A Crawler-based Study of Spyware on the Web

Alexander Moshchuk, Tanya Bragin, Steven D. Gribble, and Henry M. Levy
Department of Computer Science & Engineering
University of Washington
{anm, tbragin, gribble, levy}@cs.washington.edu

Abstract

Malicious spyware poses a significant threat to desktop security and integrity. This paper examines that threat from an Internet perspective. Using a crawler, we performed a large-scale, longitudinal study of the Web, sampling both executables and conventional Web pages for malicious objects. Our results show the extent of spyware content. For example, in a May 2005 crawl of 18 million URLs, we found spyware in 13.4% of the 21,200 executables we identified. At the same time, we found scripted "drive-by download" attacks in 5.9% of the Web pages we processed. Our analysis quantifies the density of spyware, the types of of threats, and the most dangerous Web zones in which spyware is likely to be encountered. We also show the frequency with which specific spyware programs were found in the content we crawled. Finally, we measured changes in the density of spyware over time; e.g., our October 2005 crawl saw a substantial reduction in the presence of drive-by download attacks, compared with those we detected in May.

1 Introduction

In the span of just a few years, spyware has become the Internet's most "popular" download. A recent scan performed by AOL/NCSA of 329 customers' computers found that 80% were infected with spyware programs [2]. More shocking, each infected computer contained an average of 93 spyware components. The consequences of spyware infections can be severe, including inundating the victim with pop-up ads, stealing the victim's financial information or passwords, or rendering the victim's computer useless.

Despite the severity of the problem, little is known about the nature or extent of spyware in the Internet. Previous studies have taken a desktop- or user-centric view. For example, in an earlier study, we measured the presence of a small set of spyware programs at the University of Washington by sniffing the university's Internet connection for communication between client desktops and spyware servers [16]. The AOL scan mentioned above has provided simple summary statistics by directly examining desktop infections [2], while a recent set of papers have considered user knowledge of spyware and its behavior [6, 29].

In this paper we change perspective, examining the nature of the spyware threat not on the desktop but from an Internet point of view. To do this, we conduct a large-scale outward-looking study by crawling the Web, downloading content from a large number of sites, and then analyzing it to determine whether it is malicious. In this way, we can answer several important questions. For example:

- How much spyware is on the Internet?
- Where is that spyware located (e.g., game sites, children's sites, adult sites, etc.)?
- How likely is a user to encounter spyware through random browsing?
- What kinds of threats does that spyware pose?
- What fraction of executables on the Internet are infected with spyware?
- What fraction of Web pages infect victims through scripted, drive-by download attacks?
- How is the spyware threat changing over time?

Overall, our goal is to provide a quantitative analysis of the extent of spyware-laden content in the Web.

Spyware typically installs itself surreptitiously through one of two methods. First, a user might choose to download software to which piggy-backed spyware code has been attached. Piggy-backed spyware is particularly common with file-sharing software; the Kazaa system [10] alone has been the source of hundreds of millions of spyware installations. Second, a user might visit a Web page that invisibly performs a "drive-by download" attack, exploiting a vulnerability in the user's browser to install software without the user's consent.

We have designed and implemented a scalable, cluster-based analysis platform that uses virtual machines (VMs) to sandbox and analyze potentially malicious content. By

installing and running executable files within a clean VM image, we can use commercial anti-spyware tools to determine whether a specific executable file found by our Web crawler contains piggy-backed spyware. By visiting a Web page with an unmodified browser inside a clean VM, we can use heuristic "triggers," such as the installation of a new library or the creation of a new process, to determine whether the Web page mounts a drive-by download attack. We describe our methodology in detail, including the heuristics that make the approach practical and scalable.

We carried out our study by running multiple crawler-based experiments, first in May of 2005, and then again in October 2005. This allowed us to evaluate changes in the spyware environment over that five-month period. Our results show that spyware is a significant threat in the Internet. For example, we found piggy-backed executable spyware in 4.4% of the domains we crawled in October 2005. Moreover, more than 1 in 20 of the executable files we examined contained spyware. We also saw significant changes, e.g., we found scripted drive-by download attacks in 3.4% of the domains we examined in May, but in only 1.6% of domains in October. While much of the spyware we identified is benign adware, we also found a large number of Trojan downloaders and other more malicious threats.

The rest of this paper is organized as follows. Section 2 describes previous work that places our current study in context. Section 3 presents both methodology and results for our examination of Internet executables. The methodology and results for our drive-by download study are detailed in Section 4. Finally, Section 5 summarizes our results.

2 Related Work

In our previous work, we used passive network monitoring to measure the extent to which four specific adware programs had spread through computers on the University of Washington campus [16]. In this paper, we study the spyware problem from a different perspective. Specifically, we measure the extent to which: (1) executable Web content contains spyware and (2) Web pages contain embedded drive-by download attacks. Both studies confirm the existence of a significant spyware problem.

The AOL/NCSA online safety study conducted a poll of 329 households and also examined their computers for the presence of spyware [2]. Over half of the respondents believed their machines were spyware-free; in reality, 80% of computers scanned were infected with spyware programs. The AOL/NCSA study did not attempt to identify how these computers became infected.

Both our work and the Strider HoneyMonkey project [24] are inspired by honeypot techniques [15]. Strider HoneyMonkey uses a method that is similar to ours to construct a tool to find Web sites that exploit browser vulnerabilities. While there are some differences in our methods, our study differs from theirs in several other significant ways. First, we examined executable file content for piggybacked spyware programs in addition to examining Web pages for drive-by download attacks. Second, we provide a rich analysis of the spyware that we encountered, including which areas of the Web are most infected, and the fraction of spyware that contains malicious functions, such as modem dialing or Trojan downloading. Third, we examined how spyware on the Web has changed over time. Fourth, we analyzed the susceptibility of the Firefox browser to drive-by downloads, in addition to the Internet Explorer browser.

Overall, at the time of the publication of their technical report [24], HoneyMonkey had been more focused more on the tool. In contrast, our study has focused on the analysis of our results to understand the spyware threat from several different points of view.

The Gatekeeper project [26] monitors extensibility points in the Windows operating system and its applications to detect spyware programs. This approach complements signature-based detection schemes, and bears some similarity to "trigger" mechanisms we use in our drive-by download study. It can detect arbitrary spyware programs that use monitored extension points and observe these programs as they are being installed.

A recent edition of the *Communications of the ACM* contained over a dozen articles on the spyware problem [3, 4, 6, 7, 8, 12, 14, 17, 18, 20, 23, 27, 29]. These articles discuss issues such as the public perception of spyware, security threats caused by spyware, and frameworks for assessing and categorizing spyware programs.

Many projects have examined the detection, measurement, and prevention of malware, such as worms and viruses [11, 13, 19]. Some of their techniques may ultimately be applicable to the detection and prevention of spyware. A notable example is the semantics-aware malware detection project [5], which uses an instruction-level semantic analysis of programs to match their behavior against high-level signature templates. Another example is Ghostbuster [25], which detects OS rootkit installations by comparing the file-system of an OS using a program running on the OS and scanning from an OS booted from a CD.

At least two commercial anti-spyware companies have implemented automated crawlers to seek out new spyware threats on the Web. Though we have not found detailed technical descriptions of their architecture, Webroot's Phileas [28] system appears to use a cluster of computers to scan Web content for known threats and patterns that are suggestive of new browser exploits. Sunbelt Software has announced that is is buildling a Web crawler to automate the identification of new spyware outbreaks [22].

3 Spyware-Infected Executables in the Web

This section describes our study of spyware in executable files on the Web. We first examine the tools and infrastructure that we constructed to carry out the study. We then discuss high-level results and answer the following questions:

- Which spyware programs are most prevalent and which sites distribute the most spyware?
- Are spyware executables uniformly spread on the Web or concentrated in specific areas?
- What spyware functions are more common (e.g., adware vs. keylogging)?
- How is the density of spyware-laden executables changing over time on the Web?

3.1 Study tools and infrastructure

This study required an automated solution to three problems: (1) determining whether a Web object contains executable software, (2) downloading, installing, and executing that software within a virtual machine, and (3) analyzing whether the installation and execution of the software caused a spyware infection. In addition, we required a high-performance infrastructure to solve these problems so that we could analyze a large number of executables from a variety of sources in a reasonable amount of time.

