

*Master Thesis in
Software Engineering
Thesis no: MSE-2003-05
March 2003*



Investigating Spyware in Peer-to-Peer Tools

**Martin Boldt
Johan Wieslander**

Department of
Software Engineering and Computer Science
Blekinge Institute of Technology
Box 520
SE - 372 25 Ronneby
Sweden

This thesis is submitted to the Department of Software Engineering and Computer Science at Blekinge Institute of Technology in partial fulfilment of the requirements for the degree of Master of Science in Software Engineering. The thesis is equivalent to 20 weeks of full time studies.

Contact Information:

Authors:

Martin Boldt

E-mail: mae@bth.se

Johan Wieslander

E-mail: jws@bth.se

University advisors:

Bengt Carlsson

Department of Software Engineering and Computer Science

Department of
Software Engineering and Computer Science
Blekinge Institute of Technology
SE - 372 25 Ronneby
Sweden

Internet : www.ipd.bth.se
Phone : +46 457 38 50 00
Fax : + 46 457 271 25

INVESTIGATING SPYWARE IN PEER-TO-PEER TOOLS

Master Thesis

Martin Boldt

Blekinge Institute of Technology
Department of Software Engineering and Computer Science
martin.andersson@bth.se

Johan Wieslander

Blekinge Institute of Technology
Department of Software Engineering and Computer Science
johan.wieslander@bth.se

4th April 2003

Contents

1	Introduction	2
1.1	Background	2
1.2	Problem formulation	3
1.3	Terminology	4
1.3.1	Privacy intrusion	4
1.3.2	Adware	5
1.3.3	Spyware	5
1.3.4	P2P technology	6
1.4	Related work	7
2	Investigation	8
2.1	Selected P2P tools	8
2.1.1	BearShare v4.1.1	8
2.1.2	ICQ 2002a Build 3728	9
2.1.3	iMesh 3.1	9
2.1.4	Kazaa Media Desktop 2.0	10
2.1.5	Morpheus 2.0	11
2.1.6	eDonkey 2000 v35.16.61	11
2.2	Laboratory environment	11
2.2.1	Hardware	12
2.2.2	Software	12
2.2.3	Shared files	13
2.2.4	Network	13
2.2.5	Cloning system	13
2.3	Method description	15
2.3.1	Techniques and tools	16
2.3.2	Analysis method	19
3	Analysis	23
3.1	File system lists	23
3.2	Firewall logs	24
3.3	Registry data	24
3.4	Network data	24
3.5	Ad-aware logs	28
3.6	Identified components	28

4	Discussion	30
4.1	What kind of components are a couple of well known peer-to-peer tools bundled with?	31
4.1.1	Sentry.exe in Morpheus	31
4.1.2	Eac_Rvndl in iMesh	32
4.1.3	Further thoughts	32
4.2	Do these activities intrude on user privacy?	33
4.3	How can an investigation method that discovers privacy intrusive components be constructed?	33
4.4	Planning of analysis vs. analysis results	34
4.5	Correctness of collected data	34
4.6	Analysis tool proposal	35
5	Conclusion	37
5.1	Conclusion	37
5.2	Future work	37
6	Appendices	40
6.1	Hardware specification	40
6.2	Application base specification	41
6.3	Web surfing script	42
6.4	P2P tool investigation work list	45
6.4.1	Installation	45
6.4.2	Running the tool (30 min/100 min)	45
6.4.3	Removal	46

List of Tables

2.1	Description of shared files.	13
3.1	Components that survived removal of their host software.	24
3.2	Automatically activated components in system startup.	25
3.3	BearShare packet counts.	26
3.4	BearShare protocol statistics.	26
3.5	iMesh packet counts.	26
3.6	iMesh protocol statistics.	26
3.7	Kazaa packet counts.	26
3.8	Kazaa protocol statistics.	26
3.9	Morpheus packet counts.	27
3.10	Morpheus protocol statistics.	27
3.11	Classification of identified components.	29
3.12	Identified sub-components.	29
4.1	Analysed data.	34

Abstract

Peer-to-Peer (P2P) tools are used exclusively when their users are connected to the Internet, thus constituting a good foundation for online commercials to help finance further tool development. Although software that displays ads (adware) is very common, activity monitoring or information collecting software that spies on the users (spyware) may be installed together with the P2P tool. This paper will present a method for examining P2P tool installations and present test results from a few of the most common P2P tools. It will also discuss whether these tools, with their bundled software, make any privacy intrusions. Finally, the method itself will be evaluated and suggestions of refinements will be proposed.

Keywords: Peer-to-Peer, Spyware, Adware, Privacy

Chapter 1

Introduction

1.1 Background

Some techniques that were originally created to provide sensible functionality are today misused to monitor user activity. Examples of such techniques are web browser cookies, HTTP referrers¹ and HTML image source tags. By using these techniques, and combining them, companies such as advertising agencies or media content providers can collect data to sell for their own profit. The data collected might be:

- visited web pages
- downloaded images, music or software
- items purchased online
- your installed computer hardware and software
- other personal information (e.g. street address, phone number, family members etc.).

This kind of information gathering form the basis of two new techniques called adware and spyware (see definitions below). They are a kind of software that is generally bundled with (i.e. packaged with) other software, often free- or shareware such as games, audio/video players or P2P (Peer-to-Peer) tools. Although both of them are concerned with security and privacy issues, the main purpose of adware is to display ads on a web page or in a program GUI (Graphical User Interface), rendering profit for the software copyright owners. Spyware, on the other hand, has the purpose of also gathering information about the user or the users' activities and to secretly send it to some predefined recipient. Though licence agreements of spyware bundled software attempt to specify what software is bundled and its activities, the texts tend not to be clear and unambiguous. It is therefore easy to get spyware installed along with some freeware or shareware, without knowing about it. There have been cases where the software authors have claimed their product to be spyware free when it has not which we will come back to.

¹“Web Forum Account Hijacking Vuln.”, Bugtraq post, <http://www.securityfocus.com/archive/1/223799>

1.2 Problem formulation

It has come into interest to examine widely used software, for example file sharing utilities (e.g. P2P tools) and media players, and to discuss any suspicious components found. A few good tools against adware and spyware exist, such as Ad-aware, but they typically accomplish a single or only a few tasks. There is a lack of tools when it comes to monitoring possible adware or spyware in runtime and logging their network activities.

We have chosen to examine P2P tools to find possible adware or spyware. The reason for this is that several popular P2P tools are known to have contained spyware and several of them still contain, at least, adware. Furthermore, the use of P2P tools lies entirely in networking — they have no other purpose and are useless without a network connection. It is thus a fair assumption that a network connection is available, most likely to the Internet, whenever the P2P tool is running. Any bundled software is likely to succeed in attempts to transfer data, making it possible for both adware and spyware to work as intended and possibly also go undetected. Moreover, other common activities among Internet users are web surfing, e-mail transferring and instant messaging or chatting. Any spyware could have the opportunity to eavesdrop on that communication. Only by monitoring instant messaging activities, thus not eavesdropping the content, sensitive information such as employee work habits[6] can be obtained.

The P2P tools that have been known to contain spyware have file sharing as their main purpose, in contrast to instant messaging services. Early P2P tools would share only MPEG encoded sound files but the most popular tools today have the ability to share any kind of files. Some very common file formats are mp3, wav, GIF, jpg, AVI and MPEG. Several of these file formats may contain compressed data. Depending on codec (coder/decoder), all these formats give good data compression rates. However, compressed text data would in comparison, even at moderate compression levels (Lempel-Ziv² compression using a small block sizes, typically around 100 KB) produce an output substantially smaller than an mp3-encoded song (typically 3–4 MB) or a software installation file (0.5–1 MB for a relatively simple Windows tool). Also, a simple home computer today (Pentium III 500 MHz CPU, 64–129 MB RAM) running any newer version of 32-bit Windows would not show any noticeable CPU load when compressing text corresponding to a dozen of full A4 pages. Furthermore, because ADSL and broadband Internet connections are common today, transferring such text data would typically not produce any noticeable network load either (20 KB would take 1,6 s at 100 kbit/s), but instead go undetected unless the network traffic was thoroughly examined. To estimate some shortest time a typical P2P file sharing tool user would be connected to the Internet, simple calculations reveal that transferring 3 MB data through a 100 kbit/s connection would take 4 minutes. Any other network transfers, e.g. downloading e-mail, web surfing or downloading additional files from the P2P network and indeed the P2P connectivity traffic itself, would of course increase the transfer time, thus making the user stay connected for a longer time. From this line of reasoning we conclude that it is possible to hide especially text data transfer, but generally any data depending on amount and the circumstances described above, from the user if a spyware would attempt to do so[1].

