A Dot to DDoS Attack on Cloud Computing Environment Using Adaptive WRED Congesiton Control Algorithm

David L
Department of Electronics and Communication
National Institute of Technology, Hamirpur
Hamirpur (H.P), India.
E-mail: thesisdavid@gmail.com

Ashok Kumar
Department of Electronics and Communication
National Institute of Technology, Hamirpur
Hamirpur (H.P), India.
E-mail: ashok.nithamirpur@gmail.com

***Abstract*— As companies increasingly use virtualized data centers and cloud services, new weaknesses have opened up in enterprise infrastructure. At the same time, denial-of-service attacks (DoS) are moving from brute-force floods of data to more skillful attacks on application infrastructure. The combination is increasingly threatening for the companies that are placing critical business data outside their facilities, leaving their business reliant on continuing communications. In addition, with multi-tenant services becoming more common, attacks aimed at one company could dramatically impact the services of an unrelated, but co-located, firm. This paper deals with the use of Adaptive WRed, a congestion control queuing behavior to put an end to the DDoS attacks on the cloud environment.**

Keywords- cloud computing; DDoS; RED; WRED;

I. INTRODUCTION

Cloud computing is a model for enabling convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction. This cloud model promotes availability and is composed of five essential characteristics, three service models, and four deployment models.

- On-demand self-service
- Ubiquitous network access
- Resource pooling
  - Location independence
  - Homogeneity
- Rapid elasticity
- Measured service

Amazon EC2 customers recently suffered from a concerted Distributed Denial of Service (DDoS) [1] attack that caused some consternation for the web-based code hosting service Bitbucket (news courtesy of my favorite IT tabloid, The Register). An unfortunate fact of life about the massive DDoS such as Bitbucket appears to have suffered is that there is no defense once the incoming network pipes are full other than shutting off the DDoS. Trend Micro has to wrestle with DDoS attacks as part of our antivirus business as well as our hosted security business.

Most network countermeasures cannot protect against DDoS attacks as they cannot stop the deluge of traffic and typically cannot distinguish good content from bad. Intrusion Prevention Systems (IPS) are effective if the attacks are identified and have pre-existing signatures but are ineffective if there is legitimate content with bad intentions. Similarly, firewalls typically have simple rules that allow or deny protocols, ports or IP addresses. DDoS attacks easily bypass firewalls and IPS devices since they are designed to send legitimate traffic, such as HTTP requests to a web server, and attacks generate so much traffic from so many distinct hosts that a server, or more often its internet connection, cannot handle the traffic. Cloud computing is a great stuff, but enterprises & application architects need to think carefully about security before flying into the cloud. Because the cloud service is exposed to the outside world, the cloud infrastructure should support security functions such as intrusion detection and prevention, firewalling to prevent disallowed traffic, and *Denial of Service* (DoS) prevention. The cloud service is vulnerable to *Distributed Denial of Service* (DDoS) attacks—which can effectively choke its access lines, resulting in cloud users being locked out of the cloud service. Network-based DDoS prevention is a possible solution—with one of the techniques involving distribution of the cloud infrastructure to specific geographic areas and the ability to redirect cloud users in case of DDoS lockouts.

