A Study of Mass-mailing Worms

Cynthia Wong, Stan Bielski, Jonathan M. McCune, Chenxi Wang
Carnegie Mellon University
5000 Forbes Avenue, Pittsburgh, PA, 15213
{cindywon, bielski, jonmccune, chenxi}@cmu.edu

ABSTRACT

Mass-mailing worms have made a significant impact on the Internet. These worms consume valuable network resources and can also be used as a vehicle for DDoS attacks. In this paper, we analyze network traffic traces collected from a college campus and present an in-depth study on the effects of two mass-mailing worms, SoBig and MyDoom, on outgoing traffic. Rather than proposing a defense strategy, we focus on studying the fundamental behavior and characteristics of these worms. This analysis lends insight into the possibilities and challenges of automatically detecting, suppressing and stopping mass-mailing worm propagation in an enterprise network environment.

Categories and Subject Descriptors

D.4.6 [**Operating Systems**]: Security and Protection—*Invasive software*

General Terms

Security

Keywords

Internet Worms, Network Security, Traffic Analysis

1. INTRODUCTION

In recent years the Internet has experienced an increasing number of malicious programs propagating through electronic mail. High-profile worms such as *I love you* [1] and *Sircam* [2] are but a few historic al examples. The more recent outbreak of *MyDoom* [4] substantially degraded network services and was directly responsible for a massive Distributed Denial of Service (DDoS) attack. An earlier outbreak, *SoBig* [3], utilized sophisticated hybrid propagation techniques to replicate over network shares and emails, resulting in a rapid infection rate that surpassed those achieved previously. Both SoBig and MyDoom disrupted normal network operations and caused unexpected downtime and an increase in associated IT expenses.

The severity of these attacks can be attributed to a numbe r of factors, including efficient mechanisms for targeting victims and insufficient defense mechanisms. Previous work on defendi ng email-based attacks focuses primarily on filtering techniques implemented at the outgoing mail servers [16]. Unfortunately, newer instances of mass-mailing worms come equipped with their own SMTP engines [3, 4], and therefore are capable of bypassing filters or detection mechanisms deployed at the outgoing mail server. An alternative approach would be examining incoming mails. Unfortunately, the prevalence of email messages with malicious conten t makes it difficult and costly to perform comprehensive filtering before the emails reach end users. The worm is then unlikely to be detec ted in the end user's mailbox because most users update their anti-virus software/signatures infrequently, if at all. As a resul t, we will continue to see outbreaks of email-based attacks, the frequency and sophistication of which will only increase with time.

In this paper we present an in-depth study of two mass-mailing worms, SoBig and MyDoom, from real traffic traces. The purpose of this paper is not to propose any particular defense mechanism. Rather, we focus on studying fundamental behavior and characteristics of the worms, which may lead to new insights to automatically detect, suppress and stop their propagation.

The remainder of this paper is organized as follows. Section 2 describes related work. Section 3 gives a brief background on SoBig and MyDoom. Section 4 describes the trace data set. Section 5 describes the analysis techniques and presents the the results of our analysis. Section 6 discusses the implications of our findings and we conclude in Section 7.

2. RELATED WORK

Several documented studies have investigated malicious code and the ways in which it propagates. Moore et al. [11] analyzed the propagation of the scanning worm Slammer and its effect on the Internet as a whole. Our work instead focuses on the effects of mass-mailing worms o n a single subnet.

Researchers have also studied the use of epidemiological models to model the spread of a virus or worm within a network [7, 10, 13, 14]. Staniford et al. [12] presented a study of different types of worms and how they can cause damage on the Internet. Zou et al. [18] modeled the propagation of

WORM'04, October 29, 2004, Washington, DC, USA.

the Code Red worm. These studies focus on propagation simulation and our work differs by using traffic analysis to identify anomalies and behavior of email worms.

In the area of worm defense, Williamson [15] proposed the idea of host-based rate limiting by restricting the number of new outgoing connections. He further applied this mechanism to email worms by rate limiting emails to distinct recipients [16]. Wong et al. [17] studied the effects of various rate limiting deployment strategies. Ganger et al. [8] proposed a scheme that analyzes and limits network traffic based on abnormal DNS lookup patterns. Chen et al. [6] devised a detection mechanism based on the premise that a worm-infected host will have more failed connections. Gupta et al. [9] used a heuristic based approach for detection of email worms. Our approach is different from previous work since we focus on the studying the fundamental behavior of email worms.

3. BACKGROUND

Our work is focused on email worms. In this section, we give a brief description of email worms and the general characteristics that set them apart from other instances of malicious code.

An email worm is a program that propagates by sending copies of itself to recipients via electronic mail. Once a recipient opens the email attachment, the malicious program executes on the victim's machine and further propagates itself. Typically, the worm program chooses its targets by harvesting email addresses from the victim's address book, web cache, and hard disk. The worm then sends mails containing its own code to targets in an attempt to repeat the infecti on cycle.

Email worms are different from scanning worms such as Code Red and Slammer [5, 18]. The latter typically exploits a vulnerability (e.g., buffer overflow) in the network services running on a target machine, and they must perform IP scanning to find vulnerable targets. Email worms infect hosts by tricking users into inadvertently executing malicious code. Unlike a scanning worm, which needs to aggressively search for new victims to compromise, an email worm has much higher hit rate because it obtains targets from victim machines.

Unlike traditional email viruses, mass-mailing worms do not limit their targets strictly to those in a victim's address book. W orms like SoBig and MyDoom utilize aggressive spreading techniques such as harvesting legitimate domain names from victim machines (e.g., by scanning web caches and hard disks) then attempting to construct probable addresses. These worms also use social engineering techniques (e.g., disguise as a system error message) to lure users to open t he malicious attachment. Some mass-mailing worms even spread using avenues other than mail. For in stance, both SoBig and MyDoom replicated over file sharing clients (e.g., Kazza).

