Integrated Computer-Aided Engineering 9 (2002) 333–344
IOS Press

Java-based tools for accurate bandwidth measurement of Digital Subscriber Line networks[1]

Liang Cheng[2] and Ivan Marsic*
Department of Electrical and Computer Engineering and Center for Advanced Information Processing (*CAIP*), *Rutgers, The State University of New Jersey, Piscataway, NJ 08854-8088, USA*
Tel.: +1 732 445 4208; Fax: +1 732 445 4775; E-mail: {chengl, marsic}@caip.rutgers.edu

Abstract. This paper presents the design and implementation of Java-based tools for accurate bandwidth measurement for xDSL (Digital Subscriber Line) networks, including the ADSL (Asymmetric DSL) case with different upstream and downstream bandwidths. First, a multi-packet bandwidth measurement technique is presented, based on the integration of packet-pair and filtering techniques. Considering the importance of a traffic generator to accurate measurement, a smooth and stable traffic generator is studied. We also present a stepwise algorithm designed to minimize the effect of ATM traffic shaping on the bandwidth measurements in xDSL deployments. Implementation details of the components of the Java-applet- and Java-application-based tools are described. Finally, evaluation experiments in both a laboratory testbed and a commercial ADSL network demonstrate that these tools achieve accurate bandwidth measurement in xDSL networks, and offer better performance than other popular tools such as pathchar and clink in terms of accuracy and usability.

1. Introduction

1.1. xDSL Networks

Digital Subscriber Line (DSL) is becoming a popular technology for the networking needs of home-users, small or midsize business as well as corporate telecommuters. xDSL networks refer to different variations of DSL, such as ADSL (Asymmetric DSL), SDSL (Symmetric DSL), HDSL (High bit-rate DSL), RADSL (Rate Adaptive DSL), etc. Not only does xDSL provide fast connectivity, but it can also utilize an existing copper phone line and handle voice and data simultaneously over the same line. In addition, the data part of the line has an always-on connection.

Figure 1 shows the typical deployment architecture of ADSL networks. The customer premises equipment is an ATU-R (ADSL Termination Unit – Remote) or an ADSL modem. An ATU-C (ADSL Termination Unit – Central Office) is located at a Central Office end of an ADSL service provider. The ATU-C terminates multiple subscriber loops and connects the ATU-R to the Internet through a Gateway Router. Multiple ATU-Cs in ADSL network are grouped into a DSLAM (DSL Access Module or Multiplexer) unit that terminates the multiplexed traffic into an ATM switch. The ADSL networks enable receiving data at rates up to 6.1 Mbps (of a theoretical 8.448 Mbps). For example, individual ADSL connections will provide from 512 Kbps to 1.544 Mbps downstream and about 128 Kbps upstream. In general, xDSL networks have architecture similar to the one shown in Fig. 1.

1.2. Need for accurate bandwidth measurement

Here we differentiate link bandwidth and available bandwidth: link bandwidth refers to the bottleneck-link

*Corresponding author.
[1] An earlies version of this paper appeared in [4].
[2] Present address: Department of Computer Science and Engineering, Lehigh University, 19 Memorial Drive West, Bethlehem, PA 18015, USA.

ISSN 1069-2509/02/$8.00

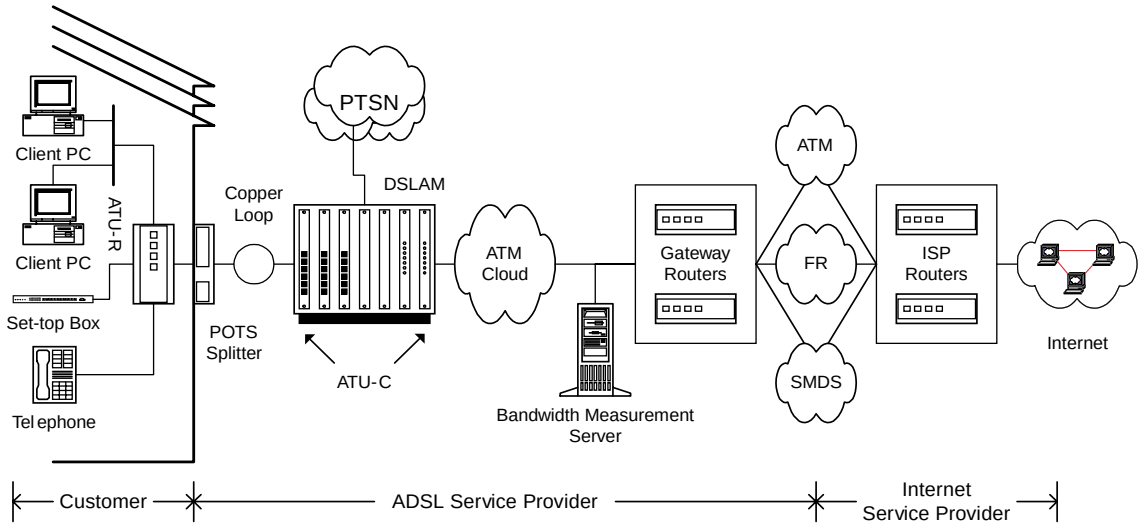

Fig. 1. Deployment architecture of ADSL networks.

bandwidth or the maximum capability of a link (or a data path) to transmit data in bits-per-second, which is always larger than or equal to the available bandwidth of the link/path considering the multiplexing nature of data networking. In this paper we study accurate measurement of the link bandwidth of an xDSL link; that is, the data path between the ATU-R and Gateway Router in Fig. 1.

