

# Software-based Internet Traffic Classification and Prioritization to Improve Network Performance in Multimedia Broadband Networks

N. Akhtar<sup>1</sup>, M. Kamran<sup>2</sup>

<sup>1,2</sup>Department of Electrical Engineering, UET Lahore, Pakistan  
<sup>1</sup>akhtar.naveed@uet.edu.pk

**Abstract**—Rapid development in multimedia broadband applications in the last decade has led towards higher bandwidth demand, which has resulted in quality of service issues for business critical applications. Researchers have suggested internet traffic classification and prioritization as a solution to the problem but still the complexity of available solutions is a fertile research area. In this paper, traffic classification and prioritization was performed using FreeBSD (v8.1) operating system for a small and medium enterprise network, the system and network performance results are presented for optimum performance conditions. Though researchers have worked on open source systems for traffic classification but the computation of impact of classification on system and network performance, and to suggest optimum performance conditions is still a fertile research area. The implementation is done by using FreeBSD (v8.1) IP Firewall (IPFW), pipes and queues, on an Intel 3.0 GHz machine and the results are presented for critique review and discussion. The key challenges faced and open issues are also discussed for future research.

**Keywords**—IP Firewall (IPFW), Deep Packet Inspection (DPI), Internet Service Provider (ISPs), Serial Line Internet Protocol (SLIP), Quality of Service (QoS)

## I. INTRODUCTION

Network congestion is among the key challenges faced by different Internet Service Providers (ISPs), aggravated further by inefficient utilization of link capacity and peer to peer applications that eat up the entire bandwidth. Present solutions are based on rate limiting different users by classifying internet traffic via their application and usage patterns. An intrusion detection system [i-ii] also works on the principle of Internet traffic classification and it is used to prevent DDoS (Distributed Denial of Service) attacks. The pre-requisite for Internet traffic classification is packet inspection. However strict privacy policies and heavy

network load coupled with high processing and infrastructure requirement for deep packet inspection have made it difficult to implement. The researchers have responded to this difficulty by working out different methods of internet traffic classification, which are based on statistical characteristics of different traffic flows without performing deep packet inspection. There are number of packet scanning applications which are implemented across different networks and they are capable of doing packet inspections like SNORT [i], Bro [ii] and Linux L-7 (Layer-7) filter. SNORT and Bro are two mostly used Intrusion detection systems, whereas Layer-7 filter is an application for application layer protocol analysis, which makes packet classification based on application layer data. Traffic classification helps to ensure network security, to filter malicious traffic flows and to offer billing solutions as per application requirement. It is one of the key requirements for Internet Service Providers (ISPs) to perform usage based billing by deploying an efficient but low cost solution which not only allocates available bandwidth based on nature of application but also ensures Quality of Service (QoS) for business critical applications [iii].

We conducted analysis of internet traffic of Pakistan Largest multimedia and broadband service provider (Pakistan Telecommunication Company Limited) with over 1.5 Million broadband customers across Pakistan, and the traffic was studied for the period June 2014 to December 2014 at PTCL core aggregation sites in Lahore and Islamabad and the analysis shows that 20% of the available bandwidth was consumed by Peer to Peer (P2P) applications; the top applications which consumed major part of overall bandwidth are shown in Table I. These P2P applications create network congestion and for optimum utilization of network bandwidth all such applications need fair sharing of available bandwidth. For these reasons researchers are working on different traffic classification techniques; which can help to classify network traffic into different traffic flows and to allocate bandwidth as per nature and priority of the

Application.

TABLE I  
BROADBAND TRAFFIC CLASSIFICATION BASED ON  
TYPE OF PROTOCOL

Protocol type	% Bandwidth Utilization
Bit Torrent UTP	20.10%
Other TCP Protocol	11.70%
Bit Torrent	10.00%
UDP	8.90%
Bit Torrent Encrypted	7.00%
Flash Video	6.50%

The rest of the paper is organized as follows: Section-II covers literature review in this area; Section-III covers the methodology and approach. Section-IV covers implementation and system design, Section-V covers results and comparison analysis, Section-VI concludes the paper with discussion on key results achieved and future scope of work.