Both SoBig and MyDoom exhibit unique and atypical traffic patterns. For instance, clients infected with SoBig and MyDoom used their own SMTP engines in propagation attempts. Under normal circumstances, connections to outside SMTP servers are highly unusual for most enterprise clients.[1]

[1]With the exception of clients using multiple mail client configurations to access addresses at different domains.

4. TRACE DATA

Our study of Sobig and MyDoom was conducted on real network traffic traces. These traces are collected from the edge router of CMU's Electrical and Computer Engineering (ECE) Department.

The ECE network consists a total of 1,128 hosts total, of which 4 are mail servers, 3 are DNS servers and around 1,000 are normal end host clients. We recorded, in an anonymized form, all IP and common second layer headers of network traffic (e.g., TCP or UDP) entering or exiting the ECE network. The header information for TCP and UDP contain the source and destination addresses and port numbers. We also recorded DNS traffic payloads which were anonymized accordingly.

For this study, we examined traces from two specific time periods. The first is from August 5, 2003 to September 5, 2003. This period contains the outbreak of the "Sobig" worm [3], unleashed on August 16th, 2003. The second is from January 15, 2004 to February 5, 2004, which corresponds with the outbreak of "MyDoom" [4], unleashed on January 24th, 2004. Residual effects of both worms lingered on for months, but the effects of the infection were most prominent during the first two weeks.

5. ANALYSIS

In this section we describe our study of the SoBig and MyDoom traces. We begin by examining traffic patterns before and during the worm outbreaks. More specifically, we investigated change in the rate of outgoing TCP, SMTP, and DNS-related traffic. We found that traffic anomalies were easily detectable from monitoring client machine traffic patterns.

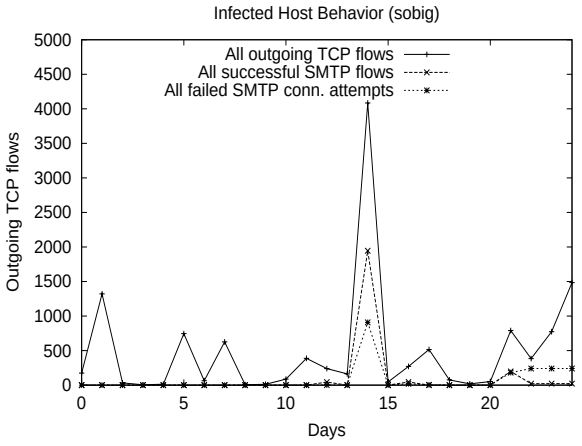
For our analysis, it is advantageous to identify the infected hosts and contrast their behavior with that of normal hosts. We devised a simple heuristic to identify infected hosts.[2] The heuristic involves contrasting outgoing SMTP connections from a host before and during the outbreak. An uninfected host should make no or very few outgoing SMTP connections (other than to its designated mail server). If a host changes its behavior by creating a large number of SMTP connections during the outbreak, it is marked as a candidate for being infected. We further refined the heuristic by considering the sizes of the message payload—those with payload similar to that of the worms reported by Symantec are suspects[3, 4]. Note that this is only a heuristic and we cannot identify infected hosts with 100% certainity. Using this heuristic, we identified 5 infected hosts during the SoBig outbreak and 6 infected hosts during the MyDoom outbreak. In the analysis that follows, those infected hosts refer to the sets identified by this heuristic.

In the following sections we analyze data before, during, and after the worm outbreaks. We study the number of outgoing TCP traffic flows, the number of distinct IPs contacted by TCP flows, and the relationship between DNS traffic and TCP flow data. In each section we present results by comparing average traffic of normal clients to the infected clients. We also present the effects of the mass-mailing worms on the outgoing traffic patterns of mail servers.

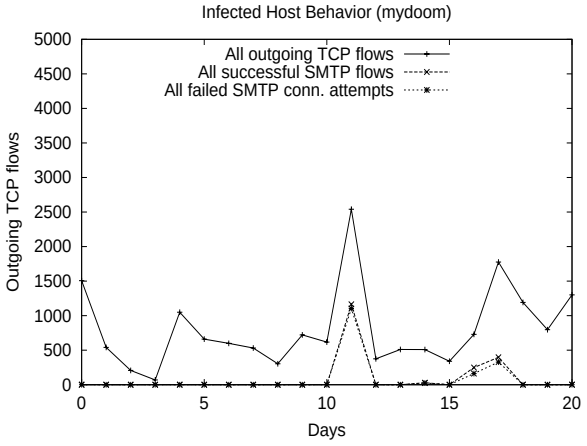
[2]Since we anonymize source and destination IP addresses, it is not straightforward to pinpoint infected hosts.

5.1 TCP Traffic patterns

In this section we study the number of outgoing TCP traffic flows observed on our network.



