Effective Traffic Control Scheme for Protecting Legitimate Traffic from Malicious Traffic on Internet

Gaeil Ahn, Kiyoung Kim, and Jongsoo Jang
Network Security Department
Electronics and Telecommunications Research Institute (ETRI)
161 Kajong-dong, Yusong-gu,Taejon, 305-350
Korea
{fogone, kykim, jsjang}@etri.re.kr

Abstract: - The greatest headache in the information-oriented society of today is security problem. This paper deals with Distributed Denial of Service (DDoS) attacks, which is executed by a malicious user with intention to prevent legitimate users of a service from using the desired resources by monopolizing network resource and resulting in network or system congestion. The existing queuing algorithms cannot solve this problem because they don't have any mechanism that distinguishes between legitimate traffic and malicious traffic. This paper proposes an effective traffic control scheme that can protect legitimate traffic from malicious traffic. The proposed scheme employs two kinds of queues, high-priority queue and low-priority queue. Our scheme can determine very quickly and correctly if network is congestion or not as well as which traffic is malicious by using traffic metering. According to the metering result, malicious traffic is served through low-priority queue and legitimate traffic is served through high-priority queue. To show our scheme's excellence, its performance is measured and compared with that of the existing queuing service through simulation.

Key-Words: - Distributed Denial of Service (DDoS), Network congestion, Protection of legitimate traffic, Malicious traffic, Traffic control

1 Introduction

The greatest headache in the information-oriented society of today is security problem. In this paper, we discuss Distributed Denial of Service (DDoS) attacks that are notorious for its destructive power on victim network and system. DDoS attack is executed by a malicious user with intention to prevent legitimate users of a service from using the desired resources by monopolizing network resource and resulting in network or system congestion [1].

Currently, malicious users execute DDoS attack by combining several well-known schemes such as SYN flooding, UDP flooding, ping of death [2-4]. Firstly, SYN flooding exploits the TCP three-way handshaking procedure. In SYN flooding, malicious user sends great number of SYN packets to a victim for the purpose of making bogus connection. SYN flooding results in the victim being unable to allow the connection request form legitimate users. UDP flooding and ping of death are based on UDP and ICMP protocol, respectively. Both schemes flood the victim, thus degrade the quality of service for legitimate user. When malicious user makes use of those schemes, he/she sometimes uses a faked source address in IP packet to hide his/her identity.

The defense solution of DDOS attack will be to block IP spoofing packet and to control malicious traffic. We don't address IP spoofing problem in this paper. Instead of it, we deal with traffic control problem.

We think the true defense of DDoS attack is to protect legitimate traffic as well as the victim. For example, in order to defeat the SYN flooding attack there may be a simple scheme that drops newly incoming all SYN packets when it receives too many TCP connection requests. Even if that simple scheme may protect the victim system/network, it can never protect the legitimate users/traffic. So, the SYN flooding attack is not failure but success.

There have been proposed queuing algorithms as traffic control scheme. These algorithms can not solve DDoS problem because they don't have any mechanism that distinguishes between legitimate traffic and malicious traffic.

This paper proposes an effective traffic control scheme that can protect legitimate traffic from malicious traffic. The proposed scheme employs two kinds of queue, high-priority queue and low-priority queue. Our scheme can determine very quickly and correctly if network is congestion or not as well as which traffic is malicious by using traffic metering. According to the metering result, malicious traffic is served through low-priority queue and legitimate traffic is served through high-priority queue. To concrete our scheme, we develop packet classification, traffic metering, and queue mapping mechanism.

The rest of this paper is organized as follows. Section 2 overviews the existing queuing algorithms. Section 3 illustrates the traffic control scheme proposed in this paper. In section 4, the performance of the proposed scheme is measured and compared with that of the existing queuing algorithms through simulation. Finally conclusion is given in Section 5.

2 Queuing Algorithms

DDoS attack monopolizes network resource, thus results in network congestion. DDoS traffic control problem can be thought of as the typical queuing discipline problem in network router. The core of the queuing discipline problem is to determine which packets get transmitted and which packets get discarded. There have been proposed several queuing algorithms such as FIFO (First-In-First-Out), FQ (Fair queuing), RED (Random Early Detection), and so on [5].

