Chapter 3

Evaluating Performance of IPv6 Migration Techniques

3.1 Introduction

The IPv6 lacks compatibility with its predecessor IPv4 as such cannot make use of the services provided by IPv6 or communication with IPv6 nodes. The migration or transition between the two protocols is expected to take considerable amount of time. The transition mechanisms come into picture as long as the two need interoperability with each other (Frankel et al, 2010). Typically three different kinds of hosts are present in a transition scenario: IPv4 only, IPv6 only and Dual Stack which support both IPv4/IPv6. Although Dual stack is formally commended, there might be instances that require single stack systems to operate. These systems normally use other mechanisms like Tunneling and Translation to operate. Transition Mechanisms altogether provide seamless integration between IPv4 and IPv6. The factors that impact the physical deployment of transition mechanisms include client capabilities, time frame, cost, transition stage, organization size etc. Rigorous research and study is being done on the transition from IPv4 to IPv6 because the process requires high level compatibility and a clear protocol for effortless deployment. Finding such a protocol is hard to achieve given the infrastructural and performance barriers in the network (Shah & Parvez, 2014). The performance of the network may be attenuated by different factors like congestion and routing loops. Also processing power in CPU and memory scalability in nodes comes into play. Before deploying any transition mechanism, it's important to measure its performance metrics and know how it will scale in a given environment. In addition to evaluating existing standard transition mechanisms, research community around the world has been in the search of a more optimized and robust transition technique. For example authors (Afreh, Mellor, Kamala & Kasasbeh,2008) have proposed a new flexible transition scheme called as BDIPS (Bi-directional Intelligent Processing System) that deals with the address mapping transition IPv4 to IPv6 and vice versa depending on an understanding of the two network environments and converting the received datagram to the destination environment. They claim that BDIPS is simple and easy to implement and also reduces the packet size compared with encapsulation, reduces the overall transmission time, and overcomes the loss of information inherent in IPv4/IPv6 header translation. A similar kind of work has been done authors (Al-Kasasbeh et al, 2010).They have proposed algorithms that deal with header processing transformation transition between IPv4/IPv6 and vice versa depending on the bi-directional identification and recognition processes of the two distinct headers. In (Liu et

al, 2008), the authors propose a hierarchical approach for integrating IPv4 and IPv6 networks. The work regarding other novel approaches can be found in (Zhai et al, 2011; Xia et al, 2014; Kong et al, 2003; Azcorra et al,2010). Thus although, a number of transition techniques have been proposed and standardized, developing an optimal one is still a distant dream. In this chapter, we review and provide comparative analysis of IPv4 to IPv6 Migration techniques. We have empirically carried out implementation of existing standard migration techniques using OPNET Modeller Simulation. Based on the calculated parameters, an approach has been made towards finding better technique among the existing migration techniques. The experiment helps us in solving the problem of choosing best IPv6 migration technique.

3.2 Transition Mechanisms

For the seamless coexistence of both protocols (IPv4 and IPv6) till the completion of migration process, several transition mechanisms are proposed by IETF which are broadly classified under Dual Stack, Tunneling, and Translation Mechanisms.

3.2.1 Dual Stack

As the name suggests Dual Stack means maintaining two protocol stacks to aid both IPv4 and IPv6 in routers and hosts (Nordmark & Gilligan, 2005).In hosts, both IPv6 and IPv4 enabled services are allowed to operate on the same device. For communicating with an IPv6 node, such node behaves like an IPv6 only node while as communicating with an IPv4 node, such nodes behaves like an IPv4 only node. Dual Stack Internetworking devices facilitate handling and forwarding of both IPv6 and IPv4 packets. The datagram header is analyzed to decide the packet forwarding decisions. Based on DNS address query and lookups, appropriate stack is chosen. A Dual stack node requires a DNS resolver having the ability to resolve both types of DNS address records. i.e. DNS A record resolves IPv4 addresses, and the DNS AAAA record resolves IPv6 addresses. Dual Stack nodes consume more system resources than single stack nodes. For example; the network bandwidth is shared between the two protocols. Also routers need to maintain two forwarding tables in addition to running routing protocols for IPv4 and IPv6.Special services like congestion control and packet filtering also needs to be implemented for both the protocols. Dual Stack is one of the simplest and conventional Transition Mechanism

used today. However, Dual Stack enables and supports communication between like nodes, i.e. IPv4 to IPv4 and IPv6 to IPv6 communication is possible. Research is being conducted on enabling IPv4 to IPv6 and IPv6 to IPv4 communication. Figure 3.1 shows the operation of Dual Stack Mechanism.

