A Performance Analysis Model of PC-based Software Router Supporting IPv6-IPv4 Translation for Residential Gateway

Ssang-Hee Seo*, and In-Yeup Kong*

Abstract: This paper presents a queuing analysis model of a PC-based software router supporting IPv6-IPv4 translation for residential gateway. The proposed models are M/G/1/K or MMPP-2/G/1/K by arrival process of the software PC router. M/G/1/K is a model of normal traffic and MMPP-2/G/1/K is a model of burst traffic. In M/G/1/K, the arriving process is assumed to be a Poisson process, which is independent and identically distributed. In MMPP-2/G/1/K, the arriving process is assumed to be two-state Markov Modulated Poisson Process (MMPP) which is changed from one state to another state with intensity. The service time distribution is general distribution and the service discipline of the server is processor sharing. Also, the total number of packets that can be processed at one time is limited to K. We obtain performance metrics of PC-based software router for residential gateway such as system sojourn time, blocking probability and throughput based on the proposed model. Compared to other models, our model is simpler and it is easier to estimate model parameters. Validation results show that the model estimates the performance of the target system.

Keywords: Performance analysis model, Software PC router, IPv6-IPv4 translator, M/G/1/K, MMPP-2/G/1/K, residential gateway

1. Introduction

We can define a software router as a general-purpose computer that executes a computer program capable of forwarding an IP datagram among network interface cards attached to its I/O bus. It is well known that software routers have performance limitations because they use a single CPU and a single shared bus to process all packets. However, due to the ease with which they can be programmed for supporting new functionality, software routers are still important at the edge of the Internet [1].

Also, the next generation Internet requires the deployment of routing nodes that support a wealth of novel telecommunication services such as differentiated services, user mobility, multicast and secured communications, to name a few. It is generally accepted that when the routing nodes are supporting these services, a considerable amount of computing resources will be consumed. Therefore, performance models of routing nodes for evaluating system improvements, performing capacity planning and overload controlling are required [2].

In particular, these routing nodes can be used as a residential gateway for a home network. Instead of using a modem or a set-top box to receive and process broadband applications, some companies are using powerful home servers. Most of the servers are based on PowerPC or Intel processors. The home servers have the home networking software suite, which includes network address translation, DHCP server, and Micro-Web server components [3].

Manuscript received September 29, 2005; accepted November 15, 2005.

* Pusan National University, Pusan, Korea ({shseotwin, leafgirl}@pusan.ac.kr)

While there is a huge amount of papers reporting on the performance of Internet routing systems and Internet protocol implementation influenced by both the host's hardware and operating system architectures, few that study PC-based IP routers [4],[5],[6],[7] are publicly available.

In [1], the authors propose a parametrical model of a PC-based software router according to their experiences. In [8], two methods of approximating its performance are investigated. But several of the previous models are complicated. It lacks a simple model that is still valid in bursty traffic.

Recent works of Internet traffic appear to be self-similar with a long-range interval. Self-similar traffic is characterized by a correlation that never vanishes on a large time-scale. Its traffic looks the same regardless of time-scales over a long-range interval. This fractal behavior makes traffic very bursty.

Following this, we investigate and build a simple performance model of a PC-based software router, supporting communication between IPv4 networks and IPv6 networks at the gateway-level. We considered all traffic to have two patterns, which is normal in burst traffic. The burst traffic exhibits self-similar traffic. We viewed it as a queueing network of a software PC router with one node. Such a simple queueing model as the M/M/1/K with First-Come-First-Served (FCFS) service discipline can predict PC router performance quite well. But conceptually it is difficult to assume that the service distribution is exponential and that the service discipline is always FCFS.

In this paper, we present M/G/1/K and MMPP/G/1/K models for a software PC router supporting IPv6-IPv4

translation. The arrival process to the server is assumed to be a Poisson Process or a two-state Markov Modulated Poisson Process (MMPP) and the service time distribution to be arbitrary. MMPP is commonly used to represent burst arrival traffic to communication systems.

We use a software IP router running Linux 2.4 operating system and the NAPT-PT transition mechanism for communication between IPv4 networks and IPv6 networks. The NAPT-PT transition mechanism source is open to IPv6 Forum Korea [9].

This paper is organized as follows. In section 2, we describe the architecture and operation and the path of a packet in an IPv6-IPv4 translator. In section 3, we explain the characteristics of the queuing model. In section 4, we describe closed form expression for PC-based software router performance metrics. In section 5, we show the results and the discussion. Finally, we discuss our conclusions.

