# Deploying IPv6 Service Across Local IPv4 Access Networks

ALA HAMARSHEH[1], MARNIX GOOSSENS[1], RAFE' ALASEM[2]
[1]Vrije Universiteit Brussel
Department of Electronics and Informatics ETRO
Building K, Office No: 4k222
Pleinlaan 2, 1050 Elsene
Belgium
{ala.hamarsheh, marnix.goossens}@vub.ac.be http://www.etro.vub.ac.be

[2] Imam Muhammad ibn Saud Islamic University
Computer Science Department
Office No: FR-90, Riyadh, 11432
Saudi Arabia
rafe.alasem@gmail.com http://www.imamu.edu.sa

*Abstract:* - This paper introduces a new protocol called D6across4, which stands for Deploying IPv6 Service across local IPv4 access networks. The protocol aims to encourage Internet Service Providers (ISPs) to start deploying IPv6 service to their customers (end-users). It utilizes the automatic IPv6-in-IPv4 tunnels to transport IPv6 traffic (data and control information) over local IPv4 access networks. The key aspects of this protocol include: providing IPv6 service to end-users equivalent to native one, stateful operation, and requiring simple configurations on both end-user's host and ISP sides at the time of setup. D6across4 connected hosts can communicate with other IPv6 hosts outside their local IPv4 access network. The simulation results showed that D6across4 performance parameters (e.g. latency and throughput) are acceptable in comparison to both IPv4 and IPv6 performance parameters.

*Key-Words:* - Local access networks, IPv6 deployment, IPv6-in-IPv4 tunnels, IPv6 host, IPv4 host, ISP, Latency, Throughput.

## 1 Introduction

Traditionally, IP addresses are used to uniquely identify network connected devices. The lack and shortage in IPv4 [1] available addresses lead to develop a new addressing scheme (IPv6 [2]) by Internet Engineering Task Force (IETF). The new IPv6 protocol has various advantages over existing IPv4 protocol. For example, it offers 128-bit addresses, better security (IPsec [3]), efficiency, QoS, mobility, etc.

Unfortunately, the two protocols are incompatible with each other, and changing the IP address structure effects on the whole network stack. Since the socket layer is the part of the network stack, it will be affected in this change as well. Most network applications are built to use specific type of socket layer functions, and changing the IP layer will lead to change all these functions in which they allow running applications to communicate over current host's network connectivity. In such a way, all applications that are using these old socket Application Programming Interface (API) functions need to be rebuilt in order to make them capable to work over new protocol. DAC [4] overcomes the incompatibility problems that might exist between network applications and host's connectivity. It allows any application type (IPv4/IPv6) to work over any communication protocol (IPv4/IPv6) without changing these applications in addressing capabilities.

However, there are many other areas of incompatibilities between IPv4 and IPv6 that arise while deploying IPv6 service. The task of adopting IPv6 is not an easy task and it is quit complex. Transition from IPv4 to IPv6 should appear in steps starting from few IPv6 nodes, and the number should gradually increase over the time until the whole network becomes IPv6 [5]. Therefore, the coexistence between IPv4 and IPv6 will remain for considerable amount of time. The IETF has proposed many transition mechanisms and specific types of IPv6 addresses to facilitate the communication between two nodes or networks operating on different IP architectures. Such mechanisms can be categorized into dual-stack, tunneling, and translation [6] [7] [8] [9].

This protocol uses tunneling technique to carry IPv6 traffic over local IPv4 access networks. Providing IPv6 connectivity to end-users requires changing/upgrading most network components on both sides (ISP and end-user sides).This protocol aims to provide IPv6 service to end-users with minimum change in the local IPv4 access network infrastructure (i.e. minimal possible cost). It does not require changing/upgrading network devices at the ISP side or Customer Edges (CEs). However it requires installing some network components at the ISP side, besides upgrading users' hosts by installing a particular application which is capable to encapsulate / de-capsulate the outgoing/ingoing traffic over IPv4 local access networks, as well as to locate the tunnel endpoint (tunnel server). ISPs can continue providing IPv4 connectivity to their customers alongside IPv6 connectivity via D6across4 protocol. With D6across4, all parties use a common IPv6 prefix. This makes the proposed mechanism distinguishable from other IPv6 deployment mechanisms. Fig. 1 shows D6across4 architecture deployed by an ISP.

