

Vulnerability Assessment of Bitcoin Architecture



By

SADIA KHALIL

NUST201362940MSEEC63013F

Supervisor

Dr. SHAHZAD SALEEM

DEPARTMENT OF COMPUTING

A thesis submitted in partial fulfillment of the requirements for the degree
of MSIS

In

School of Electrical Engineering and Computer Science,
National University of Sciences and Technology (NUST),
Islamabad, Pakistan.

(December 2016)

Approval

It is certified that the contents and form of the thesis entitled “**Vulnerability Assessment of Bitcoin Architecture**” submitted by **SADIA KHALIL** have been found satisfactory for the requirement of the degree.

Advisor: **Dr.SHAHZAD SALEEM**

Signature: _____

Date: _____

Committee Member 1: **Mr.FAHAD SATTI**

Signature: _____

Date: _____

Committee Member 2: **Mr.UBAID-UR-REHMAN**

Signature: _____

Date: _____

Committee Member 3: **Miss.HIRRA ANWAR**

Signature: _____

Date: _____

Dedication

Dedicated

to

my parents, my husband and my teachers

Abstract

Bitcoin is considered to be the world's first peer to peer and unregulated crypto-currency which has received widespread popularity in the last few years. It is considered to be the most popular way of achieving open source P2P money. A large number of businesses have started accepting bitcoins e.g WordPress, Baidu, Amazon, Reddit, VMware, Subway and SoundCloud etc. It operates in cyberspace and requires a special software called Bitcoin wallet to be installed on the client's computer. The core of the Bitcoin protocol is the mining process which is meant for verification of transactions and bringing new bitcoins into the system. It involves a Proof-of-work (PoW) mechanism which is based on a complex cryptographic puzzle.

Looking analytically into the Bitcoin protocol, there are certain security issues in the Bitcoin protocol which make Bitcoin transactions a major target of fraudsters. Incidents related to bitcoins being stolen or Bitcoin exchanges being shut down due to various attacks are observed daily. As of now, there exists no comprehensive survey which highlights the existing vulnerabilities and attack possibilities in the Bitcoin architecture. We also review existing countermeasure techniques that can make Bitcoin architecture more efficient and secure. In order to highlight the weaknesses that can make Bitcoin transactions a major target of fraudsters, STRIDE threat modeling of the Bitcoin architecture has been performed.

One of the identified problems is the security of the web based Bitcoin wallets. The web based Bitcoin wallets, if not protected properly, can become a valuable target of theft. The web based hosted Bitcoin wallets are considered to be the most vulnerable type of Bitcoin wallets since they are hosted on the servers of a trusted third party. The aim of the research is to address the authentication and authorization issues in Bitcoin wallets. As a proof-of-concept, we use Java Cryptography Extension (JCE) classes, PKCS7, PBE encryption algorithm and Shamir Secret Sharing Algorithm in such a way that no other entity would be able to carry out transactions without the intervention of the legitimate user.

Certificate of Originality

I hereby declare that this submission is my own work and to the best of my knowledge it contains no materials previously published or written by another person, nor material which to a substantial extent has been accepted for the award of any degree or diploma at NUST SEECS or at any other educational institute, except where due acknowledgement has been made in the thesis. Any contribution made to the research by others, with whom I have worked at NUST SEECS or elsewhere, is explicitly acknowledged in the thesis.

I also declare that the intellectual content of this thesis is the product of my own work, except for the assistance from others in the project's design and conception or in style, presentation and linguistics which has been acknowledged.

Author Name: **SADIA KHALIL**

Signature: _____

Acknowledgment

I am thankful to Allah Almighty for His blessings, which allowed me to accomplish this mile stone of my educational life.

I would especially like to thank my supervisor, Dr. Shahzad Saleem, whose guidance had been a constant companion throughout the course of this dissertation.

I would also like to thank my husband and my parents for their encouragement and moral support, which helped me get through this challenging phase smoothly.

I am also thankful to my friends and colleagues, especially the lab members of KTH AIS lab, without their cooperation and company, this journey would not be completed.

Sadia Khalil

Contents

1	PROLOGUE	1
1.1	Introduction and Motivation	1
1.1.1	Security of Bitcoin	2
1.2	Aim and Scope	3
1.3	Research Contributions	4
1.4	Research Methodology	5
1.5	Limitations	5
1.6	Outline of the Work	6
2	BITCOIN ARCHITECTURE	8
2.1	Components of a Bitcoin System	10
2.1.1	Bitcoin Exchanges	10
2.1.2	Bitcoin Wallets	10
2.1.3	Bitcoin Miners	10
2.2	Bitcoin Protocol	11
2.2.1	Proof-of-work	13
2.2.2	Conflict Resolution and Consensus	15
2.3	Information Propagation in Bitcoin Network	15
2.3.1	Network Topology	16
2.3.2	Propagation pipelining	17
3	VULNERABILITIES AND ATTACKS PREVALENT TO BITCOIN	18
3.1	Double Spending	19
3.1.1	Attacks in Normal Transactions	20
3.1.2	Attacks in Fast Payment Transactions	21
3.2	Selfish Mining	23
3.3	Compromising Anonymity	23
3.4	Malware Attacks	24
3.5	Comparison with other Crypto-currencies in terms of Possi- bility of Attacks	25

4	THREAT MODELING	28
4.1	Spoofing	29
4.2	Tampering	30
4.3	Repudiation	31
4.4	Information Disclosure	32
4.5	Denial-of-service(DOS)	33
4.6	Elevation of Privileges	34
5	AN OVERVIEW OF SECURITY OF WEB BASED BIT-COIN WALLETS	36
5.1	Security Of Web based Bitcoin Wallets	36
5.2	Security of Local Wallets vs. Web based wallets	37
5.3	Related Work	38
5.3.1	Naive Approach	39
5.3.2	Securing wallets through Threshold signatures	39
5.3.3	Securing wallets through Multi-signatures	40
6	DESIGN AND METHODOLOGY	41
6.1	Design and Modules	41
6.1.1	Proposed Scheme for Securing Private Keys in Hosted Wallets	41
6.1.2	High level Architecture of Proposed System	44
6.1.3	Implementation Environment	45
6.2	Use Case Scenario	46
6.2.1	Registration Process	46
6.2.2	Authentication Mechanism	46
6.3	Evaluation	46
6.3.1	Agents Used in Our Proposed Scheme	47
6.3.2	Attributes Used in Our Proposed Scheme	47
6.3.3	Security Claims and Evaluation Results of our Proposed Scheme	47
7	CONCLUSION AND FUTURE WORK	52
7.1	Contribution Summary	52
7.2	Conclusion	53
7.3	Future Research Directions	54
A	Evolution of Other Crypto-Currencies	55

B Proposed Defense Mechanisms	57
B.1 Solutions for Avoiding Double Spending	57
B.2 Solutions for Avoiding Selfish Mining	58
B.3 Solutions for Attacks on Anonymity	59
B.4 Solutions for Malware Attacks	60
C SPDL Script for Our Proposed Scheme	63

List of Figures

1.1	Top-Down Approach for the Thesis	5
2.1	Bitcoin Architecture	8
2.2	Digital Signatures in Bitcoin Transactions	12
2.3	A Bitcoin Transaction example	14
2.4	Forks in a Block Chain	15
2.5	Propagation of blocks/transactions from one node to another .	16
2.6	Improved scheme for propagation of blocks/transactions. . . .	17
3.1	Vulnerabilities in the Bitcoin architecture	19
3.2	Types of Double Spending Attacks	20
3.3	Double Spending in Fast transactions	22
4.1	Attack Tree for Spoofing Attacks on Bitcoin protocol	30
4.2	Attack Tree for Tampering Attacks on Bitcoin protocol	31
4.3	Attack Tree for Repudiation Attacks on Bitcoin protocol . . .	32
4.4	Attack Tree for Information Disclosure Attacks on Bitcoin pro- tocol	33
4.5	Attack Tree for DOS Attacks on Bitcoin protocol	34
4.6	Attack Tree for Elevation of Privileges Attack on Bitcoin pro- tocol	35
6.1	A Secure Scheme for the Protection of Hosted Bitcoin Wallets	42
6.2	High Level Architecture Diagram for The Proposed Scheme . .	45
6.3	Evaluation Results of Our Proposed Scheme	50
B.1	Relationship between Threshold and Number of honest miners that choose to mine on pool's block	58
B.2	Relationship between Threshold and Number of honest miners that choose to mine on pool's block.	59

List of Tables

2.1	Types of Bitcoin Wallets	11
3.1	Summary of Major Attacks in Bitcoin System	26
3.2	Crypto-currencies and their possible attacks	27
4.1	Classification of Threats in Bitcoin Architecture based on STRIDE Threat Model	29
6.1	Agents used in our proposed protocol	47
6.2	Attribute used in our proposed protocol	48
A.1	Table showing salient features of Bitcoin and other crypto- currencies	56
B.1	Threats/Vulnerabilities in Bitcoin protocol,and their possible countermeasures	62

Chapter 1

PROLOGUE

This chapter provides an introduction to the Bitcoin system and the motivation behind our research. It also highlights the problem statement, aims and scope of the thesis with contributions towards target research.

1.1 Introduction and Motivation

The ways through which we exchange goods have revolutionized from barter to skins and through commodity money to the fiat money. The advent of the digital currency systems has revolutionized the concept of money transfer by allowing the internet based creation, storage and transference of money. In the past few years, the digital currency systems have become an efficient means of money transfer. They have received worldwide adoption by providing a medium of exchange based on mathematical operations and by taking the currencies out of the control and manipulation of the governments. In addition to being used in the e-commerce and commercial sectors, the digital currencies have also attracted a large population of the earth which cannot get access to the formal banking systems. The crypto-currencies, being one of their types, involve different cryptographic functions for their creation and transference, in a trusted and secure environment. The use of crypto-currencies has progressed from a virtual concept to reality by the evolution of Bitcoin. The success of this concept has led to the creation of many other crypto-currencies which include Litecoin, PeerCoin, Namecoin, Quarkcoin, Primecoin and Zetacoin etc. (Stevenson, 2013). Bitcoin, along with the other crypto-currency systems, is very popular in the business world and the global economy, due to its decentralized and peer-to-peer architecture. In comparison with the other payment platforms like Visa(Evans, 2014), which maintain a private and secure communication network for sending and re-

ceiving money, Bitcoin uses the internet as its medium of transference.

The Bitcoin¹ protocol was first introduced in 2009 by a pseudonymous developer Satoshi Nakamoto (Nakamoto, 2008). Since then, it has been widely adopted as a payment procedure for the e-commerce industry as well as the regular stores. This crypto-currency along with the others, is considered to be a convenient way of achieving the open source peer-to-peer money. It operates in the cyberspace and requires Bitcoin wallets for storage purposes as well as for the generation of Bitcoin addresses. At the time of this writing, the Bitcoin market capitalization is \$5.8 billion (CoinMarketCap, 2016). Keeping in view its frequent usage, Bitcoin ATMs have been deployed in various parts of the world to facilitate its users(Perez, 2015). In comparison with the Visa transactions, where the transaction speed is 2000 tps (transactions per second) and PayPal which has 115tps transaction speed, the Bitcoin network is restricted to 7 tps(Sca, 2016). In spite of these statistics, the advantages of Bitcoin transaction over other transaction mechanisms like PayPal, Western Union and M-Paisa etc. cannot be neglected. It gives the users the advantage of carrying out instant, anonymous and irrevocable transactions with very low transaction fees. The original Bitcoin paper (Nakamoto, 2008) presents a brief overview of the architecture and the protocol but a lot of details are missing in it. With the passage of time, a number of suitable changes and ideas have been suggested through the Bitcoin Improvement Proposals (BIPs) which are incorporated after being approved by the Bitcoin community.

1.1.1 Security of Bitcoin

Despite being widely adopted by various large scale businesses, the Bitcoin transactions are still exposed to many known as well as zero-day attacks due to various vulnerabilities being exploited by the malicious entities. In order to achieve reliable and secure transactions, extensive research needs to be carried out to critically examine Bitcoin architecture and its level of security.

By Looking critically into the Bitcoin protocol, we can find a number of weaknesses that can be violated by the attackers for malicious purposes. In the past few years, a lot of vulnerabilities have been exploited causing the users to lose their bitcoins (Love, 2014),(Blasco, 2013),(from the arXiv, 2014).

¹The term bitcoin will be used in two contexts throughout the document. Bitcoin (with 'B' in capitals) refers to the protocol suggested by Satoshi Nakamoto whereas bitcoin refers to the currency or the amount (often abbreviated as BTCs).

Matthew Wilson et al. (Yelowitz and Wilson, 2015) analyze the characteristics of the Bitcoin users based on the Google search data and found that illegal activities and programming enthusiast are related to Bitcoin search but no correlation was found with political and investment motives. As of March 2014, bitcoins of worth 502,081,166.11\$ have been stolen (Love, 2014). Based on the empirical analysis of Bitcoin exchange risks, it is found that the failure rate of bitcoin exchanges is 40 % (Moore and Christin, 2013). Mt.Gox that was considered to be the largest Bitcoin exchange, got bankrupt in February 2014, allegedly due to theft, resulting in the loss of 850,000 bitcoins, out of which 20,000 were later recovered (Gox, nd).

1.2 Aim and Scope

”Bitcoin transactions are becoming a target of a number of security attacks that need to be holistically surveyed. One of the most vulnerable entities in the Bitcoin system is the web based Bitcoin wallet where users have to rely on a trusted third party for the management of their funds. A server side hack can result in the loss of users’ bitcoins without letting them know about the attack.”

This research effort targets the security aspects of the Bitcoin architecture by considering the threats/vulnerabilities prone to the Bitcoin system. The main goal of this thesis is to extensively explore the Bitcoin architecture from a security perspective and identify the vulnerabilities as well as their countermeasures. The Bitcoin system comprises of various entities each having its own security requirements.

From a practical point of view, the aim is to analyze the security of the Bitcoin entities and suggest improved schemes/mechanisms contributing to their security. However, it is infeasible to cover each and every entity as it will be beyond the scope of this thesis. The scope, therefore, is limited to the following research objectives.

- **Objective 1:** To study the Bitcoin architecture, its components and working in detail. To present the classification of different known and possible attacks based on the STRIDE threat modeling technique devised by Microsoft. This objective can be achieved by performing a survey of different attacks targeting Bitcoin and then formulating attack trees.

- **Objective 2:** To develop and design a secure scheme in order to address authentication and authorization issues in web based Bitcoin wallets. The objective could be fulfilled by analyzing their weaknesses and designing a scheme for enhancing security at the Bitcoin service end as a proof-of-concept.

1.3 Research Contributions

The primary contributions related to our research field herein are:

- **STRIDE Threat Modeling of the Bitcoin System:** A characterization of threats has been devised to evaluate the security of the Bitcoin transactions. The proposed classification is done using Microsoft's STRIDE threat modeling approach which classifies the known threats according to the exploits involved or the intention of the attacker. After performing an in depth analysis of the Bitcoin system (architecture, components, working, mechanisms) and highlighting various possible attacks that can take place during the Bitcoin transactions, attack trees have been established based on the STRIDE's categories (Spoofing, Tampering, Repudiation, Information Disclosure, Denial of Service (DoS) and Elevation of privileges. Keeping in view, the number of areas and components that come under the umbrella of the Bitcoin architecture and in order to narrow down the scope, we identified one key problematic area i.e Security of hosted bitcoin wallet services. We performed a literature survey on the proposed schemes for the security of hosted wallets and identified their shortcomings. The loopholes in these schemes led us to the designing of our proposed scheme for the protection of Hosted Bitcoin web wallets.
- **Secure scheme for web based Bitcoin wallets:** The web based hosted Bitcoin wallets, if not protected properly, can become a valuable target of theft. These Bitcoin wallets are considered to be the most vulnerable type of Bitcoin wallets since they are hosted on the servers of a trusted third party. This research addresses the authentication and authorization issues in Bitcoin web wallets. As a proof-of-concept, we have used Shamir Secret Sharing algorithm in our design and implementation in such a way that no other entity would be able to carry out transactions without the intervention of the legitimate user. The encryption key of the user's private keys is shared between the client and the service in such a way that for every transaction, the user is prompted to enter his/her part of the key.

1.4 Research Methodology

Figure 1.1 presents our top-down approach to narrow down this thesis work.