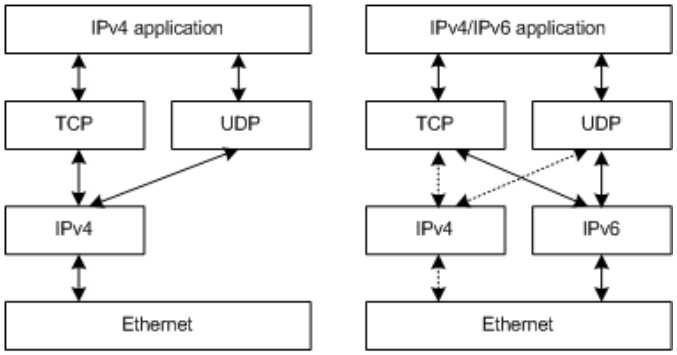

Figure 3.1 (a) Single Stack (b) Dual Stack

3.2.2 Tunneling

Tunneling makes it possible to wrap one version of IP into another version so that incompatible networks are bridged. Tunneling bridges two outlying IPv6 networks and enables them to interface over IPv4 links or vice versa. Tunneling does not require hardware up gradation and is one of the easiest ways to migrate to IPv6. Tunneling mechanism involves encapsulation at the tunnel entry point, decapsulation at the tunnel exit point and management of the tunnel. The following two step procedure is followed when an IPv6 packet needs to traverse through existing IPv4 infrastructure. At the tunnel entrance, the IPv6 hop limit gets decremented by one followed by its encapsulation inside an IPv4 packet. The encapsulated packet gets transmitted through the tunnel. If MTU is small, the IPv4 packet may get fragmented. At the tunnel exit point, the source of the packet gets checked to see if it originated from a legitimate source according to the tunnel configuration. If fragmentation was carried out, the tunnel exit point re-assembles the whole packet. Thereafter the IPv4 header is removed to retrieve back the original IPv6 packet. Further literature regarding the tunnels in documented in RFC's 2473, 4213 and 3056.

Tunneling techniques is classified under two categories: Automatic Tunnel and Configured Tunnel. Configured tunnels are manually configured IPv6-over-IPv4 tunnels where tunnel end

points are predetermined by configuration information. These tunnels are manually administered by Tunnel Administrators and are mostly bidirectional. The manual tunnel dispenses a virtual point-to-point link to the IPv6 layer by making use of configured IPv4 addresses as the end point addresses at the lower layer. Configured tunnels require a high administrative overhead than Automatic tunnels, but due to security issues are recommended.

Automatic Tunneling does not require manual configuration and endpoint specification. This allows IPv6/IPv4 nodes to transfer information over a deployed IPv4 infrastructure without the need for tunnel endpoint configuration. The information regarding the tunnel is extracted from an IPv4 address that has been embedded in an IPv6 address. Previously the tunnel destination addresses were inferred by an IPv4 compatible destination address. However; RFC 4213 has changed the interpretation of Automatic tunneling and IPv4 compatible address. Automatic Tunneling now uses 6to4 address which has a different format than an IPv4 compatible IPv6 address. Nowadays, 6to4 is not used in organizations and has been replaced by 6rd (IPv6 rapid deployment).

The operation of general Tunneling Mechanism is shown in the figure 3.2.

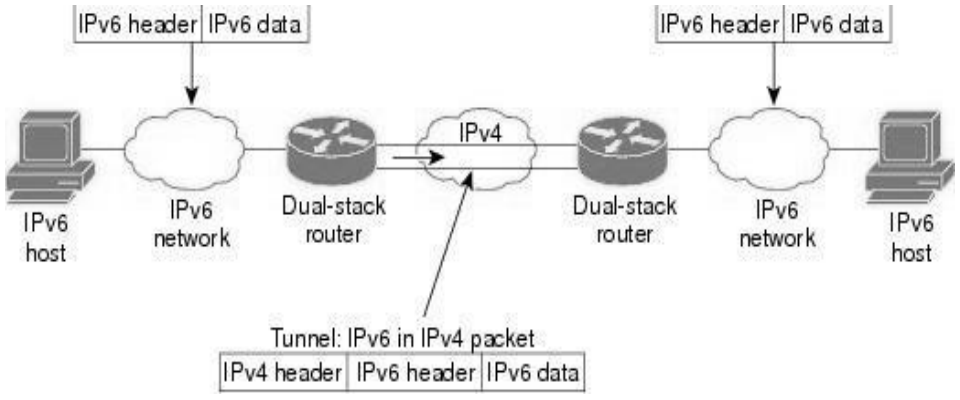

Figure 3.2 Tunneling mechanism

3.2.2.1 Types of Tunnels

The Tunnels are mainly of following types:

- *6to4* – RFC [3056] defines 6to4 as automatic router-to-router tunneling that employs address prefix of 2002::/16 for identifying a node that implements 6to4.With 6to4,the two secluded IPv6 islands are interfaced with one another using minimal overhead configuration and without any need to configure tunnels explicitly. The secluded IPv6 node is assigned a prefix of 2002:V_4ADDR::/48 having embedded IPv4 address.(V_4ADDR is unique IPv4 address configured on router).In 6to4,the IPv4 network is looked upon as Unicast point-to-point link layer and the isolated IPv6 islands communicate via 6to4 routers which are also known as 6to4 gateways. In 6to4; the IPv6 packets get encapsulated in IPv4 at 6to4 gateway. This technique requires at least one unique Unicast IPv4 address for tunnel configuration.
- *6over4*–defined in RFC [2529] is an automatic host-to-host tunneling that tethers outlying IPv6 sites through IPv6-in -IPv4 encapsulation and does not require explicit tunneling In 6over4,IPv6 datagram's are transmitted between dual stack nodes using multicast supported IPv4 as link layer.
- *ISATAP* – is an automatic tunneling mechanism that connects two IPv6 enabled devices over an IPv4 network viewing IPv4 as virtual link layer. It is designed to provide IPv6 connectivity for dual stacked nodes over an existing IPv4 infrastructure. This technique is independent of the requirement of global or private IPv4 addresses. ISATAP uses different IPv6 address format affixed with an inbuilt support to IPv4 address. Unlike 6over4, it does not require multicast supported IPv4 infrastructure. Using ISATAP, the IPv4 address is embedded in the EUI-64 interface identifier. ISATAP uses a 64 bit prefix value that can be link-local, site local or a 6to4 prefix. ISATAP address has the following format 64bitprefix **:** 5efe **:** V_4addr; where 'fe' indicates that the address contains an embedded IPv4 address and '5e' is part of IANA OUI.
- *Tunnel Brokers/ Tunnel Setup Protocol (TSP)* –simplifies the composition of validated IPv6 tunnels supported over IPv4 networks. It is an automatic tunneling mechanism that supervises tunnel creation between some dual-stack nodes and tunnel-broker servers, thus bridging with predestined IPv6 network.

