Int. J. Communications, Network and System Sciences, **2012, 5, 298-302**
http://dx.doi.org/10.4236/ijcns.2012.55039 Published Online May 2012 (http://www.SciRP.org/journal/ijcns)

Testing and Analysis of VoIPv6 (Voice over Internet Protocol V6) Performance Using FreeBSD

Asaad A. Abusin, M. D. Jahangir Alam, Junaidi Abdullah
Faculty of Computing and Informatics (FCI), Multimedia University, Cyberjaya, Malaysia
Email: asaadabusin@yahoo.com

Received February 27, 2012; revised March 23, 2012; accepted April 5, 2012

ABSTRACT

This study focuses on testing and quality measurement and analysis of VoIPv6 performance. A client, server codes were developed using FreeBSD. This is a step before analyzing the Architectures of VoIPv6 in the current internet in order for it to cope with IPv6 traffic transmission requirements in general and specifically voice traffic, which is being attracting the efforts of research, bodes currently. These tests were conducted in the application level without looking into the network level of the network. VoIPv6 performance tests were conducted in the current tunneled and native IPv6 aiming for better end-to-end VoIPv6 performance. The results obtained in this study were shown in deferent codec's for different bit rates in Kilo bits per second, which act as an indicator for the better performance of G.711 compared with the rest of the tested codes.

Keywords: VoIPv6 (Voice over Internet Protocol V6) Performance; Voice over Internet Protocol V6 Performance Testing; Voice over Internet Protocol V6 Performance Analysis; VoIPv6 Quality Testing in the Application Level

1. Introduction

In this study the G.711 was the codec technique in the VoIPv6 software that was used in both the VoIPv6 client and server couples in both the source IP and destination IP to investigate the voice packets traffic quality.

There are two main important and related considerations in selecting Audio Codec:

The delay that a codec will be introduced.

The Digital Signal Processor (DSP) speed that is required.

The DSP is measured in millions instructions per second (MIPS). Both factors affect the QoS of voice traffic, in addition to all that is the effect of the protocol performance [1,2].

2. Literature Review

With the great development of wireless communication technology and Internet, VoIP over wireless network is widely used. Providing QoS guarantees for VoIP applications is increasingly important, especially in mobile/ wireless networks due to their limited bandwidth and mobility [3].

The main difference between our study and [3] is that this study tests the VoIP performance in Mobile/Wireless Networks where as our study target the IPv6 protocol performance in wired networks, and as a holistic approach. In addition, [3] focused on studying VoIPv6 performance over Mobile networks whereas our study focus on the VoIPv6 performance in wired links.

The latest development from Cisco Systems is redefining the way businesses communicate [4]. Traffic analysis is essential to collect the statistical information about IP/UDP/RTP VoIPv6 packets such information like the arrival and inter arrival time of the voice packets, this will lead to better understanding the network and protocol performance, so that VoIPv6 networks with better QoS can be modeled. Our study encourages or pouch towards performing traffic analysis in networking systems to study the protocol performance.

IPv6 is documented in several RFCs (or request for comments) starting from RFC 2460. The IETF also publishes Experimental, Informational and Historic RFCs, and Best Current Practices. These RFC's specifies the V0IPv6 standards as mandates governing the protocol performance.

In [5], the main difference between this study and our study is that the testing tool in this study is Asterisk which is an open source/free software, whereas in our study the testing tool is FreeBSD which involve no overhead.

In [6] we are missing a capable test framework that is part of our main source tree.

With the stable release of FreeBSD 8.0 arriving last week we finally were able to put it up on the test bench and give it a thorough look over with the Phoronix Test Suite. We compared the FreeBSD 8.0 performance between it and the earlier FreeBSD 7.2 release along with Fedora 12 and Ubuntu 9.10 on the Linux side and then the OpenSolaris 2010.02 b127 snapshot on the Sun OS side [6].

