ISTANBUL TECHNICAL UNIVERSITY ★ GRADUATE SCHOOL

A SECURE PEER-TO-PEER MEDIA DISTRIBUTION USING BLOCKCHAIN TECHNOLOGY

M.Sc. THESIS

Anas MHAISH

Computer Science Department

Computer Science Programme

FEBRUARY 2022

ISTANBUL TECHNICAL UNIVERSITY ★ GRADUATE SCHOOL

A SECURE PEER-TO-PEER MEDIA DISTRIBUTION USING BLOCKCHAIN TECHNOLOGY

M.Sc. THESIS

Anas MHAISH
(704191003)

Computer Science Department

Computer Science Programme

Thesis Advisor: Assoc. Prof. Dr. Enver Özdemir

FEBRUARY 2022

İSTANBUL TEKNİK ÜNİVERSİTESİ ★ LİSANSÜSTÜ EĞİTİM ENSTİTÜSÜ

BLOCKCHAIN TEKNOLOJİSİNİ KULLANAN GÜVENLİ EŞLER ARASI MEDYA DAĞITIMI

YÜKSEK LİSANS TEZİ

Anas MHAISH
(704191003)

Bilgisayar Bilimleri Anabilim Dalı

Bilgisayar Bilimleri Programı

Tez Danışmanı: Assoc. Prof. Dr. Enver Özdemir

ŞUBAT 2022

Anas MHAISH, a M.Sc. student of ITU Graduate School Student ID 704191003 successfully defended the thesis entitled "A SECURE PEER-TO-PEER MEDIA DISTRIBUTION USING BLOCKCHAIN TECHNOLOGY", which he prepared after fulfilling the requirements specified in the associated legislations, before the jury whose signatures are below.

Thesis Advisor :	**Assoc. Prof. Dr. Enver Özdemir** İstanbul Teknik Üniversitesi	.....
Jury Members :	**Prof. Dr. Muhammed Oğuzhan Külekci** İstanbul Teknik Üniversitesi	.....
	Prof. Dr. Güneş Karabulut Kurt Polytechnique Montréal	.....

Date of Submission : **02 February 2022**
Date of Defense : **08 February 2022**

To my parents, siblings, and spouse,

FOREWORD

I dedicate this thesis to my parents and sibilings for their support and encouragment, to my spouse for her endless love and care, and to my family and friends who helped and supported me in my journey.

I would like to express my sincere gratitude to my advisor Assoc. Dr. Enver Özdemir for his support and enlightenment. And to the examining committee members for their time and for being part of my academic education.

FEBRUARY 2022 Anas MHAISH

TABLE OF CONTENTS

Page

ABBREVIATIONS

D2D	**:** Device-to-device
DDoS	**:** Distributed Denial-of-service attack
ERC	**:** Ethereum request for comment
IP	**:** Internet Protocol
MD	**:** Message Digest Algorithm
MTT	**:** Malware Tolerance Table
OHT	**:** Owner Hashed-Segments Table
P2P	**:** Peer-to-Peer
PoA	**:** Proof of Authority
PoS	**:** Proof of Stake
PoW	**:** Proof of Work
SHA	**:** Secure Hash Algorithm
TCP	**:** Transmission Control Protocol
UDP	**:** User Datagram Protocol

SYMBOLS

H(P)	**:** The Hash of the Segment
N(P)	**:** The Name of the Segment
P	**:** Segment as a Packet
U	**:** User of a node

LIST OF TABLES

Page

LIST OF FIGURES

Page

A SECURE PEER-TO-PEER MEDIA DISTRIBUTION USING BLOCKCHAIN TECHNOLOGY

SUMMARY

As demand for online services has risen, video streaming services have scored a high share and bandwidth usage. A high proportion of these online services are users using smart devices to download and cache the streamed video segments. Collaborative video stream approaches have appeared, some of them are distributed among the internet, and others are using the wireless connection of the community to distribute the video segments.

This thesis presents a method to distribute the video segments using a wireless connection to build a peer-to-peer network. It implements a method that describes the usage of Blockchain technology facilitates a secure and fast data transfer with the ability to detect and respond to malicious behaviors.

Users can join and leave the network whenever they want, once the user joins the peer-to-peer network, previous streamed video segments owners will be shared with him by the Blockchain. The user can participate in the network by sharing his downloaded streamed segments to special nodes, these special nodes ensure the security of the method and are responsible for adding blocks to the blockchain.

The thesis provides two layers of security, the first one is the blockchain itself which ensures the immutability and distribution of the data and the network contribution. The second layer is the Owner Hashed-Segments Table which helps to shared hashed segments among the nodes for validation and provides a dictionary of packets owners to ease request packets from their owners. It provides immutability and integrity to the segment. Inside the second layer, there is the Malware Tolerance Table which monitors the nodes for any malicious behavior and protects the system against Distributed Denial-of-service attacks. These are included inside the block of the blockchain and distributed to all nodes by the verifier nodes. These verifier nodes identities should be revealed and authenticated.

A peer-to-peer network is constructed that consists of any smart devices. The smart devices represent the Nodes (peers) in the network. A special node called Verifier is responsible for the security measurements of the network, and responsible for adding blocks to the blockchain and distributing the blockchain among the nodes. A Node is going to connect to a streaming service or server to download video segments. Any node can join the network by Wifi without the need to connect to stream service or to have an internet connection.

The block does not contain the segment itself but it contains the hash value of the segment, this value is fixed in size and a result of running the Message-Digest Algorithm function. The choice of this hashing algorithm was based on utilizing the bandwidth as possible as it output 128 bits.

The blockchain network follows Proof-of-Authority consensus, the verifier nodes are the validators and the moderators of the system which are responsible for validating the blocks and transactions in the network. This consensus will force the verifiers to act honestly as have their own reputation at risk, and their identity is registered and revealed in the network.

User Datagram Protocol has been used to avoid overwhelming the usage of bandwidth, the User Datagram Protocol does not ensure that all data packets arrive in order and does not require establishing handshakes with the target node. This is important to keep streaming synced on the network between the nodes as possible. And to keep and utilize the bandwidth as possible.

Tests have been obtained by implementing software. A WiFi network has been set, and a peer-to-peer network has been established. The WiFi network represents the peer-to-peer network which will be the infrastructure for the consortium blockchain network. The nodes have been defined in the network, some of them will have to log in using a simple authentication method, this is a required step for the verifiers so their identity can be validated, as verifiers should not act maliciously in the network.

In these experiments, the network latency was minimized as the focus is to calculate the time cost of the verifier to add a block to the network and validate the segment. The time cost of the verifier consist of the time that requires the segment to be transmitted to the verifier, the verifier validating the segment and adding the hash of the segment as a transaction to the block, the time requires the verifier to update the Owner Hashed-Segments Table inside the block with the new segment owner IP, and the time cost of adding a new block to the blockchain. The verifier will keep holding the segments until they reach 100 segments, and then it will release the block to the chain.

The selection of the verifier node is complex and should be carefully studied. The incentive method has not been discussed in this method, by adding incentives we will encourage the users to join and contribute to the network, enforce honesty and reduce malicious behaviors. These have been set as future work.

BLOCKCHAIN TEKNOLOJİSİNİ KULLANAN GÜVENLİ EŞLER ARASI MEDYA DAĞITIMI

ÖZET

Çevrimiçi servisler arttıkça video akış servisleri yüksek bir bant genişliğine sahip olmaktadırlar. Bu çevrimiçi servislerin yükse bir bölümü akışlı video segmentlerini indiren ve önbelleği kullanan akıllı cihaz sahipleri tarafından kullanılmaktadır. İşbirlikçi video akışı yaklaşımları ortaya çıkmıştır. Bu yaklaşımlardan bazıları internette dağıtılmakta, diğerleri ise video parçalarını dağıtmak için kablosuz bağlantı kullanılmaktadır.

Bu tez, eşler arası ağ oluşturmak için video parçalarını kablosuz ağ kullanarak dağıtan bir metodu tanıtmaktadır. Bu tezde, kötü niyetli davranışları tespit etme, ve bunlara yanıt verme yeteneği ile güvenli ve hızlı bir veri aktarımını kolaylaştıran bir Blockchain teknolojisinin kullanımını açıklayan bir yöntem uygulanmıştır.

Kullanıcılar ağa istedikleri zaman katılıp ayrılabilmektedirler. Kullanıcı eşler arası ağa katıldığında, önceki akışlı video segmentleri sahipleri Blockchain tarafından onunla paylaşılacaktır. Kullanıcı, indirmiş olduğu akışlı segmentleri özel düğümlerle paylaşarak ağa katılabilirler. Bu özel düğümler yöntemin güvenliğini sağlar ve blok zincirine blok eklemekten sorumludur.

Tez, iki güvenlik katmanı saglar; birincisi, verilerin değişmezliğini, dağıtımını ve ağ katkısını sağlayan blok zincirinin kendisidir. İkinci katman, doğrulama için düğümler arasında paylaşılan karma segmentlere yardımcı olan ve sahiplerinden paket talep etmeyi kolaylaştırmak için bir paket sahipleri sözlügü sağlayan Sahip Karma Segmentleri Tablosu'dur. Bu tablo, segmente degişmezlik ve bütünlük sağlamaktadır. İkinci katmanın içinde, dügümleri herhangi bir kötü niyetli davranış için izleyen ve sistemi Dağıtılmış Hizmet Reddi saldırılarına karşı koruyan Kötü Amaçlı Yazılım Tolerans Tablosu vardır. Bunlar blok zincirinin bloğuna dahil edilir ve doğrulayıcı düğümler tarafından tüm dügümlere dağıtılır. Bu doğrulayıcı düğüm kimlikleri ortaya çıkarılmalı ve kimliği doğrulanmalıdır.