## II. LITERATURE REVIEW

A detailed survey work has been done by Nguyen and Armitage [iv] that covers detailed work up to 2008. Most of the researchers have suggested internet traffic classification by using Machine Learning Algorithms. W. Li & A. W. Moore suggested machine learning approach based on Naive Bayes & C4.5 decision tree [v-vi] algorithms, which accurately classify the internet traffic by collecting different features at the start of internet traffic flow. Zander, Sebastian, Thuy Nguyen & Grenville Armitage [vii] used machine learning for dynamic identification of different internet traffic using their statistical characteristics. Nguyen, Thuy TT, Grenville Armitage, Philip Branch, and Sebastian Zander [viii] also used machine learning technique to analyze interactive IP traffic. In Internet traffic the P2P applications consumes most of the bandwidth and Sen, Subhabrata, Oliver Spatscheck & Dongmei Wang [ix] worked on identification of P2P traffic using application level signatures and designed online filters that were able to track P2P traffic with accuracy and robustness. But, W. Jiang & M. Gokhale [x] suggested that computation complexity of internet traffic classification using statistical approaches based on machine learning [xi] could be high, and due to their complexity, it is not practical for the Internet Service Providers (ISPs) to deploy them in their networks. W. Jiang & M. Gokhale [x] implemented Locality Sensitive Hashing (LSH) on FPGAs for real time traffic classification and Z. Li, R. Yuan & X. Guan [xii] used pattern recognition methods for traffic classifications. The researchers also worked on some other techniques for internet traffic classification like Moore, Andrew W, and Denis Zuev [v] used Bayesian analysis techniques for Internet traffic classification and they

achieved 90% accuracy on training data of different applications.

Most recent researches cover detailed working to avoid network congestion and bandwidth optimization that may not be available to different applications due to DDoS attack [xiii] or due to poor management of available bandwidth. Johnson & Christopher L [xiv] described key requirements for effective bandwidth management and implemented bandwidth management by executing the limits from socket layer to protocol layer for each application. In order to guarantee agreed service levels to business customers there are systems available to handle required content delivery and to handle differentiated business services [xv] and to ensure that agreed service levels are delivered to business customers without any major impact on their services. To handle differentiated services a plethora of different techniques including buffer management, packet scheduling etc have been worked out by different researchers and they have concluded over the time that traffic classification is the key feature of all QoS enabled architectures [iii], which are implemented by different multimedia and broadband service providers. Carbone, Marta, and Luigi Rizzo [xvi] studied detailed features of dummy net link emulator when operated under different operating systems and suggested the required operating conditions to run an emulator for the desired accuracy of results. Nussbaum, Lucas, and Olivier Richard [xvii] did comparative analysis of link emulators, but all such emulators were not tested for real time live environment where internet traffic and packet queuing is performed. But it is different from our work, in that authors have not computed the impact of traffic classification and prioritization on system performance. In this paper, we are trying to contribute in terms of evaluation of system and network performance and suggesting optimum operating conditions while optimizing the available bandwidth.

## III. RESEARCH METHODOLOGY & APPROACH

In FreeBSD, packets were read through ethernet interface and packet inspection was carried by using 4-tuple packet characteristics; i.e *source IP*, *source port*, *destination IP*, and *destination port*. The implementation block diagram is shown in Fig.1 and the detailed working of each block has been described as under

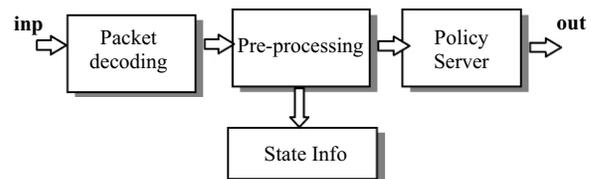


Fig.1. Implementation block diagram

A. Packet Decoding and Preprocessing

In FreeBSD operating system, IPFW (IP firewall) is user interface for controlling firewall; whereas the packets entered (depending on source and destination address) the firewall from different places in the protocol stack. The traffic which was passing through the firewall was compared against all the rules in the rule set according to the configured rule-number, order permitted and in the order of insertion of the packet. Once the match was found then the packet was treated according to the matching rule.

