

A Statistical Packet Inspection for Extraction of Spoofed IP Packets on Darknet

Masashi Eto, Daisuke Inoue, Mio Suzuki, Koji Nakao

National Institute of Information and Communications Technology,
4-2-1 Nukui-Kitamachi, Koganei, Tokyo 184-8795, Japan
{eto, dai, mio, ko-nakao}@nict.go.jp

Abstract. A darknet is a set of unused IP addresses and using it is a good way to monitor network attacks such as malware's scans. Traffic arriving at the darknet, however, contains some network attacks with spoofed IP packets, which may confuse analysis engines of monitored traffic. Therefore, it is important to distinguish the spoofed IP packets from darknet traffic in order to pursue accurate analysis results. In this paper, we propose an inspection method focusing on the Time To Live (TTL) field of each packet in order to statistically extract spoofed IP packets from whole traffic observed by the darknet. Consequently, we will provide an efficient and effective analysis engine against network attacks from the darknet by means of the proposed inspection method for the extraction of spoofed IP packets on darknet. Through a practical evaluation, we have found that at most only 1.26% of spoofed packets exist in the darknet traffic, which supports the accuracy of incident analysis with darknet monitoring.

Keywords: Spoofed IP Packets, Darknet, TTL, Standard Deviation

1 Introduction

The Internet has become an indispensable infrastructure for social, economic and cultural life, while it has been filled with unwanted traffic and attacks. The intense fountainheads of these traffic and attacks are highly functional malicious programs, i.e., malwares. To grasp the present trends of malicious activities caused by malwares, there are a number of network monitoring projects over the Internet [1]-[10]. Most of the projects are conducting continuous observations of network traffic toward *darknet*, which is a set of globally announced unused IP addresses. A darknet is a good way to monitor network attacks, such as by malwares' scans. This is because there is no legitimate host using these addresses; we can thus consider all incoming traffic to be a consequence of some kind of malicious activity or the result of misconfigurations. Moreover, since there are no common end users, a darknet is suitable for extensive monitoring coverage without any of the problems related to privacy issues encountered when monitoring the real (live) network. As the principal darknet monitoring method, we deployed the black hole sensors, which quietly monitor incoming packets without ever responding to the opposite hosts. By using these black hole sensors in the darknet IP domains, we could observe emerging network attacks,

including malware-initiated network scanning, malware infection behavior, and DDoS backscatters.

Figure 1 is an example of the statistics of incoming traffic to a /16 darknet (i.e., 65,536 unused IP addresses) during Jan. 2009. We have deployed a black hole sensor on the darknet that only captures the incoming traffic and never replies to them. The figure shows that the average number of incoming packets per day was about 2.9 million, and the average number of unique source IP addresses per day was about 41 thousand.

