

# Spiffy Spyware Stuff

Tushar Dalvi

Nilesh Dalvi

## Abstract

An increasing number of shareware softwares today come with *spyware* programs, programs that collect browsing habits of users and other information and periodically report them to remote host. In this paper, we discuss the use of network based signatures for detecting spywares. We present a survey of commonly found spywares and their working. We also propose a set of basic network signatures and demonstrate that most of the commonly found spywares satisfy them.

## 1 Introduction

Spyware is Internet jargon for Advertising Supported software (Adware). It is a way for shareware authors to make money from a product, other than by selling it to the users. There are several large media companies that offer them to place banner ads in their products in exchange for a portion of the revenue from banner sales.

While this may be a great concept, the downside is that the advertising companies also install additional tracking software on the system, which is continuously calling home, using user's Internet connection and reports statistical data. While according to the privacy policies of the companies, there will be no sensitive or identifying data collected from the user's system and the user shall remain anonymous, it still remains the fact that a live server is active on one's computer continuously sending information about the user and user's browsing habits. A spyware has complete access to one's computer and data and a malicious spyware may send user passwords and credit card information. Spywares also consume one's CPU, memory and bandwidth on someone else's behalf.

The hangover from record downloads of programs that include adware and other spywares has created a matching demand for utilities designed to block unwanted pop-up ads or remove spyware altogether.

There are several softwares available that maintain a database of all the known spywares and the files they install and use the database to detect the presence of spywares on one's system. While they have proven to be quite effective, spyware writers can easily get around them by having clever installations that change the names of the files installed to a random string.

In this work, we propose the use of network based signatures to detect spywares. This scheme works by analyzing the packets in the output network stream and correlating them with the browser activity to infer spyware activity. Network signatures can not only detect adwares that send browsing information but other spying softwares like key loggers.

Our contributions in this project are :

1. Analyzing various spywares currently in circulation and studying their network activity.
2. Proposing a set of basic signatures to look for and demonstrating their effectiveness on current spywares.
3. Providing a tool that sniffs packets and stores along with the process that generated it. This tool is useful in analyzing spywares/detecting signatures.

The rest of the paper is organized as follows. In Section 1, we describe the workings of spywares most commonly found today. In Section 3, we classify these spywares into various categories. In Section 4, we discuss the strategies for detecting these spywares, including our network based signature technique. In Section 6, we describe the related work and we conclude in Section 7.

## 2 Common Spyware

Most spyware that is included in the popular software packages available on the Internet comes from a small

set of highly used spyware. The activity of a number of such spyware (as well as some uncommon pieces of spyware) has been analyzed, and the results are presented here.

## 2.1 Gator

The Gator Advertising and Information Network (GAIN) [GC03], powered by The Gator Corporation, comes with many popular software applications and services. It delivers ads, information, and software based on the web sites viewed by the user.

Gator runs as a standalone process called `GMT.exe` and monitors user's browser activity to attempt to better target the ads shown. It randomly selects one of the following six hosts to which to send information: `bannerserver.gator.com`, `ss.gator.com`, `rs.gator.com`, `gs.gator.com`, `bg.gator.com` and `search.gator.com`. Each time a website is visited, it sends one of these hosts information about the name of the website, the local time, a uniquely identifiable machine ID, browser type, etc. Not everything that was sent was in plain text. No personal information or passwords were seen in the packets, but the plain text did not account for everything it sent.

## 2.2 Comet Cursor

Comet Cursor [CS03] is an Internet Explorer toolbar. Since it does not run as a separate process, it was difficult to isolate its network activity for study. To study Comet Cursor's traffic, the ambient network activity of the system was reduced as much as possible, and all packets that could otherwise be accounted for were discarded. It was observed that Comet Cursor randomly sends information to either `rs.cometsystems.com` or `log.cc.cometsystems.com`.