²The Lempel-Ziv compression algorithm is used in gzip, zip and PKZIP.

Having given this background, we put forward the questions if the currently most common P2P tools contain spyware and how a method for investigating them would be constructed. Although previous investigations may have produced reliable results, it is likely that the bundled software is updated or even replaced (i.e. as a counter-measure against adware/spyware removers, anti-virus software, or blacklisting). A previously adware bundled P2P tool could suddenly contain spyware or vice versa. Also, the views on what software is spyware, and what its activities are, vary. Examining all the ways in which a P2P tool could affect the operating system, not only a subset, would be sensible. Furthermore, the tool support in this area of research, for the win32 platform, is not very good. Useful tools exist but they only perform simple tasks, leaving substantial manual work to be done.

Conclusively, this thesis aims to answer the following questions:

- What kind of components are the most common peer-to-peer tools (P2P) bundled with?
- Do these activities intrude on user privacy?
- How can an investigation method that discovers privacy intrusive components be constructed?

1.3 Terminology

This section will describe the terminology of the main concepts in this report. Here, we will also give our definition of privacy intrusion.

1.3.1 Privacy intrusion

We define privacy intrusion[1][9][4] as an action, in this case performed by software, that accomplishes any of the following:

- Information about the computer or the user(s) is sent without explicit permission. Such Information could be hardware configuration, operating system and version, installed software, system configuration (computer name), user name, number of users or account settings (group).
- Explicit location information (other than traceroute equivalent results) is sent without explicit permission.
- Information about file names and file formats is collected and sent without explicit permission. (Both files on disk and files being searched for, or transferred, in the P2P tool.)
- Information is collected from browser cache, history or cookies and sent.
- Information is collected from other files on the hard drive and sent.

P2P tool licence agreements may state that third party software will be installed. If such a licence agreement does specify the name of all third party software that is installed or its functionality (e.g. *read Internet Explorer cookies and history* or *collect computer usage statistics*) and if it sends any private

information according to our definition, the third party software will not be suspected of making any privacy intrusions. However, if a description of what the software does is not correct (i.e. the software does more than what is stated) or a correct description cannot be found at either the P2P tool or third party software vendor websites, the third party software will be suspected to make privacy intrusions.

Furthermore, we chose in our definition not to include the following as resulting in privacy intrusion because they need to be exposed in order to make the networking function:

- IP address is exposed or collected.

1.3.2 Adware

Adware is any software that displays advertising banners while running the program. It is usually software that can be freely downloaded from the Internet, but it contains advertisements such as banners that create revenues for the responsible company. These advertising revenues help financing development and thereby lower the product cost for the end user. Therefore it is most often possible to download these programs free of charge on the Internet. There are however a number of problems concerning adware programs.

First, the component responsible for displaying the advertisement banners might have been imported from another company, e.g. DoubleClick³. Then there exist a connection between the end user and the ad-supplying company. These advertising components usually send marketing information (spending and surfing habits) back to their company when connected to the Internet. However, all adware include a disclosure telling the user that the company will use this information. If they fail in doing so they are classified as spyware instead.

Second, adware consume the users' computer resources. As soon as new advertising banners needs to be downloaded the users' bandwidth consumption will increase and this might be a serious problem when dealing with parsimonious bandwidth techniques, such as modems. Similarly will CPU load increase when displaying advertising banners. Both resource problems will drastically increase when dealing with new banner techniques involving both animated 3D graphics and stereo sound⁴. These problems could be further amplified if dealing with advertising components that are poorly implemented which therefore decrease its performance.

1.3.3 Spyware

It is important to understand what spyware is. However, it is equally important to understand what spyware is not and what it does not do, especially since there is much gossip and rumours travelling the Internet that involves spyware. What we need is an accurate definition of spyware. Steve Gibson⁵ has given the following definition of spyware:

³DoubleClick, <http://www.doubleclick.com>.

⁴Brilliant Digital, <http://www.brilliantdigital.com>.

⁵Gibson Research Corp., <http://grc.com/optout.htm>.

“Silent background use of an Internet ‘backchannel’ connection MUST BE PRECEDED by a complete and truthful disclosure of proposed backchannel usage, followed by the receipt of explicit, informed, consent for such use. ANY SOFTWARE communicating across the Internet absent these elements is guilty of information theft and is properly and rightfully termed: Spyware.”

It is important to understand that software that only gathers information about the user is not considered spyware. Even software that gathers information about the user and then sends this information to some central processing unit is not considered to be spyware, *as long as the user is notified about these activities in advance.* When using the Steve Gibson definition of spyware the following conclusion can be drawn:

“This means that a software can be Adware and Spyware at the same time! More importantly, not all Adware is Spyware and most Spyware is NOT easily detected by displaying ads.”⁶.

In this thesis and during our investigation we have chosen to use the Steve Gibson definition of spyware, with one addition. We don't define programs storing IP address, even without notifying the user, as spyware.

1.3.4 P2P technology

Protocols that use the peer-to-peer model[6] to communicate let all hosts (peers) communicate on equal conditions. Bo Leuf gives the following definition of P2P: *“In essence, peer-to-peer simply means equal communicating with equal”*[6]. This means that any peer in a P2P network can connect to any other peer on that network. A metaphor with the client/server model[14] would be that any peer is both a client and a server at the same time and therefore is equal communication between these nodes possible. There have been, and there still is, an increasing demand for P2P communication due to programs such as Gnutella⁷ and Kazaa⁸. However, the problem with networks built during the last decade is that these are asymmetric. One example is ADSL[14] (Asymmetric Digital Subscriber Line) which allows traffic going downstream to travel 8 times faster than traffic moving upstream. This means the network is suited for activities such as surfing the web or using news groups; technologies which work the same way as television and newspapers where users just consumes procreated material.

However, this totally disagrees with the basic meaning of P2P where all peers should communicate on equal conditions; it should be possible to both send and receive information at the same speed. And this is a great problem today since we see an increasing demand for P2P services on the Internet⁹. It might even be the case that P2P applications require these network constructions to change.

⁶iOpus adware website, <http://www.adware.info>.

⁷Gnutella protocol spec., http://www9.limewire.com/developer/gnutella_protocol_0.4.pdf.

⁸Kazaa, <http://www.kazaa.com>.

⁹Look at the download statistics for P2P tools from e.g. <http://www.download.com>.

1.4 Related work

Little academic work has been conducted in the field of adware and spyware. Our main resources are two books about P2P technology and a couple of web pages. The books are Bo Leufs “Peer to Peer - Collaboration and Sharing over the Internet” [6] and “Peer-To-Peer - Harnessing the Power of Disruptive Technologies” [8]. These books cover the P2P area. Unfortunately none of them discuss adware or spyware in any detail. To find information about this we had to find various Internet sources. All Internet sources referred to in this report have been tested and proved to be working at the 3rd of April 2003. Our foremost Internet resources are:

- Counterexploitation, www.cexx.org
Established in 1998, deals with areas like privacy, online advertising, spam and spyware.
- SANS (SysAdmin, Audit, Network, Security), www.sans.org
The site was established in 1989 as a cooperative research and education organization regarding computer security.
- Lavasoft support forums, <http://www.lavasoftsupport.com>
Discussion on spyware and adware from the authors of Ad-aware.
- Security focus and Bugtraq, <http://www.securityfocus.com>
Discussions, articles and vulnerability information related to computer security.

Chapter 2

Investigation

2.1 Selected P2P tools

We selected the P2P tools to investigate by comparing how well spread they were. To obtain this information we compared the number of downloads that had been registered on Download.com¹ for each tool. We could have looked at other sources as well, but chose not to because we considered the Download.com statistics good enough and sufficient for our investigation. All tools listed below were downloaded from Download.com on the 27th of November 2002 between 17:00 and 18:00 hours.

2.1.1 BearShare v4.1.1

BearShare is developed by Free Peers Inc.² and is based on the open Gnutella protocol. It has been downloaded more than 18 million times from download.com³. There exist two different solutions for a user that wants to use BearShare. First there exist a freeware version that includes bundled software, this is the version we have chosen for our investigation. There is also an opportunity to buy a version that doesn't contain any bundled software or advertising from Free Peers Inc.

