Trace Analyzer for NS-2

Aliff Umair Salleh, Zulkifli Ishak , Norashidah Md. Din, Md Zaini Jamaludin
Dept of Electrical Engineering, College of Engineering
Universiti Tenaga Nasional
Km 7, Jalan Kajang-Puchong
43009 Kajang, Selangor.

Abstract – **Network simulator 2 (NS-2) is an open source discrete event simulation tool used for simulating Internet protocol (IP) networks. It was developed by UC Berkeley and widely used worldwide for network simulation purposes. The NS-2 software uses TCL as a front-end interpreter and C++ as the back end network simulation engine. Network simulation scripts in TCL are used to create the network scenarios and upon the completion of the simulation, trace files that capture events occurring in the network are produced. The trace files would capture information that could be used in performance study, e.g. the amount of packets transferred from source to destination, the delay in packets, packet loss etc. However, the trace file is just a block of ASCII data in a file and quite cumbersome to access using some form of post processing technique.**

In order to ease the process of extracting data for performance study, the NS-2 Trace Analyzer is proposed. This software is a tool for extracting and presenting trace files for the network simulation environment of NS-2. The NS-2 Trace Analyzer software consists of three layers. The first layer is the source layer which consists of the trace file data. The second layer is the processing layer. This layer processes the data obtain from the source and convert it to meaningful format for the third layer. The third layer is the presentation layer. This layer presents meaningful data in the form of graph, table and report for network performance study, i.e. throughput, end-to-end delay, packet loss and jitter. Through the NS-2 Trace Analyzer the user would be able to do performance study of a network scenario through interactive GUI. This will benefit the user since he or she can concentrate on developing new algorithms or new architectures rather spending too much time on post processing of data.

I. INTRODUCTION

In the era of globalization, the borderless world now has become more intimate with the advancement in communications between countries. New applications which rely on real time internet connection such as voice over internet protocol, video on demand and video streaming makes the requirement for Internet Protocol Quality of Service (IP QoS) paramount. Important parameters that would influence a network's IP-QoS are throughput, delay, jitter and packet loss. Since data transmitted have different service requirements, thus they cannot be treated equally. Traffic prioritization is required where data which are sensitive to delay and latency would be given higher priority treatment than data that are not delay sensitive. The limitation of bandwidth makes data prioritization essential

This scenario has attracted research and development works on finding method to improve the IP infrastructure to support the required QoS. There are many possible approaches to provide service priorities but the important key is not to burden the routers but to allow the strength of the Internet Protocol to accommodate new types of application. In order to provide IP QoS the Internet Engineering Task Force (IETF) has introduced Differentiated Services (DiffServ) and Multiprotocol Label Switching (MPLS) architecture [3]. The Diffserv architecture provide QoS by dividing traffic into different categories, marking the packet with a code point that indicates its category and scheduling the packet accordingly. MPLS is based on label. The MPLS ingress router assigns the packets with labels. Label Switched Router (LSR) in the MPLS core network direct the packet based on their labels. The packets pass through the network is completely determined by initial label assigned by ingress LSR.

Network simulators facilitate the study of network architectures such as IP QoS implementations. NS-2 is one of them. NS-2 is widely used for IP network studies. One of the areas that can be enhanced in NS-2 is on the post-processing capabilities. Network simulation scripts in NS-2 are used to create the network scenarios and upon the completion of the simulation, trace files that capture events occurring in the network are produced. The trace files would capture information that could be used in performance study, e.g. the amount of packets transferred from source to destination, the delay in packets, packet loss etc. However, the trace file is just a block of ASCII data in a file and quite cumbersome to access using some form of post processing technique.

This work is an effort to ease post-processing of NS-2 network performance data. Section 2 gives an overview of NS-2 simulation process whereas Section III describes the NS-2 trace file. Section IV describes the NS-2 Trace Analyzer. Section V concludes the paper.