2. PC Router Translation System

2.1 System Architecture

The overall system architecture using S/W IPv6-IPv4 protocol translator consists of IPv6 Network, IPv4 Network and the 64Translator, which is a PC-based software router supporting IPv6-IPv4 translation.

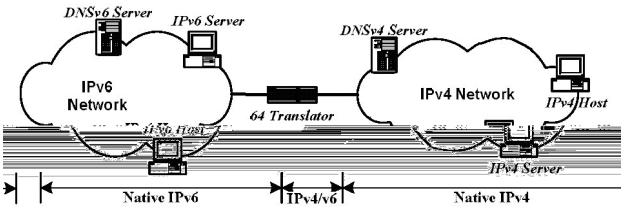

Fig. 1. The overall system architecture

2.2 IPv6-IPv4 Protocol Translator Architecture

The IPv6-IPv4 protocol translator is a gateway-level translator that exists between the IPv6 networks and IPv4 networks and provides transparent communication between the two networks. Among several translators, the IPv6-IPv4 protocol translator using NAPT-PT/SIIT is the fastest due to the IP-level translator. Fig. 2 illustrates the architecture of the IPv6-IPv4 protocol translator based on NAPT-PT/SIIT (Network Address Port Translation/ Stateless IP/ICMP Translation).

This protocol translator exists between IPv6 networks and IPv4 networks and provides the communication between the two networks. It is a dual-stack host and consists of NAPT-PT/SIIT, Mapping Table, DNS-ALG, and FTP-ALG. NAPT-PT/SIIT is the core module, which translates IP addresses and port numbers as well as IP/ICMP protocol headers. The mapping Table manages the mapping of the IPv4 address/port number and IPv6 address/port number. To support several applications extended or modified for IPv6, a protocol translator requires the ALG (Application Level Gateway). DNS-ALG translates resource records and IP addresses of DNS packets. Similarly, FTP-ALG translates commands and IP address/port numbers of FTP packets [9],[10],[11],[12].

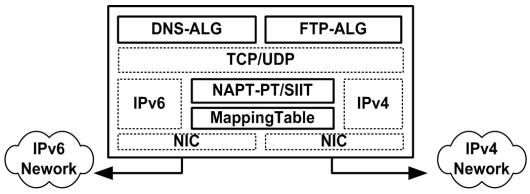

Fig. 2. IPv6-IPv4 translator based on NAPT-PT/SIIT

2.3 Functional Architecture

We show the functional architecture and the path of a packet in IPv6-IPv4 translator within a Linux-based software router in Fig. 3 [13],[14],[15].

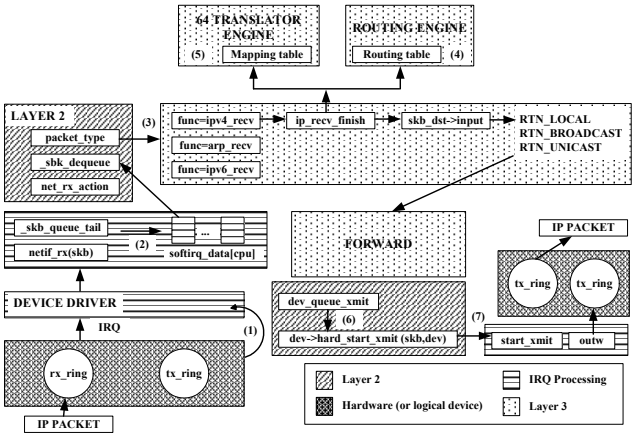

Fig. 3. The path of a packet in the IPv6-IPv4 translator

1) The packet arrives from the network. It is placed in hardware memory on the NIC. The card issues a hardware interrupt. The processor executes the device driver and copies the packet from the card to the main memory into a structure.

2) This structure is queued in a FIFO manner in the input queue. One such input queue exists per processor.

3) Before returning from the interrupt, the driver issues a soft interrupt. The filter is invoked. Once the software interrupt is allowed to execute, the structure is de-queued and information of the ISO layer 2 is analyzed. According to the value of the *type* field, the packet is passed on to the correct layer 3 function.

4) The destination IP address is extracted and the route cache is inspected. If the entry is not present in the cache, a search is performed in the routing table. The next-hop information is recorded in the structure.

5) If the packet is a packet of IPv6 host to communicate with an IPv4 host or a packet of IPv4 host to communicate with an IPv6 host, the translator allocates an address from its pool of addresses and the packet is translated to IPv4. This step requires the mapping table lookup.

6) According to the routing decision, the forwarder is then queued to the correct outgoing interface. One output queue exists per interface.

