Bi-Directional Transition Mechanism between IPv4 and IPv6

آلية تحويل ثنائية الاتجاه بين بروتوكول الانترنت الإصدار الرابع والإصدار السادس

By

Rola A. A. Haddad
200920140

Supervised By
Prof .Dr. Muzhir Shaban Al- Ani

This Thesis Submitted in Partial Fulfillment of the requirements for the Master's degree of Science in Computer Science

Department of Computer Science
College of Computer Sciences and Informatics
Amman Arab University

(2013)

Discussion Committee's Decision

This Dissertation was discussed under the title:

"Bi-Directional Transition Mechanism between IPv4 and IPv6"

It was approved in : 30/1/2013

Dissertation Committee Approval:		**Signature**
Prof.Dr.Reyadh Naoum	Chairman	R. Naoum
Prof.Dr..Akram Al-Mashaykhi	Member	a. m. Alhman
Prof.Dr.Muzhir Al-Ani	Member/Supervisor	Mizhur

Delegation

I am Rola Abdel_Jaleel Ali Haddad delegates Amman Arab University to make copy of my dissertation to libraries institutions , or people when asked.

Name: Rola Abdel_Jaleel Ali Haddad

Signature: 

Date: 30/1/2013

Acknowledgments

In the name of Allah, the Most Gracious and the Most Merciful.

It is a pleasure to thank the many people who made this thesis possible.

It is difficult to overstate my gratitude to my supervisor, Prof. Dr. Muzhir Al-Ani. With his enthusiasm, his inspiration, his ideas and his great efforts to supervise me, provide his help to make my work success.

I wish to thank Prof. Dr. Alaa Al-Hamami for helping me get through the difficult times.

Finally, I express a deep sense of gratitude to my mother, my husband and all my family for their valuable and untiring moral support.

To spirit of my father I dedicate this thesis.

Table of Contents

Table of Abbreviations

Abbreviation	Description
ARPA	Advanced Research Project Agency
BDMS	Bi-Directional Mapping System
BDIP	Bi- Directional Intelligent Processing
BDTM	Bi-Directional Transition Mechanism
BIS	Bump In the Stack
CIDR	Classless Inter-Domain Routing
DF	Don’t Fragments
DHCP	Dynamic Host Configuration Protocol
DNS	Domain Name Server
DSTM	Dual Stack Transition Mechanism

Http	Hyper Text Transfer Protocol
IANA	Internet Assigned Numbers Authority
ICANN	Internet Corporation for Assigned Names and Numbers
ICMP	Internet Control Message Protocol
ICMPv4	Internet Control Message Protocol –Version Four
ID field	Identification field
IDS	Intrusion Detection System
IETF	Internet Engineering Task Force
IP	Internet Protocol
IPsec	Internet Protocol Security
IPv4	Internet Protocol version 4
IPv5	Internet Protocol version 5
IPv6	Internet Protocol version 6
IPv7	Internet Protocol version 7
IPv8	Internet Protocol version 8
IPv9	Internet Protocol version 9
ISP	Internet Service Provider
IVI	Transition from IPv4 to IPv6, IV refers to 4,VI refers to 6

MF	More Fragments
MTU	Maximum Transmitted Unit
NAT	Network Address Translation
NAT-PT	Network Address Translation- Protocol Transition
P2P	Peer To Peer
QoS	Quality of Service
RFC	Request For Comment
RIR	Regional Internet Registries
SIIT	Stateless IP-ICMP Translation
TCP	Transmission Control Protocol
THL	The Header Length field
TOS	Type Of Service
TTL	Time To Live
URL	Uniform Resource Locator
VMware	Virtualization software

List of Figures

List of Tables

Abstract

Internet Protocol version 4 (IPv4) has been used for long time ago, but the great growth of the Internet and its change in drastic way over time makes IPv4 face many problems. The main one is the lack in address space, not to forget issues such as mobility and security. To meet requirements that didn't exist in IPv4, IP version 6 (IPv6) was designed.

The new Internet Protocol (IPv6) has been developed to replace the current Internet Protocol (IPv4) and the transition from IPv4 to IPv6 is necessary process in the realization of global Internet, with the development of IPv6 technology and continuous increase in application, but this process will take long time so transition methods will be needed. There are many IPv4/IPv6 transition methods already exist today, some of them applied in practice; others still as proposed solutions. Tunneling; Encapsulation, methods are the most techniques that used until now, but all encapsulation mechanisms suffer from increasing of the overhead traffic network as a result for either encapsulating IPv4 packet in the IPv6 packet or encapsulating IPv6 packet in the IPv4 packet.

Our main goal of this thesis is to propose a transition mechanism that makes incompatible nodes; the first based IPv4 the other based IPv6, communicate together. This mechanism is called Bi-Directional Transition Mechanism (BDTM). BDTM depends on understanding of the two environment of transmission that is, receiving the source packet then converting the information header to be adaptable to the destination end.

Our mechanism has been implemented; then we made test using simulation tool called (VMware); during this simulation, one scenario was studied, and the results have shown that BDTM make two incomputable protocol hosts communicate together.

Arabic Summary

الملخص

ما زال الإصدار الرابع من بروتوكول الانترنت يستخدم منذ فترة طويلة ولكن النمو الهائل للانترنت و التغييرات التي تطرأ عليها جعل الإصدار الرابع يواجه عدة مشاكل ، أهم مشكلة هي القصور في عدد العناوين التي يغطيها الإصدار، ولا ننسى القضايا المتعلقة بالحركة وقضايا الأمنية . و لمعالجة هذه القضايا تم تصميم الإصدار السادس من بروتوكول الانترنت.

صمم الإصدار السادس لغايات استبداله بالإصدار الرابع ، ولكن عملية الانتقال هذه ستأخذ فترة طويلة ، مما يعني الحاجة الماسة لتقديم تقنيات تحويل بين الإصدارين، وقد تم تقديم عدة تقنيات تحويل ، منها ما طبَق عمليا ومنها ما زال مجرد حلول مقترحة. تعد طريقة الأنفاق أو التغليف من أكثر الطرق المستخدمة حاليا. ولكن تطبيق هذه الطريقة سيزيد من حجم الحزمة المرسلة، وهذه الزيادة ناتجة من تغليف الحزمة المعنونة بالإصدار السادس كحزمة معنونة بالإصدار الرابع للمرور داخل بيئة الإصدار الرابع أو العكس.

إن الهدف الرئيسي من هذه الأطروحة هو اقتراح آلية تحويل تسمح لنقاط تعتمد بروتوكولات مختلفة – الإصدار الرابع و الإصدار السادس- من الاتصال معا. تم تسمية هذه الآلية (بآلية التحويل ثنائية الاتجاه) .

آلية التحويل ثنائية الاتجاه مبنية على الفهم للبيئتين المرسلة و المستقبلة ، حيث يتم التقاط الحزمة المرسلة و تحويلها إلى حزمة تتوافق مع إصدار الجهة المرسلة إليها .

تم تطبيق الآلية، ثم استخدمنا إحدى أدوات المحاكاة (VMware) في دراسة سيناريو واحد تم فيه التواصل بين جهازين مختلفين في إصدارات بروتوكولاتهما .

Chapter One
Introduction

1.1 Overview

The Internet Protocol (IP) was originally designed to facilitate connection of various organizations involved in the defense department's Advanced Research Project Agency (ARPA). During the time before IP, networks were comprised of proprietary equipment and proprietary protocols. Vendor A's mainframe did not communicate with Vendor B's mini-computer and so on. IP was conceived as a common protocol that would allow these different computers to communicate together [1].

Vinton G.Cerf and Robert E. Kahn in 1974 presented IP as a protocol that supports the sharing of resources that exist in different packet switching networks. IP is the most important protocol for communication for relaying packets of data. The relay of data packets occurs across the internet network by using a specially designed IP suite. The wide domain of Internet is managed by a primary protocol, and it helps relay of pocket data across network domains. IP has the main responsibility of carrying data pockets from the source to the destination based on their addresses [2].

Now, the Internet Assigned Numbers Authority (IANA) administers the IP address space allocations on a universal basis. It also allows to five Regional Internet Registries (RIR) to allocate IP addresses to local branches of Internet registries or Internet Service Providers (ISP) [2].

IP is very simple in its functionality. It is only designed to handle the addressing and fragmentation of data-grams. Any additional functionality such as acknowledgement of received packets and retransmission of corrupted packets is passed off to the higher-level layers such as TCP [1].

An Internet Protocol address or IP address is a numerical label tagged to devices that take part in a local or wide area computer network that actively uses IP for communicating data and information. An IP address has two purposes to perform: identify host or network interface and address various locations. An IP address is a 32-bit number, and this system is the IP Version 4(IPv4); most networks still use this format today.

No one expected that the number of Internet users will increase in this way, so when IPv4 was designed; it takes that the maximum numbers of devices that will have IP is 2^{32}. That means an IP address can provide a maximum (4,294,967,296) possible addresses, but in 1992, the Internet Engineering Task Force (IETF) realized the current IP address space was running out as a result of rapid growth of Internet size and applications, so the need for a new protocol that has larger address space and improvement features was needed; Because of that, a solution for this problem was solved by IP Version 6 (IPv6) which offers a huge address space (3.4×10^{38}) possible address which will be more than enough [22].

1.2 Internet Protocol Addresses (IP Addresses)

Every device connected to the public Internet is assigned a unique number known as an IP address. The designers of the IP defined an IP address as a 32-bit number, and this system, known as IPv4, is still used today. A new versions of IPs had been proposed by researchers some of these versions will discussed in this section. ICANN the Internet Corporation for Assigned Names and Numbers tries to manage the address space in Internet, and also they have designed different version of IP addresses after IPv4.

This list has IPv7 - (just an expansion of IPv4) IPV8, IPv9 and none of them are operational now; this means they are in draft state.

1.2.1 Internet Protocol version 4 (IPv4)

IPv4 address can provide a maximum of 4294967296 (2^{32}) possible address spaces out of which 18 million private network addresses and 270 million multi-cast addresses are reserved. A typical IP address comprises of dot-decimal notation. This notation has four decimal numbers, each one of which ranges from zero to 255, each separated a part by a dot (for example, 192.168.1.1) [11].

1.2.2 Internet Protocol version 6 (IPv6)

In 1992, the Internet Engineering Task Force (IETF) realized the current IP address space was running out. A shortage of routable IP addresses would wreak havoc on the world's e-based economy. Without routable IP addresses, one could imagine the negative

impact this could have on national security issues, electrical power delivery, or simply expansion of the Internet to new markets.

This worldwide IP address shortage posed particular risks to emerging nations such as China and India, although as stated above, China has taken bold steps toward resolving this issue. The use of measures such as Network Address Translation (NAT), Classless Inter-Domain Routing (CIDR), and Dynamic Host Configuration Protocol (DHCP) slowed IP address consumption but was still unable to prevent IP address space from eventual exhaustion. Increased usage of mobile devices such as Personal Digital Assistants (PDAs) and laptops, along with cable modems, considerably accelerated the consumption of routable public IP addresses [4].

To make address more readable, IPv6 specifies hexadecimal colon notation. In notation 128 bits are divided into eight sections, each 2 bytes in length. Two bytes in hexadecimal notation require four hexadecimal digits. Therefore, the address consists of 32 hexadecimal digits with every four digits separated by a colon. Although the IP address, even in hexadecimal format, is very long, many of the digits are zeros. In this we can abbreviate the address. The leading zeros of a section can be omitted. Only the leading zeros can be dropped, not the trailing zeros as shown in Figure 1.1[19].

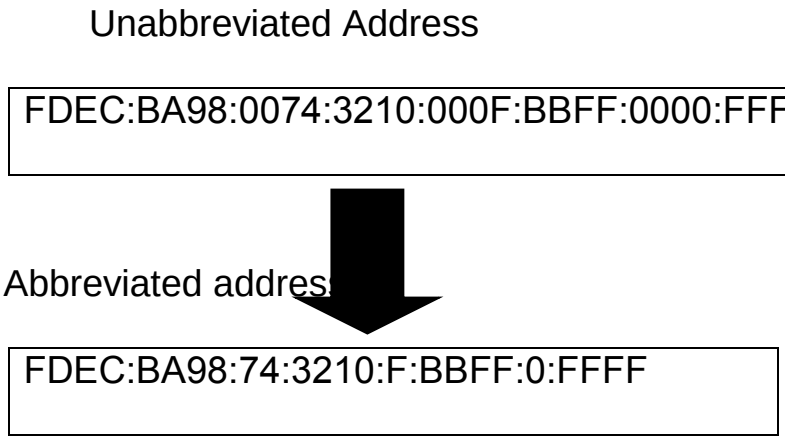


Figure 1.1 . IPv6 address notation

1.2.3 Internet Protocol version 5 (IPv5)

Everyone who starts reading about IPv6 may ask himself " if it's the next version after IPv4, why isn't it called IPv5? ". As what will be explained in details later in this thesis, there is a field in IPv4 that contains IP version number in binary (0100) - 4 in decimal - , but it's contains (0100) in IPv6 – 6 in decimal -. In 1990 a protocol was defined in RFC 1190, which used the binary pattern (0101) - 5 in decimal - in the IP version field of the packet header. It was not really a replacement for IPv4, and isn't even used today [19].

1.2.4 Internet Protocol version 8 (IPv8)

IPv8 is derived from the size of the IP address of 8 bytes. An IP packet consists of the IP header with control information and data to be transported with the package. That are 64 bit and thus a space of 2^64 addresses. There is a bigger rang in address space in comparison with IPv4, but nowadays IPv8 is a forgotten version . perhaps somebody thinks that IPv8 is new version of IP addresses or is very advanced technology, but actually IPv6 changed everything [26].

1.2.5 Internet Protocol version 9 (IPv9)

IPv9 has not been allocated for this protocol by IANA. It is designed to compete IPv6. On June 25, 2004, an announcement was made at the New Generation Internet Ten-Digit Network Industrialization & Development Seminar at Zhejiang University that the protocol had been formally adapted and popularized into the civil and commercial sectors of china. It had apparently been under development for ten years prior to the announcement and was to be used in China's National Safety Defense System and National Digital TV Network among other programs and organizations. IPv9 has been described as a protocol similar to IPv6 ,but with a 256 bit address space instead of IPv6's 128 bit address space [27].

IPv9 and IPv6 are both the Internet Protocol in order to replace the version of IPv4 currently being used on the Internet. The IPv9 referred to phone numbers that navigate the web by using Phone numbers rather than the usual way typed in the browser; This technology appears to be basically a modified DNS, and the business model is to get to registered phone numbers with them to be IP address, while IPv6 builds on the strengths of IPv4 and contains everything that was proven to be good and stable in IPv4 and extended it to meet the needs of the future Internet [27].

1.3 Literature Review

Many researches discussed IPv4, IPv6 and the migration from IPv4 to IPv6, in this section we will discuss the researches that have been done and related to this thesis.

Jesus Ibanez Parra (2004) [23] explained the main characteristics of IPv6, as well as the differences between IPv4 and IPv6 protocols. He also explained the structure of IPv6 header, and he focused in three migration mechanisms from IPv4 to IPv6: dual stack, tunnels and transition NAT/PT, which will be explained in chapter two in this thesis.