3.1.1 Finding executables

We assumed that a Web object was an executable if either: (1) the *Content-type* HTTP header provided by the Web server when downloading the object was associated with an executable (e.g., `application/octet-stream`), or (2) its URL contained an extension known to be associated with executables and installers (e.g., `.exe`, `.cab`, or `.msi`). Once we downloaded a Web object, we also looked for well-known signatures at the beginning of the file to help us identify its type. If we could not identify the file's type, we assumed it was not an executable and did not analyze it. While our approach may have missed some executables, it rarely produced false positives. Accordingly, our study may underestimate the number of executable files on the Web, but it is unlikely to overestimate it.

Some executable files on the Web are not immediately obvious to a Web crawler. Two instances of this are executables embedded in archives (such as ZIP files), and executables whose URLs are hidden in JavaScript. To handle the first case, we downloaded and extracted archive files, looking for filenames with extensions associated with executables. To handle the second case, our Web crawler scanned JavaScript content looking for URLs and added them to the list of pages to crawl. Note that JavaScript programs can dynamically construct URLs when interpreted. Since our Web crawler does not execute JavaScript code, we missed any such executables.

3.1.2 Running executables within a VM

Once we downloaded an executable, we installed and ran it in a clean VM. This was challenging; while it is simple to run a "naked" executable file, software is often distributed using an installer framework, such as *Windows Installer*. Unfortunately, installers typically interact with users, requiring them to perform manual tasks such as agreeing to a EULA, filling in demographic information, or pressing buttons to begin the installation process.

To automate the execution of installer frameworks, we developed a tool that uses heuristics to simulate common user interactions. For example, some of our heuristics identify and click on permission-granting buttons such as *next, OK, run, install,* or *I agree.* Other heuristics identify and select appropriate radio buttons or check-boxes by looking for labels commonly associated with EULA agreements. The tool also looks for type-in boxes that prompt the user for information, such as their name or email address, and fills them in with dummy information. While our tool cannot handle all installation scenarios perfectly, we verified that it successfully navigates all popular installer frameworks and we have rarely seen it fail.

Our study focuses on Windows executables. Accordingly, for each executable that we analyze, we first created a VM that contained a clean Windows XP guest operating system image. To do this, we used the "snapshot take" and "snapshot revert" functions provided in VMware Workstation 5.0 [21] running on a Linux host operating system. For each node within our cluster, we maintained a pool of VMs. When we wished to analyze an executable, we allocated a VM from this pool, rolled-back the VM to a clean checkpoint, injected the executable or installer image into the VM, and used our tool to install and execute the program.

3.1.3 Analyzing the installed executable

Once an executable was installed and run in a VM, our final challenge was to determine whether that executable had infected the VM with spyware. To do this, we ran the Lavasoft AdAware anti-spyware tool [1] in the VM, using scripts to launch the tool and collect the infection analysis from its emitted logs. The log information we collected was rich enough for us to identify which spyware programs were installed. Using online databases of spyware, we also manually classified which functions those spyware programs contained, such as keystroke logging, adware, Trojan backdoors, or browser hijacking.

Of course, AdAware can detect only those spyware programs that have signatures within its detection database. Accordingly, our analysis misses spyware programs that AdAware does not find. Also note that we only collected information about spyware software that is installed. Though many anti-spyware tools such as AdAware also identify malicious cookies or registry entries as spyware threats, we excluded these, focusing only on spyware software.

To speed up the AdAware sweep, we pruned the Windows/XP image installed in our VM so that it contained as few files and ran as few components as possible. We also disabled the host firewall and automatic updates, so as not to interfere with our analysis.

3.1.4 Performance

Our executable analysis infrastructure was hosted on a 10-node cluster consisting of dual-processor, 2.8 GHz Pentium 4 machines, each with 4GB of RAM and single 80 GB 7200 RPM disks. On average, it took 92 seconds to create a clean VM, install an executable, run it, and perform an AdAware sweep. Of this time, we spent around 1-2 seconds creating the VM, 55 seconds installing and running the executable in the VM, and 35 seconds performing the AdAware sweep. By parallelizing our analysis to run one VM per processor in our cluster, we could analyze 18,782 executables per day. In practice, we found that the bottleneck of our system was crawling the Web for executables, rather than analyzing the executables once found.

3.2 Web crawling

We used the Heritrix public domain Web crawler [9] to gather a crawl of over 2,500 Internet Web sites. To understand how spyware had penetrated different regions of the Web, we crawled sites from eight different categories: adult entertainment sites, celebrity-oriented sites, games-oriented sites, kids' sites, music sites, online news sites, pirate/warez sites, and screensaver or "wallpaper" sites. In addition, we crawled c|net's download.com shareware site, which provides a large number of downloadable executables.

Within each category, we selected sites using both the Google directory and the results of category-specific Google keyword searches. For each selected Web site, we used the top-level page as a seed and then crawled to a depth of three links away, restricting ourselves to pages hosted on the same domain. We chose a depth of three in order to balance thorough coverage of individual sites with breadth across many sites. With a depth of three, we crawled an average of 6,577 pages per site.

Many Web sites host downloadable executables on separate Web servers or outsource their distribution to third parties. Because we wanted to attribute these executables to the owner site, we allowed our crawler to fetch executable content *linked to* from the seed site but hosted on a different Web server.

For comparison with our chosen categories, we also crawled a number of "randomly selected" Web sites. For this study, we used a random walk of the link structure of the Web. We first scraped keywords from Metaspy, which lists in-progress searches occurring on the Metacrawler search engine. Next, we performed Google searches using those keywords and selected several results at random rankings from each search. Starting from the results' Web pages, we followed hyperlinks at random to a distance of 8 links away and considered the resulting sites to be "random."

3.3 Examining the changing spyware environment

To evaluate the way in which the spyware threat is changing over time, we used our methodology to conduct executable program crawls on two occasions: in May 2005, and then again 5 months later, in October 2005. In each case, we began from scratch, generating lists of crawling seeds from the Google directory and the results of category-specific Google keyword searches, as described above. Therefore, each crawl represents a partial view of the Web, informed in part by Google's page rankings at that moment in time. This allows us follow time-based trends of executable spyware in the Internet.

Note that when analyzing the May crawl, we used the most recent version of the AdAware anti-spyware tool that was available at that time (signature database version SE1R42, released on April 28, 2005). For the October crawl, we used an updated version of AdAware (signature database version SE1R70, released on October 12, 2005) that contained more recent spyware signatures.

3.4 Limitations

Our study has several limitations due to our measurement method and the nature of the Web itself. First, we did not crawl the entire Web – our results are based on a directed *sampling* of Web pages and executables. While our sampling explores what we believe are interesting parts of the Web, such as Google-selected domains and URLs in various categories, we cannot prove that this is representative of what people actually encounter while browsing the Web or those categories as a whole. Second, our goal is to study the presence or density of spyware on the Web; we cannot extrapolate any relationship between that density and the presence of threats on the desktop, since the latter is based on the behavior of real users. As previously noted, we and others have measured the desktop threat separately. Finally, because we ultimately determine the existence of spyware

crawl date	URLs crawled	domains crawled	executables found	domains w/ executables	infected executables	infected domains	unique spyware programs
May 2005	18,237,103	2,773	21,200	529 (19.1%)	2,834 (13.4%)	106 (3.8%)	82
October 2005	21,855,363	2,532	23,694	497 (19.6%)	1,294 (5.5%)	111 (4.4%)	89

Table 1: **Executable file results.** The number of pages crawled, domains crawled, executables analyzed, and infected executables found during our study of executable files on the Web.

by running a scan of an anti-spyware tool (AdAware), we are limited by what AdAware is able to detect as a threat.

Despite these limitations, we believe that our study is a significant step forward in understanding and quantifying the spyware threat from an Internet point of view.

3.5 High-level results

Table 1 shows the high-level results from our executable file study. We crawled over 18 million URLs in May 2005 and nearly 22 million URLs in October 2005. In both crawls, we found executable files in approximately 19% of the crawled Web sites and spyware-infected executables in about 4% of the sites. While the absolute number of spyware-infected executables dropped substantially between the crawls, this is due primarily to a single site whose number of infected executables declined from 1,776 in May to 503 in October. Except for that site, the amount of spyware we found did not change appreciably over the five-month period between our two crawls. Overall, we found that as of October 2005, approximately 1 in 20 of the executable files we crawled contained spyware, an indication of the extent of the spyware problem.

