DESIGN, IMPLEMENTATION, AND EVALUATION OF FRITRACE

Wayne Huang, J.L. Cong, Chien-Long Wu, Fan Zhao, S. Felix Wu
University of California, Davis, California 95616

Abstract: Denial-of-Service attacks are more prevalent than ever, despite the loss of media attention after the infamous attacks that shut down major Internet portals such as Yahoo, eBay and E*Trade in February 2000. In these flood-style denial-of-service attacks, attackers send specially formatted IP packets with forged or true IP source addresses at potentially high packet rates in order to overload/waste the resources of the routers or servers they are attacking. Determining the true sources of an attack stream, depending on the DDoS attack types, is a difficult problem given the nature of the IP protocol however it is often beneficial for the victims to acquire this information in a timely manner in order to stop the attack from further denying service to legitimate users. FRiTrace (Free ICMP Traceback) is an IP traceback implementation that can provide victims of flood-style denial-of-service attacks with sufficient information to determine the true sources of an attack, despite forged IP headers and varying attack architectures. In this paper, we present our design, implementation, and evaluation about FRiTrace.

Key words: Attack source tracing; DDoS.

1. INTRODUCTION

Despite being out of the media spotlight, denial-of-service attacks are more prevalent than ever. One study, aimed at analyzing the byproduct of denial-of-service attacks, inferred using their backscatter analysis technique that 12,805 attacks occurred against over 5,000 hosts in just a three-week period (Moore et al., 2001). Due to the lack of spoofed denial-of-service counter measures, the problem of identifying the true source of an attack in the presence of IP spoofing has become even more relevant. If the victims of an attack can discover the true sources in a timely manner, this will allow administrators to coordinate their efforts in stopping the attack by filtering it directly upstream of the source and may provide sufficient information during the investigative phase to prosecute the perpetrators. This provides two benefits. First, the sooner the sources can be identified, the sooner they can be filtered or shutdown to prevent further loss of or degraded service to legitimate users. Second, accountability provides a psychological deterrent to launching further denial-of-service attacks.

In this paper, we present FriTrace, an IP traceback system using the ICMP Traceback (Bellovin, 2000) and Intention ICMP Traceback (Wu et al., 2001) mechanisms as described in their respective IETF drafts. FriTrace, through the use of specially formatted out-of-band messages, can provide the victims of an attack with sufficient information to construct an attack graph consisting of all administrative domains supporting iTrace or Intention iTrace through which attack packets traversed thus allowing the discovery of the true victims or administrative domains containing the victims. In addition, this paper will discuss various design issues with respect to FriTrace and present the results of our deployment of FriTrace in a test-bed environment.

2. FLOODING ATTACKS

Besides all the existing DDoS attacks, we are considering a new type of statistical attack aimed at IP traceback mechanisms that probabilistically select packets for marking or the generation of an out-of-band message, such as iTrace. With probabilistic schemes, all packets seen at a given router or host are considered equally for marking or generating an iTrace message. Therefore the probability of selecting a packet that is part of an attack stream is based on the percentage of the total packet flow through the router or host that comprises attack packets. Therefore, if it is possible to artificially inflate the background "legitimate" traffic at the router or host, then its possible to decrease the probability of selecting an attack packet, thus reducing the overall effect of the IP traceback mechanism. With a slight modification to the distributed denial-of-service architecture, a statistical attack can be launched. The slaves are split into two logical groups where each slave still attacks the same victim or victims. However, in addition to the attack traffic, each slave in each group sends cover traffic to each slave in the other group.

If there are sufficient slaves, the cover traffic packet rate can be relatively small and thus less detectable. In doing this, depending on the number of slaves, the attackers can generate artificial background traffic thereby reducing the likelihood of attack packets being selected by routers closest to the slaves. In addition, this attack architecture has all of the benefits of the distributed attack including greater anonymity and the ability to generate packet rates of extremely large magnitude.

3. IP SOURCE ACCOUNTABILITY MECHANISMS

The stateless nature of the IP protocol and lack of appropriate filtering by routers makes it an extremely difficult problem to provide source accountability during spoofed denial-of-service attacks. However, there are currently a number of different techniques that provide varying degrees of source accountability. First, source accountability can be achieved through the use of link testing or other flood-based approaches whereby the victim will flood its upstream links to determine which link is passing the bulk of the attack traffic by measuring the drop in the attack traffic rate when the link is flooded. The second class of mechanisms is logging-based, such as Hash-Based IP Traceback (Snoeren, 2001). These mechanisms require routers to store, in some compact format, information about all the packets it has forwarded for a given interval of time. The last class of source accountability mechanisms select packets probabilistically in which to mark the packet or to generate an out-of-band message describing the links from which the packet arrived at the router and which link the packet was forwarded through. These mechanisms rely on the fact that flooding denial-of-service attacks often generate large packet streams for long durations and thus cannot be used to identify the source of a single packet as with the previously mentioned logging mechanisms. Some examples of these mechanisms include the advanced and authenticated marking scheme (Song et al., 2001) based on the original probabilistic packet marking scheme (Savage et al., 2000). Two other probabilistic schemes, and the schemes implemented in FriTrace are ICMP Traceback (Bellovin, 2000) and Intention-Driven ICMP Traceback (Wu et al., 2001).