In xDSL networks, accurate bandwidth measurement is useful for network management, such as for isolating line faults and verifying guaranteed QoS (Quality of Service) specifications, where the ISP (Internet Service Provider) and xDSL providers may both be involved. In the case of ADSL networks, according to Fig. 1, the deployment and maintenance of the links between the ATU-R and the Gateway Router is the responsibility of the phone company. For this reason, there is a need to measure the actual physical line speed of the ADSL link. Moreover, accurate bandwidth measurement is essential for better service provisioning. Because xDSL networks can support a random mix of services with different bit rates and traffic requirements on a per-customer basis, both customers and the service providers can use the bandwidth measurement tools to verify service quality.

Accurate bandwidth measurement is also needed for traffic engineering and xDSL network planning for traffic load balancing. By accurately measuring link bandwidth, bottleneck links can be identified so that xDSL service providers can increase the bandwidth of the bottleneck links to enhance the overall performance of their network services. Accurate bandwidth measurement is also useful for network application planning, configuration, and QoS provisioning in xDSL networks. Since xDSL networks provide fast and always-on connectivity, more and more high-bandwidth-consuming applications are used in such an environment. Most of these applications transmit multimedia data and their performance is sensitive to network bandwidth. Therefore, by accurately measuring link bandwidth in xDSL networks and combining the bandwidth requirement of the applications, the users can decide which applications can be supported by his/her xDSL connection.

1.3. Related work

A number of techniques and tools, using both implicit and explicit approaches, exist for bandwidth measurement. Implicit approaches have been used in applications and protocols, such as NWS (Network Weather Service) [13] and TCP (e.g. throughput measurement in congestion control). These approaches cannot be applied to bandwidth measurement in xDSL networks because: (i) they are not accurate, and, (ii) possible bandwidth asymmetry in xDSL networks is not considered in their designs. For example, during each measurement in NWS, the round-trip time of a single-word packet in a TCP connection is measured and the resulting value is divided by two to yield an approximation of the latency or start-up overhead associated with the communication. Obviously, the approximation is not correct in the ADSL's asymmetric case.

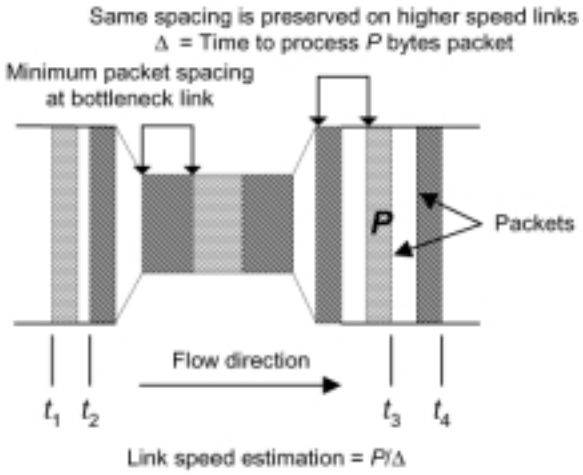

Fig. 2. Packet-pair technique for bandwidth measurement.

Most of the explicit bandwidth measurement techniques and tools can be categorized into two groups [9]. One group is a variant of pathchar [5,7,8] based on the one-packet model. The other group is a variant of packet-pair [2,3,10,12] based on the packet-pair model. In [9], a multi-packet model is presented to unify the one-packet and packet-pair model and a packet-tailgating technique is proposed. Unfortunately, all of the above bandwidth measurement techniques and tools, including the packet-tailgating technique, have low accuracy [9], which is unsatisfactory for our goals.

In our research, we designed an accurate bandwidth measurement scheme for xDSL networks, implemented it as Java applet- and application-based tools, and evaluated it in both laboratory and commercial field experiments. The accuracy is achieved by implementing: a multi-packet bandwidth measurement technique; an original traffic generator for smooth probe-traffic flow; and a novel stepwise bandwidth measurement algorithm. The experiments reported in this paper demonstrate the accuracy of the Java-based bandwidth measurement tools.

The paper is organized as follows. Section 2 presents several bandwidth measurement techniques, including our multi-packet bandwidth measurement technique, our traffic generator with stable and accurate performance, and our stepwise bandwidth measurement algorithm to address the effects of ATM traffic shaping on measurement accuracy. Section 3 describes the Java implementation of the bandwidth measurement techniques. Section 4 presents the experimental evaluation and results. Lastly, Section 5 concludes the paper.

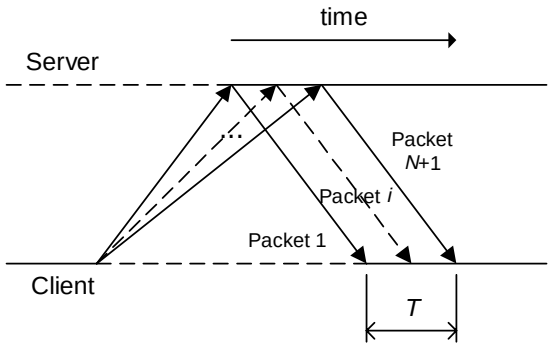

Fig. 3. Upstream scheme of bandwidth measurement.

