Journal of Engineering and Applied Sciences 8 (1): 1-8, 2013
ISSN: 1816-949X

An Embedded Network Traffic Monitoring System for Portable Applications

Mostafijur Rahman, R. Badlishah Ahmad, Zahereel Ishwar Abdul Khalib, Abid Yahya,
Md. Mijanur Rahman, Manjur Ahmed and Naufal Alee
School of Computer and Communication Engineering,
University Malaysia Perlis, P.O. Box 77,
d/a Pejabat Pos Besar, 01007 Kangar, Perlis

Abstract: This study presents an enhanced Embedded Network Traffic Monitoring (ENTM) system capable of capturing and analyzing network traffic information on data networks. The system incorporates an enhanced packet probing subsystem for low end interfacing to data network. The packet analysis engine included a sophisticated memory managing scheme to tolerate considerable bursts in network traffic. Comparative experimental results showed that the ENTM system had performance comparable with well known third party tools. In order to demonstrate that resource constraints do not significantly degrade system performances, researchers implemented the packet probing subsystem on a desktop where memory and processing power were much larger. It was found that the ENTM system had only little degradation (0.5%) in performances compared to the desktop version. Requirements of low processing power and memory make the system suitable for low end and portable applications.

Key words: Embedded operating system, network traffic monitoring, performance comparison, single board computer, ENTM

INTRODUCTION

Internet/intranet network traffic monitoring has become an indispensable topic in today's research as the network grows, the need for predicting network traffic, protocol and stack analysis poses a challenge for companies that intend to establish large communication links. Therefore, it is crucial to monitor a network in order to understand the network's behavior and to react appropriately to the need to design and provide a more efficient network in the future. The common features of network traffic monitoring include: Providing data transfer rates on the network segment, types of traffic transferred within a LAN, traffic generated per node, amount of traffic going through or coming from a system or application which is causing a bottleneck and the level of peak traffic (Hong *et al*., 1999; Kushida, 1999; Bolot, 1993; Paxson, 1999). The rapid growth of hardware technologies has resulted in the proliferation of a large variety of smaller hardware architectures and platform orientations that have been leading a large demand for embedded software (Geer, 2004). So, programmers are focusing more and more on the development of software on embedded systems in order to make them portable and platform independent. Embedded software is marked with stamps as: Timeliness, concurrency, liveliness, reactivity and heterogeneity (Lee, 2002). It is built to develop applications for a very small target market that does not require a keyboard, video, floppy disks and hard drives. The expected application of this research is to make an embedded network traffic monitoring system that system administrators, network engineers, security engineers, system operators and programmers can use.

Traditionally, internet network traffic monitoring applications have been developed to run on bulky PCs with high, often unnecessary processing power. Immediate identification and capture of traffic that causes congestion is crucial in speeding up the network problem diagnostic process. Thus, the benefit of low cost, small size and portability which an embedded system offers has never been a feature of these kinds of applications. The emergence of embedded systems in particular embedded Linux has driven developers to take up the challenge of developing a high processing power application on embedded platforms. An embedded system for this purpose should enable plug-and-play devices to provide traffic conditions in particular network segments and enable real time traffic capture and storage. At the same time, the system should act as a server to provide collected traffic statistics to enable

Corresponding Author: Mostafijur Rahman, School of Computer and Communication Engineering, University Malaysia Perlis, P.O. Box 77, d/a Pejabat Pos Besar, 01007 Kangar, Perlis

network engineers to identify network problems (Rahman *et al*., 2008). One of the main problems in developing embedded software is inadequate software architecture because better performance is necessary to reduce processing overheads, memory usage and power consumption (Lee and Yi, 2011; Lee, 2010; Xuejian *et al*., 2005; Meedeniya *et al*., 2011).

Researchers proposed a real time and historic internet/intranet network traffic monitoring system implemented on an embedded GNU/Linux-based Single Board Computer (SBC) which has recently been an attractive alternative for embedded computing because of its reduced demands on processing power and memory. Also, researchers devised a comprehensive testing and validation scheme to investigate the reliability and practicality of the proposed system. In this research, an Embedded Linux-supported SBC (TS-5400) is used. The TS-5400 SBC is selected because of its small size, portability, low power consumption, off-the-shelf availability, ruggedness, packaging, library compatibility, GNU/Linux OS support as well as its low cost.

