Session Based Packet Marking and Auditing for Network Forensics

Omer Demir, Ping Ji and Jinwoo Kim
Graduate Program in Forensic Computing
John Jay College of Criminal Justice
The City University of New York

Abstract

The widely acknowledged problem of reliably identifying the origin of network data has been the subject of many research works. Due to the nature of Internet Protocol, a source IP can be easily falsified which results in numerous problems, including the infamous denial of service attacks. In this paper, two light-weight novel approaches are proposed to solve this problem by providing simple and effective logging and IP-Traceback mechanism: Session Based Packet Logging (SBL) and SYN Based Packet Marking (SYNPM). The contribution of these schemes lies in the fact that they are easy to be implemented with little overhead and are practical under sensitive privacy regulations, since they do not need to access detailed contents of each individual communication session. Currently, SBL and SYNPM approaches support only TCP sessions.

I. Introduction

The nature of the Internet Protocol has been known to make it difficult to reliably identify the actual source of information in cyberspace. Information over the Internet travels in small units, often called packets, and the source of the information – the IP address of the source host - is contained inside the packet with other data contents. Since the IP address of the *original* source computer is often one of the most critical pieces of evidence in on-line investigation, law enforcement agencies and system administrators encounter enormous challenges due to this loophole in Network Forensics.

IP-Traceback is a method for reliably determining the origin of a packet on the Internet [1]. The major problem of determining the *real* source address of an incoming packet is that the Internet routers may replace the source IP address with their own addresses in network packets, and there is no provision in TCP/IP to discover the true origin of a packet [2]. The consequence, therefore, is an easy alteration or a masquerade of source address from an ill-minded sender to conduct criminal activities in cyberspace. One of the worst cases that this loophole can lead to is Denial-of-Service (DOS) attacks. Under DOS attacks, targeted machines suffer significantly from exhausting valuable resources by flooded spoofing messages from senders who often hide their identities through incorrect or altered IP addresses.

In most situations, this kind of criminal activity is extremely hard to prevent and trace once it is committed. The current techniques to trace the IP address of a remote machine are far from those used in telecommunication infrastructure, which is efficient and light-weight in terms of the amount of information to be logged. Specifically, telephone logging systems usually include the time of the call, the number of the caller and receiver, the duration of the call and the place of the call. Good topography of relations in the telephone system can be built based on the logged information. However, in the Internet, no IP-Traceback technology has adopted an approach that is similar to the one used in telephone auditing systems. In addition, current IP-Traceback logging techniques do not provide a long term logging mechanism, and the duration of logging is usually on the order of hours at most. If there were call logs for computer communications, it would be very helpful to law enforcement agencies for later forensics investigation.

In this paper, two novel schemes are proposed to alleviate the challenging problem of IP-Traceback: Session Based packet Logging (SBL) technique and SYN-Based Packet Marking (SYNPM) Technique. Not only are the proposed techniques novel, but they are quite practical solutions for the following reasons. First, both of the proposed techniques effectively provide log information for longer time periods (potentially on the order of months) compared to current IP-Traceback techniques which provide only up to hours of log information. Second, these techniques respect the privacy of communications, given that they do not log detailed contents of each communication session. This makes these schemes ideal to provide effective addressable evidence to the court under sensitive privacy regulations.

The SBL technique, the first proposed scheme in this paper, addresses and solves the problems residing in IP-Traceback from different perspectives compared to the previous approaches. The problem is examined by applying conventional schemes for telephone communication system in Forensics Networks. To this end, existing schema being used in telecommunication investigation are adapted to the Internet Protocol to realize simple and efficient packet logging. In the SBL approach, only critical information in network forensic investigation, the IP addresses and the duration of communication sessions, are recorded by the SYN and FIN packets over the logging period. This technique can be easily deployed at network entities for monitoring TCP traffic and requires much less storage space over conventional schemes. The second proposed technique, SYNPM, is another effective tool for the purpose, but differs from the first approach in the following manner. In the SYNPM approach, routers insert distinguishable *identifiers*, a special signature of a router, in the SYN packet whenever it routes packets. The uniqueness of this scheme from previously proposed packet marking scheme is its efficiency, as it marks only the first SYN packet for a session, as it is the packet that initializes the session and contains sufficient information for IP-Traceback in forensics networks.

The proposed scheme in this paper currently supports only TCP sessions, but these approaches could be extended to UDP connections, which have many inherent network security problems.

The rest of this paper is organized as follows: In Section II, several related IP-Traceback approaches that tried to remedy discussed problems are presented. The Session-Based packet Logging (SBL) scheme is described in detail in Section III. Section IV, details the evaluation results that are derived from experiments of SBL. In Section V, SYN-Based packet marking Protocol is presented. Section VI concludes this paper and discusses potential directions of future work.