Comet Cursor's activity immediately follows the browser activity. It is more selective in what it sends. Names of websites visited by browsers were never seen in the packets. However, it does send all the search terms entered in the forms in search engines such as Google and Yahoo. The names of the search engines are perhaps hard coded inside the Comet Cursor code.

## 2.3 Alexa

Alexa [Ale03] is another Internet Explorer toolbar, powered by Amazon. It sends its information to

`data.alexa.com`, including the names of all the URLs visited by the browser.

It uses HTTP for communication. HTTP seems to be the favorite among spywares, perhaps because many users in academic/corporate environments are behind firewalls and most ports are blocked, while HTTP is always the safest choice.

## 2.4 Google Toolbar

The Google Toolbar [Goo03] is an Internet Explorer plugin designed to give the user easier access to Google's services. Because it is a browser plugin, the toolbar does not appear in a process of its own. With the PageRank feature disabled, using the toolbar is no different from using the Google search page itself. With PageRank turned on, however, on every web access, the URL being visited is sent to Google via HTTP to retrieve the PageRank information. The toolbar is careful, however, to not relay to Google any data that was entered into web forms, or to relay intranet URLs.

## 2.5 KeySpy

KeySpy [IIP03] is a commercial keylogger application produced by IIPwr. It is invisibly installed on a victim system, set to execute whenever the system reboots, and does not appear in a process of its own. It sits in the background, silently monitoring and recording all user keystrokes, window titles, as well as all process names, IDs, and start and stop activities. At regular intervals (configurable by the attacker), KeySpy sends a report via e-mail to the attacker, optionally in an encrypted form. When sending its report, KeySpy spawns a separate process to open its TCP connection to the SMTP server. Once the report has been sent, this process terminates, eliminating any lingering `CLOSE_WAIT` or `TIME_WAIT` entries that might show up in a casual `netstat`.

## 2.6 SaveNow

SaveNow [Whe03] is adware that is often bundled as a third-party component with other software, including popular file-sharing packages such as KaZaa [SN03] and BearShare [FPI03]. It lives in a process of its own, called `save.exe`, monitoring the user's browser activity to attempt to better target the ads shown. As often as once a minute (triggered by the user's browser activity), SaveNow contacts an Akamai server via HTTP to

download its ads. It does not appear to send any information about what sites have been visited, or what data has been entered into web forms.

## 2.7 Web3000

Web3000 [Net03] is another Ad network. Its software is bundled with Netsonic Download Accelerator in the form of *Ezula TopText*. It inserts its own text advertisements as pop-ups linked to highlighted words in a Web page. It does not show up in the process list under the Windows Task Manager, but HTTP packets are seen to be transmitted to `www.ezula.com` containing the names of all the URLs visited by the browser.

## 3 Classes of Spyware

Based on the observations above, it appears that spyware can be classified into several categories according to their behaviour. This classification is useful in guiding the development of strategies for dealing with spyware.

### 3.1 Process Location

Spyware can either live in its own process, or it can be attached to one or more host process, either through a plugin, or by replacing library files. Clearly, the presence of spyware that lives in its own process can be easily detected by examining the system's process table. Network activity that is generated by such spyware can also be easily isolated for analysis and detection of suspicious activity. Spyware that lives in its own process include Gator and SaveNow.

### 3.2 Frequency of Reporting

By its very nature, spyware must periodically send over the network, reports of its observations. Spyware can be categorized according to the frequency of such reporting. Many simply send a report immediately after user activity has been observed. Others can batch up such reports and send a larger or summarized report after every few minutes or after every few instances of user activity.

Gator, Comet Cursor, Alexa, and Google Toolbar all send information with every user click in the web browser. SaveNow sends information once a minute.

KeySpy reports are sent every few minutes; their frequency is uncorrelated with user activity.

## 3.3 Content of Reports

## 4 Strategies for Detection

In this section, traditional techniques for detecting spyware are first described. These include detection based on firewalls and databases detection. Then network-based signatures are discussed.