FreeBSD 8.0 introduced support for a TTY layer rewrite, network stack virtualization, improved support for the Sun ZFS file-system, the ULE kernel scheduler by default, a new USB stack, binary compatibility against Fedora 10, and improvements to its 64-bit kernel will allow a NVIDIA 64-bit FreeBSD driver by year's end, among a plethora of other changes. With today's benchmarking—compared to our initial Ubuntu 9.10 vs Free BSD 8.0 benchmarks from September—we are using the official build of FreeBSD 8.0 without any debugging options and we are also delivering a greater number of test results in this article, along with a greater number of operating systems being compared.

In recent years, Internet Protocol (IP) telephony has been a real alternative to the traditional Public Switched Telephone Networks (PSTN). IP telephony offers more flexibility in the implementation of new features and services. The Session Initiation Protocol (SIP) is becoming a popular signalling protocol for Voice over IP (VoIP) based applications. The SIP proxy server is a software application that provides call routing services by parsing and forwarding all the incoming SIP packets in an IP telephony network. The efficiency of this process can create large scale, highly reliable packet voice networks for service providers and enterprises. We established that the efficient design and implementation of the SIP proxy server architecture can enhance the performance characteristics of a SIP proxy server significantly. Since SIP proxy server performance can be characterised by its transaction states of each SIP session, we emulated the performance model of the SIP proxy server and studied some of the key performance benchmarks such as average response time to process the SIP calls, and mean number of SIP calls in the system. We showed its limitations, and then studied an alternative based SIP proxy server performance model with enhanced performance model and studied additional key performance characteristics such as server utilisation, queue size and memory utilization. In [7], they provided the comparative results between the predicted results with the experimental results conducted in a lab environment. The slit difference between this study in [7] and our study is that this study is a study of performance and scalability metrics of a SIP proxy server as a practical approach where as our study focus on the VoIPv6 performance in application level of IPv6 networks.

In [8], Mahani *et al.* study the effects of concurrent voice connections on the performance metrics of communication network such as queue length, waiting time, packets service time and is very important. Mathematical analysis of such network especially with long-tail traffic help for a good capacity planning and also lead to an accurate admission control algorithms.

In this study a mathematical model of a communication network supporting VoIP and back-ground traffic with long-tail service time is considered. Some problems of previous mathematical models are identified and a new queueing system is proposed in which specifically the coexisting of heavy-tail and voice flows is addressed. The long-tail service time is approximated via hyper-Erlang distribution and also to achieving an accurate performance model a Markov reward model is introduced. The available bandwidth for long-tail distribution varies according to the Markov chain, describing the utilisation factor of voice connection. Numerical results show a comparison between exponential and heavy-tail service time and finally the effects of concurrent voice connections on the service time of heavy-tailed background packets is shown.

This study [9] focus on the effects of concurrent voice connections on the performance metrics of communication network such as queue length, waiting time, packets service time and is very important. Mathematical analysis of such network especially with long-tail traffic will help us for a good capacity planning and also lead to an accurate admission control algorithms. In this study a mathematical model of a communication network supporting VoIP and back-ground traffic with long-tail service time is considered. Some problems of previous mathematical models are identified and a new queueing system is proposed in which specifically the coexisting of heavy-tail and voice flows is addressed. We are missing a capable test framework that is part of our main source tree.

This means that building-in testing when working on FreeBSD's base system requires extra steps, and so is harder than should be.

We currently keep our unit tests and regression test cases under/usr/src/tools/regression/. These tests use ad-hoc ways to build and execute their test cases. Test case reporting is alsonot standardized though some tests use the Perl Test Anything Protoco. Running these tests in an automated way (a test "tinderbox") is not always possible.

At this point of time we do not archive test logs. Even if we did so, analysis of historical test data would be tedious due to the ad-hoc nature of the test reports.

The desirables from a FreeBSD test framework

- The ability to write tests that cover all the functionality of the base system.
- The ability to manage multi-machine tests (*i.e.*, *distributed* testing).
- A small, C-based test writing API that is easy to learn.

