack, how an AS-level adversary (AS8) which intercepts a part of the traffic can delay the delivery of a block for 20 minutes to a victim node C (Apostolaki et al., 2017).

The attack is as follows: (1) Nodes A and B broadcast the same block to the victim node C, using the INV message. (2) Node C request the block via a GETDATA from node A. Then attacker changes the content of the GETDATA such that it triggers the delivery of an older block from node A. (3) The older block is delivered. (4) Shortly before 20 minutes after the original bock requested by node C, the attacker triggers its delivery by modifying another GETDATA message originated by C. (5) The bloc is delivered just before the 20 minutes timeout. The victim does not disconnect from node A.

The potential damage of this attack is disturbing; among other these attacks could reduce miner's revenue and render the network much more susceptible to double-spending. These attacks could also prevent merchants, exchanges and other large entities that hold Bitcoins from performing transactions.

- Possible countermeasures. The paper by Apostolaki et al. (2017) also provides a set of countermeasures against routing attack. Such as Increase the diversity of node connections. That is, the more connected as AS is, the harder it is to attack it. So, it is better, that Bitcoin nodes are multi-homed. Another countermeasure could be select Bitcoin peers while taking routing into account. Bitcoin nodes randomly establish eight outgoing connections, while randomness is vital to avoid biased decisions, Bitcoin nodes should build a few extra connections considering routing. Furthermore, monitor the round-trip (RTT). The RTT towards the hijack destination increases during the attack, so by observing the RTT towards its peers, a node could detect sudden changes and establish extra random connections as a protection mechanism. Also, it is a good practice to use gateways in different ASes. Based on this study they found that most of the pools are using gateways in the same AS. So, by hosting these gateways in different ASes would make it harder for the attacker to launch the routing attacks against the network.

Sybil a ttack. The Sybil attack is an attack wherein a reputation system is weakened by forging identifies in P2P networks (Douceur, n.d.). That is, one attacker with many identities. In a Sybil attack, the attacker subverts the reputation system of a P2P network by creating a large number of pseudonymous identities and then use them to gain a suspiciously large influence. A Sybil attack in Bitcoin network is an attack where a single adversary is controlling multiple nodes in the Bitcoin network. The paper by Douceur (n.d.) shows that, without a logically centralized authority, Sybil attacks are always possible except under extreme and unrealistic assumptions of resources and coordination among entities. Thus, Proof of Work does not prevent a Sybil attack from occurring. However, Bitcoin uses PoW to make it infeasible for a successful Sybil attack to feed false information to the victim. Also, Bitcoin nodes initiating multiple randomly outbound connections to other peers in the network.

Attack scenario. To Sybil attack a Bitcoin node, the attacker has to first identify the victim's node and then replace that all of that node's peers with the attacker nodes. Then the attacker tries to isolate the user and disconnect the transactions initiated by the user or a user will be made to choose only those blocks that are governed by the attacker. If no nodes in the network confirm a transaction that input can be used for double-spending attack.

Possible countermeasures. Bitcoin protocol is considered Sybil resistance, because it considers the true chain to be the one with most cumulative proof of work (not the longest chain as is often incorrectly stated), the result is that a new peer joining the network only needs to connect to a single honest peer to find the true chain. This is also known as Sybil resistance, which means that it's not possible for someone to launch an attack against a node by creating many dishonest peers that provide the false information (Lopp, 2016). A worst-case scenario is when honest node is being massively Sybil attacked but still has a single connection with an honest node that is connected to the true Bitcoin network. As long as a single honest node is passing the true data to the honest full node (which is being attacked), it will ignore all the attempts from the Sybil attacker's nodes.

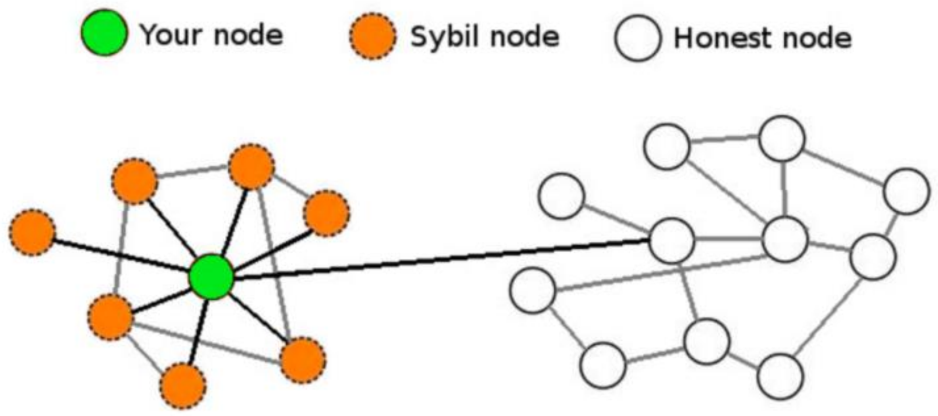

Figure 11: Sybil attack on honest node of Bitcoin network (Lopp, 2016).

The paper by Bissias, Ozisik, Levine, and Liberatore (2014), proposed a two-party mixing protocol called Xim. It is decentralized protocol to simultaneously address Sybil attack, denial of service attacks, and timing-based inference attacks. It is a multi-round protocol that includes a decentralized system for anonymously finding mix partners based on ads placed in the blockchain. Xim's design increases attacker cost linearly with the total number of participants, and by increasing the cost, it can mitigate the Sybil-based DoS attack effects. Usually, in other mixing protocols such as DarkWallet, SharedCoin, and CoinShuffle participants pay only standard transaction fees. The transaction fees are practically negligible; because the cost to the attacker is just the sum of the instances where they are successful, rather than to the number of identities they created. So, if Sybil attackers wishing to maintain a fixed success rate must pay for each participating address, and so their costs grow linearly with the total number of mix participants, while honest participant's costs are fixed and constant with the number of participants (Bissias et al., 2014).

Eclipse attack. In eclipse attack, an adversary able to control a sufficient number of IP addresses to monopolize all connections to and from a victim node (Wang, 2015). That is the attacker force network partition between the public network and a specific miner. If successful the attacker can then exploit the victim for attacks on Bitcoin's mining and consensus systems, including 0-confirmation and N-confirmation double-spending attacks, selfish mining and adversarial forks in the blockchain. To perform an eclipse attack, an attacker can manipulate the node so that all its outgoing connections are to attacker IPs. For this, the attacker need to fill victim node's peer tables with attacker IPs. Once the node is restarted it loses its current outgoing connections. Finally, the node makes new connections only to the attacker IPs. The node restarts frequently occur, because of software updates, a packet of death/DoS attacks, and power or network failures.

To understand the attack scenario for the eclipse attack, some keys Bitcoin network information is essential. For instance, a Bitcoin node has a maximum of 8 outgoing TCP connections, and 117 incoming TCP connections by default. These connections form the gossip network to propagate Bitcoin transactions and blocks. The eclipse attack targets only the Bitcoin nodes that accept incoming connections because not all nodes accept incoming connections.

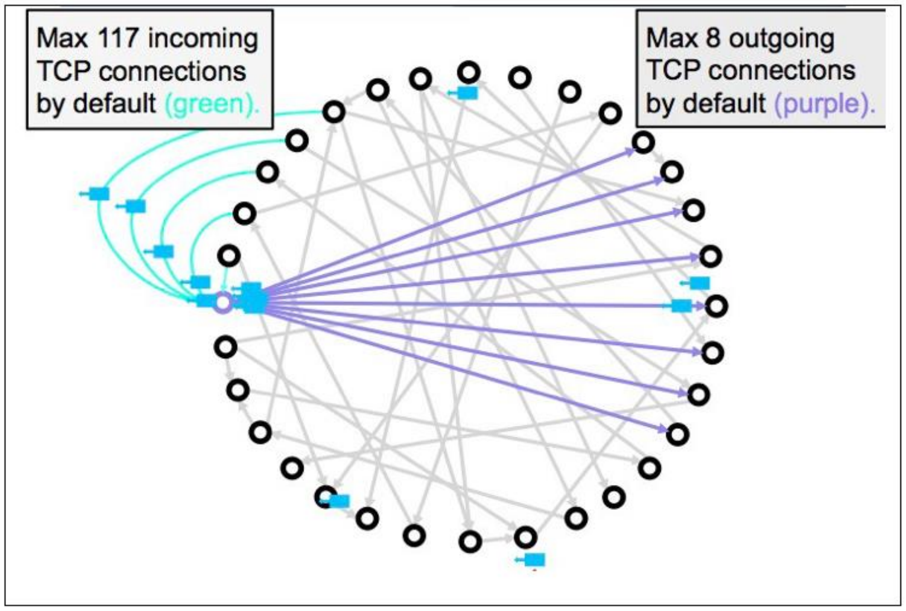
Max 117 incoming TCP connections by default (green).
Max 8 outgoing TCP connections by default (purple).

Figure 12: Bitcoin's P2P network incoming and outgoing connection (Wang, 2015).

With proper manipulation of the P2P network, an adversary can eclipse a node so that it is only communicating with malicious nodes. Moreover, in Bitcoin network, Public IPs are stored in a node's "tried" and "new" tables. These tables are stored on disk and persist when a restart. The tried table stores IP address that a node has successfully made incoming and outgoing TCP connections. The new table stores IP addresses received from DNS seeders or ADDR messages. A DNS seeder is a server that responds to DNS queries from Bitcoin nodes with a list of IP addresses for Bitcoin nodes. The maximum possible number of IP addresses that can be returned by a single DNS query is around 4000. ADDR messages, containing up to 1000 IP address and their timestamps, are used to obtain network information from peers. If more than 1000 address are sent in an ADDR message, the peer who sent the message is blacklisted. Nodes accept unsolicited ADDR messages, and an ADDR message is solicited only upon establishing an outgoing connection with a peer. When a node restarts it randomly selects an IP address from either tried or new tables. If the connection is successful, add that IP address as a new outgoing connection, and this is repeated until eight outgoing connections are established.

Attack scenario. The attack is as follows (Goldberg & Aviv Zohar, 2015): (1) The attacker first gets a large number of IP addresses, e.g., control a distributed botnet. (2) Then the attacker fills the tried table of victim node with attacker-controlled addresses. (3) Overwrite addresses in the new table with “trash” IP address that are not part of the Bitcoin network. For this, the attacker peers send unsolicited ADDR messages filled with “trash.” The “trash” addresses are unallocated or as “reversed for future use” like 252.0.0.0/8 block (“IANA IPv4 Address Space Registry,” n.d.). (4) The attacker force or wait for the victim node be restarted. (5) Once the victim node is restarted, with high probability, victim forms all of her outgoing connections with attacker controlled IP addresses.

Possible countermeasures. The paper by Wang (2015) provides eight countermeasures to makes eclipse attack difficult. Such as random selection of addresses from the tried and new tables. Moreover, diversify incoming connections, that is, a node should accept only a limited number of connections form the same IP address. As nowadays, a bitcoin node can have all of its incoming connections form the same IP address. Also, ban unsolicited ADDR messages. A node could choose not to accept large unsolicited ADDR messages (with >10 addresses) from incoming peers, and only solicit ADDR messages from outgoing connections when its new table is too empty.

