Optimization algorithms for video service delivery

Emad Mohamed Abd Elrahman Abousabea

Ecole Doctorale EDITE

Thèse présentée pour l'obtention du diplôme de

Docteur de Télécom SudParis

Doctorat conjoint Télécom SudParis et Université Pierre et Marie Curie

Spécialité:

Informatique et Télécommunication

Par :

Emad ABOUSABEA

Titre

Optimization Algorithms for Video Service Delivery

Soutenue le 12/09/2012
Devant le jury composé de :

Mme Hassnaa MOUSTAFA	**Rapporteur**	**France Telecom – Orange Labs**
Mr Ken CHEN	**Rapporteur**	**Institut GALILEE, Univ. Paris 13**
Mme Houda LABIOD	**Examinateur**	**Telecom ParisTech**
Mme Veronique VEQUE	**Examinateur**	**Faculté des Sciences d'Orsay**
Mr Guy PUJOLLE	**Examinateur**	**UPMC – Paris 6**
Mr Hossam AFIFI	**Directeur de Thèse**	**Télécom SudParis**

Thèse n°: 2012TELE0030

Abstract

The aim of this thesis is to provide optimization algorithms for accessing video services either in unmanaged or managed ways. We study recent statistics about unmanaged video services like YouTube and propose suitable optimization techniques that could enhance files accessing and reduce their access costs. Moreover, this cost analysis plays an important role in decision making about video files caching and hosting periods on the servers.

Our investigation and auditing on the servers statistics draw our attention to trends between consumer's culture, social relations and content locations. This could help in designing management and optimization algorithms for building optimized social networks. Also, it could return high gains to service providers by maximizing the profits and minimizing the costs.

Under managed video services called IPTV, we conducted experiments for an open-IPTV collaborative architecture between different operators. This model is analyzed in terms of CAPEX and OPEX costs inside the domestic sphere. Moreover, we introduced a dynamic way for optimizing the Minimum Spanning Tree (MST) for multicast IPTV service. In nomadic access, the static trees could be unable to provide the service in an efficient manner as the utilization of bandwidth increases towards the streaming points (roots of topologies). Hence, we select some new nodes (the future roots) to be streaming points based on their positions and we optimize their links with a proposed Virtual Node (VN). Then, we apply the MST on the virtual topology to calculate minimum costs for the new roots. Finally, we compare the complexity of the MST algorithm based on this virtual topology in the cases of heuristic and non-heuristic methods.

We study reliable security measures in video streaming based on hash chain methodology and propose a new algorithm. Then, we conduct comparisons between different ways used in achieving reliability of hash chains based on generic classifications. In terms of overhead complexity and packet loss rate, we use analytical and simulation methods to compare our proposed hybrid technique called *'window based hash chain with redundancy codes'* with other techniques. Our mechanism allows

the IPTV stream receivers to recover a certain amount of data in case of packet loss so as not to break the hash chain links.

Moreover, so as to assure a high degree of security using hash chain reliability, we integrate signature based window mechanism with redundancy codes. This way is compared against simple packet-based (*PB*) and block-based (*BB*) methods. Our algorithm (*window-based WB*) gives a high degree of reliability and significantly less bytes of overhead than the other methods.

In conclusion, we believe that the different contributions proposed in this thesis for both managed (*i.e. IPTV*) and unmanaged (*i.e. YouTube*) video services will improve service customization and reliability for video access and business service design algorithms in the future. Also, they could ease the evolution of next generation multimedia to a hybrid video service (i.e. the merging of managed and unmanaged).

Key words: *Video Sharing Servers, Optimization, Caching Videos, IPTV Reliability, Hash Chains.*

Thesis Publications

Book Chapters

- Emad Abd-Elrahman and Hossam Afifi; *''Chapter 2: Open-IPTV Services and Architectures''*, Book: *(Media Networks: Architectures, Applications and Standards)*, CRC Press Taylor & Francis Group, May 2012, pp.29-56.

Journals

- Emad Abd-Elrahman, Mohamed Boutabia and Hossam Afifi; *''Hash Chain Links Resynchronization Methods in Video Streaming Security: Performance Comparison''* , Journal of Mobile Multimedia JMM, Rinton Press Vol.7 No.1&2 April 6, 2011, pp.89-112.

Conferences

- Emad Abd-Elrahman and Hossam Afifi, *"Optimization of File Allocation for Video Sharing Servers"*, NTMS 3rd IEEE International Conference on New Technologies, Mobility and Security (NTMS2009),Cairo-Egypt , 20-23 Dec 2009, pp.1-5.
- Mohammed Boutabia, Emad Abd-Elrahman and Hossam Afifi, *"A Hybrid Mobility Mechanism for Heterogeneous Networks in IMS"*, IEEE ICME 2010, 19-23 July 2010, pp.1570-1575.
- Emad Abd-Elrahman, Mohammed Boutabia and Hossam Afifi, *"Video Streaming Security: Reliable Hash Chain Mechanism Using Redundancy Codes"*, ACM 8th International Conference on Advances in Mobile Computing and Multimedia (MOMM 2010) Paris, France, 8-10 Nov 2010, pp.69-76.
- Emad Abd-Elrahman, Mohamed Abid and Hossam Afifi, *"Video Streaming Security: Window-Based Hash Chain Signature Combines with Redundancy Code - YouTube Scenario as an Internet Case Study''*, The IEEE International Symposium on Multimedia (ISM2010) Taichung, Taiwan, 13-15 Dec 2010, pp.33-40.
- Emad Abd-Elrahman and Hossam Afifi, "*Moving to the Cloud: New Vision towards Collaborative Delivery for Open-IPTV"***,** The 10th International Conference on Networks ICN2011 in conjunction with GlobeNet 2011**,** St. Maarten, The Netherlands Antilles, 23-28 Jan 2011, pp.353-358.
- Emad Abd-Elrahman, Tarek Rekik and Hossam Afifi; *''Optimization of Quality of Experience through File Duplication in Video Sharing Servers",* the ACM *EuroITV2012, UP-TO-US workshop* July 2012, pp.286-291.

Research Reports

- E.Abd-Elrahman, W.BenAmeur and H.Afifi *"Optimization of File Allocation for Video Sharing Servers"* Rapport de recherché: n°:09002-RS2M, April 2009, Library of Telecom SudParis.

Acknowledgements

First, I would like to express my heartfelt gratitude to my supervisor Prof. Hossam Aifi for his academic advices and directions, constant encouragement and his support during my research and stay in France.

Second, I would like to thank all members of my PhD defense jury: Dr. Hassnaa Moustafa, Prof. Ken Chen, Dr. Houda Labiod, Prof. Véronique Veque and Prof. Guy Pujolle.

Third, many thanks to Prof. Walid Ben-Ameur for his valuable discussions and advices in optimization techniques.

Fourth, The staff of TMSP; particularly, I want to thank: Dr. Abdallah M'hamed, Dr. Vincent Gauthier, . . . and especially our head of department RS2M Prof. Djamal Zeghlache for their helpful suggestions and advices. Many thanks go to Mme Valerie Mateus for her patience and support.

Fifth, I would confirm my thanks to my friends from TMSP, especially RS2M department: Abid, Boutabia, Soua, Amira, Haykel, Ashish, Ahmad, Chedhly, Ghazi, Aroua, Teck, Alexis, Didi, Farouk, Songbo, Mamadou and Youness.

Sixth, I cannot forget to thank my colleagues and staff at National Telecommunication Institute NTI-Egypt especially Prof. Magdy El-Soudani and computer department staff, thanks for all.

Finally, I am indebted to many people during my research and my life. The most important is my family, I am extremely grateful for my parents, who have sacrificed themselves to give my brothers and me the best education. Their unreserved love and support for these many years are what make this thesis possible. For this, I believe they deserve this degree more than anyone else does. Also, to my small family: my wife 'Marwa', my daughter 'Mariam' and my son 'Mohamed' many thanks.

List of Figures

List of Tables

List of Contents

Chapter 1: Introduction

This thesis aims to investigate some optimization algorithms for different types of video delivery (either managed or unmanaged services) through the first and second contributions. Then, the third objective is to provide reliable security measures based on hash chains and digital signature methods for video stream.

This chapter lists the current issues in video services delivery under managed or unmanaged access. Also, it highlights our thesis contributions and counts the chapters included in each contribution.

1.1 General Introduction

The Future Multimedia (FM) or the Next Generation Multimedia Networks (NGMN) is one of the main points of today's research. It occupies a wider area of interest from researchers, Internet Service Providers (ISP), operators and also from some special or expert users. Several research projects like Future Internet Design FIND [1], The Global Environment for Network Innovations GENI [2] tries to imagine the future network architectures and the replacements that would be made for migration towards the whole IP based networks for new services like TV. The Future Internet Design (FID) which leads to Ubiquitous Network (UN) for users or for services has a high relation with two important subjects; the IP Multimedia services and the management of social networks. Therefore, research here has many interest points like network resources optimization, service optimization, media preparations and reliability in security measures for sensitive real time applications.

An NGMN is a packet-based network able to provide services including Telecommunication Services (TS), Data Services (DS) and Multimedia Services (MS). Also, it is able to make use of multiple broadband, QoS-enabled transport technologies where service-related functions are independent from underlying transport-related technologies. It offers unrestricted access by users to different service providers. It supports generalized mobility which will allow consistent and ubiquitous provision of service to users. The general idea behind Next Generation Network (NGN) is that all

information is transmitted via packets, like the Internet; packets are labelled according to their type (data, voice, video etc) and handled differently for Quality of Service (QoS) and security purposes by traffic management equipment. In an NGN, there is a more defined separation between the transport (connectivity) portion of the network and the services that run on top of that transport.

So, we could conclude that, the NGN is a hybrid aspect for video delivery which enables accessing videos any way. Moreover, with this type of delivery, video and Content Delivery Network (CDN) problems are solved together like video adaptations, resolutions, coding and QoS.

1.1.1 Problem Statement

Management of video content distribution through files allocation or caching in content delivery networks with some degree of reliable security measures is representing a big issue in video service delivery. Proposing a suitable algorithm is mainly depending on the type of service or environment of access. So, the nature of service provided by YouTube is different than services provided by Orange (French IPTV operator). As a result, we investigate some algorithms and techniques valid for each type. In YouTube, the files uploaded by different zones may have large hits of access from other zones far from the uploaded one. So, there is a need for redistributing the uploaded videos to YouTube by optimization techniques that take into account the hits distribution and cost of access between zones. Moreover, the performance of delivery could be enhanced by imposing some ways of files duplication in different zones rather than the best hosting ones. But, for managed service as Orange, the aspect of caching videos could be optimized by minimum spanning tree for cost optimization in multicast services like IPTV business model. In this case, a dynamic technique is required to rebuild dynamic trees of multicast based on the operators cost thresholds for CAPEX or OPEX.

After solving the optimization of service delivery in the previous types of video, the unmanaged service mainly suffers from some unreliability in case of security measures applied like signatures or hash chains. In order to achieve some degree of reliability in hash chains used in video streaming and multicast applications, we apply some redundancy codes to be piggybacked in real time packets. These codes could help the

video receivers tracking the links in hash chain and continue enjoying the stream despite the occurrence of some packets loss.

In order to correlate our views and ideas that could build an optimization technique suitable for Web-TV or IPTV, we first need to refine each service as follows:

Actually, we have two main categories of video under current business models; Web-TV and IPTV. Table 1.1 summarizes the main points of similarities or differences between the two aspects. This comparison mainly highlights the most important items which are relevant to each type of service and helps understanding the nature of each business. As cleared from the table, each service has some advantages over the other which imposes the existence of the two services together provided some times by same operator.

Table 1.1: comparison between Managed and Unmanaged TV services

	Managed TV (IPTV)	**Unmanaged TV (Web TV)**
Clients	Subscribed Users	All Internet Users
Access Network	Private Infrastructure	Open Infrastructure
Cost	- Fees per month - Pay-per-view - Subscribed channels - Triple play package	- Free - Subscribed service - Pay-per-view
Access Equipments	Any digital device behind (STB) like TV, PC or any	PC or Phone
Proposed Services	Live TV VOD	Video Sharing VOD
Quality of Service	QoS guarantee (High)	Best Effort Internet (Low)
Security of Service	Dedicated security	With or Without Authentication
Transmission Form	Multicast	Unicast
Examples	- Orange (France) [3] - Free (France) [4]	- YouTube [5] - Dailymotion [6]

1.1.2 Impairments in Video Service Delivery

In general, real time applications suffer from some impairment like: End-to-End IP-Network impairments (Packet loss, Delay (buffering technology) and Jitter). Also, video related impairments: Codec, Quality and Adaptation. Finally, impairments relevant in telecommunication networks: background noise, rates and convergence with other services. In this part, we categorize these impairments as follows:

1.1.2.1 Video Files Allocation

Video files allocation refers to an extensive set of algorithms. The goal of those algorithms is to optimize the performance of networks (through its best design) and services (through its distribution). The main objective is to assure acceptable performance that reflexes the clients' satisfactions from networks accessing. The required algorithms are supposed to solve the following issues:

- Reducing the amount of videos transactions over links connected different geographical zones
- Reducing the number of hits that coming from wide separated zones
- Reducing the access costs for video files
- Enhancing the management of video sharing servers
- Achieving high gain from the operators' point of view

1.1.2.2 Video Files Hosting

The hosting of video files is a complex task in video sharing servers or content delivery networks. The goal from this point is to introduce efficient techniques in hosting that assure the following points:

- Fast searching and recommendations
- Minimum space on servers
- Minimum accessing cost
- Efficient gain from hosting

1.1.2.3 Bandwidth Utilization

Media delivery and streaming over public or private networks are becoming the highest rank of applications consuming Bandwidth. Practically, they are rapidly increasing in network Bandwidth utilization with the huge number of Internet users concerning video access. The bandwidth problem has two faces; one related to access rate and one related to link capacity. For unmanaged service, Bandwidth utilization is little bit complex but for operators, it is under control. So, the performance optimization relevant to bandwidth utilization is a key factor for successful delivery and successful business based video.

1.1.2.4 Video Files Caching

Caching means consuming large spaces and processing from servers, it also represents good way for service delivery improvement. Actually, the main objective from caching is to make the files very near to users so as to facilitate their accessing times. The key factors for caching are:

- Files sizes
- Number of users requesting the files
- Caching cost
- Gains from caching
- Correlation between files hits and their costs
- Files popularities and their distributions over time

1.1.2.5 Video Adaptation

In the past, each user only used one device to connect the network to enjoy videos (it was the traditional TV). But in this era, each user may be having more than one device that can be used for network access like PC, Mobile IP, Digital TV, Personal Digital Assistant (PDA) or a Smartphone. So, with those varieties of devices that will depend only on same or different connections, the user may need to adapt video resolution according to device capabilities. But, this issue is difficult for user and the operators take in charge. Then, the problem transfers from video adaptation to different streams transmission issues. It affects servers' scalability for processing different flows of same video. On the other hand, there are no problems now for the scalability of servers in general after the new generation of servers' virtualization.

1.1.2.6 Nomadic Access for Managed Service

IPTV started by some channels and videos delivered to STB which was strict to physical place (user home). Now, there are many trends to consider the roaming of IPTV service. These directions enable a new way of service provisioning, called Nomadic Service (NS) provisioning. This type of service provisioning envisions the use of personal devices as a Nomadic Device (ND) for the deployment of those services called Nomadic Services (NS). The operators need to adapt their networks for supporting such type of service which means support the principle of Nomadic Networks (NN). Also, customers have to interact with their service providers to

accommodate the new culture of TV anywhere or TV anytime. In this thesis, we investigate simple software model for accessing IPTV service by Smartphone through WIFI connection without needing of STB between the server and client.

1.1.2.7 Multicast Performance

The standard for IPTV transmission for all operators is multicast. There is no problem from managing this technique inside any operator network. Only the Internet does not support such way of routing. So, the success of IPTV service is subject to the network type and way of transmission.

1.1.2.8 Security for Unmanaged Service

The IPTV operators introduce their services under some physical security measures relevant to the Set-Top-Box (STB). But, for unmanaged service, the security is a big issue. YouTube is one of famous video sharing servers over Internet but, the security measures for uploading or downloading files are very weak. No Digital Rights Management (DRM) is applied on unmanaged files. It is easy to attack any video session while the client access the service from such kinds of servers. Also, implementing security measure with free access services is not a prerequisite from business point of view. But, from the clients point view, it is important to trust the videos they are enjoying.

1.1.2.9 Reliability in Security Measures

As the video service in general is loss tolerant application, it is important to achieve some degree of reliability in the applied security measure to video streaming. There are high dependability on hashing algorithms and digital signatures. Some researchers prefer using hash chain for its simplicity and its minimum overhead. But, for such type of security if some packets drop, the chain could be broken and stop. So, searching reliability means achieving continuous security although having some drops (packet loss) in the video stream.

The last part of this thesis studies the reliability problem for using hash chain with video streaming. The study focuses on the effects of packet loss delay and replay attacks on broking the hash link. The proposed algorithm can overcome those attacks by its reliable window mechanism with the help of secure redundancy codes.

1.2 Thesis Organization

This thesis has three main contributions organized in seven chapters as follows:

1.2.1 Optimization of Video Files Distribution

This part focuses on unmanaged video services delivery like YouTube. As more and more videos are uploaded each day, the performance of such servers deteriorates with Best Effort (BE) quality. So, in this contribution we try to propose different ways for optimizing the video files distributions between different zones. This contribution is divided into two chapters:

Chapter 2: This chapter provides a brief study about some video sharing servers like YouTube. Then, an analysis about some statistics in terms of files viewers and their geographical distributions is done. Moreover, an optimization for file allocation is introduced and tested by traces collected from YouTube. Finally, some file cost hosting analysis is shown with some social relations impacts on YouTube videos.

Chapter 3: After optimizing the best locations in the previous chapter, we conduct in this chapter another optimization for the same files by duplicating the files in different zones based on users Quality of Experience levels and operator thresholds for caching costs. Two ways of file duplication are tested and analyzed in terms of costs and gains either file caching or file retrieval.

1.2.2 Open IPTV Architectures and Operators Networks Optimization

This part considers Managed IPTV services and its main elements. We propose an open IPTV management solution that can operate in managed and unmanaged ways. Also, we consider the optimization of MST as follows:

Chapter 4: This chapter illustrates some common IPTV terminologies and definitions for the new aspects like TVA, Pay-TV, TVE and others. It also introduces an implementation as a simple Testbed for live-TV or VOD based unicast delivery. Moreover, we list current and proposed architectures for collaboration design for IPTV operators. Finally, this chapter highlights the cost analysis for our collaboration model in terms of CAPEX or OPEX costs.

Chapter 5: As the Managed IPTV service is mainly delivered by their operators to Set Top Box (STB) clients based on multicast service, the operator topologies should be subject to minimum spanning tree calculations for choosing the minimum cost paths to streamer points. Therefore, the growth of this topology will affect on the overall performance of the output multicast trees. So, this chapter provides an optimization for this issue by building dynamic way that depends on dividing the original tree into different trees and duplicate the streaming points (multicast source) into some recommended nodes according to some thresholds calculated and set by the operators. Different scenarios are simulated with different nodes capacity. Finally, the complexity of MST is calculated for heuristic and non heuristic way of running spanning tree.

1.2.3 Secure and Reliable Hash Chain in Video Streaming

This part studies the reliability problem for using hash chains with video streaming. It focuses on the effects of packet loss delay and replay attacks on breaking the hash link. We propose an algorithm that can overcome on those attacks by its reliable window mechanism with the help of secure Redundancy Codes (RC). Then, we ameliorate the security degree by integrating signatures with window based-RC to eliminate some common attacks to video delivery. This work is divided into three chapters as the following:

Chapter 6: This chapter highlights the hash chain background in video streaming service and real time applications in general. Also, it provides a background about hash chains usage in video streaming service authentication.

Chapter 7: The aspect of window based hash chain will be explained in this chapter with some analysis of analytical and simulation results for different ways of achieving reliability in hash chain mechanism.

Chapter 8: This chapter introduces the integration of signature based window mechanism with RCs. Then, comparison between different options (*packet-based*, *block-based* and *window based* with RC) will be conducted in terms of reliability and additional overhead.

Finally, the thesis contributions are concluded with some future research directions in ***Chapter 9***.

Chapter 2: Optimization of File Allocation for Video Sharing Servers

Social networks instigate several research interests. This chapter focuses on one of the famous servers on the Internet that occupies a wider area than others, it is YouTube. YouTube has the third rank of the internet sites related to its traffic transactions. We want to draw the attention on the effect of these huge bases of viewers, hits, users and files of this kind of sites. We propose an optimization of its server networks design and files allocation procedure that will improve the number of hits related to this server over the Internet and the files revenue from this optimization.

2.1 Introduction

Many servers use the short videos as a business model because of their abilities to diffuse these clips to the end users in an easy manner. Users can share their own videos, upload and download very easily and also create their own profiles to know the rating of their videos evaluated by others. Many servers like YouTube[7], Google[13], Yahoo[19], Dailymotion[6] and others take good positions in ranking sites presented by Alexa [14].This evaluation shows that the ranking of content depending on uploads and downloads or just the number of videos uploaded by others.

According to these statistics, YouTube has great interest because it was ranked as the third site out of the ranked 500 sites with the highest traffic hits over the Internet, so we will focus on this service in this chapter as a good case study. Although Yahoo has an advanced rank than YouTube (the second site by Alexa), YouTube is more famous for short videos transfer. Dailymotion also occupies the rank 68 by the same measurement site.

The rest of this chapter is organized as; Section 2.2 introduces the history of YouTube and its statistics, Section 2.3 discusses the social networking concept and its data centers

design, Section 2.4 presents our proposal for file allocation optimization, Section 2.5 shows our algorithm evaluation, Section 2.6 highlights more problems in file hosting on YouTube and its proposed solutions. Section 2.7 concludes state-of-the-art. Finally, the conclusions for this chapter are presented in Section 2.8.

2.2 History of YouTube

Many statistics explained YouTube progression during last years [7, 13, 20, and 24]. YouTube is the third most visited website in the world, the leader in online video and the premier destination to watch and share original videos worldwide through the Internet. It allows people to easily upload and share video clips on *www.YouTube.com* and across the Internet through websites, mobile devices, blogs, and email. Everyone can watch videos on YouTube, upload, download and also create its own profile on this server to pursue their videos statistics. People can see first-hand accounts of current events, find videos about their hobbies and interests, and discover the quirky and unusual. As more people capture special moments on video, YouTube is empowering them to become the broadcasters of tomorrow [19].

In YouTube's history, USA registered the largest number of hits for this site, more than 15,500,000 clicks in the process of measuring the hits of this server related to USA only. YouTube drew *5 Billion U.S. Online Video Views* in last years, the measurements reporting that Americans viewed more than 11.4 billion videos for a total duration of 558 million hours during one month [8].

The press in USA indicates that YouTube Attracts 100 Million U.S. viewers. Online video continues to make gains in the U.S. with Internet users viewing 14.8 billion videos during one month, representing an increase of 4 percent over the last month of same year. According to the last statistics, YouTube was the most popular property, accounting for 91 percent of the incremental gain in the number of videos viewed versus one month, as it exceeded 100 million viewers for the first time [11].

Through Google Trends & Analytics [9, 10], we can draw some statistics about YouTube servers, hits, uploads and downloads. Those statistics will help in imagining the structure of YouTube networks and how we can suggest some changes to the structure of this huge social network. Also, we can gain from optimizing bandwidth

utilization especially on the international lines between different countries and different continent (geographical zones).

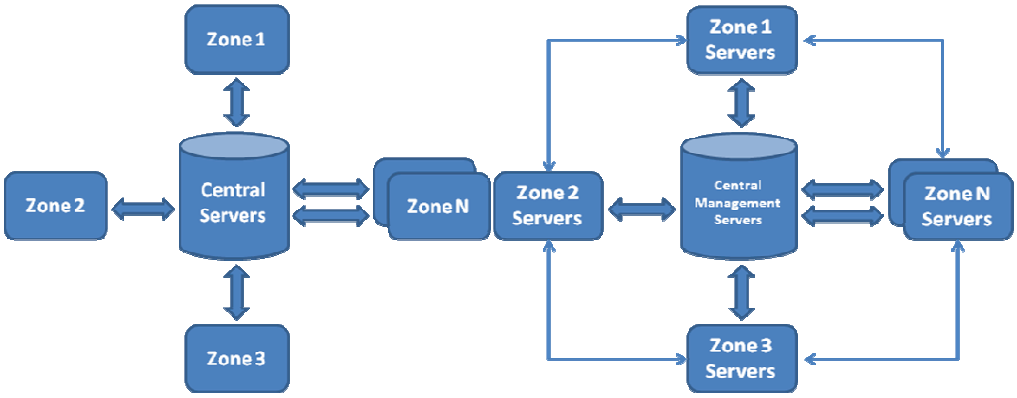
2.3 Social Networking

The field of social network design and short videos servers' delivery like YouTube became an important research domain in future multimedia service. Therefore, the aspect of social network and its data centers design is in need to be clarified as follows:

2.3.1 Data Centers Design

Trends toward centralization of data resources provide greater consistency and control across the organization. They can conflict with the need to reach a worldwide market with standardized high quality application delivery. In the past, an enterprise might respond to the need for additional capacity and performance with traditional data centre infrastructure options Figure 2.1-A. Since geography plays an important role in delivering video to global viewers, we examine costs in different world regions. The cost will provide a valuable baseline for enterprises like Google or YouTube to compare against alternatives such as Web acceleration services. Thus, enterprises may consider the expensive option of distributing hardware in regions of the world that are close to users and potential markets. In Figure 2.1-B, for example, the building of a data centre in different zones like Asia, Africa or with the much heavy loads areas. The files movements in this case may be not expensive if the infrastructure already exists all over the world.

If the enterprises increasingly replace their decentralized data architecture with large centralized data center infrastructure, the capacity becomes an issue. If the data center reaches high percent of utilization, the enterprise must find a way to scale its operation to gain more productivity from its current infrastructure [16]. We recommend the decentralized architecture to avoid the servers' burden problems and obtain a uniform distribution of data centers and not the traditional localized design.



A): Centralized design of data centers B): Decentralized design of data centers

Figure 2.1: Data centers design

2.3.2 Content Delivery Networks CDN

A Content Delivery Network is a large distributed system of servers deployed in multiple data centers in the Internet or through Internet Service Providers ISPs. The main goal of a CDN is to serve content to end users with high availability and high performance. This means that, it is considered as intermediate point between the content providers and clients. Moreover, this network can execute many video management actions like hosting/caching/fetching in order to satisfy the clients QoE or enhance the overall QoS. In video streaming systems either live-streaming or on-demand services; the CDN is key turning point factor in service delivery optimization.

2.4 Proposal for File Allocation Optimization

Our optimization starts by assuming that; we have a certain number of zones (i from 1 to N) and a number of servers (j from 1 to M). Our goal is to have minimum cost (in terms of number of hits and distance cost between zones) so as to reduce the number of hits on the main servers. So, we try to have a distribution formula that will lead to optimize the content locations on the servers in different zones according to minimum cost between zones and servers.

Let D_{ij} be the cost between server (j) and zone (i) where i is from 1 to N (max. no. of zones) and j from 1 to M (max. no. of servers). The total cost will be:

$$(C_i = \sum D_{ij} * H_{ij})$$

Where H_{ij} is the number of hits for a file coming from zone i to server j, we look for the:

$$\boldsymbol{min\ (C_i = \sum D_{ij} * H_{ij})}$$

That gives the best file allocation.

***Example**:*

Assuming we have six zones and six servers (each can be a cluster for each zone) on principal of one server per zone then:

For zone 1, where the access comes from zone 1:

$Min\ C_1 = (D_{11}*H_{11}) + (D_{12}*H_{12}) + (D_{13}*H_{13}) + (D_{14}*H_{14}) + (D_{15}*H_{15}) + (D_{16}*H_{16})$

For zone 2:

$Min\ C_2 = (D_{21}*H_{21}) + (D_{22}*H_{22}) + (D_{23}*H_{23}) + (D_{24}*H_{24}) + (D_{25}*H_{25}) + (D_{26}*H_{26})$

For zone 3:

$Min\ C_3 = (D_{31}*H_{31}) + (D_{32}*H_{32}) + (D_{33}*H_{33}) + (D_{34}*H_{34}) + (D_{35}*H_{35}) + (D_{36}*H_{36})$

For zone 4:

$Min\ C_4 = (D_{41}*H_{41}) + (D_{42}*H_{42}) + (D_{43}*H_{43}) + (D_{44}*H_{44}) + (D_{45}*H_{45}) + (D_{46}*H_{46})$

For zone 5:

$Min\ C_5 = (D_{51}*H_{51}) + (D_{52}*H_{52}) + (D_{53}*H_{53}) + (D_{54}*H_{54}) + (D_{55}*H_{55}) + (D_{56}*H_{56})$

For zone 6:

$Min\ C_6 = (D_{61}*H_{61}) + (D_{62}*H_{62}) + (D_{63}*H_{63}) + (D_{64}*H_{64}) + (D_{65}*H_{65}) + (D_{66}*H_{66})$

So, the min value between C_1 to C_6 will give us an approximate location for the best allocation for this file demands so as to save bandwidth and time for viewing or downloading the file. The evaluation results will lead us to logically move this file to the much requested region or where highest demands come from. We can see in Table 2.1 the steps of the optimization algorithm as we run them in Matlab.

2.4.1 The Correlation between Demographic Information and Web Usage

In this part we try to find a relation between the network topology and the cost calculated between zones. Assume we calculate the cost related to zone (1) by C_1 and make the same for zone 2 to 6. Then we can decide the best allocation for this file according to these values but based on the relation between zones.

2.4.2 First Assumption: Full Mesh Topology

If $C_1 <$ (C_2 to C_6) then the best location of this file is the servers in zone (1), this means that the minimum value C_i will define the best location of that file. Actually, the full mesh design suffers from the $\boldsymbol{N^2}$ links (the number of links needed for this design are equal $\boldsymbol{N(N-1)/2}$) where N is equal to number of nodes, and no one can guarantee the

full mesh design on the internet as shown in Figure 2.2.

2.4.3 Second Assumption: Not-full Mesh Topology

If the network is partially meshed then, the previous calculations and decisions will be different, and we must take into account the cost of the intermediate zones, for example if zone (1) is not directly connected to zone (2), but connected through zone (3) then, while making our calculation we must add the cost between zones 1&3 and 2&3 for the calculation of zone (1).

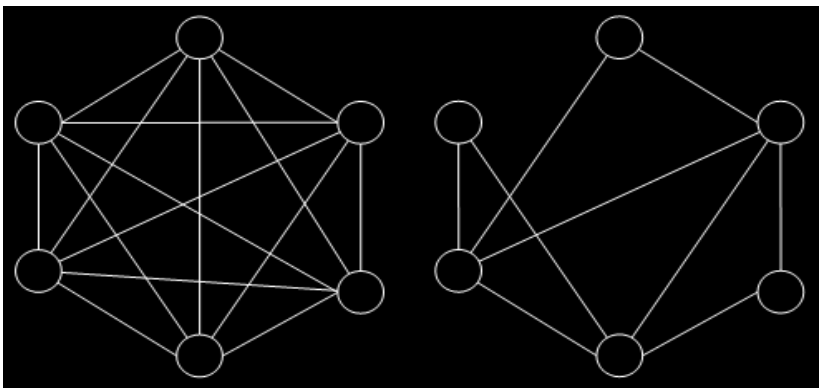


Figure 2.2: Full Mesh & Partially Mesh design for design for 6 zones topology

2.5 Proposal Evaluation

In this part we try to evaluate our algorithm by using some files chosen randomly from YouTube site that rated as top viewed during the period of test (one month). We chose 6 files from different zones as listed in Table 2.2 where each file has its own statistics related to hits and users from different zones estimated by the percentage of YouTube users for each file according to the file profile on YouTube and the statistics from Alexa [14].

By applying the algorithm in Table 2.1 on the six files selected in Table 2.2 taken file by file and by assuming the different locations of the file, i.e. we run the algorithm six times for each file by assuming the movements of the file from one zone to another and optimize the best location according to the minimum cost equation above. We finally reached the following results for the files distribution in Figure 2.3 as the hits of the six files in different zones.

Table 2.1: The Proposed Algorithm for files allocation

Algorithm

Input N (number of zones)
Input M (number of servers)
Input h_{ij} (number of hits come from zone i to sever j for specific file)
Input d_{ij} (the assumed cost between zone i and server j)
Total cost $C_i = \sum h_{ij} * d_{ij}$ // where i from 1 to N and j from 1 to M
If C_i min than C_{i+1} to C_N
Then allocate this file in zone (i)
Else moves this file to the Min (C_{i+1} to C_N) value location
End If
Return the best location for this file (best i)
End

For file 1, the best location is the servers in zone 1 where the minimum cost of that file investigated by the algorithm.

For file 2, it must be moved from its location in zone 2 to zone 3 where the algorithm gave the minimum cost and the rest of files shown in Figure 2.3; file 3 moves from zone 3 to zone 2, file 4 moves from zone 4 to zone 3, file 5 rest in the same zone 5 for achieving minimum cost and finally file 6 moves to zone 4.

At the end we can apply that algorithm on any file for which we have its statistics to define its best location on any video sharing servers.

Table 2.2: Examples of some files chosen from YouTube and related hits

Files/Hits Zones/%users	File 1 Hits 425783	File2 Hits 51340	File 3 Hits 174917	File4 Hits 27797	File5 Hits 12403	File 6 Hits 136394
Zone 1 users 30.95 %	131780	15890	54137	8603	3839	42214
Zone 2 users 9.25 %	39385	4749	16180	2571	1147	12616
Zone 3 users 30.85 %	131354	15838	53962	8575	3826	42078
Zone 4 users 23.05 %	98143	11834	40318	6407	2859	31439
Zone 5 users 2.45 %	10432	1258	4285	681	304	3342
Zone 6 users 3.05 %	12986	1566	4335	848	378	4160

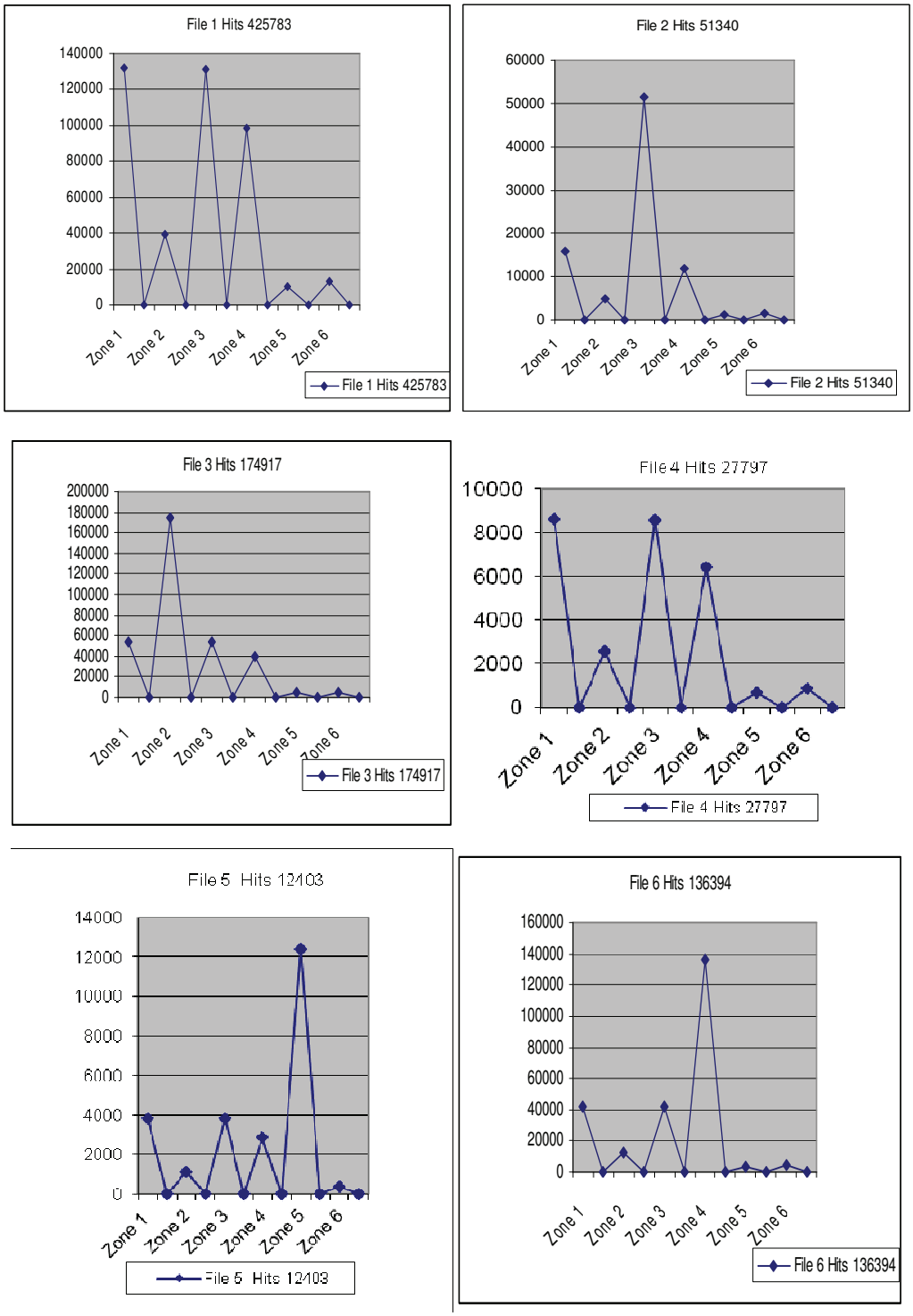


Figure 2.3: Files distribution by their hits and costs related to zones

2.6 The File Hosting Problems on YouTube

We know that users' generated content files UGC or sites that depend on YouTube only for free hosting will cause a huge problem to YouTube because the databases of the files will increase without any revenue from this hosting.

A report [22] about YouTube video growth curve assures that; YouTube currently streams more than 30 billion videos per month worldwide, this huge number of videos drew our attentions towards studying the problems of inflation of the server database and how this company will benefit from the hosting of those files. The danger is how YouTube will treat this difficulty without affecting the online video traffic marketing because there is a forecasting that; YouTube will lose about 470 Million Dollars and the company will need to change its strategies [23].

Several analyses on the continuing growth of YouTube's library versus the cost of storing files were made. Since the storage of video files that are not viewed will grow and should it be YouTube responsibility to keep these video records for the future and if so can the sponsors support additional cost for the long time hosting of those files? We believe that Google needs to study the phenomena of this growth and think how they will find a good solution for managing this problem.

2.6.1 The Proposed Solutions for Video Hosting Costs

In this part we have two problems; the first one is the long term hosting of files that became old and the second one is about hosting cost in relation to the revenue from hosting:

2.6.2 Long Term Hosting

To evaluate the hosting of each video, we can add two flags numbers related to each added file on the server; the first one is (F_h) which represents the history flag that can register the history of this file by counting the number of days this file allocated or uploaded to the server. The second flag is (F_v) which represents the number of viewers of that file. By calculating the ratio (F_v/F_h) in a certain period of time (ex. one month), we will have an indication that this file merits to remain on the server in case the ratio is high or to be omitted from the server database in the other case. This ratio express the file time-to-live on the server, so the file time-to-live is $(FTTL=F_v/F_h)$.

2.6.3 The Relation between File Hosting Cost and its Revenue

This part will focus on studying the behaviour of file hits on their hosting servers and the revenue returned from hosting this file or moving to other zone servers according to the file allocation algorithm in Table 2.1.

The following parameters will be used in optimization of the total revenue from hosting files;

H_{ij}: Numbers of hits/day for a file on server j, accessed from users in zone i.

D_{ij}: Cost of hitly a file on server j, from user in zone i.

$C_j = \sum_i H_{ij}.D_{ij}$: Total cost/day for a file put on server j.

$H_j = \sum_i H_{ij}$: Number of hits/day for a file on server j.

$\bar{C}_j = \frac{C_j}{H_j}$: Average cost/day for a file put on server j.

$\bar{R}_j$: Average revenue generated by a file put on server j.

$R_j = H_j . \bar{R}_j$: Total revenue generated /day for a file put on server j.

So we can now calculate the Revenue Cost Ratio (*RCR);*

$$RCR = \frac{R_j}{C_j} = \frac{\bar{R}_j.H_j}{\sum_i H_{ij}.D_{ij}}$$

This ratio will give us an indication for the revenue from hosting this file in terms of its hosting cost with relation to the file hits.

Finally, we can calculate the net revenue by the following equation;

$$R_j - C_j = H_j . \bar{R}_j - \sum_i H_{ij}.D_{ij}$$

And this revenue is considered as an indication for the decision of hosting the file or no.

2.6.4 Traffic and User Social Behaviour

From Table 2.3, we can see that; the traffic of YouTube has a great effect in attracting the viewers. As we noticed in the table the average time each user spent per day is 22.9 minutes and this time is not a small period if we make comparisons with other sites like news or mails sites. Moreover, all internet researchers expect that this time will increase

dramatically in the near five years. On the other hand, many countries which have a large percentage of users accessing YouTube like USA not by default give us an indication that it is the country who has the maximum percent of utilization in terms of the probability of YouTube users if we take into account the total number of internet users as a base for that calculation as the results appeared in the last column of the Table 2.3.

The social relation between YouTube viewers and videos is a significant relation and we cannot put it under any mathematical equation because the number of visitors to YouTube as it appeared in the Table 2.3 has no relation if we compared it with the last column of this table (the probability of YouTube users(x/n)). For example, if we compare USA with Ireland we can find that; the viewers' probability for Ireland is 25%:10.4% to USA although the number of internet users in USA is (220.1 M: 2.4 M) to Ireland which almost ratio 100:1.

Table 2.3: Statistics & Evaluations for the most famous countries by YouTube

COUNTRY	HITS/DAY FOR THE MOST VIEWED FILE	% OF USERS ACCESS YOUTUBE (X)	INTERNET USERS (MILION) (N)	YOUTUBE VIEWERS/ MONTH	TOTAL TIME(MIN) (AV = 22.9 M/D/U)	PROBABILITY OF YOUTUBE USERS (X/N)
Australia	136394	0.6	17	102000	2335800	0.035
N. Zealand	2522	0.5	3.4	17000	389300	0.147
UK	174917	3.8	43.8	1664400	38114760	0.087
Canada	5710	2.3	28	644000	14747600	0.082
Ireland	117914	0.6	2.4	14400	329760	0.250
Brazil	26807	3.8	67.5	2565000	58738500	0.056
France	51340	3.7	40.9	1513300	34654570	0.090
Japan	27797	7.7	94	7238000	165750200	0.082
Poland	33186	1.3	20	260000	5954000	0.065
India	28667	4.0	81	3240000	74196000	0.049
Czech R.	200085	0.5	5	25000	572500	0.100
USA	425783	22.8	220.1	50182800	1149186120	0.104
Germany	29414	4.7	52.2	2453400	56182860	0.090
Spain	40291	2.6	28.6	743600	17028440	0.091
Hong Kong	59993	0.6	4.9	29400	573260	0.122
Israel	12403	0.5	5.3	26500	606850	0.094
Italy	32884	3.8	28.4	1079200	24713680	0.134
S.Korea	14019	1.0	36.8	368000	8427200	0.027
Mexico	103461	4.0	23.9	956000	21892400	0.167
Netherland	11059	1.4	13.8	193200	4424280	0.101
Russia	20601	1.4	30	420000	9618000	0.047
Sweden	7558	0.8	7.3	58400	1337360	0.110
Taiwan	28811	0.7	15.1	105700	2420530	0.046

2.7 Related Work

Several solutions have been proposed for file allocation for videos and other heavy

loaded servers and data over the Internet by caching files like AKAMAI systems [17]. It facilitates for video delivery and streaming caching solution but still the research tries to find and design solutions for servers and files allocation procedure with lowest cost.

Many measurements and statistics have been carried out to show the grand utilisation of social networks and the increasing users numbers uploaded or downloaded videos through internet using famous sites like YouTube or Dailymotion which are the most famous now. Some examples of those studies like [15], when it comes to understanding traffic pattern, YouTube is a significant web site even among web 2.0 sites. This is due to the fact that content of YouTube is video that consumes much more bandwidth than text, picture and audio. Some statistics regarding the network of the University of Calgary and YouTube would be useful. The university has 28000 students and 5300 faculty and staff. The data for this study was collected in 85 consecutive days in spring 2007. It turns out that YouTube traffic constitutes 4.6% of the whole internet traffic of the university. YouTube is the most popular video-sharing web site and it is the source of 60% of the videos watched over the internet. 10,000,000 videos are downloaded to watch every day and 65,000 new ones are uploaded.

There is another work which presented a systematic and in-depth measurement study on the statistics of YouTube videos [16]. When authors analyzed these statistics, they found that YouTube videos have noticeably different statistics from traditional streaming videos, in aspects from video length to access pattern. They also studied some new features that have not been examined by previous measurement studies: the growth trend and active life span of videos.

In [21], the authors tried to analyze the video and the user characteristics for different geographical regions, concentrating mainly on Latin America. They developed an efficient way for collecting data about videos and users and based on the collected data, they showed that there exists a relationship between geography and the social network features available in YouTube. They presented evidence that indicates that geography creates a locality space in YouTube, which could be used to explore infrastructure improvements, such as caching mechanism and content distribution networks.

The study introduced in [25] had analyzed the YouTube as a famous User Generated Content (UGC) and VOD server. The analysis was made based on videos popularities life cycle. Also, they handled their statistics from the perspective of requests and their relationships with the video age on the system. To do so, they classified the videos into different groups according to their ages and then calculate the requests coming for each

group to find the optimal distributions. Moreover, they proposed an efficient caching technique for VOD systems like YouTube or P2P communications in general. Finally, the work drew the probability of video viewers over the time to define its popularity as a main parameter for file caching and distribution.

2.8 Conclusion

In this chapter we tried to analyze the history and statistics related to one of the most popular site for video delivery, it is YouTube.com. We also focused on the social network design for such types of servers and finally the optimization of file allocation and how to move files from server to server according to the minimum cost between zones and servers and also related to minimizing the hits directed to some servers according to file allocation and geographical distribution of servers. We also optimized the revenues from hosting and movements of those files.

We now try to optimize the allocation of files and their distribution in case of managed network for any video operator. This will lead us to analyze and propose ways for network operators to control the caching of video files.

In the next chapter, we will introduce two ways of file duplication and evaluate them by feeding same files captured from YouTube traces.