FIFO is called first-come-first-served queuing. FQ employs the algorithm that maintains a separate queue for each flow and services these queues in a round-robin manner. And RED uses the algorithm that mark or drop each arriving packet with some drop probability whenever queue length is greater than drop level.

Those queuing algorithms cannot be used as a solution for controlling malicious user's traffic. That is, FQ has a merit that a source cannot exceed its share of the networks capacity at the expense of other flow. But, the algorithm has a big problem in DDoS attack that the more increase the number of malicious user’s flows, the more decrease the legitimate user’s share of network resource because DDoS attacker can generate a large number of flows. The RED algorithm cannot solve DDoS problem either. RED has a merit that the more packets sent by a flow, the higher the chance that its packets will be selected for dropping. But, RED also has a disadvantage that the more increase the volume of malicious user's traffic, the higher the probability that legitimate user's packet will be dropped because DDoS attacker can generate a huge volume of traffic.

[6] recommended Class-based Queuing (CBQ) [7] as the queuing algorithm that can protect legitimate user from DDoS attack. Using CBQ require classification of traffic into each class. But they didn't handle the problem.

3 Traffic Control Scheme

3.1 Distinction between malicious and legitimate traffic

To protect legitimate traffic effectively, it should be a distinction between malicious traffic and legitimate traffic before everything else.

To solve this problem, we pay attention to two facts related to DDoS attack on Internet. The first fact is that it's easier for malicious user to hack systems on insecure networks than secure networks. This means that most of the compromised hosts to be used for DDoS attack will be on insecure networks. For such reasons, we propose source-traffic-trunk based metering to distinguish between malicious traffic and legitimate traffic. In this paper, source-traffic-trunk signifies the aggregate of flows that come from the same source network. Source-traffic-trunk based metering is more precise, lightweight, and flexible method than flow based metering. In flow based metering, it's almost impossible to distinguish between malicious flow and legitimate flow because there is little difference between both traffic volume in case of DDoS attacks.

The second and last fact is that DDoS attacker generates a huge volume of traffic without any consideration of network state. That is, malicious user generates heavy traffic and never decreases its transmission rate even if network congestion occurs. Legitimate user, on the other hand, has tendency to adapt its transmission rate to network state. For example, if legitimate users perceive that the response time of a Web site is very late, some of them will move to other site that provides a similar service to the Web service or give up accessing it. The relation between malicious traffic and legitimate traffic can be compared with that between UDP traffic and TCP traffic. That is, malicious traffic can be referred to as heavy and selfish traffic because a great quantity of traffic is generated regardless of network state. Legitimate traffic, on the other hand, can be referred to as obedient traffic because it is generated adjusting to network state.

For reason of that, we'll regard heavy and selfish source-traffic-trunk as traffic generated by malicious user. In the same manner, we'll regard obedient source-traffic-trunk as traffic generated by legitimate user. Table 1 shows the comparison between malicious and legitimate source-traffic-trunk by traffic property and quantity.

Table 1. Comparison between malicious and legitimate source-traffic-trunk (STT) by traffic property and quantity

Property Quantity	Selfish	Obedient
Heavy	Malicious STT	Legitimate STT
Light	Legitimate STT	Legitimate STT

In this paper, two kinds of queue are used: high-priority queue and low-priority queue. High-priority queue services packets regarded as legitimate traffic and low-priority queue packets regarded as malicious one.

3.2 Traffic Control Scheme

Fig. 1 shows the node architecture for traffic control scheme proposed in this paper. Our scheme is installed in front of the network/system to protect.

