

# Unit 5 Notes

## Risks

- **Common Risks**
  - Blockchain operation mechanisms(Blockchain 1.0, Blockchain 2.0)
- **Specific Risks**
  - Development , deployment and execution of smart contract(Blockchain 2.0)

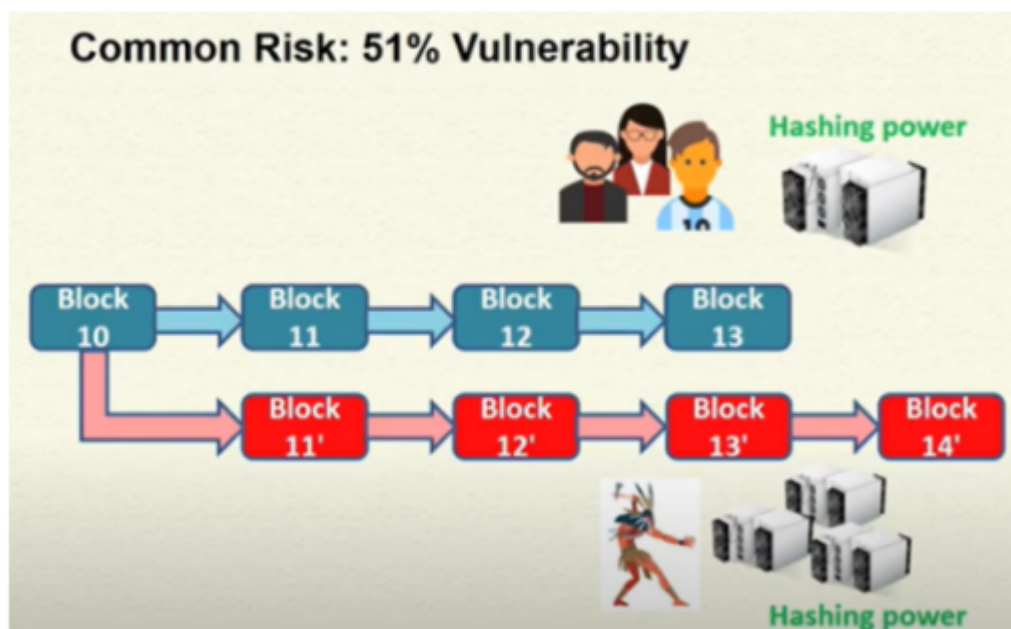
### Blockchain operation mechanisms Risks(Common Risks)

1. 51% vulnerability
2. Private key security
3. Criminal activity
4. Double spending
5. Transaction privacy leakage

### Smart Contraction execution risks(Specific Risks)

1. Criminal smart contract
2. Vulnerabilities in smart contract
3. Under-optimized smart contract
4. Under-priced operations

## RISK: 51% vulnerability



- Security vulnerability that arises when a **single entity or group gains control of more than 50% of the total computing power** (or hashing power) of a blockchain network.
- This is also known as a "**51% attack**" or "**majority attack**".
- 51% control means - they can manipulate the blockchain by **censoring transactions, double-spending coins, or altering transaction history**.
- Miners having more hashing power- more computational resources, means they will solve the problem faster- continue to win and add the blocks
- Non-malicious miners, will add to the longest chain( don't discuss, which chain they will add the block to and will **add to the attackers chain(red) without knowing** , solidifying the attackers chain)

## RISK: Private key security



- The private key is a critical component of blockchain security, as it is used to access and control ownership of digital assets, such as cryptocurrencies, stored in a blockchain wallet.
- Loss or theft of private key: owner may **permanently lose access to their digital assets**, as the private key is required to authorize transactions or transfers. This can result in **irreversible financial losses**.
- may include:
  1. **Phishing and social engineering attacks**- trick individuals into revealing their private key
  2. **Malware or keylogging attacks** -Malicious software or keyloggers can capture the private key when entered on a compromised device or network
  3. **Weak or insecure private key generation**- susceptible to brute force or guessing attacks
  4. **Insider threats** - Internal actors with access to private keys, such as employees or service providers

## RISK: Criminal Risk



## Ransomware

- They encrypt victims system data, and asks for money(in cryptocurrency), in exchange for decryption key

## Underground Market

- AKA darknet markets
- illicit online marketplaces that operate on encrypted networks and are often associated with illegal activities such as the sale of drugs, stolen data, counterfeit goods, weapons, and hacking tools.
- HIDE from government, do transactions undercount- unaccounted for

## Money Laundering

- Process of disguising the illicit origin of funds to make them appear legitimate
- Post the money on a blockchain, and add some transactions, no one will know
- Where unaccounted money, becoming accounted

## RISK: Double Spending

# Common Risk: Double Spending

1.  $TX_v$  is added to the wallet of the targeted vendor

2.  $TX_a$  is mined as valid into the blockchain

3. The attacker gets  $TX_a$ 's output before the vendor detects misbehavior

Attacker

Transaction to vendor  $TX_v$

Transaction to colluding address  $TX_a$

Bitcoin Network

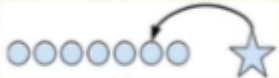
Vendor

Mining Pool

- ## RISK: Transaction Privacy Leakage

## Common Risk: Transaction Privacy Leakage

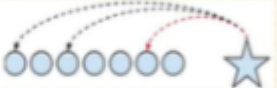
A new transaction (the star) which spends an available coin (the second circle from the right)



The diagram shows a horizontal row of seven blue circles representing coins. A solid black arrow originates from the second circle from the right and points to a blue star on the right, representing a transaction. This illustrates a direct linkage between the coin and the transaction.

Transactions and tracing in **Bitcoin**

✓ Each transaction input explicitly identifies the coin being spent, thus forming a linkage graph



The diagram shows a horizontal row of seven blue circles representing coins. A solid black arrow originates from the second circle from the right and points to a blue star on the right, representing a transaction. Additionally, several dashed lines (one red, others grey) originate from the same circle and point to other circles in the row, representing chaff coins (mixins) that are indistinguishable from the real coin.

Transactions and tracing in **Cryptonote**

✓ Each transaction input identifies a set of coins, including the real coin along with several chaff coins called "mixins."

- ✓ Many mixins can be ruled out by deduction
- ✓ The real input is usually the "newest" one

- 

- Many mixins can be ruled out by deduction
- The real input is usually the "newest one"
- Cryptonote - add fake transactions with the original transaction
- So if they want to know the status of the transaction like how much bob has now- They will get multiple states of the transaction
- In Cryptonote, **mixins refer to additional decoy inputs** used in a transaction to **obfuscate the true sender's identity**.

### Drawbacks

- how will the correct miner, know if a coin is fake or not
- unnecessary processing of transactions
- haven't fully launched this tech

## BLOCKCHAIN NETWORK ATTACKS SURFACES

- Blockchain network
- User wallets
- Smart contract
- Transaction Verification Mechanism( initiated a transaction, if you can attack when the transaction is being flooded in the network)
- Mining pool

### Cybersecurity Threats and Incidents on Blockchain Network

Identified 65 real-world cybersecurity incidents occurred between 2011 and first half-year 2019 that have adversely impacted blockchain systems.

The blockchain **cybersecurity vulnerabilities** are divided into **five categories**

- Clients' Vulnerabilities
- Consensus Mechanisms Vulnerabilities
- Mining Pool Vulnerabilities
- Network Vulnerabilities
- Smart Contract Vulnerabilities

## ATTACKS ON BLOCKCHAIN

1. Distributed denial of service
2. Transaction malleability attacks
3. Time jacking
4. Routing attacks
5. Sybil attacks
6. Eclipse attacks

## 7. Long range attacks on proof of stake networks

### 1. DISTRIBUTED DENIAL OF SERVICE

- Hard to execute on a blockchain network, but they're possible.
- Hackers intend to **bring down a server by consuming all its processing resources with numerous requests**.
- DDoS attackers aim to disconnect a network's mining pools, e-wallets, crypto exchanges, and other financial services.
- A blockchain can also be hacked with DDoS at its application layer using **DDoS botnets**.
- In **2017, Bitfinex** suffered from a massive DDoS attack. IOTA Foundation, launched their IOTA token on the platform the day before Bitfinex informed users about the attack.
- Three years later, in February 2020, Bitfinex experienced another DDoS attack just a day after the **OKEx cryptocurrency** exchange noticed a similar attack.