### 4.1 Firewall-Based Detection

One simple way of detecting spyware is to install a firewall, such as *Zone Alarm* [ZL03]. Zone Alarm blocks all the processes from connecting to the Internet or acting as servers. Access to the network can be selectively enabled by the user on a per-process basis, depending on the processes that user recognizes. Many pieces of spyware that attempt to hide themselves from the user come to surface in this way.

There are two downsides to the firewall approach. First, a process recognized by the user as requiring network access may be surreptitiously engaged in spyware activities. For instance, the FTP client *CuteFTP* [Glo03] once spied on its users in the background. Secondly, many spywares embed themselves into web browsers as either plugins or toolbars and they cannot be selectively denied access to network.

### 4.2 Signatures Based on Databases

This is the approach taken by many existing spyware detectors [Lav03, BPS03, Spy03]. They maintain an up-to-date database of the latest versions of all known spyware. The database includes lists of files installed by the spyware, the process names under which they run, etc.

Each spyware installs few dynamically linked libraries (DLLs) in the Windows system directory. This serves as an easy test for their detection. For instance *Cydoor* [Med03] installs `CD_CLINT.DLL` `CD_GIF.DLL` and `CD_SWF.DLL`.

Existing spyware detectors also examine the list of processes running in the system to detect instances of spyware that may be running. For example, Gator runs under the process name of `GMT.exe`, while SaveNow runs as `save.exe`.

Database based spyware detection is a very powerful technique for detecting existing spyware. However, it has its limitations. It cannot detect new spyware or new versions of spyware that are not in the database. Also, spyware can get around these detectors by having clever installations where the names of the installation files are changed to be random strings.

### 4.3 Network-Based Signatures

Based on the spyware analyzed in Section 2, it appears that network-based detection schemes can be more robust and flexible than detection mechanisms based on firewalls or databases. In particular, a network-based detector can analyze the outgoing packet stream for any of the following classes of signatures.

1. Network activity in the form of packets being sent to a fixed destination or to a small set of fixed destinations.
2. Network activity that is correlated with browser activity.
3. Network activity that is seen periodically at fixed intervals.
4. Packets containing the names of the websites visited by the browser.
5. Packets containing information that has recently been entered into a web form.

It is expected that Class 1 signatures should be satisfied by all spyware but perhaps also by many other benign applications. All spyware should also satisfy either Class 2 or Class 3 signatures, and these would be good indications of spyware activity. Finally, the presence of Class 4 or Class 5 signatures would be a clear indication that spyware is present in the system.

Table 4.3 summarizes the various signatures satisfied by the applications studied in Section 2.

#### 4.3.1 Detecting Network Signatures

We have implemented a packet sniffer that traces the packets back to the processes (along with their names) that created them. It serves two important purposes. First, it segregates the network activities of trusted processes and helps the user to concentrate on only those network packets that correspond to potential spyware

processes. Second, each process can be studied in isolation for spyware activity.

For spyware that runs as a separate process, it is easy to test each class of signatures. A browser workload is created, where the browser visits few fixed sites and fills out forms with some fixed information. All of the network signature classes are easily detectable by looking at the packets generated by these processes. Class 1 can be detected by looking at the total number of unique hosts that the process connects to. The packets can be scanned to look for the names of the web sites/information filled in the form.

For spyware that embed themselves into the browser, a similar approach can be taken. There could be multiple pieces of spyware running; each visit to a website could generate lots of packets to various different hosts. Class 1 can be detected by looking for destination hosts that repeat periodically in the logs. With a spyware connecting to  $n$  random hosts, each host will repeat with a frequency of  $\frac{1}{n}$  on average. Packets to each such destination host can be scanned for instances of the other signature classes.

