Unified Rate Limiting in Broadband Access Networks for Defeating Internet Worms and DDoS Attacks

Keun Park, Dongwon Seo, Jaewon Yoo, Heejo Lee*, and Hyogon Kim

Division of Computer and Communication Engineering
Korea University, Seoul 136-713, Korea
{aerosmiz,heejo,hyogon}@korea.ac.kr

Abstract. Internet worms and DDoS attacks are considered the two most menacing attacks on today's Internet. The traditional wisdom is that they are different beasts, and they should be dealt with independently. In this paper, however, we show that a unified rate limiting algorithm is possible, which effectively works on both Internet worms and DDoS attacks. The unified approach leads to higher worm traffic reduction performance than that of existing rate limiting schemes geared toward worm mitigation, in addition to the added advantage of dropping most DDoS attack packets. In our experiments with attack traffics generated by attacking tools, the unified rate limiting scheme drops 80.7% worm packets and 93% DDoS packets, while 69.2% worms and 3.4% DDoS packets are dropped at maximum by previous worm scan rate limiting schemes. Also, the proposed scheme requires less computing resources, and has higher accuracy for dropping attack packets but not dropping legitimate packets.

1 Introduction

Internet worms and DDoS attacks are considered two main threats in today's Internet. The majority of Internet Service Providers (ISPs) view Distributed Denial of Service (DDoS) attack as the most significant operational security issue of today [1], while future worm epidemics are predicted to spread at yet unprecedented rates [2].

As the broadband access technologies such as Fiber To The Node (FTTN) and Fiber To The Home (FTTH) make their way to customer premises, the problem aggravates, as higher "fire power" is given to the potential attackers. It has even become possible for an attacker to launch an attack at such high speed as 100Mbps or higher, from home. With "botnets" that can mobilize up to a few hundred thousands of these high-speed agents [3], the collective attack intensity becomes formidable. Therefore, the emergence of broadband access networks raises the pressing need to monitor, and possibly control, the attack traffic near the sources.

* To whom all correspondence should be addressed.

In this paper, we focus on *rate limiting* for the specific defense mechanism to deploy in the access networks near attack sources. Rate limiting has been used in many defense mechanisms against worm and DDoS attacks. It controls the rate of traffic so that the traffic under the specified rate is allowed, whereas the traffic exceeding the rate is dropped or delayed.

Prior works in this field such as the rate limiting applications [4] and network-side rate limiting against DDoS attacks have also been studied in various ways. Pushback [5] uses the rate limiting to drop malicious packets and notify upstream routers to drop such packets. Pushback works best against DDoS flooding-style attacks, but it could sacrifice normal packets since it does not have distinction standard between normal and malicious packets except traffic quantity. Secure Overlay Services(SOS) [6] represents a private enterprise network for the rate limiting. It offers resilience to node failure, as surviving nodes assume the role of failed nodes, plus the resilience against denial of service on the system itself. But it is designed to work with private services only, as it requires changes in client software and an extensive overlay infrastructure. D-WARD [7] is another inline system that collects two-way traffic statistics from the border router at the source network and compares them to the network traffic models built upon the specification of application and transport protocols. However, it should set up and maintain complicated procedures to apply rate limiting. A study has shown how to protect e-commerce networks with the application of D-WARD [8]. MUlti-Level Tree for Online Packet Statistics(MULTOPS) [9] is proposed as an efficient data structure against DDoS which can be used for rate limiting. MULTOPS is a tree of nodes that contains packet rate statistics for subnet prefixes at different aggregation levels. MULTOPS dynamically adapts its configuration to reflect the changes in packet rates, and can avoid memory exhaustion attack. However, the authors said, given the structure of the MULTOPS tree, the size of a table (1040 bytes), the size of a record (28 bytes), a packet size of 34 bytes, and a threshold of 1000 packets per second, an attacker is able to lead the memory exhaustion to neutralize MULTOPS.

For preventing worms, a few rate limiting defense mechanisms have been developed recently [10,11,12,13,14]. IP throttling [10] limits the sending rate at an infected end host. Failed-connection-based scheme [11] and credit-based rate limiting [12] concentrate on the fact that worm scanning activities produce high number of failed connections. DNS-based rate limiting [13,14] investigates using DNS behavior as a basis for the rate limiting. These worm rate limiting defense mechanisms focus on the deployment of a host side or an edge router mechanism, performing the distinction between worms and DDoS attacks. These mechanisms will be discussed in detail in the next section, as we design our scheme coping efficiently with worm and DDoS simultaneously.