#### Why is DDos difficult in blockchain?

There is a transaction fee. All the fake messages/request will cost transaction fee

### 2. TRANSACTION MALLEABILITY ATTACKS

- A transaction malleability attack is **intended to trick the victim into paying twice**.
- If attackers manage to alter a transaction's ID, they can try to broadcast the transaction with a **changed hash** to the network and have it **confirmed before the original transaction**.
- If this succeeds, the **sender will believe the initial transaction has failed**, while the funds will still be withdrawn from the sender's account.
- If the sender repeats the transaction, the same amount will be debited twice. This hack is successful once the two transactions are confirmed by miners.
- **Mt. Gox, a Bitcoin exchange, went bankrupt** as the result of a malleability attack in 2014.
- However, Bitcoin seems to have solved this issue by introducing the **Segregated Witness (SegWit)** process, which separates signature data from Bitcoin transactions and replaces it with a **non-malleable hash** commitment to each signature.

### 3.TIME JACKING

- Time jacking exploits a theoretical vulnerability in Bitcoin timestamp handling.
- During a time jacking attack, a **hacker alters the network time counter of the node** and forces the node to accept an alternative blockchain.
- an attacker purposefully manipulates the timestamp of a block to make it appear as if it was **created earlier or later than its actual creation time**. This manipulation can disrupt the normal functioning of the blockchain by creating **confusion about the true chronological order of blocks**.



- This can be achieved when a malicious user adds multiple fake peers to the network with inaccurate timestamps.
- However, a time jacking attack can be prevented by restricting acceptance time ranges or using the node's system time.

## 4. ROUTING ATTACKS

A routing attack can impact both individual nodes and the whole network.

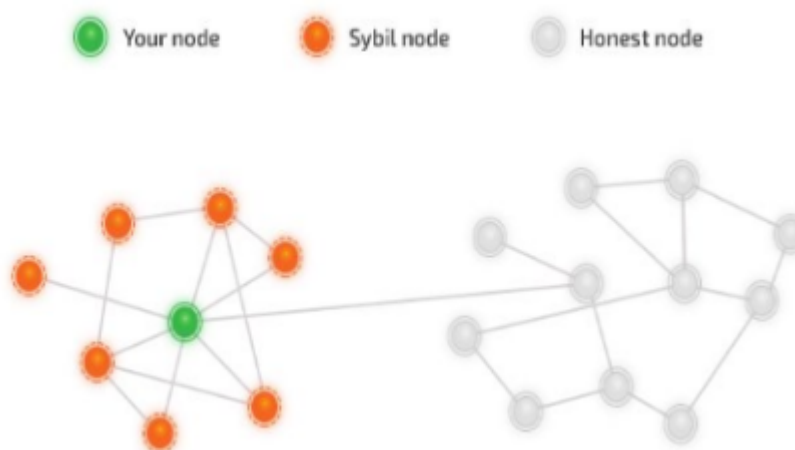
- The idea of this hack is to **tamper with transactions before pushing them to peers.**
- It's nearly impossible for other nodes to detect this tampering, as the **hacker divides the network into partitions** that are **unable to communicate** with each other.

Routing attacks consist of two separate attacks:

- **A partition attack, which divides the network nodes into separate groups**
- **A delay attack, which tampers with propagating messages and sends them to the network.**

## SYBIL ATTACKS

- A Sybil attack is arranged by **assigning several identifiers to the same node.** Blockchain networks have no trusted nodes, and every request is sent to several nodes.
- During a Sybil attack, a **hacker takes control of multiple nodes in the network.** **Then the victim is surrounded by fake nodes that close up all their transactions.**
- Finally, the victim becomes open to **double-spending attacks.**
- A Sybil attack is quite difficult to detect and prevent, but the following measures can be effective:
  - Increasing the cost of creating a new identity
  - Requiring some type of trust for joining the network
  - Determining user power based on reputation.



## ECLIPSE ATTACKS

- Hacker to **control a large number of IP addresses** or to have a **distributed botnet**.
- Then the attacker **overwrites the addresses in the “tried” table** of the **victim node** and waits until the victim node is **restarted**.
- **After restarting, all outgoing connections of the victim node will be redirected to the IP addresses controlled by the attacker.**
- This makes the victim unable to obtain transactions they’re interested in.
- Researchers from **Boston University** initiated an eclipse attack on the Ethereum network and managed to do it using just one or two machines.

## LONG RANGE ATTACKS ON PROOF OF STAKE NETWORKS

- Use the proof of stake (PoS) consensus algorithm, in which **users can mine or validate block transactions according to how many coins they hold**.

These attacks can be categorized into three types:

- **Simple** - A naive implementation of the proof of stake protocol, when nodes don’t check block timestamps
- **Posterior corruption** - An attempt to mint more blocks than the main chain in a given time frame
- **Stake bleeding** - Copying a transaction from the honestly maintained blockchain to a private blockchain maintained by the attacker

When conducting a long-range attack, a hacker **uses a purchased or stolen private key of a sizable token balance** that has **already been used** for validating in the past. Then, the hacker can **generate an alternative history** of the blockchain and increase rewards based on PoS validation.

## USER WALLET ATTACKS

User wallet credentials are the main target for cybercriminals.

- Phishing
- Dictionary attacks
- Vulnerable signatures
- Flawed key generation

## SMART CONTRACT ATTACKS

The main blockchain security issues associated with smart contracts relate to

- Bugs in source code
- A network’s virtual machine
- Runtime environment for smart contracts
- The blockchain itself.



## Attack vectors on the smart contract:

- Vulnerabilities in contract source code
- Vulnerabilities in virtual machines

## VULNERABILITIES IN CONTRACT SOURCE CODE

- Poses a risk to parties that sign the contract
- Bugs in Ethereum contract cost owners **\$80 million in 2016**
- Errors in Solidity → opens up a possibility to **delegate control to untrusted functions from other smart contracts**, known as **reentrancy attack**
- Contract A calls a function from contract B that has an undefined behavior
- In turn, B can call a function in contract A and use it for malicious purpose

## VULNERABILITIES IN VIRTUAL MACHINES

The most common vulnerabilities of the EVM are the following:

- **Immutable defects** : .If a smart contract contains any bugs in its code, they also are impossible to fix because the blockchain is immutable. There's a risk that cybercriminals can discover and exploit code vulnerabilities to steal Ether or create a new fork, as happened with the DAO attack.
- **Cryptocurrency lost in transfer** : This is possible if Ether is transferred to an orphaned address that doesn't have any owner or contract.
- **Bugs in access control** : There's a missed modifier bug in Ethereum smart contracts that allows a hacker to get access to sensitive functionality in a contract.
- **Short address attack** : This is possible because the EVM can accept incorrectly padded arguments. Hackers can exploit this vulnerability by sending specifically crafted addresses to potential victims. For instance, during a successful attack on the Coindash ICO in 2017, a modification to the Coindash. Ethereum address made victims send their Ether to the hacker's address.

## ATTACKS ON SMART CONTRACTS

### Attacks that can be used to exploit smart contract vulnerabilities

- Front-running aka transaction-ordering dependence
- DoS with block gas limit
- Block stuffing
- DoS with (unexpected) revert
- Forcibly sending Ether to a contract
- Insufficient gas griefing
- Reentrancy: Single-function reentrancy & Cross-function reentrancy

## 1. Front-running aka transaction-ordering dependence

The University of Concordia considers front-running to be “**a course of action where an entity benefits from prior access to privileged market information about upcoming transactions and trades.**”

This knowledge of future events in a market can lead to exploitation.