Wallet Security Threats

Blockchain-based currencies use private key-based authentication, though passwords remain the most common form of user authentication (Bos et al., 2013). Users’ needs their public and private keys, to access coins or make transactions in the blockchain. A Bitcoin wallet is a collection of private's keys that are used to manage those keys and to make transactions on the Bitcoin network (Eskandari et al., 2015). Bitcoin uses Elliptic Curve Digital Signature Algorithm (ECDSA) which is a variant of the Digital Signature Algorithm (DSA) to sign and validate transactions (Goldfeder, Gennaro, & Kalodner, n.d.). A digital signature of a transaction is an encryption of the transaction hash calculated with a private key. The signature of the transaction can be verified with an associated public key. The digital signature proves that the transaction has not been altered, and the owner of the private key has issued that transaction. In the process of creating signature a pre-selected random value is used along with the private key, and the random value should be different for each transaction.

Various types of key management schemes are used in the blockchain ecosystem. Typically, wallets could be either online (hot - connected to the Internet) or offline (cold-disconnected from the internet. Online and offline wallets could be summarized into four categories; software, hardware, paper and online wallets. Software wallets are applications or client's software that users can download onto their desktop, laptop, or mobile devices to store their private keys local, and are considered online wallet. Different types of software's wallets are available in the market such as Bitcoin core, Bitcoin XT, Armory. Hardware wallets are stand-alone hardware cold-storage that is used to store private key offline. Popular hardware wallets in the markets are Ledger Nano, TREZOR, PI wallet. A paper wallet is a fancy term for printing out your public and private keys on a piece of paper, and sometimes QR barcode is used to store the private key. In online wallets or web-based wallets, private keys are stored in the cloud rather than a local computer. Most of the currency exchange offers web-based wallet facilities.

Private key security threats.

Vulnerable signature. In Bitcoin, the private key is the major authentication component, and Bitcoin mainly reply on the ECDSA to sign and validate transaction. The paper (Bos et al., 2013) states that the ECDSA has insufficient randomness in signature generation, which could lead to compromise of private key. In the process of creating signature a pre-selected random value is used along with the private key, and the random value should be different for each transaction. This research found that 158 unique public keys had used the same nonce value (random value) in more than one signature, and this make it possible to compute these users' private keys. As ECDSA has poor randomness property, and Bitcoin highly used ECDSA, so securing private keys with new cryptographic algorithms besides ECDSA is still an open challenge.

Lack of control in address creation. Bitcoin address is the hash value of a public key encoded with a pair of public and private keys (Park & Park, 2017). The paper by Ateniese, Faonio, Magri, and de Medeiros (2014) states an extension of the Bitcoin protocol that preserves its decentralized nature, while enabling payers to optionally specify the involvement of a trusted authority that attests to the identity of the payee, by requiring payees to use certified Bitcoin addresses. That is, to send and receive Bitcoins only to and from certified users and control of creation of Bitcoin addresses by trusted authorities. The certified addresses are still anonymous within the Bitcoin system, but the authority can validate the legitimacy of the entity to whom it releases a certificate address. The certified addresses are allowed to co-exist with the standard auto-generation Bitcoin addresses. The paper claims that this can help in mitigating identity theft, particularly, considering the case where a man-in-the-middle (MIMT) attacker changes the payee's Bitcoin address for the attacker's address. Moreover, the certified Bitcoin address are blinded, that is, the trusted authority can mint coins on behalf of a particular user, but it cannot spend any of them. Also, the author claims that they mitigate the existing reservations against the adaptation of Bitcoin as a currency in commercial uses. But this scheme has some tradeoff of user privacy concerns with respect to trusted authority.

Possible collison and pre-image attacks. The paper by Giechaskiel, Cremers, and Rasmussen (2016) states that most of the digital currencies including Bitcoin rely on cryptographic primitives to operate, and these primitive are breakable with increased computational power and advanced cryptanalysis.

The possible effects of broken primitives could be collision attacks and pre-image attacks. A collision attack is an attack in which the adversary tries to find two inputs producing the same hash,.i.e., a hash collision, and in contrast to this a pre-image attack where a specific target hash value is specified ("Collision attack - Bitcoin Wiki," n.d.). This research analyzes and reveals the possible of impact of the collision attack on Bitcoin could be repudiate payer, or destroy coins, and pre-image attack could cause uncover address, double-spending, steal coins or completely destroy blockchain. Though for such cryptographic attacks huge computation power is required which could be possible if someone completely dedicate a large size mining pools or has high possibility with quantum computing.

Flawed key generation. Sometimes due to faulty implementations of ECDSA and the associated hash functions for key generations creates weakness in wallets which lead to exposure of private keys. This happened in December 2014 to hybrid wallet provider called Blockchain.info (Beecroft, 2015). This mistake happened during the update of the code for the Blockchain.info, and this affected the inputs to the ECDSA algorithm were not sufficiently random to generate an effective one-way function. Due to lack of randomness, it’s easy to determine private key using user’s public key. This issue was resolved within two-and-a-half hours (Beecroft, 2015). However, the flaw in the implementations is often possible which could leak of private keys.

Bugs and malware threats. The paper by Wan et al. (2017) provides a large-scale empirical study on bug characteristics in blockchain systems that examine 1,108 bugs reports. This research discussed various types of bugs in blockchain client software’s both such as semantic bugs (runtime), memory, concurrency, performance, security, GUI, configuration, build, compatibility and hard fork bugs in the blockchain systems including the client software. There are also various reports about the bugs in the hardware wallets. Recently author Rashid (2018) claims and demonstrate how an Evil Maid attack is possible on the currently popular hardware wallet named, Nano S. Ledger. This attack would allow extracting the PIN, recovery seed and BIP-39 passphrases used, and lead to disclosing of all the private keys in the wallet. To launch this attack, an attacker needs to update the MCU firmware of ledger using some social engineering or using an infected computer. This vulnerability is shared with the Nano S. ledger team to be fixed. The reports *Risks and opportunities for systems using blockchain and smart contracts* (Staples, 2017) and *Cryptocurrency Mining Malware* (2017) argue and predict the possible malware that could affect the various cryptocurrencies.

Smart Contract Security Threats

A smart contract is a program or scripts that automatically execute when certain conditions are met. Ethereum blockchain was designed as a smart contract platform, and the most well-known and used framework for smart contracts (Buterin, 2014). The Ethereum Virtual Machine (EVM) is a runtime environment where smart contracts run in the Ethereum. A good metaphor is that the EVM is a distributed global computer where all smart contracts are executed (Araoz, 2016). Every single operation that is executed inside the EVM is simultaneously executed by every node in the network. To limit the resources used by each contract run in the EVM, a mechanism is used called gas. Each operation of contracts has a cost measured in gas, and each gas unit consumed by a transaction paid for in Ether, based on the gas/Ether price which changes dynamically (Araoz, 2016). Ether is a cryptocurrency like Bitcoin used by Ethereum (Buterin, 2014).

The paper by Atzei et al. (2016) analyzed security vulnerabilities of Ethereum smart contracts and group them into the level where they are introduced such as Solidity, EVM bytecode, or blockchain. Solidity is the language for smart contracts, i.e., the source code is written in Solidity, and EVM is the virtual machine where contracts run. The research analyzes 12 types of vulnerabilities and that most of them have been already exploited. The 12 types of security vulnerabilities of Ethereum smart contracts as shown in the following table.

Table 2

Taxonomy of Vulnerabilities in Ethereum Smart Contracts

Number	Vulnerability	Cause	Level
1	Call to the unknown	Called function does not exit	Solidity
2	Gasless send	Callee's fallback function is invoked	
3	Exception disorders	Irregularity in exception handling	
4	Type casts	No expectation is thrown, if type-mismatched	
5	Reentrancy	A non-recursive function re-enter before termination	
6	Keeping secrets	Contracts private fields, secrecy is not guaranteed	
7	Immutable bugs	Defective contracts cannot be patched or recovered	EVM
8	Ether lost in transfer	Ether sent to orphan address is lost	
9	Stack size limit	The number of values exceeds 1024 in the stack	
10	Unpredictable state	Actual state of the contract is changed before invoking	Blockchain
11	Generating randomness	Malicious miner biases the outcome of random number	
12	Time constraints	Timestamp of the block is changed by malicious miner	

Vulnerabilities in the smart contract based Blockchain.

Vulnerabilities in solidity. *Call to the unknown.* It refers to use of some functions in solidity to invoke functions and to transfer ether may result in the execution of an unknown, possible malicious fallback function. That is, i the intended function to be called, does not exist, then the fallback function is executed, and malicious users can exploit this vulnerability by calling their own fallback function (Atzei et al., 2016).

Gasless send. The function send is used to transfer Ether, and it is possible to encounter out of gas expectation. If the callee of a send function is a contract with relatively expensive fallback function than the caller gas limit–which could be up to 2300 units, for sending Ether to an address and if the gas is insufficient, then an out-of-gas exception will be thrown.

*Exception disorders**.*** There are several situations in which exception may be raised. However, Solidity is not uniform in a way it handles exceptions (Atzei et al., 2016). The irregularity in how exceptions are handled may impact the security of the contract. This is usually happened due to unchecked send errors or called contracts that throw exceptions, and the effect is the calling transaction is entirely reverted, and all gas is lost (Cook, Latham, & Lee, n.d.).

Type casts. The Solidity compiler can detect some types' errors. But in some cases, the compiler does not recognize mismatched type error. Mainly, if the arguments to a direct call from one contract to a function of another contract are incorrectly typed, or the address of the contract is invalid, either nothing will be executed, or the wrong code will be executed (Cook et al., n.d.). In all cases, no exception is thrown, and the caller is unaware of the error.

Reentrancy. Due to the atomicity and sequentially execution of transactions, some programmers believe that when a non-recursive function is invoked, it cannot be re-entered before its termination. But, it is not always the case, and a fallback mechanism may allow an attacker to re-enter the caller function, and this result of repeated execution of the function and consume all gas (Atzei et al., 2016).

Keeping secrets. Some fields in contracts are public, which are readable by everyone, and some held private which should not be directly readable by other users/contracts. However, contract fields declared private, are still not guaranteed to remain secret.

Vulnerabilities in EVM.

Immutable bugs. Once the smart contracts are added to the blockchain, due to the immutable property of blockchain, it cannot be altered. The programmers consider that the contract’s runtime behavior should be as they are expecting. However, if they found the contract is defective, then there is no direct way to patch it. The immutable bugs have been exploited in various attacks to steal Ether or to make it unredeemable by any user (Atzei et al., 2016). Sometimes developers need to plan a hard-fork to resolve such issue as the Ethereum team did after the “DOA attack.”

Ether lost in the transfer. If Ether is sent to an orphan address, it is lost forever. The orphan address does not belong to any user or contract, and there is no way to detect if the address is orphan (Atzei et al., 2016).

Stack size limit. Each time a contract calls another contract or even itself, a transaction’s call stack grows, and once the stack is reached to 1,024 frames, an exception is thrown. An attacker was able to exploit this vulnerability by generating an almost-full call stack (via a sequence of nested calls), and then he invokes the victim’s function, which will fail upon a further invocation. By October 18th, 2016 Ethereum team resolves this issue by changing some gas rules and certain instruction costs (Atzei et al., 2016).

Vulnerabilities in Blockchain.

Unpredictable state. Fields and balances determine the state of a smart contract. When a user sends a transaction to invoke some contracts, he cannot be sure that the transaction will be run in the same state the contract was at the time of sending that transaction. This may happen, because at the same time other transaction may change the state of same contracts. This vulnerability is said to be available in up to 15.8 percent in all smart contracts, and this bug is also called Transaction-Ordering Dependence (TOD) (Rivlin, 2016).

