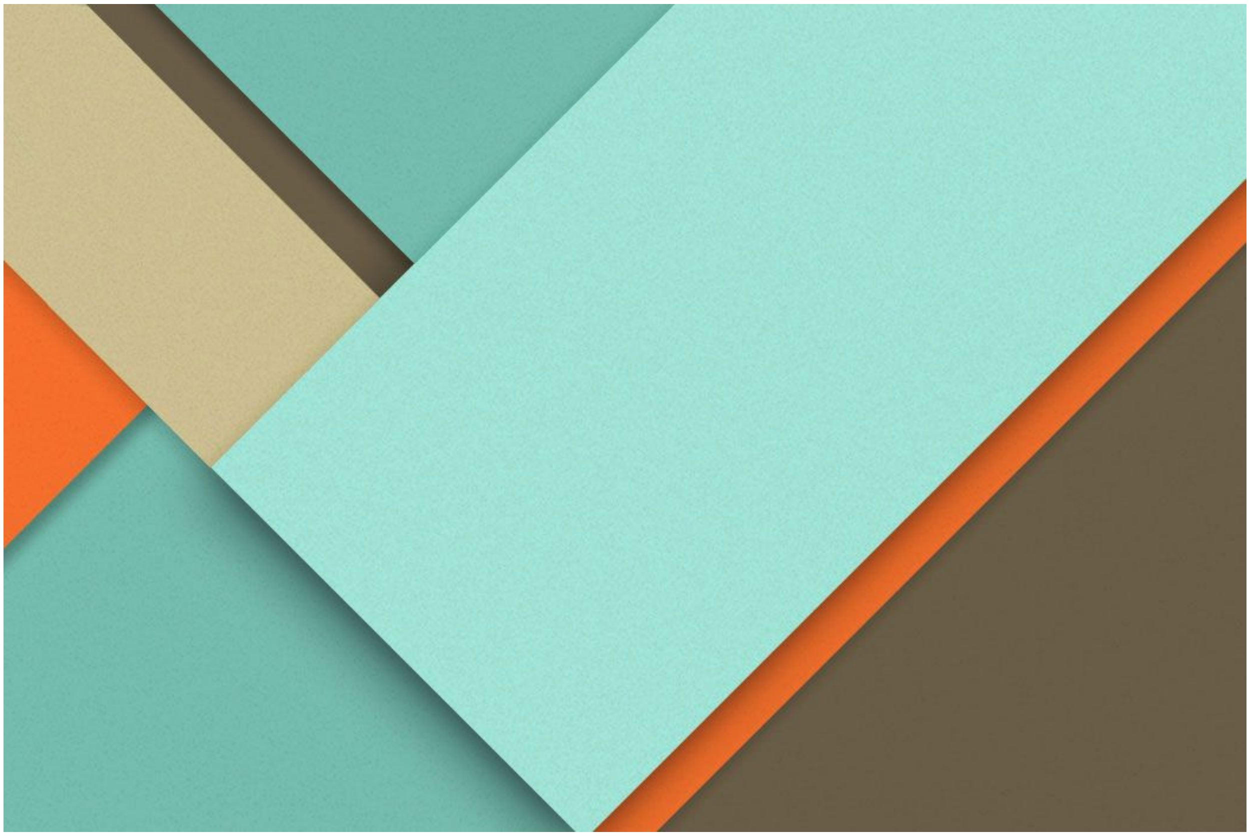


Comparing-Queue-Management-Techniques

Using NS2 simulator

NAMES -

Aman Agarwal

Dhanraj Sahu

Prashant

Prashant Meena

SIlky Priya

Introduction

In recent years, DDoS (distributed denial of service) attacks have become prevalent, and they are difficult to detect in networks. Because of their intimate, simple and effective characteristics, DDoS attacks are recognized as one of the main threats faced by network services. A queue management algorithm is important to guarantee the fair distribution of resources and quality of service (QoS) in the event of network congestion, and such an algorithm is also the first line of defense for a network when facing a DDoS attack. Consequently, the design of such algorithms is directly related to the network's ability to provide normal bandwidth allocation, latency, packet rate and other basic services. This paper analyzes three common queue management algorithms, Drop-Tail, RED, and REM, and then, in an ad hoc network environment, we conduct simulations using the network simulation software NS2 to compare the defense capabilities of these algorithms.

