IJCSE International Journal of Computer Sciences and Engineering Open Access
Research Paper Volume-4, Issue-4 E-ISSN: 2347-2693

Impact of DDoS Attacks on Different Services Using Various AQM Techniques

Arshdeep Singh[1*], Lakhvinder Kaur[2] and Kulwinder Singh[3]

[1]*Department of Computer Science Engineering, Bhai Maha Singh College of Engg., Muktsar, India*
[2] *Department Electronics and Communication Engineering, Adesh Institute of Technology, Gharuan, India*
[3]*Department of Computer Application, Bhai Maha Singh College of Engg., Muktsar, India*

Available online at: www.ijcseonline.org

Received: Mar/28/2016 Revised: Apr /06/2016 Accepted: Apr/20/2016 Published: Apr/30/2016

***Abstract*—** With the rapid development of network technology, distributed denial of service (DDoS) attacks become one of the most important issues. Distributed Denial of Service (DDoS) attacks generates enormous packets by a large number of agents and can easily exhaust the computing and communication resources of a victim within a short period of time. So congestion control mechanism is one of the key that keeps any network efficient and reliable for the users. Many mechanisms were projected in the literature over theses years for the efficient control of congestion that occur in the network. Active Queue Management (AQM) is one such mechanism which provides better control in the recent years.
The focus of this work is to study the behaviors of various queuing algorithms such as Drop Tail, Fair Queuing (FQ), Stochastic Fair Queuing (SFQ), Deficit Round Robin (DRR) and Random Early Detection (RED) using ns-2 as a simulation environment.

***Keywords*—**DDoS, AQM, FQ, SFQ, DDR and RED

I. DDOS

The techniques of DDoS attacks have been evolved since these attacks have first appeared in June of 1998 [1]. The distributed denial of service (DDoS) attack is designed to overcome victims with traffic and prevent their network resources from working correctly for their legitimate clients. For attacking big adversary, a significant amount of bandwidth is required of DDos attacks, like Web-based Media Company, so they often command thousands of hosts in a botnet to simultaneously send traffic to a victim. This action has the cause of aggregating bandwidth to match the victim's network resources, as well as making particular host filtering complicated, since the attack is coming from so many places all at once. However, the general attack model and procedures were not changed. The first reported large-scale DDoS attack via the public Internet occurred in August 1999 on a network used by faculty and students at the University of Minnesota. The attack, which shut down the network for more than two days, was launched by 227 zombies, including 114 that were part of the high-speed, high ability Internet.

DDOS [1] is a more advanced form of DOS attack. The aim is to saturate communication links and target hosts with illegitimate data. This will cause links or target hosts to drop legitimate data or request due to lack of resources. In DDOS you have the following characters:

- Client – computer or person who launches attack.
- Handler – compromised computer running attacker programs. A handler can control many agents (zombies)
- Agent - compromised computer running attacker programs and is responsible for generating large amount of traffic towards target computer.

II. AQM

Queue management plays an important role in fair bandwidth allocation [2]. From the point of dropping packets, queue management can be classifying into two types. The first type is passive queue management (PQM), which does not drop packets until the buffer is full. Drop tail is a well known example of PQM. The second category is active queue management (AQM) [3] in which drops packets probabilistically before the buffer is full. In passive queue management, packets coming to a buffer are rejected only if there is no space in the buffer to store them; hence the senders have no earlier warning on the danger of growing congestion. The Drop-Tail router has the following problems:
The Lock Out phenomenon. Drop-Tail routers [3] may prevent some TCP connections from transiting Drop - Tail routers. This Lock Out phenomenon is frequently the result of synchronization of the TCP packet sending rate. An additional problem is Full Queues. Because Drop - Tail routers drop arriving packets when the buffer of a router is overflowing, the buffer of routers maintains full for the duration. This reason increases of the end-to-end packet delay.