On the BearShare website, Free Peers Inc. state that *“BearShare says ‘NO!’ to Spyware”*. In their spyware statement⁴ Free Peers Inc. explain that they agree with Steve Gibson on the definition of spyware, just as we do (see section 1.3.3). Moreover, Free Peers Inc. denies that BearShare include any form of spyware: *“Free Peers, Inc. does not condone the use of ‘spyware’ and does not use ‘spyware’ in their BearShare products.”* However, further down in the same statement Free Peers Inc. declares that *“We do not consider SaveNow to be ‘spyware’.”* SaveNow which is a bundled software developed by WhenU⁵ is included in BearShare. Other sources such as Cexx.org claims that SaveNow do intrude on users' privacy: *“SaveNow (WhenUShop) - Installed by BearShare among others. Put quickly, an advertising toolbar that monitors what sites you*

¹Download.com, <http://www.download.com>.

²BearShare, <http://www.bearshare.com>, FreePeers <http://www.freepeers.com>.

³These figures were accurate in April 2003.

⁴BearShare spyware statement, <http://www.bearshare.com/nospy.htm>.

⁵WhenU, <http://www.whenu.com>.

visit and pops up sponsored 'deals' when products/shopping/etc. appears on those sites.”[2]. Cexx.org also classifies SaveNow as foistware that come along, trojan-style, with completely unrelated software. What is significant for foistware is that once it has sneaked into systems it is very hard to get rid of.

We also tried to find out exactly what software was bundled with BearShare. However, this information was not available on BearShare’s official website.

2.1.2 ICQ 2002a Build 3728

It is the Israeli company ICQ Inc.⁶ (which is the successor of Mirabilis Ltd.) that develops ICQ. This tool differs from the other tools in our investigation since it is an instant messaging service and not a file sharing tool. We thought that ICQ was interesting enough to be included in our investigation, mainly because of their privacy policy and since it is so well spread with over 226 million downloads⁷.

We could not find any sources claiming that ICQ include spyware or carry out any privacy violation. Version 2000b and later do contain adware since these versions contain advertising banners. ICQ Inc. privacy policy⁸ clearly states that ICQ should be considered insecure, e.g. *“Also please note that the ICQ software, as with most Internet applications, is vulnerable to various security issues and hence should be considered unsecured.”*. After this statement there is a listing of various risks that are associated with the ICQ software. Among other risks *“Unauthorized invasion of your privacy”* is mentioned.

2.1.3 iMesh 3.1

iMesh is developed by iMesh (Israel) Ltd.⁹ and has been downloaded more than 46 million times from download.com¹⁰. In the corresponding spyware statement¹¹ iMesh Ltd. declare their product as being totally free of spyware. iMesh Ltd. admit that they include Cydoor¹² as bundled software for supply of advertising banners and that Cydoor should not be confused with spyware. In the spyware statement they also explain that *“Neither iMesh or Cydoor tracks the iMesh users in any way or collects any data about them.”*. Further on, iMesh Ltd. declare that they have undertaken much labour to become spyware free, or as they state *“We’ve put a lot of effort in becoming ‘SpyWare Clean’ by talking to all the main SpyWare gurus and following their advices about telling our users ahead about the Cydoor ad engine, and making sure we and Cydoor only do ad delivery activities.”*

We tried to find information that described exactly what software iMesh bundles with on their official website. Unfortunately we couldn’t reveal this information from iMesh Ltd.

⁶ICQ Inc., <http://www.icq.com/company/about.html>.

⁷These figures were accurate in April 2003.

⁸ICQ privacy statement, <http://www.icq.com/legal/privacy.html>.

⁹iMesh, <http://www.imesh.com>.

¹⁰These figures were accurate in April 2003.

¹¹iMesh spyware statement, <http://www.imesh.com/SpyWare.html>.

¹²Cydoor, <http://www.cydoor.com>.

2.1.4 Kazaa Media Desktop 2.0

Kazaa is developed by Sharman Networks which is a company focused on P2P software development. Since Kazaa has been downloaded more than 209 million times¹³ from Download.com it was evident that it should be included in our investigation. Sharman Networks have separated their privacy statement into three different written documents. Their privacy statement¹⁴, ad support statement¹⁵ and finally a spyware statement¹⁶.

In the privacy statement Sharman Networks clearly declare that they do not leak private information about their users without obtained permission. But what if users implicitly give their permission, e.g. by accepting some obscure licence agreement or by participating in any of the online surveys and contests that Kazaa frequently carry out? These surveys and contests are fully voluntary but if a user chooses to be part in one of these he jeopardize his privacy as described in the privacy statement, *“The requested information [from surveys or contests] typically includes contact information (such as name and shipping address), and demographic information (such as zip code). Contact information will be shared with the contest or survey sponsors to notify the winners and award prizes or otherwise in accordance with the Terms and Conditions of each competition or survey.”* This means that information that is collected through surveys or contests is shared with third part sponsors. We see this as a way for Kazaa and Sharman Networks to leak private information about their users to third parties.

Kazaa Media Desktop includes search functionality. In the privacy statement it is revealed that a third party provides this search function. Therefore all search queries will be forwarded to Sharman Networks’ third parties, which could be anyone of at least Cydoor, DoubleClick and SaveNow. The privacy statement literally says *“If you choose to use the web search function [in Kazaa Media Desktop], we will provide the search keywords you enter and your country information to our web search provider. . .”*. By studying the privacy statement we were able to find out that Kazaa is bundled with *at least* Cydoor, DoubleClick and SaveNow. Of course it is fully possible that Kazaa bundles with even more software components: *“Sharman Networks licences technology from Cydoor and integrates WhenU’s SaveNow to power the advertising products on our website and in the KMD application. These applications are mandatory as they are integral to the use of the KMD software.”*. Here we are also informed that these components are mandatory and therefore can’t be opt out during installation.

Sharman Networks defines spyware by referring to Steve Gibson’s definition of spyware, see section 1.3.3. Here it is also stated that: *“No application included with your KMD installation, or KMD itself, collects personally identifiable information about users without their consent.”* What is extra interesting in this statement is the *without-their-consent* part. As discussed previously in this section Sharman Networks expects that users give their consent by participating in frequently held surveys or contents. Of course Sharman Networks also interpret that a users gives their consent by accepting the licence agreement tied to Kazaa Media Desktop. Even if this is awkwardly written and therefore hard

¹³These figures were accurate in April 2003.

¹⁴Kazaa privacy statement, <http://www.kazaa.com/us/privacy/privacy.htm>.

¹⁵Kazaa adware statement, <http://www.kazaa.com/us/privacy/adsupport.htm>.

¹⁶Kazaa spyware statement, <http://www.kazaa.com/us/privacy/spyware.htm>.

for users to understand.

2.1.5 Morpheus 2.0

Morpheus is developed by Streamcast Networks¹⁷ and has been downloaded more than 110 million times from Download.com¹⁸. Morpheus is based on the open Gnutella protocol. There was no privacy related information whatsoever on neither Streamcast Networks' nor Morpheus' web page. We believe proper companies develop privacy policies that are presented to their customers. Since Streamcast Networks fail in presenting such a policy we suspect them of at least carelessly administer their users personal information; or in worst case deliberately leak private information about their users to third parties.

Unfortunately we couldn't find any information on either Morpheus official web page nor on Streamcast Networks web page about what possible software Morpheus bundles with either.

2.1.6 eDonkey 2000 v35.16.61

eDonkey2000¹⁹ is developed by MetaMachine and has been downloaded more than 800 thousands of times from Download.com²⁰. We were not able to find any privacy related information on the official eDonkey2000 web page. When trying to connect eDonkey2000 to the P2P network we couldn't get any response. After repeating failing to connect we came to the following possible conclusions, either:

- We were not connecting to the right servers, although we tried the ones listed on the official web page.
- The whole eDonkey P2P network were closed down.
- The client was outdated, although we couldn't find any newer version.

Given this problem we had no other choice than to exclude eDonkey2000 from our investigation. Therefore eDonkey2000 will not be dealt with anymore in this thesis.

2.2 Laboratory environment

During both planning and execution of the investigation we had three main goals concerning the laboratory environment:

1. Preserving identical hardware and software configurations during all P2P tool investigations. A different tool was run on every computer, but the operating system configurations were identical. Also, the computers had exactly the same hardware and were connected to the same LAN.