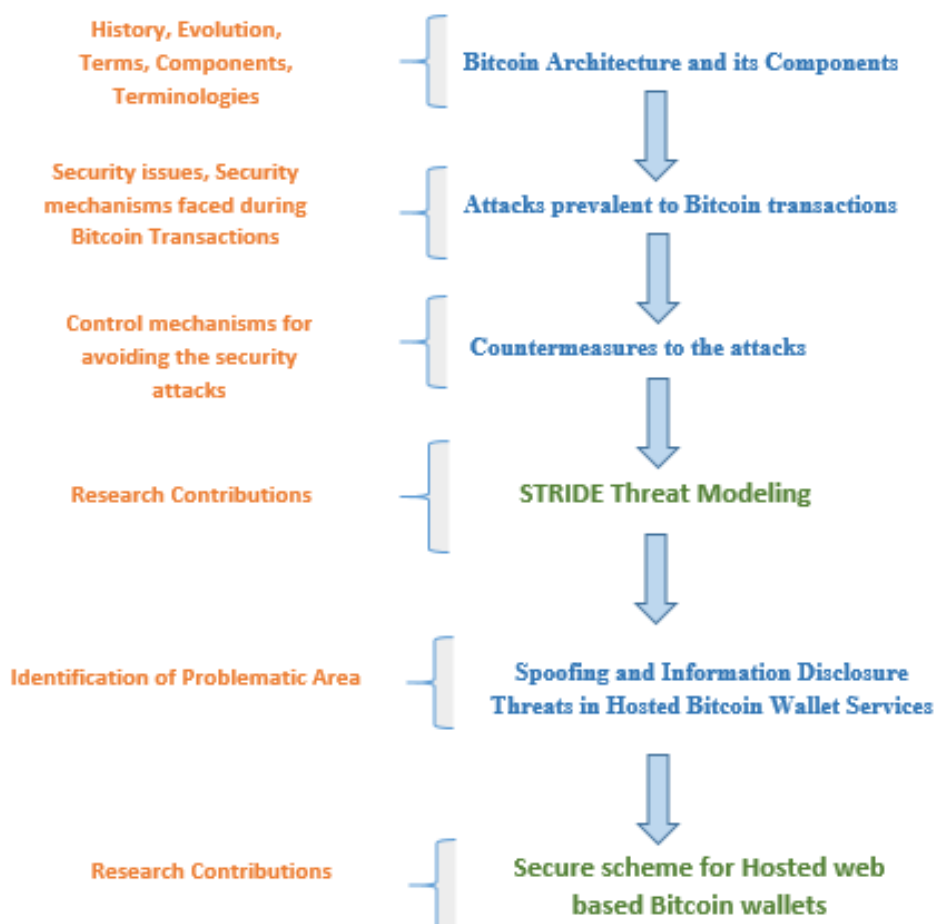


Figure 1.1: Top-Down Approach for the Thesis

1.5 Limitations

In order to define the scope of this work, we have confined our research work to the authorization and authentication issues faced in the web based wallets. Therefore, our major research contributions focus on providing security at the service end in order to avoid any server side hack. Other security concerns such as availability of the wallet service, misconfigurations, data stealing,

SQL injections attacks etc. are also the major concerns. Catering those issues in our thesis work is beyond the scope of this thesis. Hence, those security issues are not logically considered for this work.

1.6 Outline of the Work

In order to describe each of our research objectives eloquently, we classify our work over the course of seven chapters, where each chapter either provides a brief overview of concepts.

Chapter 1 presents an introduction to the Bitcoin domain as well as the motivation behind this research. The chapter discusses our research methodology which is explicitly established for the identification of problems found in the hosted web wallets. The aim and scope of this research has also been highlighted. The chapter presents our approach which we have used to conduct research for the security of the hosted web based Bitcoin wallets. Chapter 2 provides a high-level overview of the Bitcoin architecture, its key components and the Proof-of-work protocol that is the backbone of all of the Bitcoin transactions. A use case scenario of a Bitcoin transaction has been explained to briefly explain the working of the Bitcoin protocol. The mechanism behind the information propagation between the Bitcoin nodes has been explained in detail.

Bitcoin is considered to be the pioneer crypto-currency whose success has paved path for many other crypto-currencies which have been created to fulfill the shortcomings in Bitcoin. A comparative study of Bitcoin, along with three other crypto-currencies(chosen based on their high market capitalization) has been presented considering their extra features, advantages and disadvantages. In other words, this section presents the evolution of other crypto-currencies whose working backbone is same as that of the Bitcoin architecture but contains extra features that have been incorporated to overcome the weaknesses in the Bitcoin architecture.

Chapter 3 provides a holistic survey of the vulnerabilities and the security drawbacks of Bitcoin transactions. The chapter explains how attacks like double spending, selfish mining, compromising anonymity and malware attacks can result in the loss of bitcoins. On the basis of comparative study of other crypto-currencies, a comparison has been devised highlighting the possibility of various security attacks.

Chapter 4 consists of STRIDE threat modeling of Bitcoin architecture and related attack trees. In order to limit our scope, we have highlighted the problematic areas particular to the web based Bitcoin wallets.

Chapter 5 presents an overview of the local wallets and web based wallets

and why users prefer web wallets over local wallets. Security flaws in both of these types have been discussed. A literature survey of the already proposed schemes for the security of the Bitcoin wallets has been presented along with their drawbacks.

Chapter 6 provides details regarding the implementation and working of our proposed scheme as well as its formal evaluation through Scyther tool.

Chapter 7 presents a conclusion of our research by discussing the future of crypto-currencies. Moreover, the future work for our research has been proposed.

Chapter 2

BITCOIN ARCHITECTURE

In comparison with the traditional currencies that depend on a trust based model for their creation, circulation and transference, Bitcoin relies on a Proof-of-work based peer-to-peer model.

Being a crypto-currency, the Bitcoin protocol makes use of the hash functions and public key cryptography for the generation and transmission of bitcoins. A single bitcoin can be regarded as a series of digital signatures. For sending bitcoins to another entity, the sender digitally signs a hash of the involved previous transactions out of which the bitcoins are sent to the receiver as proof of possession of those bitcoins. The receiver can easily verify the series of digital signatures. Figure 2.1 shows the major components that constitute the Bitcoin Architecture.

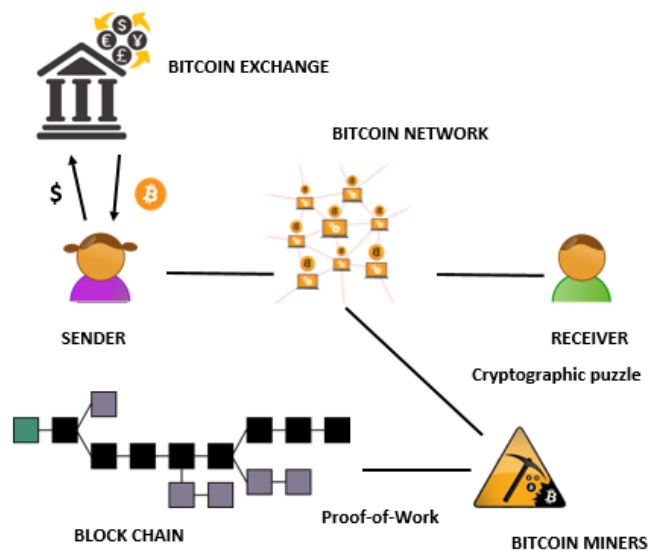


Figure 2.1: Bitcoin Architecture

The Bitcoin exchanges are not an inherent part of the Bitcoin protocol, however, they are one of the means for obtaining the bitcoins. The Bitcoin users can trade the bitcoins in exchange of the traditional or digital currency. Another way is to get bitcoins personally by asking the possessor of the bitcoins to transfer them to the buyer's Bitcoin wallet.

The Bitcoin users require Bitcoin wallets to carry out the transactions. Just like real life wallets, which are used to keep cash, Bitcoin wallets are also responsible of keeping record of the bitcoins sent and received by the owner. Bitcoin wallets generate a public/private key pair for carrying out a transaction.

The Bitcoin miners are the entities in the Bitcoin network that possess computational resources to compete in the mining process. Mining is the mechanism through which the transactions are recorded in the block chain. Mining involves a complex cryptographic mechanism known as Proof-of-Work (PoW). The motivation for the miners to perform mining is the Bitcoin reward that is granted to each miner. This reward is the source of creation of new bitcoins in the system. Initially, its value was 50 BTCs and is set in a way that it is halved after every 4 years (Con, 2016).

Individual mining is tedious and the miner has to be efficient enough to be the first one to solve the PoW puzzle successfully. If an entity is not interested in mining individually, it can join a mining pool. Joining a mining pool increases the probability of a miner in solving the PoW as the miners are assigned smaller cryptographic puzzles to solve and are able to combine their computational power with that of others in the mining pool.

In addition to the reward, the miners also receive a very small amount of bitcoins as the transaction fees. If a transaction is processed in a way that it draws bitcoins from many addresses, then greater transaction fees is associated with it due to the large transaction size. When a miner successfully generates a block, the information related to all of the transactions is incorporated in it and that miner can obtain the transaction fees.

The Block chain serves as a global ledger for keeping a record of all the confirmed transactions that have taken place since the first ever Bitcoin transaction.

2.1 Components of a Bitcoin System

2.1.1 Bitcoin Exchanges

Bitcoin users can get hold of bitcoins from Bitcoin exchanges where they can trade bitcoins in exchange of traditional currency or any other digital currency. Their working is similar to those of banks. A bitcoins' seller has to deposit some amount in an account of that exchange in the form of currencies supported by that exchange. This account balance is then used by the seller to trade bitcoins with other members of the exchange. Anyone willing to sell bitcoins can generate asks or 'sell orders' or can sell them at fixed rate. Buyers can generate offers to buy bitcoins using 'bids' or 'buy orders'. If the bid is greater than the sell order, then an exchange can take place successfully. Another way is to get bitcoins personally by asking the possessor of bitcoins to personally transfer bitcoins to the buyer's Bitcoin wallet.

2.1.2 Bitcoin Wallets

To start with a Bitcoin transaction, sender and receiver have to rely on the Bitcoin wallets . Just like real life wallets are used to keep cash, Bitcoin wallets are also responsible of keeping record of the bitcoins sent and received by the owner. It is also responsible for generating Bitcoin addresses. Bitcoin addresses can be compared with email addresses. For sending any email, the sender should know the email address of the receiver. Similarly, for receiving bitcoins, the sender should be aware of the receiver's Bitcoin address. Table 2.1 shows different types of Bitcoin wallets.

2.1.3 Bitcoin Miners

Bitcoin miners are entities in the Bitcoin network that possess reasonable resources to compete in the mining process. They are involved in solving a complex cryptographic problem known as Proof-of-Work(PoW)¹. The motivation for miners to perform mining is the Bitcoin reward that is granted to each miner. This reward is the source of creation of new bitcoins in the

¹Proof-of-work (PoW) functions/protocols are generally used to deter spam and DOS attacks. In Bitcoin, it refers to a cryptographic puzzle that requires extensive computational resources in order to be solved. Hence, protecting the system from being overtaken by attackers who introduce their allies to the network in order to get fraudulent transactions verified easily by the system.

Type of Wallet	Description
Software wallets	Software wallets can be installed on user's PC or mobile phone. After installation, the software wallets synchronizes with the Bitcoin network by installing the block chain which might take some time..
Web wallets	Bitcoin users can use the services provided by the trusted third parties. This provides convenience as users don't have to download full block chain and can access their wallets without being dependant on a specific device. Extra precautions should be taken while using this type of wallets as you are outsourcing your private keys. Two factor authentication is considered as an efficient security measure while using the web wallets.
Hardware wallets	Hardware wallets store private keys associated with the public keys and can only be accessed by maintaining a physical contact with the wallet. They provide both security and convenience as compared to other types. One of the main issues associated with software wallets is that if the media containing the private keys gets compromised due to some malware attack or theft, then it result in loss of the bitcoins. For using hardware wallets, the device containing the private keys should be attached to the user's PC using USB and should have no connection with the internet. In this way, a malware infected computer cannot do any harm to the bitcoins.
Paper wallets	The public/private key pair is printed on a piece of paper to remove it from the digital world and protecting it in the physical one. Paper wallets are used to protect the bitcoins to be stolen through internet.
Brain wallets	In brain wallets, a random pass phrase to access the bitcoins is memorized by the user. It should be long enough so that it could not be guesses easily and short enough to be memorized.
Cold wallets	Any wallet that has no connection to the internet is termed as cold wallet. It can be a paper wallet or a hardware wallet. Another possible way is to encrypt a wallet and store its key offline.

Table 2.1: Types of Bitcoin Wallets

system. Initially, its value was 50 BTCs and is set in a way that it is halved after every 4 years. If an entity is not interested in mining individually, it can join a mining pool. Joining a mining pool increases the probability of a miner in finding the nonce for PoW as miners are assigned smaller cryptographic puzzles to solve and are able to combine their computational power with that of others in the mining pool. Individual mining is tedious and the miner has to be lucky and efficient enough to be the first one to solve the PoW puzzle successfully.

2.2 Bitcoin Protocol

The Bitcoin wallets generate a public/private key pair for carrying out a transaction. The hash of the public key is referred to as Bitcoin address which can be used to send or receive transactions. The Private Key is used to digitally sign the transaction in order to add senders identity. All the transactions are signed and verified using the elliptic curve digital signature

algorithm ECDSA. In such signature algorithms, public key is derived by the multiplication of the base point of the elliptic curve with the private key. The elliptic curve used in Bitcoin is secp256k1 which is recommended in SEC 2 (Standards for efficient cryptography) (Research, 2000). Besides having the knowledge of the public key and the base point, it is not possible for anyone to derive the private key. This factor adds to the security of ECDSA. The ECDSA signatures also require a random parameter for each separate transaction made with the same private key.

Figure 2.2 explains how the Bitcoin transactions are digitally signed. A Bitcoin transaction comprises of two parts: input and output. Each part embeds certain scripts which control the future spending of the bitcoins that are being sent. The input of a transaction connects to a previous output. The Input consists of hashes of the previous transactions that client had with other users in the network. The amount of the current transaction is assigned out of these transactions. The purpose of taking hashes is to reference the output of the transactions from which this transaction is funded. It also comprises of a script called scriptSig which refers to the number of arguments that are expected by a script. The output consists of the amount to be sent to the receiver signed by the receiver's public key and the change to be sent back to the sender. A short script called scriptPubKey is also a part of the output which specifies the conditions under which the output could be redeemed. The script contains the hash of the ECDSA public key and a signature validation routine. For a valid transaction, the output script should evaluate to be true given the scriptSig provided in the input.

Each transaction is identified by a unique transaction ID which is the SHA-256 hash of the entire transaction.

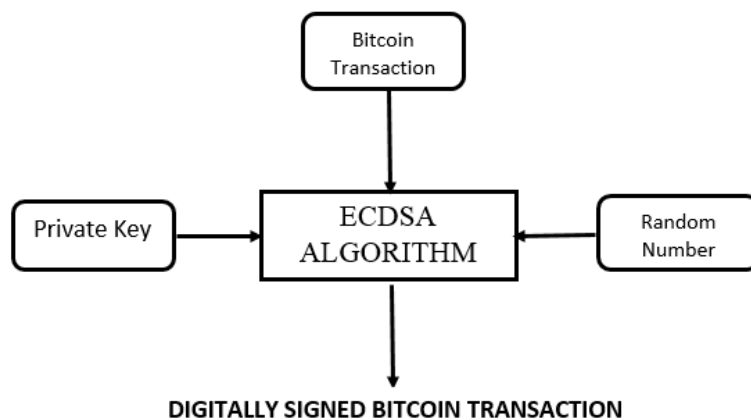


Figure 2.2: Digital Signatures in Bitcoin Transactions

2.2.1 Proof-of-work

Bitcoin transactions are verified through the Proof-of-work which is used as a countermeasure for double spending attacks (discussed in the next section). The receiver of the transaction requires confirmation in the form of a proof from the majority of the nodes, that the amount is not double spent. This proof is provided using the PoW system which is based on the idea of Adam Back's Hashcash (Back, 2002). Hashcash ensures that the requester has spent considerable amount of CPU resources before receiving the services.