MATERIALS AND METHODS

Proposed system: The proposed Embedded Network Traffic Monitoring (ENTM) system primarily consists of a network information processing engine running on TS-5400 Single Board Computer (SBC) under the operating system TS-Linux.

System overview: As shown in Fig. 1, the overall system is an interconnection of four major components:

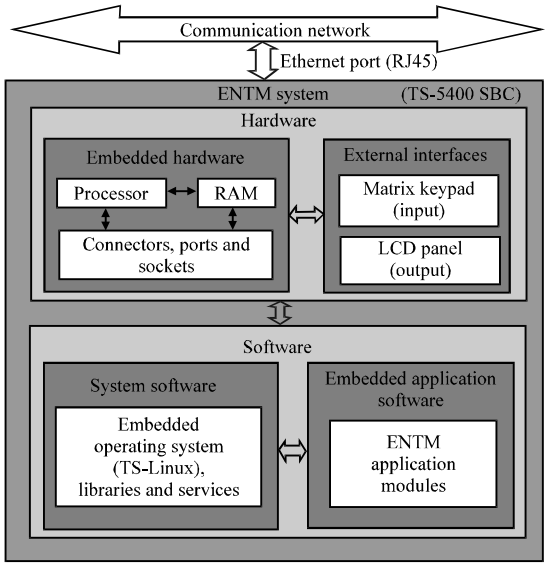


Fig. 1: Overall ENTM system architecture

- Dedicated hardware
- Peripheral devices
- System software
- ENTM application modules

With the goal of proposing a network information processing system using low processing power and memory usage, researchers used a low end embedded computer, TS-5400 which has on-board a 32 bit processor (133 MHz) and 16 MB RAM. Network traffic information is captured through Ethernet connectivity. For interaction with the management and control plans of the system, an LCD panel and a keypad are used. Further, researchers have used a light version of embedded Linux (TS-Linux) which is <18 MB in size. The major contribution consists of incorporating enhanced techniques into ENTM application module. These techniques are associated with network packet information extraction, individual host identification and peak traffic level detection.

Architectural design of the ENTM application module: The application module consists of four major modules: System Control (SC), Network Packet Probe (NPP), Packet Analyzer (PA) and view module (web based user interface). The NPP module captures packets from a network segment. The analyzer analyzes the packets' information and saves analyzed data into data log files. Network traffic information can be monitored from the data log files using any web browser. Figure 2 shows the architectural design for ENTM application modules.

System Control (SC) module: The SC module implements the management and control plan of the ENTM application module. It performs five major tasks, namely:

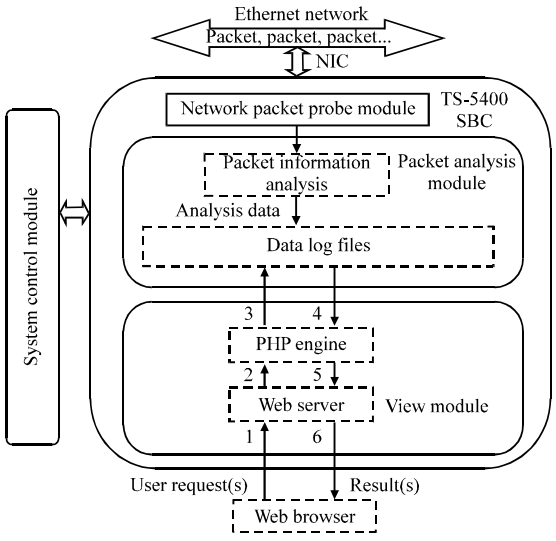
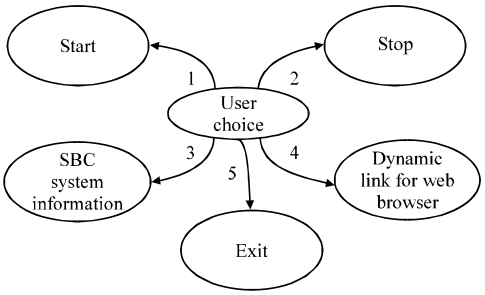


Fig. 2: Architectural design of the ENTM system

- Initialization
- System status monitoring
- Providing dynamic link to the associated web browser
- Disabling the ENTM application module
- Graceful shutdown and restart

Figure 3 shows the overall functional diagram of the SC module.

