ANALYSIS OF NETWORK PACKET LOSS USING PASSIVE MEASUREMENT TECHNIQUE AUGUMENTED WITH PARITY-CHECK

T. E. Akhigbe-Mudu, F. T. Ibharalu, O. Folorunso

Department of Computer Science, Federal University of Agriculture, Abeokuta, Nigeria

Abstract: Accurate measurement of packet loss in network traffic has not been successful - the frequently used active monitoring technique using probe packets does not capture all packets loss experienced in traffic network. Passive Measurement Technique (PMT) is attractive because of its efficiency since user traffic is observed without inserting additional test traffic (probe packets) into the network. However, the technique is handicapped by its premature subtraction that may report as lost all packets that have left their source but have not yet reached their destination. This paper proposes a passive packet loss measurement technique with augmented parity check that effectively avoids the problems of either over estimating or under estimating the network performance as a result of the premature subtraction experienced in normal PMT. The effectiveness of the proposed PMT augmented with parity check technique is confirmed through a computer simulation.

Key words: Passive, Measurement, Packet loss and Network.

1. INTRODUCTION

Packet loss measurement is a fundamental problem in network traffic. Since the performance measurement is executed by measuring some metrics of the network, it is important to develop effective measurement techniques. A considerable number of studies have examined packet loss measurement metrics [Has06, J+06], measurement methods [BK11] and described measurement results. These methods have been evaluated only by computer simulations and have not been implemented on real networks. Thus, their feasibility is unclear.

While progress has certainly been made, packet loss and its effects on network performance remain a significant issue for network researchers, system designers and operators. However, the Internet is a constant changing environment, and this requires continued evaluation of important phenomena, such as packet loss [Asa98, BK11].

There are two basic methods for measuring packet loss - Active and Passive measurement techniques, each with its own set of challenges. The first is the use of active probing, which sends probes, either in multicast or unicast [Pap08, PT10], to obtain end – to – end measurements for packet loss. Since most internet routers do not enable multicast, unicast based active probing is more appealing in practice [Pap08]. However, active probing has its own limitations, such as introducing probing overhead and requires collaboration of senders and receivers to collect measurement data [AI10]. Also the discrete nature of active probing limits the resolution of the measurements (a standard problem in any measurement methodology based on sampling). If more frequent probes are sent into the network then resolution will increase, but if the frequency is too high then the probes themselves can skew the results. Hence, passive network measurement technique is proposed, such that packet loss measurement is achieved by monitoring existing network traffic [Asa98], without generating probing traffic. While passive network measurement is attractive, it has limited control over the measurement process and these results in the degradation of the overall network performance. In particular, premature subtraction that may report as lost all packets that have left their source but have not yet reached their destination is imperfect in practice.

In this paper, we propose a Passive Measurement Technique (PMT) augmented with a parity check scheme that accounts for different functional forms of measuring packet loss.

Passive monitors can either be attached to network links or are available from network nodes. The benefit of passive monitoring systems is that they capture many of the important details of local traffic behavior, such as identifying which subsystem within a router has become overloaded [TLS04].

2. PAST EFFORTS ON PACKET LOSS MEASUREMENT TECHNIQUES

Various techniques have been proposed and developed to measure and monitor packet loss, and they can be classified into two broad classes: - Active measurement and Passive measurement techniques [ZT08].

2.1 Active Measurement Technique

Active measurement injects test traffic into the network which may cause congestion, so the network

state may become quite different from that experienced by users. Thus, active measurement may estimate performance inaccurately. In addition, the test traffic of active measurement differs from the traffic of the user applications. This difference in traffic characteristics also leads to inaccurate performance evaluations.

The benefits of active probes are that they can be run from virtually anywhere in the network, and that they give an end-to-end perspective of network behavior. The difficulty is that the discrete nature of active probing technique limits the resolution of the measurements (a standard problem in any measurement methodology based on sampling) [PS10]. [PS10] suggests that the method of [Has06] uses a discrete measurement interval and underestimates the packet loss in a network. [Pap08] proposed active measurement technique (custom measurement technique) and try to infer available bandwidth, The drawback here is that the technique compete with the application for bandwidth, which could again hurt quality of service and has not solve our problem.

The active probe tool is a scheme based on the concept of self-induced congestion. If one sends probe traffic at a rate lower than the available bandwidth along the path, then the arrival rate of probe traffic at the receiver will match their rate at the sender. In contrast, if the probe traffic is sent at a rate higher than the available bandwidth, then queues will build up inside the network and the probe traffic will be delayed. That is, the probes' rate at the receiver will be less than their sending rate [OM05, BS04].