4. INTENTION-DRIVEN ICMP TRACEBACK

ICMP Traceback (iTrace) is a mechanism where packets are selected randomly according to a probability p for generating iTrace messages. Once a packet is selected, a specially formatted out-of-band ICMP message is sent towards the destination of the selected packet. This message contains useful information such as the IP address and MAC address pair of both the upstream and downstream links of the router that selected the packet. In addition, a timestamp and a portion of the selected packet's payload are included for use by the receiver to correlate messages to received packets. Lastly, the iTrace message contains optional message authentication information to prevent a malicious user from forging iTrace messages. In the IP packet containing the iTrace message, the TTL field is always initialized to 64.

When the victim of a denial-of-service attack receives iTrace messages, they can be ordered to generate a graph from the victim to the attacker's domain including all intermediary routers that support iTrace and generated iTrace messages towards the victim. This ordering can be achieved in two ways. First, two adjacent routers will share a link, where one router's downstream link is the other router's upstream link or vice versa. By analyzing the link information in the iTrace packets, it can be inferred that two routers are adjacent if they share a common link. However, this assumes that all routers in the path support iTrace. To allow for incremental deployment of iTrace, ordering can also be deduced based on the value of TTL field in the IP header containing the iTrace message. All routers correctly following the IP protocol, regardless of whether they support iTrace, will decrement the TTL field before forwarding a packet to the next hop. After generating a graph showing all intermediary routers up to or near the attacking domain, the job of locating the actual source, usually done manually, can be expedited by eliminating the need to contact all intermediary administrative domains to determine the next upstream administrative domain passing the attack traffic.

The original ICMP traceback scheme provided sufficient information for the victims of flooding denial-of-service attacks to determine the nearest administrative domain supporting iTrace to the actual attackers, regardless of potentially spoofed IP addresses in the attack packets. In addition, authentication codes can be used to prevent malicious users from forging iTrace messages to obstruct or mislead the construction of the attack graph.

However, the original iTrace scheme is susceptible to the statistical denial-of-service attack described earlier. Specifically, iTrace will select packets randomly with probability p at any given router supporting iTrace. Therefore, the probability of selecting attack packets is simply $p(R_a)$, where R_a is the ratio of attack traffic to total traffic. By artificially increasing the total traffic without affecting the attack traffic, a malicious user can reduce R_a, and thus reduce the overall probability of iTrace selecting attack packets.

Intention-Driven ICMP Traceback (Intention iTrace) is an enhancement to iTrace where packet selection factors in an administrative domain's desire to receive iTrace messages. For each administrative domain, this desire is propagated through the use of an intention bit associated with each network prefix the administrative domain, or autonomous system (AS) advertises through BGP. Using a network intrusion detection system or even manually with an administrator, the intention bit can be dynamically set if the AS detects it is under a denial-of-service attack. BGP will automatically propagate this information to all BGP-speaking routers on the Internet for use with Intention iTrace.

The Intention iTrace consists of two different packet selection algorithms, however both algorithms achieve an average packet selection probability of p while allowing the administrator to specify the percentage p_i of selected packets to subject to the Intention iTrace criteria that an iTrace message only be generated for destinations wishing to receive iTrace messages. For all other $1 - p_i$ selected packets not subject to the Intention iTrace criteria, normal iTrace messages are generated as the normal iTrace.

The first algorithm separates traffic into two classes by analyzing all of the intention bits that are known to a particular router's routing information base (RIB). Packets with destination prefixes having an intention bit of 1 in the RIB will be considered *intention traffic* and all other packets will be considered *normal traffic* with R_{int} representing the ratio of intention traffic to total traffic. Both traffic classes will consider packets equally for normal iTrace, or a probability of $p(1 - p_i)$, however the intention traffic must also be considered for Intention iTrace. Intuitively, this probability should be $p(p_i)$, however this probability would be static. In the ideal case, if R_{int} is extremely small then the probability for generating an Intention iTrace message should be greater than $p(p_i)$, otherwise it is likely that packets from the intention traffic will not be considered. Similarly, as R_{int} approaches 1, the probability for selecting a packet to consider with Intention iTrace should be as close to $p(p_i)$ as possible or packets will no longer be considered for normal iTrace. To strike a balance between packets being considered for normal iTrace and Intention iTrace, the probability for considering a packet for Intention iTrace from the intention traffic is $p(p_i/R_{int})$.