¹⁷Streamcast Networks Inc., <http://www.streamcastnetworks.com> and <http://www.morpheus.com>.

¹⁸These figures were accurate in April 2003.

¹⁹eDonkey2000, <http://www.edonkey2000.com>.

²⁰These figures were accurate in April 2003.

2. Using default software configurations whenever possible, that including updates to device drivers and operating system.
3. Only use tools that were either freeware or shareware, i.e. anyone can use them free of charge, at least for limited time.

Our investigation system configuration was built upon common hardware and software components. Admittedly, a few programs may not be very common (package dumping library and analysis applications) and probably used only by hackers, software developers or people conducting investigations of the type we do. However, they most likely affect the system little or not at all. No custom-made software parts should exist in our application base. This also involves extreme or unusual configurations of both operating system and applications. The system should have default or, in a case where a choice must be made, standard user settings regarding software updates and security. A system with either no security potential (e.g. settings regarding cookies, scripts or possible malformed input) or one that is highly secured is undesired.

2.2.1 Hardware

The computers used in the investigation were Dell OptiPlex GX260 PC:s. Hardware configurations were identical on all the computers. Every detail such as BIOS configuration, PCI slot used for PCI cards, chipset and operating system were exact matches. The cloning system ensures identical hard drive geometry between all the computers, i.e. MBR (Master Boot Record), partition tables and FAT layouts are exactly the same on a binary level. At the time of installation of a specific P2P tool, any file in the file system was located on the exact same cylinder, track and sector on all computers.

Hardware configuration can be found in appendix, section 6.1.

2.2.2 Software

The P2P tools of interest for this investigation all run on the Win32 platform. Though P2P file sharing itself is not restricted to that platform, the adware and spyware most likely is. Moreover, the majority of users most likely do not use P2P tools on any other platform. As we have only tested the investigation tools in Windows 2000 Professional, that operating system version was the one of choice.

All available updates to Windows 2000 from the Windows Update web site were installed. Few operating system settings were changed but the general system configuration was reviewed. For accomplishing both the installation of patches and reviewing the default configuration an application called *Microsoft Baseline Security Analyzer*²¹ was used.

Exactly what software was included in our application base can be found in appendix, section 6.2.

²¹For further information, see <http://www.microsoft.com/technet/treeview/default.asp?url=/TechNet/Security/tools/tools/MBSAHome.ASP>.

2.2.3 Shared files

In order to make the P2P tool environment as real as possible, we prepared a collection of files to be shared by each P2P tool. These files were of several different types and very different sizes. Apart from the usual audio and video files, Windows programs were shared. For the purpose of sharing different types of files, a number of FreeBSD and OpenBSD files (mainly installation packages and scripts) were shared.

Type	Formats	Nr. of files	Size (MB)
Image	jpg	18	3
Audio	mp3, wav	19	56
Video	avi, mpeg, mov	215	1861
Document	pdf	2	0,1
Windows programs	exe, zip	105	442
Free/OpenBSD files	binary, text	778	396

Table 2.1: Description of shared files.

2.2.4 Network

All the computers were connected to the same switch at Blekinge Institute of Technology in Sweden. The switch operated on a subnet that was a separate (i.e. our investigation computers were the only ones running there at the time) 100 Mbit/s Ethernet[14] LAN and a part of SUNET (Swedish University Computer Network).

2.2.5 Cloning system

One of our main goals in the investigation was to provide our investigation computers with the same hardware and software configuration, one for each computer. For this purpose, we have used a cloning system developed in Seclab at BTH²². The use of this system gives our investigation a few advantages. The same P2P tool can be investigated multiple times and the results compared. As far as we can control it, the execution environment is the same for all the installations. If necessary, it would therefore be possible to compare results from the investigations of the different tools. By creating reinstallable system images, the investigation results can be reproduced. This means that installation, configuration and updating of the operating system and investigation tools is not time dependent, e.g. on what Windows patches are currently available. Another advantage of the cloning system is that system images can be created at any time during the investigation, making it possible to run any step any number of times. To summarise, there are two main uses of the cloning system:

1. Create a clone image of a selected system.
2. Recreate a previously created clone from an image.

²²Blekinge Tekniska Högskola, <http://www.bth.se>.

It is preferable to clone a system when it is shut down. There are two reasons for this.

1. An active (executing) operating system alters its state continuously. Its file system would differ between start and end of the image creation process [11]. This might result in a clone image that is a corrupt representation of the system.
2. Important information that resides only in the computers non-persistent memory and is never stored on hard disk will not be read during the clone image creation process and will therefore be lost.

So far, it is clear that the system to be cloned must be shut down. Storing its data on persistent or semi-persistent memory is the only way for a system to keep its state. In practise this means that what is stored on hard disk (or of course any other memory of the mentioned types, but only hard disks are used for reading and writing clone images) is everything and the only thing to include in the clone image creation process. What needs to be done is, simply put, to copy the contents of the hard disk on which the system is stored. However, it is important not to affect the system in any way. The issue here is **not to mount** any file system on the hard disk because that would possibly change the system state. Instead, the disk contents should be read directly as binary data from the raw device. The cloning system uses `dd`²³ for this purpose. The data is then compressed by a pipe to `gzip` and then stored to disk (note that this is any accessible hard disk other than the one read from). Because data format conversion is an option in `dd` and not a restriction, by giving no such commands no conversion is made. Of course, the successful creation of a perfectly functioning system clone shows that the cloning process is indeed working.

The Seclab cloning system is based on FreeBSD²⁴[7] Live. FreeBSD is a POSIX operating system and the Live version boots and resides entirely in RAM. It uses a read-only file system on the CD from which it was booted. The Live system is booted on the computer to be cloned and the hard disk contents is read, compressed and stored on a mounted NFS (Network File System) file system. Note again that the hard disk is only read from, not written to, and no file systems on it are mounted.

For a matter of discussion, we do acknowledge that commercial products with similar use exist. However, they are less suitable for our work than the Seclab system. The reasons for using the Seclab cloning system are:

- We have taken part in building the Seclab system and of course know how to use it.
- The Seclab cloning system gives very good control over the image creation process. We know little about other software.
- The Seclab cloning system makes it possible to verify the correctness of the clone images and has been successfully tested and run in the live environment the investigations were performed.

²³FreeBSD manual web pages, <http://www.freebsd.org/cgi/man.cgi?query=dd&apropos=0&sektion=0&manpath=FreeBSD+4.7-RELEASE&format=html>.

²⁴FreeBSD, <http://www.freebsd.org>.

- When using the Seclab cloning system we are able to ensure the reproducibility of the investigations because we know exactly what happens to the data.
- It is possible for us to adapt the Seclab cloning system to a changed environment (e.g. network or hardware configurations).
- The Seclab system is free (and this thesis is a zero budget project).

2.3 Method description

Our investigation method is based on state preservation. By preserving the state of a system together with complementing information (such as network traffic) it is later possible to retrieve a specific state for analysis. The main payback of this approach is that we avoid doing both data collection and data analysis simultaneously in real-time. Instead we are able to collect data only once. Data that later is used for analysis.

We decided to split the investigation into a number of clearly separated steps for the sake of better overview. We believe that it is easier to divide a problem into sub-problems that if possible are solved respectively.

1. Installation
2. 30 min run
3. 100 min run (including web surfing script and search queries)
4. Removal

During installation we used the default settings for all P2P tools, e.g. where to store the P2P tool on disk or what extra components to include. During the 100 min run we executed a web surfing script that automatically requested a number of web pages from the Internet. Also, during the 100 min run we used the P2P search functionality to execute 10 file searches. These search queries included the exact same keywords for all P2P tools, e.g. **Britney** or **Lord of the ring**. Both of these actions were attempts to activate spyware and no such attempts were made during the 30 minute run.

All steps listed above have common activities. Before performing the actual step (such as installing the P2P tool or starting it) a couple of files are created as to represent different important properties of the system before the step was performed. After the actual step has been performed, a new set of files is created as to represent the system properties again, tentatively with some changes. Measurements of network activity, i.e. firewall log and packet dump file, are of course created during the actual step. Below is a short list showing the procedure:

1. Create file system list 1
2. Create registry export 1
3. Clear the firewall log
4. Start dumping packets

5. Perform the actual step
6. Stop dumping packets
7. Save the firewall log to file
8. Create file system list 2
9. Create registry export 2
10. Make Ad-aware search and save log to file

These steps will recur during both data collection and analysis. An overview of the steps reveals that two steps deal with the installation and removal of our P2P tools. While the two remaining steps deal with the execution of our P2P tools and their possible ad/spyware. During the first step we will capture traces of any spyware that is installed together with some P2P tool. In step two and three we will capture any traffic from ad/spyware. Step four will reveal any persistent spyware that survives the removal part.