- Be capable of testing parts of the system that use threads.
- Integration with <bsd.*.mk>.
- Test logs should be easy to parse.
- Be available as open-source, and under a BSD-compatible license.

The rapid growth [1] of the Internet has led to the anticipated depletion of addresses in the current version of the Internet Protocol (IP), *i.e.*, IPv4. This depletion has given rise to a newer version of the IP, *i.e.*, IP version 6 (IPv6). IPv6 provides sufficient address space to meet the predicted increase of the Internet. Since IPv4 has already widely been deployed, it is required that the existing IPv4 and the newly added IPv6 can coexist and interoperate. Due to the incompatibility of the IPv4 and IPv6 headers, various mechanisms have been proposed to support the interoperability between IPv4 and IPv6. However, they are mostly designed for a static environment. Mobility support of mobile terminals in a mixed IPv4/IPv6 environment remains largely unexplored. It introduces additional overhead and delay to communications. In this paper, we analyze various handoff scenarios for a dual-stack mobile node with a predominant IPv6 home address roaming in a mixed IPv4/IPv6 environment. We investigate how handoffs can be supported and derive the handoff procedures for all scenarios. In addition, we analyze the impact of mobility support on the system performance in terms of handoff-signaling cost, handoff delay, and handoff-failure probability using our designed analytical models. Different traffic and mobility patterns are taken into account in the performance analysis. Numerical results are provided to demonstrate the performance of all handoff scenarios. Conclusions from this study can give great in-depth understanding and insights into designing new cost-effective mobility support mechanisms for IPv4/IPv6 transition and interoperability.

3. Experiments

Depending on the above two considerations in the Introductions above, Tests were conducted on the audio codec's (G.711, G.721, G.723, and G.729) in two machines with different processor speed. An audio file of nine seconds duration time was used to test the encode and decode files of all the above mentioned audio codec standards. These tests were meant to know the execution time for the different codec's codes as a performance testing trial. These tests were conducted in machines with two different processor speeds, 200 MHz and 450 MHz, to examine the effect of the processor speed on the processing time of the codes.

The following PC specifications (comparatively old machines) were used to conduct the Codec tests:

Pentium 200 MHz MMX Processor 64 Mbytes 100 MHz Random Access Memory. FreeBSD 4.5 Release operating system. KAME version 2001OS28\FreeBSD [2,10,11].

4. Results

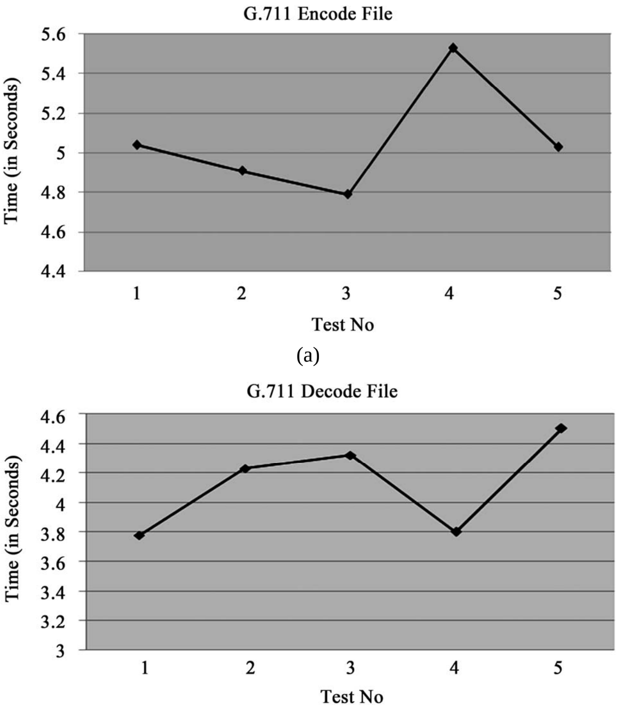
Figures 1(a) and **(b)** show the time required to execute both G.711 encode and decode files in a PC with the above specifications. **Figures 2(a)** and **(b)** show the time required to execute both G.721 encode and decode files. **Figures 3(a)** and **(b)** show the time required to execute both G.723 encode and decode files, and **Figures 4(a)** and **(b)** show the time required to execute both G.729 encode and decode files.