7) The packet is transferred to the hardware memory and the card is instructed to send the packet on the network.

3. Queueing Model Characteristics

There are several of the previous models on a PC-based software router. Our model is similar to the Jirachief-pattana model [8]. The Jirachiefpattana model is a M/M/1/ K whose service and inter-arrival time distribution are exponential, independent and identically distributed. The service discipline is ordinarily cyclical. That is, at most one packet from each queue is served in a cycle. The order of service within queues is FCFS. To approximate the time that each packet spends in the system, Jirachiefpattana follows the mean waiting time of a packet described by Boxma [16]. However, this method is complicated and is not suitable for representing the real traffic and characteristics of a PC-based software router.

Thus, we propose the M/G/1/K and MMPP-2/G/1/K queueing models, which have smaller parameters and are easy to estimate.

In the M/G/1/K model, the arrival process to the server is assumed to be the Poisson Process to represent normal traffic. And, in the MMPP-2/G/1/K model, the arrival process to the server is assumed to be the two-state Markov Modulated Poisson Process (MMPP-2) to represent burst traffic. MMPP-2 is commonly used to represent burst arrival traffic to communication systems. The service time distribution is arbitrary, since the arrived packets are translated or not translated according to the packet type. The total number of packets that can be processed at one time is limited to K. And we adopted the processor sharing (PS) scheduling algorithm for the service discipline. Processor Sharing (PS) is a good approximation for the round-robin discipline where the packets are served in turn, each for a small time slice. As known from the literature, the PS scheduling discipline offers smaller delays for customers with less work demands when compared with FIFO scheduling.

Fig. 4 illustrates the packet processing and the queuing model of the proposed PC-based software router. This sho ws that the PC-based software router is composed of two m ain parts. The first part consists of processors on network i nterface cards that have functions to receive and transmit p ackets. The second part is the CPU of the PC machine that forwards packets not destined for itself, and processes pack ets destined for itself. Thus, the basic components of the P C-based software router are receiver, transmitter and router.

The following explains packet processing and how to obtain the system sojourn time in models. The system sojourn time is one of the most important factors for performance metrics. When there is an incoming packet through the i^{th} interface $(i = 1, ..., n)$, the receiver of that interface receives such a packet and stores it in the input

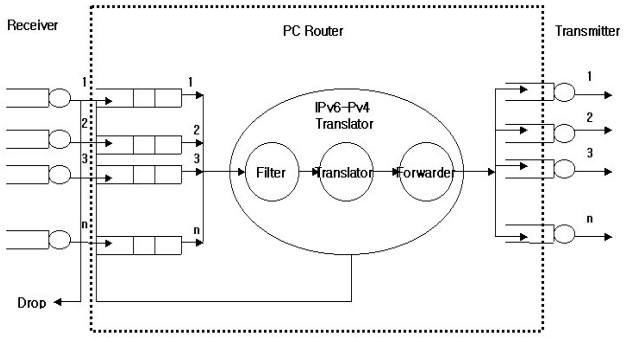

Fig. 4. A queuing model of PC-based software router

buffer of Q_i $(i = 1, ..., n)$ if the input buffer is not full; otherwise it will drop those packets. In the M/G/1/K model, the receivers which do this are independent, identically distributed stochastic variables RT_i with mean rt_i. In the MMPP-2/G/1/K model, the incoming packets are a doubly stochastic Poisson process where the arrival rate is determined by the state of a continuous-time Markov chain. The Markov chain consists of two different states, s_1 and s_2. Consumed time_of_state phases i have exponential respective distributions. Accordingly, we assumed $RT_i^j (j = 1,2)$ to be independent, identically distributed stochastic variables with mean rt_i. Once there is a packet in the output buffer of the j^{th} interface $(j = 1, ..., n)$, the transmitter of that interface picks the packet from the output buffer and transmits it. The times for transmitters to do this are independent, identically distributed stochastic variable TT_j with mean tt_j. Through which interface a packet will be transmitted is dependent on the IP destination address of such a packet, and is random. Thus the average time of picking and transmitting a packet, tt, is defined as

$$tt = \sum_{j=1}^{N} P_j tt_j \tag{1}$$

A single service facility of the PC router serves in a round-robin manner. The service times of packets of the i^{th} network interface are arbitrary, with first and second moments $E[S]$ and $E[S^2]$. Accordingly, the average time spent in the system by each packet through the i^{th} network interface $Q_i (i = 1, ..., n)$ is the expected sojourn time of a packet. This value is estimated from the measured average response time.