*Generating randomness***.** There are two types of execution of EVM bytecode, deterministic and non-deterministic (Atzei et al., 2016). Deterministic, where all miners’ executing a transaction will have the same result, and non-deterministic is to stimulate non-deterministic choices such as contracts for lotteries, games, etc. In non-deterministic execution of EVM bytecode pseudo-random numbers are generated, where the initialization seed is chosen uniquely by all miners. A malicious miner could exploit this vulnerability by arranging their block to influence the outcome of this random number generation.

Time constraints/timestamp dependency. Most of the applications use time constraints to determine which actions are permitted or mandatory at the current state of contract. The time constraints are implemented by using block timestamp, and some Dapps (Distributed Applications) use this timestamp to generate random numbers. However, the clock in Ethereum is set by local clocks of its miners. Thus, such Dapps can be affected with slight adjustments to miner's reported times (Cook et al., n.d.).

Attacks based on exiting vulnerabilities. The above 12 mentioned vulnerabilities led to six types of attacks (Atzei et al., 2016). (1) The DAO attack–DOA was a contract implementing a crowd-funding platform and this attacked happened by exploiting the "call to unknown" vulnerability of the Solidity. (2) King of the Ether Throne attack–the "King of the Ether Throne" is a game where players compete for getting the title of "King of the Ether," and this attack was happened by exploiting the "gasless send" and "exception disorders" vulnerabilities in the smart contract source code. (3) Multi-player games attack- Consider an online game between two players and suppose one of the players carry on an attack by exploiting the "keeping secret" or disclosing private field in the smart contract which can help him to reveal the opposing player state, and by this, the attacker can win the game. Game industry uses Ethereum smart contracts. (4) Rubixi attack–Rubixi is a smart contract to implement a Ponzi scheme, which is a fraudulent high-return investment program where participants gain money from the investments made by newcomers. An adversary or contract owner can collect fees, and steal Ether, and later exploit the “immutable bugs” vulnerability of contract source code. (5) GovernMental–it is another flawed Ponzi scheme, in which participant must send a certain amount of Ether to the contract to join. If no one joins the scheme for 12 hours, the last participants get all the Ether in the contract. The attack on GovernMetal could happen variously. For instance, by exploiting the “exception disorder,” “stack size limit,” and the main attack is called “Mallory attack,” in which the attacker starts invoking herself recursively, and, making the stack grow and eventually, stack overflow happened. It could also be happened by exploiting the “time constraints” vulnerability in the blockchain. (6) Dynamic libraries–This attack could happen by exploiting the “unpredictable state” of the blockchain. For example, consider a contract which can dynamically update one of its component (library), and the user does not know what the next state of the contract is. This allows an attacker to change these components with malicious ones. The attack scenario for each of the above attacks are explained in detail in the paper (Atzei et al., 2016).

Eclipse attacks on Ethereum network and client. The eclipse attacks on the Ethereum is discussed by the (Wüst & Gervais, 2016) by presenting three vulnerabilities in the Ethereum blockchain and client. First, by exploiting the block propagation design of Ethereum, an adversary can partition the P2P network without monopolizing the connections of the victim. Second, the paper presents an exploit to force a node to accept a longer chain with lower total difficulty than the main chain, and finally, identify a bug in the Ethereum difficulty calculation.

Low-source eclipse attacks on neighbor discover process. The paper by Marcus, Heilman, and Goldberg (n.d.) presented low-source eclipse attacks on Ethereum nodes that exploit the P2P network used for neighbor discovery. The attack in this scenario can be launched using only two hosts, each with a single IP address. This attack can control all the victim's incoming and outgoing connections, thus isolating from rest of the peers. This research also provides countermeasures to harden the eclipse attack on Ethereum, and the countermeasures have been consolidated in the Ethereum geth 1.8 client released on February 14, 2018.

Low-level attacks. Some attacks exploit vulnerabilities at the EVM specification level, combined with the security flaws in the Ethereum client (Atzei et al., 2016). For instance, some of the IO-heavy operations gas values are set too low, and hence these could be used by an adversary to launch a denial of service attack on Ethereum. The Rivlin study (2016) reported the denial of service attack on Ethereum due to the underpriced operation called "SUICIDE". After few other incidents due to underpriced operations the issue is fixed for seven IO-heavy operations in the EIP (Ethereum Improvement Protocol) 150 (vbuterin, n.d.).

Countermeasures solutions. The paper Luu, Chu, Olickel, Saxena, and Hobor (2016) proposed a symbolic execution tool called Oyente to find potential security bugs. This research found that among 19366 existing Ethereum contracts, Oyente flags 8,833 them vulnerable, including the DAO bug which led to a 60 million USD loss in June 2016. The also introduced four types of bugs namely. (1) Transaction-ordering dependence which is similar to unpredictable state vulnerability, (2) Timestamp dependence (3) Mishandled exceptions, (4) Reentrancy vulnerability.

The paper by Chen, Li, Luo, and Zhang (2017) identified 7 gas costly patterns and group them to 2 categories; useless code related patterns and loop related patterns. They found that under-optimized smart contracts cost more gas than necessary, and therefore users are overcharged. They developed a tool called GASPER, which automatically locating gas costly patterns by analyzing smart contracts' bytecodes. Their preliminary results on discovering the three representative patterns from 4,240 real smart contracts show that 93.5%, 90.1%, and 80% contracts suffer from these 3 patterns respectively. The three representative patterns are:

Detection of dead code. Some lines of code in a contract are never executed but they are still part of contract and deployed on blockchain. Since the consumption of gas is due to the size of bytecode, and hence the dead code wastes money. GASPER detects dead code through three steps; logs the addresses, collects the addresses, and finally reports all the blocks that are found in the CFG (Chen et al., 2017). GASPER constructs Control Flow Graph (CFG) for scanning purposes.

Detection of opaque predicates. Opaque predicates are statements in the contracts which results are known to be either true or false without execution. Since the results of opaque predicates are already known, and the results do not affect any other statement or contracts, so it should be removed for saving gas. In order to detect opaque predicates, GASPER executes the smart contracts symbolically, and records the executed branch either true or false when a conditional jump encountered. Then, the conditional jump with one never-executed branch is considered as an opaque predicate (Chen et al., 2017).

Detection of expensive operations in a loop. The expensive operations in a loop refers to execution of statement that may executed multiple times in one invocation. Moving them out of loop can save gas. GASPER detects this pattern through two steps. First, it looks for loops in the bytecode, and second, it searches the loop bodies for expensive operations. GASPER first searches for back edges in the CFG, which shows the existence of loops, and then identifies the entry bock and exit block for each loop. A block is considered to be in a loop if it is closer to the exit block than to the entry block (Chen et al., 2017).

The paper by Cook et al. (n.d.) also propose a prototype for a tool called DappGuard, which is intended to be used for live monitoring and protection for Solidity smart contracts. This tool considers all the 12 types of vulnerability discussed in the above table.

Data Analyzed: Security Threats Classification Taxonomies

Taxononmy#1. Taxonomy based on Blockchain abstract layers. This taxonomy leveraged some of the work introduced by Dinh et al. (2017) and Croman et al. (2016). The Dinh et al. (2017) design of a framework for private blockchain systems and considered four common layers of abstraction. By comparing the over 200 Bitcoin variants, Ethereum, and other permissioned blockchains, and meaningfully compare them, the study identifies four common abstraction layers in all of these systems. The consensus layer contains protocols through which a block is considered appended to the blockchain. The data layer contains the structure, content, and operations on the blockchain data. The execution layer includes details of the runtime environment that support blockchain operations. The application layer contains classes of the blockchain applications or blockchain platforms. In related work (Croman et al., 2016), blockchain is divided into several planes: network, consensus, storage, view, and side plane. This study did not consider the execution of smart contracts.

This study evaluates both studies (Croman et al., 2016; Dinh et al., 2017) and considered the five common layers of abstraction and included the network layer besides the four other layers, shown in the Figure 5. The network layer manages various protocols that are used to establish the communication channel between the nodes in the blockchain network. The common protocols included in the network layer are peer to peer (P2P) protocol- for establishing communication between the nodes connected in the decentralized environment, transmission protocol–for passing the block updates between the peers or nodes, and verification protocol responsible for verifying the transactions by each of the participant nodes or peers.

The taxonomy formed based on the abstract layers mainly considered the network, consensus, data model, and execution layers. Since the application layer includes the business model of various applications of the blockchain and could be different for each of the platform, hence we excluded it from examining. In taking into account, the four abstract layers of the blockchain, the taxonomy is formed and classified the security threats based on the nature of the vulnerabilities for each threat. The nature of each vulnerability and the corresponding security threats and attacks are discussed in detailed in next chapter, and the taxonomy based on the abstract layers are presented in Table 3.

Table 3

Taxonomy of Security Threats Affecting Blockchain Abstract Layers

Security Threats	**Attack Vectors**	**Affected Abstract Layers**
Double-Spending Threats	Race Attack	Consensus
	Finney Attack	Consensus
	Vector76 Attack	Consensus
	Alternative History Attack	Consensus
	51% Attack	Network, Consensus, Data Model
Mining/Pool Threats	Selfish Mining/Block-discard Attack	Network, Consensus
	Block-Withholding Attack (BWH)	Network, Consensus
	Fork-After-withhold Attack (FAW)	Network, Consensus
	Bribery Attack	Network, Consensus
	Pool Hopping Attack	Network, Consensus
Wallet Threats	Vulnerable signature	Data Model
	Lack of control in address creation	Data Model
	Collison & Pre-Image Attack	Data Model
	Flawed key generation	Data Model
	Bugs & Malware	Data Model
Network Threats	DDoS Attack	Network, Consensus, *External resource*
	Transaction Malleability Attack	Consensus, Data Model
	Timejacking Attack	Network, Consensus, Data Model
	Partition Routing Attack	Network, Consensus, Data Model
	Delay Routing Attack	Network, Consensus, Data Model
	Sybil Attack	Network
	Eclipse Attack	Network
	Refund Attack	Application - Bitcoin
	Balance Attack	Network, Consensus
	Punitive and Feather forking Attack	Network, Consensus, Data Model
Smart Contracts Threats	Vulnerabilities in contracts source code	Execution
	Vulnerabilities in EVM Bytecode	Execution
	Vulnerabilities in Blockchain	Network, Consensus
	Eclipse Attack on contract blockchain	Network, Consensus
	Low-level attacks	Consensus, Data Model

Blockchain architecture abstraction layers. Blockchain architecture (Figure 5) consists of a network layer, a data layer, a consensus layer, an execution layer, and an application layer. Most of the existing blockchain platform includes these five abstract layers.