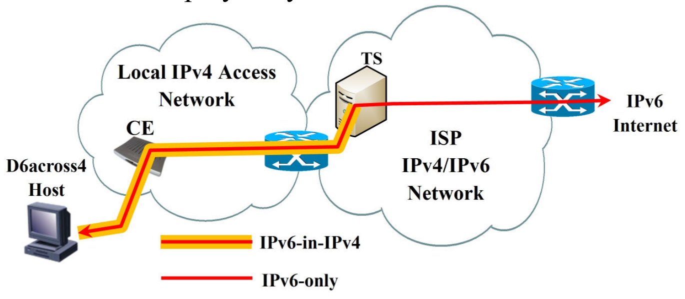


Fig. 1: D6across4 protocol architecture

## 2 Related Techniques

### 2.1 Connection of IPv6 Domains via IPv4 Clouds (6to4)

6to4 [10] is an automatic tunneling [11] technique that allows IPv6 packets to be transmitted over IPv4 network/Internet. Relay servers are used in this mechanism to connect 6to4 networks with other IPv6-only networks/Internet. 6to4 can be used by either hosts or routers. When it is configured at the host, such host must be assigned a global IPv4 address, and then it will be responsible for encapsulating the outgoing IPv6 traffic and decapsulates the incoming 6to4 packets. When it is configured at the router, it will be responsible for encapsulating/decapsulating the traffic. Finally 6to4 uses 2002::/16 prefix in creating IPv6 addresses, hence any IPv6 address begins with 2002::/16 prefix is considered as 6to4 address.

### 2.1 IPv6 Rapid Deployment (6rd)

6rd mechanism [12] [13] is used by ISPs to facilitate deploying IPv6 service across IPv4 networks. Like 6to4 mechanism, it is used to transfer IPv6 packets over IPv4 networks. Unlike 6to4, 6rd mechanism only operates within the ISP. 6to4 uses fixed IPv6 prefix (i.e. 2002::/16), whereas in 6rd mechanism, each ISP uses its own IPv6 prefixes which can reflect good network management and quality of service to end-users. 6rd tunnels are created between CEs and 6rd border relay router.

D6across4 protocol is similar in functionality to 6rd; however in D6across4 there will be no need to upgrade/change any of CEs. The end-users' hosts should be upgraded to work over D6across4 protocol. The protocol allows IPv6-in-IPv4 tunnels to be created between D6across4 based hosts and ISP tunnel server. Since there will be no need to change CEs at the customer side (minimal cost expected), this will encourage both parties (i.e. end-users and ISPs) to start adopting D6across4 protocol and hence, moving toward deploying IPv6.

## 3 ICMP Router Discovery and Solicitation Messages

In general, the ICMP router discovery messages are used by the host to discover the closed router(s), and they are used by routers to announce for their existence. These messages are categorized into ICMP router advertisement and solicitation messages. RFC 1256 [14] specifies the formats of ICMP messages. The following figures illustrate the format of the ICMP router solicitation and router advertisement messages exchanged between hosts and routers.
Traditional solicitation message format:

| | | |
|---|---|---|
| Type (10) | Code (0) | Checksum |
| Unused (sent as 0) | | |

Traditional advertisement message format:

| | | |
|---|---|---|
| Type (9) | Code (0) | Checksum |
| Number of addresses | Address ent. size (2) | Life time |
| Router address [1] | | |
| Preference level [1] | | |
| Router address [2] | | |
| Preference level [2] | | |
| ............ | | |

After exchanging a certain number of router advertisements and/or solicitation messages, the local host can configure its routing table to be able to locate its default router for its outgoing traffic.

As we can see in the previous figures, the values of type field in each of the solicitation and advertisement messages are 10 and 9 respectively, and the value of code field is left 0 for both of them.