Tezde, herhangi bir akıllı cihazdan oluşan bir eşler arası ağ oluşturuldu. Akıllı cihazlar, agdaki Düğümleri (eşleri) temsil etmektedirler. Verifier adlı özel bir düğüm, ağın güvenlik ˘ ölçümlerinden, blok zincirine blok eklemekten ve blok zincirini düğümler arasında dağıtmaktan sorumludur. Bir Düğüm, video segmentlerini indirmek için bir akış hizmetine veya sunucuya bağlanmaktadır. Herhangi bir düğüm, akış hizmetine bağlanmaya veya internet bağlantısına ihtiyaç duymadan Wifi ile ağa katılabilir.

Bu tezde, blok yapısı tartışılmıştır. Her blok işlemleri, Sahip Karma Segment Tablosunu ve Kötü Amaçlı Yazılım Tolerans Tablosunu içerir. Bunların tümü, Message-Digest Algorithm işlevi kullanılarak blok, blok zincirine eklenirken karmalanır ve değer, bloğun kendi içinde saklanır. Bir sonraki bloğu oluştururken, bu karma değer, işlemlere, Sahip Karma Segmentler Tablosuna ve bir sonraki bloğun Kötü Amaçlı Yazılım Tolerans ˘ Tablosuna dahil edilmektedir. Blok zincirinin zincirini bu şekilde oluşmaktadır. Blok madenciliği yapılmadığı için blok başlığı "nonce" dahil edilmemiştir. Versiyonu, zaman damgasını, bu tezde önemli olan önceki blok karmasını ve işlemleri koruyan Merkle kök karmasını tutan normal blok başlığı örtük olarak dahil edilmiştir.

Tez uygulamasında, normal dügüm tarafından indirilen her segment, onaylanmak üzere Dogrulayıcı düğümüne gönderilmiştir. Doğrulayıcının bunu blok zincirine bir blok veya işlem olarak eklemesi için Doğrulayıcı, video segmentini çevrimiçi akış hizmetinden indirmeyi, bu segmenti karmalamayı ve normal dügüm tarafından gönderilen segmentin karma degeriyle karşılaştırmaktadır.

Blok, segmentin kendisini içermez, ancak segmentin karma degerini içerir, bu değer boyut olarak sabitlenir ve Message-Digest algoritma fonksiyonunun çalıştırılmasının bir sonucudur. Bu karma algoritmanın seçimi, 128 bit çıktı verdigi için bant genişliğinin olabildiği kadar kullanımına olanak sağlamasına dayanmaktadır.

Blok zinciri ağı, Yetki Kanıtı konsensüsünü takip etmektedir. Doğrulayıcı düğümler, ağdaki blokları ve işlemleri doğrulamaktan sorumlu olan sistemin doğrulayıcıları ve moderatörleridir. Bu konsensüs, doğrulayıcıları, kendi itibarları risk altında olduğu ve kimlikleri kayıtlı oldugu için ve ağda ortaya çıkarıldığı için dürüstçe davranmaya zorlamaktadır.

Kullanıcı Datagram Protokolü, bant genişligi kullanımının aşırı yüklenmesini önlemek için kullanılmıştır. Kullanıcı Datagram Protokolü, tüm veri paketlerinin sırayla gelmesini sağlamaz ve hedef düğümle el sıkışmayı gerektirmez. Bu, ağda düğümler arasında mümkün olduğunca eşit akışı sağlamak ve bant genişliğini mümkün olduğunca korumak ve kullanmak için önemlidir.

Testler yazılım uygulanarak elde edilmiştir. Bir WiFi ağı kurulmuştur ve eşler arası bir ağ kurulmuştur. WiFi ağı, konsorsiyum blok zinciri ağının altyapısı olacak olan eşler arası ağı temsil etmektedir. Düğümler ağda tanımlanmıştır, bazılarının basit bir kimlik doğrulama yöntemi kullanarak oturum açması gerekecektir, bu doğrulayıcılar için kimliklerinin dogrulanabilmesi için gerekli bir adımdır, çünkü doğrulayıcılar ağda kötü niyetli hareket etmemelidir.

Yapılan deney, veri setini hazırlamak için bir youtube videosu kullandı. Youtube videosunu çevrimdışı çalıştırmak için gerekli olan youtube önbellege alınmış dosyalar indirildi. Bu önbellege alınmış dosyalar, sistem performansının çeşitli segment boyutlarında test edilmesine yardımcı olmak için 15 KB, 30 KB ve 50 KB olmak üzere 3 boyutta hazırlanmıştır. Biri doğrulayıcı, geri kalanı temel dügümler olmak üzere deneyde 10 düğüm kullanılmıştır. Genel video boyutu 3 GB'tır, bir WiFi bağlantısı ile 2.4GHz-802.11g özelliği ile yayınlanmıştır.

Bu deneylerde, amaç doğrulayıcının ağa bir blok eklemek ve segmenti doğrulamak için zaman maliyetini hesaplamak oldugu için ağ gecikmesi en aza indirilmiştir. Doğrulayıcının zaman maliyeti, segmentin doğrulayıcıya iletilmesiyle, doğrulayıcının segmenti doğrulamasıyla ve segmentin karmasını bloğa bir işlem olarak eklemesiyle ve doğrulayıcının blok içindeki Sahip Karma Segment Tablosunu yeni segment sahibinin IP'si ile güncellemesiyle geçen sürelerden oluşmaktadır. Doğrulayıcı, 100 segmente ulaşana kadar segmentleri tutmaya devam edecek ve ardından bloğu zincire bırakacaktır.

Doğrulayıcının segmenti doğrulaması ve blok zincirine eklemesi için gereken maliyet süresi düşük sayılmıştır ve iletim performansının üzerinde etkisi olmadığı kabul edilmiştir. Böylece tanıtılan yöntemin hız verimliliği ve güvenilirliği kanıtlanmıştır.

Doğrulayıcı düğümün seçimi karmaşıktır ve dikkatle incelenmesi gerekmektedir. Teşvik yöntemi bu metodda tartışılmamıştır. Gelecekteki çalışmalar, kullanıcıları ağa katılmaya ve katkıda bulunmaya, dürüstlüğü uygulamaya ve kötü niyetli davranışları azaltmaya yönelik teşvikler eklemeyi amaçlayacaktır.

1. INTRODUCTION

Many of the information on the World Wide Web is represented as visual content, these videos are shared and watched by a lot of people all over the world, especially on social media platforms. Communication and delivery of information are faster and easier using videos. Most of the online learning platforms are rely on videos to educate their courses and allow these videos to be watched on-demand. Olympics and sports use live steaming to deliver its contents to a variety of the population.

Video streaming is taking up most of the bandwidth in our mobiles. People are spending more time watching clips and videos on their phones. According to statistics, there are more than 1 billion videos are watched per day on Youtube, and more than 70% of Youtube videos is being watched on mobile devices [1]. In addition, a huge number of Youtube channels have been created, each one delivers a good amount of videos. And after the pandemic many relied on videos to watch lessons and meetings offline and the use of the streaming service Netflix has significantly risen because of the lockdown.

This work implements the security part of video streaming by using a peer-to-peer network that is hosted on a WiFi network. It facilitates blockchain technology to add a security layer to protect the users and the cached video segments being streamed. It took security measurement to protect against attacks like the majority attack and the Distributed Denial-of-service (DDoS) attack making the distribution fast and secure. The work defines a protocol for the users to use to request segments and contribute effectively and securely to the network.

1.1 Purpose of Thesis

The aim of this paper is to provide and evaluate a method for streaming videos in a fast and secure way by facilitating Blockchain technology to create a decentralized network that allows peers to exchange cached video segments within a Peer-to-Peer network, without relying on any centralized streaming service. This will provide a fast video streaming method by distributing the already downloaded cached video segments among the Peer-to-Peer network.

The peers in our method represent the smart devices that connect to each other through a Peer-to-Peer network. The employed Blockchain network is a consortium Blockchain network. We rely on specific notes called Verifiers which will ensure the security and integrity of the blockchain. The blockchain will provide transparency and security but not anonymity, which is not a concern in this environment.

The thesis provides two layers of security, the first one is the blockchain itself which ensures the immutability and distribution of the data and the network contribution. The second layer is the Owner Hashed-Segments Table which helps to shared hashed segments among the nodes for validation and provides a dictionary of packets owners to ease request packets from their owners. Inside the second layer, there is the Malware Tolerance Table which monitors the nodes for any malicious behavior and protect the system against Distributed Denial-of-service attack (DDoS) attacks.

Experimental tests and results have been provided at the end of the thesis, these experiments have been done with a minimized network latency as the thesis cares about the time cost that the Verifier nodes will take to secure the network and adding blocks to the blockchain. The thesis has not discussed the incentive methods and set these as a future plan of work.

1.2 Documentation Review

1.2.1 Blockchain

Blockchain as shown in Figure 1.1 is a chain of blocks that are linked together using cryptographic primitives. Blockchain represents a distributed database that holds transactional records, called a block. These distributed databases are shared across a network of nodes. Nodes are the electronic devices that connect to this network and possess an IP address. They are the communication endpoints that any user or application that wants to interact with the blockchain needs to go through them. Nodes can be computers, mobiles, or servers [2].

Transactions are the source purpose of the blockchain, these transactions show the asset movement that can be described as products, purchases, or even intellectual property. These transactions are blocked together and inserted as blocks into the blockchain. This block is immutable because of the blockchain properties and anyone can view its content which provides transparency.

A first major deployment for blockchains was Bitcoin as a digital currency, It first appeared at a white paper authored under Satoshi Nakamoto in 2018. Bitcoin is one of the first digital currencies to use peer-to-peer (P2P) technology to provide instant payments. Miners, who hold the computing power and participate in the network, are in charge of processing the transactions on the blockchain and are motivated by rewards [3].

Blockchains should achieve six key properties [4]:

- **Decentralized**, blockchains are spread across network nodes, it doesn't rely on a single centralized node. An asset is distributed to all entities instead of copied. Anyone can have access to this asset at the same time. No one can lock the asset by changes or modifications and changes are transparent.

- **Transparent**, anyone can join the network and view all the information inside the network. On blockchain every transaction is traceable. This provides more honesty between nodes.