II. USING NS2

NS2 undoubtedly is a very powerful network simulator that has been used widely. However, unfortunately it does not provide tools to represent results. To simulate a certain network scenario, NS-2 users need to write a simulation script, save it and invoke the NS2 interpreter. After that NS-2 will simply store the results in form of trace files. There are two types of trace files

This means that the NS-2 user will end up creating their own program to process the trace files to represent it in a presentable form. One popular approach is to produce two

1-4244-0527-0/06/$20.00 © 2006 IEEE

types of trace file, i.e. network animation (NAM) trace file and normal trace file. NAM trace file is used for network animation purposes [4]. While for trace file processing, a program can be coded using the user's own favorite software and graph plotters like GNU plot and xgraph can be used to view the results. Fig.1 shows the approach taken.

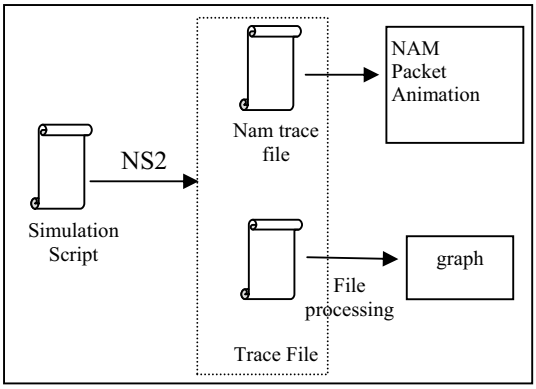


Fig. 1: Ns-2 Simulation Process Flow

In our work we developed a trace file analyzer that automates the process of extracting processing and presenting data from trace file.

III. NS2 TRACE FILE

The trace data is in ASCII code and are organized in 12 fields as shown in Fig. 2. Each trace line starts with an event descriptor followed by the simulation time (in seconds) of that event, and from and to node, which identifies the link on which the event occurred. The next information in the line are for flags. Since no flags are set here we have "------". Then we have the packet type and size (in Bytes). The next field is flow id (fid) of IP address that a user can set for each flow. Even though fid field may not be used in a simulation, users can use this field for analysis purposes. The next two fields are source and destination address in forms of "node**.** port". The last field shows the network layer protocol's packet sequence number. Note that even though UDP implementations do not use sequence number, NS-2 keeps track of UDP packet sequence number for analysis purposes. The last field shows the unique id of the packet.

Fig. 2: Ns-2 Trace File

IV. DEVELOPMENT OF NS-2 TRACE ANALYZER

The NS-2 Trace Analyzer is developed using open source tools. The platform used is Linux OS, and all the software and tools used are covered by free open source software licenses. One other effort that we see along similar line is the software Trace Graph [5] which was developed using MATLAB. The NS2 Trace Analyzer is developed using TCL\TK, an open source software which is also the platform for NS2 itself. This provides for seamless integration.

A. Architecture

Conceptually, the architecture of the Trace Analyzer is divided into three layers as shown in Fig. 3. The first layer is the source layer which handles the trace file. The second layer is the processing layer. This layer processes the data obtained from the source and convert it to meaningful format for the presentation layer. The third layer is the presentation layer. This layer presents meaningful data in the form of graph, table and report for network performance study; throughput, end-to-end delay, packet loss and jitter.

Brief description of the layers:

i. The source layer:

The Source Layer contains data from the trace file produces by NS-2 simulation. GUI scripting using TCL\TK has been done to enable NS2 trace analyzer to open selected trace file.

