

# Inline bandwidth measurements: Implementation difficulties and their solutions

Tomoaki Tsugawa, Cao Le Thanh Man, Go Hasegawa, and Masayuki Murata  
Graduate School of Information Science and Technology, Osaka University  
1-5 Yamadaoka, Suita, Osaka, 560-0871, Japan  
{t-tugawa, mlt-cao, hasegawa, murata}@ist.osaka-u.ac.jp

**Abstract**— We have proposed the concept of *inline network measurement*, which involves the concept of “plugging” an active bandwidth measurement into an active TCP connection. Mechanisms using this method have the advantage of requiring no extra traffic for measuring available bandwidth, whereas other active measurement tools cannot fundamentally avoid adding probing traffic onto the network. However, when the inline network measurement algorithms are implemented in general-purpose computers, some problems arise, such as the clock resolution of the kernel system, Interrupt Coalescence (IC) deployed in network interface cards, and the behavior of TCP receiver. In the present paper, we explain these difficulties and describe our current solutions. Furthermore, we implement the measurement algorithms and the solutions against those problems in a FreeBSD 4.10 kernel system, and present some results on experimental networks. The experimentally obtained results are used to verify the solutions and to confirm the effectiveness of our concept, inline network measurement, on actual networks. We also compare the performance of the packet interval-based approaches and packet-burst interval-based approaches, and demonstrate that using packet-burst for the measurement in high-speed networks is quite effective.

**Keywords**— Implementation, Inline network measurement, Available bandwidth, Clock resolution, Interrupt Coalescence (IC), Transmission Control Protocol (TCP)

## I. INTRODUCTION

According to [1], the term “available bandwidth” means the bandwidth information that is the unused portion of the network path between sender and receiver hosts. When the transmission speed of the network interface at the sender endhost is the lowest among all links of the network path, the available bandwidth is equal to the transmission speed. Available bandwidth information plays an important role in adaptive control of the networks. For example, Transmission Control Protocol (TCP), which is a major network transport protocol, can optimize link utilization or improve transmission performance, especially in high-speed networks, by using such information. TCP using the available bandwidth information can also provide prioritized data transmission. We have proposed some application techniques based on the available bandwidth information in [2, 3].

There exist several passive and active measurement approaches for measuring the available bandwidth [4–8]. Although active approaches are preferred because of their accuracy and measurement speed, sending extra traffic onto the network is a common disadvantage. Existing measurement algorithms also require a long time to obtain one measurement result. To counter the above problems, we have proposed the concept of *inline network measurement* [9, 10]. The concept involves “plugging” an active bandwidth measurement into an active TCP connection as shown in Figure 1. Mechanisms using this method have the advantage of requiring no extra traffic for measuring available bandwidth. Moreover, by using the active TCP connection, measurement algorithms can be implemented only by modifi-

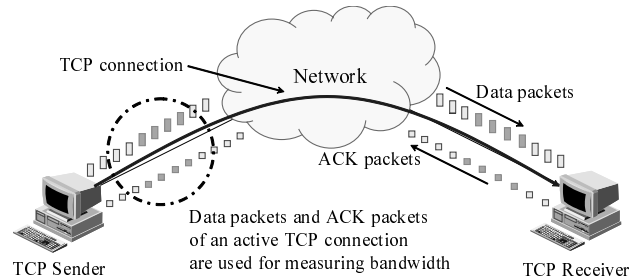


Fig. 1. Inline network measurement

cation of the sender endhost. We proposed a packet interval-based mechanism based on this concept, which is referred to as ImTCP [9].

The inline network measurement mechanisms have the advantages described above. However, some problems occur when implementing the mechanisms in real systems. The most serious problem is the resolution of the clock used in the kernel system. In bandwidth measurement algorithms based on packet transmission/arrival intervals, the sender endhost adjusts the intervals of probe packets and sends the packets to the network. The receiver endhost then observes the arrival intervals of the packets (or the sender endhost checks the arrival intervals of echoed packets from the receiver endhost) and estimates the available bandwidth by using the information of the sending/receiving intervals. Therefore, the packet interval-based algorithms require a timer with sufficiently fine-grained resolution to adjust the sending intervals of probe packets. The clock resolution used in the kernel system, however, is generally coarser than that of the clock used in upper-layer applications, and this coarse clock resolution may reduce the accuracy of measurement results.

In gigabit networks, this problem becomes more significant, and even stand-alone active measurement applications such as Pathload, do not work well [11]. Measurement in fast networks requires short transmission intervals of the probe packets (e.g., 12  $\mu$ sec for a 1 Gbps link and 1.2  $\mu$ sec for a 10 Gbps link with 1500 Byte packet). However, regulating such short intervals results in a heavy CPU load. In addition, network interface cards for high-speed networks usually employ Interrupt Coalescence (IC) [12, 13], which rearranges the arrival intervals of packets and causes bursty transmission. As a result, packet interval-based algorithms do not work properly. To counter the problems related to high-speed networks, we proposed the solution, which is referred to as Interrupt Coalescence-aware Inline Mea-

TABLE I  
PARAMETER  $HZ$ , CLOCK RESOLUTION, AND MEASURED BANDWIDTH

$HZ$	Clock resolution (1 tick) [ $\mu\text{sec}$ ]	Measured bandwidth [Mbps]				
		1 tick	2 ticks	3 ticks	4 ticks	5 ticks
100	10,000	1.2	0.6	0.4	0.3	0.24
1,000	1,000	12	6	4	3	2.4
10,000	100	120	60	40	30	24
20,000	50	240	120	80	60	48
50,000	20	600	300	200	150	120
100,000	10	1,200	600	400	300	240

TABLE II  
KERNEL COMPILATION TIME

$HZ$	Kernel compilation time [sec]
100	168.20
1,000	170.09
10,000	183.38
20,000	199.78
50,000	277.84
100,000	734.10

surement (ICIM) [10]. Unlike packet interval-based mechanisms such as Pathload, we deploy a packet-burst interval-based approach that measures the available bandwidth by adjusting the number of packets that are transmitted in a burst as a result of IC. The effectiveness of ICIM has been evaluated through the simulation experiments in our previous study [10].

