Quantifying the Performance Degradation of IPv6 for TCP in Windows and Linux Networking

Burjiz Soorty
School of Computing and Mathematical Sciences
Auckland University of Technology
Auckland, New Zealand
burjizsoorty@hotmail.com

Nurul I Sarkar
School of Computing and Mathematical Sciences
Auckland University of Technology
Auckland, New Zealand
nurul.sarkar@aut.ac.nz

Abstract – **Implementing IPv6 in modern client/server operating systems (OS) will have drawbacks of lower throughput as a result of its larger address space. In this paper we quantify the performance degradation of IPv6 for TCP when implementing in modern MS Windows and Linux operating systems (OSs). We consider Windows Server 2008 and Red Hat Enterprise Server 5.5 in the study. We measure TCP throughput and round trip time (RTT) using a customized testbed setting and record the results by observing OS kernel reactions. Our findings reported in this paper provide some insights into IPv6 performance with respect to the impact of modern Windows and Linux OS on system performance. This study may help network researchers and engineers in selecting better OS in the deployment of IPv6 on corporate networks.**

Keywords: Bandwidth, IPv6, operating systems, packet length, transmission control protocol (TCP)

I. Introduction

Transmission Control Protocol and Internet Protocol (TCP/IP) are the most widely used Internet protocols which are built into modern MS Windows and Linux OSs. There is an effort in migrating to IPv6 as is evidenced by celebrating World IPv6 launch day worldwide [1]. This deployment of IPv6 is occurring side by side with the growth of Gigabit Ethernet in commercial networks alongside the release of the newest Windows and Linux OSs. Therefore it is important to evaluate IPv6 using the latest OS developments.

In this paper we quantify the performance of IPv6 against IPv4 using Windows Server 2008 and Red Hat Enterprise Server 5.5 in a peer-to-peer Gigabit Ethernet local area network (LAN). These operating systems were selected based on their popularity and relevance to our study. Based on these implementations, we aim to shed some light on IPv6 performance for TCP. Furthermore, a very limited research on evaluating IPv6 using newer client-server OSs in Gigabit Ethernet motivates us to contribute in this area and formulate this paper.

The results of this study will be crucial to primarily those organizations that aim to achieve high IPv6 performance via a system architecture that is based on newer Windows or Linux OSs. The analysis of our study further aims to help researchers working in the field of network traffic engineering and designers overcome the challenging issues pertaining to IPv6 deployment.

II. Related Work

In this paper, we briefly review a set of literature on IPv6 performance evaluation and related issues.

Kolahi and Li [2] have evaluated IPv6 over an 802.11n network. They studied the network throughput and packet delay of IPv4 and IPv6 using and earlier version of Windows and Linux operating systems (e.g. Windows XP and Fedora 12). Kolahi and Li concluded that Fedora 12 performs better than Windows XP, however, they did not take into account a client-server network scenario and therefore the results obtained could not be translated to gauge real-world network performance such as those setup in a typical campus and commercial (Corporate and branch-offices) Gigabit Ethernet LANs. Furthermore, they did not analyze address as to why the OSs resulted in lower throughput and higher packet delays and what implementations could be made to improve it.

Another study conducted by Kolahi et al. [3] to evaluate IPv4 and IPv6 in peer-to-peer and client-server networks. This study also measured TCP throughput and delay over Windows Vista and Windows Server 2008. Their findings showed that IPv4 to perform better than IPv6 by 2.7% on a peer-to-peer setup and by 4.6% on a client-server network. Similar such studies have been carried out for IPv4 and IPv6, some evaluating over different OSs [3] whereas others over cabling systems [4] and wireless systems [5]. An earlier work evaluated IP performance using open source OSs [6]. They evaluated IPv6 using Windows Server 2003, Red Hat 9 and FreeBSD4.9. The relatively poor performance of the IPv6 on Windows 2003 was due to the datagram fragmentation greater than 1440 bytes.

Earlier works on IPv6 have focused on developing as well as evaluating address lookup algorithms for IPv6. Li and Pao [7] have evaluated three well-known approaches for IPv6 address lookup, namely the tree-based approach, the range search approach, and the hash-based approach using empirical studies. They compared the performance of three address lookup approaches using metrics, such as memory requirements, mean update time, and packet processing rate.

Table 1 lists the key researchers and their main contributions in evaluating IPv6.