- *Teredo*–is a type of automatic tunneling that dispenses IPv6 over UDP connectivity to NAT'ed IPv6 hosts which are hidden behind a NAT router and don't have a unique public IPv4 address. Teredo aids in interfacing these IPv6 nodes with global Internet. Since 6to4 mandates for a public IPv4 address at tunnel endpoints, however due to depletion of IPv4 address space, many IPv6 nodes are connected to global IPv4 internet via NAT'ed devices. In this scenario, 6to4 tunnel endpoints are to be configured on a NAT'ed device having a public IPv4 address. But due to infeasibility of hardware up gradation and economic issues, it isn't a viable option. Teredo solves the obstacle by encapsulation of IPv6 packets inside IPv4/UDP datagram's which can be forwarded by many NAT'ed devices.
- *DSTM*- resolves the panacea for IPv6 only networks in which some IPv4 dependent services are still sustained on dual stack nodes inside the infrastructure of IPv6.The IPv4 traffic gets tunneled over IPv6 only domains till it reaches a dual stack gateway carrying out encapsulation /decapsulation and forwards packets between the IPv4 only and IPv6 only Domains. DSTM offers the advantage of being transparent to the network and liberty of operating legacy IPv4 applications over IPv6 only networks. DSTM architecture consists of three main components namely a DSTM client, DSTM server and DSTM gateway. DSTM can be a flawless option for policy makers during the early phase of Migration.

3.2.3 Translation

Translation is the mechanism of converting protocol headers in-between an IPv6 and IPv4 node. Translation can occur at several layers of TCP/IP including Network, Transport and Application Layer. Translations alter or carry conversion of IP packets between IPv6/IPv4 which often result in attribute or information loss unlike tunneling which doesn't modify the tunneled datagram. Translation is often preferred in an environment where an IPv4 only node communicates with IPv6 only node or vice-versa Translation can either be state full or stateless. A stateless translation processes packets independently without any citation to previous translations. A state full translation maintains some sort of session or mapping with reference to previous translations. Several translation algorithms (Nordmark, 2000; Waddington & Chang, 2002) have been propounded based on SIIT (Stateless IP/ICMP Translation Algorithm).

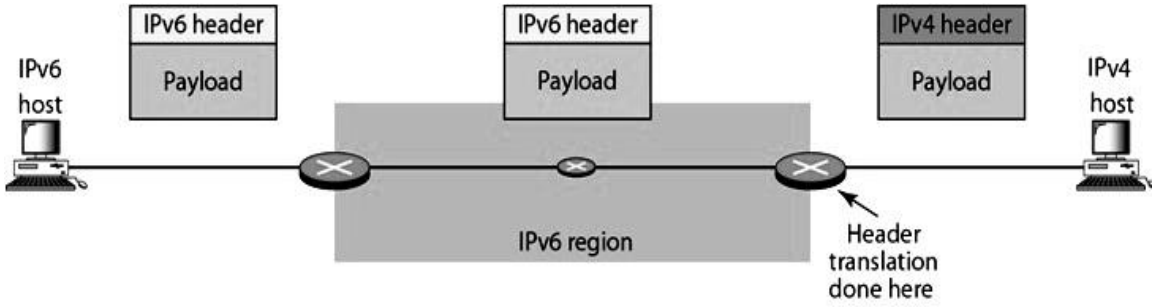

Figure 3.3 Header Translation

SIIT which operates on an IPv6 host is a bidirectional translation algorithm capable of transforming an incoming IPv4 header to IPv6 header and outgoing IPv6 header into IPv4 header. Translation also occurs between ICMPv6 and ICMPv4 messages. Translation mechanism does not affect header checksum values but ignores many extension headers and IPv4 options resulting in the loss of vital header information. SIIT algorithm acts as a building block for mechanisms like BIS, BIA and NAT-PT.

BIS (Bump-in-the-Stack) permits IPv6 links to be used as communication infrastructure for hosts running IPv4 application services.BIS investigates transmission of data between the link layer devices (NIC cards) and TCP/IPv4 modules and carries translation of IPv4 packets into IPv6 with the help of three Bump components namely Address Mapper, Translator and Extension Name Resolver as shown in figure 3.4.

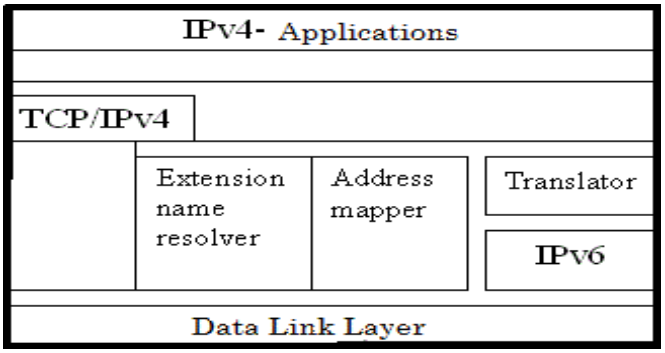

Figure 3.4 Bump in the Stack

Bump-in-the-API (BIA) similar to BIS (Tsuchiya,Higuchi & Atarashi,2000) permits IPv6 links to be used as communication infrastructure for hosts running IPv4 applications however BIA approach performs API translation between IPv6 and IPv4.The socket API calls are interrupted by the Bump layer which is placed as a segment of socket layer. The BIA layer is embedded in-between the application and TCP/UDP layer on host consisting of three components i.e. a function mapper, name resolver, and an address mapper as shown in figure 3.5.