Chapter 3: Video File Duplication Mechanisms

Consumers of short videos on Internet can have a bad Quality of Experience (QoE) due to the long distance between the consumers and the servers that hosting the videos. We propose an optimization of the file allocation in telecommunication operators' content sharing servers to improve the QoE through files duplication, thus bringing the files closer to the consumers. This optimization allows the network operator to set the level of QoE and to have control over the users' access cost by setting a number of parameters. Two optimization methods are given and are followed by a comparison of their efficiency. Also, the hosting costs versus the gain of optimization are analytically discussed.

3.1 Introduction

The exponential growth in number of users access the videos over Internet affects negatively on the quality of accessing. Especially, the consumers of short videos on Internet can perceive a bad quality of streaming due to the distance between the consumer and the server hosting the video. The shared content can be the property of web companies such as Google (YouTube) or telecommunication operators such as Orange (Orange Video Party). It can also be stored in a Content Delivery Network (CDN) owned by an operator (Orange) caching content from YouTube or DailyMotion. The first case is not interesting because the content provider does not have control over the network, while it does in the last two cases, allowing the network operator to set a level of QoE while controlling the network operational costs.

Quality of Experience (QoE) is a subjective measure of a customer's experiences with his operator. It is related to but differs from Quality of Service (QoS), which attempts to objectively measure the service delivered by the service provider. Although QoE is perceived as subjective, it is the only measure that counts for customers of a service.

Being able to measure it in a controlled manner helps operators understand what may be wrong with their services.

There are several elements in the video preparation and delivery chain, some of them may introduce distortion. This causes the degradation of the content and several elements in this chain can be considered as "QoE relevant" for video services. These are the encoding systems, transport networks, access networks, home networks and end devices. We will focus on the transport networks behaviour in this chapter.

In this chapter, we propose two ways of optimization in file duplication either caching or fetching the video files. Those file duplication functions were tested by feeding some YouTube files pre-allocated by the optimization algorithm proposed in the previous chapter.

By caching, we mean duplicate a copy of the file in different place than its original one. While, in fetching, we mean retrieve the video to another places or zones in order to satisfy instant needs either relevant to many requests or cost issues from the operators point of view.

The file distribution problem has been discussed in many works. Especially the multimedia networks, the work in [27] handled the multimedia file allocation problem in terms of cost affected by network delay. A good study and traffic analysis for inter-domain between providers or through social networks like YouTube and Google access has been conducted in [28]. In that work, there is high indication about the inter-domain traffic that comes from the CDN which implies us to think in optimizing files allocations and files duplications.

Akamai [17] is one of the more famous media delivery and caching solution in CDN. They proposed many solutions for media streaming delivery and enhancing the bandwidth optimization for video sharing systems.

The rest of this chapter is organized as follows: Section 3.2 presents the state-of-the-art relevant to video caching. Section 3.3 highlights allocation of files based on the number of hits or requests to any file. In Section 3.4, we propose the optimization by two file duplication mechanisms either caching or fetching. Numerical results and threshold propositions are introduced in Section 3.5. The hosting issues and gain from this optimization are handled by Section 3.6. Finally, this chapter is concluded in Section 3.7.

3.2 State-of-the-Art

Caching algorithms were mainly used to solve the problems of performance issues in computing systems. Their main objective was to enhance the speed of computing. After that, with the new era of multimedia access, the term caching was used to ameliorate the accessing way by offering the content close to the consumers. The Content Delivery Network CDN appeared to manage such types of video accessing and enhance the overall performance of service delivery.

For the VOD caching, the performance is very important as a direct reflection to bandwidth optimization. For [29], they considered the caching of titles belonging to different video services in IPTV network. Each service was characterized by the number of titles, size of title and distribution of titles by popularity (within service) and average traffic generated by subscribers' requests for this service. The main goal of caching was to reduce network cost by serving maximum (in terms of bandwidth) amount of subscribers' requests from the cache. Moreover, they introduced the concept of "cacheability" that allows measuring the relative benefits of caching titles from different video services. Based on this aspect, they proposed a fast method to partition a cache optimally between objects of different services in terms of video files length. In the implementation phase of this algorithm, different levels of caching were proposed so as to optimize the minimum cost of caching.

The work presented in [30] considered different locations of doing caching in IPTV networks. They classified the possible locations for caching to three places (STB, aggregated network (DSLAMs) or Service Routers SRs) according to their levels in the end-to-end IPTV delivery. Their caching algorithm takes the decisions based on the users' requests or the number of hits per time period. Caching considers different levels of caching scenarios. This work considered only the scenarios where caches are installed in only a single level in the distribution tree. In the first scenario each STB hosts a cache and the network does not have any caching facilities while, in the second and the third scenarios the STBs do not have any cache, but in the former all caching space resides in the DSLAMs, while in the latter the SRs have the only caching facility. The scenario where each node (SR, DSLAM and STB) hosts a cache and where these caches cooperate was not studied in that work. Actually, the last scenario could be little bit complicated in the overall management of the topology. Finally, their caching

algorithm was considered based on the relation between Hit Ratio (HR) and flows Update Ratio (UR) in a specific period of time.

The same authors presented a performance study about the caching strategies in on-demand IPTV services [31]. They proposed an intelligent caching algorithm based on the object popularity in context manner. In their proposal, a generic user demand model was introduced to describe volatility of objects. They also tuned the parameters of their model for a video-on-demand service and a catch-up TV service based on publicly available data. To do so, they introduced a caching algorithm that tracks the popularity of the objects (files) based on the observed user requests. Also, the optimal parameters for this caching algorithm in terms of capacity required for certain hit rate were derived by heuristic way to meet the overall demands.

Another VOD caching and placement algorithm was introduced in [32]. They used heuristic way based on the file history for a certain period of time (like one week) to guide the future history and usage of this file. Also, the decisions were made based on the estimations of requests of specific video to a specific number as a threshold value. The new placements are considered based on the frequency of demands that could update the system rates or estimated requests. Finally, their files distributions depend on the time chart of users' activities during a period of time (for example one week) and its affections on files caching according to their habits. To test this algorithm, they used traces from operational VOD system and they had good results over Least Recently Used (LRU) or Least Frequently Used (LFU) algorithms in terms of link bandwidth for caching emplacements policies. So, this approach is considered as a rapid placement technique for content replications in VOD systems.

An analytical model for studying problems in VOD caching systems was proposed in [33]. They presented hierarchical cache optimization (i.e. the different levels of IPTV network). This model depends on several basic parameters like: traffic volume, cache hit rate as a function of memory size, topology structure like DSLAMs and SRs and the cost parameters. Moreover, the optimal solution was proposed based on some assumptions about the hit rates and network infrastructure costs. For example, the hit rate is a function of memory used in cache and there is a threshold cost per unit of aggregation network points like DSLAMs. Also, they demonstrated different scenarios for optimal cache configuration to decide at which layer or level of topology.

A different analysis was introduced in [25] about data popularity and affection in videos distributions and caching. Through that study, they presented an extensive data-driven analysis on the popularity distribution, popularity evolution, and content duplication of user-generated (UG) video contents. Under this popularity evolution (from starting point), they proposed three types of caching:

- *Static caching:* at starting point of cache, handled only long-term popularity
- *Dynamic caching:* at starting point of cache, handled the previous popularities and the requests coming after the starting point in the period of trace
- *Hybrid caching:* same as static cache but with adding the most popular videos in a day

By simulation, the hybrid cache improved the cache efficiency by 10% over static one. Finally, this work gave complete study about popularity distribution and its correlations to files allocations or caching.

We will now start by analyzing the optimization of the file allocation introduced in chapter 2 in order to reduce the total access cost, taking into account the number of hits on the files. Then, we will analyze the file duplication algorithms.

3.3 Optimization of Files Best Location

We start from the topology and results in Chapter 2. In order to reduce the total access cost, we first move files so as to have them located in the node from where the demand is the highest. We can follow the steps of the algorithm in Table 3.1. The main objective from this algorithm is to define the best allocation zone i servers of file (f) uploaded from any geographical zone i where (i=1:N zones). As example, we imagine a network with 6 zones distributed as shown in Figure 3.1.

We applied that algorithm on six YouTube files [26] chosen from different zones as shown in (Table 3.2). The numbers are rounded for better clarity. Those files were preselected as the most hit files from different zones analyzed briefly in [26].

Table 3.1: Best location algorithm

```
Input N              //number of zones
Input M              //number of servers
Input h_ij           //number of hits from zone i to server j for a specific file
Input d_ij           //the assumed cost between zone i and server j
        Total cost C_i = ∑ h_ij * d_ij          // where i from 1 to N and j from 1 to M
                If C_i < C_i+1,...,C_N
                        Then allocate this file in zone i
                        Else move this to min (C_i+1,...,C_N) value location
                End If
Return the best location for this file          //best j
End
```

Table 3.2: Examples of six files chosen from YouTube and their related hits with simple approximations in hits distribution

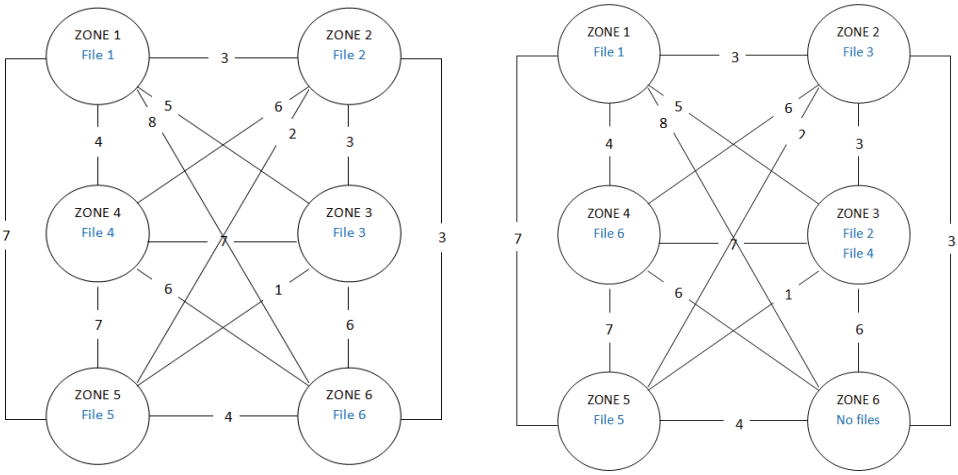
	Hits					
	File 1	**File 2**	**File 3**	**File 4**	**File 5**	**File 6**
Zone 1	135000	16000	55000	8700	3800	42000
Zone 2	40000	5000	175000	2700	1000	15000
Zone 3	132000	51000	55000	8800	3900	43000
Zone 4	100000	12000	40000	6400	2800	138000
Zone 5	12000	1000	5000	800	12400	4000
Zone 6	17000	1000	5000	900	400	5000

We reach to the following results of files' distribution or best allocation:

The best location of File 1 is the servers in zone 1 where the algorithm gave the minimum cost; File 2 moves from zone 2 to zone 3; File 3 moves from zone 3 to zone 2; File 4 moves from zone 4 to zone 3: File 5 stays in zone 5; File 6 moves to zone 4. The new files distribution is shown in Figure 3.1-B.

Since geography plays an important role in delivering video to customers, we propose the following network representation shown in Figure 3.1-A. We divide the network into six zones and we select a file uploaded from each zone to be studied by our algorithms.

The arcs connecting the nodes are physical links, and the numbers on them represent an access cost, i.e. the cost of delivering a video from zone A to a consumer from zone B. That access cost is assumed to be symmetric combination of many parameters such as the path's length, the hop count (from an edge router to an edge router), the cost of using the link (optical fibber, copper, regenerators and repeaters, link rental cost) and the available bandwidth. The files of a given zone are hosted by the servers of that zone.



A): Network Representation B): Files distribution output

Figure 3. 1: Files and zones distributions

3.4 Optimization through File Duplication

To grant a given QoE, we choose to give access to popular file (located in zone **j** servers and having a number of hits higher than a given value **Y**) for a given consumer (from zone **i**) only if this consumer can access that file with a cost $\mathbf{a_{ij}}$ lower than a given value **A** ($\mathbf{a_{ij}} =< \mathbf{A}$). If it is the case, we check the demand on that same file from the zone **i** of the customer (which is $\mathbf{y_{ij}^{f}}$). If the demand is higher than a threshold **Y**, we duplicate file **f**. Then, any consumer can access any content with good quality and minimum cost. The steps of file duplication algorithm are shown in Table 3.3.

Table 3.3: Duplication algorithm

```
Input N          // number of zones
Input A          // maximum access cost, set by the network operator
Input Y          // minimum number of hits allowing file duplication,
                 //set by the network operator

Input a_ij       // access cost to a file located in zone j from a customer located in zone i
Input p_j        // number of files in zone j
Input y_ij^f     // number of hits on file f (located in zone j) from zone i users

For i from 1 to N                                   // user's zone
      For j from 1 to N                             //file's zone
            If a_ij > A                             // we suppose that a_ii=0
                  Then For f from 1 to p_j
                        If y_ij^f > Y
                              Then duplicate(f,i,j) // duplicates file f
                                                    //in the best location
                        End If
            End If
End
```

Thus, we make a compromise between access cost and storage cost. Actually, the higher the value of **Y**, the higher the access cost and the lower the need for storage capacity. Also, the higher the value of **A**, the higher the access cost and the lower the need for storage capacity. The values of **A** and **Y** are thresholds being adjusted by the operators (Content Delivery).

Now, we are going to experiment two different duplication functions:

- ***duplicate1***(f,i,j) duplicates file ***f*** (located in zone j) in zone i which means ***(caching)*** Table 3.4.
- ***dupliacate2***(f,i,j) duplicates file ***f*** in the zone k that has the second highest number of hits on that file (located in zone j) and that grants an access cost lower than **A** at the same time, zone j being the zone with the first highest number of hits (the best location). This means ***(fetching)*** Table 3.5.

Table 3.4: Duplication 1 (caching method)

```
Function caching(f,i,j)
        Duplicate file f in zone i
        End
```

Table 3.5: Duplication 2 (fetching method)

```
Function fetching(f,i,j)

        Input k=j                       // zone k is the best location to duplicate file f
        While a_ik > A
                k:=retrieve(f,i)
                Duplicate file f in zone k  // we suppose that there's at least one zone
                                            // k that ensures a_ik =< A
        End
```

Note that, we have to make sure that the duplicated files are not taken into consideration while running the algorithm in Table 3.3, i.e. in line number 11 ***"For f from 1 to p_j"*** f can't be a duplicated file.

Also, we apply the duplication algorithm (Table 3.3) to the 6 files, starting from the file distribution on Figure 3.1.B where every file is located in its best location.

Then, we are going to look at different values of ***Y (0, 5000, 10000 and 20000)*** and compare the efficiency of the 2 duplicating methods for each value of Y. Moreover, we will also see the effect of changing Y value on the total gain associated with the optimization. For all cases of Y, we suppose ***A=5*** unit cost.

3.5 Methods and Numerical Results

This section focuses on performance investigation of the proposed methods for files duplication by applying these two functions on different files from YouTube as selected before. Moreover, we will test different use cases by changing the threshold value Y that supposes to be adjusted by the operators as the following:

3.5.1 First Case Y=0

We apply the Duplication algorithm, and we browse all the zones as the following:

- Customers from zone 1 have access to content from zones 2, 3, and 4 with an access cost that is less than or equal to the threshold A (A=5 in this example) (the access costs are respectively 3, 5 and 4, see Figure 3.1). So there's no need to go further;
- But the access cost to content from zones 5 and 6 is higher than the threshold A (the access costs are respectively 7 and 8). So we look at the files of these zones:
    - Zone 5 contains File 5. We look at the demand on that file from zone 1 customers. Table 3.2 indicates that there are 3800 hits on File 5 from zone 1. 3800 is higher than the demand threshold Y (Y=0) so we duplicate the file: If we use caching, then we duplicate File 5 in zone 1 (the customer's zone). Otherwise, if we use fetching, we look at File 5 column in Table 3.2 to see in which zone we're going to duplicate the file. The zone with the second highest demand on File 5 is zone 3 with 3900 hits. We check the access cost between zones 1 and 3 in Figure 3.1. This cost is 5, which is not higher than the access cost threshold **A** (A=5). We then duplicate File 5 in zone 3. If the access cost between zones 1 and 3 were higher than **A**, then we should have looked at the zone with the third highest demand on File 5, which is zone 1 with 3800 hits.
    - We check zone 6. There are no files in this zone so there's no more work to do for zone 1 customer. We may move to customers from zone 2, and so on.

We get the following file distribution with the caching method. In Table 3.6, 'o' indicates the file's best location, while 'x' indicates the file has been duplicated.

Table 3.6: Files distribution with Y=0 & caching method

	Zone 1	Zone 2	Zone 3	Zone 4	Zone 5	Zone 6
File 1	o				x	x
File 2			o	x		x
File 3		o		x		
File 4			o	x		x
File 5	x			x	o	
File 6		x	x	o	x	x

Let's now compute the gain achieved through the first duplication method, i.e. the difference between the access costs before and after applying the Duplication algorithm, a_{ij}^{f} being the access cost to File f located in zone j from a customer located in zone i.

Gain from the duplication of File 5 in zone 1:

File 5 is no longer delivered to zone 1 customers from zone 5 but from zone 1.

Gain from zone 1 = $(a_{15} - a_{11}) * y_{15}^{f5} = (7-0)*3800=26600$

Customers from zone 2 still access File 5 that is located in zone 5 because it is the zone with the cheapest access cost (a_{25}=2) compared with the other zones that host File 5 (zones 1 and 4 for which the access costs from zone 2 are respectively 3 and 6).

Gain from zone 2 = 0

Customers from zone 3 still access File 5 that is located in zone 5 because it is the zone with the cheapest access cost (a_{35}=1) compared with the other zones that host File 5 (zones 1 and 4 for which the access costs from zone 3 are respectively 5 and 7).

Gain from zone 3 = 0

Customers from zone 4 access File 5 that is located in zone 4 because it is the zone with the cheapest access cost (a_{44}=0).

Gain from zone 4 = 0

Customers from zone 5 access File 5 that is located in zone 5 because it is the zone with the cheapest access cost (a_{55}=0).

Gain from zone 5 = 0

#Customers from zone 6 still access File 5 that is located in zone 5 because it is the zone with the cheapest access cost (a_{65}=4) compared with the other zones that host File 5 (zones 1 and 4 for which the access costs from zone 6 are respectively 8 and 6).

Gain from zone 6 = 0

We repeat the same operation for the other duplicated files and get the following file distribution with the other method "fetching" as shown in Table 3.7.

We notice that with caching, the majority of the duplicated files are distributed among zones 4, 5 and 6, while with fetching zones 1, 2 and 3 contain all the duplicated files.

After computing the new access costs, we compare the gain achieved through the duplicating methods in the figures below (as shown in Figure 3.2).

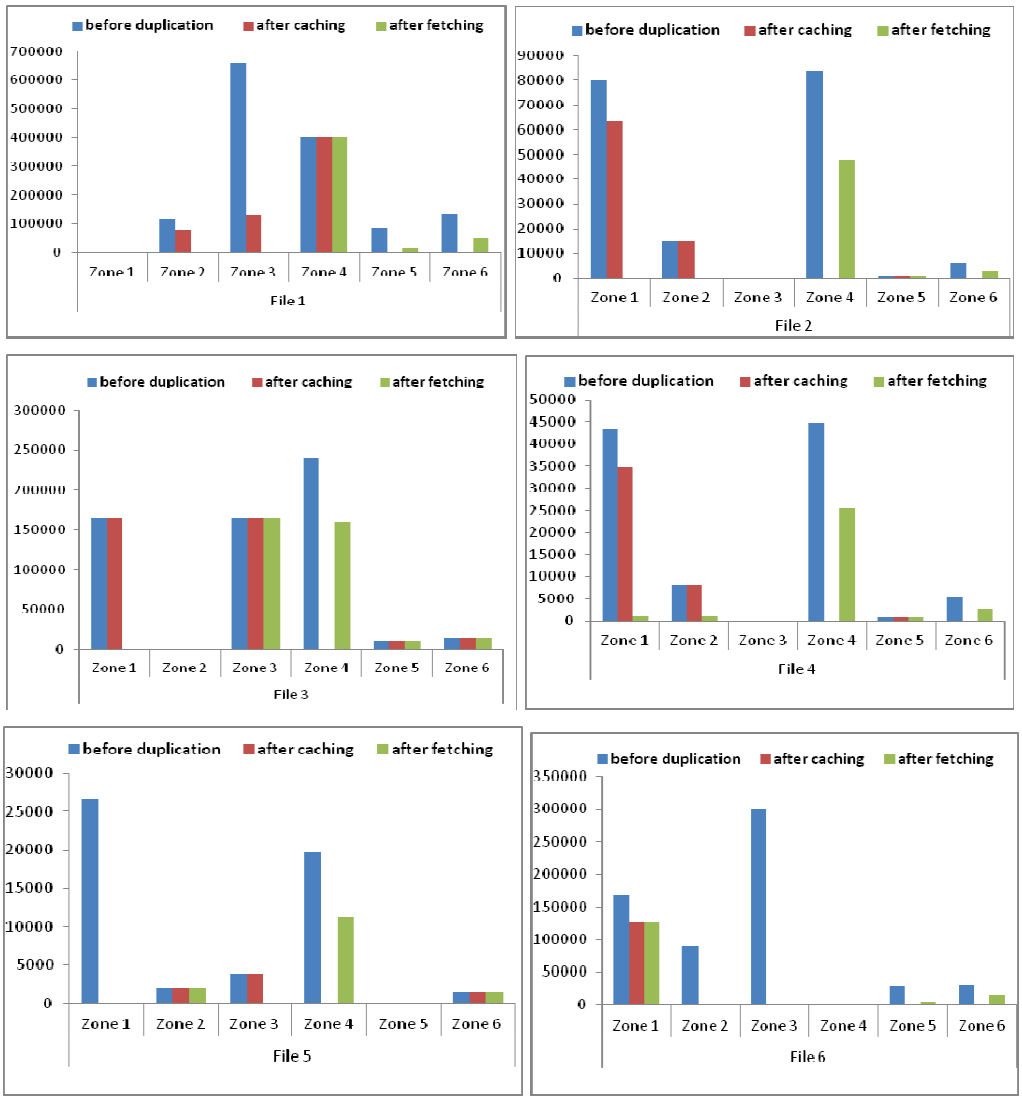


Figure 3. 2: Access cost before and after caching and fetching for Y=0

Table 3.7: File distribution with Y=0 & fetching

	Zone 1	Zone 2	Zone 3	Zone 4	Zone 5	Zone 6
File 1	o	x	x			
File 2	x	x	o			
File 3	x	o				
File 4	x	x	o			
File 5	x		x		o	
File 6		x	x	o		

If we look at file 1, we notice that the sum of the access costs from all the zones after caching (the sum of the red columns) is higher than that after fetching (the sum of the green columns). Therefore, there's a greater gain with fetching than with caching to access file 1 from all over the network.

For the delivery of file 1 to zone 4, no gain was achieved. Zone 4 customers still access file 1 from the same zone (zone 1 here). In other cases, there may be a duplication of that file on another zone that has the same access cost to zone 4 as zone 1 (the best location of file 1).

We notice that fetching is more efficient than caching for files 1, 2, 3 and 4 and zones 1, 2 and 3, while caching is better for files 5 and 6 and zones 4, 5 and 6.

3.5.2 Second Case Y=5000

For this threshold, we notice that; with caching algorithm, the majority of the duplicated files are distributed among zones 4, 5 and 6, while with fetching algorithm, zones 1, 2 and 3 contain all the duplicated files, like for Y=0.

In the Figure 3.3 below, we only show the files that were duplicated by any of the two duplication methods.

From Figure 3.3, we notice that fetching is more efficient than caching for files 1, 2, 3 and 4 and zones 1, 2 and 3 (except for file 6), while caching is better for file 6 and for zones 4, 5, 6, even if the two methods are of equal efficiency on certain zones.

We also notice that file 5 was not duplicated because there was not enough demand for it. The only zone from which the demand exceeds the demand threshold Y (5000) is zone 5 (which is the best location of file 5) with 12400 hits (as shown in Table 3.2).

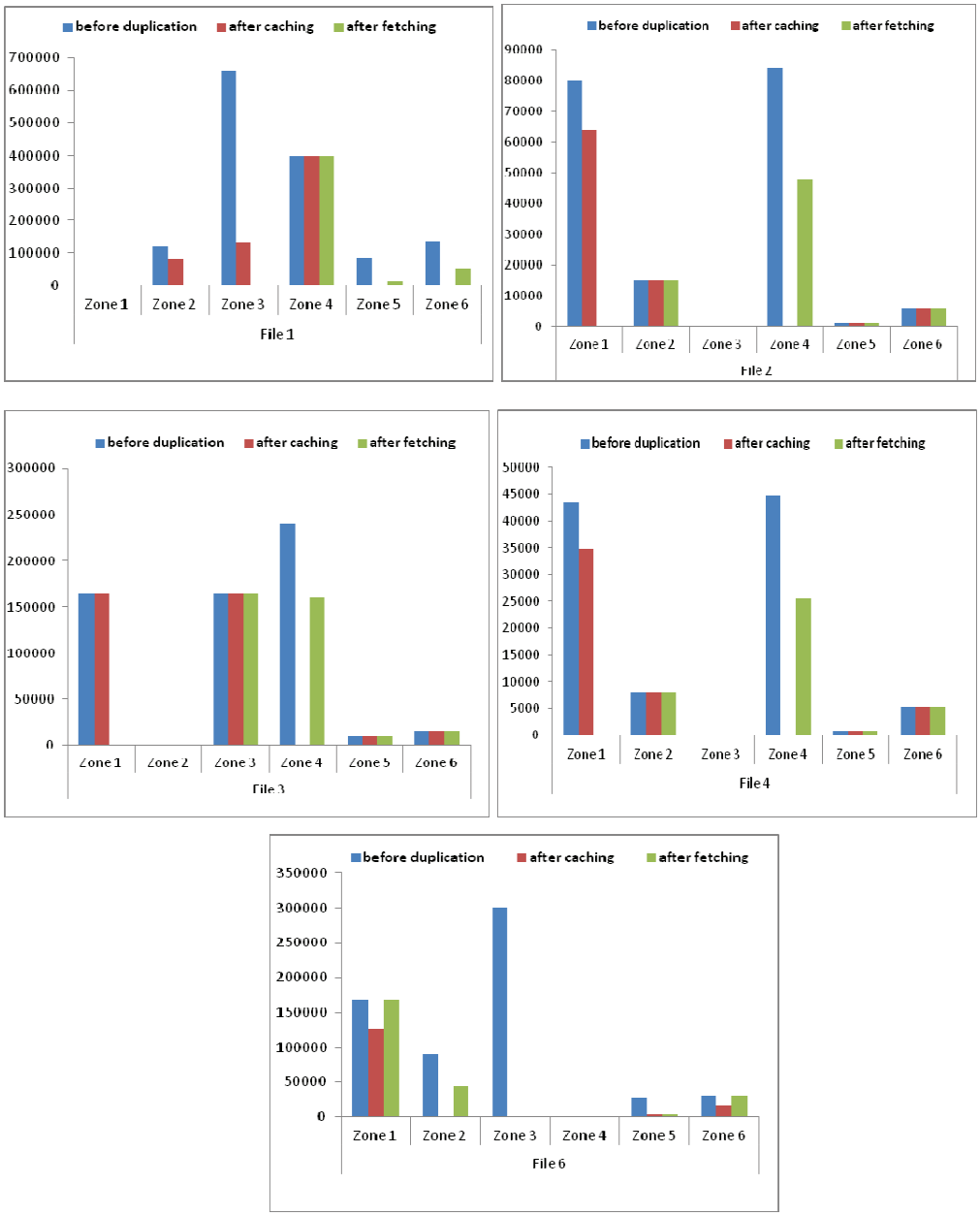


Figure 3. 3: Access cost before and after caching and fetching for Y=5000

3.5.3 Third Case Y=10000

After applying the two methods, we get the following files distribution either for caching or for fetching as shown in Figure 3.4.

Like for Y=0 and Y=5000, we notice that with caching, the majority of the duplicated files are distributed among zones 4, 5 and 6, while with fetching zones 1, 2 and 3 contain all the duplicated files.

This is understandable for fetching as the highest demands are those of Files 1, 2 and 3, and these files are located in zones 1, 3 and 2 respectively. It is also understandable for caching as the access cost from zones 4, 5 and 6 to Files 1, 2 and 3 is high compared to access costs between zones 1, 2 and 3.

In the Figure 3.4 below, we only show the files that were duplicated by any of the 2 duplication methods.

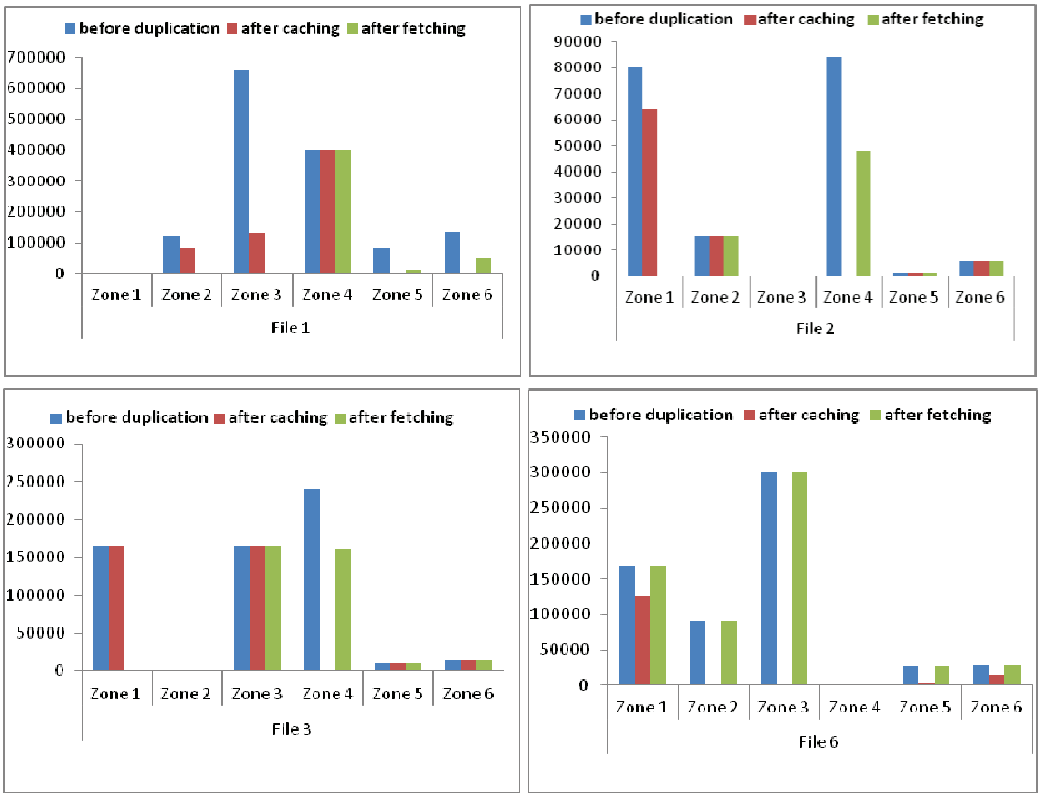


Figure 3. 4: Access cost before and after caching and fetching for Y=10000

From Figure 3.4, we notice that fetching is more efficient than caching for Files 1, 2 and 3 and zones 1, 2 and 3 (except for File 6), while caching is better for File 6 and for zones 4, 5, 6, even if the two methods are of equal efficiency on certain zones.

We also notice neither File 4 nor File 5 was duplicated because there's not enough demand for them. The highest demand for File 4 is 8800 (see Table 3.2) and doesn't exceed Y (10000), and the only zone from which the demand exceeds Y for File 5 is zone 5 (which is the best location of File 5) with 12400 hits (see Table 3.2).

3.5.4 Fourth Case Y=20000

This scenario represents the highest threshold value for **Y** as set by the operator. As we applied it, we got the following file distribution for the two methods as shown in Figure 3.5 for files 3 and 6.

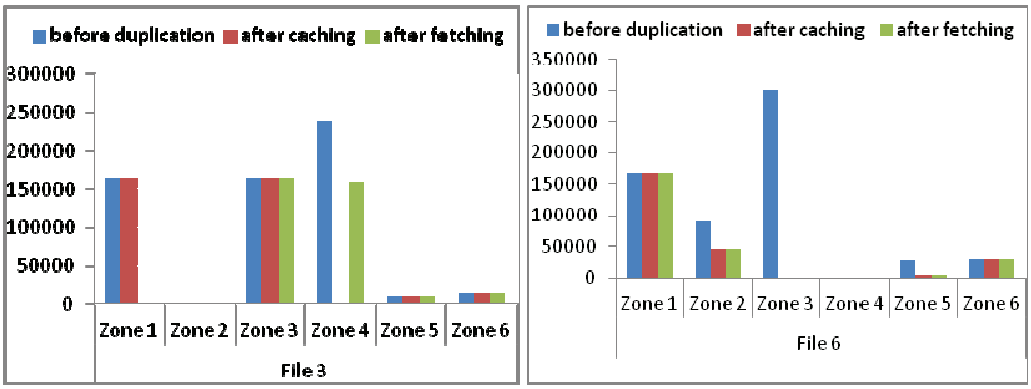


Figure 3. 5: Access cost before and after caching and fetching for Y=20000

We notice that fetching is more efficient than caching for File 3 and zones 1 and 2, while it has the same efficiency as caching on File 6 and zones 3, 5 and 6. Caching is more efficient than fetching only on zone 4.

Here, despite the fact that there's a demand higher than Y (20000) on File 1 from zones other than its best location (zone 1), such as zone 3 (132000 hits), zone 4 (100000 hits) or zone 2 (40000 hits), File 1 was not duplicated. This is due to the access cost threshold A for zones 2, 3 and 4, and due to the demand threshold Y for zones 5 and 6 as there's not enough demand on File 1 from these 2 zones (12000 and 17000 respectively).

3.6 Hosting cost and Net Gain

In this part, we take into account the file duplication cost. This cost is mainly due to file hosting. We assume the following parameters:

- A hit cost is 0.01 m.u. (money unit). Then, we multiply the access costs by 100 to have all the costs expressed in m.u. (just to scale the values) ;
- 1 TB (unit size in Bytes) hosting cost is 20 m.u. ;
- The files are sets of files, each set is 100 TB; in fact, the hosting servers contain a great number of videos, and the network operator may duplicate sets of files

instead of running the Duplication algorithm for every single file. These sets may encompass files that have almost the same viewers (all the episodes of a TV show for example).

*The hosting cost = number of duplicated files * file size in TB * 1 TB hosting cost.*

Table 3.8: Number of duplicated files with both duplication methods for different thresholds

Y	0	5000	10000	20000
Number of duplicated files through caching	13	7	6	2
Number of duplicated files through fetching	11	5	4	2

We notice that the number of duplicated files decreases if Y increases. This is due to the fact that only the most popular files are duplicated with a high value of Y as shown in Table 3.8.

Now if we compare the gain in access cost and the hosting cost (see Figure 3.6), we notice that there's a financial loss for Y=0 with both duplication methods. This is understandable as duplicating files that are not popular enough require important hosting resources and benefit to a small number of customers. Hence we need to compute the net gain, which is the difference between the gain in access cost and the hosting cost, in order to properly assess the efficiency of both duplication methods.

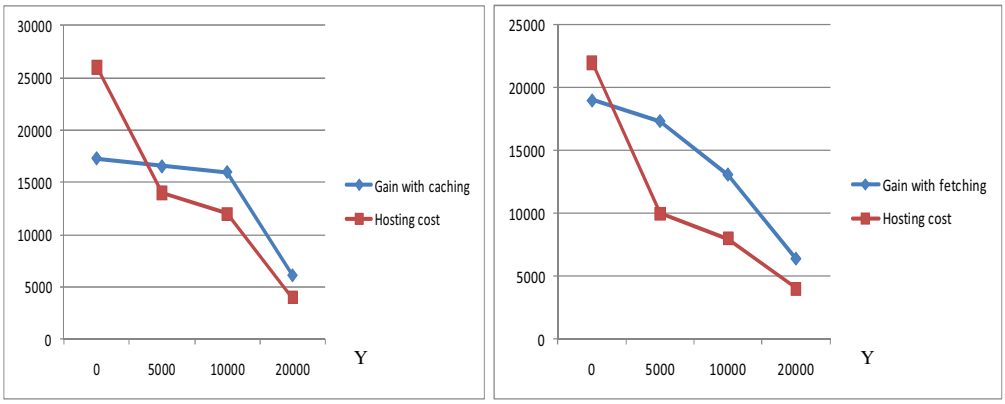


Figure 3. 6: Gain vs. hosting cost

With caching, there's a gain starting from Y≈4000 and the biggest net gain is achieved for Y=10000 according to Figure 3.7 (Net gain). While with fetching, there's a gain starting from Y≈2000 and the biggest net gain is achieved for Y=5000 according to the same Figure 3.7.

Moreover, we notice that fetching is more efficient in terms of Net gain, despite the fact that caching is better for Y between 6000 and 19000 in terms of Access gain. This is due to the fact that the duplicated files are smartly allocated with fetching, allowing more customers to access them with a minimum cost, and because there's a lower need for duplicating files than with caching.

We also notice that beyond a certain value of Y, the net gain decreases, and can even become negative beyond a certain limit (> 20000 in our example). Moreover, as Y increases beyond 10000, the advantage of fetching over caching diminishes as shown in Figure 3.7.

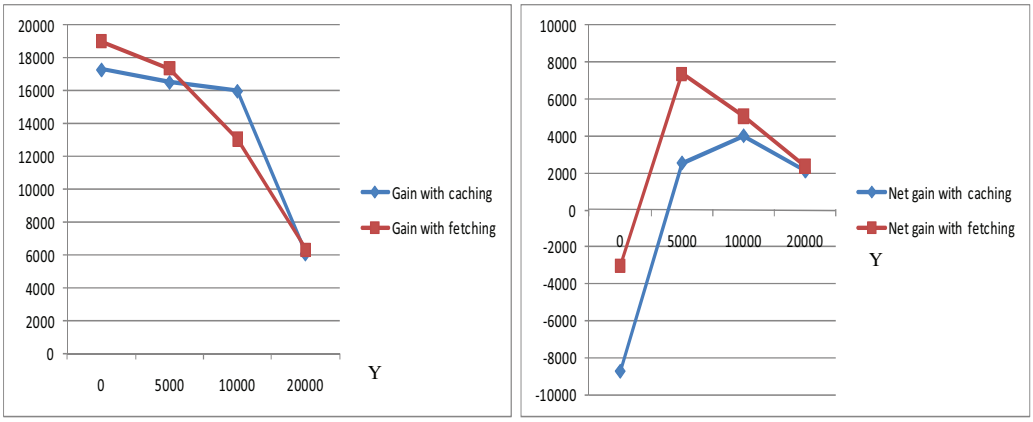


Figure 3. 7: Total gain and net gain comparison

3.7 Conclusion

This chapter provided a study for video files distribution allocation. We started by highlighting some techniques used in video files caching and distributions. Then, we proposed two mechanisms of files duplications based on files hits and some cost assumptions set by the operators.

Despite the fact that fetching is better in terms of net gain, both duplication methods can be more or less efficient, as we have seen that caching is more efficient for the delivery of File 6 for Y=0, 5000 or 1000, and has equal efficiency with fetching for Y=20000. We have also seen that, setting the right demand threshold Y is a key step to achieve cost optimization, along with the access cost threshold A (A=5 in this work) that we didn't discuss.

We can also combine duplication methods or use new ones for better results. We can, for example, duplicate a file in the zone that has the second and the third highest number of hits on that file and that grants an access cost lower than A at the same time.

However, the results found in the treated examples may vary depending on the network configuration (the access costs, the number of files and their distribution) and a duplication method that proves to be efficient for a given network may not work for another one.

In the next part of this thesis, we will consider the managed service of video (i.e. IPTV) and handle it as an open aspect with the optimization of operators' networks through Chapters 4 and 5.

Chapter 4: Open-IPTV Services and Architectures

This chapter presents the IPTV architecture and its evolutions. The logical progression of this technology should lead to the integration of nomadic services and a generalization of cloud computing as a substrate for better and fast service provisioning.

The step from IPTV to Open IPTV includes Nomadism but also several improvements such as obtaining the same home services like (LIVE-TV, VOD and WEB-TV) anywhere, anytime and on any personal device. Moreover, this approach allows us to obtain the service from any operator in a transparent means to users. Most of IPTV operators either in Triple-play or Quadruple-play services provides the IPTV services based on physical Set-Top-Box (STB) devices, restricting the users' presence in the domestic sphere. Open-IPTV aims at extending the access to services outside the domestic sphere and outside the home network. Moreover, this chapter presents the Open-IPTV motivations, concepts, services and possible architectures. Also, cost analysis issues are discussed a new architecture driven by the collaborative model. Finally, some Nomadic IPTV use-cases scenarios are illustrated.

4.1 Introduction

While the cost of deploying delivery network solutions for IPTV has increased over the last several years, the operational expense of maintaining and managing the network also continues to rise. That is why; we are searching for more open and distributed IPTV service components.

The rest of this chapter is organized as: Section 4.2 highlights some definitions and terminologies about IPTV. Section 4.3 introduces the components of the open IPTV architecture and its analysis in flow control way. Section 4.4 compares between different ways of content transmissions and conducts a Testbed for IPTV. Section 4.5 differentiates between the current model for business IPTV delivery and the proposed

one for collaborative delivery based on cloud network for domestic region. Section 4.6 handles some issues relevant to access IPTV in Nomadic situations. The model analysis, motivations from cloud design, collaborations cost analysis and some use cases in different nomadic situations are discussed in Section 4.7. Section 4.8 concludes the study of this chapter.

4.2 Definitions and Terminologies

The current status of IPTV model can be summarized in Figure 4.1. We have three models; IMS based standard model using the IP Multimedia Subsystem (IMS) [35] core as a controller, NGN [36] based standard model based on Next Generation Network architecture and finally the Internet model: Google TV [37]. We have combinations of these infrastructures like Digital Video Broadcasting DVB. The DVB project could use either IMS or NGN so we can call it a hybrid model. In DVB-IPI-based architecture, the DVB-IPTV service is the video service provided over IP like TV over IP or the Video-on-demand (VOD) over IP as specified in [38], [39] and [40].

We expect the future to have collaborative model (Cloud-Based) for IPTV delivery. It will have advantages over the other models in terms of low investments cost, better delivery performance and converged system in design.

4.2.1 Definitions

The Internet Protocol TeleVision (IPTV) term has many definitions. But, the ITU-T definition [41] is the more general one: “IPTV is defined as multimedia services such as television/video/audio/text/graphics/data delivered over IP based networks managed to provide the required level of QoS/QoE, security, interactivity and reliability.”

Actually, there are two models for general IPTV management:

- **The Managed Model:** It concerns access and delivery of content services over an end-to-end managed network by the operator (like Orange or Free Triple-play operators in France)
- **The Unmanaged Model:** It concerns access and delivery of content services over an unmanaged network (e.g., The Internet) without any quality of service QoS guarantees. YouTube represents one such type of this model.

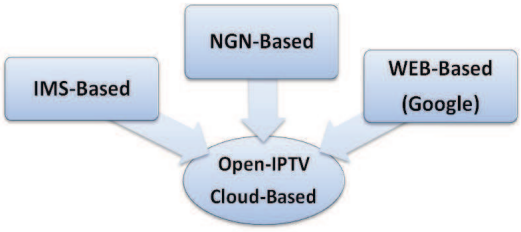


Figure 4.1: IPTV collaboration models

4.2.2 Open-IPTV Services

The Open IPTV means offering TV over both managed and unmanaged networks. The terminal (TV) is also modified to accommodate built in IP services. To obtain an open IPTV service, we need a new consideration of the Set-Top-Box (STB). In Figure 4.2, we differentiate between the current proposed services under the managed and unmanaged networks.

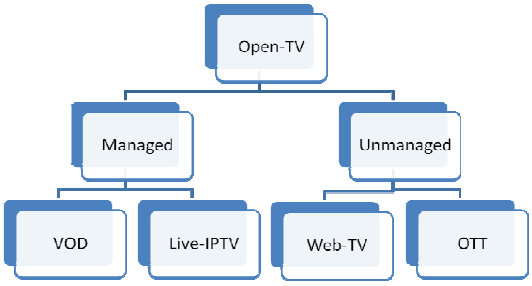


Figure 4.2: Open TV components

4.2.2.1 Physical Set-Top-Box (P-STB)

This system represents the actual implemented scenario. It mainly depends on physical Hardware of STB and leased connection between the consumer and content provider.

Advantages: The service security assurance and bandwidth guarantee are advantages of this model. Also, the good management of STB provided by the content providers is naturally guaranteed.

Drawbacks: With the present model of IPTV, the delivery is based on physical STB restricted to specific location. But, as the consumers are increasingly becoming mobile, they demand bandwidth regardless of their locations to satisfy their entertainment. So, the lack in this model is the inability to support Nomadism (Mobility and Nomadic Access aspects).

4.2.2.2 Software Set-Top-Box (S-STB)

STB in this case depends on Software instead of Hardware for controlling the received channels and videos. It could be a USB disk carrying all necessary information.

Advantages: It will satisfy the consumer desires for enjoying all their subscription videos anywhere. So, this type is highly recommended for nomadic access services.

Drawbacks: The operators cannot be in control of everything. Management of user policies and authorization while changing the location is challenging.

4.2.2.3 Virtual Set-Top-Box (V-STB)

It is a kind of S-STB but resides in the operator premises instead of user side. So, it is like an HTTP application accessed remotely by the client.

Advantages: The operator could control everything easily. For the client, no prerequisites are needed for his system to start accessing the service except for a simple IP connectivity.

Drawbacks: Delay between the client application and the STB server is the big issue. Also, some security problems for user identification can be found.

4.2.2.4 Open Set-Top-Box (O-STB)

It is a kind of hybrid STB that groups features from P-STB and S-STB. The O-STB can be accepted to run for any operator. Also, it can accept many kinds of video services like managed IPTV, Web TV, Social Networks (like YouTube) and VOD service.

Advantages: It will have an easy deployment manner and end of compatibility issues. Also, it is modern and suitable for new style of life.

Drawbacks: It may suffer from some complexity in design.