From the above **Figures 1-4** the best average execution time for both the encode and decode files is that of G.711, which act as an indicator for the better performance of G.711 compared with the rest of the tested codec's [2,10,11].

5. Quantifying Voice Quality

The voice quality was quantified and measured using a standardized ranking system called the Mean Opinion Score (MOS), which is a five-point scale described in ITU-T Recommendations P-800. On the surface, this system does not seem too scientific. The base of the MOS test is a matter of people listing to voice samples.

ITU-T P.800 makes number of recommendations regarding the selection of participants, the test environment,



(a)

(b)

Figure 1. (a) G.711 encode file execution time average execution time is 5.065732; (b) G.711 decode file execution time average execution time is 4.126231 seconds.

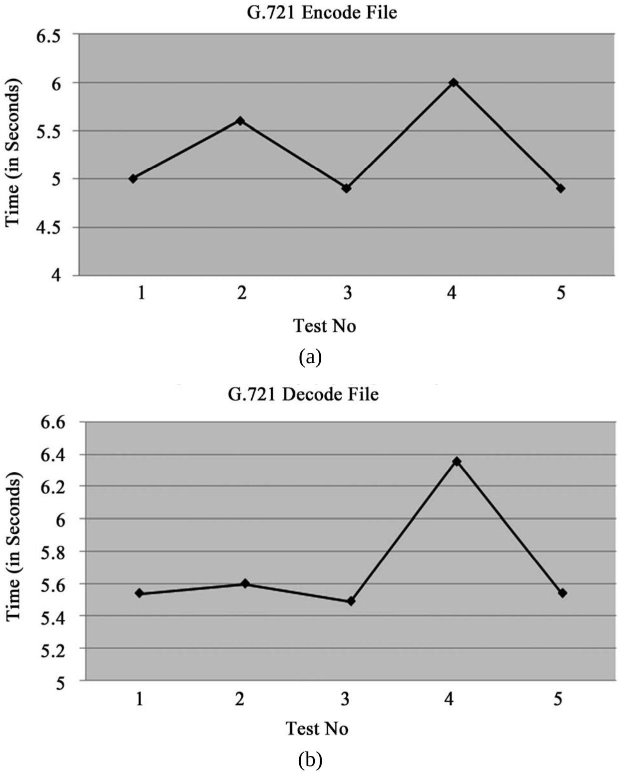


Figure 2. (a) G.721 encode file execution time average execution time is 5.267821 seconds; (b) G.721 decode file execution time average execution time is 5.709349 seconds.

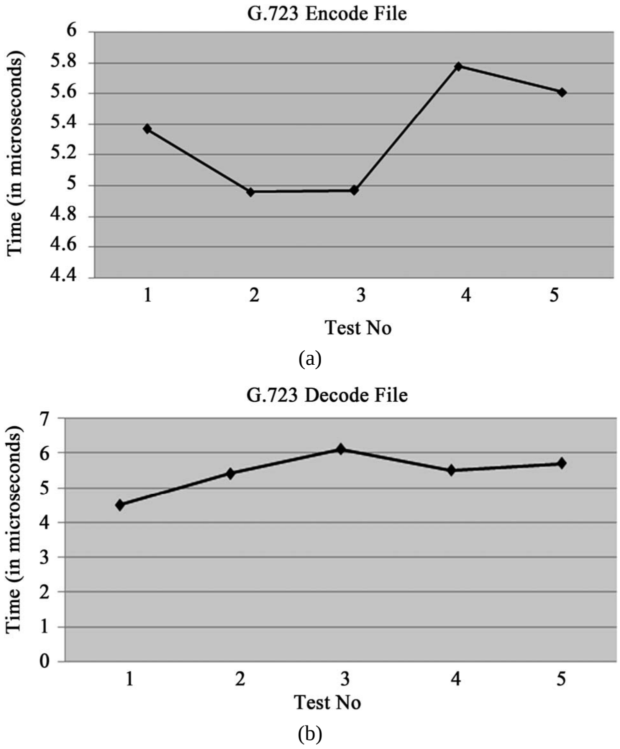


Figure 3. (a) G.723 encode file execution time average execution time is 5.338188 seconds; (b) G.723 decode file execution time average execution time is 5.421187 seconds.