Chen Yan-ge et al. (2009) [20] discussed many technologies to ensure smooth transition of IPv4 to IPv6, like IPv6 over IPv4 tunnel, 6PE tunnel, deliberation tunnel, NAT-PT technology, SIIT (Stateless IP/ ICMP Translation), BIS(Bump-In-the Stack), 4over6 technology and DSTM (Dual-stack Transition Mechanism).

Also, they insisted on Operation Departments and Research Departments to cooperate to solve the issue of IPv4/IPv6 bidirectional communication, which is important to expanding the IPv6 business and promoting the smooth transition of IPv4 to IPv6.

Punithavathani et al. (2009) [13] examined and evaluated the performance of 6to4 tunnel, configured tunnel and tunnel broker mechanisms in a real network. The results of their work was that 6to4-tunnel mechanism will be the solution of first choice because address allocation is simple and only one tunnel endpoint must to be configured.

Xianhuiche et al. (2010) [12] summarized the challenges for the IPv6 migration and proposed solutions and guidelines for IPv6 migration, also they produced ratios for IPv6 deployment in five continents as shown in Table 1.1.

Table1.1: IPv6 deployment in five continents (by April 2009) [12]

Continent	Deployment Ratio
Europe	8.8%
America	7.6%
Asia	3.6%
Africa	3.4%
Australia	6.3%

Chen Shengyang et al. (2010) [22] proposed Peer - to - Peer (P2P) communication mechanism based on request forwarding in IPv4 and IPv6 coexistence network without any change of current network devices.

The simulation results show that the proposed P2P communication mechanism can maintain the entire P2P system stable in different periods of IPv6 evolution.

Basil Al- Kassasbeh et al. (2010) [5] proposed algorithms for the header processing transition from IPv4 to IPv6 and vice versa; these algorithms depend on two systems: Bi- Directional Intelligent Processing System (BDIPS) and the Bi- Directional Mapping System (BDMS). BDIP architecture consists of the V4-V6 Domain Name Server, the V4-V6 Enabled router , the IPv4 host names which are located in the IPv4-only zone, and the IPv6 host names which are located in the IPv-6 only zone.

John M. Chasser (2010) [4] introduced the concept of a Transitional Network, defined what it is, how it is used and why a transitional network is necessary. Also, he discussed popular transition methods like: Dual stack for hosts and routers, IPv4 to IPv6 header translation (NAT-PT), IPv6 over IPv4 tunneling mechanism and 6to4 tunnel.

Srikant A.Vardhana et al. (2010) [16] stressed the fact that the main business case for migration to IPv6 is offering existing and new services to customers post IPv4 exhaustion, and ISPs providing residential services need to understand various factors listed in their paper and plan their IPv6 migration to ensure no loss of revenue and market-share.

They tried to provide a view of typical timelines and high level approach involved in IPv6 migration considering residential service as an example.

Yu Zhai et al. (2011) [21] confirmed that IPv4 addresses were depleted in IANA and will be soon exhausted in RIR while more clients are pouring into the internet. Then they proposed IVI, an IPv4/ IPv6 translation scheme in order to facilitate resource migration and protocol transition.

Khaldoun Batiha et al. (2011) [8] explained how much we need the IPv6 protocol by explaining the shortage in available IPv4 unallocated addresses and with discussing the new features and services of IPv6 protocol.

Also they mentioned that many differences between IPv4 and IPv6 protocol were declared such as packet header, addressing schema, and security issues, and they explained how these features and others give advantage for IPv6 over IPv4 which make it much flexible protocol.

On other hand, they explained some of the IPv6 protocol limitation like using existing IPv4 infrastructure to carry IPv6 packet and how servers should deal with IPv4 and IPv6 stack in the same time.

Yuan –Yuan LU (2011) [14] explained how the transition from IPV4 to IPV6 is a necessary process in the realization of global Internet, with the development of IPV6 technology and continuous increase in application, and a case was taken as an example, deals with the transitional technology from IPV4 to IPV6 and the transitional program of IPV6 campus network.

Kuobin Dai (2011) [17] focused in his paper on the excessive cost of programs and methods to achieve the minimum, and he discussed how we need to minimize the cost of a smooth transition to IPv6 networks. He predicted that transition to IPv6 networks will be a long process. He also discussed two transition technologies which currently solve the transition problem : dual stack and tunneling.

Li Zimu et al. (2012) [18] produced a new transition mechanism called IVI, and they mentioned that "There are many transition methods already exist today. This paper gives a brief description of a new mechanism called IVI to help ISP's transition from IPv4 to IPv6. IVI also has some shortcomings. For instance, it depends on DNS.

It means that if there is any embedded IPv4 or IPv6 address in an URL, IVI would not translate properly. Moreover, some protocol packets, like ICMP, should be rewritten when they pass through the Translator.

However, currently, it seems that IVI still is a fire good way for ISP's server transition for its features of stateless, as independent and incremental deployment.

1.4 IPv6 Deployment

In section 1.3 we displayed a table for IPv6 deployment in five continent, this survey had been made in 2009, in this section we will display results for IPv6 survey in February 2013. The results shown in table 1.2 [31].

Table1.2: IPv6 deployment in five continents (by February 2013)

Continent	Deployment Ratio
Europe	17.23%
North America	11.89%
Asia	19.55%
Africa	14.50%
South America	15.5%

IPv6 deployment has increased in last three years, but it still like island in ocean comparing with IPv4 deployment.

1.5 Statement of the Problem

Encapsulation mechanisms are widely- used until now for deploying IPv6, but all encapsulation mechanisms suffer from the increasing of the overhead traffic network as a result for either encapsulating IPv4 packet in the IPv6 packet or encapsulating IPv6 packet in the IPv4 packet (especially if the IPv4 packet is fragmented) [11][17].

Figure 1.2 shows how IPv6 packet encapsulated as a data in IPv4 packet; as a result the packet size will increase so network will suffer from increasing overhead traffic.

Because of that we will focus in this thesis on a transition mechanism that will reduce the packet size compared with encapsulation.

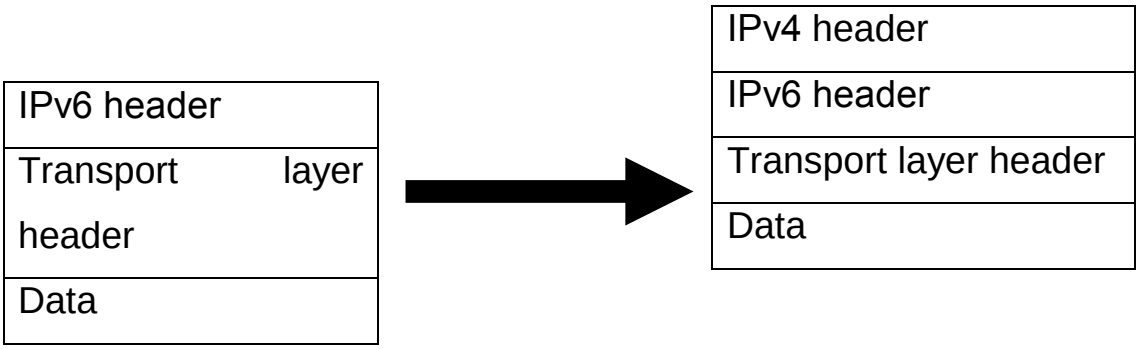


Figure1.2: Encapsulating IPv6 in IPv4.

1.6 Objectives of this Thesis

The aim of this thesis is to propose a transition mechanism between IPv4 and IPv6 that could use in migration process from IPv4 to IPv6.

The other aim is to make incompatible nodes; the first based IPv4 the other based IPv6, communicate together .

1.7 Thesis Structure

In addition to chapter one, this thesis includes the following chapters: Chapter Two where we discuss IP addresses and transition methods from IPv4 to IPv6 like: Dual Stack and Tunneling. The new transition mechanism is described in Chapter Three. In Chapter Four we discuss the results of testing. Finally in Chapter Five, based on the results obtained, some conclusions are drawn, and future work is pointed out.

Chapter Two
IPv4/IPv6 Transition Methods

2.1 Introduction

The main reason for designing IPv6 was the need to increase the number of addresses, and, as expected, IANA's pool of IPv4 addresses has been exhausted in February 2011. IPv6 has been standardized for a long time, but now it’s the solution of next generation Internet address scheme. But because IPv4 and IPv6 are incompatible, the transition process will not be quick and will need a long time [18].

Coexistence of the two protocols together will create problems and slows down the process of transition. So many transition mechanisms have been proposed, some of them used in tested real network, the others have remained so far only as suggestions that have not been adopted. The migration from IPv4 to IPv6 will take a long time, as it is impossible to stop the internet, migrate all the systems and restart all the networks again.

In this chapter IPv4 and IPv6 will explained in detail, and some transition mechanisms will explained briefly. The conversion process will be divided into three categories : dual-stack, encapsulation and translation.

2.2.IPv4 Packet Header Structure

In this section IPv4 packet header structure will be explained in details. In Figures 2.1 we represent the structures of the headers formats for IPv4.

<table>
<tr><td>Version (4)</td><td>THL(4)</td><td>Type of Service (8)</td><td>Total L (16)</td></tr>
<tr><td colspan="2">Identification (16)</td><td>Flags (3)</td><td>Fragment Offset (13)</td></tr>
<tr><td>Time to Live (8)</td><td>Protocol (8)</td><td colspan="2">Header Checksum (16)</td></tr>
<tr><td colspan="4">Source IPv4 Address (32)</td></tr>
<tr><td colspan="4">Destination IPv4 Address (32)</td></tr>
<tr><td colspan="4">Options + Padding</td></tr>
</table>

Figure 2.1: The IPv4 header format [5].

The 'version' field is a 4-bit field that designates the internet protocol version number of the packet; it contains the value 4, which in binary is “0100”. This field is important for routing since IPv4 messages must be handled differently than IPv6 messages.

The Header Length field (4 bits) indicates how long the header is, in 32 bit “words”. The minimum value is “5” which would be 160 bits, or 20 bytes. The maximum length is 15, which would be 480 bits, or 60 bytes.

The Type of Service field (8 bits) . This is used to implement a fairly simple QoS (Quality of Service). QoS involves management of bandwidth by protocol, by sender, or by recipient.

The Total Length field (16 bits) contains the total length of the packet, including the packet header, in bytes. The minimum length is 20 (20 bytes of header plus 0 bytes of data), and the maximum is 65,535 bytes (since only 16 bits are available to specify this). All network systems must handle packets of at least 576 bytes, but a more typical packet size is 1508 bytes. With IPv4, it is possible for some devices (like routers) to fragment packets (break them apart into multiple smaller packets) if required to get them through a part of the network that can’t handle packets that big. Packets that are fragmented must be reassembled at the other end. Fragmentation and reassembly is one of the messy parts of IPv4 that got cleaned up a lot in IPv6.

The Identification (Fragment ID) field (16 bits) identifies which fragment of a once larger packet this one is to help in reassembling the fragmented packet later. In IPv6 packet fragmentation is not done by intermediate nodes, so all the header fields related to fragmentation are no longer needed.

The next three bits are flags related to fragmentation. The first is reserved and must be zero. The next bit is the DF (Don't Fragment) flag. If DF is set, the packet cannot be fragmented (so if such a packet reaches a part of the network that can't handle one that big, that packet is dropped). The third bit is the MF (More Fragments) flag. If MF is set, there are more fragments to come. Unfragmented packets of course have the MF flag set to zero.

The Fragment Offset field (13 bits) is used in reassembly of fragmented packets. It is measured in 8 byte blocks. The first fragment of a set has an offset of 0. If you had a 2500 byte packet and were fragmenting it into chunks of 1020 bytes, you would have three fragments as shown in Table 2.1:

Table 2.1: Fragmentation of 2500 byte packet [19]

Fragment ID	MF Flag	Total Length	Data Size	Offset
1	1	1020	1000	0
2	1	1020	1000	125
3	0	520	500	250

The Time To Live (TTL) field (8 bits) is to prevent packets from being shuttled around indefinitely on a network. It was originally intended to be lifetime in seconds, but it has come to be implemented as "hop count". This means that every time a packet crosses a switch or router, the hop count is decremented by one. If it reaches zero, the packet is dropped. Typically if this happens, an Internet Control

Message Protocol (ICMPv4) message (“time exceeded”) is returned to the packet sender. This mechanism is how the trace route command works. Its primary purpose is to prevent looping (packets running around in circles).

The Protocol field (8 bits) defines the type of data found in the data portion of the packet. Protocol numbers are not to be confused with ports. Some common protocol numbers are:

(1) Internet Control Message Protocol (ICMP)

(6) Transmission Control Protocol (TCP)

(17) User Datagram Protocol (UDP)

The Header Checksum field (16 bits). The 16-bit one’s complement of the one’s complement sum of all 16 bit words in the header. When computing, the checksum field itself is taken as zero. The IP Header Checksum was eliminated in IPv6.

The Source Address field (32 bits) contains the IPv4 address of the sender (may be modified by NAT).

The Destination Address field (32 bits) contains the IPv4 address of the recipient (may be modified by NAT in a reply packet). Options (0 to 40 bytes) not often used.

Data – (variable number of bytes) the data part of the packet – not really part of the header. Not included in the IP Header checksum. The number of bytes in the data field is the value of ‘Total Length’, minus the value of ‘Header Length’[19].

2.3 IPv6 Packet Header Structure

The IPv6 header format is greatly simplified in comparison to the IPv4 format. This is due to the removal of several fields and the addition of the IPv6 extension headers. Figure 2.2 represents the structure of header format for IPv6.

Version (4)	Traffic Class (4)	Flow label (24)
Payload Length (16)	Next Header (8)	Hop Limit (8)
Source IPv6 Address (128)		
Destination IPv6 Address (128)		

Figure 2.2: The IPv6 header format [5].

The IP Version field (4 bits) contains the value 6 which in binary is “0110”. This field allows IPv4 and IPv6 traffic to be mixed in a single network.

The Traffic Class field (8 bits) Available for use by originating nodes and/or forwarding routers to identify and distinguish between different classes or priorities of IPv6 packets, in a manner virtually identical to that of IPv4 “Type of Service”.

The Flow Label field (20 bits) is something new in IPv6. It can be used to tag up to 220 (1,048,576) distinct traffic flows, for purposes such as fine grained bandwidth management (QoS). Its use is still

Experimental. Hosts or routers that do not support this function should set it to zero when originating a packet, or ignore it when receiving a packet.

The Payload Length field (16 bits) is the length of the IPv6 packet payload in bytes, not counting the standard packet header (as it is in IPv4 Total Length), but counting the size of any extension headers, which don't exist in IPv4, the packet extension headers is the first part of the data field (payload) of the IPv6 packet.

The Next Header field (8 bits) indicates the type of header immediately following the standard IPv6 packet header. It uses the same values as the IPv4 Protocol field, If this value contains the code for TCP, then the TCP header and packet payload (data) begins immediately after the IPv6 packet header. Otherwise, one or more IPv6 extension headers will be found before the TCP header and data begins. Since each extension header has another Next Header field (and a Header Length field), this constitutes a linked list of headers before the final extension header, which is followed by the data. UDP packets can also have extension headers.