## 4 D6across4 Protocol Specifications

D6across4 specifies a protocol mechanism to deploy IPv6 service to hosts via existing service provider's (ISP) IPv4 network. The proposed protocol relies upon an algorithmic mapping between the IPv6 and IPv4 addresses that are assigned for use within the ISP network. The tunnel endpoints could be automatically determined by resolving the Tunnel Server (TS) domain name. This protocol would help in handling the hardest scenario among all the scenarios in [15] number 4 (IPv4 network to the IPv6 Internet), hence its operation will be stateful (see later).

A D6across4 domain consists of customers' hosts and one or more TSs. Like 6to4 mechanism, the D6across4 based host encapsulates the IPv6 packets and forwards them to follow the IPv4 routing topology within the ISP network. Thereafter, the TS will decapsulate the received packets and forward them to the IPv6 network. The TS is traversed only when any of the D6across4 based hosts sends IPv6 packets or there are incoming IPv6 packets arriving from outside and destined to the ISP's D6across4 domain.

The protocol aims to use the IPv4 infrastructure to deliver IPv6 connectivity alongside native IPv4 with a little change to IPv4 networking and operation as possible. It is important to note that the ISP network can continue deploying the IPv6 within their network while delivering the IPv6 service to the end-users' hosts.

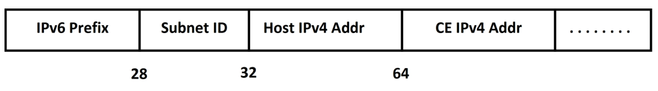
## 4.1 D6across4 Host Based Architecture and Behavior

D6across4 based hosts appear as native IPv6 hosts outside their local access networks. Such hosts must be able to automatically locate TS, create virtual IPv6 address, and (encapsulate/ decapsulate) the (outgoing/ingoing) traffic. D6across4 hosts are configured with an application called Tunneler. The Tunneler module is responsible for the followings:

- **Creating Virtual IPv6 address:**

IANA should reserve 28 bits well-known IPv6 prefix [16] belongs to this protocol. The virtual IPv6 address (i.e. not associated with IPv6 interface) for the host will be created by concatenating the D6across4 prefix and the consecutive set of bits from the IPv4 address of the host and the IPv4 address of the customer edge (e.g. ADSL router).

The following figure shows the format of IPv6 address (Section 2.5.4 of [2]) with a D6across4 prefix, host embedded IPv4 address, and customer edge embedded IPv4 address:



- **Locating Tunnel Endpoint:**

The ICMP router discovery messages (i.e. router advertisements and router solicitations messages) are used by the host Tunneler to periodically check the availability and durability of TS at the ISP side. The host Tunneler starts by resolving the domain name for the TS. Receiving a reply is an indication for the availability of D6across4 service at the ISP side.

To check the durability of the service at the ISP side, the host Tunneler generates a unicast ICMP router solicitation message destined to TS directly. Receiving a reply from TS is an indication of service availability. In order to make D6across4 ICMP messages distinguishable from traditional ICMP messages, a small change is applied by modifying the value of code field of ICMP messages from 0 to 1.

- **Encapsulation/Decapsulation:**

All outgoing IPv6 traffic that generated by D6across4 host will be encapsulated into IPv4 packets and forwarded over IPv4 infrastructure. The process of IPv4-in-IPv6 encapsulation goes as follows:

- **IPv6 Header:**
  - Source Address: the virtual IPv6 address.
  - Destination Address: the IPv6 address of the destination host.
- **IPv4 Header:**
  - Source Address: the D6across4 host IPv4 address (assigned by DHCPv4)
  - Destination Address: the TS IPv4 Address.

The Tunneler upon receiving the encapsulated packets, it will decapsulate them, and forward the generated IPv6 packets into IPv6 stack.

Forwarding manipulations and encapsulation techniques of IPv6-in-IPv4 are performed as described in [section 3.5 of [11]], with the same technique used by 6to4. Also the ICMPv4 errors are treated as described in [section 3.4 of [11]]. By default, the IPv6 traffic class field must be copied to the IPv4 Type of Service Field (ToS), or it could be overridden by configuration.

Fig. 2 shows D6across4 host architecture.