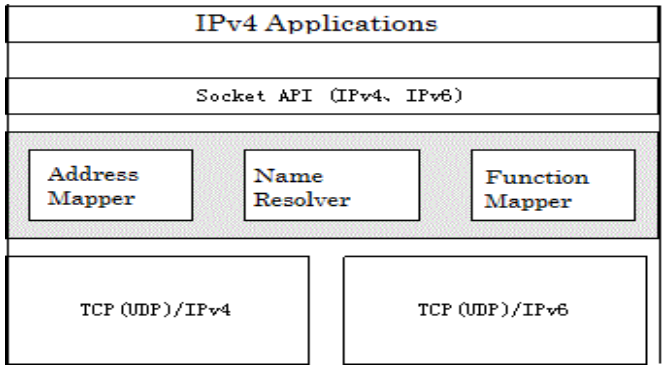

Figure 3.5 Bump in the API

NAT-PT is another translation mechanism that permits communication between IPv6 and IPv4 nodes by employing a single IPv4 address. Different IPv6 nodes are served by NAT-PT by allocating an ephemeral IPv4 address with a unique TCP/UDP port value to each node and mapping IPv6 to it thereby acting as a proxy for communication.

SOCKS64 (RFC 1928) is another translation mechanism that uses socket based applications running on dual stack devices to permit communication between IPv6/IPv4 nodes.Socks64 that works as a socket secure server relays communication between IPv6/IPv4.Applications need to be socksified at client end by using SOCKS64 library that replaces sockets and DNS API's.

TRT or Transport Relay Translator like SOCKS64 acts as a state full gateway node linking two independent connections over completely different networks. A TCP/UDP connection from the source host terminates on the TRT where it forks a separate connection to the destination host and thus acts a relay between the two connections.

3.3 A Comparative Analysis

This section provides a comparative review of the above discussed IPv4/IPv6 migration techniques. Although Dual Stack can be deployed on hosts, routers and on the same interface as IPv4, however it potentially requires 2 routing tables and processes. This comes in addition to the CPU and memory capacity of nodes. On the other hand, Tunnels are easy to deploy and are available on most platforms. But tunnels too have some issues. The tunnels must be manually configured in order to ensure security of the network. Tunnels also have issues with delay and latency through the tunnel in addition to being susceptible to single point of failure. The translation techniques although being cost effective require a significant amount of configuration from the administrative side. For network co-existence, translation techniques are not recommended by IETF.

Table 3.1 compares the transition mechanisms on the basis of six major factors:

- Hardware and Software (Upgrade in existing infrastructure).
- Complexity of the network
- Performance behavior of the Migration Techniques.
- Security Issues of Migration techniques.
- Costs Involved with the implementation of the Migration techniques.
- Advantage/Disadvantage of the Migration Technique.

	Dual Stack	**Tunneling**	**Translation**
Hardware/Software	Requires Hardware and Software up-gradation to support both IPv4 as well as IPv6	Does not require Hardware Upgrade. Likely changes on software side. Suitable for legacy equipments	Requires Specialized Translating Routers. Suitable for connecting IPv4 only and IPv6 only nodes.
Complexity	Although easy to implement, but requires complex management and high memory for storing both IPv4 as well as IPv6 routing tables.	Easy to implement over existing IPv4 infrastructure	Suffers Protocol conversion Overhead.
Performance	Can be slow due to presence of dual protocols. High end processing machines are needed.	Depends on tunnel speed. Might get slowed by encapsulation/decapsulation delay	High speed translation gateways are required
Security	Protected by IPSec	Protected by IPSec	Issues with NAT security. End to End Security Impossible
Cost	High cost of setup. Hardware costs.	Insignificant	Low cost
Advantages	Easy and Direct method for a smooth transition. Based on standard protocols.	Easy to implement over existing IP4 network. Best technique for older legacy equipment.	Provides translation at lower cost. Infrastructure need not be changed substantially to provide services to IPv6 networks
Disadvantages	Lower scalability. Entire network cannot be dual stacked. Higher Network setup Cost.	Tunnel end breakdown will fail the network.	Loss of information because some fields are discarded. Not Recommended by IETF

Table 3.1 Comparative Analysis of Transition Mechanisms

3.4 Introduction to OPNET Simulation

OPNET is an acronym for "Optimized Network Engineering Tool". It is an application software proficient of simulating large networks with protocol modeling and collecting performance analysis. It provides rich simulation features like Graphical User Interface (GUI), object based modeling, integrated data analysis tool and dynamic event simulation kernel. OPNET is composed of high level user interface which is built from C and C++ source code blocks with a huge library of OPNET specific functionality (Lu & Yang, 2012). The OPNET supports simulation at the packet level. OPNET can be used both as a research tool as well as network design/analysis tool depending upon the needs of the user. Setting of test beds and test cases using the internetworking devices is not always feasible in a real world scenario. This entails for additional costs and is also time consuming. In order to reduce the time and operational cost involved with setting up a real network, simulation environment is often the preferred choice of users. The simulation tools provide a way to model and simulate a real network inside a computer system. This tool is independent of the hardware devices required to model the network. Every internetworking device is embedded inside the simulation environment in the form of a software package available inside the GUI. The OPNET provides users with drag and drop facility to setup and monitor the network. After setting up the network, the simulations are run and results are collected which can used to study the behavior of the network. OPNET uses the hierarchical structure modeling which is divided into three main domains (Sethi & Hnatyshin, 2012):

- Network Domain which includes network and sub-networks, network topologies, geographical co-ordinates, mobility etc.
- Node Domain which includes single network nodes (e.g. workstations, routers, switches and other devices).
- Process Domain which includes single modules and source code inside the network nodes.