The Hop Limit field (8 bits) is to prevent packets from being shuttled around indefinitely on a network. Every time a packet crosses a switch or router, the hop count is decremented by one. If it reaches zero, the packet is dropped. Typically if this happens, an ICMPv6 message ("time exceeded") is returned to the packet sender. This mechanism is how the trace route command works.

The Source Address field (128 bits) contains the IPv6 address of the packet sender.

The Destination Address field (128 bits) contains the IPv6 address of the packet recipient.

Data – (variable number of bytes) the data part (payload) of the packet starts immediately after the packet header (it is not really part of the packet header). The contents of the Payload Length field contain the number of bytes in the payload. If there are any extension packet headers, they constitute the first part of the packet payload, and their length is included in the Payload Length field.

As explained above, it can be observed that the following fields from the IPv4 packet header have been eliminated in the IPv6 packet header:

1- Header Length
2- Identification (Fragment ID)
3- Fragmentation Flags
4- Fragment Offset
5- Header Checksum, and
6- Options.

The value in the Payload Length field no longer includes the length of the standard packet header. The Flow Label field had no corresponding field in the IPv4 packet header. Some of the missing fields (e.g. fragmentation information) have been pushed into an extension packet header. These exist only in fragmented packets. Unfragmented packets do not have to carry the unnecessary overhead [25].

2.4 IPv6 Packet Fragmentation and Path Maximum Transmitted Unit (MTU) Discovery

The fields related to fragmentation are now found in the fragmentation extension header of IPv6, which exists only in fragmented packets (no need to clutter up unfragmented packets, as in IPv4). In IPv6, only the originating node can fragment packets (no intervening node is supposed to do this). The originating node uses MTU Path Discovery to determine the “width” of the proposed path (the maximum packet size that it can handle). MTU stands for Maximum Transmitted Unit (maximum packet length). Any packets larger than that size must be fragmented before transmission by the originating node, and reassembled upon receipt by the destination node. There is a default packet size that any IPv6 node must be able to handle (1280 bytes). MTU Path Discovery allows the sender to determine if larger (more efficient) packets can be used. The originating node assumes the Path MTU is the MTU of the first hop in the path. A trial packet of this size is sent out. If any link is unable to handle it, an ICMPv6 Packet Too Big message is returned. The originating node iteratively tries smaller packet sizes until it gets no complaints from any node, and then uses the largest MTU that was acceptable along the entire path. This process takes place automatically in the Internet Layer. There is no corresponding mechanism in IPv4[19]

2.5 Main differences between IPv4 and IPv6

There are many differences between IPv4 and IPv6 and some of them will explained in Table 2.2 [12].

Table 2.2: Differences between IPv4 and IPv6

IPv6	IPv4
Addresses are 128 bits (16 bytes) in length	Addresses are 32 bits (4 bytes) in length.
IPSec support is not optional	IPSec is optional and should be supported externally
Header contains Flow Label field, which Identifies packet flow for QoS handling by router.	Header does not identify packet flow for QoS handling by routers
Routers do not support packet fragmentation. Sending host fragments packets	Both routers and the sending host fragment packets.
Header does not include a checksum.	Header includes a checksum.

Optional data is supported as extension headers.	Header includes options.
Does not require manual configuration or DHCP.	Configured either manually or through DHCP.
Must support a 1280-byte packet size (without fragmentation).	Must support a 576-byte packet size (possibly fragmented).

2.6 IPV4/IPv6 Transition Methods

Many transition mechanisms from IPv4 to IPv6 and vice versa had been proposed by researchers; some researchers divided these methods according to the techniques used in the transition to three transition methods: Dual-Stack method, Tunneling method and Translation method as shown in Figure 2.3 [20].

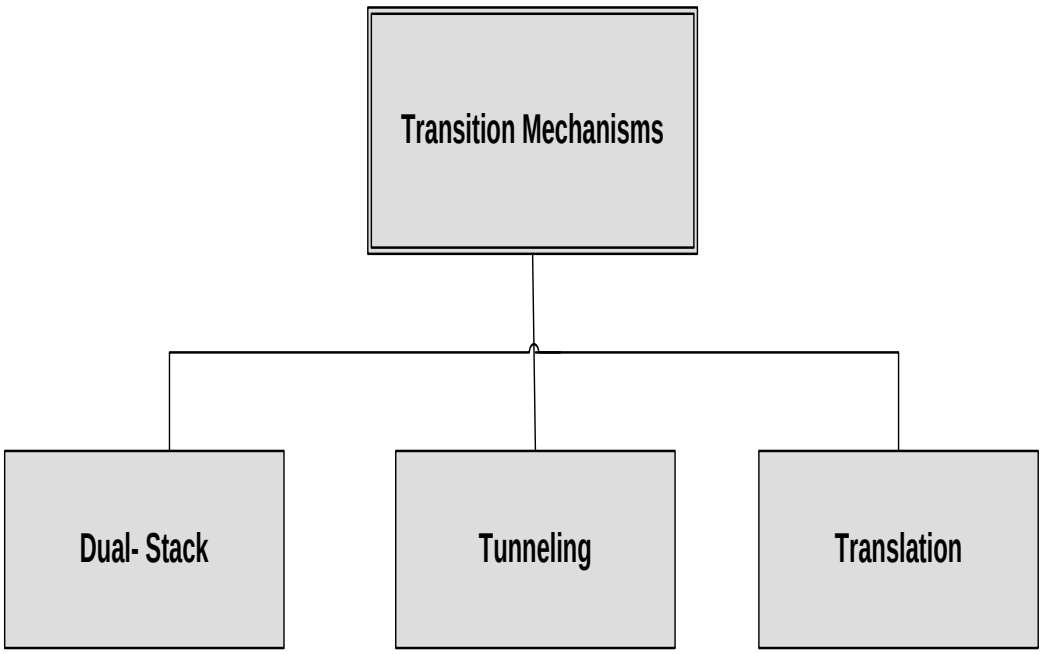


Figure 2.3 : IPv6/IPv4 transition mechanisms [6].

Unfortunately, when the IPv6 is introduced in the Internet, it faces two different sets of problems; the first one is related to having IPv6 communications among two or more IPv6 island which are isolated in the IPv4 world as shown in Figure 2.4. In this case solutions are generally based on dual stack routers and IPv6 in IPv4 tunnels.

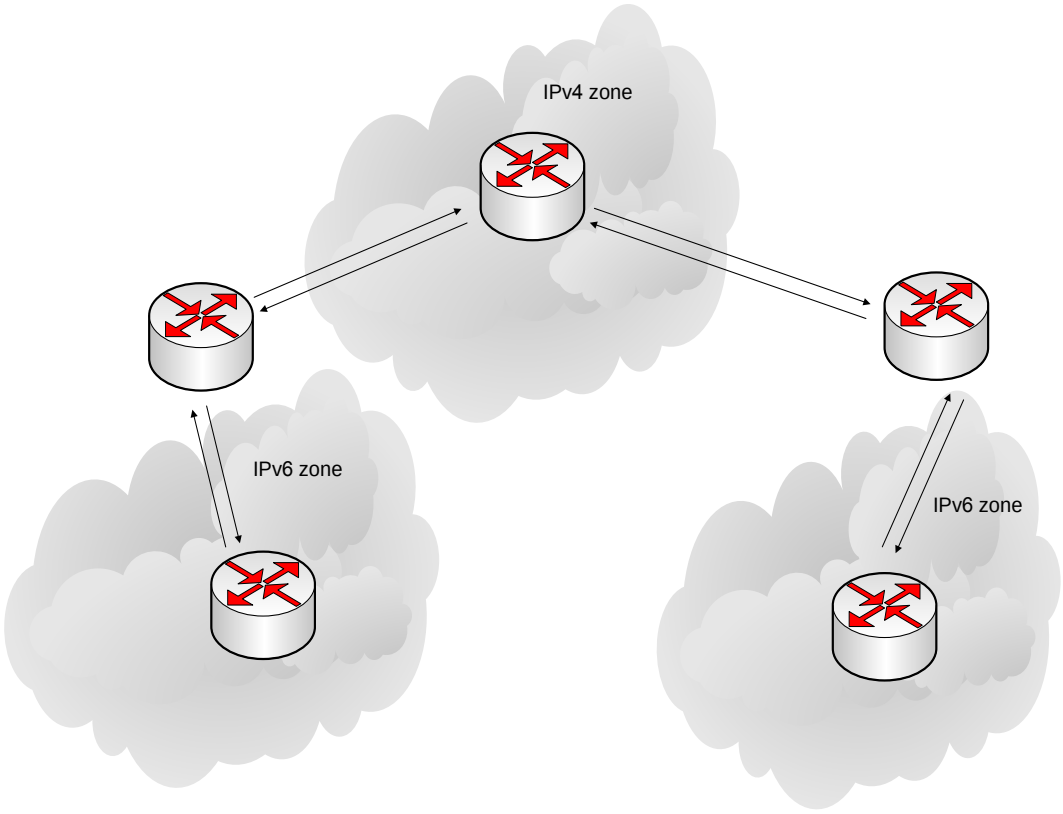


Figure 2.4: IPv6 zones isolated by IPv4 zone.

The second set is related to the establishment of communications between the existing IPv4 world and the new IPv6

world as shown in Figure 2.5. In this case Application Level Gateways will be suitable solution more than dual stack techniques or tunneling techniques [11].

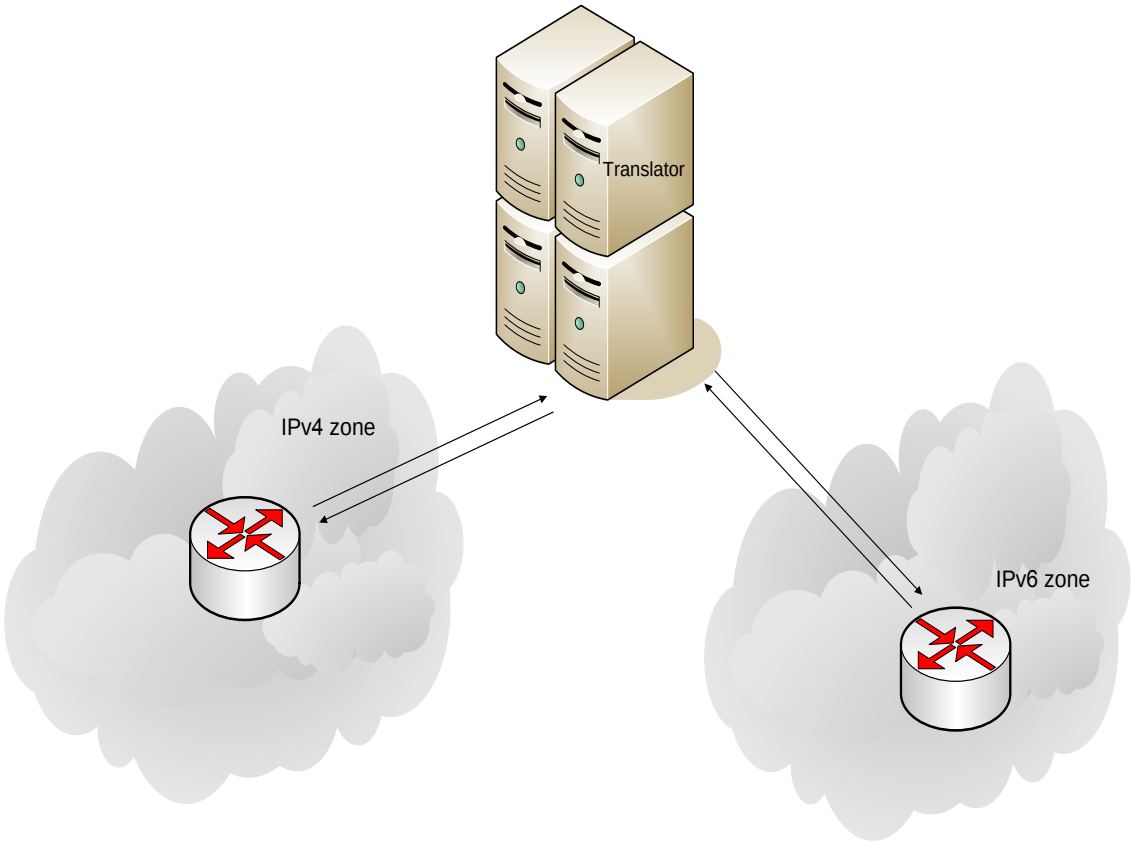


Figure 2.5: IPv6/IPv4 transition through translation mechanism.

Translation mechanism is necessary and will be used if the source host (sender) wants to use IPv6, but the destination host (receiver) does not understand IPv6, so the translation method is needed in order to convert the IPv6 header into an IPv4 header to be understood by the destination host. And in the other case, if the sender wants to use IPv4, but the receiver does not understand IPv4; then translation method is needed in order to convert the IPv4 header

into an IPv6 header, some translation methods had been proposed like NAT-PT technique, but unfortunately, this technique is not so popular within the Internet for reasons that will discussed in section 2.6.3 [5].

2.6.1 IPv4/IPv6 Dual Stack (Dual Stack)

In the dual–stack mechanism, specified in IETF (RFC 2893), a network node includes both IPv4 and IPv6 protocol stacks in parallel as shown in Figure 2.6 [24].

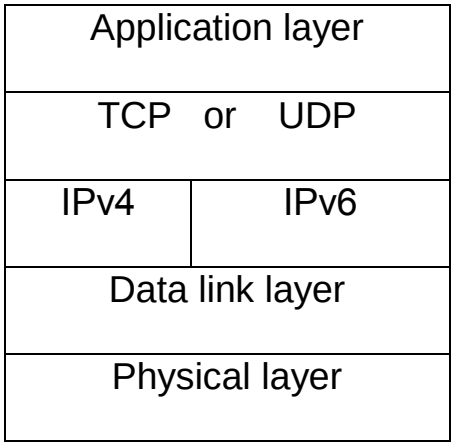


Figure 2.6: IPv4/ IPv6 dual protocol stack structure.

Technical work mechanism of dual stack approach can be simply described as: parsing the received link layer packet data segment, open and check the header; if the first field that is the version number of IP packet is 4, the packet will handled by IPv4 protocol stack processing. But if the version number in the first field of the header is 6 , then it handled by IPv6 protocol stack processing.

Dual stack mechanisms as explained include two protocol stacks that operate in parallel and allow network nodes to communicate either via IPv4 or IPv6. They can be implemented in both end system and network node, which means the network hardware runs IPv4 and IPv6 simultaneously.

Usually, the IPv4/IPv6 dual-stack model is used for transition, but this approach requires that in order to use IPv6, hosts and infrastructure should be upgraded to dual-stack. The requirement cannot be met by old devices that don't support IPv6, and the upgrade of such device would induce a huge cost on the transition scheme, also it will increase the network complexity.

On other hand, there is a shortcoming which is the most important reason for ISPs; it is that for many users who have dual-stack by default such as Win7 users, will get IPv4 and IPv6 name resolves at the DNS inquiry. Since IPv6 is prior to IPv4, IPv6 address is used to access the website first. For the reason that users do not have an IPv6 network environment, the results are often "Not Found" after a long time waiting for IPv6 time out. Obviously, this is not acceptable for ISP.

