Improved Single Packet Traceback Scheme with Bloom Filters

Jia-Ning Luo[1(✉)] and Ming-Hour Yang[2]

[1] Department of Information and Telecommunications Engineering, Ming Chuan University, Taoyuan, Taiwan
deer@mail.mcu.edu.tw

[2] Department of Information and Computer Engineering, Chung Yuan Christian University, Taoyuan, Taiwan
mhyang@cycu.edu.tw

Abstract. In response to the rapid development of the Internet in recent years, numerous new Internet services have been developed to satisfy user needs. However, numerous security issues were also emerged. Because of current Internet protocols, attackers can hide their IP addresses when initiating attacks on targets, especially on the Internet of Things (IoT) frameworks. As a result, discovering the true location of attackers is difficult, especially the attacks are initiates from the personal and private devices that previously lacked Internet connection. Numerous researchers have proposed various packet traceback schemes. Our proposed scheme is a packet marking scheme that uses a 32-bit space in the packet header to record attack paths and the time to live field to decrease the false positive rate of tracebacks. This enables single-packet tracebacks through packet marking and does not require additional storage space on routers for recording attack path data.

Keywords: Packet marking scheme · Packet logging scheme
Hybrid IP traceback · Distributed denial of service · IP spoofing

1 Introduction

In response to the rapid development of the Internet in recent years, numerous new Internet services have been developed to satisfy user needs. However, these services may be shut down by denial of service attacks. Two types of denial of service attacks exist: flood-based attacks and software exploit attacks [5,8]. Flood-based attacks send massive numbers of packets to overload a target's bandwidth or computational or storage capacities, thus rendering the server unable to accept packets from legitimate users. In contrast, software exploit attacks use fewer packets to attack vulnerabilities in a target system that can disable the system and render it unable to provide services. In addition, most routers do not verify the authenticity of the source IP address, and attackers can forge the source IP address to hide their true locations. Therefore, the development of a traceback scheme to determine the true IP address of attacks is crucial.

Traceback schemes can be classified based on the type of attack they can respond to [3,7]. Some can only trace the true source of flood-based attacks [1,2,9,11], but others can trace the true source of both flood-based and software exploit attacks [13,14].

Packet logging traceback schemes [4,15] were developed to trace the source of software exploit attacks. In these schemes, each router traversed by a packet records unaltered information from the packet. This resolves the issue of not being able to trace the source from a single packet because of insufficient packet data space. A router's internal logs can be consulted to determine whether a specific packet passed through a specific router. However, if too many packet digests are stored on a Bloom filter, two separate packets can correspond to the same field and cause false positives. Another problem with packet logging schemes is that they require too much storage space on the router to store packet information. To address this issue, hybrid IP traceback schemes were developed [2,13,14]. In this type of scheme, unused header fields are used to record a packet's path information. After all header fields have been filled, path information is stored on the router. In 2014, Yang [14] proposed using multiple tables rather than a single table to reduce the router storage space required by the HAHIT scheme. However, an additional burden is still placed on routers because of the need to store overflow data on the routers.

In this paper, we proposed a single packet traceback method that allows the source of an attack to be accurately determined with zero router storage load. We used skitter data [12] from the Center for Applied Internet Data Analysis (CAIDA) to perform network topology tests of this traceback scheme. The key contributions of this paper are as follows:

1. Tracebacks can be completed using a single packet.
2. Attacks from multiple sources can be traced simultaneously.
3. No additional router storage capacity is required.
4. The traceback scheme has a zero false negative rate, defined as $\frac{\text{number of unidentified attack paths}}{\text{total number of attack paths}}$.
5. The traceback scheme has a low false positive rate, defined as $\frac{\text{number of wrongly identified attack paths}}{\text{total number of attacks}}$.
6. CAIDA's skitter data from 1998 to 2008, comprising network topology constructed from route information obtained by sending packets from a single origin to multiple destinations, were used to validate our method.

