Quantifying TCP Performance for IPv6 in Linux-Based Server Operating Systems

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 TCP performance for IPv6 on two open source systems (OSs), namely, the Linux based server operating systems - Red Hat Enterprise Server (RHES) and Ubuntu Server. We measure and evaluate the key parameters influencing OS behavior and network performance, including TCP throughput, round trip time (RTT), CPU usage and jitter by observing OS kernel reactions. Our findings reported in this paper provide some insights into IPv6 performance with respect to the impact of modern and commonly used Linux server operating systems. 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. With the advent of IPv6 and the recent end to IPv4 addresses post-2012, there is an effort in migrating to IPv6 as is evidenced by celebrating World IPv6 launch day worldwide. This upgrade to the next generation Internet protocol has not only established a secure means of communication across the Internet but also resulted with a more efficient means to packet processing and routing due to a more enhanced and simplified packet header.

Deployment of IPv6 is occurring side by side with the growth of Gigabit Ethernet in commercial networks alongside the release of the newest Linux and Windows OSs. Therefore it is important to evaluate IPv6 using the latest OS developments. Furthermore growing corporate networks tend to prefer open source systems due to the cost benefit of not spending a significant amount in licensing [1]. Very limited research has been carried out on newer OSS systems for evaluating IPv6 on Gigabit Ethernet test-beds. This thereby motivates us to contribute in this area and to formulate this paper.

In this paper, we quantify and analyze IPv6 performance by measuring TCP throughput, RTT, jitter and CPU usage for various packet lengths. These parameters are in-sync with industry standards. For instance, packet lengths ranging from 128 to 1408 bytes are considered because standard Ethernet packet fragmentation occurs at around 1500 bytes as per RFC 1191. The effect of increasing packet length on system performance is thus also investigated. Furthermore, packet delay and packet jitter are also measured using TCP timestamp options carried in TCP headers. This helps determine overall network performance for TCP traffic over the two OSS systems. In-order to help optimize bandwidth usage on the TCP stack, these measurements are compared over IPv6 and IPv4. Because of the nature of OSS systems where the source code is open for development to anyone, this paper evaluates the TCP performance of IPv6 and IPv4 only on the Linux kernel by evaluating two Linux server operating systems, namely, Red Hat Enterprise Server 5.5 and Ubuntu Server 10.04 for which no work is published. 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 Linux OSS operating systems. The analysis of our study further aims to help researchers working in the field of traffic engineering as well as network engineers and network designers overcome the challenging issues pertaining to IPv6 deployment. This study does not evaluate closed source software such as Microsoft's (MS) windows suite of operating systems as those are only open for development to an internal MS team, however, such studies like [2] can be useful to measure the performance difference of IPv6 between open and closed source systems and can often lead to influencing key stages in the performance engineering phase of system deployment. Further to our findings in earlier study [2], this paper goes in-depth and evaluates additional performance metrics such as packet jitter and CPU Usage ('System CPU Time') to measure the difference in kernel performance between the two Linux OSS systems.

In the following sections, we review previous work on IPv4 and IPv6 and discuss our contribution towards research in this field. We describe the test bed and measurement procedure next where we detail the packet-generation and traffic-measuring mechanisms along with the evaluation methodology of our experiment. The results and comparative analysis are presented in the section entitled 'Experimental Results and

Performance Analysis'. Following which, we conclude the paper with a proposal for future work.

II. A Review of Literature

Due to the ubiquity of TCP/IP several similar studies have been carried out for IPv4 and IPv6, some evaluating over different operating systems [3, 4] whereas others over cabling systems [5]. In 2009, IPv4 and IPv6 were evaluated over the then-draft wireless standard of 802.11n [6]. Throughput and packet delay were measured over Windows operating systems such as Windows XP and Windows Vista that were common at the time. An earlier work prior to that [7] evaluated IPv6 performance over an inter-domain network that implemented routing across multiple VLAN networks. In 2006, a similar study to ours evaluated IP performance over open source systems [8]. This study included OSS systems that were not particularly limited to Linux as along with Red Hat Server they also included FreeBSD and Sun Solaris (popular at the time as server operating systems). Their study measured TCP throughput and packet delay and evaluated IPv6 performance however did not provide significant analysis on the evaluated OSS systems.