- **Open Source**, it eliminates any kind of tampering or revisioning, and it's publicly available.
- **Autonomy**, consensus provides a trust mechanism between nodes to transfer and update data. The consensus is used to achieve trust and security among the decentralized nodes. It set rules that decide the contribution legitimacy made by the contributors to the blockchain and ensure fairness among the participants.
- **Immutable**, blockchain ledger will remain unaltered and unchanged, data are kept in the blockchain and cannot be altered. The unique hash value identifies a block that is linked to the block content. Each block will hold the unique hash value of the previous block.
- **Anonymity**, nodes can exchange data without disclosing their identity information and other transactions. The wallet address owner's identity is secure and no links can indicate the person's identity.

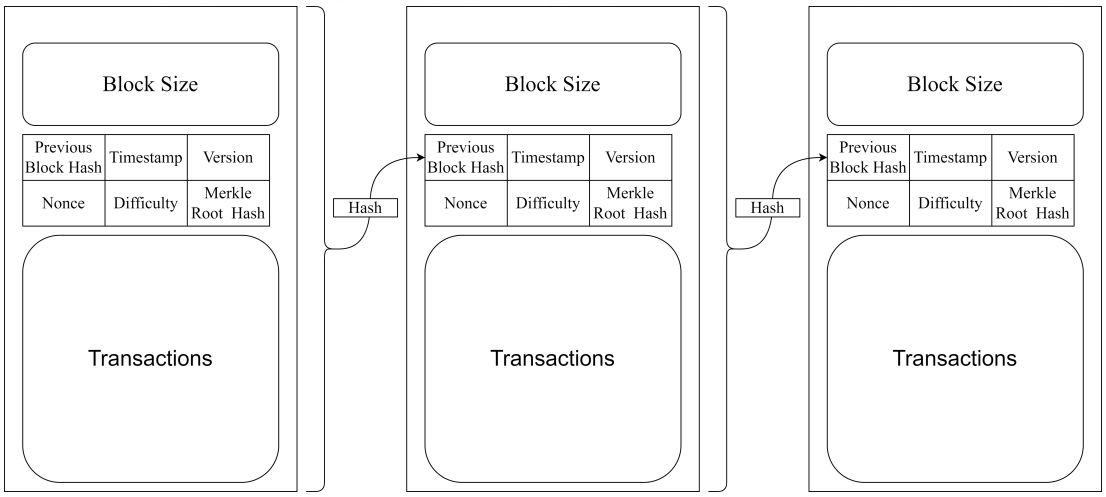

Figure 1.1 : Blockchain Structure.

1.2.2 Blockchain block

A block represents a page in a ledger. It must contain a unique cryptographic hash value that distinguishes it from other blocks. The block header contains the hash of the previous block that has been added to the blockchain, it contains a nonce variable that is incremented by the proof of work. and it holds the root hash of the Merkle tree that is named after the mathematician Ralph Merkle [5], it represents all the information of individual transactions "Leaves" and hashes of the leaves "Branches". Using Merkle tree will reduce data transmission and computing resources [6].

The block is a data structure explained in Figure 1.2 that stores information of the blockchain. Blocks need to be verified before being added to the network. The first block that gets added to the blockchain is called the Genesis block. The block contains and stores much information. The first part of the block is the header that contains the version number, the timestamp, the hash of the previous block, the nonce, and the Merkle root hash for all transactions in a block. The nonce is the complicated puzzle that miners try to solve, it is a complex mathematical problem, and finding it will allow the miners to add blocks to the blockchain.

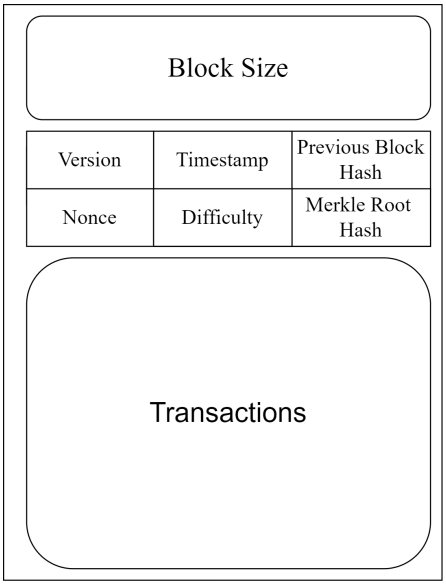

Figure 1.2 : Block Structure.

1.2.3 Blockchain classification

Blockchain technologies can be classified into three main types: Public Blockchain, Private Blockchain, and Consortium Blockchain. The table 1.1 provide a comparison between them.

1.2.3.1 Public blockchain

Public blockchains [7] are completely independent. Transactions are stored in blocks without any centralized organization. They are transparent as long as the users follow the security protocols and they are open to anyone who wants to contribute with incentives and rewards for their contribution. Bitcoin is an example of a cryptocurrency build on public blockchain. It is characterized with performance and energy consuming to preserve the anonymity and transparency in the network. Proof of Work (PoW) consensus protocol is mostly used in public blockchain networks.

1.2.3.2 Private blockchain

The main property of the private blockchain [7] is permission and partial decentralization. A single entity has access and authority over the network. It can manage and add blocks and transactions. Private networks ensure each user to have a substantiated identity since that defines the type of access they have. Therefore, Private blockchain doesn't provide anonymity in identity, however, transactions are transparent. Private network mostly uses the Proof of Authority (PoA) consensus protocol. It has advantages of stability and low fees, and PoA does not require computation power like PoW.

1.2.3.3 Consortium blockchain

The Consortium Blockchain networks are a branch of the private networks. The main difference is that consortium blockchains are governed by a group rather than a single entity. Transactions are required to be verified by a few nodes which can be trusted, these few nodes validate new blocks. [8]

One of the best features of Consortium blockchain networks is that there is no risk of 51% Attack, which happens when 51% of computational power belongs to a single organization and they can override and reverse the transaction process. In Consortium blockchain networks you have to go through proper authorization and your identity will be revealed. [9]

1.2.3.4 Hybird blockchain

This blockchain is a combination of public and private blockchains. It controls the access to the asset and can decide which users can add or view blocks on the blockchain, and even which transactions can be made public. It combines the permission property of the private network with the public blockchain transparent and security feature. However, the private entity cannot change the transactions.

Table 1.1 : BlockChain Classifications.

Property	Public	Private	Consortium
Decentralized	Yes	No	Partial
Consensus	No Permission	Permissioned	Permissioned
Immutability	Hard to alter	Possible to alter	Possible to alter
Speed	Slow	Fast	Fast

1.2.4 Blockchain consensus

1.2.4.1 Proof of work

Proof of work [10] has been used in many blockchain technologies; it has been used in bitcoin which does a randomized "leader election". It represents the process that selects one of the miners to issue the next block. And ensure that all miners should follow this consensus protocol. The selection is performed in a randomized way which is proportional to the computational power of each miner. By making parties solve a computational puzzle to add a block. Prove of work is energy consumption, and it's slow, the average time to confirm a transaction could take up to 1 hour.

1.2.4.2 Proof of stake

In Proof of Stake, miners have to invest their computational resources, instead, a process runs that select randomly a leader proportionally to their stakes. PoS differ from other blockchains technologies as it does not rely on static authorities for the next block elector, but instead, the stakes changes over time which mean trust assumption evolves with the system which results a self referential blockchain [11].

The Stakeholder's distribution is taken directly from the ledger, where a randomized selection process will determine the stakeholder(s) which may append the next block(s). Therefore, PoS is an improvement on the PoW on both time and energy consumptions. This will result many challenges like, the blocks adding have no cost and the leader election process requires randomness taken from the ledger, therefore vulnerable to attacks.

1.2.4.3 Proof of authority

The Proof of Authority [12] relied on a limited number of block validators, therefore, the purpose of the validators is to protect the network. They are chosen as trusted entities. They verify blocks and transactions and act as moderators of the system. The validators do not stake coins just like Proof-of-Stake. They have their own reputation and losing this reputation will untrust them in the network.

In order to assign a validator, identity must be validated online and should be shared among networks. They should be consistent with their procedure to establish credentials among the network nodes. This will provide the network scalability and high-speed advantage, and it's effective in terms of energy consumption. Giving all this power to validators might reduce the degree of centralization with possible manipulation of the validators, which the thesis hasn't covered.

1.2.5 Blockchain mining

Blockchain mining is used to provide security to the network, this includes validating and confirming transactions which can be then added to the blockchain. There should be a limited number of supplies which will encourage users to enter a competition and obtain the block before others. [13]

There are a number of types of mining, there are individual mining which requires individual registration. Once the transaction is ready, a complex mathematical problem is given to these single users to solve and the solution finder will be rewarded, and then the transaction will be added to the blockchain. Pool mining is another type of mining used by a group of users who works together to validate transactions, and the reward is split among these users. The cloud mining type remove the need to provide software, but invest your processer power in a cloud operation, however, the reward is considered low in this type.

1.2.6 Blockchain incentive

The incentives methods are ways to encourage users to participate in the system. By solving a complex problem and finding the nonce, they are rewarded with system tokens that are valuable. This will keep the blockchain network operational and decentralized. An example is the Ethereum incentives, participating in the Ethereum network will reward the Ether which is Ethereum token-based. Which a user can trade with other stable coins. [14]

There is another type of incentive which is Staking, this way the participants contribute to transactions validations by staking their tokens, this is used heavily in Proof-of-Stake (PoS) blockchains The participants can be rewarded with more tokens, however, their assets will be locked for a certain amount of time and then they can redeem their assets back to their wallets.

1.2.7 Hashing algorithms

The hashing algorithms are mathematical tools that are used to convert any input with any length to an output of a fixed length. The purpose of it is to scramble the data in a deterministic way, provide security, and it is a one-way encryption algorithm meaning that the input cannot be derived from the output. A common use of these algorithms is to compare large sets of data easily. [15]

1.2.7.1 Secure hash algorithm

The Secure hash algorithm 256 in short (SHA-256) which is part of the Secure hash algorithm 2 (SHA-2) is a cryptographic algorithm that accepts input of any length. SHA-256 relies on a number of constants that define its behavior, for example, the constant 256 refers to the output digested size in bits.