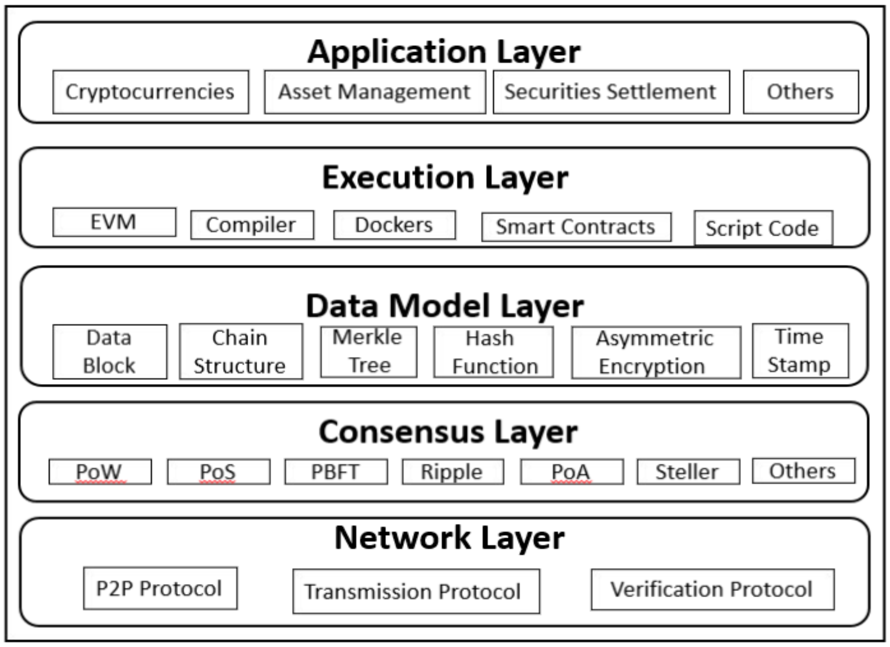

Figure 13. Blockchain abstract layers.

Network layer. Peer-to-Peer (P2P) is a decentralized communication model and is one way of distributing data in a network. In the P2P network model, each party has the same capabilities, and either party can initiate a communication session. Unlike the client/server model, in which the client makes a service request, and the server fulfills the request. The P2P network model allows each node to function both as a client and server.

In P2P network model as each peer has 100% (or as close to it as possible) of the data, and updates are shared around. Data replication, and sharing updates happen many times; once per machine, and each addition or change makes more network traffic, and due to which P2P is considered less efficient than the client-server model ("A-Gentle-Introduction-To-Bitcoin-

WEB.pdf," n.d.). However, each peer is more independent and can continue operating to some extent if it loses connectivity to the majority of other peers. In other words, P2P provides a solution to the single point of failure problem, which makes P2P network model more robust, as no central server that controls everything, so shutting down the P2P network is difficult. Another problem with P2P models is, as there is no central server to manage the updates or modification of data, so if each peer is updating at different speeds and have slightly different states, it will be difficult to determine the real state of the data. To tackle this problem, even if the peers in the P2P network are untrusted like in Bitcoin blockchain, some consensus algorithms like PoW or PoS are used to ensure the changes to the blockchain and mitigate the risk of corrupting the ledger by bad peers.

In blockchain, the P2P network layer is responsible for inter-node communications which include discovery and data transfer (usually transactions and block propagation) (Pay, 2017). Typically, a sample of this layer is the DEVp2p library used by the Ethereum blockchain (Costa, 2014/2018). DEVp2p is a wire protocol used by Ethereum to facilitate the P2P communications between the nodes in the Ethereum Blockchain network. DEVp2p nodes communicate by sending messages using RLPx (Recursive Length Prefix), an encrypted and authenticated transport protocol. RLPx is a cryptographic P2P network and protocol suite which provides a general-purpose transport and interface for applications to communicate through a P2P network.

Peers in the network are free to advertise and accept connections on any TCP port they wish, however, a default port on which connection may be listened and made is 30303 (Costa, 2014/2018). Though TCP provides a connection-oriented channel, DEVp2p nodes communicate regarding packets and RLPx facilities to send and receive packets. DEVp2p nodes find peers in the network through the RLPx discovery protocol DHT. Peer connections can also be initiated by providing the endpoint of a peer to a client-specific RPC API. Further details about RLPx is presented here (Mandeleil, 2015/2018).

Bitcoin is structured as a peer-to-peer network based on unencrypted persistent TCP connections as its foundational communication structure. The collection of nodes running the bitcoin P2P protocol is called bitcoin network. Each node in the bitcoin network maintain a list of IP address of potentials peers, and the list is bootstrapped via a DNS server, and additional addresses are exchanged between peers. Each node in the network maintain a minimum of eight unencrypted TCP connections in the overlay, and each node establish new connections if the current number of connections is lower than eight (Conti et al., 2017). The number of connection can be exceeded from 8 till 125 if incoming connections are accepted by a Bitcoin peer. By default, each peer listens on port 8333 for inbound connections, and to establish a new connection, peers perform an application layer handshake, using the *version* and *verack* messages. During the handshake, the version and verack messages are exchange, to ensure compatibility between peers. 1) The “version” message is created by a node which want to create an outgoing connection, and it will immediately advertise its version. The remote node will respond with its version, and no further communication is possible until both the peers have exchanged their version (“Protocol documentation–Bitcoin Wiki,” n.d.). 2) The “verack” message is sent in reply to version (“Protocol documentation–Bitcoin Wiki,” n.d.). This message consists of only a message header with the command string “verack”. The “verack” packet shall be sent if the version packet was accepted. The verack message. The exchange of version and verack messages are called “Version Handshake” between the peers in the Bitcoin network. For example, when a local peer (L) wants to connect to a remote peer (R), the remote peer will not send any data until it receives a version message back from the local peer. The messages include a timestamp for time synchronization, IP addresses, and the protocol version. Following is the protocol handshaking process between bitcoin's nodes in the network ("Protocol documentation– Bitcoin Wiki," n.d.).

- L -> R: Send version message with the local peer's version.
- R -> L: Send version message back.
- R -> L: if both peers' version is compatible, R send verack message.
- R: Sets version to the minimum of the 2 versions.
- L->R: Send verack message after receiving version message from R.
- L: Sets version to the minimum of the 2 versions.

Each node randomly selects its peers, and it decides a new set of peers after a fixed amount of time. This is done to minimize the possibility and effects of netsplit attack, in which an attacker creates an inconsistent view of the network at the attacked node. To detect when peers have left the network, Bitcoin uses a soft-state approach. If 30 minutes have been passed since messages were last exchanged between neighbors, peers will transmit a "hello" message to keep the connection alive.

Miners continually listen to new block announcements which are sent via INV messages, containing the hash of the mined block. If miners discover that it does not hold a newly announced block, it transmits a GETDATA message to one of its neighbors, and the neighbor responds by sending the requested information in a BLOCK message. In case the requested block does not arrive within 20 minutes, the miner triggers the disconnection of that particular neighbor and requests the same information from another neighbor. The propagation of transactions occurs in a sequence given as INV, GETDATA, and TX messages, in which nodes announces, request, and share transactions that have not yet been included in the blockchain (Conti et al., 2017).

Data model layer. The data layer contains the structure, content, and the operation of the blockchain data. According to Mastering Bitcoin, transactions are data structures that encode the transfer of value between participants in the bitcoin network (Antonopoulos, 2015). A block contains a list of transactions, and a list of smart contracts executed as well as their latest states. In the data layer, each block is identified by the cryptographic hash of its contents and linked to the previous block hash, and all blocks together create the chain of blocks or digital ledger. Transactions hashes in a block are group together in a Merkel tree. Each transaction is a public entry in the bitcoin blockchain. Transactions are the essential part of the blockchain network. Every components and service in the blockchain network is designed to ensure that transactions can be created, propagated on the network, validated, and finally added to the blockchain. Typically, data models have a block, a block header, and transactions. In Bitcoin, transactions are system states representing digital coins in the network. Bitcoin introduces the Input-Output transactions data model that is also used by other protocols such as MultiChain and Chain core. Ethereum, on the other hand, uses the state replication model, where each new block's state is the outcome of the transactions that were included in the block. For instance, in Bitcoin transferring money between sender and receiver involves searching for transaction belonging to the sender, then marking some of them as spent, whereas it is easily done in Ethereum by updating two accounts in one transaction. An account in the Ethereum has a balance as its state and is updated upon receiving a transaction. A special type of account, called “smart contract,” contains executable code and private states, and when receiving a transaction, in addition to updating its balance, the contracts' code is invoked and performed the specified arguments in the transaction. The code can read the states of other non-contract accounts, and it can send new transactions during execution (Dinh et al., 2017). Ethereum has chosen not to use the Input-Output transaction model because it does not allow multi-stage contracts or scripts that cloud keep an internal state (Dinh et al., 2017).

Consensus layer. The role of consensus layer is to get all nodes in the blockchain system to agree on the blockchain contents. That is, if a node appends a block to the blockchain, the other nodes in the network must approve and also append the same copy of the block to their blockchain. The consensus layer contains protocols through which a block is considered verified and appended to the blockchain. Blockchain systems, use a spectrum of Byzantine fault-tolerant protocols. The consensus layer works jointly with data model layer, in creating, verifying and appending a block the blockchain. The data model layer concerns about the structure of the block and consensus layer concerns about which consensus mechanism to be used for creating the adding the block to the chain.

The Proof-of-Work consensus algorithm is the broadly used consensus mechanism used in the existing blockchain systems, particularly in the cryptocurrencies area. PoW was introduced by Bitcoin and assumes that each node in the Bitcoin system votes with his “computing power” by solving proof of work instances and creating the appropriate blocks. Bitcoin uses a hash-based PoW which requires finding a nonce value, such that when hashed with additional block parameters (e.g., a Merkle hash, the previous block hash), the value of the hash has to be smaller than the current target value. When such a nonce is found, the miner creates the block and forward it to the network layer to its peers. Other peers in the network which has P2P connections, can verify the PoW by computing the hash of the block and checking whether it satisfies the condition to be smaller than the current target value (Gervais et al., 2016).

Proof-of-Work selects at each round a random node which can append a block, where the node's total computing power determines the probability of being selected. This simple scheme works against the Sybil attack; an attack in an open, decentralized environment in which adversary requires multiple identities (Bissias et al., 2014). The major problem with PoW mechanism is that as nodes spend their CPU cycle solving puzzles and it consumes a lot of energy and computing power. In addition to this problem, sometimes two nodes may be selected to append the same block to the chain, and both blocks can be accepted, and this cause fork in the blockchain. To tackle the fork problem, additional rules are used by most of the PoW-based blockchain systems, for instance, only blocks on the longest chain are considered and accepted. Ethereum, like Bitcoin uses PoW variant called GHOST (Greedy Heaviest Observed Subtree) which accepts blocks in heavy branches (Buterin, 2014). To solve the energy and resources usages problem, an alternative to PoW called Proof-of-Stake (PoW) is used. Other famous consensus algorithms used in existing blockchains are Proof-of-Authority (PoA ("Proof-of-authority," 2018), Proof-of-Burn (PoB) ("Why Proof-of-Burn | Counterparty," n.d.), Ripple (Schwartz, Youngs, & Britto, n.d.), Practical Byzantine Fault Tolerance (PBFT) (Castro & Liskov, n.d.), Stellar (RES, n.d.) and others.

Execution Layer. The execution layer contains details of the runtime environment that support blockchain operations (Dinh et al., 2017). Each blockchain system uses its own programming and scripting languages which entail runtimes environment including compiler, decoder, virtual machine and others. Most of these runtime environment operations are maintained at the execution layer of the blockchain systems. A contract (or chaincode) is executed in a runtime environment, and there are two requirements of execution. (1) Execution must be fast because there are multiple contracts and transactions in one block and they must all be verified by the node. (2) Execution must be deterministic; ideally the same at all the nodes. Deterministic execution prevents unnecessary inconsistency in transaction input and output states which leads to blocks being aborted and aborting a block usually wastes computer resources.