The main contribution of this paper is the introduction of the *unified* rate limiting scheme for the defending against both Internet worms and DDoS attacks, which could be put as "killing two birds with one stone." That is, while it is generally thought that worm and DDoS defense are separately considered due to their different attack behaviors, we attempt to effectively defend these two attacks using a single algorithm. We will demonstrate that its worm detection accuracy is higher than that of the existing worm defense schemes, and that it fares nicely in detecting DDoS attack packets. Furthermore, the overhead in terms of CPU and memory usage is shown to be affordable for end hosts and edge routers.

The remainder of this paper is organized as follows. Section 2 describes the design of a unified rate limiting scheme as well as the analysis of the traditional rate limiting schemes. Section 3 explains the implementation details of the scheme, and Sect. 4 presents the performance evaluation and comparison with other conventional rate limiting schemes. Lastly, Sect. 5 concludes the paper.

2 The Design of the Unified Rate Limiting Scheme

2.1 The Problems of Existing Scan Rate Limiting Schemes

As rate limiting against worm is usually performed close to the attack sources, we build our unified rate limiting scheme around it, adding the components necessary for DDoS rate limiting. A major observation of the existing rate limiting schemes listed above is that rate limiting decisions are always based on worm's aggressive connection attempts. For instance, they concentrate on the worm behavior that incurs a great number of TCP connections in a short period of time. We term these schemes as *scan rate limiting schemes*. While they work for even unknown worms, a drawback is that they are useless to defend against DDoS attacks, as discussed below.

Scan rate limiting schemes bring about three major problems for it to be used against DDoS attacks as well. The first is that they typically use a whitelist policy. In the whitelist policy, once a flow (such as a TCP connection) is verified as valid, then there would be no further examination so the subsequent packets in the flow bypass the rate limiter. Thus the rate limiter is incapable of preventing DDoS attacks flooding packets in an acceptable connection. The second problem is due to the lack of IP spoofing prevention. Many attackers forge, or "spoof", the IP source address of each packet they send to conceal their location, thereby forestalling an effective response [15]. The third problem is in the precision of detecting attack packets. The information used in the scan rate limiting schemes, such as TCP connection information, credit value and DNS record, is insufficient to identify DDoS attacks. See [16] for an empirical study.

2.2 The Design Principles of Unified Rate Limiting Scheme

Now we design a rate limiting scheme, with two objectives. The first is to cover both worm and DDoS attack as mentioned above. The second is to make our rate limiting algorithm fast and light-weight so that it does not interfere with the normal services.

Our unified rate limiting scheme consists of five modules to defense against Internet worms and DDoS attacks simultaneously. The five modules are shown in Fig. 1 and described as follows.