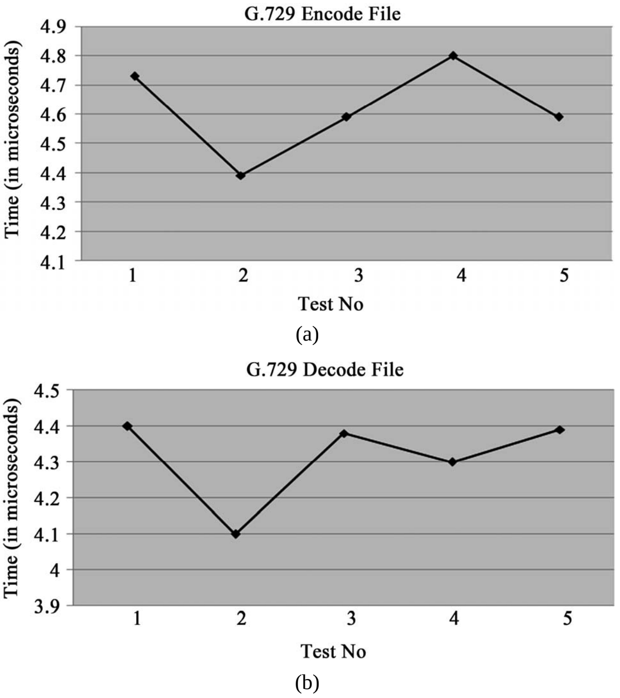


Figure 4. (a) G.729 encode file execution time average execution time is 4.617461 seconds; (b) G.729 decode file execution time average execution time is 4.309092 seconds.

explanations to listeners, analysis of results. Different MOS tests performed on the same coding algorithm tend to give roughly similar results (**Table 1**).

Test results of the MOS values for different Codec. The MOS tests conclusion is coinciding with the result of Free BSD tests [11-13].

IPv6 is documented in several RFCs (or request for comments) starting from RFC 2460. The IETF also publishes Experimental, Informational and Historic RFCs, and Best Current Practices (**Table 2**).

The text of this document about the RFC Editor function is based upon the proposal that USC ISI submitted to the Internet Society in 2006. This proposal was to provide RFC Editor services during 2007-2008, with an optional extension to 2009 (this option was approved). Note that the proposal was written during the summer of 2006; many of the proposed tasks have in fact been completed [15].

Table 1. Test results of the MOS values for different Codec.

Speech Coder	Bit Rate (Kbps)	MOS Value
G.711	64	4.3
G.726	32	4.0
G.723	63	3.8
G.728	16	3.9
G.729	8	4.0
GSM Full Rate (RPE_LTP)	13	3.7

Table 2. These RFC's specifies the VoIPv6 standards as mandates governing the protocol performance [14].

Protocol	Acronym	Purpose	RFC
Internet Protocol	IP	Physical network	RFC-791
Internet Control Message Protocol	ICMP	Status messaging	RFC-792
Transmission Control Protocol	TCP	Guaranteed delivery	RFC-793
User Datagram Protocol	UDP	Coordination, Audio	RFC-768
Telnet Protocol	TELNET	Remote login	RFC-764
File Transfer Protocol	FTP	Network utility	RFC-765
Simple Mail Transfer Protocol	SMTP	Email servers	RFC-788
Network News Transfer Protocol	NNTP	Usenet	RFC-977
Hypertext Transfer Protocol	HTTP	Web	RFC-2068

6. Conclusions and Future Works

From the above **Figures 1-4** we conclude the following:

- The best average execution time for both the Encode and decode files is that of G.711, which act as an indicator for the better performance of G.711 compared with the rest of the tested codes.
- Fluctuation and the degree of Uncertainty are high enough to affect the Quality of VoIPv6 performance.
- Any enhancement or quality improvement that could be made to the network infrastructure will lead directly to improvement in the VoIPv6 quality, Brood Band Internet is the best example of such improvement, this Brood Band Internet is been implemented in few countries in the world and Malaysia is expected to finalize this project by the beginning of the year 2012.
- Any enhancement or upgrading in the machines quality is expected to increase the overall quality of VoIPv6.
- As future work these tests could be done in different platforms other than KAME for FreeBSD such as USAGI which is IPv6 implementation in Linux kernel, and Microsoft IPv6 stack implementation.

REFERENCES

[1] J. Xie, "Performance Analysis of Mobility Support in IPv4/IPv6 Mixed Wireless Networks," *IEEE Transactions on Vehicular Technology*, Vol. 59, No. 2, 2010, pp. 962-973. doi:10.1109/TVT.2009.2034668

[2] V. Karthikeyan, N. Malmurugan and S. K. Jain, "Ambience-Based Voice over Internet Protocol Quality Testing Model," *IETE Journal of Research*, Vol. 55, No. 5, 2009, pp. 212-217. doi:10.4103/0377-2063.57598

[3] H. Qian, "Measurement and Analysis of VoIPv6 QoS Parameters over Mobile/Wireless Networks," *China Papers*, 4 October 2010.

[4] M. E. Flannagan, "Administering Cisco Qos in Ip Networks," *Syngress*, 11 April 2001, 519 p.

[5] H. A. Ibrahim, "Open Source VoIPv6," Ministry of Communication & Information Technology (MCIT)-The Egyptian IPv6 Task Force, Cairo, 2008.

[6] M. Larabel, "FreeBSD 8.0 Benchmarked against LINUX, Opensolaris," 2009. http://www.phoronix.com/scan.php?page=article&item=freebsd8_benchmarks&num=1

[7] S. V. Subramanian and R. Dutta, "A Study of Performance and Scalability Metrics of a SIP Proxy Server—A Practical Approach," *Journal of Computer & System Sciences*, Vol. 77, No. 5, 2011, pp. 884-897. doi:10.1016/j.jcss.2010.08.006

[8] A. Mahani, Y. S. Kavian, M. Naderi and H. F. Rashvand, "Heavy-Tail and Voice over Internet Protocol Traffic: Queuing Analysis for Performance Evaluation," *IET Communications*, Vol. 5, No. 18, 2011, pp. 2736-2743. doi:10.1049/iet-com.2010.0697

[9] Integration TET-FreeBSD Testing Today, 2008. http://wiki.freebsd.org/TetIntegration

[10] S. W. M. Gimgibeom, "Performance Evaluation of VoIPv6 in a Large-Scaled u-Army Network," *Korea Information and Communications Society*, Chap. 36, No. 8, 2011, p. 897.

[11] L. H. Duck and M. S. Won, "Network Performance of U-Army VoIPv6 Network Design Methodology Using Simulation," *Korea Institute of Communication*, Vol. 33, No. 10, 2008, p. 904.

[12] A. N. Moldovan and C. H. Muntaean, "Towards Personalised and Adaptive Multimedia in M-Learning Systems," *World Conference on E-Learning in Corporate, Government, Healthcare, and Higher Education*, Hawaii, 18 October 2011, pp. 782-791.

[13] P. Denisowski, "How Does It Sound," *IEEE Spectrum*, Vol. 38, No. 2, 2003, pp. 60-64. doi:10.1109/6.898801

[14] "Internet Protocol Specification A," 1981. http://www.rfc-editor.org/rfc/rfc791.txt

[15] S. Ginoza and A. Hagens, "The RFC Editor Function at ISI Bob Braden," USC Information Sciences Institute Marina del Rey, 2009. http://www.rfc-editor.org/RFCeditor.at.ISI.pdf