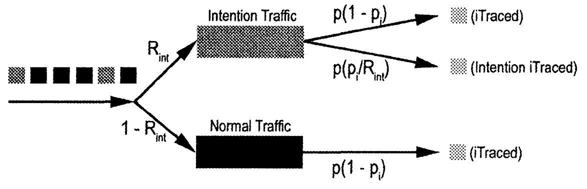

Figure 1. Intention iTrace Scheme 1 (IIS#1) Decision Process

Therefore, the total probability for an intention traffic packet to be considered for iTrace message generation is $P_i = p(1 - p_i) + p(p_i/R_{int})$ and $P_n = p(1 - p_i)$ for packets in the normal traffic class. The total overall probability for generating an iTrace message can then be expressed as $P_{avg} = R_{int} * P_i + (1 - R_{int}) * P_n = p$.

The second algorithm does not segregate traffic into traffic classes and instead, first selects a packet for either iTrace or Intention iTrace consideration with probability p as in the normal iTrace scheme. The selected packet is then sent to the appropriate decision module based on the probability pi, where with probability $p(p_i)$ the packet will be considered by the Intention iTrace decision module and with probability $p(1 - p_i)$ the packet will be considered by the normal iTrace decision module. However, within the normal iTrace decision module, the packet's destination prefix is looked up in the router's RIB and if the intention bit is set, the packet will still be considered under the Intention iTrace criteria. If the destination prefix in the RIB does not have its intention bit set, then the packet generates a normal iTrace message. Based on the ratio R_{int} of intention traffic to normal traffic, it is given that approximately R_{int} of packets sent to the normal iTrace decision module will be considered for Intention iTrace and $1 - R_{int}$ of packets will generate normal iTrace messages.

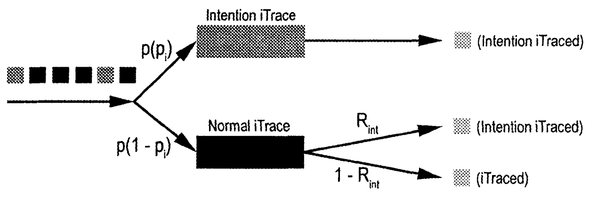

Figure 2. Intention iTrace Scheme 2 (IIS#2), Decision Process

The total probability that a selected packet will be considered using the Intention iTrace criteria is $P_i = p(p_i) + p(1 - p_i) * R_{int}$ and the probability that a selected packet will generate a normal iTrace message is $P_n = p(1 - p_i) * (1 - R_{int})$. The average probability of selected packet being considered for iTrace is thus $P_{avg} = P_i + P_n = p$.

Thus for both Intention iTrace algorithms, the average probability for a packet to be considered for generation of an iTrace message is p, but both algorithms allow administrators to determine the amount of resources to be dedicated to Intention iTrace and provide a method for randomly mixing both normal iTrace and Intention iTrace schemes. Using Intention iTrace will still allow the victim of a statistical denial-of-service attack to receive useful iTrace messages despite the artificial increase in background traffic.

In addition, Intention iTrace will still generate normal iTrace messages to destinations whom may be under a denial-of-service attack but that have not detected the attack and thus have not specified their desire to receive iTrace messages.

5. FRITRACE DESIGN AND IMPLEMENTATION

FriTrace is our open-source implementation of an IP traceback suite that supports both iTrace and Intention iTrace as well as authentication to prevent spoofed iTrace messages, source IP address spoof detection and source iTrace. The iTrace and Intention iTrace mechanisms were originally designed to work in router platforms where knowledge of upstream and downstream links would be used in the generation of an iTrace message. As such, we developed FRiTrace as a Linux 2.4 Loadable Kernel Module (LKM) to be run on Linux-based routers to provide iTrace and Intention iTrace support. However, most administrative domains use appliance-based routers such as those made by Cisco and Juniper Networks, and therefore the FriTrace LKM would not be as useful in those domains without replacing the appliance-based routers with Linux-based routers. In order to allow such domains to support iTrace and Intention iTrace, a passive implementation of FriTrace was developed that would listen on a network in order to select packets to be iTraced instead of selecting packets while forwarding them, such as in the LKM implementation. In this paper, only the passive approach is discussed.