The main contribution of [Has06, AI10] is the development of an analytical technique based on the variance of the inter- packet loss distribution which provides an accurate assessment of how many samples will be required to accurately estimate packet loss probabilities via sampling but this does not solve our problem.

2.2 Passive Measurement Technique

Passive monitoring is a means of tracking the performance and the behavior of packet streams by measuring the user traffic without creating new traffic or modifying existing traffic. It is implemented by incorporating additional intelligence into network devices to enable them identify and record the characteristics and quantity of the packets that flow through them [Has03].

In passive measurement technique, excessively large bandwidth is needed to gather the packet records stored at the sender and receiver for comparison. In the passive measurement system reported in [Pap08] [ZO12], it only addressed the costs of storage and bandwidth but do not handle the problem of premature subtraction which is the problem we addressed in this work. We propose PMT augmented with parity check scheme to address the problem of premature subtraction which causes over estimation or under estimation of the network performance.

2.3 Importance of Network Measurement

The need for measurement of Network performance to support operations, management and planning of networks has increased in recent years. One reason is that IP networks are coming to dominate the way information is brought to users on a worldwide basis. They are no longer limited to transferring data but are also the carrier for multi service communications, and many of the new applications demand high data throughput (bandwidth) with strict performance requirements [Has06]. Hence, without measurements, there will be no objective record or bench mark of how a network behaves. Measurements show whether changes improve or degrade the network's performance, and by how much. When service performance degradation occurs, these are the common questions asked by the network administrator or user;

Why is the application so slow? Why is there loss of data? What can be done to improve performance? Will additional bandwidth help?

Information must be collected by measurements in order to provide answers, to these questions. For example, measurement for capacity planning provides the information for network operators to calculate necessary capacity, in order to avoid congestion in the network. The degradation of performance may cause long packet delay time, jitter or larger packet loss ratio. Network measurement is necessary to monitor the network performance and to analyze and identify any faults.

2.4 The Simple Network Management Protocol

To use networks effectively, it is necessary to have a set of rules by which all the networks should abide. This set of rules called (Protocol). Simple Network Management Protocol (SNMP) is one among the various protocols that defines messages related to network management. SNMP has become the document standardized network management scheme in use today because of its simplicity [Has06]. It is used in the transfer of network management information between two or more network entities or nodes (routers, servers, measurement devices). All the nodes run an application called the SNMP agent. An agent is a software module in a managed device

responsible for maintaining management information and delivering that information to a manager via SNMP. A manager is a software module responsible for managing a part or the entire configuration on behalf of the network management applications and users, and is used for obtaining management information. The machine on which the manager runs is called the management station. The manager and agent communicate with each other using the SNMP protocol. The model is illustrated in figure 1 below.

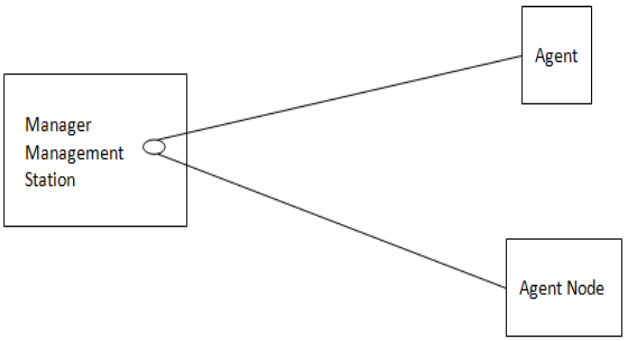

Figure 1. SNMP Network Module

2.5 Causes of Packet Loss

There are two major causes of packet loss in IP networks. One is congestion, where the network routers / switches are temporally sent more packets than their buffers can accommodate. The other is due to link failure, when all the bits currently in transit on that link will be lost. Further packets that are also transmitted down the broken link will be lost until the synchronous digital Hierarchy, (SDH), layer reconfigures a new route around the failed link [Opo11]. The effect of packet loss on the user depends mainly on the following three factors;- Packet loss rate, Packet loss distribution, and Packet size (larger packet) contains more information.
Packets loss due to congestion is a fundamental problem in IP networks. A queue forms in network transmission buffer until the router is able to transmit them on their way. It is considered that buffer overflow takes place when a queue exceeds it buffer limit.

