International Journal of Science and Research (IJSR)
ISSN: 2319-7064
SJIF (2022): 7.942

Effectuating Blockchain Network Compromise by Orchestrating a Daisy - Chain Architecture

Ujas Dhami[1], Nisarg Shah[2]

[1,2]Department of Computer Engineering, Silver Oak University, Ahmedabad - 380061, India

Corresponding Author EMail: *ujasdhami[at]gmail.com*

Abstract: ***As the surface of blockchains is increasing over the years, several vulnerabilities and attacks have been identified in the network, and several of those are currently being exploited without proper remediation from the providers. Circumstantially, the security of vendors and users is at stake from this. Investigation: This paper demonstrates a simulated daisy-chain attack on a victim who is a user of the ETH blockchain. The attack involves the participation of other exploitation techniques used to jeopardize the victim's wallet and insecurely transfer funds across the network. Method: The daisy-chain attack uses privilege escalation to first compromise the machine as an administrator, then misusing the privilege, installing multiple miners, and performing an eclipse attack into the blockchain. Later, transferring funds using a double-signing attack. Principle Result: The simulated attack was successful, and funds were transferred to another wallet through the simulated blockchain. The reverse shell remained persistent, and the ETW (Event Tracer for Windows) was disabled successfully so no traces could be found. However, a multitudinous amount of broadcast packets was needed to efficaciously perform the eclipse attack.***

Keywords: Blockchain, Daisy-Chain Attacks, Blockchain Security, Eclipse Attacks, Double Signing

1. Introduction

Blockchain is a decentralized network used to record the transaction of Bitcoin or any other cryptocurrencies. These transactions are maintained across several computers which communicate with each other in a P2P Network [1]. Anything of value can be tracked and traded on a blockchain network which reduces risk. Among different types of cryptocurrencies, Ethereum holds the second largest market capitalization. Ethereum is a decentralized open-source blockchain that validates each transaction done in it. Although holding a substantial market share, the blockchain itself is prone to several blockchain-based attacks and this research demonstrates some of them performed on a standalone system.

1.1 The Daisy-Chain Threshold

There are different types of vulnerabilities in the ETHblockchain. A couple of them can be produced through Daisy-Chain Attacks. A Daisy-Chain attack involves gaining unauthorized access to a network or a computer and then using the same network to access multiple networks or access points (APs).

Here in this research, we first built the Windows reverse ruby shell and then used it to access the Windows machine. After successfully exploiting the Windows machine, we then leveraged an eclipse attack on the blockchain.

An eclipse attack involves the creation of an artificial environment around the nodes within a P2P network. In this attack, the attacker redirects the connections from the legitimate node to the attacker nodes. These attacker nodes are completely different from the legitimate nodes. This attack can lead to double-signing of the TX 0block (also the genesis block) as it does not completely validate the transaction which leads to the encapsulation of the victim's signature by the attacker through forcefully making the node controller accept rogue digital signatures as a backup entity if the primary entity is not available.

1.2 Broadcasted Hash Flooding

The adversary here uses bots which flood the target node with a huge amount of hashes, which formulate the rogue network and connects the victim to the artificial network instead of connecting to the legitimate network.

After connecting to the artificial network by the victim, the attacker forces the victim's wallet to flush funds which is possible through several attacks as mentioned above.

2. Literature Review[2][3][4]

Title: Security and Privacy on Blockchain

Blockchain provides an innovative approach to storing information, executing transactions, perform functions, and building trust in an open environment. Many see the blockchain as a technological advance in cryptography and cybersecurity, from globally deployed cryptocurrency systems such as Bitcoin to smart contracts, smart grids, the Internet of Things, and more. There are use cases. Although blockchain has gained increasing interest in both academia and industry in recent years, blockchain security and privacy will continue to be the focus of debate when blockchain is deployed in a variety of applications.

This article provides a comprehensive overview of blockchain security and privacy. To facilitate the discussion, let's first introduce the term blockchain and its usefulness in the context of online transactions such as Bitcoin. It then describes the essential requirements of cryptocurrency systems like Bitcoin and the basic security properties supported as building blocks, followed by the additional

security and privacy properties required by many blockchain applications. Indicates.

Security and privacy techniques for achieving these security properties in blockchain-based systems, such as typical consensus algorithms, hash chain storage, merged protocols, anonymous signatures, and non-interactive zero-knowledge proofs. confirm. This research will help readers gain a deeper understanding of blockchain security and privacy in terms of concepts, attributes, techniques, and systems.