Finally, the system sojourn time $E[ST]$ spent in the system by each packet through the network interface can be defined as:

$$E[ST] = rt_i + E[W_i] + E[S_i] + tt = E[T] \quad (2)$$

$E[T]$ denotes the average response time and $E[W_i]$ denotes the mean waiting time of a packet in $Q_i (i = 1, ..., n)$.

4. Queueing Model of S/W PC Router

4.1 M/G/1/K Processor Sharing Model

We model the PC-based software router using an M/G/1/K processor sharing queue. The packets arrive according to a Poisson process with rate λ. The average service time has a general distribution with mean $E(S)$. The $E(S)$ is the inverse of μ, which is service rate. An arrival will be blocked if the total number of packets in the system has reached a predetermined value K. A packet in the queue receives a small quantum of service and is then suspended until every other packet has received an identical quantum of service in a round-robin fashion. When a packet has received the amount of service required, it leaves the queue.

Thus, such a system can be viewed as a queuing network with one node [17]. We propose the M/G/1/K queueing model as the performance model for the PC-based software router.

In the M/M/1/K FCFS model, the probability mass function (pmf) of the total number of packets in the system has the following expression, where ρ is the offered load and is mathematically defined as the packet arrival rate, λ, divided by the service rate, μ.

$$P[N = n] = \frac{(1-\rho)\rho^n}{(1-\rho^{K+1})} \quad (3)$$

We note that an M/M/1/K FCFS queue has the same pmf as M/G/1/K processor sharing [18], [19]. However the service time distribution of the M/M/1/K FCFS queue must be exponential and its service discipline must be FCFS.

From (5), we can derive the following performance metrics, average response time, throughput and blocking probability.

The probability of blocking P_b is equal to the probability that there are K packets in the system.

$$P_b = P[N = K] = \frac{(1-\rho)\rho^K}{(1-\rho^{K+1})} \quad (4)$$

The throughput H is the rate of completed packets. When the PC router reaches equilibrium, H is equal to the rate of accepted packets,

$$H = \lambda(1 - P_b) \quad (5)$$

The average response time $E[T]$ is the expected sojourn time of a packet. Following Little's law, we know that

$$E[T] = \frac{E[N]}{H} = \frac{\rho^{K+1}(K\rho - K - 1) + \rho}{\lambda(1-\rho^K)(1-\rho)}. \quad (6)$$

$E[N]$ is the mean number of packets in the system.

To get theoretical results that we can make comparisons to, we can estimate λ and μ from measurements. So by a simple simulation program, we have attained theoretical results. The actual results will appear in section 5.

4.2 MMPP-2/G/1/K Processor Sharing Model

We constructed our model for the bursty packet traffic from the two-state Markov Modulated Poisson Process (MMPP). A MMPP is a doubly stochastic process where the intensity of a Poisson process is defined by the state of a Markov chain. A two-state MMPP means that the Markov chain consists of two different states, s_1 and s_2. The Markov chain changes state from s_1 to s_2 with intensity r_1, and transits back with intensity r_2. When the MMPP is in state s_1, the arrival process is a Poisson process with rate λ_1, and when the MMPP is in state s_2, rate λ_2 is used. The packets arrive according to two-state MMPP with parameters, $\lambda_1, \lambda_2, r_1, r_2$.

The mean rate $E[\lambda]$ and the variance $Var[\lambda]$ in a two-state MMPP are given as follows by Heffes [20].

$$E[\lambda] = \frac{\lambda_1 r_2 + \lambda_2 r_1}{r_1 + r_2} \quad (7)$$

$$Var[\lambda] = \frac{r_1 r_2 (\lambda_1 - \lambda_2)^2}{(r_1 + r_2)^2} \quad (8)$$

In MMPP, $E[\lambda]$ is the mean arrival rate of burst traffic. Thus, to obtain performance metrics, λ can be redefined as $E[\lambda]$ in (7), (8).

4.3 Parameter Estimation

There are two parameters, $E[T]$ and K, in our model. We assumed that the average response time for a certain arrival rate could be estimated from measurements and had general distribution. The estimations, $\hat{E[T]}$ and $\hat{K}$, were

obtained by maximizing the likelihood function of the observed average response time.