2. Accurate bandwidth measurement scheme

2.1. General bandwidth measurement techniques

2.1.1. Packet-pair technique

The packet-pair technique is based on the FIFO (First-In-First-Out) network queuing model. As illustrated in Fig. 2, if two packets travel together so that they are queued as a pair at the bottleneck link with no packet intervening, then their inter-packet spacing is proportional to the processing time required for the bottleneck link to transmit the second packet of the pair. Therefore, by recording the inter-packet time, the link bandwidth of the bottleneck link, b, can be computed as:

$$b = P/\Delta$$

where P is the packet size. Note that the inter-packet spacing at the source must be sufficiently small so that:

$$t_2 - t_1 \leqslant P/b = t_4 - t_3 \tag{1}$$

and

$$\Delta = t_4 - t3$$

Various forms of the above-described packet-pair technique have been studied by Bolot [2], Carter and Crovella [3], Paxson [11], and Lai and Baker [9,10]. We generalized this technique to a multi-packet technique that achieves accurate bandwidth measurement in xDSL networks, including ADSL networks.

2.1.2. Multi-packet technique

Two factors could diminish the accuracy of measurement results of the packet-pair technique in terms of correctness of the inter-packet time. One factor is time compression, where the first packet of the packet pair gets delayed while the second one does not. A possible reason is that there exists some other cross traffic along

the measurement path. In this case, the time interval of the packet pair is decreased. Thus the result will be larger than the actual bandwidth. The other factor is time extension, where the second packet of the packet-pair gets delayed and the time interval is increased. Thus, the result will be smaller than the actual value. In addition to cross traffic, packet loss is also a possible reason for this phenomenon. Filtering techniques [9] have been proposed and applied to filter out inaccurate packet-pair time intervals so that the measurement results become more accurate.

As mentioned in Section 1.2, customers and administrators will use our tools to test, monitor, and manage xDSL networks. In such scenarios, the data path from the end-user's computer to the bandwidth measurement server is an ATM-circuit-like path. It is safe to assume that there is no cross traffic, meaning that time compression is not a problem in this case. However, time extension may occur because of ATM traffic shaping in xDSL service and because of a possible packet loss.

In order to reduce the effect of time extension, we propose to use multiple packets to conduct the bandwidth measurement as an integration of the packet-pair technique and the filtering technique. The multi-packet technique does not need to deploy explicit packet filtering in order to achieve accurate measurement because of its intrinsic filtering due to the averaging of results over the multiple packets. The following theorem provides the foundation for the multi-packet technique.

Theorem 1. (*Multi-packet bandwidth measurement*). *Let* $b_{\min} \leqslant b_i (\forall i, 1 \leqslant i \leqslant m)$, *where* m *is the number of links in the path,* b_i *is bandwidth of the* i*th link between node* $(i-1)$ *and node* i, *and* $b_{\min}$ *is the minimum of* b_i $(\forall i, 1 \leqslant i \leqslant m)$. *Then if multiple,* $N+1$, *packets of the same size* $(s_0 = s_1 = \ldots = s_N = s)$ *are sent back-to-back and there is no cross traffic along the path, they will arrive with a difference in time equal to the size of the total packets divided by the smallest bandwidth along the path, i.e.,* $t_{N(m)} - t_{0(m)} = (s \times N)/b_{\min}$, *where* $t_{j(\ell)}$ *is the time when packet* j *fully arrives at node* ℓ.

Proof. Because the packets are sent back-to-back at the sender and there is no cross-traffic along the path, the packets will be queued at the bottleneck link back-to-back. Note that sending packets back-to-back at the sender is not a necessary condition as long as Eq. (1) holds, i.e., the packets are queued back-to-back at the bottleneck link, as illustrated in Fig. 2.

We perform induction on N, the number of packet sent. For $N = 1$,

$$t_{N(m)} - t_{0(m)} = t_{1(m)} - t_{0(m)}$$

According to [9],

$$t_{N(m)} - t_{0(m)} = s/b_{\min}$$

that is:

$$t_{N(m)} - t_{0(m)} = (s \times N)/b_{min}$$

for $N = 1$. This proves the $N = 1$ case.

For $N > 1$, assume the equation holds for $N = n$ case, i.e.,

$$t_{n(m)} - t_{0(m)} = (s \times n)/b_{\min}$$

then for $N = n + 1$ case,

$$\begin{aligned} & t_{n+1(m)} - t_{0(m)} \\ &= (t_{n+1(m)} - t_{n(m)}) + (t_{n(m)} - t_{0(m)}) \\ &= (t_{n+1(m)} - t_{n(m)}) + (s \times n)/b_{\min} \end{aligned}$$

According to [9], since there is no cross traffic,

$$t_{n+1(m)} - t_{n(m)} = t_{1(m)} - t_{0(m)} = s/b_{\min}$$

therefore,

$$t_{n+1(m)} - t_{0(m)} = (s \times (n+1))/b_{\min}$$

This proves the $N = n + 1$ case. □

In our implementation of the bandwidth measurement tool, users can configure the number of bandwidth measurements. The maximum value is reported as the result, to handle the inaccuracy caused by time extension.

2.2. Bandwidth measurement in ADSL networks

The bandwidth measurement scheme in ADSL networks follows the client-server computing paradigm. The client is the computer at the customer site and the server is the Bandwidth Measurement Server, which is shown in Fig. 1. The asymmetric nature of the ADSL networks makes it necessary to have different bandwidth-measurement schemes for upstream and downstream cases.