Table I provides a brief overview of related work that focused on TCP performance over IPv6 using different operating systems and on similar test-beds.

TABLE I: Key Researchers and their main contributions in IPv6

Researchers	Year	Performance evaluation
B.K. Soorty et al. [4]	2012	Measured network throughput and packet delay for UDP over IPv6 on Windows and Linux client-server networks.
S Kolahi et al. [3]	2011	Measured network throughput and packet delay for TCP and UDP over IPv6 on Windows Vista.
Mohammed et al [8]	2006	Measured network throughput for TCP over IPv6 using open source systems (OSS) – FreeBSD, Sun Solaris and RHES.
Tin-Yu Wu et al. [7]	2005	Measured network throughput and packet delay for TCP and UDP over IPv6 on an inter-domain environment using Fedora Core 2.0.

All the papers reviewed in this section considered only network throughput and packet delay, with the exception of [4]. Performance metrics such as packet jitter and CPU utilization were not studied which might impact the performance of the two IP stacks according to earlier studies [3-6, 8].

Our contribution in this article is to obtain new results by investigating the network performance with respect to additional QoS metrics such as RTT and jitter and to furthermore investigate it on newer and some of the most commonly used open-source server operating systems, namely the Linux operating systems, Red Hat Enterprise Server and Ubuntu Server. Red Hat Enterprise Server more commonly known as RHES is significantly popular for commercial use on server architecture. Ubuntu Server edition being a comparatively newer server OSS reportedly also appears to have rapidly gained market popularity as a server OS due to its strong community support and a well-documented support setup from its online community. This paper analyzes the Linux IPv6 protocol stack and TCP implementations integrated into their stock mainline kernel. Furthermore unlike earlier literature [4, 6-8], our study also investigates and analyzes the behavior of the two Linux OSS systems and states why they perform the way they do and what traffic engineering techniques can be applied to improve system performance.

III. Testbed Measurement and Procedure

A. Testbed Configuration

- Topology – The network topology is a peer-to-peer Gigabit Ethernet setup consisting of server operating systems that are paired with each other, i.e. RHES with RHES and Ubuntu Server with Ubuntu Server as seen in fig. 1. This is because only those specific server-related OSS systems are evaluated and therefore it was imperative to not use any other client OS on the other end of the peer as this would instead simulate and establish a client-server connection. To avoid that, each server OS in the peer-to-peer group was setup independently with the same configuration.
- Connection – No routers, switches or hubs were used in the experimental setup so as to ensure that there was no latency over the network. Furthermore this was primarily done because the IP performance was to be measured at the network stack of the OS kernel on the server and not necessarily at the data-link layer (one that would involve use of a network switch) or at the network layer (one that would involve routing across LANs).
- Distance – Each machine was separated from the other by a distance of approximately one meter as suggested by key researchers in this field. This was also to maintain consistency with earlier research [5] and thus produce results indicative of a fair comparison with earlier systems. The client and server machines were connected using a Category 6 Crossover UTP (Unshielded Twisted Pair) cable maintaining EIA/TIA 568-B wiring configuration (Fig. 1).
- Software – All services (running on default on RHES and Ubuntu Server) consuming network bandwidth and/or CPU resources were disabled to get unbiased and more accurate results. No third-party applications were used to optimize or influence network performance in any way.
- Hardware – The hardware benchmark consisted of four workstations, all of which surpassed the minimum and recommended settings for the applicable server operating systems tested on them. Machines one and two had identical hardware specifications: 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 used the same setup for all experiments conducted. Several repeated tests revealed that the native hardware configuration met the recommended OS settings.

B. Measurement Tools and Metrics

The data-generating tool used to craft and send TCP packets across the network test-bed was a modified version of Iperf [9]. Iperf is an open source packet-generating tool written in C++ that allows users to measure network performance over several platforms including Linux. Although for the purposes of this experiment, a minor part of the code had to be appended using C++, in-order to enable the set parameters on RHES. Iperf was used to measure TCP throughput, RTT (delay) and jitter. A configuration of '1 run' was set to one million packets from the source to destination nodes. Ten such runs were carried out for each observation and a standard deviation of less than 10% was maintained to record accurate results.

D-ITG was used to measure CPU utilization on the two OSS systems.