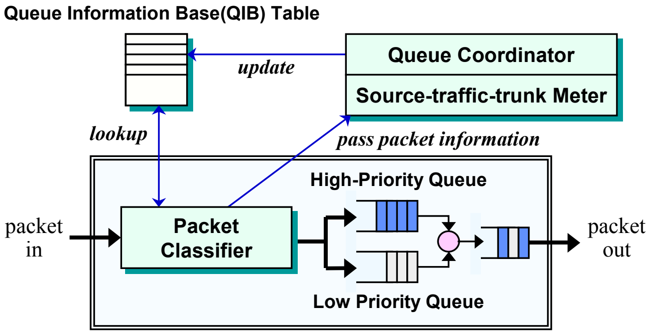

Fig. 1. Node architecture for traffic control scheme

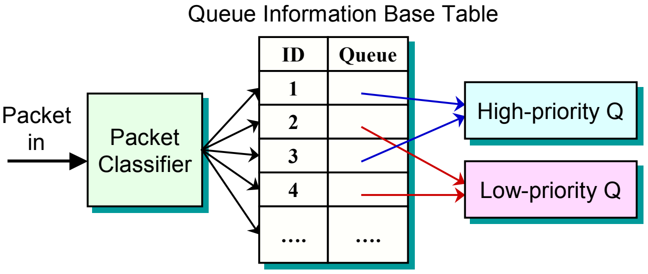

Fig. 2 Packet forwarding in our scheme

The node consists of three components: Packet-Classifier, Source-Traffic-Trunk Meter, and Queue-Coordinator. Firstly, Packet-Classifier is the component that classifies packets by their source network (i.e. the source IP address). It uses Queue Information Base (QIB) table to find the service queue for the incoming packet and then sends it to high or low priority queue according to the QIB table lookup result. Fig. 2 shows how packet is forwarded in our scheme. QIB table consists of several fields such as source-traffic-trunk-ID, service queue (i.e. high-priority or low-priority queue), and so on. Source-traffic-trunk-ID is the primary key of the QIB table and used to classify incoming packets by the source IP address prefix.

And Source-Traffic-Trunk Meter is the component that calculates the load (i.e. transmission rate) of the source-traffic-trunk corresponding to the incoming packet and the load of high-priority queue by using the information passed by Packet Classifier. Finally, Queue-Coordinator is the component that determines the queue of a source-traffic-trunk by using its load and high-priority queue's load, and then updates QIB table to reflect the result. Queue Coordinator is the core module of our scheme. Its purpose is to make malicious traffic served with low quality of service and legitimate traffic served with high quality of service. In next sub-section, we'll introduce the Queue-Coordinator in details.

3.3 Queue Coordinator

The purpose of Queue Coordinator component is to map a source-traffic-trunk to a high-priority or a low-priority queue according to the load of high-priority queue and the load of the source-traffic-trunk.

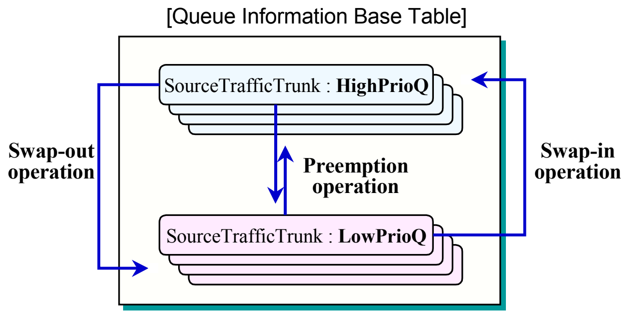
[Queue Information Base Table]
SourceTrafficTrunk : HighPrioQ
Swap-out operation
Preemption operation
Swap-in operation
SourceTrafficTrunk : LowPrioQ

Fig. 3. Three operations defined in Queue Coordinator component

The Queue Coordinator component has the following three operations as shown in Fig. 3: swap-out, swap-in, and preemption operations. First of all, the swap-out operates as follows. When a packet arrives, if its source-traffic-trunk uses high-priority queue and the load of the source-traffic-trunk is greater than *Permission_load*, QIB table is updated to set the queue of the source-traffic-trunk to low-priority. *Permission_load* means the maximum load that can be used by each source-traffic-trunk using high-priority queue. *Permission_load* is computed as

$$permission\_load = LinkBandwidth \, / \, high\_prio\_stt\_nb;$$

where *high_prio_stt_nb* indicates the number of source-traffic-trunk served through high-priority queue.