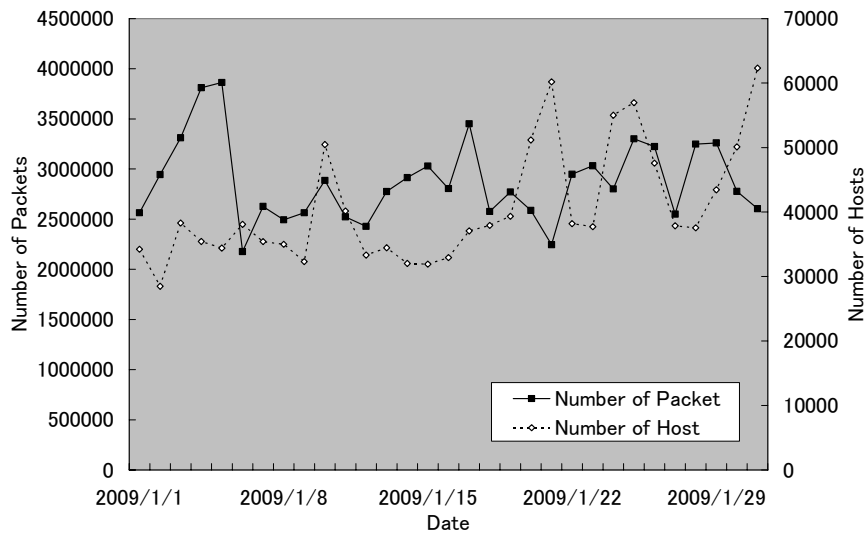


Fig. 1 Number of packets and unique hosts per day in a /16 darknet

Here we have encountered an essential question that "how many source IP addresses in the darknet traffic are properly used?" The Internet protocol is vulnerable to the source IP address spoofing in nature. Therefore it is likely that the darknet traffic contains some network attacks with spoofed IP packets. These spoofed packets may not only confuse the statistics of the darknet traffic but also be undesirable noise to analysis engines for the traffic. Consequently, it is important to distinguish the spoofed IP packets from the darknet traffic in order to pursue accurate statistics and analysis results.

In this paper, we describe a inspection method for extracting spoofed packets from the darknet traffic. The proposed method takes advantage of a statistical analysis on the Time To Live (TTL) field of each IP packets. Based on the method, we estimate the ratio of spoofed packets to the darknet traffic. The remainder of this paper is organized as follows. In Section 2, we summarize some related works. In Section 3, we describe our proposed method. Evaluation results with the darknet traffic are shown in Section 4. Finally, we conclude this paper in Section 5.

2 Related Works

There have been a lot of efforts to detect or filter spoofed IP packets in the context of mitigating distributed denial of service (DDoS) attacks to live networks (i.e., networks with legitimate hosts and/or servers). As far as we know, no literature has dealt with the spoofed IP packets in a darknet, though some mentioned backscatter from spoofed source addresses. Therefore, here we summarize some related works that take measures to the spoofed IP packets on the live networks.

Ingress filtering [11] is a filter mechanism on periphery routers. The routers check the validity of source IP addresses of each packet from their downstream networks so as to drop packets having invalid source addresses.

Route-based filtering [12] is a filter mechanism on border routers. The routers use routing information in order to detect and discard spoofed packets that have inconsistent source/destination IP addresses with their route.

IP traceback [13][14][15] is an active pursuit mechanism of source hosts launching spoofed DDoS attacks. There are several methodologies, such as use of ICMP message, logging on routers, packet marking, and so on.

The above filtering and traceback mechanisms require close collaboration from widespread routers. Therefore, the mechanisms sometimes come across issues of a large-scale deployment and legal contexts.

TTL based filtering [16][17] can be carried out in an edge router or in an end host. Jin et al. [16] examined that 1) TTL values of packets from a certain source host are stable, 2) TTL values of packets from arbitrary source hosts show some diversity, 3) It is difficult that an adversary makes spoofed packets having appropriate TTL values for each destination IP addresses. Therefore, if the edge router (or end host) knows a mapping table between source IP addresses and appropriate TTL values, it can drop spoofed IP packets that have unusual TTL values. For evaluating each TTL values of packets arriving at live networks, Jin et al. use an exact matching with the mapping table, and Templeton et al. [17] use conditional entropy of source IP addresses and expected TTL values.

Our packet inspection method takes advantage of the TTL values too, however, our focus is not on the live networks but on the darknet. This is the first trial to estimate spoofed IP packets in actual large-scale darknet traffic. Furthermore, our method uses a simple statistics, i.e., standard deviation of the TTL values of packets arriving at each subnetworks in darknet in order for detecting irregular subnetworks and finally extracting the spoofed IP packets from the subnetworks.