In the present paper, we discuss the effectiveness of inline network measurement algorithms by implementing and testing ImTCP and ICIM, on actual networks. We evaluate the relationship between the accuracy of the measurement results of ImTCP and the kernel parameters regarding the clock resolution. We then confirm that ICIM can work well in high-speed networks and clarify the advantage of the packet-burst interval-based mechanism in such networks. In particular, we demonstrate that ICIM can measure the available bandwidth when the clock resolution is not so fine. Through these experiments, we clarify the limitation and the adaptability to high-speed networks of packet interval-based and packet-burst interval-based mechanisms, and establish important objectives with respect to the implementation and development of network bandwidth measurement tools.

The remainder of the paper is organized as follows. Section II describes the difficulties related to the implementation of the bandwidth measurement mechanism in a kernel system. In Section III, we briefly introduce our solutions, which are ImTCP and ICIM, and their implementations in the FreeBSD 4.10 kernel system. Section IV presents the results of the implementation experiments in order to clarify the limitation of packet interval-based algorithms and the advantages of packet-burst interval-based algorithms in high-speed networks. Finally, we present conclusions and areas for future study in Section V.

## II. DIFFICULTIES IN IMPLEMENTING BANDWIDTH MEASUREMENT IN A KERNEL SYSTEM

In this section, we describe the difficulties involved when implementing the proposed mechanisms based on the inline network measurement in real systems.

### A. Clock resolution of kernel system

When bandwidth measurement mechanisms based on adjusting/observing the transmission/arrival intervals of packets are implemented in a kernel system, the clock resolution used in the kernel system becomes important. Packet interval-based mechanisms need to adjust the transmission intervals of data packets in order to estimate the available bandwidth. When we imple-

ment the measurement algorithm as an upper-layer application program, the application program continuously checks the system clock (e.g., using `gettimeofday()` in UNIX systems) and send the packets when the clock reaches a predetermined timing. In a Linux system with an x86-based CPU, one hardware clock access requires approximately  $1.9 \mu\text{sec}$  (in FreeBSD system, one access requires  $9 \mu\text{sec}$ ) [14]. The `write()` system call requires an average of  $2 \mu\text{sec}$ . Therefore, a Linux system can send packets in intervals of  $3.9 \mu\text{sec}$ . In the kernel system, on the other hand, adjusting the intervals of data packets is generally realized by registering the packet sending program to an Interrupt Service Routine (ISR) of the system clock interrupt. In a general-purpose UNIX OS, the ISR `hardclock()` is provided for this purpose. In FreeBSD and Linux, the `hardclock()` system call is called by the interrupt of the system clock every 10 msec, that is, the resolution of the kernel system clock is generally coarser than that of the upper-layer applications. This coarse resolution may reduce the accuracy of measurement results.

The clock resolution is determined by the parameter  $HZ$ , which is set to 100 by default, corresponding to a clock resolution of 10 msec, in the FreeBSD kernel system. Table I summarizes  $HZ$ , the clock resolution, and the rate of data transmission (or measured bandwidth) when adjusting the intervals of packets based on the resolution. When  $HZ$  is chosen to be 100, the resolution of the kernel clock becomes 10 msec. Under this setting, packet interval-based mechanisms can only measure the available bandwidth up to 1.2 Mbps. Moreover, we can see that the bandwidth resolution becomes coarse as the measured bandwidth approaches the upper limit. Therefore, when packet interval-based mechanisms measure the available bandwidth,  $HZ$  should be set so that the upper limit of the measurable bandwidth is sufficiently large with respect to the link bandwidth of the network.

However, when  $HZ$  is set to a large value, the timer interrupts by the kernel system occur frequently and the overhead for processing interrupts affects the entire performance of the system. For example, Table II summarizes the relationship between the value of  $HZ$  and the required time for the compilation of the kernel source code in the FreeBSD system in a PC having a 3.0-GHz CPU (Intel) and a memory of 1,024 MBytes. The results show that as  $HZ$  becomes large, the processing time becomes large rapidly, meaning that the performance of the system is degraded. In inline network measurement, a TCP data transfer and a bandwidth measurement task are conducted simultaneously on a single endhost, so an excessively large overhead for measure-

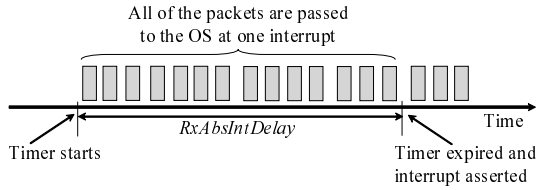


Fig. 2. Receive absolute timer

ment should be avoided. Therefore, when determining the value of  $HZ$ , the trade-off relationship between the clock resolution and the performance should be considered.

### B. Interrupt Coalescence (IC)

In high-speed networks (1 Gbps or higher), another difficulty is encountered by existing active bandwidth measurement tools, such as Interrupt Coalescence (IC), which is deployed in most gigabit Network Interface Cards (NICs) [12, 13]. IC is a technique in which NICs group multiple packets arriving in a short period of time and pass these packets to the OS in a single interrupt. Without IC, an OS interrupt occurs whenever a single packet arrives, which leads to a high CPU overhead when the system performs high-speed data transmission. In other words, IC is an important technique for reducing the CPU overhead when the arrival intervals of packets become small. However, IC has an enormous impact on packet interval-based bandwidth measurement tools because the inter-arrival intervals of packets observed by the kernel are changed.