Other shortcoming for dual stack approach is that servers can be visited from IPv4 and IPv6; this result in firewall rules must be implemented both of IPv4 and IPv6 at the same time. If there is any change of a security policy, both IPv4 and IPv6 firewall rules must be re-considered carefully consistency. It is a hard work for security engineers in reality [18][21][17][13].

2.6.2 Encapsulation approach (Tunneling)

Encapsulation approach enables the connectivity between IPv6 islands by tunneling one protocol over another. In other words, Tunneling mean encapsulating IPv6 packets within the IPv4 packets and passing them through the native IPv4 domains. The tunnels are widely used in nowadays networks; this mechanism can transport IPv6 packets through IPv4 networks as shown in Figure 2.7.

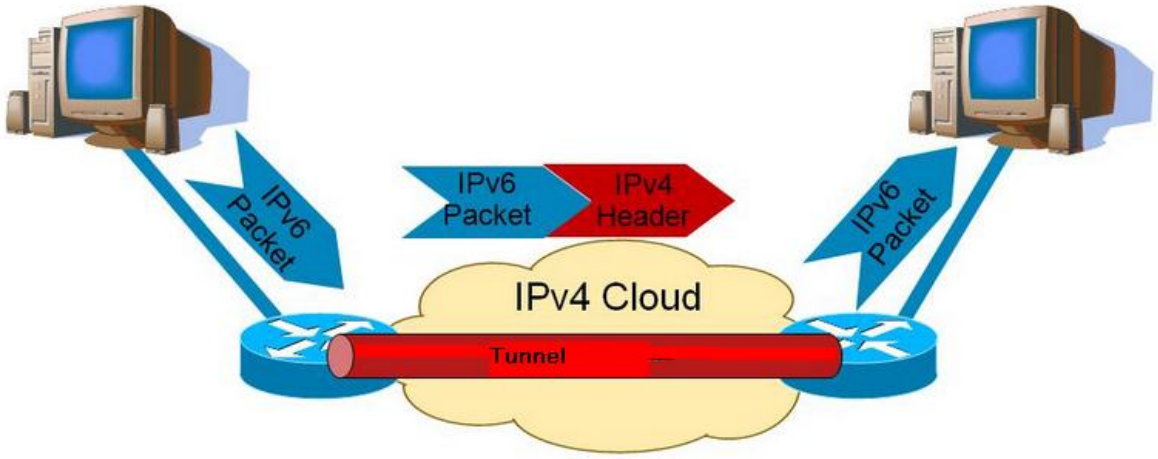


Figure 2.7: Tunneling scenario

Tunneling technology only requires both ends of the tunnel equipment to support both protocols, and each tunnel should be established between two endpoints manually or automatically. This mechanism will be used when two hosts that are using IPv6 want to communicate with each other and they intend to pass their packets through IPv4 zone, as shown in Figure 2.8 IPv6 packet will be as a data in IPv4 packet.

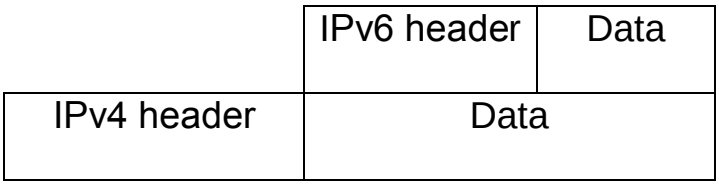


Figure 2.8: Tunneling Encapsulating

The advantages of this method that it enables the island IPv6 end systems and routers to communicate through an existing IPv4 infrastructure, and the IPv6 packets will be put and transported over IPv4 network without being modified. But this method suffers from the increasing of the overhead traffic network, and the shortcoming is in its failure to direct communication between IPv4 and IPv6 nodes [14].

2.6.3 Network Address Translation/Protocol Translation (NAT-PT)

NAT-PT is a technique that allows the communication between only IPv6 and only IPv4 systems. NAT-PT transforms datagram's corresponding field of two kinds of different protocol directly, so as to achieve the purpose which IPv4 and IPv6 can communicate each other. It consists in translating the headers of the packets; but only the common fields of IPv6 and IPv4, and for this reason NAT-PT is not so popular within the Internet [20][5].

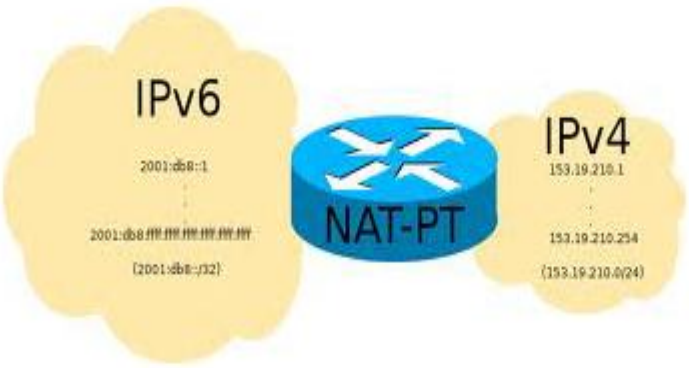


Figure 2.9**:** NAT-PT scenario[28].

Chapter Three
Implemented Transition IPv4/IPv6

3.1 Introduction

Currently the number of available IPv6 applications is very limited compared with huge applications of IPv4; this referred to as an island in the ocean. Therefore, the transition algorithms are the best options for transition from IPv4 to IPv6 and vice versa until every host or router is converted to IPv6. The scope of this research exceeds the encapsulation and tunneling which are nowadays more suitable way to perform transformation and adaptation between IPv4 and IPv6 [5]. As discussed in chapter two there are many transition methods like dual stack and tunneling but these methods have disadvantages which discussed in Sections 2.6.1 and 2.6.2.

However, in this thesis we concentrate on finding an adaptive mechanism for transition between these two versions – IPv4 and IPv6- . The proposed algorithms deal with the mechanism of transformation and adaptation between IPv4 and IPv6 that called Bi-Directional Transition Mechanism (BDTM).

3.2 The Bi-Directional Transition Mechanism (BDTM)

In this thesis a Bi-Directional Transition Mechanism (BDTM) will be proposed which is concentrated on translating and forwarding packets between the different network environments - IPv4 and IPv6 environments- as shown in Figure 3.1.

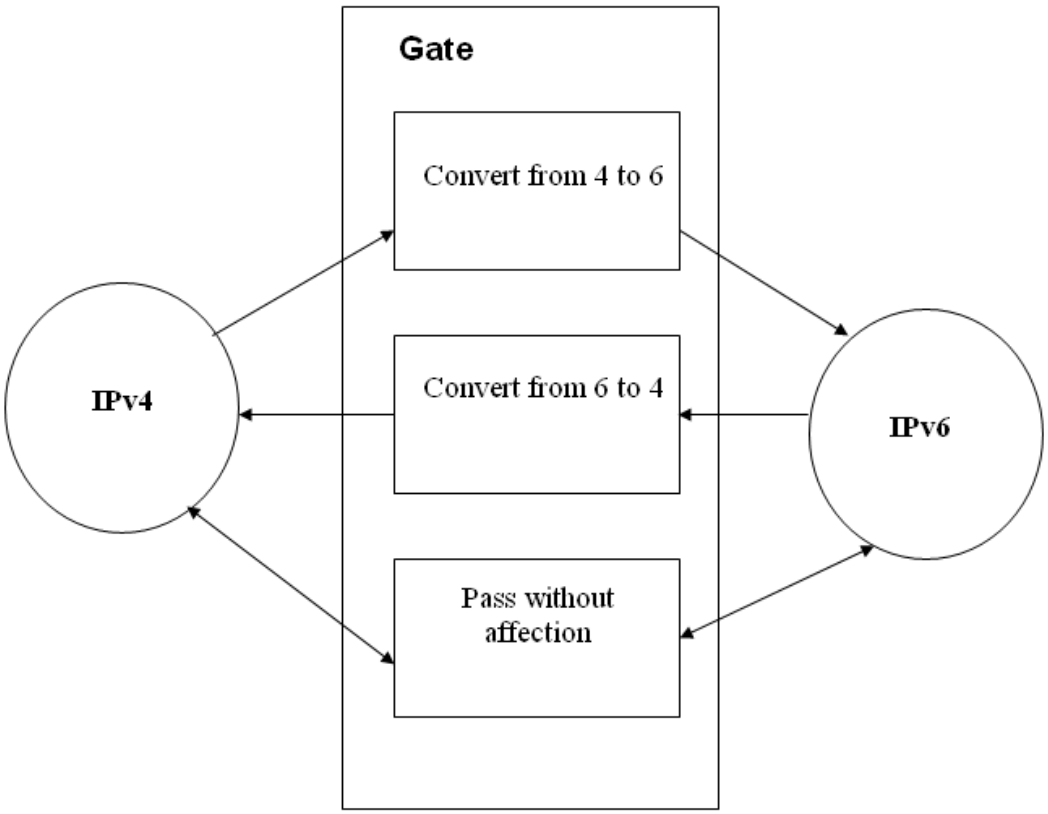


Figure 3.1: Operations in BDTM

The proposed mechanism deals with the deep understanding and analyzing the headers of both technologies IPv4 and IPv6 and the methods for managing the transformation between these technologies. Figure 3.2 shows the header structures for both protocols and the differences between them.

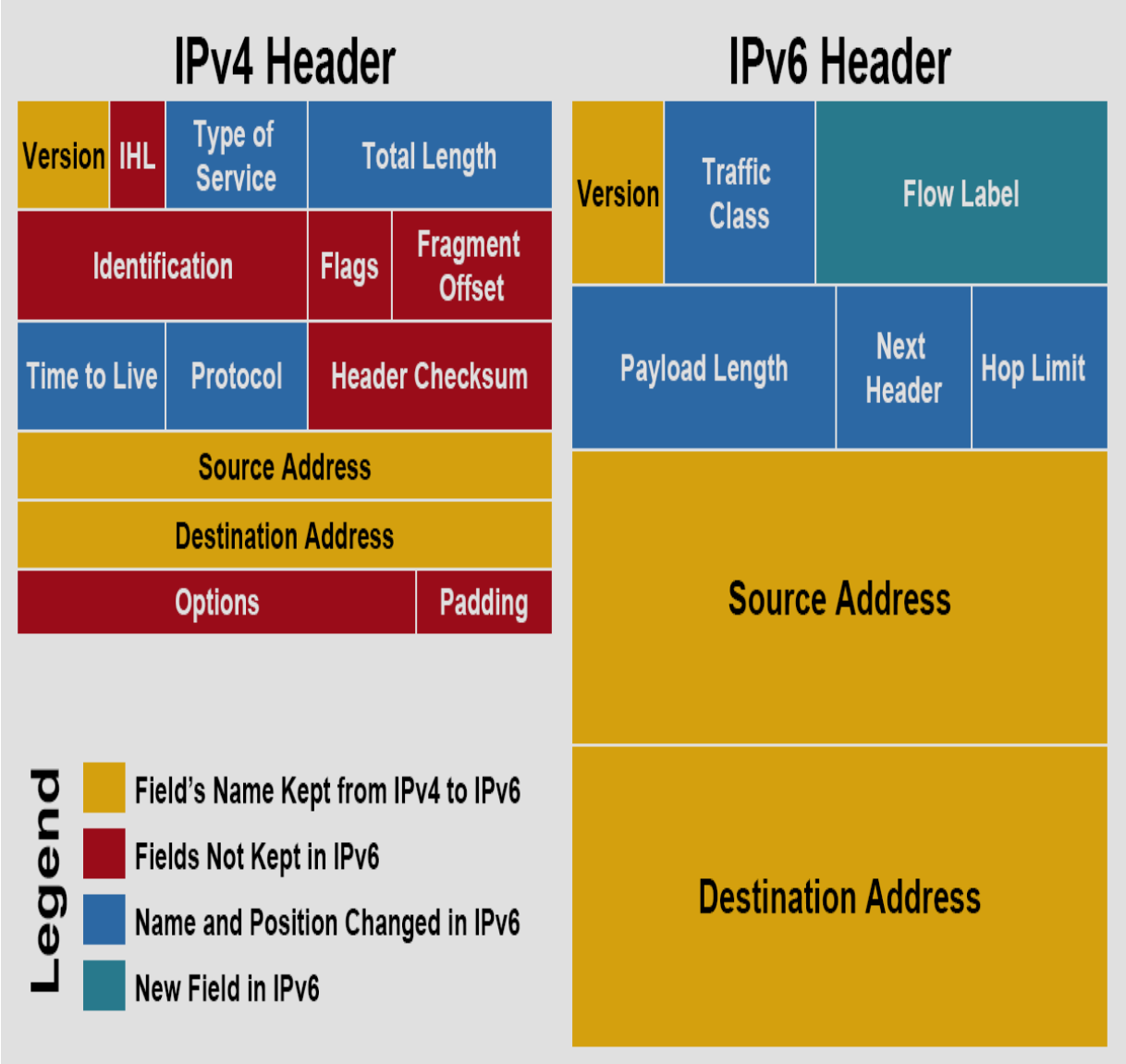


Figure 3.2: Comparison of IPv4 and IPv6 headers' structures [29].

As shown in Figure 3.2, IPv6 header format is simplified in comparison to the IPv4 format, this is due to the removal of several fields, and these fields are:

1- Header length field.
2- Identification field.
3- Flags field.
4- Fragment offset field.
5- Header checksum field.
6- Options field.
7- Padding field.

Some fields are common to both protocols, these fields are:

1- Version field, in the case of IPv4 the “version" field will be equal to (4). While in the case of IPv6 the "version" field will be equal to (6).this field is important for routing since IPv6 message must be handled differently than IPv4 message.
2- Source address field. It is expanded in IPv6 to be (128) bits, while it is (32) bits in IPv4. This change in size is due to the changes in addressing in IPv6.
3- Destination address field. It is also expanded in IPv6 to be (128) bits, while it is (32) bits in IPv6.

Some fields have changes in name, but they are the same function for both protocols, these fields are:

1- Type of Service (IPv4) and Traffic class (IPv6).
2- Time to Live (IPv4) and Hop Limit (IPv6).
3- Protocol (IPv4) and Next Header (IPv6).

4- Total Length (IPv4) and payload (IPv6).

Flow label field is new in (IPv6) , if it is equal to zero then Traffic Class (IPv6) equal to Type of Service (IPv4) and if not then use different level of type of service.

Extension headers (IPv6) is anew way in IPv6 to handle option fields, fragmentation, and security. Some of the missing fields (e.g. fragmentation information) have been pushed into an extension packet header. These exist only in fragmented packets. Unfragmented packets do not have to carry the unnecessary overhead.

The proposed mechanism depends on capturing the header, identifying the header, transformation the datagram to the destination environment and then transmitting the datagram to the destination address. This system deals with the bi-directional operation that leads to converting the received datagram to the destination environment.

3.3 BDTM Analyzing

In this section our transition algorithms will be proposed , these algorithms depends on understanding the received datagram, capturing the header, identifying the header, transition the header, and then transmitting the datagram to the destination address. The proposed header processing algorithms deal with the in-depth understanding of the two technologies of the header fields, that is, from IPv6 to IPv4 and vice versa.