3 Statistical Packet Inspection Methods

Given darknet traffic, our aim is to find spoofed IP packets and estimate how large amount of those packets are there. To realize this purpose, we propose a statistical packet inspection method based on the standard deviation of TTL values in observed packets from a single source subnetwork. Specifically, the method divides all observed packets into source subnetworks with a 24 bit netmask (class C network), then computes the standard deviation of each individual subnetwork. Additionally, the

actual input for the computation of standard deviation is the hop-count, namely the difference between the supposed initial TTL value (at the source) and the final TTL value in observed packets (at the destination). Through our own brief observation, we found that there are two possible cases of spoofing that raise anomalous final TTL values; spoof of source IP address or initial TTL value. Our goal is to detect these kinds of spoofed packets and reveal the total amount of concealed spoofed IP packets.

3.1 Spoofing IP Packet

TTL is an 8-bit field in the IP header, which is defined to control the maximum lifetime of each packet in the Internet to avoid the flood of packets that lost their destination. When a packet is transmitted from a sender to a receiver, each intermediate router decrement the TTL value one by one before he forward it to the next-hop. Therefore, the difference between the initial TTL value at a sender and final TTL value at a receiver denotes the number of intermediate routers, namely the hop-count.

Given a single packet with a final TTL ($fTTL$), the hop-count (HC) of the packet is determined by the difference between $fTTL$ and the supposed initial TTL ($iTTL$) (1).

$$HC = iTTL - fTTL \quad (1)$$

It is well known that the initial TTL value depends on the type of OSs. Although there are not any regulations about the initial TTL value, according to [18], most OSs decide its initial TTL from a few types of values as shown in Table 1. This set of initial TTL values cover most of the popular OSs, such as Microsoft Windows, Linux, variants of BSD, and many commercial Unix systems.

Table. 1 Major OS Versions and Their Initial TTL

OS and Version	Initial TTL
HP-UX	30
Windows 95	32
Mac OS 2.0.x	60
FreeBSD 5.0	64
RedHat 9	64
Mac OS X	64
Windows 98	128
Windows NT 4.0	128
Windows ME	128
Windows XP	128
MPE/IX (HP)	200
NetBSD	255
OpenBSD	255

Actually, the result of our own observation supports this condition. Figure 2 shows the numbers of packets of each final TTL, which is monitored in a certain week (from

4th to 10th in Jan. 2009). In this figure, we can see some peaks around 41 and 110 that are assumed to be derived from the initial TTL values of 60, 128 respectively. Thus, since the final TTL values are focused in some points at a certain level, we can suppose the initial TTL value only from the final TTL value.

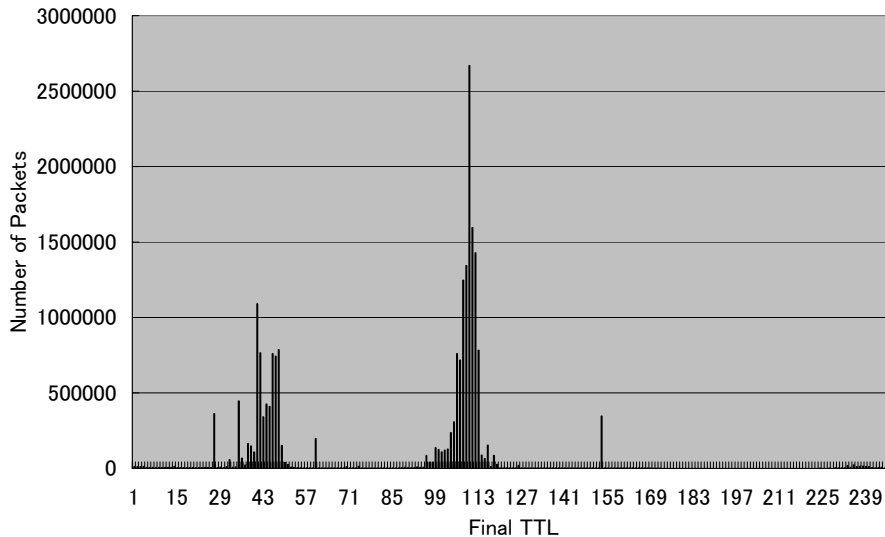


Fig. 2 Number of Packets of Each Final TTL in a Week

Since packets sent from multiple hosts in a single subnetwork to the same destination are usually transmitted along the same route, we can suppose that packets from source hosts in the same subnetwork must have similar hop-counts. Contrastingly, let us assume that an attacker in a network (A) sends a packet to a destination network (B) with spoofing its source IP address as if it was sent from another subnetwork (C). In this case, at the destination, though the packet seems to be sent from the network (B), the hop-count should be different from the other hosts in subnetwork (A) because of the difference of the distance between (A-B) and (C-B).