According to [12], although there are a number of timer setting in IC, the absolute timers are the primary source of device interrupts under sustained loads. There are two absolute timers. One is for transmit interrupts, and the other is for receive interrupts. Because transmit interrupts only inform the kernel as to the completion of packet sending, delays in transmit interrupts do not affect the real transmission intervals of the packets. In contrast, delays in receive interrupts change the intervals of all receiving packets observed by the kernel. As shown in Figure 2, the receive absolute timer starts to count down upon receipt of the first packet. Subsequent packets do not alter the countdown. Once the timer reaches zero, the controllers generate an interrupt to pass all of the packets to the OS in a bursty manner. The length of the timer is decided by the parameter  $RxAbsIntDelay$  in the FreeBSD system. Thus, all packets that have time intervals smaller than  $RxAbsIntDelay$  belong either to the same burst, in which case the time interval between the packets approach zero, or to two successive bursts, in which case the time interval becomes  $RxAbsIntDelay$  or larger.

### C. Behavior of the TCP receiver

Packet interval-based mechanisms implicitly expect the TCP receiver to send an ACK packet back to the sender immediately upon data packet arrival. However, in most TCP implementations, the receivers utilize the delayed ACK option [15], in which the TCP receiver does not generate an ACK packet for each data packet. In this case, packet interval-based mechanisms do not work properly. In high-speed networks, the effect of the delayed ACK option becomes larger. Therefore, the issue on the delayed

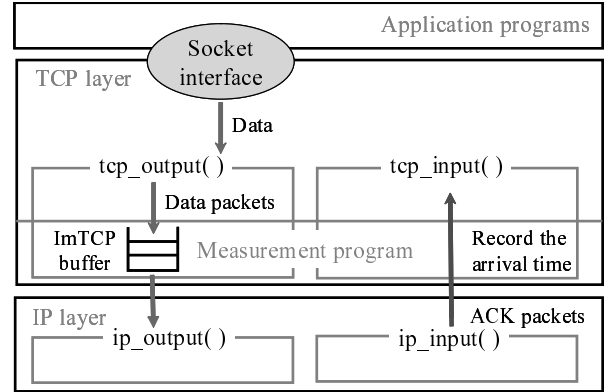


Fig. 3. ImTCP implementation architecture

ACK needs to be solved when building the bandwidth measurement algorithms into TCP.

As described in Subsection II-B, the IC mechanism at the receiver endhost also becomes a reason for degrading the measurement accuracy. When the timeout value of the absolute timer at the receiver's NIC is larger than that at the sender's NIC, the inter-arrival times of packets are greatly influenced of IC at the receiver endhost. Therefore, the measurement algorithms need to take care of the effects of IC.

Packet interval-based algorithms can not solve the above problems due to its fundamental characteristics. Therefore, we develop a packet burst-based algorithm in this paper. The proposed mechanism, ICIM, can work properly with these functions at the receiver endhost. The mechanism is described in detail in Subsection III-B.

## III. ALGORITHMS AND IMPLEMENTATIONS OF THE INLINE NETWORK MEASUREMENTS

In this section, we outline two types of bandwidth measurement algorithms based on the concept of inline network measurement. The first is Inline measurement TCP (ImTCP), which is based on the packet intervals. The other is Interrupt Coalescence-aware Inline Measurement (ICIM), which is a packet-burst interval-based mechanism that is intended for bandwidth measurement in high-speed networks. In addition, we present the implementation issues of the two algorithms in the TCP mechanism of FreeBSD.

### A. ImTCP: Packet interval-based algorithm

#### A.1 Algorithm

ImTCP measures the available bandwidth of the network path between sender and receiver hosts. In TCP data transfer, the sender endhost transfers a data packet and the receiver endhost replies to the data packet with an ACK packet. ImTCP measures the available bandwidth using this mechanism. That is, ImTCP adjusts the interval of data packets according to the measurement algorithm and then calculates the available bandwidth by observing the changes in ACK intervals. When a packet loss occurs, ImTCP stops measurement task until the lost packets are successfully retransmitted.

During each measurement, ImTCP uses a search range to find

TABLE III  
PC SPECIFICATIONS OF THE EXPERIMENTAL NETWORK ENVIRONMENT

	Sender	Receiver	Other endhosts
CPU	Intel Pentium 4 3.0 GHz	Intel Pentium 4 3.0 GHz	Intel Pentium 4 3.4 GHz
Memory	1,024 MB	1,024 MB	1,024 MB
Kernel	FreeBSD 4.10	Linux 2.6.15.1	Linux 2.6.15.1

the value of the available bandwidth. This is the concept of limiting the bandwidth measurement range based on statistical information from previous measurement results instead of searching from 0 bps to the upper limit of the physical bandwidth for every measurement. By introducing the search range, ImTCP can avoid sending probe packets at an extremely high rate, which seriously affects other traffic along the network path. ImTCP can also keep the number of probe packets for the measurement quite small. The detailed algorithm of ImTCP can be found in [9].