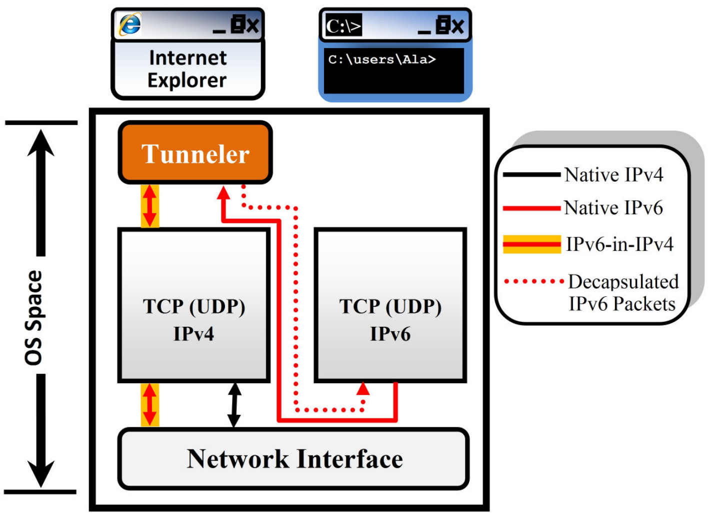


Fig. 2: D6across4 host architecture

## 4.2 ISP Network Configurations and Behavior

As mentioned earlier, ISPs can continue providing IPv4 connectivity to their customers alongside IPv6 connectivity via D6across4 protocol. Some users will be upgraded to work over D6across4 protocol and others will not. Thus, ISPs should be capable to provide different types of services (IPv4 and IPv6 services).

**D6across4 ISP should be configured with the following components:**

### • Alternative DNS:

Traditionally, each ISP has a local DNS (alternative DNS [17]) that used to resolve the domain names into IP addresses; hopefully this would be done locally. At the time of setup, the ISP network administrator should manually configure the local DNS and add a new record into the DNS records to denote for the TS. This allows the TS resolvable by other D6across4 hosts, fortunately this happens only once at the setup time. The TS must have a unique domain name added to the local DNS; this domain name must be the same for all ISPs deploying D6across4 protocol. The domain name must be chosen carefully so that it will not match any other domain name in the Internet (i.e. select a domain name with 256 characters). The ISP network administrator can assign any IP address for the TS domain name. Thus, D6across4 based hosts can automatically build their tunnels after resolving the TS domain name.

### • TS:

TS is responsible for decapsulating the received IPv6-in-IPv4 packets and then forwarding the generated IPv6 packets to their destinations inside IPv6 Internet. Furthermore, encapsulating the received IPv6 packets into IPv6-in-IPv4 packets and then forwarding them to the D6across4 host. TS always acts as a tunnel endpoint.

TS has at least one IPv4 interface and at least one IPv6 interface. Each TS has synthesized IPv6 address (IPv6 prefix + IPv4 address) assigned to it, and unicast IPv4 address.

The D6across4 is a stateful operation, this means it maintains a conceptual dynamic data structure to record address information needed for the packet submission and reception. The table structure and functionality is similar to the one that used in stateful NAT64 mechanism [18].

The following table illustrates the TS mapping table structure:

| SCR IPv4 | TS IPv4 | Host IPv4 | CE IPv4 | SCR Port | Mapped Port | SCR IPv6 | DST IPv6 |
|---|---|---|---|---|---|---|---|

The TS dynamic data structure is a mapping table to record the addressing information of the active connections initiated from D6across4 hosts and destined to IPv6-only Internet. TS is responsible for managing the mapping table (deleting or adding records). The mapping table should be supported by round-robin scheme to drop unused mappings for the longest time. This would be sufficient for operational situations.

TS extracts the host IPv4 address, CE IPv4 address, and source port number. All these values can be found in the IPv4 header of the received packet.

When the outgoing packet is received at the TS, the TS performs the following actions:

1. Copy the source IPv4 address from the IPv4 header.
2. Strip off the IPv4 header.
3. Extract the D6across4 based host IPv4 source and the CE IPv4 addresses from D6across4 IPv6 virtual address.
4. Add TS IPv6 address as source address.
5. Map the source port number into corresponding mapped port number.
6. Record all previous information in the TS mapping table.
7. Forward the extracted IPv6 packet to the IPv6 Internet.

