Performance Monitoring of VoIP with Multiple Codecs Using IPv4 and IPv6to4 Tunnelling Mechanism on Windows and Linux

Hira Sathu and Mohib A. Shah

***Abstract*—In this paper, the performance of Voice over Internet Protocol (VoIP) for five different codecs, using IPv4 without tunnelling and IPv6 with IPv6to4 tunnelling mechanism is established. The codecs tested were G.711.1, G.711.2, G.723.1, G.729.2 and the G.729.3 codec. The experiment conducted covered the two different configuration set up of networks using a fully IPv4 infrastructure and the other with IPv6to4 tunnelling mechanism (T.M). The operating systems used were Windows 7 and the Linux Ubuntu 9. The parameters covered using the above test-beds were delay, jitter and throughput. The results indicate that Linux Ubuntu 9 provides lesser delay for IPv4 compared to Windows 7 on the G.711.1, G.711.2 and G.723.1 codecs, while Windows 7 provides lesser delay on the G.729.2 and G.729.3 codecs. For the second network configuration using IPv6to4 T.M, Linux provides lesser delay across all 5 codec's than the Windows 7. The results for the throughput and jitter are also reported. Keeping in view the performance recommendations for low bandwidth networks using VoIP, the codecs like G.723.1 and G.729.2 with smaller packet size and coding speeds resulted in lower jitter and RTT delay. However, throughput results indicate G.711.1 as the preferred choice for both windows and Linux OSs with Linux having only a marginal edge over Windows for tunnelling environment.**

***Index Terms*—VoIP, performance analysis, Codec, IPv4, IPv6to4 tunnelling mechanism, Windows 7, and Linux Ubuntu.**

I. Introduction

The issue of larger number of desktop, laptop and other computing machines requiring IP addresses for access to internet and networking has been well established. The transition to IPv6 was designed by the Internet Engineering Task Force (IETF) to be the successor of IPv4. The main advantages of IPv6 is its ability to support large numbers of addresses, (2^{128}-bit address space).The delays caused due to Network Address Translation(NAT) no longer factor in as performance bottlenecks. Use of IP based networks to carry voice has gained prominence on account of economies with near circuit switched quality voice over data circuits. The main reason VoIP (Voice over Internet Protocol) has become so popular over the last few years is because of the reduced cost associated with using VoIP compared to the PSTN (Public Switched Telephone Network). VoIP prominence has necessitated evaluation of different protocols as well as confronted many companies with the dilemma of choosing either to continue using the IP4 protocol or switching over to use the new IPv6 protocol stack. The final choice would depend upon which operating system and which codecs they are using over the two communication protocol mechanisms. An earlier paper "Performance Comparison of VoIP Codecs on Multiple Operating Systems using IPv4 and IPv6" by the authors being presented at the IC4E 2011 conference discusses the performance of a range of VoIP codecs (encode speech to enable transport over internet) in purely IPv4 and IPv6 environments in a simulated network environment. This paper goes to further consider a more realistic environment. This environment includes the performance of using a range of Codecs (G.711.1, G.711.2, G.723.1, G.729.2 and G.729.3) over Tunnelling Mechanism 6to4. The end users herein used IPv6 and the 6to4 Tunnelling Mechanism, for reasons covered later across the IPv4 based Internet. VoIP communications were also tested using IPv4 addresses to provide comparative results over a similar network set up. This study also covers newer versions of operating system known as Microsoft Windows 7 and Linux (Ubuntu 9) to identify the performance of VoIP Codecs using IPv4 and IPv6to4 networks.

IPv6to4 is one of a tunnelling mechanism which is used to send IPv6 packets via IPv4 network to other IPv6 networks. This mechanism was designed to bridge between IPv6 networks through an IPv4 network. IPv6to4 tunnelling mechanism carries IPv6 packets and encapsulates into IPv4 header and sends it via IPv4 network. It de-capsulates the packets at the other end and delivers to its destination.

