An Empirical Analysis of IPv6 Transition Mechanisms

Myung-Ki Shin, Hyoung-Jun Kim, Darrin Santay, Doug Montgomery

ETRI, 161 Kajong-Dong, Yusong-Gu, Daejeon, 305-350, Korea
{mkshin, khj}@etri.re.kr
NIST, 100 Bureau Drive, Gaithersburg, MD 20899
{dougm, santay}@nist.gov

Abstract – **Numerous IPv6 transition mechanisms have been developed for supporting interoperability between IPv4 and IPv6. Although performance aspects of these mechanisms are requirements for practical deployment, they have yet to be empirically evaluated. In this paper we present the impact of IPv6 transition mechanisms on user application. Our experimental results show that though performance overheads were minimal, with small, fragmented and translation packets some performance degradation did occur.**

Keywords – **IPv6, Transition Mechanism, Application**

1. INTRODUCTION

A. Research Background

IPv6 is a new version of the Internet Protocol, designed as a successor to the current IPv4 [1]. IPv6 not only provides a larger IP address space, but also a number of fundamental features, such as security, mobility, extensibility and dynamic re-configurability, which are required by new devices. IPv6 is rapidly emerging as the preferred solution to meet the many needs of the evolving Internet.

One important key to a successful IPv6 transition from current IPv4 is the interoperability of new network nodes with the existing installed base of IPv4 nodes. The transition will be a long process during which both protocol versions will coexist. No general solution can be applied to the IPv4 to IPv6 transition process. Thus, numerous transition mechanisms have been developed and standardized with specific transition cases.

Several transition mechanisms have been developed and standardized to addresses specific transition and interoperability scenarios. As a practical matter, before we can introduce IPv6 transition mechanisms into real networks, we need to evaluate and prove that they will not adversely impact overall network security and performance. While the security implications are being extensively addressed in the IETF [2, 3], the performance implications are not well known. To date there are few reported results in this area, and those only address general performance evaluation on IPv6 protocols [4] and basic tunneling [5].

B. 2. Performance Degradation Prediction

In theory, we may predict the performance differences between IPv4, IPv6 and communications involving transition mechanisms. Basically, IPv6 has 1.41% and 1.40% higher header overhead for TCP and UDP, respectively, compared to IPv4 (for an Ethernet MTU size of 1,514 bytes) since IPv6 has a 40-byte header while IPv4 has a 20-byte header as shown in Tab. 1 (At this phase, we ignore the additional features, such as IPv6 extension headers and IPv4 options). In addition, most transition mechanisms use IP-in-IP encapsulation packets (e.g., IPv6-in-IPv4 or IPv4-in-IPv6) or IPv4-to-IPv6 translated packets. These mechanisms may incur additional overheads such as encapsulation, de-capsulation, checksum re-calculation and so on. For example, if IP-in-IP tunneling mechanisms are used in IPv6 transition mechanisms, IP header overhead will be increased to 4.81% as shown in Tab. 2. Header overhead analysis is a very simple and crude way to predict performance impacts.

We have found that the most significant differentiators of performance lie with more subtle details of protocol and transition mechanism operations. For example, if an IPv6 source sends packets larger than the path MTU, unlike IPv4, IPv6 fragmentation is performed only by the source nodes. This IPv6 behavior may improve performance on larger data transmissions. Knowing that transparency to end user applications is an important practical adoption criteria for most of these transition mechanisms, we examined the performance and behavior impact of several early implementations, including: configured tunneling [6], 6to4 [7], ISATAP [8], NAT-PT [9], DSTM [10], and Teredo [11].