### A.2 Implementation

When new data is generated at the application, the data is passed to the TCP layer through the socket interface. The data (packet) is passed to the IP layer after TCP protocol processing by the `tcp_output()` function and is injected onto the network. Because the program for inline network measurement must know the current size of the congestion window of TCP, it should be implemented at the bottom of the TCP layer, as shown in Figure 3. When a new TCP data packet is received from the application and is ready to be transmitted, it is stored in an intermediate FIFO buffer before being passed to the IP layer. The stored packets are passed to the `ip_output()` function in the intervals based on the measurement algorithm. For adjustment of the intervals of data packets, ImTCP uses the task scheduling function provided by the kernel system. When using the task scheduling function, the accuracy of the first tick is unreliable, which is pointed out in [15]. In our implementation, we solve the problem by delaying transmitting the head of probing packets for one tick. The method, however, decreases the throughput when the  $HZ$  is small and the length of one tick is large. For example, when  $HZ$  is set 100, which is the default value of FreeBSD kernel system, one tick is equal to 10 msec. This causes the serious influence in the throughput of TCP especially when a TCP connection has a small RTT.

On the other hand, the measurement program should also be implemented in the `tcp_input()` function. An ACK packet that arrives at the IP layer of the sender endhost is passed to the `tcp_input()` function for TCP protocol processing. The measurement program records the time when the ACK packet arrives at the `tcp_input()` function. The measurement program also guesses the current available bandwidth based on the sending time of data packets and the receiving time of ACK packets according to the algorithm described in [9].

### B. ICIM: Packet-burst interval-based algorithm

#### B.1 Algorithm

As we described in Subsection II-B, packet interval-based algorithms do not work properly in high-speed networks. In order to solve the problems, we have proposed a packet-burst interval-

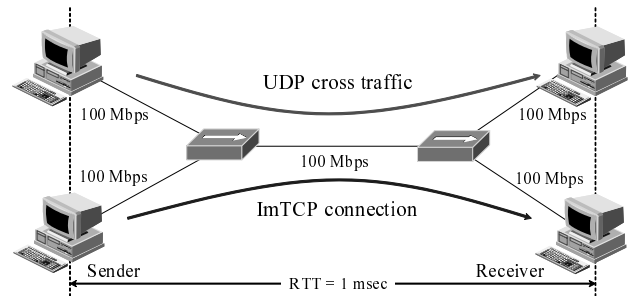


Fig. 4. Low-speed network environment

based algorithm, ICIM. In order to measure the available bandwidth of the network path between sender and receiver hosts ICIM utilizes the burst of data packets in TCP, which is generated by various functions including IC. The sender TCP adjusts the number of packets involved in a burst to estimate the available bandwidth, and checks whether the inter-arrival intervals of the bursts of corresponding ACK packets are increased. When packet losses occur in the network, ICIM interrupts measurement temporarily as is the case with ImTCP. The detailed algorithm of ICIM can be found in [10].

One important advantage of ICIM is that the mechanism can measure the available bandwidth even when the delayed ACK option is enabled in the TCP receiver, as described in Subsection II-C, because ICIM compares the number of transmitting data packets and the corresponding ACK packets in bytes by using the sequence number.

#### B.2 Implementation

When implementing ICIM in the FreeBSD 4.10 kernel system, we inherit the architecture of ImTCP shown in Figure 3, to which some modifications are added. First, we change the packet sending program. The measurement program of ICIM adjusts the number of packets stored in the FIFO buffer, and then passes all of the packets to the `ip_output()` function at one time. The measurement program also records the number of data packets in the burst.

We then modify the program for observing ACK packets. ICIM counts the number of ACK packets when the last ACK packet of the probing burst arrives at the `tcp_input()` function. The measurement program then estimates the available bandwidth by comparing the number of data packets in the burst with that of ACK packets.

Although ICIM can essentially measure the available bandwidth independent of the value of  $HZ$ , ICIM has a limitation with respect to the clock resolution. ICIM measures the available bandwidth by observing the inter-arrival intervals of the