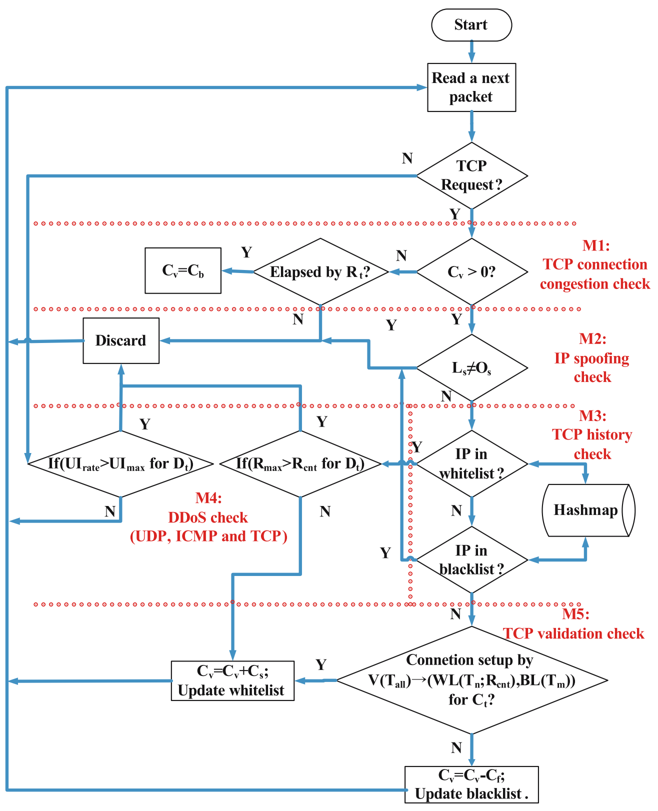

Start
Read a next packet
TCP Request?
N
Y
M1: TCP connection congestion check
$C_v > 0$?
N
Elapsed by R_t?
Y
$C_v=C_b$
N
Y
Discard
M2: IP spoofing check
$L_s \neq O_s$
N
M3: TCP history check
IP in whitelist?
Hashmap
IP in blacklist?
Y
If($UI_{rate}>UI_{max}$ for D_t)
If($R_{max}>R_{cnt}$ for D_t)
M4: DDoS check (UDP, ICMP and TCP)
M5: TCP validation check
Connetion setup by $V(T_{all}) \rightarrow (WL(T_n;R_{cnt}),BL(T_m))$ for C_t?
$C_v=C_v+C_s$; Update whitelist
$C_v=C_v-C_f$; Update blacklist.

Fig. 1. Unified rate limiting algorithm

- **TCP connection congestion check(M1):** monitoring whether connection failures excessively occurred.
- **IP spoofing check(M2):** checking source address spoofing to discard suspicious packets as many as possible before establishing a connection.
- **TCP history check(M3):** utilizing black and white lists to reduce the execution time of rate limiter through the reuse of the existing lists.
- **DDoS check(M4):** allowing a connection only if the transmission rate does not exceed the predefined threshold.
- **TCP validation check(M5):** updating connection information such as ACK response time and request count, and deciding which list the IP should belong to; whitelist or blacklist.

Using these five modules, it is possible to screen excessive traffic which is caused by Internet worms and DDoS attack.

Even though the scheme prevents those attacks very well, it is useless if the algorithm overhead is unaffordable. Figure 2 shows the reason why we set the module order like Fig. 1. In Fig. 2, our scheme examines credit value first that

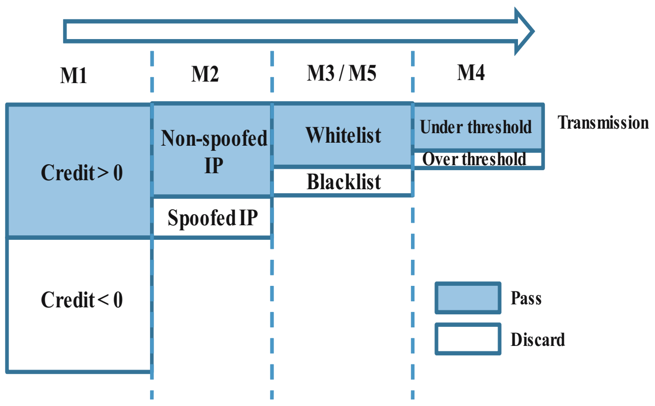


Fig. 2. Unified rate limiting policy order

holds the largest number of packets to decide whether the packet should be passed or blocked. IP spoofing check module handles second largest number of packets. After that, only if a packet is in whitelist, we check the packet transmission rate of the connection. From this order of module execution, we are able to consume the smallest computation on the average for handling one packet.

Unified rate limiting scheme can be deployed in the various places such as client PC's, NAT boxes and edge routers. In Sect. 4, we will show that the unified scheme has higher attack packet dropping ratio and lower false detection than the existing rate limiting schemes.

2.3 The Details of Unified Rate Limiting Scheme

- **M1(TCP connection congestion check):** When a new TCP connection setup request is made, we first check the "congestion" status of TCP connections by the credit value C_v. A TCP packet is allowed to pass if $C_v > 0$. We compute C_v as follows. We initialize $C_v = C_b$ at the beginning, and bound the maximum value of C_v to C_{max}. C_b is a default credit value which is normally set to small positive integer like 5. If the initiated TCP connection fails to set up, C_v is subtracted by C_f. Therefore, if a worm generates a number of failed connections, we eventually have $C_v < 0$. Then further connection attempts are blocked. As soon as we reach a negative value of C_v, we run a timer, and the connection setup blocking is enforced until the timer reaches R_t, when the credit value is reinitiated to C_b. If the three-way handshake succeeds on the other hand, C_v is incremented by C_s. $C_v > 0$ upon the connection attempt means that there are not too many failed or ongoing connection attempts, so we allow new attempts.
- **M2(IP spoofing check):** for the allowed connection attempt, we validate the source address for forgery. We check the local address L_s against the source address of an outgoing packet O_s, and if $L_s \neq O_s$, we drop the packet.