Title: A Survey on Blockchain for Information Systems Management and Security

Blockchain technology has gained a lot of attention in recent years, and many experts list potential applications of technology related to various aspects of the industry, market, government, or government. In the short history of blockchain, there have been an incredible number of achievements regarding how blockchain is used and its potential impact on various industries. The sheer number and complexity of these aspects can make it difficult to address the potential and complexity of blockchain, especially when trying to consider its purpose and suitability for a particular task. This study provides a comprehensive overview of blockchain applications as a service to today's information system applications. This research gives readers deeper insights into how blockchain can help protect and manage today's information systems.

This study covers a wide range of cases of blockchain research and applications proposed by the research community, their respective impacts on the blockchain, and their use in other applications and scenarios. The main findings of this study are that blockchain structures and the latest cloud and edge computing paradigms enable widespread adoption and development of new player blockchain technology in today's unprecedented and dynamic global market. Includes the fact that it is important to be. Making blockchain widely available through public and open-source code libraries and tools allows us to unlock the full potential of our technology and further develop it toward the long-term goals of blockchain enthusiasts.

Title: On the Security and Scalability of Bitcoin's Blockchain

Blockchain has proved to be an innovative tool that has been proven in many application scenarios. Many large companies such as IBM, Microsoft, Intel, and NEC are currently investing in the use of blockchain to enhance their product portfolio. Many researchers and practitioners speculate that blockchain technology may change the way we look at various online applications today. Sure, it's still early to say, but the blockchain is expected to make a big difference in a wide range of products and have a positive impact on the digital experience of many people around the world.

This article describes and evaluates some measures to thwart threats to the system. Some of them are already built into the system. Note that Bitcoin has been forked multiple times to optimize consensus (i.e. block generation time and hash function) and network parameters (e.g. block size). Therefore, the results reported in this tutorial are not limited to Bitcoin, but also apply to many "altcoins" that are clones/forks of Bitcoin source code. As alternative blockchain proposals increase, this tutorial aims to facilitate better design and analysis of next-generation secure blockchain currencies and technologies, from the Bitcoin system to basic security.

3. Methodology

The simulated attack surface was a Windows 10 21H1 standalone system which was vectored to perform commercial operations inside an enterprise network. The system was penetrable by delivering a malicious payload by using tools and spear-phishing techniques. Once the victim executed the payload, the system retained a reverse shell connected back to the attacker by the use of a ruby shell socket.

3.1 Privilege Escalation

The ruby socket opened up a connection to the attacker by using this particular command line:

```
ruby -rsocket –e
'c=TCPSocket.new("RHOST","9001");while(cmd=c.gets);IO.pope
n(cmd,"r"){|io|c.print io.read}end'
```

The given command initiates a new TCP socket, with the remote host being the IP address of the attacker, and the assigned port as 9001. Ruby uses the GETS method to read user input, which in this case would be the attacker’s commands. IO.POPEN will make Ruby run the designated command by the attacker as a subprocess and in the ‘read’ mode, and will print the output to the screen.

Now, for attaining SYSTEM privileges, the adversary will need to escalate the normal user’s privileges to SYSTEM [5], and for this, we used the MCC improper file permission assignment-based privilege escalation technique, which was successful.

3.2 The Eclipse Attack

After SYSTEM thread compromise, we now targeted the victim’s cryptocurrency wallet, by using techniques based on 51% attacks and improper signature validation control. One such attack to compromise the victim’s transactional network is by jeopardizing its network controller by forcing it to accept attacker’s malicious nodes’ hashes instead of the legitimate ones. This attack is called the eclipse attack.

We as adversaries used the SYSTEM privileges to establish several nodes based on the ETH blockchain into the same system. All the miner clones received unique hashes and then were ready to validate transactions. But here, the adversary intentionally configured nodes which had a high broadcast rate, so they could perform DoS into the network and force the network controller to accept themselves.

```
ERR_SLEEP = 15
MAX_NONCE = 1000000L
```

```
class ETHRPC:
        OBJID = 1
        def __init__(self, host, port, username, password):
                credpair = "%s:%s" % (username, password)
                oauth = "Basic %s" % (base64.b64encode(credpair))
                oconn = httplib.HTTPConnection(host, port, False, 30)
defbytereverse(x):
        return uniint32(( ((x) << 24) | (((x) << 8) & 0x00ff0000) |
                        (((x) >> 8)& 0x0000ff00) | ((x) >> 24) ))
```

This snippet demonstrates the configuration of a miner which will broadcast itself more than 24 times inside the network, technically creating a buffer between the legitimate nodes and the node controller. The controller will have to accept the nodes after restarting couple times.