When the reply IPv6 packet is received at the TS device arriving from IPv6 host located in the IPv6 Internet (because TS IPv6 address starts with D6across4 assigned prefix), TS performs the following actions:

1. Lookup in the mapping table to check if there was a previous mapping for the IPv6 destination and port number or not.
2. In case a mapping entry related to this packet is found, retrieve the related record immediately.
3. Use retrieved information to create D6across4 IPv6 virtual address.
4. Configure IPv4 and IPv6 headers as follows:
    - **IPv4 Header:**
        - Source Address: TS IPv4 Address.
        - Destination Address: Closest router IPv4 address (it got this address when the outgoing packet is received at the TS device).

- **IPv6 Header:**
    - Source Address: destination host IPv6 address.
    - Destination Address: D6across4 virtual address.

5. Forward the packet to D6across4 based host.

The IPv4 header is then added as it normally would be for any packet destined to D6across4 based host. That is, the IPv4 destination address is the D6across4 based host, and the source address is the TS IPv4 address. If the TS cannot find a record in the table, then it silently drops the packet. As explained earlier, both D6across4 based host and TS use the same IPv6 prefix in generating D6across4 virtual IPv6 addresses. TS is algorithmically configured to create D6across4 IPv6 virtual address using IPv6 assigned prefix. Moreover, the same prefix is used when synthesizing the IPv6 address of the TS device. The TS configures the source address of the outgoing IPv6 packet header by changing the source address with its synthesized IPv6 address. Fig. 3 shows a flowchart to illustrate the functionality of TS.