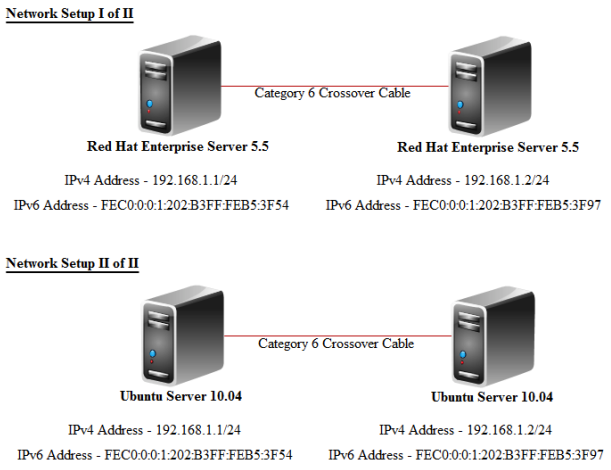

Fig. 1. Network Testbed

IV. EXPERIMENTAL RESULTS AND PERFORMANCE ANALYSIS

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. Fig. 2 displays and compares TCP throughput (in Mbps) for IPv6 and IPv4 on RHES and Ubuntu Server OSS for packet lengths of 128, 384, 640, 896, 1152, and 1408 bytes. We observe an increase in throughput with the increase in packet length. This is because larger packets can carry more payloads and require less number of transfers to move the data from the source to the destination. Thus a higher throughput is achieved through means of lower packet fragmentation by effectively increasing the MTU (Maximum Transmission Unit) and customizing the kernel to force fragmentation to occur at higher-packet sizes. Such form of packet crafting would be efficient for services and applications involving data transfer.

We also observe that for IPv4 Red Hat Enterprise Server handles larger packets more efficiently than the IPv4 packet handler in Ubuntu Server. For example, Red Hat server achieves a higher throughput (812 Mbps) compared to Ubuntu Server (750 Mbps) at the packet length of 1408 bytes. The throughput difference being approximately 7.6% better using Red Hat server.

For IPv6, the reversal holds true of what was observed for IPv4. Here, we observe Ubuntu Server processing packets with a higher payload more efficiently. For example, Ubuntu server achieves a higher throughput (799 Mbps) than Red Hat Server (698 Mbps) at a packet length of 1408 bytes. The throughput difference being significantly 12.6% better using Ubuntu server.

The higher throughput on IPv4 over Red Hat server and the higher throughput on IPv6 over Ubuntu server may be a result of the high TCP send/receive buffer in the kernel respective to each OS. 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.

By looking at Fig. 2, one can observe that TCP link throughput is overall slightly higher for IPv4 than IPv6 for both OSS systems at packet length smaller than 896 bytes. Perhaps IPv6 deteriorates throughput as a result of its high transmission overhead (i.e. larger header), however the IPv6 header though larger is much simpler compared to the IPv4 header which explains its higher throughput on larger packets (Higher payload transfer per packet) wherein the difference in TCP throughput is fairly low between the two IP versions. Previous studies have shown that IPv4 performs significantly better than IPv6 on an older version of client/server Windows and Linux networks [8]. This trend seems to be continuing with the release of newer operating systems such as Windows 7 and Ubuntu 10.04 [4]. Now we observe a similar trend here with Ubuntu Server and Red Hat server. The main conclusion is that IPv6's TCP throughput is lower than IPv4 for both OSS systems and for most packet lengths however this degradation is insignificant with smaller packets but significant on larger packets. This observation is critical in-terms of packet crafting OSS applications that are bandwidth intensive. When the two OSS systems are compared Ubuntu server performs significantly better overall (i.e. higher throughput) to Red Hat server. Interestingly though, we observe Red Hat server to achieve a similar throughput to that of Ubuntu for packets smaller than 640 bytes and Ubuntu Server achieving a higher throughput for packets larger than 640 bytes. This is largely due to the kernel implementations unique to the respective OS.