Once we achieved the eclipse attack, we then had the ownership of the network, and now could begin the third phase of the compromise, which was the double-signing attack. Manipulating digital signatures can give any adversary access to a component, impersonated as a legitimate user.

3.3 Entity Double-Signing

Double-Signing occurs when the front-facing primary user's signature couldn't get validated for the transaction, and hence, the blockchain takes the backend signature for use [6]. The backend signature is easily replaceable by any particular signature which could then be signed by the victim's private key and then authorized as the signature of the user, which can be of an adversary and can cause significant harm to the victim's transaction.

The victim's wallet can be poisoned by using nodes which are improperly configured. Poor node implementation might allow multiple signatures to be accepted from the network operator in case of an attempt of signing as a legitimate user and transacting across the network as one.

In this scenario, the node's nonce validation was deteriorated, and so its validation of the multiple digital signatures passed on by the private key. The node implementation required functions which weren't configured properly but were very important for the determination of the right signature for use.

```
id = self.sender.id
        return collections.OrderedDict( { 'sender': id, 'recipient': self.recipient, 'value': self.value, 'time' : self.time } )
defsign_transaction( self ):
privkey = self.sender._privkey
pki = PKCS1_v1_5.new( privkey )
            result = SHA.new( str( self.to_dict( ) ).encode( 'utf8' ) )
self.pki = binascii.hexlify( signer.sign(result) ).decode( 'ascii' )
        return self.pki
defvalidate_transaction( self ):
        return
```

The self-signer functions above created an arbitrary signature which got passed by the victim's private key as a validation message, which then were used to perform functions only authorized to be used by the victim. This snippet triggered when the primary entity was not available to load its digital signature, and so the operator took the backup entity, which was controlled by the adversary.

Hence, after this, the victim's transactions were easily manipulated by making decisions through a rogue signature inside a dominant network controlled by us as adversaries.

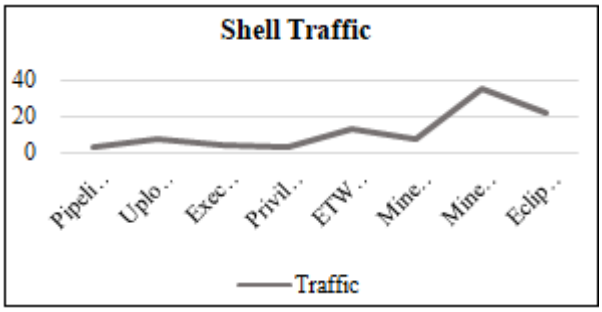
4. Outcomes and Analysis

After the compromise and when the eclipse attack was at peak, the compromised system transmitted a lot of SYN-bound packets requested from the Bloom Filter. However, with the compartmentalization of the network, the filter was forced to reset its previous node information with the rogue network's data by a series of Denial-of-Service (DoS) attempts.

4.1 Reverse Shell Persistence

Once the malicious vector was executed, the reverse TCP shell script was executed inside victim system. However, to bypass security defenses and access controls, the script was made to escalate local privileges to SYSTEM, hence, it used the exploit wired to the Microsoft Connected Cache (MCC), which allowed actors to gain the highest system privilege by exploiting improper file permission assignment.

Given below, is the traffic associated with the reverse shell in terms of persistence and error handling.



To exploit the surface, C:\Doinc\ was accessed and a PowerShell script was implanted. Upon execution, the web Administration module was replaced by a counterfeit, and imported by the SetDrivesToHealthy.ps1 script, providing NT\AUTHORITY\SYSTEM to the lower-privileged user.

However, after the privileges were jeopardized, ETW was disabled [7], and miner scripts were uploaded to the victim from the attacker's machine. Midway, several ETW permission errors were raised, which then were suppressed by restarting the MCC service.

4.2 Miner Counterfeiting

The ETH miners didn't require unique IP addresses, but only unique clients to get assigned an identity hash. This is a major advantage to the adversary, as he can clone and run various mining instances at once. Mitigation to this can be the usage of an IP address assigned to a miner, which could

be its own IP, so that malicious clones would not be formulated and node integrity would increase.