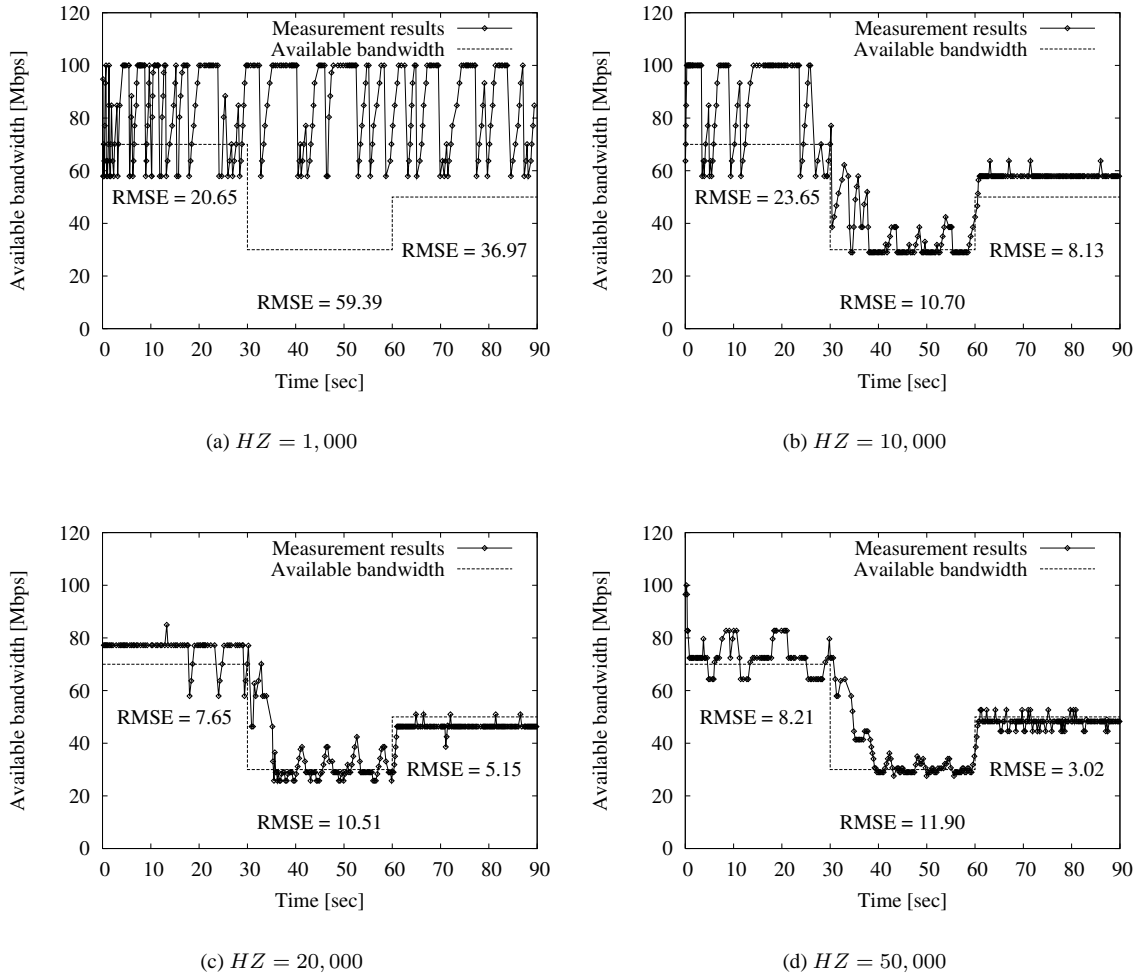


Fig. 5. Measurement results of ImTCP

bursts. Therefore, when the clock resolution is coarser than that of the absolute timer of the network interface card, the mechanisms cannot find the space between two bursts correctly. Consequently,  $HZ$  should be set to be sufficiently large, so as not to be coarser than the absolute timer ( $RxAbsIntDelay$ ). One possible solution to this problem is that the precise clock be used only for judging the space between two bursts, instead of changing the value of  $HZ$ , and the normal system clock is used in other functions. This mechanism, however, is difficult to implement in real systems, so that we do not introduce the mechanism into the implementation of ICIM. In Subsection IV-B, we demonstrate that although the measurement accuracy degrades, ICIM can measure the available bandwidth even when  $HZ$  is set to a small value.

#### IV. PERFORMANCE EVALUATION AND DISCUSSION

In this section, we evaluate the effectiveness of mechanisms based on the concept of inline network measurement on actual networks. Note that we do not compare the performance with any other measurement tools in this section because there are no other tools to measure the available bandwidth based on the

inline measurement concept.

The main contribution of the present paper is to confirm the difficulties when implementing the measurement mechanisms in real systems. The most serious problem is the clock resolution used in the kernel system. Therefore, we mainly test the effects of the measurement accuracy by the clock resolution. In order to evaluate the effects of the clock resolution we make relatively reasonable network conditions (e.g., simple topology and constant bit rate cross traffic) in Subsection IV-A and Subsection IV-B. In addition, we evaluate the effectiveness of ICIM in the little more actual condition in Subsection IV-C. Through experiments and discussions we clarify the limitations of packet interval-based algorithms and the advantages of packet-burst interval-based algorithms in high-speed networks.

##### A. Results of ImTCP in low-speed networks

We first evaluate the effectiveness of ImTCP on a low-speed experimental network. Figure 4 shows the network environment. This network environment consists of two 100Base-TX Ethernet switches, two endhosts that generate cross traffic, an endhost that measures the available bandwidth (Sender), and an endhost that

TABLE IV  
AVERAGE CPU UTILIZATION

$HZ$	ImTCP [%]	TCP Reno [%]
1,000	3.07	11.28
10,000	13.21	12.22
20,000	14.19	14.01
50,000	24.42	22.86

receives packets from Sender (Receiver). All endhosts are connected by a 100-Mbps Ethernet connection. Table III shows the specifications of the experimental network environment. In this experiment, we use Iperf [16] and generate UDP traffic for the cross traffic between the two endhosts. During the experiment, the rate of the cross traffic is changed so that the available bandwidth of the bottleneck link is 70 Mbps from 0 sec to 30 sec, 30 Mbps from 30 sec to 60 sec, and 50 Mbps from 60 sec to 90 sec.

In the experimental network, we change the value of  $HZ$  and observe the relationship between the clock resolution in the kernel system and the accuracy of the measurement results. Figure 5 shows the measurement results of the available bandwidth when the value of  $HZ$  is set to 1,000, 10,000, 20,000, and 50,000, respectively. In each figure, we also show the correct values of available bandwidth and the Root Mean Square Error (RMSE) values. Figure 5(a) shows that ImTCP cannot measure the available bandwidth when the value of  $HZ$  is set to 1,000. In this case the clock resolution is 1 msec and the upper limit of measurable bandwidth becomes 12 Mbps, so that the clock resolution is too coarse to measure in the current network. In Figure 5(b) we can see that the measurement program cannot work properly when the available bandwidth is 70 Mbps (or higher). When  $HZ$  is set to 10,000, the upper limit of the measurable available bandwidth of ImTCP becomes 120 Mbps. However, bandwidth resolution is coarse as the measurement result approaches the upper limit as shown in Table I. This means that  $HZ = 10,000$  is still insufficient in this experimental network. In Figure 5(c) and Figure 5(d) we can see that the measurement results in the case when  $HZ$  is set to 20,000 are more accurate slightly than that in the case when  $HZ$  is set to 50,000. One possible reason is that as  $HZ$  becomes large, the overhead for the kernel system to make interrupts becomes a large task. The overhead affects the accuracy of the length of one tick (one tick is equal to  $\frac{1}{HZ}$ ), which degrades the measurement accuracy.