#### 4.3.2 Packet Sniffer

This section describes the implementation of the per-process packet sniffer. There are two aspects of the implementation:

1. Getting Packets: The *WinPcap* [Win03b] Windows packet capture library is used. This library is used by many current packet sniffers, such as *WinDump* [Win03a] and *Ethereal* [Eth03]. It includes a kernel-level packet filter, a low-level dynamic link library, and a high-level system-independent library. We use the library to isolate the TCP packets sent from the given machine. These packets have the information about the TCP ports used to send the packets but do not reveal the names of processes that generated them.
2. Getting Process Names: The IP helper API of the Microsoft Platform SDK can be used to access the data structure storing the list of open TPC ports on the system. Windows XP has an extended query feature that gives the additional information about the PIDs of the processes using these ports. This information, joined with the port information of the packets, helps in tracing the packets back to the originating processes.

	1	2	3	4	5
Alexa	✓	✓		✓	✓
Comet Cursor	✓	✓			✓
Gator	✓	✓		✓	✓
Google Toolbar	✓	✓		✓	
KeySpy	✓		✓		
SaveNow	✓	✓	✓		
Web3000	✓	✓		✓	✓

Figure 1: Signature classes satisfied by various pieces of spyware

## 5 Removal of Spywares

After spyware activity has been detected in the system, an interesting and intriguing problem deals with blocking the spyware activity. For the database based spyware detectors, it is easy to uninstall the spywares as they contain the list of files installed by the spywares. For network signature based detectors, it is a much harder problem. One possibility is to block the spywares at network level. Care has to be taken to block the packets sent by the spywares as well as packets sent to spywares. The incoming traffic to spywares need to be blocked because remote hosts may be sending them advertisements to display or even executables to run on the user machine. For spywares identified as separate processes, all the packets sent to the process or the packets sent by the process can be dropped. For spywares sitting inside the browser, a list of remote hosts corresponding to the spyware activity can be made and any packets with those hosts as source or destination can be dropped.

Spyware blocking at network level is of course not foolproof and complete removal of these spywares from the system is sought. Again, for spywares with process names, the names of the executables are known and the Windows Installer can be summoned up to remove the corresponding program. For spywares running inside browser, this is an almost intractable problem as it is impossible to know the application whose installation brought the spyware. Two approximate strategies can be employed here. First, the spyware detector can run continuously in the background and coordinate with the Windows Installer. If spyware activity suddenly crops up in the system, the detector can relate it to the application Installer installed and try removing it. Another possible strategy is to perform a DNS lookup for the remote host that spyware is contacting and try to relate its name with names of applications. For example,

Gator contacts `gator.com` servers and Comet Cursor contacts `cometlabs.com` servers.

## 6 Related Work

The general problem of detecting attacks against a computer system over a network is called *network intrusion detection*. Monitoring network activity to detect spyware is a special case of network intrusion detection in which an illicit program, installed on the system alongside legitimate software, sends personal information about the user to a remote site over the network.

Most network intrusion detection systems (NIDSs) [MHL94] work by inspecting IP packets and reconstructing the higher level interactions between end hosts and remote users. For example, Heberlein et al. [HDL<sup>+</sup>90] have developed a network security monitor that compares current network usage patterns with a historical profile to detect abnormal activity.

The EMERALD environment [PN97] is a distributed NIDS that can track malicious activity in large networks. EMERALD surveillance and response monitors are distributed over a network and can be independently tuned. These monitors provide data to an event-analysis system that combines signature analysis with statistical profiling to detect suspicious network activity.

Bro [Pax98] is a passive network monitoring system for detecting intruders in real-time on high-speed networks. The network traffic stream is filtered and reduced to a series of events. These events are then processed by event handlers, written in a specialized scripting language. Event handlers in Bro can update state information, synthesize events, generate logging information, and generate real-time notifications.

Sekar et al. [SGVS99] have designed a rule and pattern-based NIDS that features a specification lan-

guage with a strict typing discipline. The specification language allows the user to define rules and patterns in network packets corresponding to anomalous behavior.