To avoid this situation a new technique called Active Queue Management (AQM) implemented [4]. Active Queue Management (AQM) is most commonly used in wired networks, mainly in backbone routers where packet loss is due to network congestion [5]. Active Queue Management (AQM) is intended to achieve high link utilization with a low queuing delay. Active Queue Management (AQM) proposes to replace drop-tail queue management in order to improve network performance in terms of delay, link utilization, and packet loss rate and system fairness [6]. The mechanisms to solve the congestion problem at the intermediate nodes are called active queue management (AQM) algorithms.

A. Drop Tail

Due to the simplicity of the FIFO queuing mechanism, drop-tail [7] queues are the most widely used queuing mechanism in Internet routers today. Drop tail queuing method is by far the simplest approach to router queue management. Drop Tail, is a simple queue management algorithm used by Internet routers to decide when to drop packets. In Tail Drop all the traffic is not differentiated. Every packet is treated identically. With tail drop, when the queue is overflowing to its maximum capacity, the recently arriving packets are dropped until the queue has enough room to accept incoming traffic. The name arises from the effect of the policy on incoming datagrams. Once a queue has been full, the router begins discarding all other datagrams, thus dropping the tail of the sequence of datagrams. Drop tail are implicitly updated at the links and implicitly fed back to sources through end-to-end loss or delay, respectively Queues are used in routers to absorb transient bursts in incoming packet rates, allow the router enough time for packet transmission.

The Drop-Tail routers have the following problems:

1. The Lock Out phenomenon. Drop-Tail routers may prevent some TCP connections from transiting Drop Tail routers. This Lock Out phenomenon is frequently the result of synchronization of the TCP packet sending rate.
2. Full Queues. Because Drop-Tail routers drop arriving packets when the buffer of a router is full, the buffer of routers maintains full for the duration. This reason increases of the end-to-end packet delay.

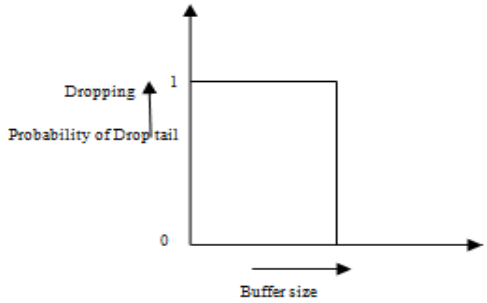

Fig. 2.1 Queue Size Of Drop Tail Algorithm

B. RED

RED (Early Detection Scheme) is proposed by Floyd and Jacobson for congestion avoidance in packet-switch networks. Random early detection (RED) [8], also known as random early discard or random early drop is an active queue management algorithm. It is as well as a congestion avoidance algorithm. Random Early Detection (RED) is a mechanism for active queue management that has been proposed to detect incipient congestion and is currently being deployed in the Internet backbone. RED drops packets based on the average queue length greater than a threshold, rather than only when the queue overflows. However, when RED drops packets before the queue really overflows, RED are not compulsory by memory limitations to discard the packet. It tries to avoid problems like global synchronization, lock-out, busty drops and queuing delay that exists in the traditional passive queue management i.e. Drop tail scheme[9]. Although the effectiveness of the RED gateway is fully dependent on a choice of its four control parameters, it is difficult to configure them. RED has four control parameters called minth, maxth, maxp, and qw . Minth and Maxth are the minimum and the maximum thresholds, respectively. These thresholds be used to determine a packet marking probability, according to which RED randomly drop an arriving packet.

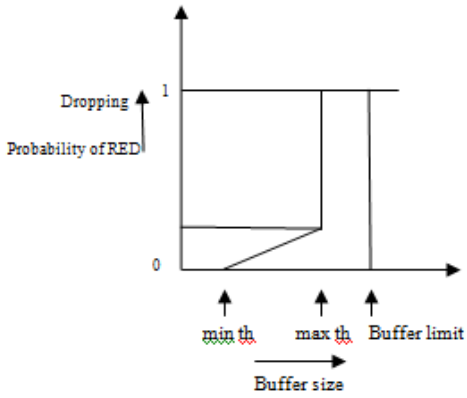

Fig. 2.2 Queue Size Of RED Algorithm

The template is used to format your paper and style the text. All margins, column widths, line spaces, and text fonts are prescribed; please do not alter them. You may note peculiarities. For example, the head margin in this template measures proportionately more than is customary. This measurement and others are deliberate, using specifications that anticipate your paper as one part of the entire proceedings, and not as an independent document. Please do not revise any of the current designations.