5.1 FriTrace Probabilistic Packet Selection

Regardless of whether normal iTrace or Intention iTrace is activated in FriTrace, both modes require the ability to probabilistically select packets given a probability p. The naïve approach to the selection method would be to maintain a counter and to select a packet every time the counter modulo $1/p$ was equal to $1/p - 1$. While this maintains an average probability of p to select a packet, it is not secure to specially timed attacks. By selecting packets statically using this method, a crafty attacker can setup a flooding denial-of-service attack where the attack packets are sent periodically with the same frequency. If timed correctly, this attack can exploit the static packet selection technique to avoid selection of packets from the attack stream.

To counter this timed attack, packet selection must occur randomly with probability p. This can be implemented by computing a random number for each packet and selecting the packet if the generated random number is less than *p*. Assuming a uniformly distributed pseudo random number generator (PRNG), attack packets will be selected at random with a probability *p*. However this approach suffers in that a random number must be generated for each packet, thus causing per-packet overhead, which should be avoided in order not to affect packet-processing rates.

In FriTrace, random packet selection is achieved by pre-computing a series of *n* random numbers using a uniformly distributed PRNG for integer values between *0* and $n(1/p) - 1$, inclusive. The window of *n* random numbers is then sorted. For each packet that is read, a counter is incremented. When the counter value equals a value from the random number window, a packet is selected. When the counter value reaches $n(1/p)$ it is reset to zero and a new window of random numbers is computed. This approach allows packets to be selected at random with a probability *p*, without incurring the overhead of computing a random number for each packet.

5.2 Source iTrace

IP traceback mechanisms have primarily focused on allowing the victims of denial-of-service attacks to discover the true sources of their attackers. However, depending on the specific attack used, there may be other administrative domains adversely affected. Specifically, the use of most spoofed denial-of-service attacks causes the victim to generate responses towards the IP addresses that were spoofed. Therefore, if iTrace messages are also generated with an independent probability *p* towards the source of a packet, this would allow administrative domains to discover that their address space is being used by a malicious user in a spoofed denial-of-service attack.

Another distributed denial-of-service attack architecture involves the use of *reflectors*, or innocent hosts on the Internet used to reflect attack packets towards a victim. Instead of the slaves generating the attack traffic directly towards the victim, they instead reflect their attacks off of random hosts on the Internet by using the functionality of the TCP and UDP transport layer protocols. For example, if an attacker sends a SYN packet to a reflector with a forged source address of the victim, any response generated by the reflector will be sent to the victim. With the use of iTrace or Intention iTrace the victim would simply trace back to the reflectors used in the attack, which is not especially useful to the victim. However, with the use of Source iTrace, the packet stream from the slaves to the reflectors will be iTraced with the iTrace messages being sent towards the victim. This information can then be used to discover the slaves in a reflector-based denial-of-service attack.

FriTrace supports the Source iTrace feature and maintains an independent probability to generate Source iTrace messages.

5.3 The Passive Approach and its Implementation

The passive implementation of FriTrace is run entirely in user space as a single process. There are three main components within the passive FriTrace implementation.

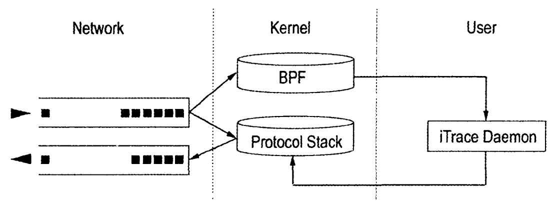

Figure 3. FriTrace, Passive Architecture

The first is the network stack interface using *libpcap* that allows FriTrace passive access to all packets on a network segment using the BSD Packet Filter (BPF) (McCanne et al., 1993) or packet socket functionality. The *libpcap* interface then calls the iTrace decision module for each packet that is observed on the network, similar to the functionality of Netfilter in the active implementation. The iTrace decision module then applies the appropriate packet selection algorithm and if a packet is selected to be iTraced, it will be sent to the iTrace generation component prior to returning control to the *libpcap* interface. The second component, or iTrace generation component, performs the logging of packet information to syslog and generates iTrace messages. Lastly, the third component validates and responds to authentication requests.

It is important to note that because packets are read from the network passively, there is no accompanying information that passive FriTrace can use to truly identify either an upstream or downstream IP link. However, packets intercepted through *libpcap* retain their link layer information, which provides passive FriTrace with information that can be used to generate a MAC address upstream link. Since passive FriTrace will be operating on a network segment, the receiver of an iTrace message can determine the administrative domain from which the message was sent based on the IP address of the host running passive FriTrace. Once the offending administrative domain is located, the MAC address pair can be used to locate individual machines from within that domain.