TABLE I: Multiple VoIP Codecs and Their Frame Sizes [1] (Milliseconds)

Codec	G.711	G.723.1	G.729
Coding speed (Kbps)	64	5.3/6.3	8
Frame size (ms)	20	30	10
Processing Delay (ms)	20	30	10
Lookahead Delay (ms)	0	7.5	5
DSP MIPS	0.34	16	20
Payload (bytes)	160	20/24	20
Number of flows	7	84/71	56
Subscribed Rate packet time (ms)	20	30.2/30.5	20

In [1], the authors compare the jitter and delay for VoIP performance generated by common voice codecs both under Differentiated services with expedited forwarding and best-effort service. The codecs used in this paper are the

Manuscript received April 9, 2012; revised May 30, 2012.

H. Sathu is with Unitec Institute of Technology, Auckland, New Zealand (e-mail: hsathu@nitec.ac.nz).

M. A. Shah was with Unitec Institute of Technology, Auckland New Zealand. He is now tutoring at Unitec, New Zealand (e-mail: mshah.itexpert@gmail.com mshah@unitec.ac.nz).

ITU-T (International Telecommunications Union) standard voice codec algorithms G.711.1, G.723.1, and G.729. Each codec has its own speed, frame size, delay and payload as shown in Table I above.

The organisation of this paper is as follows: next Section 2 covers related works and contribution this paper makes, Section 3 covers the network set up for the current study, and Section 4 covers the traffic generating tool description. Section 5 outlines the results of the experiment and the last Section6 covers the discussion and conclusion followed by the references.

II. Related Work

In [2] the authors carried out an experiment using IPv4 and IPv6 networks and established a SIP (Session Initiation Protocol) based VoIP system on each network to identify the performance of SIP on IPv4 and IPv6. The results obtained indicate that delay using SIP performed better on IPv4 than IPv6. Another experiment was carried out including two different tunnelling mechanisms IPv6to4 and Teredo. However outcome clarifies that Teredo tunnelling mechanism has higher delay than IPv6to4 tunnelling mechanism tested. This was the reason leading to the choice of using 6to4 tunnelling mechanism for the current study.

In [3], the authors discussed the connection of IPv6 domains using the current IPv4 network without setting up an explicit tunnel between the two connected domains. The report explains the use of the IPv6to4 pseudo-interface, which is when the IPv6 packet is encapsulated in an IPv4 packet at one end, and is then sent over the IPv4 cloud. When the packet reaches the other end, it is then unpacked. The mechanism is intended as a start-up transition tool used during the period of co-existence of IPv4 and IPv6 .

The authors in [4] compared different aspects of VoIP between IPv4 and IPv6. The authors considered jitter, delay, packet loss and throughput on different systems using 0 – 200 Mbps traffic. The authors showed that for windows XP, the average delay between packets for IPv4 and IPv6 is almost the same, except at 100Mbps when IPv6 delay is approximately 0.002ms more than IPv4. From 0 – 50Mbps of traffic, packet loss was equal between the two IP versions, at 0 lost packets, but for 100, 150 and 200Mbps, IPv6 packet loss rose to 4, 13, and 17 packets lost respectively, whereas IPv4 rose to 0, 12 and 17 packets lost respectively. The average jitter reduced fairly consistently up to 100Mbps, but from 100Mbps to 200Mbps IPv4 showed less jitter than IPv6 by 0.05ms. Overall, IPv4 had better performance across the tests compared to IPv6. In general, their results indicate that the difference in VoIP performance for IPv6 and IPv4 is negligible. Results for the bare PC softphone confirm that reducing system and application overhead lowers delta and jitter values regardless of the IP version.

In [5] the researchers have conducted the tests to identify the performance of audio and speech compression. The investigation included GSM (Global System for Mobile Communications) full rate, G.711, G.723.1 and MPEG coders. The results identified that MPEG transcoding impair speech recognition for low bitrates and sustain the performance of speech coders like GSM and G.711.