Tab.1 IPv4 vs. IPv6 Header Overhead Comparison

	IPv4		IPv6	
	TCP	UDP	TCP	UDP
Ethernet Header(bytes)	14	14	14	14
IP Header (bytes)	20	20	40	40
TCP/UDP Header(bytes)	20	8	20	8
Data Payload(bytes)	1460	1472	1440	1452
IP Header Overhead(%)	1.36	1.35	2.77	2.75

Tab.2 Header Overhead Comparison between Transition Mechanisms

	IPv6-in-IPv4 Tunneling		IPv4-in-IPv6 Tunneling		(UDP)IPv6-in-IPv4	
	TCP	UDP	TCP	UDP	TCP	UDP
Ethernet Header(bytes)	14	14	14	14	14	14
Basic IP Header	40	40	20	20	40	40
Additional UDP Header	0	0	0	0	8	8
Tunneled Header(bytes)	20	20	40	40	20	20
TCP/UDP Header(bytes)	20	8	20	8	20	8

Data Payload(bytes)	1420	1432	1420	1432	1412	1424
IP Header Overhead(%)	4.25	4.18	4.25	4.18	4.81	4.77

2. MEASUREMENT TARGET AND PROCEDURES

A. *Measurement Target*

Each IPv6 transition mechanism was designed with different assumptions and objectives, and thus apply to different deployment scenarios. In order to meaningfully compare them, we have classified these mechanisms considering the following factors: architectural goal, scope, and scenarios. The mechanisms have one or more goals: (a) an IPv6 node connecting to an IPv6 node through an IPv4 network; (b) an IPv6 node connecting to an IPv6 node through an IPv4 network supporting NAT traversal; (c) an IPv4 node connecting to an IPv4 node through an IPv6 network; (d) an IPv6 node connecting to an IPv4 node; (e) an IPv4 node connecting to an IPv6 node. Also, these mechanisms should be introduced with a specific scope – inter-site or intra-site and scenarios - 3GPP, ISP, enterprise or unmanaged networks.

Tab. 3 shows the grouping results: IPv4-in-IPv6 Tunneling for Inter-site, IPv4-in-IPv6 Tunneling for Intra-site, Translation/IPv6-in-IPv4 Tunneling , and UDP tunneling.

Tab 3. IPv6 transition mechanisms classification results

Goal	Scope	Scenarios	Candidate Mechanisms	Grouping Results
(a)	Inter-site	3GPP, Enterprise, ISP	Configured tunnels[6], 6to4[7]	(1)
	Intra-site	Enterprise	ISATAP[8]	(2)
(b)	NA	Unmanaged, Enterprise	Teredo[11]	(4)
(c)	Inter-site	Enterprise, Unmanaged	DSTM[10]	(3)
	Intra-site	Not Defined Yet	No Candidates	NA
(d)	NA	3GPP, Enterprise	NAT-PT[9], DSTM[10]	(3)
(e)	NA	3GPP, Enterprise	NAT-PT[9], DSTM[10]	

B. *Testbed Configuration*

In order to empirically measure the impact of these transition mechanisms on end-to-end performance, we constructed a testbed using five systems, equipped with two AMD 32/64-bit Opteron 244 DP processors, 64-bit 800-MHz Front Side Bus, 1GB DDR333 RAM, and NetGear 100Mbps NICs. The two routers were installed with Linux (2.4.18 kernel) and the three hosts were configured as dual-boot configurations system running both Linux and Windows XP. Fig. 1 shows the complete testbed configuration.