Network Packet Probe (NPP) module: The NPP module functions are to capture and extract network packet information and store it in a data buffer for further processing. The processes of this module involve network adapter setting, user input processing, packet capture, packet information extraction and packet information storing in a data buffer for analysis. This module operates at the network layer and captures network packets physically through the Network Interface Card (NIC). Figure 4 shows the flowchart of the NPP module. In the probe module, two inputs are given through the matrix keypad and displayed on the LCD panel. The packet capture time interval is used to set the time interval limit for packet capturing in seconds. Another input is level of peak traffic in kb/sec. The peak traffic level means a data transmission rate on a network segment has reached a certain level of data transmission given by the user. The peak traffic measurement process is to compare the current data capture rate with the given value of peak traffic level and the current data capture rate exceeds the given peak traffic level. The matrix keypad is used to enter character input. Figure 5 shows the algorithm that converts sequential character inputs into an integer value. The probe module extracts data from network packets in each layer and saves it to a preconfigured temporary buffer. Each packet information consist of packet capture time, source MAC address, destination MAC address, source IP, destination IP, network layer protocols, transport layer protocols, application layer protocols, source port, destination port and packet length. These pieces of information are grabbed by an algorithm as shown in Fig. 6. Information on the Ethernet, IP, TCP and UDP header structures (Hassan and Jain, 2004). In the implementation, the functionality of the probe module is realized through the use of libpcap open source library (WestNet, 2001).



1- Start: Start ENTM system application module 2- Stop: Stop ENTM system application module 3- SBC system information: • IP address • Running time • RAM free space • CF free space • Cancel	4- Network traffic information: Display URL where network traffic information can be accessible 5- Exit: • Shutdown • Restart • Cancel

Fig. 3: Overall functional diagram of system control module

Packet Analysis (PA) module: The PA module performs two jobs: Finding and updating individual host information from the captured packets and sorting the information according to its data exchange value.

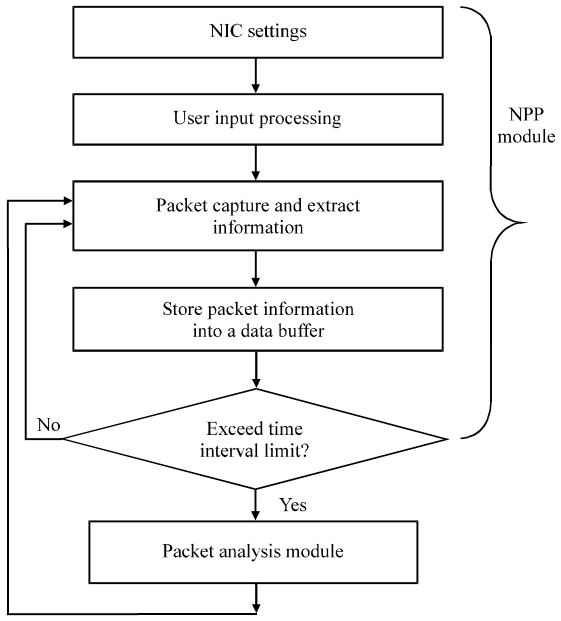


Fig. 4: Flow chart of the NPP module

1: Set output variable, X = 0
2: Read a character (C) from matrix keypad
[Character must be 0-9]
3: Check if input character is "Enter" or "D" then goto step 7
4: Convert the character C into integer I
5: Apply the equation, X = X * 10+I
6: Goto step 2
7: Exit