3.3.1 Transition from IPv4 Header to IPv6 Header

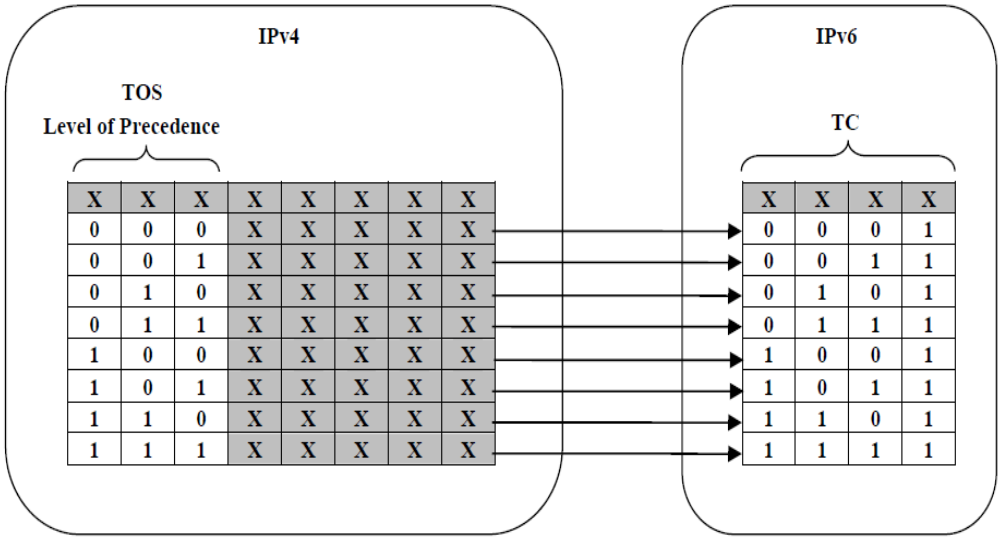
The header processing proposed algorithm deals with understanding of the two structures of the header fields. Figure 3.4 shows the algorithm of the header processing transition from IPv4 to IPv6, the process of transition are listed below:

Process 1: version transformation, inserting (0110) in the version field of IPv6 to denote that the IP version is 6.

Process 2 : computing payload length , it will be computed by subtracting the contents of the “header length “ field of IPv4 header from the contents of the “total length “ field of IPv4 , then , the resulted value will be saved in the “payload length” field in IPv6 [5].

Process 3: inserting Flow Label field, this will done by copping the content of Identification Field of IPv4 in Flow Label field of IPv6.

Process 4 : filling the Traffic Class field of IPv6, by mapping the content of the TOS field of IPv4 to the TC field of IPv6 as shown in Figure 3.3 .



X	X	X	X	X	X	X	X
0	0	0	X	X	X	X	X
0	0	1	X	X	X	X	X
0	1	0	X	X	X	X	X
0	1	1	X	X	X	X	X
1	0	0	X	X	X	X	X
1	0	1	X	X	X	X	X
1	1	0	X	X	X	X	X
1	1	1	X	X	X	X	X

X	X	X	X
0	0	0	1
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	1

Figure 3.3: Mapping the contents of the TOS to TC [5]

Process 5 : checking the Fragment Offset field of IPv4 .However , if the Fragment Offset field is not equal zero then process 6 will done before process 7 and 8 .Else process 7 and 8 will done without doing process 6.

Process 6: coping the contents of the “Flag” and “ Fragment Offset “ fields of IPv4 header to the corresponding Fields in the “ Fragment Extension Header “ of IPv6.

Process 7: copping the contents of “Time To Live” field of IPv4 header to the “Hop Limit “ field of IPv6.

Process 8 : convert the” IPv4 Destination Address and Source Address” to “IPv6 Destination Address and Source Address “ then save the result in IPv6 Destination Address and Source Address fields.

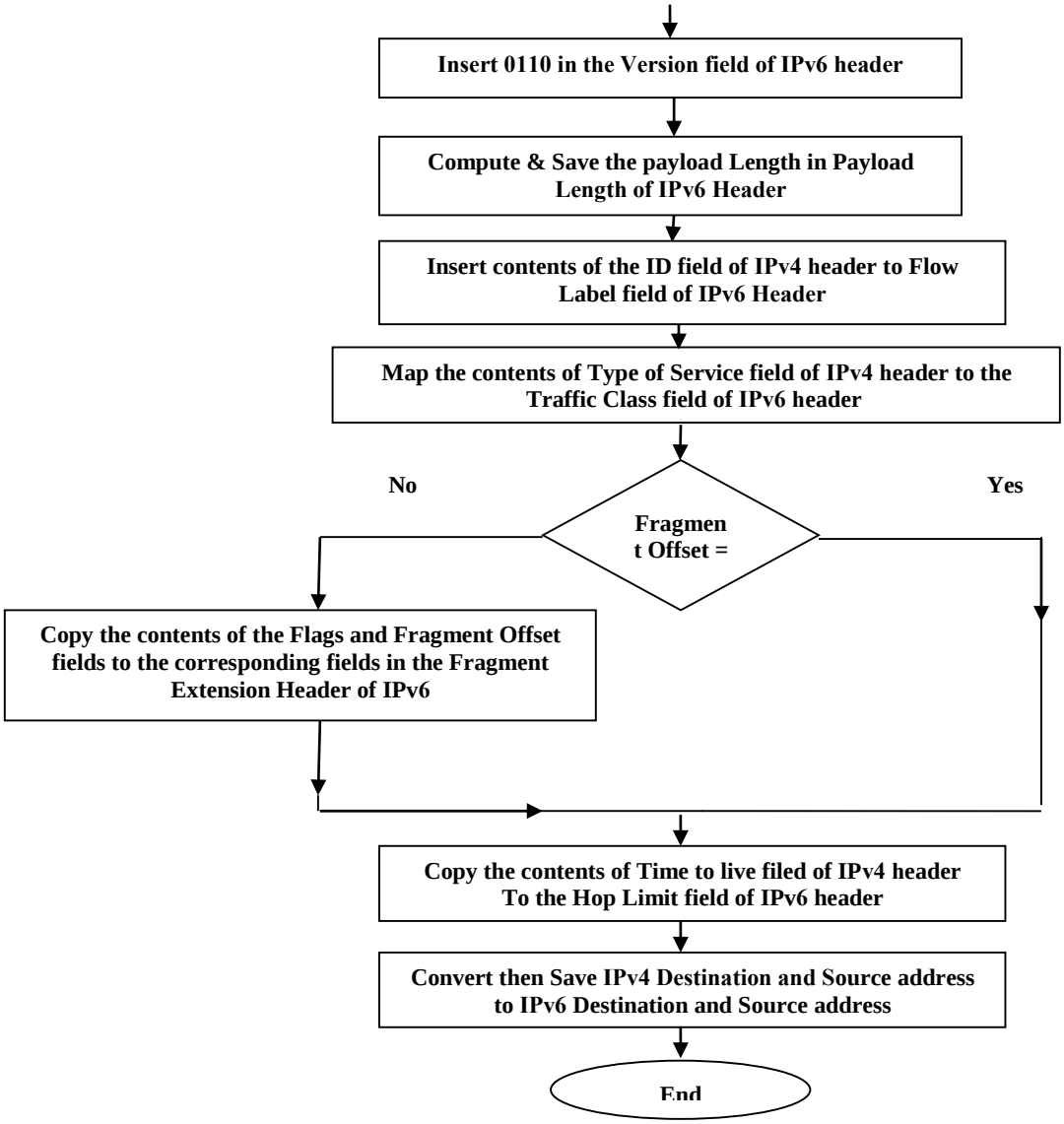


Figure 3.4 : The algorithm of the header processing transition from the IPv4 to IPv6

3.3.2 Transition from IPv6 Header to IPv4 Header

Figure 3.6 shows the detailed algorithm of the transition process from IPv6 header to IPv4 header. Again, some fields of the IPv4 header will contain the same values from the corresponding fields of the IPv6 header. The main process of transition is listed below:

Process 1: version transformation, inserting (0100) in the version field of IPv4 to denote that the IP version is 4.

Process 2: mapping the content of TC field in IPv6 header to the TOS field in IPv4 header as explained in Figure 3.5.

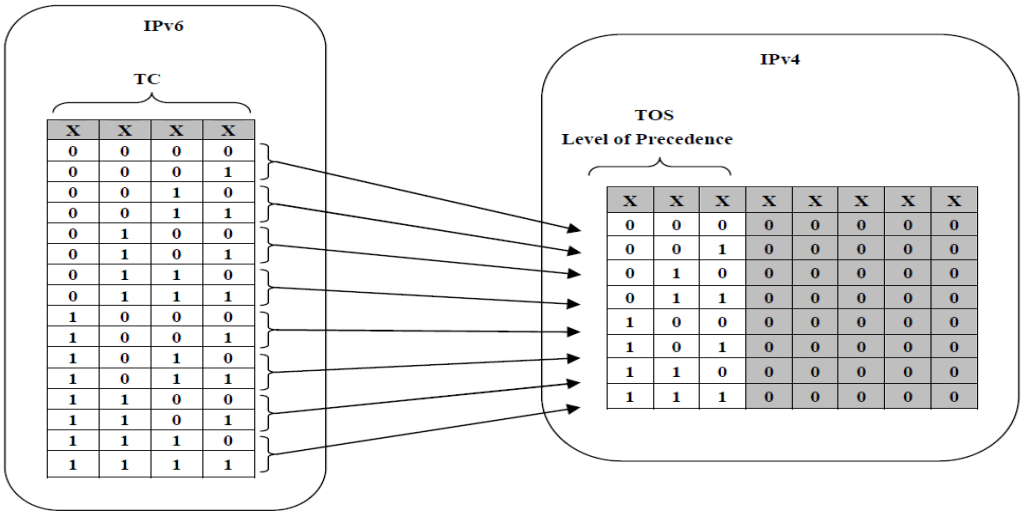


Figure 3.5: Mapping the contents of the TC to TOS [5].

Process 3: copping the content of Next Header field of IPv6 to the Protocol field in IPv4.

Process 4: copping the contents of Hop Limit field in IPv6 header to the Time To Live field in IPv4.

Process 5: compute the header length and save the result in THL field in IPv4 header.

Process 6: compute the Total Length for IPv4 header by adding the value of Payload Length of IPv6 header to the result that computing in process 5 then save the result in Total Length field for IPv4.

Process 7: convert the" IPv6 Destination Address and Source Address" to "IPv4 Destination Address and Source Address" then save the result in IPv4 Destination Address and Source Address fields.

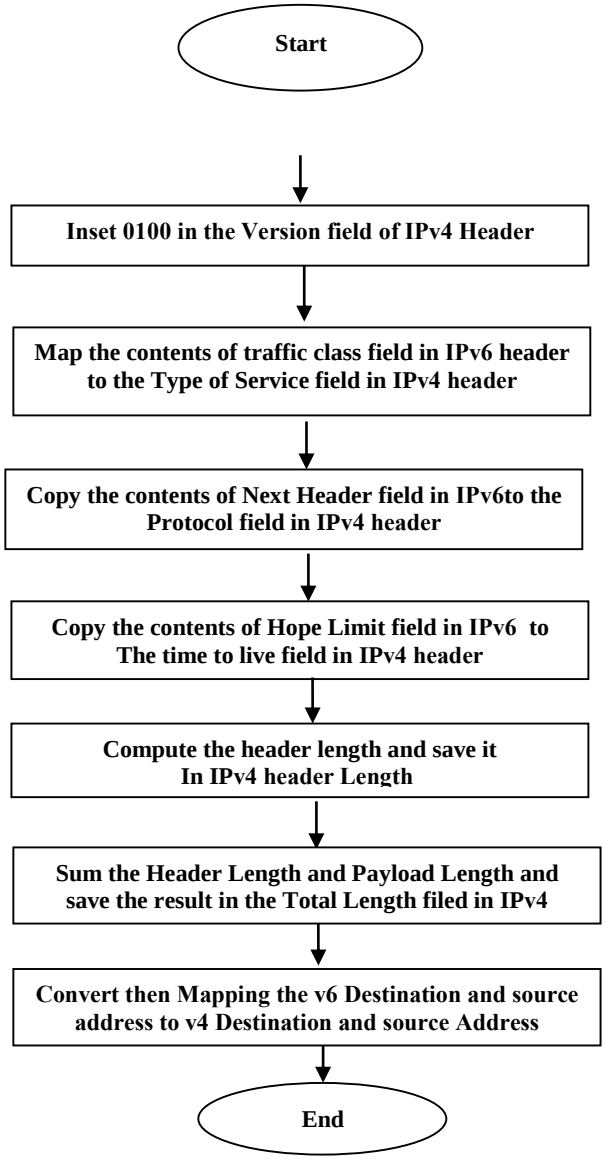


Figure 3.6 : The algorithm of the header processing transition from the IPv6 to IPv4

3.4 Implementation

As mentioned before, our proposed mechanism deals with the deep understanding and analyzing of the headers for both technologies; IPv4 and IPv6 implement this Mechanism; BDTM, depends on the two algorithms that proposed in sections 3.3.1 and 3.3.2 .

3.4.1 Implementation of BDTM

The implementation of BDTM contains three applications that are listed below:

1. Sender Application.
2. Gate Application.
3. Receiver Application.

Both Sender and Receiver Applications must exist in the sender and receiver nodes.

3.4.1.1 Sender Application

The function of this application is to identify the source address and destination address and to select the files that will be sent. The sender interface represents the sender application our implementation. This interface has many objects, as shown in Figure 3.7 , the functions of these objects are listed below:

1. Public IP text: this text box requests the public IP address for the current network and that by "http://automation.whatismyip.com/n09230945.asp". website.
2. The name of the sender : name for sender computer by using microsoft library "myIP.ToString" .
3. The IP address of the sender: identify the IP address for the sender by using microsoft library "myIPs.HostName".
4. IP address for receiver : it filled by user .
5. Txt message : filled manually by user ; the user can insert any text he want.
6. Port No: inserting port number; this is not mandatory step. If this step has been selected, it is determined. The port that will be used to send data and to increase the degree of safety sometimes, to speed up the transmission of data , or for other reasons may be of interest to the user. Figure 3.8 shows the interface for changing the port.
7. Browse: to select file that will be send with message to the receiver.
8. Send Button: to send data and IP address to gate application then to destination device.
9. Exit Button : pressing this button will come out of the screen.

Sender application sends the request to distination device and gets the feed back if areceiver applies and ready to receive data.