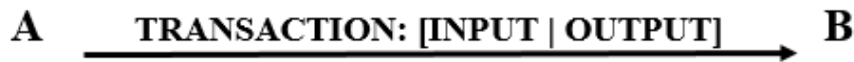
Bitcoin mining involves finding the PoW, in which a nonce is calculated by the miners using a random number generator (RNG) algorithm ran at a colossal rate. The nonce is calculated corresponding to an automatically generated challenge/difficulty. The block including the nonce when hashed under SHA 256, generate a string whose initial bits are consecutive zeroes(as specified in the challenge). The hash of the block (including the nonce) should satisfy the challenge/difficulty. The computational resources required for doing the PoW are directly proportional to the number of required zeroes. The first miner whose RNG spits out the correct nonce is the miner who gets the reward for solving the PoW. Any node of the Bitcoin network can become a miner if it possesses reasonable computing resources.

Once the PoW is solved, the block is broadcast to the network. The block will be accepted by the nodes only if the transactions are valid and are not double spent. Broadcasting the block also informs the other Bitcoin miners to quit mining this block and start working on another one. The verified block is then appended to the block chain. Each block is a reference to the previous one and results in formation of a continuous chain which serves as public ledger.

The average time for mining is about 10 minutes. Therefore, the memory pool of each node will contain all the unlogged transactions that have taken during the time period of these 10 minutes. For every 2016 blocks that have been mined, the network checks the performance of the miners. If it is greater than normal, the difficulty level of the PoW is increased. The difficulty in solving this cryptographic puzzle provides security as it ensures that no malicious entity can flood the block chain with conflicting blocks.

Suppose a merchant running an online business accepts bitcoins payments. In order to receive the payments, Bitcoin address of the merchant should be known to all of the customers. Practically, it is quite inefficient to

have a separate transaction for each bitcoin transfer, considering the small size of such transactions. In this regard, the Bitcoin protocol allows splitting and combining of transactions. Generally, there can be one input if previous transaction is large enough for the amount to be sent. If the previous transactions are small and amount to be sent is large, then there will be multiple inputs combining previous smaller transactions. Figure 2.3 explains a scenario of a Bitcoin transaction in which an entity A wants to send 50 bitcoins to entity B . Previously, A has received 20 BTCs from C , 25 BTCs from D and 10 BTCs from E . As discussed earlier, the input part of the transaction contains hashes of the previous transactions which is digitally signed with private key of $A(PR_A)$. Output contains the amount which is to be sent to B and the change amount that is to be sent back to A . In order to ensure that the desired amount is only received by B , the amount is encrypted using public key of $B(PU_B)$. The change amount is encrypted using public key of $A(PU_A)$.



where

INPUT: $\{H(20_C), H(25_D), H(10_E)\} PR_A$

OUTPUT: $\{50\} PU_B, \{5\} PU_A$

Figure 2.3: A Bitcoin Transaction example

For the validation of a transaction following steps take place:

1. Each node gets the transaction that was broadcast by the sender.
2. Each node checks the formatting of the transaction and confirms if the transaction is already present in the block chain. The unlogged transactions are temporarily stored in memory pools(memory buffers)of each node.
3. The miners compete in the formation of a block by participating in the mining process. PoW puzzle is calculated which involves a distributed

time stamping mechanism to prevent double spending. This work requires a lot of computational power and hence miners are rewarded with a small percentage of transaction fee. If the nonce that solves the PoW is successfully found, then all the transactions in the memory pool are transferred to the block, which is then broadcast. The purpose of the broadcast is to tell other nodes to start working on a new block.

4. The nodes validate the block by confirming the PoW and it is added to the block chain. The miners work on the construction of new blocks.

2.2.2 Conflict Resolution and Consensus

If two miners succeed in finding the PoW simultaneously, then after broadcasting, some of the nodes will receive one of the blocks whereas others will receive the other one. This results in the formation of a fork as shown in fig2.4. The nodes with the first block will work on it and keep a copy of the second one. Similarly, the nodes that receive the second block first will mine further blocks on this block and keep a copy of the first block. The conflict will be solved after a new PoW is found and one branch becomes longer than the other one. In such case, the blocks in the shorter branch will be discarded and the miners will shift to working on the longer one. The discarded blocks are called orphan blocks.

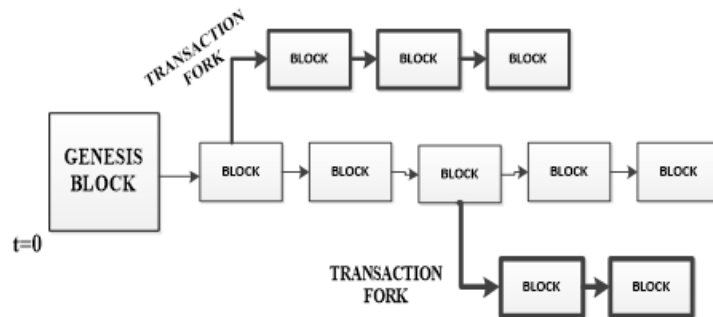


Figure 2.4: Forks in a Block Chain

2.3 Information Propagation in Bitcoin Network

Nodes in a Bitcoin network keep a record of all the transactions. Such information is used to verify the validity of the transactions that are taking place

in the network.

2.3.1 Network Topology

When a new node wants to enter the network, it has to query some DNS servers known as DNS seeds which are hard coded to the network. These DNS seed return a list of IP addresses of Bitcoin nodes that can be used for bootstrapping. After joining the network, the new node can ask its neighbor nodes for information of other nodes and known addresses. There is no specific policy for nodes leaving the network. So IP addresses of the node that have left the network remain stray in the network for several hours until nodes remove them from their list of addresses.

Whenever a new transaction takes place or a new block is created, the transaction and block messages are broadcast so that all nodes can synchronize and update their ledger. These update messages are not broadcast directly rather they follow a propagation protocol so that there is no duplication of information. When a new transaction or a block is introduced, the origin node will verify it and send an *inv* message to its neighbors. This *inv* message will contain hashes of the transactions or the blocks created by that node or received by it from other nodes. If the recipient node already has the updated information in its ledger, it will not respond to that message, otherwise it will send a *getdata* message to request for the transaction information or the block (Decker and Wattenhofer, 2013).

Figure B.2 depicts the flow of information regarding a block/transaction from one node to another.

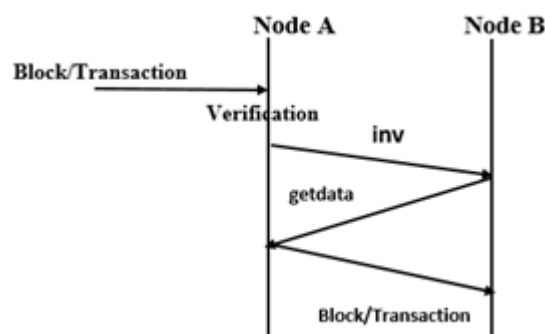


Figure 2.5: Propagation of blocks/transactions from one node to another

To make the propagation protocol faster and to minimize the propagation delay incurred while verifying transactions, verification process can be split into two parts. First part will be calculation of target/nonce corresponding

to the hash of the block in proof-of-work calculation and second will be verification of the transactions. Once the target/nonce is verified, block can be broadcast to the neighbors without individually verifying the transactions. This indicates that *inv* message will be sent whenever the target/nonce is verified. The actual verification of transactions will be done once the block has been sent to the neighbor.

2.3.2 Propagation pipelining

The protocol can be further improved by immediately sending the *inv* message once it has been received by that node. The response of the *getdata* message from the recipient will be queued until the first node receives the actual block and the target/nonce is verified. Individual verification of transactions will be done once the block has been sent as discussed above. Figure 2.6 indicates an improvised mechanism for information propagation from one node to another.

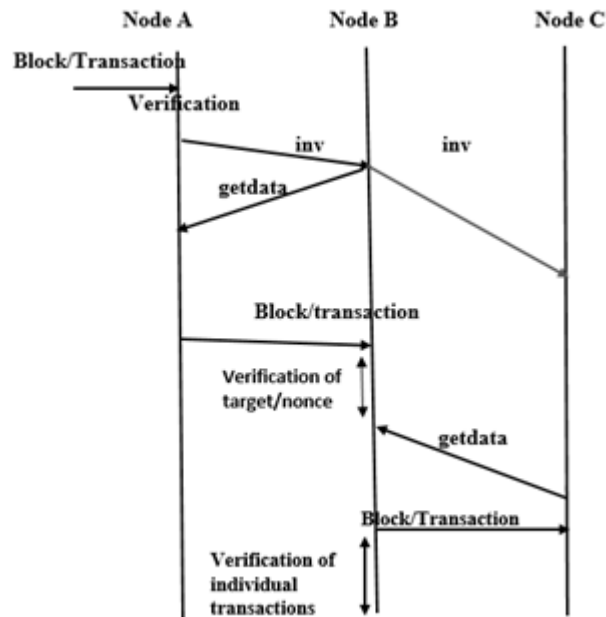


Figure 2.6: Improved scheme for propagation of blocks/transactions.

The chapter presents an overview of the Bitcoin architecture, its components and the protocol in detail. Keeping in view these details, we present a detailed survey of vulnerabilities targeting the Bitcoin components in the next section.

Chapter 3

VULNERABILITIES AND ATTACKS PREVALENT TO BITCOIN

Despite being an efficient and convenient way of electronic funds transfer, the Bitcoin transactions are becoming a target of many known and zero day attacks. No holistic solution has been proposed up till now to cater these attacks. This leads to the need of certain modifications in the protocol by the consent of the Bitcoin community. Figure 3.1 illustrates the vulnerabilities particular to the main entities in the Bitcoin architecture. The flow starts from a sender who has installed a Bitcoin wallet on his system in order to send some bitcoins to the receiver. The sender's bitcoin wallet is prone to malware attacks and theft which can result in the loss of private keys generated by the wallet. The sender can trick the receiver by double spending the amount of bitcoins he is going to send to the receiver. Hence, the sender can deprive the receiver from the sent bitcoins and can reuse those bitcoins. Malicious mining is a major threat in the Bitcoin transactions. If malicious miners have a large hash rate, then they can easily add conflicting transactions in block chain and can steal bitcoins. Bitcoin transactions which were initially claimed to be completely anonymous, can reveal the identity of the communicating ends. In order to buy bitcoins from the exchanges, the bitcoin users have to provide some personal information. This personal information can be revealed through man in the middle attack and malicious exchanges. Following are some of the attacks that are possible in Bitcoin transactions.

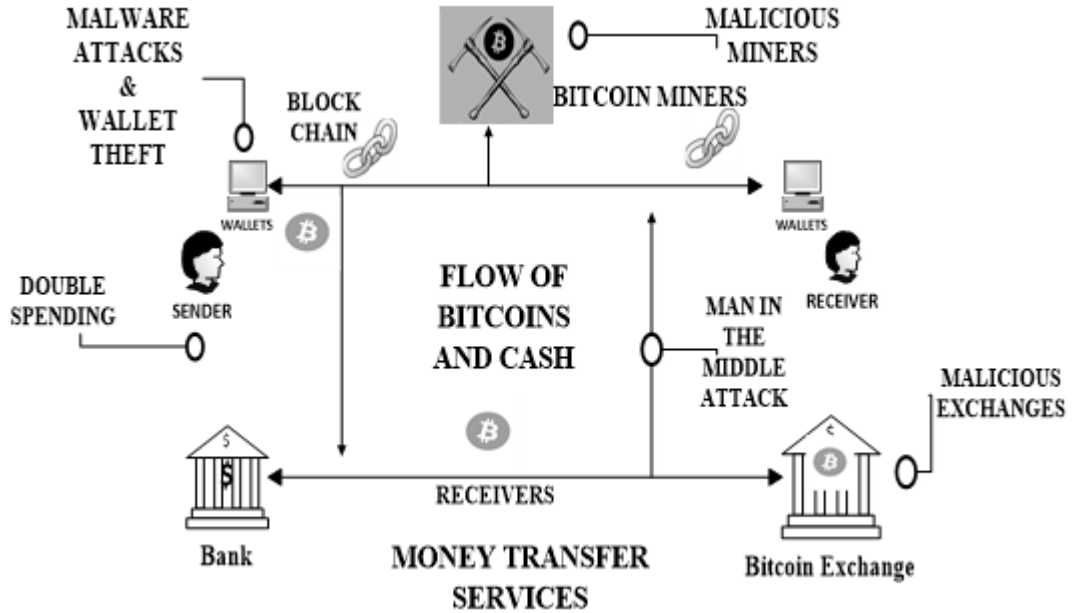


Figure 3.1: Vulnerabilities in the Bitcoin architecture

3.1 Double Spending

While a transaction is taking place, there is a possibility that the sender is sending same bitcoins to two receivers at a time. This leads to a problem known as double spending. Double spending attacks occur when a user assigns same amount of bitcoins (BTCs) to more than one user at a time, allowing users to spend the same amount of bitcoins twice. Bitcoin systems use hash based PoW schemes to protect against double spending attacks, but attackers have found ways to bypass such schemes.

Double spending attacks are more prevalent in fast payment mechanisms where time for exchange between money and services is very less. Figure 3.2 shows some of the main double spending attacks observed in Bitcoin systems.

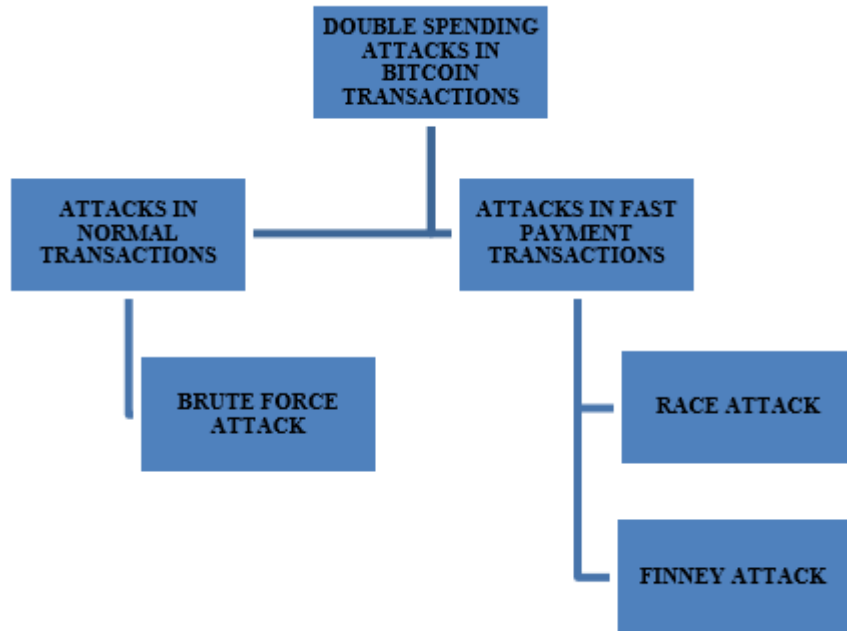


Figure 3.2: Types of Double Spending Attacks

3.1.1 Attacks in Normal Transactions

When a transaction is made, the sender broadcasts it to the entire network. Each node checks whether the transaction is properly structured and the amount is not previously spent somewhere else. After this confirmation, transaction is stored in the memory pool of the nodes. The miners then compete in mining process by doing the PoW(Nakamoto, 2008). If a peer succeeds in finding a nonce that solves POW, it includes all the transaction in its memory onto the newly created block. This block is then broadcast to the whole network. Each peer then ensures that the block is valid and every transaction incorporated in it is not double spent. If verification is successful, block is added to the block chain. In spite of this protection mechanism, attackers are successful in carrying double spending attack.

Brute Force Attack

The chances of a brute force attack being successful depend on the attacker's hash rate(Dou, 2016). If an attacker can control more than half of the networks hash rate, then this attack can take place with a high probability. The

attacker makes a transaction to the vendor and meanwhile keeps on mining conflicting block in which a double spending transaction is included. In the presence of some conflicting transactions, miners have to vote for the valid block by appending further blocks to the block the miners think is valid. The longest block chain is accepted whereas the other one is discarded. The attacker can get the false blocks verified by the network, by generating blocks in order to keep the chain longer than that of the legitimate peers (Drainville, 2012). After receiving the BTCs, the vendor waits for some confirmations and then provides the service. Once the service is provided, the attacker releases the fork by overtaking the number of blocks on the legitimate chain. Figure 2.4 highlights two conflicting blocks resulting in the formation of a transaction fork.