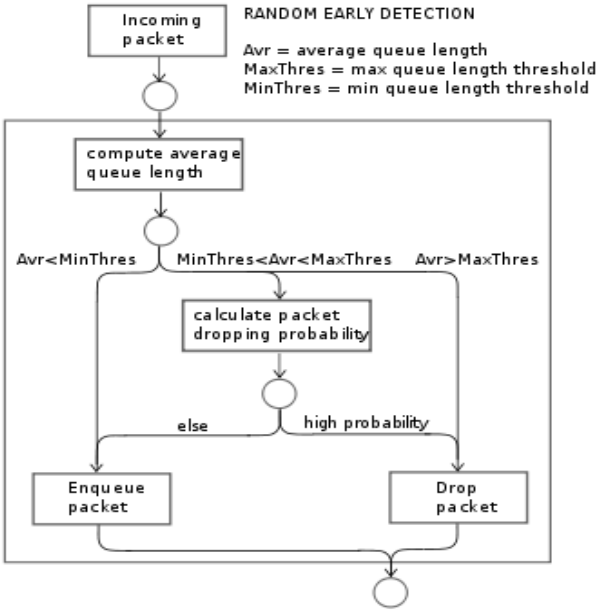

Fig. 2.3 Working Of Red Algorithm [9]

C. *Fair Queuing*

It was proposed by John Nagle in 1985, and has since been one of the most studied scheduling algorithms. Fair Queuing [10] algorithms for QoS-based resource allocation falls into two categories.

First is a class of scheduling algorithms for proportionate bandwidth allocation such as SFQ which guarantee weight-proportional throughput to clients by dividing up the server bandwidth fairly between them. The second class of scheduling algorithms are latency-sensitive in that both throughput and response time constraints may be independently specified provided certain capacity constraints are met. Fair queuing [11] is a scheduling algorithm used in computer and telecommunications networks to allow multiple packet flows to fairly share the link capacity.

D. *Stochastic Fair Queuing*

(SFQ) is similar to an SFB queue with only one level of bins. The biggest difference is that instead of having separate queues, SFB [11] uses the hash function for accounting purposes. SFB maintains accounting bins. The accounting bins are used to keep track of queue occupancy statistics of packets belonging to a particular bin. SFQ [12] by the dynamic allocation of FIFO queues which formed a queue for one conversation. The main advantage of SFQ is that it allows different programs to share an equal connection, and to avoid the bandwidth being occupied by a single client program.

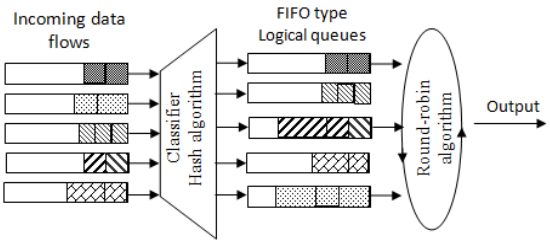

Fig. 2.4 SFQ Operations

E. *DDR*

DRR is a simple scheduling algorithm. The server in DRR [13] rotationally selects packets to send out from all flows that have queued packets. DRR maintains a service list to keep the flow sequence being served in a round and to avoid examining empty queues. If a flow has no packets in its queue, its identifier will be deleted from the service list. The next time a packet arrives to the flow that has an empty queue, the identifier of the flow will be added to the tail of the list. Deficit Round Robin uses three parameters, weight, Deficit Counter and quantum.