6. FRITRACE EVALUATION

The experimental setup consisted of four hosts, a 100 Mbps hub and a router. The *Attacker*, *Background* and *FRiTrace* hosts were Pentium III 700 MHz servers with 512 MB of memory and a 100 Mbps network interface card, running Red Hat Linux release 7.0 with the Linux 2.2.16 kernel. All three hosts are connected to a 100 Mbps hub, which is in turn connected to a router. The *Victim* host is a dual Pentium III 1000 MHz with 512 MB of memory and a 100 Mbps network interface card, running Red Hat Linux release 7.2 with the Linux 2.4.7 kernel.

Table.1 – Experiment Setup, Experiment Parameters

#	Mode	p_i	Background	Attack Rates
1	Normal	*n/a*	16,000 pps	1,000 – 16,000 pps
2	Normal	*n/a*	2,000 – 16,000 pps	2,000 pps
3	IIS#1	1/10	16,000 pps	1,000 – 16,000 pps
4	IIS#1	1/10	2,000 – 16,000 pps	2,000 pps
5	IIS#1	1/4	16,000 pps	1,000 – 16,000 pps
6	IIS#1	1/4	2,000 – 16,000 pps	2,000 pps
7	IIS#1	1/2	16,000 pps	1,000 – 16,000 pps
8	IIS#1	1/2	2,000 – 16,000 pps	2,000 pps
9	IIS#2	1/10	16,000 pps	1,000 – 16,000 pps
10	IIS#2	1/10	2,000 – 16,000 pps	2,000 pps
11	IIS#2	1/4	16,000 pps	1,000 – 16,000 pps
12	IIS#2	1/4	2,000 – 16,000 pps	2,000 pps
13	IIS#2	1/2	16,000 pps	1,000 – 16,000 pps
14	IIS#2	1/2	2,000 – 16,000 pps	2,000 pps

As shown above, separate experiments with different parameters were conducted using FRiTrace on this experimental topology. In the first experiment, the background traffic was fixed at 16,000 pps with the attack rate doubling with each run from 1,000 pps to 16,000 pps. In the second experiment, the attack rate was fixed at 2,000 pps with the background traffic rate doubling with each run from 2,000 pps to 16,000 pps. These experiments were conducted with normal iTrace and both Intention iTrace schemes with p = 1/20,000. For the Intention iTrace schemes, p_i was set to 1/10, 1/4 and 1/2. Each experiment was conducted a total of ten times.

Using a modified FRiTrace daemon, information about each iTrace message generated was stored to a log file. This information contained the iTrace packet size and the destination IP address. Once all of the experiments were conducted, the log files, one generated for each run, were post-processed to obtain the total number of iTrace messages generated, the total number of useful iTrace messages, the total packet overhead of FRiTrace and the total data overhead of FRiTrace.

One method of determining the effectiveness of FRiTrace as an IP traceback suite is to measure how many useful iTrace messages were generated with the different schemes, where a useful iTrace message is defined as one generated from a packet belonging to a denial of service attack stream. The higher the number of useful iTrace messages received by the victim of an attack, the more confidence the victim can place on the path generated by post-processing the iTrace messages. Once the data was compiled, all three schemes were compared to each other to show the relative effectiveness of one scheme versus another and to also show how the p_i parameters relates to the overall effectiveness of the Intention iTrace schemes.