The incoming packets were read by **ether\_demux()** function and afterwards the traffic was passed to **ether\_input()** function for packet processing [xviii] to upper layers as shown in Fig. 2. The mapping of different OSI layers for **network code** in FreeBSD [xviii] is shown in Fig. 3 which shows how the traffic was captured from physical layer till the application layer process was invoked to handle the traffic. In this paper, we used the simplistic approach with lesser computations to capture and to identify the traffic and thus minimizing load on network resources.

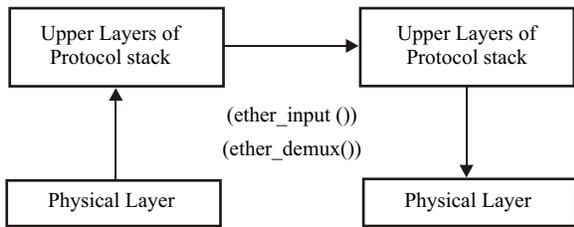


Fig. 2 Packet flow from physical layer to application layer

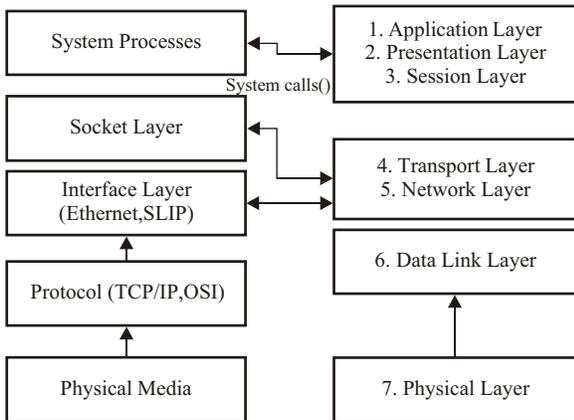


Fig. 3 FreeBSD-Network Code Mapping with OSI Layers

A. Traffic Policing

After the identification of different applications, they were passed from two objects i.e. pipe and a queue to rate limit and prioritize as per available bandwidth. The objects which were configured were pipes; the pipe control the link with certain delay and bandwidth, and

traffic was further passed to the scheduler; afterwards the packets were passed to the queue with configured queue size and loss rate. As packets arrived out of ipfw, they were transmitted over the configured outgoing link. The traffic flow diagram from different pipes has been shown in Fig. 4. The key variables for a pipe are

- queue size
- link bandwidth
- network end to end delay

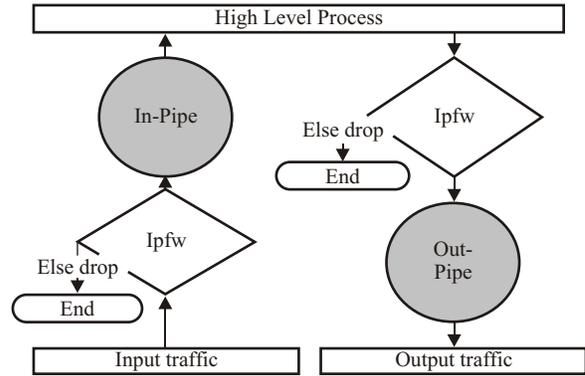


Fig. 4. Flow Diagram for In & Out traffic

IV. IMPLEMENTATION & SYSTEM DESIGN

The implementation network topology is shown in Fig. 5. The FreeBSD server has two network interfaces **fxp0** and **fxp1**. The **fxp0** was connecting inter network devices whereas **fxp1** was connected to the public internet. Internet traffic classification and prioritization has been done through FreeBSD server which was working as a policy server for internet traffic.

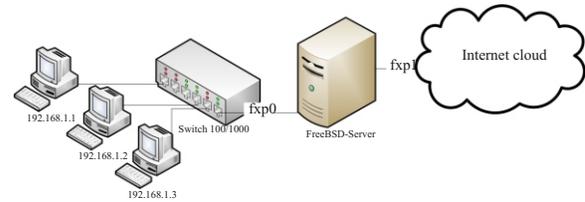


Fig. 5 Network Topology

Following options were enabled in FreeBSD kernel configuration to enable dummynet.