Fig. 5: Algorithm for user input processing

1: Initialize Ethernet, IP, TCP and UDP header structure
2: Capture packet
3: Extract packet length
4: Extract source and destination MAC address
5: Check the network layer protocol (IP or ARP)
6: If the packet is an IP packet then:
   i) Save source and destination IP and MAC addresses
   ii) Check Transport Layer Protocol (TCP or UDP)
   iii) Save the source and destination port address
7: Check the application layer protocol based on source and destination port
8: Exit

Fig. 6: Algorithm for packet information grabbing

/*This algorithm consists of two structure arrays, PacketInfo[ ] and HostInfo[ ]. At the initial time, the PacketInfo[ ] contains packet information and HostInfo[ ] is empty. The variable no. of pack contains the number of packets captured. */
1: Set M = 0, Host No. = 0
2: If (M> = No. of pack) Goto step 12
3: Set N = 0 and Check = 0
4: (i) I = Compare (PackInfo [M]->Source IP, HostInfo [N]->SourceIP)
   (ii) J = Compare (PackInfo [M]->Destination IP, HostInfo [N]->Source IP)
5: If (I = True OR J = True)
   (i) Update HostInfo [N]
   (ii) Set Check = 1
   (iii) End If.
6: Set N = N+1
7: If (Check = 1) Then
   (i) M = M+1 and Goto step 3
8: If (N<Host No.) Goto step 4
9: If (Check = 0) Then
   (i) Save the packet info as a new host info
   (ii) Set Host No. = Host No.+1
   (iii) End If
10: Set M = M+1
11: Goto step 2
12: Exit

Fig. 7: Algorithm for finding and updating individual host information

Researchers used quick-sort algorithm for sorting all hosts' information because it is the fastest sorting algorithm and is used for most sorting applications (McMillan, 2007). Figure 7 shows the algorithm for obtaining and updating individual host information. From the individual and sorted host information users can monitor all hosts' activities and bandwidth usage. The grabbed hosts' information is sorted in descending order according to amount of data exchanged and all sorted information is stored into files for viewing. This sorted host information helps users to identify hosts using higher bandwidths on a particular network segment.

Web based user interface: Architecturally, researchers use a client-server web system (Fig. 8) (Welling and Thomson, 2005) for facilitating users to ubiquitously monitor traffic information through this system. The web based user interface comprises the server part of this client-server web system. This module enables the user to monitor important pieces of traffic information including the protocols at network, transport and application layers, bandwidth usage for these protocols as well as traffic between host pairs. Because of the web based user interface, the analyzed network traffic information can be viewed through any web browser. The web pages are developed using PHP and HTML code inside the SBC. The PHP script reads the log files according to user selection and displays the network traffic information. As mentioned, this information was computed and saved into log files by the PA module. This interface enables the user to monitor two important types of information, namely; network traffic information and system health information. As shown in Table 1, the network traffic information

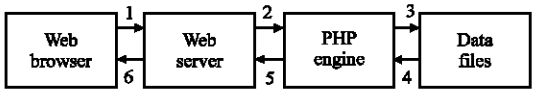


Fig. 8: Data transaction between web browser and server

Table 1: Network traffic information

Information types	Description
Real time (displays the network status in real time, online)	
1: Total traffic	Display: Packet capture time interval, peak traffic level, start time, endtime, No. of packets captured, No. of hosts, peak traffic level exceeded (Yes/No), data exchanged (kb), data capture rate (kbps), packet capture rate (packets sec^{-1}), average packet size (bytes), No. of broadcast packets and protocol information (Network, transport and application layer protocols)
2: Host info.	Display (according to max data exchanged): Host No., host IP address, MAC address, data exchanged and protocol information
Historic (displays the accumulated traffic history)	
3: Total traffic info.	In contrast to info type 1 which shows instantaneous information, info type 3 represents historic data, i.e., same pieces of information collected over a predetermined amount of time
4: Top 5 hosts info.	Same as information type 2 but only selected top five host's information is stored for every time interval
5: Traffic history accumulated every 15 min	Same as info type 3 but accumulated over a larger amount of time (15 min)
6: Peak traffic info.	User input, peak traffic time and traffic information at the peak traffic time

includes both real time and historic traffic information. The system health information includes system date and time, running time, memory status, network interface status and available running processes.