3.1.2 Attacks in Fast Payment Transactions

In fast payments, most of the times, the transaction is not of high value and the required response from the vendor should be immediate, so that the transactions are carried out instantly. On average it takes 10 minutes for a transaction to be verified and added to the block chain. Vendors like fast food restaurants, supermarkets, ATM machines, vending machines etc. cannot wait too long in order to provide services. Therefore, Bitcoin developers allow zero confirmation transactions so that vendor can provide immediate services to its customers for faster payments. Previous (till version 0.5.2) and current Bitcoin systems do not have protection mechanism against double spending in fast payments. Attacker can be clever enough to create conflicting transactions in which one of the transactions sends money to the vendor whereas the other one sends the same amount to the nodes which are controlled by the attacker.

Following is the description of some of the important attacks related to double spending in fast transactions.

Race Attacks

Degree of success of race attacks in Bitcoin protocol is very high. Karame et al (Karame et al., 2012) discuss a scenario in which attackers can successfully double spend bitcoins despite developers solutions to prevent it. The main supposition is that the attacker controls some of the nodes in the network. Remaining nodes are honest ones and their computational power is far greater than that of attacker and his allied peer nodes. This means that attacker is not capable of adding conflicting block in the block chain. Suppose attacker A wants to double spend the amount he wants to send to vendor V . In order

to carry out a successful race attack, the attacker A will create two transactions TR_A and TR_V . Both of the transactions have same input (hashes of the previous transactions from which bitcoins are being transferred) but different outputs (amount to be sent encrypted with public key of the recipient). If both transactions are made simultaneously, then they have an equal probability of being verified as nodes cannot accept both transactions having same input fields. Only the transaction that reaches first will be accepted after verification. If V receives TR_V and all majorities of the peers receive TR_A , then TR_A will be more likely be verified and included in the block. In this scenario it is necessary that the time at which V receives TR_V is less than the time at which V receives TR_A , which means that TR_V should reach V before TR_A . If V receives TR_A earlier than TR_V , then V will first store TR_A in his memory pool and reject TR_V . Hence, asking the attacker to re-issue the payment. After the attack is successful, the amount goes back to the attacker and by the time the vendor has realized that he has received an invalid transaction, the services have already been carried out. Figure 3.3 depicts a double spending attack in fast transactions.

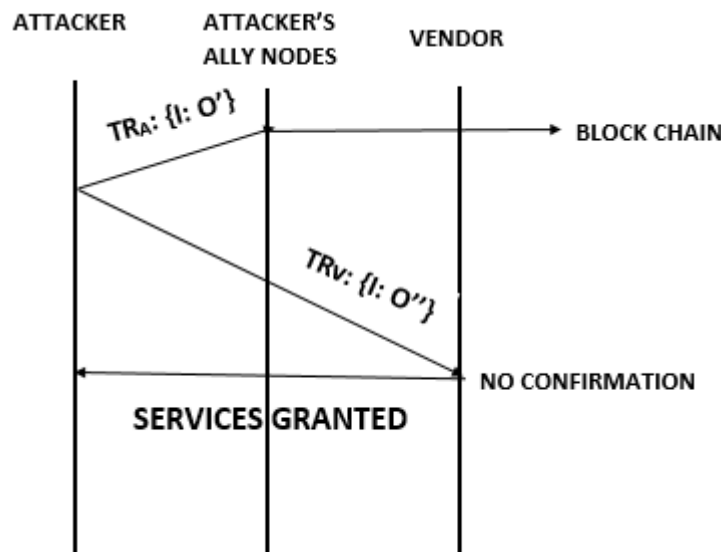


Figure 3.3: Double Spending in Fast transactions

Finney Attack/Block Withholding Attack

This attack is known to be the most expensive double spending attack (approx. 1\$ per second required by attacker to hold a calculated block)(Perry, 2012)When a transaction is made, the sender broadcasts it to the whole network. A transaction is not confirmed until it is added on a block in the block

chain by the Bitcoin miners. This attack requires an attacker to be a fraudulent miner in a way that he includes a conflicting transaction which sends some bitcoins back to himself. Meanwhile, he does not release that block due to which the block is not there in the block chain. The attacker then sends those bitcoins to a merchant to get some services. At the same time, the attacker will broadcast the unreleased block which will cause the transaction to the merchant to be overridden by the one back to the attacker.

3.2 Selfish Mining

Mostly, the Bitcoin miners work in the form of groups called mining pools and share the revenue according to their computational power. This honest mining process is the backbone of Bitcoin protocol. A lot of research has been done to analyze the behavior of bitcoin miners. According to the researchers, Ittay Eyal et al. (Eyal and Sirer, 2014), Bitcoin mining is no longer incentive compatible. A large number of selfish miners can join a mining pool and get their mining reward larger than their fair share.

In this attack, if the selfish miners find a block '**X**', they do not reveal it to the network, keeping it secret. All the other miners keep on finding that block, as without discovering it, they cannot move to the next block '**Y**'. In the meantime, the selfish miners start working on next blocks ('**Y**', '**Z**' etc.). When the honest miners find block '**X**' and distribute it to the network, the selfish miners immediately release all the work they have done so far, which might be several blocks (i.e. '**X** → **Y** → **Z**'). As the revenue goes to the miner of longer chain, the selfish miners win over the honest miners and get the incentive (Narayanan, 2013).

The revenue that the selfish miners get increases super-linearly with the size of the group. So, the colluding miners usually try to add more miners into their group to increase their cumulative computational power. This snowball scenario helps them to avoid the situation in which the honest miners find the block before the selfish miners would find the second block, due to enhanced computational resources.

3.3 Compromising Anonymity

Initially, when the Bitcoin protocol was designed, it was claimed that the protocol provides complete anonymity of both the sender and the receiver and they cannot be traced back. But recently, the researchers have found ways

to identify the sender and the receiver by using some extra computational power. The Bitcoin community itself states that the current implementation is not very anonymous (Ano, 2016). The Bitcoin protocol is now considered to be pseudo-anonymous.

Each Bitcoin transaction has a list of inputs and outputs. The input contains previous transactions so that the miners would verify that the bitcoins are not already spent (to avoid double spending). The transactions usually have two outputs; one output belongs to the receiver while the other output contains address of sender, to return the extra bitcoins. Using the reference to previous transactions from the input, transaction graphs can be built. These graphs can help in tracking the sender (Möser, 2013). People usually have to provide their personal information such as, copy of National ID card etc., to bitcoin exchanges, in order to buy bitcoins. Also, the transactions are stored publicly in the block chain, so anyone can analyze these transactions to find the links between the previous owners of bitcoins.

3.4 Malware Attacks

Attacks on Digital Wallets

Cyber-criminals use malicious software to attack systems. In most cases, the purpose is to steal critical information. They try to expand the attack by targeting more and more machines, creating a botnet. Bitcoin wallets are the digital place to store the private key that is used to access bitcoin address. If the key is compromised, all of the bitcoins are lost. To attack the bitcoin digital wallets, the attacker writes some malicious code, and spreads it to the botnet. The malware steals data, including wallet.dat file from the computers, stealing the bitcoins. The users can encrypt their files to prevent bitcoins loss but recent malwares have key logging capability added into them as well. They steal the password and decrypt the required file. Khelios and IRC are some known malware that do this job for the attackers (Blasco, 2013). MtGox, allegedly got bankrupt due to this attack. It lost 850,000 bitcoins. Out of which, 750,000 bitcoins belonged to customers (?). The attackers can launch attacks on the servers storing critical information like encryption keys of the Bitcoin users and can steal bitcoins from users' wallet accounts.

Mining BotNets Attacks

Apart from stealing the bitcoin digital wallets, some malware are designed to attack the computational resources of different machines. As, bitcoin miners get incentive in the form of bitcoins during the mining process, they always try to increase their computational power, to get more and more reward. The attackers have now found a way to win the competition. They write a malicious code and spread it to the botnet. The code installs Bitcoin daemon on victims machine and connect it to the mining pool. The attacker then uses victims computation power to mine bitcoins. As, the computation power is enhanced, the chance to win, by verifying blocks, is increased. Some of the known malwares are Zeus, Dorkbot and Ufasoft(Blasco, 2013). The malware is spread through fake emails, Skype, and phishing websites. All of the above mentioned attacks are the major ones and are constantly being modified by attackers to launch new attacks.

Table 3.1 shows summary of the major attack possibilities in Bitcoin along with their description.

3.5 Comparison with other Crypto-currencies in terms of Possibility of Attacks

Based on the features of each of the crypto-currencies as discussed in the appendix A, the following table 6.1 illustrates a list of crypto-currencies along with the possibilities of different attacks associated with them. The tick mark ✓ indicates the existence of a particular attack, whereas cross mark ✕ indicates its absence. As discussed in 3, Bitcoin is prone to attacks like double spending, selfish mining, compromising anonymity and malware attacks etc. Altcoins are designed either to provide additional functionalities on the top of the Bitcoin protocol or to fix the issues found in the protocol. Litecoin transactions have faster block generation rate and provide greater resilience to double spending attacks. As a security measure against double spending attacks, it is a standard practice for a merchant to wait for a sufficient number of blocks/confirmations to be added to its transaction's block. Litecoins transactions can have more confirmations during the same time that is used with Bitcoin, therefore the probability of double spending attacks is lesser. Litecoin, Peercoin and Namecoin are all pseudo-anonymous like Bitcoin. Malware attacks can be observed in all crypto-currencies and can be avoided by adopting efficient end user security practices. Peercoin prevents 51% attacks through proof-of-stake scheme which allows miners to generate

CHAPTER 3. VULNERABILITIES AND ATTACKS PREVALENT TO BITCOIN26

Attacks	Description
Brute Force Attack	It requires an attacker to control more than half of the network's hash rate in order to be successful. Adversary sends a transaction to the desired vendor and at the same time secretly keeps on mining further blocks that contain the double spent transaction. Vendor waits for n confirmations and provides the service. If the attacker has mined more than n block, he will release the fork. Hence the bitcoins that were spent earlier are regained by the attacker.
Race Attacks	This attack is specific to fast payment transactions in which services are granted without any transaction confirmation. Attacker creates two similar transactions with different recipients. One of the transaction is sent to the vendor whereas the other one goes to majority of the colluding nodes that have alliance with the attacker. The later has more probability of getting accepted in the block chain. So after the service has been granted, the attacker is able to spend those bitcoins again.
Finney Attack- /Block Withholding block	This attack takes place in fast payment transactions. Attacker sends some BTCs to the vendor and meanwhile keeps on mining block that contain transaction that sends those bitcoins back to himself. The attacker does not release the block until the service is granted. Once the service is granted without confirmation, attacker releases the block and is able to double spend those bitcoins.
Selfish Mining	Miners can join mining pools to increase their computational power. Selfish miners secretly keep on mining blocks to form a longer chain. When an honest miners mines a block which they already have mined, they immediately release the block. This results in all the PoW reward going to selfish miners.
Compromising Anonymity	Attacker is capable of tracing the sender and recipient of a transaction by manipulating different block chain services. This rejects the claim that Bitcoin transactions are completely anonymous.
Malware Attacks on Digital Wallets	Attack on confidentiality in which an adversary gains illegal access to critical information like private keys. Attacker uses malware to steal bitcoins from victim's wallet.
Mining Botnets At- tacks	Attackers spread malicious code through botnets in order to steal computational resources of victim systems. These resources are used for mining in order to get reward.

Table 3.1: Summary of Major Attacks in Bitcoin System

coins based on their share in the network. A successful double spending attack would require an attacker to possess a very large number of coins, which is practically infeasible. Peercoin prevents selfish mining by providing a fair distribution of coins scheme through proof-of-stake.

CHAPTER 3. VULNERABILITIES AND ATTACKS PREVALENT TO BITCOIN27

Crypto-currency	Attacks			
	Double Spending	Selfish Mining	Compromising Anonymity	Malware Attacks
Bitcoin	✓	✓	✓	✓
Litecoin	×	✓	✓	✓
Peercoin	×	×	✓	✓
Namecoin	✓	✓	✓	✓

Table 3.2: Crypto-currencies and their possible attacks

Chapter 4

THREAT MODELING

In software development, security features are mostly considered as non-functional requirements. Either they are added during the design phase as built-in features or incorporated afterwards as add-ons. The system is then tested by attacking its different components from security perspective, using different mechanisms like test cases, penetration testing and ethical hacking etc. Organizations hire red teams in order to view the system from the point of view of adversary. The system is evaluated by testing various components. One of the mechanisms to identify threats is through threat modeling.

In order to ensure secure working of any software, threat modeling is done during different phases of software development lifecycle(Möckel and Abdallah, 2010). It involves identification, assessment and documentation of vulnerabilities, threats, attacks and countermeasures during software development life cycle. Effective threat modeling results in considerable reduction of vulnerabilities. Considering the working of Bitcoin, security cannot be neglected. Bitcoin, itself has a strongly secure architecture due to the involvement of different cryptographic mechanisms. However, there are certain loopholes that can still be exploited, exposing the system to many attacks. After analyzing the Bitcoin protocol in detail and performing a survey on different attacks, we have classified the threats using STRIDE (Spoofing, tampering, Repudiation, Information Disclosure, Denial of Service and Elevation of Privileges) threat model(Microsoft, 2005). Furthermore, in order to present a clear picture of the threats, we have used Amenaza Secure-ITree(Limited,) for creating attack trees. The attack trees are used for providing a graphical and tree based models to present the ways in which the system can be compromised. The green shapes indicate 'OR' or logical disjunction, the blue shapes depict 'AND' or logical conjunction and the grey squares represent leaf nodes.

Table 4.1 presents each threat category and its definition with respect to the Bitcoin architecture.

Threat Category	Definition	Vulnerable Entity
Spoofing	Spoofing is referred to a situation in which an adversary impersonates another entity in order to access privileges that are only entitled to that entity. Attacker can do so by stealing the credentials of a legitimate user. If an attacker successfully spoofs the identity of a legitimate user while accessing Bitcoin wallets, then it can result in loss of bitcoins.	Bitcoin wallets.
Tampering	Tampering involves damaging the data integrity. Tampering can let an adversary or any other unauthorized entity to modify data. It may involve tampering the transactional data being sent or modifying the entries in the block chain.	Block chain.
Repudiation	Repudiation refers to the denial from sending or accepting. Bitcoin transactions are considered to be irrevocable. However, an entity can reverse a transaction by modifying the block chain which serves as a global log to keep a record of all the transactions. If an attacker receives more than half of the networks hash rate, then he can easily repudiate transactions.	Block chain.
Information disclosure	Information disclosure leads an attacker to reveal confidential information. Bitcoin transactions are claimed to be anonymous. However, malicious exchanges can reveal users critical information that has been provided by the user during buying of bitcoins. Block chain information services can be manipulated in order to trace a transaction back to its creator.	Bitcoin exchange, Block chain.
Denial of Service (DOS)	Denial of service refers to the disruption of a service. Attacker can overwhelm the Bitcoin network by flooding it with bogus transactions. This will result in slowing down the verification process for legitimate transactions.	Bitcoin network, Transaction confirmation process.
Elevation of Privileges	An attacker can acquire elevated privileges to access systems which are not available under normal circumstances. An adversary may also access Bitcoin wallets to make transactions on behalf of a legitimate user.	Bitcoin wallets, Normal users or Bitcoin users.

Table 4.1: Classification of Threats in Bitcoin Architecture based on STRIDE Threat Model

4.1 Spoofing

Initially, Bitcoin wallets were not encrypted by default. In most of the cases, Bitcoin wallets are unprotected or weakly protected using short length pass-

words, exposing the wallets to Brute force attack. Wallet theft can result in the loss of private keys by depriving the user from his bitcoins. Malware/spyware with key logging capabilities can also extract user credentials in order to get access to Bitcoin wallets. Online wallets can be exposed to session hijacking attacks through man in the middle attack and cookie theft. The highlighted areas indicate situations under which Bitcoin hosted wallet services can be attacked. In case of weak security measures implemented at the service end, an adversary can spoof the identity of a legitimate user by getting access to legitimate user's private keys and carrying out the transactions on his/her behalf. In all of these cases, the attacker can spoof the identity of the actual user to send and receive transactions. Figure 4.1 shows the attack tree for spoofing attacks on Bitcoin architecture.