We let $E[T]_i$ be the average response time predicated from the model and $E[\hat{T}]_i$ be the average response time estimated from the measurements when the arrival intensity is λ_i, $i = 1...m$. Since the estimated response time $E[\hat{T}]$ is the mean of the samples, it is approximately a normal distributed random variable with mean $E[T]$ and variance σ_T^2 / n when the number of samples n is very large. Hence $E[T]$ and K could be estimated by maximizing the log-likelihood function of the observed average response time.

$$\log \prod_{i=1}^{m} \frac{1}{\sqrt{2\pi\, \sigma_i^2 / n_i}} \exp\left[\frac{(E[\hat{T}]_i - E[T]_i)^2}{2\sigma_i^2 / n_i} \right] \tag{9}$$

Maximizing the log-likelihood function is equivalent to minimizing the weighted sum of square errors as follows,

$$\sum_{i=1}^{m} \frac{(E[\hat{T}]_i - E[T]_i)^2}{\sigma_i^2 / n_i} \tag{10}$$

In this paper, we used a truncated Newton approach. Our approach is based on [21].

5. Experimental Results

The method developed in section 4.3 was used to estimate the parameters from the measurements. The result is presented in Table 1. Using the estimated parameters, we can predict the PC-based software router performance and compare it with the measurements.

Table 1. Estimated Parameter of the Model

Bandwidth (Mbps)	E[T] (sec)	Bandwidth (Mbps)	E[T] (sec)	K
1	0.000257	50	0.006226	10
5	0.000503	60	0.006144	10
10	0.001546	70	0.006367	10
20	0.004546	80	0.006645	10
30	0.005695	90	0.006480	10
40	0.006010	100	0.006586	10

We were interested in the following performance metrics: system sojourn time, throughput and blocking probability. The throughput is estimated by taking the number of bits per second between the total number of successful packets and the time span of measurement. The system sojourn time is the time difference between when the IPv6 host sends a packet and when the IPv4 host receives the packet. This is the expected sojourn time of a packet. The blocking probability is estimated as the drop ratio between when the IPv6 host sends a packet and when the IPv4 host does not receive the packet.

Also, we assume that the average response time for a certain arrival rate can be estimated from the measurements. We chose to set the mean arrival rate for the MMPP process, and then to determine MMPP parameters from that value. r_1 and r_2 were set to 0.5 and 0.5 respectively. The low rate, λ_1, was set to $0.7 \bullet E[\lambda]$. λ_2 is a high rate and can be seen as a sudden burst rate. λ_2 was used 50% of the time according to settings of r_1 and r_2.

In order to analyze the performance metrics of the PC-based software router, we adopted the public domain traffic generator called *Multi-Generator 4.x(MGEN)* [22]. Offered bandwidth from 1Mbps through 100Mbps, the payload size of 1440 bytes, and the duration of 30 sec. were used. Since the current implementation of *MGEN* supports the UDP only, we conducted the experiment using the UDP.

The IPv6 host works on a 2.6GHz Pentium PC running Linux. The IPv4 host works on a 2.0GHz Pentium PC with Linux. The IPv6 host and IPv4 host connect to 100Mbps Ethernet respectively and communicate with each other via the IPv6-IPv4 translator. The IPv6-IPv4 translator works on an AMD 2400+ MP running Linux.

Using the estimated parameters, we could predict the PC router performance and compare it with the measurements. Fig. 5, 6 and 7 show the blocking probability, the throughput, and average response time of M/G/1/K model and Fig. 8, 9 and 10 show those of the MMPP-2/G/1/K model. Also, Fig. 11, 12 and 13 show the blocking probability, the throughput, and average response time of the M/G/1/K model when K is varied.