Windows application installation software often log what files are installed and what registry keys are set. Such log files could show if any known ad/spyware components are installed. However, it is not guaranteed that these logs really contain everything that was installed by the installation software. Therefore, we will disregard these logs.

2.3.1 Techniques and tools

For successfully preserving the states of the systems we needed tools that gathered information about the file system and Windows registry. We also needed tools that allowed us to dump network traffic into files. We only selected tools that were either freeware or shareware, i.e. anyone can use them free of charge, at least for limited time. The exception is of course the operating system, Windows 2000.

File system

It is very important that our method is capable of storing accurate information about what is stored in the file system during all steps of the investigation. The information needed should include what files and folders that exist in the file system at a given time. In addition to this we also need information about file size and about the latest modification time. We decided to install Cygwin²⁵ for solving this task. Cygwin is a Unix environment for Windows. To save the information needed about the file system we used the `ls`²⁶ command with arguments to operate recursively and to produce detailed output: `ls -lR . > file`

This command lists all files (except for filenames beginning with a dot) in the specified directory, and recurses through all sub-directories. For each file, `ls` prints the file properties. Instead of sending the output to the screen we redirect it to a text file. This means that, when started from the file system

²⁵Cygwin, <http://www.cygwin.com>.

²⁶FreeBSD manual web pages, <http://www.freebsd.org/cgi/man.cgi?query=ls&apropos=0&sektion=0&manpath=FreeBSD+4.7-RELEASE&format=html>.

root directory, the properties of all files in the file system will be output to a file. Listed properties are (in order) file mode (directory flag), file permissions, number of hard links, owner, size, date and time of last modification and name. Below is an excerpt from a list file:

```
-rwxr-xr-x    1 Administ None      192 Oct  7 14:14 AUTOEXEC.BAT
-rw-r--r--    1 Administ None      782 Oct  8 19:06 history.dat
-rw-r--r--    1 Administ None    8803 Oct  8 19:06 store.rdf
drwxr-xr-x    4 Administ None    4096 Dec  9 16:46 Vim
```

Since we created two new list files during each step we could look at the differences between these files and thereby find out what files had been modified. Such modifications could involve creation, removal or modification of files and directories. For modifications we could not derive how the content inside the affected files had changed. However we could see at what time the file was modified and the change in size (by comparing size before and size after the change). This way we determine if data had been added to or removed from a certain file. For performing these tasks we used a graphical diff tool called ExamDiff Pro²⁷.

By using this method we could find not only which files and directories were changed during a certain task, but also any components that survived removal. The last step of the investigation was to remove the P2P tool. Since we created a list file after removal we could check for differences between this list file and the list file created before installation. By doing so we could see if any files were added during installation or execution that survived the removal process.

Windows registry

In order to perform a later analysis of the Windows registry we needed to store this information into a file. For doing so we used a tool called `regedit` that is included in Windows 2000. By using this tool we could save all data stored in the registry into a file of our choice.

Scripted web browsing

Because adware and spyware is likely to have functionality related to web browsers, and specifically Internet Explorer, a list of URLs to visit when running the P2P tools was created. By visiting web pages, possibly any adware or spyware components could be activated. And inversely, we might discover no or little activity from such components if no browsing was made. Also, by only visiting these URLs, we could keep a coherent environment for the P2P tools and their components.

In order to make the executions of the different P2P tools as similar as possible, automation of the web browsing was needed. For this purpose, Cygwin was used. A simple but sufficient method for automating the web browsing was using a shell script. The list of URLs was modified to make bash, the Cygwin standard shell, run it. In short, bash started Internet Explorer which then visited the URLs specified in the script, waited a while and then closed Internet Explorer. Then, the same process was repeated for the next URL. For a more detailed description of the script, see appendix, section 6.3.

²⁷ExamDiff, <http://www.prestosoft.com/examdiff>.

The script was only executed during the 100-minute run of the P2P tools. The reason for not executing the script also during the 30-minute run was that we wanted to see if there were any differences between the two P2P tool runs. If a component was active without the script being executed, web surfing probably would not have any effect on the component.

Network packet data

Collecting network data in this investigation is very important because of its focus on detecting any transfer of information (which could lead to privacy intrusion as discussed in the introduction). During each step of the investigation, network packets are dumped to files using Win32 ports of Tcpdump and libpcap called WinDump and WinPcap. WinPcap is an architecture for packet capture and network analysis and gives WinDump the ability to sniff and analyse network packets. We used a filter to make WinDump filter the packet stream before dumping it to a file. The purpose of this filter was to let through IP-based protocol traffic (i.e. TCP and UDP) and packets with the local host as source or destination. The quite simple filter syntax is:

```
ip host 194.47.XXX.YYY
```

where XXX is the subnet and YYY is the host address. The filter makes windump save only packets with the specified address as source or destination.

The packets are stored in Tcpdump format, which is a binary format. It is therefore necessary to use some tool that can read dump files and display the data in a more human readable format. We used Ethereal²⁸ for accomplishing this. Ethereal makes it possible to easily browse and filter capture data. It can also show detailed information for each packet and a summary covering the entire dump file.

Firewall

The ZoneAlarm firewall is capable of capturing both inbound and outbound network traffic and also of connecting this data to a specific process. An excerpt from one of our firewall log files look like this:

```
PE,2002/12/19,22:21:48 +1:00 GMT,Windows Explorer,66.28.234....  
72:80,N/A  
FWIN,2002/12/19,22:21:50 +1:00 GMT,66.28.234.72:80,194.47.XYZ...  
YX:137,UDP  
FWIN,2002/12/19,22:29:08 +1:00 GMT,210.182.154.153:1025,194....  
47.XYZ.YX:137,UDP
```

These log files contain information about event, date, time, process name, source address and destination address. The event specifies if the firewall blocked in-/outbound traffic or if a process asked the user for permission to initiate an outbound connection.

²⁸Ethereal, <http://www.ethereal.com>.

Ad/spyware removal tool

Ad-aware²⁹ is a program that detects and removes both spy- and adware. We included Ad-aware in our investigation to help us locate interesting information among our data. We could then continue analysing this data in more depth by using our file list data, registry data or network data. We only used the identification functionality in Ad-aware, i.e. we didn't use it for removing any ad/spyware. The log files produced by Ad-aware included information about suspicious registry keys, running processes and files on the file system. All such entities, except registry keys, include information about name, size, version number, author and creation date. Information about registry keys included name and its corresponding value.

2.3.2 Analysis method

Analysis method of file system lists

We anticipated that this data should be quite large. Prior tests of the list file technique showed that file sizes could be estimated at about 2,5 MB. These figures concern a clean installation of our investigation system, i.e. no P2P tools or shared files included. The size of this data combined with the fact that file systems are under frequent change when running Windows made us realise that we needed to restrict the analysis of this data. We decided to analyse at least two P2P components thoroughly.

We should however still, for each P2P tool, investigate what components that survived removal. Furthermore should we use the data in the list files to verify all suspicious files found by Ad-aware.

Analysis method of registry data

The Windows registry includes very dynamic structures of data. The registry export files we created were about 8–11 MB in size and contained around 300 000 lines of text. Therefore we had to limit our examination of the registry. We simply could not check all changes that occurred since there should be a total of about $9 * 8 * 5 = 360$ MB of such data to analyse. The calculation assumes each registry file has a size of 9 MB and that there are five P2P tools to investigate. Each P2P tool investigation includes creation of 8 such files. Therefore we chose only to investigate the parts of the registry that handles what programs should be automatically started during system startup. Even though we didn't analyse all data we still collected it to have a choice of doing a later in depth analysis, if necessary.

For analysing the registry we used `regedit` and `RegCleaner`. `Regedit` is included in Windows 2000 and `RegCleaner` is a tool from www.jv16.org. Both tools are good for exploring and analysing the Windows registry.

Analysis method of network data

Because we define privacy intrusion not only as the gathering itself, but also as the transfer of gathered information, the data in focus in the analysis is the packet traffic.

²⁹Lavasoft, <http://www.lavasoftusa.com>.