The Contribution and Application of Queue Management

Different from the passive queue management algorithm, the contribution of active queue management algorithm is to discard the packet before the queue is full, therefore we can regulate the discharge velocity of the original transmission end which could control congestion in order to avoid the negative effects like the long time end-to-end delay caused by the state of full queues or the reduction of utilization ratio and so on. Common active queue management algorithms include random early detection algorithm (RED) and random exponential marking algorithm (REM).

Queue Management Algorithms

The composition of a network is mainly based on the technology of store-and-forward and statistical multiplexing because data are inherently unexpected; thus, it is necessary to allow the transmission of unexpected data packets. The important role of the queue in a router is to absorb unexpected packets; a larger cohort can absorb more unexpected data and thus improve the throughput, but it also increases the data packet delay. Thus, we must manage the queue through the router to maintain a smaller queue length; therefore, a number of queue management algorithms have been generated.

Significance of Queue Management Algorithms

In contrast to passive queue management algorithms, active queue management algorithms discard packets before the queue is full to regulate the discharge velocity of the original transmission source. This approach helps control congestion to avoid other negative effects such as long end-to-end delays caused by full queues and a reduction of the utilization ratio. Common active queue management algorithms include the random early detection algorithm (RED) and the random exponential marking algorithm (REM).

Passive Queue Management Algorithms

Passive queue management algorithms, the traditional way to manage queue length, set a maximum length for each queue (in packets) and then accept packets into the queue until the queue length limit is reached. Thus, any subsequent packet that arrives will be rejected by the queue until the queue length has decrease. This technique includes the traditional “Drop-Tail” algorithm. Because packets must be dropped when the queue is full, this algorithm is called passive queue management. Drop-Tail queue management mechanisms are still the most widely used method for queuing and discarding network packets.

RED Queue Management Algorithm

The RED queue management algorithm uses the average queue length to predict impending network congestion and drops packets using random selection so that the transmission side, which has the function of congestion control, can be influenced to regulate the discharge velocity before congestion occurs, thus avoiding congestion.