TABLE I: KEY RESEARCHERS AND THEIR MAIN CONTRIBUTIONS IN EVALUATING IPV6

Researchers	Year	Performance Metric	Transport Protocol	Type of LAN
Kolahi et al. [2]	2011	Throughput and packet delay on Fedora Linux 12.0	TCP, UDP	802.11N Wireless LAN
Kolahi et al. [3]	2010	Throughput and packet delay on Windows Vista with Windows Server 2008	TCP, UDP	Fast Ethernet
Mohammed et al. [6]	2006	Throughput and Packet delay on Win. Server 03, Red Hat 9.0, FreeBSD 4.9	TCP	Gigabit Ethernet
Li and Pao [7]	2006	Memory requirements, mean update time, packet processing rate on Linux systems	TCP	Ethernet

All the papers reviewed in this section only evaluated the transport layer protocols (e.g. TCP or UDP) over IPv6 with very little or no supposition and hypothesis as to why the performance is high or why performance degrades and where exactly the bottlenecks lie.

Our main contribution in this paper is to quantify the performance of IPv6 for TCP using newer Windows and Linux client/server OSs (Windows Server 2008, and Red Hat Enterprise Server 5.5) for which no published work is available. We obtain new results and quantify the performance degradation of IPv6 with respect to IPv4 in peer-to-peer Gigabit Ethernet. We not only quantify the performance degradation but also discuss the implications for system design and deployment.

III. TESTBED MEASUREMENT AND PROCEDURE

A. *Testbed Configuration*

Figure 1 shows the experimental setup (test-bed) for evaluating IPv6 performance. The network topology is a peer-to-peer Gigabit Ethernet link between a client and a server machine. To avoid additional delays caused by switching/routing devices, we did not use any routers in the test-bed measurements. This allows us to focus on IPv6 evaluation and to investigate the impact of modern OSs on system performance more accurately. In this regards, all the unwanted services (running on default) that consume network bandwidth were disabled. No third-party applications were used to optimize network performance.

The experimental setup consisted of four machines that surpassed the minimum and recommended settings for the selected OSs tested on them. We connected two Windows machines (e.g. Windows 7 connected to Windows Server 2008) using a Category 6 crossover cable. The separation between the client and server was set to 1 meter as suggested by network researchers [3-5]. We use the same connection for linking two Linux machines (Ubuntu 10.04 and Red Hat Server 5.5).

These machines had identical hardware configurations: Intel® Core™ 2 Duo processors with 4 GB 800 MHz DDR-2 Corsair® RAM modules. All four machines had Gigabit Ethernet (GBE) network interface cards. To eliminate the effect of network performance associated with hardware process and design, we benchmarked the hardware and use the same setup for all experiments conducted. Several repeated tests reveal that the native hardware configuration met the recommended OS settings.

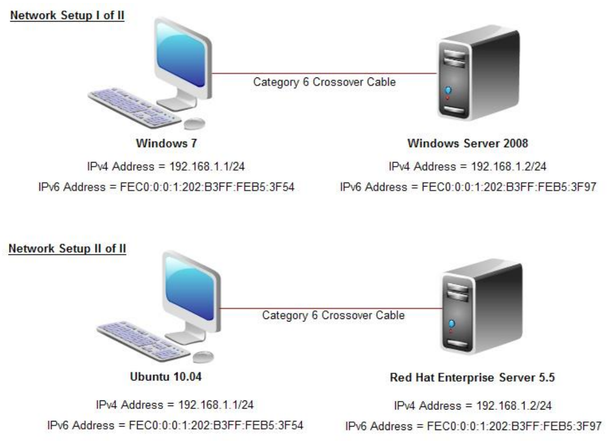
Network Setup I of II
Category 6 Crossover Cable
Windows 7
IPv4 Address = 192.168.1.1/24
IPv6 Address = FEC0:0:0:1:202:B3FF:FEB5:3F54
Windows Server 2008
IPv4 Address = 192.168.1.2/24
IPv6 Address = FEC0:0:0:1:202:B3FF:FEB5:3F97
Network Setup II of II
Category 6 Crossover Cable
Ubuntu 10.04
IPv4 Address = 192.168.1.1/24
IPv6 Address = FEC0:0:0:1:202:B3FF:FEB5:3F54
Red Hat Enterprise Server 5.5
IPv4 Address = 192.168.1.2/24
IPv6 Address = FEC0:0:0:1:202:B3FF:FEB5:3F97

Figure 1. Network Testbed setup for evaluating IPv6

B. *Measurement Tools and Metrics*

Several data generating and traffic measuring tools were researched for the purpose of evaluating IPv6 and IPv4 on Windows and Linux systems. For measuring the performance of Windows client-server networks, IP Traffic [8] was used because of its popularity and measurement accuracy [2-5]. Another motivation of using IP Traffic was to compare the results with previous studies. IP Traffic has also been used in the previous studies [2, 5].