Except IP address or TTL spoofing, there might be some cases that induce disorganization of final TTL values. For example, suppose a situation which is often seen in medium-scale enterprise networks; a single IP address in a /24 subnetwork is configured as a NAT router, and the router holds several private networks behind itself. In this case, the final TTL values of packets from the hosts behind the NAT router differ from the packets of the other source IP addresses in the /24 subnetwork. However, since the depths of private network behind NAT router is usually only a few hops, we can ignore those differences among TTLs. Furthermore, as another example, since a network environment for a given IP address may change as time advances (e.g., shift of a client host (OS) by DHCP), the final TTL value may also fluctuate. Even in this case, however, we can determine the hop-count by supposing the accurate initial TTL value unless the packet is spoofed. In other words, if a hop-

count of a packet differs from other packets in a single source subnet, the packet can be assumed to be spoofed.

3.2 Initial TTL Determination

As shown in Table 1, most of the defined initial TTL values are sufficiently far apart each other. Furthermore, since previous works [19, 20] show that the distances among most of Internet hosts are within 30 hops, we can estimate initial TTL values from given final TTL values. More specifically, we determine the initial TTL value by selecting the smallest value in Table 1 that is larger than its final TTL. For example, given the final TTL value 105, the initial TTL value is determined as 128, the smaller of the two possible initial values, 128 and 255. Note that when an obtained hop-count was between 30 and 32, 60 and 64, and between 32 and 60, it is difficult to determine the exact initial TTL value. To resolve these ambiguous cases, we take the one which is more similar to the hop-count of other packets from the same subnetwork.

Additionally, through a preliminary survey, we found that there are a number of packets which seem to have initial TTL values {48, 100, and 168} that are not listed in Table 1. To get rid of influences of these packets, we decided to add these values to the initial TTL list.

As a result, we defined a rule to decide initial TTL values as shown in Table 2.

Table. 2 Determination Rule of Initial TTL

Final TTL	Supposed Initial TTL
0 ~ 21	32
22 ~ 39	48
40 ~ 59	64
60 ~ 89	100
90 ~ 120	128
120 ~ 159	168
160 ~ 189	200
190 ~ 255	255

Based on this rule, we determine the supposed initial TTL value and the hop-count for the computation of the standard deviation.

3.3 Inspection Algorithm

The inspection method estimates the total amount of spoofed IP packets. At first, the inspection method divides all observed packets based on source IP addresses into subnetworks with a 24 bit netmask (/24 network), which stands on a fundamental assumption of this paper. Namely, in many cases, terminal networks are segmented into class C or smaller subnetworks, therefore the final TTL values must be same or very close if the source hosts belong to the same subnetwork. Unfortunately, since it

is impossible to passively figure out the size of source subnetwork, the inspection method roughly classifies observed packets into /24 source subnetworks. Subsequently, the standard deviation will be computed for each /24 subnetwork. Where a set of packets from the same source subnetwork (N) is given, the standard deviation (SD) of the network is derived with hop-counts (HC) of each packet as follows.

$$SD_N = \sqrt{\frac{\sum (HC_i - \overline{HC})^2}{n}} \quad (2)$$

With a result of the computation, the inspection method extracts subnetworks whose standard deviations are greater than a certain threshold (T_1). This means that the packets from the subnetworks contain spoofed IP packets. After a subnetwork is extracted, the inspection method seeks the spoofed IP packets by calculating deviation scores (DS s) for each packet with Formula 3. The spoofed IP packets are determined based on the threshold (T_2).

$$DS_i = \frac{HC_i - \overline{HC}}{SD_N} \quad (3)$$

With this inspection method, we can detect and estimate spoofed IP packets. The next section analyzes the practical traffic data observed in darknet, and shows the total amount of spoofed IP packets.