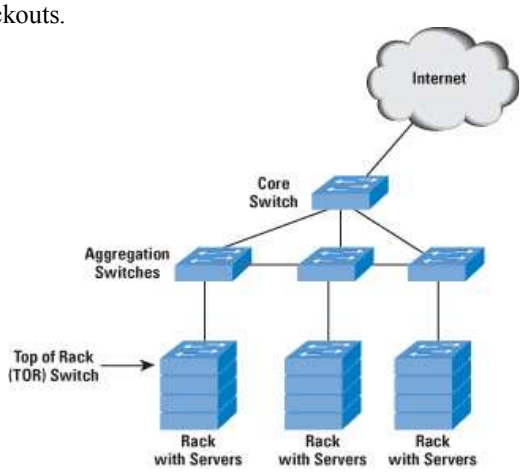

Fig: 1 Server and Switches in a Cloud

II. DISTRIBUTED DOS (DDOS) ATTACK

A distributed denial-of-service attack is composed of four elements. First, it involves a victim, i.e., the target host that has been chosen to receive the brunt of the attack. Second, it involves the presence of the attack daemon agents. These are agent programs that actually conduct the attack on the target victim. Attack daemons are usually deployed in host computers. These daemons affect both the target and the host computers. The task of deploying these attack daemons requires the attacker to gain access and infiltrate the host computers. The third component of a distributed denial of service attack is the control master program. Its task is to coordinate the attack. Finally, there is the real attacker, the mastermind behind the attack. By using a control master program [5], the real attacker can stay behind the scenes of the attack. The core switch which is having the main connection from the internet cannot be able to handle the traffic when it is DoS attacked. Hence the Rack servers in the cloud environment won't be able to get the internet services from the provider and hence it affects the customers under that particular cloud environment.

III. RANDOM EARLY DETECTECTION (RED)

Traditionally, a Tail Drop (TD) scheme is used in most of the routers for queue management. Packets will only be dropped when there is no more buffer space for the packets to be enqueued into the queue. This can lead to global synchronization and lock-out problems. Hence to avoid these problems RED[3] is used. Random early detection (RED), also known as random early discard or random early drop is an active queue management (AQM) algorithm. It is also a congestion avoidance algorithm. In the traditional tail drop algorithm, a router or other network component buffers as many packets as it can, and simply drops the ones it cannot buffer. If buffers are constantly full, the network is congested. Tail drop distributes buffer space unfairly among traffic flows (as the number of packets lost is proportional to the number sent - irrespective of their size). Tail drop can also lead to TCP global synchronization as all TCP connections "hold back" simultaneously, and then step forward simultaneously. Networks become under-utilized and flooded by turns. RED addresses these issues. It monitors the average queue size and drops (or marks when used in conjunction with ECN) packets based on statistical probabilities. If the buffer is almost empty, all incoming packets are accepted. As the queue grows, the probability for dropping an incoming packet grows too. When the buffer is full, the probability has reached 1 and all incoming packets are dropped.

RED is more fair than tail drop, in the sense that it does not possess a bias against bursty traffic [4] that uses only a small portion of the bandwidth. The more a host transmits, the more likely it is that its packets are dropped as the probability of a host's packet being dropped is proportional to the amount of data it has in a queue. Early detection helps avoid TCP global synchronization. RED (Random Early Detection) is a congestion avoidance algorithm that can be implemented in routers. The basic queue algorithm for routers is known as Drop Tail. Drop Tail queues simply accept any packet that arrives when there is sufficient buffer space and drop any packet that arrives when there is insufficient buffer space. RED gateways instead attempt to detect incipit congestion by computing a weighted average queue size, since a sustained long queue is a sign of network congestion.

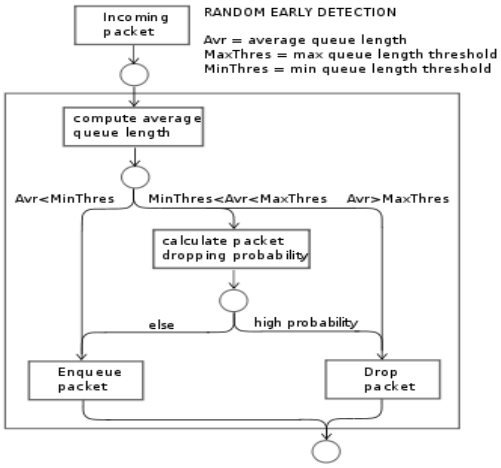

Fig 2: Dropping Probability of RED

Upon packet arrival, a RED gateway checks the weighted average queue size against specified minimum and maximum thresholds. If there is congestion, it notifies, either by dropping a packet or by setting a bit in a header field of the packet, probabilistically. For a RED gateway that drops packets, rather than marking a congestion bit, the following three phases sum up its algorithm:

Phase1: Normal Operation

If the average queue size is less than the minimum threshold, no packets are dropped.

Phase2: Congestion Avoidance

If the average queue size is between the minimum and maximum thresholds, packets are dropped with a certain probability. This probability is a function of the average queue size, so that larger queues lead to higher drop probabilities.

Phase3: Congestion Control

If the average queue size is greater than the maximum threshold, all incoming packets are dropped.