In [6] experiment was conducted by researchers to test the performance of VoIP with IPv4 and IPv6 using IPv6to4 tunnelling and NAT (Network Address Transition) mechanism to identify the delta, jitter, packet loss, MOS (Mean Opinion Score) and throughput. Their results demonstrate that “VoIP quality due to using IPsec with IPv6, 6to4, and NAT in VPNs during the IPv4/IPv6 transition is not significantly different from using IPsec with IPv4, and that there is a minimal impact on voice quality as long as the network capacity is not exceeded” [6].

In the next paper [7] the authors have discussed about VoIP technology as a technology is fundamentally changing telephony, enabling not just cheaper calls but also richer and more flexible services.” The authors also pointed out that VoIP still has some challenges in business communication environment. The two main challenges in VoIP technologies are security and NAT (Network Address Transition); however SIP (Session Initiation Protocol) based VoIP network has improved many of the challenges and it also has replaced PBX (Private Branch Exchange) network system.

In this [8] article authors have studied about various tunnelling mechanisms and designed a network to calculate the performance of the VoIP based applications on the mechanisms. The design of the network includes NAT (Network Address Transition), Teredo Tunnel and 6to4 Tunnel. They mainly focused on the impact of these tunnels and translation mechanisms on the SIP network.

As of mid-2010, very little is known about the comparative performance of 5 different VoIP codecs on IPv6to4 tunnelling mechanism using Windows 7 and Linux Ubuntu 9. The contribution and motivation of this paper is to compare the performance of above codecs with IPv4 and IPv6to4 tunnelled networks using Windows 7 and Linux Ubuntu 9 operating systems.

III. Network Setup

The proposed network test-bed was setup based on two different configurations with IPv4 and IPv6to4 tunnelling mechanism. The first setup was based on the IPv4 configuration, where all the nodes on the networks had IPv4 addresses and connected via our campus network (Fig. 1 below). Second setup configuration was based on IPv6to4 tunnelling mechanism network where two nodes with IPv6 were connected through the campus’s IPv4 Network, using standard Category 5e cables (Fig. 2 below).

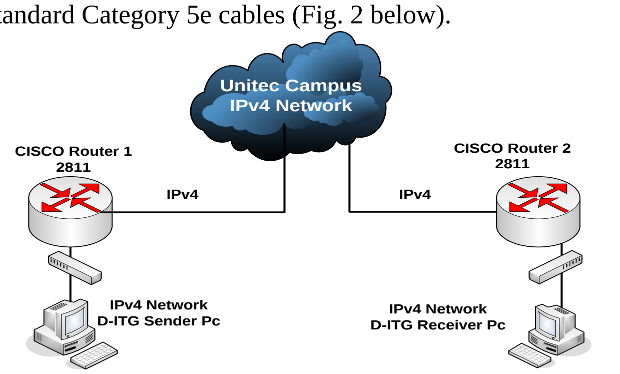

Fig. 1. Network test bed based on IPv4 Unitec’s Campus IPv4 Network

In Fig. 1, the network included two workstations using one of the operating system (Linux or Windows 7) and was configured with IPv4 addresses. The workstations then wired to a Cisco 2811 router using IPv4 address and RIP 2 (Routing Information Protocol 2). The other side of router was wired to the Campus IPv4 network as illustrated in Fig.1 above.

In the second test bed (Fig. 2), IPv6 addresses were specified to two identical workstations with Linux or Windows 7 operating system. The computers were then connected to the Cisco 2811 routers. The router was configured to act as IP6to4 tunnelling and was connected to Campus IPv4 network as before.

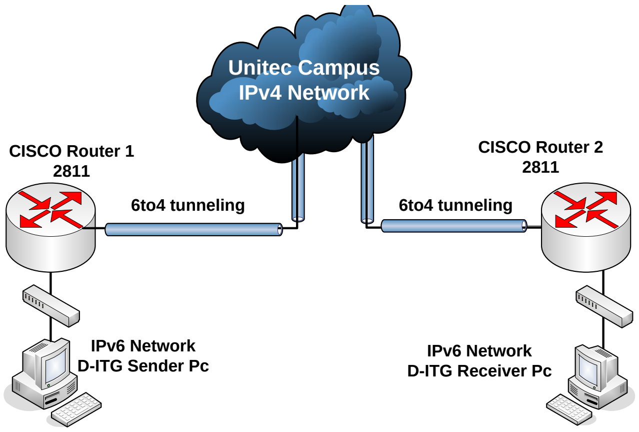

Fig. 2. Network test bed based on IPv6to4 Tunnelling Mechanism via Unitec's Campus IPv4 Network