RESULTS AND DISCUSSION

This study describes experimental results to show the performance evaluation of the proposed ENTM system. The hardware used is a 32-bit SBC coupled with an LCD panel, matrix keypad and Ethernet connectivity. The software part is a combination of system software and embedded application software. The hardware components used are: TS-5400 SBC, LCD panel, matrix keypad, compact flash memory card, development PC, HUB and switch. TS-5400 is a compact, full-featured PC-compatible, single board computer based on the AMD Elan520 processor at 133 MHz. This small size (4.1"× 5.4") system can operate over a wide range of temperature (-20 to +70°C). Also, it consumes nominal power (5V DC @ 800 mA). The embedded OS used for the TS-5400 is TS-Linux (version 3.07a).

The experimental setup as shown in Fig. 9 involves a HUB (10 BASE-T) and a switch (10/100 3COM 3C17304 SuperStack@3) directly connected by means of an Ethernet link. The ENTM system is connected to the HUB through an Ethernet port. The HUB is directly connected to the internet. All of the experimental data were taken from a network segment of the University of Malaysia Perlis (UniMAP) network.

The performance is presented in terms of hardware performance and network packet capturing performance. Measurement of hardware performance is based on CPU and memory utilization. For comprehensive evaluation of system performances, researchers tested the system in two configurations. Performances of these two configurations were compared between them as well as with a third party network analyzer.

Hardware utilization efficiency: Hardware utilization efficiency was measured in terms of CPU and memory utilization expressed as percentage of usable resources at full capacity. These measurements in turn facilitate detection of bottlenecks or blocking areas as well as identification of resources having low utilization. The top program in TS-Linux is used to measure CPU and memory load of the SBC.

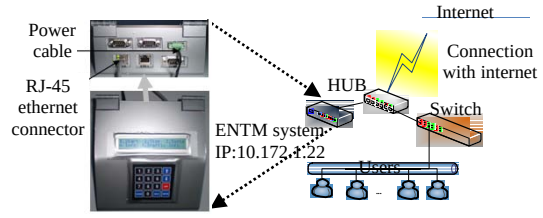


Fig. 9: Experimental setup for the ENTM system

CPU utilization: The CPU time is often measured in clock ticks or as a percentage of the capacity. In this research, researchers adopted percentage of the CPU capacity as the utilization criteria. Figure 10 shows the CPU utilization rate graph for TS-5400 SBC without running ENTM system modules. As seen in Fig. 10, the bare system routines utilized 2.1% of the CPU power on average. Out of this usage, 1.4% was used by the default system and 0.7% by user processes.

Figure 11 shows the CPU utilization when the ENTM system modules were running on TS-5400 SBC. As seen in Fig. 11, the average CPU utilization rate was 14.62% with 7.8 and 6.8% for the system and user processes, respectively. In the test scenario, average CPU utilization rate for the proposed ENTM modules was estimated to be 6.2%.

It can be concluded from the results as shown in Fig. 12 and 13 that the packet capturing, filtering and processing modules can be executed on the TS-5400 low end SBC.