2.2.1. Upstream scheme

A fixed number, $N + 1$, of UDP packets of uniform size, P bytes, are sent from the client at a rate slightly higher (about 10%) than the nominal bottleneck bandwidth of the ADSL network to saturate the bottleneck link. A server process on the Bandwidth Measurement Server (see Fig. 1) echoes back the packets as they arrive at the server end. Figure 3 illustrates this scheme. The time difference, T, in milliseconds, between the arrival of the first packet at the client end and the arrival of the last packet at the client end is measured. Then the upstream bandwidth, b_1, is computed as:

$$b_1 = N \times P \times 8/T \text{ Kbps} \tag{2}$$

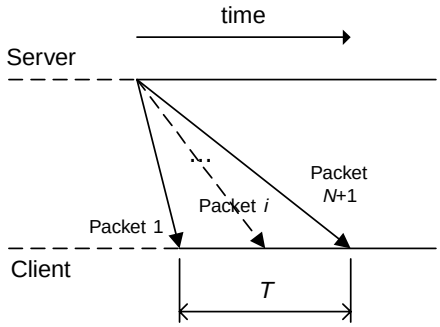

Fig. 4. Downstream scheme of bandwidth measurement.

2.2.2. Downstream scheme

A traffic generator at the Bandwidth Measurement Server generates a downstream traffic. A receiving process at the client measures the arrival time of the packets. Because of the nature of ADSL that the upstream bandwidth is smaller than the downstream bandwidth, the client does not echo the probe packets back. Instead, the client computes the downstream bandwidth using the same Eq. (2) as for the upstream case. Figure 4 illustrates the downstream scheme.

2.3. Traffic generator with stable performance

According to our experience, the performance of the traffic generator in generating probe traffic affects the accuracy of bandwidth measurement in xDSL networks because of ATM traffic shaping described in Section 2.4.1 below. We implemented a stable traffic generator generating smooth data traffic instead of using services such as chargen and/or ping because: (i) currently there is no Java API to access and/or control these services, and, (ii) the data traffic generated by these services is not always sent at a uniform rate with uniform spacing due to the different scheduling mechanisms of operating systems, which may result in bursts in traffic generation.

A simple method of generating packets at a known rate employs Algorithm 1 shown below. Although it seems correct in theory, it does not give the stable and smooth behavior as expected. For example, when the call to `sleep()` returns and the process is scheduled out, the packet is sent only when the operating system reschedules the process rather than sending a packet immediately. Moreover, delay computed in Algorithm 1 may not be an integer number while `sleep()` generally only accepts integers as input values.

To overcome the randomness caused by the scheduling mechanisms of operating systems and coarse granularity of timers, an alternative way of generating traffic at the desired rate was devised as shown in Algorithm 2. The algorithm achieves the expected behavior, i.e., it stably generates a smooth flow of traffic at the expected rate. The performance comparison between the new traffic generator and the simple one is shown in Fig. 5.

Algorithm 1. Pseudo-code for the simple traffic generator.

```
Initialize:
// rate in Kbps, packet size in byte
// delay in millisecond
delay = 8 x packet size x 1000 / rate;
for (i = 0; i < # of packets; i++)
{
     send_packet();
     sleep(delay); // in milliseconds
}
```

2.4. Stepwise bandwidth measurement algorithm

2.4.1. Effect of ATM traffic shaping

According to Fig. 1, a DSLAM unit terminates the multiplexed traffic from multiple xDSL lines into an ATM switch and to the Internet. Since xDSL deployments are typically done over ATM to provide guaranteed QoS, the effect of ATM traffic shaping on bandwidth measurement in xDSL networks should be taken into consideration.

The ATM Forum [1] specifies five service categories in ATM, i.e., Unspecified Bit Rate (UBR), Available Bit Rate (ABR), Constant Bit Rate (CBR), Real-Time Variable Bit Rate (rt-VBR), and Non-Real-Time Variable Bit Rate (nrt-VBR) services. In some service categories, such as in the CBR service class setting, an increase in the data rate beyond the provisioned bandwidth results in a heavy packet loss. This is caused by the ATM traffic shaping. Its effect on bandwidth measurement is that it distorts the results since the accuracy of the packet-pair technique depends on the success of receiving back-to-back packet pairs. To minimize the effect of ATM traffic shaping, we designed a stepwise bandwidth measurement algorithm to make the probe traffic have a sending rate closely matching the link bandwidth. The stable traffic generator is implemented as described above to make the probe traffic less bursty.

Algorithm 2. Pseudo-code for the new traffic generator.

```
Initialize:
// maxNumOfPackets: the maximum number of packets allowed to be sent
// packetSize: the size of packet in bits, and rate is the expected traffic rate
// pps: the number of packets to be sent per sec

pps = rate / packetSize;
fraction = 0;
stopSleep = getTimeMillis(); // get the current time in milliseconds
startSleep = stopSleep;
i = 0; // number of packets have been sent

do
{
   sleep(1);      // sleep for 1 ms
   stopSleep = getTimeMillis();

   packetCount = (int) (fraction + pps * (stopSleep – startSleep) / 1000);
   fraction = (fraction + pps * (stopSleep – startSleep) / 1000) - packetCount;

   startSleep = stopSleep;
   upLimit = i + packetCount;
   if (upLimit > maxNumOfPackets)
       upLimit = maxNumOfPackets;
   for (; i < upLimit; i++)
       sendPacket();
} while(i < maxNumOfPackets);
```

2.4.2. Stepwise bandwidth measurement algorithm

The stepwise algorithm shown as Algorithm 3 below consists of at least two steps. It can be used for both the downstream and upstream measurements in the ADSL case. During the first step, a small number of packet-pair sequences, e.g., 5, are sent back-to-back from the client to the server. This small number of packets will not suffer significant packet loss, as ATM switches are capable of handling such small bursts. The computed result at the client in the first step is used as the trial bandwidth of the xDSL link.