3. THE METHODOLOGY (AUGMENTED TECHNIQUE)

The augmented technique assumes a network path equipped with two passive monitoring sensors at its endpoints, then, measuring packet loss is simply achieved by subtracting the number of packets arrived at the destination from the number of packets that were originally sent.

The PMT technique observes sender – to – receiver packets and receiver – to – sender acknowledgements (ACK) at a measurement point. It seeks to identify whether each packet is delivered or lost from the sender to the receiver. Determining whether a packet is lost is based on packet retransmission mechanism: if a sender successfully delivers a packet to a receiver, then the receiver will reply a corresponding ACK; if the sender does not receive the ACK after a time out, then it deduces that the packet is lost and retransmits the same packet. Thus intuitively, packet retransmissions can be viewed as indicators of packet losses. Since retransmissions can also be triggered by spurious time outs or ACK losses in addition to packet losses, there are also more robust approaches of measuring packet loss. Our proposed PMT augmented with parity check scheme will correctly determine whether a packet is lost by observing the network traffic. We develop a Packet Monitor Algorithm (see 3.1) that monitors the network traffics. In the algorithm, an array of temporary files to store traffic packets is created. The main loop in the algorithm captures packets into temporary files, analyses the packets and display them in a readable format. Example of an output from a run of this algorithm is displayed in table 1.

3.1 Packet Monitor Algorithms

(A) Algorithm for Port Scanning

Step 1: *Variable Declaration*
1.1 Declare variables for storing IP Address and host name and set them to null

Step 2: *Input*
2.1 Enter value of Host name (or IP Address)

Step3: *Scanning*
3.1 Declare variable port = 0
3.2 Declare initial port = value.
3.3 Declare final port = value.
3.4 Check if the port is available between initial port and final port.
3.5 increment port by 1
3.6 Repeat step 3.4 up to final port.

Step4: *Display*
4.1 Display all the active ports in GUI format

(B) Algorithm for packet capturing

Step 1: *Obtaining the list of network interfaces*
1.1 Create a variable array of devices
1.2 Detect network interfaces present in user
1.3 Store the above list in devices variable.

Step 2: *Displaying the list of network interfaces*
2.1 Declare loop counter integer variable i and initialize to 0
2.2 While the value of i is less than the length of the array of devices, do Step 2.3

2.3 Print out the name and description of the captured Network Interface.

Step 3: *Open the network interface.*

3.1 Declare integer variable J and initialize to zero (J=0)

3.2 While J < length of array of devices, Goto Step 3.3 else Goto Step 3.7

3.3 Check if the network interface at **J***th* index number in devices array is selected. If yes goto Step 3.6 else goto Step 3.4

3.4 J=J+1

3.5 Goto Step 3.2

3.6 Open the selected network interface i.e. Network Interface at **J***th* index, then Goto Step 4

3.7 Display that the network interface has not yet been selected by the user.

3.8 Goto ***Step 8***

Step 4: *Capture packets from the network interface*

4.1 Is the menu button of stop capture packet selected? If yes goto Step 3.8 else goto Step 4.2

4.2 Capture the upcoming single packet from the network

4.3 Display the captured packet by going to Step 5

Step 5: *Display the captured packet to the user in proper GUI format.*

5.1 Detect user' menu choice of the format in which captured packet's to be displayed

5.2 Analyze the packet. Display in Hexadecimal format

5.3 Goto Step 6 to save the packets to a temporary file

5.4 Go back to Step 4.1

Step 6: *Save captured packets into a file*

6.1 Create a temporary file say

6.2 Save captured packets into the opened file

6.3 Go back to Step 5.4

Step 7: *Close all the open network interface*

7.1 Delete the temporary file.

7.2 Close the network interface.

Step 8: End

3.2 Packet Loss Estimation

Let S be the set of packets whose features are recorded at the sender. Similarly, let R denote the set of the packets whose features are recorded at the receiver. Additionally, let L be the set of the packets lost between the sender and receiver. Then,

$$S \subseteq L \cup R \qquad (1)$$

for accurate packet loss measurement [OM05, ZO11].