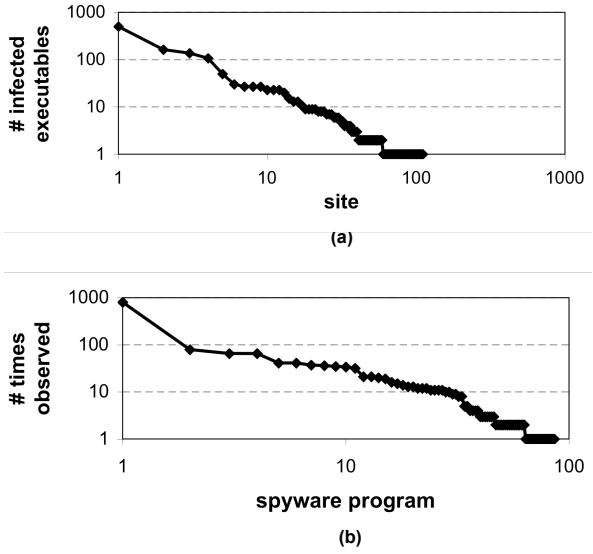

Figure 1: **Spyware prevalence (October 2005).** (a) The number of spyware-infected executables found in crawled sites. The x-axis is sorted by the number of executables found on that site. (b) The number of times a given spyware program was found; the x-axis shows the number of times the program was found. Both graphs' axes are drawn on a log-scale.

3.6 Who are the main culprits?

Table 1 shows that spyware appears on a small, but non-negligible fraction of the Internet Web sites we crawled (3.8% in May 2005, 4.4% in October 2005). However, some sites are much more egregious than others in presenting infected content. Figure 1a plots, on a log/log scale, the number of infected executables we found on each site that we crawled during October 2005; the results from May are similar. While some sites offer a large number of infected executables, most just offer a handful.

Our crawl found a total of 2,834 infected executables in May and 1,294 in October. However, those infected executables contained only 82 (May) and 89 (October) different spyware programs; the total number of distinct spyware threats we encountered is relatively small. Figure 1b plots the prevalence of each spyware program in the infected executables. The 10 that appear most frequently are shown in Table 2. (We removed data for the outlier site scenicreflections.com from the spyware program lists in Tables 2a and 2b; this single site contained 1,776 instances of "TurboDownload" and 1,354 of "WhenU" in May 2005).

Most spyware programs are rare; during our May 2005 crawl, only 15 spyware programs were found that were present in more than twenty infected executables. However, the most prevalent programs appeared very frequently: we detected 364 executables that contained WhenU in May, and 340 such executables in October. This data suggests that signature-based anti-spyware techniques should be effective, as relatively few spyware variants are commonly encountered when Web browsing.

Looking at the change in these lists over time, six of the top-ten offending sites in the May 2005 crawl also appeared in October's top-ten list. The remaining four sites

site	# infected executables
scenicreflections.com	1,776
screensaver.com	191
celebrity-wallpaper.com	136
screensavershot.com	118
download.com	116
gamehouse.com	111
galttech.com	38
appzplanet.com	37
megspace.com	36
download-game.com	30

spyware program	times observed
WhenU	364
180Solutions	236
EzuLa	214
Marketscore	143
BroadCastPC	67
Claria	44
VX2	41
Favoriteman	36
Ebates MoneyMaker	31
NavExcel	24

(a) executable file study, May 2005 crawl

site	# infected executables
scenicreflections.com	503
gamehouse.com	164
screensavershot.com	137
screensaver.com	107
hidownload.com	50
games.aol.com	30
appzplanet.com	27
dailymp3.com	27
free-games.to	27
galttech.com	23

spyware program	times observed
WhenU	340
Marketscore	47
Claria	41
BroadCastPC	37
Aurora	36
FOne	35
Zango	34
EzuLa	33
Web3000	32
180Solutions	25

(b) executable file study, October 2005 crawl

Table 2: **Top 10 spyware programs and sites.** The top 10 spyware-laden sites, and the top 10 spyware programs found, in the (a) May and (b) October 2005 crawl. Programs and sites common across the two crawls' top-ten lists are italicized. Note that the top 10 spyware program lists exclude data from the outlier site scenicreflections.com, which contained 1,776 instances of "TurboDownload" and 1,354 of "WhenU" in the May crawl.

were still functioning and serving spyware, but three were not encountered during our October crawl, and the fourth ($c|net$) was serving far less spyware. Similarly, six of the top-ten offending programs from May also appeared in October's list. No offenders disappeared: all ten from the May crawl were encountered at least once in the October crawl. Interestingly, one of the "newcomers" in October's top-ten list, Aurora, was first released in April 2005, and has gained significant "popularity" since then.

Overall, while the absolute rank of the top offenders changed over time, we found that the list of the most egregious sites and programs was fairly stable.

3.7 Are some Web categories more dangerous than others?

Anecdotal evidence suggests that some zones of the Web are more dangerous than others. For example, one might expect to encounter more spyware on freeware and shareware sites than on commercial news-reporting sites. Table 3 shows the frequency with which we encountered spyware-infected programs in ten different categories of Web sites, as defined in Section 3.2, during our October 2005 crawl. While all categories except "news" contained at least one spyware executable, our results confirm that some Web site categories do appear more spyware-laden than others.

Our data shows that the most high-risk category is "games." Approximately 60% of these sites contain executable content, which presumably consists of free games available for download. Though only a small fraction of these executables contain spyware (5.6%), one in five game sites include spyware programs. Another high-risk category is "celebrity," for which over one in seven executables are infected with spyware.

We have not included a similar detailed breakdown for the May 2005 crawl, since we saw few qualitative changes between the May and October crawls. Two cells in Table 3 did experience a substantial change, however: the fraction of infected celebrity executables dropped from 73.5% in May to 16.3% in October, and the number of infected screensaver executables dropped from 2,256 in May to 789 in October. In both cases, the change is attributable to a single Web site that offered an anomalously high number of infected executables in May, but far fewer in October.

The $c|net$ Web portal has a large number of free and shareware programs available for download. In May, we examined 2,370 executables at $c|net$ and found spyware in 110 of them (4.6%). In October, we re-crawled the site and examined 1,944 executables, but we found only 6 infected with spyware (0.3%). Sometime in between our two crawl dates, $c|net$ had implemented a policy of scanning file submissions to ensure they are adware and spyware free. While a few programs seem to have slipped past their scans, they have substantially reduced how much spyware is available through their site.

3.8 What kinds of spyware do we find?

Adware may be an annoyance and can degrade performance, but it typically poses no significant security threat. In contrast, keylogging spyware is dangerous, since it puts at risk a victim's passwords, account numbers, and other sensitive information. To understand the danger posed by the spyware typically found on the Web, we categorized the spyware-infected executables according to the kind of spyware they installed.

In addition to adware and keyloggers, we categorized *Trojan downloaders*, which download and install additional software chosen by the attacker; *browser hijackers*, which modify browser functions, such as search engine tools and the user's default home page, or redirect URLs to differ-

	adult	celebrity	games	kids	music	news	pirate	wallpaper	c|net	random
URLs crawled	3,465,024	3,131,497	872,686	733,648	3,421,796	458,079	3,042,390	678,506	193,118	5,858,619
domains crawled	157	144	125	183	220	20	311	125	1	1,356
executables found	158	153	4,872	112	4,218	19	3,422	6,860	1,944	2,000
domains with executables	26 (16.6%)	28 (19.4%)	76 (60.8%)	24 (13.1%)	72 (33.2%)	7 (35.0%)	111 (36.0%)	51 (40.8%)	1 (100%)	102 (7.5%)
infected executables	18 (11.4%)	25 (16.3%)	272 (5.6%)	3 (2.7%)	149 (3.5%)	0 (0%)	74 (2.2%)	789 (11.5%)	6 (0.3%)	6 (0.3%)
infected domains	12 (7.5%)	11 (7.6%)	25 (20.0%)	3 (1.6%)	24 (11.4%)	0 (0%)	21 (7.1%)	12 (9.6%)	1 (100%)	5 (0.4%)
unique spyware programs	12	10	32	5	55	0	43	34	5	2

Table 3: **Executable infections across Web categories.** This table shows the percentage of executables and domains infected with spyware across different Web categories, based on our October 2005 crawl.

spyware function	May 2005	October 2005
keylogging	0.04%	0.15%
dialer	0.14%	0.9%
Trojan downloader	9.1%	13%
browser hijacker	60%	85%
adware	91%	75%

Table 4: **Spyware functions.** The fraction of spyware-infected executables found that contain various spyware functions. Note that since a given spyware-infected executable may have more than one function, the columns in these tables do not sum to 100%.