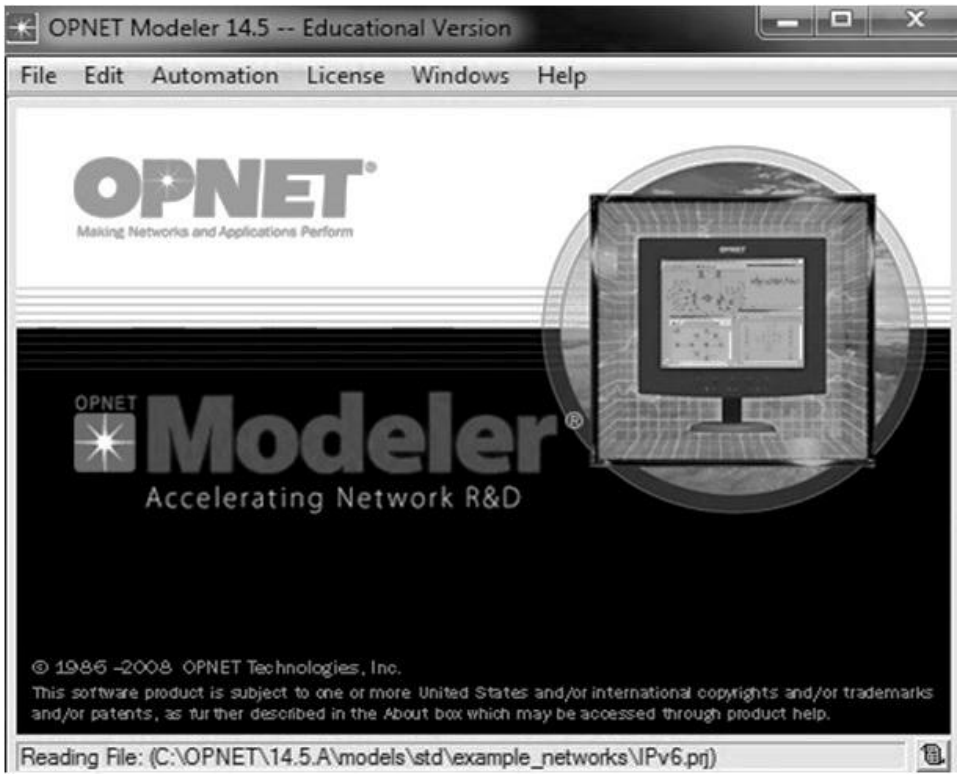

Figure 3.6 OPNET Snapshot

The OPNET Modeller simulation tool consists of the following components:

- Network Editor which is a tool for building network models. A network can host sub-networks with nodes and links. The network editor specifies the overall scope of the system to be simulated.
- Node Editor is a tool for modeling network nodes. This tool defines the internal structure of a network node and behavior of the network object. Typical nodes include workstations, routers, switches, terminals etc.
- Process Editor is a tool that is used to simulate process models. These models are described by Finite State Machines (FSM). These machines consist of states with transitions and conditions between them.
- Packet Format Editor Tool is used to define the internal structure of the packet.
- Probability Density Function Editor permits us to describe the spread of probability over a range of different outcomes.
- Probe Editor is another tool that is used to specify what data has to be collected from the simulation output. It is necessary so as to analyze the behavior of the network.

- Simulation Sequence Editor is used to add constraints to the simulation behavior.
- Analysis Tool provides the results of the simulation in the graphical format.