We then check the CPU load during the experiments. Table IV shows the average CPU utilization when we use ImTCP or the original TCP Reno for the data transfer. These results show that ImTCP can be realized without a heavy load on the CPU. Although we omit the detailed results due to space limitation, most of the load on the CPU are caused by the transmission of data packets. Table IV also shows that the average CPU utilization of ImTCP is much smaller than that of TCP Reno when  $HZ$  is set to 1,000. When  $HZ$  is set to 1,000, the clock resolution becomes 1 msec. Therefore, the intervals of probe packets becomes more than 1 msec when ImTCP adjusts the intervals of probe packets by using the kernel scheduling, which degrades the data transmission throughput. Consequently, the value of  $HZ$  is required to be large not to degrade the throughput when using ImTCP. By

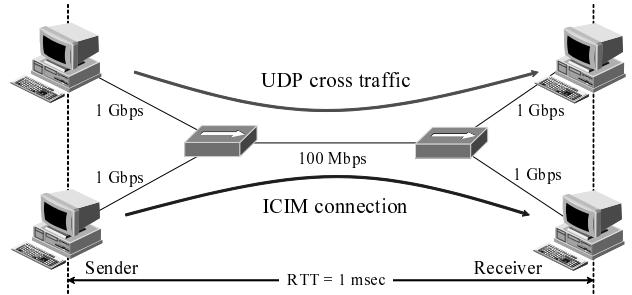


Fig. 6. Gigabit network environment

considering the measurement accuracy, the CPU overhead, and the throughput, we conclude that  $HZ = 20,000$  is a good choice in this environment.

In Figure 5 we can see that when the available bandwidth suddenly decreases, the measurement results slowly follow the change. When the available bandwidth decreases, packet losses occur and the measurement trials are interrupted. Therefore, ImTCP cannot obtain the measurement results immediately after decreasing the available bandwidth, and needs some time to reflect the change in the measurement results.

### B. Results of ICIM in gigabit networks

We next evaluate the effectiveness of the inline measurement algorithm on the gigabit experimental network. Note that the fundamental characteristics of ICIM are clarified through the simulation experiments in [10]. Here, we demonstrate that ICIM can measure the available bandwidth in the high-speed network with coarse clock resolution.

Although ImTCP requires  $HZ$  to be larger than 100,000 for measuring the gigabit network bandwidth, ImTCP cannot work properly in such setting because the CPU load becomes too heavy. ICIM, on the other hand, cannot measure the available bandwidth on the low-speed networks because the intervals of packets are too large and packet-bursts are not generated without IC mechanism at the NIC. Therefore, the performance of two mechanisms cannot be evaluated simply. In the current subsection, we clarify the advantage of ICIM to ImTCP on the gigabit networks.

Figure 6 shows the network environment of the current experiment. This network environment consists of two 1000Base-T gigabit Ethernet switches, three endhosts that generate cross traffic, an endhost that measures the available bandwidth (Sender), and an endhost that receives packets from Sender (Receiver). All endhosts are connected by a 1-Gbps Ethernet connection. The Intel PRO/1000 adapter [17] is used as the network interface card of Sender and Receiver, which supports the Interrupt Coalescence (IC), and the  $RxAbsIntDelay$  parameter of IC in both network interface cards is set to 128, which corresponds to  $131.2 \mu\text{sec}$  according to [17]. The specifications of the current network environment are the same as those in the Table III. In the current experiment, we inject the UDP traffic as cross traffic in the experimental network by using Iperf [16]. During the experiment, we change the rate of cross traffic so that the available bandwidth is 500 Mbps from 50 sec to 100 sec and 200 Mbps from 100 sec to 150 sec. Although we omit the detailed results