Smart contracts are programs that serve to negotiate, facilitate or enforce the performance and state of an agreement (contract). The term of the contract is coded as a program in the programming language provided by the underlying blockchain platform. These smart contracts can complement or substitute legal contracts. Ethereum develops its own machine language (bytecode) and a virtual machine called Ethereum Virtual Machine (EVM) for executing the code. EVM is a runtime environment, which provides all the needed support and services for executing the smart contracts in Ethereum blockchain system. EVM is optimized for Ethereum specific operations. For instance, executing a code (running a transaction or contract) in Ethereum costs a certain amount of gas, and the total cost must be properly tracked and charged to the transaction’s sender. Gas is the internal pricing for running a transaction or contract in Ethereum blockchain system. The gas system is not very different from the use of KW (Kilo Watt) for measuring electricity. One difference from actual energy market is that the originator of the transaction sets the price of gas, to which the miner may accept or ignore. That is, miners are free to ignore transactions whose gas prices limit is too low, on the other hand, with Bitcoin miners' priorities transaction with the highest mining fees. The idea behind using gas is to limit infinite loops. For instance, 10Szabo (which is about 1/100,000 of an Ether), or 0.00001 Ether (Ethereum currency) or 1 gas can execute a line of code or some commands in the contract. So, if there is not enough Ether in the account to perform the transaction or execute the smart contract, then it is considered invalid. The idea is to stop denial of service attacks from infinite loops and to make an attacker pay for the resources they use, from bandwidth to CPU calculations through to storage [56].

Party (Dixon, n.d.) another Ethereum based blockchain system also uses EVM for executing the code. However, Hyperledger ("Hyperledger_Arch_WG_Paper_1_Consensus.pdf," n.d.) does not consider these semantic in its design, so it merely supports running of compiled machine codes inside Docker image.

Application layer. The application layer includes the use-cases of the blockchain application. Most of these applications of blockchain are due to key features of blockchain mainly two features. First, data in the blockchain is immutable and transparent to the participants, meaning that once a record is appended, it can never be changed and this feature achieved data integrity. Second, it is resilient to dishonest and malicious participants. The most extensive application of blockchain is still cryptocurrencies. According to Coinmarketcap site, currently, more than 1500 different cryptocurrencies projects exist in the market. Besides cryptocurrencies, there are other use-cases of the blockchain in the market such as asset management, supply change, identity management, security settlements, IoT, Cloud security, financial services, Healthcare and many more.

Taxonomy #2. Taxonomy based on Blockchain primary processes. This taxonomy is inspired from the work (Ellervee, 2017), which designed a comprehensive reference model for blockchain-based distributed ledger technology. The reference model was designed with intension to better understand the standardization and uniform processes involved in the blockchain. In order to designed a reference model the study (Ellervee, 2017) separated the blockchain based platforms into four groups: permissionless (public) blockchain, permissioned (private) blockchain, blockchains with smart contracts, and blockchain with transaction only as shown in the Table 4.

Table 4

Classification of Blockchain Platforms

	Permissionless	**Permissioned**	
Transaction only	Bitcoin	MultiChain	**Blockchain 1.0**
With Smart Contract	Ethereum	Chain Core	**Blockchain 2.0**

The four groups of blockchain platforms are characterized as follows:

- **Permissionless.** The permissionless blockchains are fully public blockchains, where anyone can read, write data (Ellervee, 2017).
- **Permissioned.** In permissioned blockchain, rules are pre-defined for different permissions on different users on the network. There can be different permissions for reading data, creating transactions, validating blocks, creating new blocks and others (Ellervee, 2017).
- **Blockchain with transaction only.** These blockchain platforms are built for transaction capabilities such as Bitcoin and MultiChain. They support transferring

value from one account to another (Ellervee, 2017). These types of blockchain are also named as Blockchain 1.0.

- **Blockchain with Smart Contracts.** These blockchain enable “smart contract” like capabilities and allow building business logic process mechanism into the chain (Ellervee, 2017). With the invent of smart contract concept, these blockchain platforms are considered as Blockchain 2.0.

The reference model summarized the four platforms provide similar four general process: *Network discovery*, *Transaction creation, and Mining* (Block generation and submission), and *block validation process*.

Process are realization of services that the actors (users) of the technology use. The typical services in the blockchain are creating transaction, mining or creating blocks, and validating blocks. The table 5 provides overview of the processes from different platforms.

Table 5

Blockchain Processes in Blockchain Platform

Platform	**Processes**
Bitcoin	Network discovery process, Transaction creation process, Mining process, Block verification process
MultiChain	Handshake process, Transaction creation process, Mining process
Ethereum	Network discovery process, Transaction creation process, Mining process, Block validation process
Chain Core	Network discovery process, Transaction process, Chain consensus process

- **Network discovery process.** Each platform has a network discovery process, which includes four main steps (Ellervee, 2017):

- Peer discovery: finding peers to connect to, either user already knows the IPs or acquires them
- Handshake: version check, establishing connection, providing ownership of private key
- Network discovery: finding neighboring peers and letting the network know that a new node has connected.
- Synchronization: downloading the latest block data from the network.

- **Transaction creation process.** In bitcoin to create a transaction user, need to enter value and the receivers address, and then the transaction is signed by the user and broadcasted to the network (to neighboring nodes). The neighboring nodes check the transaction for validity and if valid, then they will propagate it forward to other peers. In Ethereum the same process as Bitcoin is followed up, i.e., supports the regular value transactions. The input parameters for the transaction are similar to Bitcoin (i.e., amount and the address of the receiver). Besides input parameters Ethereum also support creating contracts and calling contract functions, which is an additional metadata added to a transaction (Ellervee, 2017).
- **Mining process or Block generation process.** Mining is based on the proof-of-work consensus mechanism. In mining process or block generation process a miner add collected unverified transaction and metadata, and calculate the computationally exhaustive proof-of-work for the block and build a new block. If he is the first one to solve the task and mine the block, he can submit the block to the network and receive the reward for the work (Ellervee, 2017; Nakamoto, n.d). In Ethereum, the mining

process also requires a state transaction process, since it keeps a state of the blockchain. In the state transaction process, transactions are validated and in case of contracts, code execution is also performed. Finally, when all the state transaction function, miner in Ethereum will provide the proof-of-work to the blockchain (Buterin, 2014; Ellervee, 2017). In MultiChain the proof-of-work is optional and mining is permissioned (Greenspan, n.d.).

- **Block validation process.** In public blockchain such as Bitcoin and Ethereum the validation process is performed by every node once the miner broadcast a new block (Buterin, 2014; Nakamoto, n.d.). Since in the public blockchains the miners are anonymous, there has to be a guarantee that the miner has truly produced a valid block. In Bitcoin, this is called a *consensus* if all the nodes validate the new blocks against the same rules. In Ethereum, the validation is similar to Bitcoin, but it also includes the state transaction process, which has to be performed by each before accepting the block (Ellervee, 2017).

This taxonomy classified the security threats based on the primary affected blockchain process and shown in Table 6:

Table 6

Taxonomy of Security Threats Affecting Blockchain Processes

Security Threats	**Attack Vectors**	**Primary Affected Blockchain Processes**
Double-Spending Threats	Race Attack	Transaction Creations, Mining
	Finney Attack	Transaction Creations, Mining, Block Validation
	Vector76 Attack	Transaction Creations, Mining, Block Validation
	Alternative History Attack	Transaction Creations, Mining, Block Validation
	51% Attack	Network discovery, Mining, Transaction Creations, Block Validation
Mining/Pool Threats	Selfish Mining/Block-discard Attack	Mining
	Block-Withholding Attack (BWH)	Mining
	Fork-After-withhold Attack (FAW)	Mining
	Bribery Attack	Transaction Creations, Mining, Block Validation
	Pool Hopping Attack	Mining
Wallet Threats	Vulnerable signature	Others -Private Key security threat
	Lack of control in address creation	Others -Private Key security threat
	Collison & Pre-Image Attack	Others -Private Key security threat
	Flawed key generation	Others -Private Key security threat
	Bugs & Malware	Others -Private Key security threat
Network Threats	DDoS Attack	Network discovery, Mining, Transaction Creations, Block Validation
	Transaction Malleability Attack	Transaction Creations, Mining
	Timejacking Attack	Transaction Creations, Mining
	Partition Routing Attack	Network discovery
	Delay Routing Attack	Network discovery
	Sybil Attack	Network discovery
	Eclipse Attack	Network discovery
	Refund Attack	Others-Payment protocol authentication & refund
	Balance Attack	Network discovery
	Punitive and Feather forking Attack	Mining
Smart Contracts Threats	Vulnerabilities in contracts source code	Transaction Creations - change of state
	Vulnerabilities in EVM Bytecode	Transaction Creations - change of state, Mining
	Vulnerabilities in Blockchain	Transaction Creations - change of state, Mining
	Eclipse Attack on contract's blockchain	Network discovery
	Low-level attacks	Transaction Creations - change of state, Mining

Taxonomy #3. Taxonomy based on Blockchain affected business users. This taxonomy is based on the affected target of security threats. It mainly focuses on the affected users by each of the security threats. In design of this taxonomy the taxonomy leveraged the (Ellervee, 2017) work with some modification. Blockchain a decentralized network of individual nodes, and each node has different purposes and different roles. The study (Ellervee, 2017) shows an overview of the actors who are present in the analyzed blockchain platforms in Table 7.

Table 7

Overview of Actors in Blockchain Platforms

Platform	Actors
Bitcoin	Client (sender or receiver of Bitcoins), Miners
MultiChain	Client, Miners
Ethereum	Externally Owned Account, Contract Account, Miner
Chain Core	Client (Issuer/Spender of assets), Blockchain operator (Generator/Signer)

Generally, each of the platform involves a Client and Miner. The client interacts with the blockchain and exchanges or adds value by creating and broadcasting transactions. In Ethereum, an Externally Owned Account (EOA) can be understood as physical actor, and Contract Account (CA) can be understood as a system user which acts upon a request by an EOA, or by another CA. Miners deal with validating transactions and building new blocks.

With keeping in mind the (Ellervee, 2017) study, this study defined and considered the actors as follows:

- **User.** The human act who interacts with the blockchain by creating transaction can be called a “User”.

- **Miner.** Human or system actor responsible for verification and validation of transactions, building new blocks, signing new blocks and publishing new blocks to the blockchain. This actor supports trust between the parties involved.
- **Mining Pool.** When miners share their computer power to solve together the proof-of-work, miners create a pool. Since the difficulty of some blockchain such as Bitcoin and Ethereum is very high, and it is difficult for a solo miner to solve the problem, so they share their computer resources and collectively solve the problem. The rewards for the solving difficulty or mining block is then shared among all the participant miners in the pool.
- **Merchant.** Any business which accept digital currency as payment systems.
- **Blockchain Network.** It includes the nodes, which created the decentralized network of the blockchain and maintain the data of the ledger.
- **Exchange.** The cryptocurrency or digital currency exchanges are the business that provides services to customers to trade digital currencies for other assets, such as conventional fiat money, or different digital currencies.

The affected business users or actors based on each security threats or attacks are shown in Table 8.