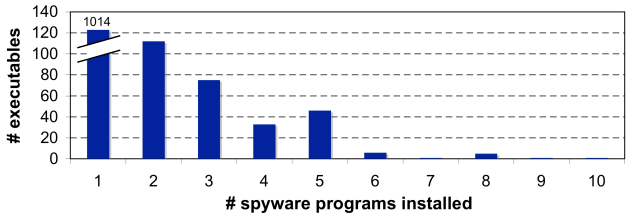

Figure 2: **Number of programs installed.** The number of spyware programs installed per executable, for the October 2005 crawl data. Most infected executables install only one or two spyware programs, but some install many.

ent sites; and *dialers*, which use a victim's modem to call expensive toll numbers. A given executable might contain more than one function, either because it contains several spyware programs or because it contains one spyware program that has multiple functions.

Table 4 shows our results. In both the May and October crawls, most infected executables consisted of relatively benign adware or browser hijackers. However, a significant fraction of infected executables contained Trojan downloaders, which do pose a serious threat.

Comparing the results from the two crawls, we see little absolute change. Though the fraction of dangerous keyloggers and dialers went up by nearly an order of magnitude, they both still remain rare relative to the more benign adware and browser hijacker functions.

Spyware-infected executables can cause a single spyware program to be installed, or they can cause multiple infections. To explore this, we counted the number of spyware programs installed by each infected executable. Note that some spyware programs download additional spyware at a later date. In this study, we waited 10 seconds after installing the original executable before running our AdAware scan. As a result, our study primarily observed programs installed directly, but we captured some of these cascaded downloads as well. Figure 2 shows our results for the October crawl data; the May crawl data showed qualitatively similar results. The majority of infected executables installed only one spyware program, as shown in the figure. Some executables cause a larger number of infections; for example, sixty of the spyware-infected executables installed five or more spyware programs.

3.9 Can signature-based tools keep up?

One common defense against spyware is a signature-based anti-spyware tool. By comparing a database of spyware signatures to files and processes running on a client computer or network traffic arriving at the computer, signature-based tools can detect when the computer is infected with known spyware programs. Because new spyware programs and variants emerge over time, the vendors of these tools are forced to compete in a perpetual

crawl date	URLs crawled	domains crawled	executables found	domains w/ executables	infected executables	infected domains	unique spyware programs
May 2005	14,441,999	6,272	5,534	823 (13.1%)	653 (11.8%)	279 (4.4%)	125

Table 6: **Executable file results (blacklisted sites).** The number of pages crawled, domains crawled, executables analyzed, and infected executables found during our study of executable files on blacklisted Web sites.

October 2005 executable pool	May 2005 AdAware signature DB	October 2005 AdAware signature DB
# infected executables found	880	1,294
# unique spyware programs found	80	89

(a)

spyware program	# times found using October AdAware DB	new program or new variant?
WhenU	398	new variant
WebHancer	6	new variant
Spyware.WebDir	2	new program
Lycos Sidesearch	2	new variant
AdBlaster	2	new variant
WinFixer	1	new program
BroadcastPC	1	new variant
Whazit	1	new variant
iDonate.BHO	1	new program

(b)

Table 5: **New spyware variants.** (a) Infections detected within the October 2005 executable pool, using either the May or October AdAware signature databases. (b) Spyware programs detected by AdAware's October signature database that eluded its May signature database.

"arms race" with spyware purveyors to keep their signature databases fresh.

To determine how rapidly new spyware programs and variants surfaced over the period of our crawl, we re-analyzed all of the spyware-infected executables we found in October, but using the *older* AdAware signature database that was available in May. If new spyware threats were released between May and October, this older version of AdAware might not have signatures that match them.

Table 5 shows our results. Of the 1,294 infected executables that the newer October AdAware signatures detected, only 880 were detected by the older May signatures: 414 infected executables contained new programs or variants that required updated signatures. However, only nine spyware programs were responsible, as shown in Table 5b. Of these nine, three were newly introduced programs, and six were new variants of older spyware programs.

While it is important to keep an anti-spyware signature database up-to-date, the rate at which we found new spyware threats piggybacked on Web executables was slightly less than two per month. In contrast, the total rate at which new spyware software is written must be significantly faster, as confirmed by the fact that leading commercial anti-spyware tools now add anywhere from a few dozen to several thousand new signatures to their signature databases per month.[1] While many new spyware programs are created per month, our results suggest that in practice only a small number of them become wide-spread enough to be encountered by a typical Web user.

Because it may be difficult to predict in advance which new spyware threats will actually be encountered by users, signature-based anti-spyware tools must either use automated tools to find all new threats and generate signatures for them, or use automated techniques to focus human attention on the emerging "important" threats if manual signature generation is necessary. Without such automated techniques, anti-spyware tool vendors might become overwhelmed by the effort of ensuring their signature databases have enough coverage to protect against the specific threats that end up mattering in practice.

3.10 How effective is blacklisting?

Another defense against spyware is to construct *blacklists* of URLs or domains that are suspected to contain spyware. Given a blacklist, a firewall or proxy can block clients from accessing listed sites. To evaluate the potential of blacklists to block spyware infections, we crawled domains listed in two third-party blacklists ("IE-SPYAD" and "methlabs," both readily available on the Web), and repeated our executable file analysis. We performed our blacklist crawl in May 2005.

[1] For example, Webroot uses automated Web crawling to find new spyware threats [28]. Fed by this, their "Spy Sweeper" tool currently contains over 100,000 signatures, and added 3,863 new signatures between October 11th and November 11th of 2005. Note that any given spyware program may have many signatures associated with it, so these numbers do not directly reflect how many distinct spyware programs are in their database. In comparison, AdAware has added or updated signatures for only 70 spyware programs per month, as they rely on manual detection.

spyware function	May 2005 blacklisted sites	May 2005 non-blacklisted sites
keylogging	0.1%	0.04%
dialer	7.2%	0.14%
Trojan downloader	30%	9.1%
browser hijacker	74%	60%
adware	72%	91%

Table 7: **Spyware functions (blacklisted vs. non-blacklisted sites).** The fraction of spyware-infected executables found that contain various spyware functions, for the May 2005 blacklist crawl and the previously presented May 2005 non-blacklist crawl. Note that since a given spyware-infected executable may have more than one function, the columns in these tables do not sum to 100%.

Table 6 shows the summary statistics from our blacklist crawl. Comparing these results to our May crawl of other Web categories (Table 1), we see that a similar fraction of blacklisted sites and URLs contain spyware-infected executables as non-blacklisted sites. The overall conclusion we can draw is that blacklists are ineffective in two ways: many blacklisted sites contain no spyware, and many non-blacklisted sites do contain spyware.

Table 2, presented earlier in the paper, showed that there were a large number of sites that each provide a handful of infected executables. This further suggests why blacklisting is only partially useful: while a blacklists can find and block egregious "heavy hitter" sites, it is difficult for them to find and block the much larger number of sites that contain small numbers of infected executables.

We were interested in understanding the kinds of spyware on which blacklists focus attention. Table 7 shows the fraction of blacklisted, spyware-infected executables found to contain various spyware functions. Compared to what we previously observed in the non-blacklisted crawl from May 2005, blacklisted spyware programs tend to contain a greater fraction of keyloggers, dialers, and Trojan downloaders. It appears as though blacklists tend to focus on spyware that contains more dangerous functions.

3.11 Summary

To study the threat of executable-based spyware on the Web, we examined approximately 20 million URLs in each of two crawls in 2005. The crawls explored 2,773 and 2,532 domains in May and October of 2005, respectively. Our results show that spyware piggybacked on executables is a significant threat. While we found only a small number of distinct executable spyware programs (89 in October 2005), we discovered instances of those programs in approximately 4% of the domains we visited. Overall, 1 in 20 of the executables we identified were infected with spyware in our October crawl – a surprisingly high fraction. We also showed that some Internet zones, such as game or celebrity sites, have a higher incidence of executable spyware than others. While some changes have occurred in the time between our crawls, at a high level they show a similar level of risk to Web users.

We examined potential issues facing two spyware defenses: signature-based anti-spyware tools and spyware blacklists. By comparing the threats found in our May and October crawls, we found only nine spyware programs in October that would not have been detected using a signature database from May, in spite of the fact that leading commercial anti-spyware tools add thousands of signatures per month. As well, we found that while blacklisted sites often contain more dangerous variants of spyware, many blacklisted sites do not contain any spyware at all, and many non-blacklisted site do contain spyware.

4 Drive-by Downloads

This section measures and analyzes drive-by download attacks on the Web. We begin by presenting the tools and infrastructure we constructed to carry out our study. Next, we present our results and answer specific questions about the source and frequency of drive-by induced infections.