II. Related work

There have been several IP-Traceback techniques. Snoeren et al. [3] present a hash-based IP-Traceback approach called Single packet IP-Traceback. In this technique, a hash based scheme is offered for single packet IP-Traceback and it is claimed to require a space of 0.5% of link capacity for recording. This technique enables single packet IP-Traceback, but it has several limitations. First, the original attacking packet has to be found in order to search the audit. The original packet's digest is acquired and then the query is done using that digest. It is not common in Forensic Analysis to have an original copy of an attacking packet. Most of the time only the source IP address and the time of the packet is known. Another problem is false positives. In the case of having the original packet for IP-Traceback, the possibility of false positives will make it difficult to see the log in the court. In order to use them they have to be certain, with no chance of false positives. Memory limitation is another drawback. The amount of time during which queries can be supported is directly dependent on the amount of memory dedicated to its implementation.

Savage et. al. [4] describes a general purpose IP-Traceback mechanism based on probabilistic packet marking in the network. Their approach allows a victim to identify the network path(s) traversed by attack traffic without requiring interactive operational support from Internet Service Providers (ISPs). Moreover, they claim it is possible to trace back the packet after the incident has finished with its scheme. They proposed several different marking algorithms, but there are some problems with their approach. The first is a backward compatibility issue, since the IP header encoding has several practical limitations and it might negatively impact users that require fragmented IP datagram. The second problem is its effectiveness under distributed attack. The implementation of this technique inherently has serious limitations, due to the difficulty in correctly grouping fragments together. Consequently, the probability of misattributing an edge, as well as the amount of state needed to evaluate this decision, increases very quickly with the fan out of an attack. The third problem is with path validation. Some number of the packets sent by the attacker are unmarked by intervening routers. The victim cannot differentiate between these packets and genuine marked packets. Therefore, an attacker could insert "fake" edges by carefully manipulating the identification fields in the packets it sends. The fourth problem is detection of the attack's origin. While this IP-level traceback algorithm could be an important part of the solution for stopping denial-of-service attacks, it is by no means a complete solution. This algorithm attempts to determine the approximate origin of attack traffic. There are a number of reasons why this may differ from the true source of the attack. Attackers can

hide their true identities by "laundering" attacks through third parties, either indirectly (e.g., “smurf attacks” [5] or “DNS reflectors” [6]) or directly via compromised "stepping stone" machines or IP-in-IP tunnels.

Ingress filtering is another approach towards eliminating the ability to forge source addresses. As the name implies, the incoming packets are filtered at the routers. Routers filter the packets depending on the source address and the real network segment that the router is connected to. One of the possible problems of this method is that it requires routers with knowledge of legitimate and illegitimate traffic. Ingress filtering must be deployed as much as possible for best and effective filtering. Another problem is that even if ingress filtering were universally deployed at the customer-to-ISP level, attackers could still forge addresses from the hundreds or thousands of hosts within a valid customer network [7].

Generally, IP-Traceback techniques start from the router closest to the victim and independently test its upward links until they determine all that are used to transmit the attackers’ traffic. This procedure is repeated recursively on the upward routers until the source is founded. This scheme is generally called link testing. There are two varieties of link testing schemes, input debugging and controlled flooding. In input debugging, the victim is supposed to recognize that it is being attacked and develop an attack signature that describes common features of all the attack packets. The victim sends this signature to a network operator. The network operator then installs a corresponding input debugging filter on the victim's upstream output port. This way the router from which this packet has arrived can be determined. The process is then repeated recursively on the upstream router, until the originating site is reached or the trace leaves the ISP's. In case of leaving ISP border, the upstream ISP must be contacted and the same procedure must be repeated. Unfortunately, management is a considerable problem for this method. Controlled flooding is a link-testing IP-Traceback technique developed by Burch and Cheswick which does not require any support from network operators [8]. This technique is called controlled flooding since it tests links by flooding them with large bursts of traffic and observing how this affects traffic from the attacker. The inherent problem with this technique is that controlled flooding itself happens to be a denial-of-service attack

Another technique for IP-Traceback is to log packets at routers and then use data mining techniques to determine the path that the packets traversed ([9], [10]). This technique could be quite effective as it allows tracing an attack even long after the attack has completed. But the problem is the storage of these log files. Another potential question which might be raised by privacy advocate groups is that the log file might contain sensitive data of peoples’ private conversations.