IV. WEIGHTED RANDOM EARLY DROP (WRED)

WRED is a network congestion control algorithm which follows the IP precedence [6] rule for dropping the packets.

This is the only difference between (Random Early Drop) RED and WRED. The packets in the buffer are dropped down based on their priority. The IP precedence is chosen by the WRED algorithm implemented in the router.

WRED configuration in the routers will drop the packets for low priority traffic. That is, the packets with low priority are the packet which just entered the queue. When the packet is dropped it will inform the source which sent the particular packet. Hence the source can resend the particular data packet again. Generally Random Early Drop (RED) basically deals with dropping of

the packets before the buffer is full. The basic idea is that one should not wait until the buffer is full in order to drop the packets. When RED is not configured, output buffers fill during periods of congestion. When the buffers are full, tail drop occurs; all additional packets are dropped. Since the packets are dropped all at once, global synchronization [7] of TCP hosts can occur as multiple TCP hosts reduce their transmission rates. The congestion clears, and the TCP hosts increase their transmissions rates, resulting in waves of congestion followed by periods where the transmission link is not fully used. RED reduces the chances of tail drop by selectively dropping packets when the output the output interface begins to show signs of congestion. By dropping some packets early rather than waiting until the buffer is full in the link, RED avoids dropping large numbers of packets at once and minimizes the chances of global synchronization. Thus, RED allows the transmission line to be used fully at all times [8]. And also the main thing that is required is to drop the packets in the queue which are sent by the attacker. Hence changes have to be done in the router to drop exactly the attackers' packets

A. WRED Packet Dropping Probability

The idea behind WRED invention was to take full advantage of TCPs congestion control mechanism by eliminating buffer tail drops. Without congestion avoidance implementation, TCP synchronization may occur. This takes place when an interfaces output queue is full causing newly arriving packets to be dropped. As a consequence, all active TCP flows go into TCP slow start resulting in bandwidth underutilization. The packet drop probability is based on the minimum threshold, maximum threshold, and mark probability denominator.

When the average queue depth is above the minimum threshold, WRED starts dropping packets. When the average queue size is above the maximum threshold, all packets are dropped. The minimum threshold value should be set high enough to maximize the link utilization. Here minimum threshold is taken and acceptable as Standard and Premium minimum threshold. If the minimum threshold is too low, packets may be dropped unnecessarily, and the transmission link will not be fully used.

The difference between the maximum threshold and the minimum threshold should be large enough to avoid global synchronization. If the difference is too small, many packets may be dropped at once, resulting in global synchronization.

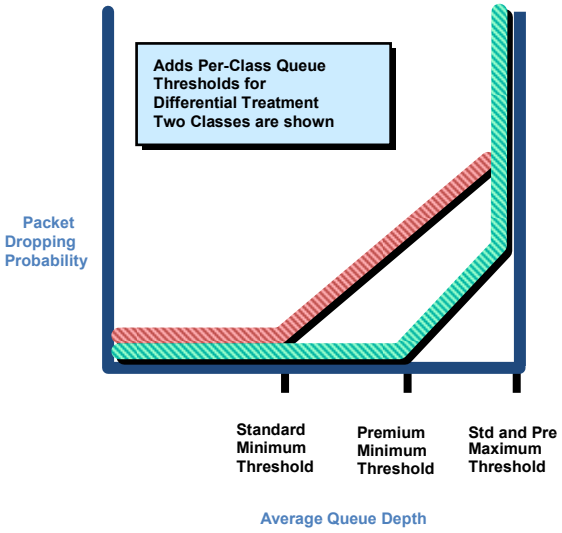

Fig: 3 WRED Dropping Probability

B. WRED with Weighted Fair Queuing

Before further to Weighted Fair Queuing, Fair Queuing (FQ) is to be introduced first. FQ is designed for fairly sharing network resources, which will try to reduce the delay and jitter of all traffics to their optimum levels. It has taken all the aspects into consideration, which has the following features:

- Different queues have fair opportunity of dispatching to equilibrate the delay of each stream on the whole.
- Short packets and long packets are treated fairly while dequeuing: if there are long packets in a queue and short packets in another queue waiting simultaneously to be sent out, the short packets should also be cared, and statistically the short packets should be treated preferentially, and the jitter between packets of every traffic will be reduced on the whole.