1.2.7.2 Message digest algorithm

The Message digest algorithm in short (MD5) generated 128 bits of output. Recently it has been found that this algorithm suffers from some vulnerabilities and collisions. The reason why this algorithm is used in the thesis is its short output size, as the thesis tries to utilize the bandwidth as possible. And it is not a security breach to revert the hash, as the content of that hash is actually the packet itself.

1.2.8 The majority attack

This also known as a 51% attack, happens when a single entity or group of entities gain control over 50 percent of the hashing power of the blockchain. This will help them to block confirming new transactions, changing the order of the transaction, and effecting branching of the blockchain network by triggering a double-spending issue. [16]

1.2.9 Distributed denial-of-service attack

This attack's main objective is to overwhelm the system, server, or network with traffic that will lead to malfunctioning, it can be as incoming requests, messages, or connections. It can be fake packets just like the case of this thesis. This attack can tacket any system including websites, emails, banking, and any service rely on a server or network. The attacker can use and rent botnets, which is a hijack group of connected internet devices that can do attacks at a large scale. Distributed Denial-of-service attack (DDoS) can target many layers, it can target Application, Presentation, Session Transport, Network, Datalink, and Physical layers with various attack types. [17]

2. RELATED WORKS

2.1 Video Streaming

The caching distribution has been divided into two types of information-theoretic caching systems. The first type is centralized that relies on a couple of entities to share caches among other nodes, the second type is decentralized which distributes power. In the centralized the cached files are stored in users' devices, some of them store all the cached files into user devices [18], others have divided the data and distributed part of the data among users devices [19].

2.1.1 Wireless social caching

In this paper [20], they used D2D (Device-to-device) sharing to minimized the bandwidth and interaction between users and servers, which performed much better than regular topology connection. They have calculated that the priority of two users to meet many times is high within a time range. They focused on supplying the data to high users' likelihood of sharing cached data to other users. Users download caches from the wireless network and rely on random meetings to exchange caches between them using D2D connection.

2.1.2 Cooperative mobile devices

This work [18] evaluated the performance of mobile devices that establish a short-range link to neighbor mobile devices. These mobile devices form a cluster where they can exchanges packages between themselves fastly with minimum energy consumption. It implements a network coding between the devices in one cluster and between the clusters themselves, the main idea is to take a video file and split it into multiple streaming services and distribute these among the clusters, and a cooperation clusters network will be established. This technology is similar to

BitTorrent technology as each cluster will represent a seed. And it's vulnerable to Distributed Denial-of-service attacks (DDoS) and viruses.

2.1.3 Flixxo

Flixxo [21] was developed in 2016 by creating a distributed peer-to-peer a network where every user is a distributor of content. Flixxo provides a decentralized payment infrastructure where it replaces the middle-men layer costs and introduces a platform that empowers the community to be the distributors. An author is able to upload content and define the distribution rules, like the price and the incentives for the users, and the users will earn rewards by participating in the community and distributing. Flixxo facilitates smart contracts to provide peer-to-peer payment solutions, and the distribution rules are set into the smart contracts, this will help multiple authors to split the payment of a single content view to a different address by distributing Flixx token that follows ERC20 standardization.

2.1.4 MicroCast

Microcast [19] is a system that employs a peer-to-peer network to share video segments among users. It uses a WiFi network that each smart mobile device will connect to. Each user will download segments of the video from the internet and use the WiFi network to share these segments with other users. All mobile devices will be able to watch the same video at the same time. Microcast assumes that the connected users should trust each other, which means Microcast cannot be used anonymously and it is not secure. This will affect its scalability and usage as it would be hard to form trust groups to join the network. This thesis is using a close MicroCast process, but it adds security, scalability, and anonymity for the users.

2.1.5 Popcorn time

Popcorn Time utilizes BitTorrent technology [22] to provide video stream service, each user is defined as seed. A seed will be going to upload streaming video while watching and downloading. The method provides high-quality video streaming using a peer-to-peer network. However, the platform is vulnerable to the common BitTorrent security attacks like Distributed Denial-of-service attacks (DDoS) and viruses.

3. PROPOSED APPROACH

In this thesis, smart mobile devices have been used to create a peer-to-peer network. These devices connect to a shared WiFi network. Video segments are prepared and set in the network as we are ignoring network latency. Some of the nodes will login into the network as Verifiers and others as regular nodes. A consortium blockchain network is set where the verifier will contribute to adding blocks to the blockchain, distributing the blockchain, and making sure that security measurements are set and validated.

3.1 Structure

A peer-to-peer network in Figure 3.1 is constructed that consists of any smart devices. The smart devices represent the Nodes (peers) in the network. A special node called Verifier is responsible for the security measurements of the network, and responsible for adding blocks to the blockchain and distributing the blockchain among the nodes. A Node is going to connect to a streaming service or server to download video segments. Any node can join the network by Wifi without the need to connect to stream service or to have an internet connection. Once it connects to the network, it will receive the blockchain from the verifiers, and then it can lookup segments inside a table called Owner Hashed Table (OHT) inside blockchain blocks, and request segments from their owner nodes. This will result in a wise usage of the network resources and faster video streaming with high quality and minimum internet bandwidth usage.

The blockchain network follows Proof-of-Authority consensus, the verifier nodes are the validators and the moderators of the system which are responsible for validating the blocks and transactions in the network. This consensus will force the verifiers to act honestly as have their own reputation at risk, and their identity is registered and revealed in the network.

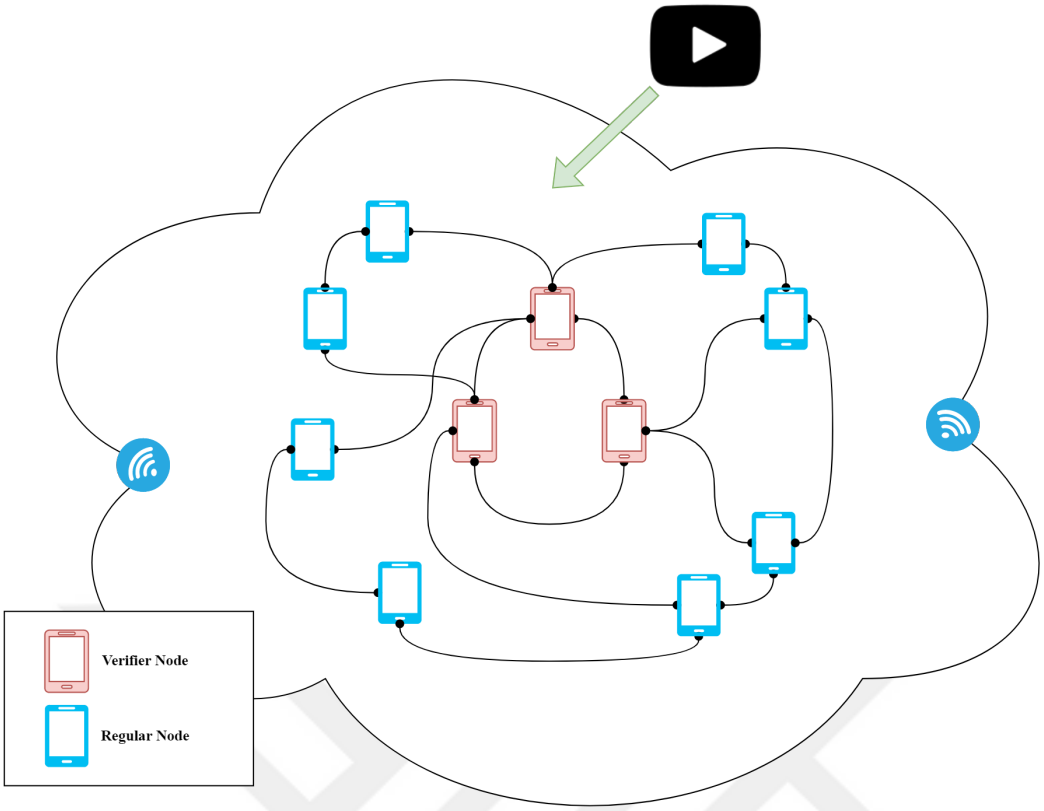

Figure 3.1 : Network Structure.

3.1.1 Block structure

In Figure 3.2, each block contains transactions, the Owner Hashed-Segments Table (OHT), and the Malware Tolerance Table (MTT). All of these are hashed when adding the block to the blockchain using the Message-Digest Algorithm (MD5) function, and the value is stored inside the block itself. When building the next block, this hashed value will be included alongside the transactions, the Owner Hashed-Segments Table (OHT), and the Malware Tolerance Table (MTT) of the next block, this will build up the chain of the blockchain. The block header nonce is not included as the block is not mined. The regular block header is implicitly included which holds the version, the timestamp, the previous block hash which is important in this thesis, and the Merkle root hash to protect the transactions.

It is important to mention that the block does not contain the segment itself but it contains the hash value of the segment, this value is fixed in size and a result of running the Message-Digest Algorithm (MD5) function. The choice of this hashing algorithm was based on utilizing the bandwidth as possible as it output 128 bits.