The next step(s) assumes that the accurate xDSL bandwidth is close to the trial result obtained from the first step. In fact, only packet loss impacts the result. If the packet loss in the first step is minimal or zero, which has also been taken into account by our implementation, then the result of the first trial is accurate. In the next step(s), probe packets are sent at a rate slightly higher than the nominal bandwidth of the xDSL link. More packet pairs are used to make the

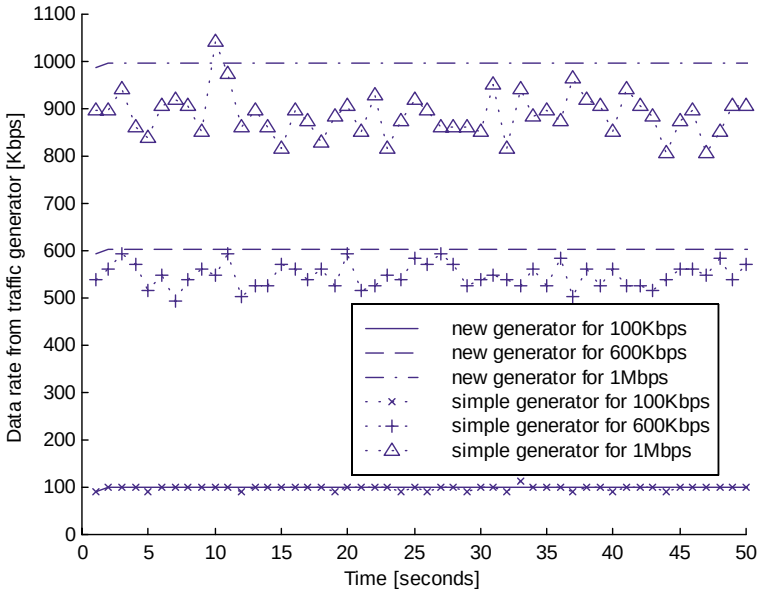

Fig. 5. Performance comparison of traffic generators.

measurement to ensure that the results are convergent and finally consistent.

In the field experiments descried in Section 4.2 below, to demonstrate the necessity for the stepwise algorithm, we performed bandwidth measurements using the single-step algorithm with the stepwise algorithm disabled. In these experiments, the traffic generator's sending rate at a client is explicitly pre-set using different values, and the client sends the sustained traffic at that rate. The Bandwidth Measurement Server records the measurement results. The results show that the measurement inaccuracy of the single-step algorithm ranges up to 25%, which is significantly larger than that of stepwise algorithm, 5%. This demonstrates the stepwise algorithm's prevention of packet loss caused by ATM traffic shaping in xDSL networks and improvement of the accuracy of bandwidth measurement.

3. Java implementation

Our bandwidth measurement tools include both server side and client side implementation. The client side of the tools is implemented as a Java application or a Java applet, which can be loaded in a Web browser. A Java applet is used for client side implementation so that (i) the client software does not need to be physically installed on the user's computer and it can be downloaded and run from the user's web browser, and (ii) upgrades to the client software can be easily facilitated by this feature. On the server side, the Java programming language is also used for implementation because of its multi-platform inter-operability. Figure 6 illustrates the sequence of interactions between the user, the client software, and the server software. The user logs in, configures the tool, and initiates the upstream and downstream bandwidth measurement of xDSL links.

3.1. Components of client software

3.1.1. Authentication

The client software asks the user for user name and password and then validates the information from the server to check whether the user is allowed to use this tool. Figure 7 shows the login interface.

3.1.2. Tool configuration

The client software presents an interface for the user to configure parameters for the bandwidth measurement, such as probe packet size, number of probe packets per measurement, number of stop packets, etc.

3.1.3. Traffic generator

Since probe packets need to be generated and sent to the server in the upstream bandwidth measurement, the traffic generator described in Section 2.3 is implemented using UDP.

Algorithm 3. Stepwise bandwidth measurement algorithm.

```
get bandwidth b_0 by trial bandwidth measurement; // initial estimation of nominal link bandwidth;

do
{
    // probe packets sent at a rate slightly higher than b_0
    get bandwidth b_i by bandwidth measurement;
} while (sequence b_i converges)
```