4 Evaluation Results

Through the development of the inspection method in the previous section, this section firstly demonstrates a further inspection of the source IP address spoofing, and then introduces the estimation of spoofed IP packets. The data used for this evaluation is traffic monitored during (4/Jan/2009 - 10/Jan/2009) at a black hole sensor on a /16 darknet. During the observation period, the sensor has monitored 20,422,842 packets from 261,131 unique source IP addresses.

4.1 Statistical Result

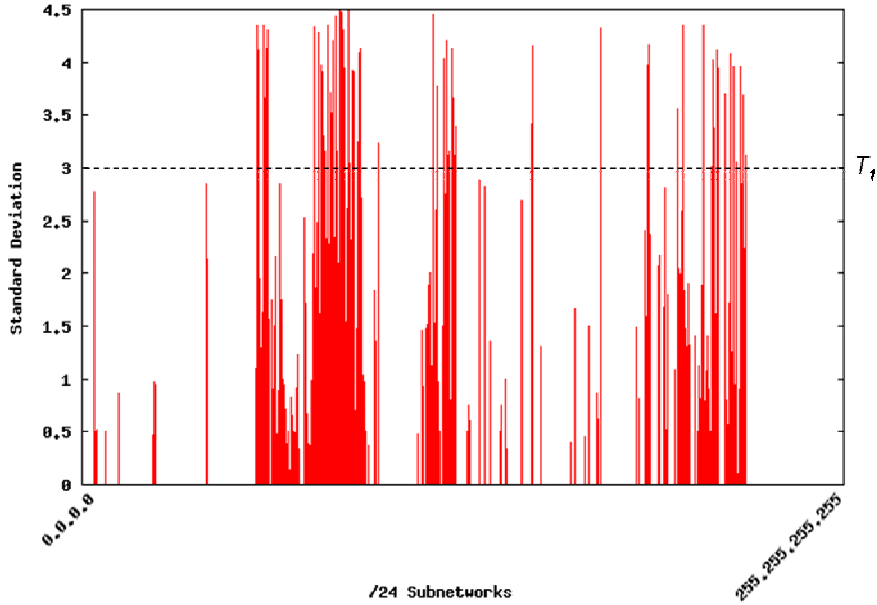


Fig. 3 Standard Deviations of Each /24 Subnetwork

The statistical analysis result is shown in Figure 3. In the figure, we can see that in 847 subnetworks, their standard value exceeded the temporary threshold (T_1) of 3.0. In those subnetworks, we found some cases of spoofing; source IP address spoofing, initial TTL spoofing and even packets generated by the traceroute command. Note that traceroute estimates the route to a destination by cheating initial TTL values.

Let us see the detail of packets of the two subnetworks, whose standard deviations were higher than the threshold (T_2), and their source IP addresses are assumed to be spoofed. At first, Figure 4 shows packets from a /24 subnetwork whose standard deviation was 4.5, where ten packets from three source IP addresses were observed. While two hosts (xxx.yyy.152.106 and xxx.yyy.152.179) have similar hop-count (24 and 23), the hop-count of the host xxx.yyy.152.6 was distinctly-different from other two hosts. Based on the difference of the hop-counts among the three hosts, we can assume that any of the three hosts must be spoofing their source IP addresses or the initial TTL.