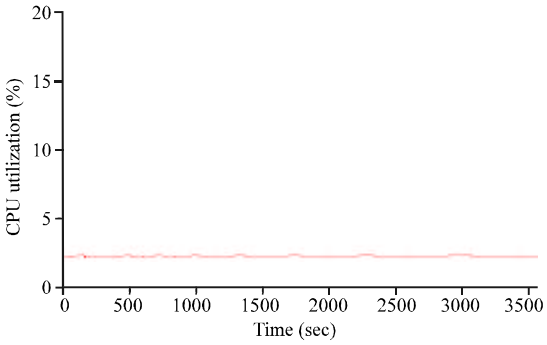


Fig. 10: CPU utilization graph for the TS-5400 SBC

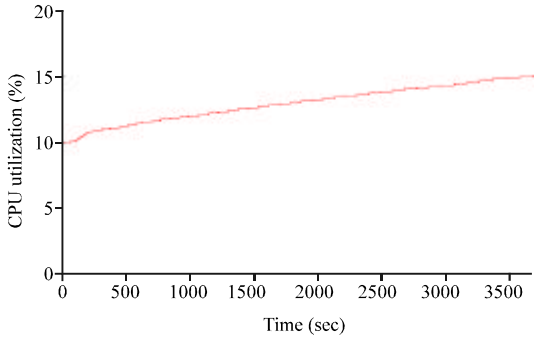


Fig. 11: CPU utilization graph during the ENTM application module's execution

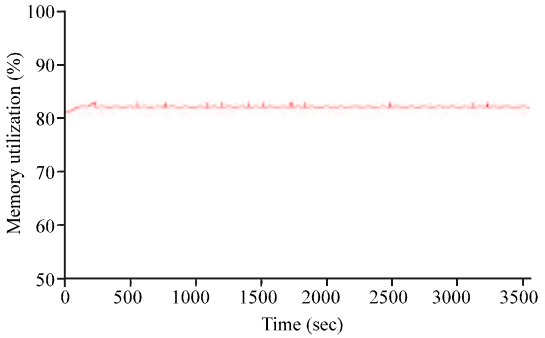


Fig. 12: Memory (RAM) utilization graph for the TS-5400 SBC

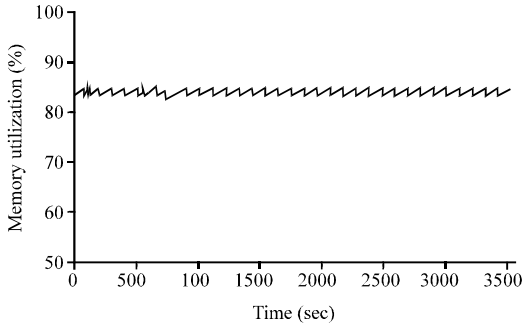


Fig. 13: Memory (RAM) utilization graph during application module's execution

Memory utilization: Similarly to the CPU utilization, researchers carried out measurement of memory (RAM) utilization as a percentage of total system memory (16 MB). Experimental results are shown in Fig. 12 and 13. As shown in Fig. 12, memory utilization for bare system (without ENTM system modules) was 82.1% on the average. When the ENTM system application modules' was running, the average memory usage was increased to 84.76% (Fig. 13).

It can be concluded that the ENTM system modules require considerably small amount of memory (<16 MB).

Performance comparison of the proposed system module: The performance of the NPP was evaluated by comparing the packet capture rate and data capture rate with the performance of a desktop PC and desktop based network traffic monitoring software. The SBC and desktop PC differ in processor type, processing speed, RAM size, kernel and OS. Table 2 shows comparison between SBC and the desktop PC used in the experiment to measure NPP performances.

ENTM module on SBC and desktop PC: The NPP performance evaluation was done by measuring the packet and data capture rate on different platforms. The measurement was performed by executing the code on both SBC and desktop platforms at the same time. Figure 14 shows the packet capture rate (pps) for both SBC and desktop PC. The x-axis represents the number of packets examined. The y-axis indicates the packet capture rate for the examined number of packets. The Packet Capture Rate (PCR) is a specific measure of how many packets are captured within a specific range of time.