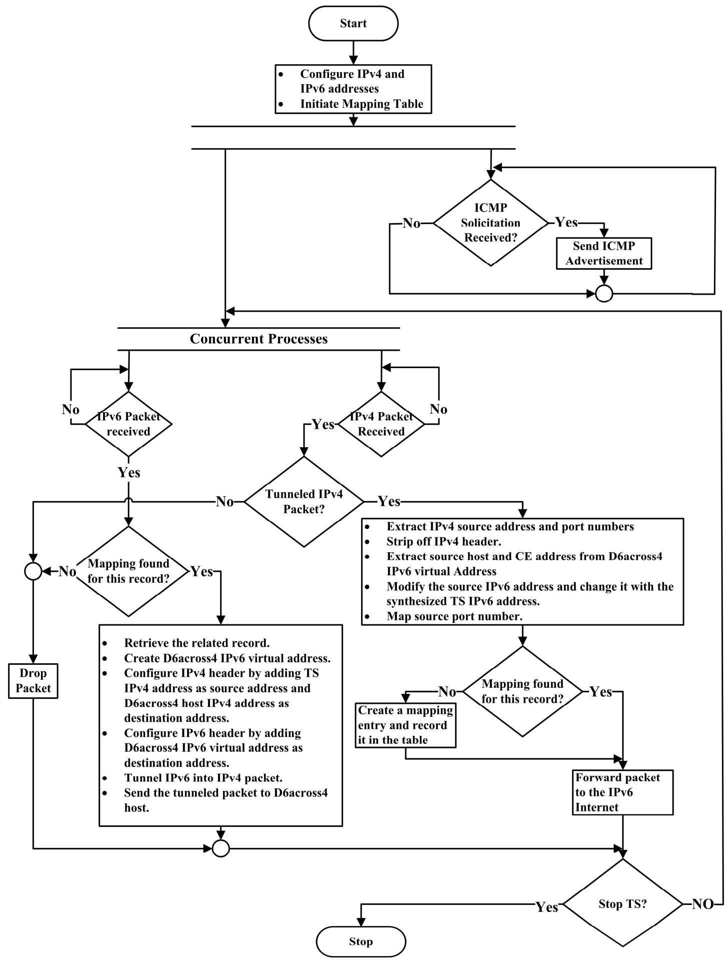

Start
Configure IPv4 and IPv6 addresses
Initiate Mapping Table
ICMP Solicitation Received?
No
Yes
Send ICMP Advertisement
Concurrent Processes
No
IPv6 Packet received
Yes
IPv4 Packet Received
No
Yes
Tunneled IPv4 Packet?
No
Yes
Extract IPv4 source address and port numbers
Strip off IPv4 header.
Extract source host and CE address from D6across4 IPv6 virtual Address
Modify the source IPv6 address and change it with the synthesized TS IPv6 address.
Map source port number.
Mapping found for this record?
No
Yes
Drop Packet
Retrieve the related record.
Create D6across4 IPv6 virtual address.
Configure IPv4 header by adding TS IPv4 address as source address and D6across4 host IPv4 address as destination address.
Configure IPv6 header by adding D6across4 IPv6 virtual address as destination address.
Tunnel IPv6 into IPv4 packet.
Send the tunneled packet to D6across4 host.
Mapping found for this record?
No
Yes
Create a mapping entry and record it in the table
Forward packet to the IPv6 Internet
Stop TS?
Yes
NO
Stop

Fig. 3: TS functionality

## 5 Performance Analysis

The OMNET++ network simulation framework [19] has been used to simulate D6across4 protocol. The goal of our simulation is to compare the performance of D6across4 based networks on with both IPv4 and IPv6 based networks. The performance is measured in terms of latency and throughput [20] parameters (see the next two subsections).

The simulator considers different packet sizes (64, 128, 256, 512, 768, and 1024) that are transmitted between the following scenarios:

a) D6across4 client and IPv6 server (D6across4 network).
b) IPv4 client and IPv4 server (IPv4 network).
c) IPv6 client and IPv6 server (IPv6 network).

In addition to the design architecture that appears in Fig. 1, the simulator measures the latency and throughput parameters of IPv6-only client while communicating with IPv6 server. Tests were conducted for each scenario by generating a traffic using TrafGen [21] tool. We used TrafGen tool to generate TCP traffic between client and server. The tool has the ability to define the parameters for certain traffic flows (i.e. packet size, destination, ON and OFF in seconds, etc.) between client and server. For the sake of reliability and consistency, every test was repeated 50 times. The reported value at a specified packet size is the average of all recorded values.

Series of simulation results have been performed in measuring the latency and throughput parameters at the above-mentioned packets' sizes. The D6across4 based network performance parameters (latency and throughput) are compared to both IPv6 and IPv4 based networks performance parameters. In general, the results showed that D6across4 parameters are roughly equal to both IPv4 and IPv6 performance parameters. However, D6across4 performance tests showed a small lower performance in comparison to IPv6 and IPv4 tests. We clarify this difference due to the overhead generated by encapsulating/decapsulating IPv6/IPv6-in-IPv4 packets and due to the frequent access to mapping table for each outgoing/ingoing packet arrived at TS.

### 5.1 Latency

The latency was measured as the time taken for a packet to be transmitted between D6across4/IPv6/IPv4 host and IPv6 server. The reported value of the latency at a specified packet size is the average of the recorded latency values. It was measured by sending packets of sizes varied from 64 to 1024 bytes (see Eq. (1)). Tests were generated for all protocols (D6across4, IPv6, and IPv4).

$$\text{Latency} = \frac{\sum_{i=1}^{N}\left(Td_i - Ts_i\right)}{N} \quad \text{……..…….. (1)}$$

Where,
Td: is the time at destination host.
Ts: is the time at source host.
N: number of received packet at the source host (round trip)

Fig. 4 shows a comparative latency for IPv6 packets transmitted over D6across4 and IPv6 networks, moreover for IPv4s packets transmitted over IPv4 network.