- **knowing a very large purchase of a specific token is going to occur**, a bad actor can purchase that token in advance and sell the token for a profit when the oversized buy order increases the price.

## 2. DoS with block gas limit

In the Ethereum blockchain, all blocks have a gas limit.

- One of the benefits of a block gas limit is it **prevents attackers from creating an infinite transaction loop, and if the gas usage of a transaction exceeds this limit, the transaction will fail.**

## 3. Unbounded operations

A situation in which the block gas limit can be an issue is in **sending funds to an array of addresses.**

Even without any malicious intent, this can easily go wrong. Just by having too large sized, **an array of users to pay can max out the gas limit and prevent the transaction from ever succeeding.**

- the transaction can be blocked indefinitely, possibly even preventing further transactions from going through.

## 4. Block stuffing

In some situations, your contract can be attacked with a block gas limit even if you don't loop through an array of unspecified length.

- An **attacker can fill several blocks** before a transaction can be processed by using a sufficiently **high gas price.**
- prevent other transactions from being processed.
- increase gas price and increase transactions

Ethereum transactions require the sender to pay gas to disincentivize spam attacks.

- Eg. a block stuffing attack was used on a **gambling Dapp, Fomo3D.**

The app had a countdown timer, and users could win a jackpot by being the last to purchase a key—except **every time a user bought a key, the timer would be extended.** An attacker bought a key

then **stuffed the next 13 blocks in a row** so they could win the jackpot.

## 5. DoS with (unexpected) revert

- DoS (denial-of-service) attacks can occur in functions **when you try to send funds to a user and the functionality relies on that fund transfer being successful**.
- This can be problematic in the case that the funds are sent to a smart contract created by a **bad actor**, since they can **simply create a fallback function that reverts all payments**.

```
//Art NFT
A (wallet) --→$10
B(smart contract)---→$11
C(wallet)--→$12
//unsuccessful (wrong fall back function)
```

## 6. Forcibly sending Ether to a contract

- Occasionally, it's **unwanted for users to be able to send Ether to a smart contract**.
- Unfortunately for these circumstances, it's possible to **bypass a contract fallback function** and forcibly send Ether.

## 7. Insufficient gas grieving

- **Griefing** is a type of attack often performed in video games, where a malicious user plays a **game in an unintended way to bother other players, aka trolling**. This type of attack is also used to prevent transactions from being performed as intended.
- This attack can be done on contracts which accept data and use it in a subcall on another contract. This method is often used in **multisignature wallets as well as transaction relayers**.
- If the subcall fails, either the whole transaction is reverted, or execution is continued.

## 8. Reentrancy

- Reentrancy is an attack that can occur when a bug in a contract function can **allow a function interaction to proceed multiple times when it should otherwise be prohibited**.
- This can be used to **drain funds from a smart contract** if used maliciously. In fact, reentrancy was the attack vector used in the DAO hack.

### Single -function reentrancy

A single-function reentrancy attack occurs when a **vulnerable function is the same function an attacker is trying to recursively call**.

### Cross-function reentrancy

A cross-function reentrancy attack is a more complex version of the same process. Cross-function reentrancy occurs when a **vulnerable function shares a state with a function an attacker**

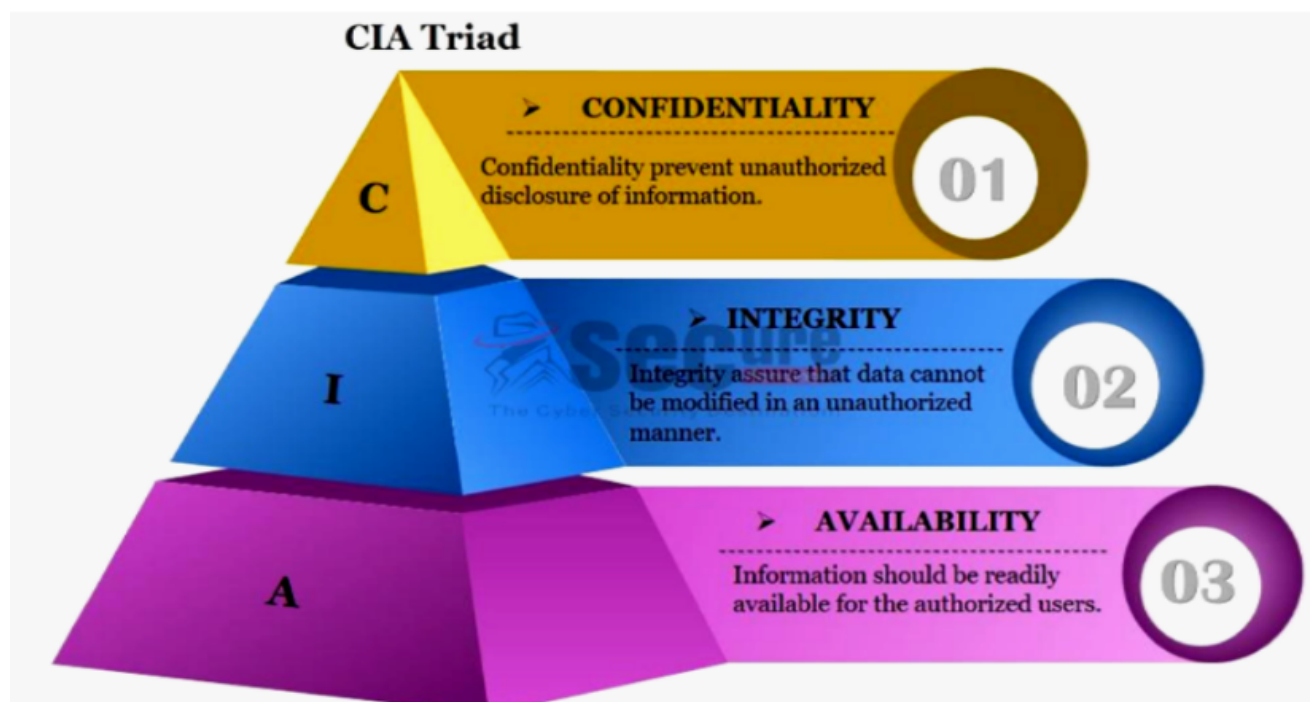
can exploit.

## VULNERABILITIES

- Integer overflow and underflow
- Timestamp dependence
- Outdated compiler version
- Unchecked call-return value
- Unprotected Ether withdrawal
- Unprotected self destruct instruction
- State variable default visibility
- Uninitialized storage pointer
- Assert violation
- Use of deprecated Functions
- Delegate call to untrusted callee
- Signature malleability

## BLOCKCHAIN ON CIA SECURITY TRIAD

**Confidentiality, Integrity, and Availability (CIA)** security triad model is one of the oldest and most popular security frameworks connected with the blockchain structure. **The CIA triad model is a model that helps organizations structures their security posture.**



## Confidentiality

Confidentiality is a way to **keep information hidden from unauthorized people**.

However, because of the open and Permissionless nature of the public blockchain such as Bitcoin, achieving a better confidentiality rank can be extremely difficult.

## Integrity

Integrity is a way to **protect the unauthorized tampering of information**. It is a mandatory compliance for every infosec body. It is also a method to maintain the consistency, accuracy, and trustworthiness of the respective data over its entire life cycle.

- Certain measures of prevention include file permission and user access controls.

## Availability

Availability refers to **on-time and reliable access to data**. The path of going from data to information and information to value means that the value will be illegitimate if the information is not available at the right time.