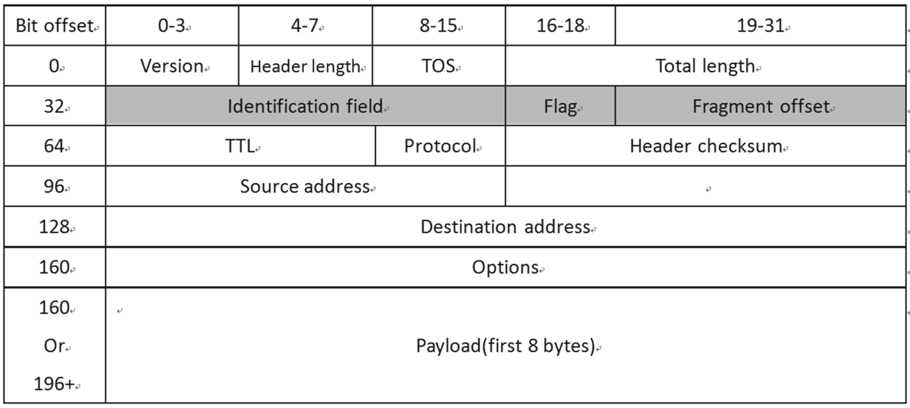
2 Improved Single Packet Traceback Scheme with Bloom Filters

Our scheme assumes that attackers initiate multiple software exploit attacks or one or more distributed denial of service attacks on a single target. Thus, the proposed scheme must be able to simultaneously trace multiple attack sources. In addition, only the final destination of the packet is considered when routers determine which downstream router to forward a packet to; the packet's originating IP is not authenticated. Attackers can spoof the source IP to hide their location, thus allowing all attacks to reach the victim. Furthermore, because we assume our traceback scheme is openly accessible, attackers can disguise their location by designing a forged mark in packet headers in an attempt to mislead the traceback.

To be able to perform tracebacks on attackers with these qualities, and to define the scope of problems that can be resolved by the proposed traceback scheme, our traceback scheme can only work if the following conditions are met:

- Routers are safe from intrusions.
- Routers can identify whether a packet was forwarded from another router or from a local area network.
- All routers support this traceback scheme.
- The target has an intrusion detection system because an attack must be identified before a traceback can begin.

To simplify assumptions and focus on the main proposal of this paper, we assume that the victim has an intrusion detection system to detect when attacks occur, and we do not discuss how the intrusions are detected in the present study. To trace the source of an attack, a packet must have space to record the routers traversed by the packet. As shown in Fig. 1, we use the 32-bit space in the IP header occupied by the identification, flag, and fragment offset fields to mark and store path information. Because only 0.06% to 0.25% of all packets exceed the maximum transmission unit and must be fragmented [6,10], storing path information in these fields will not affect normal network functioning in most cases.



Bit offset	0-3	4-7	8-15	16-18	19-31
0	Version	Header length	TOS	Total length	
32	Identification field			Flag	Fragment offset
64	TTL		Protocol	Header checksum	
96	Source address				
128	Destination address				
160	Options				
160 Or 196+	Payload(first 8 bytes)				

Fig. 1. Packet marker field in the IP header

2.1 Packet Marking Scheme

When router R_i receives a packet P, the router first determines whether the packet came from a local area network. If so, router R_i sets packet P's marker field, $P.mark32$, to 0 and packet P's TTL field, $P.ttl$, to the maximum value (255). These values prevent passing a non-zero marker to the Internet, which would result in the inability to accurately determine when to stop tracing and the difficulty of determining whether a packet has exceeded the hot count caused by different initial TTL values. Table 1 lists all symbols used in this method.

Table 1. List of symbols

R_i	$\{R_1, R_2, \ldots, R_i, \ldots, R_n\}$; a router on the path between the origin and the destination
P	a packet on the Internet
$P.mark32$	A 32-bit marker field in packet P
$P.mark30$	A 30-bit marker field in packet P
$P.mark15$	A 15-bit marker field in packet P
$P.ttl$	The TTL field in packet P's IP header
IP_{R_i}	The IP address of router R_i
$h()$	Hash function
MSM	Maximum size of a mark
BF	Bloom filter
MS	A flag that indicates the marker field is full and no additional marks can be included in this field
$P.id$	An identifier that is identical for all fragments of packet P
$P.no$	The sequence of fragments with the same identifier
SF	A flag that indicates the marker field has been fragmented
$threshold$	The threshold for fragmenting packets
$\|\|$	OR operator
%	Mod operator

As shown in the Algorithm 1, when R_i receives a packet P that is not from a local area network, the router hashes its IP address IP_{R_i} to calculate multiple indexes that need to be marked in $P.mark32$. A bitwise OR operation is performed on 1 and the bits referenced by those indexes in $P.mark32$, and packet P is forwarded to the next router. This process repeats until the packet P reaches its destination.