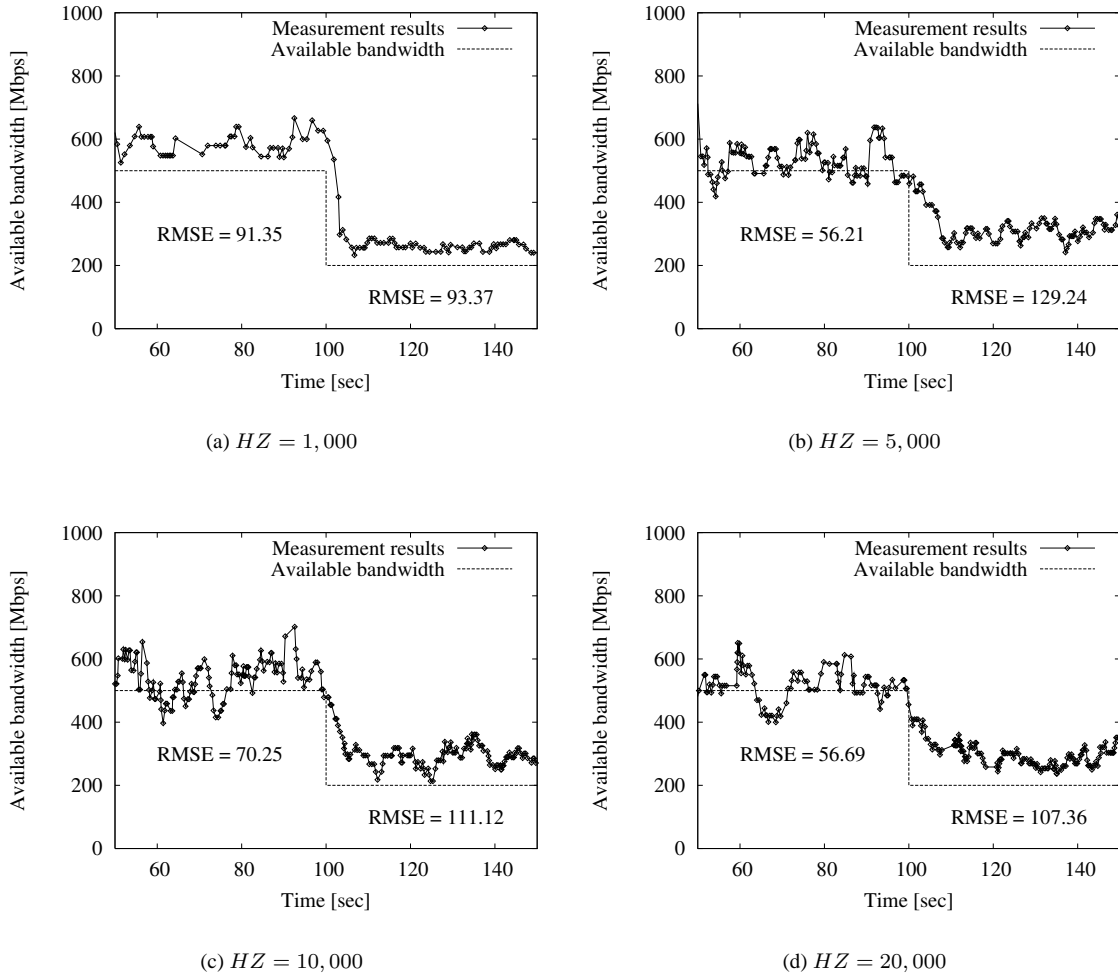


Fig. 7. Measurement results of ICIM

due to space limitation, ICIM can be realized without a heavy load on the CPU.

Figure 7 shows the measurement results of the available bandwidth when the value of  $HZ$  is set to 1,000, 5,000, 10,000, and 20,000, respectively. In each figure we also show the correct values of the available bandwidth and RMSE values. In these figures we can see that ICIM can measure the available bandwidth independent of the value of  $HZ$  compared with the results of ImTCP shown in Figure 5. This result apparently explains the advantage of the packet-burst interval-based algorithm in high-speed networks. Recall from Table I that the packet interval-based approaches require  $HZ$  to be larger than 100,000 for measuring the gigabit network bandwidth.

In addition, We can see the results from Figure 7(a) that although the measurement results tend to overestimate the available bandwidth, ICIM can be performed even when  $HZ$  is set to 1,000. In this case, the clock resolution in the kernel system becomes  $1,000 \mu\text{sec}$ , while  $RxAbsIntDelay$  is set to  $131.2 \mu\text{sec}$ , and the ICIM algorithm does not work well. ICIM can measure the available bandwidth in such case because when the clock resolution is too coarse to find the space between the bursts, ICIM

adjusts the number of data packets in one tick (tick means  $\frac{1}{HZ}$  second) and estimates the available bandwidth by observing the number of ACK packets in the tick. This behavior is similar to Bprobe [18], rather than the mechanism of Pathload [5]. Therefore, the measurement result in such a case tends to be overestimated [19].

Figure 7(d) shows that the accuracy of the measurement results is low when the available bandwidth is 200 Mbps. Packet-burst interval-based algorithms, measure the available bandwidth by adjusting the number of data packets contained in the probing burst. However, the number of packets in the burst decreases as the available bandwidth degrades. Therefore, adjusting the number of packets in the bursts becomes difficult as the available bandwidth degrades, and so the measurement accuracy degrades.

### C. Results of TCP cross traffic

We finally evaluate the measurement accuracy of ICIM for the case in which TCP traffic exists as the cross traffic. Note that we have already evaluated the measurement accuracy of ImTCP with TCP cross traffic on the low-speed network in [20]. There-

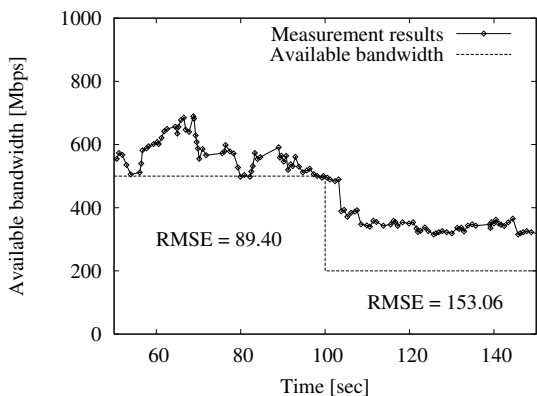


Fig. 8. Measurement results of ICIM (case of TCP cross traffic)

fore, we confirm the performance of ICIM on the gigabit network in the present paper. We use the network and machine environment shown in Figure 6 and Table III. The value of  $HZ$  at the sender endhost (Sender) is set to 20,000, and  $RxAbsIntDelay$  is set to 128. In the experiment, we limit the maximum data transmission rate of each of a cross traffic TCP connection to 100 Mbps by setting the receive endhost. During the experiment, the number of cross traffic TCP connections is changed so that the available bandwidth of the bottleneck link is 500 Mbps from 50 sec to 100 sec and 200 Mbps from 100 sec to 150 sec.

Figure 8 shows the measurement results of the available bandwidth. In this figure we also plot the correct values of the available bandwidth and RMSE values. Although ICIM can measure the available bandwidth in this condition, the measurement accuracy is less than the case of UDP cross traffic as shown in Figure 7. This is caused by the bursty nature of TCP traffic co-existing in the network. When competing traffic is bursty, the intervals of ICIM probe packets are disturbed by the bursty traffic and the measurement accuracy is degraded. We will investigate the effectiveness of ICIM in detail on various networks (e.g., the Internet) and consider the solution against the bursty cross traffic in future studies.