- IPFIREWALL ; to enable IP firewall
- IPFIREWALL\_VERBOSE; to enable firewall output
- IPFIREWALL\_VERBOSE\_LIMIT; to limit firewall Output
- DUMMynet ; to enable dummynet operation
- HZ ; to set the timer granularity

In order to enable firewall rules, following were added to **/etc/rc.conf** file, the code is mentioned in below Fig. 6.

```
# vi /etc/rc.conf
firewall_enable= "YES"
firewall_loggin= "YES"
firewall_script= "/usr/local/etc/ipfw.rules"
#ipfw.rules will contain the complete listing of
rules
; After saving the basic firewall configuration the
firewall services are started as under
# service ipfw start
```

Fig. 6. Code to enable firewall

A. Reading Input Traffic

The data at each interface (“fxp0” in our case) was being held at *ifnet* structures, which were connected in a linked list form and following were some of the key *ifnet* functions [xix] which were used to handle the internet traffic.

- if\_init* ; To initialize the interface (fxp0)
- if\_start* ; To initiate transmission of packets
- if\_output* ; To queue outgoing packets
- if\_ioctl* ; To Interface ioctl function

All incoming packets were read and handled by following functions

- a) *ether\_input()* & *ether\_demux ()* ; to read input traffic and to add or remove headers
- b) *ipntr-* Interrupt to identify nature of application as per defined priority

B. Pipe Configuration

In order to measure results, we configured pipe no 10 & 11 for the client machine 192.168.1.1. We configured following output rate and input rate for client machine 192.168.1.1.

- (1) pipe no. 10 was configured for maxim output rate of 2500 Kbps with delay of 5 ms.
- (2) pipe no. 11 was configured for maximum input rate of 512 Kbps with delay of 15 ms.

The configuration steps and code for client machine 192.168.1.1 has been mentioned in below Fig. 7.

```
# cd /etc
# vi ipfw.ruleset
ipfw add 00005 allow all from any to any via
fxp0
; allowing all traffic to inner network interface
# Output pipe 10 and input pipe 11 were
configured for outgoing and incoming traffic for
host 192.168.1.1
ipfw add 110 pipe 10 out 192.168.1.1
ipfw add 220 pipe 11 in 192.168.1.1
# client 192.168.1.1 was restricted to output rate
of 2.5Mbps and input rate of 0.5Mbps
ipfw pipe 10 config bw 2500Kbit/s delay 5ms
ipfw pipe 11 config bw 512Kbit/s delay 15ms
# save and exit file
:wq!
```

Fig.7. pipe configuration code for client 192.168.1.1

C. Network Applications and Prioritization

We implemented UDP traffic priority over TCP traffic using wf2q+ (Worst case weighted fair queuing). The results were measured by originating the test traffic from these machines. The prioritized traffic passed through the Interface “fxp0”, which was acting as a gateway for all the applications. The traffic prioritization code is shown in Fig.8.

```
# vi ipfw.ruleset
ipfw add 00017 sched 10 config type wf2q+
ipfw queue 5 config weight 20 sched 10
ipfw queue 6 config weight 10 sched 10
ipfw add 00018 queue 5 out proto udp
ipfw add 00019 queue 6 out proto tcp
# save and exit file
:wq!
```

Fig.8. Priority configuration for tcp and udp traffic

TCP and UDP test traffic was generated at 200 kilo packets per second (packet size as 64 bytes) from client 192.168.1.1 towards yahoo server (IP 76.13.28.70) and we used Wireshark (v.1.10.2) to capture internet traffic [xx-xxi], the Timestamp sent and its echo reply time moving average was used to measure RTT. The results are shown in Table IV.

V. RESULTS & PERFORMANCE ANALYSIS

In order to measure the results, TCP and UDP traffic was generated traffic from client machine 192.168.1.1; the system performance, i.e. CPU and memory utilization was measured by using *vmstat*. The packet sizes were kept 64 bytes and 256 bytes whereas the packet rate was varied from 10 Kilo packets per second to 400 Kilo packets per second. The results achieved are shown in Table II and Table III. The results show that when traffic classification was performed, the system utilization remained upto 50% but as we go above 500 kilo packets per seconds, the system utilization increased substantially, i.e. reaching upto 75%. The CPU utilization at different packet rates is shown in Fig. 9.