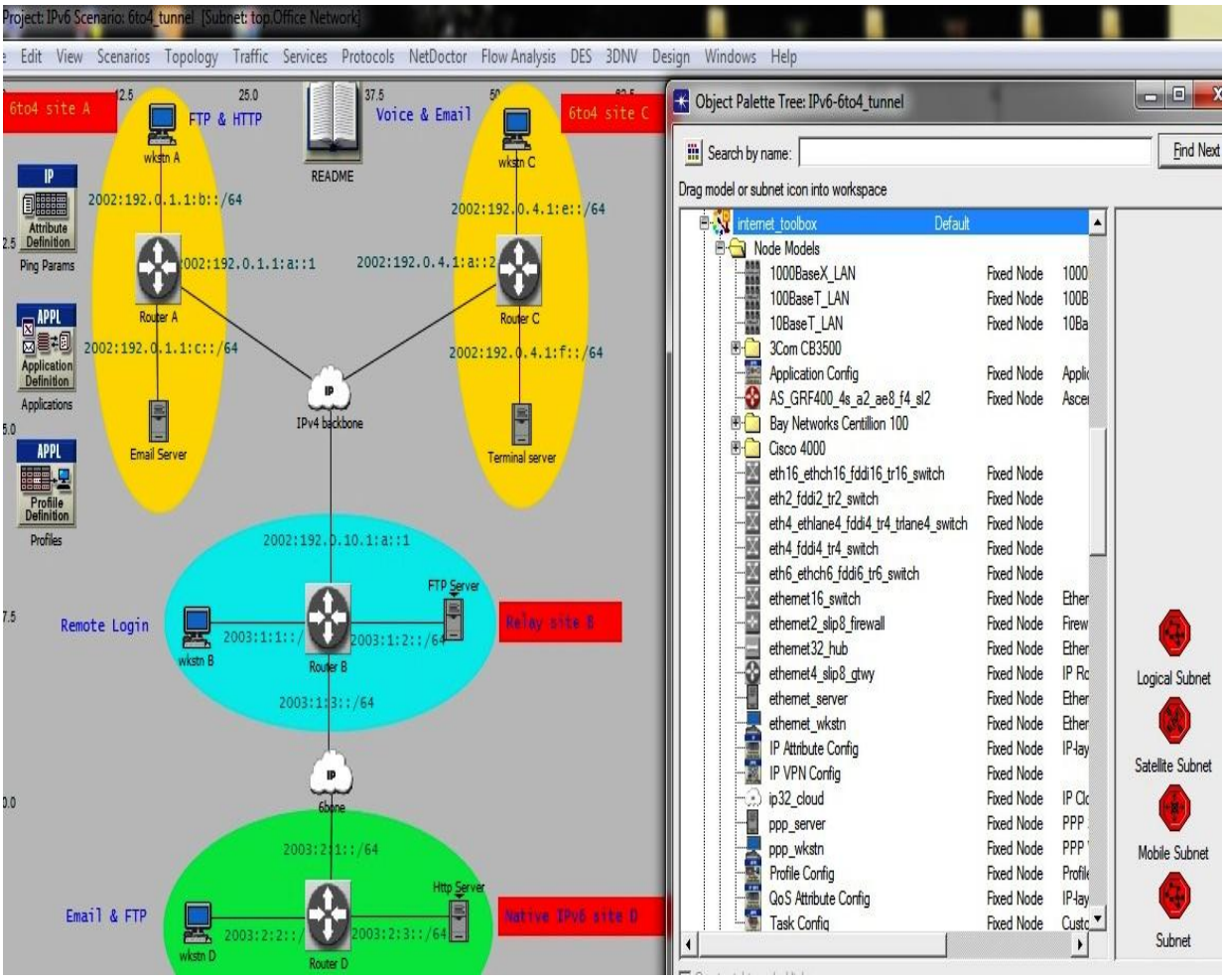

Figure 3.7 OPNET Design Panel

In addition to the above mentioned components, OPNET supports four different kinds of simulation environments.

- *Discrete Event Simulation (DES)* executes the protocol in much the same way as the real environment. It provides highly detailed models that explicitly simulate packets and protocol messages. Although DES provides higher accuracy results, the simulation runtimes are often longer than others.

- *Hybrid Simulation* is a combination of two distinct modeling techniques (analytical and discrete) to provide accurate detailed results for target flows. Hybrid simulation depends on background and explicit traffic. The execution runtime is notably faster than DES.
- *Flow Analysis* model steady state network behavior using different analytical techniques and algorithms. This type of environment is not used for modeling protocol messages or packets; however it's used for studying routing and reachability across the network in the steady state. The runtime execution time is usually faster than DES.
- *ACE Quick Predict* studies impact on application response time of changing network parameters using different analytical techniques.(e.g. bandwidth, latency, utilization, packet loss).

Using any of the simulation environments, the behavior of the network can be studied. The workflow of OPNET is divided into four major steps; creating a new network topology, choosing which statistics to collect, running the simulation scenario and obtaining/analyzing the results.

3.5 Performance Analysis of Migration Techniques

Transitioning from IPv4 to IPv6 is a gradual process necessitating remodeling the network infrastructure. For the successful deployment of IPv6, the next logical step is to vote out the most acceptable transition method. As discussed in literature above, a number of transition mechanisms are available and each has its own advantages and disadvantages; but the most difficult problem is to make a decision for choosing a method that will be smooth and will allow seamless integration. The method so chosen should also not degrade the QoS and performance of the network. In this section, we will be evaluating the performance analysis of some existing IPv4 to IPv6 migration techniques with the help of OPNET Modeller simulation. The simulation results are significant and give an insight about choosing best transition technique and an idea about network migration and capacity planning.

3.5.1 Related Work

The wide interest in Next Generation Networks and Internet has put increasing demands on real time multimedia applications. Due to the increasing adoption of IPv6, a significant interest has been on measuring performance metrics and behavior of IPv6 and migration networks (Wang et al,2005;Sathu et al,2011;Pezaros et al,2004;Zeadally et al,2003;Shiwani et al,2013).

(Bilski, 2010) discusses about network performance issues in Internet Protocol Migration. The paper surveys about negative and positive impacts of IPv6 implementation with special emphasis on the IP transition phase. One negative impact is the larger address of IPv6 (128 bits) which require more computational power on routers and servers to process it.

In (Aziz et al, 2011), the authors have measured throughput of voice/video Traffic in IPv4/IPv6 networks. The experiment is carried in OPNET simulation environment. The result shows IPv6 produces 3% higher throughput than IPv4 but the difference is insignificant. A similar kind of study is carried in (Sathu et al, 2011) where the performance of various video formats like MPEG-1, MPEG-2 and MP-4 is studied in different IP transition mechanisms like Dual Stack, 6to4, 6in4.Results show that Dual Stack performs better than 6to4 and 6in4 tunneling mechanisms.6to4 tunneling had less impact on video packets but 6in4 greatly affected the data.