And, the swap-in operation operates as follows. When a packet arrives, if its source-traffic-trunk uses low-priority queue and the load of the source-traffic-trunk is less than *Permission_load*, QIB table is updated to set the queue of the source-traffic-trunk to high-priority. And also, periodically (every user-defined time) a source-traffic-trunk using low-priority is randomly chosen at QIB table and its queue is set to high-priority by updating QIB table.

Finally, the preemption operation operates as follows. When a packet arrives, if the packet is supposed to use high-priority queue, a source-traffic-trunk is randomly chosen among those using low-priority. If the load of the source-traffic-trunk for the incoming packet is greater than the load of the randomly chosen source-traffic-trunk, then QIB table is updated to set the queue of the source-traffic-trunk for the incoming packet to low-priority and the queue of the randomly chosen source-traffic-trunk to high-priority. In case that the incoming packet is supposed to use low-priority, its source-traffic-trunk is compared with a source-traffic-trunk chosen randomly among those using high-priority for the purpose of preempting the queue of the randomly chosen source-traffic-trunk.

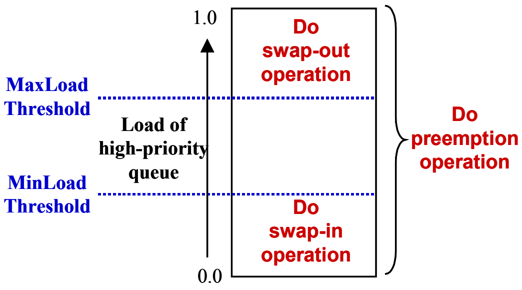

Fig. 4. Rules to trigger each operation

Fig. 4 explains when each operation is executed. Queue Coordinator component has two high-priority queue load thresholds that trigger certain operation: *MinLoad_Threshold* and *MaxLoad_Threshold*. The swap-out operation is executed only in case that the load of high-priority queue is greater than *MaxLoad_Threshold*. The swap-in operation is executed only in case that the load of high-priority queue is less than *MinLoad_Threshold*. And the preemption operation is executed whenever a packet arrives, without regard to the load of high-priority queue

The purpose of the rules shown in Fig. 4 is to let the load of high-priority queue be between two thresholds. Thus, the packets that belong to source-traffic-trunks using high-priority queue (i.e. legitimate traffic) can be guaranteed their quality of service because the load of high-priority queue cannot exceed *MaxLoad_Threshold*.

The proposed scheme has the following advantages:

1) Fast, correct, and lightweight attack detection -- Our scheme employs source-traffic-trunk based metering instead of flow based metering. The flow based metering is resource-consuming operation and difficult scheme to distinguish between malicious flow and legitimate flow because the difference in traffic rate is very trivial.
2) Quick attack defense -- We proposed swap-out as an operation for defeating DDoS attack in this paper. The operation makes all malicious traffic served through low-priority queue as soon as DDoS attack starts
3) Protection of legitimate traffic -- We proposed both preemption and swap-in operations in this paper. Both operations guarantees that legitimate traffic are served through high-priority queue.

4 Performance Evaluation

In order to evaluate the performance of the existing queuing algorithm and our scheme during a DDoS attack, we use ns-2 Network Simulator [8]. Even if the ns-2 simulator support various type of queuing service such as DropTail, Random Early Detection, Fair Queuing, Class-based Queuing, and so on, it has no function of metering and mapping scheme proposed in this paper. So we have implemented such things by extending ns-2.

4.1 Simulation Configuration

The network topology for the simulation of DDoS attack is shown in Fig. 5. The topology consists of twelve insecure source networks, eight secure source networks, and one victim network. Twenty nodes from node 4 to node 23 mean source networks. Node 3 means victim network (or system). In this simulation, each source network corresponds to a source-traffic-trunk. So there are twenty source-traffic-trunks on the simulated network.