TABLE II  
CPU LOAD OF CORE 0 DEPENDING ON PACKET RATE,  
PACKET SIZE AND SAMPLING RATE

Packet Rate (kilo packets/s)	Core 0 CPU Load (%age) with 64 bytes packet size	Core 0 CPU Load (%age) with 256 bytes packet size
10	5	6
100	14	17
200	26	28
300	36	37
400	42	48

TABLE III  
MEMORY UTILIZATION DEPENDING ON PACKET RATE,  
PACKET SIZE AND SAMPLING RATE

Packet Rate (kilo packets/s)	Memory Utilization (%age) with 64 bytes packet size	Memory Utilization (%age) with 256 bytes packet size
10	5	7
100	15	19
200	23	29
300	37	39
400	44	54

Looking at the CPU utilization, 500 kilo packets per second is the maximum supported packet rate while working under current system conditions. Similarly, the memory utilization also increased as packet size was increased and it is shown in Fig.10. It was observed that memory utilization was slightly higher as compared to the CPU utilization due to queuing

implementation effect. The system performance results show that FreeBSD (v8.1) dummynet can be a good candidate to be part of multi-service traffic classification system but its major limitation is to filter out encrypted traffic with maximum packet rate of 500 packets per seconds. The results measurements were taken on an Intel 3.0 GHz machine with 4 GB RAM and in order to analyze queuing impact, the traffic priority was implemented for TCP and UDP traffic.

The bandwidth throttling and optimization results for client 192.168.1.1 are shown in Fig.11 and Fig.12, the client 192.168.1.1 traffic was passed through configured pipe number 10 and 11. The traffic usage of client 192.168.1.1 was captured and is shown in Fig.11 and Fig.12. In Fig.11 and Fig.12, the X-axis is representing time (hours) and Y-axis is representing the download rate in bits per seconds (download rate). The Fig. 8 shows that the download rate of client 192.168.1.1 was reaching to 4Mbps and such peer to peer high download rate would impact other customer's experience.