Algorithm 1. Packet marking algorithm

```
Input : A Packet header $P$
1 if $P$ comes from local network then
2     $P.mark32 = 0$
3     $P.ttl = 255$
4 end
5 $P.mark[h(IP_{R_i})\%MSM]||1$
6 $P.ttl = P.ttl - 1$
7 Forward this packet to the next router
```

Figure 2 shows an example in which five senders send packets through eight routers to the same destination. The IP hash values for routers R_1 through R_8 are

5, 1, 3, 3, 7, 1, 5, and 6, respectively. From Fig. 2, we see that attack packet sent by Attacker 1 was passed to the Internet via router R_1 and traversed routers R_3 and R_6 to reach its final destination. Because this packet traversed three routers, its TTL value $P.ttl$ is 3 less than the maximum, or 252. The string in $P.mark32$, the packet marker field, is 10101000. Three bits were set to 1 to record that this packet has traversed three routers. Attacker 2's packet also traversed three routers, and its TTL value $P.ttl$ is also 3 less than the maximum, or 252. However, because the IP hash values of routers R_2 and R_6 are both 1, only the first and third bits were set to 1, and the string in this packet's $P.mark32$ is 10100000. Attacker 3's packet was passed to the Internet via router R_7 but then took the same path as attacker 2's packet to reach the destination. Thus, its TTL value is 1 less than attacker 2's packet, or 251. However, three bits were set to 1 in the packet marker field, and the string in its $P.mark32$ is 10101000. During packet marking, the router does not distinguish whether a packet is part of an attack. Thus, the target also receives legitimate packets. For example, in Fig. 2, the destination also received packets from Legitimate users 1 and 2 with $P.mark32$ strings of 10100100 and 10010000, respectively.