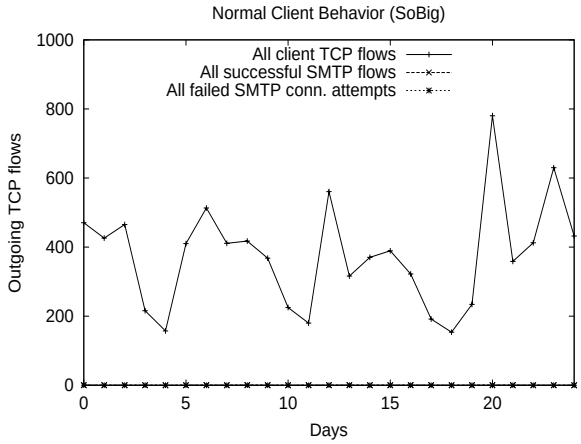
(a) SoBig



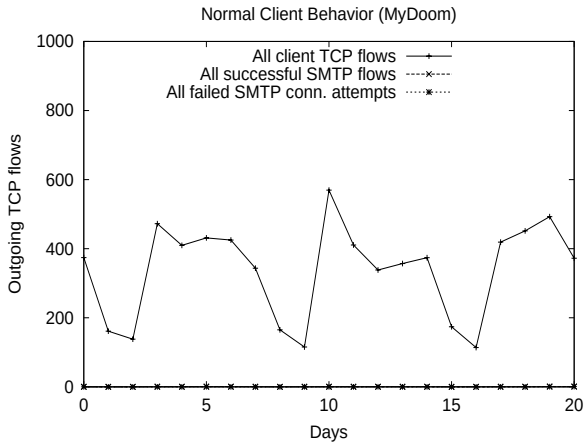
(b) MyDoom

Figure 1: Average outgoing TCP flows for infected hosts

TCP behavior of infected hosts Figure 1 shows the average outgoing TCP flows (both successes and failures) for infected clients. As shown, before the worm outbreak (Day 0 - 12), the hosts made very few outgoing SMTP connections. Both the SoBig and MyDoom outbreaks (around day 12 - 15 in the graph) caused the volume of SMTP flows to jump to almost 50% of the total TCP flows. For SoBig, we also see a particularly pronounced increase in the volume of outgoing TCP flows outside of the increase in SMTP. This additional traffic can be attributed to the worm periodic ally contacting worm servers to download new malicious code. We also see a large number of SMTP failures for both worms. In SoBig's case SMTP failure s amounted to about 25% of total TCP flow traffic and approxiamately 40% in MyDoom's case. We speculate that MyDoom's higher failure rate is due to its attempts to "guess" the names of mail servers at particular domains, which may have resulted in resolving IP addresses protected by firewalls or otherwise unequipped to receive traffic on SMTP. In contrast, Figure 2 shows the average



(a) Sobig



(b) MyDoom

Figure 2: Average outgoing TCP flows for normal hosts

outgoing TCP flows of normal hosts during the same period. These graphs include all the outgoing TCP flows of the client and the subset of SMTP flows. The graphs indicate a cyclic weekly pattern of traffic, a higher amount of traffic during weekdays and dipping to a lower amount on the weekend. Furthermore, the uninfected client very rarely attempted to make connections to outside SMTP servers, which resulted in a minimal number of SMTP flows.

Behavior of mail servers Figure 3 shows all the outgoing SMTP flows of the mail servers. Intuitively, since

both worms carried their own SMTP engine, the outgoing traffic of mail servers should not increase. Yet, there is a pronounced spike during the SoBig outbreak whereas mail server traffic during MyDoom's outbreak remains relatively flat. We believe this is due to the 'spoofed mail' phenomenon, as described below.

A newly infected host scans the victim's hard drive for email addresses of prospective targets. The "From:" address is occasionally spoofed or hard coded in by the worm author (e.g. "admin@fake.com"). The "To:" address can be partially guessed (e.g. "John.Doe@victim.com") or it may be taken directly from a victim's address book. If the recipient does not exist on the destination mail server, that server will send an email back to the mail server listed in the "From:" address. We believe that the reason for the increase in the TCP flows is due to this phenomenon – both SoBig and MyDoom aggressively guess destination email addresses, which resulted in an increase in bounced email message flows. The data in Figure 3 suggests that SoBig was either more aggressive in guessing email addresses or that it more often provided return addresses in legitimate domains.[3]

Discussion From the TCP flows statistics, we can see that SoBig employed a more aggressive propagation strategy than MyDoom. Although there were less clients infected by SoBig in our network, the total volume of flows generated by its infected clients was 50% greater than MyDoom's. Our data suggests that traffic analysis on the mail server would be ineffective against MyDoom, since mail server traffic during the MyDoom outbreak does not appear to be distinctively unusual. However, for Sobig, it is possible that one can detect its presence due to the large volume increase in the SMTP flows.

5.2 Distinct IPs

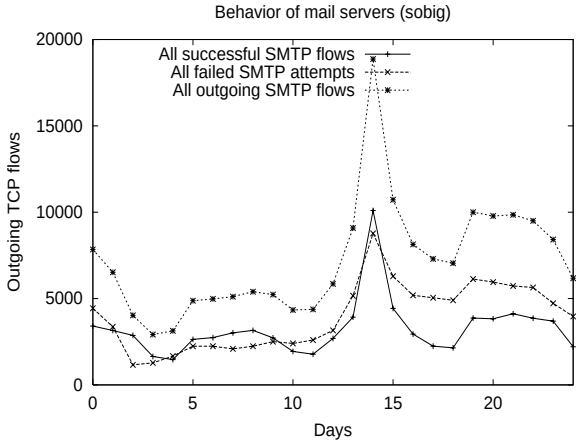
In this section we study the number of distinct destination IPs of outgoing TCP flows and relate this statistics to the TCP flow traffic analyzed in the previous section.

Behavior of infected hosts Figure 4 shows the average number of distinct destination IPs for the outgoing TCP flows of the infected hosts. Comparing the average number of successful TCP flows and the average number of distinct IPs from Figure 1 in the previous section, we can see that at SoBig's peak an infected client on average had 2000 successful SMTP flows corresponding to only 200 distinct IPs. At MyDoom's peak infected clients averaged 900 successful SMTP flows to 115 distinct IPs. These data suggest that multiple infection attempts were directed to the same target (perhaps different email addresses within the same domain). On average, the infected hosts sent about 10 infected emails to each server.

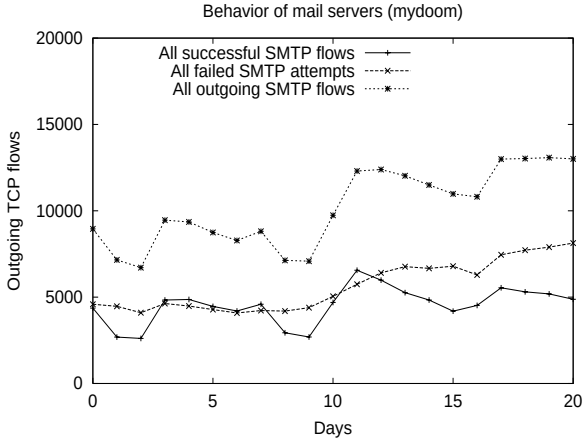
Plots in Figure 5 show that for a normal host, very few of the distinct IPs contacted involved SMTP. A typical uninfected client initiated about 400 successful TCP flows per day. Since these flows were to approximately 20 unique IP addresses, around 20 flows per IP address were observed.