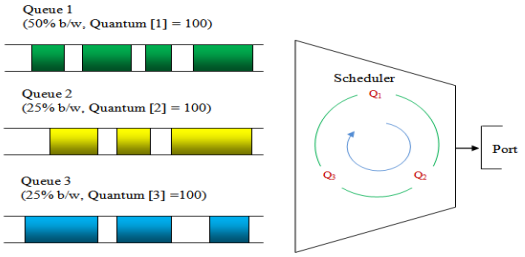

Fig. 2.5 Deficit Round Robin [14]

This queue mechanism used a well-designed idea to get better performance and can also be implemented in a cost effectiveness manner. It provides a basic framework to implement fair queuing efficiently. Although DRR [11] serves fine for throughput fairness, but when it comes to Latency bounds it performs rather poorly. Also it does not operate well for real time traffic. The queuing delays introduce through DRR can have exciting results on the congestion window sizes.

III. IMPLEMENTATION DETAILS

Experiment is performed in NS2.35 with the integration of AQM platform Simulation environment. The experimental platform is with Sony PC, the system of core linux10.0, the CPU of Intel Core i3 1.83GHz, hard disk of 320GB, RAM 4GB and integrated network card. The network structure for attacking simulation is indicated in Fig. 3.1.This scenario is created with 11 nodes out of which 4 are attackers, that are node 7, 8, 9, 10 rests are normal users which are node 1, 2, 3, 4, 5, 6 and node 0 is a target node on which attackers performed attack. In simulation normal user starts sending data or packet at 0 second and attacker nodes start performing attack after the simulation time of 60 second

and ends at 100 second. Overall simulation is performed for 120 seconds. In this simulation parameters are as follow, bandwidth used in the network is 10 Mbps except bottleneck link, delay between the nodes is 2ms except the bottleneck link, delay and bandwidth of bottleneck link is 5ms and 1Mbps respectively. Bottle neck queue size is 100 packets.

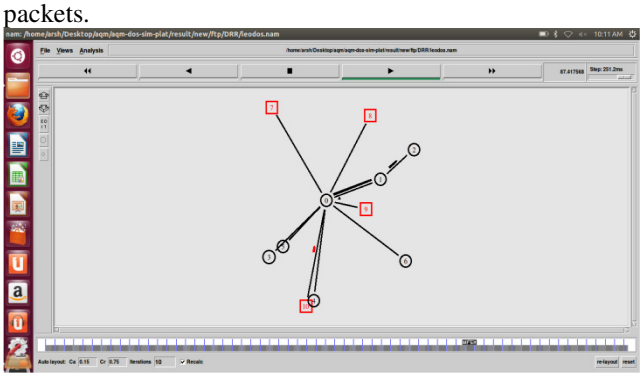

Fig. 3.1 Simulation Scenarios

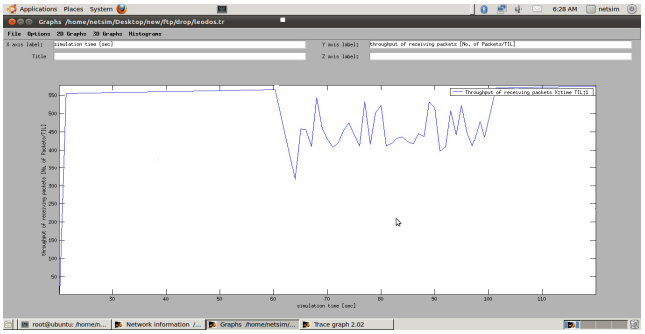

Fig. 3.2 Drop Tail Algorithm with FTP Application Over TCP Flow.

In Fig. 3.2 the graph shows the throughput of receiving packets at target node 0. Drop Tail Algorithm is used for this simulation using TCP flow and FTP application. During this experiment total no. of packet sent 38564, dropped 5005, average packet size 374.

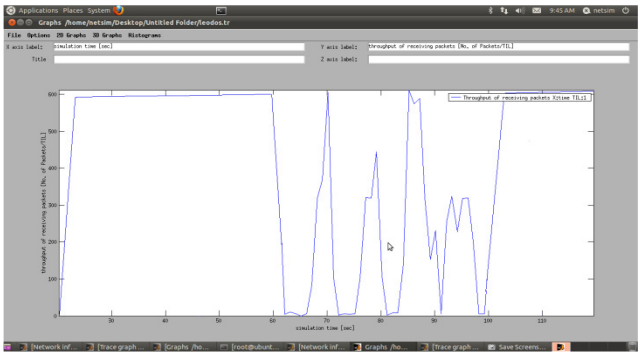

Fig. 3.3 Drop Tail Algorithm with TELNET Application Over TCP Flow

In Fig. 3.3 the graph shows the throughput of receiving packets at target node 0. Drop Tail Algorithm is used for this simulation using TCP flow and TELNET application. During this experiment total no. of packet sent 8272, dropped 63, average packet size 231.