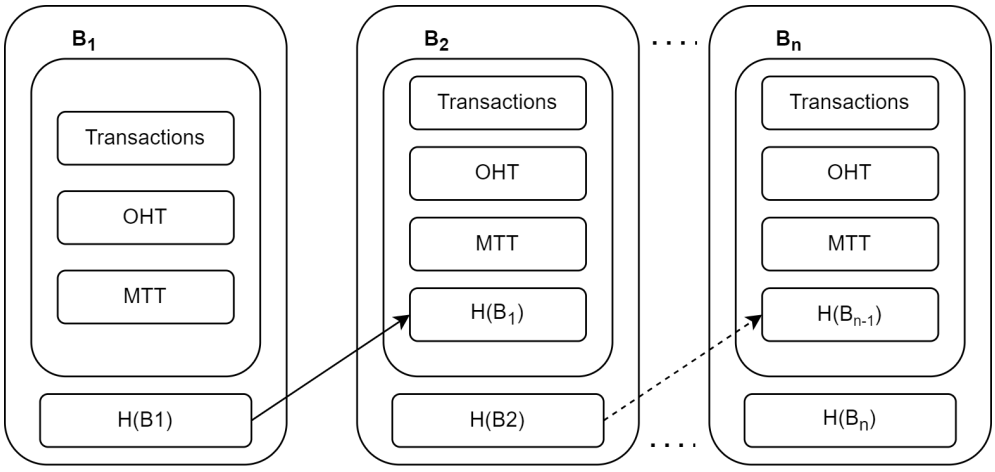

Figure 3.2 : Block Structure.

3.1.2 Owner hashed-segments table

The Owner Hashed-Segments Table (OHT) contains the segments hashes that have been downloaded by each basic node in the network during the video streaming. The data structure is a dictionary where the keys represent the name of the segment as demonstrated in Figure 3.3. The symbol N is the function of getting the segment name, and P is the segment as a packet. The Owner Hashed-Segments Table (OHT) in Figure 3.3 also contains the IP address of the basic node owners, each one of them contain a list of segment names functions, this list represents the segments owner, a segment could exist in one or more owner lists.

The Owner Hashed-Segments Table (OHT) is important as it provides a second security layer. It keeps the packets immutable along with the network as each node can validate the integrity of the segment by comparing and validating the hash of it with the hash of the segment that has been received. For checking the validity of segment using the first part of the Owner Hashed-Segments Table (OHT) it costs O(1). For the second part, it cost O(1) x O(n) as it has to look up the owner in the dictionary, then it has to skim the list to find the wanted packet.

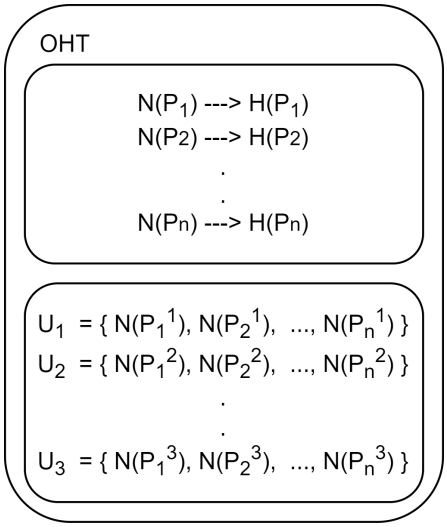

Figure 3.3 : Owner Hashed-Segments Table.

3.1.3 Malware tolerance table

The Malware Tolerance Table (MTT) in Figure 3.4 is a table is where the verifier marks a node as malicious. The network will define a tolerance limit and the verifiers will keep monitoring nodes' contributions. If the limit has reached for a node whether by sending malicious segments or by trying Distributed Denial-of-service attack (DDoS), the verifier is going to block this node contribution. The Malware Tolerance Table (MTT) will be added in the block and then in the blockchain, and the table will be shared among all verifiers in the network.

The Malware Tolerance Table (MTT) is a dictionary that contains the nodes' IPs as the dictionary key, and the tolerance value as the dictionary value. It required O(1) to look up the dictionary and know the tolerance value for a certain node.

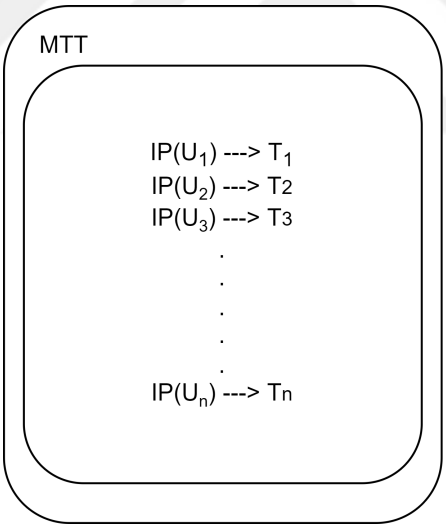

Figure 3.4 : Malware Tolerance Table.

3.2 Security

It is important for the consortium blockchain network to establish a single entity that is responsible for the security of the network, this may result in less decentralization, however, the nodes are not relying on a centralized stream service anymore, and consortium blockchain network is faster, more efficient, and more cost-effective than public blockchains, which require a lot of time and energy to validate transactions

In the thesis implementation, each downloaded segment by the regular node will be sent to the Verifier node to be confirmed. In order for the verifier to add it as a block or transaction into the blockchain, the Verifier requires to download the video segment from the online streaming service, and hash this segment and compare it with the hash of the segment sent by the regular node

The Verifier will group these segments and patch them together into one block, and add this block to the consortium blockchain network. Then the verifier will distribute the blockchain to all the nodes available in the network. This will help the Verifiers in the network to get the up-to-date blockchain and will help regular nodes to look into the blockchain to find missing segments for them, and it will request this segment from its owner regular nodes.

As the Verifier nodes are being authenticated and their identity will be revealed, this will help to protect from 51% attack. The Malware Tolerance Table (MTT) has been employed to detect malicious nodes contributions as discussed in 3.1.3 and 3.3.3. The Malware Tolerance Table (MTT) has a significant usage of protecting from A distributed Denial-of-service attack (DDoS) as the malicious node will be announced to all the Verifiers, and all the Verifiers will block its contribution. In addition, there is more than one entity as a verifier the nodes will not rely on a single centralized entity, which will help against Distributed Denial-of-service attack (DDoS) too.

3.3 Algorithm

3.3.1 Basic node process

In Figure 3.5, a basic node will login into the peer-to-peer network, it's going to broadcast and announce its IP into the network. This will notify the verifiers that this node has logged into the network and the verifiers in return will going to send the blockchain to that Node. This will help the node to discover the already downloaded segments by looking up the Owner Hashed-Segments Table (OHT) inside the blockchain, therefore, it will be able to request the segments from their basic node owners. Once this basic node receives the segment from the owner basic node, it's going to validate it by comparing its hash using MD5 with the existing hash inside the blockchain for that segment, if they are equal then the segment has not been altered.

The basic node will connect to the streaming service and request the missing segments, the successfully downloaded segments will be hashed using MD5 by the basic node, and these hash will be sent to the verifiers in the network, this will be considered as a contribution from the basic node.

The Algorithm 1 describes Basic Node algorithm steps using Pseudocode. Three main functions have been defined. The **SendSegmentToVerifier** function inputs are the segment and the verifier recipient, the segment is fetched from the local database by the segment name. The local database is the place where the node stores the downloaded or requested segments. The **SendSegmentRequest** function is where the node requests the missing segment, the node was able to find the missing segment and the owner node by looking inside the Owner Hashed-Segments Table (OHT), it sends the owner the request. The **SendSegmentToNode** function is called when the node receives the missing segment request from the requested node, and it responsed back the requested segment by fetching it from its local database. The **ReceiveSegment** function triggered when the node receives the requested missing segment. It hashes this segment using MD5 and compares it with the segment hash stored inside the Owner Hashed-Segments Table (OHT). If it's valid then it's not a malicious segment, and it can add it to its local

database. It will also send this segment to the verifier so the verifier can add him as the owner of this segment.

The first loop keeps running while the node is connected to the peer-to-peer network. It checks for missing segments and downloads new segments from the streaming service. In the second loop the node listens for requests for missing segments from other nodes in the network.

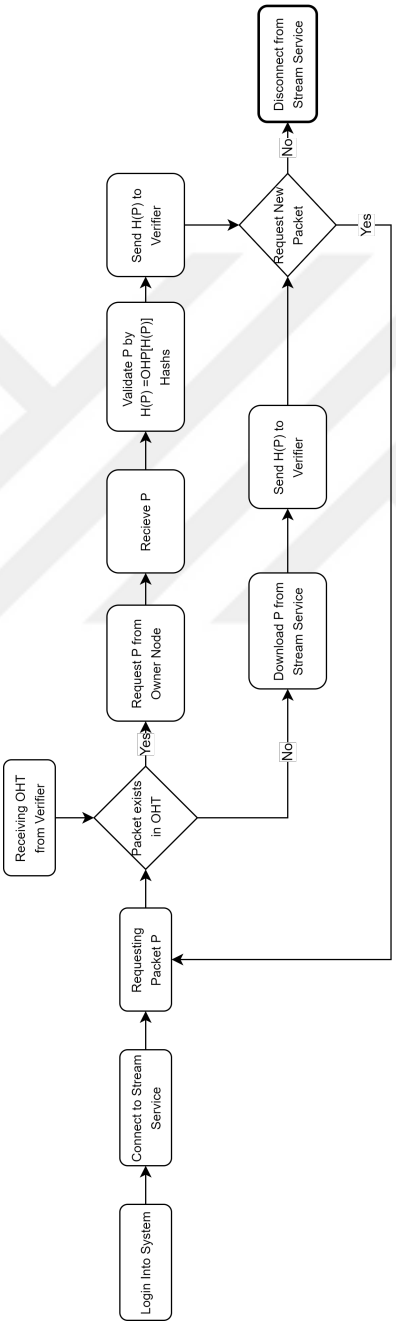

Figure 3.5 : Node Flow Diagram.

3.3.2 Basic node workflow

The basic node workflow in Figure 3.6 describes the interaction between two nodes. The time direction is from top to bottom. Node1 has logged in to the system and broadcasts its IP. Node1 has received the latest blockchain from the verifier. The node wants packet 1 and finds out that the packet P1 exists in the blockchain and the owner of it is Node2, it sends a request to Node2 to inform it of her request. Once it receives the request packet P1, it is going to hash it using Message-Digest Algorithm (MD5) and validate that its hash is equal to the packet P1 hash that is stored inside the Owner Hashed-Segments Table (OHT) in the blockchain. If it is valid, Node 1 is going to send that hash to the verifier to notify it of its ownership.