Iperf [9] has been used as a primary tool to evaluate both IPv4 and IPv6 in Linux client-server networks [10]. However, it did not support Windows networking and therefore we modified the settings of Iperf to match with IP Traffic [9]. Iperf is an open-source network performance measurement tool that creates both TCP and UDP data streams to measure network throughput and packet delays.

For each observation, a total of one million packets were sent over the Gigabit Ethernet link using IP Traffic and Iperf. Ten such runs were recorded for both Windows and Linux Servers. A total of ten million packets were sent for measuring RTT of both IPv6 and IPv4. The same method was used to measure the link throughput and RTT of Ubuntu 10.04 and Red Hat Server 5.5.

IV. RESULTS AND DISCUSSION

We measure empirically both TCP throughput and RTT. Throughput is a measure of system's capacity (i.e. actual data rate as opposed to theoretical data rate) and is the most crucial metric in terms of core system performance.

Figure 2 compares TCP throughput (in Mbps) for IPv6 and IPv4 on Windows and Linux Servers at packet length of 768 bytes. We observe that IPv4 achieved slightly higher throughput than IPv6 for both Windows and Red Hat Servers. We also observe that for both IPv4 and IPv6, TCP throughput is consistently higher for Red Hat Server 5.5 than Windows Server 2008.

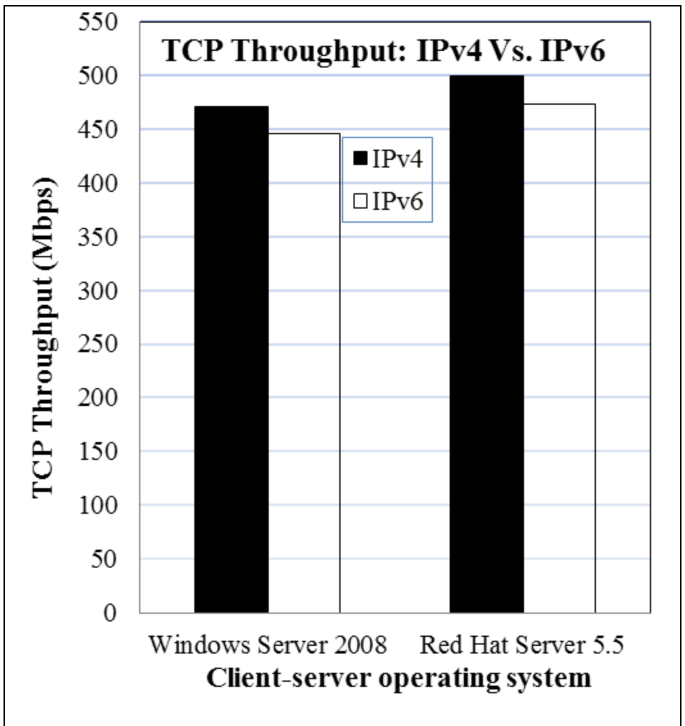

Figure 2. TCP throughput comparison of IPv6 and IPv4 for both Windows Server 2008 and Red Hat Enterprise Server 5.5.

Let us now quantify TCP throughput over IPv6 Wndows and Linux networking. The mean throughput was computed by taking an average of all the throughput measurements obtained at the packet length of 768 bytes. We found that IPv6 achieved a 5.2% lower throughput than IPv4 on both, the Windows and Linux networks. We also found that Red Hat Server 5.5 achieved approximately 6% higher throughput than Windows Server 2008 for both IPv6 and IPv4. This is because Red Hat 5.5 Server kernel is more efficient with respect to overall IPv6 throughput. The higher throughput on IPv6 for Red Hat Server is a result of the high TCP send/receive buffer in its kernel. This buffer size can be modified to accommodate more packets based on the type and length of a packet.

Customizing the send/receive buffer in the kernel accordingly can enable TCP segments to be sent/received faster per unit of time in-order to gain good client-server communications in achieving higher throughput. Potential to increase the gain in throughput in Windows Server may exist by means of customizing the IP handler to define and accommodate larger packets for services pertaining to TCP segments. Due to the 'closed-source' nature of Microsoft OSs, any such tweaking or development is restricted to the system engineers and only open to an internal MS team.