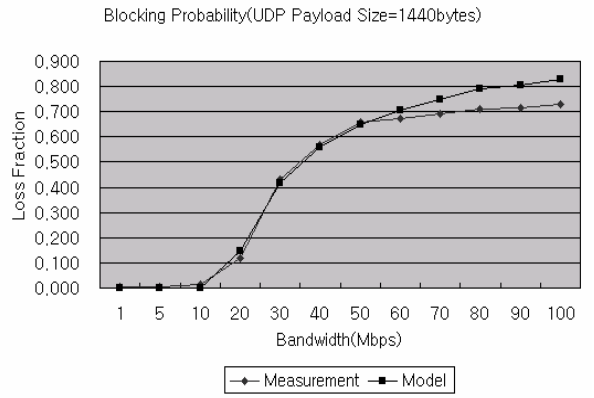

Fig. 5. Average Blocking Probability

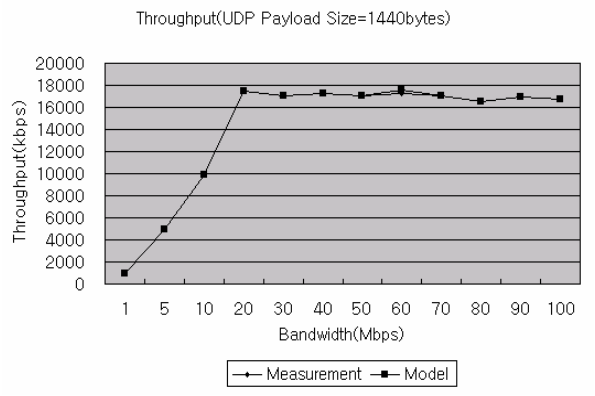

Fig. 6. Average Throughput

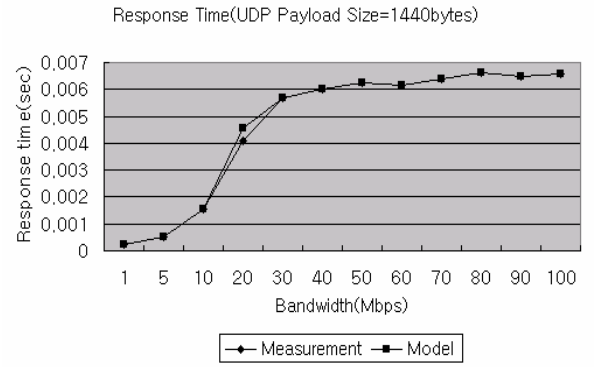

Fig. 7. Average Response Time

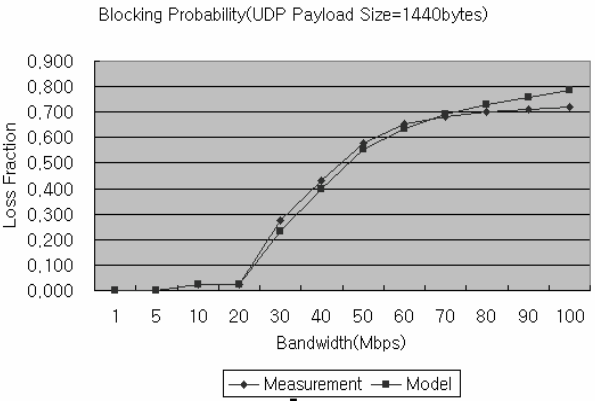

Fig. 8. Average Blocking Probability

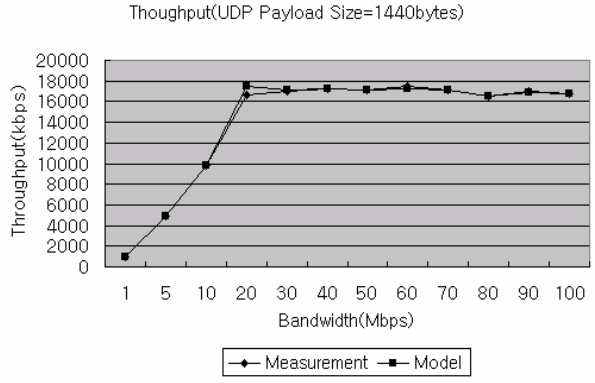

Fig. 9. Throughput

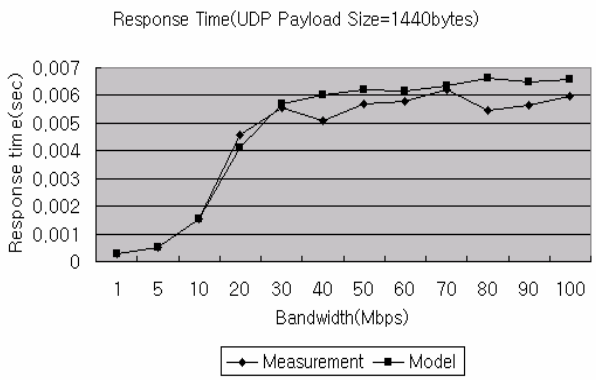

Fig. 10. Average Response Time

Fig.s 5, 6, and 7 all show that the measured and the predicted data were saturated when the bandwidth was more than a certain bandwidth. Fig. 5 shows that the average blocking probability was slightly greater than the measurement. The error in the average blocking probability was expected since we only use the measured values in our parameter estimation. Fig.s 6 and 7 are a good fit.