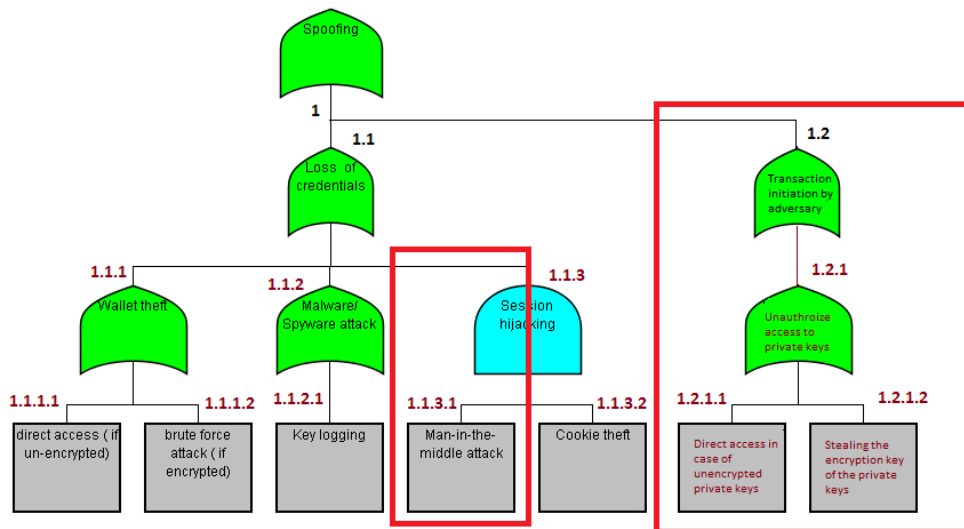


Figure 4.1: Attack Tree for Spoofing Attacks on Bitcoin protocol

4.2 Tampering

Transaction malleability attacks allow attackers to tamper the unique transaction ID of a transaction. This results in receiving of the transaction at the desired recipient end as the output of the transaction remains. However, the sender will not be aware of it. A fraudulent miner having more than half of the network's hash rate can tamper the entries in the block chain in order to get the transactions of his interest accepted by the network. Figure 4.2 shows the attack tree for tampering attacks on Bitcoin architecture.

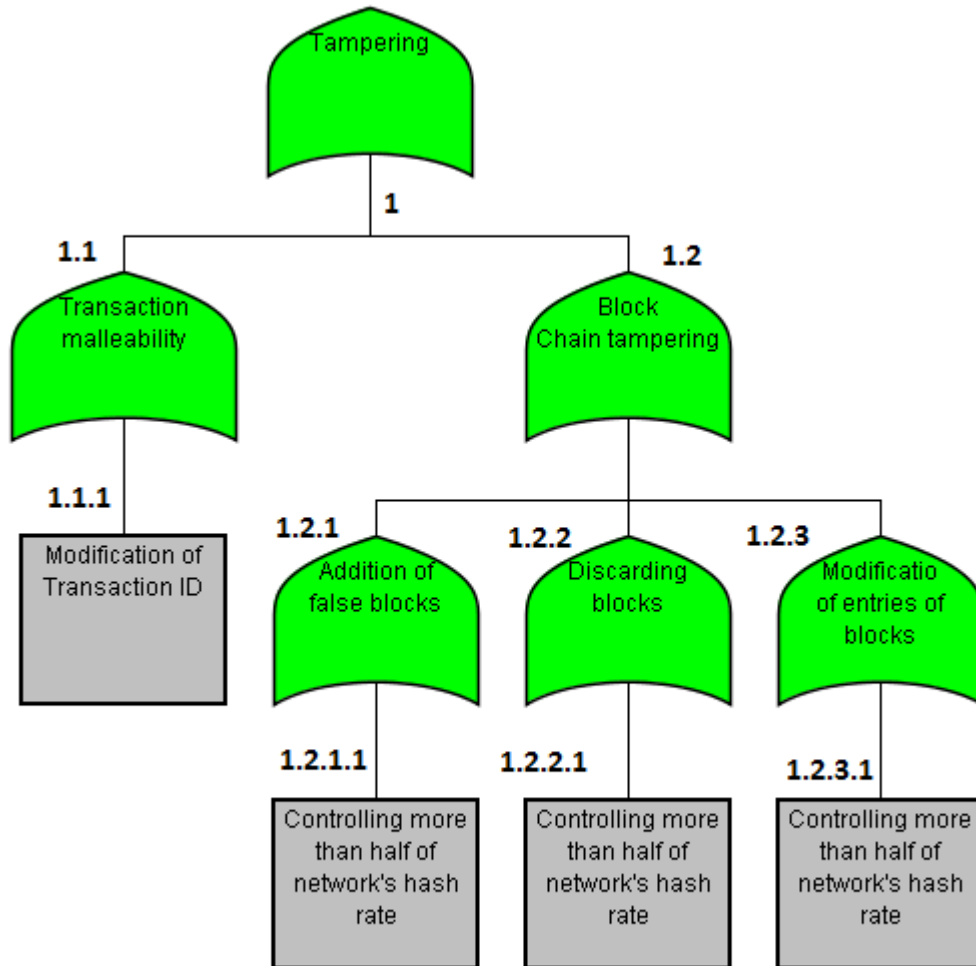


Figure 4.2: Attack Tree for Tampering Attacks on Bitcoin protocol

4.3 Repudiation

An attacker with large hash rate can modify the blocks in a block chain and hence can repudiate transactions. Figure 4.3 shows the attack tree for repudiation attacks on Bitcoin architecture.

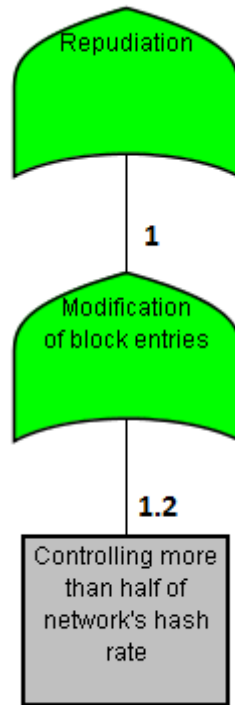


Figure 4.3: Attack Tree for Repudiation Attacks on Bitcoin protocol

4.4 Information Disclosure

Most of the Bitcoin exchanges require user's address and credentials as their policy for providing security or to cater legal issues. Scam exchanges can reveal personal information that can violate anonymity of transactions. Anonymity claims of Bitcoin transactions have been rejected by researchers by manipulating the block chain which is publically available to everyone in the Bitcoin network. Specialized services like blockchain.info (Blo, 2016) provide information regarding block chain and Bitcoin activity and can indicate the nodes which relay any transaction. The obtained IP addresses of the nodes can be manipulated to trace back the origin of the transaction. Block chain and transaction data can be analyzed to create transaction graphs that can identify the sender and receiver of a transaction. The highlighted areas indicate situations under which Bitcoin hosted wallet services can be attacked. Figure 4.4 shows the attack tree for information disclosure attacks on Bitcoin architecture.

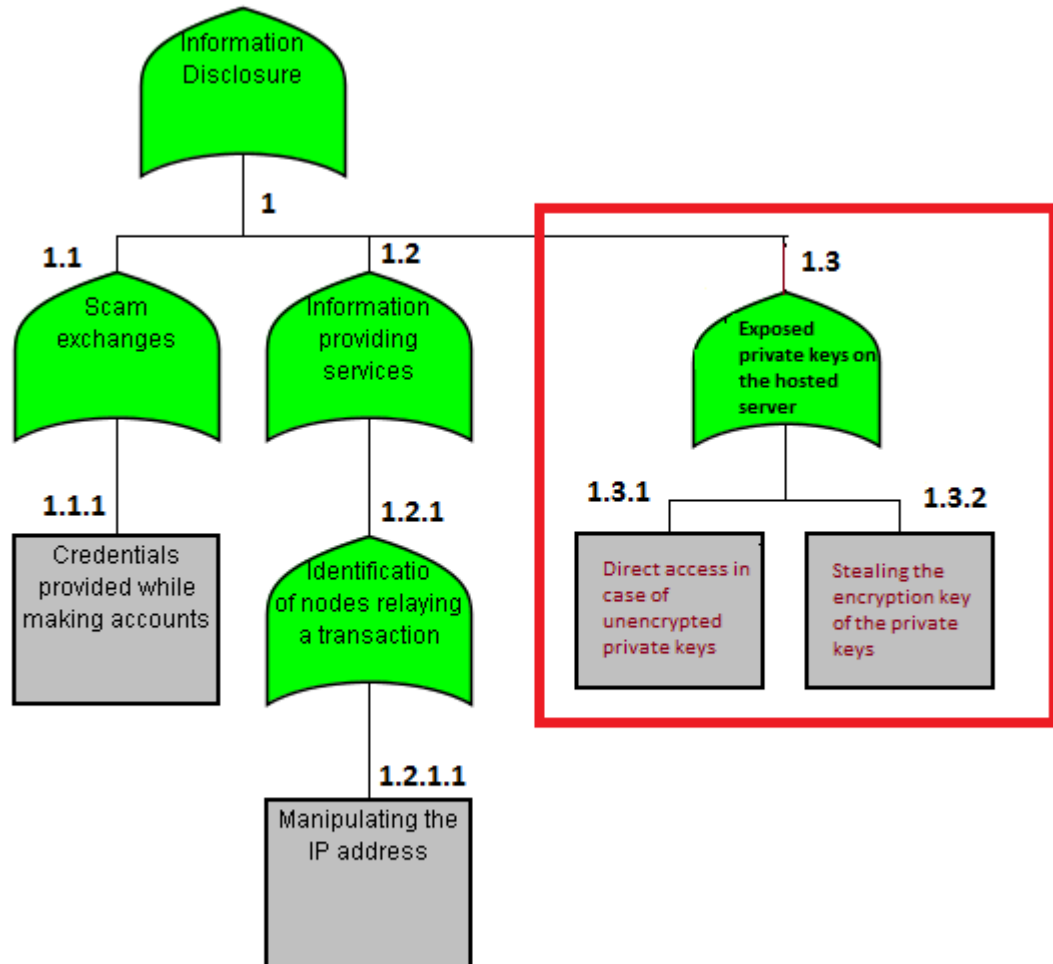


Figure 4.4: Attack Tree for Information Disclosure Attacks on Bitcoin protocol

4.5 Denial-of-service(DOS)

It is quite easy for any Bitcoin user to send transaction to back himself. The spammers can flood the network by sending transactions back to their wallets which can fill the storage capacity of a block i.e. 1MB. This causes DOS attacks resulting in delay in verification of other transactions.

Transaction malleability can temporarily cause DOS by disrupting the veri-

fication process. Modifying the transaction ID causes the sender to believe that the transaction is not sent, which in actual has been sent. This causes fog of confusion within the network, thus slowing it down. Figure 4.5 shows the attack tree for DOS attacks on Bitcoin architecture.

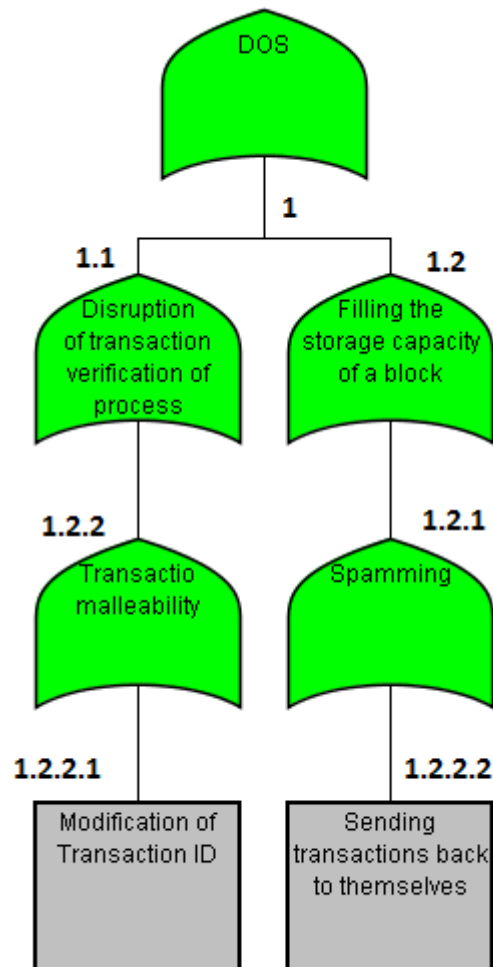


Figure 4.5: Attack Tree for DOS Attacks on Bitcoin protocol

4.6 Elevation of Privileges

Distributed Botnet attacks can cause an attacker to get privileged access to victims' computers by stealing the computational power. This computational

power, when combined, can be used by the attackers for mining process. An attacker can spoof the identity of a legitimate user by stealing the device on which Bitcoin wallet is installed. Figure 4.6 shows the attack tree for elevation of privileges attacks on Bitcoin architecture.

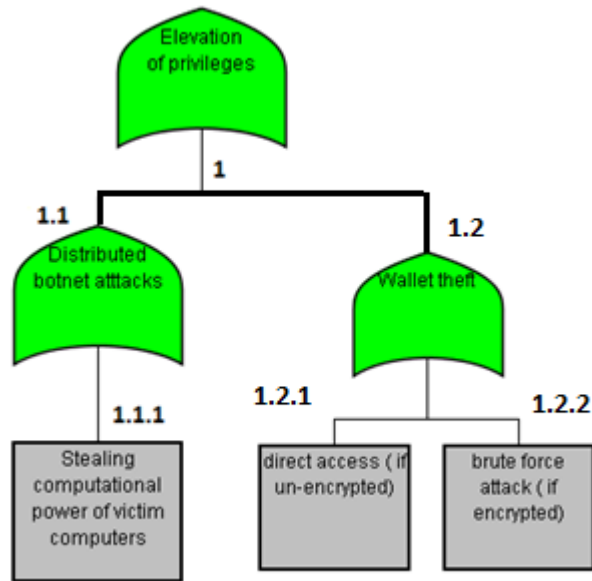


Figure 4.6: Attack Tree for Elevation of Privileges Attack on Bitcoin protocol

In this section, we have analyzed the possibility of different attacks based on the threats associated with different Bitcoin components. The threats, if not eliminated, may expose Bitcoin transactions to different attacks leading to the loss of bitcoins. The areas highlighted in the Spoofing and Information Disclosure attack trees, serve as our problem statement. In order to limit our research to a particular area, we have considered Bitcoin hosted wallets as our problematic areas which are prone to attacks like unauthorized access to private keys and man-in-the-middle attacks.

Chapter 5

AN OVERVIEW OF SECURITY OF WEB BASED BITCOIN WALLETS

Based on the threat modeling presented in chapter 4, we narrow down the scope of our thesis by addressing the Spoofing and Information Disclosure threats that pertain specifically to web based Bitcoin wallets. In this regard, the following chapter presents an overview of the security aspects of web based wallets as well as the related work that has been done in providing authentication and authorization while initiating Bitcoin transactions.

5.1 Security Of Web based Bitcoin Wallets

As discussed in chapter 2, the Bitcoin wallets can be considered as a digital equivalent of a bank account. They are responsible for generating a set of Bitcoin addresses and their corresponding private keys, that are necessary for sending, receiving and storing bitcoins. The security of the Bitcoin wallet is very critical due to the private keys involved which are necessary to initiate transactions using a set of Bitcoin addresses.

During the initial days of Bitcoin, the users had the only option of first installing Bitcoin client software called Bitcoin-QT(now known as Bitcoin core)(license, 2016) on their systems in order to become a part of the Bitcoin network and to be able to send or receive bitcoins. By doing so, the updated blockchain till that day was also downloaded along with the Bitcoin client. It comprises of a full node for validating a blockchain, as well as a Bitcoin wallet.

With the passage of time, the Bitcoin wallets moved from desktop PCs to mobile phones and web platforms. The biggest advantage of this shift is that the users do not have to download whole blockchain and can carry out Bitcoin transactions from anywhere around the world. While each of the types of the Bitcoin wallets provides its own advantages and disadvantages, the web wallets have attracted a large number of Bitcoin users who want to access their Bitcoin wallets from anywhere in the world without the hassle of downloading the blockchain, thus saving space and memory. Moreover, the users can instantly carry out transactions.

By choosing web wallets as hosted Bitcoin wallets, the users rely on a trusted third party to secure their bitcoins. Therefore, before choosing a web wallet, the users have to be sure about the credibility of the service provider. A large number of security breaches in web based wallets have been observed in the last few years, resulting in the loss of bitcoins.

5.2 Security of Local Wallets vs. Web based wallets

Not all of the local wallets have to download blockchain like in the case of Bitcoin-QT. In some cases, a local wallet connects to a server which has a copy of the blockchain. There are a lot of security threats associated with the local wallets. Some of them are discussed below.