Behavior of mail servers For the mail server, the ratio of successful outgoing SMTP flows to distinct IPs contacted

[3]MyDoom so metimes left the sender field blank or filled it with random characters.



(a) Sobig

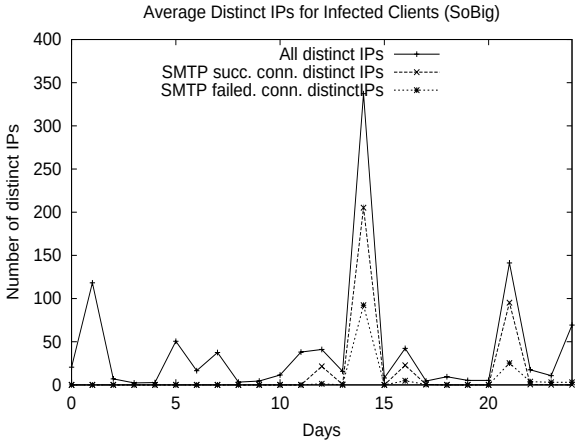


(b) MyDoom

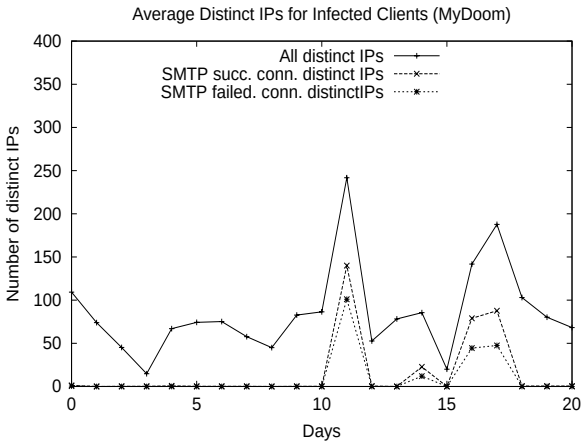
Figure 3: Number of outgoing SMTP flows for mail servers

is smaller for mail servers compared to normal clients. At the peak of the SoBig outbreak, there were about 9500 successful outgoing SMTP flows corresponding to 1600 distinct IPs. The increase in distinct IPs contacted can be attributed to contact with additional mail servers resulting from the "spoof ed mail" effect, discussed in Section 5.1.

Discussion Contrary to our original belief that distinct IPs contacted should remain relatively stable during outbreaks of email worms (since domain names are scanned from web caches and files from victims machines), hosts contacted an noticeably large number of distinct mail servers. This is interesting because it means that rate limiting schemes such as Williamson's [15] that work by limiting the rate of distinct IPs contacted can still be effective. However, it should be noted that an email worm may send a large number of emails to the same mail server, hence straightforward

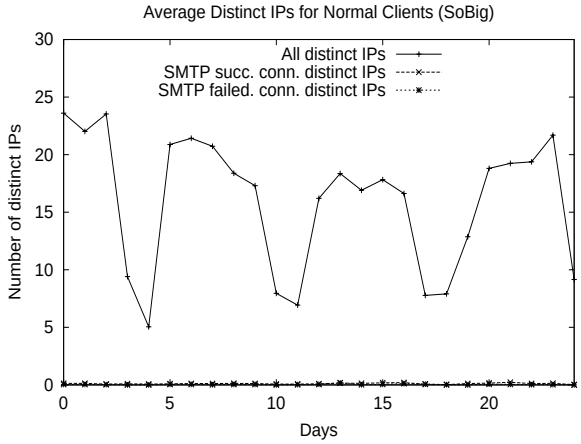


(a) Sobig

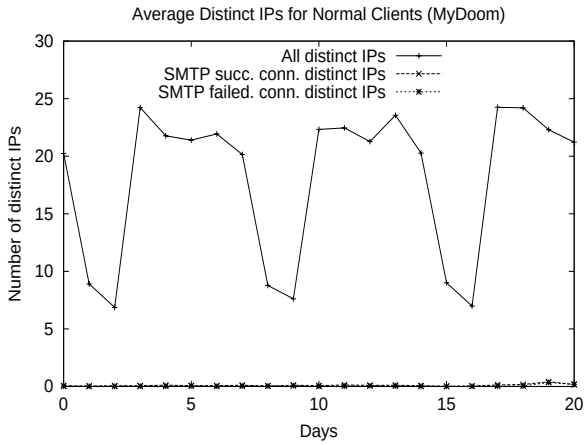


(b) MyDoom

Figure 4: Distinct outgoing IPs on average for infected hosts



(a) Sobig



(b) MyDoom

Figure 5: Distinct outgoing IPs on average for normal hosts

rate limiting on distinct IPs would not be as effective as against random scanning worms.

5.3 DNS and related traffic

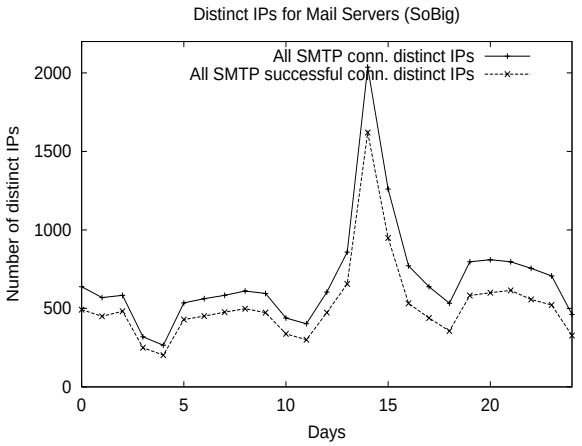
In this section we study DNS and related traffic. A worm attempts to infect targets by using a list of mail addresses. In order for the infection to reach its target, the worm's SMTP engine (on the infected host or on a designated mail server) needs to contact DNS servers to obtain the IP address of the destination mail server. Thus, we expect that a disproportionately large number of DNS queries will be made during the outbreak of an email worm. We con jecture that the patterns of DNS traffic during an outbreak may yield interesting insights into the behavior of an email worm. The detailed analysis can be found in the following subsections.