In (Bahaman et al, 2012), capabilities of 6to4 tunneling are compared with IPv6/IPv4 networks. The authors investigates performance of TCP and UDP and calculate parameters like Round Trip Time (RTT), Throughput and overhead caused by tunneling. The results show that tunneling performance is considerably below average than IPv4 and IPv6 in context of TCP data transmission. However incase of UDP, its insignificant.

An investigation of QoS measurement in IPv6 domain has been presented in (Bouras et al, 2004).The results show that IPv6 produces a higher throughput than IPv4 and better QoS parameters. A similar kind of result can be found in (Sasanus et al, 2012) where the authors have tried to evaluate bandwidth requirement of different internet applications in 6to4 automatic tunneling environment.

(Lihua et al, 2012) provide a comparative analysis of IPv6 migration techniques. They evaluate end-to-end application performance by calculating transmission Ethernet latency and server

response time using OPNET simulation. A similar kind of work can be found in (Radley et al, 2013; Kalwar et al, 2015).In (Hanumanthappa et al, 2010), the authors have evaluated two most common transition mechanism BD-SIIT and DSTM. Their findings reveal that that no unique solution exists for transition problem between IPv4 and IPv6.The solutions for transitioning from IPv4 to IPv6 varies according to the needs and the requirements of the users. Different transition mechanism will be appropriate for different requirements in different networks at different points.

More work on evaluating performance metrics can be found in the literature (Wu & Zhou, 2011; Elich et al, 2013; Srinidhi et al, 2014).

3.5.2 Simulation Setup

This section describes the experimental setup for evaluating four migration techniques namely Dual Stack, Automatic 6to4 tunneling, Manual 6in4 tunneling and NAT-PT and makes comparative analysis with native IPv6 environment. The implementation is carried out in a controlled environment using basic internetworking devices and network components. The components that we are using include workstations, gateway routers, IP backbones, Ethernet 100baseT and PPP_DS3 communication links. The experiment is carried out in five different network scenarios. The topology used for all scenarios is shown in figure 3.8.

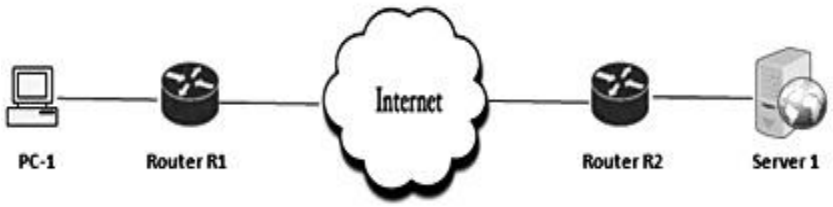

Figure 3.8 Network Topology for Simulation

All the five scenarios are modeled using OPNET simulator v14.5.The PC1 is using IPv6 protocol which attempts to communicate with server1 using the same protocol. In first four scenarios (Automatic 6to4, Manual 6in4, Dual Stack and NAT-PT), the communication network in-between is IPv4 based and transition techniques are configured on Router R1 and R2. In fifth scenario IPv6 only network is configured unescorted by any migration technique. In all the scenarios, HTTP traffic is being generated by PC1 with page inter arrival time showing an exponential distribution of mean 60.HTTP application traffic is configured for heavy browsing

along the network. The parameters that we are going to measure in this experiment are *Throughput*, *Response Time* and *TCP Delay*. The Throughput is defined as the average data transferred across a medium per unit time. In our scenario link throughput is calculated at the link connecting R1 and IP backbone because all packets from the PC1 are forwarded to Global Internet via this link. The application Response time represents the time elapsed between sending a request and receiving the response packet for the application. In our case, it's measured on workstation PC1. TCP Delay (in seconds) suffered by the TCP layers is the delay of packets in the complete network and is calculated from the time the source TCP layer sends an application packet data to the time it is received by the TCP layer at destination node.

3.5.3 Results

All the scenarios are run separately and performance metrics are calculated on individual basis. The total simulation runtime for each scenario is 30 minutes. Figure 3.9 to 3.11 shows the result of performance metrics that are collected.