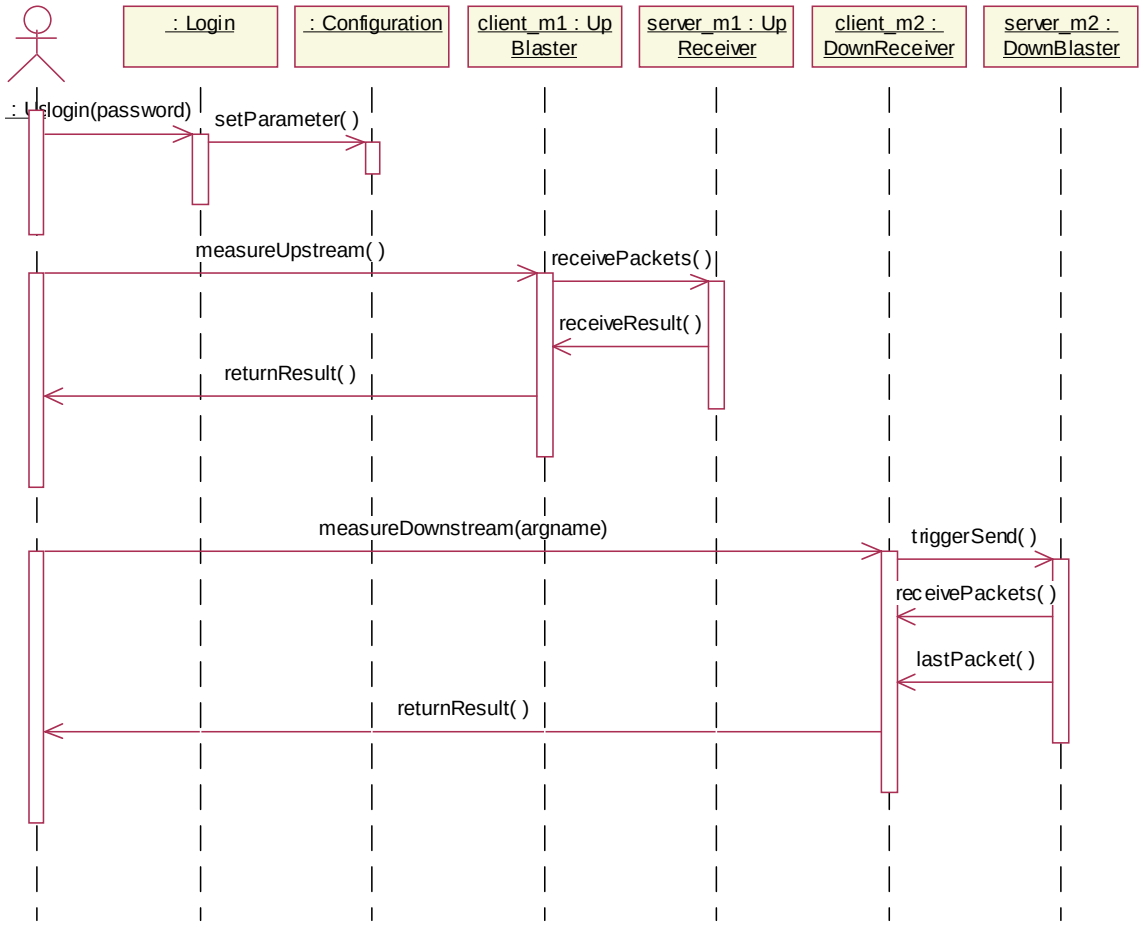

Fig. 6. UML sequence diagram for interactions between the user and the bandwidth measurement system.

3.1.4. Measurement

The measurement component implements the stepwise algorithm for upstream bandwidth measurement. The measurement results are displayed in the user interface as shown in Fig. 8. The results are also reported back to the server for further data analysis and history log maintenance.

3.2. Components of server software

3.2.1. Authorization

The server software receives user name and password from the client software, checks this informa-

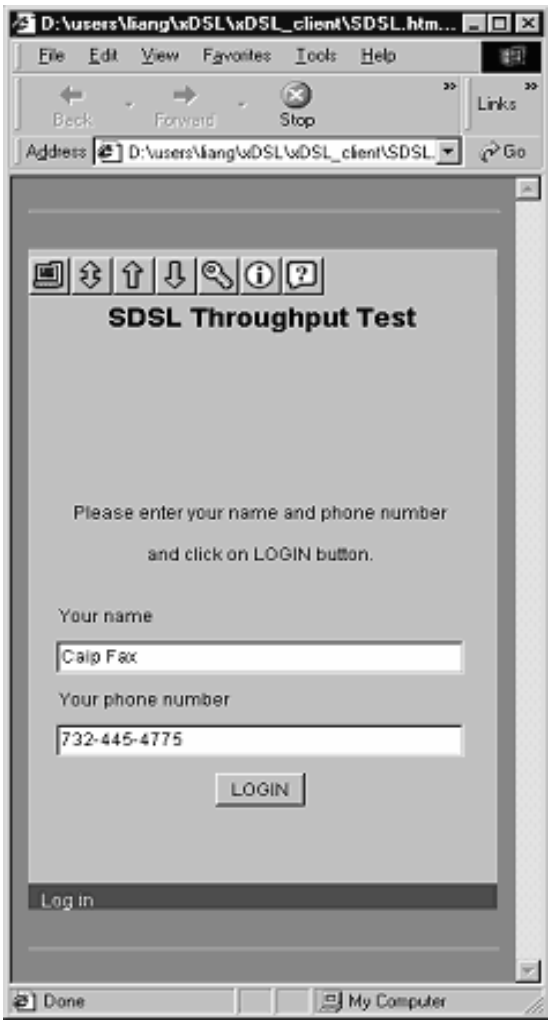

Fig. 7. Login interface of the bandwidth measurement tool.

tion against the user database and, if confirmed valid, authorizes the user to start a measurement session.

3.2.2. *Traffic generation*

Since probe packets need to be generated and sent to the client in the downstream bandwidth measurement, a stable traffic generator needs to be implemented. Again, UDP packets are used instead of ICMP packets because of Java's lack of support for raw sockets. The generator described in Section 2.3 is implemented. It is activated by a trigger packet from the client. The last packet from the server is marked as the "stop packet" so the client knows when the stream of packets from the server ends. Since UDP is an unreliable protocol, the stop packet can be lost. For this reason we also set a timer when the first packet is received (default value

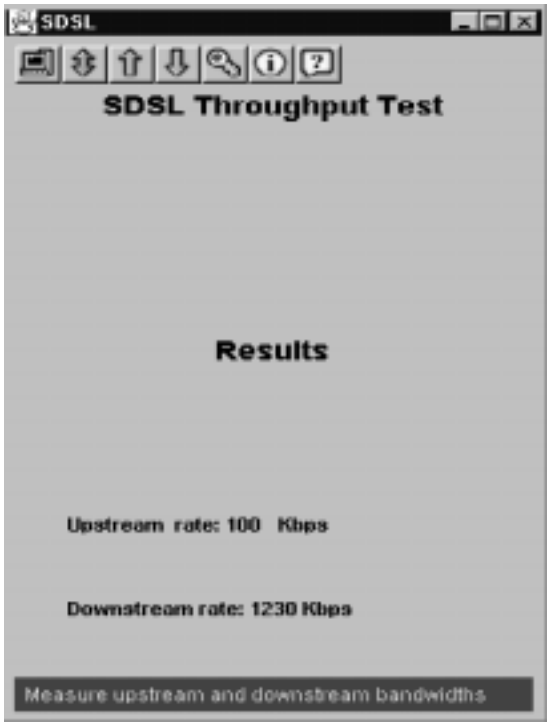

Fig. 8. Bandwidth measurement result panel.