For the Node2 case, after login into the system and broadcasting its IP. It is going to download packet P1 from the streaming service as it does not exist in the Owner Hashed-Segments Table (OHT), and it will send its hash to the verifier to register its ownership. After that Node2 has received the Node1 request for packet P1 and it is going to send it over to Node 1. After a while, it got the latest blockchain which says that Node1 is the ownership of packet P1 too.

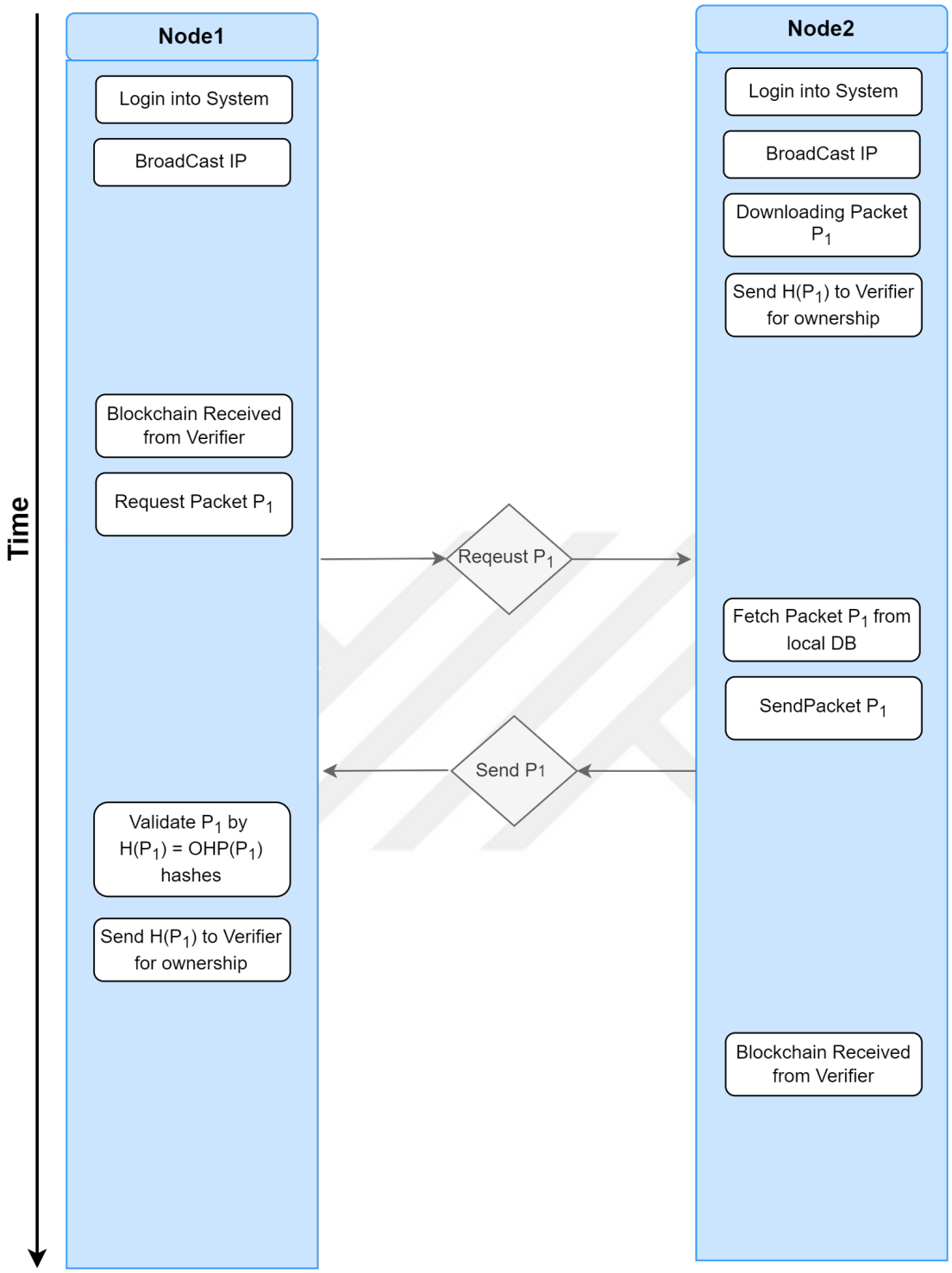

Figure 3.6 : Basic Node WorkFlow.

Algorithm 1 Basic Node Algorithm

Input Verifier, Node, StreamService, LocalDB, BlockChain
Output LocalDB

```
1: function SENDSEGMENTTOVERIFIER(verifier, segment)
2:     segmentHash ← H(segment)
3:     UDP.Send(verifier, segmentHash, segment.Name)
4: end function
5: function SENDSEGMENTREQUEST(segmentName)
6:     node ← BlockChain.OHT.OwnerSegment[segmentName]
7:     UDP.Send(node, segmentName)
8: end function
9: function SENDSEGMENTTONODE(segmentName, node)
10:    segment ← LocalDB[segmentName]
11:    UDP.Send(node, segment)
12: end function
13: function RECEIVESEGMENT(segment)
14:    segmentHash ← H(segment)
15:    ohtSegmentHash ← BlockChain.OHT.HashedSegments[segment.Name]
16:    if segmentHash = ohtSegmentHash then
17:        LocalDB.Save(segment)
18:        SendSegmentToVerifier(verifier, segment)
19:    end if
20: end function
21: while Connected do
22:    requestableSegments ← BlockChain.OHT.HashedSegments.Names
23:    - LocalDB.Segments.Names
24:    for Each segment in requestableSegments do
25:        SendSegmentRequest(segment.Name)
26:    end for
27:    newSegment ← StreamService.Get()
28:    LocalDB.Add(newSegment)
29:    SendSegmentToVerifier(verifier, segment)
30: end while
31: while ListeningToNodes(Node) do
32:    segment, requestSegmentName ← UDP.Recieve()
33:    if segment ≠ NULL then
34:        ReceiveSegment(segment)
35:    else
36:        if requestSegmentName ≠ NULL then
37:            SendSegmentToNode(requestSegmentName, Node)
38:        end if
39:    end if
40: end while
```

3.3.3 Verifier node process

In Figure 3.7, the verifier node will log in to the network, it's going to broadcast and announce its IP into the network. This will notify the verifiers and the basic nodes of its existence. The verifier node is responsible for validating the segments, inserting them into the blockchain, distributing the blockchain to other nodes, and monitoring nodes for any malicious behavior.

It's required for the verifier node to have a connection to the streaming service in order to validate the segments. The verifier node will receive the segment hash from the basic node, it will download this segment from the streaming service and hash it using MD5 algorithm, and then it will compare it to the hashed segment sent by the basic node. After validating the segment, the verifier node is going to insert this hashed segment with its basic node owner IP into the Owner Hashed-Segments Table (OHT) table, and insert it into a transaction inside a block, once this block reaches the maximum number of transactions, the verifier node will insert this block into the blockchain, and it's going to distribute the updated blockchain to all nodes.

If the verifier node found that the hashed segment send by the basic node is invalid and doesn't match the hashed segment downloaded from the streaming service. It's going to increase the Malware Tolerance Table (MTT) for that node, after exceeding the configured tolerance, it's going to block that node, which means it will no more receive hashed segments, and it will not accept its contribution to the blockchain anymore.

The Algorithm 2 describes Verifier Node algorithm steps using Pseudocode. It defines two functions, the **SendBlockchainToNodes** function broadcasts the blockchain to all nodes and verifiers, this will ensure keeping the network up to date. The **ReceiveSegmentHashFromNode** function is where the verifier receives a segment from a node to be added to the blockchain. First, it downloads the segment from the streaming service and hashes it using MD5 algorithm. Second, it compares this hash with the hash value sent by the node, if it's valid then it adds this segment hash to the Owner Hashed-Segments Table (OHT) and it adds the segment name to the Owner table and marks the node as the owner of the segment, it keeps stacking these data until it reaches the possible limit of adding a new block. The limit has

been defined as 100 segments per block. After reaching the limit it adds the Owner Hashed-Segments Table (OHT) to a block and inserts this block into the blockchain. Afterward, it calls **SendBlockchainToNodes** to broadcast the blockchain to all users. If the hash validation has failed, the verifier will increase the malware tolerance for that node. It will keep tracking this value until it bypasses the **MaximumTolerance**, then the verifier will block that node and mark it as malicious, and it won't receive any new segments from it anymore. In the loop, the verifier keeps listening to the nodes for new segments.

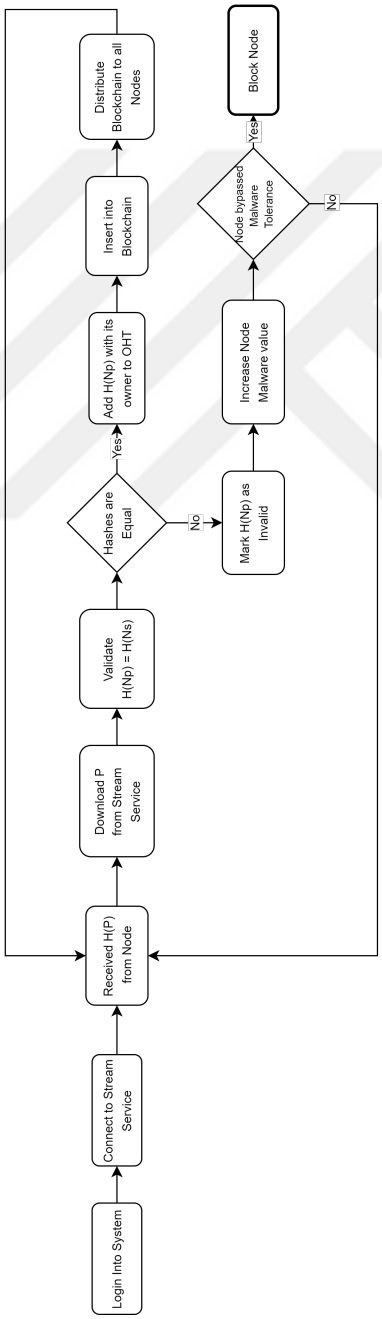

Figure 3.7 : Verifier Flow Diagram.