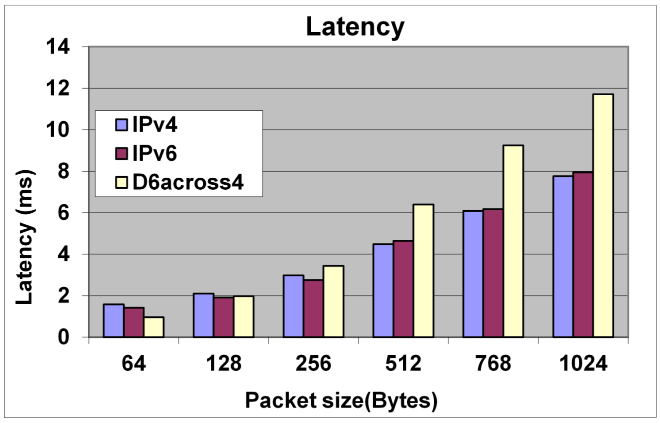


Fig. 4: Latency analysis

Fig. 4 shows a comparative latency values for IPv6, IPv4, and IPv6-in-IPv4 (D6across4) packets transmitted across IPv6, IPv4, and D6across4 protocols as packet size is varied from 64 bytes to 1024 bytes. In general the latency values for all conducted tests are nearly equal. D6across4 protocol showed small higher latency values among other protocols, this happened due to the extra overhead generated by encapsulating IPv6 packets and decapsulating IPv6-in-IPv4 packets.

### 5.2 Throughput

Throughput is the amount of data traffic that is transmitted over communication path per unit of time. It is calculated from Eq. 2.

$$\text{Throughput} = \frac{\sum_{i=1}^{50} \frac{N_{\mathrm{Rev}(i)} \times P_{size(i)} \times 8}{(T_{end(i)} - T_{start(i)}) \times 1 \times 10^6}}{50} \quad \text{……… (2)}$$

Where,
$N_{Rev}$: number of packets received at destination point.
$P_{size}$: packet size in bytes.
$T_{end}$: time at destination host (second).
$T_{start}$: time at source host (second).

In general, the throughput increases with the increase in the packet size. The reported value of the throughput at a specified packet size is the average of the recorded throughput values.

Fig. 5 shows a comparative throughput for IPv6, IPv4, and IPv6-in-IPv4 (D6across4) packets transmitted across IPv6, IPv4, and D6across4 protocols as the size of the packet varied from 64 to 1024 bytes.

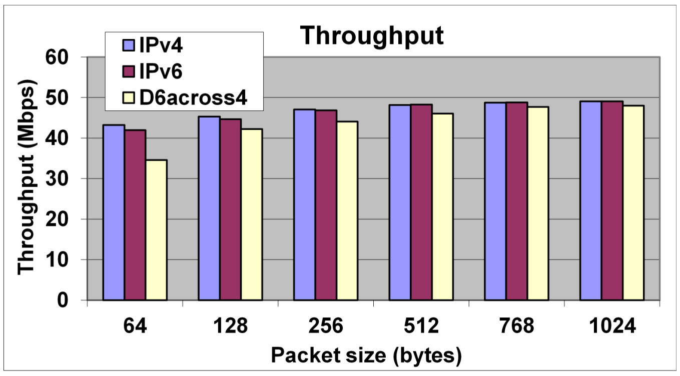


Fig. 5: Throughput analysis

In analyzing the throughput parameter among IPv6, IPv4, and D6across4 protocols, we noticed that the IPv4 throughput is better than other protocols especially in the lower packets sizes. The standard IPv6 header is 20 bytes larger than the standard IPv4 header, which produces an extra overhead. This is in fact an expected and known result, since the larger IPv6 header introduces some overhead, especially in relatively small packet sizes in the transmissions. When considering larger packets sizes, approximately both IPv4 and IPv6 protocols showed the same throughput values. On the other hand, D6across4 showed a lower throughput in comparison to other protocols. We clarify this throughout the extra overhead that was generated by encapsulation and decapsulation processes.

## 6 Conclusion

This paper introduced a new protocol for ISPs called D6across4. It allows ISPs to continue providing IPv4 service to their customers alongside IPv6 service via D6across4. The D6across4 based hosts will be able to communicate with other IPv6 hosts while they are connected to local IPv4 access networks. Many simulation tests were conducted in measuring and comparing the performance of D6across4 based network with both IPv4 and IPv6 based networks in terms of latency and throughput parameters. In general, all tests showed acceptable D6across4 performance in comparison to IPv6 and IPv4 protocols. However, the D6across4 protocol showed a lower performance compared with IPv4 and IPv6 protocols. The authors

believe that it is worth to use this protocol for the sake of smooth transition to IPv6.
The main drawbacks of this protocol are bottleneck and single point of failure at the TS module. However, the future work might go towards installing multiple TSs in the network. Then, develop an appropriate load balancing algorithm to distribute the IPv6-in-IPv4 tunnels among these TSs. Therefore, this will help in achieving a better quality of service in these networks and avoiding the single point of failure that caused by considering single TS in the network.