- Either the wallet is not encrypted or the implemented encryption scheme is very weak and can be bypassed by the attacker.
- The media containing the wallet e.g. tablets, smart phones, ipads gets lost, stolen or damaged. In all of the cases, users will not be able to access their Bitcoin wallets.
- The Bitcoin users forget the password of their Bitcoin wallets and are somehow unable to reset the password. In these cases, the wallet.dat file can be decrypted and there is not a backward mechanism so that that the keys associated with that wallet could be accessed.
- The media containing the wallet gets compromised through some malware that can steal users' credentials to get access to their wallets or a key logger can also steal user's password in order to steal bitcoins.

The biggest reason why users prefer the web based wallets over local wallets is that there is no need to synchronize with the Bitcoin network. The users do not have to know about the inner workings of the software and there is no installation required. The users trust a trusted third party for the management and security of their Bitcoin wallets. This, however, leads to a number of security threats; few of them are mentioned below:

- The web wallet service becomes unavailable and users can not access their wallets. This can be due to the service downtime, any technical fault or due to DOS attack on the server by any attacker.
- The wallet gets deleted on the server either by the user or by the service (deliberately or unknowingly).
- The server gets hacked and the attacker carry out transactions on users' behalf. Either the attack gets undetected or when it is detected, the attacker has already stolen the bitcoins.

5.3 Related Work

There are different scenarios of dealing with the security of the web wallets. The foremost solution is to protect the wallet using a secret key. The security of the secret key is critical as an adversary can get access to the private keys of a user if the password is stolen. There are two ways of dealing with the security of the password. Either the password is created and stored at the third party service or it stays with the user and the service has no knowledge of it at all.

If the password is with the user, then its security relies on the user's security practices. If it is lost, then the service could not do any thing to retrieve it. If on the other hand the password is stored at the online service, the user has nothing to do with its security. He/She logs in to the service with another authentication method and then the password for the wallet is automatically retrieved when a transaction is carried out. The problem here occurs for the user when for some reason the service loses the password. This can be due to a server side hack or a technical error, etc. In this case, the user will lose access to his/her accounts and his/her Bitcoins.

Following are the schemes that have been proposed to cater the authentication and authorization issues faced by the Bitcoin wallets. These security schemes ensure that only the legitimate user will be able to initiate a Bitcoin transaction.

5.3.1 Naive Approach

The naive approach in the case of the security of the hosted web wallets is that the service asks the user an encryption password and encrypts the associated private keys with that password. The password is stored in a database as a reference key along with the user's credentials so that the password can be fetched every time the user requests the service to carry out the transactions.

Disadvantage:

From an adversary point of view, the passwords can be stolen and can easily be accessed in case of a server side hack. The attacker can decrypt the private keys and carry out transactions on user's behalf.

5.3.2 Securing wallets through Threshold signatures

A simple way for private keys protection in the Bitcoin wallets is through threshold cryptography. According to the technique (Goldfeder et al., 2014), the users can divide their private keys into random parts and locate each one on different devices like smartphone, PDAs or desktop computers. This scheme allows the users to spend bitcoins only when the threshold number of devices are active in order to reconstruct the keys.

Disadvantage:

In such a scenario, there can be a tradeoff between security and efficiency as certain number of devices must be active in order to make any transaction. Moreover, the security of each of the involved devices has to be ensured.

(Goldfeder et al., 2014) present a wallet security scheme that provides a joint control over a certain wallet. In order to prevent a single point of failure, where loss of a single key will result in the loss of the bitcoins, this scheme uses threshold cryptography for the security of private keys. Consider a collaborative environment where the approval of t employees is necessary to carry out a transaction. The scheme is based on (t, n) secret sharing scheme. The private keys of a single wallet are split across the platform of these t employees. Thus a business can implement joint control of a Bitcoin address by distributing shares of the private key among multiple employees' devices. In order to initiate a transaction, the key parts will be reconstructed to create a private key which will digitally sign the transaction in a normal way. The reconstructed signature is similar to a normal signature and the transaction is not different from a normal transaction.

Disadvantage:

The proposed scheme poses a number of security threats. The greater the number of people involved, the greater will be the risks. There is a need to secure the platforms that hold the parts of the private key. A compromise

of any of these systems will result in a hurdle in the creation of the final private key. And hence, a new key has to be generated every time one of the constituent keys is compromised. Moreover, the scheme also raises usability concerns as the transactions are carried out in an interactive environment and there is a need for all of the platforms/users to be active in order to carry out a transaction.

5.3.3 Securing wallets through Multi-signatures

A number of Bitcoin wallet services like BitGo(Bit, 2016), Copay(Cop, 2016), Armory(Arm, 2016) etc. use multi-signature schemes in order to secure Bitcoin transactions. The scheme ensures a distributed control over a Bitcoin transaction by requiring a certain number of digital signatures in order to initiate a transaction. Multi-signatures are implemented in such a way that in order to spend bitcoins, a transaction needs to be signed by x out of the t designated entities

Disadvantage:

The disadvantages in this scheme involve the increase in the transaction size due to additional signatures. This is a big concern in case of micro-payments where the amount to be sent is very small. The increase in the transaction size due to additional signatures may result in additional transaction fees which may exceed the amount to be sent in some cases.

Moreover, the scheme also raises usability concerns as the transactions are carried out in an interactive environment and there is a need for all of the platforms/users to be active in order to carry out a transaction. Also, The greater the number of people involved, the greater will be the risks.

Chapter 6

DESIGN AND METHODOLOGY

This chapter covers the implementation details of the proposed protocol prototype. We have used the idea of Shamir's secret splitting technique(Shamir, 1979) in our research to secure the private keys. The Secret splitting scheme provides high confidentiality by splitting secret among different entities. The chapter covers the design and modules of the implemented scheme as well as the formal evaluation of the designed architecture through the Scyther tool.

6.1 Design and Modules

Following section presents the architecture details of our proposed scheme.

6.1.1 Proposed Scheme for Securing Private Keys in Hosted Wallets

Figure 6.1 shows the detailed architecture diagram of our proposed scheme.

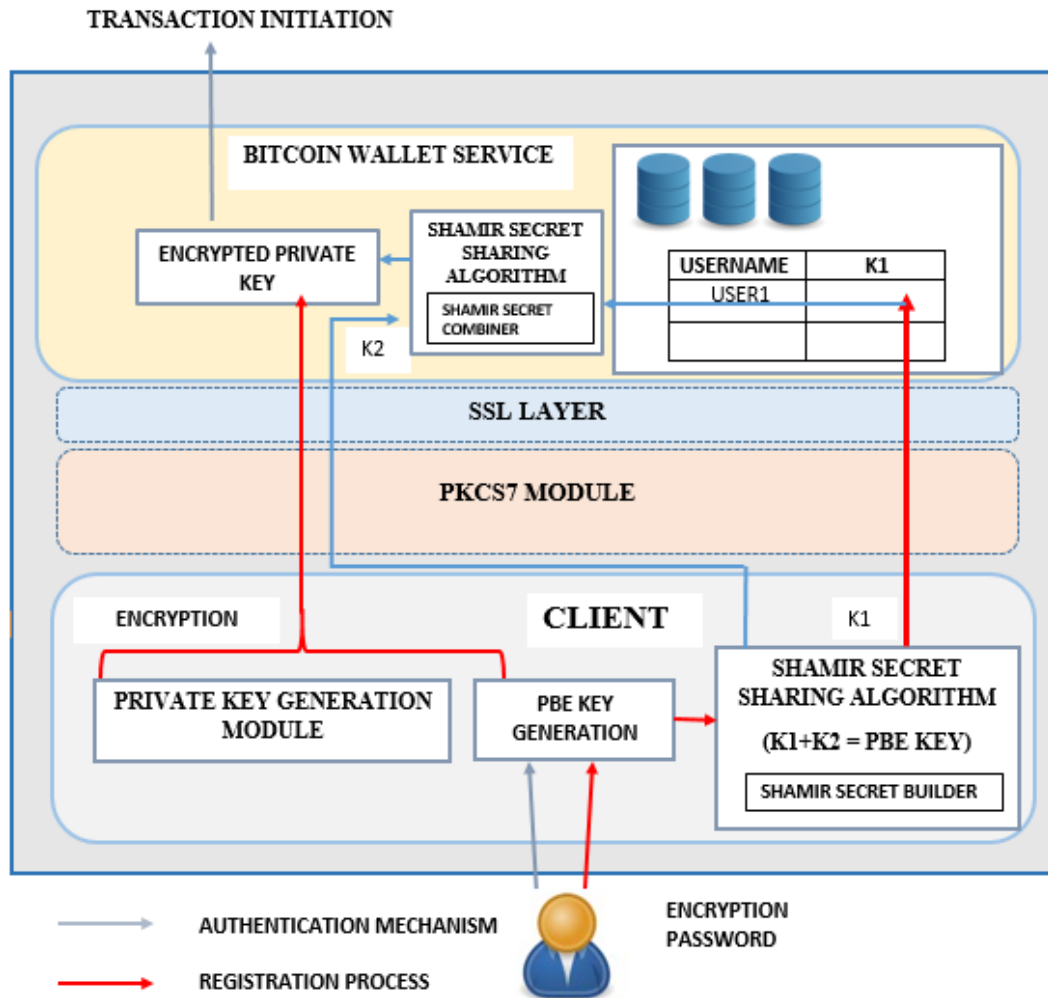


Figure 6.1: A Secure Scheme for the Protection of Hosted Bitcoin Wallets

Our proposed scheme consists of four main constituent modules, each of which are described below.

Private Key Generation Module

Unlike, most of the hosted web wallets schemes that generate private keys on their own and ask users for an encryption key, our scheme provides full control to the users over the private keys by generating them at the client end and sending it in an encrypted format so that the service provider can not access it. The private keys can be generated locally using browser based key generation schemes. For implementation purpose, we use Tomcat server for the host-

ing service provider and generated a RSA keypair using the *KeyGenerator* class of Java Cryptographic Extension(JCE)(JCE, 2004). The keysize is set to be 1024 bits. Listing illustrates the concept of RSA keypair generation. The server class functions are accessed through the proxy class.

PBE Key Generation

Password based encryption schemes have been used to encrypt and decrypt files using an easy to remember password. The PBE encryption algorithms generate a complex key from the user provided password. A random number called the salt is added to the password to avoid Brute force attacks. For implementation, we have used a *PasswordBasedEncryption* algorithm "AES" in order to encrypt the generated encoded private key. Any common algorithm supported by OpenSSL can be used as discussed in the OpenSSL documentation(Ope, 2016). The PBE encrypted private keys are securely transferred using PKCS7 encryption to the service provider.

Shamir Secret Sharing Algorithm

Our proposed scheme implements Shamir Secret Sharing Algorithm(Shamir, 1979) which divides a data D into n parts in such a way that n or k where $k < n$ parts are required to reconstruct D . The PBE algorithm generated encryption key is split into two parts at the client end and only one part $K2$ is sent to the server end. This server part is stored in a database corresponding to the user's credentials. The other part will only remain at the client end and will only be sent when a user wants to initiate a transaction. In order to avoid replay attacks, a nonce will be generated sent every time as a challenge and in response the client side will send the other part of the key along with the nonce. The Shamir Secret Sharing algorithm has been implemented in our protocol in the form of following two module.

1. Shamir Share Builder

The ShareBuilder class takes the number of shares and the threshold value for share combination set by the user, which in our case is two, and generates a list of components of the PBE key. The class uses Polynomial class which is used to generate a polynomial based on the number of shares, co-efficients and secure random numbers generated. The class returns a list of shares according to the share size set by the user. We have implemented the Shamir share builder module on the client end where PBE key will be split into two components, one for the user and other for the wallet service.

2. Shamir Share Combiner

The ShareCombiner class retrieves the list of shares and regenerates the PBE key. The number of shares that retrieved is equal to the threshold value set by the user, which in our case is two. The class uses the LagrangeInterpolator class in order to regenerate the PBE key. The Shamir Secret Combiner module has been implemented at the wallet service end where the PBE key will be regenerated by retrieving the Server's component and getting the User's component as input. The regenerated PBE key will decrypt the encrypted private key and hence the transaction could be initiated.

PKCS7 Module

Public Key Cryptography Standards (PKCS) are designed and published by RSA Security Inc. to promote the use of the cryptography techniques. PKCS7(Technologies, 2016) is used to sign and envelop a message before exposing it to the network in order to provide authentication and integrity. Hence, PKCS7 ensures that only authorized users can access the message being sent and no other entity can modify it. There are different JAVA based APIs that provide the functionality of signing and enveloping the data and BouncyCastle is one of them. PKCS7 has two modules a data signer and a data verifier. Data signer digitally signs the key component before travelling to the network. The signed text is wrapped in PKCS7 wrappers. Data verifier unwraps the message from PKCS7 wrapper.

6.1.2 High level Architecture of Proposed System

The high level architecture of proposed techniques is illustrated in figure 6.2.

```

Input:
    1. User entered password for PBE Encryption
    2. User Key Component

Output:
    1. Decrypted Private Key
    2. Concatenated PBE key (User Component and Server Component)

Begin:
If (Request=="Registration")
{
    Private Key generation;
    Getting the user input;
    PBE encryption;
    Key Splitting ();
    PKCS7 protocol ();
    Storage of Server's Component at the Service end;
}
Elseif (Request == "Transaction Initiation")
{
    PKCS7 protocol ();
    Collect the User's Component;
    Concatenate with the Server's Component;
    Decrypt Private Keys using the regenerated key;
}
DisplayOutput();
End

```

Figure 6.2: High Level Architecture Diagram for The Proposed Scheme

6.1.3 Implementation Environment

Following are the details regarding the tools and technologies that pertain to the development of the prototype.

Tools and Technologies Involved

In order to implement the designed scheme, a modular approach is followed using the following tools and technologies.

- Java Cryptographic Extension(JCE) for the password based Encryption(PBE)
- Eclipse Mars.2 for the software development

- BouncyCastle API for implementation of PKCS7 module
- Scyther tool for the evaluation of the protocol

6.2 Use Case Scenario

6.2.1 Registration Process

The user first registers himself/herself to the Bitcoin service and selects a user name and password. These credentials can be used to log in to his account. During the registration phase, the user creates a Bitcoin wallet in such a way that private keys are locally created at the client end. The user provides a password which is first converted into a complex cipher key using PBE encryption algorithm and is then used to encrypt the private keys which are securely transferred to the trusted third party service.

The cipher key created from the password provided by the user is split into two parts $K1$ and $K2$. One of the parts is sent to the service to be stored in a database corresponding to the corresponding username.

6.2.2 Authentication Mechanism

When a transaction request is made from the user end, the service first generates a nonce as a challenge and as a response, the split key pair along with the nonce is securely sent from the client to the Bitcoin wallet service. The service confirms the nonce, concatenates the key part with the other part and decrypts the private keys. Hence, the transaction is initiated. It should be noted that key is not stored anywhere on the server during the authentication mechanism. For better security practice, it is recommended to frequently change the encryption password.

PKCS7 (Technologies, 2016) protocol is used to wrap and unwrap data objects between transfer's entities for data confidentiality. Furthermore SSL is used to send and receive request between client and the wallet service to protect data from unauthorized used.

6.3 Evaluation

In order to formally verify our proposed protocol, we have used Scyther (Cremers, 2008) tool which is used for the formal analysis and evaluation of the security protocols. It is used to find security problems in the design of a protocol. Scyther works under the assumption that all of the cryptographic functions

are perfectly designed and are secure. For identification of a set of traces, Scyther uses pattern refinement algorithm. Scyther uses SPDL (Security Protocol Description Language), which defines different roles to describe a security protocol (Cremers, 2014). These roles are characterized by a sequence of events (sending and receiving of terms). Scyther takes an SPDL format file as input in which intended security properties are specified. These intended security properties are referred to as security claims and are evaluated by Scyther. For the validation of the designed protocol, an SPDL format file is run, after which the claims are evaluated and results are displayed. In case of incorrect claims, Scyther provides an option to view the attacks on the security claim.

6.3.1 Agents Used in Our Proposed Scheme

The agents serve as the interacting entities in the protocol specified by a behavior. We have defined two agents in our proposed design, each having a specific role. These agents will send and receive items with each others. Table 6.1 represents the agents defined in our proposed protocol.

Agent	Role
Client	This agent will perform the role of client.
WalletService	This agent will perform the role of a third party service hosting the wallet services.