By looking at Figure 2, one can observe that TCP link throughput is slightly higher for IPv4 than IPv6 for both Windows and Linux systems at packet length of 768 bytes. This is because IPv6 deteriorates throughput as a result of its high transmission overheads (i.e. larger header). However, even though IPv6's header is larger but it is much simpler than IPv4's header which explains why the difference in throughput keeps getting lower with the increase in packet length (more payload delivery).

The main conclusion is that IPv6's TCP throughput is lower than IPv4 on both Windows and Linux networking and for smaller packet lengths. However this throughput degradation becomes insignificant as we increase packet lengths. This observation is critical in terms of packet crafting. For instance, with open source software (OSS) such as a Linux, the system kernel is open for development to anyone. In such instances where OSS systems are implemented to structure an open-source client-server environment, system engineers could craft packets by increasing payload data and setting the packet-length of 1408 bytes. By comparing Windows and Linux, we also found that Linux performs significantly better overall (i.e. higher throughput). The main reason for this can be attributed largely due to the kernel implementations in Red Hat Server.

We now focus on RTT performance of IPv6. RTT is a measure of latency or packet delay from a sending node to a destination node across the network. Figure 3 compares TCP RTT for IPv6 and IPv4 using Windows and Red Hat Servers.

For IPv4, the lowest RTT (1.73 ms) was recorded for Windows Server at packet length of 768 bytes. In contrast, Red Hat Server obtained RTT of 1.84 ms. The RTT difference is about 6% (Windows Server is better in achieving lower RTT). The difference becomes smaller (i.e. insignificant) as we increase the packet length. For example, Windows Server achieves less than 5% lower RTT than Red Hat Server at packet length of 1408 bytes.

For IPv6, the lowest RTT (2.1 ms) was recorded for Windows Server at packet length of 768 bytes. In contrast, Red Hat Server had RTT of 2.34 ms. The difference in RTT between Windows and Red Hat Servers is about 10% (Windows Server 2008 is better in achieving lower RTT). However, RTT difference between Windows and Red Hat Server becomes smaller as we increase packet lengths.

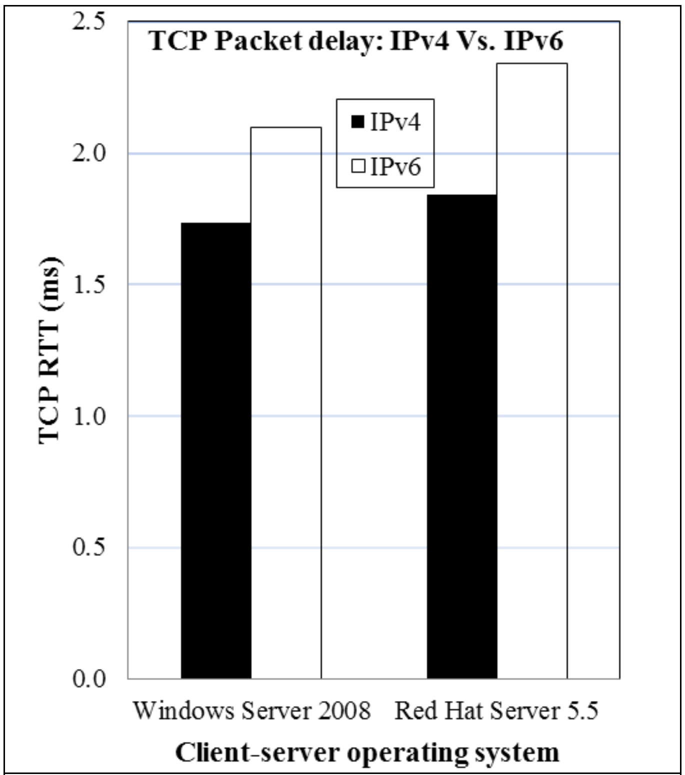

Figure 3. TCP RTT comparison of IPv6 and IPv4 for both Windows Server 2008 and Red Hat Enterprise Server 5.5.

V. Practical Implications

In this paper, we investigated the data performance of IPv6 for TCP over Windows Server 2008 and Linux Red Hat Server 5.5. The aim of our research was not only evaluate the performance of those systems but also to quantify the performance difference between IPv6 and IPv4 with respect to modern Windows and Linux OSs. We discussed the reasons for performance differences and what changes could be made to help improve system performance. We measure IPv6 performance using a customized test-bed setting and our findings could be useful for network engineers and designers for selecting the right client-server OS in the deployment IPv6. Our research furthermore analyzed the limitations in the newer operating systems to justify the performance degradation over IPv6 and what approach software developers and system engineers could take to rectify and improve IPv6 performance.