The studied protocols are TCP, UDP and HTTP. Non-routable or network management protocols will not be included in the analysis. Examples of such protocols are DNS, STP, NetBIOS or SMB. Besides, the packet dump files only contain IP-based protocol traffic.

UDP does not include acknowledgement of received packets. Therefore it is less suitable than TCP for P2P file transfer. This is because UDP file transfer functionality would require more application-level logic because e.g. packet order is not guaranteed. Indeed, TCP is by far the most used protocol. We expect to find few or no UDP packets and any found packets, apart from the DNS traffic, are considered interesting and will be included in the analysis.

Gnutella and other P2P protocols are not used for transferring “payload” data, i.e. the files themselves, but only for setting up connections between peers to perform such transfers. Thus, P2P protocols will be disregarded.

The initial analysis includes counting the number of addresses and the number of packets to be analysed. The following display filter is applied in Ethereal:

```
(ip.proto == 0x11 || (tcp.flags.syn == 1 && tcp.flags.ack == 0)) &&
!(ip.src == 194.47.0.0/16 && ip.dst == 194.47.0.0/16) &&
!(ip.addr == 255.0.0.0/8)
```

The filter applies the following rules:

1. Display only UDP traffic or initiating TCP connections (first packet of the TCP handshake)
2. Do not display packets to local addresses
3. Do not display broadcast packets

The filter will, in a sense, normalise the packet dump files from the different P2P tools and make them a little easier to compare because anything other than relevant traffic is removed. Although traffic created by the web surfing script is present in the dump files, even after filtering, it is still possible to compare results between the P2P tools since the script was run when running every P2P tool. It is however not a good idea to compare such properties as packet count or packet count per protocol between investigation parts of the same P2P tool (note here that the script was active only during the 100 minute P2P tool run). On the other hand, those measurements can give a hint about the network activity.

After applying the filter, we extract the following information:

- Total packet count
- Filtered packet count (i.e. the number of packets that passed the filter)
- Number of external addresses found. External means that the IP address should not be 194.47.*.*.

- For TCP, UDP, and HTTP, extract:
 - Percentage of entire packet dump file
 - Number of packets
 - Data (bytes)

The number of external addresses is calculated in the following manner: A summary list of the filtered packets is saved to a text file with the following layout³⁰:

No.	Time	Source	Destination	Protocol	Info
1816	179.293781	194.47.XXX.XXX	80.128.53.XXX	TCP	1271 > 6346 [SYN] Seq=512...
1817	179.499238	194.47.XXX.XXX	63.226.177.XXX	TCP	1292 > 6347 [SYN] Seq=516...
1820	179.731278	194.47.XXX.XXX	216.228.184.XXX	TCP	1286 > 6347 [SYN] Seq=515...
1821	179.999077	194.47.XXX.XXX	157.238.69.XXX	TCP	1293 > 6346 [SYN] Seq=516...
1825	180.278142	194.47.XXX.XXX	24.127.0.XXX	TCP	1273 > 6346 [SYN] Seq=513...
1826	180.498765	194.47.XXX.XXX	216.165.22.XXX	TCP	1294 > 6346 [SYN] Seq=516...
:					

This text file is processed by the following (Unix) commands:

```
cat <file> | awk '{print $4;}' | sort | uniq -c | wc
```

First, the file is printed to the standard output by `cat`. The output is then sent to `awk`, which applies a filter and prints the fourth text block in every line, thus printing only the “Destination” column. The resulting list is then sent to `sort` and printed out sorted. Next, `uniq` removes duplicate lines and finally, `wc` calculates the number of lines and prints the result to the standard output. This number is equal to the number of unique addresses, and *plus one* for most of the files. The lab computer’s own IP address may be present in the list (because of the bi-directional communication output from Ethereal). This is checked by replacing `wc` with `grep 194.47.XXX` (where XXX is the local network address):

```
cat <file> | awk '{print $4;}' | sort | uniq -c | grep 194.47.XXX
```

Instead of counting words, `grep` searches for the specified input string, in this case the local network address. Thus, if the local network address is found, our result is the number of lines minus one.

For every packet dump file we then do the following:

1. Start by applying the display filter in Ethereal (if not already applied).
2. Extract the target IP address and trace it (`traceroute`). Trying to resolve its name (`nslookup`) is not necessary because it has already been attempted by Ethereal. If the name implies that the target address is

³⁰The host addresses have been replaced by XXX for privacy reasons. They are not important in this context.

dynamic (e.g. dialup or in some cases DHCP) or used by an private ISP customer, the session will not be further analysed because the target is considered a P2P network node. Also, typical university addresses like (.bth.se or .edu) are not considered interesting.

3. For TCP-based protocols, do a TCP Stream Analysis. If this analysis reveals nothing or it is evident that the target is a private computer or a normal P2P node, this target will not be further analysed.
4. Can you identify any clear text information that matches our definition of privacy intrusion in the TCP stream analysis?
5. Can you identify any encrypted or compressed data and possibly the encryption/compression type (e.g. SSL)?
6. Try to connect to the target address on ports 21, 80, 137 and 138 to find out target platform and server software. If a connection is accepted, check any output for these details.
7. Search the Internet (using the Google search engine) for information about the target address (IP address or name if available) or domain. Try to answer the following questions:
 - (a) Is the target a company (e.g. an ISP) or is it a private computer running P2P software?
 - (b) Is any information about the target and its activities available? If so, is the information consistent with our findings?

Since both the web surfing script and the search queries generated network traffic during the 100 min run we had to keep this in mind when analysing the network data. However this didn't cause much problem since we already had the names of all web servers that our web surfing script visited, see appendix section 6.3.

Analysis method of firewall log

When analysing the firewall log data we specifically looked for suspicious traffic. Because ZoneAlarm provides us with the ability to associate certain network traffic with a program we should be able to find any traffic originating from suspicious components, i.e. components usually not included in standard Windows systems. If any such traffic was found we should be able to determine a destination IP address. This IP address could be used to find interesting traffic in the packet dump files.

Analysis method of Ad-aware logs

There were no direct analysis of this data since it already was filtered by Ad-aware. However as described before we verified all files and components that Ad-aware found by using our own file list and registry data.

Chapter 3

Analysis

The Ad-aware and firewall log files and the file system list files are rather easy to analyse. They contain human readable text and are easy to filter and search through using any standard text editor. The network packet dump files, however, are more difficult to analyse. Because of their binary format, they must be analysed using a tool capable of parsing `tcpdump` format dump files. These dump files also contain a lot of information compared to e.g. the firewall logs. Analysing them is likely a time-consuming task.

In the following sections of this chapter, the analysis results for the different types of collected data are presented. Reflections regarding the data collecting and analysis methods are also given.

3.1 File system lists

A typical list file includes about 100 000 entries in one single file. There were a total of 147 MB of data captured this way. This information was divided in 40 list files that give an average size of 3.675 MB. As described before we should investigate all list files of at least two P2P tools thoroughly, we chose iMesh and Morpheus. We based this choice on the fact that Ad-aware found most suspicious components in iMesh. We therefore thought it would be interesting to investigate iMesh further. Based on the same line of reasoning we also choose to investigate Morpheus further.

We didn't manage to identify any new components for iMesh, other than the ones already found by Ad-aware. However we found out that the file `msbb.exe` (Web3000) wasn't installed during the installation of iMesh. Instead this file was installed during the 100 min execution, which is interesting see section 4. This file was located inside `C:\WINNT\System32\` together with a file called `msbb.dll`.

When analysing Morpheus we managed to identify one component that Ad-aware didn't find. The component was called `sentry.exe` and it was installed during installation. This component was located inside the `C:\WINNT\` directory together with a file called `sentry.ini`.

Apart from these findings the list files also showed that several components manage to survive removal of their host software. The list below specifies exactly what components this applied for.

Component	Host	Related file(s)
Web3000	iMesh	msbb.exe, msbb.dll
Cydoor	iMesh	cd_clint.dll, cd_htm.dll
eZula	iMesh	ezstub.exe, ezinstall.exe
SaveNow	BearShare	save.exe
WeatherCast	BearShare	weather.exe
Cydoor	Kazaa	cd_clint.dll
SaveNow	Kazaa	savenow.exe, savenow.db, savenow.htm
BrilliantDigital	Kazaa	bdeclean.exe, bdedetect1.dll
Gator	Morpheus	CMESys.exe, GStartup.lnk
DateManager	Morpheus	DateManager.exe
GMT	Morpheus	gmt.exe
PrecisionTime	Morpheus	PrecisionTime.exe
WurldMedia	Morpheus	mbho.dll

Table 3.1: Components that survived removal of their host software.