The ECE network we study has three DNS servers. To simplify the analysis, we use the concept of a "virtual DNS cache", an abstract entity intended to model the combined behavior of the name servers as comprising a single cache.[4] There are a number of events in the cache that are of interest to this study:

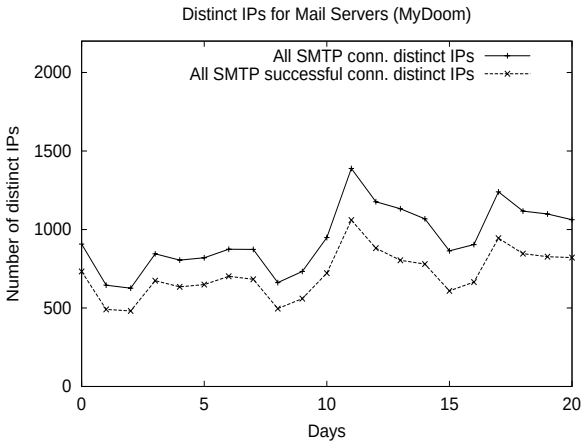
1. New cache entry. This event occurs when a previously unseen or expired IP address is returned in a DNS query. It is added to the cache along with its associated Time to Live (TTL).

2. Refreshed cache entry. This event occurs when an unexpired IP address is returned in a DNS query with a TTL that may or may not increase its lifetime in the cache

[4]In other words, all DNS traffic was combined into a single pool, regardless of the specific DNS server responsible for it.

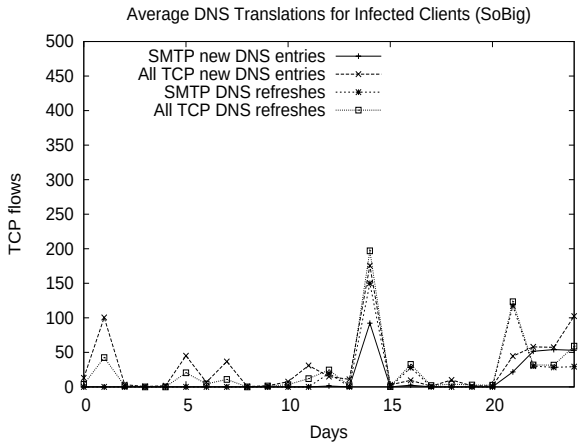


(a) Sobig

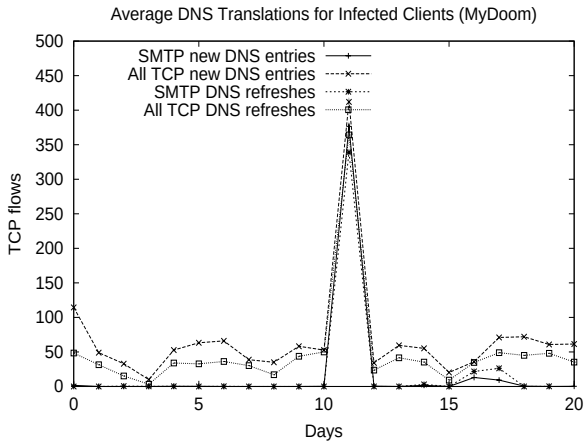


(b) MyDoom

Figure 6: Distinct outgoing IPs for mail servers



(a) Sobig



(b) MyDoom

Figure 7: Average TCP-related and SMTP-related DNS translations for infected hosts

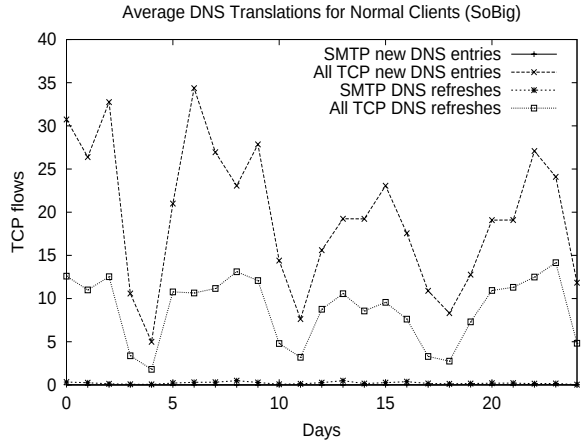
3. Cache entry expiration. This event occurs whenever the TTL of an IP address expires.

Modeling the cache in such a manner allows us to emulate the type of statistics and events that are already available to a nameserver internally. For the purposes of our discussion, if a host first contacts a freshly translated IP on a SMTP destination port, we refer to the translation as "SMTP-related", and if the host first contacts the IP using TCP, the translation is "TCP-related".

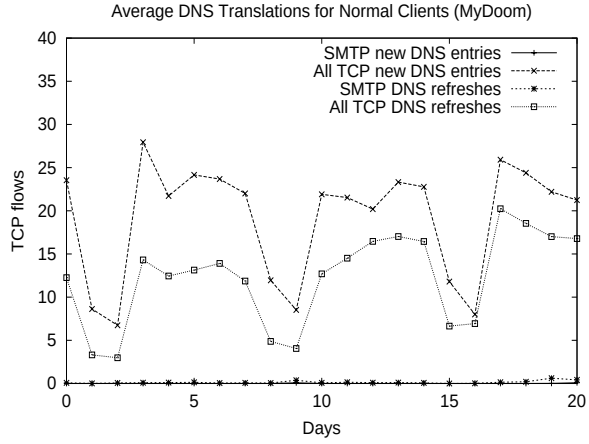
Behavior of infected hosts Figure 7 shows the TCP-related DNS translations corresponding to new or refresh cache events for infected hosts. Figure 8 shows the same data for normal hosts. Before the outbreak, for both SoBig and MyDoom, SMTP-related translations were virtually non-existent. This behavior is consistent with the behavior of a normal client, as seen in Figure 8.