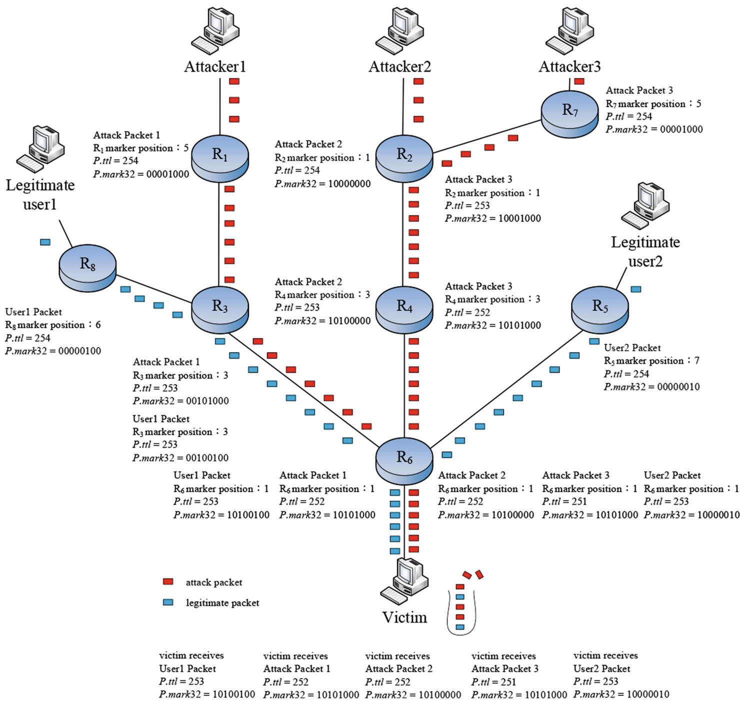

Attacker1
Attacker2
Attacker3
Attack Packet 3 R7 marker position : 5 P.ttl = 254 P.mark32 = 00001000
Attack Packet 1 R1 marker position : 5 P.ttl = 254 P.mark32 = 00001000
Attack Packet 2 R2 marker position : 1 P.ttl = 254 P.mark32 = 10000000
Attack Packet 3 R2 marker position : 1 P.ttl = 253 P.mark32 = 10001000
Legitimate user1
Legitimate user2
R1
R2
R3
R4
R5
R6
R7
R8
Attack Packet 2 R4 marker position : 3 P.ttl = 253 P.mark32 = 10100000
Attack Packet 3 R4 marker position : 3 P.ttl = 252 P.mark32 = 10101000
User1 Packet R8 marker position : 6 P.ttl = 254 P.mark32 = 00000100
Attack Packet 1 R3 marker position : 3 P.ttl = 253 P.mark32 = 00101000
User1 Packet R3 marker position : 3 P.ttl = 253 P.mark32 = 00100100
User2 Packet R5 marker position : 7 P.ttl = 254 P.mark32 = 00000010
User1 Packet R6 marker position : 1 P.ttl = 253 P.mark32 = 10100100
Attack Packet 1 R6 marker position : 1 P.ttl = 252 P.mark32 = 10101000
Attack Packet 2 R6 marker position : 1 P.ttl = 252 P.mark32 = 10100000
Attack Packet 3 R6 marker position : 1 P.ttl = 251 P.mark32 = 10101000
User2 Packet R6 marker position : 1 P.ttl = 253 P.mark32 = 10000010
attack packet
legitimate packet
Victim
victim receives User1 Packet P.ttl = 253 P.mark32 = 10100100
victim receives Attack Packet 1 P.ttl = 252 P.mark32 = 10101000
victim receives Attack Packet 2 P.ttl = 252 P.mark32 = 10100000
victim receives Attack Packet 3 P.ttl = 251 P.mark32 = 10101000
victim receives User2 Packet P.ttl = 253 P.mark32 = 10000010

Fig. 2. An example of 3 attack and 2 legitimate packets forwarded to the same destination

When the target detects a successful denial of service attack from three attacking packets, the path reconstruction algorithm explained in the next section can be used to find the sources of these three attackers.

2.2 Path Reconstruction

When the victim detects an intrusion, the attack packet P is sent to the traceback server to find the source of the attack. The traceback server transmits the fields necessary for tracing the source, $P.mark32$ and $P.ttl$, to a router R_i that is one hop upstream from the victim for path reconstruction. As shown by the algorithm 2, after R_i receives $P.mark32$ and $P.ttl$, it uses its own IP address, IP_{R_i}, in a hash function to calculate its marker position in a BF. If R_i's marker position is unflagged (i.e., $P.mark32[h(IP_{R_i})] = 0$), then router R_i is not in the path of the attack packet P, and R_i no longer needs to assist with tracing the source of the attack. In contrast, if R_i's marker position is flagged (i.e., $P.mark32[h(IP_{R_i})] = 1$), then R_i transmits the BF that includes R_i's marker position to the traceback server. R_i also checks whether $P.ttl$ plus 1 is equal to the initialization value of 255; if so, then R_i is the boundary router for the attacker and the traceback is complete. If not, R_i transmits $P.mark32$ and $P.ttl$ to all other linked upstream routers to continue tracing the source of the attack. After the traceback server receives all attack packet BFs from the routers, the server integrates all BF values to find a router combination that completely matches the $P.mark32$ in the BF and in which the $P.ttl$ is equal to 255. This comprises the routing of an attack packet.

Algorithm 2. Path reconstruction algorithm

```
Input : $P.mark32$, $P.ttl$
1  if $P.mark32[h(IP_{R_i})\%MSM] = 1$ then
2      $P.ttl = P.ttl + 1$
3      Send $((BF[h(IP_{R_i}\%MSM], P.ttl)$ to traceback server
4      if $P.ttl < 255$ then
5          Send $P.mark32$ and $P.ttl$ to all upstream routers
6      end
7  end
8  else
9      End trackback
10 end
```

3 Conclusion

Our proposed scheme, an improved packet traceback scheme with bloom filters, uses a 32-bit space in the packet header to record attack path information.