4.2.3 Some IPTV Terminologies

The common IPTV categories used could be classified as:

- **Pay-TV:** This service refers to the subscription-based TV service delivered in either traditional analog forms, digital or satellite. In different countries, we have similar terms referring to ‘Packs’ and channels like Canal-Satellite, Showtime, ART and so on.
- **TV-OTT:** TV Over-The-Top; it is one of the American TV services that provides a seamless consumer experience for accessing linear content through

the broadcast network on a TV set, as well as non-linear services such as Catch-up TV and Video on Demand (VOD) through a broadband IP network. It is also designed to allow the provider to extend content and the consumer experience to additional platforms including PCs, mobiles, gaming consoles and connected TVs.

- **IPTV ‘‘Follow-me’’:** It allows the user to continue access his IPTV service while moving and changing his screen (content adaptation while Mobility is an issue).
- **Personal IPTV "My Personal Content Moves with Me":** It allows the user to access his personalized IPTV content in any place in his domestic region with the reception of the bill on his own home subscription (like Nomadic Access).
- **TVA:** TV-Anytime is developed by a specific IPTV group [42] which is interesting in the interoperability and security for future TV.
- **Open IPTV:** It is a model of TV service that will be based on borderless technology. A hybrid model that merges the traditional Broadcasting TV with the Web-TV in one thing. OIPF (Open IPTV Forum) [43] is a well known group in this field.

4.3 Open-IPTV Architectural Components

Open IPTV term could be defined as: an integral solution for both managed and unmanaged IPTV services. It could be considered as a kind of TV anywhere/anytime but it goes beyond the subscriber domestic home region to include all possible access.

We can follow the state diagram in Figure 4.3 which explains the steps of accessing video service based on an open IPTV architecture. For the details see ***Appendix A***.

4.4 Content Transmission Forms

The modern flow control of contents in data centers or cloud computing environments will depend on OpenFlow [131]. Through this communication protocol, the forwarding plan of data will be simple and easier in managements by network elements. Moreover, it will help the organization in building virtualized networking infrastructure. Then, the

customization and programming of this infrastructure will be easy and adaptable according to the needs.

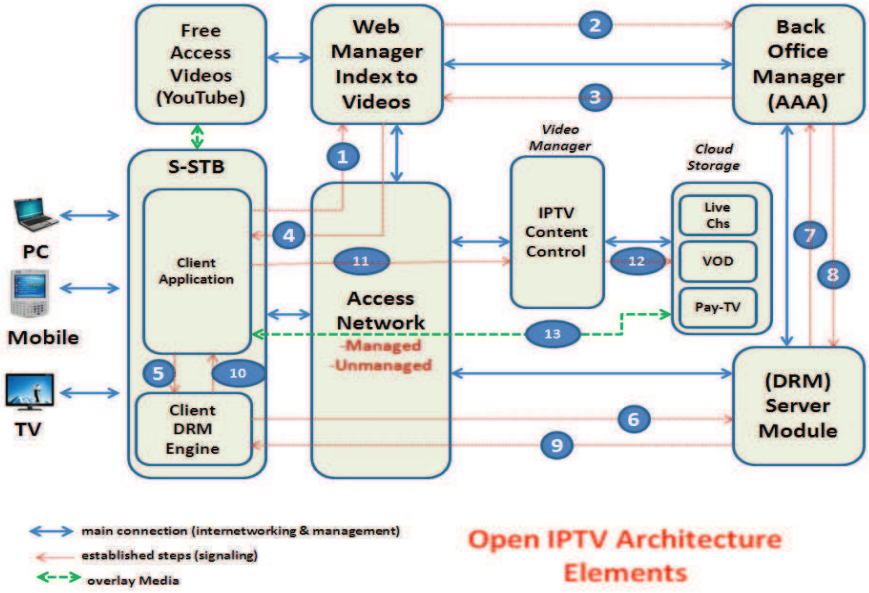


Figure 4.3: Flow Control for Open IPTV architecture

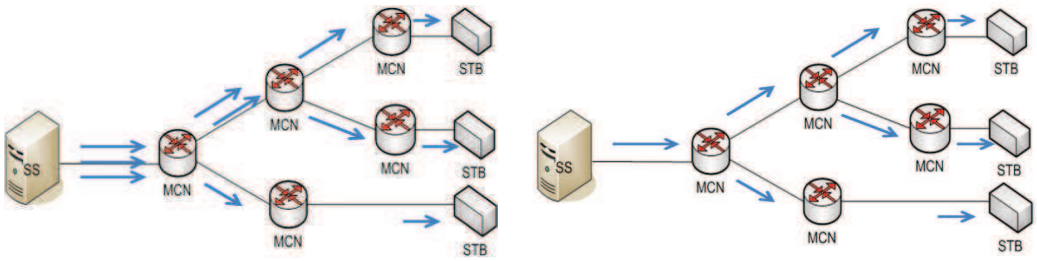
For classical IPTV, it has different forms of transmission that depends mainly on the type of access (Managed or Unmanaged).

- HTTP/TCP (for WEB-TV)
- Unicast/RTP/UDP (for VOD)
- Multicast/UDP (for LIVE-TV)

The first two scenarios could be controlled by using unicast transmission and the last one is controlled using multicast techniques in IPTV business model as the following:

4.4.1 Unicast versus Multicast Transmissions

Unicast is a point-to-point transmission methodology as shown in Figure 4.4-A. So, it is not efficient for video streaming. If we have N clients that join the server (Streaming Server SS) then, this server must handle N sessions for the same copy of data packets. While Multicast is an efficient way for transmission as only one copy of traffic is generated and passed to all devices inside the paths between source and destination. Also, there is no linear increase in traffic as the number of client increases as shown in Figure 4.4-B.



A): UNICAST transmission B): MCAST transmission

Figure 4.4: Unicast versus Multicast transmissions

4.4.2 Testbed Validation

In this section, we focus on validating the open-TV aspects by implementing the following three scenarios:

1. WEB-TV (using HTTP/TCP as a reliable transmission protocol)
2. VOD (using HTTP/TCP as a reliable transmission protocol)
3. Live-TV (using RTSP and UDP)

The objective from this Testbed is to validate different scenarios of IPTV access. Through the first scenario, we conduct the test based HTTP protocol to validate the VoD. Then, we conduct live-TV using RTSP to confirm the live access of IPTV without multicast. In conclusion, the Testbed validates all IPTV scenarios based unicast mode. The details of this Testbed and all analysis are shown in ***Appendix A***.

4.5 Open IPTV Business Model

Cloud computing is a concept that fits well into an IPTV architecture. The concept of collaborative resource has been discussed in different works such as the one presented in [55]. The authors proposed the concept of Alliance as a general aspect of Virtual Organizations. Their Alliance concept is based on integration and collaboration between clients' requests or demands and providers for resources. Moreover, they study the motivations from reforming the distinction in the current situation of organizations that will lead to good business model. The work mainly discussed the collaboration problems and some security aspects towards virtual organizations. Also, the work in [56] proposed the idea of on-demand cloud service within IPTV based servers'

virtualization. But, this work did not touch the area of domestic collaboration between different providers. So, what are the factors that affect design of an open IPTV model?

4.5.1 Design Factors

We believe that two factors press on the providers decisions while taking a new infrastructure investment:

- *Capital Expenditure (CAPEX):* It is representing the cost of network foundation and all non-consumable system devices and infrastructure.
- *Operational Expenditure (OPEX):* It is representing the running cost for provider network including all cost of operation and maintenance.

For the long term investments, the operators will reduce those costs in the domestic Cloud. Moreover, the new added services related to quality and interactivity will be costless.

4.5.2 Collaborative Architecture

The competitive space between different IPTV operators pushes them to implement high similarity in clients' services. This means that, the majority of VoD and IPTV channels are the same corresponding to the culture and social interests of each country. Thus, if we make some convergence between the different providers, it will not affect the overall policy of these operators. Moreover, it will enhance the service delivery and reduce incremental cost for future services investments.

4.5.2.1 Current Architecture

The current architecture of content providers (as shown in Figure 4.5 as France case study) has mainly three layers:

Layer 1: Interconnection Layer (Infrastructure Providers, like France Telecom in France)

Layer 2: Control Layer (Content Providers islands, like Orange, Free …)

Layer 3: Management Layer (The 3rd party hosting and cashing videos and channels servers, like Akamai [17])

The isolations and different islands are the main features of this model. Each operator has a huge investment and local management for the same services that provided by all other competitors in many cases.

The open models require an open infrastructures design and also an open management policy. So, the providers must avoid their selfishness and think for two important things:

- The great benefits from the collaboration through cloud computing for example
- The satisfactions of consumers with new services

Thus, the Open-IPTV model needs cooperation between different partners for achieving success. The next section highlights the collaborative architecture.

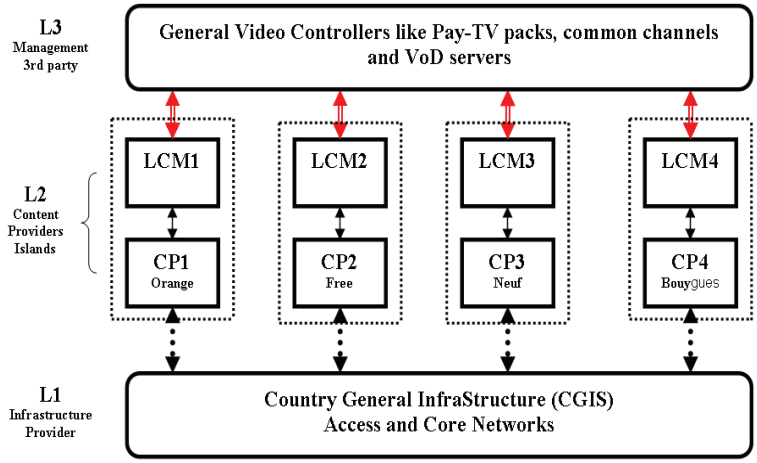


Figure 4. 5: The current model of different islands content providers.
CP: Content Provider, LCM: Local Content Management and GCM: General Content Management

◀•▶ : is the interface between CPs access network and core network

⟺ : is the interface between LCMs and GCM

4.5.2.2 Collaborative Open Architecture

The collaborative model design is illustrated in Figure 4.6. This model proposes more interactions and collaborations between different operators. As mentioned, the S-STB is the best match for IPTV delivery; we use it as the interface point to multi-screen access client.

The collaboration exists in the form of common access to CCI and DGCM layers by the client access layer. This case will lead us to a new methodology of accessing called Resource-On-Demand (ROD) that will save the time and cost for service configuration. Moreover, UAR and UAA processes for clients' services authentication and authorizations pass mutually and independently of user access network. The Content Adaptation (CA) for different screen has two aspects; one based the capability of S-STB and the other on the access device specifications. The CA process for the domestic sphere for the client is mainly done by STB.

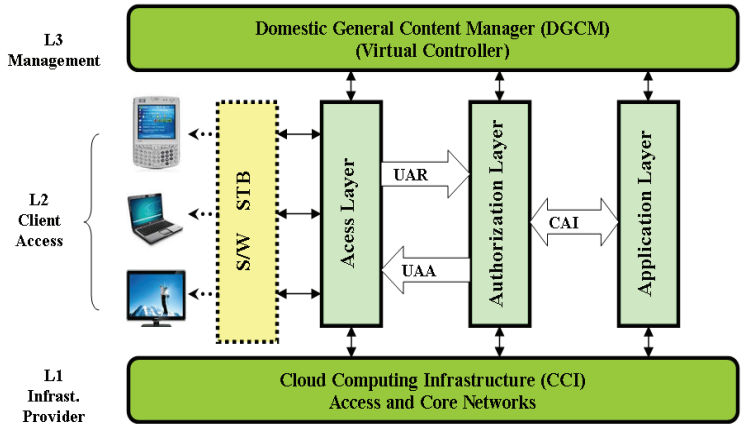


Figure 4. 6: The proposed collaborative model for domestic cloud network to new IPTV system.
UAR: User Authorization Request, UAA: User Authorization Answer
CAI: Content Adaptations Interactions

4.6 Nomadic IPTV Services

Most of IPTV operators either in Triple play or Quadruple play services provide the video services based physical Set-Top-Box (STB) device. This box mainly restricts the users to be inside their homes for having the service. Some operators give their clients the rights to access their IPTV service based on PC or Mobile phones devices [50] inside their own managed networks using something like VLC media player [47]. This access mainly based on RTP/UDP stack because the UDP is fast but it is unreliable.

The operators in this scenario could guarantee the network reliability because the network is managed (end-to-end). But, the customers still need to have the same access outside the home network which means anywhere (Nomadic Access).

The Nomadic Access to Nomadic Service (NA-to-NS) aspect has four elements that need to be clarified:

1. **Nomadic Access:** The Nomadic Access is the way of accessing a home service outside the home network.
2. **Nomadic Network:** The Nomadic Network is the network that provides network access for roaming and remote users to the home network or the home services. Remote users are those with a portable, desktop PC or DTV at home that is already connected to the Internet (for example, via a domestic ISP) and need to access the same subscribed service in nomadic situation.
3. **Nomadic Service:** The Nomadic Service is the service that can be obtained from a nomadic network and can be accessed independently of the home service

provider. The nomadic service may be a service provided by the nomadic device to the network.

4. **Nomadic Device:** Traditionally, it was mainly the mobile devices but it could be extended to include any digital or IP devices like DTV, PC or any device that can obtain the service behind an anonymous IP-STB.

4.6.1 Nomadism & Roaming

'Nomadism' is an equivalent term to 'Roaming' but under some conditions. So, it is a kind of mobility called 'Personal Mobility' which means; the user could access his service in any place under 'any access network' by using same credential or accounts. If the service is obtained outside the operator network based on mobility of Mobile phones it will be considered as Roaming. But, if the service is obtained either from same operator or from different operator based on any access device, this can be considered as Nomadism. So, Nomadism can be defined as the global definition for Roaming services and we can call it service 'Portability'. Figure 4.7 "use-case 4" represents the Nomadism in UP-TO-US project [57] from the perspectives of IPTV service personalization.

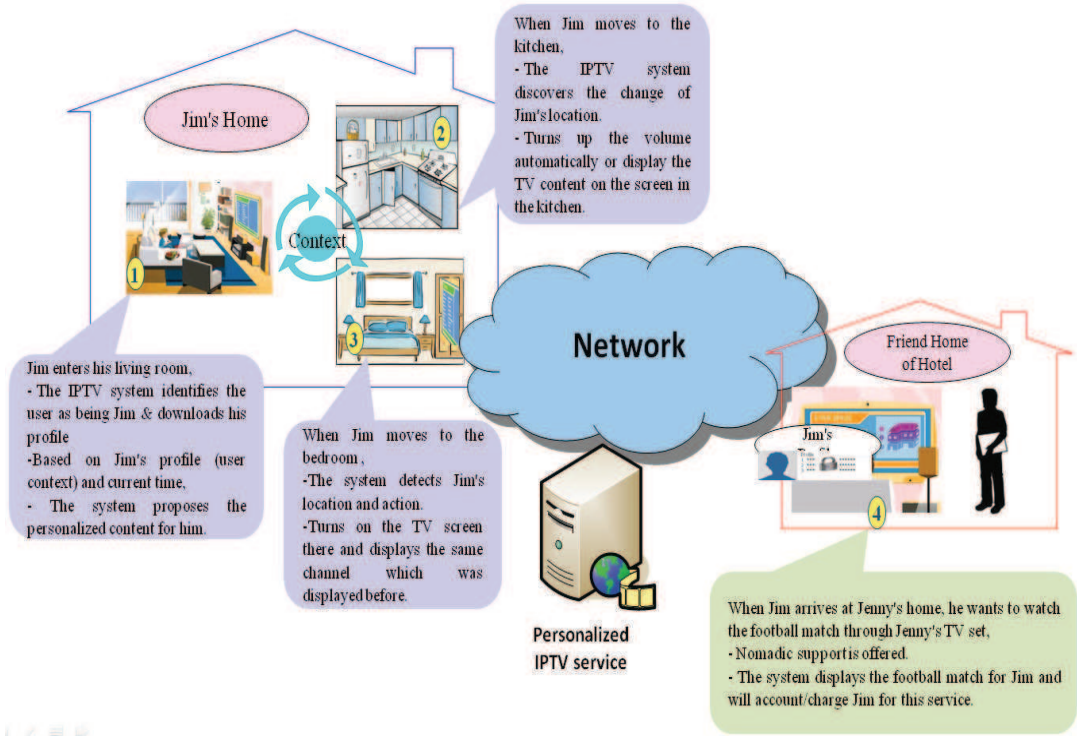


Figure 4. 7: UP-TO-US project Nomadism

4.6.2 Nomadic Issues

1. **Network Address Translation (NAT):** The main issue in the all-IP based home network is the NAT problem. Because in the nomadic situation the user may have a private IP address and this adds more complexity to build VPN. Also, if it is possible to build the VPN with NAT we may have some problem relevant to the service itself and the ports numbers used.
2. **Privacy Problem:** The privacy is a big issue in the nomadic network. All user need to use the minimum sensitive information while they are in nomadic situation. This is to keep and guarantee your privacy and prevent others from knowing your sensitive information and personal choices.
3. **Digital Rights Management (DRM):** All operators guarantee the DRM issues while providing their IPTV service by depending on the physical STB settled at the user home. But, for nomadic situations, they have some troubles for controlling the DRM and on which criteria they will build their guarantees.
4. **Cross Domain Authentication (CDA):** Obtaining a nomadic access rights is similar to the roaming service in mobile. So, the authentication can be regulated by the home network AAA servers or by the visited networks according to the roaming policy. Cross Authentication was provided by the roaming group in [51] [52] [53].
5. **Multicast Tree Structure:** As the live TV is diffused based on multicast technology like PIM-SM [44] multicast routing protocol with management of the clients joining and leaving by IGMP [54], this tree could be changed and updated so as to optimize the operator network performance. This will require adopting a new way of rearranging the nodes and the MCAST trees (***Chapter 5***).

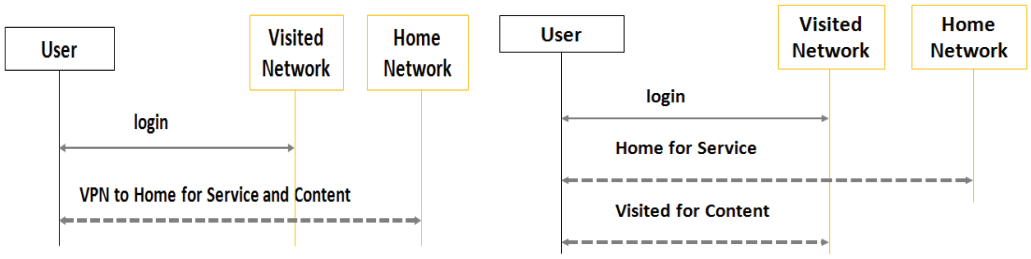
4.6.3 Nomadic Access

For accessing nomadic services from any place we mainly have two ways:

1. **All Based Home Network:** In this scenario, both service and content could be provided by the home network. So, the nomadic network uses specific software to connect to the home network and to forward the service. This connection establishment has two phases. In the first phase, the client connects either to his ordinary Internet Provider (from home) or to the nomadic network using any dial-in point-to-point connection like PPPoE (PPP over Ethernet). Then, the

PPPoE makes a connection to a LAN, and provides authentication, authorization and accountability issues for this client from his home network. After the user passed this verification phase over multiple Internet providers and had the settings to specific service, he could make a VPN (Virtual Private Network) connection, using for example PPTP (Point-to-Point Tunneling Protocol). This will encrypt the traffic by passing it through a 'tunnel'. This tunnel is secure so that other people can't intercept the traffic and read it as it passes across the Internet or over different wireless connections as shown in Figure 4.8-A.

2. **Service Based Home Network:** Service is provided by the home network while content is provided by the visited network or nomadic network as shown in Figure 4.8-B. In this scenario, the client has to pass a secure authentication and authorization connection with his home service providers. Then inter policy between providers will give the content requested from the very near point to the visited network.



A): All based Home B): Service Based Home

Figure 4. 8: Nomadic access ways

Those two ways could be supported by an access from managed network or unmanaged network. Each type has its pros and cons as the following:

- **Access under Managed Network**

By using a specific infrastructure built and managed by the operators, we can guarantee the QoS, security and privacy. Also, we can apply multicast transmission for scalability issue. On the other hand, this way suffers from high access cost and locality of access which means having the service strictly to the physical STB.

- **Access under Unmanaged Network**

The Unmanaged network is including any Internet access like Internet cafe or any shared DSL connection. This way gives low access cost and high portability which

means it is easy to access the service from any place. Apart from those two advantages, the QoS and security have some limitations and can not be guaranteed. Also, we can not apply multicast in transmission.

4.7 Collaborative Architecture for Open-IPTV Services

To have collaboration, we need to have convergence. Figure 4.9 illustrates the relations between converged networks 'Infrastructure' with converged service 'Management Tools'. The access networks and storage or caching systems are integrated with the converged infrastructure. Media servers and management tools or software are however still in need to move towards convergence. This is due to the isolation islands for each operator systems.

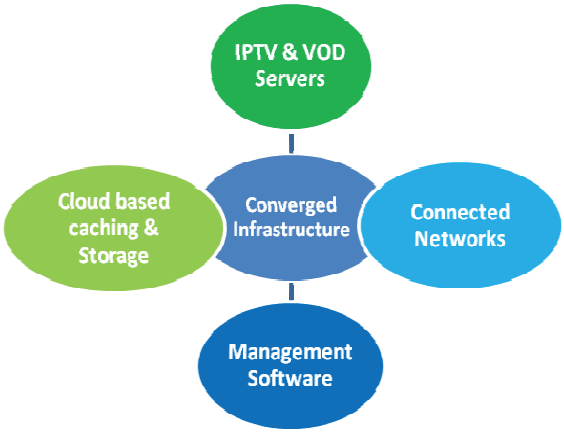


Figure 4. 9: Future Network-Service Convergence

4.7.1 Cost Analysis: CAPEX vs. OPEX

All direct and indirect infrastructures and servers costs are representing the whole part of CAPEX cost. Cloud provides an economic solution for data centres design [34]. We estimate that, for the Cloud design under collaboration, it can have more reduction of the essential costs till high percent of the total cost. Moreover, the cost reduction mainly depends on the degree of collaboration between the providers in the same domestic region.

Let the following assumptions:

N: is the number of providers' links in the domestic region

C: is the total cost for each provider

Then, the total CAPEX for all providers = ***N*C***

For collaboration model it will equal: ***N*c/N*C***

Where $c \approx L*C^2$ (c: is the cost of link between two operators as a result from N-Square problem). So,

$$CAPEXc = (L*C^2)/C = L*C$$

Where: CAPEXc: is collaboration CAPEX cost

L: is the infrastructure Link foundation between providers after collaboration.

So, the profit (***Revenue R***) from collaboration as a reduction in CAPEX:

$$R = N*C - L*C = (N-L)*C$$

If we have a third parity (Cloud provider), then the CAPEX for the current operators will be Zero and all costs will just be OPEX costs.

Therefore, the typical cost optimization regarding to traditional data center design versus Cloud design is really remarkable. The long term costs will be reduced. Also, the benefits from adoption of Cloud are the utilization principles of servers based needs. This means that, the infrastructure is used only when there is a real need and it is released otherwise.

4.7.2 Open-IPTV Use-cases Analysis

This part mainly discusses some analysis about the two use cases for accessing nomadic IPTV service validated in UP-TO-US project [57] as the follows:

- *User tries to access his subscribed channels and videos from another place but relevant to his operator. Using same URI host part in this address (userA@operatorA.com) for operatorA network but from userB access device STB.*

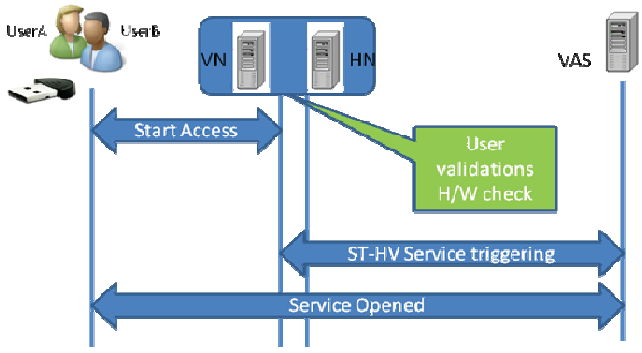


Figure 4. 10: Flow control for user accesses his service outside his home by same operator

This scenario gives the clients with the same IPTV operator the same home privileges for accessing their own subscribed or personalized channels outside their home networks. As the main issue of access is the operator guarantee that the user requests the nomadic access is the home user itself. This problem could be solved by using additional hardware H/W like USB Key or Dongle that can be used as a personal badge to give the client his rights as soon as it accepted by the visited network STB as shown in Figure 4.10. So, after starting the access by the nomadic user (userA), the Visited Network (VN) and Home Network (HN) are considered as one thing. After the validation of this user H/W the service triggering (ST) could be started for initiating the service from the (VN) easily through Visited Application Server (VAS).

- *User tries to access his subscribed channels and videos from another place but irrelevant to his operator. Using different URI host part. userA@operatorA.com in operatorB network.*

This scenario looks like roaming in mobile operators. So, we propose collaborative access between different providers to enable their customers having the right access of their personal services in any operator network. It is suitable to apply either Federation Identity (FI) or Multi-Identity (MI) according to the policy agreement between operators. The software based solutions is more practical in this case or we can apply the hardware based solution but on condition of compatibility solved. So, after passing the access verification by the visited network (VN), the service could be proposed by the home network or the visited network according to the inter-domain policy between the two operators as shown in Figure 4.11.

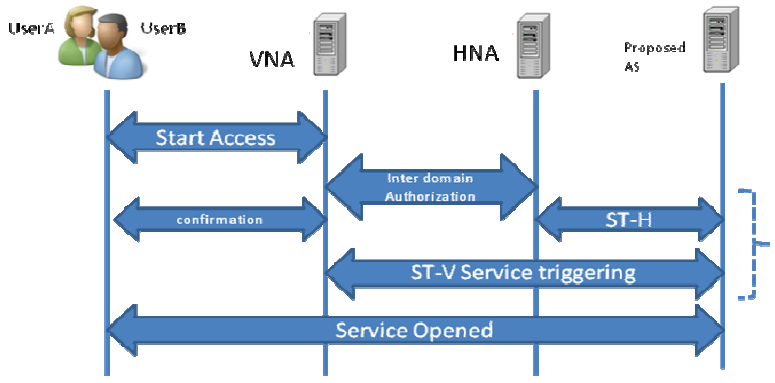


Figure 4. 11: Flow control for user accesses his service outside his home by different operator

4.8 Conclusion

Today, collaboration is considered as a vital solution in IPTV business models. It has high impacts on IPTV market share, seamless mobility between different operators, common way of security, cost reduction in investments and good network performance. This chapter contributions were to: present state-of-the-art for new IPTV terminologies like (OTT, Pay-TV, TVA and Open TV); illustrate the benefits in convergence networks design and their impacts on CAPEX and OPEX costs. Also, demonstrate the performance of future IPTV service against traditional TV and introduce the Multi-screen idea and the bandwidth optimization for multimedia delivery to different consumer's screens and then analyze the impacts of using cloud computing infrastructure in the converged network to different providers. Moreover, a simple Testbed for open TV aspect was conducted and analyzed in terms of control messages exchanged between server and clients. Apart from normal access, we also introduced the aspect of Nomadism and its access problems, ways and some use-cases for IPTV delivery. Finally, we expect the demise of isolated content providers' islands over the Internet at least in the domestic regions.

In the next chapter, we will go to in depth of operator network optimization through multicast trees in IPTV service. This chapter will conduct dynamic optimization for minimum spanning tree topologies so as to enhance the performance and decrease the costs of access.

Chapter 5: Optimizing Dynamic Multicast Spanning Trees for IPTV Delivery

This chapter aims to optimize the tree structure of IPTV Multicast nodes. In Multicast, clients join/leave the Multicast IPTV source of channels according to their requests. During Nomadic Access (NA), the multicast tree has to be restructured according to the new demands. We propose a new method that divides the operator main tree into different short trees based on new sources of video in different locations (source replications) by adding Virtual Nodes (VN) and changing the tree roots. Also, we optimize the investment cost of the new streaming points and compare the total minimum spanning tree costs before and after our optimization.

5.1 Introduction

The IPTV service consists in video streaming over IP networks. Last years, this service achieved high gain and success. First, operators proposed the service in their networks called *'Managed Networks'* so as to guarantee the service delivery and the Bandwidth requirements for service quality and customers satisfactions. Then, clients pushed the operators to search for new solutions for delivering same service outside their home networks *'Unmanaged Networks'* or *'Managed'* under roaming policies. This type of access is called *'Nomadic Access'* or TV Anywhere/Everywhere [59].

The structure of IPTV networks as explained in [61, 62] consists of access networks and core networks. Multicast protocols are the key element in communication between elements inside the two networks. Most operators adopt Protocol Independent Multicast - Sparse Mode PIM-SM [44] protocol for multicast routing in their IPTV topologies. While PIM-SSM (Source Specific Multicast) [63] also found high appreciation from the operators inside multicast networks to enhance and optimize the multicast processes.

Since Multicast is the base for commercial live TV in all managed IPTV delivery, operators search for dynamic solutions to cope with the changing number of clients joining a multicast node. The issue of broadcast *'Nomadism'* confronted with the guarantee of video quality and dynamic behaviour of multicast topology outside the user home network. So, we need to differentiate between the home network and the visited one in IPTV as follows:

Home Network: (HN) refers to the user local home network where he resides with his STB. This STB is installed by an operator which means 'restricted place inside the home' by a physical ADSL connection.

Visited Network/Nomadic Network: (VN or NN) refer to any access network other than the user home network. Operators can be used it to give the service for nomadic users. So, it could be same operator network outside the user local home or any other operators' networks.

The rest of this chapter is organized as follows: In Section 5.2, we introduce the work motivations, challenges in spanning tree, problem statement and main objective of our work. The virtual aspect of node insertion is introduced in Section 5.3 with its mapping procedures. Our Algorithm of virtual node insertion with MST optimization is explained in Section 5.4. The proposed algorithm simulation results and analysis of complexity issue are introduced in Section 5.5. Finally, this chapter is concluded in Section 5.6 with some highlights are found in future directions.

5.2 Work Motivations and Challenges

The operators need to broadcast required TV channels in a customized manner by adapting their trees to the clients' behaviours and also to minimize the access cost. To do this, there are two ways:

1. The clients should build Virtual Private Networks (VPN) with their home networks to obtain their videos. This solution is not scalable in case of increasing number of clients and multi-domain hops. Also, it is not efficient in video applications for delay and performance issues.
2. The operators should reconstruct the MCAST trees to match the clients' needs. This solution has more benefits in terms of network scalability and resources

optimization. It matches our objectives where we study some solutions and algorithms to reconstruct the multicast spanning tree topology.

Economically, the operators could evaluate any proposed solution by three criteria:

1. ***CAPEX*** cost: is the essential and fixed investments in main infrastructure
2. ***OPEX*** cost: is the running cost for long term operation
3. ***Revenue***: is the operator benefits from new business adaptation

So, we can divide the IPTV operators' service optimization into two parts:

- *The optimization of network resources by maximizing the utilization of network resources especially in multicast services. This may require some IPTV trees reconstructions.*
- *The optimization of network costs by minimizing the running and operational costs. By that, there is a way to gain high revenue from the investments.*

5.2.1 Managed Service

Although the majority of clients hope accessing their services anywhere and anytime, the operators are still the dominant for their services under managed mode inside their managed networks. Moreover, their networks suffer from some delivery problems especially in the scalability of multicast service. There are still some challenges in spanning tree for constructing multicast topologies. This section presents some challenges in IPTV networks and spanning tree aspect globally.

5.2.1.1 Challenges in Managed IPTV Service

The study introduced in [65] about the main challenges in delivering multicast IPTV focused on the network complexity. It handled the effects of increasing the number of channels on the multicast nodes capacities and its negative effects on the bandwidth performance.

This section highlights the background of IPTV service from the view of multicast mode and spanning tree protocol as the following:

- **Spanning Tree MCAST**

We have two famous Multicast service models used in IPTV delivery:

1. **SSM**: Source-Specific Multicast. In this model, the terminal (like STB) subscribes to a multicast group to receive data sent to the group by a specific source (only one source for the tree) which means specific source to specific

group (S,G) [63]. Therefore, with SSM the network is able to build a separate distribution tree for each multicast source.

2. **GSM:** Group-Shared Multicast. In this model, each node in the multicast group could send/receive videos to/from any member in the tree (many sources of traffic in the trees). This type traditionally called Any Source Multicast (ASM) as explained in [66].

- **Background about STP**

Spanning Tree Protocol (STP) is used in layer-2 switches to avoid loops in physical topologies [69]. As a result, it could be used in multicast networks to select the minimum cost paths to the root node and also preventing the loop issues in topology structure. Also, it could change and determine those paths in a dynamic way. When STP is initially enabled in a network, one of its first actions is to use the STA (Spanning Tree Algorithm) to select a root node and a direction to the root. The root point could be selected or forced as the administrator or operator needs (i.e. it could be node-x preselected or automatically chosen). After the root node is selected, the root links to all other nodes are determined one by one according to the shortest path algorithm used. More details about MST algorithms used here will be explained in Section 5.5.3.

5.2.1.2 Minimum Spanning Tree

Assume that any network is represented by one type of graph G (directed or undirected). This graph is represented by G = (V, E) where, V indicates the nodes and E represents the edges. The weights function *w* assigns a weight *w(e)* to each edge *e*. For our optimization, we will assume that the weights are real numbers between {0, 1}. MST objective is to find the minimum spanning tree of G, that is, the spanning tree T minimizing the following function:

$$W(T) = \sum w(e) \qquad where\ e \epsilon T$$

This MST aspect could be used in physical or virtual environments as the following:

In physical environment like TV systems, spanning tree was used by cable TV companies for laying cable to a new place. A graph represents points connected on different new paths. Some of those paths might be more expensive, because they are longer, or require the cable to be buried deeper; these paths would be represented by edges with larger weights. A spanning tree for that graph would be a subset of those paths that has no cycles but still connects to every house (final node in the path) . There

might be several spanning trees possible. A minimum spanning tree would be one with the lowest total cost [64].

For virtual environment, multicast overlay in MANET networks [60] is a means of virtualization by building an overlay virtual multicast topology over the physical one which is the Internet. This work proposed the aspect of virtual topology through MANET networks to reduce the cost and enhance the delay of transmission in a dynamic manner. The adaptation and dynamicity is done here in terms of mobility issue as a nature of MANET.

5.2.2 Operators Challenges in Spanning Tree

The increase in IPTV and Multimedia data in general imposes the operators either to find other solutions to replace spanning tree in complicated network or to conduct Multicast Replications (MR) in different layers of topology. This vision also comes from the new applied virtual cloud computing. Some of these proposals are discussed here after:

One of the recent proposed technologies to replace spanning tree in the data center is Multi System Link Aggregation (MLAG). MLAG works by extending the link level redundancy and load sharing mechanism of Link Aggregation (LAG), to support device and network level redundancy, active-active load sharing for full utilization of network bandwidth, and fast convergence [67].

To avoid drawbacks and pitfalls associated with STP, there are Avaya's Switch Clustering (SC) [68] and the IEEE's Shortest Path Bridging (SPB). SPB is based on a standardized data plane, offers comprehensive Operation Administration and Maintenance OA&M, and supports efficient Multicast distribution. It also enables secure traffic separation through the creation of Virtual Service Networks (VSNs).

On the other hand Multicast Service Replication (MSR) is an essential component in the efficient delivery of IPTV Multicast service. In general, all topology elements including aggregation switches, DSLAMs and IP edge routers perform Multicast Replication. This replication is either to be in the last parts of the network like edge routers or DSLAMs based on the service rate subscription or number of clients requests for a specific node.

5.2.3 Problem Statement

The Multicasting technique (MCAST) is adopted for the Managed Networks to deliver IPTV service. The advantage of MCAST over UNICAST is the saved Bandwidth (BW) and the processing along the network paths to reach the source node. The conventional MCAST schemes for Multicast applications over networks face several difficulties in dynamic networks. The dynamicity here means changing the number of clients joining the MCAST tree for obtaining the video service according to the clients' situations. Moreover, increasing the nodes capacity will lead to an increasing in the operator investments in multicast routes and processing. The MCAST tree topology can be adopted by creating virtual nodes capable of separating the original tree to different trees with minimum costs to reach final nodes (i.e. leaves of tree). Hence, the main goal is to propose different nodes to act as new sources of streaming based on the nodes capabilities. This means that, the assumption of virtual node is to insert a node inside the tree between some nodes but with some conditions on any of them like:

- The infrastructure of this node supports to be a streamer (Node Resources): $\boldsymbol{N_R}$
- The additional cost of investment in this node (CAPEX cost): $\boldsymbol{C_{CA}}$
- The running cost of this node after being root of a MCAST tree (OPEX cost): $\boldsymbol{C_{OP}}$
- The number of clients that are supposed to benefit from the new change of topology (as a factor of profit): $\boldsymbol{N_{CL}}$
- The Revenue from this adaptation: $\boldsymbol{R_{VN}}$

We can summarize those parameters as shown in Figure 5.1 below. In this figure, there are some nodes (n) that are supposed to be new roots for video service streaming. Those nodes are marked by (N_R) (i.e. Node Real). The connections between those nodes and the Virtual node (N_V) are called (L_{vn}) links to virtual node.

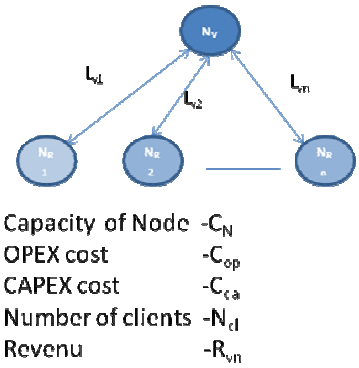


Figure 5.1: VN propositions

5.2.4 Revenue Cost Ratio (RCR)

From the economic view of IPTV business models, there are two economic issues:

1. ***Revenue :*** it is calculated /estimated by the size/capacity of clients/node joining process in IPTV multicast network
2. ***Cost :*** it has two types (capital CAPEX and operational OPEX) and all of them are affiliated to the technical requirements:

   Total Cost for any Node (i) is (N^i): $C^i_T = C^i_{CA} + C^i_{OP}$

   Each Node Revenue:

   $$R_{RR} = [N_{CL} . \textit{Client Revenu}]$$

   Virtual Resources for each node i in the running time:

   $$N^i_{RV} = [1 - (N^i_{CL} / N^i_{CL(MAX)})]. N^i_{RR}$$

   Total VN resources for the elected nodes (n):

   $$N^T_{RV} = \sum^i ([1 - (N^i_{CL} / N^i_{CL(MAX)})]. N^i_{RR}) \text{ where } i = \{1:n\}$$

   Total Additional operational cost:

   $$C^T op_{NV} = \sum^i [1 - (N_{CL} / N_{CL(MAX)})]. Cop_{RR}$$

   $$\textit{Where } i = \{1:n\}$$

   Total Revenue of Virtual Resources:

   $$R^T_{RV} = N^T_{RV}. R_{RV}$$

   Revenue Cost Ratio: it can be defined as the following:

   RCR= Total revenue from virtual resources/Total costs

   $$RCR = [R^T_{RV} / (C^T_{CA} + C^T op_{NV})]$$

   This RCR could be considered as a good indication in our optimization algorithm.

   Where:

N_{RV}: Node Virtual Capacity Resources

N_{RR}: Node Real Capacity Resources

R_{RV}: Revenue of Root Virtual

R_{RR}: Revenue of Root Real

Cop_{NV}: Operational Cost for Virtual Node

Cop_{RR}: operational cost for any root real

C^{T}_{CA}: total CAPEX (additional for new roots)

5.2.5 Objective Function

Assume that, the Virtual Node (VN) cost is a summation of weights between VN (v) and (n) real ones so, the total cost value that needs to be minimized is:

$$W^{T}_{vj} = \sum^{j} F(C^{j}_{CA\ \vee vj} + C^{j}_{OP\ \vee vj}) \text{ where } 1<j>=n \quad (1)$$

Where:

$C^{j}_{CA\ \vee vj}$ = CAPEX cost (where j any node to be selected as a new MCAST source)

$C^{j}_{CA\ \vee vj}$ = 0 if node j is able to be a root (new source) without any additional cost

$0< W^{j}_{vj} <=1$ according to the cost function in (1)

$C^{j}_{OP\ \vee vj}$ = OPEX cost as additional cost to run (j) as a new root.

The weight matrix [W^{T}_{vj}] is a key factor in the optimization of number of roots generated by the minimum spanning tree as shown in results Section 5.6. We propose different values of those weights on the principle vector of costs as {0 to 1}.

5.3 Steps of Proposed Algorithm

We describe here the technique for optimizing the cost associated with the virtual node insertion procedure.

The Virtual Node insertion algorithm is mainly depending on the mapping that will be explained in Section 5.4.2 and the assumptions for the elected nodes to act as new roots. So, the weights of virtual links and the virtual resources calculations are the key factor in this algorithm. In Figure 5.2, there are three factors which in the election phase {*node position, link quality and CAPEX cost*}. Those parameters besides the weights of virtual links help in our optimization procedure as we will explain in the next phases:

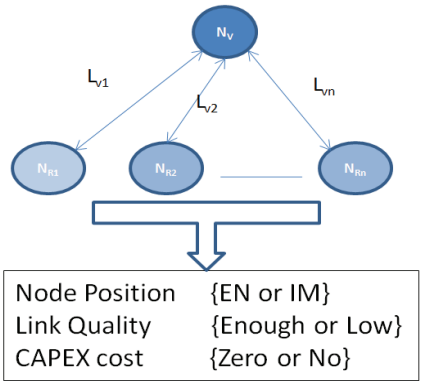


Figure 5.2: Elected nodes propositions

5.3.1 Initialization Phase (I)

We first select some nodes to participate in the election phase of new roots. As shown in Table 5.1, there are some criteria to judge on the preselected nodes like:

Node Position: This position could be end node in the tree or intermediate one.

Link Quality: The QoS of the link.

CAPEX cost: This cost is an indication for additional cost investment for this node to be a root.

Table 5.1: Initialization Phase for elections

Define Node Attributes (NA)
 # Node Position:
EN="End Node"
IN="Intermediate Node"
 # Link Quality Average (only one link):
Low="Low"
Enough="Enough"
 # CAPEX Cost:
Zero="Zero"
No="Not"
*If NA = ((**IN** and **Enough** and **Zero**) OR (**EN** and **Enough** and **Zero**))*
Then "This node is preselected"
Else
"This node is out of scope"
Return "Elected Nodes List"

5.3.2 Node Candidate List Phase (II)

This phase is used to finalize the nodes selection and filtrations according to the threshold parameters defined by the operators for accepting new nodes to act as roots for

video streaming. Table 5.2 explains the steps and procedure for obtaining final nodes list of the selected nodes.

Table 5.2: Selected nodes filtration

#operator threshold: Y (for cost decision)
Define (Y)
#Standard fixed cost for node construction: CAPEX:
Define (CAPEX)
 # Legacy cost for node i = Lc(i)
CAPEX- Lc(i) = X(i)
If Xi>Y
Then "neglect this node"
Else
Add "this node to the list[i]"
End
Return list[n]

5.3.3 Operational Phase (III)

This phase is representing the final phase which applies the MST on the virtual topology. The operational phase steps are planned in Table 5.3.

Table 5.3: Final phase

For i=1:n
(VN,i) = $W_{LV}(i)$
Run Minimum Spanning Tree (MST) on virtual topology
#Output: MST of Virtual Tree
Remove VN from the output tree

In this algorithm, we propose running the MST taking into consideration the VN as a new root for the virtual topology (T^{v}) and check the second level after the virtual root which actually represents the real roots for the divided trees. We then remove the virtual root and the result represents the new trees generated by our method. Moreover, we recalculate the MST costs for the new trees and compare them with the original MST costs to judge if the algorithm successeded in minimuzing the total costs and enhanced the overall network performance as shown in Section 5.5.

5.4 Virtual Nodes and Multiple Trees

This section discusses the virtual node insertion procedure and its mapping with the new tree roots.

5.4.1 Virtual Node Insertion

We use the following notations as explained in Table 5.4.

Table 5.4: Notations of VN insertion and roots optimization

Original Tree	T^o	Original tree
	N^o	Nodes of original tree
	L^o	Links of the original tree
	Nw^o	Weight of nodes in the original tree
	Lw^o	Weight of links in the original tree
	C^o	Current cost
Virtual Tree	T^v	Virtual tree after inserting VN
	N^v	Nodes of virtual tree
	L^v	Links of the virtual tree
	Nw^v	Weight of nodes in the virtual tree
	Lw^v	Weight of links in the virtual tree
	C^v	Additional cost of VN
	C^s	Streaming cost (part of C^v)
Constraints	N_R	Node Resources
	N_{CA}	Node CAPEX cost
	N_{OP}	Node OPEX cost
	N_{CL}	Node clients capacity (Actual)
	$N_{CL(MAX)}$	Node clients capacity (Maximum)

Assume that the distribution network is represented by a graph $G=(V,E)$ with V represents the number of nodes and E represents the links between them. Then, the spanning tree (T) of (G) is the shortest path topology based on the minimum weight or spanning tree cost. This cost could be defined as the sum of routing costs from all source nodes (roots) to other nodes (destinations). The output tree will be denoted by (T^o) and can be written as the following:

Original spanning tree:

$$\boldsymbol{T^o=(N^o, C^o, L^o, Nw^o, Lw^o)}$$

Then, the virtual spanning tree could be expressed by:

$$\boldsymbol{T^v=(N^v, C^v, L^v, Nw^v, Lw^v)}$$

So, the objective now is to map the Virtual Tree ($\boldsymbol{T^v}$) onto the Original Tree (T^o) by inserting the virtual root based on the constraints explained in Table 5.4.