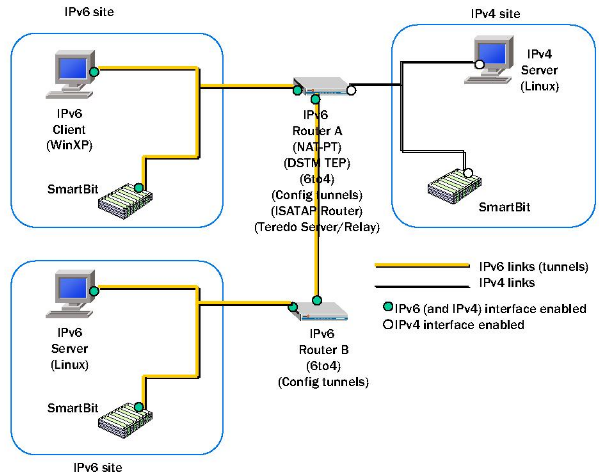

Fig.1 Testbed configuration

C. *Mechanisms Analysis*

To ascertain which mechanism operations cause overhead we analyzed them prior to measurement. Fig. 2 shows configured tunnels and 6to4 data packet transmission procedures. The overhead of Group-1 is mainly due to steps 3 and 5, which involve IPv6-in-IPv4 encapsulation/ decapsulation processes. IPv6-in-IPv4 tunneling mechanisms have higher header overhead (1.48% and 1.43% for TCP and UDP, respectively) than IPv6-only packet transmission. When an IPv6 packet is encapsulated in IPv4, with an Ethernet MTU size of 1,514 bytes, 40 bytes of IPv6 header are included. In case of Group-2, ISATAP, operations and encapsulation/decapsulation overhead are very similar to 6to4, but the ISATAP overhead is mainly due to step 1, 2 and 4, since ISATAP is implemented based on host-to-router tunneling mode, as illustrated in Fig. 3.

Also, we understand there might be a performance difference in Group 3, since NAT-PT overhead is mainly due to steps 2 and 3 while DSTM overhead is mainly due to steps 1, 2, 3 and 4 as shown in Fig. 4 and Fig. 5. Basically, DSTM has more header overhead than NAT-PT while NAT-PT performance is more dependent on a translator performance compared to DSTM.

Fig. 6 shows the procedure of Teredo, which uses a UDP tunneling mechanism. Teredo has 2.04% and 2.02% higher header overhead for TCP and UDP, when compared to IPv6-only packet transmission for an Ethernet MTU size of 1,514 bytes. The operational overhead for encapsulation/ decapsulation is similar with that of ISATAP, since the same host-to-router tunneling mechanisms are used.

Another important testing aspect is to figure out the fragmentation impact on each mechanism. If an IPv6 source attempts to send packets larger than the path MTU, unlike IPv4, fragmentation in IPv6 is performed only by source nodes. We need to see whether the mechanisms are benefited by the use of IPv6 since IPv6 behavior may improve performance on larger packet transmission. As well, all of the mechanisms would also be dependent of performance on IPv4 networks.

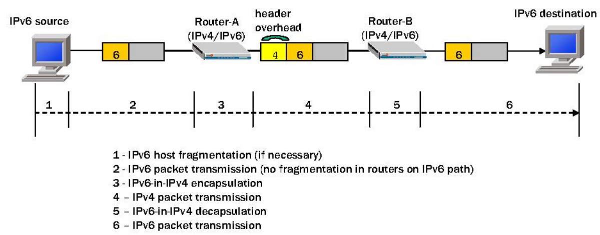

Fig. 2 Configured tunnels and 6to4 data packet transmission(router-to-router mode)