III. Session based IP-Traceback

In this section, a light-weighted IP-Traceback scheme - Session Based packet Logging (SBL) - that can be easily deployed at network entities for monitoring TCP traffic is

proposed. Since storage is one of the major concerns in logging systems of network forensics, SBL is designed to record only the *necessary* contents of a communication session. Specifically, in network forensics, the data transmitted and the session numbers or the fragments of a transaction are not usually needed. All that is necessary are the source and destination addresses and the time stamps of the communication duration. Once the traces of illegal actions in the victim system (e.g. IP address) are known, it is only necessary to check the IP addresses against the log to investigate the attack. Since the purpose here is to log the IP addresses and the duration of *communication sessions* between hosts, it is enough to log the TCP sessions by recording the first session establishment packet (SYN packet) and the session termination packet (FIN packet) for each connection.

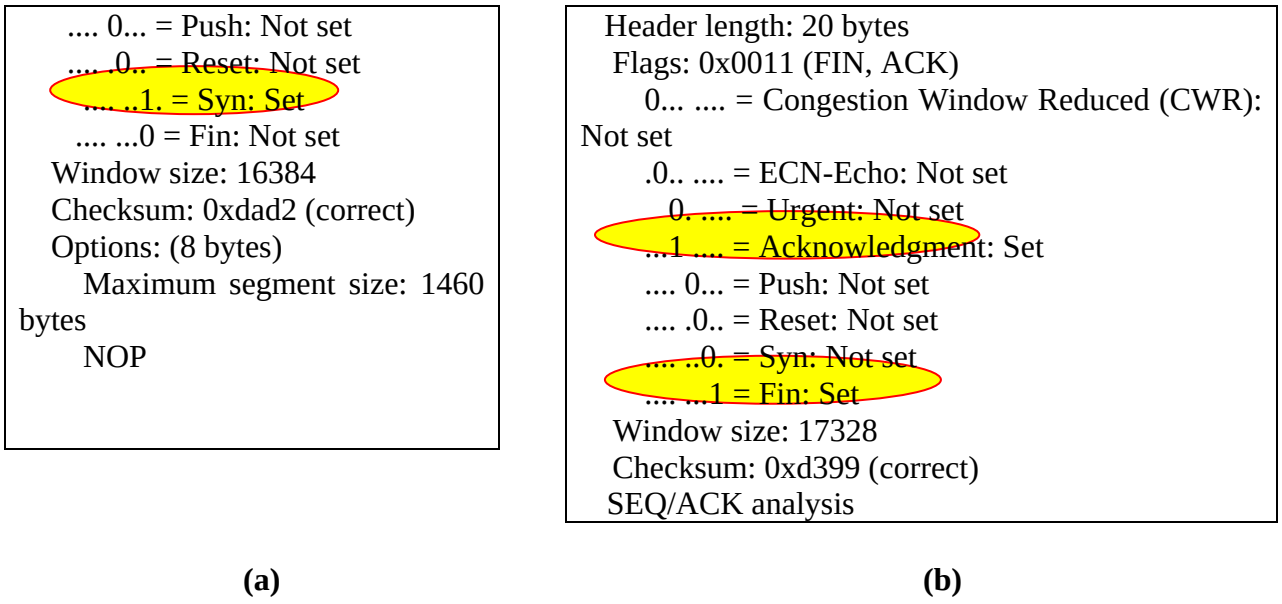
In the SBL scheme, the source IP, Destination IP, incoming port of the router, outgoing port of the router and the time stamps are logged for each logged packet. When keeping the logs, the packet encapsulation which happens at each network entity (e.g., a router) must be considered. When performing encapsulation, instead of replacing a previous IP with its own IP address, each router envelopes the IP packet sent by the neighboring network entity (e.g., end system, adjacent routers) as data payload into a new packet and attaches its IP header to the new packet. In this situation the original source IP (e.g., the IP address from the previous hop) is located in the first four bytes of the payload in the encapsulated packet. Therefore, in the SBL logging approach, the header information and the first four bytes of the data payload of each logged packet are recorded.

The storage space (in the unit of bytes) that each captured packet may need must be considered next. The source and destination IP addresses for each SYN packet must be recorded, which consumes 8 bytes space in total. In addition, as discussed, the first four bytes of the data payload should be recorded. It is safe to assume that there are 256 incoming ports and 256 outgoing ports for each network router. Thus, two more bytes are needed to record the input and output port numbers. Therefore, there are 14 bytes of information to log. It is also important to keep track of the time stamp of each packet. Since one day has 86400 seconds, which can be represented by 3 bytes with a 1/128 sec precision, combined with the 14 bytes information, for each packet, 17 bytes are required to record the information of interest. Provided that a SYN packet size is 62 bytes and a FIN packet is 54 bytes, the ratio of the logged data size to the total size of a logged packet is “17/62” or “17/54,” which saves the storage space significantly. In the experiments, the log files were kept with each data entry as shown in Table 1. Ethereal [11] was used to monitor the traffic, where a SYN packet can be observed, when the corresponding “Syn” field is set to be “1”; and a FIN packet can be observed if the “Fin” field is set to be “1”. Figure 1 shows the partial contents of packet information obtained by Ethereal, where Figure 1(a) corresponds to a SYN packet, and Figure 1(b) corresponds to a FIN packet.