If the above relation holds, some packets in S are correctly delivered to the receiver but not recorded at the receiver. Such packets are counted as lost though they are not lost. Thus, equation (1) must be satisfied to avoid overestimating packet loss. To prove statistically that equation (1) holds, a test at 95% level of confidence interval is carried out using the data given in table 1. The entries in table 1 are the output of a run of the Packet monitor algorithm in 3.1. Values in column 2 represents the number of packets captured at the time in seconds recorded in column 1, while column 4 and 5 are the source (IP addresses) and destination respectively. Column 6 is the type of protocol (Transport Control Protocol) used in the network traffic.

If we let n denote the number of sample packets taken, then in our example, n = 10, and the mean packet generated ($\bar{x}$) is 54.9 with a standard deviation (s) of 11.87. The population we are dealing is a normal distribution since $n \leq 30$, and we justify the reliability of equation (1) above with the t-test statistics $\bar{x} \pm t\left(n-1, \frac{\alpha}{2}\right).\frac{s}{\sqrt{n}}$ with n- 1 degree of freedom. We carried out the hypothesis test at 95% of significance where $H_0 : S = L \cup R$ the null hypothesis and $H_a : S \neq L \cup R$ is the alternative hypothesis. The result of the test confirms the null hypothesis that

S = L U R. Hence the relationship stated in (1) is satisfied.

3.3 Augmented Counting Algorithm

Executing the recording sequence (Augmented Counting Algorithm) described below in paragraph 3.5, yields two lists of packet features. One list, V_s, includes the data captured at the sender while the other list V_R includes the data captured at the receiver. The size of the buffer used is 8192bytes. Sent packet is displayed on the client output window (figure 2) and increases the number of sent packets by 1, that is, set $V_s = V_s + 1$. TCP socket reads the received message from buffer and converts it to characters which is then stored in variable chksum. The algorithm compare the value of chk (packet sent) and chksum (packet received), if the chk = chksum, displays the received message on the standard output device. An acknowledgement message with the format chksum# ACK is sent to the client. Let L be the number of the lost packets. The algorithm successfully counts the packets loss by subtracting the number of received packets from the sent packets, see 3.5 (Packet Counter Algorithm).

Table 1. Output of Packets monitor algorithm

Time taken to monitor traffic (s)	No. of packets (x)	The source address	The destination address	Type of Protocol
1	60	10.51.16.66:1209	173.194.34.64:88	Tcp
2	40	10.51.16.66:1209	173.194.34.64:.88	Tcp
3	52	10.51.16.66:1209	173.194.34.64:88	Tcp
4	52	10.51.16.66:1209	173.194.34.64:88	Tcp
5	66	10.51.16.66:1209	173.194.34.64:88	Tcp
6	40	10.51.16.66:1209	173.194.34.64:88	Tcp
7	61	10.51.16.66:1209	173.194.34.64:88	Tcp
8	40	10.51.16.66:1209	173.194.34.64:88	Tcp
9	68	10.51.16.66:1209	173.194.34.64:88	Tcp
10	70	10.51.16.66:1209	173.194.34.64:88	Tcp

3.4 Packet Counter Algorithm: Client

Step 1: Obtain server IP and port number
- 1.1.1 Create a variable to hold the server IP address
- 1.1.2 Create tcp client socket to use in connecting to server.
- 1.1.3 Declare and initialize variables: Sent, Vs=0; Received, Vr=0

Step 2: Connect to the tcp server using the tcp client
Step 3: Generate the data to send (Here we generate system GUID as our data)
Step 4: Compute the checksum of the data to send
Step 5: Make the checksum value a 10-digit string by padding 0s to the beginning of the value if the number of digits of the computed value is less than 10.
Step 6: Append the 10-digit checksum value to the beginning of the generated data to be sent
Step 7: Convert the data to send to array of bytes.
Step 8: Send the byte data to the server
Step 9: Display sent data on the client output window and increase the number of sent packets by 1
- 9.1 Set Vs=Vs+1
- 9.2 Wait for response from server. Server response is of the format CHECKSUM#ACK

Step 10: Receive response from server
- 10.1 Check if received message ends with the string ***#ACK***
- 10.2 If message ends with #ACK, get the checksum by replacing #ACK in the received message with empty string
- 10.3 Display received data on the server output window and increase the number of received packets by 1. Set Vr=Vr+1

Step 11: Close tcp client socket
Step 12: End

3.5 Packet Counter Algorithm: Server

Step 1: Obtain IP and port number
- 1.1 Create a variable to hold the server IP address
- 1.2 Create tcp listener to use in listening to connection from clients.
- 1.3 Create tcp socket to use in receiving data/sending response to client