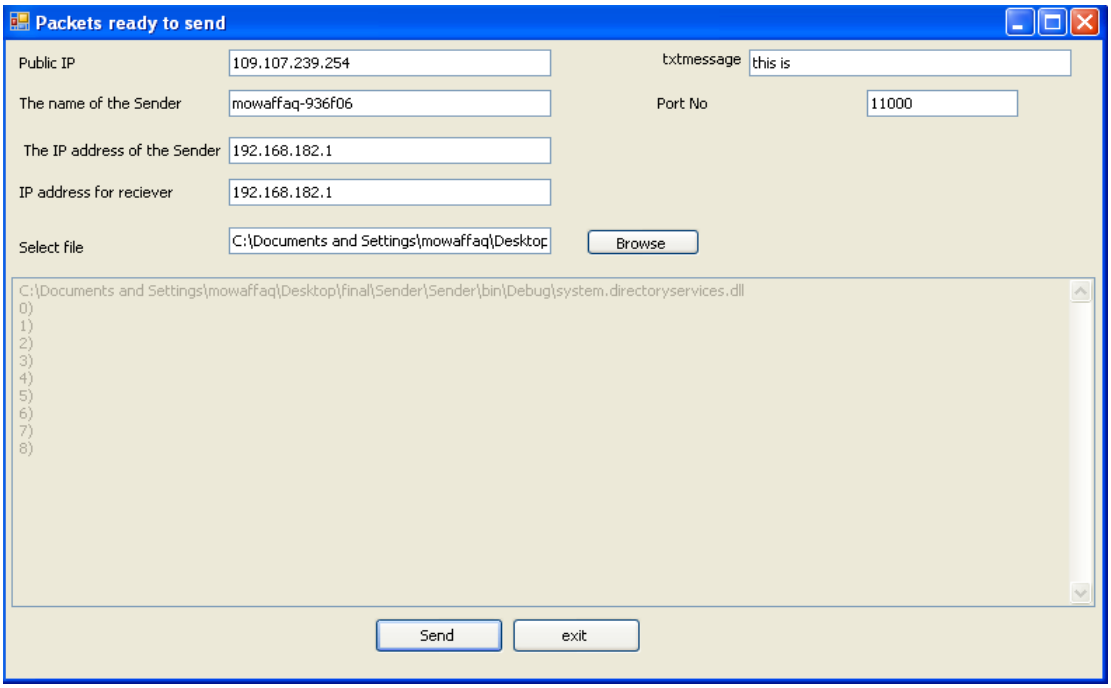


Figure 3.7: Sender Interface

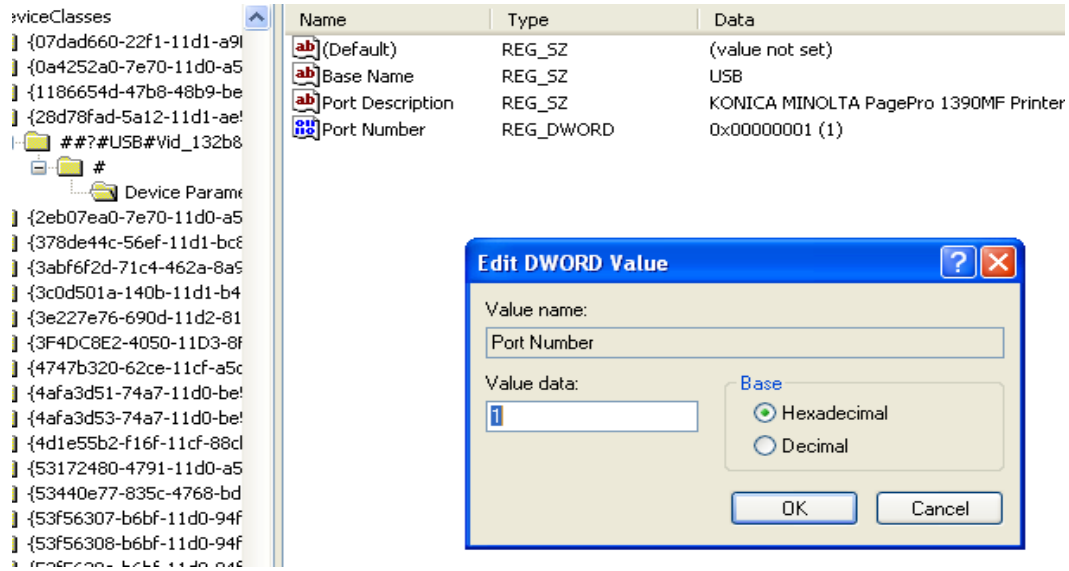


Figure 3.8 : Determine the port number

3.4.1.2 Gate Application

The Gate application used to process the algorithms of (BDTM) according to the sender and receiver node. These algorithms were discussed in sections (3.3.1) and (3.3.2) in details.

3.4.1.3 Receiver Application

The function of this application is receiving the packets that had been sent then store it in the receiver device. The receiver interface represents the receiver application in (BDTM). This interface has many objects as shown in Figure 3.9; the functions of these objects are listed below:

1. Public IP: this text field display the public IP for the receiver network and that by "http://automation.whatismyip.com/n09230945.asp" website.
2. IP address for receiver : its value determine by using microsoft library "myIPs.HostName".
3. The IP address of the sender: that by receiving the request from sender application.
4. Start: to start recieving data .
5. Save : this choice used to save the received data as shown in Figure3.10.

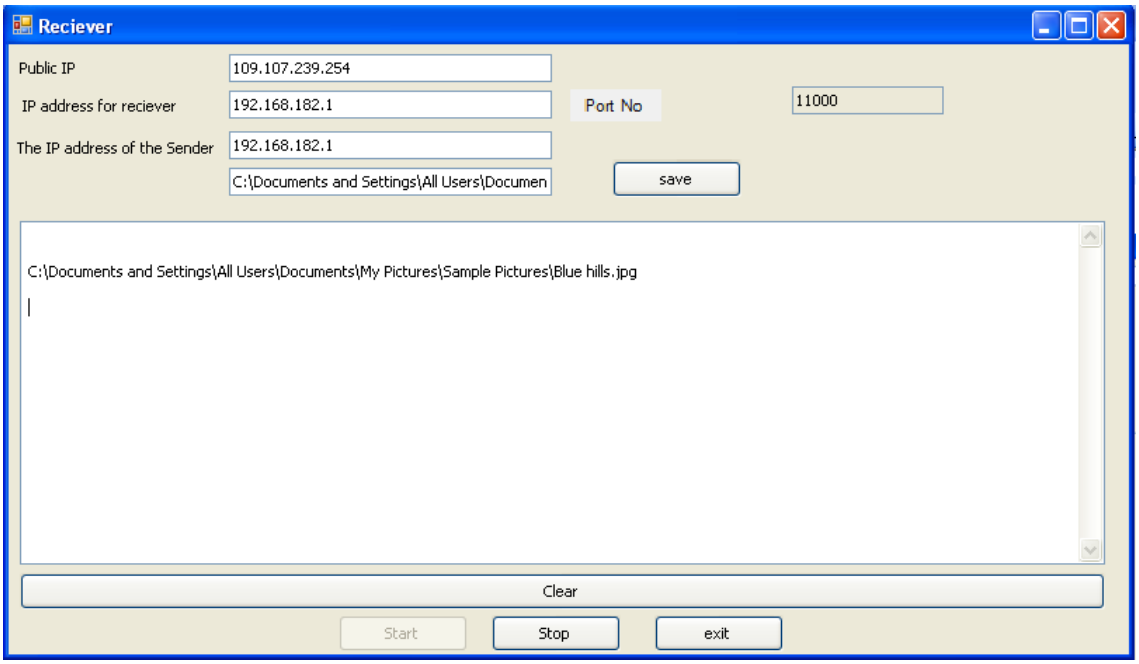


Figure 3.9 : Reciever Interface

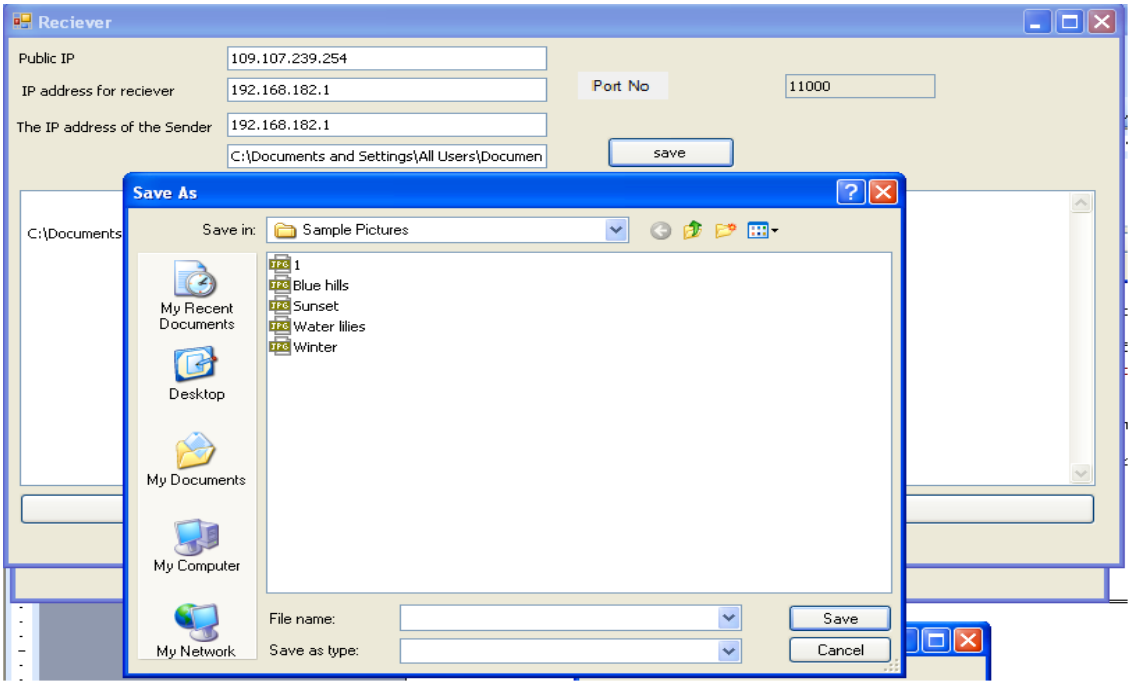


Figure 3.10 : Saving received file

3.4.2 Steps of running BDTM implementation

Our implementation is very simple to use. But the user should download the sender and receiver applications in his device to use this implementation.

Steps of the implemented system are ordered as below:

1- When the user wants to send file to destination environment he will browse the files.

2- Next step the user will insert IP address of destination device in the IP address of the receiver field.

3- The sender application will get the IP address automatically for the current device and save it into "IP address of the sender field".

4- User will press start (the application send request message to gate application to convert the packets).

5- Gate application will verify the address of sender and receiver, converting the packet then transforming the packets to receiver destination.

6- Gate application sends new packets to receiver destination.

7- When Gate application send new packets to destination device the Receiver Application will receive the IP address of sender in same version of receiver.

8- Then Receiver Application shows the message box to tell other user if he wants to accept the packets or not. If yes save dialog box will show, else cancels receiving.

Chapter Four
Simulation & Testing

4.1 Introduction

A network simulator is a piece of software or hardware that predicts the behavior of a network without an actual network being present. A network simulator is a software program that imitates the working of a computer network. In simulators, the computer network is typically modeled with devices, traffic etc. and the performance is analyzed [30].

Most of the commercial network simulators are GUI driven, while some network simulators require input scripts or commands (network parameters). The network parameters describe the state of the network (node placement, existing links) and the events (data transmissions, link failures, etc.). An important output of simulations is the trace files. Trace files can document every event that occurred in the simulation and are used for analysis. Certain simulators have added functionality of capturing this type of data directly from a functioning production environment, at various times of the day, week, or month, in order to reflect average, worst-case, and best-case conditions. Network simulators can also provide other tools to facilitate visual analysis of trends and potential trouble spots [30].

In communication and computer network research, network simulation is a technique where a program models the behavior of a network either by calculating the interaction between the different network entities (hosts, packets, etc.) using mathematical formulas,

or actually capturing and playing back observations from a production network. The behavior of the network and the various applications and services it supports can then be observed in a test lab; various attributes of the environment can also be modified in a controlled manner to assess how the network would behave under different conditions. While a simulation program is used in conjunction with live applications and services in order to observe end-to-end performance to the user desktop, this technique is also referred to as network emulation.

There are a wide variety of network simulators, ranging from the very simple to the very complex. Minimally, a network simulator must enable a user to represent a network topology, specifying the nodes on the network, the links between those nodes and the traffic between the nodes. More complicated systems may allow the user to specify everything about the protocols used to handle traffic in a network. Graphical applications allow users to easily visualize the workings of their simulated environment.

4.2 Testing Environment

BDTM has been implemented in visual studio 2005 environment, and a simulation tool has been selected to perform simulation experiment as one of the performance evaluation techniques. This tool is (VMware) tool; it is a representation of a real machine using software that provides an operating environment which can run or host a guest operating system [30].

Once the project has been implemented; the simulation has been done to test this system.

4.3 Simulation Scenario

In this section, a description of the architecture of the simulation environment will be introduced. The scenario given in Figure 4.1 depicts a conversation between two hosts, IPv4 host and IPv6 host.

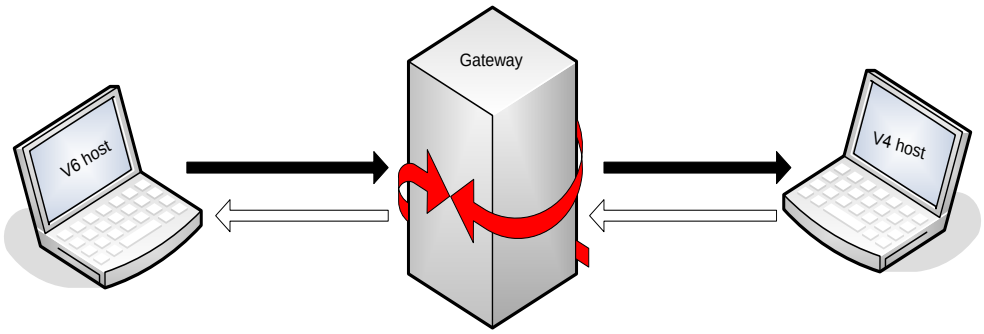


Figure 4.1: Simulation Scenario

As shown in Figure 4.1, our virtual network contains three parts : two hosts and gateway machine; we determined the operating system of the hosts as windows 7, and the operating system of the gateway as Linux 2.6.x kernel.

The test that has been done by order as below:

Step 1: download VMware program.

Step 2: identify (host 1) as IPv4 machine.

Step 3: identify (host2) as IPv6 machine by determining the IP address for the device in version 6, as shown in Figure 4.2.