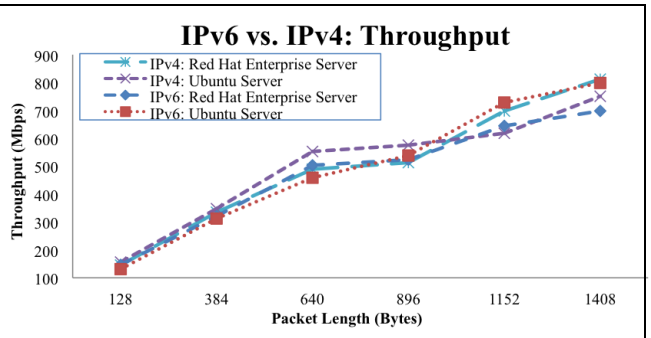

Fig. 2. TCP throughput comparison for IPv6 and IPv4. We measured throughput for both protocols comparing Red Hat Enterprise Server with Ubuntu Server OSS systems.

We now quantify the throughput of IPv6 on the two OSS systems (see Table II). Table II compares TCP throughput for the highest packet length of 1408 bytes for IPv6 and IPv4 over the two OSS systems. We found that IPv6 achieves a 15% lesser throughput than IPv4 using RHES. In comparison, IPv6 achieved a 6.17% higher throughput than IPv4 using Ubuntu Server. By comparing both OSS systems, one can observe that Ubuntu Server achieved approximately 13.41% higher throughput than Red Hat Server for IPv6 albeit Red Hat server achieved a 7.85% higher throughput gain over Ubuntu server for IPv4. Based on these empirical findings, our readings find the Ubuntu server kernel more efficient in-terms of overall IPv6 throughput. Potential to increase the gain in throughput on Red Hat server does exist via means of customizing the IP handler to define and accommodate larger packets for services pertaining to TCP segments.

TABLE II: TCP THROUGHPUT COMPARISON OF IPV6 AND IPV4 FOR RED HAT ENTERPRISE SERVER AND UBUNTU SERVER.

Open Source Software (OSS)	**Throughput (Mbps)**			
	IPv4	***IPv6***		
Red Hat Enterprise Server	812.23	698.26	**15.09%**	**IPv4 is better**
Ubuntu Server	750.85	798.67	**6.17%**	**IPv6 is better**
	7.85%	**13.41%**		
	RHES better	**Ubuntu is better**		

Recall that 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 Ubuntu and Red Hat servers. For IPv4, the lowest RTT (1.65 ms) was recorded for Red Hat Server at packet length of 128 bytes. In contrast, Ubuntu Server obtained RTT of 1.73 ms at packet length of 128 bytes. The RTT difference is about 4.7% (Red Hat Server is better in achieving lower RTT). The highest RTT for Ubuntu and Red Hat Servers at packet length of 1408 bytes are 5.56 ms and 2.07 ms, respectively. Again Red Hat Server performs better in achieving about 91.5% lower RTT than Ubuntu Server at packet length of 1408 bytes. Based on these readings, our findings show that the kernel for Red Hat server is far significantly optimized for TCP throughput and RTT on IPv4 compared to Ubuntu server.

For IPv6, the lowest RTT (1.38 ms) was recorded for Ubuntu Server at packet length of 128 bytes. In contrast, Red Hat Server had RTT of 1.58 ms at packet length of 128 bytes. The difference in RTT between the two server OSS systems is about 13.5% (Ubuntu Server is better). The highest RTT for Ubuntu Server and Red Hat Server at packet length of 1408 bytes are 7.53 ms and 3.11 ms, respectively. One can observe that Ubuntu Server offers about 83% lower RTT than Red Hat Server at packet length of 1408 bytes.

Now let us discuss how IPv6 RTT reacts with packet lengths. As shown in Fig. 3, the RTT increases with packet length for IPv6 and it becomes more significant at packet length greater than 640 bytes. For example, IPv6's RTT difference for the lowest (128 bytes) and the highest (1408 bytes) packet lengths is about 138% for Ubuntu Server. In contrast, the IPv6's RTT difference between the lowest and the highest packet lengths is 65% for Red Hat Enterprise Server.