The above two test beds were done in order to evaluate the performance of a pure IPv4 network and network with IPv6to4 tunnelling using Linux and Windows 7 operating systems. Parameters calculated were RTT, jitter and throughput. All tests were conducted under same circumstances (same low Campus traffic as tests were performed after business hours.)

The hardware benchmark comprised of an Intel® Core™ 2 Duo 6300 1.87 GHz processor with 2.00 GB RAM for the efficient operation of Windows 7 and Linux Ubuntu 9, an Intel Pro/100 S Desktop Adapter NIC and a Western Digital Caviar SE 160 GB hard-drive on the two workstations. In order to make comparisons, we used identical hardware for all our tests. A benchmarking tool known as CPU-Z was used to determine if all computers were identical. Two routers, two Switches and cat5e fast Ethernet cables were also used for creating the test-bed.

IV. Data Generation and Traffic Measurement Tool

TABLE II: D-ITG Codecs for VoIP packet generator [8]

Codecs	Samples	Framesize	Packets (per sec)
G.711.1	1	80	100
G.711.2	2	80	50
G.729.2	2	10	50
G.729.3	3	10	33
G.723.1	1	30	26

D-ITG (Distributed Internet Traffic Generator) [9] was the tool that was selected to generate and measure the traffic. This tool was the one selected as it could support both IPv4 and IPv6 traffic, and worked across a range of operating systems including Linux Ubuntu and Windows 7. D-ITG tool was designed with fixed frame size and packets per second for each VoIP codec as in Table II above.

D-ITG command mode version was installed on both networks to send and receive VoIP traffic. D-ITG sender was installed on a workstation and D-ITG receiver was installed on another workstation. The experiments comprised of performing 10 flows with number of runs for every codec type, on every operating system, for IPv4 and IPv6. A flow contains 1000 packets of a codec and (is equivalent to a VoIP call) sending from one workstation to another. A script was used to send 10 flows at the same time and average results were obtained. The number of runs is continued until 95% confidence interval in results is achieved. Each codec has its own standard packet size, which effects the results obtained (Table II).

The jitter, RTT (Round Trip Time) and throughput were calculated for IPv4 and IPv6to4 tunnelling mechanism on Windows 7 and Linux (Ubuntu), for the G.711.1, G.711.2, G.723.1, G.729.2, and G.729.3 codecs over a fast Ethernet VoIP network as shown in the network test bed diagram (Fig. 1 & 2) above.

V. Results

As may be noted from Figure 3 below, G.711.1 codec using Windows with IPv6to4 tunnelling had the highest delay out of all the tests, at approximately 0.78 milliseconds, and was closely followed by the G.711.2 codec using Windows with IPv6to4 tunnelling as well, at approximately 0.77 milliseconds. The lowest delay was calculated by the G.723.1 codec using IPv4 on Linux, at approximately 0.44 milliseconds, and was closely followed by the same codec using IPv4 on Windows, at approximately 0.45 milliseconds. Overall, Windows 7 using IPv6to4 had the highest amount of delay across all the different codecs, and was generally followed by Ubuntu Linux using IPv6to4, except for the G.711.1 codec, where Windows 7 using IPv4 had the second highest amount of delay.

Generally, Ubuntu Linux using IPv4 had the lowest amount of delay across the different codecs, except for the G.729.2 codec, where Windows 7 using IPv4 had the lowest amount of delay.