3.3.4 Verifier node workflow

The verifier workflow in Figure 3.8 describes the interactions between the verifier node and the basic node. The Node1 will login into the system and broadcast its IP. It is going to download packet P1 from the streaming service, then it is going to hash it using Message-Digest Algorithm (MD5) and send the hash to the verifier. It is going to do the the same process once it receives a requested packet P2 from Node2.

After The verifier login to the system and broadcasts its IP, it receives the hash value of packet P1 that has been sent from Node1. It is going to download the packet from the stream service SP1, then it is going to hash it using Message-Digest Algorithm (MD5) and compare this hash with the hash sent by Node1. If they are equal then it is going to insert Node1 ownership into the Owner Hashed-Segments Table (OHT). The verifier will also receive an already existing packet P2 from Node1, therefore, it is going to look up inside the Owner Hashed-Segments Table (OHT) and pick up its hash and compare it with the send has of packet P2. If they are equal it is going to insert the ownership of packet P2 to be both Node1 and Node2 and update the Owner Hashed-Segments Table (OHT). Once the verifier adds the block to the blockchain it is going to send it over to all nodes.

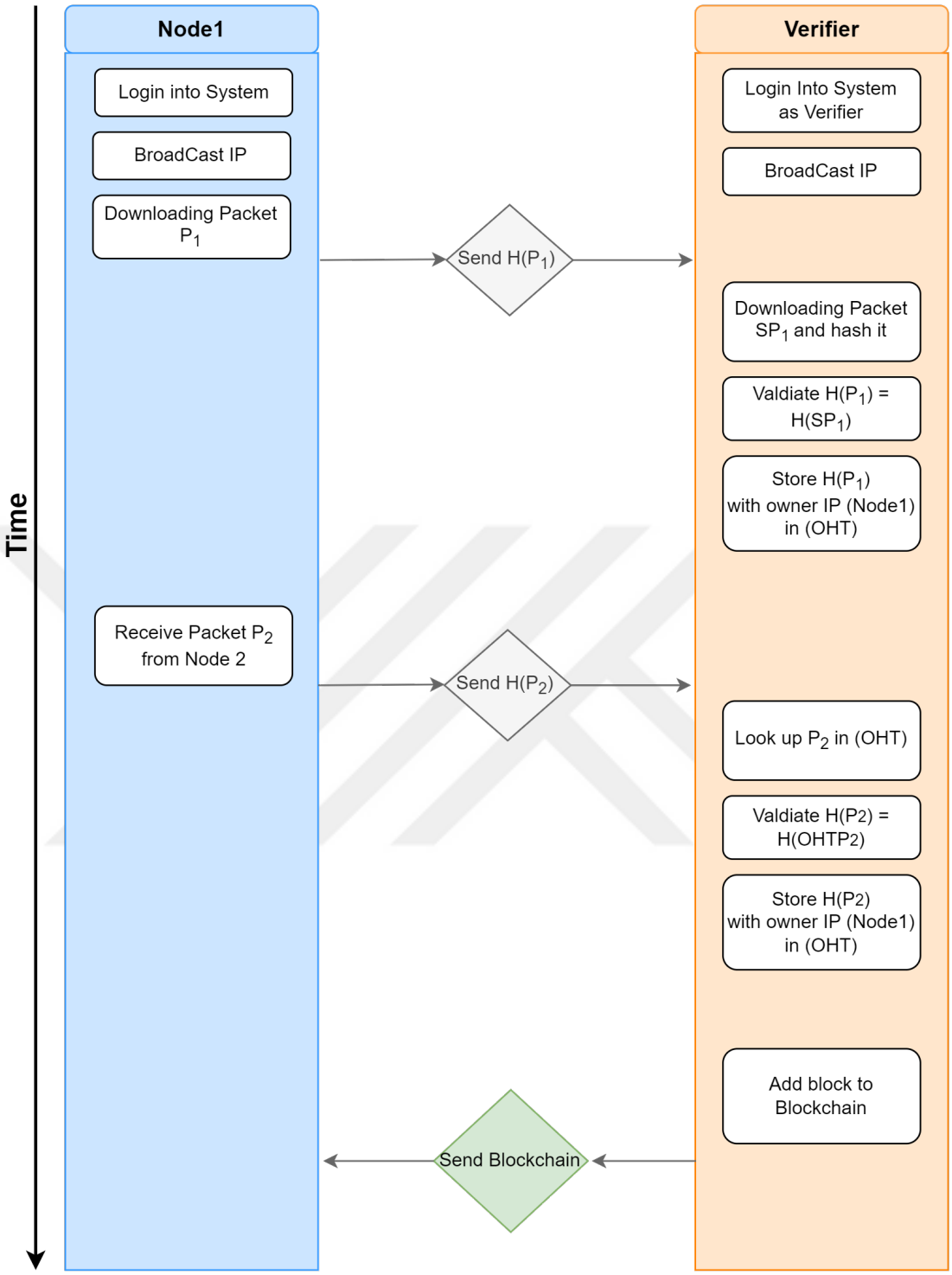

Figure 3.8 : Verifier WorkFlow.

Algorithm 2 Vertifier Node Algorithm

Input BlockChain, MTT, StreamService
Output BlockChain

```
1: function SENDBLOCKCHAINTONODES
2:     UDP.BroadCast(Blockchain)
3: end function
4: function RECEIVESEGMENTHASHFROMNODE(node, segmentHash, segmentName)
5:     streamServiceHash ← StreamService.Get(segmentName)
6:     streamServiceSegmentHash ← H(streamServiceHash)
7:     if streamServiceSegmentHash = segmentHash then
8:         BlockChain.OHT.HashedSegments[segmentName] ← segmentHash
9:         BlockChain.OHT.OwnerSegment[segmentName] ← node
10:        if BlockChain.CanAddBlock() then
11:            BlockChain.MTT ← MTT
12:            BlockChain.AddBlock()
13:            SendBlockchainToNodes()
14:        end if
15:    else
16:        MTT[node].IncreaseMalwareValue()
17:        if MTT[node].Tolerance > MaximumTolerance then
18:            UDP.Block(node)
19:        end if
20:    end if
21: end function
22: while ListeningToNodes(Node) do
23:    node, segmentHash, segmentName ← UDP.Recieve()
24:    if segmentHash ≠ NULL then
25:        ReceiveSegmentHashFromNode(node, segmentHash, segmentName)
26:    end if
27: end while
```

4. EXPERIMENTAL DATA ANALYSIS AND RESULTS

Tests have been obtained by implementing software. A WiFi network has been set, and peer-to-peer network has been established. The WiFi network represents the peer-to-peer network which will be the infrastructure for the consortium blockchain network. The nodes have been defined in the network, some of them will have to log in using a simple authentication method, this is a required step for the verifiers so their identity can be validated, as verifiers should not act maliciously in the network. The basic node will not have to login into the system. In section 3.3, we have demonstrated the flow and processes of the system.

4.1 Data Set

The experiment used a youtube video to prepare the data set, the youtube cached files that are required to run the youtube video offline have been downloaded. These cached files have been prepared with 3 sizes, 15 KB, 30 KB, and 50 KB to help test the system performance on various segment sizes. The table 4.1, demonstrate the test-bed parameters for the tests. The experiment used 10 nodes. One as a verifier, and the rest as basic nodes. The overall video size is 3 GB that streamed in a WiFi connection with property of 2.4GHz-802.11g.

Table 4.1 : Test-bed parameters.

Description	Value
Number of Nodes	10
Shared File Size	3 GB
Packet sizes	15K, 30K, 50K
Hashing Algorithm	MD5
Wireless Connection	2.4GHz-802.11g

4.2 Experimental Data Analysis

User Datagram Protocol (UDP) has been used to avoid overwhelming the usage of bandwidth, as UDP does not ensure that all data packets arrive in order and does not require establishing handshakes with the target node. This is important to keep streaming synced on the network between the nodes as possible.

As the streamed video size is 3 GB, three tests have been conducted per segment size. The first test has used a 15 KB segment size with 200000 segments, the second test has used a 30 KB segment size with 100000 segments and the third test has used a 50 KB segment size with 60000 segments. The segment size decision has been based on the UDP characteristic of a maximum packet size of 65,535 bytes. These three tests were conducted to find the minimum network latency.

In these experiments, the network latency was minimized as the focus is to calculate the time cost of the verifier to add a block to the network and validate the segment. The time cost of the verifier consist of the time that requires the segment to be transmitted to the verifier, the verifier validating the segment and adding the hash of the segment as a transaction to the block, the time requires the verifier to update the Owner Hashed-Segments Table (OHT) inside the block with the new segment owner IP, and the time cost of adding a new block to the blockchain. The verifier will keep holding the segments until they reach 100 segments, and then it will release the block to the chain.

The Figures 4.1, 4.2, and 4.3 demonstrate the time cost of adding new block into the blockchain. Nodes have connected to this verifier and send the segments using UDP protocol with the peer-to-peer network. The Figures contain 3 histograms per test instance. The cost time includes validating the segments, hashing the segments using MD5, updating the Owner Hashed-Segments Table (OHT) table, adding transactions to the block, and inserting the block to the consortium blockchain.

Figure 4.1 shows a test instance of 15 KB segment size, with 2000 blocks, each block will hold 100 segments. Each bar represents the number of blocks that have been added in a specific time range, for instance, the verifier has added 250 blocks to the blockchain in the 60-70 millisecond range. Likewise, Figure 4.2 shows the test instance

of 30 KB segment size with 1000 blocks being added to the blockchain. And Figure 4.3 shows the test instance of 50 KB segment size with 600 blocks added to the blockchain.

The cost time requires for the verifier to validate the segment and add it to the blockchain is considered low, which will not have an effect on the performance of the transmission. This proves that this method is considered fast and secure.