Table 8

Taxonomy of Security Threats Affecting Business Users

Security Threats	**Attack Vectors**	**Primary Targets**	**Blockchain Versions**
Double-Spending Threats	Race Attack	Sellers or Merchants	Blockchain1.0, 2.0
	Finney Attack	Sellers or Merchants	Blockchain 1.0, 2.0
	Vector76 Attack	Exchange	Blockchain 1.0, 2.0
	Alternative History Attack	Sellers or Merchants	Blockchain 1.0, 2.0
	51% Attack	Network, Sellers, Merchants, Exchange, Miner	Blockchain 1.0, 2.0
Mining/Pool Threats	Selfish Mining/Block-discard Attack	Miners or Mining pools	Blockchain 1.0, 2.0
	Block-Withholding Attack (BWH)	Miners or Mining pools	Blockchain 1.0, 2.0
	Fork-After-withhold Attack (FAW)	Miners or Mining pools	Blockchain 1.0, 2.0
	Bribery Attack	Miners or Mining pools	Blockchain 1.0, 2.0
	Pool Hopping Attack	Miners or Mining pools	Blockchain 1.0, 2.0
Wallet Threats	Vulnerable signature	Users and merchant's wallets	Blockchain 1.0, 2.0
	Lack of control in address creation	Sellers or Merchants	Blockchain 1.0, 2.0
	Collison & Pre-Image Attack	Users and Merchants wallets	Blockchain 1.0, 2.0
	Flawed key generation	Users and Merchants wallets	Blockchain 1.0, 2.0
	Bugs & Malware	Users and Merchants wallets	Blockchain 1.0, 2.0
Network Threats	DDoS Attack	Network, miners, pools, users, exchange, businesses	Blockchain 1.0, 2.0
	Transaction Malleability Attack	Exchanges	Blockchain 1.0, 2.0
	Timejacking Attack	Miners	Blockchain 1.0, 2.0
	Partition Routing Attack	Miners, users, pools	Blockchain 1.0, 2.0
	Delay Routing Attack	Miners, users, pools	Blockchain 1.0, 2.0
	Sybil Attack	Network, miners, pools, users	Blockchain 1.0, 2.0
	Eclipse Attack	Miners, users	Blockchain 1.0, 2.0
	Refund Attack	Exchanges, users, sellers	Blockchain 1.0, 2.0
	Balance Attack	Users, miners	Blockchain 1.0, 2.0
	Punitive and Feather forking Attack	Users	Blockchain 1.0, 2.0
Smart Contracts Threats	Vulnerabilities in contracts code	Contracts owner, businesses	Blockchain 2.0
	Vulnerabilities in EVM Bytecode	Contracts owner, businesses	Blockchain 2.0
	Vulnerabilities in Blockchain	Users, contracts owner, businesses	Blockchain 2.0
	Eclipse Attack on Smart contract	Miners, users	Blockchain 2.0
	Low-level attacks	Network, miners, pools, users, exchange, businesses	Blockchain 2.0

Summary

In this chapter, the author explained the data collected for the systematic survey about the security aspect of the blockchain. It also included the technical challenges and advancements. Analyzed the security threats and attack vectors for each security threats, and how adversaries can use these existing vulnerabilities and launch various types of the attacks. Each of the attack is discussed in detail by examining the popular blockchain platforms and provided the attack scenario and possible countermeasures. Based on data analyzed the author formed taxonomies and classified the security threats and attacks in term of affected abstract layers, processes and business users involved in the blockchain platforms.

Chapter V: Results, Conclusion, and Recommendations

This chapter concludes this study. It also suggests the lessons learned, future research directions and conclusion of the work.

Results

One of the major contribution of blockchain is the degree of transparency and decentralization that it provides along with the adequate level of security and privacy, which previously considered impossible. However, Blockchain is still some evolving technologies and besides many interesting security and privacy features, there are still huge security and privacy concerns available in the Blockchain ecosystem. Based on the comprehensive survey on the security aspect of the Blockchain, there are security threats existing due to the available vulnerabilities in the Blockchain. Besides, the security threats that lead to double-spending attacks, there are security threats to Blockchain network, blockchain miner or mining pools, private key security threats and smart contracts. The study also analyzed based on the nature of each vulnerabilities discussed in this study, and classified the security threats to different fields of the blockchain.

Future Research Directions

Consensus mechanism. As most of the popular public Blockchain such as Bitcoin, Ethereum and others used proof-of-work (PoW) based consensus algorithm to protect the user’s actions. Proof-of-work also provide a practical solution for the Byzantine Generals problem and achieve distributed consensus. However, PoW based blockchain expose itself to a number of security threats. The main threat is double-spending which is almost always possible in Bitcoin. Another major disadvantage of PoW based blockchain is the waste of computer resources. To solve this problem some blockchain such as Ethereum plan to use the hybrid consensus mechanism of PoW and PoS. But as Bitcoin is still dominate blockchain and PoW always requires huge wastage of resources both electricity and computer resources. Thus, providing more robust, secure and easily scalability consensus algorithms are a possible research direction.

- One the major contribution of blockchain like Bitcoin is the degree of transparency and decentralization, which was previously considered impossible.

Mining pool protocols. The original concept of mining, which could be based on Proof of Work, Proof of Stack, Proof of Burns or some other scheme, not only secures the blockchain but it also provides distributed consensus. Without mining schemes, the fake identities would be able to easily disturb the consensus process and ruined the normal functions of systems by a Sybil attack. However, the rapidly increasing mining pools threatens the decentralized of some of the blockchain like Bitcoin.

Game theory and stability. Since most the blockchain functions depends on the miner and miners can behave selfishly by holding on to their blocks and releasing it whenever they want. This kind of selfish behavior may pose a game theoretic problem between the selfish miners and the honest miners in the network. The selfish miners or attacker may try to contribute to an increase of their chain length compared to honest chain miners in the network, and eventually launch different types of attacks including double-spending. Thus, achieving equilibrium to bring stability in the network is a possible research direction.

Cryptographic and keying techniques. The current Simplified Payment Verification (SPV) protocol which is a lightweight protocol used for the verification of the transactions send from the user, is often vulnerable to attacks like Sybil or double-spending. A more robust verification protocol is required. For the key manipulations and calculations, a distributed approach is always preferred more than the centralized one. Thus, in this direction, innovative means of key computation and storage of the Blockchain is a possible research direction. Also, was discussed Bitcoin highly rely on ECDSA algorithm and it has poor randomness property and hash functions like SHA-256, creates another research direction.

Improving blockchain protocol. Despite blockchain potential, it still faces significant concerns in term of privacy and scalability. The immutable nature of the blockchain makes it impractical for many other applications. Recently, the research called “Redactable Blockchain” (Ateniese, Magri, Venturi, & Andrade, 2016) present modification in blockchain techniques that allows operations such as re-writing one or more blocks, compressing any number of blocks into a smaller one.

Fastness. Bitcoin PoW is designed to validate a new block on average within 10 minutes, and it is recommended to wait for six confirmations before accepting a transaction, which makes it impractical for many real-word application such as a point of sale payments. Fasting mining with the same robustness is a future requirement.

Blockchain data. The blockchain will produce a lot of data, including block information, transaction data, contract bytecode, etc. However, not all of the data in the blockchain is valid. For instance, a smart contract can delete its code by SUICIDE or SELFDESTRUCT, but the address of the contract will not be erased. Moreover, there are a lot of smart contracts containing no code or totally the same code (redundant), or some smart contracts are never being executed after its deployment. Thus, an efficient data cleanup and detection mechanism is desired to improve the execution efficiency of the blockchain systems.

No practical defense for attacks. Throughout our comprehensive and systematic survey, we provided the reported defense mechanism or countermeasures of each of the attack discussed in the study. However, some of the attack such as Fork after Withholding (FAW) (Kwon et al., 2017) attack, and Punitive and Feather forking attacks (Narayanan, 2017) remains open challenges.

Conclusion

This study focuses on the security issues of the blockchain technology. By studying the different fields of the blockchain such as consensus mechanisms, blockchain network, mining process, data storage and key management, and smart contracts functionality, the study reviewed all the existing vulnerabilities in this area. Additionally, based on the understanding from the survey the study classified the security threats in two various domains.

References

A-Gentle-Introduction-To-Bitcoin-WEB.pdf. (n.d.). Retrieved from https://bravenewcoin.com/assets/Reference-Papers/A-Gentle-Introduction/A-Gentle-Introduction-To-Bitcoin-WEB.pdf.

Andrychowicz, M., Dziembowski, S., Malinowski, D., & Mazurek, Ł. (2015). On the malleability of bitcoin transactions. In M. Brenner, N. Christin, B. Johnson, & K. Rohloff (Eds.), *Financial cryptography and data security* (Vol. 8976, pp. 1-18). Berlin, Heidelberg: Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-662-48051-9_1

Antonopoulos, A. (2015). *Mastering bitcoin*. Retrieved November 15, 2017, from http://chimera.labs.oreilly.com/books/1234000001802/ch07.html.

Apostolaki, M., Zohar, A., & Vanbever, L. (2017). *Hijacking bitcoin: Routing attacks on cryptocurrencies* (pp. 375–392). IEEE. https://doi.org/10.1109/SP.2017.29

Araoz, M. (2016, July 29). *The hitchhiker's guide to smart contracts in Ethereum*. Retrieved March 30, 2018, from https://blog.zeppelin.solutions/the-hitchhikers-guide-to-smart-contracts-in-ethereum-848f08001f05.

Asia, O. (2018, January 29). *Tracing back stolen cryptocurrency (XEM) from Japan's Coincheck*. Retrieved April 7, 2018, from https://www.forbes.com/sites/outofasia/2018/01/29/tracing-back-stolen-cryptocurrency-xem-from-japans-coincheck/.

Ateniese, G., Faonio, A., Magri, B., & de Medeiros, B. (2014). Certified bitcoins. In I. Boureanu, P. Owesarski, & S. Vaudenay (Eds.), *Applied cryptography and network security* (Vol. 8479, pp. 80-96). Cham: Springer International Publishing. https://doi.org/10.1007/978-3-319-07536-5_6

Ateniese, G., Magri, B., Venturi, D., & Andrade, E. (2016). *Redactable blockchain–or–rewriting history in bitcoin and friends* (No. 757). Retrieved from http://eprint.iacr.org/2016/757.

Atzei, N., Bartoletti, M., & Cimoli, T. (2016). *A survey of attacks on Ethereum smart contracts* (No. 1007). Retrieved from http://eprint.iacr.org/2016/1007.

Bag, S., Ruj, S., & Sakurai, K. (2017). Bitcoin block withholding attack: Analysis and mitigation. *IEEE Transactions on Information Forensics and Security, 12*(8), 1967-1978. https://doi.org/10.1109/TIFS.2016.2623588

Bahack, L. (2013). Theoretical bitcoin attacks with less than half of the computational power (draft). *ArXiv:1312.7013 [Cs]*. Retrieved from http://arxiv.org/abs/1312.7013.

Bamert, T., Decker, C., Elsen, L., Wattenhofer, R., & Welten, S. (2013). *Have a snack, pay with bitcoins* (pp. 1-5). IEEE. https://doi.org/10.1109/P2P.2013.6688717

Banafa, A. (2017, August 17). *How to secure the internet of things (IoT) with Blockchain*. Retrieved November 13, 2017, from https://datafloq.com/read/securing-internet-of-things-iot-with-blockchain/2228.

Beecroft, N. (2015). *Bitcon|emerging risk report–2015*. Retrieved from https://www.lloyds.com/~/media/files/news-and-insight/risk.../2015/bitcoin--final.pdf.

