Available online at www.sciencedirect.com
ScienceDirect
Journal of Network and Computer Applications 31 (2008) 585–602
Journal of NETWORK and COMPUTER APPLICATIONS
www.elsevier.com/locate/jnca

An upper bound model for TCP and UDP throughput in IPv4 and IPv6

Eric Gamess*, Rina Surós

Central University of Venezuela, Escuela de Computación, Caracas, Venezuela

Received 10 July 2007; received in revised form 21 November 2007; accepted 26 November 2007

Abstract

Due to the shortage of public IPv4 addresses, the IETF has developed a new version of the Internet Protocol called IPv6. Many institutions all over the world had already started the migration to IPv6. Since this migration has to be done slowly, the first step is the coexistence of the two protocols (IPv4 and IPv6) for some years. One important issue for IPv6 to gain acceptance, is its performance in end-user applications. Hence, due to the availability of a variety of IPv6 implementations on different operating systems, it is important to evaluate the performance of the different IPv6 stacks, and compare it to the one shown by IPv4. In this paper, we present an upper bound model to compute TCP and UDP throughput for IPv4 and IPv6, in a full-duplex point-to-point connection. Our model can be used for any variant of Ethernet technology (10, 100, and 1000 Mbps). To validate this model, we did experiments and compared the maximum theoretical throughput with the experimental ones. Experiments were done with Windows XP SP2, Solaris 10, and Debian 3.1, which are very popular operating systems. The results show that 10 Mbps Ethernet technology is already very mature, since it gave performance very close to the maximum theoretical throughput. Experiments with FastEthernet (100 Mbps) show a TCP and UDP throughput close to the maximum theoretical throughput, especially for large payload. In the case of GigaEthernet (1000 Mbps), experimental results are not far from the maximum throughput for large TCP and UDP payload. However, for small TCP and UDP payload, the differences between our model (the maximum throughput) and the experiments are important. These differences should significantly decrease with the release of faster technology (processors and RAM).

Keywords: IPv4; IPv6; Benchmarks; TCP throughput; UDP throughput; Performance evaluation; Ethernet

*Corresponding author. Tel.: +58 212 6051061.
E-mail addresses: egamess@kuaimare.ciens.ucv.ve (E. Gamess), rsuros@ucv.ve (R. Surós).

1084-8045/$ - see front matter
doi:10.1016/j.jnca.2007.11.009

1. Introduction

Due to its rapid and unexpected growth, the Internet is facing serious problems in the last few years. Lack of adequate IPv4 address space may be slowing down the development of the Internet and new applications. Several proposals have been developed and implemented to solve these problems. A popular solution to the IPv4 address shortage is Network Address Translation (NAT) (Srisuresh and Egevang, 2001) which consists of hiding networks with private IPv4 addresses behind a NAT-enabled router with few public IPv4 addresses. As traffic passes from the private networks to the Internet, the source address in each packet is translated on the fly from the private addresses to the public addresses by the NAT-enabled router. Similarly, the destination address in each packet for incoming traffic is translated by the NAT-enabled router. However, NAT is a partial solution since it has drawbacks. Hosts behind a NAT-enabled router do not have true end-to-end connectivity and cannot participate in some Internet protocols. Services that require the initiation of connections from the Internet can be disrupted.

Another solution to the problem of the shortage of public IPv4 addresses that faces the Internet consists to migrate to the new version of the Internet protocol (Davies, 2002; Deering and Hinden, 1998; Popoviciu et al., 2006), called IPv6, or the coexistence between both protocols (Blanchet, 2006). IPv6 fixes a number of problems in IPv4, such as the limited number of available IPv4 addresses. IPv6 has a 128-bit address, while IPv4 has a 32-bit address. IPv6 also adds many improvements to IPv4 in areas such as routing and network autoconfiguration. In recent years, IPv6 has received significant attention from researchers, educational institutions, software vendors, and end users. All modern operating systems (Windows 2003/XP/Vista, Linux, Mac OS, AIX, Solaris, FreeBSD, etc.) support IPv6. For many applications, the overhead introduced by the operating system over the network performance is critical. However, only a few works had been presented to evaluate the performance of IPv6 at the operating system level. Ettikan (2000) and Ettikan et al. (2000) analyzed IPv6/IPv4 performance using simple applications (ping and FTP). They used computers with FreeBSD and KAME[1] IPv6 protocol stack to simulate routers. They reported latency using the ping utility, and throughput using the FTP application. Zeadally and Raicu (2003) evaluated IPv6/IPv4 performance on Windows 2000 (Microsoft IPv6 Technology Preview for Windows 2000) and Solaris 8. They connected two identical workstations using a point-to-point connection and reported results such as throughput, round-trip time, CPU utilization, socket-creation time, and client–server interactions, for both TCP and UDP. They used packets ranging from 64 to 1408 bytes. Their experimental results show that IPv6 for Solaris 8 outperform IPv6 for Windows 2000, while IPv4 outperform IPv6 for TCP and UDP for both operating systems. Zeadally et al. (2004) evaluated IPv6/IPv4 performance on Windows 2000, Solaris 8, and RedHat 7.3. The authors experimentally measured throughput of TCP and UDP, latency, CPU utilization, and web-based performance characteristics. Mohamed et al. (2006) evaluated IPv6/IPv4 performance on Windows 2003, FreeBSD 4.9 and RedHat 9. They measured throughput, round-trip time, socket-creation time, TCP-connection time, and number of connections per second in three different test-beds. The first test-bed consisted of a single computer and communication was limited to processes running in this computer using the loopback interface. In the second test-bed, two computers were connected

[1] http://www.kame.net