Table 1: Sample Table for Logs

Time	Incoming Port	Source IP	First 4 bytes of IP payload	Destination IP	Destination Port
00000.001	2	X.Y.Z.D	X.X.X.X	T.B.G.V	6
00000.012	3	A.B.C.D	Y.Y.Y.Y	E.F.G.H	7

Figure 1: Sample portion of packet logging file from Ethereal Labs. (a) Observation of a SYN packet; (b) Observation of a FIN packet



(a) **(b)**

Given the packet format of SYN and FIN packets, we define the filter format for SBL session logs as the following: **1.** Set SYN bit to "1" and the Ack bit to "0." This is to get rid of the SYN ACK packets sent by the server back to the client, as it would not provide additional information other than the server's acceptation of the connection; **2.** FIN bit must be set to "1". A sample capture filter for session logging for ethereal is shown below:

"tcp[tcpflags] & (tcp-syn) = 1 and tcp[tcpflags] & (tcp-ack) = 0 OR tcp[tcpflags] & (tcp-ack) = 0"

IV. Experimental results for session-based IP-Traceback

In the experiments, Ethereal was utilized to sniff network communications between one static residence host and other network end hosts. For each round of the experiment, the SYN and FIN packets were captured using the SBL scheme as mentioned in the previous section. The experiment was run 33 times with varied duration for each run. Ideally, the SBL session logging scheme should be implemented at not only the end hosts but also intermediate routers. However, tabbing fast machines to the network to

sniff the aforementioned information is more efficient than putting additional computation burdens on the routers. Thus, the experiments using Ethereal based on end-to-end systems are sufficient to investigate the performance of SBL logging. Table 2 shows the experiment results of the 33 experiments conducted. For each individual experiment, the number of packets captured, the number of logged packets, the size of all captured packets, the size of logged packets, and the duration of the experiment was recorded.

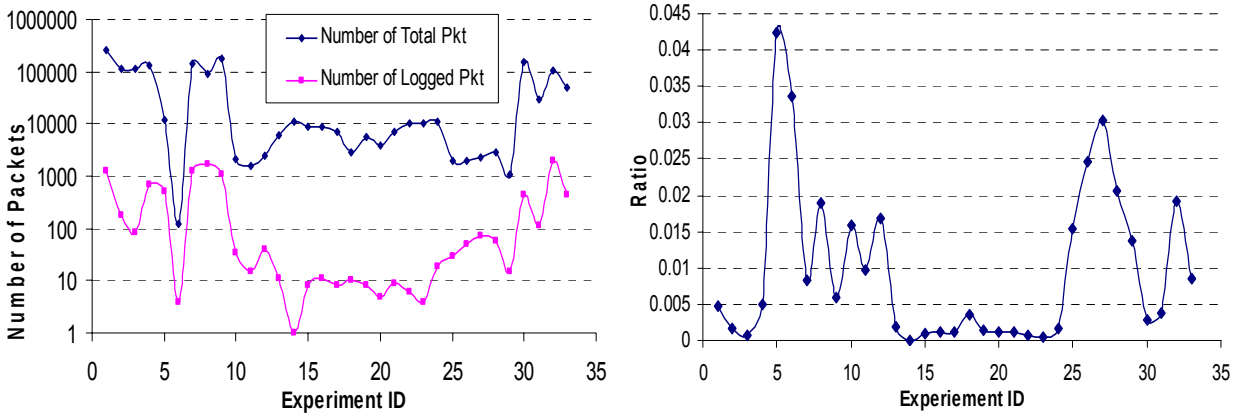
Figure 2(a) shows the number of total packets vs. the number of logged packet for each run of the experiment. Log-scale is used in the y-axis for Figure 2(a) to display the data clearly. Figure 2(b) shows the ratio of the number of logged packets and the number of total packets in different experiments. Furthermore, Figure 3(a) shows the packet size of total packets vs. the packet size of logged packet in log-scale for each run of experiment, and Figure 3(b) shows the ratio of the packet size of logged packets to the total packets for each experiment.

From the Figures, we observe that the average logged packet number and the average logged packet size are very low (0.0097 and 0.0027 respectively). Even for a large amount of total number of packet (or total packet size), the size of logged file remains limited. The maximum ratio between logged number of packets and the total number of packet is only around 0.043, and the maximum ratio between logged packet size and the total packet size is only 0.025.