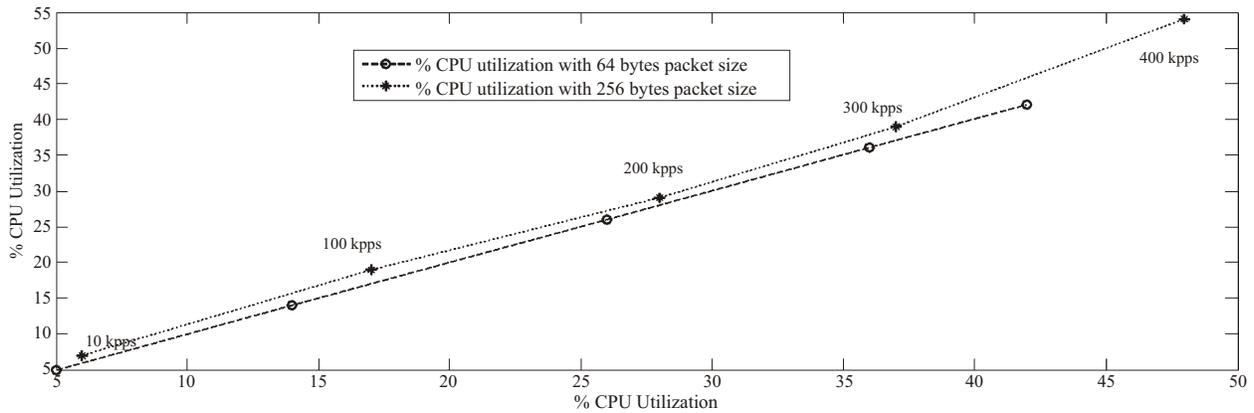


Fig. 9. CPU (Core 0) Utilization for packet size of 64 bytes and 256 bytes

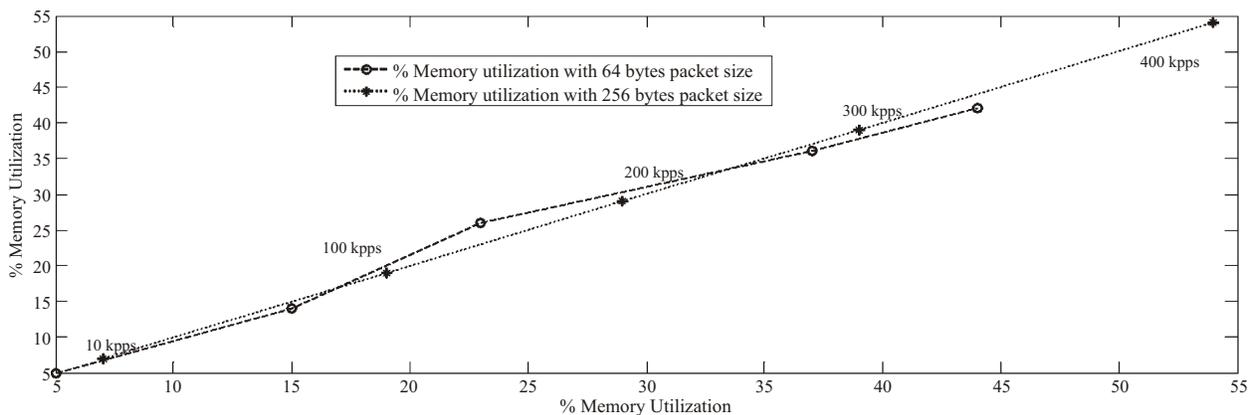


Fig. 10. Memory Utilization for packet size of 64 bytes and 256 bytes

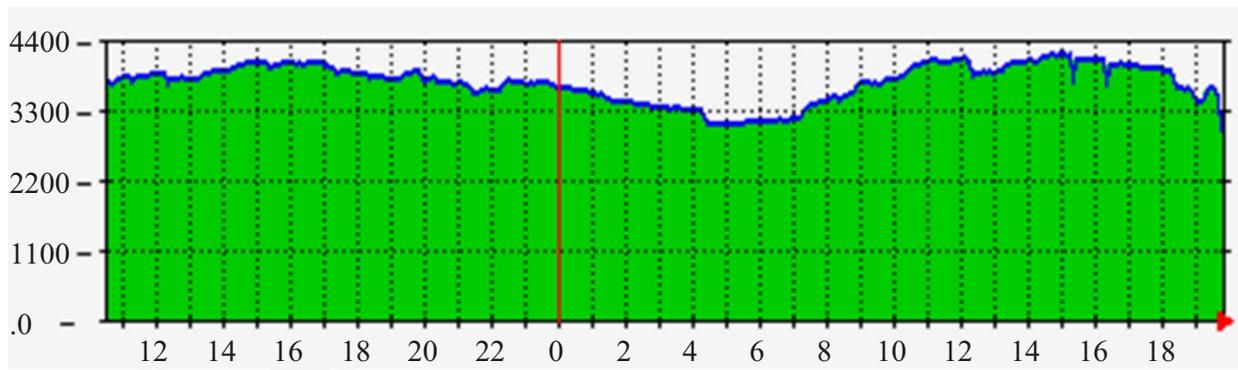


Fig. 11. Traffic pattern before applying rate limitation

The Fig.12 shows total traffic of client 192.168.1.1 during different hours of the day and it shows that whenever the traffic of client 192.168.1.1 tried to exceed 2500Kbps, it was rate limited to 2500Kbps for uniform allocation of available bandwidth. Thus bandwidth allocation was made uniform for all the clients. To test the traffic prioritization, TCP and UDP traffic was prioritized and their results are shown in Table IV. The results were quite encouraging with 10% higher throughput with 1% lesser packet loss was

achieved for UDP traffic in comparison to TCP traffic.