Bentov, I., Lee, C., Mizrahi, A., & Rosenfeld, M. (2014). *Proof of activity: Extending Bitcoin's proof of work via proof of stake* (No. 452). Retrieved from http://eprint.iacr.org/2014/452.

BGP hijacking. (2018, January 16). In *Wikipedia*. Retrieved from https://en.wikipedia.org/w/index.php?title=BGP_hijacking&oldid=820773357.

Bissias, G., Ozisik, A. P., Levine, B. N., & Liberatore, M. (2014). *Sybil-resistant mixing for bitcoin* (pp. 149-158). ACM Press. https://doi.org/10.1145/2665943.2665955

Bitcoin block reward halving countdown. (n.d.). Retrieved March 17, 2018, from http://www. bitcoinblockhalf.com/.

Bitcoin charts & graphs–Blockchain. (n.d.). Retrieved March 23, 2018, from https:// blockchain.info/charts.

Bitcoin energy consumption index. (n.d.). Retrieved April 6, 2018, from https://digiconomist. net/bitcoin-energy-consumption.

Bitcoin, Litecoin, Namecoin, Dogecoin, Peercoin, Ethereum stats. (n.d.). Retrieved April 6, 2018, from https://bitinfocharts.com/.

Bitcoincore development–How much of BIP 62 ("Dealing with malleability") has been implemented?–Bitcoin stack exchange. (n.d.). Retrieved March 26, 2018, from https://bitcoin.stackexchange.com/questions/35904/how-much-of-bip-62-dealing-with-malleability-has-been-implemented.

Blockchains & distributed ledger technologies. (n.d.). Retrieved February 27, 2018, from https://blockchainhub.net/blockchains-and-distributed-ledger-technologies-in-general/.

Bonneau, J. (n.d.). Why buy when you can rent? *Bribery attacks on Bitcoin style consensus* (p. 8). Stanford University and Electronic Frontier Foundation.

Bos, J. W., Halderman, J. A., Heninger, N., Moore, J., Naehrig, M., & Wustrow, E. (2013). *Elliptic curve cryptography in practice* (No. 734). Retrieved from http://eprint.iacr.org/2013/734.

Boverman, A. (2011, May 25). *Culubas: Timejacking & Bitcoin*. Retrieved March 26, 2018, from http://culubas.blogspot.com/2011/05/timejacking-bitcoin_802.html.

Bruce, J. D. (n.d.). The Mini-Blockchain Scheme, p. 13. Retrieved from https://www.weusecoins.com/assets/pdf/library/The%20Mini-Blockchain%20Scheme.pdf.

Buldas, A., Kroonmaa, A., & Laanoja, R. (2013). *Keyless signatures' infrastructure: How to build global distributed hash-trees* (No. 834). Retrieved from http://eprint.iacr.org/2013/834.

Buterin, V. (2014). A next-generation smart contract and decentralized application platform. *White Paper*.

[Case Study] 51% attack and double spending. (2014, May 15). Retrieved April 6, 2018, from https://www.reddcoin.com/case-study-51-attack-and-double-spending/.

Castro, M., & Liskov, B. (n.d.). Practical Byzantine Fault Tolerance. In the *Proceedings of the Third Symposium on Operating Systems Design and Implementation*, New Orleans, USA, February 1999. Retrieved from http://pmg.csail.mit.edu/papers/osdi99.pdf.

Chen, T., Li, X., Luo, X., & Zhang, X. (2017). Under-optimized smart contracts devour your money. In *2017 IEEE 24th International Conference on Software Analysis, Evolution and Reengineering (SANER)* (pp. 442-446). https://doi.org/10.1109/SANER.2017.7884650

Collision attack–Bitcoin Wiki. (n.d.). Retrieved March 29, 2018, from https://en.bitcoinwiki.org/wiki/Collision_attack.

Comprehensive Blockchain Glossary: From A-Z. (n.d.). Retrieved April 7, 2018, from https://blockgeeks.com/guides/blockchain-glossary-from-a-z/.

Conti, M., E, S. K., Lal, C., & Ruj, S. (2017). A survey on security and privacy issues of Bitcoin. *ArXiv:1706.00916 [Cs]*. Retrieved from http://arxiv.org/abs/1706.00916.

Cook, T., Latham, A., & Lee, J. H. (n.d.). *DappGuard : Active monitoring and defense for solidity smart contracts.* Retrieved from https://pdfs.semanticscholar.org/7438/ffd4c3b45a6d239815df377a453adfa890fa.pdf.

Costa, R. (2014/2018). W*iki: The Ethereum Wiki*–ethereum. Retrieved from https://github.com/ethereum/wiki (Original work published 2014).

Courtois, Bahack, N. T., & Lear. (2014). On subversive miner strategies and block withholding attack in Bitcoin digital currency.pdf. Retrieved from https://arxiv.org/pdf/1402.1718v2.pdf.

Cryptocurrency Mining Malware. (2017). Retrieved from https://sucuri.net/documentation/Sucuri-eBook-Cryptomining-Malware.pdf.

Decker, C., & Wattenhofer, R. (2014). Bitcoin transaction malleability and MtGox. *ArXiv:1403.6676 [Cs]*, *8713*, 313–326. https://doi.org/10.1007/978-3-319-11212-1_18.

Dickson, B. (2017, January 22). Blockchain’s brilliant approach to cybersecurity. Retrieved February 26, 2018, from https://venturebeat.com/2017/01/22/blockchains-brilliant-approach-to-cybersecurity/.

Dinh, T. T. A., Wang, J., Chen, G., Liu, R., Ooi, B. C., & Tan, K.-L. (2017). Blockbench: A framework for analyzing private blockchains. In *Proceedings of the 2017 ACM International Conference on Management of Data* (pp. 1085-1100). ACM.

Dixon, B. (n.d.). *Block party on the Blockchain*. Retrieved from https://static1.squarespace.com/static/58536224b3db2b6a3ffa6657/t/585700fc03596ed9d7689c39/1482096892985/Block+Party+On+The+Blockchain.pdf.

Douceur, J. R. (n.d.). *sybil.pdf*. Retrieved from http://www.divms.uiowa.edu/~ghosh/sybil.pdf

Ellervee, A. (2017). *A reference model for Blockchain-based distributed ledger technology*. (Unpublished master's thesis), University of Tartu.

Eskandari, S., Clark, J., Barrera, D., & Stobert, E. (2015). A first look at the usability of bitcoin key management. *ArXiv:1802.04351 [Cs]*. https://doi.org/10.14722/usec.2015.23015

Ethereum Charts and Statistics. (n.d.). Retrieved March 23, 2018, from https://etherscan. io/charts.

Ethernodes.org–The ethereum node explorer. (n.d.). Retrieved March 25, 2018, from https://www.ethernodes.org/network/1.

Eyal, I., Emin, A. E. G., & Sirer, G. (n.d.). Bitcoin-NG: A scalable Blockchain protocol. In the *Proceedings of the 13th USENIX Symposium on Networked Systems Design and Implementation* (NSDI).

Eyal, I., & Sirer, E. G. (2014). Majority is not enough: Bitcoin mining is vulnerable. In N. Christin & R. Safavi-Naini (Eds.), *Financial cryptography and data security* (Vol. 8437, pp. 436-454). Berlin, Heidelberg: Springer Berlin Heidelberg. https://doi.org/10.1007/ 978-3-662-45472-5_28

Eyal, I. E., & Sirer, E. G. (n.d.). *How to disincentivize large Bitcoin mining pools*. Retrieved March 16, 2018, from http://hackingdistributed.com/2014/06/18/how-to-disincentivize-large-bitcoin-mining-pools/.

Fromknecht, C., & Yakoubov, S. (n.d.). CertCoin: A NameCoin Based Decentralized Authentication System 6.857 Class Project, 19.

Gervais, A., Karame, G. O., Wüst, K., Glykantzis, V., Ritzdorf, H., & Capkun, S. (2016). On the security and performance of proof of work blockchains. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security* (pp. 3-16). ACM.

Giechaskiel, I., Cremers, C., & Rasmussen, K. (2016). *On Bitcoin security in the presence of broken crypto primitives* (No. 167). Retrieved from http://eprint.iacr.org/2016/167

Global Bitcoin nodes distribution. (n.d.). Retrieved March 23, 2018, from https://bitnodes.earn.com/nodes/.

Global DDoS Threat Landscape | Q3 2017 | Incapsula. (n.d.). Retrieved March 21, 2018, from https://www.incapsula.com/ddos-report/ddos-report-q3-2017.html.

Goldberg, E. H. A. K., & Aviv Zohar, Sharon. (2015). *Eclipse attacks on Bitcoin's peer-to-peer network* (No. 263). Retrieved from http://eprint.iacr.org/2015/263.

Goldfeder, S., Gennaro, R., & Kalodner, H. (n.d.). Securing Bitcoin wallets via a new DSA/ECDSA threshold signature scheme, p. 26.

Greenspan, D. G. (n.d.). MultiChain private blockchain. *White Paper,* p. 17.

Halpin, H., & Piekarska, M. (2017). Introduction to security and privacy on the Blockchain. In *Security and Privacy Workshops (EuroS&PW), 2017 IEEE European Symposium on* (pp. 1–3). IEEE.

Hashrate Distribution. (n.d.). Retrieved March 11, 2018, from https://blockchain.info/pools.

Heilman, E. (2014). *One weird trick to stop selfish miners: Fresh Bitcoins, A Solution for the Honest Miner.* (No. 007). Retrieved from http://eprint.iacr.org/2014/007.

Hileman, G., & Rauchs, M. (2017, September 22). *2017 Global Blockchain Benchmarking Study*. Available at SSRN: https://ssrn.com/abstract=3040224 or http://dx.doi.org/10.2139/ssrn.3040224.

Hyperledger_Arch_WG_Paper_1_Consensus.pdf. (n.d.). Retrieved from https://www.hyperledger.org/wp-content/uploads/2017/08/Hyperledger_Arch_WG_Paper_1_Consensus.pdf.

IANA IPv4 Address Space Registry. (n.d.). Retrieved March 27, 2018, from https://www.iana.org/assignments/ipv4-address-space/ipv4-address-space.xhtml.

Irreversible Transactions–Bitcoin Wiki. (n.d.). Retrieved March 12, 2018, from https://en.bitcoin.it/wiki/Irreversible_Transactions.

Ittay, E. (n.d.). The miner's dilemma.pdf. Retrieved from https://www.cs.cornell.edu/~ie53/publications/btcPoolsSP15.pdf.

Karame, G. O., Androulaki, E., & Capkun, S. (2012). *Two Bitcoins at the price of one? Double-spending attacks on fast payments in Bitcoin* (No. 248). Retrieved from http://eprint.iacr.org/2012/248.

Karame, G. O., Androulaki, E., Roeschlin, M., Gervais, A., & Čapkun, S. (2015). Misbehavior in Bitcoin: A Study of double-spending and accountability. *ACM Transactions on Information and System Security, 18*(1), 1-32. https://doi.org/10.1145/2732196

King, S. (2013, July 7). Primecoin: Cryptocurrency with prime number proof-of-work. Retrieved from http://primecoin.io/bin/primecoin-paper.pdf.

King, S., & Nadal, S. (2012, August 19). *Ppcoin: Peer-to-peer crypto-currency with proof-of-stake*. Self-published Paper.