Table 2: Experiment Data

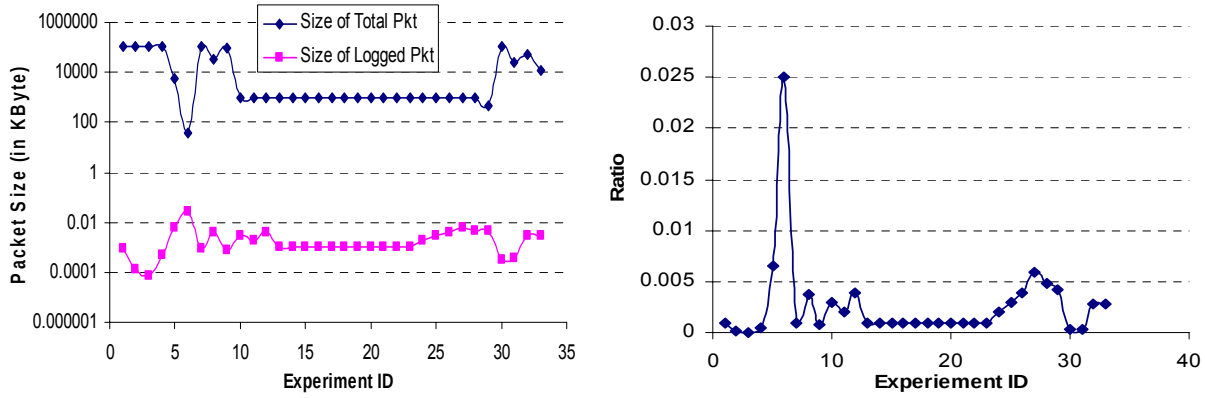
Experiment ID	# of All Packets	# of Logged Packets	Size of All Packets (KB)	Size of Logged Packets (KB)	Duration of the Experiment(hh:mm:ss)
1	263370	1242	102402	95	24:10:21
2	113036	183	102402	14	1:12:07
3	111939	84	102007	7	2:46:58
4	135588	678	102401	52	15:14:14
5	12016	509	5988	39	1:03:57
6	119	4	40	1	0:00:51
7	146143	1221	102401	94	16:28:27
8	88193	1671	33850	128	3:59:37
9	173895	1044	99423	80	6:52:46
10	2190	35	1025	3	0:39:57
11	1529	15	1025	2	0:10:58
12	2373	40	1025	4	22:47
13	5967	11	1025	1	18:23
14	11204	1	1025	1	6:37
15	8554	8	1025	1	5:03
16	9060	11	1025	1	5:20
17	7178	8	1025	1	3:51
18	2863	10	1025	1	1:02
19	5607	8	1025	1	3:12
20	3898	5	1025	1	1:54
21	7232	9	1025	1	3:42
22	9934	6	1025	1	5:34
23	10490	4	1025	1	6:11
24	11091	19	1025	2	13:05
25	1936	30	1025	3	4:17
26	2031	50	1025	4	8:18
27	2318	70	1025	6	3:57
28	2808	58	1025	5	0:26:34
29	1101	15	473	2	0:29:37
30	151003	439	102401	34	3:42:28
31	29551	115	23946	9	2:17:33
32	105662	2018	54827	154	23:59:54
33	50264	428	11971	33	6:11:21

Figure 2: (a) Number of Packets Observed Over Different Experiments (b) The Ratio Between Number of Logged Packets and Total Packets Over Different Experiments (average ratio = 0.0097)



(a) **(b)**

Figure 3: (a) The Size of Packets (in KB) Observed Over Different Experiments (b) The Ratio Between The Size of Logged Packets and The Size of Total Packets Over Different Experiments (average ratio = 0.0027)



(a) **(b)**

This experiment method is suitable for a regular user. However, for large ISPs, the log files may be huge and stored at multiple network entities (e.g., intermediate routers). Nonetheless, from a simple analysis, using SBL for a large amount of data logging, the storage space consumed by the log files is also reasonable. For example, for the network communication with a total data of 782*KB*, an experiment with about 110 hours in duration may be needed. Accordingly, an average log size for the experiment should be about 782/110×24=171*KB* per day. Since the fraction between the size of logged data and the size of a SYN packet is "17 / 62," the average size of the corresponding log file can be estimated as 171*KB*×17/62≈50*KB*. For a 1000 user environment the log file size would be approximately 50*KB*×1000=50*MB* per day, and about 1.5GB for a 30

day month. For a 1 Million user environment it would take approximately 50 GB per day and 1500 GB per month. These values are very reasonable for the log files recording session-based traffic for large ISPs. When compression techniques are used, another 2/3 of the storage space can be saved.