Generally, for VN insertion, we have global mapping (M) between the virtual tree ($\boldsymbol{T^v}$) and the shortest spanning original tree ($\boldsymbol{T^o}$) as the following:

$$M: \boldsymbol{T^v} \rightarrow (\boldsymbol{N'}, \boldsymbol{C'}, \boldsymbol{L'}, N_R, N_{CA} - N_{OP}, N_{CL})$$

Where:

$$\boldsymbol{N' \subset N^o}$$

$$L' \subset L^o$$
$$C' \subset C^o$$

For MST, we can consider each output tree ($\mathbf{T^o}$ or $\mathbf{T^v}$) as an Undirected Graph subjects to minimizing the total weights on all links ($\mathbf{Lw^o}$ or $\mathbf{Lw^v}$) as follows:

For original tree (T^o):

Inputs: Undirected Graph G=(V,E) with [V= N^o]and [E =$\mathbf{Lw^o}$]

Output: MST ($\mathbf{T^o}$) that minimizing the total weights:

$$\mathbf{Weights(T^o)=min} \sum(\mathbf{Lw^o}) \quad \forall L^o$$

For virtual tree (T^v):

Inputs: Undirected Graph G=(V,E) with [V= N^v]and [E =$\mathbf{Lw^v}$]

Output: MST ($\mathbf{T^v}$) that minimizing the total weights:

$$\mathbf{Weights(T^v)=min} \sum(\mathbf{Lw^v}) \quad \forall L^v$$

5.4.2 Mapping Types

According to the previous general mapping, we have three sub-maps to satisfy our optimization:

- ***Node Mapping:***

This map selects the new roots to be video streamers based on the Nodes Resources N_R (like servers, CPU and streaming fees) which all are parameters of nodes. So this mapping could be written as:

$$\mathbf{M^n}: (N^v, Nw^v) \rightarrow (N', N_R)$$

- ***Cost Mapping:***

This map considers cost issues either CAPEX (N_{CA}) or OPEX (N_{OP}) to optimize the profits from adopting this technique:

$$\mathbf{M^c}: (C^v, C^s) \rightarrow (C', N_{CA} + N_{OP})$$

- ***Link Mapping:***

This map considers links and weights or metrics values on the links affected by the number of clients benefiting from the node service (N_{CL}).

So, the Link mapping is:

$$M^l: (L^v, Lw^v) \rightarrow (L', N_{CL})$$

Finally, the revenue of Mapping (R_M) could be expressed in the following manner:

$$R_M(T^v) = \sum^i R_M(T^i) \quad where\ 1<i<n$$

Where (n) is the number of generated trees (new sources).

5.5 Results and Analysis

In this section, we give examples for networks structure for delivering multicast service (live TV) in nomadic situation.

Actually, we have three types of graph representations for our problem as the following:

- **DG:** Directed Graph (for n nodes, the total number of links equal ***[n*(n-1)]*** which is almost equal to ***[n^2]***.
- **UG:** Undirected Graph (for n nodes, the total number of links equal ***[n*(n-1)]/2*** which means half number of edges than directed graph DG.
- **SG:** Sparse Graph (for n nodes the maximum number of links ***[2*n]***) and if it reaches ***[n^2]*** the graph will be dense graph.

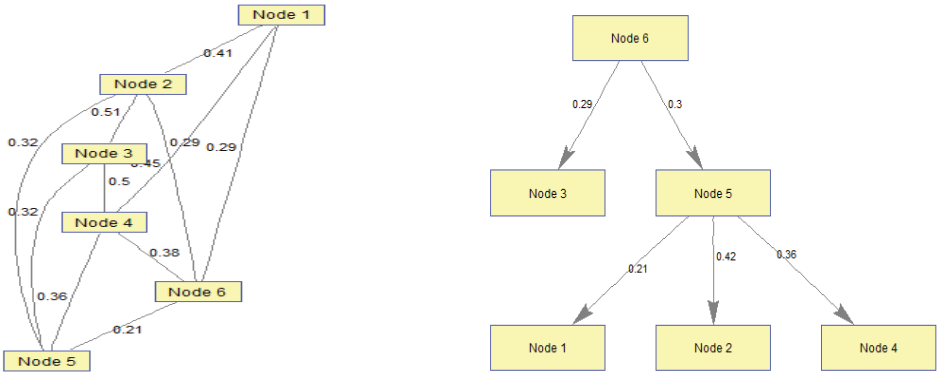
In our implementations, we mainly work on DG, then we transform it to UG and finally, we apply minimum spanning tree algorithm on SG which is more practical and complies with actual network operators design.

Moreover, we propose the virtual links cost (i.e. weights) between VN and other nodes. Then, we study the evolution of the new tree with respect to the weights as:

- The VN has equal costs to the proposed nodes [from 0 to 1 step size 0.1]
- The VN has different costs to the proposed nodes

We run our simulations using Matlab and get the following results:

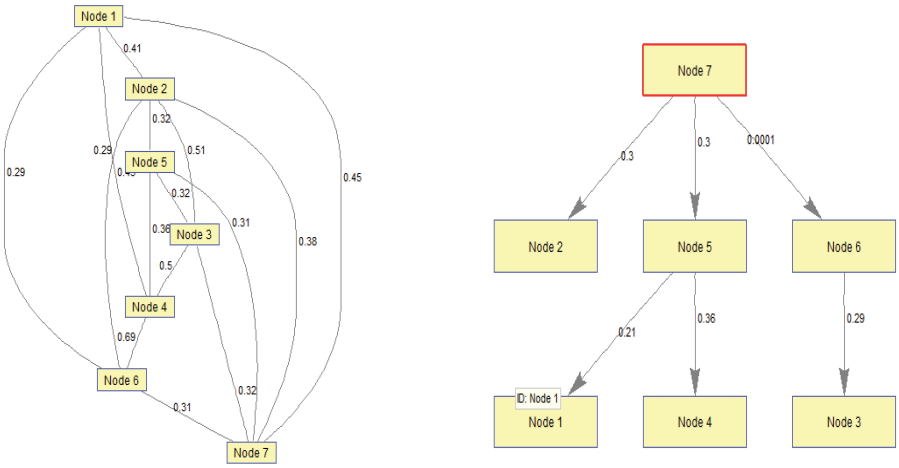
As shown in Figure 5.3-A, the network is represented by a simple topology of 6-nodes with some interconnection links. After running MST to define the shortest paths between nodes on the assumption of node 6 as a network root, we have the output minimum spanning tree structure as shown in Figure 5.3-B. The arcs weights represent the metric values or link costs between nodes.



A): The original graph B): The MST output graph

Figure 5.3: Main topology with six nodes

In Figure 5.4-A, we insert node-7 as a virtual node to act as a root position between nodes-1, 2, 3, 5 and 6 with same assumption of nodes weights. After that, we run same MST over the new virtual topology and have the output shown in Figure 5.4-B. The VN creates new sub trees; one with the old root node-6 and another with new root node-2 and last one with node-5. In Figure 5.5, we have different sub-trees (only two) as we change the virtual link costs.



A): Main virtual graph B): MST for virtual graph

Figure 5.4: Virtual topology by inserting VN (node-7)

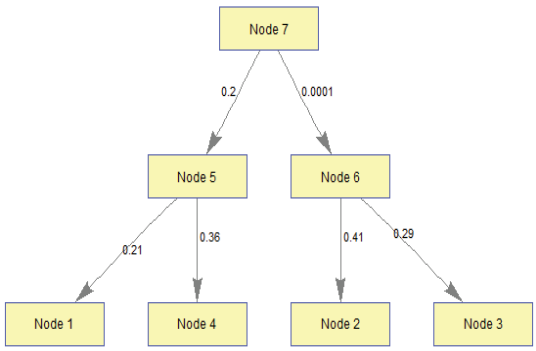


Figure 5.5: Two trees built over the previous one with 2 new roots (node 6 and 5) after inserting node-7 as a VN root and changing the weights of links between VN and proposed ones.

5.5.1 Virtual Links Cost-Analysis

In this section, we will vary the costs on virtual links between the virtual node and the new selected roots to prove that all the simulations give MST costs for virtual tree less than the original one. Also, this means that the summation of MST costs for the generated sub-trees will be less than or at maximum equal the total MST for the original tree as the following:

$$\sum MSTCost(Sub-Trees) \leq MSTCost(OriginalTree)$$

Actually, the proposed costs on virtual links are depending on the network state. This state has a dynamic aspect which means the costs variations could not be deteriminstic process but, it could be considered as a stochastic one. Moreover, the predections of network states and nodes capacities could be changed at any time according to users join/leav of network or the costs estimations. Hence, we will suppose different scenarios of topologies and assigne different costs on virtual links between 0 and 0.9 on equal and non-equal bases. The equal base assumes that the supposed costs are equal as the virual root resources are divided equally between the new roots. But, for non-equal base, we assume different costs based on the total virtual resources are not equal and each new root has different base of resources which means different costs on virtual links. In the following three figures, we simulate the two scenarios with different number of nodes and all give MST costs less than or equal the original one as shown in Figure 5.6 for 6 nodes, Figure 5.7 for 15 nodes and Figure 5.8 for 25 nodes.

Form the three figures below, we can conclude that; the MST cost for orignal graph is represented by point zero which means no VN. After adding the VN with different links cost, we have MST cost equal or less than the original ones either for equal or different costs. This confirms the previous equation that, the generated sub-trees have MST cost less than or equal the original one.

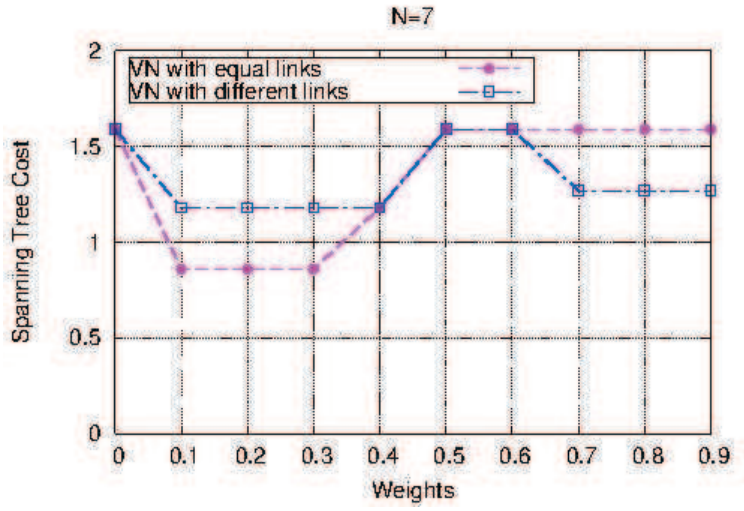


Figure 5.6: MST cost calculation for 7 nodes [N=7] (6 real and 1 virtual)

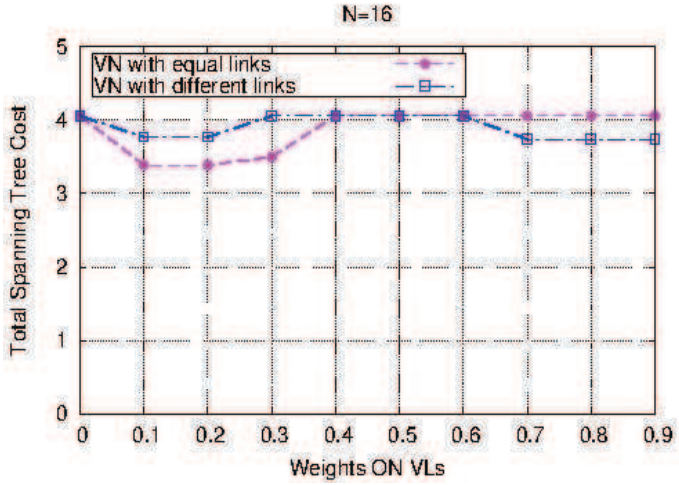


Figure 5.7: MST cost calculation for 16 nodes [N=16] (15 real and 1 virtual)

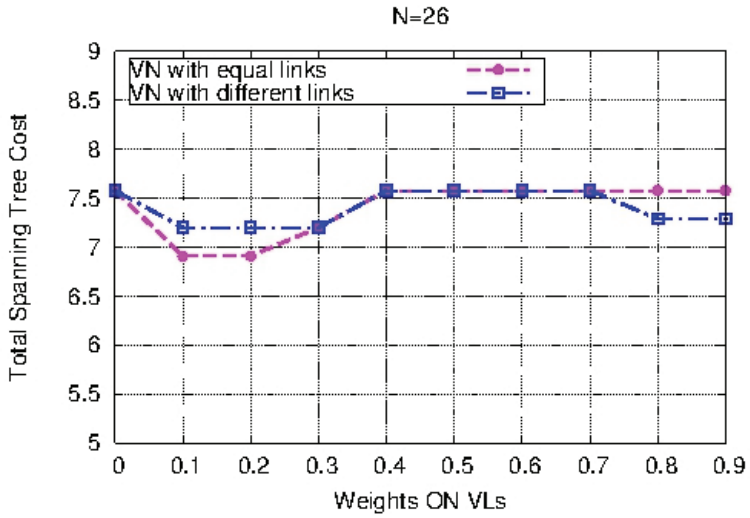


Figure 5.8: MST cost calculation for 26 nodes [N=26] (25 real and 1 virtual)

5.5.2 Source Replications

In this part, we simulate different scenarios of Virtual Node (VN) positions by changing the weights links (Wvj) where (v) is the virtual node and (j) is the preselected node to be a new source. The simulation results of spanning tree are summarized in Table 5.5 using MST algorithm in Matlab. The objective from this variation is to test the algorithm with different networks architectures and measure the number of replications as new sources for multicast service. The output number of replications is complying with the practical scenarios of IPTV source replications in many operators.

Table 5.5: Three trees with capacity 6, 15 and 25 nodes and with different VN positions, the last two columns specify the number of output replications as new sources or roots

Number of Real Nodes	**Virtual Node Links**	**weights**	**New Sources**	**Replications OUTPUT**
6	5(1,2,3,5,6)	Wv1, Wv2,Wv3,Wv5,Wv6	2,3,5	3
15	2(1,7)	Wv1,Wv7	1,7	2
	3(1,7,8)	Wv1,Wv7,Wv8	7,8	2
	4(1,7,8,9)	Wv1,Wv7,Wv8,Wv9	7,8	2
	5(1,7,8,9,10)	Wv1,Wv7,Wv8,Wv9,Wv10	7,8,10	3
25	2(1,7)	Wv1,Wv7	7	1
	3(1,7,10)	Wv1,Wv7,Wv10	7,10	2
	4(1,7,10,13)	Wv1,Wv7,Wv10,Wv13	7,10	2
	5(1,7,8,10,13)	Wv1,Wv7,Wv8,Wv10,Wv13	7,10,13	3

5.5.3 Complexity

The complexity of the algorithm used to find the minimal spanning tree (MST) is function of number of nodes , number of links and the type of algorithm:

- **'Kruskal algorithm [70]'**: Grows the minimal spanning tree (MST) one edge at a time by finding an edge that connects two trees in a spreading forest of growing MSTs. Time complexity is ***O(E+X*log(N))***, where X is the number of edges no longer than the longest edge in the MST, N and E are the number of nodes and edges respectively.

- **'Prim algorithm [71]'**: Grows the minimal spanning tree (MST) one edge at a time by adding a minimal edge that connects a node in the growing MST with any other node. Time complexity is ***O(E*log(N))***, where N and E are the number of nodes and edges respectively.

We applied the second algorithm for its simplicity in the computation times and also its practicality in finding nodes ranking.

In terms of complexity time consumed by MST algorithm to build the final trees, we improved the algorithm convergence time as follows:

5.5.3.1 Heuristic Way

To improve the convergence time, we can guide the starting point of the algorithm by forcing it to chose node (R) as a main root for (T^o). Then, the VN will be the new root (R) connected to pre-selected nodes only. Finally, we run MST on this graph to obtain (T^v) which is the output virtual tree.

5.5.3.2 Non-Heuristic Way

The Non-Heuristic way inserts the VN between all nodes in the graph. This means that, the VN has full mesh connection with all nodes. So, it will have large numbers of links and more processing time as shown in the Figure 5.9.

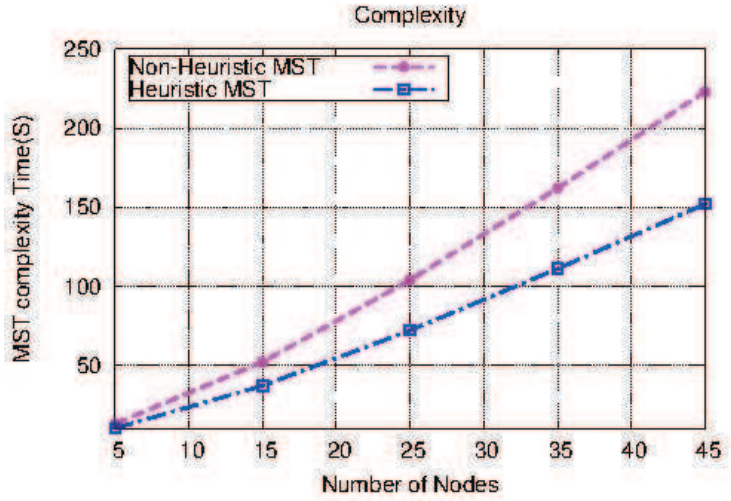


Figure 5.9: Heuristic versus Non-Heuristic test for MST algorithm (prime)

Clearly, the proposed heuristic way gives lower processing and computation times compared to the Non-Heuristic way (Figure 5.9).

5.5.4 GUI Optimization Tool

We implemented a graphical tool to be used for network design and optimization as Graphical User Interface (GUI). This graphical tool is based on Python libraries [58].

Figure 5.10 illustrates the main construction steps:

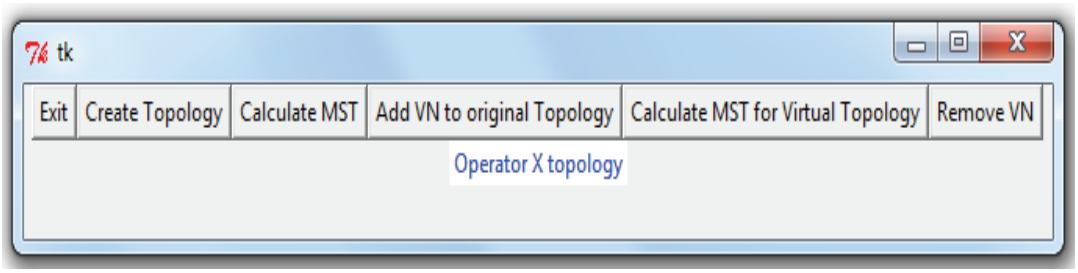


Figure 5. 10: The GUI interface of the optimization tool

In this GUI, we can create different topologies or graphs. Then, the MST can be calculated for the original network structure. After that, we can add VN between some proposed nodes and run the MST again to have the new virtual tree MST. Finally, we can remove the VN to obtain the sub trees generated by our optimization. For more details and examples shown in ***Apendix B***.

5.6 Conclusion

We have presented an optimization solution for multicast trees used for IPTV. It is adapted to nomadism cases where diffusion trees have to be changed.

The main idea is to add a virtual node where we think a replication server should be put and to recalculate the new minimum spanning tree. The root of virtual tree is removed afterwards. We have also defined new cost metrics for CAPEX and OPEX to be used in our algorithm. The results show an optimal diffusion IPTV tree in all situations.

In the next contribution, we enrich our video delivery in general with reliability and security measures. As the security mechanisms mainly depend on hashing techniques, we will introduce some hash redundancy techniques achieving reliability in case of video packets loss.

Chapter 6: Hash Chains Background for Video Streaming

Hash chain procedures use symmetric block encryption mechanisms that create an output message (Digest) which is dependent not only on the current plaintext input message and key, but also on earlier information. By block chaining, two instances of the same plaintext input block produce different cipher text blocks, making cryptanalysis difficult. Therefore, it could be considered as an efficient and light weight mechanism to enforce control. However, the main problem of hash chain with video applications is its reliability and this is our objective in this contribution as the video is a loss tolerant application.

6.1 Introduction

Most of video streaming research about security concentrates on the user authentication and verification as a basis for accounting and billing systems. But, one of the problems that draw our attention is the verification of received data and the authentication of the source of video streaming which is called data integrity & data origin authentication. We may have some attacks while receiving the real time data. We need to confirm each time that; there is no “man-in-the-middle” capable of changing the streaming of the data and resend it again to the receiver. It is also useful against video copying and retransmission.

A direct solution for this problem is to encrypt data between two pairs. But encryption and decryption algorithms need processing time and memory while the modern devices like mobile and real time applications need to optimize those parameters. So, solutions usually stop at authentication mechanisms for practically solving those problems.

6.1.1 Data Authentication

Data authentication means to authenticate the data and source of data so as to assure that the data is sent from a real sender and also not being changed.

Mechanisms developed to achieve a high degree of Data Integrity with IP packets like IPSec [72] anti-replay have certain ability to verify the origin of data transmission (Data Origin Authentication). Also, there is another security protocol for real time applications called SRTP [73] (Secure Real-Time Transport Protocol) for securing media streams.

The main idea of using the hash chains in data authentication is due to the security of the used hash functions. In general, when you have a hashed value, you are unable to calculate the original value because of its nature as it is one way algorithm. So, you just able to calculate some values (Digest) and compare it with the received hash source value to assure that your data is not altered or intercepted in the transmission path between source and destination if and only if your hash results is the same hash value you received from the source.

6.1.2 Hash Chain Methodology

The hash chain is a method used to produce many one-time keys from a single key or password. So, it is a successive application of a cryptographic hash function $h(x)$ to a string. For example, given the initial value (x) and hash function h, then we can calculate: $Xn=h(h^{n-1}(x))$.

So, if $n=5$ then $h(h(h(h(x))))$ gives a hash chain of length 5, often denoted by $h^5(x)$. Then, the use of hash chains is very simple. Once a random initial value is generated then, we are able to calculate a chain of n values by repeatedly using the hash function, i.e. given the initial value x and hash function h, then you can calculate: $\boldsymbol{Xn=h(h^{n-1}(x))}$**.** The last value in the chain, i.e. the one that will be given to the other party first, is called the anchor value [74].

The procedure used is that the sender (may be any application server) binds itself to the data (video streaming) by signing it with its own identity. Next, the client on reception creates a hash chain anchor, which is bound to the identity of the user itself. This is also bound to the given flow or stream, so it basically could be a hash of the pre-agreed statement. Note that at this point, the client and the server have already increased

trust to their identities, because the home operator has acknowledged the user as its own subscriber and access network has been approved by the home network.

The rest of this chapter is organized as follows: Section 6.2 gives an overview on the concept of using hash chain with real time applications. In Sections 6.3, we highlight the related work used hash chain methodology with a brief study on the pervious works introducing hash chain links. Also, we introduce our work motivations based on the previous works comparison. Section 6.4 handles the process of real time application delivery with the specifications of senders' preparations or receivers' verifications. In Section 6.5, we conclude this chapter and list the next steps of this contribution through our objectives and motivations.

6.2 Hash Chain and Video Streaming

Hash chain concept was first proposed in [74] for securing password authentication. Later, it was used in applications such as authentication of multicast traffic [75, 76], the routing in sensor networks and sensors applications [77], the privacy of RFID authentication [78], data streaming [79], micropayment systems [80] and one time password [74]. The main advantage of hash chains is the lower calculations compared to other cryptographic algorithms like encryption algorithms.

Streaming has two ways of operation online or offline as the following:

Online or live videos: The live events, such as concerts, speeches, and lectures, are commonly streamed live over the Internet via broadcasting software. The broadcasting software encodes a live source, such as video from a camera, in real time and delivers the resulting stream to the server. The server then sends the live stream to the clients. Regardless of when different clients connect to the stream, each one could see the same point in the stream at the same time of his joining.

Offline or On-demand video: For an On-demand video delivery, the files are archived and pre-stored on the server. Each client initiates the stream from the beginning, so no one ever comes in "late" to the stream. In this case, no need for broadcasting software from the server side. YouTube is a good example for such case which contain a huge database of short videos and represent a successful model for file allocation business that attract many users daily as studied in [26].

6.3 Related Work

Digital Signature (DS) are used in video streaming authentication and verification. Its role is to assure that the receiving data is not altered or changed during its route from source to destination. It has especially been applied in multicast application delivery but the problem is the large overhead produced by it according to the keys size used in data signing [81].

Hash Chains have been used in SIP [82] for Real Time Payment [83]. In that system, authors integrated some messages of SIP and Hashes. They extended some SIP messages and added the calculated chains to them so as to achieve secure payment for SIP based real time services.

Hash Chains were also used in the accounting and billing systems [84]. They tried to assure the accounting information between the source and the destination by the help of hash chains. Their algorithm detects any change by another third parity in the route while transferring the accounting information so as to secure the billing information.

In another security project "IS-Manet" [85], the contributors tried to secure the video streaming received from satellite link by combining the digital signature with watermarking to achieve authentication and data integrity for mobile applications. They achieved the security by digital signature combined with a pre-computed hash chain and embedded this hash chain inside the video streaming by the means of watermarking technique.

In [86] Timed Efficient Stream Loss-tolerant Authentication (TESLA), they allow all receivers to check the integrity and authenticate the source of each packet in multicast or broadcast data streams. TESLA requires no trust between receivers, uses low-cost operations per packet at both sender and receiver, can tolerate any level of loss without retransmissions, and requires no per-receiver state at the sender. TESLA can protect receivers against denial of service attacks in certain circumstances. Each receiver must be loosely time-synchronized with the source in order to verify messages, but otherwise receivers do not have to send any messages. TESLA alone cannot support non-repudiation of the data source to third parties.

In [87], the authors assumed Multicast Message Authentication Code (MMAC) for achieving message integrity. They developed an advanced signature scheme for

multicast data. Their system reduced the buffering mechanism on the sender and also reduced the packet loss.

6.3.1 State-of-the-art

In general techniques, to authenticate stream of packets each packet must carry its own signature and thereby each received packet is individually verified. However, the main disadvantage in this way is the high complexity and overhead, as cryptographic signature operations require high computation power and its size is in the order of hundreds of bytes. In another method, a single signature is computed from a bit string which is the concatenation of all packets. While it has very low complexity and low overhead, it does not tolerate any packet loss, which means any packet loss causes cession in security measure (i.e. the receiver could not be able to continue packets verification process).

So as to overcome those limitations in classical ways, there is a solution proposed in [88]. The authors observed that; one-time signatures can be used in combination with a single digital signature to authenticate a sequence of packets. Each packet carries a public-key, which is used in a one-time signature scheme to sign the following packet. Only the first packet needs to be signed with a regular digital signature. Since one-time signatures are an order of magnitude faster to apply than digital signatures, and can also be verified somewhat more efficiently, this solution offers a significant improvement in execution speed.

The major drawbacks of the previous technique are the one-time signatures overhead and the low reliability. If some packets were dropped from the stream, there is no compensation method and the chain will be broken immediately.

The work proposed in [89] had two efficient solutions to the problem of authenticating streams in a lossy environment. In the first scheme (TESLA,) packets are authenticated with Message Authentication Codes MACs. The MAC keys are disclosed after a certain time interval, to enable verification. The delay before disclosure is chosen long enough to ensure that the keys can no longer be used to corrupt packets. TESLA is highly efficient and versatile, but it requires the ability to synchronize the clocks of the sender and the receiver within some margin. The second scheme is called Efficient Multi-chained Stream Signature (EMSS) which used one-way hashes in combination with digital signatures to achieve authentication, following the idea introduced in [88].

To resist loss, the hash of each packet is stored in multiple locations. EMSS proposed to choose such locations at random, and provides probabilistic guarantees that a packet can be authenticated by given a certain amount of loss in the stream.

The main advantage of this scheme is the redundancy of the digest for each packet. This could lead to some degree of reliability in receiver verifications as there is a chance for compensating the hash value for same packet from different packets. But, as the locations of storing those values are randomized there is huge difficulty in the implementation of this algorithm. Also, for achieving high degree of reliability it is assumed to have six hashes per packet which is considered as an overhead.

The deterministic technique proposed in [90] introduced some solutions for increasing the reliability in case of packet loss and provides simple implementations than the previous work [89]. They considered a stream exchanged between a sender and a receiver over an insecure, unreliable channel such as UDP. Lost packets could not be retransmitted, and packets may arrive out of their order. Their work assumed that loss occurs in bursts. A burst starts at a location randomly chosen in the sequence and lasts for a randomly chosen number of packets. Their proposed approach to signing the stream followed the idea introduced in [88]. They used a combination of one-way hashes and digital signatures to authenticate packets. The idea is as follows: if a collision-resistant hash of packet P_1 is appended to packet P_2 before signing P_2, then the signature on P_2 guarantees the authenticity of both P_1 and P_2.

The pros of this algorithm are its simplicity of implementation and the achieved degree of reliability. But, the overhead is its main drawback. As the number of packets increases the overhead will increase linearly.

The evolution of EMSS algorithm was proposed in same work under new name; Augmented Chain technique [90]. They proposed a systematic method of inserting hashes in strategic locations so that the chain of packets formed by the hashes will be resistant to a burst loss.

The LMCM (Layered Multiple Chaining Model) proposed in [91] had some degree of reliability by distributing the hash values and signatures to different packets. They proposed a novel approach to authenticate multicast streams based on (LMCM). It

divides a stream of n packets into some groups, and each group includes m (m>1) blocks that contain d (d>1) packets equally. A packet P_i is defined as a message Mi, which a sender sends to receivers along with the additional authentication information. A sender appends the hash $H(P_i)$ of a packet Pi to specific other packets to achieve robustness against packet losses. For each group the sender then concatenate the hashes of specific packets together and signs them using its private key. For that, high authentication probability can be achieved by increasing the number of packets that contain the hash of a previous packet and the number of hashes that are appended to the signature packets. Their algorithm is secure and provides non-repudiation. It is efficient and practical to be applied to many multicast applications. But, practically, it suffers from large overhead in terms of the number of hash values appended to the packets.

In [92], authors introduce the Butterfly Graph (BG). They divided the packets into groups and each group has one signature calculated based on hashing. The redundancy is achieved by sending the signed packets several times. Their overall concern is to keep a good performance as the amount of redundancy is increased.

In BG, they amortized a digital signature among a group of packets which are connected as a butterfly graph. It has lower complexity, low overhead and very high verification probability even in the presence of packet loss, because it inherits the nice fault tolerance property from the butterfly graph.

Also, in [93, 94] they examine the problem of streaming of authenticated video over lossy public networks using Graph theory and taking into account the quality of wireless channels. It is a kind of optimization technique for authenticating the streaming packets called Rate-distortion-Authentication (R-D-A). Moreover, they achieved good optimization in media quality and packet overhead.

They also proposed a Generalized Butterfly Graph (GBG) for authentication, which supports arbitrary number of packets and arbitrary overhead and at the same time, retains the high verification probability of the Butterfly Graph authentication.

In Hybrid Hash Chain (H2A) [95], the authors introduced a hybrid algorithm for authenticating data-streaming based on hash chain. They tried to achieve some degree of

redundancy in hash links by sending the hash value of each packet in other packets as the following:

The hash of packet P_i is $H(P_i)$ is included in packet P_{i+1} which is the next packet to P_i. Also, for adding redundancy, the hash $H(P_i)$ will be included in some randomized packets which are selected within a specific scope (d) which is also randomized. The robustness of this algorithm is achieved by carrying the hash link of each packet in different locations which are randomized and achieved the hybrid property. The last issue is the signature packet. It is sent after specific period or number of packets. This means amortizing a single digital signature over many packets. Also, this protocol has achieved high degree of redundancy in verification with minimum added overhead as compared in their simulation results with EMSS [83] algorithm.

The main advantage of this hybrid technique is its high robustness against packet loss. But, it is still suffering from overhead per packets which is more than 3 hashes. This overhead could be different according to the type of hashing algorithm used.

In RLH [96] Receiver-driven Layered Hash-chain technique, the hash values are categorized into different levels. The layers of this technique define two issues; the repeating degree of hash values per level which means the distribution of packet hash to other packets in same level. The second issue is concerning the indexing way of cross layering hash including which means the periodicity of packet hash values distributed into another layer. For the signature packet, the source will concatenate different hashes from different levels to obtain this packet which is the main objective of this algorithm. Moreover, the source has to send this packet periodically so as to achieve the required rate of robustness.

The main advantage of this technique is its high degree of reliability in terms of different paths to hash links. This means that, the receivers could achieve the reliability by deciding at which layer could join signature verification. Also, each layer has its specific hash chains. But, the main drawback is the algorithm sophistication and randomized way of embedding the hash values or the cross layering indexing. This could be considered as the realization is little bit complex for such way.

A Time-Critical multicast authentication scheme was proposed in [97], which combines hash chains with one time signature to authenticate streaming of packets. The

algorithm provides short end-to-end computational latency, perfect tolerance to packet loss, and strong resistance against malicious attacks. They used long key for achieving high security which leads to large overhead but their algorithm has high robustness against packet loss. This algorithm is called Time Valid One-Time Signature for Time-Critical Multicast Data Authentication (TV-OTS).

Finally, we can conclude the main differences and similarities between the previous algorithms in terms of delay consumed, overhead and degree of robustness as shown in the following Table 6.1.

Table 6.1: Comparison between different authentication algorithms based on hashing

Algorithm	Processing Delay	Overhead of Authentication	Robustness to Loss
One-Time signature [1997]	Only at the senders (Packet-Based PB)	Only one packet overhead (first)	No robustness
TESLA [2000]	Buffering at senders and receivers (PB)	According to hash size	Medium
EMSS [2000]	Only at the receivers (PB)	According to the hash indexing level (6 hashes per packet)	Medium
Augmented Chain [2001]	For both senders and receivers (BB)	Less than EMSS	Medium
MAC [2001]	For both senders and receivers (BB)	According to block size	High
Hybrid Hash Chain H2C [2004]	Less delay at senders (PB)	Varied (more than 3 hashes per packet)	High
RLH [2005]	Only at the receivers	Varied	High
Butterfly-Graph (BG) [2007]	For sender and receiver (Block-Based BB)	According to group size	High
Generalized BG [2007]	For sender and receiver (BB)	According the random number of packets selected	High
LMCM [2008]	Only at the senders (BB)	Increased according to number of packets carried previous hash values	High
TV-OTS [2009]	No buffering as processing of packet-based method (PB)	Overhead per packet according to signature size	High tolerance

6.3.2 Work Motivations

The video streaming or IPTV service applications are loss tolerant applications. This means that, the flow of packets could miss some of them without affection on the streaming process. Although, the application is loss tolerant, the security measures that will be applied like hash chain or digital signatures are not tolerant for these losses. This could lead to break the security measure especially in hash chains methodology.

Our main objective is to simplify the verification method used by the senders and receivers for confirming that the data is not intercepted or changed while its transmission with high degree of hash chain reliability.

We propose using the Hash Chain method for this task under optimised hash functions that keep the user under low overhead to verify sender and receiving data in an integral way. We will also focus on the reinitiating mechanism (resynchronization) for sequence numbers in case some parts of the video-stream have been lost and how this will affect the receiver decision about discarding data.

The chaining mechanism will improve the security. It will add some degree of complexity due to the calculation or estimation necessary for an intruder to try to attack the streamed video.

6.4 Real Time Applications Delivery

Most real time applications in Future Internet and NGN architectures mainly use SIP protocol as a complete signalling and service delivery controlling mechanism. This protocol is highly recommended by 3GPP group for IMS architecture and also NGN for convergence network. The Real-Time Transport Protocol (RTP) [98] is the mainly used protocol for transport audio and video packets over IP network. The RTP deals with the video stream as; two flows encoded by different codec one for audio and the other for video or the visual part of stream. But, the main problem is the synchronization between the two parts of stream. Some of the most common ways of coding are MPEG series. In MPEG-4 [98] the (RTP) payload formats for carrying each of MPEG-4 Audio and MPEG-4 Visual bit-streams without using MPEG-4 Systems. For the purpose of directly mapping MPEG-4 Audio/Visual bit streams onto RTP packets, it provides specifications for the use of RTP header fields and also specifies fragmentation rules. These RTP payload formats enable transport of MPEG-4 Audio/Visual streams without using the synchronization and stream management functionality of MPEG-4 Systems.

The two standard protocols SIP and RTSP are mainly used for signalling protocols for streaming IP environment. SIP is used more in interactive media system that is why it is one of the core signalling protocols of NGN architecture.

- **Packets transmission and verification**

We have mainly two ways of packets preparation in the authentication process, either to be packet-based signature [99, 100] or block-based signature [89, 101-109]. In packet-based the level of authentication is done packet by packet. It has advantages like the absence of buffering but it has large overheads and minimum robustness. On the other hand, the block-based algorithms are built based on block level which means group of packets signed together. The main advantage of this method is its lowest overhead but it suffers from some large buffering for block processing. The one time signature proposed in [110-112] is representing common ways for digital signatures creation and verification procedures for block level transmission.

6.5 Conclusion

This chapter provided state-of-the-art about hash chain methodologies and their usage in video streaming security or authentication in general. Also, we introduced some relevant works with their analysis for achieving reliability in the applied chain. The majority of algorithms built their robustness on some redundancy in calculation and piggybacked hash values or signature in different packets. Moreover, a comparison between different algorithms was conducted in terms of processing delays, overheads and robustness. Based on this comparison, we will organize the work in the next two chapters for building our comparative results on same basis of assumptions.

In the next chapter, we will propose different methods to assure hash chain reliability in case of having packets lost. Our algorithm will be compared against different general techniques for achieving hash chain links not broken while having some losses.

Chapter7: Hash Chain Links Resynchronization Methods in Video Streaming Security

Hash chains provide a secure and light way of security to data authentication including two aspects: Data Integrity and Data Origin Authentication. The real challenge of using the hash chains is how it could recover the synchronization state and continue keeping the hash link in case of packet loss? Based on the packet loss tolerance and accepted delay of video delivery representing the permitted tolerance for heavy loaded applications, we propose different mechanisms for such synchronization recovery. Each mechanism is suitable to use according to the video use case and the low capabilities of end devices. This chapter proposes comparative results between them based on the status of each one and its overhead. Then, we propose a hybrid technique based on Redundancy Code (RC). This hybrid algorithm RC-SRS (RC-Synchronization Recovery State) is simulated and compared analytically against the other techniques (SHHC, TSP, MLHC and TSS). Moreover, a global performance evaluation in terms of delay and overhead is conducted for all techniques.

7.1 Introduction

Many security mechanisms were proposed for securing the delivery of real time applications based on Hash Chains methodology. The main advantage of hash chains is the light calculations compared to other cryptographic algorithms like the encryption methods. It also provides a fast and secure way for the real time applications that are very sensitive to any delay caused by the security overhead.

7.1.1 Video Streaming Security Measures

According to the recommendations by National Institute of Standards and Technology (NIST) [115] for securing sensitive applications and also for defining the degree of

security, there are four levels of security. These levels were organized based on group of roles define the co-relation between the operators and the provided services. That standard provides four increasing, qualitative levels of security: Level 1 (basic security measures), Level 2 (high physical layer security), Level 3 (Identity-Based security and more services authorization), and Level 4 (high level security applications).

For video streaming, the objective is different because the security module or in general the cryptographic module must take into account the nature of the application (time sensitive application), and the quality of delivery affected by the security measures delay. So, we can build our prospection to secure video streaming on the degree of importance of this stream and also the capabilities of the low end devices that will be used by the clients to access this stream like PDA devices.

Figure 7.1 illustrates the security measures classification for our video streaming study. We divide the security modules into two levels according to the nature of video diffusion (online or offline video stream) and who diffused it (important speech or normal speech). We propose a suitable hashing technique for each level based on hash chains mechanism as shown in Figure 7.1.

Our solution adapts to different cases like when the stream is important but is not online. In this scenario, we will apply a cross layer security mechanism between the two levels shown in the figure.

Therefore, the applied security measure for video streaming will not be fixed for all types of streams but it will vary according to the video requirements, the video status and the network conditions.

We may combine hashing and watermarking so as to assure a high degree of security. These measures could be used and combined with our hash chain methodology according to the type of applications under security as follows:

1. **Digital Rights Management (DRM):** is mainly designed to prevent illegal accessing, copying or converting of multimedia materials into other formats using digital devices. DRM is a generic term for access control technologies that used copyright protections. Signatures and watermarks are classes of DRM.
2. **Cryptographic Signature:** used for authentication purposes like detection of any alteration of the signed data and to authenticate the sender's data.

3. **Watermarks:** are used for authentication in especial applications and are designed to resist alterations and modifications in data.
4. **Fragile Watermarks:** are watermarks that have only very limited robustness. They are used to detect modifications of the watermarked data (like image applications).

The ability to achieve good security for real time applications requires some security measures from the above items merged with the hashing mechanism.

Hash function is a simple way to ensure data confidentiality. A hash function transforms a string of characters into a usually shorter, fixed-length value or key that represents the original string. The difference between hashing and encryption is in how the data is stored. With encrypted mode, the data can be decrypted with a key. With the hash functions, after the data is entered and converted using the hash function, the plaintext is gone. Therefore, the hashed values are only used in comparison. We have two base standards for hash building as follows:

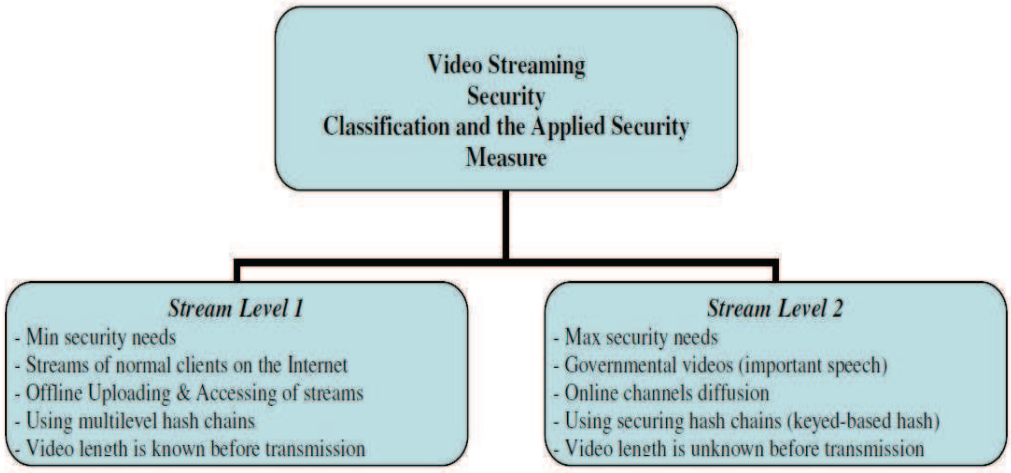


Figure 7. 1: Video streaming security levels classification

- *SHA-1* [119]: When a message of any length < 2^64 bits is input, the SHA-1 produces a 160-bit (20 Bytes) output called a message digest. So, 2^160 operations are needed for knowing the digest value (the number of possibilities that can be generated with 160 bit length).
- *MD5* [118]: When a message of any length < 2^64 bits is input, the MD5 produces a 128-bit (16 Bytes) output called a message digest. So, 2^128 operations are

needed for knowing the digest value (the number of possibilities that can be generated with 128 bit length).

For simplicity, our work relies on using SHA-1 and comparing the overhead with MD5 as they used the same block-based calculations. But, the proposed algorithm could accommodate any type of new hashing like SHA-256 or SHA-512. Although, there are several critics of MD5 like the trials to break it within max 1 hour proposed in [124], it is still implemented in many security applications. Also, the US declared that, it will gradually change to high series of SHA for the government applications. More collision resistance analysis about SHA family is listed in [125]. Actually, our objective is the reliability by finding solution to the hash links in hash chains technique. So, some details about attack mitigations will be shown in Section 7.5.

Practically, we have two use cases of video streaming under study as follows:

Offline Video Streaming: The video is in this case on the server side and has a definite length. So, the server could calculate any security measure for that total length before starting the client accessing it. YouTube and Dailymotion are good examples for this category of video sharing servers which have a huge database of short videos [26]. The Video-on-Demand (VoD) is representing this case study.

Online Video Streaming: This scenario is more complex. The length of video file in this case is unknown, so the sender can not calculate any measures of security when it is receiving the video from the up-loader or diffusing it at different time periods. This scenario represents the personal TV or live video. It corresponds also to video streaming channels hosted by some Internet content providers.

7.1.2 Related Work

Hash chain is a successive application of a cryptographic hash function *h(.)* to any string. The link of chain means that; the initialization value currently input to *h(.)* will be the output of the previous hash calculated from the previous part of data. So, the hash calculations could not continue if the previous hash digest missed.

Hash chains for video streaming have been considered extensively in the literature. The survey in [122] provides a good study about using hash chain in video streaming. It mainly conducted a comparison between many algorithms which proposed handling the issues of hash chain with video stream authentications. However, handling of the

resynchronization problem for broken hash links still needs additional work. Our previous work [116] highlighted the resynchronization issue in hash chain links and categorized some solutions for it. Then, we added some security measures based on signing specific packets in the video stream [117]. The work in [88] gives a good starting point for how to sign a digital streaming video. Authors proposed two cases; offline and online streams. They chain blocks based on the packets inside the block. Each block carries the hash of the next one (online case). For the offline case they calculate the hash based on the whole video and the receiver must have some buffer so as to start the verification after a specific length of the video. Their algorithm does not handle the redundancy of chain links.

In [92], authors introduce the Butterfly Graph. They divided the packets into groups and each group has one signature calculated based hashing. The redundancy is achieved by sending the signed packets several times. Their overall concern is to keep a good performance as the amount of redundancy is increased. Also, in [93, 94] they examine the problem of streaming of authenticated video over lossy public networks depending on the ideas of Graph and taking into account the quality of wireless channels. It is a kind of optimization technique for authenticating the streaming packets which called Rate-distortion-Authentication (R-D-A). Moreover, they achieved remarkable optimization in media quality and packet overhead.

In [89], the work is based on signing a small number of special packets in data stream; each packet is linked to a signed packet via multiple hash chain. The links depend on six hashes per packet. Hence six packets carry the same hash value and this represents a large overhead. Two solutions for securing the video stream are compared. The first solution is called TESLA (Timed Efficient Stream Loss-tolerant Authentication). The second scheme is called EMSS (Efficient Multi-chained Stream Signature).

Another work [90], handled the video stream authentication by assuming a combination of one-way hashes and digital signatures to authenticate packets. Their idea can be explained as follows; for collision resistant, the hash of packet P_i is appended to packet P_{i+1} before signing P_{i+1}, then the signature on P_{i+1} guarantees the authenticity of P_i and P_{i+1} at the same time. The drawback of that proposal is the large overhead as it increased linearly with the growth numbers of packets.

A time-critical multicast authentication scheme was proposed in [97], which combines hash chains with one time signature to authenticate streaming of packets. The algorithm

provides short end-to-end computational latency, perfect tolerance to packet loss, and strong resistance against malicious attacks. They used long key for achieving high security which leads to large overhead.

7.1.3 Work Motivations and Organization

Our objective in this chapter is to study and design novel solutions for hash chain resynchronization in case of some packets loss. For conducting this study, we assume some parameters that will be repeated in many sections as shown in Table 7.2 (Section 7.3.1).