4.1 Study tools and infrastructure

This part of our research examined *drive-by download attacks*, a common method for infecting victims with spyware. A drive-by download attack occurs when a victim visits a Web page that contains malicious content. An example is JavaScript embedded in HTML that is designed to exploit a vulnerability in the victim's Web browser. A successful drive-by download lets the attacker install and run arbitrary software on the victim's computer.

The primary challenge in detecting drive-by attacks is performing an automated analysis of Web content to determine whether it contains attack code. Fortunately, we found a simple solution: we assumed that a drive-by download attack would attempt to break out of the security sandbox implemented by the Web browser, e.g., modifying system files or registry entries. To recognize such an attack, we rendered the Web page using an unmodified browser and tried to detect when the sandbox had been violated.

4.1.1 Detecting browser sandbox violations

Our method for detecting browser sandbox violations is based on the notion of *event triggers*. A trigger fires when an event matching a trigger condition occurs in the guest OS or an application running on it. For example, if visiting

Trigger condition
Process creation: a new process is launched, excluding known browser helper processes.
File system activity: a file is created or modified, excluding those in "safe" folders such as the browser cache, browser cookies, system event logs, and paths associated with helper processes.
A suspicious process writes to a file: a process besides the browser and its known helper processes creates or modifies any file.
Registry activity: sensitive registry entries are modified, such as those that control programs that are launched on reboot, browser helper objects (BHOs) loaded when the browser runs, initialization scripts that are executed when certain programs are launched, etc.
Browser or OS crash: the browser or OS crash, or otherwise stop responding to events.

Table 8: **Trigger conditions.** If any of these trigger conditions occurs within the guest OS, we mark the associated URL as suspicious and run a spyware scan.

a Web page causes file-system activity to occur outside of a small set of prescribed folders associated with the browser (such as its local cache), a trigger will fire. Table 8 describes the trigger conditions that we defined and implemented for detecting drive-by downloads.

In our VM environment, we have precise control over the software that runs in our guest operating system. This lets us reduce the number of trigger conditions that are generated by the base software. For example, we configured the guest OS to disable all unnecessary background system services. This increases the likelihood that a trigger firing is the result of a drive-by download. However, some of our defined trigger conditions can occur naturally, e.g., from background services that we failed to disable or from system or browser crashes. In addition, not all drive-by downloads install spyware; some install benign software. Accordingly, when a trigger fires, we consider the page to be suspicious and perform an AdAware scan of the VM to detect installed spyware.

As a performance optimization, we attempted to reduce the number of new VMs that must be created. Therefore, as we crawled Web pages to scan for drive-by downloads, we continued loading pages within the same browser and VM instance until a trigger fired or we visited 100 Web pages. In either case, we then garbage collected the VM and created a new one from our clean, checkpointed VM image. In practice, we observed that triggers fire often enough that we have rarely reached the 100-page limit.

4.1.2 Dealing with complex web content

Some Web pages contain scripted content that could confound our analysis. One example is a "time bomb" used by some drive-by attacks; when a browser renders the page, JavaScript within it causes the browser to set a timer that will trigger some activity (such as a page load) at some defined point in the future. As another example, some Web pages contain JavaScript that is executed when the page closes. As a third example, some Web pages cause pop-up windows to open, which in turn may contain malicious code. If we mishandle these complexities, we could potentially attribute a trigger firing to the incorrect URL.

To deal with time bombs, we sped up the virtual time on the guest OS by a factor of fifteen. Thus, each second of wall-clock time that elapsed caused fifteen seconds to appear to elapse on the guest OS and its applications. All time-bombs that we have observed in drive-by downloads had a fuse-length of less than fifteen seconds, so we ensured that at least a second elapsed between fetching one URL and beginning the analysis of the next URL in the same VM.

Coping with page-close code is straightforward. Before concluding the analysis of a URL, we caused the browser to fetch a known, clean Web page, thereby triggering the page-close handlers of the previous Web page.

To correctly handle pop-up windows, we waited for all pop-up windows to finish loading and then closed them in order to trigger any page-close JavaScript handlers associated with the pop-ups. Some pop-up windows caused an endless sequence of additional pop-up windows to be opened; in this case, we iterated through our pop-up closing procedures 10 times before halting the analysis for that URL and resetting the VM image.

There are several additional complexities that we handle, including dealing with browser dialog boxes that prompt the user for input (e.g., when a Web page asks the user to accept a license agreement or to agree to change their default home page). We developed a set of automated heuristic solutions to this and other problems, but we omit a detailed description here due to space constraints.

4.1.3 Browser configuration

We analyzed two different browser configurations, both based on Microsoft's Internet Explorer (IE) version 6.0, running on Windows XP without either SP1 or SP2 installed. We deliberately chose to use unpatched versions of XP, since the majority of existing exploits attack vulnerabilities in such older system configurations. In addition,

		May 2005	October 2005 (recrawl May URLs)	October 2005 (new URLs)
URLs crawled		45,000	45,000	45,000
domains crawled		1,353	1,353	1,420
unique spyware programs found		48	26	36
say yes to prompts ("cfg_y")	infectious URLs	2,675 (5.9%)	1,548 (3.4%)	186 (0.4%)
	infectious domains	46 (3.4%)	27 (2.0%)	23 (1.6%)
say no to prompts ("cfg_n")	infectious URLs	690 (1.5%)	37 (0.1%)	92 (0.2%)
	infectious domains	16 (1.2%)	5 (0.4%)	9 (0.6%)

Table 9: **Drive-by download results.** The number of infectious pages and domains found by our drive-by download study. We report results for two browser configurations: IE configured to automatically say "yes" to security dialog boxes (*cfg_y*), and IE configured to say "no" (*cfg_n*). We also report on three traces: our May 2005 crawl, the same URLs from the May trace re-crawled in October 2005, and a new set of URLs gathered and crawled in October 2005.

most (but not all) newly found exploits affect both patched and unpatched systems.

For the first configuration (*cfg_y*), the browser behaves as though the user *grants* permission in all security-related IE dialog boxes. For example, when a Web page tries to download and run ActiveX controls, IE requests the user's approval for the action. Sometimes a Web page attempts to "push" an executable file to the user's computer, using either inline JavaScript or pop-up windows. In this case as well, IE asks for permission to install or run the executable.

For the second configuration (*cfg_n*), the browser behaves as though the user *refuses* permission in security-related dialog boxes. Spyware that installs itself despite the user's refusal typically exploits browser flaws, bypassing IE's security framework.

None of the URLs examined in this section link to executable content. Accordingly, any spyware infections found were the result of a drive-by download. Note that if an executable were installed in the *cfg_n* configuration, the user will not have the opportunity to refuse the installation, and, in most cases, no notification will occur. This is the most malicious form of a drive-by download: simply visiting a Web page will cause an executable to be installed and run on the victim's system.

4.1.4 Performance

Using the same cluster of machines described in Section 3.2, we found that analyzing a single Web page took on average 6.3 seconds, including restarting the browser and loading the page and its pop-ups. For those pages that fire a trigger, performing an AdAware scan took on average an additional 108 seconds. We observed that 5% of Web pages caused a trigger to fire, leading to an average latency of 11.7 seconds per page. We could run two VMs per CPU without loss of performance, and, accordingly, we analyzed approximately 14,769 pages per CPU per day.

4.2 High-level results

Our drive-by download study examined Web pages selected from the Website categories we described in Section 3.2, using the same methodology based on Google keyword searches. We performed three crawls overall. To begin, in May 2005 we crawled 45,000 URLs from 1,353 domains and analyzed their content. In October 2005, we performed two additional studies for comparison. First, we re-crawled these *same* URLs to understand how their content had changed. Second, we generated a *new* list of 45,000 URLs from 1,420 domains, seeded by re-executing the original Google keyword search. This second October crawl accounts for the constant change in domains, pages, and popularity on the Web.

Our high-level results are shown in Table 9.[2] In our May crawl, with IE configured to say "yes" to security prompts, 2,675 URLs in 46 domains caused spyware infections. With IE configured to say "no," we found 690 infectious URLs in 16 domains. That is, 1.5% of the URLs we crawled in May exploited IE security flaws to install spyware *without* prompting the user. While this may seem like a small percentage, consider that 1 in 67 Web pages that we examined contained malicious content targeting browser flaws.

Our examination of the same URLs in October saw a reduction in the number of drive-by attacks, with the drop significantly more pronounced for the *cfg_n* configuration. Many of the pages and domains that previously exploited browser vulnerabilities no longer did so: e.g., of the 690 *cfg_n* infectious URLs we found in May, only 37 were still infectious in October. Through manual examination, we found that some of the formerly infectious sites had been removed, some were still functioning but had substantially changed in content, and some had the same user-perceived content but no longer performed drive-by download attacks.