TABLE IV  
TRAFFIC PRIORITY IMPLEMENTATION

Application	TCP	UDP
Throughput (%age)	80	90
Packet Loss (%age)	3	2
Kpps (Kilo Packets per Seconds)	<b>200</b>	<b>200</b>

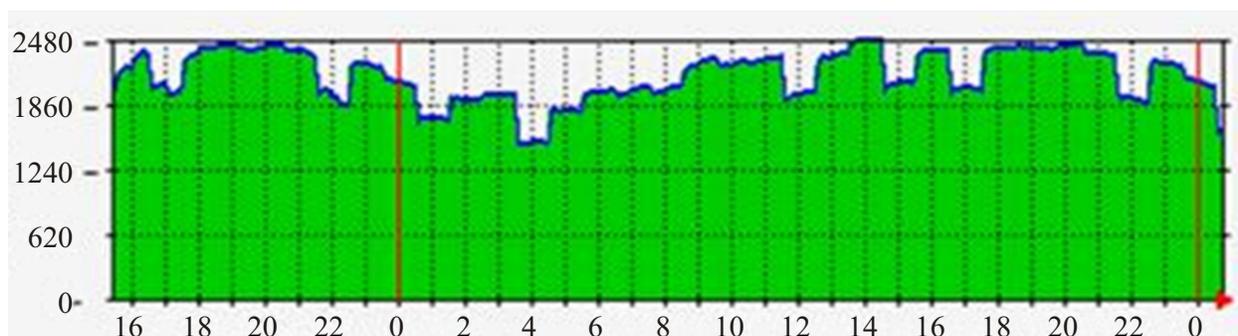


Fig. 12. Traffic pattern after applying rate limitation

## VI. CONCLUSIONS

In this paper we have presented an overview of system and network performance while implementing network bandwidth optimization using FreeBSD (v8.1) operating system. Though deep packet inspection is among the available solutions but for Internet Service Providers; solution complexity, cost, network load (utilization) and legal requirements are major constraints towards DPI (deep packet inspection) implementation. In addition to the default features of FreeBSD (v8.1) operating system, we were able to rate limit, classify and prioritize different network flows as per their nature and priority to ensure Quality of Service (QoS) for business critical applications. We presented different system level measurement results and showed the system behavior

by varying packet rate and packet size of different applications. Although FreeBSD (v8.1) is a mature operating system and is used by different mid-sized service providers for the optimization of network bandwidth, but still system and network performance under different set of operating conditions to get optimum results is a fertile research area.

This paper will help researchers to evaluate further FreeBSD operating system for the implementation of network security, which can be an area of further work for the researchers. Moreover, we plan to do extended comparison of our suggested technique against other classifiers.