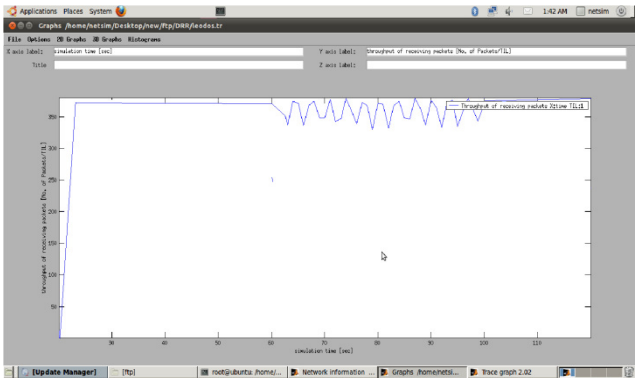

Fig. 3.4 DRR Algorithm with FTP Application over TCP Flow

In Fig.3.4 the graph shows the throughput of receiving packets at target node 0. DRR Algorithm is used for this simulation using TCP flow and FTP application. During this experiment total no. of packet sent 36848, dropped 8002, average packet size 424.

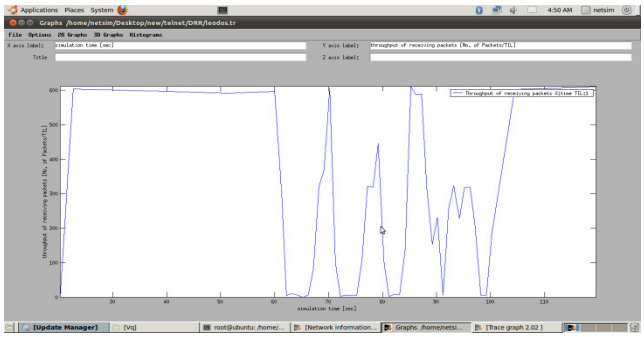

Fig. 3.5 DRR Algorithm with TELNET Application Over TCP Flow

In Fig.3.5 the graph shows the throughput of receiving packets at target node 0. DRR Algorithm is used for this simulation using TCP flow and TELNET application. During this experiment total no. of packet sent 8278, dropped 54, average packet size 231.

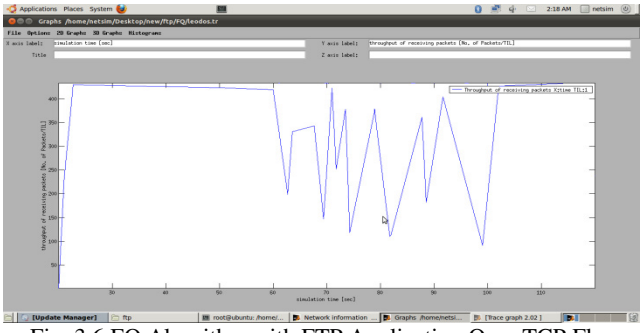

Fig. 3.6 FQ Algorithm with FTP Application Over TCP Flow

In Fig. 3.6 the graph shows the throughput of receiving packets at target node 0. FQ Algorithm is used for this simulation using TCP flow and FTP application. During this experiment total no. of packet sent 69387, dropped 37573, average packet size 325.