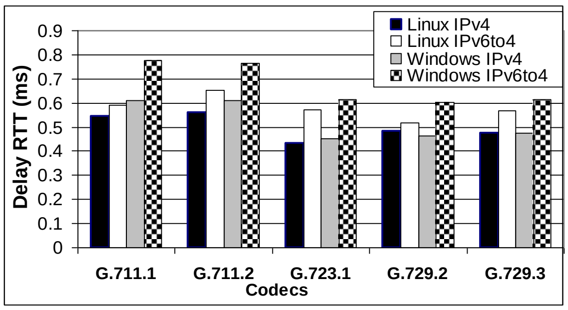

Fig. 3. RTT Comparison for IPv4 and IPv6to4 tunnelling mechanism on Windows 7 and Linux Ubuntu

Jitter was fairly different across the different codecs, with Windows performing better than Linux across all codecs except for the G.711.1. The highest jitter was seen on the G.729.3 codec using IPv4 running on Linux, which was at 0.21 milliseconds, and the next highest jitter was seen on the G.723.1 codec using IPv4 on Linux, at approximately 0.2 milliseconds. The least amount of jitter was the G.729.3

codec using IPv4 running on Windows, which had jitter of approximately 0.07 milliseconds, and was closely followed by the G.729.3 codec using IPv4 on Windows, with jitter of approximately 0.065 milliseconds.

As visible below in Fig. 4, Windows 7 using IPv4 had the lowest amount of jitter across all five of the codecs (shown as the grey bar below).

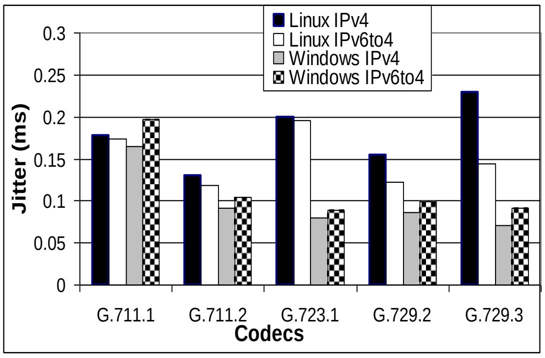

Fig. 4. Jitter Comparison for IPv4 and IPv6to4 tunnelling mechanism on Windows 7 and Linux Ubuntu

As visible in Table III below, the throughput for the G.711.1 codec and the G.711.2 codec is much higher than the other three codecs, averaging in the mid to high 600kbps, while the other codecs average in the high 90kbps and lower 100kbps. From the table below, it is visible that the highest throughput was seen on the G.711.1 codec, using Ubuntu Linux with the IPv6to4 tunneling mechanism, at approximately 692kbps throughput, while the lowest throughput is seen on the G.723.1 codec using Linux with IPv4, at approximately 77kbps.

TABLE III: THROUGHPUT COMPARISON FOR IPV4 AND IPV6TO4 TUNNELLING MECHANISM ON WINDOWS 7 AND LINUX UBUNTU (KBPS)

	Throughput IPv4		**Throughput IPv6to4**	
Codec Type	Linux	Windows	Linux	Windows
G.711.1	681.64	687.59	692.99	687.45
G.711.2	651.48	656.40	662.11	657.97
G.723.1	76.99	77.58	78.05	77.71
G.729.2	108.85	109.31	110.27	109.53
G.729.3	97.41	98.25	98.90	98.42

VI. DISCUSSION AND CONCLUSION

The results obtained in the above section indicate that Linux OS fared better than Microsoft so far a RTT delay parameter, across the complete range of codecs tested for both IPv4 pure network as well as for the tunnelling test. However, whenever tunnelling was introduced additional delay resulted as would be expected. As far as Jitter parameter was concerned, results varied, Windows OS performed better than Linux across most codecs except in the case of Codec G.711.1 for the 6to4 tunnelling test.

The best throughput of the codecs tested across the two OSs was for G.711.1 under Linux OS using 6to4 Tunnelling. The performance of this codec under windows 7 was just lower than 692.99 Kbps but comparable. The worst performance was that of G.723.1 codec across the complete range of tests, with the lowest throughput at 76.99 Kbps for Linux OS while using IPv4. Considering that this study is primarily for VoIP performance, the delay and jitter are the more significant parameters. The choice of codecs would therefore vary for different situations. For low speed networks or congested networks VoIP packets (being smaller) lower rated codecs perform better. For Integrated services (data, voice and video with heavy traffic) higher throughput requirement would dictate a different choice of OS and codec. It may be reasonable to assume that the performance of Linux Ubuntu 9 being better than Windows 7 could also be attributed to greater compatibility of the Linux OS system with the Cisco Router 2811 that was used for establishing the tunnel for the tests.