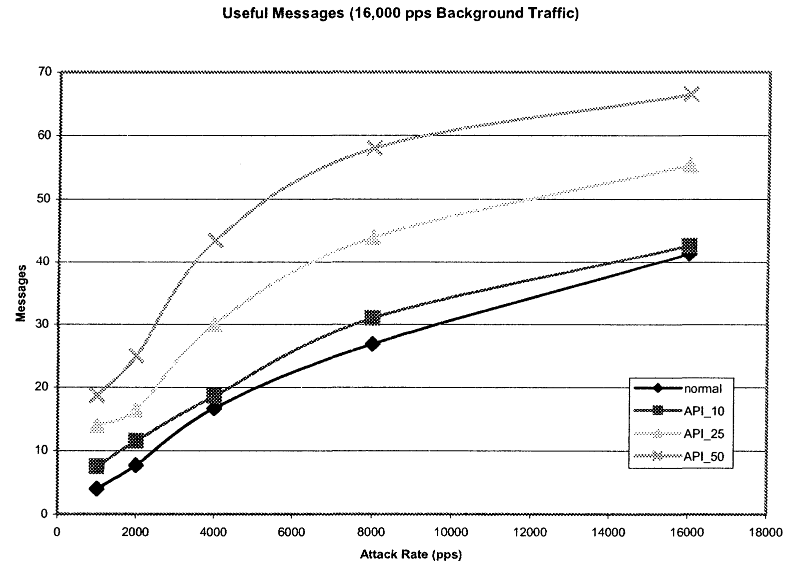

Figure 4. Number of Useful Messages for Normal vs IIS#1

The first comparison was between normal iTrace and Intention iTrace scheme 1 with a fixed background rate and shows how the number of useful messages is affected by changes in the attack rate. As can be seen from this comparison (Figure 4), overall the Intention iTrace scheme 1 showed a higher number of useful iTrace messages. Based on the functionality of Intention iTrace this behavior was expected. In addition, as p_i was increased from 1/10 to 1/2, the number of useful messages also increased. In scheme 1, the Intention iTrace decision module classifies packets into two groups based on the packets' destination addresses. As a result, when the ratio of Intention traffic to total traffic, R_i, is small, the relative increase in the number of useful messages is also small when comparing normal iTrace to Intention iTrace and also between the Intention iTrace runs with varying p_i. However, as R_i increases, the increase in the number of useful messages becomes more realized. Lastly, with Intention iTrace scheme 1, there is roughly a linear relationship between the number of useful messages and p_i at high attack rates. For example, the number of useful messages roughly doubled when p_i was increased from 1/4 to 1/2.

We have also compared the normal iTrace scheme and IIS#2. This scheme does not classify packets into traffic groups, but instead generates a trigger with probability p, after which p_i decides whether the packet should be treated as an Intention packet or a normal packet. Because of this, the change in the number of useful messages as the attack rate increases is not affected by the ratio of Intention traffic to total traffic, R_i.

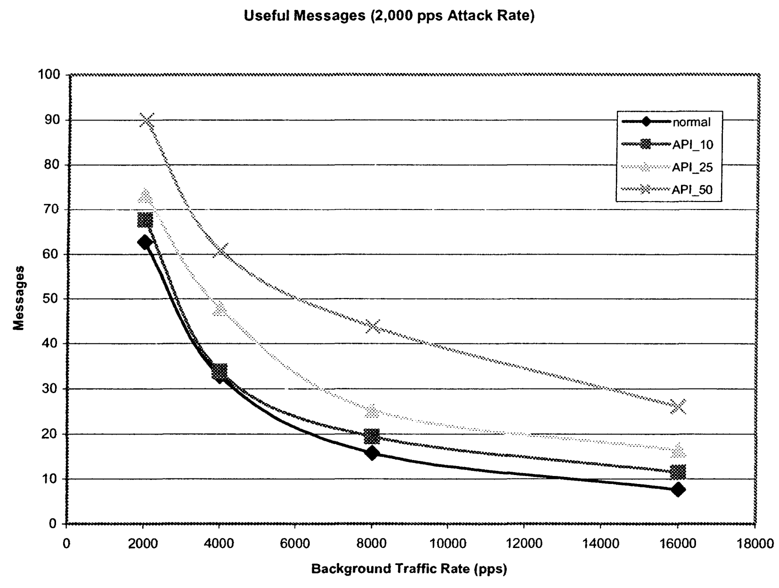

Figure 5. Normal vs. IIS#1, with Variable Background Traffic

Figure 5 shows how the number of useful messages is affected by an increase in the background traffic rate in all the schemes. This experiment is important for determining how effective a statistical denial of service attack is on hiding the true attack packets. In general, as the background traffic rate increases, there is a decrease in the number of useful messages in all runs. However, the rate of decrease as the background traffic rate increases varies for each run. For example, using normal iTrace, as the background traffic rate doubles, the number of useful messages decreases by half. But for Intention iTrace, the rate of decrease decreases as p_i is increased despite increasing the background traffic rate. This can be seen from the scheme 1 run with $p_i = 1/2$. An initial doubling of the background traffic rate drops the number of useful messages from about 90 to 60, a decrease of about 1/2. The next doubling of the background traffic rate yields a decrease of only 1/3, and so on. In general, as p_i is increased towards 1.0, the rate of decrease approaches 0. This is consistent with the functionality of Intention iTrace, which only generates iTrace messages for prefixes with an intention bit of 1. We have also performed the same experiments with IIS#2, and the results are similar.