- **M3(TCP history check):** two lists are used for the faster processing of subsequent packets. If a TCP packet is transmitted, we check if the destination address of the packet is recorded in $WL(T_n, R_{cnt})$ or $BL(T_m)$. If former, it is allowed to pass and R_{cnt} is incremented. If latter, the packet is immediately dropped. The packet belonging to the blacklist should not be dropped permanently so that the blacklist is reset in BL_t, blacklist timeout.
- **M4(DDoS check):** If R_{cnt} above exceeds a predefined rate R_{max} within time D_t, it means that an excessive number of connections are made to the destination of T_n in $WL(T_n, R_{cnt})$, *i.e.* a TCP flooding attack. Then, the packets are dropped. In case of UDP or ICMP, we need to compare the current sending rate UI_{rate} to the predefined maximum rate UI_{max}. If $UI_{rate} > UI_{max}$ within D_t, packets will be dropped. After passing P_t time period, UI_{rate} is reset to 0. To reduce the false positives incurred by lots of legitimate retransmissions, D_t can be defined as a small period, e.g. 1 or 2 seconds. Thus, it is not possible that a legitimate user retransmitting packets is regarded as an attacker.
- **M5(TCP validation check):** If not listed in neither the whitelist nor the blacklist, the TCP packet should be validated to be registered in either list. In case that the source address is validated and the outgoing TCP connection attempt is allowed to pass the filter, we check if it gets an ACK within the time C_t. Depending on the result of the check, all outgoing TCP packets T_{all} are classified into two groups – normal group T_n or malicious group T_m. T_n and T_m are stored with the request counter R_{cnt} in the whitelist $WL(T_n, R_{cnt})$ and the blacklist $BL(T_m)$, respectively. This is depicted in Fig. 1 as $V(T_{all}) \rightarrow \{T_n, T_m\}$ for C_t.

3 Prototype Implementation

3.1 Packet Filter Driver and Application

To evaluate the performance of the unified rate limiting, we have implemented a prototype as an application program based on the algorithm[1]. The program takes control of packets in the user and the kernel mode in Microsoft Windows system.

We implemented the packet filter driver with filter hook driver provided by Windows Driver Development Kit (DDK). For the implementation, we let the functions in filter driver use a number of control codes which determines to call appropriate functions.

3.2 Network Simulator

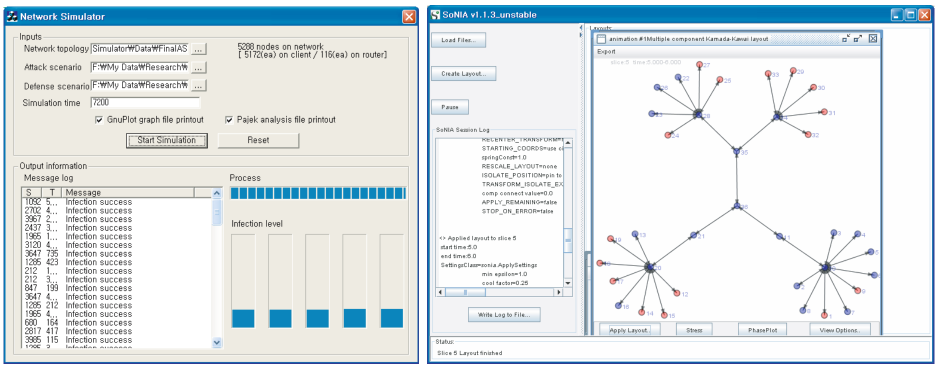
We developed a network simulator in order to measure the possible impact of the proposed algorithm on the Internet under partial deployment, and to find an

[1] The application program and the network simulator developed are available at `http://ccs.korea.ac.kr/URL`

effective deployment strategy for a better performance. We utilize the Internet AS connectivity graph of year 2006 obtained from the RouteView project [17] as a network topology, which consists of 21,211 nodes. Even though the AS graph represents the connectivity between AS's instead of routers or computers, we use it since it is closer to the real Internet topology than any artificially generated graphs. The network simulator does not consider asymmetric routing that exists in the real network because the routing path does not take any effect on the proposed mechanism. The reason is that our mechanism only controls the rate limiting of outgoing packets at the edge of network. Among 21,211 nodes, 21,022 terminal nodes are regarded as clients and 189 central nodes are considered routers.