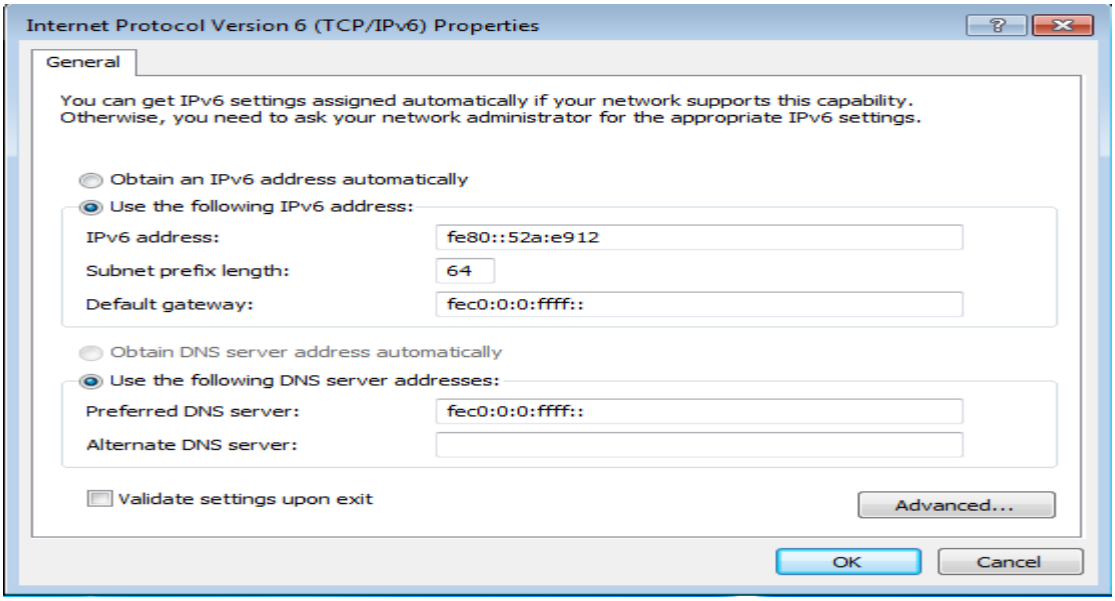


Figure 4.2 : Windows 7 IPv6 Configuration

Step 4: download (BDTM) implementation in (host1) and (host2) as shown in Figure 4.3.

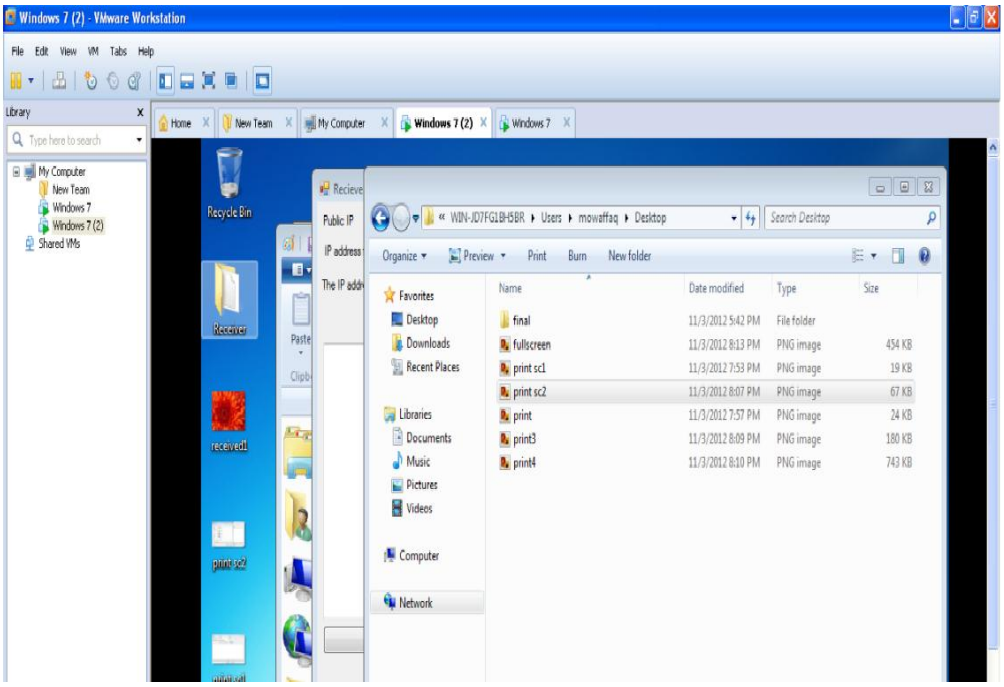


Figure 4.3: Full outside screen for VMware workstation

Step 5: run (BDTM) implementation in (host1)- Main Screen will appear- as shown in Figure 4.4. The Main Screen contains the buttons to start Sender or Receiver applications, and Exit button.

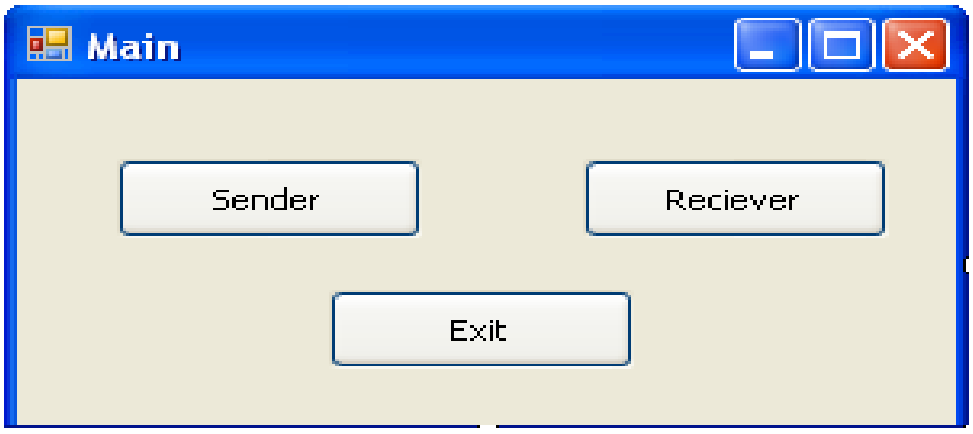


Figure 4.4 : Main Screen of BDTM implementation

Step 6: select (sender) command from the Main Screen.

Step 7: insert the IP address of (host2) –receiver- in the (IP of the receiver) in sender interface.

Step 8: select the file that will be sending from (browse) command as shown in Figures 4.5 and 4.6.

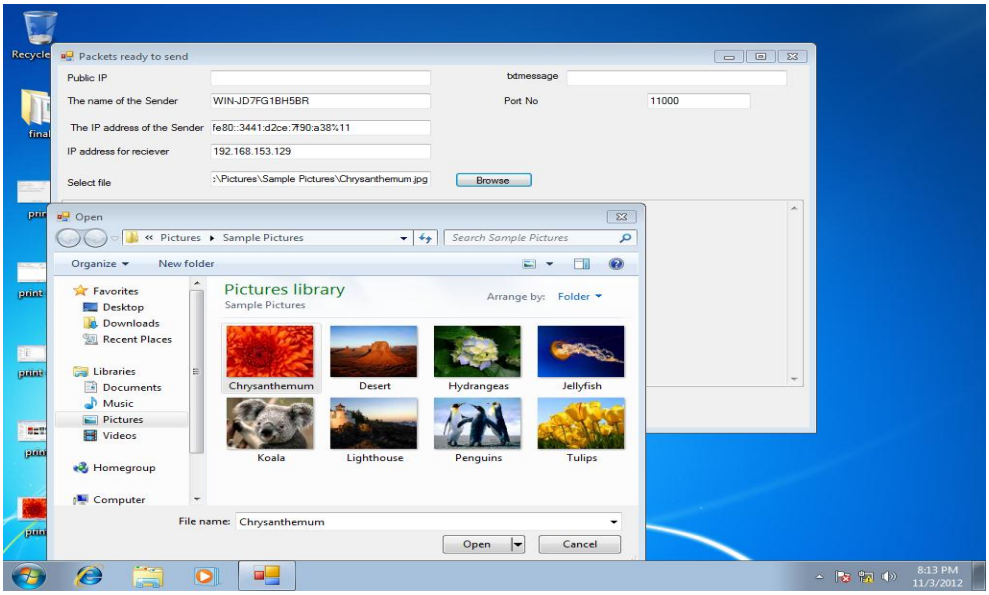


Figure 4.5 : Sending Operation

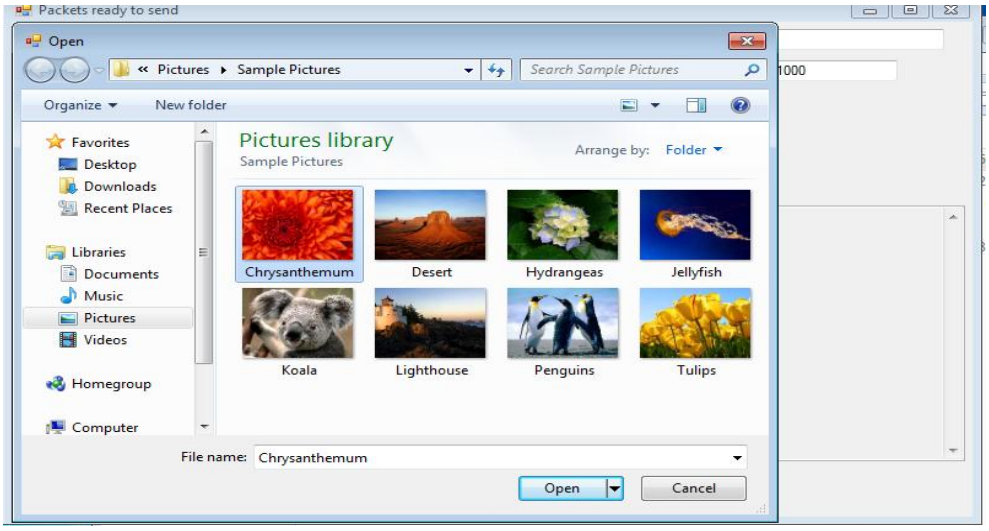


Figure 4.6: Browsing for files

Step 9: press (start) command.

Step 10: run (BDTS) in (host2).

Step11: select (receiver) command from the main menu.

Step 12: press (save) command from the receiver interface.

Step 13: save the file in (host2) as shown in Figures 4.7 and 4.8.

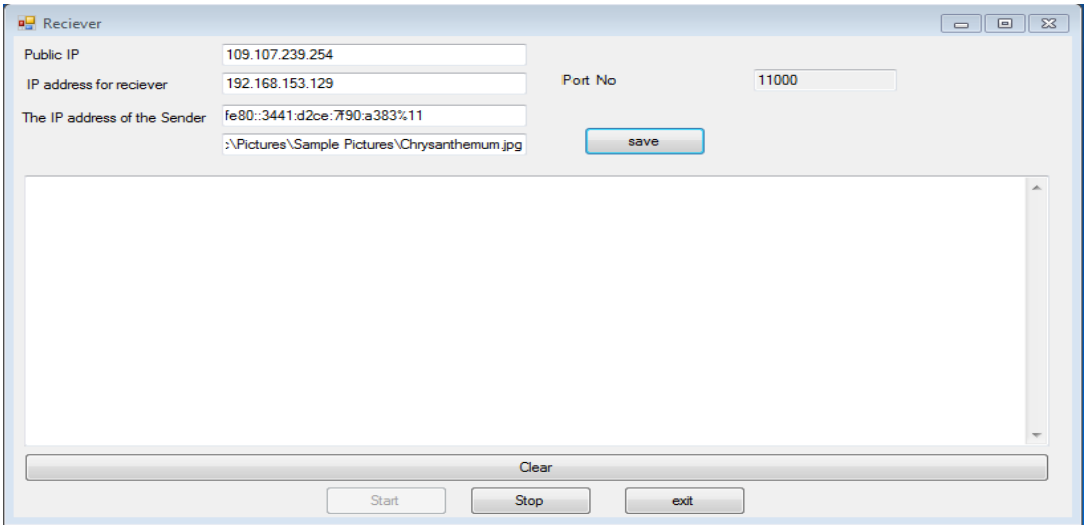


Figure 4.7: Receiving Operation

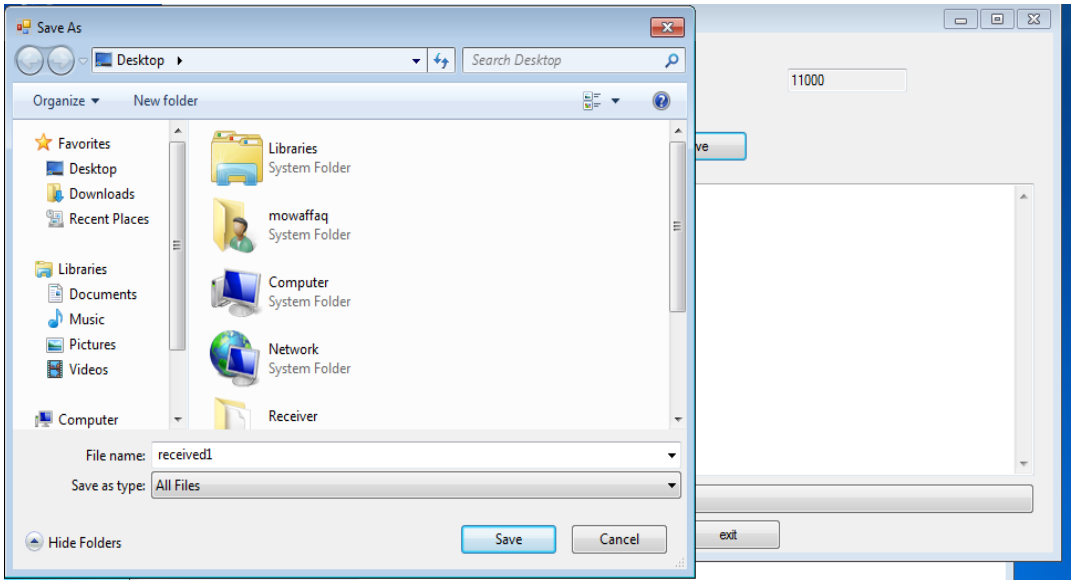


Figure 4.8 : Saving the received file

Step14: open the file to insure that it is the same file that has been sent from (host1), and then we could make a comparison between the properties of the files before sending them and their properties after receiving them to ensure that the receiving file is not corrupted.

Chapter Five

Conclusions & Future Work

5.1 Conclusions

In this thesis a transition mechanism has been proposed. The proposed algorithms deal with the method of transformation and adaptation between IPv4 and IPv6 that called Bi-Directional Transition Mechanism (BDTM) .

In this thesis we proposed algorithms that depend on understanding of the two environment of transmission, that is, received the source packet then converting the information header to be adaptable to the destination end.

The algorithms have been implemented, and then a simulation test had been done by a network simulation tools called VMware. During this simulation, BDTM was studied in one scenario, and the results shown that BDTM could make two incompatible protocol hosts communicate together.

5.2 Future Work

The work reported in this thesis opens the way for continued research in a number of directions. A particularly interesting area for future research is studding security issues while using BDTM as the Internet Protocol Security (IPsec) is mandatory in IPv6 but optional in IPv4 [15], as a result, there will be sending secure data when converting from IPv6 to IPv4, but the reverse may not be true.

References

[1] RFC 791 ,"Internet Protocol DARPA Internet Program Protocol Specification" ,www.tools.ietf .org, 1981.

[2] Vinton.G.Cerf and Robert.E.Kahn " Aprotocol for Packet Network Intercommunication" ,1974 IEEE Reprinted ,with permission, from IEEE Trans on Comms, vol com-22, no 5 May 1974.

[3] R.Glenn, J . Wack , and H.Fang. " project : IPv6 technology ". Technical report, National Institute of Standards and Technology , 1996.

[4] John M.chasser, " Security Concernce in IPv6 and Transision Networks",Information Security Journal : A Global Perspective , 2010.

[5] Al-Kasasbeh, Al-Qutaish,Muhairat, " Innovation Algorithms for the Header Processing Transition from IPv4 to IPv6 and Vice Versa", The International Arab Journal of Information Technology , Vol.7 , No 3 , July 2010.