This work will focus on the handling mechanisms for the re-initialization problem to keep the continuous hash chain in case of packet loss. This loss can break the link of hash chains and may lead to restart the process again. We propose adding some redundancy codes that will be calculated based on the hash values of different Blocks from the video. Those redundancy values will be inserted in some packets inside the Block. Hence, those values will help the receiver side to extract the hash values for each Block without correctly received the whole packets of this Block. This means that, in case of some packets are lost from the Block, this loss will not affect on the continuity of hash chain used to authenticate the video transmission and also will not lead to stop the streaming.

The rest of this chapter is organized as follows: Section 7.2 gives overview on the hash chain synchronization problem and the proposed solutions comparison. Section 7.3 illustrates our algorithm architecture and assumptions for achieving redundancy of hash link. Section 7.4 introduces the window mechanism capabilities. Some attacks analyses are studied in Section 7.5. Section 7.6 shows the results and Section 7.7 concludes this chapter and its future directions.

7.2 Hash Chains Resynchronization

Hash chain is a known technique used in many applications. Our work depends on a simple forward one way hashing system as shown in Figure 7.2. This type of sequential hashing is often used in practice and requires relatively less memory than other types of parallel hashing. Moreover, it is more convincible for low capacity end devices. If we adopt the traditional hash chain in its typical way it will request a complex

synchronization system. So, it will need large memory and buffering capacities from the clients. So, by adopting of forward and sequential way in hashing system will avoid these difficulties. Also, the buffering sizes will depend on our redundancy factors for how many Blocks/Window under processing as we will explain later in results section.

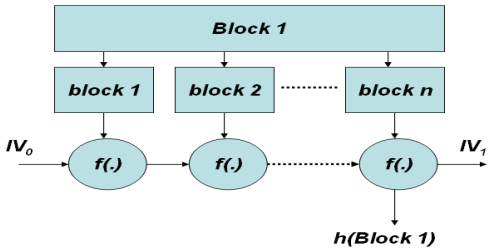


Figure 7. 2: Simple construction of hash chain mechanism for processing one Block/Window

IV_0: is the standard Initialization Vector according to the type of hash algorithm

IV_1: is the final output from hashing Block 1 and will be the initialization vector of Block 2

block: is representing standard unit for processing hash function (for example 512 bits for MD5 or SHA-1 Modulo 512)

Block: is representing a group of packets (for example one Block=100 packets)

When the hash chains are applied to the video streams like VoD or IPTV, it faces several problems for keeping the hash link continuity in case of packet drops. As some packets are lost, this will cause mismatch calculation in hash link or Message Digest (MD) between sender and receiver. So, we are looking for continuity of hash chain in case of that loss happened.

In the next sections we categorize the solutions for this problem into four categories (SHHS, TSP, MLHC and TSS). Then, we discuss the advantages and the disadvantages of each technique. Moreover, we suggest a new hybrid technique that collects the best features from the four methods of hash link synchronization.

7.2.1 Self-Healing Hash Chain (SHHS)

Self-Healing is a modern technique that has many uses in the smart networks. It means that creating a process that has the ability to recover its state in case of system failures without external help. This technique was used with hash chain in many applications like ‘‘Self-Healing Key Distribution’’ [114] which focus on the users

capability to recover the lost group keys on their own, without requesting additional transmissions from the group manager.

Video streaming can be delivered based on TCP or UDP transport protocols. TCP is mainly used to overcome Network Address Translation (NAT) filters but, the most appropriate is UDP. Usually, during online streaming, we have some packet drops that can be considered with respect to some packet loss tolerance. The acceptable tolerance may not affect the streaming quality. However, the loss could affect on synchronization of the hash chains of the stream.

The Self-Healing Hash Chain (SHHC) can overcome this problem by re-synchronizing the chains despite the loss of some packets from the stream. The SHHC is a robustness system able to resolve the synchronisation problem of chains. As in the Figure 7.3; the stream is divided into specific time blocks and at each time a hash must be calculated for this period of time. The reliability will depend on the redundancy factor of the concatenated hashes. To guarantee the synchronisation, three hashes may be concatenated together.

This procedure will add some redundancy for tracking the link synchronization points of the stream. Also, the advantage of this scheme is the low overhead for memory and calculations.

As in the Figure 7.3; the stream is divided into specific time Blocks and for each period of time a hash must be calculated. The sending parity will depend on some redundancy of the concatenated hash values that will be transmitted. We strongly recommend that three hashes must be concatenated so as to guarantee the synchronisation of links between sender and receiver. The concatenated values are inserted in the last packet of each Block of the video.

The sequence for concatenation mode can be explained as follows:

- Hash of time zero will be $sign \,||\, h(t_1)$, means the signed packet of the first part.
- Hash of t_1 time-end will be $h(t_1) \,||\, h(t_2)$
- Hash of t_2 time-end will be $h(t_1) \,||\, h(t_2) \,||\, h(t_3)$
- Hash of t_3 time-end will be $h(t_2) \,||\, h(t_3) \,||\, h(t_4)$
- Hash of t_n time-end of stream will be $h(t_{n-1}) \,||\, h(t_n)$

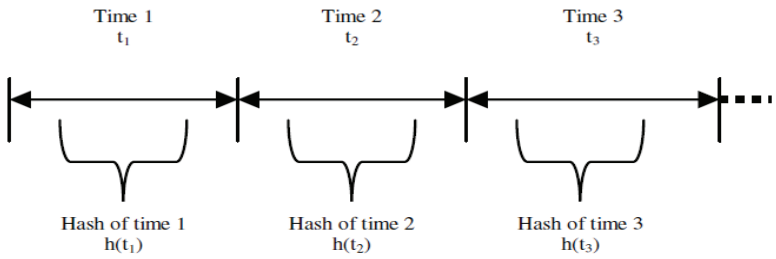


Figure 7. 3: The time hash-chain of sending party

The pros of concatenations are: concatenating outputs from multiple hash functions provides collision resistance as good as the strongest of the algorithms included in the concatenated result.

For less overhead, we can replace the concatenation process by XORing process. This replacement will reduce the overhead sent with in the stream by 1/3 for the base of 3-hash concatenated together.

The self-healing feature comes from the receiver's ability to extract or recalculate any hash without receiving the total Block or Window (group of Blocks).

So, at any time the sender transmits three hashes to link the time Block of this time with the previous time Block and the coming one. This procedure will add some redundancy for tracking the synchronisation points of the stream.

7.2.2 Time-Synchronization Point (TSP)

This mechanism is used to assure synchronisation of hash chains in case of packet loss. It depends on adding additional information bits in the stream. The stream must be divided into a pre-defined specific time Blocks. After each Block, a synchronisation point must be inserted in the sender side as shown in Figure 7.4. Those inserted points can help the receiver tracking the synchronisation of the stream.

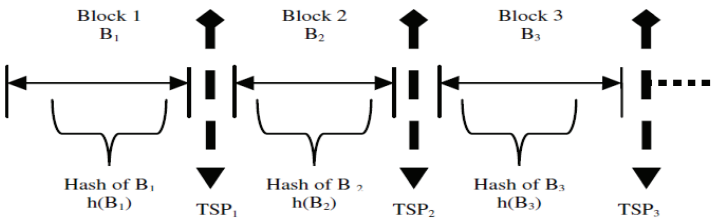


Figure 7. 4: The time synchronisation point of sending party

In this case, the receiver must keep tracking those times synchronisation points (TSP) so that not to lose the stream synchronisation or hash link breaks. The drawback of this technique is the large bits overhead added for synchronisation because it depends on adding an extra packet for this purpose. For more information about overhead comparison see Table 7.3.

In this method, if we assume that each Block or Window has (N) packets, then the packet number (N+1) will be redundant packet. As, we insert this packet after each Block, it will add a global overhead on the stream depends on the total number of Blocks in the video file.

7.2.3 Multi Layer Hash Chain (MLHC)

This technique was gave good results for the problem of security assurance for the E-lottery winners and their serial numbers generation tickets [113]. The multi layer means to calculate hash chains where each value represents one layer according to the base of calculation. Although the objectives are different (e-lottery and video streaming), this technique could be very effective in offline video streaming security mode. When we use this technique in video streaming the layers conception will completely be different so as to match the specific nature of the real time applications. We can highlight the impacts of two layers hash by the example in Figure 7.5. In this structure, we have two concurrent layers of hashing as:

- $H`_i = h(W_i, IV_{st})$: *unkeyed hashing step which depends on standard initialization vector.*
- $H_i = h(H`_i, IV_{sec})$: *keyed hashing step where the key is equal the* IV_{sec} *(secure IV).*

And if the round function used is h(.), then:

$$H`_i = h(h(.....h(h(IV_{st}, W_1), W_2), ... W_{i-1}), W_i)$$
$$H_i = h(h(.....h(h(IV_{sec}, H`_1), H`_2), ... H`_{i-1}), H`_i)$$

This nested double layer hash chain can thwart many high level attacks to the stream and the hashing itself.

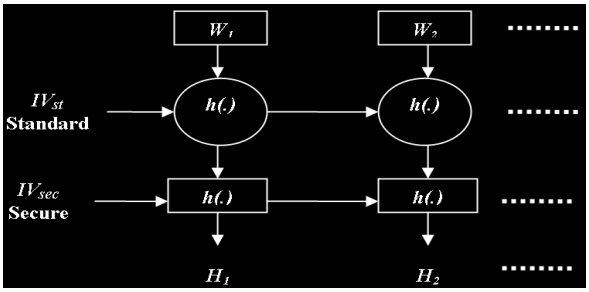


Figure 7. 5: Two layers hash chain mechanism

7.2.4 TimeStamp Synchronization (TSS)

The sequence of packets could be used as a key for achieving video synchronization and also keeping the link of hash chain. This can be efficient with lesser cost and time overhead because the timestamp is a mandatory field in Real Time Protocol (RTP) packet as described in [98] for RTP packets of MPEG-4 streams. In this case, The RTP packets are responsible for sequence numbers and timestamp synchronisation (TSS) between the source and destination. The benefit of this technique is the reuse of parameters from RTP standard packets header. Therefore, the sequence number and timestamp for each packet are good indication or index to where is the lost point. So, the added digest value plus the original packet timestamp in all stream packets or in one packet per Block or Window of packets represent the link between sequential hash chain outputs.

V=2|P|X| CC |M| PT | sequence number
Timestamp
Synchronization source (SSRC) identifier
contributing source (CSRC) identifiers ....
MPEG-4 stream (byte aligned) ...OPTIONAL RTP padding

Figure 7. 6: RTP packet headers for MPEG-4 stream as in [98]

Window time-stamp: the time stamp that will help in hash chain link synchronisation is 4 bytes per packet as shown in the Figure 7.6. But, for our Window based of calculations, we will consider the 4 bytes only overhead per Window for achieving hash link synchronization which nothing added else the digest value according the cryptographic hash algorithm used. Therefore, if we have two parities under synchronization of timestamp, then the last packet of each Window or what is called the

Window timestamp packet will responsible of hash link synchronization as the following:

Let $\mathbf{T_i}$ is the timestamp of Window $\mathbf{W_i}$, then the Window Digest will be:

$$WD_i = h(W_i, H_{i-1})$$

Where: H_{i-1} is the previous Window Digest $\mathbf{W_{i-1}}$. But, we need to add the timestamp to this calculation by concatenating it to the previous Digest. So, the final Window hash will be:

$$H_i = h(\ WD_i \parallel T_i\ ,\ H_{i-1})$$

So, the hash chain links relations H_{i+1}, H_i, H_{i-1} could be built based on the Windows timestamps:

$$T_{i+1},\ T_i\ ,\ T_{i-1}\ .$$

To conclude those four techniques, we can make performance comparison between them as shown in Table 7.1. This comparison is the base for building the proposed hybrid technique.

Table 7.1: comparison between different techniques for resynchronisation in term of performance

	Robustness	**Recovery Time**	**Memory Overhead**	**Bits Overhead**	**Calculation Overhead**	***Video Streaming Mode***
Self-healing	*High*	*Low*	*Very Low*	*Very Small*	*Small*	*Online & Offline*
TSP	*Moderate*	*Very Low*	*Small*	*Large*	*Low*	*Online & Offline*
Multi-Layer	*Low*	*Large*	*Large*	*Small*	*High*	*Offline*
TimeStamp	*Moderate*	*Very Low*	*Low*	*Small*	*Very Low*	*Online & Offline*

7.2.5 The Hybrid Technique Based (RC)

Finally, we suggest a hybrid technique RC-SRS capable of inheriting the advantages of the previous solutions and also overcoming most of their drawbacks. This technique could be useful for slow processor endpoints so as to minimise the calculation needs to be performed during every session. Also, it could be used to rapidly re-establish the link

synchronization in case of packet loss session problems or delay time. Another criterion is packet and hash information caching that have more CPU intensive compared with using time synchronization that may be important for low-memory mobile platforms.

Our proposal is inspired from redundancy code techniques. Redundancy Code (RC) is a generic concept introducing some redundancy in the system to overcome hash chain break in case of packet loss and to increase the reliability of the security system. In the next section, we introduce more details about using RC with hash chain.

7.3 The Proposed Redundant Hash Chain Method

Before the description of the proposed architecture, we must have a look on the packetization sequence of the video stream. Figure 7.7 illustrates the simple sequence in standard manner based on real time transport protocol as in [98]. The video is considered as a group of chunks output from the coder such as MPEG-TS [123]. This gives better clarification on which the stream word represents for us and what is the packet structure for our proposal. The hashing calculations will be done after the highlighted row in Figure 7.7 (for transmitter) and before in the case of the receiver.

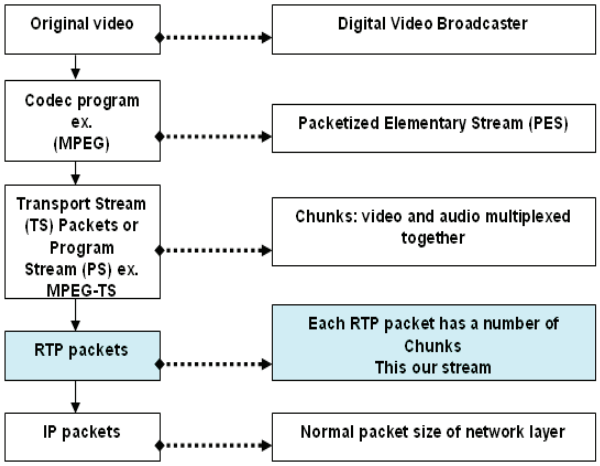


Figure 7. 7: The sequence diagram for video packetizing in the transmitter side

The proposed architecture in Figure 7.8 has many parameters that need to be initialized:

Stream of Chunks: are the output blocks after MPEG-TS (like RTP packets).

Blocks B_i: is a group of packets that have a relation with their numbered Chunks. For example; each Block=10 Packets and each Packet=7 Chunks in case of RTP Packet.

IV_0: is the initial victor for starting the hash chain.

h(.): is the hash functions used for calculate the output hash like MD5 or SHA series.

h_i: is the output hash value or the output digest.

Hi: is the output hash digest for two layer hash technique.

Combination Code: is the coding process that will be used to calculate a redundancy code for generating the hash value in case of missing a hash value of Block (ex. XOR function).

RC_i: is the output Redundancy Code that is responsible for recalculate the missed hash value so as to keep the hash link not broken. It is something like Forward Error Correction (FEC) codes.

The added Redundancy Code will allow the receiver to detect and correct the errors in the digest values (under some restrictions). This code is the key factor for solving the resynchronization problem of hash link.

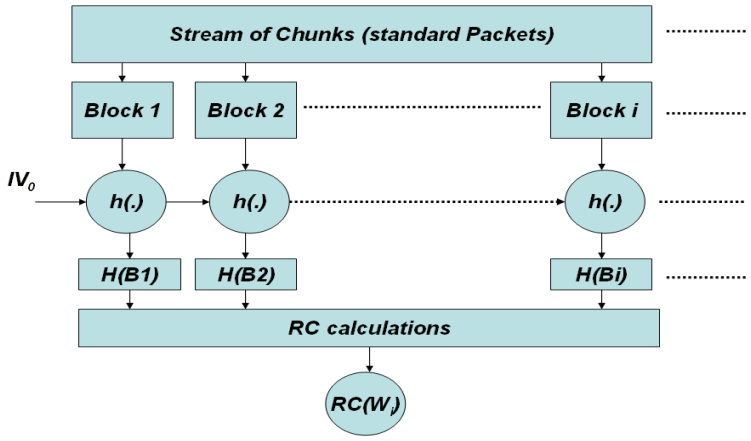


Figure 7. 8: Block diagram for the hash chain redundancy for video streaming

The original stream is divided into chunks (series of packets) then assembled them to specific Blocks Bi after that the hash chain applied to the Blocks; finally, the RCs calculated based on static Window size.

Hash links recovery could happen without asking the sender for additional data retransmission because the sender is memory-less in case of online video streaming. The advantages of RCs are that; buffering is not required and the retransmission of hash values can often be avoided (which reduces the bandwidth requirements, time calculations and the buffering memory). RC is therefore applied in this situation where the retransmissions are difficult to achieve in real time applications and memory-less devices. The main objective from RCs is the hash link synchronization and finding the

recovery point of synchronization by obtaining the hash value of this time which represents the initial vector (IV) for next hash calculation in our chain.

7.3.1 Assumptions

All the previous proposals mentioned in Section 7.2 differentiated between the concept of ***offline*** and ***online video streaming***. They made their calculations based on the pre-known video length in case of offline stream. Also, the online stream has an infinite length assumption. In both cases, if the receiver has some restrictions about the processing capabilities and the buffering capacities (Memory Buffers), the two cases lead to one case only which is the ***online*** scenario.

Our proposed solution is built to fit the new generation of handheld devices that have some limitations in all processing capabilities compared to the normal PCs. So, the treatment of any video will be considered as an online one (from the receiver side) although if in some cases the sender knows all videos lengths accessed by the others. This assumption will eliminate the need of buffering of data at the receiver side before starting the playing of video in case of large videos.

We assumed that, the redundancy in this case is mandatory for synchronization matter. But, when we calculate the RC for some part of data, this calculation will mainly depend on the degree of redundancy and the accepted overhead.

For example, if RC calculated based on 3 hashes values like RC_1= combination (H_1, H_2, H_3) and RC_2= combination (H_3, H_4, H_5) then we have redundancy 3/4 with dynamic sliding Window. But, if we consider RC_1= combination (H_1, H_2, H_3) and RC_2= combination (H_4, H_5, H_6) then, it will represent the static sliding Window which means that no relation between the two Windows. If we take 4 hashes values the redundancy will be 4/5 and so on. So, which factor will be control the calculation of the RCs codes? This is one of the most effective factors in the calculations.

In Table 7.2, we assumed some parameters and values that we used in the calculation of hashing and RC values. All of the assumed parameters in the table were preselected based on the packet standardization size for Real Time Protocol (RTP). The calculated sizes for the Block and the Window are output result from the analytical and simulation results conducted in Matlab.

Table 7.2: Proposed parameters

Parameters	***Symbol***	***Definition***
Packet	*P*	*Standard packet size like MTU size of 1500 Bytes*
block	*b*	*Standard block size for hashing algorithm like 512 bits for MD5 or SHA-1*
Block	*B*	*The number of packets to be processed together*
Packet Rate	*PR*	*= VBR/MTU (packets per sec)*
Block Rate	*BR*	*= PR/ Block Size (Blocks per sec)*
Video Bit Rate	*VBR*	*For example 512 Kbps or 1 Mbps*
Window	*W*	*Is dynamic buffer contains number of Blocks*
Hash Function	*h(.)*	*Is the hash algorithm used like MD5 or SHA-1*
Hash Output	*H*	*Is the output digest or hash value of each Block*
Packet Error Rate	*PER*	*Probability of packets loss or error in the Block*
Hash Error Rate	*HER*	*HER = (PER/Block size).RF this for any Window*
Redundancy Factor	*RF*	*The number of Blocks per Window processing in scanning*

7.3.2 Sender Security Tasks

This part focuses on how the sender prepares the packets and puts the calculated hash values and redundancy code in the packets? Also, how each packet will have an index to that place in the packet which caries this redundancy code?

The Window mechanism adopts a technique of dynamic buffering. This buffer depends on some parameters like video rate and client processing capabilities. After an agreement done between the sender and receiver, the sliding Window mechanism will be conducted according to the redundancy factors adopted.

The complete steps for the implemented algorithm are:

1. Input video file.
2. Divide the file into chunks by MPEG-TS coder each (188 Bytes).
3. Define the Packet size (each packet 7 chunks).
4. Define the Block size (variable from 10 to 100).
5. Define the used hash function MD5 or SHA-1.
6. Start hashing Block by Block with initialization vector of current Block is the hash value of previous Block (chain mechanism).
7. Calculate the Redundancy Code (XOR two or 3 hash together).
8. Insert the RC code in specific packet (or more than one).

9. Index each packet with the location of RC place [116].
10. Add transport headers and send the packets of Block according to the Window size.

7.3.3 Receiver Verification Tasks

All processing of the received packets are after RTP layer. The receiver will handle the verification of Blocks or Windows according to the Window size which is controlled by the Redundancy Factor (RF). Therefore, it can process one Block and compare its hash digest with the received one. If they are identical, this means that the received Block is correct and it will process the next. Otherwise, it will wait till the Window completes its size and then use the RC value to drive the hash of the previous Block to help the receiver continuing its verifications for the next Block.

The total procedure for verification is as follows:

1. Read the received RTP packets.
2. Group the Block size (ex. from 10 to 100 packets).
3. Verify the whole Block secured hash and the index for each packet.
4. Compare the receive hash with the calculated one.
5. Drop the packets that not have the correct index.
6. Use the redundancy code RC in case of packet loss to know the hash value or the signature of that Block.
7. Divide the packets to chunks for MPEG decoder.
8. Decode the packet elements of the video.
9. Run the application to view the video in case of (OK) for the predefined tolerance for packet loss.
10. Return verification pass (OK).

7.3.4 The Recovery Time

The recovery time is the receiver waiting time before recovering the missed hash link based on the RC value. This time must be less than the standard Round Trip Time (RTT) value.

We have two scenarios for recovery:

1. ***Best Case:*** In this case, the delay time for recovery is very small as compare to RTT so as to avoid requesting new (IV) for re-initialization process.

2. ***Worst Case:*** In this case, the delay time is larger than the best case because the loss happened in the beginning of the Window and the receiver will have to wait some times till it receives the entire Window. But, in this case, the delay will be less than RTT or the receiver will prefer to reinitialize than recovering.

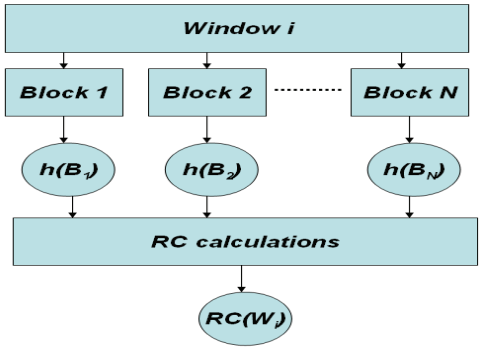


Figure 7. 9: Processing of static Window sliding over N Blocks

If we define the Recovery Time as the waiting time for the receiver to recover the hash link in case off any Block error and as shown in Figure 7.9, any Window consists of (N) Blocks and the Block Rate is BR Block/sec as shown in Table 7.2, then we can calculate the waiting time as:

Waiting Time: $W_T = (N-i)/BR$ where (i) is the error Block position in the Window

For the best case: (i = N), the error occurred in the last Block of the Window (ideal scenario for recovery)

$$W_{Tb} = (N-i)/BR = (N-N)/BR = Zero$$

For the worst case: (i=1), the error occurred in the first Block

$$W_{Tw} = (N-i)/BR = (N-1)/BR$$

In all the cases the W_{Tb} or W_{Tw} must be less than the RTT value so as to prove that; it is best for the receiver to depend on RC value for recovering any missed hash link rather than to reinitialize (IV).

7.3.5 Offline Access Initialization

From the Figure 7.10 below, we have two phases: the uploader-server phase and the client-server access phase. We assume that; the first phase is pre-secured at the server side. Moreover, this phase can be secured more and more using encryption techniques especially in the offline scenario as the online real time feature not exist.

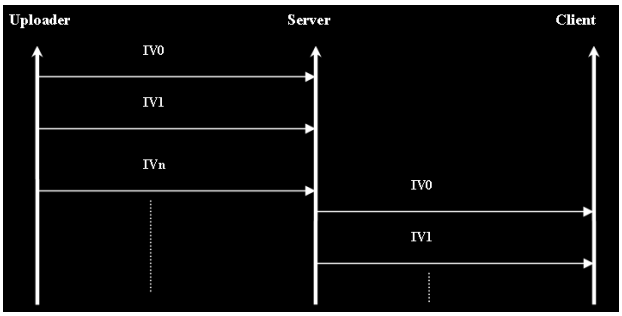


Figure 7. 10: Offline joining case

Client's joining procedure: When a user wants to join the media streaming server, he/she should first pass the authentication phase in the secured manner as it explained in [117]. Then, it will be assigned directly to the first Window number and starting the indexing from zero (IV0) because there is no need for its timestamp as the access is offline and must start from the beginning of the video.

7.3.6 Online Access Initialization

At any time, an uploader can start its online video diffusion and any user can access this stream from the hosting server at the instant of its joining. The server initializes the IV_0 for this client by IV_t where (t) is current time of the server side.

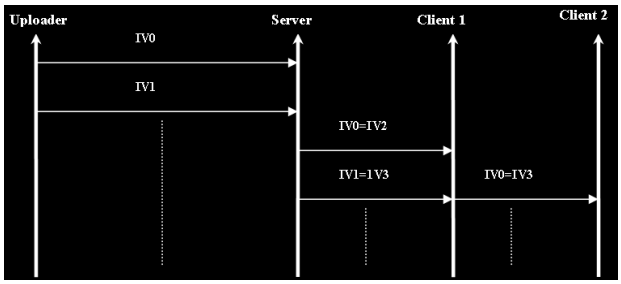


Figure 7. 11: Online case with different joining time access for clients

In Figure 7.11, client 1 joined the online stream at time t_2 which means that; he missed 2 *Windows* from the beginning of the stream and client 2 joined at t_3 which means he missed 3 *Windows* from the starting time of diffusion.

Client's joining procedure: When a user wants to join the media streaming server, he/she should first pass the authentication phase in the secured manner explained in [117]. Then, it can find the trusted starting point according to its time-stamp for assigning to the nearest Window index number [116].

7.4 Window Mechanism Capabilities

This mechanism proposes a high degree of reliability for hash chains. The algorithm depends on an agreement between the server and clients about the processing capabilities for a number of packets or bytes per window. This window size also has a direct relation with the video streaming rate. The RC code is calculated based on this size and each window has sequence number for reliability considerations. Figure 7.12 illustrates this window mechanism. So, the *window* is a dynamic buffer which its size defined by video playing rate and receiver processing memory.

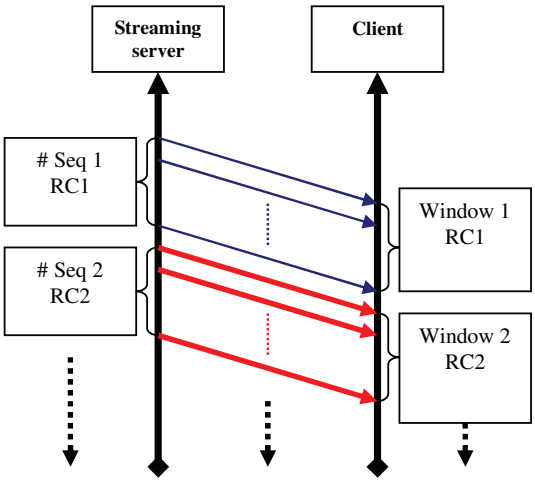


Figure 7. 12: The window redundancy mechanism

For example, if the RC combines 3 hash values of three Blocks then the window size will be 3xBlock size. So, the window has a direct relation with the redundancy calculation. Also, the buffer has a relation with the delivery rate of playing stream and

processing capabilities of the clients. The following parts discuss the challenges of synchronisation and their treatments.

7.4.1 Online versus Offline Synchronization

Online: It is not loosely time synchronized because the joining time must be the initial point of synchronization between the hashing generator in the sender and the verification process in the receiver.

Offline: It is loosely time synchronized case because any receiver can freely join the stream at any time it likes. At any initial time value (*Ti)*, the verification side can negotiate the starting point with the generator side.

We have two scenarios of joining clients:

- ***Clients start playing from initial value of stream:*** in this case the initialization vector will be the standard one proposed by the type of hash method. For example SHA-1 has a specific 160 Bit IV and MD5 has also standard 128 Bit IV.
- ***Clients start playing after some time passed from the stream:*** in this case, the client joins the online stream after some time of the broadcasting start. So, the problem is the initialization vector for starting the process of chain. Each window has RC value that will help the receiver in finding the IV for its joining time.

7.4.2 The Reliability: Anti-replay and Anti-delay

The reliability and robustness could be achieved by the sequence number of the redundancy code. Each time the sender prepares window size packets, it defines the sequence number for those bundles of packets. The receiver does not check the bundle that did not arrive in its correct order so as to prevent the replay attack. This sequence number is embedded by the sender side as shown in Figure 7.12 for each window. For the drop or loss packet attack, we have in this case; each packet has an index to the location of RC code that will help the receiver verification for this packet. Each window has specific number (N) and each packet has an index number (n) where each ($n \leq N$). The packet that will come within out of its order ($n \succ N$) it will be directly dropped by the receiver verification procedure.

If we have any file with (n) packets it will be divided to N windows or Blocks where each (N=n/m) where (m) is the number of packets per window. The index number of

packets (P_1 till P_m) for each window has a direct relationship with the generated sequence number of any window (N).

7.4.3 Collision Avoidance

Let us assume that a hash function 'H' having two different messages x, y (where $\boldsymbol{x \neq y}$). If $\boldsymbol{H(x) = H(y)}$ this means a collision occurs.

As we said before, the simple idea of hash chains is that each packet will carry the hash of the previous packet, hence we called it chains.

We propose a simple hash algorithm that will decrease the computations layer into two layers only. The first one will hash the original message and the output will be the ***Ho*** (hash original) and the second one will be the ***Hn*** (hash new) to the previous ***Ho***.
For example, if we have message X=(x1,x2,…….,xn) the hashing output will be H(X)=h(x1),h(x2),……..,h(xn), which represent the normal hash (the first phase in Figure 7.8).

In the second phase, we will apply hash chains Hc(X)=hc1(h(x1)), hc2(h(x1)),……, hcn(h(xn)). That is why RC-SRS algorithm has the layering features. The final message X that will be sent is:

X={x1 || hc1(h(x1)), x2 || hc2(h(x2)),……, xn || hcn(h(xn))}

If any one (i.e. attacker) detects this message and tries to calculate the hash of X, he will find H(X) and not Hc(X) because Hc(X)=H(H(X)) on the base of Hc is hash chains (i.e. double layers hashing). The hash chains used IV that is generated locally on the client and no one can detect it easily. So, the degree of security for the second phase is one of our objectives to enhance the data integrity sent or received. The summary of the layers process of hashing are shown in Figure 7.13.

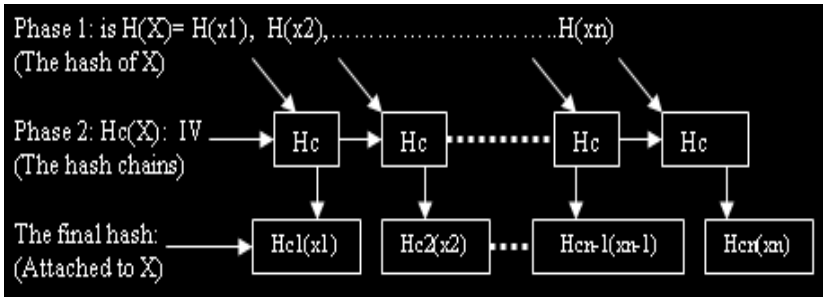


Figure 7. 13: The layering process for collision avoidance

For any man-in-the-middle (MITM) captures the data X and calculates its hash H(X), it will not recover Hc(H(X)). So, the data integrity will then be assured.

Till now, we have H(X) the output of first hashing and H(IV,X) the output of second hashing (pure hashing). But, we can add more security if we use H(Ks,IV,X) which Ks is security key applying on the hash function and this leads us to Digital Signature (DG) which will be handled in the next chapter. This Ks value can be generated by private key generator PKG and distributed by Diffie-Hellman (DH) key management algorithm [127] or Elliptic Curve Cryptography (ECC) technique [128].

7.4.4 Indexing Mechanism

For MPEG-TS, the output chunks (shown in Figure 7.7) are equal 188 Bytes/chunk. The RTP captured from the packet analyser during the simulation was of size 1370 Bytes which is equal to 7(chunks) x188 Bytes + 54 (total headers rest). So, on the base of 1500 Bytes standard packets which called the Maximum Transmission Unit (MTU), we still have 1500-(1370+2 bytes for index to the place of RC) = 128 Bytes. We will depend on using two bytes in our indexing mechanism. The RC value will be put in any random packet inside the window so as to overcome the packets tracking attack. Each packet inside the window has hierarchal indexing towards the RC location starting with window number (as shown in Figure 7.12) followed by packet order inside this window that contains this redundancy code.

For the sender, it prepares the packets and piggybacks the calculated hash values and redundancy code in the packets. Also, each packet will have an index to that place in the packet which carries this code.

For the receiver, as each packet contains an index to RC and window number. It drops the packets that not belonged to the receiving window sequence and have not the right indexing.

7.5 Attacks Analysis

As our hashing technique uses keyed-hash functions, the majority of attacks can be thwarted. This part gives an overview on some high level attacks that can affect on the hashing or the link of chain. Those attacks may help in breaking the hash link and causing non-synchronous video.

- **Replay attack:** (the attacks produced by delaying or deleting some video packets and resend them or anther to the destination along the same path). The *time-stamp* property can eliminate this attack. The Window transmission timestamp can resolve this problem by checking in the receiving side.
- **Padding attack:** (the attacks generated from adding some bytes to the original data and recalculate and resend the hash of new padded data). This attack can be easily eliminated by pre-pending the *Window Size or Length* because it is impossible to pre-pending the whole video length in case of online but it is possible in offline streaming.
- **Packet loss problems:** The UDP transmissions are unreliable and can cause some packet loss and others come within different order. The *indexing mechanism* of the *Window algorithm* can overcome on this problem.
- **Collision attacks:** this attack relevant to the hash algorithm used. MD5 and SHA-1 suffer from this attack which includes two aspects (Preimage and Birthday attacks). But, our hash construction can overcome these attacks as the following:
  - For Preimage attack: the RC calculation based concatenated 2, 3 or 4 Digest values gives impossibility to this attack. The concatenation gives some strongest to the final hash value. More over, the secure IV used will add some complexity to cryptanalysis attackers' procedures.
  - For birthday attack: the Multilayer construction increases the complexity of finding two messages having same Digest value.

Moreover, the hashing structure plays an important role in the degree of security. For using keyed-hash over unkeyed-hash have the following pros:

Keyed hash mechanism proposed in [120, 121] which called HMAC is a good example for Keyed-Hashing for Message Authentication Codes based on MD5 or SHA-1. It depends on secure shared key used with any standard cryptographic hash function between two parities to add some security measure for the message integrity and source-destination authentications. The degree of security could be increased if we used secure initialization vectors for hashing the Windows of video stream. This IV can be created and defined by the same manner explained in [117] based on PKG private key generation system with the elliptic curve secured manner. Therefore, the value added to cryptographic hash functions by the keying system used can overcome many weakness and some attacks related to normal hashing or what is called unkeyed hash.

7.6 Results

In general, we built our analytical and simulation results (using Matlab) based on the assumed parameters and values in Table 7.2. Moreover, these assumptions were assumed based on some standards like packet size equal MTU and the delay times for video streaming within 1 to 2 sec maximum. But, it is important again to re-mention the difference in structure between packet, bock, Block and Window as the following:

- **packet:** is the standard packet size 1500 Bytes
- **block:** is the standard size of block used by hash algorithm which is 512 bits for MD5 or SHA-1
- **Block:** is the number of packets to be processed together
- **Window:** is the total buffer which consists of number of Blocks depending on some parameters like: video rate, processing delay and RTT value

The Round Trip Time (RTT) is delay time consumed by the client to join the streaming server. It is important for our proposed algorithm to have total Delay time based on RC calculations and buffering or de-jittering less than the RTT or the client will prefer to initiate the session by requesting initialization vector. In this case the total Delay may be greater than RTT. The following equation expresses the total Delay related to buffering based static Window calculations:

$$D_{buffer} = D_{dejitter} + D_{calcul}$$

We have simulated the Redundancy Code Synchronization Recovery State (RC-SRS) algorithm and analyse some preliminary results. The obtained results are based on some videos assumptions. Assume that, we have video file that needs buffer size (Bs) equal 2Mbit, then (Bs=2Mbits) and transmitting rate (R=1Mbit/Sec) then the total delay = (Bs/R) = 2 sec. So, if we have two delay times as the following:

$\mathbf{D_{calcul}}$: processing time for calculate hash (sender) and verification (receiver).

$\mathbf{D_{wait}}$ **:** total delay time before using the RC to recalculate the hash link of any Block inside the Window according to the Block order in the Window (Best or Worst case as explained in Section 7.3.4).

Then $D_{calcul}+D_{wait}$ must be less than (Bs/R) which 2 sec. So, our threshold condition will be:

$$(D_{calcul} + D_{wait}) < 2 \text{ sec}$$

$$D_{wait} = (\text{No packets x PS Packet Size}) / R \text{ (bits/sec)}$$

We will put the two second in this case as Max threshold allowed delay time and change the number of packets to find the max number of Block size or buffer under the above conditions.

Figure 7.14 shows the relationship between the numbers of packets/Window or Block versus our assuming delay time from 1 to 2 sec. The curve gives 85 packets as optimum number for Window or Block size.

Figure 7.15 compares the total overhead (the added bytes to stream as a redundancy code) by using MD5 or SHA-1 hash algorithms. As shown, if we assume the number of packets per block is equal to 10, so the full redundancy means sending the RC 10-times (means with each packet). But, this will lead to very high overhead.

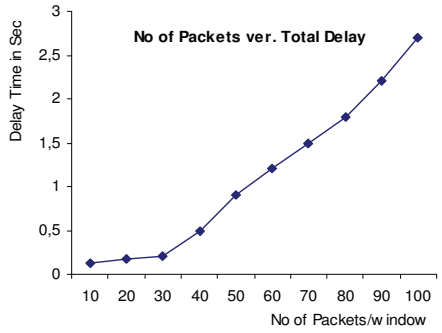


Figure 7. 14: The optimal number of packets under maximum allowable delay time (1 to 2 sec)

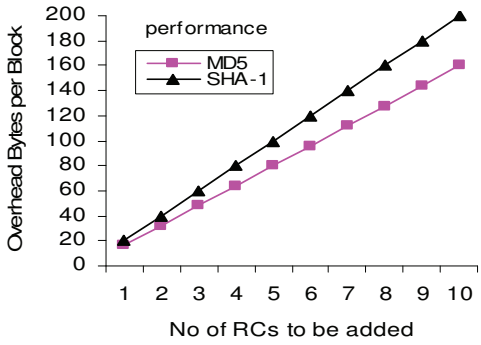


Figure 7. 15: The overhead bytes in terms of number of redundancy trials

If we take an example; for MPEG-TS the output chunks are equal 188 Bytes/chunk. The RTP packet captured from the packet analyzer during the simulation has size of

1370 Bytes which is equal to 7(chunks) x188 Bytes + 54 (total headers rest). So, for 1500 Bytes standard packets we still have 1500-(1370+2 bytes for index to the place of RC) = 128 Bytes. Those 128 Bytes give us the probability of sending the RC 8 times in case of MD5 and 6 times in case of SHA-1 as shown in Figure 7.15. Those results were obtained under our assumptions of packet size 1500 Bytes and Block size 85 packets.

In Figure 7.16, a comparison between different methods has been made in terms of processing time for each Block of video against different video rates. This Block is almost is almost 2 sec in case of SHHC technique and 85 packets (for each packet size 1500 Bytes) in case of TSP, MLHC, TSP and RC techniques. The results indicate minimum accepted calculation time for our proposal based RC which the average time about 200 msec for each Block.

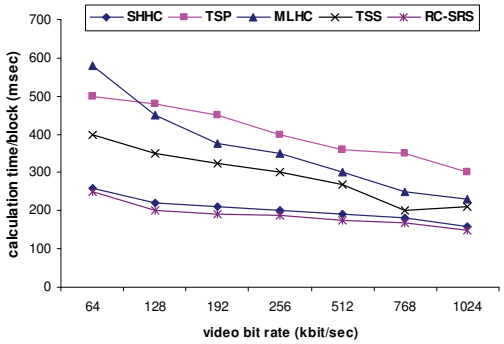


Figure 7. 16: The computation time comparison between the different methods against different videos bit rates

For the error rate and its impact on hash recovering or reliability of our technique, we have two parameters control this process:

Packet Error Rate (PER): is the probability of an error occurred in any packet of the Block. So,

$$PER = 1/Block\ size$$

If the *Window* has (M) *Blocks* then the total will be:

$$PER_T = (1/Block\ size)/M$$

Hash Error Rate (HER): is the probability of an error occurred exactly in the packet which carried the RC:

$$\boldsymbol{HER = PER_T/Block\ size}$$

This calculation is valid for the case of each Window contains one Block. But, if we have different Redundancy Factors (RF) like 2/3, 3/4 and 4/5 then, we must multiply the ***HER*** by the RF as:

$$\text{Total: } \boldsymbol{HER_T = (PER_T/Block\ size).RF}$$

Table 7.3 summarizes the behaviour of each algorithm by comparing between them in terms of overhead and processing delay per each Window or Block. The overhead for the hybrid technique based RC is almost the same overhead of the others or less than them. Also, the max delay time is 4.X is less than RTT value for the clients.

Table 7.3: Overhead comparison: In this comparison N is the packet size and X is the processing time for each Block/Window buffers and X<<RTT

Resynchronization Technique	**The Method Used**	**MD5 16-Bytes /window**	**SHA-1 20-Bytes/window**	**Overhead processing Delay**
SHHC	-Hashes Concatenation (3 hashes) -Hashes XOR	48-Bytes/window 16-Bytes/window	60-Bytes/window 20-Bytes/window	3.X
TSP	One packet/window	N-Bytes	N-Bytes	X
MLHC	2 Layers hash	16-Bytes	20-Bytes	≈2.X
TSS	4 Bytes/packet for timestamp	20-Bytes	24-Bytes	X
RC-SRS	-Redundancy 2/3 -Redundancy 3/4 -Redundancy 4/5	16-Bytes 16-Bytes 16-Bytes	20-Bytes 20-Bytes 20-Bytes	2.X 3.X 4.X

In Figure 7.17, the performance of the hybrid technique based RC in terms of probability of recovery against different packet loss error rates has been illustrated. The simulation has been done using SHA-1 hashing technique. The three redundancy factors used are 2/3, 3/4 and 4/5 (which means combines 2 hashes, 3 hashes or 4 hashes values per static Window) give high probability of chain recovery till error rate 0.2 % for all RF. This gives us good indication for the robustness of our proposal and its high degree of recovery state of chain link.

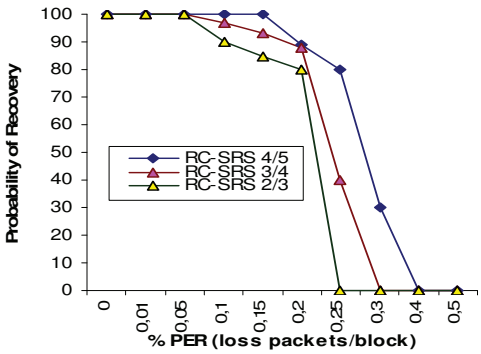


Figure 7. 17: The percentage of recovery state against the packet error rate PER for video rate 1024 kbps with different redundancy factors 2/3, 3/4 and 4/5

7.6.1 Advantages of Small Window or Block Size

As the Window size is varied based on the video rate, the achieving degree of reliability has some how relation with the increasing of Window size. It is best to keep the Window as small as possible to enhance the reliability of transmission. This is very important in transmission over UDP because its nature is unreliable by default. But, for TCP as it is reliable transmission, the impact of increasing Window will not have great effects on the reliability.

Although many previous works simulated the large number of packets per Block, our work benefits by adopting small Window buffers like:

- More reliable with unreliable transmission environment like UDP transport systems
- Quick calculation and verification time
- Small overheads
- The Block with 85 packets seems a small size, but this assumption has a good features on PER or losses till complete Window
- The HER will be controllable under these assumptions

7.6.2 Adaptive Window Size

This work adopts static sliding Window which means that; each Window consists of fixed number of Blocks. The Window size is negotiated between the server and client

according to the video rate and the client capabilities during the session establishment. The client will process the Window for specific number of Blocks then empty it. So, there is no relation between the current Window and the next one else the hash value of the last Block of this Window. So, the receiver only cashes small information from the previous Window which is its digest value to use it as initialization vector for the next one.

As we agreed before from Figure 7.14 that, we build our comparison on Window or Block size 85 packets for video bit rate 1024 Kbps and for a delay time 1 sec, this buffer will be duplicated if we assume 2 sec delay as shown in Figure 7.18. In this figure, there is a comparison between different video bit rates against the elevation effects of Window size. As the rate increased the window size must be increased and vice verse.

But, decreasing the video rate below 512 Kbps will lead to Window size less than 85 packets which by default has negative impact on the delay time augmentation.

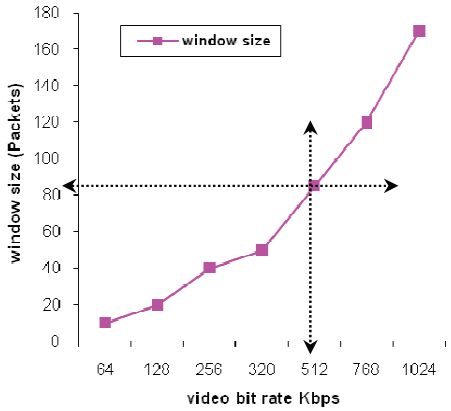


Figure 7. 18: Window size for different video rates against video rates (64K, 128K, 256K, 512K, 768K and 1M)

7.7 Conclusion

This chapter presented a complete study about the hash chain resynchronisation problem and solutions that could be used for its recovery with any kind of data streaming. After that, the comparison between different methods led to a hybrid algorithm which called RC-SRS.