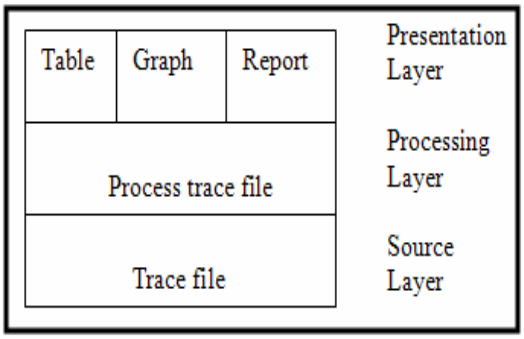
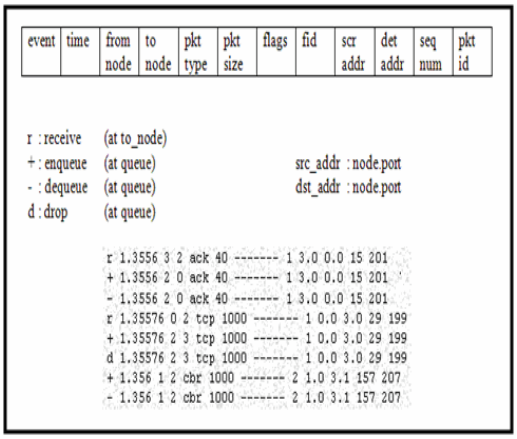


Fig. 3: NS-2 Trace Analyzer Layers

ii. The processing layer:

Since processing layer involves text processing, Unix awk utilities [7] are used extensively. The awk utilities



allows users to do operation on data files i.e. Filtering data column by column, averaging the value, summation, and data reformatting tasks. In this software, the utility is used for data extraction, calculation of performance parameter, and reorganization of the results data so that it will be compatible with the tools used in presentation layer.

iii. The presentation layer:

The presentation layer reads result data created from processing layer and then present it in these three forms:

- Report
- Table
- Graph

Report:

Report is used to display important parameter which is in single valued. There are two reports available, the first one is used to display the statistical simulation and network node statistics and the second report provide the analysis results.

The report for simulations result displays the statistical value for jitter, end-to-end delay, throughput and packet loss. Its also includes other significant information for network analysis purposes. TCL\TK label widget has been used to display the results.

Table:

Table is used to present the DiffServ Policy Table. The executable tcl script used to simulate the DiffServ will produces result in form of table. This table is base on the Policer Table [8]. The table consists of packets statistic that have been defined with DiffServ Policy Table. In post processing segment, the DiffServ tables are important for researcher using DiffServ to evaluate the performance of the DiffServ Policer. Therefore it is essential to include it into the software. TkTable widget, an extension TCL\TK is used to generate the table.

Graph:

For graph, the Trace Analyzer links to Xgraph for plotting purposes. Xgraph usually comes together with NS2 installation package.

Extra features

The trace analyzer also provides filtering tools to display the trace file for certain traffic only. Using this, user can filter the trace file for interested traffic.

B. Performance Parameter

The performance parameters that can be obtained through the NS2 Trace Analyzer are as follows, which are the main parameters of interest for IP QoS [6]:

- Throughput
- Packet Loss
- End to End Delay
- Jitter

Throughput

Throughput is the rate at which a network sends or receives data. It is a good channel capacity of network connections and rated in terms bits per second (bit/s).

Throughput, $T_p = \frac{P_a}{P_f}$,

where P_a is the packets received and P_f is the amount of forwarded packets over certain time interval.

Packet Loss

Packet loss is where network traffic fails to reach its destination in a timely manner. Most commonly packets get dropped before the destination can be reached.

Packet dropped/loss, $P_d = P_s - P_a$

where P_s is the amount of packet sent and P_a amount of packet received.

Jitter

Jitter is the fluctuation of end-to-end delay from one packet to the next packet of connection flow.

Jitter, $J = \left| D_{i+1} - D_i \right|$

where D_{i+1} is delay of i_{th}+1 packet and D_i is the delay of the i_{th} packet.

End to End Delay

End-to-end delay refers to the time taken for a packet to be transmitted across a network from source to destination.

End-to-end delay, $D = T_d - T_s$

where T_d is the packet receive time at the destination and T_s, packet send time at source node.

C. GUI layout

The layout of for the NS2 Trace Analyzer is fully designed using TCL\TK. It is designed in such a way that reports, graph, tables and the main layout is in its own window.