Fig. 8, 9 and 10 show the performance metrics of the MMPP-2/G/1/K model. Fig.s 8, 9 and 10 are similar to Fig.s 5, 6 and 7. Fig.s 8, 9 and 10 show that the difference in performance metrics by arrival process of the PC-based software router is small. It shows that the MMPP-2/G/1/K model is valid as a PC-based software router model with self-similar traffic. Fig.s 8 and 9 show that performance metrics predicated fit well to the experimental outcome. But, Fig. 10 shows some difference. This difference is based on several factors. First, link delays on the communication line between the IPv4 host and PC router, and between the PC router and IPv6 host can be ignored. They are directly connected. Second, in our test bed, there was little background traffic except for intermittent traffic for network management. Because most traffic offered was loaded by the generator, overall traffic showed a little invariance. For these reasons, the predicated data were greater than the measurement data. Fig. 7 has some difference for the same reasons as Fig. 10.

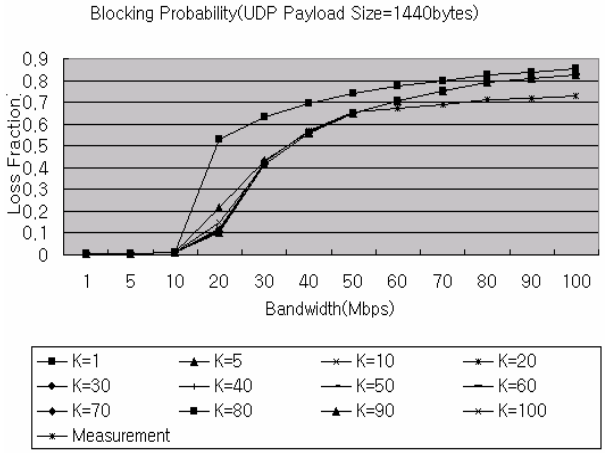

Fig. 11. Average Blocking Probability

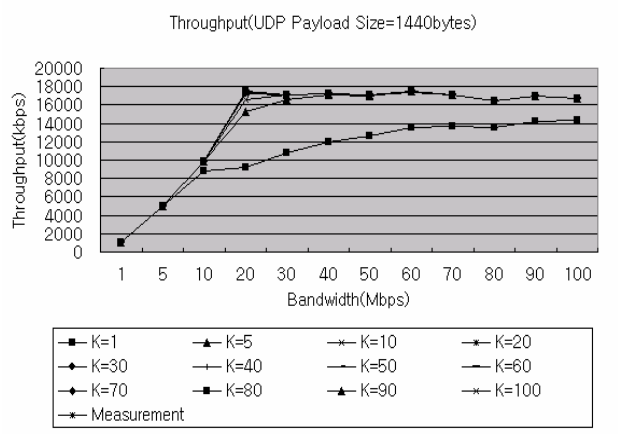

Fig. 12. Average Throughput

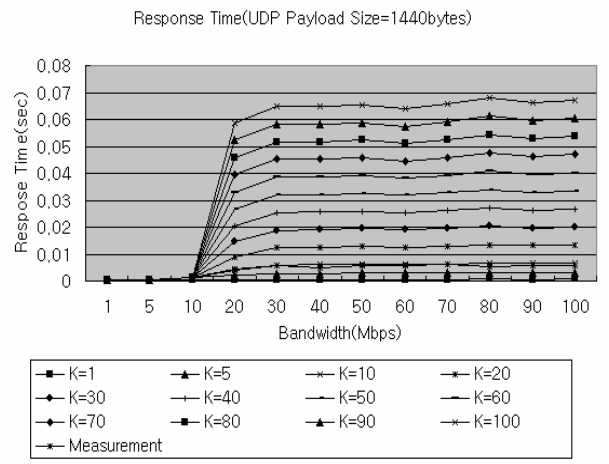

Fig. 13. Average Response Time

Also, we measured the change in performance metrics of the M/G/1/K model when K is varied. Fig.s 11, 12 and 13 show the change of average blocking probability, throughput and average response time by K , respectively.

Fig.s 11 and 12 show that average blocking probability and throughput are hardly influenced by K . But, Fig. 13 shows the greater K is, the greater the average response time. The average response time is calculated by sum of mean service time and mean waiting time. So, the greater K is, the greater mean waiting time. Also, the mean service time is related to the service discipline of the server. Since the processor is sharing discipline services at the rate $1/n$ if there are n packets in system, the greater K is, the greater the total service time. Hence, the greater K is, the greater the average response time.