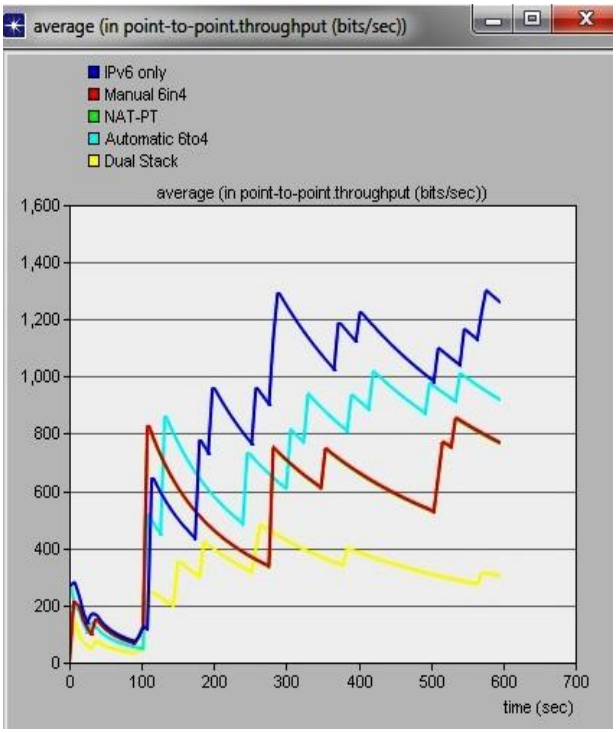

Figure 3.9 Throughput

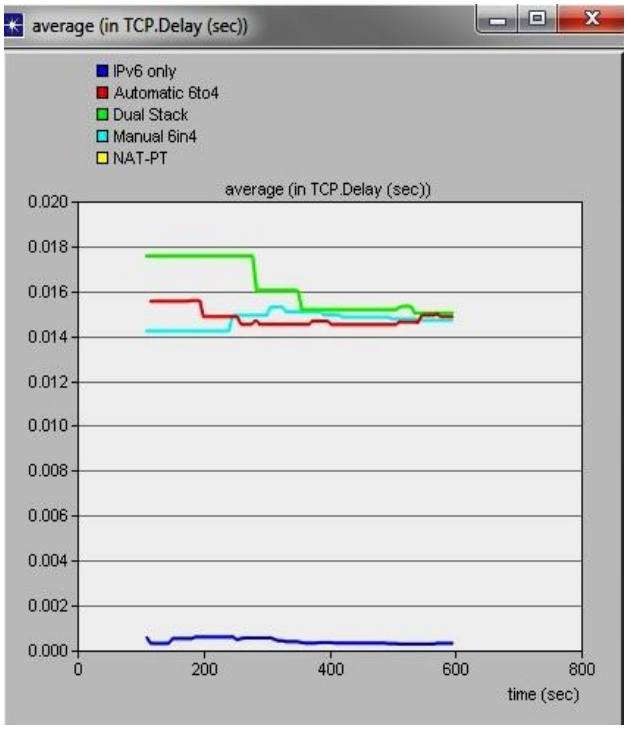

Figure 3.10 TCP Delay

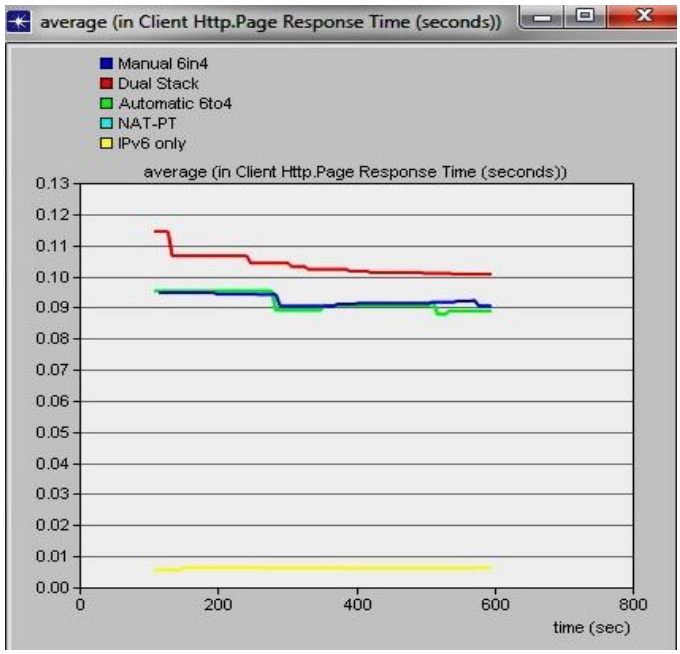

Figure 3.11 Response time

The average of each calculated value is shown in table 3.2 for simplification. Analyzing table 3.2, we notice that IPv6 only environment yields higher throughput than remaining four network scenarios. This is obvious because IPv6 has been designed for faster packet processing. Also Response Time and TCP delay suffered by IPv6 is quite low as compared to migration networks. In case of migration networks, Automatic 6to4 performs better than Dual Stack and Manual 6in4 Tunneling.Worst Case performance metrics are shown by Dual Stack network.

Scenario	**TCP Delay (ms)**	**Throughput (bits/sec)**	**Response Time (ms)**
Automatic 6to4	14.0	900	90
Manual 6in4	14.55	720	97
Dual Stack	16.63	335	110
NAT-PT	14.55	710	96
Native IPv6	1.0	1100	9.0

Table 3.2 Average values of simulation for experiment

3.6 Summary

As mentioned earlier, the chapter examined current IP transitioning techniques and engaged in a thorough review about comparative analysis of IPv4 to IPv6 Migration techniques. The paper also made an empirical evaluation of four most commonly used transition mechanisms namely Dual Stack, Automatic 6to4 Tunneling, Manual 6in4 Tunneling and NAT-PT and made comparison of performance metrics with native IPv6 environment. The results have shown that IPv6 only network produces a higher throughput and suffers a minimum delay compared to transitioning networks. This also means that IPv6 has faster packet processing and forwarding capability than transitioning networks. In case of migration networks, Automatic 6to4 performs better than Dual Stack and Manual 6in4Tunneling. The worst performance is shown by Dual Stack Network. This is obvious due to presence of two protocols in dual stack. The result from this experiment infers that transition techniques can only be seen viable for the migration period and may not be suitable for the long term deployment of applications on the internet. Till date no best feasible solution for transition plan has evolved. Although a number of IPv6 transition techniques are available, they all seem to be scenario specific. The only feasible solution for bandwidth efficiency and higher throughput is complete implementation of IPv6.Future work in this area may be carried towards developing a better migration plan during the transition phase and evaluating it against current transition mechanisms. Focus is to be put on software side rather than upgrading existing hardware architecture which will be economical from implementation perspective.