Compared with FQ, WFQ considers priority in addition when calculating the dispatching sequence of packets. Statistically, with WFQ, high priority traffic takes priority over low priority packets in dispatching. WFQ can automatically classify traffic according to the "session" information of traffic (protocol type, source/destination TCP or UDP port number, source/destination IP address, preference bits of ToS field, etc), and try to provide more queues so that each traffic will be equably put into different queues and equilibrate the delay of every traffic on a whole.

While dequeuing, WFQ can assign the bandwidth of egress interfaces occupied by each flow according to IP precedence.

The bigger the numerical value of the precedence is, the more bandwidth can be obtained. Because WFQ can balance the delay and jitter of every flow when congestion occurs; For instance, in the assured services using the RSVP (resource reservation protocol), generally, WFQ will be used as the dispatching policy. And also in GTS, WFQ is used to dispatch buffered packets. To drop the packets through comparing the length of the queue with the maximum/minimum limitations will treat the bursting data stream unfairly and influence the transmission of data stream. WRED uses the average queue and maximum/minimum limitations comparison to determine the dropping probability.

C. *Average Queue Length*

The average queue length is the result of low pass filtering of queue length. The average queue length reflects the changing of queue and is insensitive to bursting change of queue length, preventing the unfair treatment for the bursting data stream. When WFQ is adopted, you can set index, maximum limitation, minimum limitation, and packet-dropping probability when calculating average queue length for different queues that has different priorities. So packet with different priority will have different packet dropping characters.

When FIFO, PQ and CQ are adopted, you can set index, maximum limitation, minimum limitation, and packet-dropping probability when calculating average queue length for each queue. So packet with different priority will have different packet dropping characters.

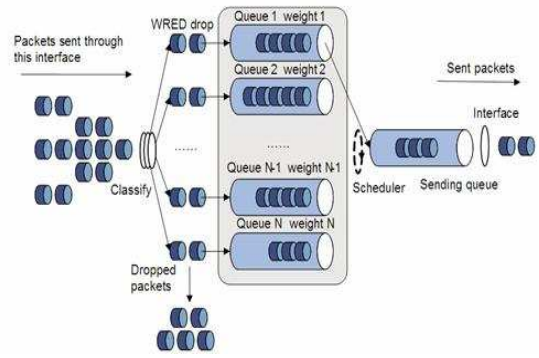

Fig: 4 Relation between WRED and queue mechanism

Associating WRED with WFQ, the flow-based WRED can be realized. Because different flow has its own queue during packet classification, the flow with small traffic always has a short queue length, so the packet dropping probability will be small. The flow with high traffic will have the longer queue length and will drop more packets, so we can protect the benefits of the flow with small traffic.

V. CONTROLLING CONGESTION WITH WRED

The congestion in the network can be controlled only by configuring the edge routers with WRED implications. The router commands [2] that are to be configured for WRED to drop the attacker's packet are given below.

random-detect precedence {*precedence* | **rsvp**} *min-threshold max-threshold mark-prob-denominator*

```
Router(config)#interface HSSI0/0

Router(config-if)#random-detect
Router(config-if)#random-detect precedence 7
40 50 100
```

The first argument after the *precedence* keyword here is the IP Precedence value. The options are any integer between 0 and 7, or the keyword *RSVP* [9]. After this are the minimum threshold, maximum threshold, and the so-called *mark probability* denominator. The minimum threshold is the number of packets that must be in the queue before the router starts to discard. The probability at the minimum threshold is essentially zero, but it rises linearly as the number of packets in the queue rises. The maximum probability occurs at the maximum threshold. We specify the actual value of the probability at this maximum by using the mark probability denominator. In this case we have set the value to 100, which means that, at the maximum, we will discard one packet in 100. This means that halfway between the maximum and minimum thresholds, the router will drop one packet in 200. The router doesn't necessarily drop packets when the queue depth reaches the minimum threshold. Rather, it uses a moving average so that temporary bursts of data are not dropped. This configured minimum is the lower limit of this moving average, which is reached only when the congestion continues for a longer period of time.