Table 6.1: Agents used in our proposed protocol

6.3.2 Attributes Used in Our Proposed Scheme

Attributes define the atomic terms that are used in send and receive events of each of the defined roles. Table 6.2 represents the attributes defined in our proposed scheme

6.3.3 Security Claims and Evaluation Results of our Proposed Scheme

In order to model security properties of designed scheme, claim statements are used. Claims show what properties need to be evaluated in the de-

Attribute	Description
nC	Nonce of client.
tC	TimeStamp of client.
nWS	Nonce of hosted Wallet service.
tC	Time stamp pf the Wallet Service.
skey	Session key shared between the client and the Wallet service.
ServerPKCS7Comp	Part of the PBE encryption key stored at the Wallet service.
UserPKCS7Comp	Part of the PBE encryption key kept with the client.
EncryptedPrivKeyPKCS7Packet	The private key encrypted with the Password based encryption(PBE).

Table 6.2: Attribute used in our proposed protocol

sign. Following security claims have been used on client end in our proposed scheme.

- **Claim 1:** $\text{claim_c1}(\text{Client}, \text{Secret}, \text{ServerPKCS7Comp})$;
 The secrecy claims ensures the value is secret only to the communicating entities and not any third party. This claim evaluates the secrecy of ServerPKCS7Comp at the client end.
- **Claim 2:** $\text{claim_c2}(\text{Client}, \text{Secret}, \text{EncryptedPrivKeyPKCS7Packet})$;
 The secrecy claims ensures the value is secret only to the communicating entities and not any third party. This claim evaluates the secrecy of EncryptedPrivKey at the client end.
- **Claim 3:** $\text{claim_c3}(\text{Client}, \text{Secret}, \text{UserPKCS7Comp})$;
 The secrecy claims ensures the value is secret only to the communicating entities and not any third party. This claim evaluates the secrecy of UserPKCS7Comp at the client end.
- **Claim 4:** $\text{claim_c4}(\text{Client}, \text{Niagree})$;
 The non-injective agreement claim adds the condition that the two agents agree as to which roles each was taking, and that they agree upon some of the data items used in the exchange. This claim evaluates the Non-injective agreement at the client end.
- **Claim 5:** $\text{claim_c5}(\text{Client}, \text{Weakagree})$;
 Weak Agreement of one entity to another is guaranteed when one entity

completes a run of the protocol, apparently with the other entity, then the other entity has previously been running the protocol apparently with the first entity. This claim insists that one entity agreed that he was running the protocol with the other entity. This claim evaluates the Weak agreement property at the client end.

- **Claim 6:** `claim_c6(Client, Alive)`;
The aliveness or recentness claim between two entities ensures that when one entity completes a run of protocol apparently with another entity, then the other entity has also been running the same protocol. This claim evaluates the Aliveness at the client end.
- **Claim 7:** `claim_c7(Client, Nisynch)`;
The non-injective synchronization ensures that the send message has been completely received before the occurrence of the read event. This claim evaluates the Non-injective synchronization at the client end.

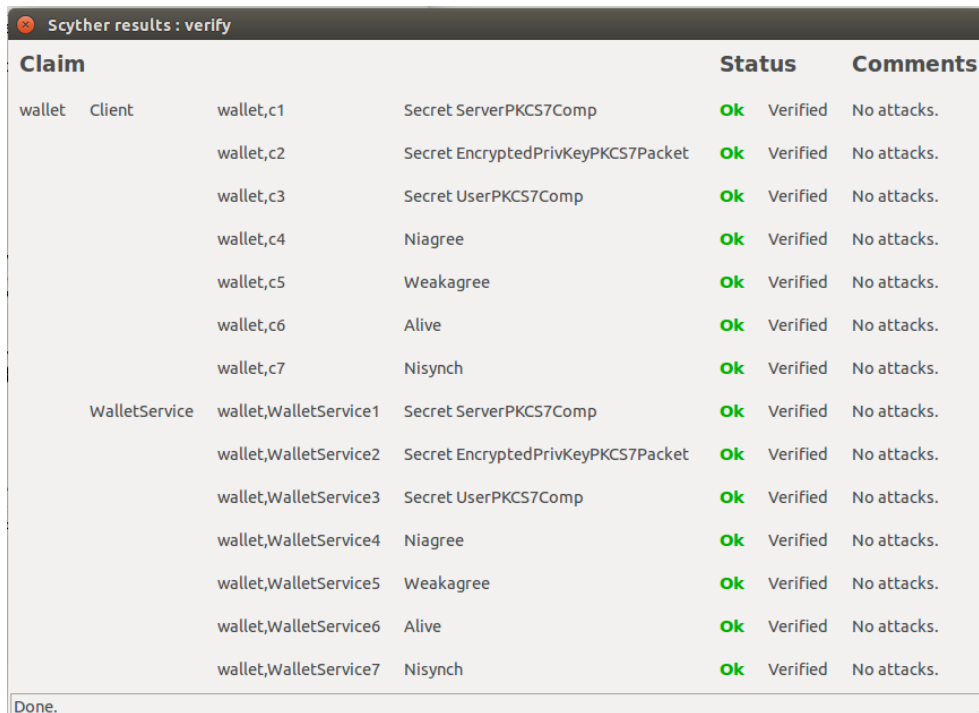
Following security claims have been used on Wallet Service end in our proposed scheme.

- **Claim 1:** `claim_c1(WalletService, Secret, ServerPKCS7Comp)`;
The secrecy claims ensures the value is secret only to the communicating entities and not any third party. This claim evaluates the secrecy of `ServerPKCS7Comp` at the `WalletService` end.
- **Claim 2:** `claim_c2(WalletService, Secret, EncryptedPrivKeyPKCS7Packet)`;
The secrecy claims ensures the value is secret only to the communicating entities and not any third party. This claim evaluates the secrecy of `EncryptedPrivKeyPKCS7Packet` at the `WalletService` end.
- **Claim 3:** `claim_c3(WalletService, Secret, UserPKCS7Comp)`;
The secrecy claims ensures the value is secret only to the communicating entities and not any third party. This claim evaluates the secrecy of `UserPKCS7Comp` at the `WalletService` end.
- **Claim 4:** `claim_c4(WalletService, Niagree)`;
The non-injective agreement claim adds the condition that the two agents agree as to which roles each was taking, and that they agree upon some of the data items used in the exchange. This claim evaluates the Non-injective agreement at the client end.
- **Claim 5:** `claim_c5(WalletService, Weakagree)`;
Weak Agreement of one entity to another is guaranteed when one entity

completes a run of the protocol, apparently with the other entity, then the other entity has previously been running the protocol apparently with the first entity. This claim insists that one entity agreed that he was running the protocol with the other entity. This claim evaluates the Weak agreement property at the client end.

- **Claim 6:** `claim_c6(WalletService, Alive);`
The aliveness claim between two entities ensures that when one entity completes a run of protocol apparently with another entity, then the other entity has also been running the same protocol. This claim evaluates the Aliveness at the client end.
- **Claim 7:** `claim_c7(WalletService, Nisynch);`
The non-injective synchronization ensures that the send message has been completely received before the occurrence of the read event. This claim evaluates the Non-injective synchronization at the client end.

The SPDL script for our proposed scheme has been provided in Appendix A. After running the spdl file of our designed protocol, figure 6.3 shows the evaluation results of our security claims.



Scyther results : verify				Status	Comments
Claim					
wallet	Client	wallet,c1	Secret ServerPKCS7Comp	Ok	Verified No attacks.
		wallet,c2	Secret EncryptedPrivKeyPKCS7Packet	Ok	Verified No attacks.
		wallet,c3	Secret UserPKCS7Comp	Ok	Verified No attacks.
		wallet,c4	Niagree	Ok	Verified No attacks.
		wallet,c5	Weakagree	Ok	Verified No attacks.
		wallet,c6	Alive	Ok	Verified No attacks.
		wallet,c7	Nisynch	Ok	Verified No attacks.
	WalletService	wallet,WalletService1	Secret ServerPKCS7Comp	Ok	Verified No attacks.
		wallet,WalletService2	Secret EncryptedPrivKeyPKCS7Packet	Ok	Verified No attacks.
		wallet,WalletService3	Secret UserPKCS7Comp	Ok	Verified No attacks.
		wallet,WalletService4	Niagree	Ok	Verified No attacks.
		wallet,WalletService5	Weakagree	Ok	Verified No attacks.
		wallet,WalletService6	Alive	Ok	Verified No attacks.
		wallet,WalletService7	Nisynch	Ok	Verified No attacks.

Done.

Figure 6.3: Evaluation Results of Our Proposed Scheme

The figure shows that our proposed scheme fulfills the claims of secrecy, aliveness, non-injective agreement, weak agreement and non-injective synchronization both at the client and the service end.

Chapter 7

CONCLUSION AND FUTURE WORK

This chapter presents the contribution summary of our research by providing an overview of the targets of research that we have achieved. It also provides a conclusion to our targeted research and future work directions. Future work includes the areas that are still unattended in our research work.

7.1 Contribution Summary

This thesis work targeted the area of Bitcoin security which is a hot topic now a days among the researchers. Bitcoin security is a vast field and catering the security of each of its components was out of the scope of this work. Therefore, we confined our research our work to the security of hosted Bitcoin wallets.

- In this work, we have studied the Bitcoin protocol from a security perspective and have highlighted various vulnerabilities that need to be addressed. We have presented a holistic survey of attacks that can take place during Bitcoin transactions. We have critically analyzed the Bitcoin architecture and carried out threat modeling to highlight all the possible weaknesses. Based on the possible threats on the Bitcoin components, we have created attack trees in order to present a clear picture of the attacks. Our work also mentions the possible countermeasures that can be adopted to make Bitcoin a safer protocol.
- A comparative analysis of Bitcoin and other crypto-currencies has also been presented by highlighting their specific features, advantages and

disadvantages and possible attacks. Based on our analysis, it is suggested that changes need to be made to the Bitcoin protocol by the consensus of open source community in order to overcome the weaknesses.

- We also present a secure and efficient scheme for addressing authentication issues in web based hosted Bitcoin wallets. We have used technologies like Java Cryptography Extension (JCE) classes, PKCS7, PBE encryption algorithm, Shamir Secret Sharing Algorithm and SSL in order to practically implement our proposed scheme. The proposed architecture ensures that no other entity will be able to access a hosted Bitcoin wallet even in the case of a server side security compromise. User's intervention is mandatory for every transaction initiation.

7.2 Conclusion

It goes without saying that digital currencies like Bitcoin are of growing interest to economists, cyber security experts, speculators and media. Never in history have people experienced a decentralized money transfer system that offered anonymous, unregulated transactions to be carried out with very less transaction fees. World's unbanked population finds no choice other than Bitcoin for sending money to other people in the world. On a larger scale, worldwide national adoption of decentralized and virtual currencies will result in economic neutrality and political transparency.

It cannot be said with surety that Bitcoin or any other crypto-currency will be able to replace real currency but there is a great desire for digital currencies. Human beings are considered to be the weakest link in the security chain, therefore there is a dire need of a decentralized and trustless system as it is better to trust on cryptographic operations than humans. A large number of credible researchers and entrepreneurs are investing their time and money in this domain. The advantages that digital currency systems provide over physical monetary systems cannot be overlooked. However, certain economic and security challenges act as a hurdle in world-wide adoption of virtual currencies. In order to ensure successful and world-wide adoption in future, security weaknesses in the protocol need to be dealt with. Moreover, price fluctuations are causing inconvenience for businesses accepting bitcoins as they have to adjust their prices accordingly.

In short, Bitcoin is evolving with progress in economic and technological con-

ditions and its future is quite vague. If Bitcoin community puts its efforts in resolving inherent security issues in its protocol, then Bitcoin surely has the capability of being a strong competitor of real currency.

7.3 Future Research Directions

After concluding this research work, it can be concluded that there are a number of areas that require future work. Few of them are as follows.

- The attack trees suggest a number of possible threats. The research could be further extended by considering other components of Bitcoin architecture and their relevant security issues. The possible attacks can be simulated to quantify the extent of damage they can cause and suitable countermeasures should be suggested.
- Different cryptographic mechanisms used within the Bitcoin protocol should be analyzed in order to check the possibilities of their compromise.
- There is a lack of privacy and trust management for Hosted Bitcoin wallet service. A future research can help in establishing a framework for security and privacy management for Bitcoin web wallet services.
- The client end applications like Bitcoin wallets are the most vulnerable entities and their security also depends on the security practices followed by the users. Identifying those security practices and then evaluating their strengths and weaknesses can further extend this research.
- A detailed comparative analysis of different cryptographic currencies in terms of their characteristics and how secure they are is required. A scheme for quantifying the usability of crypto-currencies over traditional currency systems is required. Based on the analysis, future of crypto-currencies can be predicted.

Appendix A

Evolution of Other Crypto-Currencies

Bitcoin became the pioneer crypto-currency by introducing the concept of decentralized peer-to-peer cryptography based digital currency. The idea of Bitcoin has paved the way for the creation of many other crypto-currencies which have been collectively termed as 'altcoins'. These altcoins are considered to be modified and improvised versions of Bitcoin but still lag behind in terms of acceptance, market capitalization and liquidity. Table A.1 highlights distinguishing features of Bitcoin along with top three altcoins with respect to market capitalization.

¹SCRYPT is slightly simpler algorithm as compared to SHA 256 which is less susceptible to ASICs (Application Specific Integration Circuits) designed for Bitcoin mining(Scr,). It is designed to make mining accessible to everyone without the requirement of computational resources)

²A hybrid of proof-of-work and proof-of-stake is used by Peercoin in the mining process. According to the design, proof-of-work is used only in the initial generation of coins but in the long term proof-of-stake would be used. Proof-of-stake reduces the chances of 51% attack by allowing coin generation based on miner's holding in the network

Crypto-currency	Features				Advantages	Disadvantages
	Hashing Algorithm	Mining Process	Current Mining Reward	Maximum Supply		
Bitcoin(Bit, 2013)	SHA 256	Proof-of-work	25 coins (halves after every 210,000 coins)	21 million	<ul style="list-style-type: none"> • Greatest market capitalization among crypto-currencies. • Widely accepted by businesses. 	<ul style="list-style-type: none"> • Requires extensive computational resources (ASICs) for mining (in comparison with Litecoin, CPU mining is not possible in Bitcoin). • Prone to various attacks. (51% attack, selfish mining, compromising anonymity).
Litecoin(Lit, 2016)	SCRYPT ¹	Proof-of-work	50 coins (halves after every 840,000 coins)	84million	<ul style="list-style-type: none"> • Greatest market capitalization among alt-coins. • Faster transaction confirmation. • Higher transaction volume in the blockchain. • Market entry costs are very low which allow anyone with a computer and internet to mine litecoins. 	<ul style="list-style-type: none"> • Lesser market acceptance. • Exposed to similar attacks as of bitcoin (51% attack, selfish mining, and compromising anonymity).
Peercoin(King and Nadal, 2012)	SHA 256	Proof-of-work and Proof-of-Stake ² (King and Nadal, 2012)	25 coins (halves after 16 times increase in the network) but as the reward decreases, reward will be proportional to the miner's stake in the currency.	No limit (designed to reach an annual 1% inflation rate)	<ul style="list-style-type: none"> • No limit on maximum supply. • Lesser chances of 51% attack/monopoly or due to proof-of-stake system (as new coins are generated based on the holding of the individual.) • Energy efficiency (lesser power consumption required for proof-of-stake as compared to proof-of-work that involves resource intensive hashing functions). 	<ul style="list-style-type: none"> • Lesser market acceptance. • Rich get richer. • People get rewarded for hoarding peercoins.
Namecoin(Kaloudis et al., 2015)	SHA 256	Proof-of-work	25 coins (halves after every 210,000 coins)	21million	<ul style="list-style-type: none"> • Same implementation as Bitcoin. • Can be used for money transfer as well as for storing information (DNS or identification/authorization) in the blockchain. • Provides decentralized DNS to prevent internet censorship. 	<ul style="list-style-type: none"> • Lesser market acceptance. • Users have to pay for network fees along with transaction fees. • Prone to attacks like 51% attack and compromising anonymity.

Table A.1: Table showing salient features of Bitcoin and other cryptocurrencies

Appendix B

Proposed Defense Mechanisms

Following section contains the description of the proposed defense methodologies for the attacks mentioned in chapter 3.

B.1 Solutions for Avoiding Double Spending

One possible way to deal with double spending is to introduce a trusted third party or a mint that keeps a record of all transactions and continuously checks each transaction for double spending. This defeats the decentralized behavior of Bitcoin architecture as the entire monetary system will be then dependent on the mint. The best possible countermeasure for double spending in current scenario is public announcements of all the transactions to all the nodes in the Bitcoin network.