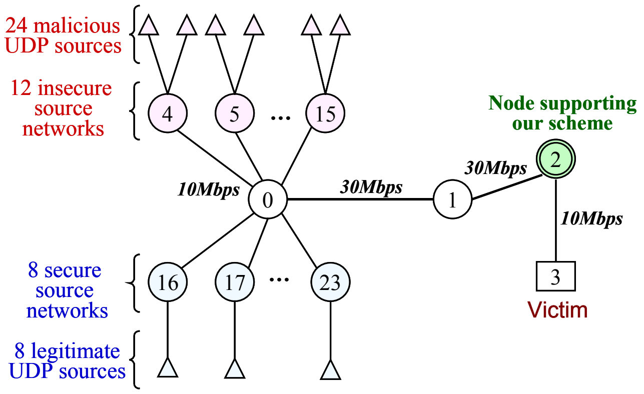

Fig. 5. The simulated Networks

All source and victim network links have the bandwidth of 10Mbps and a delay of 10ms. Links between node 0 and node 2 mean core network and have the bandwidth of 30Mbps and a delay of 20ms. The simulation scenario is as follows. First, eight legitimate users on the secure networks each generate 128-byte-long UDP packets at a rate of 400 to 600Kbps at 1.0 seconds. And then, the twenty-four malicious users on the insecure networks each generate 128-byte-long UDP packets at a rate of 400Kbps at 2.0 seconds. Finally, the malicious users start DDoS attack at the same time and periodically. That is, they each increase the transmission rate from 400Kbps to 1Mbps at 4, 8 and 12 seconds, respectively. This results in network congestion because the victim network supports only a maximum bandwidth of 10Mbps.

Malicious users stop the DDoS attack during 2 seconds at 6 and 10 seconds, respectively. That is, only half of them generate UDP traffic.

4.2 Simulation Results

Fig. 6, Fig. 7, and Fig. 8 show the simulation results of FIFO, FQ, and our scheme, respectively.

FIFO has the problem that it cannot guarantee bandwidth requested for the legitimate user. That is, after the beginning of the DDoS attack, the total bandwidth of the legitimate users falls from about 3.6 Mbps to about 1.2 Mbps as shown in Fig. 6. We've also simulated RED. The result of RED is almost same as that of FIFO except that there is no traffic oscillation.