Writing the rules for a rule-based NIDS, however, requires expert security knowledge and is an expensive process that often involves hours of tedious programming and debugging. Lee et al. [LPS99] have developed a data mining framework for adaptively learning these rules for use in Network Flight Recorder [NFR97].

There does not appear to be any previous work done on spyware detection, either by identifying such rules, or by applying network intrusion detection techniques to the problem. Our work deals with extending these techniques for spyware detection by identifying the network signatures common to spywares.

There are many commercial software products available that detect the presence of spyware by examining the host's file system. These are based on signature databases and look for the installations of the currently known spywares. These systems are limited in the scope as they cannot detect new or unknown spyware. We plan to extend the scope using our techniques.

## 7 Conclusions

### References

- [Ale03] Alexa. Alexa Toolbar. <http://www.alexa.com/>, 2003.
- [BPS03] Bullet Proof Soft. BPS Spyware Remover. <http://www.bulletproofsoft.com/>, 2003.
- [CS03] Comet Systems. Comet cursor. <http://cometcursor.cometsystems.com/>, 2003.
- [Eth03] The Ethereal Network Analyzer. <http://www.ethereal.com/>, 2003.
- [FPI03] Free Peers Inc. BearShare. <http://www.bearshare.com/>, 2003.
- [GC03] Gator Corporation. Gator. <http://www.gator.com/>, 2003.
- [Glo03] GlobalSCAPE. CuteFTP. <http://www.cuteftp.com/>, 2003.
- [Goo03] Google. Google Toolbar. <http://toolbar.google.com/>, 2003.
- [HDL<sup>+</sup>90] L. Heberlein, G. Dias, K. Levitt, B. Mukherjee, J. Wood, and D. Wolber. A network security monitor. In *IEEE Symposium on Research in Security and Privacy*, pages 296–304, May 1990.
- [IIP03] IIPwr. IIPwr Package. <http://www.iipwr.com/IIPwrPackage.htm>, 2003.
- [Lav03] Lavasoft. Ad-aware. <http://www.lavasoftusa.com/>, 2003.
- [LPS99] Wenke Lee, Christopher T. Park, and Salvatore J. Stolfo. Automated intrusion detection methods using NFR. In *USENIX Intrusion Detection Workshop*, 1999.
- [Med03] Cydoor Desktop Media. Cydoor. <http://www.cydoor.com/>, 2003.
- [MHL94] B. Mukherjee, L. Heberlein, and K. Levitt. Network intrusion detection. In *IEEE Network*, pages 26–41, May/June 1994.
- [Net03] Netsonic. Web3000. <http://www.web3000.com/>, 2003.
- [NFR97] Network flight recorder. <http://www.nfr.com/>, 1997.
- [Pax98] Vern Paxson. Bro: A system for detecting network intruders in real-time. In *Computer Networks*, pages 2425–2463, December 1998.
- [PN97] Phillip A. Porras and Peter G. Neumann. Emerald: Event monitoring enabling responses to anomalous live disturbances. In *NIST-NCSC National Information Systems Security Conference*, 1997.
- [SGVS99] R. Sekar, Y. Guang, S. Verma, and T. Shanbhag. A high-performance network intrusion detection system. In *ACM Conference on Computer and Communications Security*, pages 8–17, 1999.
- [SN03] Sherman Networks. KaZaa media desktop. <http://www.kazaa.com/>, 2003.
- [Spy03] Spybot. Spybot S&D. <http://security.kolla.de/>, 2003.
- [Whe03] WhenU.com. SaveNow. [http://www.whenu.com/about\\_savenow.html](http://www.whenu.com/about_savenow.html), 2003.
- [Win03a] Windump: tcpdump for Windows. <http://windump.polito.it/>, 2003.
- [Win03b] Winpcap: the free packet capture architecture for Windows. <http://winpcap.polito.it/>, 2003.
- [ZL03] Zone Labs. Zone Alarm. <http://www.zonelabs.com/>, 2003.