3.2 Firewall logs

Unfortunately ZoneAlarm did not capture all network connections that was made. We could not correlate all sessions captured by our network packet log to the entries in our firewall log. We don't know if this depends on any misconfiguration of the firewall, although it was carefully configured before use, or if there is any reduction of functionality since we used a freeware version of ZoneAlarm. The data collected by ZoneAlarm had a total size of 0,6 MB that was divided over 20 different files.

3.3 Registry data

We checked the registry for what components that should be started automatically after each step in the investigation. By automatically start each time the system restarted allowed these components to constantly run in the background performing their business. Table 3.2 lists these components together with their host software.

In table 3.2 `Eac_rvnd1` is of special interest since this component wasn't found by Ad-aware, see section 4 for more information.

3.4 Network data

The overall analysis described in section 2.3.2 was performed on all the packet dump files. The results are found in the tables below. Each P2P tool is represented by two tables. The "packet count" tables contain:

Component Name	Host Software
MediaLoads Installer	Kazaa
New.net Startup	Kazaa, iMesh
PromulGate	Kazaa
SaveNow	Kazaa, BearShare
WheatherCast	BearShare
Eac_rvndl	iMesh
eZmmod	iMesh
Hotbar	iMesh
Trickler	iMesh
Zenet	iMesh
Msbb	iMesh
CMESys	Morpheus
DateManager	Morpheus
GStartup	Morpheus
PrecisionTime	Morpheus
Sentry.exe	Morpheus

Table 3.2: Automatically activated components in system startup.

Total packet count	The total number of packets in the dump file.
Filtered packet count	The total number of packets in the dump file after applying the Ethereal display filter.
Nbr. of ext. addr.	The number of external addresses contacted. This number is calculated after applying the display filter.

The “protocol statistics” tables contain the same type of information for three different protocols, TCP, UDP and HTTP:

count	The number of packets of the specific protocol.
%	How large percentage the specific protocol traffic was of the entire traffic in the dump file.
size	The total size of all the packets (the entire packets, not only the payloads) in kB for the specific protocol. (1 kb equals 10^3 bytes.)

These statistics were calculated without any filter applied, i.e. from the entire packet dump file.

Note that traffic originating from the web surfing script is included in the packet dump files, and thus in the calculations. Filtering that traffic out is, however, not only a matter of removing all entries matching the web servers in the web surfing script. It is common practice to let web browsers load items such as ad banners from other web servers than where the actual web page resides. Therefore, loading a page specified in the web surfing script might result in communication with more than one server, making it difficult to tell if a packet originated from an ad/spyware component or the web surfing script.

Step	Total packet count	Filtered packet count	Nbr. of ext. addr.
Installation	115	5	3
30 min. run	25821	738	179
100 min. run	178574	1099	178
Removal	0	0	0

Table 3.3: BearShare packet counts.

Step	TCP			UDP			HTTP		
	count	%	size	count	%	size	count	%	size
Installation	52	45,2	6	54	47,0	9	12	10,4	3
30 min. run	25519	98,8	11325	302	1,2	49	5821	22,5	5712
100 min. run	178066	99,7	77661	501	0,2	76	5737	3,2	5014
Removal	0	0	0	0	0	0	0	0	0

Table 3.4: BearShare protocol statistics.

Step	Total packet count	Filtered packet count	Nbr. of ext. addr.
Installation	5952	148	80
30 min. run	15965	648	81
100 min. run	27636	1022	127
Removal	0	0	0

Table 3.5: iMesh packet counts.

Step	TCP			UDP			HTTP		
	count	%	size	count	%	size	count	%	size
Installation	5682	95,5	4594	184	3,1	23	3467	58,6	4464
30 min. run	15662	98,1	9356	264	1,7	43	5635	35,3	5362
100 min. run	27121	98,1	16541	466	1,7	79	7545	27,3	6945
Removal	0	0	0	0	0	0	0	0	0

Table 3.6: iMesh protocol statistics.

Step	Total packet count	Filtered packet count	Nbr. of ext. addr.
Installation	9555	165	74
30 min. run	17414	624	73
100 min. run	21029	833	90
Removal	0	0	0

Table 3.7: Kazaa packet counts.

Step	TCP			UDP			HTTP		
	count	%	size	count	%	size	count	%	size
Installation	9328	97,6	8611	175	1,8	20	2485	26,0	3525
30 min. run	17169	98,6	10851	244	1,4	40	5258	30,2	4966
100 min. run	20689	98,4	12753	334	1,6	59	5742	27,3	4923
Removal	0	0	0	0	0	0	0	0	0

Table 3.8: Kazaa protocol statistics.

Step	Total packet count	Filtered packet count	Nbr. of ext. addr.
Installation	11555	15	9
30 min. run	67469	725	102
100 min. run	51666	8368	2576
Removal	11	1	1

Table 3.9: Morpheus packet counts.

Step	TCP			UDP			HTTP		
	count	%	size	count	%	size	count	%	size
Installation	11519	99,7	11062	36	0,3	5	7575	65,6	10848
30 min. run	67178	99,6	31138	276	0,4	40	3993	5,9	3716
100 min. run	51006	98,7	15653	608	1,2	84	6098	11,8	5251
Removal	7	63,6	0,9	4	36,3	0,5	2	18,1	0,6

Table 3.10: Morpheus protocol statistics.

In the tables above, no data is available for the removal step of BearShare, iMesh or Kazaa. When examining the packet dump files we discovered that they contained no packets. This could have two explanations. Either the filter syntax was incorrect or there was simply no network traffic to or from the computer during removal. After performing the removal step twice for the three tools, it was affirmed that the filter syntax was correct. Thus, no packets were transmitted or received during removal. It should be noted that removal of any of the P2P tools rarely took more than one minute. The lack of packets is therefore reasonable.

Only the dump files from installation and 30 minute run of BearShare were further analysed, but not completely. When analysing these two files it became clear that our analysis method for packet dumps was not very good. The discovered shortcomings are described next.

Although the amounts of packets to analyse after applying the display filter were about 3% of the original¹, the amount of data is still very large.

As described previously in this section, the web traffic created by the web surfing script appears in the packet dump files. Again, it was hard to tell exactly what traffic the script created. For this to be possible, the script should have been executed on a computer with no other networking application (P2P tool, web browser) running and all traffic collected by windump. An analysis in Ethereal would have produced a list of what traffic to exclude from the P2P tool investigation.

Furthermore, we believe that should all the packet dump files have been analysed, it would not have resulted in any well-founded conclusions. When dealing with the amount of data and performing the degree of manual work originally proposed by our method, it is possible that some data will be missed.

Finally, we found that not collecting timestamped socket information (which sockets were opened, when they were opened and closed, and by which process) made the network data analysis time consuming and ineffective. This is because the data cannot be associated to any processes. The lab computers were

¹This is $\frac{\text{filt. packet count}}{\text{tot. packet count}}$. The median value is 3%, and 75% of the values are within the interval [1..4].

running something about 15–25 processes, where the majority were operating system specific (i.e. Windows 2000 services), at any point in time. Any traffic originating from other processes than the ad/spyware components should be removed. Again, the necessary information for this to be possible was not collected.

It should now be clear to the reader that no conclusions regarding the first two questions of the problem formulation (i.e. what components were installed and do they intrude on user privacy) could be made. However, this part of the analysis provided a good evaluation of the investigation method. After performing this analysis, it became clear that the method must be modified and where it is lacking. A proposal for solving these problems can be found in section 4.6.

3.5 Ad-aware logs

Ad-aware found a number of suspicious components. A list of the components identified by Ad-aware is showed in table 3.11. In addition to these components we manage to find two more components, `Sentry.exe` and `Eac_rvnd1` by analysing the data from our investigation.