The role of Bitcoin daemons is to continuously search for transactions which have similar input part and generate error messages. These errors are not propagated to the bitcoin users. The attacker can only be successful if the vendor ignores the flagged transactions processed by the bitcoin daemons.

As suggested by (Myt, 2016), one of the countermeasures to combat double spending in fast payment transactions is that the vendor V maintains a listening period before providing services to the customer²². During this listening period, the V can monitor each transaction to check if it is trying to double spend the amount he has previously received from the customer. One of the intuitions for this attack is that the vendor will receive TR_A and TR_V during the listening period. Connectivity of V acts as a security parameter as lesser the neighbors of V , more likely it is that peers will get TR_V before TR_A . Number of neighbors of vendor in a network depend on network churn

so V has to make sure that there are considerable number of helper neighbor nodes.

However, Karame et al.(Karame et al., 2012) have highlighted a flaw in this countermeasure. The attacker A can create a delay while sending TR_A in such a way that $t = tV_A - tV_V$ is more than the listening period and TR_A still spreads in the network, where tV_A is the time for TR_A to reach V and tV_V is the time for TR_V to reach V .

With the increase in the delay, majority of the neighbors of vendor V will have TR_V in their memory pools and when they receive TR_A , they will not add it in their memory pools and hence TR_A will not be forwarded to V . A has to ensure that enough peers accept TR_A so it can be incorporated in the block chain. This can be done by helper peers of the attacker.

Another countermeasure for double spending that can be used along with listening period is that V control an adequate number of 'observer node' in the same network which will forward all the transactions they receive, to V . In this way V can detect double spending if he himself or any observer receives TR_A .

B.2 Solutions for Avoiding Selfish Mining

It is suggested that in case of two equal length branches in a block chain, if half of the honest miners mine on pool's branch and others on the other branch, then selfish miners will require a considerable amount of extra computational resources to succeed. The following figureB.1 shows the relation between threshold (minimum power required by selfish pool to succeed) and γ .

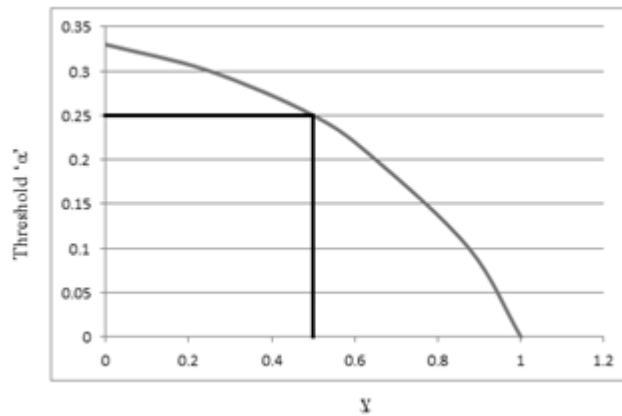


Figure B.1: Relationship between Threshold and Number of honest miners that choose to mine on pool's block

The Bitcoin protocol currently allows γ to be 1 and threshold of almost 0, i.e. selfish miners always succeed. To minimize the possibility of 'Selfish mining' attack, γ should be set to 0.5. That is, when honest miners learn of two equal length branches, half of them would mine on pool's branch while the other half would start mining on the other branch. This, in turn, yields the threshold to be 0.25 (which is 0 for $\gamma=1$).

This solution raises the threshold from 0 to 25%, by reducing γ from 100% to 50%. Now, the selfish miners will have to spend 25% additional computational power to get control of the Bitcoin mining process. This limitation decreases the selfish miner's ability to increase the revenue above their fair share (Eyal and Sirer, 2014) , (Springer, 2014).

B.3 Solutions for Attacks on Anonymity

The problem of anonymity can be solved by using a mixing service. A mixing service provides anonymity by mixing bitcoins of different senders together, and sending them to the destination using a different address. This strategy makes it harder to build the transaction graph and find links between previous transactions.

FigureB.1 shows the basic working of mixing services. The sender sends bitcoins to Address 'A' of mixing service. After the verification of transaction, the mixing service sends bitcoins to receiver using another address i.e. Address 'C', hiding the identity of sender.

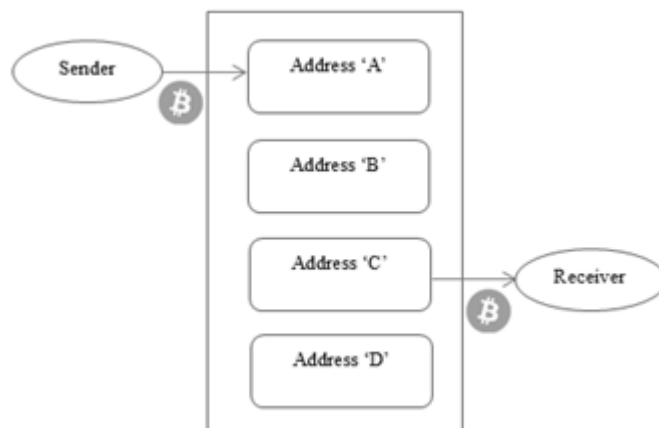


Figure B.2: Relationship between Threshold and Number of honest miners that choose to mine on pool's block.

There are a lot of mixing services available. The degree of anonymity that a mixing service provides can be calculated by comparing the information an attacker can get by analyzing the transaction, to maximum entropy. The information of the attacker can be obtained by analyzing the transaction. Formula for calculating the degree of anonymity d is shown in equation (B.1)

$$d = \frac{H(X)}{H_M}, \quad (\text{B.1})$$

where $H(X)$ is the entropy of the attacked system and H_M is the maximum entropy in the system. In addition to mixing services, another way to avoid links between transactions is to use a new address, every time a transaction has to be made. (Möser, 2013).

B.4 Solutions for Malware Attacks

The bitcoin protocol does not provide any countermeasure for malware attacks. The users themselves have to be careful to avoid these attacks. One of the most effective ways to avoid such attacks is to install effective antivirus and to download software from trusted sources.

A simple way for private keys protection in the Bitcoin wallets is through threshold cryptography. According to the technique, users can divide their private keys into random parts and locate each one on different devices like smartphone, PDAs or desktop computers. This allows the users to spend bitcoins only when the threshold number of devices are active. In such scenario there can be a tradeoff between security and efficiency as certain number of devices must be active in order to make any transaction.

Simon Barber et al. (Barber et al., 2012) suggest that better efficiency and availability can be achieved by slightly modifying the above scheme. User can have his own personal bank in the form of a super wallet having all his bitcoins. In addition to that, the user will have a sub-wallet installed on his smart phone in such a way that pre-approved transactions can be managed and bitcoins can be transferred from super wallet to the sub-wallet. This situation can be compared to original banks which let users draw cash through ATMs. Larger amount of bitcoins in the super wallet can be secured using threshold of devices. If an adversary gets hold of user's smartphone, then

only small amount of bitcoins will be lost.

Tobias Bamert et al. (Bamert et al., 2014) suggest a fast and secure alternative to credit cards in the form of BlueWallet, which also contributes to security of Bitcoin wallets. It serves as a hardware token that stores private key which is meant to digitally sign the transactions. BlueWallet delegates the task of creation of an unsigned transaction to another entity. Therefore, there is no need to connect it to the Bitcoin network. This prevents the BlueWallet from various attacks from the network. It serves as a storage place for private key which can only be accessed by the user who knows the valid PIN. Transactions can be created on a device which is directly connected to the Bitcoin network and BlueWallet can be used to sign such transactions. Bluetooth Low Energy communication is required for interaction between the two entities.

Table B.1 shows the threats/vulnerabilities in Bitcoin protocol and their possible countermeasures.

Threat/Vulnerability	Countermeasure
Double Spending	Vendor should maintain a listening period before provision of services and deploy some listener nodes that will forward the transactions they receive, to the vendor.
Selfish Mining	The protocol should enforce the policy in which, upon discovering two equal length branches, half the miners start mining on one branch, while the other half on the other branch.
Lack of Anonymity	Users of Bitcoin protocol should <ol style="list-style-type: none"> 1. Use mixing service 2. Use a new address, every time a transaction has to be made
Malware Attacks	<ol style="list-style-type: none"> 1. Keep antivirus database up-to-date. 2. Download software(applications) from a trusted source. 3. Verify the hash if downloading from mirror site.

Table B.1: Threats/Vulnerabilities in Bitcoin protocol, and their possible countermeasures

Appendix C

SPDL Script for Our Proposed Scheme

```
usertype SessionKey, TimeStamp, EncryptedPrivateKey, ServerKeyComponent,
  UserKeyComponent;

protocol wallet (Client, WalletService){

  role Client
  {
    fresh nC:Nonce;
    var nWS:Nonce;
    fresh tC:TimeStamp;
    var tWS:TimeStamp;
    hashfunction H;
    var skey:SessionKey;
    fresh ServerPKCS7Comp:ServerKeyComponent;
    fresh UserPKCS7Comp:UserKeyComponent;
    fresh EncryptedPrivKeyPKCS7Packet:EncryptedPrivateKey;

    send_1(Client, WalletService, (nC, tC, {H(nC,tC)}sk(Client)));
    recv_2(WalletService, Client, nWS, tWS, {H(nC,nWS,tWS)}sk(WalletService));
    recv_3(WalletService, Client, {skey, tWS, {H(skey,tWS)}sk(WalletService)
    }pk(Client) );
    send_4(Client, WalletService, {ServerPKCS7Comp}skey );
    send_5(Client, WalletService, {EncryptedPrivKeyPKCS7Packet}skey);
    send_6(Client, WalletService, {UserPKCS7Comp}skey );

    claim_c1(Client, Secret, ServerPKCS7Comp);
```

```

claim_c2(Client, Secret, EncryptedPrivKeyPKCS7Packet);
claim_c3(Client, Secret, UserPKCS7Comp);
claim_c4(Client, Niagree);
claim_c5(Client, Weakagree);
claim_c6(Client, Alive);
claim_c7(Client, Nisynch);
}

role WalletService
{
fresh nWS:Nonce;
var nC:Nonce;
fresh tWS:TimeStamp;
var tC:TimeStamp;
hashfunction H;
fresh skey:SessionKey;
var ServerPKCS7Comp:ServerKeyComponent;
var UserPKCS7Comp:UserKeyComponent;
var EncryptedPrivKeyPKCS7Packet:EncryptedPrivateKey;

recv_1(Client, WalletService, (nC,tC,{H(nC,tC)}sk(Client)));
send_2(WalletService, Client, nWS, tWS, {H(nC,nWS,tWS)}sk(WalletService));
send_3(WalletService, Client, {skey, tWS, {H(skey,tWS)}sk(WalletService)
}pk(Client));
recv_4(Client, WalletService, {ServerPKCS7Comp}skey);
recv_5(Client, WalletService, {EncryptedPrivKeyPKCS7Packet}skey);
recv_6(Client, WalletService, {UserPKCS7Comp}skey);

claim_c1(WalletService, Secret, ServerPKCS7Comp);
claim_c2(WalletService, Secret, EncryptedPrivKeyPKCS7Packet);
claim_c2(WalletService, Secret, UserPKCS7Comp);
claim_c4(WalletService, Niagree);
claim_c5(WalletService, Weakagree);
claim_c6(WalletService, Alive);
claim_c7(WalletService, Nisynch);

}
};

```

Bibliography

Scrypt.cc.

(2004). *Java™ Cryptography Extension (JCE) Reference Guide for the Java™ 2 Platform Standard Edition Development Kit (JDK) 5.0*. Oracle.

(2013). Bitcoin.

(2013). How cybercriminals are exploiting bitcoin and other virtual currencies.

(2016). Anonymity.

(2016). Armory secure wallet.

(2016). Bitgo.

(2016). Blockchain info.

(2016). Controlledsupply.

(2016). Copay.

(2016). Double-spending.

(2016). Litecoin.

(2016). Myths.

(2016). Openssl cryptography and ssl/tls toolkit.

(2016). Scalability.

Back, A. (2002). Hashcash - a denial of service counter-measure. Technical report.

Bamert, T., Decker, C., Wattenhofer, R., and Welten, S. (2014). Bluewallet: The secure bitcoin wallet. In *Security and Trust Management*, pages 65–80. Springer.

Barber, S., Boyen, X., Shi, E., and Uzun, E. (2012). Bitter to better how to make bitcoin a better currency. In *Financial cryptography and data security*, pages 399–414. Springer.

Blasco, J. (2013). How cybercriminals are exploiting bitcoin and other virtual currencies. [online].

CoinMarketCap (2016). Crypto-currency market capitalizations.

Cremers, C. J. (2008). The scyther tool: Verification, falsification, and analysis of security protocols. In *International Conference on Computer Aided Verification*, pages 414–418. Springer.

- Cremers, C. J. (2014). scyther-manual.
- Decker, C. and Wattenhofer, R. (2013). Information propagation in the bitcoin network. In *Peer-to-Peer Computing (P2P), 2013 IEEE Thirteenth International Conference on*, pages 1–10. IEEE.
- Drainville, D. (2012). An analysis of the bitcoin electronic cash system.
- Evans, D. S. (2014). Economic aspects of bitcoin and other decentralized public-ledger currency platforms. *University of Chicago Coase-Sandor Institute for Law & Economics Research Paper*, (685).
- Eyal, I. and Sirer, E. G. (2014). Majority is not enough: Bitcoin mining is vulnerable. In *Financial Cryptography and Data Security*, pages 436–454. Springer.
- from the arXiv, E. T. (2014). The troubling holes in mtgox’s account of how it lost \$600 million in bitcoins.
- Goldfeder, S., Bonneau, J., Felten, E. W., Kroll, J. A., and Narayanan, A. (2014). Securing bitcoin wallets via threshold signatures. *Princeton University*, [http://www.cs.princeton.edu/stevenag/bitcoin threshold signatures.pdf](http://www.cs.princeton.edu/stevenag/bitcoin%20threshold%20signatures.pdf).
- Gox, M. (n.d.). Mt. gox [online].
- Kalodner, H., Carlsten, M., Ellenbogen, P., Bonneau, J., and Narayanan, A. (2015). An empirical study of namecoin and lessons for decentralized namespace design. Technical report, Citeseer.
- Karame, G., Androulaki, E., and Capkun, S. (2012). Two bitcoins at the price of one? double-spending attacks on fast payments in bitcoin. *IACR Cryptology ePrint Archive*, 2012:248.
- King, S. and Nadal, S. (2012). Ppcoin: Peer-to-peer crypto-currency with proof-of-stake. *self-published paper*, August, 19.
- license, M. (2016). Bitcoin core.
- Limited, A. T. Attack trees.
- Love, D. (2014). 500 million worth of bitcoin has been stolen since 2010 [online].
- Mann, C. and Loebenberger, D. (2014). Realizing two-factor authentication for the bitcoin protocol. *IACR Cryptology ePrint Archive*, 2014:629.
- Microsoft (2005). The stride threat model.
- Möckel, C. and Abdallah, A. E. (2010). Threat modeling approaches and tools for securing architectural designs of an e-banking application. In *Information Assurance and Security (IAS), 2010 Sixth International Conference on*, pages 149–154. IEEE.
- Moore, T. and Christin, N. (2013). Beware the middleman: Empirical analysis of bitcoin-exchange risk. In *Financial cryptography and data security*, pages 25–33. Springer.
- Möser, M. (2013). Anonymity of bitcoin transactions: An analysis of mixing

- services. In *Proceedings of Münster Bitcoin Conference*.
- Nakamoto, S. (2008). Bitcoin: A peer-to-peer electronic cash system.
- Narayanan, A. (2013). Why the cornell paper on bitcoin mining is important.
- Perez, Y. B. (2015). Bitcoin atm industry: A look at the numbers.
- Perry, D. (2012). Bitcoin attacks in plain english.
- Research, C. (2000). Standards for efficient cryptography - sec 2: Recommended elliptic curve domain parameters. Technical report, Certicom Corp.
- Shamir, A. (1979). How to share a secret. *Communications of the ACM*, 22(11):612–613.
- Springer, M. (2014). Is bitcoin currently experiencing a selfish miner attack?
- Stevenson, J. (2013). *Bitcoins, litecoins, what coins? A global phenomenon*.
- Technologies, D. (2016). Pkcs7 cryptographic message syntax standard.
- Yelowitz, A. and Wilson, M. (2015). Characteristics of bitcoin users: an analysis of google search data. *Applied Economics Letters*, 22(13):1030–1036.