Based on the results it may be concluded that use of the IPv6to4 tunnelling mechanism, increased the RTT delay as compared to its IPv4 counterpart. This delay being mainly due to the need to encapsulate at the sending side and de-capsulate at the receiving side. However, the delay is within reasonable or tolerable limits if the correct codec and OS combination is chosen. Companies who move to IPv6, and are communicating across the Internet using VoIP are recommended to consider use of either the G.723.1 or G.729.2 codec over Linux and Windows. It may be noted that where users continue using IPv4 protocol there would be no tunnelling, hence a choice of Windows 7 OS is preferable for all codecs other than G.711.1.

Future work in this area should also include study and comparison of alternative methods used to put IPv6 traffic on IPv4 core network. Another area is the effect of increased traffic load on packet loss to further introduce the realistic environments of operational network performance interest

REFERENCES

[1] J. K. Muppala, T. Bancherdvanich, and A. Tyagi. “VoIP Performance on Differentiated Services Enabled Network,” (ICON 2000). In *Proceedings, IEEE International Conference on Networks*. 2000, pp. 419-423.

[2] T. Hoeher, M. Petraschek, S. Tomic, and M. Hirschbichler. “Evaluating Performance Characteristics of SIP over IPv6,” *IEEE Journal of Networks*, vol. 2 no.4, pp.10, 2007.

[3] B. Carpenter and K. Moore. "Connection of IPv6 Domains via IPv4 Clouds ," Internet proposed standard RFC 3056, 2001.

[4] R. Yasinovskyy, A. L. Wijesinha, R. K. Karne, and G. Khaksari. “A Comparison of VoIP Performance on IPv6 and IPv4 Networks,” *.IEEE/ACS International Conference on Computer Systems and Applications*, 2009, pp. 603-609.

[5] L. Besacier, C. Bergamini, D. Vaufreydaz, and E. Castelli. “The effect of speech and audio compression on speech recognition performance,” *IEEE 4th Workshop on Multimedia Signal Processing, Aug 2002*, pp. 301-306.

[6] R. Yasinovskyy, A.L. Wijesinha, and R. Karne. “Impact of IPSec and 6to4 on VoIP Quality over IPv6,” *IEEE International Conference on 10th, Telecommunications, ConTEL*, 2009, pp. 235–242.

[7] T. Zourzouvillys and E. Rescorla. “An Introduction to Standards-Based VoIP”. *IEEE Journal on Internet Computing*, vol. 14 no.2, pp. 69-73, 2010.

[8] S. Tomic, T. Hoeher, R. Menedetter, R. Maslenka, M. Banfield, and R. Lauster. “Study of SIP-based VoIP Application Interworking with IPv4-IPv6 Transitioning Mechanisms”. *IEEE Sarnoff Symposium*, 2006, pp. 1-4.

[9] B. Alessio, D. Alberto, and P. Antonio. (2009). Multi-Protocol and Multi-Platform Traffic Generation and Measurement. [Online]. Available: http://www.grid.unina.it/software/ITG/, 2009.

H. Sathu is a Principal Academic staff member at Unitec Institute of Technology, Auckland, New Zealand for over 17 years. Prior to his academic roles at Unitec he worked for the House of Siemens,(4 years) and the Department of Defence, India (20 years). Of recent he has been an invited speaker at International Conferences, NZ Info Security Forum and helped chair sessions at various National and International conferences.

M. A. Shah recently completed his Master's degree in Computing and he is currently tutoring at Unitec, New Zealand. He is interested in NGN activities such as VoIP, VVoIP, Wireless Network and Mobile communication based on IPv6 infrastructure. He received a BCS degree and a Post Graduate Dip in Information Technology from Unitec, New Zealand.