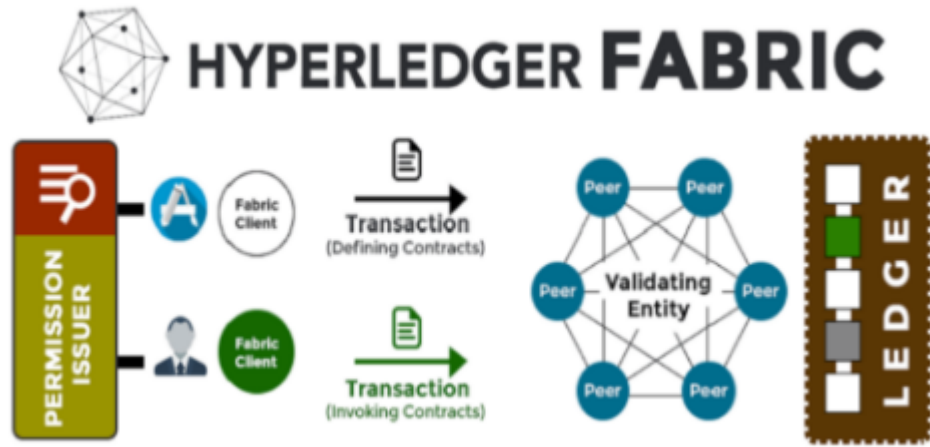
**Distributed denial-of-service (DDoS) and ransomware attacks** are used to keep information away from people who have authorized and legitimate access.

Organizations make several attempts to combat these attacks, including **web application firewalls, DDoS protection, content delivery network (CDN), and even disaster recovery**.

## Businesses, blockchain, and confidentiality

- When it is about business, confidentiality becomes a critical pillar in the cyber security space to achieve better **trust among customers and other stakeholders**
- While considering **Hyperledger Fabric**, **IBM suggests** that certain points should be kept in mind:
  - With each transaction, it is important to know **whether a participant can see complete information, a part of it, or no information at all**. It has to be mentioned under a smart contract.
  - If the **regulator** has been assigned, then they must confirm the extent of data accessed by the regulator.
  - It is important to understand the nature of your network—**static or flexible**—as confidentiality parameters may change in the future, based on new participant roles and needs.

## CONFIDENTIALITY WITH HYPERLEDGER FABRIC



Hyperledger Fabric provides features to **achieve confidentiality** with the ease of **calling a set of library files**:

1. **Attribute-based access control (ABAC)**
2. **Attribute Certificate Authority (ACA)**
3. **Hyperledger Fabric encryption library**

## BLOCKCHAIN ON INTEGRITY

Blockchain uses **cryptographic hashing** to ensure that the ledger remains tamper-proof.

- **Hashing function is one way**, which means it is logically impossible to get the data back from the hash result or from the message digest.
- **Difficult to analyze the pattern of message digest** and predict the original data as even a slight change in the actual message can result in a big difference.

An **Ethereum** account identifier is created by hashing a public key with the **Keccak-256** hashing algorithm

A **Bitcoin** address is computed by hashing a public key with the **SHA-256** algorithm

## BLOCK ARRANGEMENT AND IMMUTABILITY

Each node stores the **ledger in the form of connected blocks**, and the creation of a new block depends on the hash of the previous block.

This stops the possibility of malicious attempts to disturb, alter, or delete any blocks in the ledger.

## ACHIEVING INTEGRITY WITH HYPERLEDGER

**Committing a peer** always **validates the new block** before adding it to the ledger.

- A situation where a **peer is hacked** means that the block may get compromised from the ledger.



To avoid such a situation, there are certain methods to correct the way a block gets added in the ledger.

## VERIFYING CHAIN INTEGRITY

In this method, **each peer periodically validates its blockchain** and asks the Peer to recheck whether a **broken block is detected**. A function named **CheckChainIntegrity()** has to be called to keep the integrity check running:

```
// Periodically checks the integrity of the block chain on this peer
func CheckChainIntegrity() {
    var l ledger.PeerLedger
    blockchainInfo, err := l.GetBlockchainInfo()
    quit := make(chan struct{})

    for {
        select {
        case <- time.After(600*time.Second): //check the integrity of the blockchain every 10 minutes
        case <- quit:
            return
        }

        for k, v := range chains.list {
            l = v.cs.ledger
            ledgermgmt.VerifyChain(k, l, 0, blockchainInfo.Height)
        }
        break
    }
}
```

## BLOCKCHAIN ON AVAILABILITY

- **On-time and reliable access to information** resembles availability.
- Cyberattacks such as **DDoS** cause **huge disruption to internet services and result in websites becoming inaccessible, which costs businesses a lot of money.**

The decentralization nature of blockchain makes it harder to disrupt these applications. WHY?

- **No single point of failure**  
Even if one node in the blockchain goes down, the information can be accessed and used by the rest of the nodes in the network. As all of the nodes keep the exact copy of the ledger, it will always be up-to-date.

### Business and availability

A blockchain's **availability is determined by valid and successful transactions**. For every business, keeping record of all transactions is a core function, and these transactions could be the entries of business activities, asset entries, supply chain management records, and many more.

## Blockchain-Based DNS Security Platform

- The Domain Name System (DNS) is mainly designed to **resolve a host name query to an IP address**.
- like a phonebook of the internet and allows everyone to use it globally at the same time
- high possibility of it getting misused.

## DNS

**Anti-spam:** DNS mechanisms, **Sender Policy Framework (SPF)** and **DomainKeys Identified Mail (DKIM)**, ensure only a **predefined list of domains** should be allowed to send emails on behalf of a specific organization.

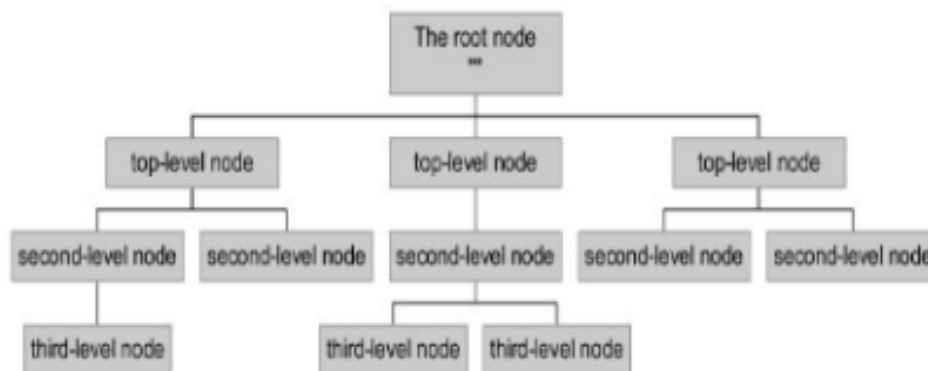
**Load sharing:** DNS services can optimize the server infrastructure by load sharing the traffic of highly utilized servers with other Underutilized servers.

**Privacy:** privacy of an organization's namespace information by **masking addresses** with different names, depending on whether they are accessed from inside or outside of the network.

## Understanding DNS components

### Namespace

A namespace is a structure of the DNS database. It is represented in the form of an **inverted tree with its root node at the top**. Each node in the tree has a label and the root node has a null label.



### Name Servers

Name servers are responsible for **storing information about the namespace in the form of zones**.

- There can be multiple name servers and ones that **load a complete zone** are said to be **authoritative for the zone**

There are two main types of name servers:

1. Authoritative servers
2. Caching servers

#### 1. Authoritative name servers

- It **provides responses to DNS queries**. It is responsible for delivering original and definitive answers to each DNS query.

There can be two types of authoritative name servers:

1. **Master server (primary name server):** It stores the original copies of all zone records. An administrator can only make changes to the master server zone database.
2. **Slave server (secondary name server):** A slave server keeps a copy of master server files. It is used to share DNS server load and to improve DNS zone availability.