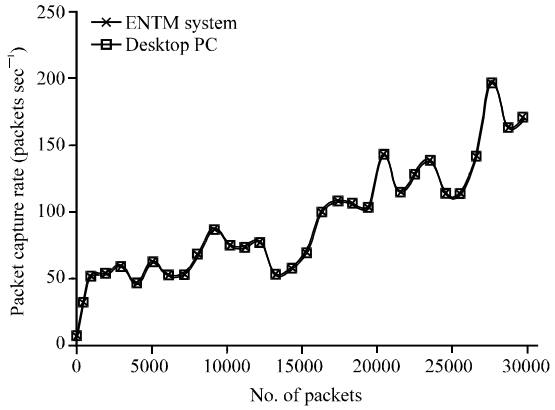


Fig. 14: Packet capture rate comparison between ENTM system and desktop PC

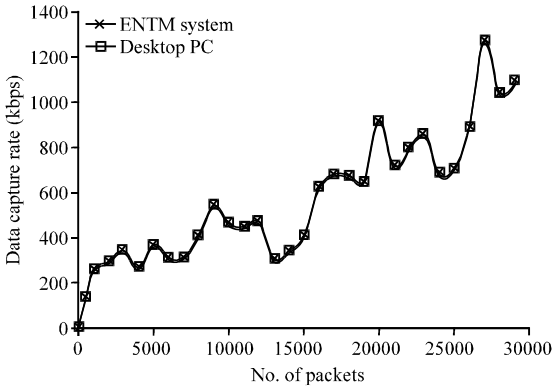


Fig. 15: Data capture rate comparison between ENTM system and desktop PC

Table 2: Comparative specification of SBC and desktop PC

Items	TS-5400	Desktop PC
Processor type	AMD Elan520	Intel (R) Core (TM) 2 Duo CPU
Processor speed	133 MHz	2.2 GHz
RAM size	16 MB	2 GB
Secondary storage	1 GB	250 GB
Operating system	TS-Linux 3.07a	Ubuntu 7.10
Linux kernel	2.4.23 ts	2.6.20-16 generic
NIC	10/100 Ethernet	VIA Rhine II Fast Ethernet

Figure 14 shows that the packet capture rates were quite similar for both NPP on ENTM system and desktop PC. Within 27,000 packets capture time, the maximum packet capture rate was 195.89 (pps) for ENTM system and 194.47 (pps) for desktop PC. The PCR average

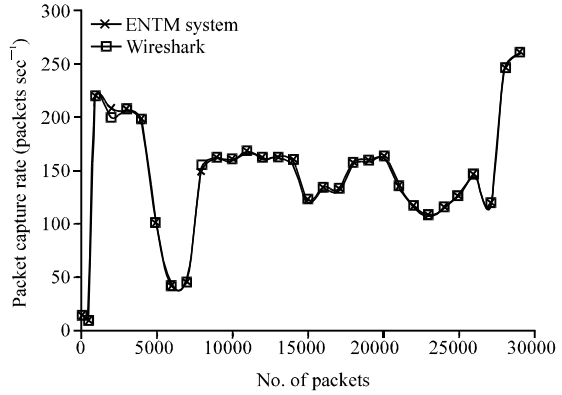


Fig. 16: Packet capture rate comparison between ENTM system and Wireshark

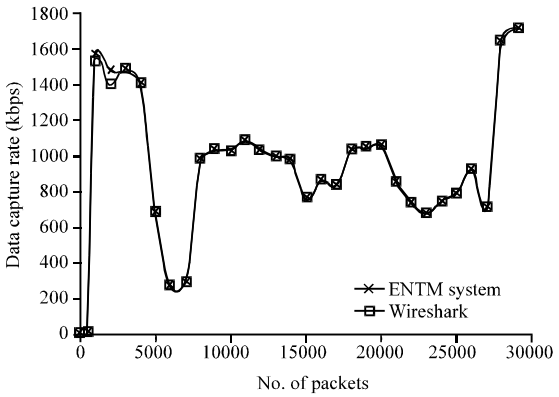


Fig. 17: Data capture rate comparison between ENTM system and Wireshark

difference between the ENTM system and desktop PC was 0.22 (pps) or 0.12%. Figure 15 shows the Data Capture Rate (DCR) which is a specific measure of how much data are captured within a specific time interval for both ENTM system and desktop PC. The data capture rates were almost similar for both ENTM system and desktop PC. At 27,000 packets capture time, the maximum DCR was 1.28 Mbps for ENTM system and 1.27 Mbps for desktop PC. The average DCR difference between the ENTM system and desktop PC was 1.65 (kbps) or 0.14%. It can be concluded from the above comparison that the packet capturing efficiency using NPP on ENTM system is not much different from NPP on a high performance desktop PC. Thus, the NPP on the ENTM system is well suited for packet capturing purposes, although the processing speed and memory are much lower than the desktop PC.