Subnetwork	Protocol	Port	fTTL	iTTL	HC	DS
xxx. yyy. 152. 106	TCP-SYN	139	104	128	24	-0. 78
xxx. yyy. 152. 106	TCP-SYN	445	104	128	24	-0. 78
xxx. yyy. 152. 106	TCP-SYN	445	104	128	24	-0. 78
xxx. yyy. 152. 106	TCP-SYN	445	104	128	24	-0. 78
xxx. yyy. 152. 106	TCP-SYN	445	104	128	24	-0. 78
xxx. yyy. 152. 179	UDP	24813	105	128	23	-1
xxx. yyy. 152. 6	TCP-SYNACK	1024	222	255	33	1. 22
xxx. yyy. 152. 6	TCP-SYNACK	1024	222	255	33	1. 22
xxx. yyy. 152. 6	TCP-SYNACK	3072	222	255	33	1. 22
xxx. yyy. 152. 6	TCP-SYNACK	3072	222	255	33	1. 22

Average: 27.5 / Standard Deviation: 4.5

Fig. 4 Observed Packets and Deviation Values of a Subnetwork (1)

Subsequently, Figure 5 shows packets a /24 subnetwork whose standard deviation was 3.90. We observed twelve packets from four hosts in the subnetwork. While the hop-count of packets from most hosts is 15, packets from the host yyy.www.3.83 have distinctly-different hop-count, 24. Because of the existence of this host, the standard deviation took a higher score. Same as the previous case, we determined that any of the four hosts spoofed their IP addresses or the initial TTLs.

Subnetwork	Protocol	Port	fTTL	iTTL	HC	DS
vvv. www. 3. 17	TCP-SYN	445	113	128	15	-0. 58
vvv. www. 3. 17	TCP-SYN	445	113	128	15	-0. 58
vvv. www. 3. 17	TCP-SYN	445	113	128	15	-0. 58
vvv. www. 3. 179	TCP-SYN	445	113	128	15	-0. 58
vvv. www. 3. 179	TCP-SYN	445	113	128	15	-0. 58
vvv. www. 3. 179	TCP-SYN	445	113	128	15	-0. 58
vvv. www. 3. 246	TCP-SYN	445	113	128	15	-0. 58
vvv. www. 3. 246	TCP-SYN	445	113	128	15	-0. 58
vvv. www. 3. 246	TCP-SYN	445	113	128	15	-0. 58
vvv. www. 3. 83	TCP-SYN	445	104	128	24	11. 68
vvv. www. 3. 83	TCP-SYN	445	104	128	24	11. 68
vvv. www. 3. 83	TCP-SYN	445	104	128	24	11. 68

Average: 17.25 / Standard Deviation: 3.90

Fig. 5 Observed Packets and Deviation Values of a Subnetwork (2)

In the both cases, the deviation score of the packets from the suspicious hosts were higher than the other. Thus, we can detect spoofed IP packets by deviation scores of each packet.

Finally, we estimate the rate of spoofed IP packet from all of observed packets based on the deviation score. Table 3 examines the rate of spoofed IP packets where the thresholds of deviation scores (T_2) are at 1 ~ 3. As a result, we found that the rate of spoofed IP packets was at most only 1.26% where the threshold was at 1. As a result, we can tell that the rate of spoofed IP packets are in ignorable level for darknet analysis engines.

Table. 3 Number of Spoofed IP Packets

Threshold of Deviation Score (T_2)	Number of Spoofed IP Packets	Rate of Spoofed IP Packets
1	257037	1.26%
2	54593	0.27%
3	39217	0.19%

5 Conclusion

In order to pursue accurate statistics and analysis results, it is important to estimate the amount of spoofed IP packets in darknet traffic. As the first trial to estimate spoofed IP packets in actual large-scale darknet traffic, we proposed a statistical inspection method that distinguishes spoofed packets from darknet traffic. Our method uses a simple statistics, i.e., standard deviation of the TTL values of packets arriving at each subnetwork in darknet in order for detecting irregular subnetworks and finally extracting the spoofed IP packets from the subnetworks. Through a development of the proposed algorithm, we estimated the rate of spoofed packets in practical traffic.