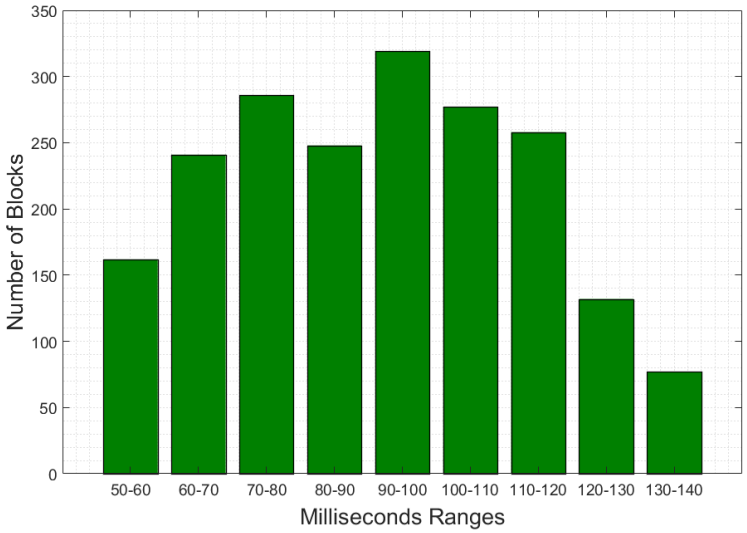

Figure 4.1 : Verifier Block Insert for 15KB with UDP.

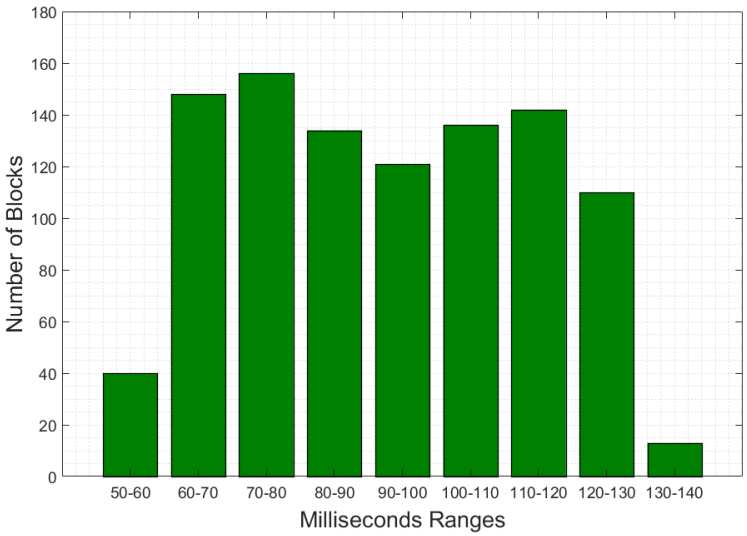

Figure 4.2 : Verifier Block Insert for 30 KB with UDP.

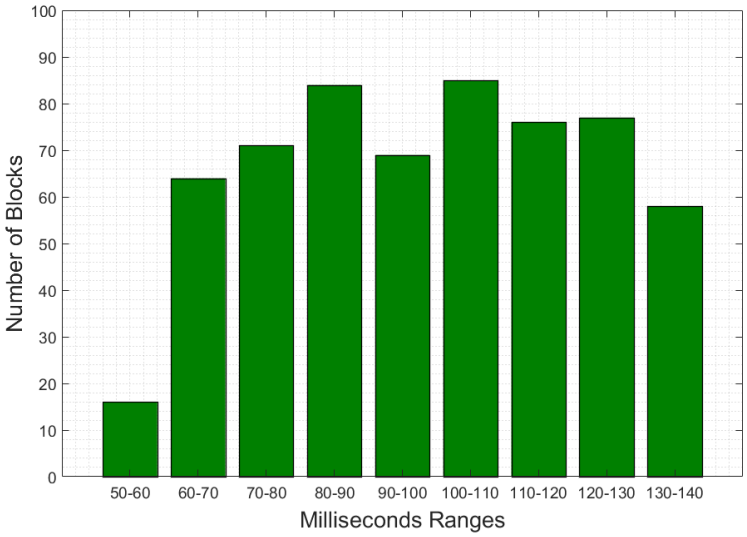

Figure 4.3 : Verifier Block Insert for 50 KB with UDP.

5. CONCLUSIONS AND RECOMMENDATIONS

The work presents a secure and fast distribution method using a decentralized peer-to-peer network. It uses Blockchain to facilitate a secure layer for transactions. It tries to solve the problem of video streaming in the same network and try to utilize the bandwidth as possible.

Incentive methods have been put as future work, adding these incentives will help to encourage the users to join and participate in the network. It will enforce honesty and help to reduce malicious behaviors among the nodes. Selecting verifiers methods can be studied in depth to ensure security, as we are relying on these nodes to validate the network, the current method require the verifier to authenticate himself and share his identity, however, this can be improved in a better way.

REFERENCES

[1] **Google**. Youtube for Press.

[2] **Nofer, M.**, **Gomber, P.**, **Hinz, O. and Schiereck, D.** (2017). *Blockchain*, volume 59, Springer.

[3] **Nakamoto, S.** *et al.*, (2008), Bitcoin: A peer-to-peer electronic cash system.(2008).

[4] **Lin, I.C. and Liao, T.C.** (2017). A survey of blockchain security issues and challenges., *Int. J. Netw. Secur.*, *19*(5), 653–659.

[5] **Merkle, R.C.** (1987). *A digital signature based on a conventional encryption function*, *Conference on the theory and application of cryptographic techniques*, Springer, pp.369–378.

[6] **Kim, S.**, **Kwon, Y. and Cho, S.** (2018). *A survey of scalability solutions on blockchain*, *2018 International Conference on Information and Communication Technology Convergence (ICTC)*, IEEE, pp.1204–1207.

[7] **Buterin, V.** (2015). On public and private blockchains, *Ethereum blog*, *7*.

[8] **Dib, O.**, **Brousmiche, K.L.**, **Durand, A.**, **Thea, E. and Hamida, E.B.** (2018). Consortium blockchains: Overview, applications and challenges, *International Journal On Advances in Telecommunications*, *11*(1&2).

[9] **Sayeed, S. and Marco-Gisbert, H.** (2019). Assessing blockchain consensus and security mechanisms against the 51% attack, *Applied Sciences*, *9*(9), 1788.

[10] **Liu, D. and Camp, L.J.** (2006). *Proof of Work can Work.*, *WEIS*, Citeseer.

[11] **Kiayias, A.**, **Russell, A.**, **David, B. and Oliynykov, R.** (2017). *Ouroboros: A provably secure proof-of-stake blockchain protocol*, *Annual International Cryptology Conference*, Springer, pp.357–388.

[12] **Pass, R. and Shi, E.** (2017). *Rethinking large-scale consensus*, *2017 IEEE 30th Computer Security Foundations Symposium (CSF)*, IEEE, pp.115–129.

[13] **Wang, W.**, **Hoang, D.T.**, **Hu, P.**, **Xiong, Z.**, **Niyato, D.**, **Wang, P.**, **Wen, Y. and Kim, D.I.** (2019). A survey on consensus mechanisms and mining strategy management in blockchain networks, *Ieee Access*, *7*, 22328–22370.

[14] **He, Y.**, **Li, H.**, **Cheng, X.**, **Liu, Y.**, **Yang, C. and Sun, L.** (2018). A blockchain based truthful incentive mechanism for distributed P2P applications, *IEEE access*, *6*, 27324–27335.

[15] **Li, P.**, **Shrivastava, A.**, **Moore, J. and König, A.** (2011). Hashing algorithms for large-scale learning, *Advances in neural information processing systems*, *24*.

[16] **Dey, S.** (2018). *Securing majority-attack in blockchain using machine learning and algorithmic game theory: A proof of work*, *2018 10th computer science and electronic engineering (CEEC)*, IEEE, pp.7–10.

[17] **Lau, F.**, **Rubin, S.H.**, **Smith, M.H. and Trajkovic, L.** (2000). *Distributed denial of service attacks*, *Smc 2000 conference proceedings. 2000 ieee international conference on systems, man and cybernetics.'cybernetics evolving to systems, humans, organizations, and their complex interactions'(cat. no. 0*, volume 3, IEEE, pp.2275–2280.

[18] **Pedersen, M.V.**, **Fitzek, F.H. and Larsen, T.** (2008). *Implementation and performance evaluation of network coding for cooperative mobile devices*, *ICC Workshops-2008 IEEE International Conference on Communications Workshops*, IEEE, pp.91–96.

[19] **Keller, L.**, **Le, A.**, **Cici, B.**, **Seferoglu, H.**, **Fragouli, C. and Markopoulou, A.** (2012). *Microcast: Cooperative video streaming on smartphones*, *Proceedings of the 10th international conference on Mobile systems, applications, and services*, pp.57–70.

[20] **Karmoose, M.**, **Cardone, M. and Fragouli, C.** (2018). Simplifying wireless social caching via network coding, *IEEE Transactions on Communications*, *66*(11), 5512–5525.

[21] **Team, F.** (2018). *Community video distribution*, https://www.allcryptowhitepapers.com/wp-content/uploads/2018/05/Whitepaper_0.6.pdf.

[22] **Cohen, B.** (2003). *Incentives build robustness in BitTorrent*, *Workshop on Economics of Peer-to-Peer systems*, volume 6, Berkeley, CA, USA, pp.68–72.

CURRICULUM VITAE

Name Surname: Anas MHAISH

EDUCATION:

- **B.Sc.:** 2013, Damascus University, Faculty of Computer Engineering, Department of Artificial Intelligence
- **M.Sc.:** 2022, Istanbul Technical University, Informatics Institute, Computer Science Department

PUBLICATIONS, PRESENTATIONS AND PATENTS ON THE THESIS:

- **Sueda Guzey, Gunes Karabulut Kurt, Anas Mhaish, Enver Ozdemir, Nasim Tavakkoli** Secure Device-to-Device Caching with Blockchain, *IEEE Internet of Things Journal*