V. A SYN-Based Packet Marking Scheme

The session-based packet logging scheme requires all the routers along the communication path to keep a log for the session traffic, which requires a significant amount of storage resource at each router. When IP-Traceback is performed for a particular session, every single router along the route must be visited and specific data must be retrieved. However, by using the SYN packet based logging scheme, the storage space required for keeping the logs has been significantly reduced and the searching algorithm for identifying specific sessions in the database is simplified, the overall task of log-based IP-Traceback for the entire network is still complicated. Therefore, in this section, another Protocol, the SYN-Based Packet Marking (SYNPM) Protocol is proposed to enhance the performance of the session based IP-Traceback approach.

In the SYN-Based Packet Marking Protocol, instead of logging packets, the routers insert distinguishable *identifiers* in the SYN packet hop-by-hop. The identifiers are special signatures of routers and are appended to the packets to record the fact that the corresponding router is along the communication path of the TCP session. However, in the meantime, no router keeps any log of the passing-by packets. The data storage burden is pushed all the way to the end systems. By doing this, the storage problem existing in a packet logging scheme is resolved easily. Other packet marking techniques have also been proposed along this line of research. However, in the existing packet marking schemes, either all of the packets or statistically chosen packets are marked. In this work, it is proposed that session based packet marking scheme by marking only the first SYN packet for a session be conducted, as it is the packet that initializes the session and contains sufficient information for the IP-Traceback scheme in forensics networks.

Using SYNPM, routers attach their *1-Byte Identifiers* to the SYN packet of each session; thus, every connection is logged with the reverse path to the source, which makes some malicious attacks such as Denial of Service attacks easy to be detected. The SYNPM scheme is presented in detail here.

- At first, the communication initiating peer of a session (e.g., the TCP client) generates the SYN packet for establishing a TCP connection. It is assumed that the initiating TTL value set by the end system is *T,* and the variable *t* is used to denote the changing TTL value of the SYN packet. Thus the SYNPM Protocol originally reserves *T* bytes in the data payload for packet marking to be conducted along the communication path. The *T* bytes of data are initialized as all zeros. For instance, if the TTL value is set to be 255 by the system, then the

initiating peer would reserve 255 bytes of space in the data payload of the first SYN packet and initialize these data into zeros.

- At each router, a piece of software is designed so that the router may perform a very simple modification on the data payload and attach its identifier to the data field of the SYN segment; thus, when the SYN packet arrives at a router, the router attaches a1 byte unique identifier at the last *available* byte field (i.e., the *t-th* byte field) of the reserved bytes (e.g., the 255-th byte for the first hop router when the initial TTL value is set to be 255) in the data. In the meantime, the router decrements the TTL value by one (i.e., $t=t-1$).
- The SYN packet is thereafter marked by routers attaching their special 1-byte identifiers one hop after another onto the *t-th* byte of the reserved packet marking field in data, where *t* is in fact used to keep track of the updated TTL value. This marked packet will eventually be stored in the target computer (e.g., the data sender) as the log entry of the session.

By conducting the above steps, a trace of routers along the communication path of a session has been logged in a reverse order at the end of the reserved data payload in the SYN segment, and a log entry has been created by the destination system of the SYN packet. In order to conduct IP-traceback for a session, an end system takes one log entry as input and checks its recorded final TTL value (stored in variable *t*) which can be used to calculate the number of hops (i.e., *T-t*) that the packet traversed easily. The information that should be used for IP-traceback thereafter is stored from the *t-th* byte to the last recorded byte of the data payload in the logged SYN packet. For example, if for a log entry the variable *t* is recorded as 242, it indicates that from the 242^{nd} byte until the very last recorded byte in the data payload of the corresponding SYN packet, *unique identifiers* are attached by the routers along the communication path of that particular session.