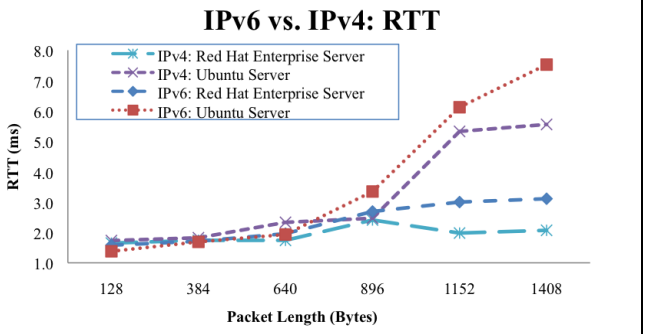

Fig. 3. TCP Round-Trip Time (RTT) for IPv6 and IPv4. We measured RTT for both protocols comparing Red Hat Enterprise Server with Ubuntu Server OSS.

Table III compares TCP RTT for the highest packet length of 1408 bytes for IPv6 and IPv4 over the two OSS systems. We found that IPv4 achieved about 40.15% and 30.09% lower RTT than IPv6 for RHES and Ubuntu server, respectively. By comparing RTT for both OSS systems, one can observe that Red Hat Enterprise Server achieved about 83% and 91.5% lower RTT than Ubuntu Server for IPv6 and IPv4, respectively. Based on these findings, our results conclude Red Hat Enterprise Server effectively maintains a lower RTT and therefore less packet delay to Ubuntu Server. It can be hypothesized that this significant difference in packet delay on RHES may be attributed to the fact that Ubuntu Server produces a significantly higher throughput (13.41% higher throughput as shown in table III) compared to the throughput achieved on RHES. The RHES kernel would be therefore more geared towards network applications and web services involving use of secure delay-sensitive data.

TABLE III: TCP RTT COMPARISON FOR IPV6 AND IPV4 USING RED HAT ENTERPRISE SERVER AND UBUNTU SERVER.

Open Source Software (OSS)	TCP RTT (ms)			
	IPv4	*IPv6*		
Red Hat Enterprise Server	2.07	3.11	**40.15%**	**IPv4 is better**
Ubuntu Server	5.56	7.53	**30.09%**	**IPv4 is better**
IPv4 and IPv6 – RHES is better	**91.5%**	**83.0%**		

TABLE IV: MEAN TCP JITTER COMPARISON FOR IPV6 AND IPV4 USING RED HAT ENTERPRISE SERVER AND UBUNTU SERVER.

Open Source Software (OSS)	Mean TCP Jitter (ms)			
	IPv4	*IPv6*		
Red Hat Enterprise Server	0.16	0.16	**0%**	**No Difference**
Ubuntu Server	0.21	0.22	**0.01%**	**IPv4 is better**
IPv4 and IPv6 – RHES is better	**27%**	**31.5%**		

Both Jitter and CPU utilization are important measures of network performance. Jitter is an important performance metric for delay-sensitive traffic such as VoIP. It is a measure of packet delay variance. Figure 4 compares TCP jitter of IPv6 and IPv4 for Red Hat Enterprise Server and Ubuntu Server. Recall that TCP jitter was measured (in millisecond) at the receiving node for both OSS systems. We observe that Red Hat Enterprise Server achieves about 30% lower jitter on the average (across all the packet lengths considered) than Ubuntu Server for both IPv6 and IPv4. This may explain why the overall packet delay was lower on RHES than it was on Ubuntu Server. Another observation is that TCP jitter of IPv4 is slightly better (e.g. lower jitter) than IPv6, especially for packet length greater than 640 bytes for both server operating systems. Furthermore our results indicate an increase in TCP jitter with the increase in packet length. As observed with both OSS systems, TCP jitter increases with packet length and becomes saturated at packet length of 1408 bytes for IPv6. The difference in TCP jitter for IPv6 between the lowest (128 bytes) and the highest (1408 bytes) packet length is about 53% on RHES and Ubuntu Server.

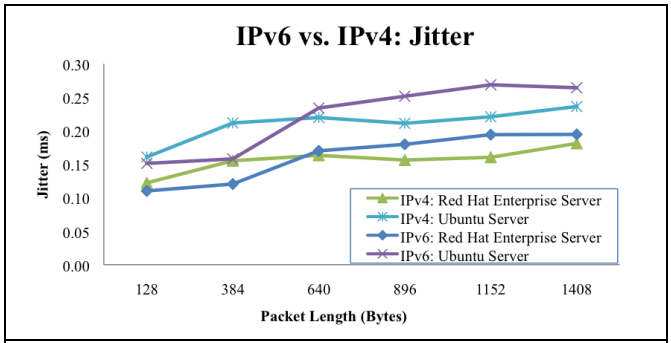

Fig. 4. TCP jitter comparison for IPv6 and IPv4 using Red Hat Enterprise Server and Ubuntu Server.