## 7 Acknowledgment

This work was funded by European Union, Erasmus Mundus External Cooperation Window (EMECW) programme under project number 141085-EM-1-2008-BE-ERAMUNDUS-ECW-L02, Belgium.

*References*

[1] Postel J., “Internet Protocol”, *Internet Engineering Task Force RFC 791*, 1981.

[2] Deering, S., R., Hinden, "IP Version 6 Addressing Architecture*", Internet Engineering Task Force RFC 4291*. 2006.

[3] S. Kent, K. Seo, “Security Architecture for the Internet Protocol”, *Internet Engineering Task Force RFC 4301*, 2005.

[4] A. Hamarsheh, M. Goossens, R. Alasem, “Decoupling Application IPv4/IPv6 Operation from the Underlying IPv4/IPv6 Communication (DAC)”, American Journal of Scientific Research, Eurojournals Press, Issue 14, PP. 101-121, 2011.

[5] J. Chen, Y. Chang, C. Lin, “Performance Investigation of IPv4/IPv6 Transition Mechanisms”, Journal of Internet Technology, Volume 5 No.2, pp. 163-170. 2004.

[6] B. Forouzan;”TCP/IP Protocol Suite”; McGraw-Hill, 2003.

[7] J. F. Kurose, K. W. Ross, ”Computer Networking A Top-Down Approach Feachering the Internet”, Pearson Education, New York, 2003.

[8] N. Oliver, V. Oliver; “Computer Networks Principle, Technologies and Protocols for Network Design”, John Wiley and Sons, England, 2006.

[9] J. Chen, Y. Chang, C. Lin, “Performance Investigation of IPv4/IPv6 Transition Mechanisms”, Journal of Internet Technology, Volume 5 No.2, pp. 163-170. 2004.

[10] B. Carpenter, K. Moore, “Connection of IPv6 Domains via IPv4 Clouds”, Internet Engineering Task Force RFC 3056, 2001.

[11] E. Nordmark, R. Gilligan, “Basic Transition Mechanisms for IPv6 Hosts and Routers”, Internet Engineering Task Force RFC 4213, 2005.

[12] W. Townsley, O. Troan, “ IPv6 Rapid Deployment on IPv4 Infrastructures (6rd) -- Protocol Specification”, Internet Engineering Task Force RFC 5969, 2010.

[13] R. Despres, “IPv6 Rapid Deployment on IPv4 Infrastructures (6rd)”, Internet Engineering Task Force RFC 5569, 2010.

[14] S. Deering, “ICMP Router Discovery Messages”, Internet Engineering Task Force RFC 1256, 1991.

[15] F. Baker, X. Li, C. Bao, K. Yin, “Framework for IPv4/IPv6 Translation”, Internet Engineering Task Force Draft, 2010.

[16] Bao C., Huitema C., Bagnulo M., Boucadair M., Li X., "IPv6 Addressing of IPv4/IPv6 Translators", Internet Engineering Task Force RFC 6052, 2010.

[17] Internet Architecture Board, “IAB Technical Comment on the Unique DNS Root”, Internet Engineering Task Force RFC 2826, 2000.

[18] M. Bagnulo, P. Matthews, I. van Beijnum,”Stateful NAT64: Network Address and Protocol Translation from IPv6 Clients to IPv4 Servers”, Internet Engineering Task Force Draft, 2010.

[19] “OMNeT++ Network Simulation Framework”, http://www.omnetpp.org.

[20] I. Raicu, S. Zeadally, “Evalating IPv4 to IPv6 transition mechanisms”. IEEE International Conference on telecommunications, 2003.

[21] Isabel Dietrich, “OMNET++ Traffic Generator”, http://www7.informatik.uni-erlangen.de/~isabel/omnet/modules/TrafGen/