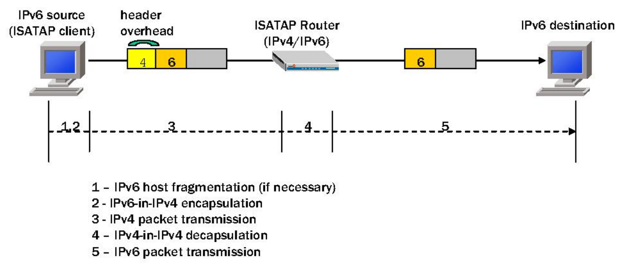

Fig. 3 ISATAP data packet transmission

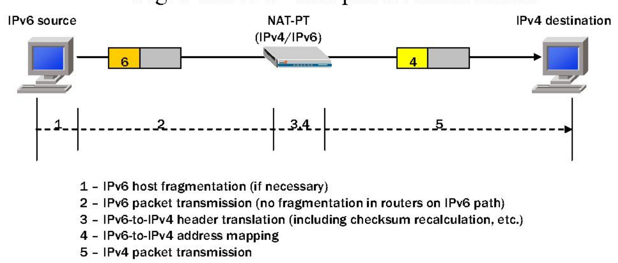

Fig. 4 NAT-PT data packet transmission

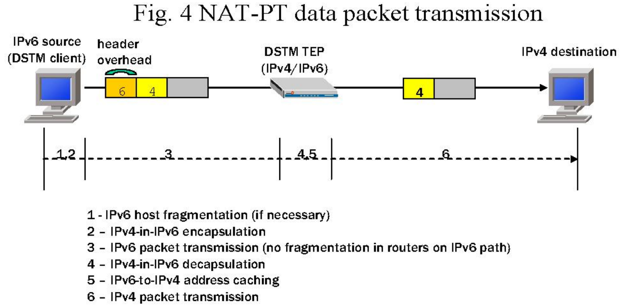

Fig. 5 DSTM data packet transmission

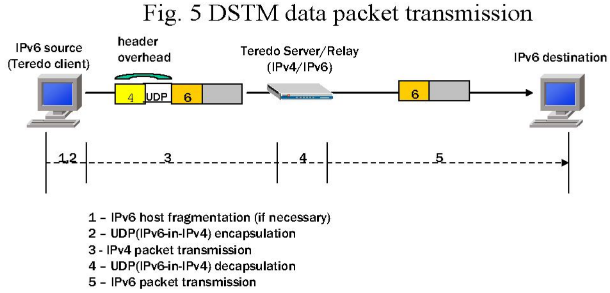

Fig. 6 Teredo data packet transmission

D. *Measurement Procedures*

Our primary performance metrics in this paper are throughput, CPU utilization, round-trip time, and connect/request/response transaction rate. We used iperf [12], netperf [13], ping and top [14] applications as measurement tools to see performance impacts on user applications. Experiments were executed for a period of 60 seconds and each test was repeated several times using data ranging from 64 to 3,072 bytes in order to see small data and fragmentation impacts on tunneling and translation mechanisms.

3. Results on End-to-End Application Performance

A. *Throughput*

Throughput is for measuring the TCP/UDP network throughput from one host to another. It is the rate at which bulk data transfers can be transmitted from one host to another for a long time period. Fig. 7 and Fig. 8 illustrate TCP and UDP application throughput for transition mechanisms in Group 1. As Fig.1 shows, with small data frames (e.g., 256 bytes or less), the difference on TCP between IPv6-only and IPv6-in-IPv4 tunneling was a considerable 36.2%. As we increased data sizes to link MTU and beyond, the differences were minimal. The curves for configured tunneling and 6to4 are very close to each other because they were implemented using the same tunneling techniques on Linux. The TCP transport protocol avoids fragmentation using the Maximum Segment Size (MSS) option. The UDP result shows minimal performance difference among the mechanisms, regardless of data size. The introduction of increased hops and/or heavier traffic in the path may increase the performance difference. Still, we found that IPv6-in-IPv4 tunneling (router-to-router mode) mechanisms have very little overhead. In addition, the results for configured tunneling and 6to4 were very similar as they too were implemented using the same tunneling techniques on Linux. Unlike TCP, UDP packets larger than MTU size were sent as a set of fragmented packets. We did not see fragmentation impacts on throughput of IPv4, IPv6, and IPv6-in-IPv4 tunneling mechanisms.

Fig. 9 shows TCP throughput for the Group 2 mechanism. The results were similar to those of Group 1, since they were implemented using the same IPv6-in-IPv6 tunneling mechanisms, though the TCP throughput difference was approximately 10% higher. This result appears to be due to the fact that ISATAP is very dependent of host performance (implementation) since the mechanism uses a host-to-router tunneling technique and our testbed configuration is different from that of Group 1 (see the difference between Fig. 2 and Fig 3). The UDP result of Group 2 was similar with that of Group 1.

Fig. 10 shows the TCP results comparing NAT-PT and DSTM. When we compare NAT-PT and DSTM with IPv6-only, with small packets, while NAT-PT causes about 7.7%, DSTM causes about 15.0% performance degradation on TCP throughput. But, when larger data was sent, DSTM overhead was more reduced, and DSTM performs better than IPv6-only, as well as NAT-PT. We think if data size is small, the IPv4-in-IPv6 tunneled header used in DSTM overhead may be more serious than IPv6-to-IPv4 header translation overhead, but when larger data was sent, DSTM overheads were more relatively decreased, compared with that of NAT-PT. We obtained a bit different result for UDP throughput as shown in Fig 11. The UDP result shows minimal performance difference among the mechanisms.