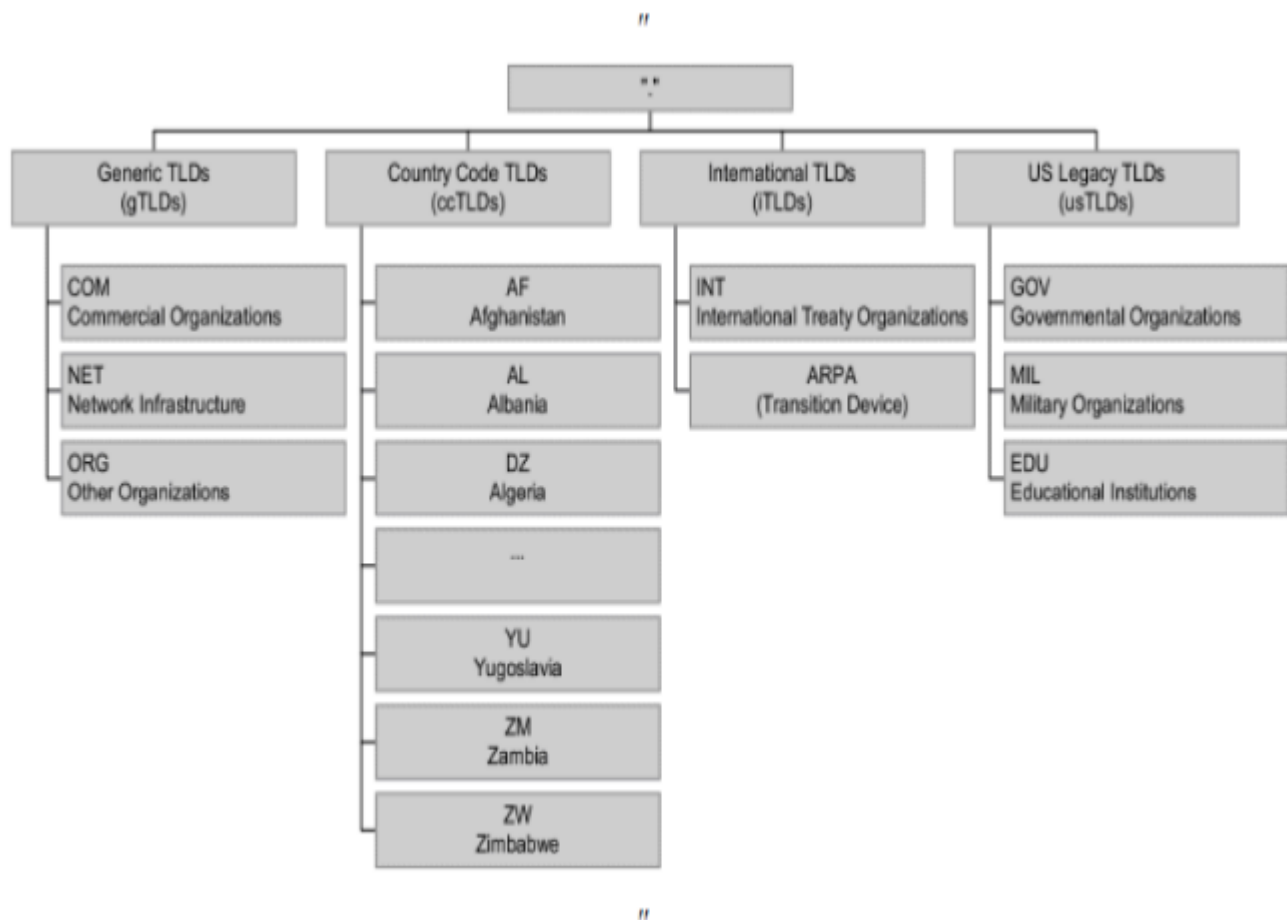
## 2. Caching name server

It brings the **name service closer** to the user and **improves overall name lookup performance**.

- It also provides a comprehensive mechanism for **providing private namespaces to local users**, by allowing users to obtain all names mapping from **local caching**.

## Resolver

- The name resolver is required to **find out the name and IP address of the name servers for the root zone**.
- The root name servers store information about **top-level zones** and **direct servers** in whom to contact for all top-level domains (TLDs).
- The resolver basically **breaks the name** into its labels from **right to left**. The first component, the **TLD**, is **queried using a root server** to obtain the designated authoritative server.



The Internet Corporation for Assigned Names and Numbers (ICANN) ensures that TLDs are

managed by delegated organizations. The Internet Assigned Numbers Authority (IANA) is operated by ICANN and is responsible for managing the DNS root zone.

IANA is responsible for managing the following TLDs:

**ccTLD** — country-code TLDs

**gTLD** — generic TLDs

**.arpa** — infrastructure TLDs

## Registries, registrars, and registrants

The DNS stores a massive database of domain names.

In order to perform registration, there are three entities working together—**registry, registrar, and registrant**:

### Registry:

An organization **maintaining the database of namespaces** that has **edit rights to that database**. The registry runs the authoritative NS for the namespace and manages the TLD names. Their role is in creating domain name extensions, setting up rules for the domain names, and working with registrars to provide domain names to the public.

- For example, **Verisign** manages the registration of .com domain names and their DNS.

### Registrar:

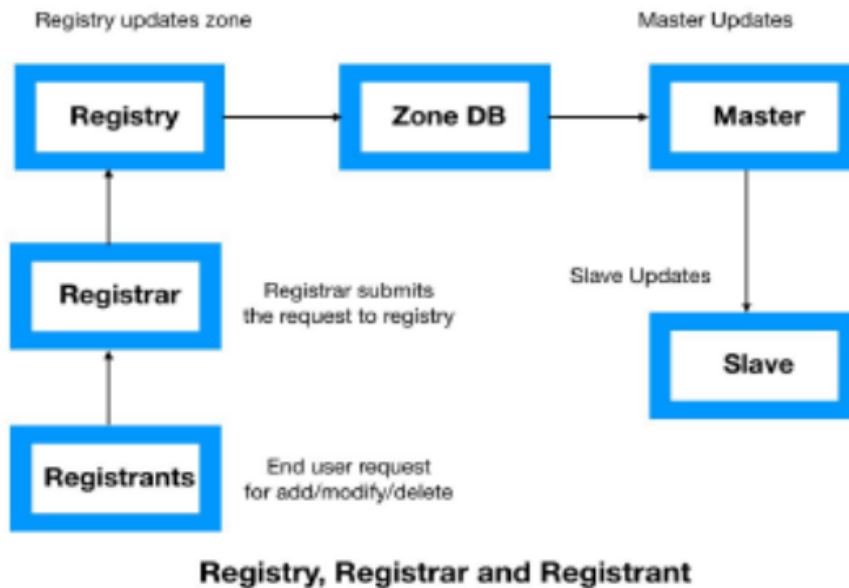
An organization that **reserves domain names and is accredited to sell domain names to the public**. This registrar must be accredited by a generic top-level-domain (gTLD) registry or a country code top-level domain (ccTLD) registry. A registrar works under the guidelines provided by domain name registries.

Only a designated registrar can modify or delete information about domain names in the central registry database. End users buy domains directly from the registrar and the end user has complete rights to switch registrar, invoking a domain transfer process between registrars.

- Some of the most popular registrars are **GoDaddy, HostGator, BigRock**, and many more.

### Registrant:

This is simply the **end user who holds the rights to a domain name**. As a domain name registrant, every person has certain rights and responsibilities, like access to information from the user's registrar regarding processes for registering, managing, transferring, renewing, and restoring the domain name registration.



## DNS records

DNS records are map files that associate with DNS server whichever IP addresses each domain is associated with, and handle requests sent to each domain.