## REFERENCES

- [i] Snort, <http://www.snort.org>, accessed 15<sup>th</sup>

- December 2014.
- [ii] Bro, <http://bro-ids.org>, accessed 20<sup>th</sup> December 2014.
- [iii] I. Stoica, Stateless Core, "A Scalable Approach for Quality of Service in the Internet", Doctoral Dissertation Competition, Winning Thesis of the 2001, *ACM*, vol. 2979, 2004. Available: <http://www.springer.com/computer/swe/book/978-3-540-21960-6>
- [iv] T. T. Nguyen and G. Armitage, "A Survey of Techniques for Internet Traffic Classification using Machine Learning", *IEEE, Communications Surveys & Tutorials*, vol. 10, no. 4, 2008, pp. 56-76. Available: [http://ieeexplore.ieee.org/xpls/abs\\_all.jsp?arnumber=4738466](http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=4738466)
- [v] W. Moore and D. Zuev, "Internet traffic classification using bayesian analysis technique", *ACM SIGMETRICS Performance Evaluation Review*, vol. 33, no. 1, 2005, pp. 50-60. Available: <http://dl.acm.org/citation.cfm?id=1064220>
- [vi] W. Li and W. Moore, "A machine learning approach for efficient traffic classification. Modeling, Analysis, and Simulation of Computer and Telecommunication Systems" MASCOTS'07. *IEEE 15<sup>th</sup> International Symposium*, 2007, pp. 310-317. Available: <http://www.ieeeexplore.com/xpl/articleDetails.jsp?tp=&arnumber=4674432>
- [vii] S. Zander, T. T. Nguyen and G. Armitage, "Automated traffic classification and application identification using machine learning", *IEEE, Conference on Local Computer Networks (LCN)*, 2005, pp. 250-257. Available: <http://www.ieeeexplore.com/stamp/stamp.jsp?tp=&arnumber=1550864>
- [viii] T. T. Nguyen, G. Armitage, P. Branch and S. Zander, "Timely and continuous machine-learning-based classification for interactive IP traffic", *Transactions on Networking (TON), IEEE/ACM*, vol. 20, no. 6, 2012, pp. 1880-1894. Available: <http://www.ieeeexplore.com/xpls/icp.jsp?arnumber=6164288>
- [ix] S. Sen, O. Spatscheck and D. Wang, "Accurate, scalable in-network identification of p2p traffic using application signatures", *ACM, Proceedings of the 13<sup>th</sup> international conference on World Wide Web*, 2004, pp. 512-521. Available: <http://dl.acm.org/citation.cfm?id=988742>
- [X] W. Jiang and M. Gokhale, "Real-time classification of multimedia traffic using fpga", *International Conference on Field Programmable Logic and Applications (FPL), IEEE*, 2010, pp. 56-63. Available: <http://www.ieeeexplore.com/xpls/icp.jsp?arnumber=5694221>
- [xi] S. Zander and G. Armitage, "Practical machine learning based multimedia traffic classification for distributed QoS management", *IEEE, 36<sup>th</sup> Conference on Local Computer Networks (LCN)*, 2011, pp. 399-406. Available: <http://www.ieeeexplore.com/xpls/icp.jsp?arnumber=6115322>
- [xii] Z. Li, R. Yuan and X. Guan, "Traffic classification-towards accurate real time network applications". *Human-Computer Interaction (HCI) Applications and Services, Springer Berlin Heidelberg*, 2007, pp. 67-76. Available: <http://link.springer.com/book/10.1007%2F978-3-540-73111-5>
- [xiii] R. Mahajan, S. M. Bellovin, S. Floyd, J. Ioannidis, V. Paxson, and S. Shenker, "Controlling high bandwidth aggregates in the network", *ACM, SIGCOMM Computer Communication Review* 32, no. 3, pp. 62-73. 2002. Available: <http://dl.acm.org/citation.cfm?id=571724>
- [xiv] L. Johnson, "Bandwidth management". *U. S. Patent 6, 820, 117*, 2004. Available: <http://www.google.com/patents/US6820117>
- [xv] N. Hartsell, "Systems and methods for providing differentiated business services in information management environments". *U.S. Patent Application 09, 879, 848*, 2001. Available: <http://www.google.com/patents/US20020049608>
- [xvi] M. Carbone and L. Rizzo, "Dummysnet revisited". *ACM, SIGCOMM Computer Communication Review* 40, no.2, 2010, pp 12-20. Available: <http://dl.acm.org/citation.cfm?id=1764876>
- [xvii] L. Nussbaum and O. Richard, "A comparative study of network link emulators". Society for Computer Simulation International. *In Proceedings of the 2009 Spring Simulation Multiconference*, 2009, p. 85. Available: <http://dl.acm.org/citation.cfm?id=1639898>
- [xviii] <http://www.manpages.info/freebsd/ipfw.8.html>, accessed November 2014
- [xix] <https://www.freebsd.org/doc/en/books/>

developers-handbook/book.html, accessed Nov 2014

[xx] C. Sanders, "Practical Packet Analysis, Using Wireshark to solve real-world network problems", 2011, No Starch Press.  
Available:  
<http://www.nostarch.com/packet2.htm>

[xxi] A. Orebaugh, G. Ramirez and J. Beale, "Wireshark & Ethereal network protocol analyzer toolkit", 2006, Syngress.  
Available:  
<http://www.sciencedirect.com/science/book/9781597490733>

<b>Authorship and Contribution Declaration</b>			
	<b>Author-s Full Name</b>	<b>Contribution to Paper</b>	
1	Mr. Naveed Akhtar	Proposed topic, basic study Design, Literature review, methodology and manuscript writing	
2	Prof. Dr. Muhammad Kamran	Referencing and quality insurer	