The inputs to the worm attack scenarios are scanning speed, the ratio of infection success, and attack strategy. Attack strategy can be categorized as random or local. In the random strategy, infection targets are chosen randomly. In the local strategy, 50% of the targets are chosen randomly, whereas the other 50% of the targets are chosen in the local subnet where the infected node resides. As to the DDoS attack scenario, attack packets per second, the number of attack nodes, and attack strategy (random or local) are given as input. Meanwhile, the defense configurations have the following elements – deployment ratio of a rate limiting algorithm, deployment strategy (random or local), false positive and negative rate of a rate limiting algorithm with respect to the worm and DDoS attacks. In particular, the local deployment strategy represents a rate limiting algorithm fully installed in a specific subnetwork (deployment ratio of 100%). On the other hand, the random deployment strategy is to install the rate limiters randomly.

The output of the network simulator includes a network graph with the infection status of each node, a sequence of infection steps for animating the progress of infection on a network. In our experiments, statistics are collected for 10 minutes simulation in each case of worm and DDoS attacks. The working example and sample output of the simulator is shown in Fig. 3.



Network topology: Simulator₩Data₩FinalAS
Attack scenario: F:₩My Data₩Research₩
Defense scenario: F:₩My Data₩Research₩
Simulation time: 7200
[✓] GnuPlot graph file printout
[✓] Pajek analysis file printout

5288 nodes on network
[5172(ea) on client / 116(ea) on router]

```
S     T     Message
1092  5,..  Infection success
2702  4,..  Infection success
3967  2,..  Infection success
2437  3,..  Infection success
1965  1,..  Infection success
3120  4,..  Infection success
3647  735   Infection success
1285  423   Infection success
212   1,..  Infection success
212   3,..  Infection success
847   199   Infection success
3647  4,..  Infection success
1285  212   Infection success
1965  4,..  Infection success
680   164   Infection success
2817  417   Infection success
3985  115   Infection success
```

```
        RECENTER_TRANSFORM=
        STARTING_COORDS=use c
        springConst=1.0
        RESCALE_LAYOUT=none
        ISOLATE_POSITION=pin to
        TRANSFORM_ISOLATE_EX
        comp connect value=0.0
        APPLY_REMAINING=false
        STOP_ON_ERROR=false


<> Applied layout to slice 5
start time:5.0
end time:6.0
SettingsClass=sonia.ApplySettings
        min epsilon=1.0
        cool factor=0.25
```

Slice 5 Layout finished

Fig. 3. Network simulation: (left) the operation of the network simulator, (right) the output of a network infection animation file

4 Evaluation

4.1 Experimental Setup

We use the malicious worms such as Blaster, CodeRed, Sasser and Welchia. As to DDoS attack traffic, we generate TCP, UDP, ICMP packets by the use of publicly available attacking tools. The experimental system is equipped with Pentium4 CPU running at 3.0GHz, 512MB main memory, on Microsoft Windows XP. Additionally, we also run the network simulator with the results from the experimental system as the input values, in order to measure the effect on network.

We use the following settings for each rate limiting schemes in our experiment. IP throttling has a five-address working set and a delay queue length of 100 [10]. Credit-based connection rate limiting is configured with its original setting in the reference [12]. DNS-based rate limiting scheme is implemented with 100,000 DNS lists and the rest is the same as in the previous work [13]. Our unified rate limiting scheme has the following configuration: $C_b = 20$, $C_o = -1$, $C_s = +3$, $C_f = -1$, $C_t = 1\text{sec}$, $D_t = 1\text{sec}$, $P_t = 10\text{sec}$, $BL_t = 10\text{sec}$, $C_{max} = 100$, $R_{max} = 500$, and $UI_{max} = 1,000$. This is not necessarily an optimal setting, which can be acquired by further tests. For the whitelist and the blacklist, hashmap [18] is used in our experiments, since it performs well to record and retrieve IP address and the request counter. The hashmap has dynamic size depending on the number of connection, and its searching time guarantees O(1).

In the experiments, we consider four performance metrics. The first metric is the dropping ratio of attack packets. This metric allows us to see defense performance when we adopt a rate limiting scheme. Another performance metric is false alarm rates - false positive and false negative. This shows the precision of a rate limiting scheme. The third metric is the effectiveness of deployment strategy such as local deployment or random deployment. The fourth metric is the overhead.