Moreover, the work focused on the resynchronization needs for hash chain mechanisms in video security by using redundancy codes (RCs). Therefore, we

categorized the methods that can be used in hash chain link recovery into four (SHHC, TSP, MLHC and TSP). Then, we proposed a hybrid technique called RC-SRS that inherits from the pros of all previous techniques. The analytical and simulations results for this algorithm assure the reliability and efficiency of it over other algorithms (SHHS, TSP, MLHC and TSS)

In terms of complexity, a comparison has been done between the different ways for achieving the resynchronization of hash chain. This is followed by an evaluation of our proposed method RC-SRS for resynchronisation based on redundancy codes and a study on attack mitigations. Our results indicted that; the RCs will not cause additional computation time for the sender and receivers and the overhead added is accepted in terms of packet size. Moreover, the delay time consumed by the receiver to deduce the hash link based on received RC is less than standard RTT.

As this work is focused mainly in the hashing for achieving the reliability in security measure for video streaming, it still lacks some degree of security. In the next chapter, we will also integrate security signatures and analyse their robustness versus their verification time.

Chapter 8: Window-Based Hash Chain Signature Combines with Redundancy Code

This chapter provides a performance study for securing media streaming based on hash chain methodology. We introduce a new technique that combines the signature of window-based hash chain with redundancy codes for achieving high reliability and robustness against many attacks. Also, the Window technique integrates the Time-Stamped which strongly eliminates the anti-replay attack. It will also control the management of many users accessing the same video in different instant times. The Window-Based algorithm with redundancy code will be compared against Packet-Based or just Block-Based video streaming security ways. The analytical and simulation results indicate that, the Window-Based Hash Chain Signature combines with Redundancy Code (WB & RC) is a good solution for video streaming security in terms of reliability and robustness. The times of signature creation and verification are accepted under the standard delay recommendations of real time applications. Our case study provides YouTube as a successful scenario over Internet. The privacy of YouTube will rely on a secure email in user access which represents an efficient way in mobility issue.

8.1 Introduction

Nowadays, the majority of media sharing servers over Internet face many challenges for securing their media delivery. Several mechanisms exist and may partially solve this problem using Digital Signature and cryptographic hashing algorithms. This work aims to introduce an Efficient Hash Chains (EHC) mechanism to guarantee the media delivery between server and client. This mechanism will assure the Data Authentication (DA) which includes two guarantees: Data Integrity (DI) and Data Origin Authentication (DOA). We have mainly two scenarios for video streaming models; the

first one which occupies a large band of Internet access based on HTTP/TCP packets like accessing YouTube servers and the second one which represents the business operators' part for video streaming delivery like IPTV operators (for examples; Free or Orange boxes in France). The second scenario mainly depends on Real Time Protocol (RTP) delivery over User Datagram Protocol (UDP) by using Session Initiation Protocol (SIP) as a signalling protocol.

In this chapter, we have designed a *Window-Based* hash chain digital signature scheme for securing video streaming either in online or offline broadcasting. As, the majority of hash chain digital signature works for multicast applications and video streaming mainly depend on *Packet Based Signature* (which means that; each packet is signed with its hash digest), we propose a new alternative way that we call *Window Based* combining *Signatures* and *Redundancy Codes*. It depends on signing the digest of a group of packets represented by the *Window* size. The new algorithm will decrease the time of calculations and will reduce the overhead bytes besides its high reliability and good robustness against packet loss.

8.1.1 Digital Signature

The digital signature is a public key cryptography algorithm using two different keys that have a mathematical relation [81]. Those two keys are: the private key used for generating the signature by the signer part and the other key is called the public one used by the verifier or the receiving part to verify the signature.

So, the digital signature involves two processes, one performed by the signer and the other one by the receiver:

Signature creation uses a hash result (Message Digest) derived from and unique to both the signed message and a given private key. To have a secure hash result, there must be only a negligible possibility that the same digital signature could be created by the combination of any other message and private key.

Signature verification is the process of checking the digital signature by reference to the original message and a given public key, thereby determining whether the digital signature was created for that same message using the private key that corresponds to the referenced public key.

8.1.2 Redundancy Codes

Redundancy Codes (RCs) are the additional information that will be added to the original data by the sender to help the receiver recovering the transmission error. They are some codes equivalent to Forward Error Correction (FEC) codes. Those codes will save time for the receiver calculations and reordering data lost or damaged. As the specific nature of video streaming does not allow the retransmissions of packets lost, RC will help the receiver to be self-healing and continue playing the video without interruptions. Also, the Hash Chain with RC will keep the hash link unbroken in case of packet loss as explained in the previous chapter (Chapter 7).

The rest of this chapter is organized as follows: Section 8.2 gives an overview on the related work by comparing packet-based versus block-based signature. Section 8.3 discuses the security case study of YouTube video accessing and its privacy. Section 8.4 illustrates our algorithm architecture and assumption for achieving redundancy of hash chain using the RCs. Also, it highlights the window mechanism challenges against some popular attacks. Section 8.5 shows the results and Section 8.6 concludes this chapter.

8.2 Related Work

Hash chains together with digital signature have been used in many security applications. Especially, real time and some sensors applications are the most interesting ones. The fast and light calculations are the advantages of Hash Chain that attracted many applications to use it in security. This technique of security was successfully introduced in [80] for password authentication security. Then, the adaptability of Hash Chain with signature encouraged its use in multicast traffic authentication [75, 90]. Also, it was used largely in streaming authentication [75], sensors applications [77], RFID authentication [78] and micropayment systems [80] for real time applications. The chaining technique with watermarking has given good results for the authentication of data streaming [86]. The one way function has been found much interesting to design a good construction that matches the real time application. One of the remarkable works in this construction was the work illustrated in [126].

For the hash chain based signature, we can classify the video streaming security based on digital signature into two categories as:

8.2.1 Packet-Based Signature

In this category, each packet must piggyback the digital signature of its content. Although this type provides a perfect security measure for video streaming, it suffers from some drawbacks like large overhead and security calculation or verification times.

One of the most frameworks used in multicast authentication is Timed Efficient Stream Loss-tolerant Authentication (TESLA) [86]. TESLA allows all receivers to check the integrity and authenticate the source of each packet in multicast or broadcast data streams. It requires no trust between receivers, uses low-cost operations per packet at both sender and receiver, can tolerate any level of loss without retransmissions, and requires no per-receiver state at the sender. TESLA can protect receivers against denial of service attacks in certain circumstances. Each receiver must be loosely time-synchronized with the source in order to verify messages; otherwise, receivers do not have to send any messages. TESLA alone cannot support non-repudiation of the data source to third parties. So, it must be combined with other security measure for achieving good security.

Another work [90] handled the video stream authentication. The authors assumed a combination of one-way hash and digital signature to authenticate packets. Their idea explained as follows; for avoiding collision in hash links, the hash of packet P_i is appended to packet P_{i+1} before signing P_{i+1}, then the signature on P_{i+1} guarantees the authenticity of P_i and P_{i+1} at the same time. Also, they proposed some additional packets to resist the packet loss. The hash of each new packet is appended twice: to the packet preceding it and to the packet from the original chain following it. The drawback of that proposal is the large overhead as it increases linearly with the growth numbers of packets.

In another security project [85], the authors tried to secure the video streaming received from satellite link by combining the digital signature with watermarking to achieve authentication and data integrity for mobile applications. They achieved the security by digital signature combined with a pre-computed hash chain and embedded this hash chain inside the video streaming by the means of watermarking technique to all packets.

A time-critical multicast authentication scheme was proposed in [97], which combines hash chains with one time signature to authenticate streaming of packets. The algorithm

provides short end-to-end computational latency, perfect tolerance to packet loss, and strong resistance against malicious attacks. They used long key for achieving high security which leads to large overhead.

8.2.2 Block-Based Signature

In this category, only some random packets from a block which may contain many packets could carry the signature of the whole block information.

In [88], the authors sign video streams depending on their type: offline or online. They chain blocks based on the packets inside the block. Each block carries the hash of the next one (online case). For the offline case, they calculate the hash based on the whole video and the receiver must have some buffer so as to start the verification after a specific length of the video. They signed the hash of each block and only the first packet of each block will carry this signature.

One of the contributions that adopted the principles of block-based is [89]. This work adopted two solutions to secure the video stream and compared between them. The first solution based on TESLA which belongs to packet-based technique. The second scheme, called Efficient Multi-chained Stream Signature (EMSS). This work was based on signing a small number of special packets in a data stream; each packet is linked to a signed packet via multiple hash chains. The authors achieved the link of chain by appending the hash of each packet (including possible appended hashes of previous packets) to a number of subsequent packets. The best choice of parameters which controls that scheme guarantees was the authentication for the arrival packets, even if the transmission was over highly lossy channels. So, the robustness against packet loss was achieved by verifying that each packet contains multiple hashes of previous packets, and furthermore, the final signature packet signs the hash of multiple packets. Their links depend on putting six hashes per packet which represents large overhead. Hence six packets carry the same hash value.

8.3 YouTube Case Study as Video Streaming Internet Model

When using Internet, the majority of streaming are TCP/IP based delivery for firewall security issues. This model represents the scenario of accessing videos over Internet independently of content providers. We will provide a brief study for this model

including the analysis of privacy access, uploading videos and playing videos online or offline.

YouTube provides a good scenario for video streaming which represents 90% of media delivery over the Internet. When we analysed the session of media streaming for some servers like YouTube, we found that all packets were HTTP/TCP so as to facilitate establishing the media sessions over Internet firewalls. Many works handled the YouTube phenomena from the prospection of files hits and huge statistics related to such types of servers. They also discussed the benefits and revenues from YouTube files hosting [26]. That work proposed a good analysis for YouTube statistics and optimized its best file allocation according to numbers of files' hits. The huge statistics of YouTube were the incentive for its proposing to this study.

8.3.1 YouTube Privacy

When a user creates an account through YouTube, he will use his email address and password to access his private part on the server. So, if his name is Bob and he has an email address through Gmail: *Bob@gmail.com* and password: (******) then the server will create the unique YouTube address for him like: ***vlff1x979cp8@m.youtube.com***. This unique YouTube address will be used by the server for all of his privacy to-and-from the server. It will also be used for securing the upload sessions of his videos on the server. The security degree of this account is guaranteed by the server.

The user can use this unique address for all the uploads or at any time, he can request another one by sending request to the server. He can also request one for each uploaded file if he needs this from the server [130].

8.3.2 Security Policy Management

In this part, we will analyse the proposed security mechanism by explaining the procedures and the requirements to secure video streaming uploaded or downloaded by another client. Figure 8.1 explains the phases proposed for assuring the security policy to our videos in case of uploading or viewing. We have three phases to study after the login access phase. The login access uses the YouTube email address as an identifier for each client. After client login success, the security phases will start as following:

Login Access: In this phase, the secure login will be executed according to YouTube privacy that will generate a unique identifier as explained in YouTube Privacy section 8.3.1 above.

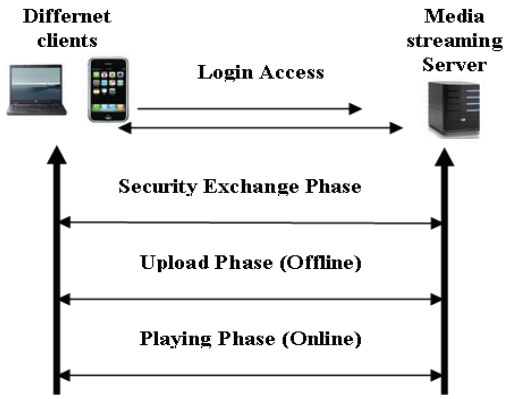


Figure 8. 1: The whole scenario for any client access media streaming server

Security Exchange Phase:

This phase focuses on the key agreement between the server and clients as shown in Figure 8.2. Also, its objective is to generate either a secure (IV) to be used in hash chain or a secure private key that will use to sign the hash value. There are many security algorithms that can handle this process like Diffie-Hellman (DH) [127] or Elliptic Curve Cryptography (ECC) [128]. Figure 8.2 illustrates all the steps needed in this phase. We use an elliptic curve key agreement based on Identity Based Cryptography (IBC) [129]. As each user has a YouTube email address (for example: bob@gmail.com), we will use it to generate a public key. This will lead to a personalized key agreement to access to YouTube services.

In this phase, there are two important steps. The first one is the key agreement to generate a shared key (Ks) and the second one is the generation of client private key K_{priv} by the Private Key Generator (PKG). The two steps need to be realized in a secure manner.

After the login access, an SSL (Secure Sockets Layer) session starts. The server chooses an elliptic curve defined in Galois Field GF(p) where p is a 128 bits prime number. This elliptic curve has A and B like coefficient. It will be defined as E(p,A,B): $y^2 = x^3 + Ax + B \bmod p$. The server chooses a public point P in E and computes the public point $X = S_x.P$ where S_x is the server private key. E(p,A,B), P and X are sent to the client as shown in Figure 8.2.

This latter calculates a point $Y = S_y.P$ where S_y is his private key and, then, sends Y to the server.

The server and the client calculate the point $Z = S_x.Y = S_y.X = S_x.S_y.P$. Z has the form of $Z(z_x,z_y)$ where z_x is the abscissa and z_y is the ordinate. The shared key Ks can be z_x or z_y with a 128 bits length.

After this step, the PKG will generate the client private key K_{priv} using his security parameters and the client YouTube address (as a unique identity). The public PKG security parameters are: E(p,A,B), p and $P_{pub} = s.P$ where s is the PKG private key. There is also a hash function called MapToPoint MTP which convert a simple string into a point in E(p,A,B). Then, the client's public key is MTP (client YouTube address) and $K_{priv} = s.MTP$(client YouTube address).

The server sends the K_{priv} and the Initialisation Vector (IV) encrypted with the shared key Ks.

Now, each client has a secure IV to start his scenario of accessing the server as explained in Chapter 7 for joining procedure of clients (Sections 7.3.5 and 7.3.6).

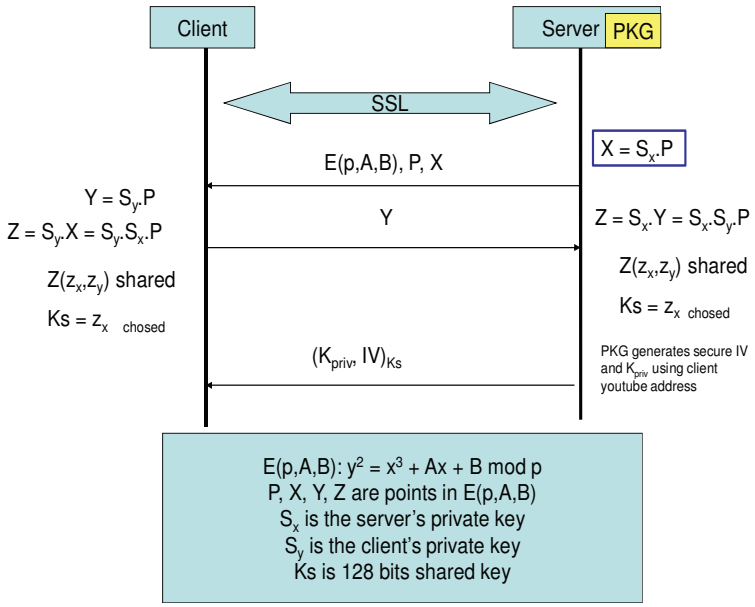


Figure 8. 2: Security exchange phase

Upload Phase (Offline): In this phase, the client must upload his video offline (since no online access is allowed to this video during the upload). We will apply the hash chains mechanism using the security parameters initiated and generated in the first phase. This

case can be considered as Video on Demand (VOD) because the video uploaded will not represent an online session.

Playing Phase (Online): For this phase, the client will upload his video and in parallel, other clients are accessing it. The synchronization in this case must be controlled by the server side so as to assure the delivery of this video or channel to many users at the same time. Also, how the server will treat many clients at the same time with the same security policy for the same video? The different identities for the different clients will not affect the security policy for controlling the hash chains.

8.3.3 The Security Measure for YouTube Videos

We can assume that, each client has a single Client Identity (CI) pre-registered in the server side while creating the user profile. The server uses this identity with some information derived from the uploaded video file in the server (like the File Name Identity (FNI)) to create a unique file identity. Each File Identity (FI) is a very secure unique number that will be saved on the server. This (FI) could be used as an Initialization Vector (IV) for starting hash chains of this file. But, it must have the standard length format for (IV).

So, the file identity FI = f (CI, FNI) where (f) is a hash function. FI will be the (IV) which is used to start the hash chains of this file. This function (f) is secured and predefined between the server and clients. The server is responsible for saving the content from changes or attacks when it is allocated to the server database.

So, each client will have at the end a secure reference record which contains entire client files index and their initial hash values to start the hash chains of this file as shown in Table 8.1. Those metadata for each file describe the security policy parameters for that content of the file.

When we chose YouTube as an example, we find that the Content Management program on YouTube is responsible for creating something very similar to our (FNI) for each uploaded file in this server. And the advantage from that is that YouTube's copyright tool is free, and does not require any commercial partnership with the application [130].

Table 8.1: Reference record for each user on server side

File Index	File#1	File#2	File#3	File#4
Client Identity CI	*CI (unique value for all files)*			
File Name id FNI	*FN#1*	*FN#2*	*FN#3*	*FN#4*
File Initial Value IV	*IV#1*	*IV#2*	*IV#3*	*IV#4*

If any client wants to access this video file, the server will distribute the (CI) and (FNI) of this file to the client. Using the same function which represents the securing part, this client will calculate the (IV) to start the hash of this file. So, the (IV) will be regenerated locally on the client side so as to avoid sending it by the server in case of insecure access. In this case, there is no entry point for man-in-the-middle (MITM) attack to change this stream of data because the attacker could not expect the function used to create (IV) of this file.

The benefit from distributing the (CI) of the file by the server to all clients which want to access this file is to give them the rights to authenticate its owner (the uploader of the file in the server). Moreover, they can authenticate the distributor of the file which is the server in this case. So, we have two data origin authentication: the owner of the video file and its distributor.

The main advantage of using our algorithm to identify the user access is its high efficiency in the mobility issue. As the main access depends on unchangeable parameters like email address and not on changeable ones like IP address or the Subscriber Identity Module (SIM) access card information, so this will be very helpful in the handover process.

8.4 Proposed Redundancy Hash Chain Method

Figure 8.3 illustrates the steps of the applied hash chain based window mechanism and redundancy codes (WB&RC). As shown in the figure, the original stream is divided into chunks composed of some RTP standard packets [98, 123], then assembled to specific Windows (W_i). After that, the hash chain is applied per window. Finally, the RCs are calculated according to the redundancy factor like 2/3 (which means combine two hashes) or 3/4 (which means combine three hashes) or 4/5 (which means combine four hashes) which controls the degree of reliability.

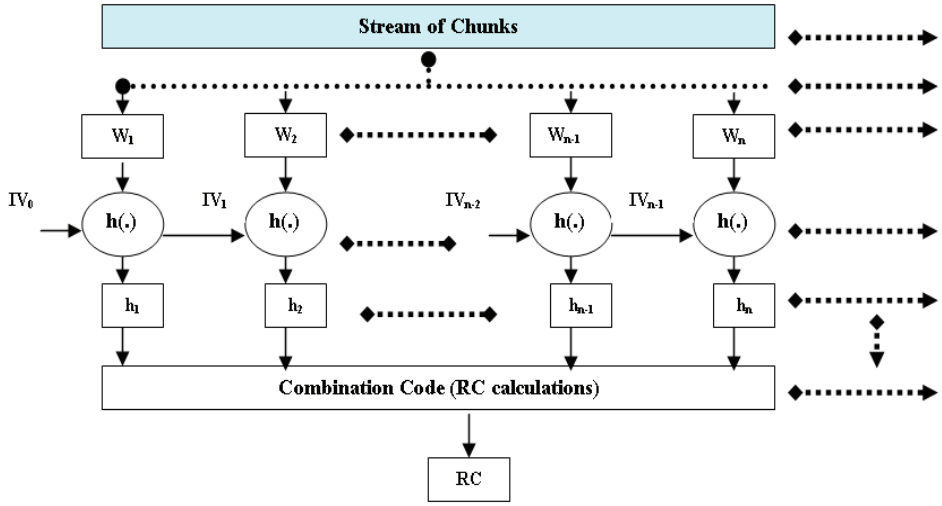


Figure 8. 3: Block diagram for the hash chain architecture based redundancy for video streaming

The choice of piggybacking the RC with any packet inside the window could be randomized if the system could build on pure hashing. But in our proposal, we will integrate signature in the system. So, we preferred to fix the last packet of each *Window* to carry this signature. After finishing the redundancy process according to the capabilities and the permitted overhead to achieve a specific degree of reliability, we have two scenarios of signing the stream:

1. ***Signing the hash value:*** this method follows all previous works in signing video streaming. It depends on: using pre-generated much secured keys by PKG as in Section 8.3.2 then, applying the signature algorithm like RSA [99]. The details of this step will be explained in Section 8.4.1 below.
2. ***Signing the RC value:*** although, the redundancy codes RC size is the same as the digest value size, we will send the RCs in clear mode. This is because the objective of RCs is to achieve the reliability in the hash link of hash chain method while some packets of the stream were lost (as explained in Chapter 7). So, there is no urgent need from signing the RC. This code will be piggybacked in each Window but within the first packet. It will be the supporter for the receivers to track time-stamped for each window.

8.4.1 The Window Signing Sequence

In Figure 8.4, the sequence process applied to generate a signature for any Window (Block of packets from the stream) is illustrated. The process details are:

- The Window ***(W_n)*** is firstly hash chained with ***(IV)*** input which represents the output hash of the previous Window ***(W_{n-1})***.
- This output will be called Message Digest (MD) which has fixed size like 20-Bytes in case of SHA-1 [119] or 16-Bytes in case of MD5 [118] hashing techniques.
- The (MD) will then be subject to Digital Signature process with security algorithm **S(.)** like RSA [99] and security key *Ks* as shown in the Figure 8.4.
- The final output that will be used for signing the window is **$Sig(MD_n)$, MD_n is the Message Digest of** ***(W_n)*** which has output overhead Bytes to be piggybacked in each window according to the RSA key standard length used.

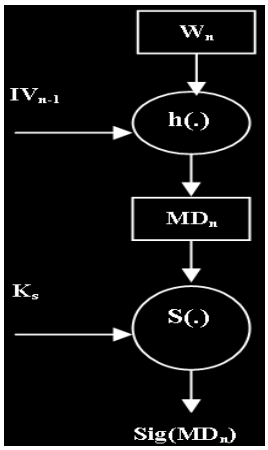


Figure 8. 4: Flow chart to create Digital Signature for any window of the video stream

8.4.2 General Scheme Overview

Any video will be divided into ***(λ)*** Windows and each Window has ***(n)*** packets ***{P_1, P_2, …, P_n}***. So, each ***W_i= {$P_{i,1}$, . . ., $P_{i,n}$}*** where **i=1,...,λ**.

The calculation will depend on the video access mode:

- For offline mode ***(λ)*** is pre-known
- For online mode ***(λ)*** is undefined

Thus, we first calculate the hash for each Window: ***H_i=h(W_i, IV_{i-1})***

Then, we create the signature of this Window:

$$s_i = S(H_i, Ks)$$

Where S is the signature function like RSA and Ks is the signer private key used for signature creation (this is the key generated in Section 8.3.2).

In case of RSA, we compute the standard ciphertext (***C***) corresponding to message (M) by:

$$C = m^e \bmod n$$

Where, $0 < m < n$ and (m) is the integer representation of (M).The public key Ks is (e,n).

In our case m is equal h_i and the signature will be:

$$s_i = h_i^e \bmod n$$

Where *(n)* is used as the modulus for both public and private keys

8.4.3 Signature Verification

The receiver can recover the hash value (h_i) from the ciphertext signature (s_i) by using the signer public key which is the exponent ***(d)*** by the following computation:

$$h_i = s_i^d \bmod n$$

Then, the receiver compares this value with the window calculated hash $h_i(W_i)$ and if they are equal this means that correct window has been received. Given (m), the receiver can recover the original message (M) by reversing the padding scheme.

8.4.4 The Window Verification Sequence

When the receiver assembles the window size, it will verify this window following these steps:

- It will assure that, all packets inside the window have the right index to the location of signature.
- It calculates the hash of this window using the previous window hash value as (IV),

$H_n=h(W_n, IV_{n-1})$

- It verifies the signature to obtain the plain text value which represents the hash of this window.
- It compares this hash with the calculated hash. If they are equal (plain text=H_n) this means that this window is correctly received and not altered or changed in the routing path between the server and the receiver.
- It gives this window to the decoder to complete the processing and play the video

8.4.5 Window Based Time-Stamped Signature

For many users access, managed at different time instant of joining the server stream, we add the Time-Stamped technique to control this process. As we mentioned in Section 8.3.2, in the security policy management, each user must pass the login access phase and key management phase. Then, to access an online stream, the server will give the client the Initialization Vector (IV) of that time instant (t). So, we use IV_t ($0 <= t <= \lambda$), where t_0 indicates that, the client access the file from its beginning and (t_λ) means access in the final *Window*.

8.4.6 Window Overhead

This part focuses on the total overheads of our algorithm either for signature overhead bytes or the time calculation overhead.

The latest RSA recommends key length between 1024 bits and 2048 bits which gives 128-Bytes overhead in the first case and 256-Bytes overhead in the second case as an output signature.

As an example; for MPEG-TS [123] the output chunks are equal 188 Bytes/chunk. The RTP packet size we captured from the packet analyser during the simulation was 1370 Bytes which equal 7(chunks) x188 Bytes + 54 (total headers rest). So, on the base of 1500 Bytes standard packets we still have 1500-(1370+2 bytes for index to the place of RC) = 128 Bytes. Those 128 Bytes give us the probability of sending the RC 8 times in case of MD5 or 6 times in case of SHA-1. Those results were obtained under assumptions of packet size (1500 Bytes) and block size (512 bits). Also, the redundancy codes overhead are different from the signature overhead.

So, in case of using RSA key length 1024 bits, we can piggyback the signature in any packet inside the Window. But, for the key 2048 bits, we can divide this signature into two parts each one 128 Bytes that can be piggybacked to only one packet or send the signature in one special packet which means more overhead. In our work, we chose RSA with key length 1024 bits that means, we need only one packet for piggybacking the signature inside which is the last one in the Window.

8.4.7 Assumptions and Attacks Analysis

In this part of the work, we analyze some problems and assume some attacks that could face our proposal and how it could handle them:

- **Man-In-The-Middle (MITM)**

The security procedure that handles the login access with (PKG) on the server side could prevent such type of attacks. Also, we used (SSL) to secure the way of key distribution between the server and clients.

- **Delay:**

Any packet comes within *Window Number* that is not belonging to will be discarded and this will protect the receiver from any packet delay problems.

- **Robustness:**

The RCs will help the receiver to be Self-Healing so as to grantee finding the hash link in case of some packets lost from the Window. The RCs will help the receiver in extracting the initial value needed to continue the hashing process.

- **Anti-Replay:**

If some attackers alter some packets from stream, modify them, and then resend the packets again to the receivers along the path, the *Time-Stamped Window* will treat this problem by checking the time and discarding those packets. Also, the signature match will judge on the whole window validity and which packet has no right index to the signed packet.

8.5 Results

The results focus on the performance comparison between the different models which are Packet-Based, Block-Based and Window-Based & RC. The analytical and

simulation results by Matlab are based on standard block size 512-Bytes of MD5 [118] or SHA-1 [119] hashing techniques. Assuming that, we have a video file which has (N) packets and is divided into (λ) Blocks or Windows. If we use hash chain method to calculate the hash value which will be used to calculate the signature by the above three methods, the signature overhead and time calculation will be different according to the way of piggybacking this signature to the stream. This overhead will vary according to the type of modulus used for signature length (for example 1024-bits key-length gives an overhead 128-Bytes signature and 2048-bits key-length will give 256-Bytes signature). We assume that, the signature size will be (X-Bytes) for each packet, (N) is the total number of packets per video file and (λ) is the number of Blocks or Windows per video file. Table 8.2 explains the proposed parameters and summarizes the overheads produced by the calculation times for the three models PB, BB and WB&RC.

Table 8.2: Analytical comparison between the three ways of adding the signature based (Packet, Block, and Window & RC).
The (N_{RC} and T_{RC}) are the overhead bytes and time added due to redundancy used RC

	Signature Overhead Bytes	**No of Signed Packets**	**Processing Time (T_p)msec**	**Verification Time (T_v) msec**
Packet-Based (PB)	$N.X$	N	T_p	T_v
Block-Based (BB)	$\lambda.X$	*One*	T_p/λ	T_v/λ
Window-Based & RC (WB&RC)	$\lambda.X + N_{RC}$	*One or More*	$T_p/\lambda + T_{RC}$	$T_v/\lambda + T_{RC}$

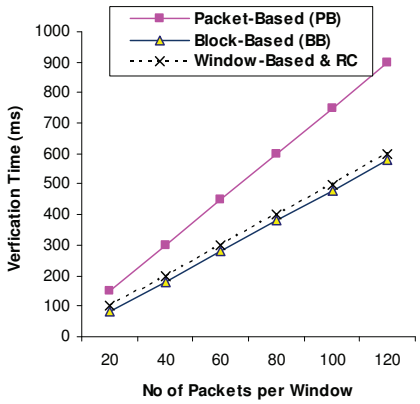


Figure 8. 5: The Signature verification time versus the number of packets per block or window

The analytical and simulation results indicate that, the WB&RC technique not adds remarkable verification time as shown in Figure 8.5. We assume that, the total delay for

our technique not exceed 2 sec which gives us up to 85 packets per window or block as shown in Figure 7.14 *(Chapter 7: The preselected window size for our assumption of no more than 2 sec delay between streaming source and destination which gives acceptable number 85 packets of standard size 1500 Bytes MTU*).

For comparing the three models (PB, BB and WB&RC), we simulate those models based on standard packet size MTU and window buffer of size 85 packets as obtained before. The results as shown in Figure 8.6 indicate that, the processing time decreases as the video rate increases so as to keep the rate of video playing not affected by the security measure processing time.

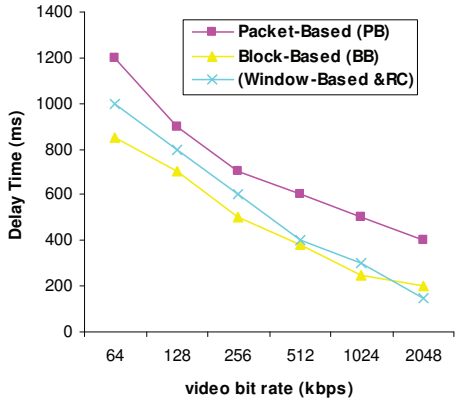


Figure 8. 6: Signature verification time versus video bit rate for preselected window 85 packets

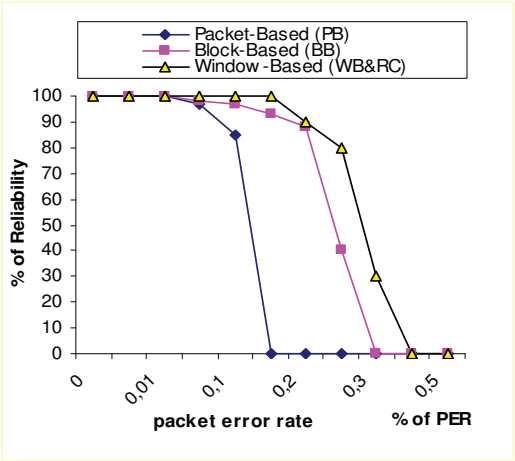


Figure 8. 7: The reliability percentage for the three mechanisms against packet error rate PER

In Figure 8.7, the performance comparison between the three techniques in terms of reliability against different rates of packet shows that the window-based algorithm provides good reliability till 0.25% PER (Packet Error Rate).

8.6 Conclusion

This chapter introduced a new methodology for treating the video streaming security based on hash chain. It combines the window based hash chain signature with redundancy codes so as to obtain high reliable security measure. Our proposed algorithm called *Window-Based* hash chain digital signature with *Redundancy Code* (WB&RC) has high resistance to many video streaming security attacks.

Moreover, a brief case study analysis about YouTube video files accessing and security clients' management were introduced. This case study includes the adding of PKG and distributing keys with ECC mechanism by the server. In the end, the analytical and simulation results for (WB&RC) algorithm indicate minimum time verification that will not affect on the streaming performance.

For comparison, this work considers the video streaming security from the prospective of enhancing the robustness against packet loss and delay time. It also reduces the computation times and the overall bytes overhead. Our work briefly introduces three modes of security based hash chain (Packet-Based, Block-Based and Window-Based combines with Redundancy Codes). The window-based proposal indicates good security measure for video streaming in terms of PER till 0.25%. Also, it provides good robustness and resistance against many attacks like MITM, Delay and Anti-Replay as a result of using *Window Time-Stamped*.

The future directions will focus on the second scenario for business model of IPTV delivery. This scenario will depend on the SIP protocol as a standard signalling for the business model. It will study the clients' authentication wherever they are and give them their rights to access channels or videos they subscribed without the physical restriction of strict STB (Set-Top-Box) location.

Chapter 9: Conclusion and Future Work

It is the time to conclude this thesis and construct the perspectives.

9.1Thesis Conclusions

This chapter concludes our three contributions:

1. Optimization of Video Files Distribution
2. Open IPTV Architectures and Operators Networks Optimization
3. Reliability of using Hash Chain in video streaming security

9.1.1 Optimization of Video Files Distribution

In this contribution, we provided a complete study about video sharing servers through optimizing file allocation. This optimization included two aspects:

- *File Allocation Mechanism*

Through this part, we introduced an optimization algorithm for video file allocation based on the maximum number of hits related to each file and its viewing rates according to file's profile information calculated by YouTube counters. Our algorithm provided simple way for files movements between different zones according to minimum cost of file access. Also, we provided the social behaviours effects and geographical information impacting on the utilization rates/accesses of YouTube files.

- *File Duplication Mechanism*

The second step was to analyze the duplications of video files that could achieve better user satisfaction Quality of Experience (QoE) and high gain for network operators.

9.1.2 Open IPTV Architectures and Operators Networks Optimization

In open IPTV architectures, we proposed a collaborative design for modern business model. Then, this model is validated and analyzed through a Testbed. The benefits from collaboration in design either in CAPEX or OPEX costs are analyzed. Also, the impacts of this design are derived to fit in the nomadic access design of IPTV service delivery.

Moreover, open IPTV requirements and dynamic behaviour of Multicast topologies need to combine new MST networks that take cost and mobility into design consideration. We propose a new root selection algorithm that replaces the old methods. Using known multicast implementations (like ASM or SSM); the multicast sources must be learned in advance via some external methods (e.g. manual configuration). Our dynamic way for changing the sources of multicast enhances the performance of the network. The proposed algorithm depends on introducing Virtual Node (VN) in the original network graph. Then, we recalculate the MST cost for the virtual topology. Through the optimization and simulation results, we get minimum spanning tree cost for sub-trees compared to the original one. Also, we enhance the complexity of the optimization by guiding the algorithm through the non-heuristic test.

9.1.3 Reliability of using Hash Chain in Video Streaming Security

This part studies different methods for achieving resynchronisation state for hash chain links. It also proposes a hybrid algorithm based on Redundancy Codes (RC) and windows flow called *'Redundancy Code Synchronization Recovery State (RC-SRS)'*. This technique merges the pros of all methods (SHHC, TSP, MLHC and TSS) and avoids their weakness. The analytical and simulation results for the hybrid algorithm indicate that, it has a good overall performance in terms of complexity and calculation time compared to other solutions.

In terms of complexity, a comparison has been made between those different ways for achieving the resynchronisation of hash chain. The RC-SRS for resynchronisation based on redundancy codes gives acceptable results which indicate that the RCs will not cause harmful computations time for sender or receiver verifications. Also, the overhead added was acceptable in terms of the standard packet size and MTU 1500 Bytes. The final contribution for RC-SRC is its hybrid features of (SHHS, TSP, MLHC and TSS) as; it is self-healing and depends on the RC for deriving the hash values at the receiver.

It used an indexing mechanism for achieving reliability like TSP. Also, it builds the technique of MLHC for overcoming the collision avoidance and finally the sequences numbers in packets and windows like TSS for preventing the delay and replay attacks.

An added value to our algorithm in the signature integrated to window calculation. Moreover, we classified the integration ways into three methods (PB, BB and WB&RC). Those methods were analytically compared in their overhead and performance. Also, the reliability of our algorithm was high in case of packet error rate above 0.2 % loss.

9.2 Future Directions

In terms of video files distributions, we hope to apply our algorithm on more statistics belonging to YouTube, Dailymotion or any video sharing server over Internet. More and more caching and content management optimization needs to be studied for helping in performance optimization to those types of social networks. Also, it will be feasible to formulate this problem by queuing network model and study the performance of files movement or caching into different places or geographical zones.

For open IPTV and Managed video services, new work needs to be added like building an advanced graphical tool that could help the operators in managing their multicast nodes in the running times. Also, we hope doing some optimizations in the cloud based delivery for IPTV either for enhancing the performance or securing the media. Finally, a constrained MST could be applied with forced number of nodes as roots which will lead to NP-hard problem in the complexity.

As reliability is an issue in security measure based hash chain methodology, this work adopted static sliding Window technique for calculating the RCs and our perspective is to simulate the dynamic use case. This scenario will be built using dynamic sliding Window. We expect that, this scenario will add more robustness besides increasing the degree of reliability and the degree of recovery. Also, we hope integrate our solution in real video streaming system like VLC (server and client) to test its efficacy.

Finally, the future for video service will be the merge between managed with unmanaged creating more service interactivity and security or privacy issues for content over NGN. Beside this, the issue of personalization in multimedia and social networks design can be considered in future work.

Thesis Bibliography

References

[1] FIND Project: http://www.nets-find.net/projects.php

[2] GENI Project: http://www.geni.net/

[3] Orange Box TV: http://www.orange.fr/Livebox-ADSL-TV

[4] Freebox France: http://www.free.fr/adsl /index.html

[5] YouTube: http://www.youtube.com /

[6] Dailymotion: http://www.dailymotion.com /

[7] YouTube Members: http://www.youtube.com/members

[8] YouTube Press: http://www.youtube.com/press_room

[9] Google Trends: http://www.Google.com/trends

[10] Google Analytics: http://www.Google.com/Analytics

[11] WebProNews: http://www.webpronews.com/topnews/2009/03/05/youtube-attracts-100-million-us-viewers.

[12] Google Blog: http://www.google.com/blog

[13] Comscore Press: http://www.comscore.com/press

[14] Alexa: http://www.alexa.com/topsites

[15] P. Gill, M. Arlitt, Z. Li, A. Mahanti, 'YouTube Traffic Characterization: A View From the Edge', ACM Internet Measurement Conference (IMC) San Diego, CA, USA October 2007.

[16] X. Cheng, C. Dale, J. Liu, 'Statistics and Social Network of YouTube'. IWQoS 2008. 16th International Workshop on Videos, Quality of Service, IEEE 2008.

[17] Akamai : http://www.akamai.com/

[18] Yahoo : http://www.yahoo.com/

[19] http://mediatedcultures.net/ksudigg/?p=108

[20] http://www.pcworld.com/article/158949/

[21] F.Duarte, F.Benevenuto, V.Almeida, J.Almeida, 'Geographical Characterization of YouTube: a Latin American View', Web Conference, 2007. LA-WEB 2007. Latin American, Oct. 31 2007-Nov.2 2007 Page(s):13 – 21.

[22] ''YouTube currently streams more than 30 billion videos per month worldwide'': http://www.fierceonlinevideo.com/

[23] http://www.multichannel.com/article/191223YouTube_May_Lose_470_Million_In_2009_Analysts.php

[24] ''20 Hours of Video Uploaded Every Minute'', http://www.youtube.com/blog.

[25] M. Cha, H. Kwak, P. Rodriguez, Y.-Y. Ahn, and S. Moon; "I Tube, You Tube, Everybody Tubes: Analyzing the World's Largest User Generated Content Video System", In Proceedings of ACM-IMC'07, October 24-26, 2007.

[26] E.Abd-Elrahman and H.Afifi, 'Optimization of File Allocation for Video Sharing Servers', NTMS 3rd IEEE International Conference on New Technologies, Mobility and Security (NTMS2009),Cairo-Egypt , 20-23 Dec 2009, pp.1-5.

[27] Nakaniwa, A.; Ohnishi, M.; Ebara, H. and Okada, H.;"File allocation in distributed multimedia information networks," Global Telecommunications Conference, 1998. GLOBECOM 98. The Bridge to Global Integration. IEEE, vol.2, 1998, pp.740-745.

[28] C.Labovitz, S.Iekel-Johnson, D.McPherson, J.Oberheide and F.Jahanian; ''Internet Inter-Domain Traffic'', SIGCOMM'10, 2010, pp.75-86.

[29] L.B. Sofman, B. Krogfoss and A. Agrawal; "Optimal Cache Partitioning in IPTV Network", CNS 2008, pp.79-84.

[30] D. De Vleeschauwer, Z. Avramova, S. Wittevrongel, H. Bruneel ; "Transport Capacity for a Catch-up Television Service"; EuroITV'09, June 3–5, 2009, pp.161-169.

[31] D. De Vleeschauwer and K. Laevens; "Performance of caching algorithms for IPTV on-demand services Transactions on Broadcasting", IEEE TRANSACTIONS ON BROADCASTING, VOL. 55, NO. 2, JUNE 2009, pp.491-501.

[32] D. Applegate, A. Archer and V. Gopalakrishnan; "Optimal Content Placement for a Large-Scale VoD System", ACM CoNEXT 2010, November 30 – December 3 /2010, pp.1-12.

[33] L.B. Sofman, and B. Krogfoss; "Analytical Model for Hierarchical Cache Optimization in IPTV Network," Broadcasting, IEEE Transactions on, vol.55, no.1, March 2009, pp.62-70.

[34] Michael Armbrust, Armando Fox, Rean Griffith, Anthony D. Joseph, Randy Katz, Andy Konwinski, Gunho Lee, David Patterson, Ariel Rabkin, Ion Stoica, Matei Zaharia; 'A View of Cloud Computing', Communications of the ACM Vol. 53 No. 4, Pages 50-58, April 2010.

[35] Draft ETSI TS 182 027 V0.0.9 (2007-04), Telecommunications and Internet converged Services and Protocols for Advanced Networking (TISPAN); IPTV Architecture; IPTV functions supported by the IMS subsystem, ETSI Technical Specification Draft, 2007

[36] Telecommunications and Internet converged Services and Protocols for Advanced Networking TISPAN); Service Layer Requirements to integrate NGN Services and IPTV, ETSI TS 181 016 V3.3.1, July 2009.

[37] Google TV: http://www.google.com/tv/

[38] Digital Video Broadcasting (DVB); ''Transport of MPEG-2 TS Based DVB Services over IP Based Networks'', ETSI TS 102 034 V1.4.1, August 2009.

[39] Digital Video Broadcasting (DVB); ''Carriage of Broadband Content Guide (BCG) Information over Internet Protocol (IP)'', ETSI TS 102 539 V1.3.1, April 2010.

[40] Digital Video Broadcasting (DVB); ''Remote Management and Firmware Update System for DVB IPTV Services (Phase 2)'', ETSI TS 102 824, February 2010.

[41] ITU-T Recommendation Y.1901 (01/2009) – Requirements for the support of IPTV services, clause 3.2.15. Page 4.

[42] TV-Anytime Forum: http://www.tv-anytime.org/

[43] OIPF (Open IPTV Forum): http://www.openiptvforum.org/

[44] Internet Engineering Task Force, 'Protocol Independent Multicast (PIM) Sparse Mode', 2006: http://www.ietf.org/rfc/rfc4601.txt.

[45] Internet Engineering Task Force, 'Protocol Independent Multicast (PIM) Dense Mode', 2005: http://www.ietf.org/rfc/rfc3973.txt.

[46] Wireshark Tools: http://www.wireshark.org/

[47] Video Lan Client VLC, http://www.videolan.org/

[48] LIVE555 Media Server: http://www.live555.com/mediaServer/

[49] Internet Engineering Task Force, 'Real Time Streaming Protocol (RTSP)', 1998: http://www.ietf.org/rfc/rfc2326.txt.

[50] Free Box TV: http://www.free.fr/assistance/2236-freebox-multiposte-executer-vlc-media-player-sur-votre-ordinateur.html

[51] B. Aboba, J. Lu, J. Alsop, J. Ding, W. Wang, "Review of Roaming Implementations", RFC 2194, September 1997: http://tools.ietf.org/html/rfc2194

[52] B. Aboba, J. Vollbrecht, "Proxy Chaining and Policy Implementation in Roaming", RFC 2607, June 1999: http://tools.ietf.org/html/rfc2607

[53] B. Aboba, G. Zorn, "Criteria for Evaluating Roaming Protocols", RFC 2477, January 1999: http://tools.ietf.org/html/rfc2477

[54] Internet Engineering Task Force, 'Internet Group Management Protocol, Version 3', RFC 3376, October 2002: http://www.rfc-editor.org/rfc/rfc3376.txt

[55] John M. Brooke and Michael S. Parkin; ''Enabling scientific collaboration on the Grid''; Original Research Article Future Generation Computer Systems, Volume 26, Issue 3, March 2010, Pages 521-530.

[56] P.Yee Lau, S.Park, J.Yoon and J.Lee;'' Pay-As-You-Use On-Demand Cloud Service: An IPTV Case''; International Conference on Electronics and Information Engineering (ICEIE 2010); pp. V1-272 - V1-276.

[57] UP-TO-US project: https://up-to-us.rd.francetelecom.com/

[58] Python Programming Language: http://python.org/

[59] Parks Associates' Report ''TV Everywhere: Growth, Solutions, and Strategies'', February 2011: http://www.parksassociates.com/report/tv-everywhere-report2011

[60] C. Gui and P. Mohapatra, "Overlay multicast for MANETs using dynamic virtual mesh"; Wireless Networks , Volume 13, Number 1, 2007 , pp77-91.

[61] Technical White Paper: 'High Quality and Resilient IPTV Multicast Architecture', Nokia Siemens Networks (2008).