For comparison, we also crawled and examined the new set of 45,000 URLs that we generated in October. During this crawl, both browser configurations observed a significantly lower number of drive-by download attacks than we found in May. For example, in May, 5.9% of the crawled URLs performed *cfg_y* attacks and 1.2% of sites performed *cfg_n* attacks; in October, these percentages dropped to 0.4% and 0.6%, respectively.

[2]We repeated our drive-by study for blacklisted domains, similar what we presented in Section 3.10 for executable files. Blacklisted domains had drive-by results that are qualitatively similar to non-blacklisted domains, so we have chosen not to present them for the sake of brevity.

	category	adult		celebrity		games		kids		music		news		pirate		wallpaper		random	
		cfg_y	cfg_n	cfg_y	cfg_n	cfg_y	cfg_n	cfg_y	cfg_n	cfg_y	cfg_n	cfg_y	cfg_n	cfg_y	cfg_n	cfg_y	cfg_n	cfg_y	cfg_n
	URLs crawled	5,000	5,000	5,000	5,000	5,000	5,000	5,000	5,000	5,000	5,000	5,000	5,000	5,000	5,000	5,000	5,000	5,000	5,000
May 2005	domains crawled	67	67	99	99	105	105	125	125	145	145	21	21	78	78	83	83	716	716
May 2005	infectious URLs	35 0.7%	30 0.6%	182 3.6%	18 0.4%	493 9.9%	500 10%	0 0%	0 0%	439 8.8%	127 2.5%	0 0%	0 0%	1451 29%	2 0.04%	30 0.6%	7 0.1%	45 0.9%	6 0.1%
May 2005	infectious domains	3 4.5%	3 4.5%	10 10%	1 1.0%	4 3.8%	3 2.9%	0 0%	0 0%	10 6.9%	5 3.4%	0 0%	0 0%	11 14%	2 2.6%	3 3.6%	1 1.2%	7 1.0%	4 0.6%
October 2005 (new URLs)	domains crawled	101	101	103	103	93	93	164	164	116	116	17	17	185	185	92	92	621	621
October 2005 (new URLs)	infectious URLs	2 0.04%	0 0%	24 0.5%	16 0.3%	14 0.3%	5 0.1%	1 0.02%	0 0%	0 0%	0 0%	0 0%	0 0%	139 2.8%	68 1.4%	1 0.02%	0 0%	5 0.1%	3 0.1%
October 2005 (new URLs)	infectious domains	2 2.0%	0 0%	4 3.9%	1 1.0%	2 2.2%	1 1.1%	1 0.06%	0 0%	0 0%	0 0%	0 0%	0 0%	8 4.3%	4 2.2%	1 1.1%	0 0%	5 0.8%	3 0.5%

Table 10: **Drive-by downloads across Web categories.** Drive-by download attacks that cause spyware infections across Web categories, for both the May 2005 and October 2005 crawls. Note that the number of infectious "games" URLs in May 2005 is higher for *cfg_n* than *cfg_y*. In "games," many of the examined URLs non-deterministically send different content on subsequent downloads.

Overall, our summary statistics suggest that the density of drive-by download attacks on the Web has declined over the five-month period of our study.

4.3 From where do drive-by downloads originate?

To understand whether certain Web categories are more likely to perform drive-by download attacks than others, we examined URLs from eight different Web categories and calculated the fraction of URLs and domains that were infectious in each. Table 10 shows our results for the May crawl and the October crawl of new URLs.[3]

Focusing on the fraction of May domains that are infectious, we saw a blend of high-risk and no-risk categories. At the high end, 14% of "pirate" sites installed drive-by infections when the user responded "yes" to prompts. At the low end, we found no drive-by download attacks in either the "kids" or the "news" sites. The "random" category gives us an approximate view of the state of Web pages reachable through a Web crawl. Only 1% of "random" Web sites contained drive-by download attacks in May (0.8% in October), confirming that these attacks are focused on specific portions of the Web rather than being spread more uniformly throughout it.

It is common for infectious sites to attempt drive-by downloads on a large fraction of their Web pages. For each infectious domain we found using the *cfg_y* browser configuration, Figure 3 shows the fraction of pages in that domain that performed drive-by downloads, for the May crawl. This fraction ranges between 0.29% and 100%, however, approximately half of the infectious domains installed spyware through 50% or more of their URLs.

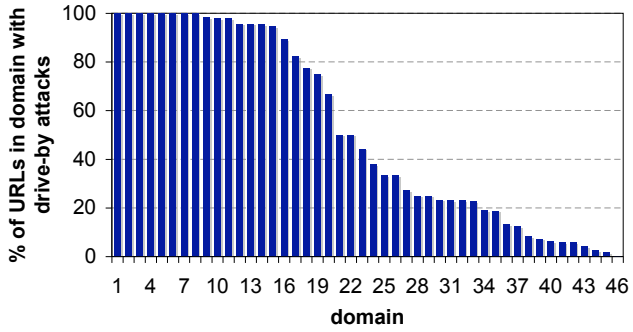

Figure 3: **Fraction of URLs in infectious domains that contain drive-by attacks (May 2005 crawl).** For each infectious domain, this graph shows the fraction of URLs within that domain that install spyware when the user responds "yes" to security prompts.

The "games" zone shows an interesting anomaly: in our May crawl, we observed slightly more *cfg_n* drive-by downloads occurring in this category than *cfg_y* drive-bys. In principle, the *cfg_n* drive-bys should be a subset of the *cfg_y* drive-bys, since exploits that affect a *cfg_n* browser should work equally well on a *cfg_y* browser. In practice, however, some URLs behave non-deterministically, serving different content each time the URL is visited. Their infectiousness depends on chance, and there were enough of these in the "games" category to visibly affect our overall results.

[3]The remainder of this paper presents data for these two crawls and ignores the October re-crawl of the May URLs. The re-crawl told us what changed in sites that were infectious in May; however, we believe that the new October crawl is more representative of the state of the Web in October, since it was generated with new seeds.

spyware program	times observed (cfg_y)
DyFuCA	1,176
PeopleOnPage	1,132
Adintelligence	1,114
istbar	1,008
Hijacker.TopConverting	905
BargainBuddy	889
SideFind	805
WindUpdates	628
IEHijacker.Hotoffers	556
ZyncosMark	421

spyware program	times observed (cfg_n)
IEHijacker.Hotoffers	250
Win32.TrojanDownloader.Agent.jq	43
CoolWebSearch	28
Security iGuard	22
AdRotator	12
Aurora	10
VX2	6
TIB Browser	5
Dialer.WSV	5
Win32.Trojan.Delprot.a	2

(a) drive-by download study, May 2005 crawl

spyware program	times observed (cfg_y)
Aurora	63
EffectiveBrandToolbar	61
WindUpdates	56
WinAD	55
ImlServer IEPlugin	33
Roings	27
DyFuCA	20
Win32.TD.TSUpdate	15
istbar	14
CoolWebSearch	9

spyware program	times observed (cfg_n)
Aurora	63
EffectiveBrandToolbar	53
VirtualBouncer	14
AdDestroyer	12
VX2	12
CasinoClient	9
CoolWebSearch	8
BookedSpace	4
BroadCastPC	4
Virtumonde	4

(b) drive-by download study, October 2005 crawl (new URLs)

Table 11: **Top 10 spyware programs.** Top 10 drive-by download infections for (a) the May 2005 crawl, and (b) the October 2005 crawl (new URLs). Programs that were found in the top-ten lists of both crawls are shown italicized.

We now turn our attention to the change in drive-by downloads over time. In nearly all categories, the density of drive-by download attacks we found in October was lower than in May. In many categories, the density had dropped substantially. For example, in the May crawl, a noticeable fraction of music sites and URLs performed drive-by attacks in both browser configurations, while in October, no music sites or URLs performed attacks.