This enables single-packet tracebacks via packet marking and does not require additional storage space on routers for recording attack path data. Because no data is stored on routers, the router load is reduced compared to packet logging or hybrid IP traceback schemes. In addition, using the TTL field in packet headers decreases the false positive rate caused by marker field conflicts. We also proposed a dynamic marking space to further improve upon the traceback accuracy of the 32-bit marker field. Using 120-bit marking space results in a false positive rate of approximately 20%, which is 1/3 lower than the false positive rate observed in the scheme developed by Takurou et al.; using a 240-bit marking space results in a false positive rate of approximately 6%, which is five times lower than that observed in the scheme developed by Takurou et al. Furthermore, our proposed scheme has a 100% marker delivery ratio and only requires 16 packets to trace the source of an attack with 94% accuracy. Our proposed method successfully achieves the objectives of single packet traceback, zero router storage load, zero false negative rate, and low false positive rate.

Acknowledgements. The authors gratefully acknowledge the support from Ministry of Science and Technology under the grants MOST 105-2221-E-130-005 and 105-2221-E-033-051.

References

1. Aghaei-Foroushani, V., Zincir-Heywood, A.N.: Ip traceback through (authenticated) deterministic flow marking: an empirical evaluation. EURASIP J. Inf. Secur. **2013**(1), 5 (2013)
2. Cheng, L., Divakaran, D.M., Lim, W.Y., Thing, V.L.: Opportunistic piggyback marking for IP traceback. IEEE Trans. Inf. Forensics Secur. **11**(2), 273–288 (2016)
3. Cusack, B., Tian, Z., Kyaw, A.K.: Identifying DOS and DDOS attack origin: IP traceback methods comparison and evaluation for IoT. In: Mitton, N., Chaouchi, H., Noel, T., Watteyne, T., Gabillon, A., Capolsini, P. (eds.) InterIoT/SaSeIoT -2016. LNICST, vol. 190, pp. 127–138. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-52727-7_14
4. Hilgenstieler, E., Duarte, E.P., Mansfield-Keeni, G., Shiratori, N.: Extensions to the source path isolation engine for precise and efficient log-based IP traceback. Comput. Secur. **29**(4), 383–392 (2010)
5. Hussain, A., Heidemann, J., Papadopoulos, C.: A framework for classifying denial of service attacks. In: Proceedings of the 2003 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications, pp. 99–110. ACM (2003)
6. John, W., Tafvelin, S.: Analysis of internet backbone traffic and header anomalies observed. In: Proceedings of the 7th ACM SIGCOMM Conference on Internet Measurement, pp. 111–116. ACM (2007)
7. Prakash, P.B., Krishna, E.P.: Achieving high accuracy in an attack-path reconstruction in marking on demand scheme. i-Manager's J. Inf. Technol. **5**(3), 24 (2016)
8. Prasad, K.M., Reddy, A.R.M., Rao, K.V.: DoS and DDoS attacks: defense, detection and traceback mechanisms-a survey. Global J. Comput. Sci. Technol. **14**(7) (2014)

9. Savage, S., Wetherall, D., Karlin, A., Anderson, T.: Network support for IP traceback. IEEE/ACM Trans. Netw. **9**(3), 226–237 (2001)
10. Stoica, I., Zhang, H.: Providing guaranteed services without per flow management, vol. 29. ACM (1999)
11. Tian, H., Bi, J., Xiao, P.: A flow-based traceback scheme on an as-level overlay network. In: 2012 32nd International Conference on Distributed Computing Systems Workshops (ICDCSW), pp. 559–564. IEEE (2012)
12. UCSD, T.C.: The caida ucsd macroscopic skitter topology dataset. http://www.caida.org/tools/measurements/skitter
13. Yang, M.H.: Hybrid single-packet IP traceback with low storage and high accuracy. Sci. World J. **2014** (2014)
14. Yang, M.H., Yang, M.C., Luo, J.N., Hsu, W.C.: High accuracy and low storage hybrid IP traceback. In: 2014 International Conference on Computer, Information and Telecommunication Systems (CITS), pp. 1–5. IEEE (2014)
15. Zhang, L., Guan, Y.: Topo: a topology-aware single packet attack traceback scheme. In: Securecomm and Workshops, pp. 1–10. IEEE (2006)