During the outbreak of SoBig, SMTP-related translations increased dramatically to almost 50% of TCP-related translations for new entries and 80% for refreshes. We also observed that refreshes dominated new entries, which is the opposite of what was observed before the outbreak.

The effects of MyDoom are similar to those of SoBig. During the outbreak, the number of TCP-related translations resulting in new and refreshed entries amount to equal volumes of traffic, whilst before the outbreak DNS lookups resulting in new entries are significantly larger in volume than DNS refreshes. This is presumably due to the DNS lookups from different infected clients for the same mail server which resulted in DNS refreshes. Figure 8 shows the translation patterns for normal clients, which are consistent with the behavior of the infected clients prior to the worm outbreak.

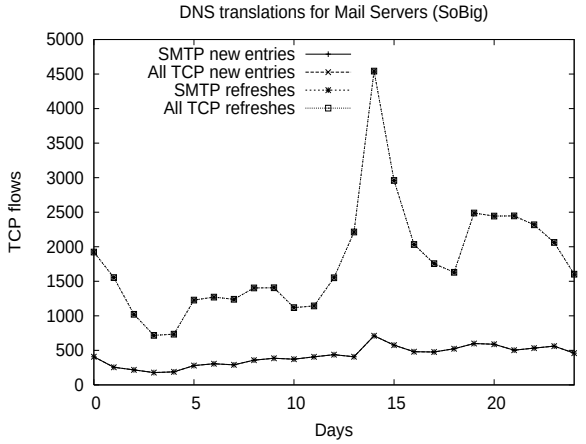


(a) Sobig

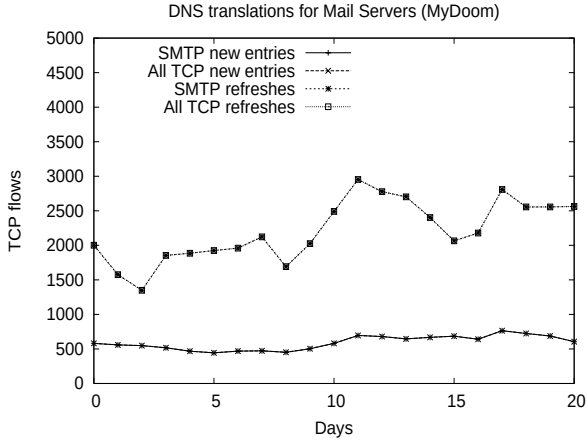


(b) MyDoom

Figure 8: Average TCP-related and SMTP-related DNS translations for normal clients



(a) Sobig



(b) MyDoom

Figure 9: TCP-related and SMTP-related translations for mail servers

Behavior of Mail Servers Figure 9 shows TCP-related and SMTP-related translations corresponding to new and refreshed DNS entries. The traffic patterns on the mail servers are quite different from what is observed on normal clients. On the mail servers, refreshes dominate the translations while the number of new entries remain consistently low. This is because there is virtually no other traffic on the mail server besides SMTP, whose destinations are mail servers with typically large TTLs. In contrast, as we will show in Section 5.4, client flows are dominantly HTTP.

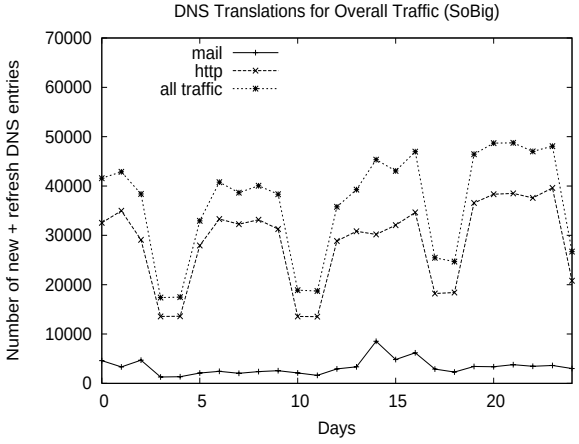
As previously mentioned, our original belief is that there should be no increase in the outgoing TCP flows on the mail servers. Yet for the Sobig traces, there is a pronounced traffic spike occurring around day 13-15 in the graph, indicating an abnormally high volume of DNS translations. This corresponds to results seen in previous sections, where we speculate that the traffic is due to the mail server sending out an unusually high amount of bounced mails to addresses spoofed by the email worm, which resulted in the increase of DNS refreshes.

Discussion Our study of the infected hosts shows that associating DNS translations with TCP flow data reveals a spike in DNS related traffic, and it hints at the "second-order" effects of the worm, such as additional DNS traffic generated and additional data loaded into the local name-servers' caches. We also observed that during times of infection client translations uncharacteri stically resulted in more refreshes than new entries. This phenomenon could provide the basis for an interesting detection or de fense strategy, such as rate-limiting DNS responses that only contain refreshed entries to clients.

What was not accounted for in our study was local DNS traffic, which may look similar to the the figures in Sec-

tion 5.1 if little address caching is performed on the end hosts. Also unaccounted for are intermediate translations that result from "walking the DNS hierarchy" to arrive at the DNS server delegated for a particular host name. If these translations are included, we could see a sizable increase in DNS traffic attributed to worm activity.