To improve IPv6 throughputs, system engineers could craft packets to increasing payload by setting appropriate packet-lengths. This would result in a slight increase in packet delays, but significant throughput gain can be achieved. This study can also help network developers working on open source projects to configure Linux kernel to further improve IPv6 efficiency. For instance, by increasing buffer-size in the socket and thereby accommodating more packets to be delivered and consequently improve the performance of IPv6 on Red hat Linux Server.

In conclusion, a higher throughput can be achieved as a result of lower packet fragmentation and customizing the kernel to force fragmentation to occur at higher-packet lengths. Such form of packet crafting would be efficient for services and applications involving data transfer.

VI. Conclusions

In this paper we quantify the performance gain and/or degradation of IPv6 with respect to IPv4 in peer-to-peer Gigabit Ethernet Windows and Linux networking. Our research findings revealed that IPv6 throughput was about 6% higher over the Linux based client-server architecture using Red Hat Enterprise Server than the Windows network running Windows Server 2008. IPv6 throughput degrades by approximately 5% due to its high transmission overheads with respect to IPv4. This degradation was insignificant when the packet-length was set for encapsulation at 1408 bytes. Analyzing the kernel’s TCP/IP stack on each OS, we found that the Linux kernel processed IPv6 packets more efficiently thereby resulting in higher overall throughput. We also measured RTT for both IPv4 and IPv6 and found that IPv6 degrades performance (about 21% higher RTT in Red Hat Server) with respect to IPv4. When comparing the two networks, Windows Server 2008 had a lower RTT than Red Hat Server 5.5. This could be due to a higher queue buffer to the Red Hat Server. However, a lower RTT on Red Hat can be achieved by TCP segment fragmentation, decreasing the Maximum Transmission Unit, and customizing the kernel to force fragmentation to occur at smaller packet lengths. Such form of packet crafting would be efficient for services and applications involving delay sensitive information such as voice and video authentication. Future works on TCP could include measuring memory (RAM) usage by the kernel and further behavioral analysis of TCP structures over IPv6. Other methods of TCP tuning could also be investigated.

References

[1] S. Ahmed. (2013, Jun. 04). Taking the rocky road to IPv6. Available: http://computerworld.co.nz/news.nsf/news/taking-the-rocky-road-to-ipv6

[2] S. S. Kolahi and P. Li, "Evaluating IPv6 in Peer-to-Peer 802.11n Wireless LANs," *IEEE Internet Computing*, vol. 15, pp. 70-74, 2011.

[3] S. S. Kolahi and B. K. Soorty, "Evaluation of Gigabit Ethernet Local Area Networks in Windows Vista-Server 2008 Environment," in *2011 IEEE Workshops of Int. Conf. Advanced Information Networking and Applications (AINA)*, pp. 308-312.

[4] B. K. Soorty, S. S. Kolahi, N. Chand, and Z. Qu, "Performance Comparison of Category 5e vs. Category 6 Cabling Systems for both IPv4 and IPv6 in Gigabit Ethernet," in *10th IEEE Int. Conf. Computer and Information Technology (CIT)*, Bradford, West Yorkshire, UK, 2010, pp. 1525-1529.

[5] S. S. Kolahi, Z. Qu, B. K. Soorty, and N. Chand, "The Impact of Security on the Performance of IPv4 and IPv6 Using 802.11n Wireless LAN," in *3rd Int. Conf. New Technologies, Mobility and Security (NTMS)*, Cairo, Egypt, 2009, pp. 1-4.

[6] S. S. Mohamed, M. S. Buhari, and H. Saleem, "Performance comparison of packet transmission over IPv6 network on different platforms," *IEE Proc. Comm.*, vol. 153, pp. 425-433, June 2006.

[7] Y. K. Li and D. Pao, "Address lookup algorithms for IPv6," *IEEE Proc. Comm.*, vol. 153, pp. 908-919, 2006.

[8] Z. Telecom. (2013, Feb. 27). *IP Traffic - Test & Measure*. Available: http://www.zti-telecom.com

[9] NLANR/DAST. (2013). *Iperf*. Available: http://iperf.sourceforge.net/

[10] B. Soorty and N. I. Sarkar, "Evaluating IPv6 in peer-to-peer Gigabit Ethernet for UDP using modern operating systems," in *2012 IEEE Symp. Computers and Communications (ISCC)*, Cappadocia Turkey, pp. 534-536.