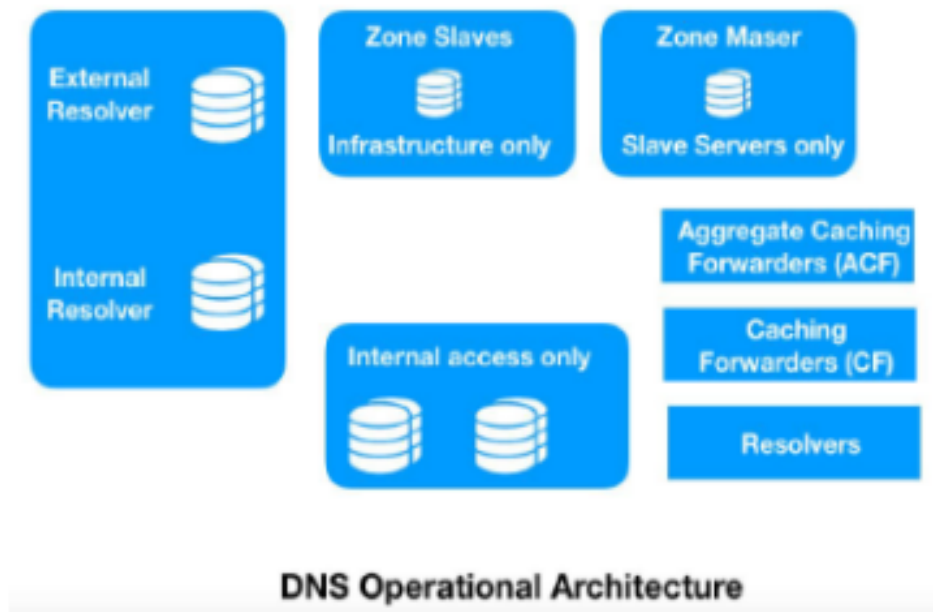
**MX records:** These records identify the servers that can exchange emails.

**TXT records:** These records deliver a method to expand the information provided through DNS. This text record stores information about the SPF that can identify the authorized server to send email on behalf of your organization.

**CNAME:** CNAMEs are essentially **domain and subdomain text aliases to bind traffic**. They indicate that the **Secure File Transfer Protocol (SFTP)** server is on the same system as the mail server. CNAME plays an important role, particularly when the server is not under organizational control such as a hosted or managed web server.

**PTR records:** These records provide the **reverse binding from addresses to names**. PTR records should exactly match the forward maps.

## DNS Architecture



The preceding standard DNS architecture can be described as follows:

### Master DNS zone:

The master zone **contains a read/write copy of zone data**. Only one master zone is allowed in a network. All the DNS records have to be written in the master zone manually or automatically. This data is then stored in a standard text file.

### Slave DNS zone:

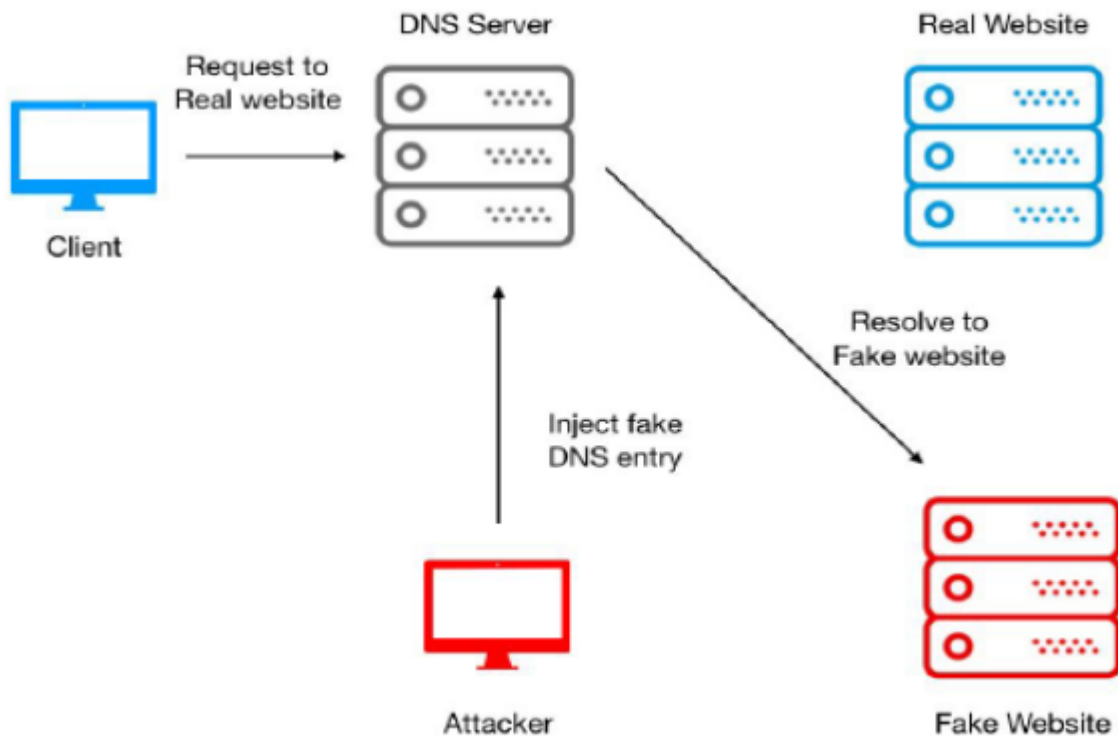
The slave zone is a **read-only copy of the zone data**. Usually, it is a copied version of master zones. If an attempt is made to change the DNS record on the secondary zone, it can redirect to another zone with read/write access. The slave DNS zone serves the purpose of backing up the DNS zone file.

### Aggregate Caching Forwarder (ACF):

It basically **forwards the requests** instead of processing them. When the server sends a response, it passes it back to its own client. In some situations, the resolver can also be a forwarder or caching forwarder. It may or may not cache the data; however, it is useful for systems such as small office home office (SOHO) gateways that want to provide DNS data to DHCP clients that don't have a predefined address for the DNS server.

## Challenges with current DNS





DNS remains a vulnerable component in the network infrastructure that is often used as an attack vector. **Firewalls leave port 53 open** and **never look inside each query**. Let's look at one of the most widely used DNS-based attacks:

## DNS spoofing

**DNS cache poisoning:** An attacker can take advantage of cached DNS records and can then perform spoofing by injecting a forged DNS entry into the DNS server. As a result, all users will now be using that forged DNS entry until the time the DNS cache expires.

**Compromising a DNS server:** A DNS server is the heart of the entire DNS infrastructure. An attacker can use several attack vectors to compromise a DNS server and can provide the IP address of a malicious web server against each legitimate DNS query.

**Man-in-the-middle (MITM) attack:** In this type of attack, a threat actor keeps listening to conversations between clients and a DNS server. After gathering information and sequence parameters, it starts spoofing the client by pretending to be the actual DNS server and provides the IP addresses of malicious websites.

## Blockchain-based DNS solution

### DNSChain

**DNSChain** is one of the most active projects to transform the DNS framework and protect it from spoofing challenges.

- DNSChain is a blockchain-based DNS software suite that replaces X.509 **public key infrastructure (PKI)** and **delivers MITM proofs of authentication**. It allows internet users to set a **public DNSChain server for DNS queries** and access that server with domains ending in **.bit**.

## X.509 PKI replacement

- X.509 is a standard framework that defines the format of PKI to identify users and entities over the internet.

## MITM-proof DNS infrastructure

- This uses a **public key pinning technique to get rid of the MITM attack problem**. Public key pinning specifies two pin-sha256 values; that is, it pins two public keys (one is the pin of any public key in the current certificate chain and the other is the pin of any public key not in the current certificate chain).

## Deploying Blockchain-Based DDoS Protection

### DDoS attacks

Attackers never legitimately control their attacking machines, but rather they **infect millions of computers worldwide with some tailored malware** and then get **complete access to launch a massive DDoS attack**.