It is difficult to attribute this decline to a specific cause, but there are several recent trends that we believe may be partially responsible. The adoption of anti-spyware tools has increased, shrinking the potential market reach of spyware programs. Automated patch installation tools such as Windows Update have made it simpler for typical users to administer security updates, reducing the number of Internet Explorer installations susceptible to known browser vulnerabilities. Finally, a recent set of civil lawsuits brought against spyware distributors or affiliates that use drive-by download mechanisms might have discouraged some potential attackers.

spyware function	May 2005	October 2005 (new URLs)
keylogging	0%	0%
dialer	0.02%	0%
Trojan downloader	51%	37%
browser hijacker	84%	60%
adware	75%	55%

(a) cfg_y

spyware function	May 2005	October 2005 (new URLs)
keylogging	0%	0%
dialer	2.4%	0%
Trojan downloader	22%	12%
browser hijacker	83%	78%
adware	73%	37%

(b) cfg_n

Table 12: **Spyware functions.** The fraction of drive-by downloads that cause various spyware functions to be installed. Note that a given drive-by download can install more than one function, so columns do not sum to 100%.

4.4 What kinds of infections do drive-by downloads cause?

Table 11 shows the spyware programs installed most frequently during drive-by download attacks for both the May and October crawls. There is little overlap between these lists and the executable file study (Figure 2) – in fact, only VX2, BroadCastPC, and Aurora appear in both. The set of spyware programs that tend to be piggybacked with executables are largely disjoint from those that tend to be installed via drive-by downloads.

Table 12 shows the spyware functions that are present in drive-by downloads. The results are similar to the executable study: keylogging and dialer functions are rarely observed, and the more benign adware and browser hijacking functions are much more common. Adware functions declined noticeably between May and October, dropping from 75% to 55% for *cfg_y*, and from 73% to 37% for *cfg_n*. While promising at first blush, in practice many Trojan downloaders retrieve and install adware over a long period of time. Our results may therefore underestimate the ultimate number of adware installs.

In Figure 4, we show histograms of the number of spyware programs installed by drive-by downloads using each of the *cfg_y* and *cfg_n* browser configurations, for both the May and October crawls. Most drive-by downloads installed only a few spyware programs, though a few caused

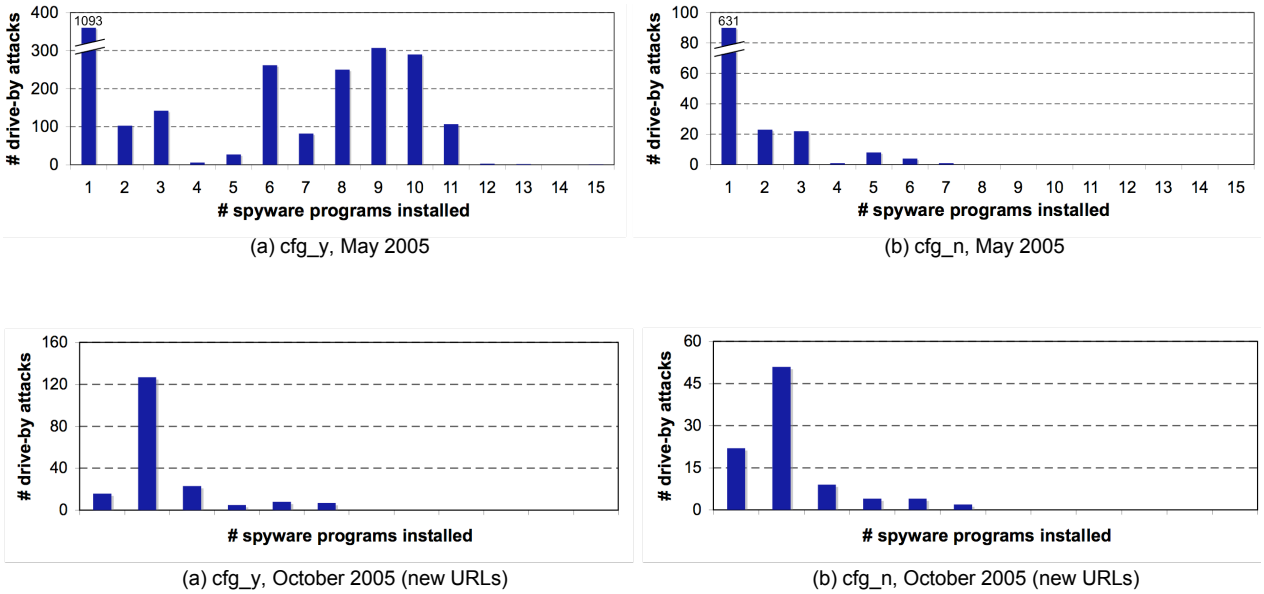

Figure 4: **Number of programs installed.** The number of spyware programs installed per drive-by download for *cfg_y* and *cfg_n*, and for the May 2005 and October 2005 (new URLs) crawls. Most drive-by downloads install only a few programs, but some install many.

many to be installed. Interestingly, in May, most drive-by downloads installed only a single program, while the trend was towards the installation of several programs in October.

The May data shows a cluster of *cfg_y* drive-by attacks that installed many spyware programs per attack. For example, 307 of May *cfg_y* drive-by attacks installed nine programs. These incidents can be attributed to a set of Web sites within the "pirate" category that installed similar bundles of spyware. The spyware programs that these sites install dominate our May *cfg_y* top-ten spyware program list in Table 11a.

4.5 Is the Firefox browser susceptible?

Thus far, we have focused on the susceptibility of the Internet Explorer browser to drive-by download attacks. We now turn our attention to the Firefox browser, which is currently the second-most popular browser in use. [4] A common perception about Firefox is that it is more secure against drive-by download attacks, in part because it does not support ActiveX components, a common contributing factor to IE browser vulnerabilities.

To explore this issue, we modified our experimental infrastructure, creating a virtual machine instance that contains Firefox version 1.0.6 on Windows XP with no service packs installed. We replicated all of the heuristics we previously built for IE, such as those that handle JavaScript "time bombs." Finally, we created both *cfg_y* and *cfg_n* Firefox configurations.

In October 2005, we gathered a new crawl of the same Web site categories that we explored for the IE drive-by download study. Our methodology for selecting seed domains to crawl was identical to our other crawls, except we tuned the crawler to favor breadth across sites rather than depth within a site. We did this in anticipation of there being far fewer malicious domains that target Firefox, and accordingly we wanted exposure to a larger number of domains.

Table 13 shows the results of our study. Out of the 45,000 URLs we examined, we found 36 (0.08%) that performed drive-by spyware installs on Firefox. These spyware installs affected only the *cfg_y* browser configuration. We found no *cfg_n* attacks, i.e., we did not observe any Web pages that exploit Firefox vulnerabilities to install spyware without the user's consent.

The few *cfg_y* Firefox drive-by downloads that we observed were based on a Java applet created and distributed by Integrated Search Technologies (IST), a developer of several advertising and browser search redirection software products. The Java applet attempts to install a bundle containing several spyware and adware programs, including DyFuCA and SideFind. An applet is normally prevented from installing new software by Java's security sandbox. Sun's Java Runtime Environment will allow digitally signed applets to run outside the sandbox in some circumstances. In particular, if the applet contains a previously unknown signature, the user is prompted whether or not to trust the

[4] MSIE currently has an 83% market share, while Firefox has an 8% market share, according to `http://www.thecounter.com`.

		URLs crawled	domains crawled	infectious URLs	infectious domains
adult	cfg_y	5,000	744	0	0
	cfg_n	5,000	744	0	0
celebrity	cfg_y	5,000	319	**4 (0.08%)**	**1 (0.3%)**
	cfg_n	5,000	319	0	0
games	cfg_y	5,000	659	0	0
	cfg_n	5,000	659	0	0
kids	cfg_y	5,000	164	0	0
	cfg_n	5,000	164	0	0
music	cfg_y	5,000	392	**7 (0.14%)**	**1 (0.26%)**
	cfg_n	5,000	392	0	0
news	cfg_y	5,000	136	0	0
	cfg_n	5,000	136	0	0
pirate	cfg_y	5,000	300	**25 (0.5%)**	**5 (1.6%)**
	cfg_n	5,000	300	0	0
wallpaper	cfg_y	5,000	272	0	0
	cfg_n	5,000	272	0	0
random	cfg_y	5,000	4,494	0	0
	cfg_n	5,000	4,494	0	0

Table 13: **Drive-by downloads with the Firefox browser.** Drive-by download attacks that cause spyware infections with the Firefox browser in the October 2005 crawl. The few successful drive-by attacks we found used Java applets to attempt to download executables, but required a user to consent to the download.

applet. If the user says yes, the applet is granted permission to execute outside the sandbox. In this particular example, granting permission results in the installation of spyware.