[62] Internet Engineering Task Force, 'Requirements for Multicast in Layer 3 Provider-Provisioned Virtual', 2007: http://www.ietf.org/rfc/rfc4834.txt

[63] Internet Engineering Task Force, 'An Overview of Source-Specific Multicast (SSM)', 2003: http://www.ietf.org/rfc/rfc3569.txt.

[64] Bang Ye Wu and Kun-Mao Chao, "Spanning Trees and Optimization Problems", (2004), Chapman & Hall/CRC Press, USA.

[65] J. Caja; "Optimization of IPTV Multicast Traffic Transport over Next Generation Metro Networks," Telecommunications Network Strategy and Planning Symposium, 2006. NETWORKS 2006. 12th International, vol., no., pp.1-6, Nov. 2006.

[66] Internet Engineering Task Force, 'Host Extensions for IP Multicasting'; 1989: http://www.ietf.org/rfc/rfc1112.txt.

[67] Extreme Networks White Paper: ` Exploring New Data Center Network Architectures with Multi-Switch Link Aggregation (M-LAG) `, 2011.

[68] AVAYA White Paper,' Network Virtualization using Shortest Path Bridging and IP/SPB', 2011.

[69] IEEE Standard for Local and Metropolitan Area Networks: 'Media Access Control (MAC) Bridges', IEEE802.1D, 2004: http://standards.ieee.org/getieee802/download/802.1D-2004.pdf

[70] J.B. Kruskal, "On the shortest spanning sub-tree of a graph and the traveling salesman problem", Proceedings of the American Mathematical Society, Volume 7, pp. 48-50, 1956.

[71] R.C. Prim, "Shortest connection networks and some generalizations", Bell System Technical Journal, Volume 36, pp. 1389-1401, 1957.

[72] RFC 2401, '' Security Architecture for the Internet Protocol'', Nov. 1998: http://www.ietf.org/rfc/rfc2401.txt

[73] Baugher, M., McGrew, D., Naslund, M., Carrara, E., and K. Norrman, "The Secure Real-time Transport Protocol (SRTP)", RFC 3711, March 2004.

[74] L.Lamport; 'Password authentication with insecure communication'; Communications of the ACM, 24(11), pp.770-772, November 1981.

[75] Heba K. Aslan; 'A hybrid scheme for multicast authentication over lossy networks'; Computers & Security 23, pp.705-713, 2004.

[76] Yacine Challal, Abdelmadjid Bouabdallah and Yoann Hinard; 'RLH: receiver driven layered hash-chaining for multicast data origin authentication'; Computer Communications 28, pp.726–740, 2005.

[77] Adrian Perrig, Robert Szewczyk, Victor Wen, David Culler, J. D. Tygar; 'SPINS: Security Protocols for Sensor Networks', ACM Mobile Computing and Networking, Rome, Italy, 2001.

[78] Syamsuddin, I.; Dillon, T.; Chang, E.; Song Han; 'A Survey of RFID Authentication Protocols Based on Hash-Chain Method'; Third International Conference on Convergence and Hybrid Information Technology ICCIT 08; Vol. 2, 11-13, pp.559 – 564, November 2008.

[79] Huiping Guo , Yingjiu Li , Sushil Jajodia; 'Chaining watermarks for detecting malicious modifications to streaming data'; Information Sciences, pp.281–298, 2007.

[80] Min-Shiang Hwang and Pei-Chen Sung ; 'A Study of Micro-payment Based on One-Way Hash Chain'; International Journal of Network Security, Vol.2, No.2, pp.81–90, March 2006.

[81] Pinkas, et al.; ''Electronic Signature Formats'', RFC 3126, September 2001.

[82] RFC 3226; ''SIP: Session Initiation Protocol'', June 2002: http://www.ietf.org/rfc/rfc3261.txt

[83] Jianguo Hao; Jia Zou; Yiqi Dai;'' A Real-Time Payment Scheme for SIP Service Based on Hash Chain'' E-Business Engineering, 2008. ICEBE '08. IEEE Conference, 22-24 Oct. 2008 Page(s):279 – 286.

[84] Seppo Heikkinen; Book Chapter in (Wired/Wireless Internet Communications), chapter: ''Security and Accounting Enhancements for Roaming in IMS'' Volume 5031/2008, pages 127-138

[85] Chessa, S.; Di Pietro, R.; Ferro, E.; Giunta, G.; Oligeri, G.;" Mobile Application Security for Video Streaming Authentication and Data Integrity Combining Digital Signature and Watermarking Techniques", Vehicular Technology Conference,2007. VTC2007-Spring, IEEE 65th, 22-25 April 2007 Page(s):634 – 638.

[86] A. Perrig, D. Song, R. Canetti, J. D. Tygar and B. Briscoe; 'Timed Efficient Stream Loss-Tolerant Authentication (TESLA): Multicast Source Authentication Transform Introduction', RFC 4082, June 2005.

[87] D. Boneh, G. Durfee, M. Franklin, 'Lower Bounds for Multicast Message Authentication' ;Lecture Notes In Computer Science; Vol. 2045, Proceedings of the International Conference on the Theory and Application of Cryptographic Techniques: Advances in Cryptology, Springer-Verlag London UK ,Pages: 437 - 452, 2001.

[88] Rosario Gennaro and Pankaj Rohatgi; 'How to sign digital streams', In Proceedings of the Advances in Cryptology CRYPTO'97, pp. 180-197, 1997.

[89] A. Perrig, R. Canetti, J. Tygar and D. Song, "Efficient authentication and signing of multicast streams over lossy channels," in Proc. of IEEE Symposium on Security and Privacy, pp. 56-73, 2000.

[90] P. Golle and N. Modadugu, 'Authentication streamed data in the presence of random packet loss', ISOC Network and Distributed System Security Symposium, pp.13-22, 2001.

[91] Jinxin He; Gaochao Xu; Xiaodong Fu; Zhiguo Zhou; Jianhua Jiang; , "LMCM: Layered Multiple Chaining Model for Authenticating Multicast Streams," Software Engineering, Artificial Intelligence, Networking, and Parallel/Distributed Computing, 2008. SNPD '08. Ninth ACIS International Conference on , vol., no., pp.206-211, 6-8 Aug. 2008.

[92] Zhishou Zhang; Apostolopoulos, J.; Qibin Sun; Wee, S. and Wai-Choong Wong; ''Stream authentication based on generalized butterfly graph''; In Proceedings of the IEEE International Conference on Image Processing (ICIP'07), Vol. 6. 2007, pp.121–124.

[93] Zhishou Zhang; Qibin Sun; Wai-Choong Wong; Apostolopoulos, J.; Wee, S.; , "Rate-Distortion-Authentication Optimized Streaming of Authenticated Video," Circuits and Systems for Video Technology, IEEE Transactions on , vol.17, no.5, May 2007, pp.544-557.

[94] Qibin Sun; Zhi Li; Yong Lian; Chang Wen Chen; , "Joint Source-Channel-Authentication Resource Allocation for Multimedia overWireless Networks," Circuits and Systems, 2007. ISCAS 2007. IEEE International Symposium on, 27-30 May 2007, pp.3471-3474.

[95] Yacine Challal, Hatem Bettahar, and Abdelmadjid Bouabdallah; ''Hybrid and Adaptive Hash-Chaining Scheme for Data-Streaming Source Authentication'', HSNMC 2004, LNCS 3079, pp. 1056–1067, 2004.

[96] Yacine Challal, Abdelmadjid Bouabdallah and Yoann Hinard; 'RLH: receiver driven layered hash-chaining for multicast data origin authentication'; Computer Communications 28, pp.726–740, 2005.

[97] Q.Wang, H.Khurana, Y.Huang, K.Nahrstedt, ‘‘Time Valid One-Time Signature for Time-Critical Multicast Data Authentication’’, IEEE INFOCOM 2009, pp.1233-1241.

[98] Y. Kikuchi, T. Nomura, S. Fukunaga,Y. Matsui, H. Kimata; ‘RTP Payload Format for MPEG-4 Audio/Visual Streams’, RFC 3016, November 2000.

[99] R. L. Rivest and A. Shamir and L. M. Adleman, Method for obtaining digital signatures and public-key cryptosystems, Commun. ACM, 21(2), pp. 120-126, 1978.

[100] A. Shamir and Y. Tauman, Improved online/offline signature schemes, CRYPTO’01, pp. 355-367, 2001.

[101] Y. Zhou and Y. Fang, Multicast authentication over lossy channels, MILCOM’07, 2007.

[102] A. Perrig and R. Canetti and D. Song, and J. D. Tygar, Efficient and secure source authentication for multicast, NDSS’01, 2001.

[103] S. Miner and J. Staddon, Graph-based authentication of digital streams, IEEE Security & Privacy, 2001.

[104] P. Golle and N. Modadugu, Authenticating streamed data in the presence of random packet loss, NDSS’01, 2001.

[105] D. Song and D. Zuckerman and J. D. Tygar, Expander graphs for digital stream authentication and robust overlay networks, In Proceedings of IEEE Symposium on Security & Privacy, pp. 258-270, May 2002.

[106] J. M. Park and E. K. P. Chong and H. J. Siegel, Efficient multicast packet authentication using signature amortization, In Proceeding of IEEE Symposium on Security & Privacy, pp. 227-240, 2002.

[107] A. Pannetrat and R. Molva, Efficient multicast packet authentication, NDSS’03, 2003.

[108] A. Lysyanskaya and R. Tamassia and N. Triandopoulos, Multicast Authentication in Fully Adversarial Networks, In Proceeding of IEEE Symposium on Security & Privacy, pp. 241- 255, 2004.

[109] Chris Karlof and Naveen Sastry and Yaping Li and Adrian Perrig and J. D. Tygar, Distillation codes and applications to DoS resistant multicast authentication, NDSS’04, pp. 37-56, 2004.

[110] L. Lamport, Constructing digital signatures from one-way function, Technical Report SRI-CSL-98, SRI International Computer Lab, 1979.

[111] R. C. Merkle, A certified digital signature, CRYPTO’89, 1989.

[112] D. Bleichenbacher and U. M. Maurer, Directed acyclic graphs, one-way functions and digital signatures, CRYPTO’94, pp. 75-82, 1994.

[113] Y.LIU, L.HU, and H.LIU;’ Using an efficient hash chain and delaying function to improve an e-lottery scheme ‘; International Journal of Computer Mathematics; Vol. 84, No. 7, July 2007, pp.967–970.

[114] J. Staddon, S. Miner, M. Franklin, D. Balfanz, M. Malkin and D. Dean, ‘‘Self-healing Key Distribution with Revocation’’, proceedings of IEEE Symposium on Security and Privacy, pp. 224-240, 2002.

[115] National Institute of Standards and Technology (NIST), 2002, FIPS 180–2: ‘Secure hash standard’ (Washington, DC: NIST, US Department of Commerce).

[116] E.Abd-Elrahman, M.Boutabia and H.Afifi, 'Video Streaming Security: Reliable Hash Chain Mechanism Using Redundancy Codes', The ACM 8th International Conference on Advances in Mobile Computing and Multimedia (MOMM 2010) Paris, France, 8-10 Nov 2010, pp.69-76.

[117] E. Abd-Elrahman, M. Abid and H.Afifi, '' Video Streaming Security: Window-Based Hash Chain Signature Combines with Redundancy Code'', The IEEE International Symposium on Multimedia (ISM2010) Taiwan, Dec 2010, pp.33-40.

[118] L.Ronald, R.Rivest. The MD5 message-digest algorithm. Internet Request for Comment RFC 1321, Internet Engineering Task Force, April 1992.

[119] National Institute of Standards and Technology (NIST). Secure Hash Standard (SHA), Federal Information Processing Standards (FIPS) Publication 180-1, May 1993.

[120] H. Krawczyk, M. Bellare, R. Canetti; ''HMAC: Keyed-Hashing for Message Authentication'', Request for Comments: 2104, Feb 1997.

[121] American Bankers Association, Keyed Hash Message Authentication Code, ANSI X9.71, Washington, D.C., 2000.

[122] M.Hefeeda and K.Mokhtarian; ''Authentication Schemes for Multimedia Streams: Quantitative Analysis and Comparison'', ACM Transactions on Multimedia Computing, Communications and Applications, Vol. 6, No. 1, Article 6, February 2010.

[123] M.-J. Montpetit, G. Fairhurst, H. Clausen, B. Collini-Nocker, H. Linder ;'' A Framework for Transmission of IP Datagrams over MPEG-2 Networks'' , RFC 4259, Nov. 2005.

[124] X. Wang and H. Yu, "How to Break MD5 and Other Hash Functions", Proceedings of EuroCrypt 2005, Lecture Notes in Computer Science, Vol. 3494, 2005.

[125] NIST Special Publication (SP) 800-107, Recommendation for Applications Using Approved Hash Algorithms, July 2008.

[126] Yih-Chun Hu, M.Jakobsson and A.Perrig, ''Efficient Constructions for One-Way Hash Chains'', ANCS'05, July 2005.

[127] E. Rescorla, ''Diffie-Hellman Key Agreement Method'', RFC 2631, June 1999.

[128] S. Blake-Wilson, N. Bolyard, V. Gupta, C. Hawk, B. Moeller; 'Elliptic Curve Cryptography (ECC) Cipher Suites for Transport Layer Security (TLS)', RFC 4492, May 2006.

[129] A.Shamir, ''Identity-Based Cryptosystems and Signature Schemes'', 1984.

[130] Privacy at YouTube: http://www.youtube.com/t/privacy_at_youtube.

[131] OpenFlow: http://www.openflow.org/

Appendix A: Open-IPTV

A.1 Open-IPTV Architectural Components

Open IPTV term could be defined as: an integral solution for both managed and unmanaged IPTV services. It could be considered as a kind of TV anywhere/anytime/anydevice but it goes beyond the subscriber domestic home region to include all possible access.

We have mainly two models of IPTV service delivery:

Managed Service: this is the service provided by IPTV service providers based on standard STB and reserved link BW. The contents are mainly generated by the operators but in some situations the clients could generate and diffuse their contents like 'Personalized TV'.

UnManaged Service: this may be called Internet IPTV which has no guarantee for QoS. YouTube is one of the famous representations of this model. The majority of contents in these methods could be generated by the customers themselves.

So, Open IPTV architecture means providing an integral solution for both managed and unmanaged IPTV services in one model. As a result, this will require searching new attaching point with the access network based on software Set-Top-Box (S-STB). Moreover, this kind of S-STB will add more scalability in the access. Video services can either be based on Managed or Unmanaged scenarios.

We can follow the state diagram in Figure 4.3 which explains the steps of accessing video service based on an open IPTV architecture as:

1. The client application regardless of the accessing device (PC, Mobile or TV) initiates a request for the web manager index. This manager is just a repository for indexing different videos, channels or multimedia services. This manager also integrates the functionality of the 'recommendation system' that recommends information items like (movies, TV program/show/episode, video on demand, music, books, news, images, web pages etc.) that are likely to be of interest to the user. Finally, the recommender system gives its decisions based on comparing the user profile with some social reference characteristics.

2. By clicking on a specific choice from the previous manager, a request will pass to the back office manager asking about the client privacy.
3. The answer will either be positive or negative for accessing the client choice based on AAA decision and will forward a notification to index system.
4. The web manager notifies the client application to start accessing the video but after accepting the DRM agreement.
5. The client application starts activating the client DRM engine.
6. The client DRM engine asks the DRM server module about the rights for this video.
7. The DRM server and the back office authorization center agree about this client rights.
8. The back office center gives the DRM server a resume about this client and file security information.
9. The DRM server notifies the client DRM engine the DRM policy and rights for the requesting video.
10. The DRM client engine sends those permissions to the client application to start playing the selected video.
11. The client application starts the access by passing a request to IPTV controller.
12. The Controller redirects this request to storage cloud system who is caching the videos and channels.
13. Finally the cloud management system builds an overlay session for accessing the video based on a specific resource optimization.

A.2 Content Transmission Forms (Managed & Unmanaged Networks)

The IPTV has different forms of transmission that depend mainly on the type of access (Managed or Unmanaged).

- HTTP/TCP (for WEB-TV)
- Unicast/RTP/UDP (for VOD)
- Multicast/UDP (for LIVE-TV)

The first two scenarios could be controlled by using unicast transmission and the last one is controlled using multicast techniques in IPTV business model as the following:

A.2.1 Unicast Transmission

Unicast is point-to-point. It can be an effective way in light applications like e-mail. But, for heavy loaded applications like video streaming, where it will be inefficient if more than one user is streaming. If we have N clients that join the server (Streaming Server SS) then, this server must handle N sessions for the same copy of data packets as shown in Figure 4.4-A. Also, all routers (Multicast Nodes MCN) located near the traffic source will handle large numbers of sessions and increase the burden of those devices. HTTP and RTP/UDP are two different kinds of unicast. Although they are not efficient, half of the Internet traffic is today video streaming coming from the main short video services such as YouTube. That is why; unicast is representing the main mode of transmission over Internet.

A.2.2 Multicast Transmission

It is a kind of Broadcast (BCAST) transmission but it specifies a group of receivers not all like BCAST. As shown in Figure 4.4.B, only one copy of traffic is generated and passed to all devices inside the paths between source and destination. Also, there is no linear increase in traffic as the number of client increases.

For this type of access, we have two basic elements: Management protocol to manage users joining and leaving multicast groups like protocol IGMP [54] and MCAST routing protocols to forward MCAST packets like Protocol Independent Multicast (PIM) which has two popular modes: Sparse Mode PIM-SM [44] for using in light LANs and Dense Mode PIM-DM [45] for using in populated networks.

A.2.3 Validations-Based Testbed

In this section, we focus on validating the open-TV aspects by implementing the three scenarios:

- WEB-TV (using HTTP/TCP as a reliable transmission protocol)
- VOD (using HTTP/TCP as a reliable transmission protocol)
- Live-TV (using RTSP and UDP)

We highlight the important aspects that are needed to be present in a typical prototype. This implementation depends on Android smart phones as clients. Also, all

scenarios are simulated based on unicast mode. We built a simple architecture (like a Testbed shown in Figure A.1) that consists of an integrated local area network of our sub-network inside our campus (Telecom SudParis) as follows:

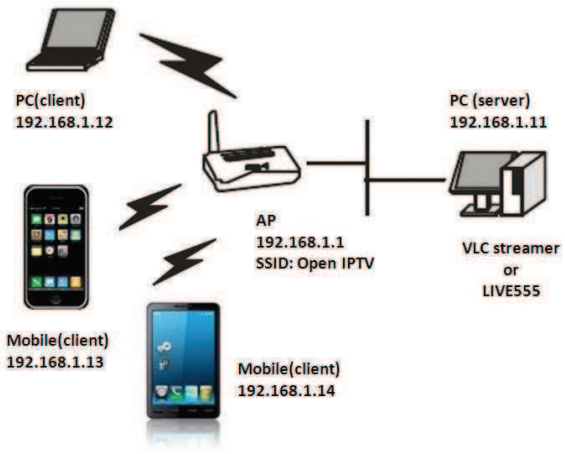


Figure A. 1: Testbed architecture for accessing IPTV

1. One Server to stream either via VLC or LIVE555 under Linux.
2. A client computer using Android emulator or VLC client.
3. An access point to test Mobile IP clients (Wifi (IEEE 802.11) via the access point with an SSID configured as: Open IPTV).
4. Two Mobile clients as: Sony Ericsson Xperia - Android 2.1 and Samsung S Galaxy - Android 2.2 that are interconnected to WLAN by the previous access point Wifi (IEEE 802.11) via the access point SSID preconfigured.

A.2.3.1 WEB-TV and VOD Scenarios

We conducted this test using HTTP/TCP as a reliable transmission protocol that can match transmission over the unreliable networks like Internet. The server side streams the videos via VLC server. Clients use Web browser application built by Android under Linux. One of the advantages of this scenario is its capability to support VOD which permits clients to fast-forward or rewind the video sequence.

We used Wireshark tools [46] to analyze the control messages and observed the following sequence of queries between the client and server as shown in Figure A.2:

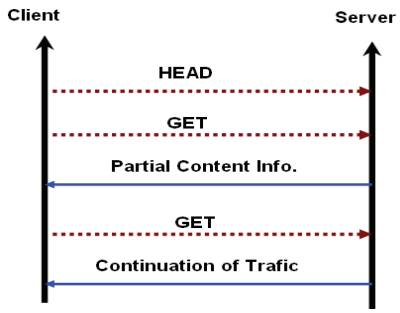


Figure A. 2: Flow control messages in accessing WEB-TV by Mobile nodes under Android

- **HEAD:** Query the header of the media
- **GET:** Query to the media content
- **Partial Content Information:** Response from the server response about the type of media which is (Video MP4)
- **GET(2):** the second request following the translation of the navigation bar on the video to an area not stored in memory buffer
- **Continuation of Traffic:** Responses of video content in continuation

A.2.3.2 Live-TV Scenario

We simulate this scenario by using VLC [47] as client and LIVE555 [48] as a streaming server.

In this part, we used same Testbed in Figure A.1 but with LIVE555 server [48] under Linux as a streamer. The client uses VLC as a client to view the stream in an on-line mode without any options of fast-forward or rewind. Then, we analyzed the sequence of messages between the two parties as the following in Figure A.3. The control part is done using standard Real Time Streaming Protocol (RTSP) commands [49] as follows:

- **3-way handshaking of TCP/RTSP:** there are the 3 packets used normally to open any TCP connection with the sequence control bits SYN/SYN/ACK
- **RTSP options:** it is an RTSP query sent by the client to the server so as to tell him about the client software used (VLC) and asking about the server software as (LIVE555)
- **RTSP reply:** this is the server confirmation for the previous message from the server side to the client side and inside it the server proposed some public options like DESCRIB, OPTIONS, SETUP, TEARDOWN and PLAY

- **RTSP describe:** the client tried to start by describing and accepting the server application used as a streamer
- **RTSP reply with SD:** the server answers using session description protocol all information about the media session and descriptions.
- **RTSP setup:** the client initiated request for session establishment with the server based on the previous descriptions
- **RTSP reply 200 ok:** the server confirms the request of RTSP setup by sending back ok
- **RTSP play:** the client uses the method play to start viewing the video
- **Bundle of UDP RTP packets:** this bundle of packets represent the video packets encoded by MPEG-TS with as each RTP packet contains 7 chunks with 188 Bytes as a Packet Elementary Stream PES for each one.
- **RTSP goodbye:** it is a control report sent by the receiver based on RTCP message to close a connection
- **4-way handshaking closing TCP:** those are the ordinary 4-way handshaking for closing any TCP connection with the sequence control bits ACK/FIN-ACK/FIN-ACK/ACK

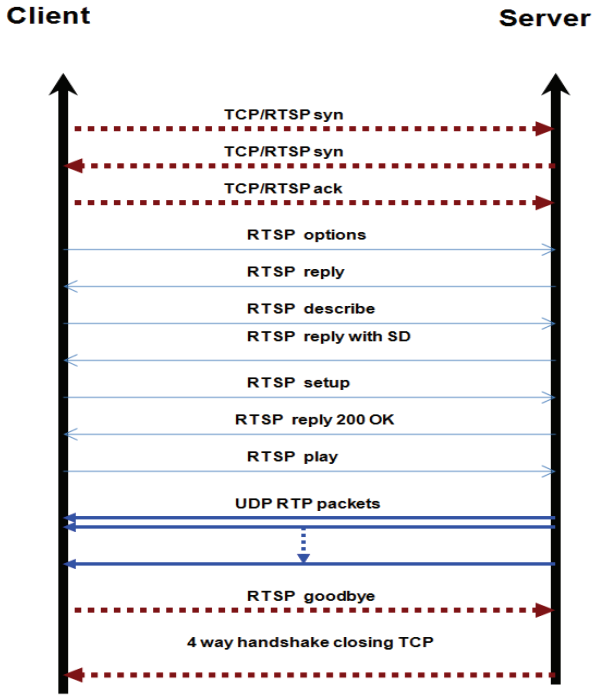


Figure A. 3: Flow control messages in accessing Live-TV by VLC clients under Unicast mode for firewall issues that prevent MCAST

Appendix B: Code for MST-GT

B.1 GUI Optimization Tool

We implemented a complete Graphical Tool (GT) to be used by the operators in their networks design and optimization. This graphical tools is built based on Python libraries [58].

Figure 5.10 illustrates the main construction steps:

- Creating Topology: which means defining the basic topology elements (i.e. number of nodes and links between them
- Calculating MST for original topology: which means running MST algorithm on the pre-created topology
- Adding Virtual Node (VN) to the original topology: which means adding a virtual node to the original topology , this node called VN and has some limited virtual connections to some pre-selected nodes to act as new roots
- Recalculating MST for the virtual topology: which means rerunning the MST on the virtual topology to get the minimum tree cost
- Removing the VN and comparing the MST cost for the output trees with the MST cost of the original tree

Example:

In this example, we create simple topology which has 10 nodes connected in Sparse Graph as shown in Figure B.1. Then, the MST is shown in Figure B.2 for the original network structure. After that, we add VN in Figure B.3 between some proposed nodes (0, 4 and 7) and run the MST again to have the new virtual tree shown in Figure B.4. Finally, we remove the VN as shown in Figure B.5to have the output trees which means two generated trees one with root node-0 and the other with node-7 while node-4 could not act as a root according to the optimization though it is pre-selected.

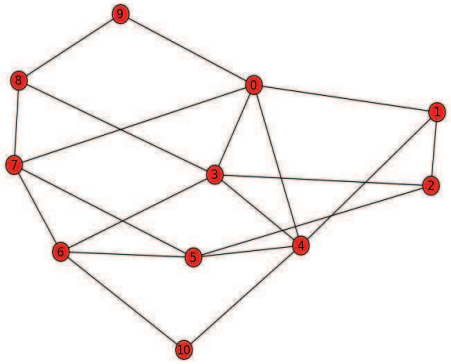


Figure B. 1: Original Topology

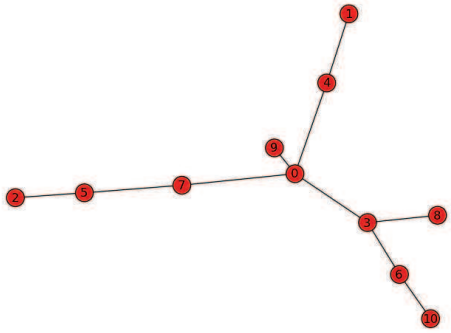


Figure B. 2: MST of original topology based on node-0 as a root

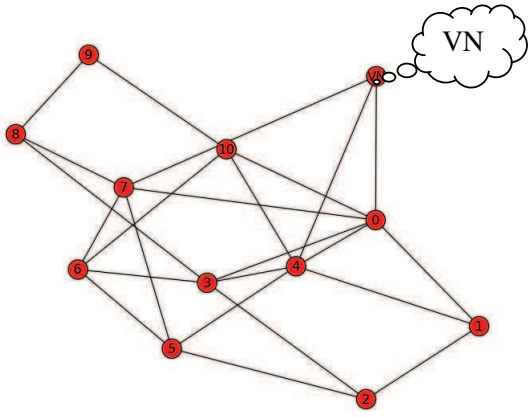


Figure B. 3: Virtual topology by inserting VN linked to node-0, node-4 and nod-7

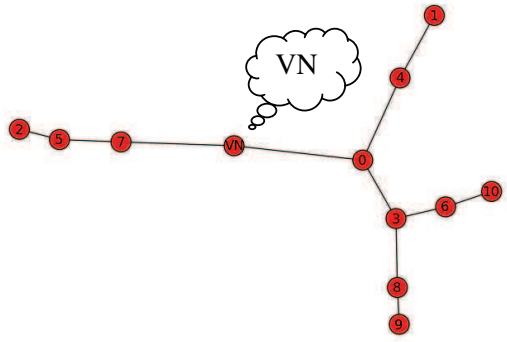


Figure B. 4: MST of virtual topology based on VN as a root

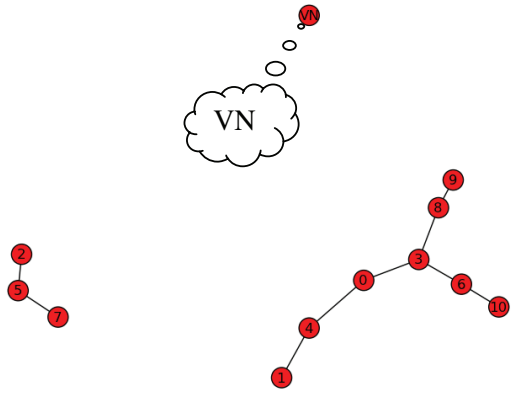


Figure B. 5: Removing the VN connections to obtain the divided trees

B.2 Python Code

 # GUI for MST Optimization Based Virtual Node: code simulates the previous example #

```
from Tkinter import *
root=Tk()
import networkx as nx
import matplotlib.pyplot as plt
f1=Frame(root)

class Draw(Frame):
   #    "classe defination for main topology"#
```

```
    def __init__(self):
        Frame.__init__(self)
        G=nx.cycle_graph(10) # create graph with 10 nodes
        #G=nx.MultiGraph()# for multi graph
        #G=nx.DiGraph() # for directed graph
        G.add_edge(0,4,weight=0.2) # assign weight .2 to edge 0-4
        G.add_edge(0,3,weight=0.5) # assign weight .5 to edge 0-3
        G.add_edge(0,7,weight=0.3) # assign weight .3 to edge 0-7
        G.add_edge(2,5,weight=0.4) # assign weight .4 to edge 2-5
        G.add_edge(3,6,weight=0.6) # assign weight .6 to edge 3-6
        G.add_edge(1,4,weight=0.2) # assign weight .2 to edge 1-4
        G.add_edge(2,3,weight=0.5) # assign weight .5 to edge 2-3
        G.add_edge(5,7,weight=0.3) # assign weight .3 to edge 5-7
        G.add_edge(8,3,weight=0.4) # assign weight .4 to edge 8-3
        G.add_edge(4,10,weight=0.6) # assign weight .6 to edge 4-10
        G.add_edge(10,6,weight=0.5) # assign weight .6 to edge 10-6
        plt.figure(1)
        pos=nx.spring_layout(G)
        nx.draw(G)
        #plt.plot(xrange(10))
        plt.show(G)

class Draw1(Frame):
    #    "classe defination for MST on To original topology"#
    def __init__(self):
        Frame.__init__(self)
        G=nx.cycle_graph(10) # create graph with 10 nodes
        G.add_edge(0,4,weight=0.2) # assign weight .2 to edge 0-4
        G.add_edge(0,3,weight=0.5) # assign weight .5 to edge 0-3
        G.add_edge(0,7,weight=0.3) # assign weight .3 to edge 0-7
        G.add_edge(2,5,weight=0.4) # assign weight .4 to edge 2-5
        G.add_edge(3,6,weight=0.6) # assign weight .6 to edge 3-6
        G.add_edge(1,4,weight=0.2) # assign weight .2 to edge 1-4
        G.add_edge(2,3,weight=0.5) # assign weight .5 to edge 2-3
        G.add_edge(5,7,weight=0.3) # assign weight .3 to edge 5-7
        G.add_edge(8,3,weight=0.4) # assign weight .4 to edge 8-3
        G.add_edge(4,10,weight=0.6) # assign weight .6 to edge 4-10
        G.add_edge(10,6,weight=0.5) # assign weight .5 to edge 10-6
        T=nx.minimum_spanning_tree(G) # calculate MST with node zero as a root
        print(sorted(T.edges(data=True)))
        plt.figure(2)
        pos=nx.spring_layout(T)
        nx.draw(T)
        plt.show(T)

class Draw2(Frame):
    #    "classe defination for inserting 'Virtual Node”#
    def __init__(self):
        Frame.__init__(self)
        G=nx.cycle_graph(11) # insert virtual node 11
        G.add_edge('VN',4,weight=0.2, name = 'Virtual Node') # assign weight 0.2 to edge 11-4
        #G.add_edge(11,3,weight=0.5) # assign weight 0.5 to edge 11-3
        G.add_edge('VN',7,weight=0.3, name = 'Virtual Node') # assign weight 0.3 to edge 11-7
        G.add_edge('VN',0,weight=0.1, name = 'Virtual Node') # assign weight 0.1 to edge 11-0
        G.add_edge(0,4,weight=0.2) # assign weight .2 to edge 0-4
```

```
        G.add_edge(0,3,weight=0.5) # assign weight .5 to edge 0-3
        G.add_edge(0,7,weight=0.3) # assign weight .3 to edge 0-7
        G.add_edge(2,5,weight=0.4) # assign weight .4 to edge 2-5
        G.add_edge(3,6,weight=0.6) # assign weight .6 to edge 3-6
        G.add_edge(1,4,weight=0.2) # assign weight .2 to edge 0-4
        G.add_edge(2,3,weight=0.5) # assign weight .5 to edge 0-3
        G.add_edge(5,7,weight=0.3) # assign weight .3 to edge 0-7
        G.add_edge(8,3,weight=0.4) # assign weight .4 to edge 2-5
        G.add_edge(4,10,weight=0.6) # assign weight .6 to edge 3-6
        G.add_edge(10,6,weight=0.5) # assign weight .6 to edge 3-6
        plt.figure(3)
        pos=nx.spring_layout(G)
        nx.draw(G)
        #plt.plot(xrange(10))
        plt.show()

class Draw3(Frame):
    #    "classe defination for MST on VT virtual topology"
    def __init__(self):
        Frame.__init__(self)
        G=nx.cycle_graph(11) # insert virtual node 11 with name VN
        G.add_edge('VN',4,weight=0.8) # assign weight 0.2 to edge 11-4
        #G.add_edge(11,3,weight=0.5) # assign weight 0.5 to edge 11-3
        G.add_edge('VN',7,weight=0.1) # assign weight 0.3 to edge 11-7
        G.add_edge('VN',0,weight=0.1) # assign weight 0.1 to edge 11-0
        G.add_edge(0,4,weight=0.2) # assign weight .2 to edge 0-4
        G.add_edge(0,3,weight=0.5) # assign weight .5 to edge 0-3
        G.add_edge(0,7,weight=0.3) # assign weight .3 to edge 0-7
        G.add_edge(2,5,weight=0.4) # assign weight .4 to edge 2-5
        G.add_edge(3,6,weight=0.6) # assign weight .6 to edge 3-6
        G.add_edge(1,4,weight=0.2) # assign weight .2 to edge 0-4
        G.add_edge(2,3,weight=0.5) # assign weight .5 to edge 0-3
        G.add_edge(5,7,weight=0.3) # assign weight .3 to edge 0-7
        G.add_edge(8,3,weight=0.4) # assign weight .4 to edge 2-5
        G.add_edge(4,10,weight=0.6) # assign weight .6 to edge 3-6
        G.add_edge(10,6,weight=0.5) # assign weight .6 to edge 3-6
        T1=nx.minimum_spanning_tree(G)
        print(sorted(T1.edges(data=True)))
        plt.figure(4)
        pos=nx.spring_layout(T1)
        nx.draw(T1)
        plt.show()

class Draw4(Frame):
    #    "classe for removing the Virtual Node VN from the output VT virtual tree"#
    def __init__(self):
        Frame.__init__(self)
        G1=nx.cycle_graph(11) # insert virtual node 11
        G1.add_edge('VN',4,weight=0.8) # assign weight 0.2 to edge 11-4
        #G.add_edge(11,3,weight=0.5) # assign weight 0.5 to edge 11-3
        G1.add_edge('VN',7,weight=0.1) # assign weight 0.3 to edge 11-7
        G1.add_edge('VN',0,weight=0.1) # assign weight 0.1 to edge 11-0
        G1.add_edge(0,4,weight=0.2) # assign weight .2 to edge 0-4
        G1.add_edge(0,3,weight=0.5) # assign weight .5 to edge 0-3
        G1.add_edge(0,7,weight=0.3) # assign weight .3 to edge 0-7
```

```
        G1.add_edge(2,5,weight=0.4) # assign weight .4 to edge 2-5
        G1.add_edge(3,6,weight=0.6) # assign weight .6 to edge 3-6
        G1.add_edge(1,4,weight=0.2) # assign weight .2 to edge 0-4
        G1.add_edge(2,3,weight=0.5) # assign weight .5 to edge 0-3
        G1.add_edge(5,7,weight=0.3) # assign weight .3 to edge 0-7
        G1.add_edge(8,3,weight=0.4) # assign weight .4 to edge 2-5
        G1.add_edge(4,10,weight=0.6) # assign weight .6 to edge 3-6
        G1.add_edge(10,6,weight=0.5) # assign weight .6 to edge 3-6
        T1=nx.minimum_spanning_tree(G1)
        #T1.remove_edge(11,4)
        T1.remove_edge('VN',7)
        T1.remove_edge('VN',0)
        #G1.add_edge(11,4,weight=0.0) # assign weight 0.2 to edge 11-4
        #G.add_edge(11,3,weight=0.5) # assign weight 0.5 to edge 11-3
        #G1.add_edge(11,7,weight=0.0) # assign weight 0.3 to edge 11-7
       # G1.add_edge(11,0,weight=0.0) # assign weight 0.1 to edge 11-0
        #T2=nx.minimum_spanning_tree(G1)
        print(sorted(T1.edges(data=True)))
        plt.figure(5)
        pos=nx.spring_layout(T1)
        nx.draw(T1)
        plt.show()

but1=Button(f1,text="Exit",command=root.quit)
but2=Button(f1,text="Create Topology", command=lambda: Draw())
but3=Button(f1,text="Calculate MST", command=lambda: Draw1())
but4=Button(f1,text="Add VN to original Topology", command=lambda: Draw2())
but5=Button(f1,text="Calculate MST for Virtual Topology", command=lambda: Draw3())
but6=Button(f1,text="Remove VN",command=lambda: Draw4())
lab= Label(root,text="Operator X topology",fg="blue",bg="white")
but1.pack(side="left")
but2.pack(side="left")
but3.pack(side="left")
but4.pack(side="left")
but5.pack(side="left")
but6.pack()
f1.pack()
lab.pack()
root.mainloop()
```

Appendix C: Résumé de Thèse

C.1Résume

L'objectif de cette thèse est de fournir des algorithmes d'optimisation pour l'accès aux services vidéo qu'ils soient non-gérés (Internet TV) ou gérés (IPTV). Nous étudions des statistiques récentes concernant les services vidéo non-gérés comme YouTube et nous proposons des techniques d'optimisation appropriées qui pourraient améliorer l'accès aux fichiers vidéos et réduire le coût de cet accès. En outre, l'analyse des coûts joue un rôle important dans les décisions qui concernent la mise en cache des fichiers vidéos et celles liées au choix des périodes temporelles d'hébergement de ces fichiers sur les serveurs.

L'analyse des statistiques des serveurs attire notre attention sur l'influence de la culture du consommateur et ses relations sociales sur les emplacements du contenu qu'il demande. Cela pourrait aider à concevoir des algorithmes de gestion et d'optimisation qui permettraient la construction des réseaux sociaux optimisés. De plus, cela rapporterait des gains plus élevés aux fournisseurs de services vidéo en maximisant les profits et en minimisant les coûts d'accès.

En ce qui concerne les services vidéo gérés appelés IPTV, nous avons mené des expériences sur une architecture ouverte IPTV-collaboration entre différents opérateurs. Ce modèle est analysé selon un critère de coût d'investissement et d'exploitation à l'intérieur de la sphère domestique. En outre, nous avons introduit une solution d'optimisation dynamique de l'arbre « minimum spanning tree » (MST) pour le service IPTV multicast. Lors d'un accès nomade, les arbres statiques pourraient être incapables de fournir le service de manière efficace vu que l'utilisation de la bande passante augmente aux côté des points de streaming (racines de la topologie). Dans notre solution, nous choisissons certains nœuds (les futures racines virtuelles) en nous basant sur leurs positions, pour qu'ils soient des points de streaming et nous optimisons leurs liens avec un nœud virtuel donné (VN). Puis, nous appliquons le MST sur la topologie virtuelle pour calculer les coûts minimaux pour les nouvelles racines. Enfin, nous comparons la complexité de l'algorithme MST basée sur cette topologie virtuelle dans les cas heuristique et non-heuristique.

Nous étudions des mesures de sécurité fiables en streaming vidéo basées sur la méthodologie de la chaîne de hachage et nous proposons un nouvel algorithme hybride. Nous effectuons des comparaisons entre les différentes manières utilisées dans la réalisation de la fiabilité des chaînes de hachage basées sur les classifications génériques. En termes de complexité des *overheads* et taux de perte de paquets, nous utilisons des méthodes d'analyse et de simulation pour comparer notre technique hybride proposée appelée « Chaîne de hachage de fenêtre avec codes de redondance » avec d'autres techniques. Notre mécanisme permet aux récepteurs de flux IPTV de récupérer une certaine quantité de données en cas de perte de paquets afin de ne pas rompre les liens de la chaîne de hachage.

En outre, afin d'assurer un degré élevé de sécurité grâce à la fiabilité de la chaîne de hachage, nous intégrons le mécanisme de la signature basée sur la fenêtre avec les codes de redondance. Cette façon est comparée à de simples méthodes à base de paquets (*Packet-Based*) et à base de blocs (*Block-Based*). Notre algorithme (à base de fenêtres *Window-Based*) donne un degré élevé de fiabilité et nécessite moins d'octets d'*overhead* comparé aux autres méthodes.

En conclusion, nous pensons que, dans le futur, les différentes contributions proposées dans cette thèse (pour les deux services ; gérés (IPTV) et non-gérés vidéo (YouTube)) permettront d'améliorer la personnalisation du service et la fiabilité de l'accès vidéo et la conception des algorithmes de services par les entreprises. De plus, ils pourraient faciliter l'évolution du multimédia de nouvelle génération en un service de vidéo hybride (c'est à dire la fusion des services gérés et non-gérés).

C.2 Introduction Générale

Le Multimédia de futur (Future MultiMedia) ou les réseaux multimédias de prochaine génération sontt l'un des principaux axes de recherche de nos jours. En effet, ce domaine attire de plus en plus l'attention des chercheurs, des fournisseurs des services internet ainsi que les opérateurs et les experts. Par conséquence, plusieurs projets de recherche ont été amorcés dans les quatre coins du monde pour constituer une base pour le Multimédia de futur. Parmi ces projets nous citons les projets FIND (*Future Internet Design*) [1] et GENI (*Global Environment for Network Innovations*) [2] qui ont essayé de définir des nouvelles architectures réseaux et les modifications à y apporter au afin de faciliter la migration vers les services basés en entier sur des réseaux IP tel que la TV.

En outre, Le concept de l'Internet du futur, qui introduit la notion des réseaux ubiquitaires relatifs aux utilisateurs ainsi que pour les services, est en relation forte avec deux problématiques importantes: les services IP multimédias et la gestion des réseaux sociaux. Par conséquent, beaucoup de défis s'imposent dans ce domaine de recherche à savoir l'optimisation des ressources du réseau et des services, l'adaptation des médias et des mesures de sécurité aux applications temps réel.

Le Next Generation Network MultiMedia (NGNM) est un réseau basé sur paquets permettant la mise en place de certains services tels que les services de télécommunication, les services de données (DS) et les services multimédia. En outre, il est capable de faire usage de multiples larges bandes. Il permet également d'offrir des services de QoS technologies de transport (*Quality of Service*) QoS où le service des fonctions liées est indépendant de sous-jacents liés aux transports des technologies. Il offre un accès illimité par les utilisateurs aux fournisseurs de services différents et il prend en charge la mobilité généralisée qui permet la fourniture cohérente et partout de service aux utilisateurs. L'idée générale derrière NGN est que toutes les informations sont transmises via des paquets, comme sur Internet; les paquets sont étiquetés en fonction de leur type (données, vidéo, et voix) et traités différemment pour la qualité de service (QoS) et des raisons de sécurité par des équipements de gestion du trafic. Dans un réseau NGN, il y a une séparation plus définie entre le transport (connectivité) partie du réseau et les services qui s'exécutent au-dessus de ce transport.

Donc, on pourrait conclure que, le NGN est un aspect hybride pour la livraison de vidéo qui permet l'accès aux vidéos d'aucune façon. En outre, avec ce type de réseau de livraison, la vidéo et de contenu (*Content Delivery Network* CDN) les problèmes sont résolus ensemble comme les adaptations vidéo, résolutions, de codage et de QoS.

C.3 Énoncé du Problème

La gestion de la distribution du contenu vidéo à travers l'allocation des fichiers ou la mise en cache dans les réseaux de distribution de contenu tout en assurant une certaine garantie au niveau de la fiabilité des mesures de sécurité constitue un challenge dans la prestation de services vidéo. En effet, un algorithme approprié à ce type de problème est souvent dépendant du type de service ou de l'environnement de l'accès. Par exemple, le service fourni par YouTube est différent de celui des services fournis par Orange (opérateur français d'IPTV). En conséquence, nous étudions des algorithmes et des techniques appropriés à chaque type de service. Pour YouTube, les fichiers chargés à

partir de zones différentes peuvent fournir des résultats importants sur l'accès à partir d'autres zones éloignées pour la télécharger. De ce fait, la nécessité de redistribuer les vidéos téléchargées sur YouTube par des techniques d'optimisation qui tiennent compte de la répartition efficace et le coût de l'accès entre les zones. En outre, la performance de la livraison pourrait être renforcée par l'imposition de certaines façons de duplication des fichiers dans les différentes zones, plutôt que de se concentrer sur celles des plus adaptées à son hébergement. Mais, pour le service géré comme celui d'Orange, l'aspect de la mise en cache des vidéos pourrait être optimisé par l'arbre de recouvrement minimal pour l'optimisation des coûts dans les services de multidiffusion comme modèle d'affaires IPTV. Dans ce cas, une technique dynamique est nécessaire pour reconstruire les arbres dynamiques de multidiffusion sur la base des seuils de coûts pour les opérateurs CAPEX ou OPEX.

Après avoir résolu l'optimisation de la prestation des services dans les précédents types de vidéo, le service non géré souffre principalement de certaine non-fiabilité en cas de mesures de sécurité appliquées comme des signatures ou des chaînes de hachage. Afin d'atteindre un certain degré de fiabilité dans les chaînes de hachage utilisées dans le streaming vidéo et les applications multicast, nous appliquons certains codes de redondance pour être jumelé dans des paquets en temps réel. Ces codes pourraient aider les récepteurs vidéo de suivi des liens dans la chaîne de hachage et continuer à profiter du flux, malgré la survenance d'une certaine perte de paquet.

C.4 Analyses des Contributions de la Thèse

Dans cette partie, nous analysons nos trois contributions en se basant sur les objectifs et les résultats obtenus.