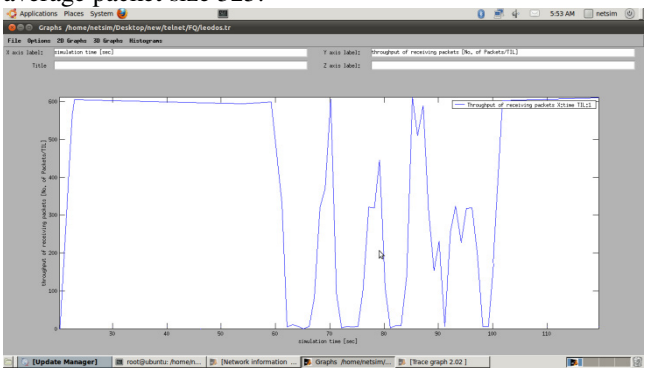

Fig. 3.7 FQ Algorithm with TELNET Application Over TCP Flow

In Fig. 3.7 the graph shows the throughput of receiving packets at target node 0. FQ Algorithm is used for this simulation using TCP flow and TELNET application. During this experiment total no. of packet sent 8272, dropped 134, average packet size 231.

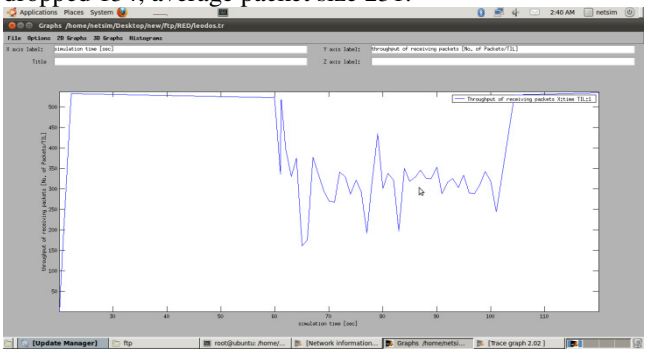

Fig. 3.8 RED Algorithms with FTP Application over TCP Flow

In Fig. 3.8 the graph shows the throughput of receiving packets at target node 0. RED Algorithm is used for this simulation using TCP flow and FTP application. During this experiment total no. of packet sent 29604, dropped 2546, average packet size 459.

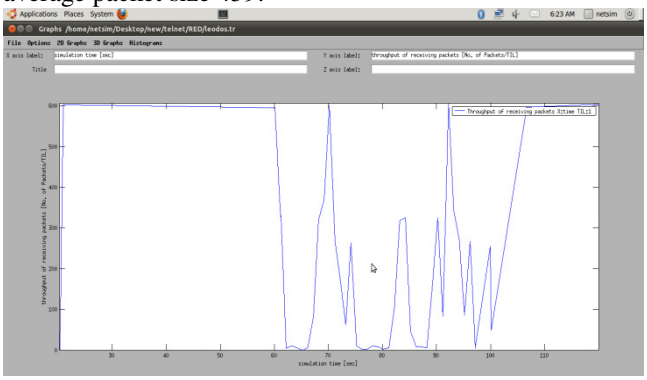

Fig. 3.9 RED Algorithms with TELNET Application Over TCP Flow

In Fig. 3.9 the graph shows the throughput of receiving packets at target node 0. RED Algorithm is used for this simulation using TCP flow and TELNET application. During this experiment total no. of packet sent 6110, dropped 31, average packet size 241.

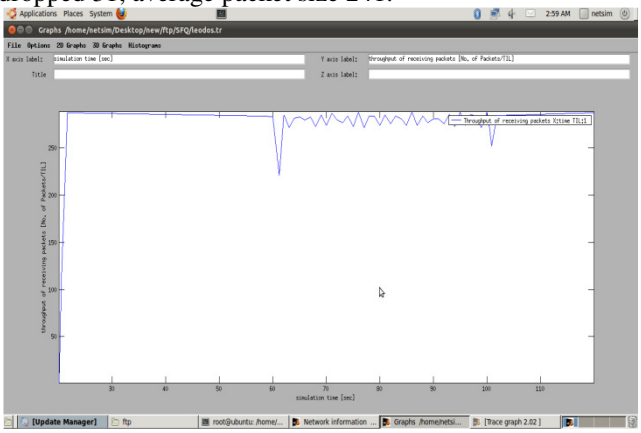

Fig. 3.10 SFQ Algorithm with FTP Application Over TCP Flow

In Fig. 3.10 the graph shows the throughput of receiving packets at target node 0. SFQ Algorithm is used for this simulation using TCP flow and FTP application. During this experiment total no. of packet sent 42726, dropped 17052, average packet size 432.