## V. CONCLUSIONS

In the present paper, we clarified the difficulties when implementing the measurement algorithms in real systems and showed the current solutions. We also implemented the proposed algorithms and confirmed the effectiveness of our concept, inline network measurement, on actual networks. Through experiments using ImTCP, we obtained some guidelines for setting kernel parameters when implementing packet interval-based mechanisms in a kernel system. In addition, ICIM experiments confirmed that the packet-burst interval-based mechanism could work well in high-speed networks. Furthermore, we demonstrated that ICIM could measure the available bandwidth even when the clock resolution is coarse. The discussions in the present paper clarified the limitation and the adaptability to high-speed networks of packet interval-based and packet-burst interval-based mechanisms. The source codes of ImTCP and ICIM can be found at our web site: <http://www.anarg.jp/imtcp/>.

In future studies, we will evaluate the effectiveness of ICIM

in detail on various networks (e.g., the Internet and the more high-speed networks). Addition to this, we will propose mechanisms for measuring other bandwidth information based on the proposed inline network concept and will evaluate these mechanisms through simulation experiments and in actual networks.

## REFERENCES

- [1] R. Prasad, M. Murray, C. Dovrolis, and K. C. Claffy, "Bandwidth estimation: Metrics, measurement techniques, and tools," *IEEE Network*, vol. 17, pp. 27–35, Dec. 2003.
- [2] T. Iguchi, G. Hasegawa, and M. Murata, "A new congestion control mechanism of TCP with inline network measurement," in *Proceedings of International Conference on Information Networking (ICOIN 2005)*, Jan. 2005.
- [3] T. Tsugawa, G. Hasegawa, and M. Murata, "Background TCP data transfer with inline network measurement," in *Proceedings of Asia-Pacific Conference on Communications (APCC 2005)*, Oct. 2005.
- [4] B. Melander, M. Bjorkman, and P. Gunningberg, "A new end-to-end probing and analysis method for estimating bandwidth bottlenecks," in *Proceedings of IEEE GLOBECOM 2000*, Nov. 2000.
- [5] M. Jain and C. Dovrolis, "End-to-end available bandwidth: Measurement methodology, dynamics, and relation with TCP throughput," in *Proceedings of ACM SIGCOMM 2002*, Aug. 2002.
- [6] V. J. Ribeiro, R. H. Riedi, R. G. Baraniuk, J. Navratil, and L. Cottrell, "pathChirp: Efficient available bandwidth estimation for network paths," in *Proceedings of Passive and Active Measurement Workshop (PAM 2003)*, Apr. 2003.
- [7] J. Navratil and R. L. Cottrell, "ABWE: A practical approach to available bandwidth estimation," in *Proceedings of Passive and Active Measurement Workshop (PAM 2003)*, Apr. 2003.
- [8] A. Shiram, M. Murray, Y. Hyun, N. Brownlee, A. Broido, M. Fomenkov, and K. C. Claffy, "Comparison of public end-to-end bandwidth estimation tools on high-speed links," in *Proceedings of Passive and Active Measurement Workshop (PAM 2005)*, Mar. 2005.
- [9] C. L. T. Man, G. Hasegawa, and M. Murata, "Available bandwidth measurement via TCP connection," in *Proceedings of IFIP/IEEE Workshop on End-to-End Monitoring Techniques and Services (E2EMON 2004)*, Oct. 2004.
- [10] C. L. T. Man, G. Hasegawa, and M. Murata, "ICIM: An inline network measurement mechanism for high-speed networks," in *Proceedings of IEEE/IFIP Workshop on End-to-End Monitoring Techniques and Services (E2EMON 2006)*, Apr. 2006.
- [11] R. Prasad, M. Jain, and C. Dovrolis, "Effects of interrupt coalescence on network measurements," in *Proceedings of Passive and Active Measurement Workshop (PAM 2004)*, Apr. 2004.
- [12] Intel, "Interrupt moderation using intel gigabit ethernet controllers." available from <http://www.intel.com/design/network/applnotes/ap450.pdf>.
- [13] Sysconnect, "SK-NET GE Gigabit Ethernet Server Adapter." available from [http://www.sysconnect.com/sysconnect/technology/SK-NET\\_GE.PDF](http://www.sysconnect.com/sysconnect/technology/SK-NET_GE.PDF).
- [14] G. Jin and B. L. Tierney, "System capability effect on algorithms for network bandwidth measurement," in *Proceedings of Internet Measurement Conference (IMC 2003)*, Oct. 2003.
- [15] W. R. Stevens, *TCP/IP Illustrated, Volume 1: The Protocols*. Reading, Massachusetts: Addison-Wesley, 1994.
- [16] NARANR, "NARANR/DAST: Iperf 1.7.0 - The TCP/UDP bandwidth measurement tool." available from <http://dast.nlanr.net/projects/Iperf/>.
- [17] Intel(R) PRO/1000 Adapter, "README file." available from <http://support.intel.co.jp/jp/support/network/adapter/1000/linux.readme.htm>.
- [18] R. L. Carter and M. E. Crovella, "Measuring bottleneck link speed in packet-switched networks," *International Journal on Performance Evaluation*, vol. 27–28, pp. 297–318, Oct. 1996.
- [19] C. D. P. Ramanathan and D. Moore, "What do packet dispersion techniques measure?," in *Proceedings of IEEE INFOCOM 2001*, Apr. 2001.
- [20] T. Tsugawa, G. Hasegawa, and M. Murata, "Implementation and evaluation of an inline network measurement algorithm and its application to TCP-based service," in *Proceedings of IEEE/IFIP Workshop on End-to-End Monitoring Techniques and Services (E2EMON 2006)*, Apr. 2006.