6. Conclusion

In order to analyze the performance of the PC-based software router supporting IPv6-IPv4 translation for a residential gateway, we present the M/G/1/K and MMPP-2/G/1/K processor sharing queueing models. We have derived expressions for performance metrics such as system sojourn time, blocking probability and throughput using measured average response time. Moreover, we obtained the experimental results, which were similar to the results induced by our model. As a result of our experiment, we can verify our model. Future works include Quality of Service (QoS) supporting of a PC-based software router. The QoS allows home networking applications to prioritize individual services. The QoS guarantee of a residential gateway is essential for entertainment-based applications delivery over home networks. Also, a study of overhead factors is needed to improve the performance of the PC-based software router.

References

[1] O. I. Lepe, and J. Garcia "A Performance Model of a PC Based IP Software Router," IEEE ICC 2002, New York, 2002.

[2] O. I. Lepe, and J. Garcia "I/O bus usage control in PC-based Software Router," IFIP Networking 2002, Pisa, 2002.

[3] Gerard O'Driscoll, The Essential Guide to Home Networking Technologies. Prentice Hall, 2001.

[4] D. D. Clark, Van Jacobson, John Romkey, and Howard Salwen. "An analysis of TCP processing overhead," IEEE Communications Magazine, Vol. 27, No. 6, pp. 23-29, IEEE, June, 1985.

[5] C. Papadopoulos, and G. M. Parulkar. "Experimental evaluation of SUNOS IPC and TCP/IP protocol implementation," IEEE/ACM Transactions On Networking, Vol. 1, No. 2, pp. 199-216, April 1993.

[6] J. Kay, and J. Pasquale. "Profiling and reducing overheads in TCP/IP," IEEE/ACM Transactions On Networking, Vol. 4, No. 6, pp. 817-828, December 1996.

[7] X. Qie, A. Bavier, L. Peterson and S. Karlin, "Scheduling Computations on a Software-Based Router," Proc. of SIGMETRICS 2001, June 2001.

[8] A. Jirachiefpattana, P. County, T.S. Dillon, and R. Lai, "Performance evaluation of PC routers using a single-server multi-queue system with a reflection technique," Computer Communications '97, Elsevier Science, 1997.

[9] IPv6 Forum Korea, "Linux-based Userspace NAT-PT," http://www.ipv6.or.kr/.

[10] G. Tsirtsis, and P. Srisuresh, "Network Address Translation - Protocol Translation (NAT-PT)," *IETF RFC 2766*, February 2000.

[11] E. Nordmark, "Stateless IP/ICMP Translation Algorithm (SIIT)," *IETF RFC 2765*, February 2000.

[12] Marc E. Fiuczynski, Vincent K. Lam, and Brian N. Bershad, "The Design and Implementation of an IPv6/IPv4 Network Address and Protocol Translator," *Proc. of the USENIX Conference*, June 1998.

[13] Daniel P. Bovet and Marco Cesati, Linux kernel, O'Reilly, 2001.

[14] Paul Gortmarker, Linux Ethernet-howto, 2000.

[15] Alessandro Rubini and Jonathan Corbet, Linux Device

Drivers, O'Reilly, 2001.
[16] O.J. Boxma and B. Meister, "Waiting-time approximations for cyclic-service systems with switch-over times," Performance '86, Elsevier Science, 1986.
[17] P. J. B. King, Computer and Communication Systems Performance Modeling, Prentice Hall, 1990.
[18] L. Kleinrock, *Queueing Systems, Volume 1: Theory*. John Wiley & Sons, 1975.
[19] S. Lam, "Queueing networks with population size constraints," *IBM Journal of Research and Development*, vol. 21, no. 4, pp. 370-378, July 1977.
[20] H. Heffes, "A Class of Data Traffic Processes – Covariance Function Characterization and Related Queuing Results", The Bell System Technical Journal, Vol. 59, No. 6, July-August, 1980.
[21] J. Cao, M. Andersson, C. Nyberg, M. Kihl, "Web server Performance Modeling Using an M/G/1/K*PS Queue", at the 10th International Conference on Telecommunications, 2003, Papeete, Tahiti.
[22] Naval Research Laboratory (NRL), *The Multi-Generator (MGEN) toolset*, http://manimac.itd.nrl.nay.mil/MGEN/.

Ssang-Hee Seo
She received her PhD in Computer Engineering from Kyungnam University, Masan, Republic of Korea, in 2000. She is now the BK21 assistant professor of Pusan National University. Her research interests are in IPv6, IPv6/IPv4 translator, home network and performance analysis.

In-Yeup Kong
She received her bachelor's degree and MS degree in Computer Engineering from Pusan National University, Pusan, Republic of Korea, in 2000 and 2002. She is now a PhD student at Pusan National University. Her research interests are in IPv6, IPv6/IPv4 translator, and home network.