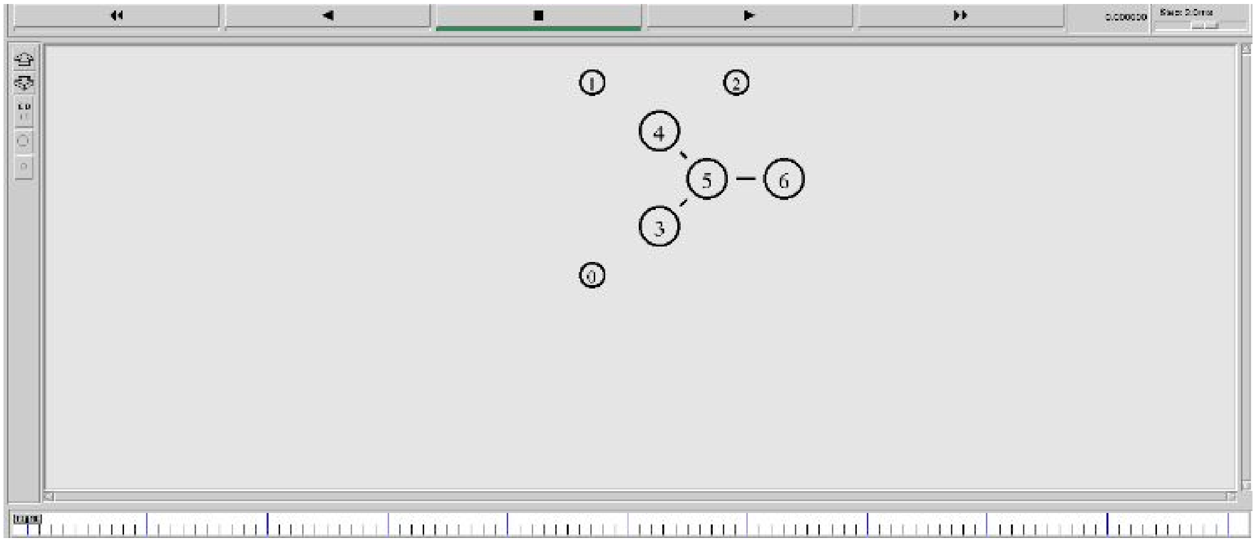
REM Queue Management Algorithm

The random exponential marking (REM) algorithm management mechanism stems from “optimal flow control” theory. The REM algorithm uses the concept of “price” from the network traffic optimization theory proposed by F. Kelly to detect and control the network’s congestion state. The change of “price” is determined by two factors: one is the difference between the actual queue length and the expected queue length, and the other is the difference between the input rate of aggregated flow and the output rate of the port. For the deepening algorithm of “price,” REM uses a simple proportional relationship. The sum of the probability of a single node and the “price” on the link shows an approximate proportional relationship. REM stabilizes the input rate to remain close to the link capacity and maintains the queue length within a small target range, regardless of the number of connections sharing the link, because the REM algorithm is based on the solution of gradient optimization. There are two core features for the load-based REM algorithm.

EXPERIMENT

Simulation of three Queue Management techniques – Tail Drop, RED and REM, for a mesh network. The results obtained are analysed over 3 parameters: Throughput, End-2-End delay and Packet Loss/Drop. The goal of the simulations performed using NS2, is to determine which Queueing mechanism, Taildrop or RED or REM, is better for managing queues and thereby more effective. These two mechanism are tried on a single network scheme and are compared on the basis of various performance parameters. RED and REM came out as a queue management technique with the target to outperform the performance of Tail Drop.

Network Topology and Simulation Scenario



Schematic of the network simulated A Mesh network with 3 wireless nodes, 3 static and 3 mobile nodes.Description of the different flows/packets/protocols Routing protocol : AODV Interface queue type : Droptail,RED.REM End to end link specs: 2Mb 10ms and 1.7Mb 20ms End to End protocol: TCP and UDP

Varying parameters

Protocol (Taildrop/RED/REM)

Sent Data Rate

Packet size

Drop Tail Simulation length in seconds: 9.77363029 Number of nodes: 7 Number of sending nodes: 7 Number of receiving nodes: 4 Number of generated packets: 640 Number of sent packets: 316 Number of forwarded packets: 289 Number of dropped packets: 326 Number of lost packets: 4 Minimal packet size: 32 Maximal packet size: 1040 Average packet size: 870.7948 Number of sent bytes: 246072 Number of forwarded bytes: 244808 Number of dropped bytes: 330808 Packets dropping nodes: 0 1 2 5

RED Simulation length in seconds: 5.321942771 Number of nodes: 7 Number of sending nodes: 7 Number of receiving nodes: 4 Number of generated packets: 619 Number of sent packets: 285 Number of forwarded packets: 266 Number of dropped packets: 336 Number of lost packets: 1 Minimal packet size: 40 Maximal packet size: 1040 Average packet size: 875.2949 Number of sent bytes: 223204 Number of forwarded bytes: 222304 Number of dropped bytes: 339796 Packets dropping nodes: 0 1 2

REM Simulation length in seconds: 4.5628372 Number of nodes: 7 Number of sending nodes: 7 Number of receiving nodes: 4 Number of generated packets: 602 Number of sent packets: 265 Number of forwarded packets: 254 Number of dropped packets: 346 Number of lost packets: 1 Minimal packet size: 42 Maximal packet size: 1040 Average packet size: 879.2949 Number of sent bytes: 200204 Number of forwarded bytes: 200304 Number of dropped bytes: 387796 Packets dropping nodes: 0 1 2.

CONCLUSION

Compared three types of traditional network queue management algorithms, Drop-Tail, RED and REM. Using the network simulation software NS2 in an ad hoc network environment; we performed a simulation to compare the packet rate etc. The results showed that the active queue management algorithms, such as REM and RED, exhibited stronger capabilities than the passive queue management algorithm Drop-Tail; however, under large-scale DDoS attacks, all three algorithms exhibited insufficient defense capabilities.