This collection of **millions of infected computers** is named a **botnet** and the individual infected computers are named bots.

- Significant first attack occurred in 1999, and it targeted the **University of Minnesota**. It impacted more than 220 systems and brought down the entire infrastructure for several days.

## How does DDos work? CYBER KILL CHAIN

A computer scientist at Lockheed-Martin Corporation coined a term called cyber kill chain that lays out the **stages of a cyber attack**.

These stages are:

- 1. Reconnaissance:** The attacker identify its target device and starts searching for vulnerabilities in it.
- 2. Weaponization:** The attacker uses a remote tool kit and malware such as virus or worm to address the vulnerability.
- 3. Delivery:** The threat actor inject the cyber weapons to the victim network through several methods such as phishing email, drive-by download, USB drives, insiders and so on
- 4. Exploitation:** The malware code is used to trigger the attack, taking action on target network to exploit vulnerabilities
- 5. Installation:** Malware is now installed in the victim machine

**6. Command and control:** This malware allows the remote threat actor to gain access to victim machine

## Types of DDoS attack

- Attacks targeting network resources
- Attacks targeting server resources
- Attacks targeting application resources

### Attacks targeting network resources

- User datagram protocol (UDP) flood
- ICMP flood
- Internet Group Management Protocol (IGMP) flood
- Amplification attacks

### Attacks targeting server resources

- TCP SYN Flood
- TCP RST attack
- Secure sockets layer (SSL) based Attack
- Encrypted HTTP attacks

### Attacks targeting application resources

- DNS flooding
- Regular expression DoS attacks
- Hash collision DoS attacks

## Challenges with current DDoS solutions

- As per the recent report by Radware, **43% of organizations experienced burst attacks**, but the **rest were unaware** of whether they were attacked.
- On February 28, 2018, **GitHub**, the code hosting website, was hit with the **largest-ever DDoS attack**, recorded at 1.35 Tbps.

As DDoS attack **falls under cyber threat category** that makes it **unfeasible to deploy any security prevention mechanism** as system vulnerabilities are under control of organizations but threats cant be controlled.

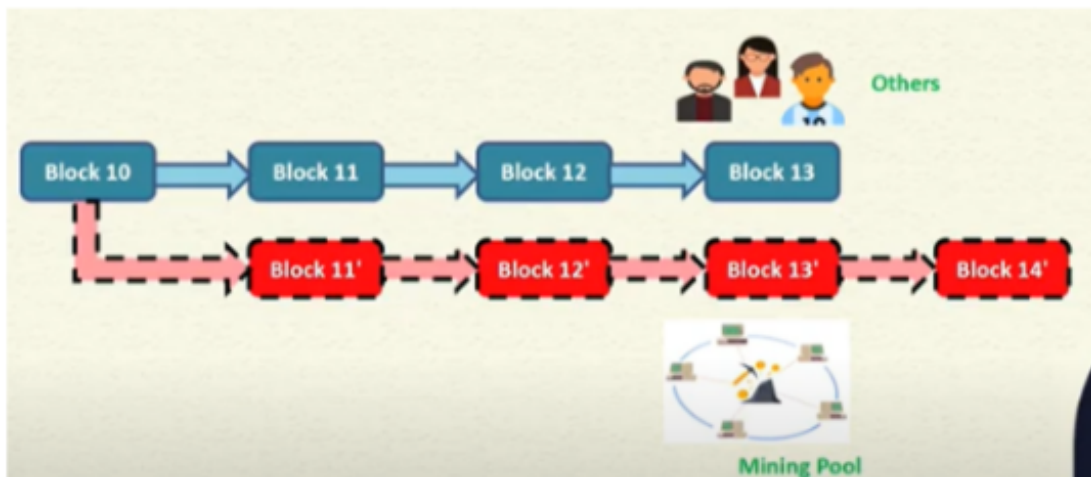
## How blockchain can transform DDoS protection?

- Blockchain technology can be used to deploy a decentralized **ledger to store blacklisted Ips**
- Blockchain technology **eliminates** the risk of a **single point of failure**

## Selfish Mining Attack

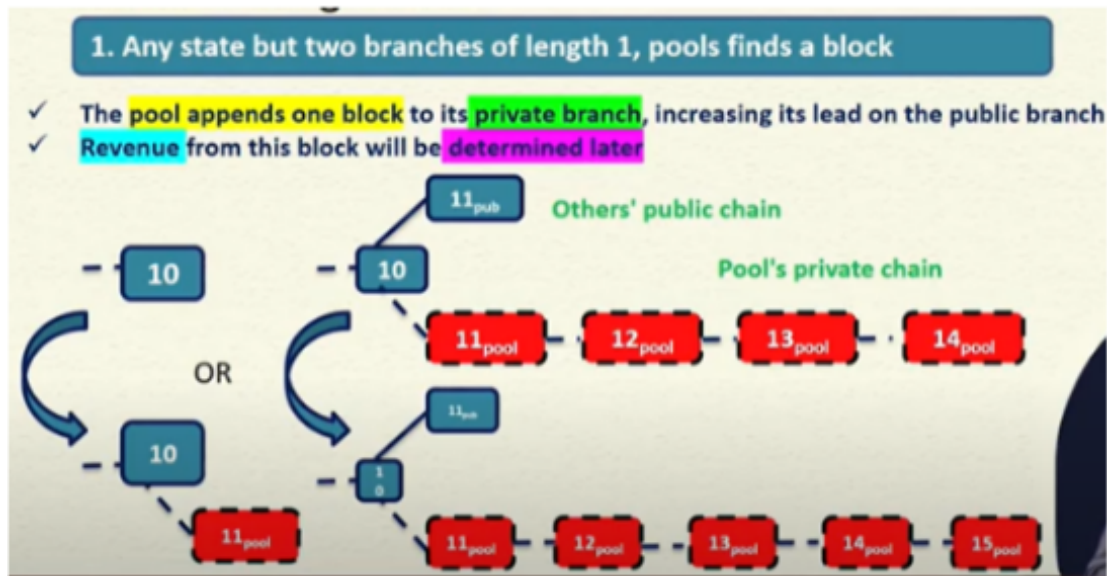
The selfish mining attack is a potential attack on blockchain networks where an **attacker secretly mines blocks in private, without broadcasting them, and then releases them to the network to overtake the public chain.**

- This results in the other miners' efforts being wasted, as their blocks become invalidated, and the **attacker's longer chain is accepted** by the network as the new valid chain
- Pool intentionally forking the chain for keeping discovered blocks private
- The honest nodes continue to mine on the public chain. The pool mines on its own private branch
- When the public branch approaches the pool's private branch in length the selfish miners reveal blocks from their private chain to the public



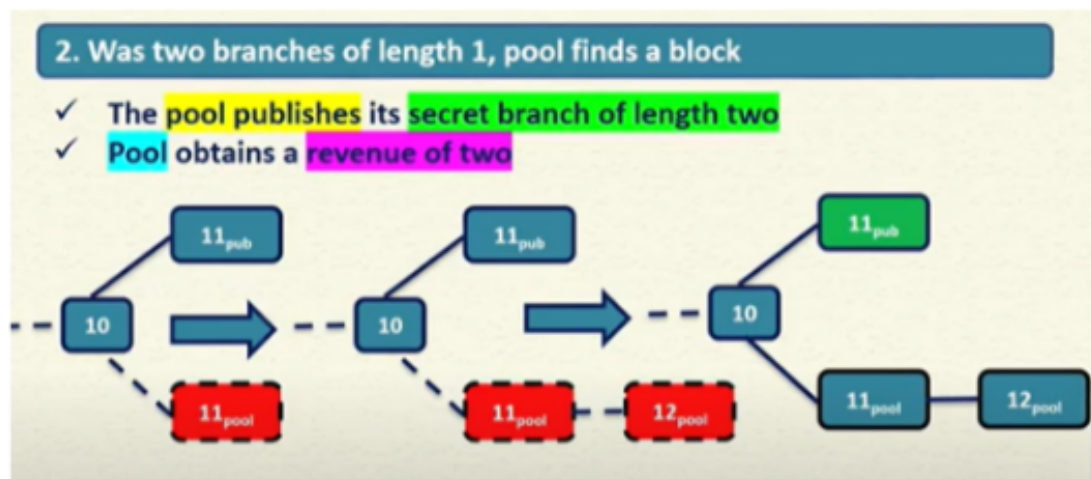
**CASE 1: Any state but two branches of length 1, pools finds a block**