4.2 Simulation Results

In the first place, we measure the attack packet dropping ratio and the detection accuracy, *i.e.*, false positive and negative. Figure 4 shows the results for the worm attack, with rate limiting schemes[2], at the end host.

As shown in the figure, the unified rate limiting scheme has the highest attack packet dropping ratio. On the other hand, it shows the lowest false rates. In Fig. 4 (right), the curves represent the sum of false positive and false negative ratios. The increased accuracy in the unified rate limiting is owing to the two validation checks, *i.e.*, with IP address and the credit value.

For the DDoS attack, we measured the attack packet dropping ratio and the detection accuracy as well. The DDoS attack packets were comprised of 25%

[2] IP_RL: IP Throttling Rate Limiting, CB_RL: Credit Based Rate Limiting, DNS_RL: DNS based Rate Limiting and UNI_RL: Unified Rate Limiting (our proposed scheme).

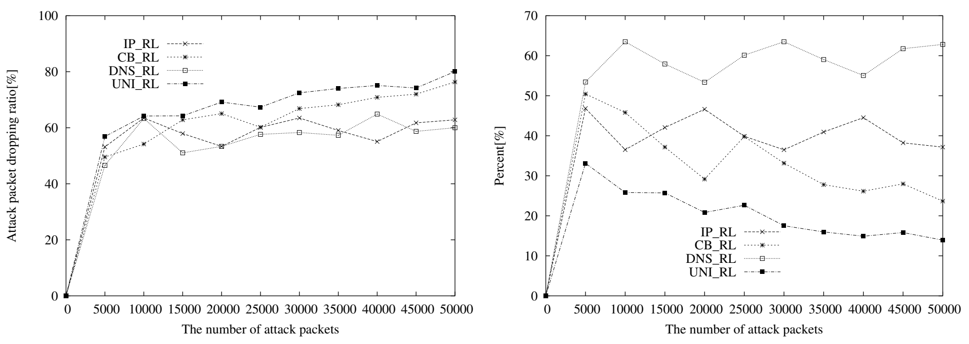


Fig. 4. Simulation results for rate limiting schemes on worm attack in local environment: (left) attack packet dropping ratio, (right) sum of false positive and false negative ratio

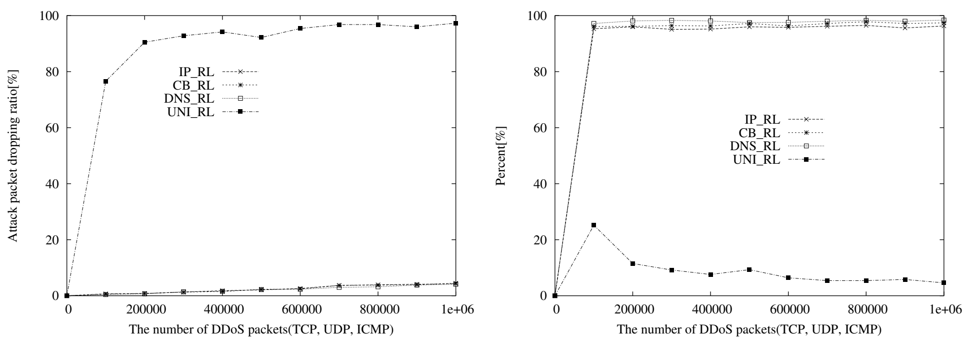


Fig. 5. Simulation results for rate limiting schemes on DDoS attack in local environment: (left) attack packet dropping ratio, (right) sum of false positive and false negative ratio

IP-spoofed TCP packet, 25% normal TCP packet, 25% UDP packet, and 25% ICMP packet.

Figure 5 shows that the unified rate limiting scheme is highly effective for dropping DDoS attack packets in both measures. Namely, most IP-spoofed TCP packets are detected and prevented from going out into the network. In particular, the blacklist helps drop excessively requested IPs. In contrast, the existing worm scan rate limiting schemes are quite ineffective, as only 5% packets are dropped at maximum. This demonstrates the advantage of the unified limiting scheme.