Kwon, Y., Kim, D., Son, Y., Vasserman, E., & Kim, Y. (2017). *Be selfish and avoid dilemmas: Fork After Withholding (FAW) attacks on Bitcoin* (pp. 195-209). ACM Press. https://doi.org/10.1145/3133956.3134019

Lamport, L., Shostak, R., & Pease, M. (1982). The Byzantine generals problem. *ACM Transactions on Programming Languages and Systems (TOPLAS), 4*(3), 382-401.

Lei, M. (n.d.). *Exploiting Bitcoin's topology for double-spend attacks*. (Bachelor's thesis), Swiss Federal Institute of Technology, Zurich.

Lin, I.-C., & Liao, T.-C. (2017). A survey of Blockchain security issues and challenges. *IJ Network Security*, *19*(5), 653-659.

Lopp, J. (2016, November 13). *Bitcoin's security Model: A deep dive*. Retrieved April 6, 2018, from https://www.coindesk.com/bitcoins-security-model-deep-dive/.

Luu, L., Chu, D.-H., Olickel, H., Saxena, P., & Hobor, A. (2016). *Making Smart Contracts Smarter* (No. 633). Retrieved from http://eprint.iacr.org/2016/633

Luu, L., & Velner, Y. (n.d.). SMARTPOOL: *Practical decentralized pooled mining*. At 26th {USENIX} Security Symposium ({USENIX} Security 17):1409-1426

Mandeleil, R. (2015/2018). *devp2p: ÐΞV's p2p network protocol & framework*. *ethereum*. Retrieved from https://github.com/ethereum/devp2p (Original work published 2015).

Marcus, Y., Heilman, E., & Goldberg, S. (n.d.). *Low-resource eclipse attacks on Ethereum's peer-to-peer network* (p. 15). Retrieved from https://www.cs.bu.edu/%7Egoldbe/projects/eclipseEth.pdf.

Mcleod, S. (2008). *Qualitative vs quantitative data | Simply psychology*. Retrieved April 7, 2018, from https://www.simplypsychology.org/qualitative-quantitative.html.

Mearian, L. (2018, January 18). *What is blockchain? The most disruptive tech in decades*. Retrieved February 23, 2018, from https://www.computerworld.com/article/3191077/security/what-is-blockchain-the-most-disruptive-tech-in-decades.html.

Nakamoto, S. (n.d.). Bitcoin: A Peer-to-Peer Electronic Cash System (p. 9). Retrieved from https://bitcoin.org/bitcoin.pdf.

Narayanan, A. (2017). *Bitcoin and cryptocurrency technologies*. Retrieved from https://press.princeton.edu/titles/10908.html.

Park, J., & Park, J. (2017). Blockchain security in cloud computing: Use cases, challenges, and solutions. *Symmetry*, *9*(8), 164. https://doi.org/10.3390/sym9080164

Pattison, I. (2017, April 11). *4 characteristics that set blockchain apart*. Retrieved February 24, 2018, from https://www.ibm.com/blogs/cloud-computing/2017/04/characteristics-blockchain/.

Pay, S. (2017, November 23). *Towards common blockchain architecture—an "ISO OSI for blockchain" primer*. Retrieved March 2, 2018, from https://medium.com/@scanpayasia/towards-common-blockchain-architecture-an-iso-osi-for-blockchain-primer-778db4e5b35c.

Piscini, E., Guastella, J., Rozman, A., & Nassim, T. (2016, February 24). *Blockchain: democratized trust*. Retrieved November 13, 2017, from https://dupress.deloitte.com/dup-us-en/focus/tech-trends/2016/blockchain-applications-and-trust-in-a-global-economy.html.

Podolanko, J. P., Ming, J., & Wright, M. (n.d.). *Countering double-spend attacks on Bitcoin fast-pay transactions* (p. 7). Retrieved from http://www.ieee-security.org/TC/SPW2017/ConPro/papers/podolanko-conpro17.pdf.

Pool vs. solo mining–Bitcoin Wiki. (n.d.). Retrieved March 17, 2018, from https://en.bitcoin.it/wiki/Pool_vs._solo_mining.

Poon, J., & Buterin, V. (n.d.). Plasma: scalable autonomous smart contracts (p. 47). Retrieved from https://www.plasma.io/.

Proof-of-authority. (2018, February 27). In *Wikipedia*. Retrieved from https://en.wikipedia.org/w/index.php?title=Proof-of-authority&oldid=827938861.

Protocol documentation–Bitcoin Wiki. (n.d.). Retrieved March 4, 2018, from https://en.bitcoin.it/wiki/Protocol_documentation#version.

Rashid, S. (2018, March 20). *Breaking the ledger security model*. Retrieved March 30, 2018, from https://saleemrashid.com/2018/03/20/breaking-ledger-security-model/.

Ream, J., Chu, Y., & Schatsky, D. (2016, June 8). Upgrading blockchains: Smart contract use cases in industry. Retrieved April 6, 2018, from https://www2.deloitte.com/insights/us/en/focus/signals-for-strategists/using-blockchain-for-smart-contracts.html.

REMME Technical Paper. (2017, November 16). Retrieved from https://longcatchain.com/data/files/181217050_452110203.pdf.

RES, D. M. (n.d.). The stellar consensus protocol: A federated model for internet-level consensus (p. 32).

Rivlin, B. (2016, November 28). *Vitalik Buterin on empty accounts and the Ethereum state*. Retrieved April 1, 2018, from https://www.ethnews.com/vitalik-buterin-on-empty-accounts-and-the-ethereum-state.

Rosenfeld, M. (2011). Analysis of Bitcoin pooled mining reward systems. *ArXiv:1112.4980 [Cs]*. Retrieved from http://arxiv.org/abs/1112.4980.

Schwartz, D., Youngs, N., & Britto, A. (n.d.). *The ripple protocol consensus algorithm* (p. 8). Retrieved from https://cryptorum.com/threads/ripple-whitepaper-ripple-protocol-consensus.42/.

Seco, A. (n.d.). *BLOCKCHAIN: Concepts and potential applications in the tax area* (1/3). Retrieved April 4, 2018, from https://www.ciat.org/blockchain-concepts-and-potential-applications-in-the-tax-area-13/?lang=en.

Sedgwick, K. (2018, April 5). *Verge is forced to fork after suffering a 51% attack*. Retrieved April 6, 2018, from https://news.bitcoin.com/verge-is-forced-to-fork-after-suffering-a-51-attack/.

Solat, S., & Potop-Butucaru, M. (2016). ZeroBlock: Timestamp-free prevention of block-withholding attack in Bitcoin. *ArXiv:1605.02435 [Cs]*. Retrieved from http://arxiv.org/abs/1605.02435.

Sompolinsky, Y., & Zohar, A. (n.d.). *Bitcoin's security model revisited* (p. 24). Retrieved from http://www.cs.huji.ac.il/~yoni_sompo/pubs/16/security_model.pdf.

Srinivasan, P. (2017, November 9). *Healthcare Blockchain: How smart contracts could revolutionize care delivery | Prolifics*. Retrieved April 6, 2018, from https://www.prolifics.com/blog/healthcare-blockchain-how-smart-contracts-could-revolutionize-care-delivery.

Staples, M., Chen, S., Falamaki, S., Ponomarev, A., Rimba, P., Tran, A. B., Weber, I., Xu, X., Zhu, J…. (2017). *Risks and opportunities for systems using blockchain and smart contracts*. Data61 (CSIRO), Sydney.

Swan, M. (2015). *Blockchain: blueprint for a new economy* (1st ed.). Beijing : Sebastopol, CA: O'Reilly.

van den Hooff, J., Kaashoek, M. F., & Zeldovich, N. (2014). VerSum: Verifiable computations over large public logs (pp. 1304-1316). ACM Press. https://doi.org/10.1145/2660267.2660327

Vasek, M., Thornton, M., & Moore, T. (2014). Empirical analysis of denial-of-service attacks in the Bitcoin Ecosystem. In R. Böhme, M. Brenner, T. Moore, & M. Smith (Eds.), *Financial cryptography and data security* (Vol. 8438, pp. 57–71). Berlin, Heidelberg: Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-662-44774-1_5

Vasin, P. (2014). Blackcoin's proof-of-stake protocol v2. *URL: Https://Blackcoin. Co/Blackcoin-Pos-Protocol-v2-Whitepaper. Pdf*.

vbuterin. (n.d.). Long-term gas cost changes for IO-heavy operations to mitigate transaction spam attacks. Issue #150 · ethereum/EIPs. Retrieved April 1, 2018, from https://github.com/ethereum/EIPs/issues/150.

Vorick, D., & Champine, L. (n.d.). Sia: Simple Decentralized Storage. Retrieved from https://cryptorum.com/threads/sia-whitepaper-simple-decentralized-storage.91/.

Vyas, C. A., & Lunagaria, M. (2014). Security concerns and issues for Bitcoin. In *the Proceedings of National Conference cum Workshop on Bioinformatics and Computational Biology, NCWBCB-2014*. Citeseer.

Wan, Z., Lo, D., Xia, X., & Cai, L. (2017). *Bug characteristics in Blockchain systems: A large-scale empirical study* (pp. 413–424). IEEE. https://doi.org/10.1109/MSR.2017.59

Wang, F. (2015, October 4). *Eclipse attacks on Bitcoin's peer-to-peer network*. Retrieved March 27, 2018, from https://medium.com/mit-security-seminar/eclipse-attacks-on-bitcoin-s-peer-to-peer-network-e0da797302c2.

Weaknesses–Bitcoin Wiki. (n.d.). Retrieved March 11, 2018, from https://en.bitcoin.it/wiki/Weaknesses#Attacker_has_a_lot_of_computing_power.

whitepaper.pdf. (n.d.). Retrieved from https://docs.zilliqa.com/whitepaper.pdf.

Why Proof-of-Burn | Counterparty. (n.d.). Retrieved April 6, 2018, from https://counterparty.io/news/why-proof-of-burn/.

Wilkinson, S., Boshevski, T., Brandoff, J., Prestwich, J., Hall, G., Gerbes, P., … Pollard, C. (n.d.). Storj A peer-to-peer Cloud storage network. *White paper*, 37.

Woolf, N. (2016, October 26). DDoS attack that disrupted internet was largest of its kind in history, experts say. Retrieved April 6, 2018, from http://www.theguardian.com/technology/2016/oct/26/ddos-attack-dyn-mirai-botnet.

Wüst, K. (2016). *Security of Blockchain technologies*. ETH Zürich.

Wüst, K., & Gervais, A. (2016). *Ethereum eclipse attacks*. ETH Zurich. https://doi.org/10.3929/ethz-a-010724205

Xu, X., Weber, I., Staples, M., Zhu, L., Bosch, J., Bass, L., … Rimba, P. (2017). A taxonomy of Blockchain-based systems for architecture design. In *Software Architecture (ICSA), 2017 IEEE International Conference on* (pp. 243-252). IEEE.

Yli-Huumo, J., Ko, D., Choi, S., Park, S., & Smolander, K. (2016). Where is current research on blockchain technology?—a systematic review. *PLOS ONE*, *11*(10), e0163477. https://doi.org/10.1371/journal.pone.0163477

Zheng, Z., Xie, S., Dai, H., Chen, X., & Wang, H. (2017). An overview of Blockchain technology: Architecture, consensus, and future Trends (pp. 557-564). IEEE. https://doi.org/10.1109/BigDataCongress.2017.85