The miners got associated with different hashes under the same system, and the attacker executed the miners spontaneously. Since the machine was based on a cloud service, the miners could take the necessary resources to perform the eclipse attack and perform malfunctioned transactions with respect to rogue blockchain nodes created by these miners across the limited network the legitimate node was in. During the time of miner creation, the reverse TCP pipeline underwent a log of lag because of the high resource consumption.

```
char error[256] = "Failed.";
char persist[256] = "Persistence creation complete at location: \\HKEY_CURRENT_USER\\SOFTWARE\\Microsoft\\Windows\\CurrentVersion\\Run\n";
RegCKey(Var);
send(sock,persist, sizeof(persist),0);
return 0;
```

This is a snippet demonstrating the persistence of the system. The persistent backdoor was installed on the above mentioned location into the HKCU registry, hence, it wasn't killed by the lag happened under miner generation.

4.3Eclipse Attack

After the miners were set, the adversary now had a complete botnet to manipulate the victim's transactions right from TX 0. Eclipse attacks are possible when the network table is filled up with rogue IP addresses, which isolate the victim node from the rest of its legitimate peers.

4.3.1 Attack Formation

Achieving eclipse attacks is done through three steps of non-colliding synthesis done on the network table. To prioritize inserting the table with rogue hashes, the attacker first needs to perform a DoS attack onto it to force the Bloom Filter and the network to restart. This is possible with a shorter range of compromised miners, because the ETH network takes less than ten nodes inside a network to validate a transaction to the other side.

After the DoS is complete and the network restarts, the rogue nodes would have already started broadcasting SYN packets across the mainframe, hence, the network controller recognized the nodes and filled them inside the table. But, in many cases, this has to be done multiple times, because the network may switch very fast, concerning the location and availability of volunteers within a transaction.

The adversary may need to broadcast an abnormal amount of packets to get attained priority inside a network to be included in the table. This also works in the Simplified Byzantine Fault Tolerance (SBFT) parties, and the Proof of Stake (POS) protocol.

4.3.2 Leveraging Double-Signing

Alongside broadcast, Distributed Ledger Technology (DLT) can also be utilized to validate whether the attack works perfectly or not [8]. The adversary can also use double-signing to malfunction the POS consensus algorithm for validating blocks, as an additional leverage of improperly configured Public Key Infrastructure (PKI).

```
func (n *N) procSlCanMsg(msgPl []byte) {
if !node.IsRunningBeaconChain() {
return
}
can := slash.Records{}
if err := rlp.DecodeBytes(msgPl, &can); err != nil {
utils.Logger().Error().
Err(err).Msg("slash message decode error.")
return
}
if err := node.Blockchain().AddPendingSlashingCandidates(
candidates,
); err != nil {
utils.Logger().Error().
Err(err).Msg("slash candidates cant be added to pending ")
}}
```

Above is the code snippet for slashing a penalty to any user who performs double-signing on a blockchainnode, which was installed inside the victim's node configuration. This code was leveraged by the miners, as the entire network was affected by them, hence, they created rogue digital signatures to validate both the private keys which were passed onto the consensus nodes.

Because the nodes were improperly configured, the consensus algorithm was manipulated into accepting two private keys in one transaction, which were manipulated as two keys for separate isolated transactions.

4.4 Signature Manipulation

Many of the consensus algorithms use Elliptic Curve Cryptography (ECC) for validating digital signatures encoded in the same format. In this scenario, the attacker's ECC private key can be injected into the victim's TX node by assigning the rogue private key as a backup entity, supporting the primary entity which is the victim's signature, and then performing a DoS onto the primary entity, so that the backup entity gets invoked.

ECC secrets are Base64 strings carrying information about the transactions. Given below is an example.

```
AOvxP5+fZcyyDdrocqhPyk23d5ZICo0aTOKBAfY0TqjNweHivjl
2abSoehNYjiEbU/nj+g==
```

ECC private and public keys are also encoded in Base64 after an iteration of the EC cipher. The following would be the signatures of the attacker and the victim.

Attacker:

```
-----BEGIN PUBLIC KEY-----
MH4wEAYHKoZIzj0CAQYFK4EEACQDagAEAE88SPdOQi6p
A5ZL/Jkz6wzvwEwDVuGw
8KH70MFCa/TynV/RUoYovBYCjL9G2CqNKtjLbgGk64+R3S98
Q9KnGOuRYvdbx89i
yhRsSIzlIkfWaOHSCkvaYlWzkHabpML0vL/IuVfy4j0=
-----END PUBLIC KEY-----

-----BEGIN EC PRIVATE KEY-----
MIGvAgEBBDNU0fMni8qMS8o0uNTdr8NPyCsG2m1/XeM13X
```