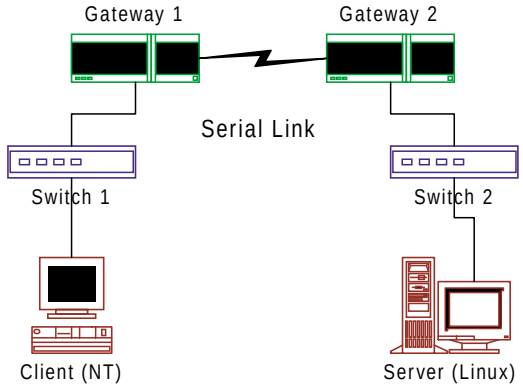

Fig. 9. Laboratory testbed topology.

is 5 sec), which will be used as substitute for the time of the stop packet arrival.

3.2.3. *Echo server*

The UDP Echo Server for upstream measurements waits for Echo Request packets on a specific server port and echoes back the received packet back to the client. Essentially it acts as a rate generator generating packets at a rate closely matching the upstream bandwidth. Since the standard port 7 service of Unix servers discards Echo Request packets beyond a certain rate as a

Table 1
Laboratory experiment results in case of a 10 Mbps link

Measurement tool	Version	Mean value	Standard deviation	Number of probe packets	Mean time per measurement
Rutgers tool	1.0	9.95 Mbps	197 Kbps	450[1]	0.09 min
pathchar [5]	Alpha; April 21, 1997	8.34 Mbps	336 Kbps	$h \times 2880$[2]	7.50 min
Clink [6]	1.0; August 14, 1998	9.26 Mbps	1.10 Mbps	$h \times 1488$	1.33 min[3]

[1]The number of probe packets can be configured. In all experiments reported here, 450 packets have been transmitted for each measurement. This number is statistically significant enough to offset the effect of the system timer inaccuracy.
[2]h is the number of hops between the two endpoints.
[3]This time is variable and reaches 8.50 minutes for smaller HSSI bottleneck link bandwidths.

security measure, this standard Echo server cannot be used in our application. A version of the Echo Server has been coded to handle throughput rates as high as 640Kbps which satisfies the ADSL network Upstream Throughput rates of the commercial ADSL provider we worked with.

3.2.4. Measurement

The measurement component implements the stepwise bandwidth measurement algorithm described in Section 2.4.2.

3.2.5. Logging information for data analysis

The Log Server collects upstream and downstream measurement results and stores them into the database for future data analysis. Logs also enable technicians at the Central Office to immediately verify the results of the test being performed by the customer.

4. Experimental evaluation

4.1. Laboratory testbed

For experiment evaluation, we first tested our bandwidth measurement tool in a laboratory testbed network since its bottleneck link bandwidth could be altered manually. Figure 9 shows the topology of the testbed. The client and the server are in different subnets that are connected by a serial link by introducing a High-Speed Serial Interface (HSSI) between two gateways. The serial link is the bottleneck link. We compared the performance of our measurement tool implemented in Java with those of other popular tools, such as pathchar and clink. The server in the experiments runs on an Intel Pentium II 266 MHz machine with Linux operating system. The test was repeated 100 times for the Rutgers tool and 10 times for both pathchar and clink.

Table 1 shows the experimental results for the case of a 10 Mbps link. Figure 10 compares the performance of our bandwidth measurement tool to that of pathchar and clink [6] on a range of bottleneck link bandwidths. The HSSI clock is only able to accurately set the link speeds up to 4000 Kbps. Consequently there is a gap from 4000 Kbps to 10000 Kbps, which is the link speed for a 10 Mbps Ethernet LAN. The numbers in Table 1 and the figure show that our tool compares favorably with the other tools in terms of measurement accuracy. Both pathchar and clink show significant deterioration of measurement accuracy starting at about 10 Mbps. We did not perform measurements for higher bandwidths, but it is reported in [9] that the accuracy becomes worse with increasing bandwidth.

In addition, our tool runs much faster, consumes much less bandwidth, and is platform independent. Moreover, pathchar and clink use ICMP packets in estimating the link characteristics. The Rutgers tool is implemented as a Java applet and it is thus platform-independent and very easy to use. Unlike pathchar and clink, no root or super-user privileges are required to use our tool.