Teredo performance is illustrated in Fig. 12. Teredo is a complex mechanism that uses additional packets, including bubble packets and ICMP request/response packets, to manage its behavior. Also, the Teredo IPv6 address scheme includes additional information such as NAT type, port, IPv4 address, etc. Our results show that these Teredo mechanisms have a detrimental impact on performance, especially with small packets.

B. *Round-trip time (RTT)*

RTT is for measuring the length of time it takes to forward an ICMP request/response packet from one host to another. Fig. 13, Fig. 14. Fig 15, and Fig 16 illustrate RTT for Group 1, Group 2, Group 3 and Group 4, respectively. These figures imply that IPv6 performs better than IPv4 on larger data. This proves the IPv6 design goal may be realized (e.g., no fragmentation/checksum re-calculation on routers) when there are more hops and larger data transmission. It is also important to see there was no fragmentation/checksum re-calculation impact on IPv6-in-IPv4 tunneling (i.e., Group 1, Group 2 and Group 4) like IPv6-only, because fragmentation in IPv6 is performed only by source nodes. On the contrary, the Group 3 results show that they could not take the benefits of IPv6 on larger packet transmission. Nevertheless, DSTM performs better than IPv4 and NAT-PT, because DSTM packets are encapsulated in IPv6 with host fragmentation, even though applications generate IPv4 data packets. In addition, we found the overhead caused by IPv6-to-IPv4 header translation was increased on larger packets. It appears checksum re-calculation and fragmented header processing on NAT-PT router create additional delays. For the Group 3 case, (we call it IPv6-dominant network scenario), DSTM might be a lighter weight solution than NAT-PT, and, in particular, DSTM performs better than NAT-PT when there are larger data transmission and heavier traffic.

C. *CPU Utilization*

We measured CPU utilization increase on the router during the throughput tests. The results show the amount of increased CPU load the router used to process transition mechanisms. Tab. 4 and Tab. 5 show CPU utilization increase for Group 1 and Groups 2, 3 and 4, respectively. The results were very similar with results of throughput and round-trip time tests. Also, we see there was a considerable CPU utilization increase on transition mechanisms with small UDP packets processing. ISATAP and Teredo have a considerable CPU utilization increase though we feel the increase rate should have been similar with configured tunnels and 6to4. We believe this was due to problems in the current unofficial implementations on Linux. For example, we found the current ISATAP router implementation on Linux kernel 2.4.18 had problems with interrupt processing.

D. *Connect/request/response transaction rate*

It's for measuring transaction rate (e.g, transactions/sec). A transaction is defined as the exchange of a single request and a single response. Tab. 6 and Tab. 7 show TCP/UDP request/response and TCP connect/request/response transaction rate for Group 1 and Group 3, respectively. From a transaction rate, we can refer one-way round-trip average latency. Also, we can mimic the http protocol used by most web servers from TCP connect/request/ response transaction rate. The results were also very similar with results of round-trip time tests.

Tab.4 CPU utilization increase for Group -1 (%)

		IPv4 - only	IPv6 - only	Configured tunnels	6to4
TCP (average)		2.8	3.4	4.5	4.5
UDP	average	2.4	1.8	2.8	2.9
	data size < 128 bytes	10.1	12.7	16.5	16.9

Tab.5 CPU utilization increase for Group -2, 3, and 4 (%)

		ISATAP	NAT-PT	DSTM	Teredo
TCP (average)		15.2	5.8	5.1	9.2
UDP	average	12.51	3.5	2.9	10.5
	data size < 128 bytes	21.1	17.3	15.2	18.1

Tab. 6 Request/response transaction rate for Group 1 (bytes/sec)

	IPv4 - only	IPv6- only	Configured tunnels	6to4
TCP request/ response	2443.38	2442.19	2441.69	2439.70
UDP request/ response	2448.77	2443.12	2437.80	2434.36
TCP connect/ request/ response	2434.12	2432.28	2430.80	2429.58

Tab. 7 Request/response transaction rate for Group 3 (bytes/sec)

	IPv4 - only	IPv6- only	NAT-PT	DSTM
TCP request/ response	2474.69	2462.69	2456.02	2457.44
UDP request/ response	2472.32	2468.42	2462.02	2462.44
TCP connect/ request/ response	2465.41	2463.49	2454.01	2456.49

4. CONCLUSION

Our goal was not to evaluate a specific router or host, but to empirically analyze impacts on transition mechanisms compared with IPv4-only and IPv6-only network performance. There is significant research on the IPv6 protocol [4] and IPv6-in-IPv4 tunneling mechanism evaluations [5], but there is a lack of performance analysis of various transition mechanisms, even though these mechanisms are becoming more widespread as standards for IPv6 deployment.