ENTM module on SBC and Wireshark: The ENTM system performance was also compared with that of Wireshark (ver. 0.99.4) running on desktop PC. Figure 16 shows the packet capture rate (in packet per sec) for both ENTM system and Wireshark; the packet capture rates were similar for both SBC and Wireshark. Within 29000 packets capture time, the maximum packet capture rate was 259.83 (pps) for ENTM system and 258.92 (pps) for Wireshark. In this experiment, the average packet capture rate difference between the ENTM system and Wireshark was 0.97 (pps) or 0.34%.

Figure 17 shows the data capture rate (kb sec^{-1}) for both ENTM system and Wireshark. It shows that the data capture rates are almost similar for both these systems. At 29000 packets capture time, the maximum data capture rates was 1.72 Mbps for ENTM system and 1.71 Mbps for Wireshark. The average data capture rate difference was 4.18 (kbps) or 0.22%.

It has been observed from the above comparison that packet capturing on SBC is not much different from high performance desktop based software (Wireshark). These comparisons clearly shows usability of the ENTM system for network traffic analysis although, it runs at a very low power (5V DC @ 800 mA).

CONCLUSION

Researchers proposed an enhanced Embedded Network Traffic Monitoring (ENTM) system capable of performing network probing and packet analysis. A very desirable feature of the system is that significant reduction in processing power and memory, above bare minimum requirements does not significantly degrade the system performance. The efficacy of the system was validated through a comprehensive experimental scheme where performances of the system were compared with that of Wireshark, a well known third party network traffic analysis tool. The claim that resource constraints do not significantly degrade system performances was validated though implementing the system both in high end desktop PC and in low end embedded platform and comparing their performances. The experimental results showed that difference of the performance of the system on the low end platform from that of high end one was well within 0.5%. Thus, it may be concluded that the design and implementation of the ENTM system was highly competitive particularly for application in the low power embedded platforms.

REFERENCES

Bolot, J.C., 1993. End-to-end packet delay and loss behaviour. Proceedings of the SIGCOMM, September, 1993, San Francisco, CA., USA., pp: 289-298.

Geer, D., 2004. Survey: Embedded Linux ahead of the pack. Distrib. Syst., Vol. 5. 10.1109/MDSO.2004.28.

Hassan, M. and R. Jain, 2004. High Performance TCP/IP Networking. Pearson Prentice Hall, New Jersey, USA.

Hong, J.W.K., S.S. Kwon and J.Y. Kim, 1999. Web traf mon: Web-basedinternet intranet network traffic monitoring and analysis system. Comput. Commun., 22: 1333-1342.

Kushida, T., 1999. An empirical study of the characteristics of Internet traffic. Comput. Commun., 22: 1607-1618.

Lee, E.A., 2002. Embedded Software: Advances in Computers. Vol. 56, Academic Press, London, UK.

Lee, J., 2010. On-Chip bus Serialization method for low-power communication. ETRI J., 32: 540-547.

Lee, J. and J. Yi, 2011. Improving memory efficiency of dynamic memory allocators for real-time embedded systems. ETRI J., 33: 230-239.

McMillan, M., 2007. Data Structures and Algorithms Using C#. Cambridge University Press, New York, USA.

Meedeniya, I., B. Buhnova, A. Aleti and L. Grunske, 2011. Reliability-driven deployment optimization for embedded systems. J. Syst. Software, 84: 835-846.

Paxson, V., 1999. End-to-end internet packet dynamics. IEEE/ACM Trans. Network., 7: 277-292.

Rahman, M., Z.I.A. Khalib and R.B. Ahmad, 2008. A portable network traffic Analyzer. Proceedings of the International Conference on Electronic Design, December 1-3, 2008, Penang, Malaysia, pp: 1-6.

Welling, L. and L. Thomson, 2005. PHP and MySQL Web Development. Sams, USA.

WestNet, 2001. Protocol Analysis (WB77.0). West Net Learning Technologies, USA.

Xuejian, L., Y. Jing and W. Minghui, 2005. A heterogeneous evolutional architecture for embedded software. Proceedings of the 5th International Conference on Computer and Information Technology, September 21-23, 2005, Shanghai, China, pp: 901-905.