- The pool appends one block to its private branch, increasing its lead on the public branch
- Revenue from this block will be determined later



**CASE 2: Was two branches of length 1, pool finds a block**

- The pool publishes its secret branch of length two
- Pool obtains a revenue of two

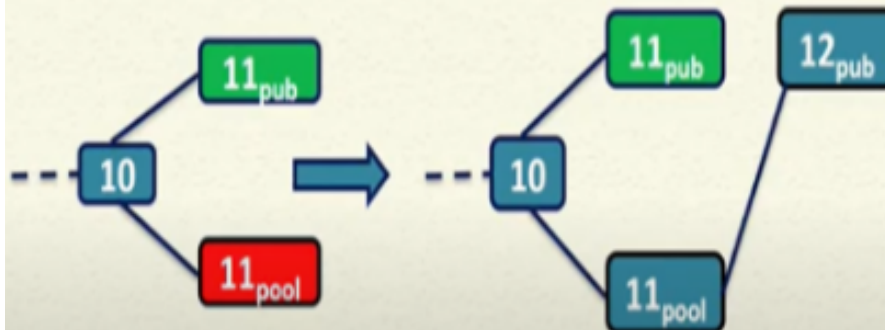


**CASE 3: Was two branches of length 1, others find a block after pool head**

- The pool and the others obtain a revenue of one each
- The others for the new head, the pool for its predecessor

### 3. Was two branches of length 1, others find a block after pool head

- ✓ The pool and the others obtain a revenue of one each - the others for the new head, the pool for its predecessor

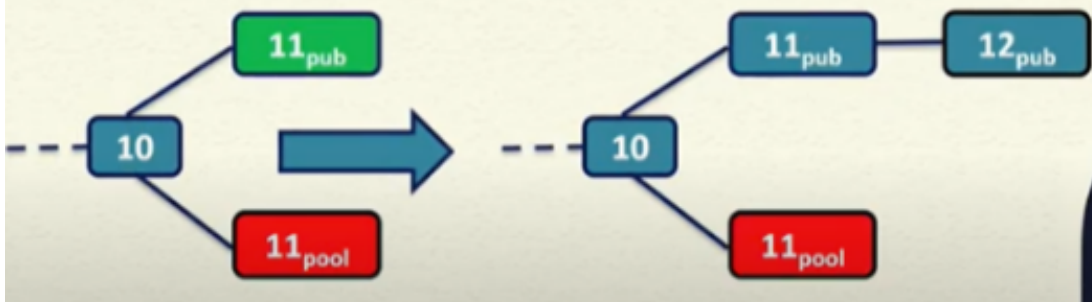


### CASE 4: Was two branches a length 1, others find a block after others' head

- The others obtain a revenue of two

### 4. Was two branches of length 1, others find a block after others' head

- ✓ The others obtain a revenue of two



### CASE 5: No private branch, others find a block

- Both the pool and the others start mining on the new head
- The others obtain a revenue of one

### 5. No private branch, others find a block

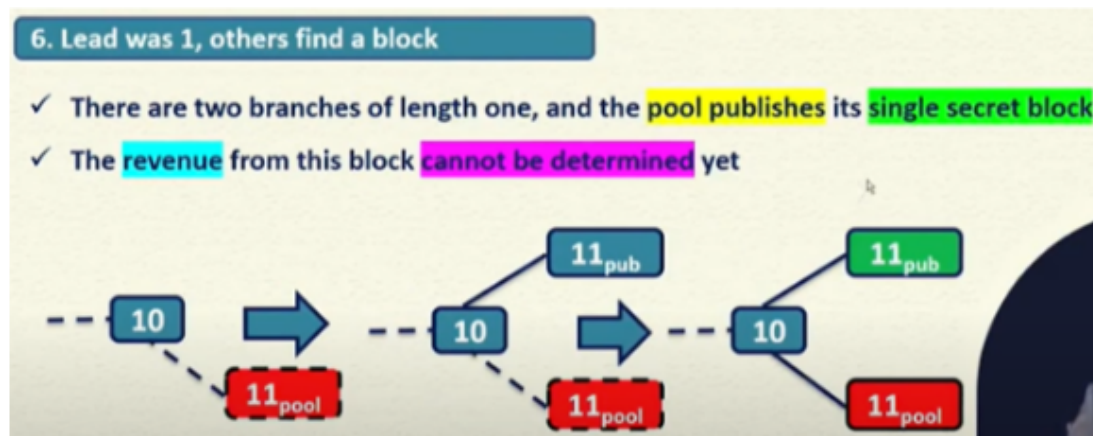
- ✓ Both the pool and the others start mining on the new head
- ✓ The others obtain a revenue of one





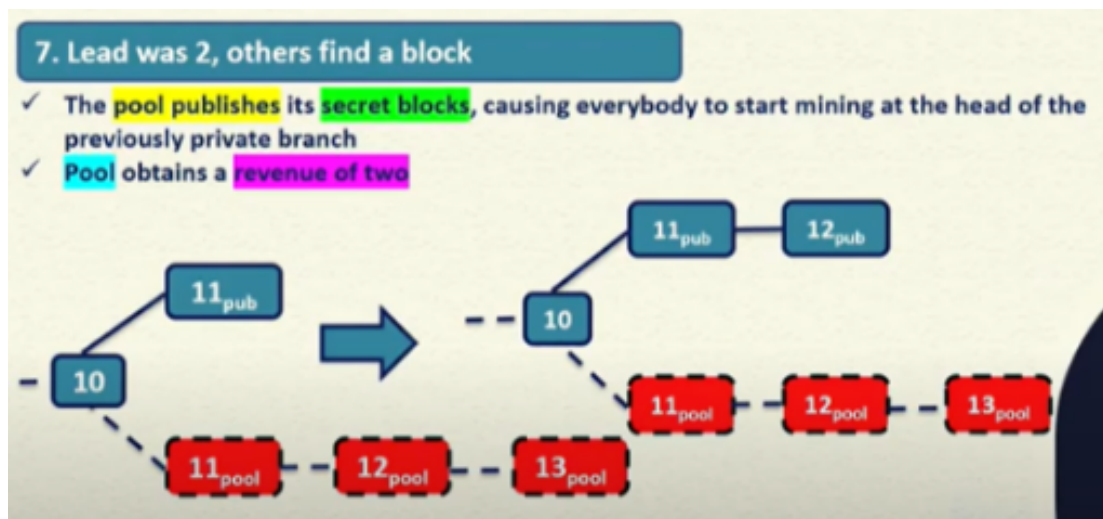
### CASE 6: Lead was 1, others find a block

- There are two branches of length one, and the pool publishes its single secret block
- The revenue from this block cannot be determined yet



### CASE 7: Lead was 2, others find a block

- The pool publishes its secret blocks, causing everybody to start mining at the head of the previously private branch
- Pool obtains a revenue of two



### 7. Lead was 2, others find a block (Contd.)

- ✓ The pool publishes its secret blocks, causing everybody to start mining at the head of the previously private branch
- ✓ Pool obtains a revenue of two



### CASE 8. Lead was more than 2, others win

- The pool now reveals its i'th block
- Pool obtains a revenue of one