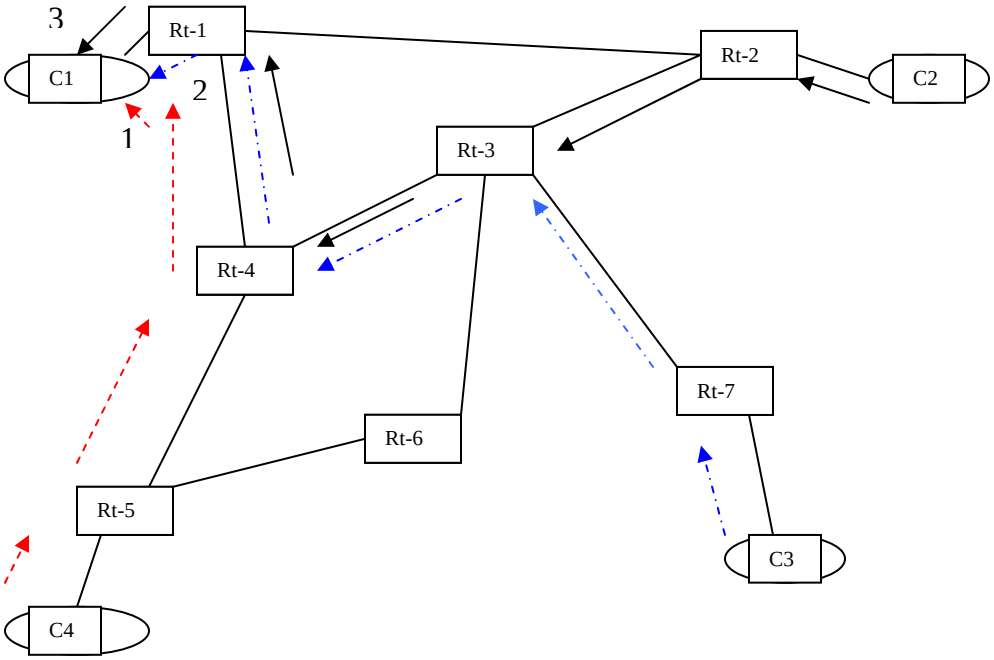
After obtaining the useful data portion from a SYN log, the end system may initiate an IP-traceback process by sending out a *trace command* with a unique query ID (*qID*) to the routers attached directly to the end system. Each router that receives this command extracts the *first* byte of the input data, which is supposed to be the identifier of the first-hop router of the end system, and the router checks whether the identifier matches one of its own. If a router has identified that itself is indeed a router logged by the SYN packet for the corresponding session, the router conducts the following three steps:

1. The router erases its ID (the first byte of the data portion in the query) from the query;
2. The router broadcasts the same *trace command* with the same query ID (*qID*) to the neighboring routers attached to it. In this new trace command, modified query data is attached, where the router has erased its own record from the logged trace data;
3. The router sends its own IP address directly to the query-originating end host with the query number (*qID*).

The above process is conducted hop-by-hop iteratively from the "first-hop" router(s) attached to the query-initiating host until the *original source* recorded by the SYN packet. By having every identified router send back its real IP address, the query originator may recover the original routing trace of a logged session and discover the authentic source of the TCP connection.

Figure 4 shows a sample topology to demonstrate the Packet Marketing mechanism. In Figure 4, *Rt* represents Router, C1-C4 represents computers 1 to 4, and the numbers 1, 2, and 3 beside the arrows represent the connection initialization requests from different computers. The TTL field is originally set as 255.

Figure 4: A sample Topology



According to Figure 4, the data of SYN packets logged at computer C1 could be:

First log: 00000000……0000000 00000001 00000110 00000111

252×8 zeros; 253rd byte, 254th byte, 255th byte

Second log: 0000…000 00000001 00000110 00000011 00001111

251×8 zero; 252nd byte, 253rd byte, 254th byte, 255th byte

Third log: 0000…000 00000001 00000110 00000011 00000010

251×8 zero 252nd byte, 253rd byte, 254th byte, 255th byte

When the second log is queried for IP-traceback, the following steps should be conducted:

1. Extract 255-251 = 4 bytes of data from the log, which is: "00000001 00000110 00000011 00001111", and attach a *qID* to the query data.
2. The end system then will send the *trace commend* with "00000001 00000110 00000011 00001111" plus the *qID* to first-hop router(s) which is Rt1 in the example shown in Figure 4.
3. When it receives the query, Rt1 extracts the first byte of the query data and checks whether itself is the first router recorded. When identified that it is the corresponding router, Rt1 does the following:
    a. sends its IP with the query ID *qID* back to C1;
    b. erases its record from the data which modifies the query data to the new value: 00000110(Rt4) 00000011(Rt3) 00001111(Rt7);
    c. broadcasts this new query to all attached routers; in the example shown by Figure 4, Rt1 sends the modified query to routers Rt2 and Rt4.
4. When they receive the query, both Rt2 and Rt4 extract the first byte of the data and check whether they are the logged router. In this case, Rt2 discovers that it is not the recorded router and simply gets rid of the query. However, Rt4 finds out that it indeed is a logged router of the query; Rt4 thus conducts similar steps as shown in items a. to c. in step 3 and sends out a new modified query to Rt3 and Rt5.
5. Rt3, Rt5 and other routers perform the same set of actions independently, and eventually the query reaches C3 through Rt7.

The query-initiating host at the same time collects the returned IP addresses with the same *qID*, and reconstructs the trace route easily.