MENfqY8BkYVHa9YZ5c
howrX9sa0y/8BDWgBwYFK4EEACShbANqAAQATzxI905CLq
kDlkv8mTPrDO/ATANW
4bDwofvQwUJr9PKdX9FShii8FgKMv0bYKo0q2MtuAaTrj5HdL
3xD0qcY65Fi91vH
z2LKFGxIjOUiR9Zo4dIKS9piVbOQdpukwvS8v8i5V/LiPQ==
-----END EC PRIVATE KEY-----

Victim:

-----BEGIN PUBLIC KEY-----
MH4wEAYHKoZIzj0CAQYFK4EEACQDagAEAA09M6ozLzak
3tPSE+VpO78+EQ4Yi4eh
/nyLHesWc7TC56/NsdsPnsv5CTamx4WkVib4CAEC+rQqQ6msa
xrJl/FOBJ3i5rLD
acyhPLmL9c7RaGtiVNHZ+mt+2hZ7lOzybPX+nitB2Z8=
-----END PUBLIC KEY-----

-----BEGIN EC PRIVATE KEY-----
MIGvAgEBBDNBFOI8oYHiSH3XUo+rDNesLQxd7R2KlyJGhR
EJnIiCdQs1Y/Hysi2F
IOBWfmqO9vPjr+mgBwYFK4EEACShbANqAAQADT0zqjMv
NqTe09IT5Wk7vz4RDhiL
h6H+fIsd6xZztMLnr82x2w+ey/kJNqbHhaRWJvgIAQL6tCpDqax
rGsmX8U4EneLm
ssNpzKE8uYv1ztFoa2JU0dn6a37aFnuU7PJs9f6eK0HZnw==
-----END EC PRIVATE KEY-----

Both were interchanged by using the above mentioned tactic of manipulation of the network operator and then the surface was leveraged to perform authenticated transactions across the network without the consent of user.

All of the attack execution was performed from the reverse ruby shell, which was persistent with no pipeline termination occurred in middle of any of the operations.

5. Conclusion

Several blockchains have not yet established a secure landscape of vectors which can successfully mitigate attacks related to the network or the consensus algorithm. Like the double-signing attack, improper nonce validation can also trigger double-spending attacks under other circumstances.

This can be mitigated by establishing strict validation rules and acceptance of miners inside a network shall be thoroughly monitored to determine whether they constitute elements within the guidelines of the network. However, if any type of Denial-of-Service attacks occur, the miners involved into the same shall be terminated and the operator shall be restarted to accept only legitimate outside connections. Effective switching and moderation can help mitigate the risk of eclipse attacks.

Furthermore, ruby shells are extremely fruitful on windows machines, since they undergo less anti-malware detections than all of their alternatives. The persistence of the shell is consistent and powerful, when it comes to uploading and downloading content.

References

[1] Donet, Joan & Pérez-Solà, Cristina & Herrera-Joancomartí, Jordi. (2014). The Bitcoin P2P Network. 8438. 10.1007/978-3-662-44774-1_7.
[2] Rui Zhang, RuiXue, and Ling Liu. 2019. Security and Privacy on Blockchain. ACM Comput. Surv. 52, 3, Article 51 (May 2020), 34 pages. https://doi.org/10.1145/3316481
[3] David Berdik, SafaOtoum, Nikolas Schmidt, Dylan Porter, YaserJararweh, A Survey on Blockchain for Information Systems Management and Security, Information Processing & Management, Volume 58, Issue 1, 2021, 102397, ISSN 0306-4573, https://doi.org/10.1016/j.ipm.2020.102397.
[4] GhassanKarame. 2016. On the Security and Scalability of Bitcoin's Blockchain. In Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security (CCS '16). Association for Computing Machinery, New York, NY, USA, 1861–1862. https://doi.org/10.1145/2976749.2976756
[5] Ul, Hidayat& Mohamad Zain, Jasni. (2018). WINDOWS PRIVILEGE ESCALATION THROUGH NETWORK BACKDOOR AND INFORMATION MINING USING USB HACKTOOL. MALAYSIAN JOURNAL OF COMPUTING. 3. 10.24191/mjoc.v3i1.4811.
[6] Tomasz Hyla, Jerzy Pejaś, Long-term verification of signatures based on a blockchain, Computers & Electrical Engineering, Volume 81, 2020, 106523, ISSN 0045-7906, https://doi.org/10.1016/j.compeleceng.2019.106523.
[7] Park, Insung. (2004). Event Tracing for Windows: Best Practices.. 565-574.
[8] Masood, Faraz&Faridi, A.. (2018). An Overview of Distributed Ledger Technology and its Applications. International Journal of Computer Sciences and Engineering. 6. 422-427. 10.26438/ijcse/v6i10.422427.