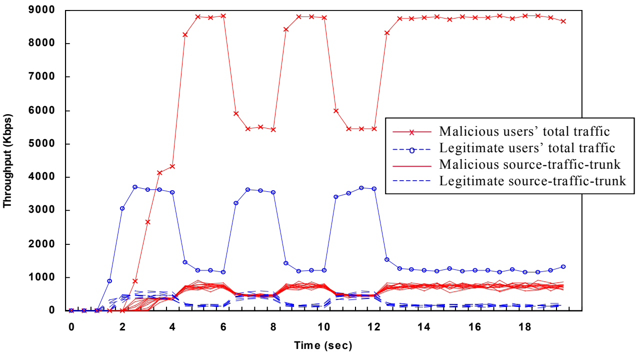

Fig. 6. Simulation results of FIFO

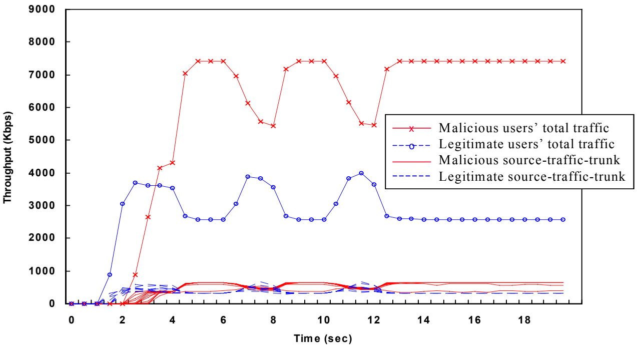

Fig. 7. Simulation results of FQ

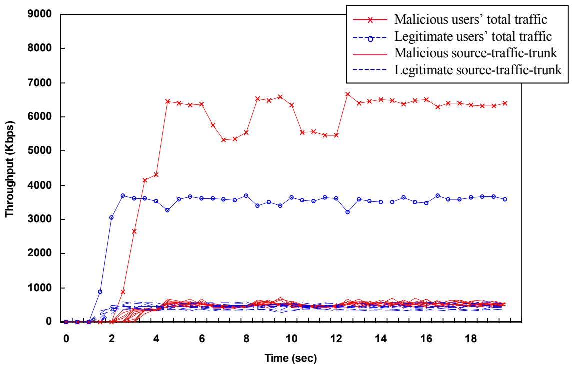

Fig. 8. Simulation results of the proposed scheme

The performance of FQ also is not good in DDoS attack even if it is better than that of FIFO. That is, after the beginning of the DDoS attack, the total bandwidth of the legitimate users falls from about 3.6 Mbps to about 2.5 Mbps as shown in Fig. 7. During the DDoS attacks, each legitimate user is allocated only about 310Kbps. The reason is that the number of malicious users increases from 12 to 24 during the attack. So, each user is allocated only about 310Kbps (the link bandwidth of the victim network (10Mbps) /

the number of all users (32)). That means the more increase the total number of malicious flows, the more decrease the bandwidth share allocated to legitimate flows. This is why FQ cannot prevent DDoS attack.

Fig. 8 shows the simulation results of our scheme. Our scheme provides the almost full bandwidth that the legitimate users requested even if there is trivial performance degradation during a very short time after the beginning of the DDoS attack. Fig. 8 shows that our scheme quickly detects and defeats DDoS attack, protecting legitimate traffic. The simulations results demonstrate that our scheme is better than any other schemes in performance.

5 Conclusion

Currently, Internet is changing from experimental to commercial network and expanding its domain from simple text to multimedia service. The one of the biggest barrier that hinders Internet development will be security problem caused by malicious user.

In this paper, we discussed DDoS attack, which are notorious for its destructive power on victim network and system.

The main purpose of this paper is to effectively control malicious and legitimate traffic in order to protect legitimate user's traffic. For this, we proposed source-traffic-trunk based metering for fast and correct attack detection, and three operations (i.e. swap-in, swap-out, and preemption operations) based traffic control for defeating DDoS attack. We simulated our scheme and the existing queuing schemes to examine the performance of each scheme. The simulation results show that our scheme is better than any other queuing schemes in case of DDoS attack.

Our future work is to verify and refine our scheme by analyzing and simulating more DDoS attacks.

References:

[1] X. Geng and A. B. Whinston, "Defeating Distributed Denial of Service Attacks", IT Pro, July-August 2000, pp 36-41

[2] K. J. Houle and G. M. Weaver. "Trends in Denial of Service Attack Technology," The fall 2001 NANOG (The North American Network Operators' Group) meeting, Oct. 2001

[3] S. Dietrich, N. Long, and D. Dittrich, "Analyzing Distributed Denial Of Service Tools: The Shaft Case," Proceedings of the 14th Systems Administration Conference (LISA 2000), Dec. 2000, pp. 329-339.

[4] A. Householder, A. Manion, L. Pesante, and G. M. Weaver, "Managing the Threat of Denial-of-Service Attacks," CERT® Coordination Center, Oct. 2001, http://www.cert.org/archive/pdf/Managing_DoS.pdf

[5] S. Keshav, "An Engineering Approach to Computer Networking: ATM Networks, the Internet, and the Telephone Network", Addison Wesley, 1997.

[6] F. Lau, S. H. Rubin, M. H. Smith, and L. Trajkovic, "Distributed Denial of Service Attacks," IEEE International Conference on Systems, Man, and Cybernetics, 2000.

[7] S. Floyd and V. Jacobson, "Link-sharing and Resource Management Models for Packet Networks," IEEE/ACM Transactions on Networking, Vol. 3 No. 4, pp. 365-386, Aug. 1995.

[8] UCB/LBNL/VINT, "ns Notes and Documentation," http://www.isi.edu/nsnam/ns.