In the evaluation, we have found that at most 1.26% of spoofed packets exist in the darknet traffic. This result supports the accuracy of incident analysis with darknet monitoring.

As the future work, we have to discuss about how to decide appropriate thresholds (T_1 , T_2) for the more precise spoofed packet detection. Furthermore, we will deploy the proposed method to our analysis system in order to get rid of spoofed packets completely in real time.

References

1. Bailey, M., Cooke, E., Jahanian, F., Nazario, J., Watson, D.: The internet motion sensor: a distributed blackhole monitoring system, 12th Annual Network and Distributed System Security Symposium (NDSS05), 2005.
2. Moore, D.: Network telescopes: tracking denial-of-service attacks and internet worms around the globe, 17th Large Installation Systems Administration Conference (LISA'03), USENIX, 2003.

3. Yegneswaran, V., Barford, P., Plonka, D.: On the design and use of Internet sinks for network abuse monitoring, 7th International Symposium on Recent Advances in Intrusion Detection (RAID 2004), LNCS 3224, pp 146 - 165, 2004.
4. Inoue, D., Eto, M., Yoshioka, K., Baba, S., Suzuki, K., Nakazato, J., Ohtaka, K., Nakao, K.: nictcr: An Incident Analysis System toward Binding Network Monitoring with Malware Analysis, WOMBAT Workshop on Information Security Threats Data Collection and Sharing (WISTDCS 2008), pp. 58 - 66, 2008.
5. SANS Internet Storm Center: <http://isc.sans.org/>
6. REN-ISAC: Research and Education Networking Information Sharing and Analysis Center: <http://www.ren-isac.net/>
7. Leurcom.org: Honeypot project, <http://www.leurcom.org/>
8. National Cyber Security Center, Korea: <http://www.ncsc.go.kr/eng/>
9. ISDAS: Japan Computer Emergency Response Team Coordination Center, <http://jpcert.jp/isdas/index-en.html>
10. @police: http://www.cyberpolice.go.jp/english/obs_e.html
11. Ferguson, P., Senie, D.: RFC 2827: Network Ingress Filtering: Defeating Denial of Service Attacks which employ IP Source Address Spoofing, 2000.
12. Park, K., Lee, H.: On the Effectiveness of Route-Based Packet Filtering for Distributed DoS Attack Prevention in Power-Law Internet, ACM SIGCOMM 2001, pp. 15 - 26, 2001.
13. Savage, S., Wetherall, D., Karlin, A., Anderson, T.: Practical Network Support for IP Traceback, ACM SIGCOMM 2000, pp. 295-306, 2000.
14. Snoren, A. C., Partridge, C., Sanchez, L. A., Jones, C. E., Tchakountio, F., Kent, S. T., Strayer, W. T.: Hash-based IP traceback, ACM SIGCOMM 2001, pp. 3 - 14, 2001.
15. Song, D., Perrig, A.: Advanced and Authenticated Marking Schemes for IP Traceback, IEEE INFOCOM 2001, pp. 878 - 886, 2001.
16. Jin, C., Wang, H., Shin, K. G.: Hop-Count Filtering: An Effective Defense Against Spoofed DDoS Traffic, 10th ACM Conference on Computer and Communications Security (CCS 2003), pp. 30-41, 2003.
17. Templeton, S. J., Levitt, K. E.: Detecting Spoofed Packets, DARPA Information Survivability Conference and Exposition 2003, pp. 164 - 175, 2003.
18. Claffy, K., Monk, T. E., McRobb, D.: Internet Tomography, In Nature, 1999. <http://www.caida.org/Tools/Skitter/>
19. Davids, N.: Initial TTL values, 2009. http://members.cox.net/~ndav1/self_published/TTL_values.html
20. Cheswick, B., Burch, H., Branigan, S.: Mapping and Visualizing the Internet, 2000 USENIX Annual Technical Conference , 2000.