Table IV compares the mean TCP packet jitter for IPv6 and IPv4 over the two OSS systems. As observed there is no difference in packet jitter for IPv4 and IPv6 using RHES and a very insignificant difference that is low enough to be discounted using Ubuntu Server. It is however noteworthy to observe that RHES has a lower packet-drop by 27% for IPv4 and by 31.5% for IPv6 when compared with Ubuntu Server. Thus in conclusion, the RHES kernel handler is more efficient in-terms of packet jitter similar to how it is in packet delay with comparison to Ubuntu Server.

CPU utilization is an important resource that should be managed to run OSs efficiently. Fig. 5 compares the CPU processing resources consumed by sending node to transfer TCP segments over IPv6 and IPv4 networks. We observe that CPU utilization is higher on the smaller packets than larger packets. This is due to the smaller packets carrying a smaller payload of data and therefore requiring more transfers per TCP session. Comparatively larger packets have a higher payload thereby requiring fewer transfers per TCP session and a smaller TCP window. Packet processing is therefore much higher with smaller packets thereby in-turn resulting in higher CPU utilization on smaller packets as observed in Fig. 5.

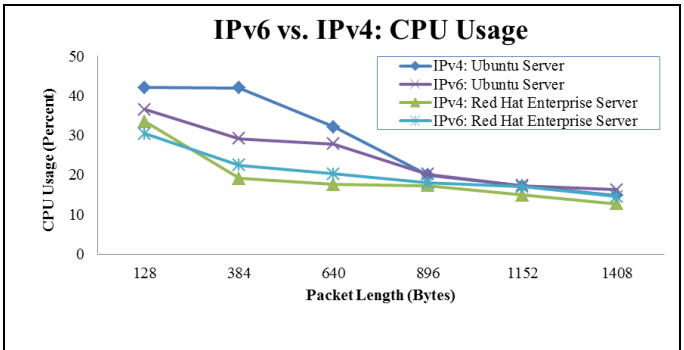

Fig. 5. CPU Usage comparison for IPv6 and IPv4 using Red Hat Enterprise Server and Ubuntu Server.

Figure 5 furthermore provides an indicative overview of CPU usage to Linux networking. As observed, when the two OSS systems are compared the kernel for the Ubuntu server utilizes higher processing in handling the TCP/IP stack compared to the RHES kernel. We also observe the CPU utilization to be lower with IPv6 on smaller packets however it is found to be slightly higher to IPv4 with a larger payload and higher packet-length.

'System Time' can be defined as the percentage of time the CPU spends executing kernel threads and interrupts. Table V compares TCP CPU usage for IPv6 and IPv4 using the two OSS systems. We observe that Red Hat Enterprise Server consumes about 4.5% less system time than Ubuntu server for IPv6 and 8% less system time for IPv4. Another observation is that the CPU usage on IPv6 is about 2.5% lower than IPv4 for Ubuntu Server and an insignificant 0.5% lower for IPv4. Overall, IPv6 handles CPU usage more efficiently on Ubuntu

Server, as the source code for its socket creation time appears to record a higher number of TCP connections in play at the time of measurement.

TABLE V: MEAN TCP CPU USAGE: SYSTEM TIME FOR IPV6 AND IPV4 USING RED HAT ENTERPRISE SERVER AND UBUNTU SERVER.

Open Source Software (OSS)	**Mean TCP Jitter (ms)**			
	IPv4	***IPv6***		
Red Hat Enterprise Server	20	20	**0%**	**No Difference**
Ubuntu Server	28%	24.5%	**3.5%**	**IPv6 uses less CPU power**
IPv4 and IPv6 – RHES is better	**8%**	**4.5%**		

V. COMPARATIVE ANALYSIS

In the earlier section, our empirical results provided an overview of IPv6 performance at the network stack of the Linux kernel. Based on these measurements our paper aims to shed some insight into IPv6 performance on two of the evaluated OSS systems, namely Red Hat Enterprise Server (kernel 2.6.18) and Ubuntu Server (kernel 2.6.32) and the choice of server OS best configured for IPv6. Furthermore the earlier section also delved into deeper analysis of where, why and how the system performance could be tuned and improved for IPv6. Table VI summarizes the performance of IPv6 with IPv4 for TCP over the two OSS systems.