While (*server is still running*)
Begin
Step 2: Initialize tcp listener
Step 3: Start listening for connection at a specific port (5002 used for this implementation)
Step 4: Initialize the receiving buffer. Buffer size used is 8192 bytes
Step 5: Tcp socket accepts an incoming connection
Step 6: Read the received message from the buffer and convert it to characters.
Step 7: Store the converted message in a string variable
Step 8: Extract the first 10 digits which is the checksum value and store in a variable ***chk***
Step 9: Extract the main message from position 11 to the end and store in variable ***msg***
Step 10: Compute the checksum value for the message stored in ***msg*** and store in variable ***chksum***. Pad up the checksum value with 0s to make it a 10-digit string
Step 11: Compare the value of ***chk***(received checksum) and ***chksum***(computed checksum)
Step 12: If (***chk*** = ***chksum)***
- 12.1 Display the received message on the standard output device
- 12.2 Send an acknowledgement message with the format ***chksum#ACK*** to the client

Step 13: Close tcp socket
Step 14: Close tcp listener
EndWhile
Step 15: End

4. IMPLEMENTATION

To implement the proposed algorithm, some issues must be addressed. The first is the creation of TCP client socket for receiving packets and sending response to client. We declare and initialize variables: such as, sent packets,$V_S = 0$ and received packets, $V_R = 0$. TCP socket accepts an incoming connection, read the received message from buffer. The sent packets are displayed on the client output window and increase the number of sent packets by 1, that is, $V_S = V_S + 1$.

Tcp chksum (sent packets) and chksum#ACK (received packets) are counted during an appropriate time period. This paper examines the following:- assume that the lost packet (L) is computed every T seconds from the numbers of chksum and chksum#ACK messages arriving during the most recent mT seconds (m:integer). This means that packet loss (L) is computed from chksum and chksum#ACK messages that have arrived in the time window whose size is mT seconds.

Let S_0 , S_1 , . . . ,S_{m-1} be m counters to count the chksum messages, while R_0 ,R_1 . . . , R_{m-1} are m counters to count chksum#ACK messages. In addition, let t denote the current time in seconds. Counter S_k (or R_k) is incremented by 1 for each chksum (or chksum#ACK) messages arriving during the period.

$$(N_m + k)\, T \le t \le (N_m + k + 1)\, T \qquad (2)$$

where N is an integer. For example, if T is 10 seconds, S_0 is incremented for $0 \le t \le 10$, S_1 is incremented for $0 \le t \le 20$ and so on. Then, when the current time satisfies $t = (N_m + K^*)T$ (3)