If we do not change these values, the defaults take IP Precedence values into account. The default mark probability denominator is 10, so the router will discard one packet in 10. The default maximum threshold depends on the speed of the interface and the router's capacity for buffering packets, but it is the same for all Precedence values. So, by default, the only differences between WRED's treatment of different IP Precedence levels is in the minimum threshold. The default minimum threshold for packets with an IP Precedence of 0 is 50 percent of the maximum threshold. This value rises linearly with Precedence so that the minimum threshold for Precedence 7 and packets with RSVP reserved bandwidth allocations are almost the same as the maximum threshold.

SIMULATION RESULTS

The comparison between the Droptail queue and the RED queue is made in NS-2. The simulation results shows that the droptail queue length during the time of 10 sec duration has larger values which represents that if the attacker's packet is filled in the transmission link, droptail can drop the packets when the link is full irrespective of the legitimate is full irrespective of legitimate of non-legitimate packets. If RED is used with IP precedence, then average queue length is maintained in the given link, and for dropping the attacker's packet, we can use the IP precedency of the WRED. Hence RED queue length shows better dropping probability compared to droptail and in addition to that WRED can be implemented using this RED queue results with IP precedence.

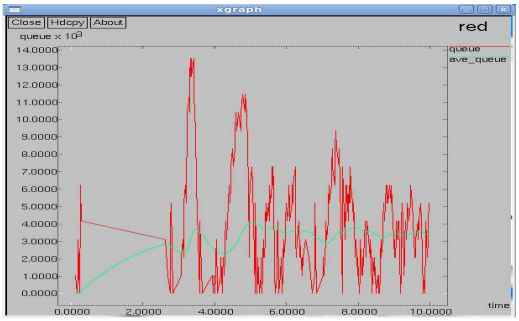

Fig: 5 Comparision of Average Queue Length of Drop tail queue and RED

CONCLUSION

Thus in order to prevent DDoS attacks in the Cloud Computing environment, the edge routers deployed in the network of the Date Centers in a cloud should be configured with Adaptive WRED that can be used for dropping the attacker's packet in the network in addition to that of Network Congestion Control. In order to detect the attacker in the network and to apply firewall to prevent the attacker in the cloud environment, WRED has to be adopted with the Access Control Lists (ACL) functionality of the routers. Hence by using the WRED behavior of the routers, we can stop the DDoS attack in a cloud computing environment.

REFERENCES

[1] Cisco Systems, Inc., "Defining strategies to protect against TCP SYN denial of service attacks," July 1999, http://www.cisco.com/en/US/tech/tk828/technologies_tech_note09186a00800f67d5.shtml

[2] Cisco white paper, "Distributed WRED", *https://www.cisco.com/en/US/docs/ios/11_1/feature/guide/WRED.html*, October 2009.

[3] L. Guan, I. Awan, M. Woodward, and X. Wang, "Discrete-time performance analysis of a congestion control mechanism based on RED under multi-class bursty and correlated traffic," *The Journal of System and Software*, vol. 80, no. 10, pp. 1716–1725, 2007..

[4] S. Floyd and V. Jacobson, "Random early detection gateways for congestion avoidance," *Networking, IEEE/ACM Transactions on*, vol.1, no. 4, pp. 397–413, Aug 1993.

[5] CERT® Coordination Center, "Results of the distributed systems intruder tools workshop," November 1999, http://www.cert.org/reports/dsit_workshop.pdf

[6] Cisco Systems, Inc., "Weighted Random Early Detection on Cisco 12000 Series Routers," http://www.cisco.com/en/US/docs/11_2/feature/guide/wred_gs.html.

[7] Chung-Hsin Liu, Po-Cheng Teng, Chun-Lin Lo, "The study of the Wireless Networks DoS Attacks". ICIS 2009, November 24-26, 2009 Seoul, Korea

[8] Xiao-Hui Lin, Kai-Yu Zhou, Hui Wang, Gong-Cao Su, "Scalable Fair Random Early Detection," Wireless Communications, Networking and Mobile Computing, 2006,WiCOM2006, International Conference on 22-24 Sept. 2006 Page(s):1 – 4

[9] Lixia Zhang, Stephen Deering, "RSVP: A New Reservation Protocol", *IEEE Network Magazine* September 1993, vol. 3, no.5.