Fig. 4 shows the main window which consists of a button to load trace file, and also a button to read the loaded trace file. It also gives the options for the user to view reports, graph and/or table. The main window also displays the trace file property. A report window (Fig. 5) allows the user to request for per traffic flow information based on the source, destination and flow id. The trace file filter window enables user to filter traffic based on certain search field as seen in Fig. 6.

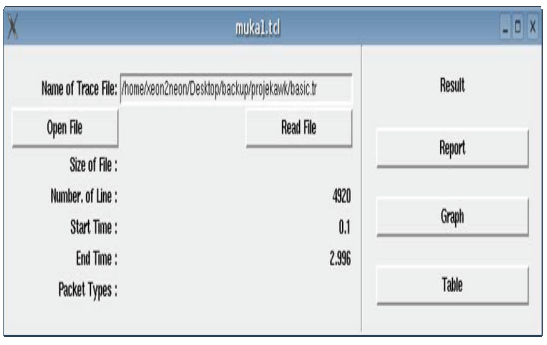


Fig. 4: Main window

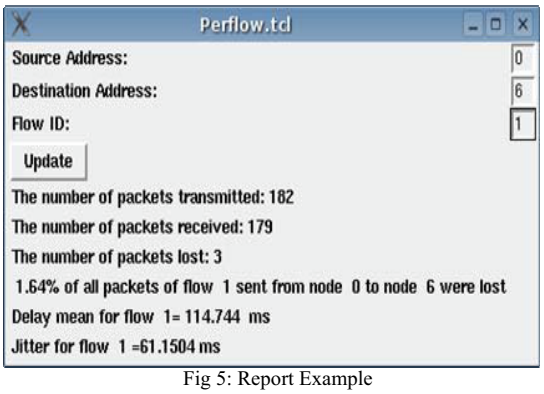


Fig 5: Report Example

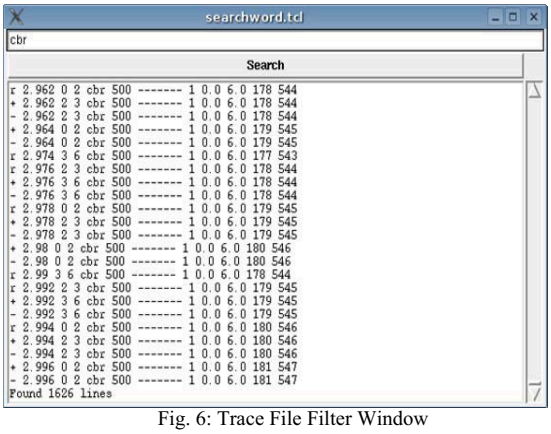


Fig. 6: Trace File Filter Window

V. CONCLUSION

The NS-2 Trace Analyzer is software meant to ease the task of analyzing network performance after network simulation on NS-2. By the click of a button, the software caters the need of users through easy to use GUI and eliminates the manual processing task. This could enhance the users productivity such that more time and focus can be given to networking studies with lesser hassle on the performance analysis part.

VI. REFERENCES

[1] Kevin fall, Kannan Varadhan. "The ns Manual (formerly ns notes and documentation)". The VINT project, July 2003.
[2] TCL/TK programming language, http://wiki.tcl.tk
[3] E. Rosen, A. Viswanathan, and R. Callon, "Multiprotocol Label Switching Architecture", RFC 3031, Jan. 2001.
[4] Network Simulator Manual, http:www.isi.edu/nsam/ns/index.html
[5] J. Malek, http://www.tracegraph.com/
[6] Christopher Y. Metz. "IP switching: Protocols and Architecture". McGraw Hill Publisher, USA 1998.
[7] AWK scripting for Network simulator 2 http://www.sop.inria.fr/maestro/personnel/Eitan.Altman/ns.htm
[8] S.Blake et al "An Architecture for Differentiated Services", RFC 2475, Dec 1998