Our experimental results show that though performance overheads were minimal, with small, fragmented and translation packets some performance degradation did occur. In this paper, we did not consider control planes for the mechanisms or performance impact on the mechanisms with security (e.g, IPsec). For the next steps, we intend to investigate them.

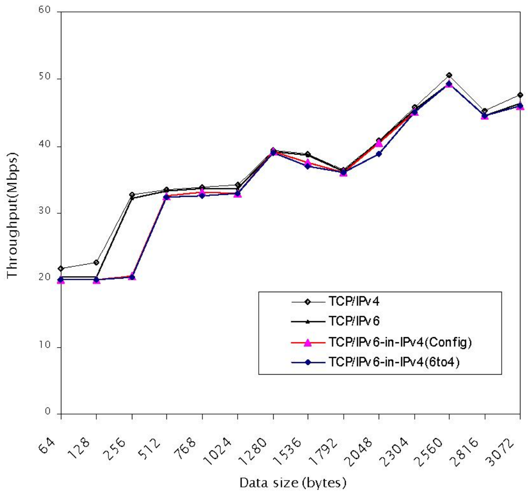

Fig 7. Group –1 TCP throughput results

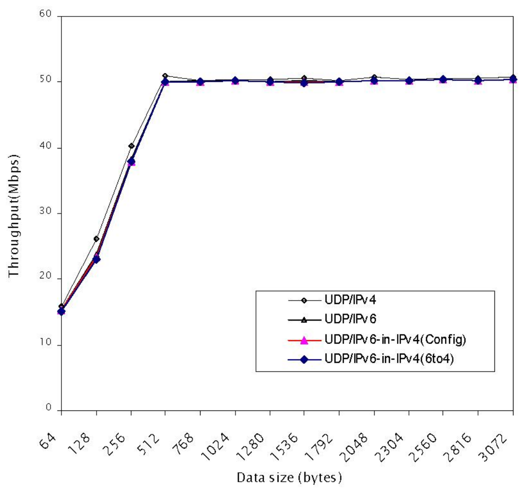

Fig 8. Group –1 UDP throughput results

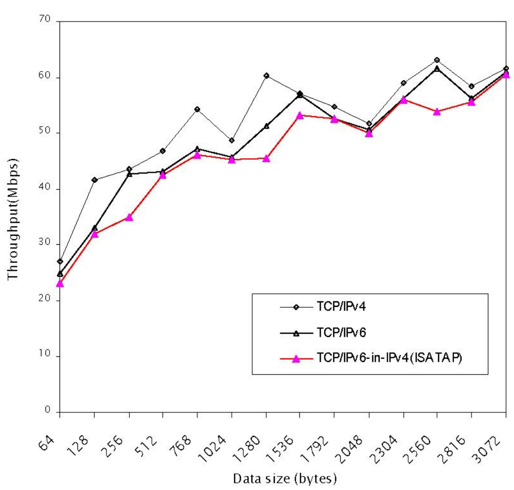

Fig 9. Group –2 TCP throughput results

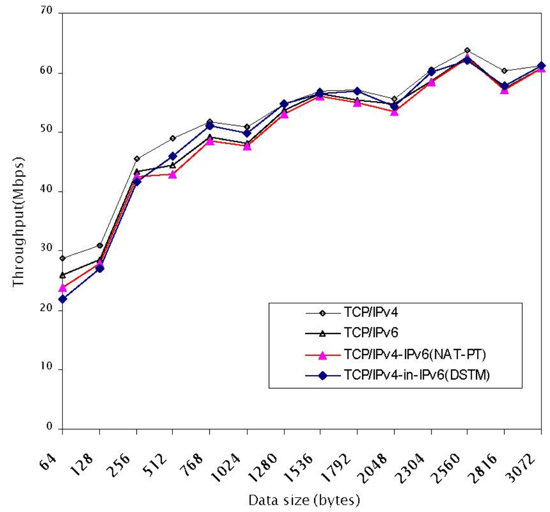

Fig 10. Group –3 TCP throughput results

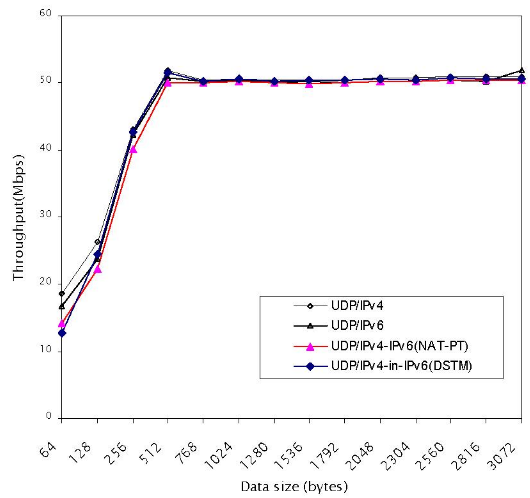

Fig. 11. Group –3 UDP throughput results

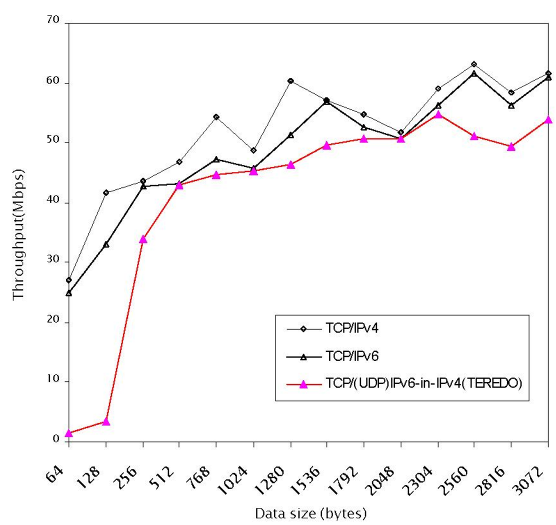

Fig. 12. Group –4 TCP throughput results

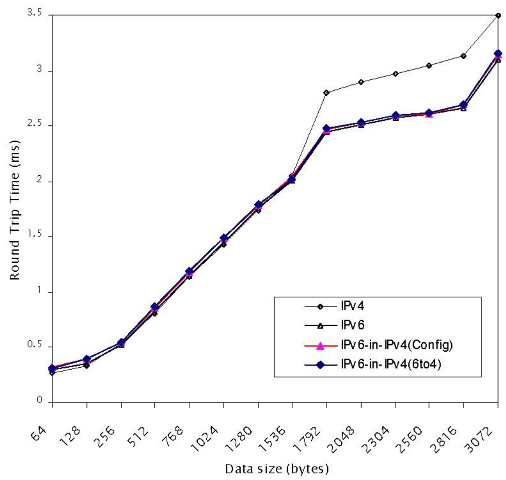

Fig. 13. Group –1 RTT results

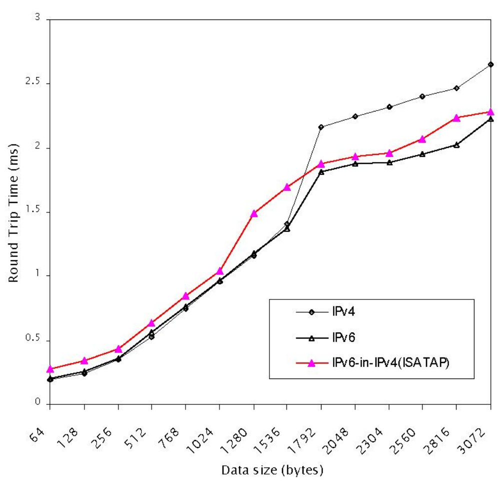