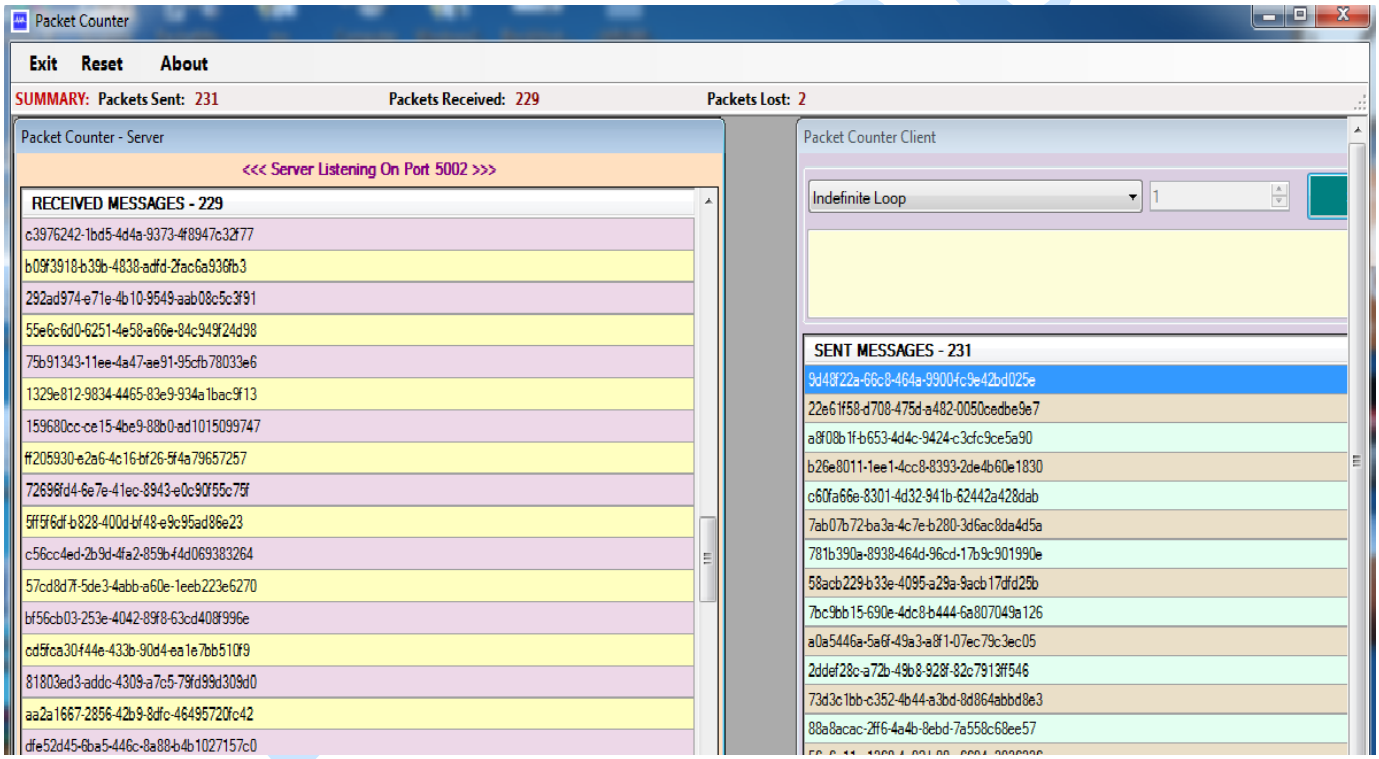

Figure 2. Output window of the Packet Counter

For integer K, the metric L for the most recent mT seconds is computed by:

$$L = \sum_{k=0}^{m-1} S_k - \sum_{k=0}^{m-1} R_k \qquad (4)$$

Simultaneously, S_k and R_k, which hold the oldest data among the counters are set to zero. The size of the time window, mT, must be sufficiently large to obtain a reliable value of L. If the time window size is too small, very few lost messages are counted during the time window.

4.1 Evaluation

To show the effectiveness of our proposed PMT scheme, an experiment was executed. The experiment employed an Intel Pentium Personal computer (PC) running windows at a minimum speed of 2.0GHz and 512MB of RAM with Network Interface Card, see figure 3. The connection time is

measured by the client programs and the proposed metric L (loss packet) is measured on the PC. The metric L is measured using the TCP socket by recording chksum (sent packets) and chksum#ACK (received packets) from user traffic,
the numbers of chksum and chksum#ACK packets, V_S and V_R are counted. Then the metric L is computed by equation (5):

$$L = V_S - V_R \quad (5)$$

It is observed that packet loss is identified correctly through the proposed scheme and which confirms that the packet loss (L) is determined by the mechanism described in section 3.3 of this paper.

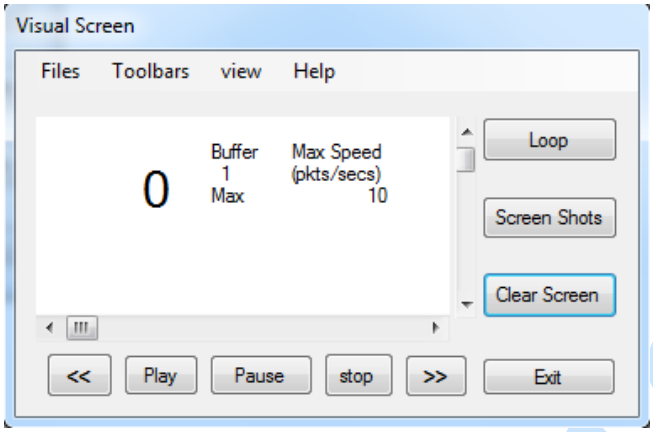

Figure 3. Main visualization window

CONCLUSION

PMT augmented with parity check is significant for packet loss measurement because it enables operators to take appropriate action against network performance degradation. This paper focuses on two points passive measurements - the transmitted packets are recorded at the sender host while the received packets are recorded at the receiver host. The packet records captured at these hosts during a specified period are gathered and compared. The result of the comparison allows the performance parameters to be estimated. The packet loss is detected by checking whether the same packet exists in the sender and receiver records. An accurate traffic model is introduced to assist the network performance analysis to overcome the problem of either over estimating or under estimating the network performance. The correctness of the analysis is confirmed through a computer simulation.