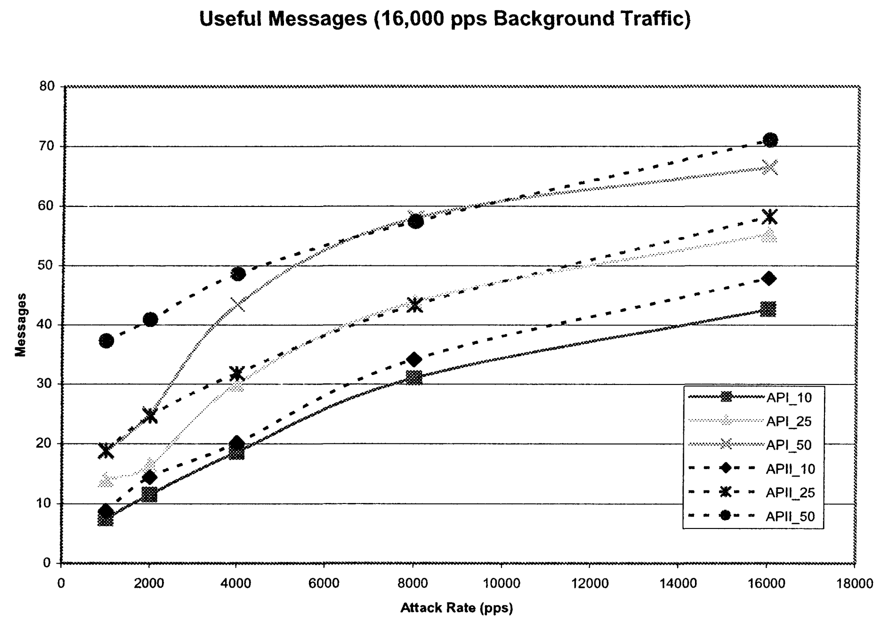

Figure 6. Number of Useful Messages for IIS#1 versus IIS#2.

From these comparisons, the most important result is that both Intention iTrace schemes are resilient to statistical denial of service attacks because as the background traffic rate increases, the rate of decrease of the number of useful messages also decreases. In addition, as p_i is increased, the amount of decrease increases until $p_i = 1$ where a decrease in the number of useful messages no longer occurs.

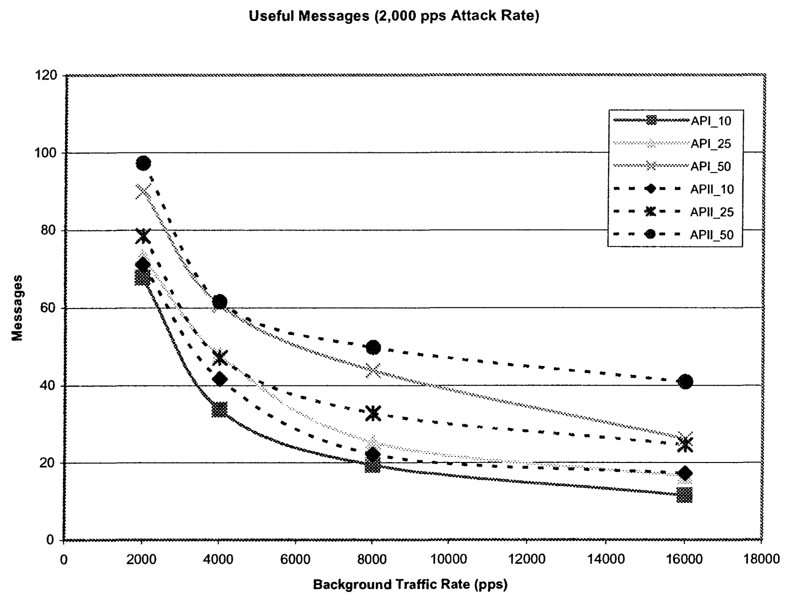

Figure 7. Scheme 1 vs Scheme 2, with Variable Background Traffic

The next comparisons show how the two different Intention iTrace schemes compare in terms of useful messages when there is a fixed background traffic rate of 16,000 pps and a variable attack rate and when there is a fixed attack rate of 2,000 pps and a variable background traffic rate.The solid lines in Figure 6 represent scheme 1 results and the dashed lines represent scheme 2 results with varying p_i. As can be seen, scheme 2, in general generated more useful messages than scheme 1 under identical testing conditions. From this comparison, the effect of classifying traffic into groups and selecting packets independently is apparent by the significantly lower number of useful messages produced by scheme 1 compared to scheme 2 when the ratio of Intention traffic to total traffic R_i is low. For higher values of p_i, it appears that scheme 1 achieves its maximum usefulness around $R_i = 0.5$ when compared to scheme 2. Up to and following $R_i = 0.5$, the number of useful messages decreases compared to scheme 2.