5.4 Overall Traffic



(a) Effect of overall traffic (Sobig)

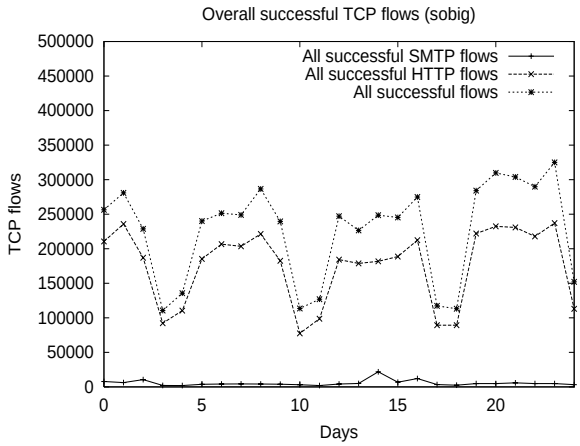
Figure 10: DNS effects on overall traffic

Given the increase in both TCP flows and DNS translations on the infected clients and mail servers, one would expect that similar trends exist in the aggregate traffic of the network. However, we found that HTTP traffic significantly dominates TCP network traffic and DNS translations. Figure 10 shows aggregate SMTP-related, HTTP-related, and TCP-related DNS translations in our network for the period of SoBig. Figure 11 shows the overall successful outgoing TCP flows in our network for the same period.[5] From inspecting new and refreshed DNS entries, we found that DNS events related to mail are rather insignificant in comparison to other traffic. As seen in Figure 10 there is a slight spike in SMTP-related traffic, but this comprises a very small portion of overall DNS traffic.

Moreover, we can see that HTTP makes up around 90% of outgoing TCP flows. During Sobig's infection peak, the number of successful flows attributed to SMTP was about 20,000, while HTTP amounted to about 200,000 flows. For MyDoom (graph not present) the number of SMTP flows averaged about 10,000 and HTTP amounted to on average about 300,000 flows.

By looking at overall traffic flows it is hard to meaningfully discern network behavior and the misbehavior of infected hosts. Selective traffic filtering must be employed to successfully detect the presence of a mail worm, or the noise of other traffic (e.g., HTTP) will make a worm extremely difficult to detect. In addition, to understand which hosts are

[5]In both graphs, the data from the t ime of MyDoom is nearly identical to that of the time of SoBig. We only present one set due to space constraints.



(a) Effect of overall traffic (Sobig)

Figure 11: TCP flows of overall traffic

making abnormal contributions to overall traffic, one must separate them into different groups (e.g., clients, servers, p2p clients, ...).

6. DISCUSSION

In the previous section we studied the effects of mass mailing worms from a variety of perspectives. We found that mass-mailing worms' activity on end host machines is apparent whether one monitors the number of outgoing TCP flows, the number of distinct IP addresses contacted by those TCP flows, or the DNS lookup pattern required to resolve the address. An advantage of monitoring DNS patterns to detect a worm is that one could implement a mail-worm watchdog at the DNS server, rather than analyzing all outgoing packets at the router. Containment possibilities at the DNS server are also intriguing. In particular, the DNS server may be able to slow down the rate that mass-mailing worms propagate by delaying answers to translation requests. Another possibility is that a DNS server could attempt to "quarantine" suspicious emails by replacing the address of the requested mail server with that of a mail "holding bay". These suspicious mails could then be inspected by a more heavy weight filtering mechanism.

The analysis in Section 5.1 shows that mail worms that create messages with a legitimate return address will place additional load on mail servers by causing noticeable "bounced" mail messages to the domains. In our analysis, we discovered that SoBig caused more "backscatter" mail traffic than MyDoom, presumably because SoBig was more aggressive in attempting to guess addresses or because MyDoom often did not provide a properly formed return address. Any assessment of the load placed on a network by a worm should factor in this "spoofed mail" effect, as well as the increase of DNS traffic attributable to it (see Figure 9). In order to limit the load on the mail server, one could employ a strategy limiting the number of bounced mail messages (for example, those in response to mail with attachments).

Additionally, we found that most DNS traffic generated

by the worms was to refresh unexpired entries in the DNS cache rather than to input new entries. This is intuitive, because addresses associated with mail servers often have longer TTLs. This suggests that the additional DNS traffic generated by worms may be superfluous, and that loads on DNS servers may be minimized by honoring the TTLs of their current entries rather than refreshing them prematurely.

Finally, the total number of outgoing flows of overall traffic shows that DNS traffic attributed to TCP is dominated by HTTP, not SMTP. Therefore, in order to detect the presence of a mass mailing worm, it is necessary to apply additional filtering techniques to overall traffic. By examining traffic on the SMTP port or looking at sets of particular hosts or servers, we were able to distinguish the worms' presence in an academic network setting. Future work will focus on a more in depth investigation of the DNS traffic, with an eye towards separating IPs corresponding to MX (mail exchange) records from others. If this proves fruitful, it may be helpful to add logic to the DNS server that flags internal hosts requesting an abnormal number of mail server address translations as part of a wider defense or detection mechanism.

Our study suggests existing defenses specifically designed for monitoring outgoing mail on SMTP servers do not work for newer types of mass mailing worms, because almost all of these worms have their own SMTP engine. Defenses should instead be deployed where most network traffic can be seen – at the edge router or at individual hosts. In addition, selective filtering should be employed, or noise from other TCP traffic (e.g. HTTP) will obscure the traffic patterns of email worms.

Williamson's [15] and Ganger's [8] schemes will be less effective against email worms than random scanning worms. Ganger's detection scheme is based on using a secure network interface to detect and rate-limit connections to IP addresses that were not introduced to the host by a DNS translation. Since email worms require DNS lookups (for MX records) to send mail to their targets, this scheme would be ineffective.