REFERENCES

[Asa98] **M. Asawa** – *Measuring and Analyzing Service Levels: A Scalable Passive Approach*, in Proceedings of Sixth IEEE/FIP International Workshop on Quality of Service, page: 3 -12, (1998).

[AI10] **M. Aida, K. Ishibashi** – *Compact Monitor: Change of Measure Based Passive/Active Monitor - Weight Active Sampling Scheme to Infer QOS*, in Proceedings of IEEE, SAINT Workshop on Measurement Technology for the Internet Applications, pages 119-125 (2010).

[BK11] **N. Brownlee, Andrzej Kwiecien** – *Methodology for Passive Analysis of a University Link*, in Proceedings of the Computer Networks, Ustron, Poland (2011).

[BS04] **P. Barford, J. Sommers** – *Comparing Probe Based and Router - Based Packet Loss Measurement*, Internet Computing. IEEE, vol. 8, Issue 5: 50-56, (2004).

[Has03] **M. Hasib** – *Limitations of Passive and Active Measurement Methods in Packet Networks*, in London Communications Symposium (2003).

[Has06] **Maheen Haib** – *Analysis of Packet Loss Probability in Packet Networks*, PhD thesis, University of London, UK (2006).

[J+06] **Sharad Jaiswal, Gianluca Lannaccones, Jim Kurose, Don Towsley** – *Formal Analysis of Passive Measurement Inference Techniques, in Proceedings of the 25th IEEE International* Conference on Computer Communication (2006).

[Opo11] **Samuel King Opoku** – *Parallel Self-Sorting System for Objects*, Cyber Journals: Multidisciplinary Journals in Science and Technology, journals of Selected Areas in Telecommunications, 2, 12, pages 1 – 8 (2011).

[OM05] **Satoru Ohta, Toshiaki Miyazaki** – *Passive Packet Loss Monitoring that Employs the Hash-Based Identification Technique*, Network Innovation

Laboratories, NTT Cooperation, Yokosuka, Japan (2005). In posters of IM2005, paper P2_09, Nice, France.

[Pap08] **Pavlos Papageorgiou** – *The Measurement of Manager: Modular and Efficient End-to-End Measurement Services*, Ph.D. Thesis, University of Maryland College Park (2008)

[PS10] **R. Padmalatha, G. Screedhar** – *A Novel Algorithm for Improving the End-to-End Active Packet Loss Measurements in Computer Networks*, International Journal of Computer Applications, India, vol. 6, No. 1:1-8 (2010).

[PT10] **Maria Priscilla, Antony Selvados Thanamani** – *Managing Network Congestion with a Modified Kohonen-Based Red Queue*, International Journal of Engineering Science and Technology, vol. 2, No. 11: 6747-6752 (2010).

[Q+10] **M. A. Qadeer, A. Iqbal, M. Zahid, M. R. Siddiqui** – *A Network traffic analysis and Intrusion Detection Using Packet sniffing*, in the Proceedings of Communication Software and Networks (ICCSN'10), Second International Conference, Department of Computer Engineering, Aligarh Muslim University, Aligar, India, pages: 313-317 (2010).

[TLS04] **T. Timotijevic, C. M. Leung, J. A. Schormans** – *Accuracy of Measurement Techniques Supporting Qos in Packet Based Intranet and Extranet VPN*, in proceedings of IEEE Communications, Special Edition on VPNs and Broadband Technology vol. 151, No. 1 (2004).

[ZO11] **Shan Zhu, Satoru Ohta** – *Real Time Measurement of Flows for IP Networks*, Cyber Journals: Multidisciplinary Journals in Science and Technology, Journal of Selected Areas In Telecommunications 2, 12, Dec Edition, pages 22 – 33, (2011).

[ZO12] **Shan Zhu, Satoru Ohta** – *Real Time Flow Counting In IP Networks: Strictly Analysis And Design Issues*, Cyber Journals: Multidisciplinary Journals In Science And Technology, Journal Of Selected Areas In Telecommunications 3, 2, Feb. Edition, pages 7 – 17, (2012).

[ZT08] **M. Zhangs, S. Tafvelin** – *Analysis of UDP Traffic Usage on Internet Backbone Links*, in Proceedings of 9th Annual International Symposium on Applications and the Internet, IEEE Computer Society, Washington DC, USA, page: 280-287 (2008).