With variable background traffic, as in Figure 8, scheme 2 achieves a higher number of useful messages at all levels of p_i and for all values of R_i. However, more important to note is the fact that scheme 2 is much more resilient to statistical denial of service attacks, which artificially inflate the background traffic rates. Not only do the scheme 2 results generate more useful iTrace messages, but the rate of decrease of the number of useful messages as the background traffic rate increases is much smaller in scheme 2 than in scheme 1. The quicker stabilization of scheme 2 even for small R_i indicates that much more of the background traffic is ignored by scheme 2 and that further increases of the background traffic rate past 16,000 pps would yield only slight decreases in the number of useful messages.

7. SUMMARY

To show the overall effectiveness of FRiTrace, several experiments were conducted to measure the number of useful iTrace messages and the total traffic overhead generated by the various FRiTrace modes. For the Intention iTrace schemes, p_i was also varied to observe its effects on the number of useful messages and total traffic overhead. Our experiments showed that even when the attack traffic represents a small percentage of the overall background traffic that FRiTrace is still able to generate a substantial amount of useful iTrace messages. While it only takes a single message from any of the routers along the path from the victim to the true attacker, the more messages received from the routers along the path the greater confidence the victim can place in the final traced path to the attacker or attackers. By increasing p_i throughout the first set of experiments, it was shown that not only does the Intention iTrace scheme generate more useful messages, but there is also a near-linear relationship between p_i and the relative amount of useful messages, regardless of the attack rate or background traffic rate. Lastly, the first set of experiments showed the true benefits of the Intention iTrace schemes. With a fixed attack rate, as the background rate was increased, the number of useful messages did not drop linearly as in the normal iTrace scheme. Instead, the decrease in the number of useful messages decreased as either p_i or R_i increased and eventually reached a stable value where additional increases in the background traffic would still generate approximately the same number of useful messages. This behavior was not expected or observed in normal iTrace, where there was a linear relationship between the decrease in useful iTrace messages and the increase in the background traffic rate.

Our experiments also showed that even in the worse case, FRiTrace only generated approximately 0.0064% total network traffic overhead. This equates to an increase of approximately 6.5 kbps for every 100 Mbps of traffic analyzed by FRiTrace, a very small amount of overhead. In addition, the amount of overhead is a function of the probability to generate an iTrace message p. A decrease in p would result in a decrease of the network traffic overhead. This parameter is intended to be set by the network administrators of the domain being analyzed based on traffic patterns observed on the network. Aside from the overall low overhead introduced by FRiTrace, several logical trends were observed. As p_i increased for both Intention iTrace schemes, the amount of total traffic overhead also dropped. As R_i increased, total traffic overhead stabilized regardless the increase in either the background traffic rate or attack rate. Lastly, it was observed that scheme 2 produced slightly higher overhead than scheme 1.However, this is a result of the higher efficiency of scheme 2 and is also shown in the first set of experiments where scheme 2 generated more useful iTrace messages than scheme 1 regardless of the parameters to the experiments.

These initial results show that FRiTrace can be an effective tool for allowing victims of denial of service attacks to discover the true attackers despite IP spoofing. While these experiments were conducted on a simple test network representing only a single FRiTrace host, the results would simply scale up to the number of hosts or routers along the attack path that supported iTrace because the probability p to generate an iTrace message is independent of all other hosts or routers along the path. Further, since iTrace messages are sent out of band, there is no interference from downstream routers on previously generated iTrace messages as in the probabilistic packet marking schemes described in an earlier chapter. Because of this, as more routers along the attack path support iTrace, more iTrace messages will be sent to the victim allowing faster path reconstruction and correlation.

REFERENCES

Bellovin, S. M., ICMP Traceback Messages, *IETF Internet Draft*: draft-ietf-itrace-01.txt, October 2000.

McCanne, S, and Jacobson, V., The BSD packet filter: a new architecture for user-level packet capture, in *Proceedings of the Winter 1993 USENIX Conference*, January 1993.

Moore, D, Voelker, G. M., and Savage, S. Inferring Internet Denial-of-Service Activity, in *Proceedings of the 10th USENIX Security Symposium*, August 2001.

Savage, S., Wetherall, D., Karlin, A., and Anderson, T., Practical Network Support for IP Traceback, in *Proceedings of ACM SIGCOMM,* August 2000.

Snoeren, A.C., Partridge, C., Sanchez, L.A., Jones, C.E., Tchakountio, F., Kent, S.T., and Strayer, W.T., Hash-Based IP Traceback, in *Proceedings of SIGCOMM,* August 2001.

Song, D.X., and Perrig, A., Advanced and Authenticated Marking Schemes for IP Traceback, in *Proceedings of IEEE Infocom,* April 2001.

Wu, S. F., Zhang, L., Massey, D., and Mankin, A., Intention-Driven ICMP Traceback, *IETF Internet Draft*: draft-ietf-itrace-intention-00.txt, November 2001.