Fig. 14. Group –2 RTT results

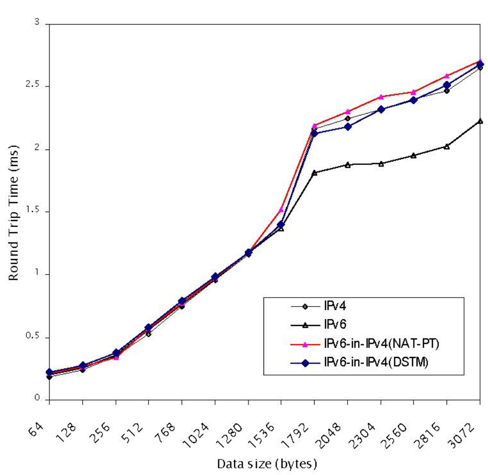

Fig. 15. Group –3 RTT results

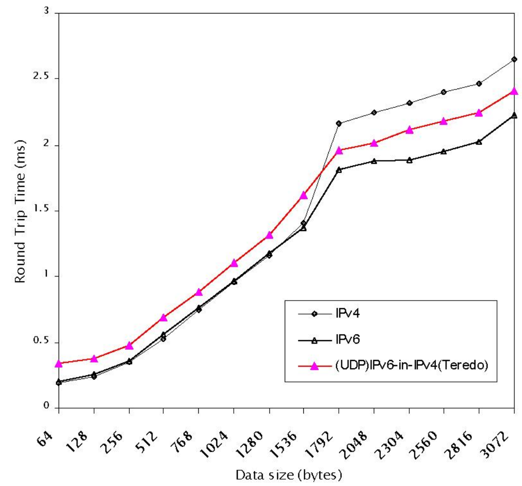

Fig. 16. Group –4 RTT results

REFERENCES

[1] S. Deering, R. Hinden, “Internet Protocol, Version 6 (IPv6) Specification”, RFC 2460, December 1998.

[2] E. Davies, S. Krishnan, P. Savola, “IPv6 Transition/Co-existence Security Considerations,” <draft-savola-v6ops-security-overview-03.txt>, October 2004, Work-in-Progress.

[3] G. Van de Velde, T. Hain, R. Droms, B. Carpenter, E. Klein, “IPv6 Network Architecture Protection,” <draft-ietf-v6ops-nap-00.txt>, March 2005, Work-in-Progress.

[4] S. Zeadally, I. Raicu, “Evaluating IPv6 on Windows and Solaris,” IEEE Internet Computing, pp. 51-56, May/June 2003.

[5] I. Raicu, S. Zeadally, “Evaluating IPv4 to IPv6 Transition Mechanisms, ICT 2003, February 2003.

[6] E. Nordmark, R. E. Gilligan, “Basic Transition Mechanisms for IPv6 Hosts and Routers,” <draft-ietf-v6ops-mech-v2-06.txt>, September 2004, Work-in-Progress.

[7] B. Carpenter, K. Moore, "Connection of IPv6 Domains via IPv4 Clouds," RFC 3056, February 2001.

[8] F. Templin, "Intra-Site Automatic Tunnel Addressing Protocol (ISATAP)," <draft-ietf-ngtrans-isatap-22.txt>, May 2004, Work in Progress.

[9] G. Tsirtsis, P. Srisuresh, "Network Address Translation-Protocol Translation (NAT-PT)," RFC 2766, February 2000.

[10] J, Bound, et al., Dual Stack Transition Mechanism,” <draft-bound-dstm-exp-02.txt>, January 2005, Work in Progress.

[11] C. Huitema, "Teredo: Tunneling IPv6 over UDP through NATs," <draft-huitema-v6ops-teredo-04.txt>, January 2005, Work in Progress.

[12] Iperf, http://dast.nlanr.net/Projects/Iperf/

[13] Netperf, http://www.netperf.org/netperf/NetperfPage.html

[14] Top, A Top-CPU Usage Display, http://www.groupsys.com/topinfo/

[15] “QoS Performance Tester for SmartBits, SmartFlow,” 2004, Spirent Communications, Inc., http://www.spirentcom.com/documents/94.pdf