C.4.1 Optimisation de la Distribution des Fichiers Vidéo

Dans cette partie, nous nous sommes concentrés sur la prestation des services vidéo non géré en prenant comme exemple YouTube. Chaque jour, le nombre de vidéos téléchargées depuis YouTube est de plus en plus important ce qui entraîne une détérioration au niveau des performances des serveurs fournissant un tel service surtout avec l'utilisation de la qualité de service *Best Effort* au niveau réseau IP. Ceci dit, nous proposons à travers cette contribution différentes techniques d'optimisation de la distribution des fichiers vidéo entre des différentes zones. Cette contribution se divise en deux parties :

C4.1.1 Algorithme d'Allocation des Fichiers Vidéo

Cette partie fournit une brève étude sur certains serveurs de partage de vidéo comme 'YouTube'. Nous enchaînons ensuite avec une analyse de quelques statistiques relatives aux nombres de personnes visionnant les vidéos et leurs distributions géographiques. En outre, une optimisation de l'allocation de fichiers est introduite et évaluée à l'aide de traces recueillies à partir de YouTube. Enfin, une analyse de certains coûts d'hébergement de fichiers vidéo est menée avec une étude de l'impact des relations sociales sur les vidéos regardées sur YouTube.

Nous supposons tout au long de notre proposition que nous avons un certain nombre de zones (i de 1 à N) et un certain nombre de serveurs (j de 1 à M). Notre objectif est d'avoir un coût minimum (en termes de nombre de visites 'hits' et de coût de distance entre les zones) afin de réduire le nombre de hits sur les serveurs principaux. Nous essayons donc d'avoir une formule de distribution qui optimise les emplacements de contenu sur les serveurs dans des zones différentes en fonction du coût minimum entre les zones et les serveurs (les étapes de l'algorithme d'optimisation sont présentées dans le Tableau C.1).

En supposant que D_{ij} est le coût entre le serveur (j) et la zone (i) où i est de 1 à N (max. nombre de zones) et j de 1 à M (max. nombre de serveurs), le coût total sera:

$$(C_i = \sum D_{ij} * H_{ij})$$

Où H_{ij} est le nombre de hits pour un fichier provenant de la zone i au serveur j, nous cherchons:

$Min\ (C_i = \sum D_{ij} * H_{ij})$ **;** qui donne la meilleure allocation de fichiers.

Tableau C.1: Algorithme d'Allocation

```
Input N                 //number of zones
Input M                 //number of servers
Input h_ij              //number of hits from zone i to server j for a specific file
Input d_ij              //the assumed cost between zone i and server j
        Total cost C_i = ∑ h_ij * d_ij          // where i from 1 to N and j from 1 to M
                If C_i < C_i+1,...,C_N
                        Then allocate this file in zone i
                        Else move this to min (C_i+1,...,C_N) value location
                End If
Return the best location for this file          //best j
End
```

C4.1.2 Algorithme de la Reproduction de Fichiers Vidéo

Cette partie représente une étape postérieure à l'optimisation des meilleurs emplacements décrite dans le chapitre précédent. En effet, nous effectuons dans ce chapitre une autre optimisation pour les mêmes fichiers en dupliquant les fichiers dans

les différentes zones en se basant sur le niveau de la qualité d'expérience des utilisateurs et les seuils de coût de la mise en cache imposés par les opérateurs. Deux techniques de duplication de fichier sont testées et analysées en termes de coûts ou de gains pour les opérations de mise en cache des fichiers vidéo (*caching*) ou de la récupération du fichier (*retrieving*).

Afin de garantir une QoE donnée, nous choisissons de donner l'accès au dossier populaire (situé dans les serveurs de la zone j et ayant un nombre de hits supérieure à une valeur donnée Y) pour un consommateur donné (à partir de la zone i) que si ce consommateur peut accéder à ce fichier avec un a_{ij} coût inférieur à une valeur donnée **A** **(a_{ij} = <A).** Si c'est le cas, nous vérifions la demande sur ce même fichier à partir de la zone i du client (ce qui est $\mathbf{Y_{ij}^{F}}$). Si la demande est supérieure à un seuil **Y**, on duplique le fichier **F**. Par conséquent, tout consommateur peut accéder à tout contenu avec une bonne qualité et un coût minimal. Les étapes de l'algorithme de duplication des fichiers sont présentées dans le Tableau C.2.

Ainsi, nous faisons un compromis entre le coût d'accès et de coût de stockage. En fait, plus la valeur de **Y** est importante, plus le coût de l'accès est grand et la nécessité d'une capacité de stockage est faible. La même observation est relative à la valeur de **A**. Les paramètres **A** et **Y** sont des valeurs seuils ajustées par les opérateurs (*Content Delivery Network*).

Maintenant, nous allons expérimenter deux fonctions de duplication différentes:

• **duplicate1** f(f, i, j) du fichier f des doublons (situé dans la zone j) dans la zone i ce qui signifie (*cache*)

• **duplicate2** f(f, i, j) du fichier f doublons dans la zone de k qui a le deuxième plus grand nombre de hits sur ce fichier (situé dans la zone j) et qui accorde un coût d'accès inférieur à la **A** en même temps, la zone j étant la zone avec le plus grand nombre premier succès (le meilleur emplacement). Cela signifie (*retrieve*).

Pour tester notre algorithme, nous avons appliqué les deux fonctions duplication sur certains fichiers vidéo prises à partir de YouTube. Ces fichiers sont analysés par leurs profils afin d'obtenir leurs distributions de hits.

Tableau C.2: Algorithme de Reproduction

```
Input N          // number of zones
Input A          // maximum access cost, set by the network operator
Input Y          // minimum number of hits allowing file duplication,
                 //set by the network operator

Input a_ij       // access cost to a file located in zone j from a customer located in zone i
Input p_j        // number of files in zone j
Input y_ij^f     // number of hits on file f (located in zone j) from zone i users

For i from 1 to N                                        // user's zone
      For j from 1 to N                                  //file's zone
            If a_ij > A                                  // we suppose that a_ii=0
                  Then For f from 1 to p_j
                        If y_ij^f > Y
                              Then duplicate(f,i,j)      // duplicates file f
                                                         //in the best location
                        End If
            End If
End
```

C4.2 Architectures Ouvertes d'IPTV et l'Optimisation des Réseaux des Opérateurs

Cette partie se concentre sur les services gérés 'IPTV' et ses principaux éléments. Nous proposons une solution de gestion ouverte pour l'IPTV qui peut fonctionner de manière gérés et non gérés. En plus, nous considérons l'optimisation de *minimum spanning tree* (MST) comme suit:

Le premier chapitre illustre quelques terminologies communes à l'IPTV et des définitions pour les nouveaux aspects tels que TVA, Pay-TV, TVE et d'autres. Il présente également une implémentation vue comme un banc d'essai simple (TestBed) pour la livraison unicast en direct-TV ou VOD. Nous listons également les architectures proposées pour la conception collaborative pour les opérateurs IPTV. Enfin, ce chapitre met en évidence l'analyse des coûts pour notre modèle de collaboration en termes de coûts CAPEX et OPEX.

Dans la deuxième partie, puisque le service géré IPTV est principalement délivré par les opérateurs de Set Top Box (STB) basé sur un service de multidiffusion '*Multicast*', les topologies de l'opérateur devrait être soumises à des calculs de *Minimum Spanning Tree* (MST) pour choisir les chemins de coût minimum pour les points de streaming. Par conséquent, la croissance de cette topologie aura une incidence sur la performance globale des arbres multicast de sortie. Ainsi, ce chapitre fournit une solution dynamique pour cette problématique qui dépend de la division de l'arbre original dans différents arbres et de la duplication des points de streaming (les sources de multicast) dans certains nœuds recommandées en fonction de certains seuils calculés et fixés par les opérateurs. Différents scénarios sont simulés en variant le nombre de nœuds dans

Matlab. Enfin, la complexité des MST est calculée pour des techniques heuristiques et non-heuristiques pour le parcours du spanning tree.

Le Multicast est la base de toute solution pour la télévision en direct dans tous les systèmes de IPTV gérés, les opérateurs sont à la recherche de solutions dynamiques afin de faire face à l'évolution du nombre de clients joignant un nœud multicast. La problématique en «nomadisme» est la garantie de la qualité vidéo et le comportement dynamique de la topologie multicast en dehors du réseau domestique de l'utilisateur. De ca fait, nous devons faire la distinction entre le réseau domestique et le réseau visité dans l'IPTV comme indiqué ci après:

Réseau Mère (d'origine): (HN) réfère au réseau domestique local de l'utilisateur, c'est à dire le réseau où il réside avec sa STB. Ce décodeur est installé par un opérateur qui signifie «endroit restreint à l'intérieur de la maison» par une connexion ADSL.

Réseau Visité/Réseau Nomade: (VN ou NN): Il s'agit de n'importe quel réseau d'accès autre que le réseau domestique de l'utilisateur. Les opérateurs peuvent l'utiliser pour donner le service à des utilisateurs nomades. Ainsi, il pourrait être le même réseau de l'opérateur ou des réseaux servis par d'autres opérateurs.

C4.2.1 Minimum Spanning Tree

Supposons que le réseau est représenté par un type de graphe G (*directed ou undirected*). Ce graphe est représenté par G = (V, E) où, V indique les nœuds et E représente les arêtes. La fonction poids w attribue un poids w(e) pour chaque arête e. Pour notre optimisation, nous supposons que les poids sont des nombres réels entre {0, 1}. L'objectif de MST est de trouver l'arbre couvrant minimal de G, qui est, l'arbre couvrant T minimisant la fonction suivante:

$$W(T) = \Sigma w(e) \text{ lorsque } e \varepsilon T$$

La technique de multidiffusion (MCAST) est adoptée pour les réseaux gérés afin de fournir un service IPTV. L'avantage de MCAST avec Unicast est le gain en termes de quantité de bande passante (BW) ainsi que le calcul effectué le long des chemins d'accès réseau pour atteindre le nœud source. Les techniques conventionnelles de MCAST pour des applications multicast se trouvent face de plusieurs difficultés dans les réseaux dynamiques. La dynamicité ici signifie la variation du nombre de clients joignant l'arbre MCAST pour obtenir le service de vidéo selon l'une des situations des clients. En outre, l'augmentation la capacité des nœuds engendre une augmentation dans les investissements des opérateurs dans les routes multicast et ainsi que des capacités de traitement plus importantes. La topologie de l'arbre MCAST peut être utilisée pour la

création des nœuds virtuels capables de partitionner l'arbre original en différents arbres avec des coûts minimaux qui permettent d'atteindre les nœuds finaux (feuilles de l'arbre). Par conséquent, l'objectif principal est de proposer des nœuds additionnels capables d'agir comme de nouvelles sources de diffusion en continu (streaming) en se basant sur les capacités des nœuds. Ainsi, l'hypothèse du nœud virtuel consiste finalement à insérer un nœud dans un arbre, entre certains nœuds, mais en respectant certaines conditions.

C4.2.2 Analyse des Coûts des Liens Virtuels

Dans cette section, nous allons varier les coûts sur les liens virtuels entre le nœud virtuel et les nouvelles racines sélectionnées dans le but de prouver que toutes les simulations donnent les coûts MST pour les arbres virtuels inférieurs que celui d'origine. En outre, cela signifie que la somme des coûts MST relatifs aux sous-arbres générés sera inférieure ou égal au coût total MST relatif à l'arbre original :

$$\sum MSTCost(Sub-Trees) \leq MSTCost(OriginalTree)$$

En effet, les coûts proposés sur les liens virtuels dépendent de l'état du réseau. Cet état a un aspect dynamique qui signifie que les variations des coûts ne pouvaient pas être un processus déterministe mais, il pourrait être considéré comme un processus stochastique. En outre, les prédictions des différents états du réseau ainsi que les capacités des nœuds peuvent être modifiées à tout moment en fonction des actions des utilisateurs en joignant (quittant) le réseau ou aussi les estimations des coûts. Par conséquent, nous considérons différents scénarios de topologies et nous assignons différents coûts sur les liaisons virtuelles. Ces coûts varient entre 0 et 0,9 sur des bases égales et non égales. La base de l'égalité considère que les coûts sont égaux puisque les ressources du nœud virtuel sont divisées à parts égales entre les nouvelles racines. Mais, en cas de non-égalité de base, nous supposons différents coûts en se basant sur le fait que les ressources virtuelles ne sont pas égales et chacune des nouvelles racines possède une base de ressources différente des autres ce qui signifie des coûts différents. Dans les deux figures suivantes Figure C.1, nous simulons les deux scénarios avec un nombre différent de nœuds. Les deux scénarios donnent tous des coûts MST inférieurs ou égal à celui d'origine comme le montre la Figure C.1 pour 16 et 25 nœuds plus un seul nœud virtuel. Dans cette figure le coût MST est représenté par le point zéro ce qui signifie l'absence de VN. Après avoir ajouté le VN avec des différents coûts de liens, nous obtenons un coût MST égal ou inférieur à celui d'origine dans le cas des coûts égaux ou

différents. Cela confirme l'équation précédente : les sous-arbres générés donnent un coût MST inférieur ou égal à celui d'origine.

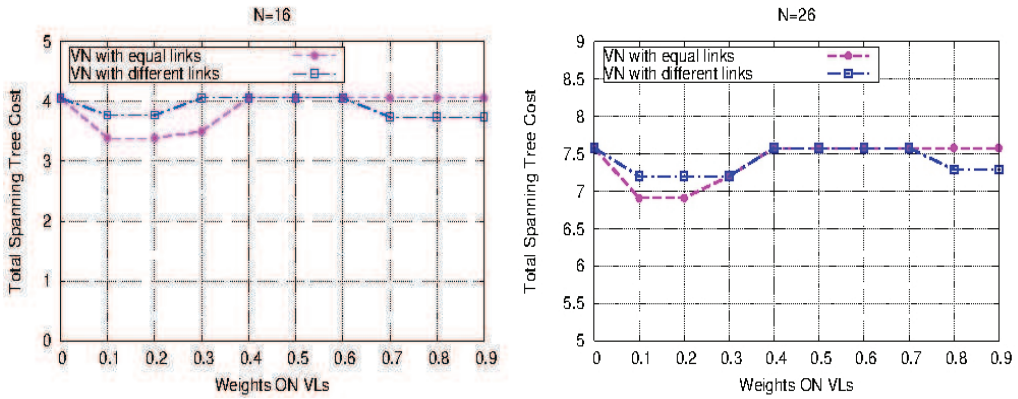


Figure C.1: MST calcul des coûts pour les 16 et 26 nœuds

C4.2.3 Complexité d'Algorithme

La complexité de l'algorithme utilisé pour trouver l'arbre couvrant minimal (MST) dépend du nombre de nœuds, le nombre de liens et le type d'algorithme:

• **«L'algorithme de Kruskal [70]»:** Agrandir l'arbre couvrant minimal (MST) par un arc à la fois en trouvant un arc qui permet de relier deux arbres dans un ensemble d'arbres MST. La complexité temporelle est (O (E + X * log (N)), où X est le nombre d'arêtes ne dépassant pas la plus grande arête du HNR, N et E sont le nombre de nœuds et des arêtes, respectivement.

• **«L'algorithme de Prime [71]»:** Faire agrandir l'arbre couvrant minimal (MST) par un arc à la fois en ajoutant un arc minimal permettant de relier un nœud avec un autre nœud dans le MST. Complexité en temps est (O (E * log (N)), où N et E sont le nombre de nœuds et d'arêtes, respectivement.

Nous avons appliqué le second algorithme pour sa simplicité dans les temps de calcul et aussi pour sa pratique dans la recherche du classement des nœuds.

En termes de temps de complexité consommé par l'algorithme MST pour construire les arbres finaux, nous avons amélioré le temps de convergence de l'algorithme comme suit:

- **Méthode Heuristique**

Pour améliorer le temps de convergence, nous pouvons orienter le point de départ de l'algorithme en le forçant à choisir un nœud (R) en tant que racine principale pour (T^o). Ensuite, le VN sera la nouvelle racine (R) connectée seulement à quelques nœuds présélectionnés. Enfin, nous appliquons l'algorithme MST sur ce graphe pour obtenir

(T^V) qui est l'arbre de sortie virtuelle.

- **Méthode Non-heuristique**

La manière non-heuristique insère le VN entre tous les nœuds dans le graphe. Cela signifie que le VN dispose d'une connexion complète avec tous les nœuds. Ainsi, il y aura un grand nombre de liens et plus de temps de traitement, comme indiqué dans la Figure C.2.

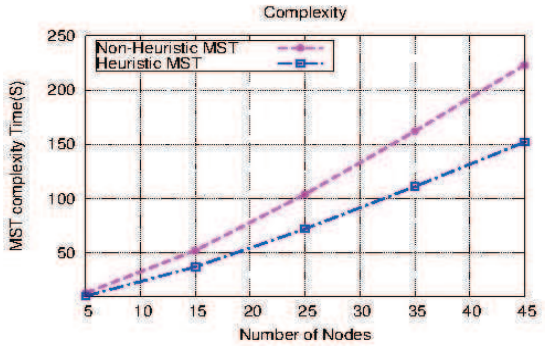


Figure C.2: Méthode heuristique versus méthode non-heuristique pour le test d'algorithme MST

Clairement, la manière heuristique proposée donne un temps de traitement et de calcul plus faible par rapport à la façon non-heuristique (Figure C.2).

En conclusion, nous avons présenté une solution d'optimisation pour les arbres multicast utilisés pour l'IPTV. Il est adapté à des cas de nomadisme où les arbres de diffusion doivent changer. L'idée principale est d'ajouter un nœud virtuel où nous pensons qu'un serveur de réplication doit être mis et de recalculer le nouveau *minimum spanning tree*. La racine de l'arborescence virtuelle est ensuite éliminée. Nous avons également défini de nouvelles métriques de coût pour CAPEX et OPEX utilisées dans notre algorithme. Les résultats montrent un arbre de diffusion optimal IPTV dans toutes les situations.

C4.3 Chaîne de Hachage Sécurisée et Fiable pour le Streaming Vidéo

Cette partie étudie la fiabilité de l'utilisation des chaînes de hachage avec le streaming vidéo. Elle se concentre sur les effets du retard ou de la perte de paquets et les attaques basées sur le fait de rejouer sur la rupture du lien de hachage. Nous proposons un algorithme qui permet de surmonter ces attaques par son mécanisme fiable utilisant une fenêtre avec l'aide de codes de redondance (RC) sécurisés. Ensuite, nous améliorons le degré de sécurité en intégrant les signatures avec fenêtre basée sur RC afin d'éliminer certaines attaques communes à la livraison de vidéo. Ce travail est divisé en trois chapitres :

La première partie constitue un état de l'art sur les chaînes de hachage utilisées pour

les services de streaming vidéo et les applications en temps réel en général. En outre, il fournit quelques informations sur l'utilisation des chaînes de hachage dans l'authentification du service de streaming vidéo comme indiqué dans le Tableau C. 3 suivant :

Tableau C.3: Comparaison entre les différents algorithmes d'authentification basés sur le hachage

Algorithm	Processing Delay	Overhead of Authentication	Robustness to Loss
One-Time signature [1997]	Only at the senders (Packet-Based PB)	Only one packet overhead (first)	No robustness
TESLA [2000]	Buffering at senders and receivers (PB)	According to hash size	Medium
EMSS [2000]	Only at the receivers (PB)	According to the hash indexing level (6 hashes per packet)	Medium
Augmented Chain [2001]	For both senders and receivers (BB)	Less than EMSS	Medium
MAC [2001]	For both senders and receivers (BB)	According to block size	High
Hybrid Hash Chain H2C [2004]	Less delay at senders (PB)	Varied (more than 3 hashes per packet)	High
RLH [2005]	Only at the receivers	Varied	High
Butterfly-Graph (BG) [2007]	For sender and receiver (Block-Based BB)	According to group size	High
Generalized BG [2007]	For sender and receiver (BB)	According the random number of packets selected	High
LMCM [2008]	Only at the senders (BB)	Increased according to number of packets carried previous hash values	High
TV-OTS [2009]	No buffering as processing of packet-based method (PB)	Overhead per packet according to signature size	High tolerance

C4.3.1 Motivations de Travail

Le streaming vidéo ou les applications de service IPTV sont des applications tolérantes aux pertes. Cela signifie que le flux de paquets pourrait manquer certains paquets sans affecter le processus de streaming. Bien que, le service de streaming tolérant aux pertes de paquets, les mesures de sécurité qui seront appliquées comme la chaîne de hachage ou les signatures numériques ne sont pas tolérantes à ces pertes. Cela pourrait nuire aux mesures de sécurité en particulier la méthodologie des chaînes de hachage.

Notre objectif principal est de simplifier la méthode de vérification utilisée par les expéditeurs et les récepteurs pour confirmer que les données ne sont pas interceptées ou modifiées alors que sa transmission à haut degré de fiabilité de la chaîne de hachage.

Nous proposons d'utiliser la méthode de la chaîne de hachage pour cette tâche avec des fonctions de hachage optimisées qui maintiennent des coûts faibles pour l'utilisateur pendant la vérification des données reçues et expédiées. Nous nous concentrerons également sur le mécanisme de réinitialisation (*resynchronisation*) des numéros de

séquence dans le cas où certaines parties du flux vidéo ont été perdues et comment cela va influer sur la décision du récepteur concernant les données perdues. Le mécanisme de chaînage permettra d'améliorer la sécurité. En effet, il va ajouter un certain degré de complexité en raison du calcul ou l'estimation nécessaire pour un intrus d'essayer d'attaquer la vidéo en streaming.

Dans la deuxième partie, l'aspect de la chaîne de hachage basée sur la fenêtre sera expliqué dans ce chapitre avec la mise en clair d'une analyse de certains résultats analytiques et de simulation pour différentes façons d'atteindre la fiabilité dans le mécanisme de la chaîne de hachage. L'objectif dans cette partie est d'étudier et de concevoir de nouvelles solutions pour la resynchronisation de la chaîne de hachage dans le cas d'une certaine perte de paquets. Pour mener cette étude, nous supposons certains paramètres qui seront répétés dans de nombreuses sections, comme indiqué dans l'algorithme de redondance.

Ce travail sera complètement concentré sur les mécanismes relatifs au problème de afin de conserver la chaîne de hachage continue en cas de perte de paquets. Cette perte peut briser le lien des chaînes de hachage et peut conduire à relancer le processus de nouveau. On propose d'ajouter des codes de redondance, calculés en se basant sur les valeurs de hachage des différents blocs de la vidéo. Ces valeurs de redondance seront insérées dans certains paquets à l'intérieur de l'édifice. Par conséquent, ces valeurs aideront le récepteur à extraire les valeurs de hachage pour chaque bloc sans avoir reçu correctement les paquets entiers de ce bloc. Cela signifie que, dans le cas de certains paquets qui sont perdus de l'édifice, cette perte n'affecte pas la continuité de la chaîne de hachage utilisée pour authentifier la transmission vidéo et aussi ne conduira pas à arrêter le streaming.

C4.3.2 L'Algorithme de Redondance des Codes

La Redondance des Codes (RC) sont les informations supplémentaires qui seront ajoutées aux données d'origine par l'expéditeur afin d'aider le récepteur à récupérer l'erreur de transmission. Ils sont équivalent aux codes '*Forward Error Correction*' (FEC). Ces codes permettent de gagner du temps au niveau du calcul coté récepteur et l'ordonnancement des données perdues ou endommagées. Par définition, le service de streaming vidéo ne permet pas les retransmissions de paquets perdus, la RC aidera le récepteur à l'auto-guérison et continuer à jouer la vidéo sans interruptions. En outre, la chaîne de hachage avec la RC garde le lien de hachage ininterrompu en cas de perte de paquets.

L'architecture proposée à la Figure C.3 a de nombreux paramètres qui doivent être initialisés:

Stream de Morceaux: les blocs de sortie après le MPEG-TS (comme les paquets RTP).

Block Bi: est un groupe de paquets qui ont une relation avec leurs gros morceaux numérotés. Par exemple, chaque bloc = 10 paquets et chaque paquet = 7 morceaux dans le cas des paquets RTP.

IV_0: est le vainqueur initial pour le démarrage de la chaîne de hachage.

h (.): ce sont les fonctions de hachage utilisées pour calculer le hachage de sortie comme MD5 ou SHA série.

h_i: c'est la valeur de sortie de hachage ou le résumé de sortie.

H_i: est le condensé de sortie de hachage pour deux techniques de couche de hachage.

Code de combinaison: c'est le processus de codage qui sera utilisé pour calculer un code de redondance pour générer la valeur de hachage au cas où une valeur de hachage du bloc est manquante (ex. XOR fonction).

RC_i: est le code de redondance de sortie qui est responsable pour recalculer la valeur de hachage interrompue de manière à maintenir le lien de hachage et éviter sa cassure. C'est semblable aux codes *Forward Error Correction* (FEC).

Le code de redondance ajouté permet au récepteur de détecter et de corriger les erreurs dans les valeurs à digérée (*Digest value*) (sous certaines restrictions). Ce code est le facteur clé pour résoudre le problème de resynchronisation du lien de hachage.

La récupération des liens de hachage peut se produire sans demander à l'expéditeur de retransmettre des données supplémentaires parce que l'expéditeur est sans mémoire dans le cas du streaming vidéo en ligne. Les avantages de la RC sont deux : le tampon n'est pas nécessaire et la retransmission de valeurs de hachage peut souvent être évitée (ce qui réduit les besoins en bande passante, les calculs de temps et la mémoire tampon). La RC est donc appliquée dans cette situation où les retransmissions sont difficiles à réaliser dans les applications temps réel et d'appareils sans mémoire. L'objectif principal de la RC est la synchronisation de la liaison de hachage et de trouver le point de récupération de la synchronisation par l'obtention de la valeur de hachage de ce temps qui représente le vecteur initial (IV) pour le calcul de hachage suivante de notre chaîne.

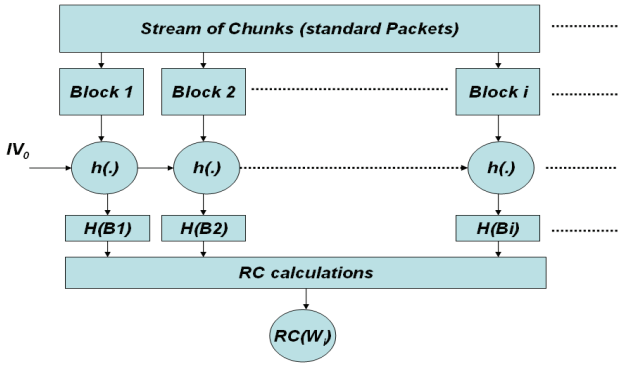


Figure C. 3: Schéma de principe de la redondance de la chaîne de hachage pour le streaming vidéo. Le flux original est divisé en morceaux (série de paquets), puis les assembler à des blocs spécifiques Bi, après que la chaîne de hachage appliquée à des blocs, enfin, la RC calculé en fonction de la taille de la fenêtre statique.

C4.3.3 L'Algorithme de Redondance des Codes Mélangé avec Signature

Troisième partie; cette section présente l'intégration d'un mécanisme de signature basé sur fenêtre avec RC. Ensuite, la comparaison entre différentes options (basé sur paquets, basé sur des blocs et la fenêtre en fonction avec RC) sera menée en termes de fiabilité et la charge supplémentaire.

- **Signature À base de Paquets (Packet-Based)**

Dans cette catégorie, chaque paquet doit se greffer la signature numérique de son contenu. Bien que ce type fournisse une mesure de sécurité parfaite pour le streaming vidéo, il souffre de quelques inconvénients, tel qu'un overhead lourd et un temps de calcul de la sécurité de vérification très important.

- **Signature À base de Bloc (Block-Based)**

Dans cette catégorie, seuls quelques paquets aléatoires à partir d'un bloc contenir de nombreux paquets pourrait porter la signature de l'information bloc entier.

- **Signature À base de Fenêtre (Window-Based)**

Dans cette catégorie, nous considérons le bloc-base de haut niveau en regroupant certains blocs ainsi que dans le mécanisme de la fenêtre. En outre, nous intégrons la signature avec le mécanisme de redondance des codes expliqué dans la partie précédente.

Dans tous les cas, nous avons deux phases importantes comme suit:

Phase d'échange Sécurisée:

Cette phase se concentre sur l'accord entre le serveur et les clients. En outre, son objectif est de générer soit une connexion sécurisée (IV) pour être utilisée dans la chaîne de hachage ou une clé privé sécurisée qui permettra de signer la valeur de hachage. Il y a

de nombreux algorithmes de sécurité qui peuvent gérer ce processus comme Diffie-Hellman (DH) [127] ou la cryptographie à courbe elliptique (ECC) [128]. Nous utilisons une clé d'accord de courbe elliptique fondée sur l'identité de la cryptographie (IBC) [129]. Comme chaque utilisateur dispose d'une adresse e-mail YouTube (par exemple: bob@gmail.com), nous allons l'utiliser pour générer une clé publique. Cela conduira à un accord clé personnalisée pour accéder à des services de YouTube.

Phase d'Intégration :

Le choix de superposer la RC avec un paquet à l'intérieur de la fenêtre pourrait être aléatoire si le système pourrait s'appuyer sur les hachages purs. Dans notre proposition, nous allons intégrer la signature dans le système. Donc, nous avons préféré fixer le dernier paquet de chaque fenêtre pour effectuer cette signature. Après avoir terminé le processus de réplication en fonction des capacités et de l'overhead autorisé pour obtenir un niveau spécifique de fiabilité, nous avons deux scénarios de la signature du flux:

- **La signature de la valeur de hachage**: cette méthode suit tous les travaux antérieurs en signant le streaming vidéo. Cela dépend de l'aide pré-généré beaucoup de clés sécurisées par PKG comme dans la section précédente, puis, en appliquant l'algorithme de signature comme RSA [99].
- **La signature de la valeur RC**: la taille de la redondance des codes RCs est la même que la valeur de la taille condensée, nous envoyons les RCs en mode clair. C'est parce que l'objectif des RCs est de parvenir à la fiabilité dans le lien de hachage de la méthode de la chaîne de hachage alors que certains paquets du flux ont été perdus (comme expliqué dans le chapitre précédent). Donc, il n'y a pas de besoin urgent de signer le RC. Ce code sera greffé dans chaque fenêtre, mais dans le premier paquet. Il sera le moyen pour les récepteurs de suivre l'horodatage pour chaque fenêtre.

C4.3.4 Résultats

Les résultats portent sur la comparaison des performances entre les différents modèles qui sont à base de paquets, à base de blocs et à base de fenêtres & RC. Les résultats analytiques et de simulation par Matlab sont basés sur la taille de bloc standard de 512 octets des techniques de hachage de MD5 [118] et SHA-1 [119] respectivement. En supposant que, nous avons un fichier vidéo qui a (N) paquets et est divisé en (λ) Blocs ou fenêtres. Si nous utilisons la méthode de la chaîne de hachage pour calculer la valeur de hachage qui sera utilisée pour calculer la signature par les trois méthodes ci-dessus, la surcharge de signature et le calcul du temps seront différents selon la façon dont on

greffe cette signature dans le flux. Cette surcharge peut varier selon le type de module utilisé pour la longueur de la signature (par exemple une clé de longueur 1024 bits donne une surcharge de 128 octets de signature et de une clé 2048 bits donnera 256 Octets de signature). Nous supposons que, la taille de la signature sera (X-octets) pour chaque paquet, (N) est le nombre total de paquets par fichier vidéo et (λ) est le nombre de blocs ou fenêtres per vidéo. Tableau C.4 explique les paramètres proposés et résume l'overhead produits par les temps de calcul pour les trois modèles PB, BB et la WB & RC.

Tableau C.4: Comparaison analytique entre les trois façons d'ajouter la signature sur la base (Paquet, Bloc, et la Fenêtre & RC).

	Signature Overhead Bytes	**No of Signed Packets**	**Processing Time** (T_p) ***msec***	**Verification Time** (T_v) ***msec***
Packet-Based (PB)	$N.X$	N	T_p	T_v
Block-Based (BB)	$\lambda.X$	*One*	T_p/λ	T_v/λ
Window-Based & RC (WB&RC)	$\lambda.X + N_{RC}$	*One or More*	$T_p/\lambda + T_{RC}$	$T_v/\lambda + T_{RC}$

Les résultats analytiques et de simulation indiquent que, la technique basée sur la fenêtre & RC n'ajoute pas un temps de vérification remarquable comme le montre la Figure C.4. Nous supposons que, le délai total pour notre technique ne dépasse pas 2 secondes ce qui nous donne jusqu'à 85 paquets par une fenêtre ou par un Bloc (la taille présélectionnée de la fenêtre pour notre hypothèse d'un délai ne dépassant pas 2 sec de retard entre le streaming source et la destination donne le nombre acceptable de 85 paquets de taille standard 1500 octets MTU dans la contribution précédente).

Pour la comparaison entre nos trois modèles (PB, BB et la WB & RC), nous simulons les modèles basés sur la taille du paquet MTU standard et tampon de fenêtre 85 paquets qui ont été obtenu précédemment. Les résultats, comme le montre la Figure C.4, indiquent que le temps de traitement diminue à mesure que le débit vidéo augmente de façon à maintenir le taux de lecture vidéo pour qu'il ne soit pas affecté par le temps de traitement et des mesures de sûreté.

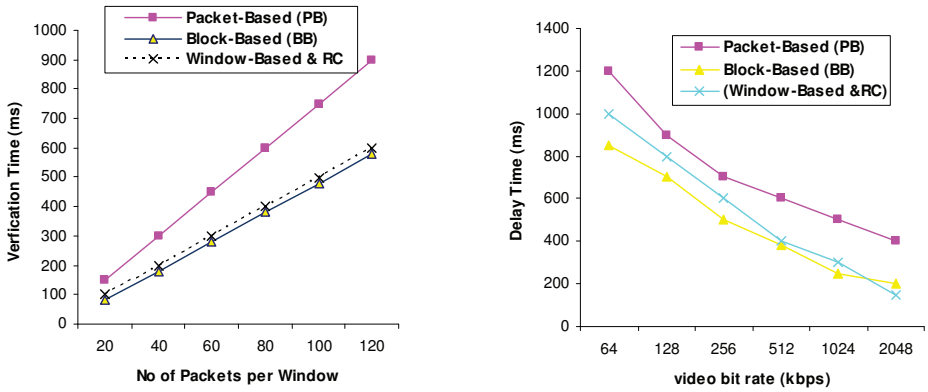


Figure C.4: Le temps de vérification de signature en fonction du nombre de paquets par Bloc ou d'une fenêtre et le débit binaire vidéo pour la fenêtre présélectionnée 85 paquets

C4.3.5 Hypothèses et Analyse des Attaques

Dans cette partie du travail, nous analysons certains problèmes et nous considérons certaines attaques que notre proposition pourrait faire face et nous essayons de définir des solutions pour les traiter:

• Man-In-The-Middle (MITM)

La procédure de sécurité qui a traité l'accès à la connexion avec (PKG) sur le côté serveur pourrait empêcher ce type d'attaques. Nous avons utilisé (SSL) pour sécuriser la voie de la distribution de clé entre le serveur et les clients.

• Retard

Tout paquet ayant un numéro n'appartenant à la plage de nombre de la fenêtre sera rejeté. Cela va protéger le récepteur de tous les problèmes de retard de paquets.

• Robustesse

Le RC aidera le récepteur dans l'auto-guérison de manière à l'aider à trouver le lien de hachage dans le cas de certains paquets perdus de la fenêtre. Le RC aidera le récepteur dans l'extraction de la valeur initiale nécessaire pour poursuivre le processus de hachage.

• Anti-Replay

Si certains attaquants modifient certains paquets du flux, les modifient, puis les renvoient de nouveau aux récepteurs le long du chemin, la fenêtre horodatée traitera ce problème en vérifiant le temps et le rejet de ces paquets. En outre, la correspondance des signatures jugera la validité de toute la fenêtre et qui n'a pas droit d'indexer les paquets signés.

C.5 Conclusions

Cette thèse propose trois contributions majeures:

1. **Optimisation de la Distribution des Fichiers Vidéo**

Dans cette contribution, nous avons fourni une étude complète sur les serveurs de partage de vidéos par l'optimisation de l'allocation des fichiers. Cette optimisation comprend deux aspects:

- ***Mécanisme d'Allocation des Fichiers***

Grâce à cette partie, nous avons introduit un algorithme d'optimisation pour l'allocation de fichier vidéo basé sur le nombre maximal de résultats liés à chaque fichier et ses taux de visualisation selon les informations de profil calculées par les compteurs de YouTube. Notre algorithme permet de façon simple des mouvements des fichiers entre les différentes zones en fonction du coût minimum de l'accès aux fichiers. En outre, nous avons mis en évidence les effets sociaux des comportements et de l'impact de l'information géographique sur les taux d'utilisation et d'accès de fichiers de YouTube.

- ***Mécanisme de Reproduction des Fichiers***

La deuxième étape a consisté à analyser les duplications de fichiers vidéo qui vont permettre d'atteindre une meilleure qualité de satisfaction de l'expérience des utilisateurs (QoE) et un gain élevé pour les opérateurs de réseaux.

2. **Architectures Ouvertes et l'Optimisation de Réseaux IPTV**

Dans les architectures IPTV ouvertes, nous avons proposé une conception collaborative pour le modèle d'affaires moderne. Ensuite, ce modèle est validé et analysé à travers un banc d'essai. Les avantages de la collaboration dans la conception, soit avec les coûts CAPEX ou OPEX sont analysés. En outre, les impacts de cette conception sont dérivés pour s'adapter à la conception accès nomade de la prestation de services IPTV.

Les exigences IPTV et le comportement dynamique des topologies de multidiffusion ont besoin de combiner de nouveaux réseaux MST qui prennent en considération les coûts et la mobilité. Nous proposons un algorithme de sélection de nouvelles racines qui remplace les anciennes méthodes. En utilisant des implémentations connues de multidiffusion (comme l'ASM ou SSM); les sources de multidiffusion doivent être apprises à l'avance via des méthodes externes (configuration manuelle, par exemple). Notre façon dynamique pour modifier les sources de multidiffusion améliore les performances du réseau. L'algorithme proposé repose sur la mise en place d'un nœud virtuel VN dans le graphe du réseau d'origine. Ensuite, nous recalculons le coût du MST

pour la topologie virtuelle. Grâce aux résultats de l'optimisation et la simulation, nous obtenons un coût minimal (Spanning Tree) pour les sous-arbres par rapport à l'original. En outre, nous améliorons la complexité de l'optimisation par l'algorithme de guidage à travers le test non-heuristique.

3. **Fiabilité de l'Utilisation de Chaîne de Hachage dans la Sécurité Vidéo en Streaming**

Cette partie étudie différentes méthodes pour atteindre l'état de resynchronisation pour les liens de la chaîne de hachage. Il propose également un algorithme hybride basé sur la redondance des codes (RC) et les flux Windows appelé «code de redondance de récupération de synchronisation état (RC-SRS)». Cette technique combine les avantages de toutes les méthodes (SHHC, TSP, MLHC et TSS) et évite leur faiblesse. Les résultats analytiques et de simulation pour l'algorithme hybride indiquent qu'il donne de bonnes performances globales en termes de complexité et de temps de calcul par rapport à d'autres solutions.

En termes de complexité, une comparaison a été faite entre ces différentes manières d'atteindre la resynchronisation de la chaîne de hachage. Le RC-SRS pour la resynchronisation sur la base de codes de redondance donne des résultats acceptables qui indiquent que la RC ne causera pas des temps de calcul supplémentaires pour l'expéditeur ou pour des vérifications de réception. En outre, l'overhead ajouté est acceptable en termes de taille de paquet standard et MTU 1500 octets. La contribution finale pour RC-SRC réside dans ses caractéristiques hybrides de (SHHS, TSP, MLHC et TSS); il est caractérisé par auto-guérison et dépend de la RC pour la génération des valeurs de hachage au niveau du récepteur. Il utilise un mécanisme d'indexation pour atteindre la fiabilité comme TSP. En outre, il construit la technique de MLHC pour surmonter l'évitement de collision et, enfin, les numéros de séquences de paquets et les fenêtres comme TSS pour prévenir du retard et les attaques par rejoué.

Une valeur ajoutée à notre algorithme de la signature intégrée au calcul fenêtre. En outre, nous avons classé les moyens d'intégration en trois méthodes (PB, BB et la WB&RC). Ces méthodes ont été comparées analytiquement au niveau de leurs overheads et de leurs performances. En outre, la fiabilité de notre algorithme a été améliorée en cas de taux d'erreur supérieur à 0,2%.

C.6 Perspectives

En termes de distributions de fichiers vidéo, nous espérons appliquer notre algorithme sur des statistiques à plusieurs serveurs tels que YouTube, Dailymotion ou n'importe quel serveur de partage de vidéos sur Internet. De plus, la mise en cache et l'optimisation de gestion de contenu doivent être étudiées pour aider dans l'optimisation des performances de ces types de réseaux sociaux. En outre, il sera possible de formuler ce problème en modèle de réseau de files d'attente et d'étudier la performance de mouvement ou de la mise en cache des fichiers dans différents endroits ou zones géographiques.

Pour l'IPTV ouverte et les services vidéo gérés, de nouveaux travaux doivent être ajoutés comme la construction d'un outil graphique avancé qui pourrait aider les opérateurs dans la gestion de leurs nœuds de multidiffusion. En outre, nous espérons faire quelques optimisations dans la prestation de nuages pour l'IPTV en se basant soit sur l'amélioration des performances ou la sécurisation des médias. Enfin, une contrainte MST pourrait être appliquée avec un nombre de nœuds forcés comme des racines qui mèneront à NP-difficile dans la complexité.

Comme la fiabilité est un problème dans la mesure de sécurité basée méthodologie de la chaîne de hachage, ce travail a adopté la technique statique de fenêtre coulissante pour le calcul de la RC et notre point de vue est de simuler le cas d'utilisation dynamique. Ce scénario sera construit en utilisant dynamique fenêtre coulissante. Nous nous attendons à ce que ce scénario va ajouter plus de robustesse et augmenter le degré de fiabilité et le degré de récupération. Aussi, nous espérons intégrer notre solution dans le système réel de la vidéo en streaming comme VLC (serveur et client).

Enfin, l'avenir pour le service vidéo sera la fusion entre les services géré avec non-géré afin de créer plus d'interactivité entre les services et les questions de sécurité ou de confidentialité des contenus sur les réseaux NGN continuent toujours de s'imposer. A côté de cela, la question de la personnalisation dans la conception des réseaux multimédias et sociaux peuvent être considérés dans les travaux futurs.

Appendix D: Acronyms

D.1 Glossary

3GPP	3rd Generation Partnership Project
AAA	Authentication Authorization and Accounting
ADSL	Asymmetric DSL
ASM	Any Source Multicast
BB	Block-Based
BE	Best Effort
BG	Butterfly Graph
BW	Bandwidth
CAPEX	Capital Expenditure
CCI	Cloud Computing Infrastructure
CDA	Cross Domain Authentication
CDN	Content Delivery Network
CP	Content Provider
DG	Directed Graph
DH	Diffie-Hellman
DI	Data Integrity
DOA	Data Origin Authentication
DRM	Digital Right Management
DSL	Digital Subscriber Line
DSLAM	DSL Access Multiplexer
DVB	Digital Video Broadcasting
ECC	Elliptic Curve Cryptography
EHC	Efficient Hash Chain
EMSS	Efficient Multi-chained Stream Signature
FEC	Forward Error Correction
FI	Federation Identity
FID	Future Internet Design
FIND	Future Internet Design
FTTL	File Time To Live
GBG	Generalized Butterfly Graph
GCM	General Content Management
GENI	Global Environment for Network Innovations
GF	Galois Field

GSM	Group Shared Multicast
GUI	Graphical User Interface
H2A	Hybrid Hash Chain
HN	Home Network
IBC	Identity Based Cryptography
IMS	IP Multimedia Subsystem
IPI	Internet Protocol Infrastructure
IPSec	IP Security
IPTV	Internet Protocol Television
ISP	Internet Service Provider
IV	Initialization Vector
LCM	Local Content Management
LFU	Least Frequently Used
LMCM	Layered Multiple Chaining Model
LRU	Least Recently Used
MAC	Message Authentication Code
MCAST	Multicast
MCN	Multicast Node
MD	Message Digest
MI	Multi Identity
MITM	Man-in-the-middle
MLHC	Multi Layer Hash Chain
MPEG	Moving Picture Experts Group
MPEG-TS	MPEG-Transport Stream
MR	Multicast Replication
MST	Minimum Spanning Tree
NA	Nomadic Access
NAT	Network Address Translation
ND	Nomadic Device
NGMN	Next Generation Multimedia Networks
NGN	Next Generation Network
NIST	National Institute of Standards and Technology
NN	Nomadic Network
NS	Nomadic Service
OPEX	Operational Expenditure
OTT	Over The Top TV
PB	Packet-Based
PDA	Personal Digital Assistant
PER	Packet Error Rate

PIM	Protocol Independent Multicast
PIM-DM	PIM Dense Mode
PIM-SM	PIM Sparse Mode
PKG	Private Key Generator
PPP	Point to Point Protocol
PPPoE	PPP over Ethernet
QoE	Quality of Experience
QoS	Quality of Service
RC	Redundancy Code
RC-SRC	Redundancy Code Synchronization Recovery State
RDA	Rate Distortion Authentication
RF	Redundancy Factor
ROD	Resource On Demand
RSA	Rivest Shamir Adleman algorithm
RTP	Real Time Protocol
RTT	Round Trip Time
SG	Sparse Graph
SHA	Secure Hash Algorithm
SHHC	Self-Healing Hash Chain
SIP	Session Initiation Protocol
SR	Switching Router
SRTP	Secure Real-Time Transport Protocol
SS	Streaming Server
SSL	Secure Socket Layer
SSM	Source Specific Multicast
ST	Service Triggering
STA	Spanning Tree Algorithm
STB	Set-Top-Box
STP	Spanning Tree Protocol
TESLA	Time Efficient Stream Loss-tolerant Authentication
TSP	Time Synchronization Point
TSS	Timestamp Synchronization
TVA	Television Anytime
TVE	Television Everywhere
TV-OTS	Time Valid One-Time Signature
UAA	User Authorization Answer
UAR	User Authorization Request
UG	Undirected Graph
UGC	User Generated Content
UN	Ubiquitous Network
UP-TO-US	User-Centric Personalized IPTV UbiquitOus and SecUre Services

URI	Uniform Resource Identifier
VLC	Video LAN Client
VN	Virtual Node
VN	Visited Network
VOD	Video on Demand
VPN	Virtual Private Network
VSN	Virtual Service Network
WB	Window-Based