Although the data did suggest a spike in the distinct IP addresses contacted by the infected hosts, Williamson's scheme [15] that restricts connections to different IP addresses could be easily circumvented: the worms we observed sent an average of 10 infected emails to each IP address it contacted, and this ratio could be further increased by sending more infection attempts to a single friendly (or compromised) external SMTP server. Williamson also proposed a different scheme that would rate limit mail to distinct email addresses [16]. Other work has shown such rate limiting schemes are ineffective unless implemented on almost all host machines [17]. Another defense mechanism is for a router to filter out all outgoing SMTP traffic from non-mail servers. This solution would be possible and quite effective in an enterprise network setting where clients are mostly homogeneous. However, it may prove to be too restrictive for academic computing environments.

Future worms may attempt to hide under the radar by sending emails at a slower rate, or by intelligently delegating targeted addresses across other infected intranet hosts. We believe that an effective traffic monitoring scheme would use different filtering mechanisms and correlate their results. For instance, traffic monitoring could be done on both per-client and aggregate basis, and traffic could be monitored using a variety of metrics, including the number of total flows and unique destination IPs per each destination port. The more specifically we can examine traffic, and the more vantage points we employ to examine it, the easier it will be to mitigate false positives and detect abnormal traffic.

7. CONCLUSIONS

From our network trace data analysis, it is clear that SoBig and MyDoom both induced noticeable abnormalities in the traffic of the infected hosts. In addition, during mass-mailing worm outbreaks, the normal traffic patterns of outgoing mail on the mail servers were also disr upted, despite the fact that infected hosts used their own SMTP engines to forward mail. We conjecture that these disruptions are due to the backscatter mail traffic. In addition, anti-virus software deployed on the mail server is not only ineffect ive against mail worms, it also generates excess traffic by sending infection notifications to spoofed addresses.

We examined our trace data in a number of ways, all of which exhibited abnormalities at the time of the worm outbreaks. These included exploring the aggregate number of outgoing TCP flows, the number of outgoing TCP flows to distinct destination ad dresses, and the relationship between TCP flows and DNS traffic. We found that each "viewpoint" provided us with unique insight into characterizing and contrasting each worm's behavior and im pact on our local network. One interesting observation gathered from our data is that SoBig was generally more virulent than MyD oom. SoBig caused more outgoing infection attempts per infected host and contacted more distinct destination addresses. Recent work in rate limiting schemes such as traffic throttling [15] and secure NICs [8] show promise for mitigating widespread worm attacks. However, these techniques have not been tested using sufficiently realistic network data. In addition, we have outlined some problems in using these techniques to defend specifically against mail worms. Future work will focus on alternative defense approaches that are more tolerant of a diversified network environment.

8. REFERENCES

[1] Network Associates and 2000-05. Vbs/loveletter@mm. World Wide Web, http://vil.nai.com/vil/content/v_98617.htm, 2000.

[2] Network Associates and 2001-07. W32/sircam@mm. World Wide Web, http://vil.nai.com/vil/content/v_99141.htm, 2001.

[3] Network Associates and 2003-08. W32/sobig.f@mm. World Wide Web, http://vil.nai.com/vil/content/v_100561.htm, 2003.

[4] Network Associates and 2004-01. W32/mydoom@mm. World Wide Web, http://vil.nai.com/vil/content/v_100983.htm, 2004.

[5] CERT. CERT Advisory CA-2003-04 MS-SQL Server Worm. World Wide Web, http://www.cert.org/advisories/CA-2003-04.html.

[6] Shigang Chen and Yong Tang. Slowing down internet worms. In *Proceedings of 24th International Conference on Distributed Computing Systems*, Tokyo, Japan, March 2004.

[7] Zesheng Chen, Lixin Gao, and Kevin Kwiat. Modeling the spread of active worms. In *Proceedings of IEEE INFOCOM 2003*, San Francisco, CA, April 2003.

[8] Gregory R Ganger, Gregg Economou, and Stanley M Bielski. Self-securing network interfaces: What, why and how, Carnegie Mellon University Technical Report CMU-CS-02-144, August 2002.

[9] A. Gupta and R. Sekar. An approach for detecting self-propagating email using anomaly detection. September 2003.

[10] Jeffrey O Kephart and Steve R White. Directed-graph epidemiological models of computer viruses. In *Proceedings of the 1991 IEEE Computer Society Symposium on Research in Security and Privacy*, pages 343–359, May 1991.

[11] David Moore, Vern Paxson, Stefan Savage, Colleen Shannon, Stuart Staniford, and Nicholas Weaver. Inside the slammer worm. In *IEEE Security and Privacy journal, 2003*, 2003.

[12] Stuart Staniford, Vern Paxson, and Nicholas Weaver. How to 0wn the internet in your spare time. In *Proceedings of the* 11^{th} *USENIX Security Symposium*, August 2002.

[13] Yang Wang, Deepayan Chakrabarti, Chenxi Wang, and Christos Faloutsos. Epidemic spreading in real networks: An eigenvalue viewpoint. In *Proceedings of the 22nd International Symposium on Reliable Distributed Systems*, 2003.

[14] Yang Wang and Chenxi Wang. Modeling the effects of timing parameters on virus propagation. In *Proceedings of the 2003 ACM workshop on Rapid Malcode*, pages 61–66. ACM Press, 2003.

[15] Matthew M Williamson. Throttling viruses: Restricting propagation to defeat malicious mobile code. In *Proceedings of the 18th Annual Computer Security Applications Conference*, Las Vegas, Nevada, December 2002.

[16] Matthew M Williamson. Design, implementation and test of an email virus throttle. In *Proceedings of the 19th Annual Computer Security Applications Conference*, Las Vegas, Nevada, December 2003.

[17] Cynthia Wong, Chenxi Wang, Dawn Song, Stanley M Bielski, and Gregory R Ganger. Dynamic quarantine of internet worms. In *Proceedings of DSN 2004*, Florence, Italy, June 2004.

[18] Cliff Changchun Zou, Weibo Gong, and Don Towsley. Code red worm propagation modeling and analysis. In *Proceedings of the* 9^{th} *ACM Conference on Computer and Communication Security*, November 2002.