To measure the performance of a rate limiting algorithm and deployment strategy in broadband access networks, we obtained the results as shown in Fig. 6. Figure 6 shows the attack packet dropping ratio in various rate limiting schemes with respect to the deployment ratio. In the figure, the unified rate limiting scheme shows a strong detection capability, not only for worm attacks

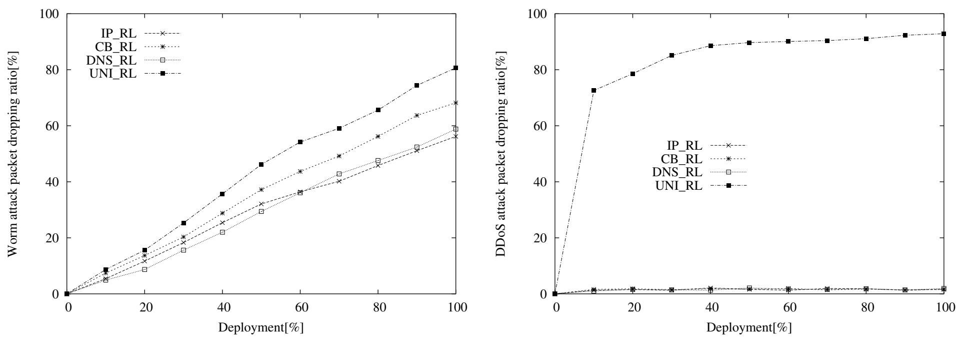


Fig. 6. Simulation of attack packet dropping ratio of rate limiting schemes with respect to deployment ratio: (left) worm attack, (right) DDoS attack

but also for DDoS attacks. The unified rate limiting scheme drops 80.7% worm scanning packets and 93% DDoS packets at the full deployment, while at most 69.2% worm scanning packets and 3.4% DDoS packets are dropped by existing rate limiting schemes.

To show the effectiveness of the rate limiting schemes with respect to the deployment strategy and the deployment ratio for worm and DDoS attack, we make an experiment with 21,022 nodes in subnetworks measured for identifying worm and DDoS attack. In the local deployment scenario of worm attack, worm infection is more mitigated than in the random deployment. This implies the installation of a rate limiting scheme in subnetwork can be effective for reducing worm traffic. However, the wider the deployment goes, the less the performance difference between local and random strategy results.

In addition, we notice that worm scan rate limiting schemes have virtually no DDoS detection capability, and that they record more than 20,000 DDoS attack nodes even though 100% deployment is provided, regardless of the deployment strategy. However, the unified rate limiting scheme takes effect both in the local and random deployment. Unlike in the worm attack scenario, the deployment strategy has no significant impact on the performance.

4.3 Overhead Analysis

For each rate limiting scheme considered in this paper, its time and space complexity is summarized in Table 1. Q represents the length of the queue for IP throttling and DNS-based rate limiting scheme. L represents the length of the DNS list in DNS-based rate limiting scheme. M and N represent the length of the blacklist and whitelist of unified rate limiting scheme respectively. In terms of the time complexity, IP throttling complexity includes the search time of the queue and the whitelist. The complexity for the DNS-based rate limiting includes the DNS list search time along with the queue and whitelist search time. Credit-based rate limiting searches the whitelist when $C_v < 0$, which costs

Table 1. Comparison of complexities and CPU usages of rate limiting algorithms

Rate limiting algorithms	Complexity		CPU usages	
	Time	Space	Worm(5,000 TCP packets)	DDoS(10,000 TCP packets)
IP_RL	O(N)	O(Q+N)	6%	2.5%
DNS_RL	O(N+L)	O(Q+N+L)	3.2%	2.4%
CB_RL	O(N)	O(N)	7.5%	2.3%
UNI_RL	O(N+M)	O(N+M)	4.7%	2.7%

$O(N)$. The unified rate limiting searches the whitelist and the blacklist so that the complexity becomes $O(N + M)$.

During the experiments, the overhead is measured as shown in CPU usages of Table 1. We notice that CPU usage is tightly coupled with time complexity. But even if 5,000 IP addresses should be simultaneously searched, it consumes only 3.2–7.5% of the CPU cycles in the hashmap data structure. Moreover, in case of the credit-based rate limiting and the unified rate limiting scheme, there is no need for the search with respect to the value of C_v. It drastically reduces the CPU usage. In terms of the memory usage, all rate limiting schemes consume 6.5MB of memory, which is readily affordable on the modern computers.