TABLE VI: SUMMARY OF IPV6 EVALUATION FOR TCP - OVERALL PERFORMANCE BETWEEN OOS SYSTEMS.

Open Source Software (OSS)	**Throughput**		**RTT**		**Jitter**		**CPU Usage**	
	IPv6	***IPv4***	***IPv6***	***IPv4***	***IPv6***	***IPv4***	***IPv6***	***IPv4***
Red Hat Enterprise Server		Better		Better	Better		Better	
Ubuntu Server								

We observe the following system performance characteristics:

1. **Throughput (IPv6 vs. IPv4):** For RHES, IPv6 achieved 15% lower throughput than IPv4 (IPv4 is better). For Ubuntu Server, IPv6 achieved about 6% higher throughput than IPv4 (IPv6 is better). When the performance of the two OSS systems was compared, TCP throughput was higher for IPv4 on RHES (by 8%) and higher for IPv6 on Ubuntu Server (by 13%).
2. **RTT (IPv6 vs. IPv4):** For RHES, IPv6 achieved about 40% higher TCP delay than IPv4 (IPv4 is better). For Ubuntu Server, IPv6 achieved about 30% higher delay than IPv4 (IPv4 is better). When the performance of the two OSS systems was compared, RTT for TCP was lower for IPv4 on RHES (by 91.5%) and lower for IPv6 also on RHES (by 83%).
3. **Jitter (IPv6 vs. IPv4):** For RHES, there was no difference observed in overall TCP packet jitter between IPv4 and IPv6, the same could be stated for Ubuntu server where IPv4 produced a slightly lower drop by 0.01%. When the performance of the two OSS systems was compared, TCP Jitter was lower for IPv4 on RHES (by 27%) and lower for IPv6 also on RHES (by 31.5%).
4. **CPU Usage (IPv6 vs. IPv4):** For RHES, there was no difference observed in TCP system time between IPv6 and IPv4. For Ubuntu Server, CPU usage on IPv6 was 2.5% lower than IPv4 (IPv6 is better). When the performance of the two OSS systems was compared, CPU usage was lower for IPv4 on RHES (by 8%) and lower for IPv6 also on RHES (by 4.5%).

As shown in table VI, implementing IPv6 on Gigabit Ethernet networks will have drawbacks, such as lower bandwidth (throughput) and higher latency (RTT). IPv6 obtained about 15% lower throughput than IPv4 on Red Hat Enterprise Server, however IPv6 obtained a 6.17% higher throughput compared to IPv4 on Ubuntu Server. This thereby validates our research question 1 that not every OSS system is optimally configured for IPv6 at the kernel level of the network stack. This leads us to answer the research question 2 posed earlier as to which modern, commonly used, open-source, server operating system is geared for high IPv6 performance. Based on empirical results carried out through this experimental research we can now state that the Linux based Ubuntu Server is optimized for IPv6 performance as it achieves a higher TCP throughput over IPv6 (see Table II) however this does include a citation and one stating that it is so for the highest packet-length of 1408 bytes. Red Hat Enterprise Server achieved excellent system performance over IPv4, but not necessarily for IPv6 networks. Our third research question poses the analytical