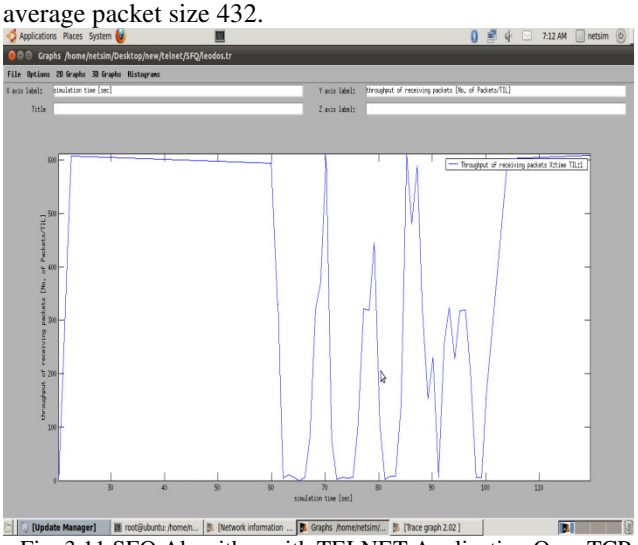

Fig. 3.11 SFQ Algorithm with TELNET Application Over TCP Flow

In Fig. 3.11 the graph shows the throughput of receiving packets at target node 0. SFQ Algorithm is used for this simulation using TCP flow and TELNET application. During this experiment total no. of packet sent 8269, dropped 182, average packet size 231.

ALGO	No. of sent packets		No. of dropped Packets		Dropping probability		Average end to end delay		Average packet size	
Application →	FTP	TELNET	FTP	TELNET	FTP	TELNET	FTP	TELNET	FTP	TELNET
Drop Tail	385564	82722	50005	63	12.997	.776	.22234	.02775	374.45	231.35
DRR	368448	82778	80002	54	21.711	.65	.09443	.02990	424.41	231.31
FQ	693887	82722	375573	134	54.115	1..62	.20007	.02221	325.43	231.47
RED	296004	61110	25546	31	8.600	.50	.06116	.01555	459.70	241.58
SFQ	427266	82669	170052	182	39.911	2..201	.09006	.01779	432.94	231.56

Table 3.1 Result analysis based on varies parameters

IV. CONCLUSION

In this section, we summarize the research presented in this thesis and present directions for future works. We discuss problems which AQM mechanisms have. While AQM mechanisms solve problems which conventional Drop-Tail routers have, AQM mechanisms have several problems. We have analyzed the impact of TCP flow under different algorithm (Drop Tail, RED, DRR, FQ, and SFQ) variation on the behavior the packet loss probability which is varying during network simulation. We use two different applications of TCP flow i.e. FTP and TELNET for comparative analysis of Active Queue Management algorithms. It is conclude that based on the dropping probability of packet Random Early Detection algorithm is better than all the rest four algorithms. It states that average packet size of Random Early Detection algorithm is larger than other algorithms. It reveals that average end to end delay of Random Early Detection algorithm is less than rest of all algorithms.

V. FUTURE SCOPE

1. The efficiency of RED may be considered for more improvement.
2. FTP and TELNET may be replaced with other applications of TCP/IP.
3. AQM's other algorithm may be used for better results.

REFERENCES

[1] K. Lee, J. Kim, K. H. Kwon, Y. Han, and S. Kim , "DDoS attack detection method using cluster analysis", Elsevier, Expert System with Applications, Vol.34, 2008, pp.1659-1665.

[2] M. Nabeshima, and K. Yata , “Performance improvement of active queue management with per-flow scheduling" ,IEEE Proc.-Commun., Vol. 152, No. 6, 2005.