5 Conclusion

As the broadband access technology brings high-speed pipes to the customers, it also raises a security concern because the attackers can exploit the increased bandwidth to mount stronger attacks. Thus the attack mitigation strategy that places defense mechanisms close to the potential attack sources becomes important in the Internet with emerging broadband access networks. The unified rate limiting scheme that we propose in this paper works close to the attack sources, and deals with the two most threatening attacks, Internet worms and DDoS attacks. Unlike existing rate limiting schemes, it is effective to both, and the performance from the unified monitoring is also higher than any existing rater limiters. In particular, it sharply contrasts with the existing rate limiting schemes by drastically reducing DDoS attack packets. Through extensive simulations, we show that the unified approach drops more worm packets than any other rate limiting schemes, with less false alarm. We also show that the unified rate limiting scheme is most effective in the locally and highly deployed networks. As being exploited unknowingly in the worm propagation or DDoS attacks is an unpleasant experience to anybody, we believe that ISPs can include the unified rate limiting scheme in their distribution packages to deploy on subscriber PCs. Or, the ISPs can deploy the scheme in the ingress routers as the pipes which are not too thick, thereby rendering the deployment cost low. Since a single piece of software can deal with both worms and DDoS, separate installation is not necessary. As for the deployment cost, the proposed unified scheme also operates with affordable CPU and memory overhead, making it easily integrable into existing networks elements.

Acknowledgments

This work was supported in part by KT (Korea Telecom), Defense Acquisition Program Administration and Agency for Defense Development under the contract UD060048AD. Additionally supported by the ITRC program of the Korea Ministry of Information & Communications.

References

1. Arbor Networks: Worldwide Infrastructure Security Report (September 2007)
2. Chen, T.M., Robert, J.-M.: Worm Epidemics in High-Speed Networks. IEEE Computer 37(6), 48–53 (2004)
3. Yaneza, J., Sancho, D.: The trend of threats today: 2005 annual roundup and 2006 forecast. Trend Micro White Paper (2005)
4. CISCO SYSTEMS: ONS 15327 Multi-Service Provisioning Platform (November 2002)
5. Mahajan, R., Bellovin, S.M., Floyd, S., Ioannidis, J., Paxson, V., Shenker, S.: Controlling High Bandwidth Aggregates in the Network. ACM SIGCOMM Computer Communications Review 32(3), 62–73 (2006)
6. Keromytis, A., Misra, V., Rubenstein, D.: SOS: Secure Overlay Services. In: Proc. of ACM SIGCOMM, pp. 61–72 (August 2002)
7. Mirkovic, J., Prier, G., Reiher, P.: Attacking DDoS at the source. In: Proc. of 10th IEEE International Conference on Network Protocols (November 2002)
8. Kang, J., Zhang, Z., Ju, J.: Protect e-commerce against DDoS attacks with improved D-WARD system. In: Proc. of the e-Technology, e-Commerce and e-Service conference, March 2005, pp. 100–105 (2005)
9. Gil, T., Poletto, M.: MULTOPS: a data-structure for bandwidth attack detection. In: Proc. of 10th Usenix Security Symposium (August 2001)
10. Williamson, M.M.: Throttling Viruses: Restricting propagation to defeat malicious mobile code. In: Proc. of the 18th Annual Computer Security Applications Conference (ACSAC) (June 2002)
11. Chen, S., Tang, Y.: Slowing Down Internet Worms. In: Proc. of the 24th International Conference on Distributed Computing and Systems (ICDCS) (March 2004)
12. Schechter, E., Jung, J., Berger, A.W.: Fast Detection of Scanning Worm Infections. In: Proc. of the 7th International Symposium on Recent Advances in Intrusion Detection (RAID) (October 2004)
13. Whyte, D., Kranakis, E., van Oorschot, P.C.: DNS-based detection of scanning worms in an enterprise network. In: Proc. of the 12th ISOC Symposium on Network and Distributed Systems Security(NDSS) (February 2005)
14. Granger, G., Economou, G., Bielski, S.: Self-securing network interfaces: What, why and how. Technical report, Carnegie Mellon University, CMU-CS-02-144 (May 2002)
15. Moore, D., Voelker, G.M., Savage, S.: Inferring Internet Denial-of-Service Activity. ACM Transactions on Computer Systems (TOCS) 24(2), 115–139 (2006)
16. Wong, C., Bielski, S., Studer, A., Wang, C.: Empirical Analysis of Rate Limiting Mechanisms. In: Proc. of the 11th International Symposium on Recent Advances in Intrusion Detection (2006)
17. Meyer, D.: University of Oregon Route Views archive project (2006), `http://archive.routeviews.org`
18. SGI: Standard Template Library Programmer's Guide; hashmap containers, `http://www.sgi.com/tech/stl/hash_map.html`