4.2. Field experiments

We performed field experiments in a commercial ADSL network with the experimental scenario abstracted as in Fig. 1. Three ATM service categories in the ADSL network, CBR, rt-VBR, and UBR, are chosen to evaluate the accuracy of our bandwidth measurement tool. For each category, the upstream and the downstream link bandwidths are measured for several nominal ADSL service rates: 90 Kbps and 680 Kbps in the upstream case, and 640 Kbps, 1.7 Mbps and 7.1 Mbps in the downstream case. Each measurement was repeated 100 times and the mean values and standard deviations were computed. Results show that in all experiments the mean measurement errors, i.e., the ratios of the difference between the mean value and the nominal value to the nominal value, are within 5%. Also the coefficients of variation (COVs), which are the standard deviations divided by the means, are within 5% range. Possible reasons for not achieving 100%

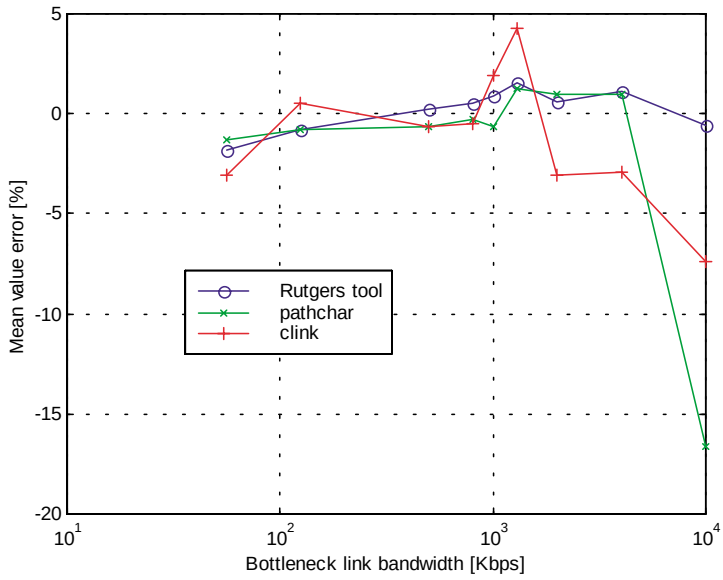

Fig. 10. Comparison of the performance of different tools for bandwidth measurement.

accuracy could be some of those discussed in [9]. Of those, the most significant ones seem to be unreachable nominal values and the timing resolution of the operating system.[1]

5. Conclusion

The growth in deployment of high-bandwidth applications over the Internet has created a need for reliable and fast local loops for both home-user communities and corporate communities, which connect to the Internet through a dial-in ISP. In most cases, a serial modem solution that offers a maximum bandwidth of about 56 Kbps is not sufficient. Therefore, various technologies, such as xDSL, have emerged to achieve high-speed connectivity to the Internet.

This paper presents a design and implementation of Java-based tools for accurate bandwidth measurement for xDSL networks, including the ADSL case. To achieve measurement accuracy, a stepwise algorithm deploying a multi-packet bandwidth measurement technique and a stable traffic generator have been designed and evaluated. Our Java-based implementation has been tested by extensive experimentation both in a laboratory testbed and a commercial ADSL network. The experimental results show that our tool achieves accurate bandwidth measurement in xDSL networks and has better performance than popular tools such as pathchar and clink.

[1] In addition to the timer used in traffic generation, there are also times recorded at the beginning and end of each measurement cycle.

Acknowledgments

The authors are grateful to Senthilkumar Rajasekharan and the research group at the CAIP Center, Rutgers University for helping implement the tools and perform the field measurements. This research is supported by grants from Verizon Communications (formerly Bell Atlantic Corp.), Cisco Systems, Inc., and NSF KDI IIS-98-72995, and by the Rutgers Center for Advanced Information Processing (CAIP).

References

[1] ATM Forum, Online at: http://www.atmforum.com/.
[2] J.C. Bolot, End-to-end packet delay and loss behavior in the Interent, *Proceedings of ACM SIGCOMM*, San Francisco, CA, September 1993, pp. 289–298.
[3] R.L. Carter and M.E. Crovella, Measuring bottleneck link speed in packet-switched networks, Technical report TR-96-006, Department of Computer Science, Boston University, Online at: http://www.cs.bu.edu/faculty/crovella/papers.html, March 1996.
[4] L. Cheng and I. Marsic, Bandwidth measurement in xDSL networks, *Proceedings of the 4th IEEE International Conference on ATM* (*ICATM2001*), Seoul, Korea, April 2001, pp. 222–226.

[5] A. Downey, Using pathchar to estimate Internet link characteristics, *Proceedings of ACM SIGCOMM '99 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*, Cambridge, MA, August/September 1999, pp. 241–250.
[6] A.B. Downey, Clink: a tool for estimating Internet link characteristics, Wellesley College, Online at: http://rocky.wellesley.edu/downey/clink.
[7] V. Jacobson, Pathchar, Online at: ftp://ftp.ee.lbl.gov/pathchar, 1997.
[8] V. Jacobson, Pathchar – a tool to infer characteristics of Internet paths, Presented at the Mathematical Science Research Institute, Online at: ftp://ftp.ee.lbl.gov/pathchar, April 21, 1997.
[9] K. Lai and M. Baker, Measuring link bandwidths using a deterministic model of packet delay, *Proceedings of the ACM SIGCOMM 2000 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*, Stockholm, Sweden, August 2000, pp. 283–294.
[10] K. Lai and M. Baker, Measuring bandwidth, *Proceedings of IEEE INFOCOM 1999*, New York, NY, March 1999, pp. 235–245.
[11] V. Paxson, Measurements and Analysis of End-to-End Internet Dynamics, Ph.D. Dissertation, University of California, Berkeley, April 1997.
[12] V. Paxson, End-to-end Internet packet dynamics, *Proceedings of the ACM SIGCOMM '97 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*, Cannes, France, September 1997, pp. 139–152.
[13] R. Wolski, Dynamically forecasting network performance to support dynamic scheduling using the network weather service, *Proceedings of the 6th IEEE Symposium on High-Performance Distributed Computing*, Los Alamitos, California, August 1997, pp. 316–325.