question as to why the OSS systems perform the way they do and what modifications can be applied to the OS kernel to help improve TCP performance over IPv6. The answer although covered in earlier sections can be justified and summarized as follows. On both OSS systems mean TCP throughput was lower on IPv6 than it was on IPv4. This observation was noted in earlier studies evaluating IPv6 performance over Windows systems. This trend continues with newer Linux OSS systems as evident from empirical results gained through our experiment. This degradation over IPv6 was proposed in earlier studies to be attributed due to the larger overhead of IPv6. Our study can now confirm this as we observe IPv6 to have a significantly lower throughput to IPv4 on smaller packets however, this difference decreases with the increase in packet-length. This observation can be noted on both the Linux operating systems. One means of increasing IPv6 throughput is by reducing packet overhead. This enables the kernel to craft larger packets enabling them to carry a higher payload over each session. This change can be significant to bandwidth intensive applications that operate over an IPv6 network. In-terms of packet delay and jitter, our results show an average of 30% lower jitter on RHES compared to Ubuntu (across all packet lengths). This is likely the reason why RHES also achieves a significantly lower RTT than on Ubuntu. The low RTT on IPv4 compared to IPv6 is a direct co-relation to the higher throughput achieved over IPv4. In terms of packet jitter, there was no notable difference between IPv4 and IPv6. CPU usage i.e. the 'system time' for processing packets was also lower over IPv6 when compared to IPv4 on RHES and similar on both over Ubuntu. Furthermore it can also be noted that with both operating systems, the CPU utilization was higher on smaller packets than it was on the larger packets. This is due to the smaller packets having a smaller payload and therefore requiring more packet generation and more transfers per TCP session. There can be an increased efficiency in packet processing when packets are crafted to carry a higher payload. This results in lower CPU utilization and higher throughput. When we compare the two OSS systems, we find IPv6 handles CPU usage more efficiently on Ubuntu server compared to RHES. This, as justified earlier is because the source code for its socket creation time appears to record a higher number of TCP connections in play compared to RHES.

VI. Practical Implications

In this paper, we investigated the data performance on TCP for IPv6 over two Linux based open-source server operating systems. The aim of our research was to not only evaluate and investigate the performance of those systems but also to analyze and report what changes could be made to help improve system performance. Our research evaluated the performance in-order to help network engineers, network architects and network administrators deploy the right server OS. 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 by implementing the above mentioned kernel level modifications. For instance, to gain an increase in IPv6 throughput, system engineers could craft packets by increasing payload data and setting the packet-length to the size of 1408 bytes. This would result in a slight increase in packet delay, however, also result in a significant gain in throughput. Since most network applications relate to delay insensitive data, unless the network appliance used requires a form of secure authentication, most system engineers would be benefitted to make this change and experience a gain in overall IPv6 performance. This study can also enable developers working on open source projects for the Linux kernel implement suggested changes to further improve IPv6 efficiency. For instance, by increasing buffer-size in the socket and thereby accommodating more packets the performance of IPv6 on RHES can be increased similar to that noticed on Ubuntu Server.

VII. Conclusion

The findings based on our empirical study revealed that IPv6 throughput was 13% higher on the Ubuntu Server architecture compared to RHES. Comparing the kernel's TCP/IP stack on each OS, we found that the Ubuntu kernel processed IPv6 packets more efficiently compared to RHES, which maintained higher throughput and packet delays over IPv4. With respect to packet delays, jitter and CPU utilization, the kernel structure in RHES is better handled for performance. This is no surprise as the RHES and Fedora Core kernel comparatively have been longer under development and use as a server OS. Ubuntu Server OS being relatively new still managed to gain higher throughput on IPv6 and had lower system time in processing IPv6 packets. We further quantified the degradation on performance for both OSS systems by each metric that was evaluated and produced our analysis on the bottlenecks and changes that could be implemented in the kernel stack such as packet handling writes on the socket based on the type of traffic desired on a network, and modifying packet reception by increasing the queue buffer in RHES.

A lower packet delay on Ubuntu Server could similarly be achieved by through means of lower packet fragmentation by decreasing the MTU (Maximum Transmission Unit) limit and customizing the kernel to force fragmentation to occur at lower-packet sizes. 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] StatOwl. (2010). *Operating System Version: Usage*. Available: http://statowl.com/operating_system_market_share_by_os_version.php?1=1&timeframe=last_6&interval=month&chart_id=4&fltr_br=&fltr_os=&fltr_se

[2] N. Sarkar and B. Soorty, "Evaluating IPv6 in Peer-to-Peer Gigabit Ethernet for TCP using Modern Windows and Linux Systems," *International Journal of Business Data Communications and Networking*, vol. 9, pp. 50-63, 2013.

[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. 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.

[5] 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.

[6] 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.

[7] T.-Y. Wu, H.-C. Chao, T.-G. Tsuei, and Y.-F. Li, "A measurement study of network efficiency for TWAREN IPv6 backbone," *International Journal of Network Management*, vol. 15, pp. 411-419, 2005.

[8] 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.

[9] NLANR/DAST. (2012). *Iperf*. Available: http://iperf.sourceforge.net/