[3] B. Braden, D. Clark, J. Crowcroft, B. Davie, S. Deering, D. Estrin, S. Floyd, V. Jacobson, G. Minshall, C. Partridge, L., Peterson, K. Ramakrishnan, S. Shenker, J. Wroclawski, and L. Zhang, "Recommendation on queue management and congestion avoidance in the Internet" . RFC2309, 1998.

[4] V. Santhi, and A. M. Natarajan, “A New Approach to Active Queue Management for TCP with ECN”, IEEE 2009 First International Conference on Advanced Computing, ISSN. 2377-6927, 2009, pp. 76-81.

[5] G. Pibiri, C. M. Goldrick, and M. Huggard , "Using Active Queue Management to Enhance Performance in IEEE802.11", 2009.

[6] Basant Kuamr Verma and Binod Kumar2, "An Improved Weighted Clustering for Ad-hoc Network Security New", International Journal of Computer Sciences and Engineering, Volume-03, Issue-03, Page No (51-55), Mar -2015, E-ISSN: 2347-2693

[7] Singh, Umesh Kumar, et al. "An Overview and Study of Security Issues & Challenges in Mobile Ad-hoc Networks (MANET)." International Journal of Computer Science and Information Security, Volume-9, Noi-4 (2011): 106-110.

[8] S. Floyd and V. Jacobson, “Random early detection gateways for congestion avoidance,” ACM/IEEE Transaction on Networking, Vol. 1, 1993, pp. 397-413.

[9] G. A. Ramachandra, R. Banu and G. F. A. Ahammed, “Analyzing Marking Mod RED Active Queue Management Scheme on TCP Applications”, International Conference on Information and Network Technology, Vol. 37, 2012, pp. 251-257.

[10] F. Lau, R. H. Stuart, and S. H. Michael,., "Distributed Denial of Service Attacks," in Proceedings of 2000 IEEE International Conference on Systems, Man, and

Cybernetics, Nashville, TN, Vol.3, 2000, pp.2275-2280.
[11] http://nms.csail.mit.edu/6.829-f06/lectures/bruce-queue.pdf
[12] M. Shreedhar, G. Varghese, E_cient Fair Queuing Using De_cit Round Robin, IEEE/ACM Transactions on Networking, Vol.4, No.3, June 1996.
[13] C. Semiria, "Supporting Differentiated Service Classes: Queue Scheduling Disciplines", Juniper Networks, Inc.
[14] S. Kumar, A. Bhandari, A. L. Sangal and K. K. Saluja, "Queuing Algorithms Performance against Buffer Size and Attack Intensities", Global Journal of Business Management and Information Technology. Volume 1, No. 2, 2011, pp. 141-157

AUTHORS PROFILE

Arshdeep Singh has received the Bachelor degree of Technology from Punjab Technical University, (Jalandhar) India in 2010 and Master of Technology from University College of Engineering, (Patiala) India in year 2013. He is working as Assistant Professor in Department of Computer Science and Engineering, BMSCE, Muktsar, Punjab since 2015. Before that he had performed duty as Assistant professor in AIT Chandigarh. His main research focus is on Computer Networking (DDoS). He has 3 years' experience in teaching and research.

Lakhvinder Kaur has received the Bachelor degree of Technology from Punjab Technical University, (Jalandhar) India in 2010 and Master of Technology from Yadwindra College of Engineering, (Patiala) India in year 2012. She is working as Assistant Professor in Department of Electronics and Communication Engineering, AIT, Chandigarh, India since 2013. Before that she had performed duty as Assistant professor in BHSBIET, Lehragaga, Punjab. Her main research focus is on VLSI. She has 4 years' experience in teaching and research.

Kulwinder Singh has received the Bachelor degree in Computer Applications from Punjab Technical University, (Jalandhar) India in 2007 and Master degree in Computer Applications from Punjab Technical University, (Jalandhar) India in year 2010. He is working as Assistant Professor in Department of Computer Science and Engineering, BMSCE, Muktsar, Punjab since 2011. He is a life time member of ISTE. His main research focus is on Computer Networking. He has 5 years' experience in teaching and 2 years in research.