Interestingly, the URLs that perform this drive-by download attack use JavaScript to customize the attack based on the browser that is retrieving the URL. If IE is used, an Active-X based attack is attempted. Otherwise, the Java applet-based attack is used. The applet-based attack apparently works on several different browsers, including Firefox, Mozilla, Netscape, and Avant.[5]

4.6 Summary

To study drive-by installations of spyware using the Internet Explorer browser on Windows, we performed a crawl of 45,000 URLs in May and two crawls of 45,000 URLs in October 2005. Our study found a reduction in the fraction of domains hosting drive-by downloads across the categories we examined. In general, a small number of infectious domains are responsible for the majority of infectious links. Once a user browses an infectious domain, they are very likely to be hit with a spyware infection, often whether or not they respond "yes" to a security prompt. Overall, in our most recent crawl, we found drive-by downloads attempted in 0.4% of the URLs we examined and drive-by attacks that exploit browser vulnerabilities in 0.2% of the examined URLs.

We also examined whether the Firefox browser was susceptible to drive-by installations. We found that only 0.08% of examined URLs performed a drive-by download installation, but all of these required user consent in order to succeed. We found no drive-by attacks that exploited vulnerabilities in Firefox.

5 Conclusions

The goal of this research is to quantify the nature of the spyware threat from a Web-centric point of view. To do this, we conducted a crawler-based examination of both executable content and scripted page content in the Web over a five-month period in 2005.

In the first part of our study, we crawled approximately 20 million URLs in May and again in October of 2005, looking for executable content. Our crawls found executable programs in 19% of the domains we examined. In October, approximately 5.5% of executables and 4.4% of the domains we crawled contained piggy-backed spyware. While the largest portion of that spyware is adware, we found that 14% of the spyware contained potentially malicious functions, such as Trojan downloaders and dialers. We also showed that some Web categories, such as games and wallpaper sites, have a higher concentration of executable spyware than others.

In the second part of our study, we performed three crawls of 45,000 URLs in eight Web categories to examine drive-by download attacks over a period of 5 months. Our results showed a 93% reduction in pages carrying drive-by attacks between May and October, from 5.9% to 0.4%. Approximately 0.2% of the pages crawled in October exploited browser vulnerabilities to install spyware even when the user denied permission for a download or script execution. In the most recent crawl, we found that pirate sites have the greatest risk of drive-by attacks (4.3% of pirate domains), with celebrity sites following close behind (3.9% of domains). For some of the most dangerous sites, we found that the majority of URLs in the domain pose a risk.

Overall, our results show that even with some of the reductions we have seen, the density of spyware on the Web is still substantial. It is difficult to generalize from a single study, but in the zones that we crawled, over one in twenty executables contained piggy-backed spyware. In our Web page crawl, on average, 1 in 62 domains contained at least one scripted drive-by download attack. If these numbers are even close to representative for Web areas frequented

[5]This particular attack has been well-documented. See, for example, the description at `http://www.vitalsecurity.org/2005/03/firefox-spyware-infects-ie.html`.

by users, it is not surprising that spyware on the desktop continues to be of major concern.

6 Acknowledgements

We would like to thank the Computer Science and Engineering support staff and the Computing and Communications staff at the University of Washington for enabling our research. In particular, we want to thank Erik Lundberg and Warren Jessop for fielding the countless security alarms that our crawling generated, and David Sinn and David Richardson for helping set up elements of our crawler infrastructure. This research was supported in part by the National Science Foundation under grants CNS-0430477, CCR-0326546, and ANI-0132817, by an Alfred P. Sloan Foundation Fellowship, by the Wissner-Slivka Chair, and by gifts from Intel Corporation and Nortel Networks.

References

[1] Ad-Aware. `http://www.lavasoftusa.com/software/adaware/`.

[2] America Online and the National Cyber Security Alliance. AOL/NCSA online safety study. `www.staysafeonline.info/news/safety_study_v04.pdf`, October 2004.

[3] Kirk P. Arnett and Mark B. Schmidt. Busting the ghost in the machine. *Communications of the ACM*, 48(8):92–95, 2005.

[4] Neveen Farag Awad and Kristina Fitzgerald. The deceptive behaviors that offend us most about spyware. *Communications of the ACM*, 48(8):55–60, 2005.

[5] Mihai Christodorescu, Somesh Jha, Sanjit A. Seshia, Dawn Song, and Randal E. Bryant. Semantics-aware malware detection. In *Proceedings of the 2005 IEEE Symposium on Security and Privacy*, Oakland, CA, May 2005.

[6] Lee A. Freeman and Andrew Urbaczewski. Why do people hate spyware? *Communications of the ACM*, 48(8):50–53, 2005.

[7] Steve Gibson. Spyware was inevitable. *Communications of the ACM*, 48(8):37–39, 2005.

[8] Qing Hu and Tamara Dinev. Is spyware an internet nuisance or public menace? *Communications of the ACM*, 48(8):61–66, 2005.

[9] Internet Archive. The Heritrix web crawler project. `http://crawler.archive.org/`.

[10] Kazaa. `http://www.kazaa.com`.

[11] Darrell M. Kienzle and Matthew C. Elder. Recent worms: A survey and trends. In *Proceedings of the 2003 ACM Workshop on Rapid Malcode*, Washington, DC, October 2003.

[12] Younghwa Lee and Kenneth A. Kozar. Investigating factors affecting the adoption of anti-spyware systems. *Communications of the ACM*, 48(8):72–77, 2005.

[13] David Moore, Vern Paxson, Stefan Savage, Colleen Shannon an d Stuart Staniford, and Nicholas Weaver. Inside the slammer worm. *IEEE Security and Privacy*, 1(4):33–39, July 2003.

[14] Robin Poston, Thomas F. Stafford, and Amy Hennington. Spyware: a view from the (online) street. *Communications of the ACM*, 48(8):96–99, 2005.

[15] Niels Provos. A virtual honeypot framework. In *Proceedings of the 13th USENIX Security Symposium*, San Diego, CA, August 2004.

[16] Stefan Saroiu, Steven D. Gribble, and Henry M. Levy. Measurement and analysis of spyware in a university environment. In *Proceedings of the First Symposium on Networked Systems Design and Implementation (NSDI '04)*, San Francisco, CA, March 2004.

[17] Mark B. Schmidt and Kirk P. Arnett. Spyware: a little knowledge is a wonderful thing. *Communications of the ACM*, 48(8):67–70, 2005.

[18] Sudhindra Shukla and Fiona Fui-Hoon Nah. Web browsing and spyware intrusion. *Communications of the ACM*, 48(8):85–90, 2005.

[19] Prabhat K. Singh and Arun Lakhotia. Analysis and detection of computer viruses and worms: An annotated bibliography. *ACM SIGPLAN Notices*, 37(2):29–35, February 2002.

[20] Thomas F. Stafford. Introduction to the special issue on spyware. *Communications of the ACM*, 48(8):34–36, 2005.

[21] Jeremy Sugerman, Ganesh Venkitachalam, and Beng-Hong Lim. Virtualizing I/O devices on VMware workstation's hosted virtual machine monitor. In *Proceedings of the 2001 Annual USENIX Technical Conference*, Boston, MA, June 2001.

[22] Sunbelt Software, Inc. Sunbelt Software Aquires Web Spidering Technology. Available at `http://www.sunbelt-software.com/Press.cfm?id=114`, April 2005.

[23] Roger Thompson. Why spyware poses multiple threats to security. *Communications of the ACM*, 48(8):41–43, 2005.

[24] Yi-Min Wang, Doug Beck, Xuxian Jiang, and Roussi Roussev. Automated web patrol with strider honeymonkeys: Finding web sites that exploit browser vulnerabilities. Technical Report MSR-TR-2005-72, Microsoft Research, August 2005.

[25] Yi-Min Wang, Doug Beck, Binh Vo, Roussi Roussev, Chad Verbowski, and Aaron Johnson. Detecting stealth software with Strider GhostBuster. Technical report, Yokohama, Japan, July 2005.

[26] Yi-Min Wang, Roussi Roussev, Chad Verbowski, Aaron Johnson, Ming-Wei Wu, Yennun Huang, and Sy-Yen Kuo. Gatekeeper: Monitoring auto-start extensibility points (ASEPs) for spyware management. In *Proceedings of the 18th Large Installation System Administration Conference (LISA '04)*, Atlanta, GA, November 2004.

[27] Merrill Warkentin, Xin Luo, and Gary F. Templeton. A framework for spyware assessment. *Communications of the ACM*, 48(8):79–84, 2005.

[28] Webroot Software, Inc. Automated threat research. Described at http://research.spysweeper.com/automated_research.html.

[29] Xiaoni Zhang. What do consumers really know about spyware? *Communications of the ACM*, 48(8):44–48, 2005.