[6] Muzhir Al-Ani , Basil Kasasbeh , " Efficient Header Processing Transition between IPv4 and IPv4 " ,International Journal of Science Engineering and Technology ,Vol.1 ,No.2 , 2008.

[7] J.Hanumanthappa and Manjaiah D.H , " Astudy on Comparison and Contrast between IPv6 and IPv4 Feature sets",International Conference on Computer Network and Security (ICCNS), 2008.

[8] Khaldoun batiha, Khaled batiha, Amer Abu Ali," The need for IPv6", International Journal of Academic Research, Vol.3. No.3.May 2011.

[9]Laura Denaris , Questioning IPv6 Security , Business Communications Review, July 2006.

[10] Viney Sharma, “IPv6 and IPv4 Security Challenge Analysis and Best- Practice Scenario” , Int.J. of Advanced of Networking and Applications , volume:01,issue: 04,pages : 258-269 , 2010.

[11] Ra'ed AlJa'afreh , John Mellor , Mumtaz Kamala , Basil Kassasbeh, Muzhir Al-Ani , "A novel IPv4/IPv6 Transition Mechanism which Supports Transparent Connections" , ISBN: 1-9025-6016-7© 2007 PGNet.

[12] Xianhuiche, Dylan Lewise ," IPv6 :Current Deployment and Migiration statuse" , International Journal of Research and Reviews in Computer Science (IJRRCS), Vol.1,No.2, June 2010.

[13] D.Shalini Punithavathani , K.Sankaranarayanan, " IPv4/IPv6 Transition Mechanisms", European Journal of Scientific Research , 2009.

[14] Yuan-yuan LU, "Transition from IPV4 to IPV6 Network Application ", International Conference on Intelligense Science and Information Engineering, 2011.

[15] Abidah Hj Mat Taib ,"Security Mechanisms for the IPv4 to IPv6 Transition", The 5th Student Conference on Research and Development –SCOReD ,2007

[16] Srikant A.Vardhana ,Sib S.Roy,Vidyadhar Chakravarthy," IP Upgrade-Engineering Exercise or Necessity?", 2010 Sixth International Conference on Networking and Services

[17] Kuobin Dai, "IPv4 to IPv6 Transition Research Based on the Campus Network" , 2011 International Symposium on Intelligence Information Processing and Trusted Computing.

[18] Li Zimu, Peng Wei, Liu Yujun, "An Innovative Ipv4-ipv6 Transition Way for Internet Service Provider", IEEE Symposium on Robotics and Applications(ISRA), 2012.

[19] Lawrence E. Hughes,"The Second Internet", Infoweapons , 2010.

[20] Chen Yan-ge,Tan Shui-mu, Guo Jun-Ying," IPv4/IPv6 Intercommunication technology Research", International Conference on Environmental Science and Information Application Technology, 2009.

[21] Yu Zhai,Congxiao Bao,Xing Li," Transition from IPv4 to IPv6 : A Translation Approach", Sixth IEEE International Conference on Networking ,Architecture ,and Storage ,2011.

[22] Chen Shengyang, Shang Yanlei ,"P2P Communication Mechanism Based on Request Forwarding in IPv4 and IPv6 Coexistance Network", International Conference on e- Education, e-Business, e-Management and
e-Learning , 2010.

[23] Jesus Ibanez Para,"Comparison of IPv4 and IPv6 Networks Including Concepts for Deployment and Interworking", INFOTECH Seminar Advanced Communication Services (ACS), 2004.

[24] RFC 2893," Transition Mechanisms for IPv6 Hosts and Routers" , August 2000 .

[25] Norshakinah Bin Timd Nasir, “Performance Evaluation of IPsec Implementation in IPv4 and IPv6 Networks”, University Utara Malaysia,July 2007.

[26] www.ipv6area.com, [accessed; 30 December 2012].

[27] www.nationmaster.com/encyclopedia/IPv7,[accessed ;1 march 2012]

[28] www.tomikl.net/naptd.docs.php, [accessed ; 30 December 2012].

[29] www.pixmule.com/ipv6, [accessed ; 11 November 2012].

[30] www.vmware.com, [accessed ; 12 December 2012].

[31] www.RIPE.net,[accessed ; 10 February 2013].

Appendices

Appendix A: Code Summary

1. The next shows general imports files for .net framework:

```
Imports System.Net.Dns
Imports System.Net
Imports System.IO
Imports System
Imports System.Windows.Forms
Imports System.Net.Sockets
Imports System.Text
Imports System.Net.Sockets.Socket
Imports Microsoft.VisualBasic
Imports System.Runtime.InteropServices
```

2.The next shows general declarations:

```
Const betweenPings As Integer = 10 'seconds between pings
    Dim isClosing As Boolean = False
    Dim splitter As String = "'\\'"
    Dim GLOIP As IPAddress
    Dim GLOINTPORT As Integer
    Dim bytCommand As Byte() = New Byte() {}
    Dim bytCommand2 As Byte() = New Byte() {}
    Dim fName As String
    Dim udpClient As New UdpClient
```

```
    Dim split() As String = Nothing
```

3.The next shows load form action:

```
Dim _IPAddress As System.Net.IPAddress
        _IPAddress = New System.Net.IPAddress(New Byte() {125,
23, 23, 4})
        Console.WriteLine(_IPAddress.ToString)

Console.WriteLine([Enum].GetName(GetType(System.Net.Sockets.
AddressFamily), _IPAddress.AddressFamily)) 'ipv4
        _IPAddress = New System.Net.IPAddress(New Byte() {125,
23, 23, 4, 125, 23, 23, 4, 125, 23, 23, 4, 125, 23, 23, 4})
        Console.WriteLine(_IPAddress.ToString)

Console.WriteLine([Enum].GetName(GetType(System.Net.Sockets.
AddressFamily), _IPAddress.AddressFamily)) 'ipv6
        Console.ReadLine()
        ComboBox1.Text = "1"
        ComboBox2.Text = "2"
        Text = "netwok test"
        pingThread.Start()
        Dim myHost As String = System.Net.Dns.GetHostName
        Dim myIPs As System.Net.IPHostEntry =
System.Net.Dns.GetHostByName(myHost)
        TextBox1.Text = myIPs.HostName
```

```
        For Each myIP As System.Net.IPAddress In myIPs.AddressList
            TextBox2.Text = myIP.ToString
                Next
        Dim client As New WebClient
        '// Add a user agent header in case the requested URI contains
a query.
        client.Headers.Add("user-agent", "Mozilla/4.0 (compatible;
MSIE 6.0; Windows NT 5.2; .NET CLR1.0.3705;)")
        Dim baseurl As String =
"http://automation.whatismyip.com/n09230945.asp"
        ' with proxy server only:
        Dim proxy As IWebProxy =
WebRequest.GetSystemWebProxy()
        proxy.Credentials =
CredentialCache.DefaultNetworkCredentials
        client.Proxy = proxy
        Dim data As Stream
        Try
            data = client.OpenRead(baseurl)
        Catch ex As Exception
            MsgBox("open url " & ex.Message)
            Exit Sub
        End Try
        Dim reader As StreamReader = New StreamReader(Data)
        Dim s As String = reader.ReadToEnd()
```

```
        data.Close()
        reader.Close()
        s = s.Replace("<html><head><title>Current IP
    Check</title></head><body>", "").Replace("</body></html>",
    "").ToString()
        'MessageBox.Show(s)
        TextBox5.Text = s
```

4.The next shows translation steps in vb.net code:

```
    Dim WSHShell
    Dim objNTInfo
    Dim GetComputerName
      Set objNTInfo = CreateObject("WinNTSystemInfo")
    strComputer = lcase(objNTInfo.ComputerName)
  strMyEmailAddr = InputBox("This script grabs your computer's serial,
express code and model number. Please enter in your email address")
  Set objWMIService = GetObject("winmgmts:\\" & strComputer &
"\root\cimv2")
Set colItems = objWMIService.ExecQuery("Select * from
Win32_BIOS",,48)
  For Each objItem in colItems
strSerial = objItem.SerialNumber
Next
 Function model(strComputer)
```

```
'Set objWMIService = GetObject("winmgmts:\\" & strComputer &
"\root\cimv2")
Set objWMIService = GetObject("winmgmts:" _
    & "{impersonationLevel=impersonate}!\\" & strComputer &
"\root\cimv2")
Set colItems = objWMIService.ExecQuery("Select * from
Win32_ComputerSystem",,48)
Set colAdapters = objWMIService.ExecQuery _
    ("SELECT * FROM Win32_NetworkAdapterConfiguration WHERE
IPEnabled = True")
 For Each objItem in colItems
model = objItem.Model
next
End Function
 Function Base2Base(InputNumber,InputBase,OutputBase)
Dim J, K, DecimalValue, X, MaxBase, InputNumberLength
Dim NumericBaseData, OutputValue
NumericBaseData =
"0123456789ABCDEFGHIJKLMNOPQRSTUVWXYZ"
MaxBase = Len(NumericBaseData)
if (InputBase > MaxBase) OR (OutputBase > MaxBase) then
Base2Base = "N/A"
Exit Function
end if
```

```
'Convert InputNumber to Base 10
InputNumberLength = Len(InputNumber)
DecimalValue = 0
for J = 1 to InputNumberLength
for K = 1 to InputBase
if mid(InputNumber, J, 1) = mid(NumericBaseData, K, 1) then
DecimalValue = DecimalValue+int((K-
1)*(InputBase^(InputNumberLength-J))+.5)
end if
next
next
'Convert the Base 10 value (DecimalValue) to the desired output base
OutputValue = ""
while DecimalValue > 0
X = int(((DecimalValue/OutputBase)-
int(DecimalValue/OutputBase))*OutputBase+1.5)
OutputValue = mid(NumericBaseData, X, 1)+OutputValue
DecimalValue = int(DecimalValue/OutputBase)
Wend
Base2Base = OutputValue
Exit Function
End Function
 ' Get IP and MAC Address
 strComputer = "."
```

```
Set objWMIService = GetObject("winmgmts:" _
    & "{impersonationLevel=impersonate}!\\" & strComputer &
"\root\cimv2")
 Set colAdapters = objWMIService.ExecQuery _
    ("SELECT * FROM Win32_NetworkAdapterConfiguration WHERE
IPEnabled = True")
 n = 1
 For Each objAdapter in colAdapters
    strMACAddress = objAdapter.MACAddress
    If Not IsNull(objAdapter.IPAddress) Then
      For i = 0 To UBound(objAdapter.IPAddress)
        objValue = objAdapter.IPAddress(i)
      Next
   End If
 Next
 '==Message boxes.
 WScript.Echo "Your Serial: " & strSerial
Wscript.Echo "Express code: " & Base2Base(strSerial, 36, 10)
WScript.Echo "Model:" &  model(strComputer)
WScript.Echo "IP Address:" &  objValue
WScript.Echo "MAC Address:" &  strMACAddress
 MsgBox("")
 '==Email
 Set objMessage = CreateObject("CDO.Message")
objMessage.Subject = "Inventory Info"
```

```
objMessage.From = strMyEmailAddr
objMessage.To = ""
objMessage.TextBody = "Your Serial: " & strSerial & vbCrLf &
"Express Code: " & Base2Base(strSerial, 36, 10) &  vbCrLf & "Model:"
& model(strComputer)& vbCrLf & "IP Address:" & objValue & vbCrLf &
"MAC Address:" & strMACAddress
'==This section provides the configuration information for
the remote SMTP server.
'==Normally you will only change the server name or IP.
objMessage.Configuration.Fields.Item _
("http://schemas.microsoft.com/cdo/configuration/sendusing") = 2

'Name or IP of Remote SMTP Server
objMessage.Configuration.Fields.Item _
("http://schemas.microsoft.com/cdo/configuration/smtpserver")
="mail.domain.com"
'Server port (typically 25)
objMessage.Configuration.Fields.Item _
("http://schemas.microsoft.com/cdo/configuration/smtpserverport"
        ) = 25
objMessage.Configuration.Fields.Update
'==End remote SMTP server configuration section==
```

```
objMessage.Send
'================
foreach (IPAddress IPA in
Dns.GetHostAddresses(HttpContext.Current.Request.UserHostAddres
s))
if (IPA.AddressFamily.ToString() == "InterNetwork")
IP4Address = IPA.ToString();
break;

if (IP4Address != String.Empty)
return IP4Address;
foreach (IPAddress IPA in
Dns.GetHostAddresses(Dns.GetHostName()))
if (IPA.AddressFamily.ToString() == "InterNetwork")
IP4Address = IPA.ToString();
break;
String strHostName = Dns.GetHostName();
if (IP4Address != String.Empty)
{
return strHostName + IP4Address;
```

5.The next code to show progress and that in class object:

```
    Inherits Stream
        Private ReadOnly file As FileStream
        Private ReadOnly m_length As Long
        Public Class Progress Changed
    EventArgs
```

```
        Inherits EventArgs
        Public BytesRead As Long
        Public Length As Long
        Public Sub New(ByVal BytesRead As Long, ByVal Length As
Long)
            Me.BytesRead = BytesRead
            Me.Length = Length
        End Sub
    End Class
    Public Event ProgressChanged As EventHandler(Of
ProgressChangedEventArgs)
    Private bytesRead As Long
    Public Sub New(ByVal file As FileStream)
        Me.file = file
        m_length = file.Length
        bytesRead = 0
        RaiseEvent ProgressChanged(Me, New
ProgressChangedEventArgs(bytesRead, m_length))
    End Sub
    Public Function GetProgress() As Double
        Return CDbl(bytesRead) / file.Length
    End Function
    Public Overloads Overrides ReadOnly Property CanRead() As
Boolean
```

```
        Get
            Return True
        End Get
    End Property
    Public Overloads Overrides ReadOnly Property CanSeek() As
Boolean
        Get
            Return False
        End Get
    End Property

    Public Overloads Overrides ReadOnly Property CanWrite() As
Boolean
        Get
            Return False
        End Get
    End Property
    Public Overloads Overrides Sub Flush()
    End Sub
    Public Overloads Overrides ReadOnly Property Length() As Long
        Get
            Throw New Exception("The method or operation is not
implemented.")
        End Get
    End Property
```

```
    Public Overloads Overrides Property Position() As Long
        Get
            Return bytesRead
        End Get
        Set(ByVal value As Long)
            Throw New Exception("The method or operation is not
implemented.")
        End Set
    End Property
    Public Overloads Overrides Function Read(ByVal buffer As
Byte(), ByVal offset As Integer, ByVal count As Integer) As Integer
        Dim result As Integer = file.Read(buffer, offset, count)
        bytesRead += result
        RaiseEvent ProgressChanged(Me, New
ProgressChangedEventArgs(bytesRead, m_length))
        Return result
    End Function
    Public Overloads Overrides Function Seek(ByVal offset As Long,
ByVal origin As SeekOrigin) As Long
        Throw New Exception("The method or operation is not
implemented.")
    End Function
    Public Overloads Overrides Sub SetLength(ByVal value As Long)
        Throw New Exception("The method or operation is
not
```

```
implemented.")
    End Sub
    Public Overloads Overrides Sub Write(ByVal buffer As Byte(),
ByVal offset As Integer, ByVal count As Integer)
        Throw New Exception("The method or operation is not
implemented.")
    End Sub
End Class
```