VI. Conclusion

Two novel techniques have been presented, Session Based packet Logging (SBL) and SYN Based Packet Marking (SYNPM), to log traffic for TCP sessions for network forensics investigation purposes. Compared to related research on IP-Traceback, SBL and SYNPM adopt the data auditing principle used in traditional telephone networks, which makes proposed schemes easy to implement and has a significant advantage in terms of saving storage space. Specifically, the SBL approach only records the IP addresses and the duration of communication sessions by recording the SYN and FIN packets over the logging period. Thus, SBL logs provided limited, yet sufficient information for Network Forensics investigation to deal with network-based criminal cases. Furthermore, the SBL approach does not need to capture detailed contents of each individual communication session and therefore, protects people's privacy very

well. Using SBL approach, there is no need to install any agent software on the logging machine. The regular logging mechanism with filtering capacity will work fine. The SYNPM technique further enhances the performance of SBL by effectively simplifying the overall task of log-based IP-Traceback through inserting unique signatures of each router during the SYN segment transmission. Since a router does not need to keep any log information of the passing-by packets, the storage problem existing in traditional packet logging scheme is resolved easily. Furthermore, in this scheme only the first SYN packet for a session is marked and recorded, which is different from other pattern marking schemes that usually mark either all of packets or statistically chosen packets. Here, only the first SYN packet is marked, as it is the packet that initializes the session and it contains a sufficient amount of information for IP-Traceback in forensics networks.

The two approaches proposed in this paper focus on logging packets particularly in TCP sessions. However, many network security problems reside in UDP connections, and having session based logs for UDP traffic would be very useful for investigating UDP based attacks. In our future work, we would like to extend our research on Session Based Logging scheme in the direction of recording UDP traffic efficiently.

About the Authors

Omer Demir is a PhD student in the Computer Science program of the Graduate Center of the City University of New York (CUNY). He received his Master's degree from the Forensic Computing graduate program at John Jay College of Criminal Justice, CUNY. Prior joining to John Jay, he worked at the Information Systems Department of the Turkish National Police for seven years as a Police Lieutenant. His e-mail address is odemir@jjay.cuny.edu.

Dr. Ping Ji is an Assistant Professor in the Department of Mathematics and Computer Science at the John Jay College of Criminal Justice, and the Graduate Center of the City University of New York. She has been teaching courses at the graduate program in forensic computing at John Jay College and her research interests include forensic network, network security, network signaling protocols design and network measurement. Her e-mail address is pji@jjay.cuny.edu.

Dr. Jinwoo Kim is an Assistant Professor in the Department of Mathematics and Computer Science at John Jay College of Criminal Justice, The City University of New York. Like Dr. Ji, he is also a faculty at the graduate program in forensic computing at John Jay and his research interest includes security in computing, digital forensics, and embedded systems. His e-mail address is jwkin@jjay.cuny.edu.

References

[1] Wikipedia, www.wikipedia.org

[2] R. T. Morris, "A weakness in the 4.2BSD Unix TCP/IP Software," AT&T Bell Labs, Tech. Rep. Computer. Sci. 117, 1985.

[3] Alex C. Snoeren, Craig Partridge,Luis A. Sanchez, Christine E. Jones, Fabrice Tchakountio, Beverly Schwartz, Stephen T. Kent, and W. Timothy Strayer, "Single-Packet IP Traceback"

[4] S. Savage, D. Wetherall, A. Karlin, and T. Anderson, "Network support for IP traceback," IEEE/ACM Trans. Networking", vol. 9, pp. 226–237, June 2001.

[5] CERT Advisory CA-98.01 "'smurf' IP Denial-of-service attacks" (1998, Jan.). Online. Available: http://www.cert.org/advisories/CA-97.01. smurf.html

[6] CERT Incident Note IN-20004)4 "Denial-of-service attacks using name servers" (2000, Apr.).Online. Available: http://www.cert.org/incident notes/IN-200-04.html

[7] CERT Advisory CA-2000--01 Denial-of-service developments (2000,Jan.). Online. Available: http://www.cert.org/advisories/CA-2000-01.html

[8] H. Burch and B. Cheswick, "Tracing anonymous packets to their approximate source" in Proc. 2000 USENIX LISA Conf., Dec. 2000, pp. 319-327.

[9] G. Sager, "Security Fun with OCxmon and cflowd," presented at the Internet 2 Working Group, Nov. 1998.

[10] R. Stone, "CenterTrack: An IP overlay network for tracking DoS floods," in Proc. 2000 USENIX Security Syrup., July 2000, pp. 199-212

[11] Ethereal Packet Sniffer, http://www.ethreal.com