3.6 Identified components

In the Ad-aware log files we found the components displayed in the table below together with a classification. The classification of every component is based on different Internet resources. The following web sites were used:

- SANS Reading Room, <http://www.sans.org/rr>
- Securityfocus and Bugtraq, <http://www.securityfocus.com>
- Counterexploitation, <http://www.cexx.org/adware.htm>
- Securitytracker, <http://www.securitytracker.com>
- Lavasoft Support Forums, <http://www.lavasoftsupport.com>

The table below table presents all components found by Ad-aware:

Component Name	Host Software	Adware	Spyware
B3D Projector	Kazaa	x	— ¹
CommonName	iMesh	x	x
Cydoor	iMesh, Kazaa, Morpheus	x	x
DownloadWare	Kazaa	x ²	—
eZula	iMesh	x	—
FlashTrack	iMesh	x	x
Gator	iMesh, Morpheus	x	x
Hotbar	iMesh	x	x
NewDotNet	iMesh, Kazaa	—	—
Network Essentials	Kazaa	x	x
PromulGate Universal	Kazaa	x	—
SaveNow	BearShare, Kazaa	x	x
WeatherBug	BearShare	—	—
Web3000	iMesh	x	x
WurldMedia	Morpheus	—	x

Table 3.11: Classification of identified components.

- (1) Technically, not spyware. However, the component authors prepare to include the users in distributed computation projects without their permission.
(2) Technically, not adware. However, it downloads and installs adware.

Due to the scarcity of reliable information it is hard to say exactly what business the components in table 3.11 carry out. For classifying these components as adware and/or spyware we used information from the sources listed above. In addition to these components a number of sub-components were also found. Table 3.12 presents a list of these together with which components they are part of.

Component Name	Part of . . .
CMESys	Gator
Date Manager	Gator
GStartup	Gator
MediaLoads Installer	DownloadWare
Precision Time	Gator
Trickler	Gator
Zenet	CommonName

Table 3.12: Identified sub-components.

Chapter 4

Discussion

This paper has dealt with the problem domain by working with the three questions put forward in the problem formulation. A lot of people on the Internet are discussing different issues in the adware and spyware domain¹. However little academic research has been directed towards this area. Though the investigation presented in this report is of a technical nature, the problem domain is tightly associated with non-technical issues such as privacy. Our treatment of privacy has been from a technical perspective.

Though privacy is not at all trivial in the adware and spyware context, there are other important issues related to those technologies, for example information security. More than one of the bundled ad/spyware components have the ability to download and run software from servers in some way affiliated with the component author. Apart from consuming resources (CPU time, network bandwidth and disk space), this functionality poses a big threat to the P2P tool users. They can only hope that destructive viruses, trojans or worms are not spread in large networks like the Kazaa network. A rapidly spreading worm could not only fill its victim P2P network but possibly also cause extreme loads in larger networks, e.g. ISP nets. Also, consider what could happen if 10% of the P2P nodes were infected by a DDoS (Distributed Denial of Service) trojan, and the network contains 10 million nodes. A large-scale attack could be performed with nobody to hold responsible³.

Furthermore, because the described download-and-run functionality in essence means that the users' CPU time is sold by the downloading component's author, the users will most likely have no control over what software is actually running in their systems. It would probably not be too much trouble, not including legal ones, to sell all this CPU time to the highest bidder no matter the purpose, be it weapons research, genome research or crypto-cracking. Distributed computation projects have existed several years and proven successful².

¹Some examples are <http://www.spyware.co.uk>, <http://www.adware.info> or <http://www.spywareonline.org>.

²SETI@home, <http://setiathome.berkeley.edu> and [distributed.net](http://www.distributed.net), <http://www.distributed.net>.

4.1 What kind of components are a couple of well known peer-to-peer tools bundled with?

Almost all of the P2P tools contain adware and spyware, see table 3.11. Some of the found components are classified as both adware and spyware. One P2P tool, ICQ, contained no adware or spyware components. We found two suspicious components that were not detected by Ad-aware, **Sentry.exe** and **Eac_Rvnd1**. Out of these two components at least **Sentry.exe** should be classified as spyware. Descriptions of these two components are presented below.

4.1.1 Sentry.exe in Morpheus

Sentry.exe was found when investigating the list file data from Morpheus. This file appeared in our list files after the installation of Morpheus. From Cexx.org^[2] we found out that a company called IPinsight³ develops this software. On Cexx.org we found a description stating that **Sentry.exe**: *“Provides Web sites with demographic and geographic information about you (the company brags that it can determine what city you live in to 90% accuracy), along with connection-speed and other data.”* IPinsight presents a privacy policy on their web page⁴ that validates the information presented on Cexx.org.

In this privacy policy they also describe how they manage to locate what city a person resides in with such high accuracy. It turns out that the **Sentry.exe** software eavesdrop on web forms that are submitted by a user. Or as stated in the privacy policy: *“IPinsight estimates neighbourhood-level geography of the user based upon postal code. The IPinsight Software monitors the users’ form submissions for the presence of street name and postal code name value pairs. On a periodic basis these street names and postal code values are communicated to the IPinsight server, . . .”* By using this sneaky method IPinsight also collects information regarding users age, gender, birthday and birth year. However this is not all. Further down the privacy policy IPinsight acknowledge that they, apart from the information just mentioned, also collects *“information about the computer’s hardware configuration, such as the amount of free space on your hard drive, and information about the computer’s software configuration, such as the name and version of the operating system.”* Exactly what this information includes is impossible to conclude, it could be anything from lists of all files located the computer to what programs and versions that are installed on a computer.

The information collected by IPinsight is of private nature. This is actually one of the worst concrete examples of spyware we have seen. IPinsight is probably using the information gathered by **Sentry.exe** to create very sophisticated user profiles that are sold to third parties. What adds an extra amount of excitement to this finding is that Ad-aware failed in locating this file.

We can only speculate in the reason for this failure⁵. However, what is clear is that IPinsight uses **Sentry.exe** to collect information about users that are of private nature. There is no doubt about this since IPinsight describes it

³IPinsight, <http://www.ipinsight.com>.

⁴IPinsight consumer policies, <http://www.ipinsight.com/consumer.asp>.

⁵We did find one clue, a discussion board at <http://www.computing.net/windowsme/wwwboard/forum/29443.html>. However, the there information is too vague for being presented in this thesis.

quite accurately on their web site. There could be little or no opposition when categorising `Sentry.exe` as spyware. Some might mean that since IPinsight describes what `Sentry.exe` does it should not be categorised as spyware. There are two reasons why this is not the case. First, the information presented on IPinsight's web page should be presented to users in advance of installation. Second, the information presented to users should be a complete description stating exactly what `Sentry.exe` does and exactly what information it gathers. Both of these points correspond with the Steve Gibson definition of spyware, see section 1.3.3.

4.1.2 Eac_Rvndl in iMesh

When analysing the Windows registry data collected from iMesh we found a suspicious registry key called `Eac_Rvndl`. This key pointed to `C:\Documents and Settings\Administrator\LocalSettings\Temp\EACDownload\ANTIVI~1.EXE`. First we made sure this key was not in our system prior installation of iMesh. The next step were to validate the existence of `ANTIVI~1.EXE` with our list file data. We found `ANTIVI~1.EXE` in the list file created after installation of iMesh. After performing a diff between this file and the list file created prior the installation of iMesh we could conclude that this file was not in the file system before installing iMesh. Therefore we could conclude that `ANTIVI~1.EXE` was bundled with iMesh, in one way or the other.

After locating this suspicious file we tried to find why it was located on our system, i.e. what business it had there. Unfortunately this proved to be quite difficult. We managed to find out that `ANTIVI~1.EXE` might be a trojan backdoor called `Troj/Canary`. This information came from a web page called Sophos virus analysis⁶. Even though the source is quite vague we decided to use it since it was the only information we could find.

Sophos described the trojan as *“Troj/Canary is a backdoor Trojan client program that can be used to download and run files from a remote server.”*. If this turns out to be true it is pretty serious. Further on Sophos states that the file should be stored inside a directory called `C:\<temp>\EACDownload\`. This part matched `ANTIVI~1.EXE` perfectly. However, Sophos stated that the trojan file should be called either `can_temp.exe` or `can_temp.dat` while our file was called `ANTIVI~1.EXE`. However it is quite reasonable to assume that this file was renamed to confuse anyone looking for suspicious files. Therefore it might be a good idea to rename it `ANTIVI~1.EXE` just to mislead someone into thinking this is an antiviral file of some sort. We had reached the end of this road so we decided to stop doing any further analysis of this file.

4.1.3 Further thoughts

One conclusion we made from the two findings presented above was that Ad-aware doesn't find all spy/adware circulating the Internet. We discovered two components, where at least one was spyware, that Ad-aware did not find. After performing this investigation we understand that the team behind Ad-aware must find it difficult to keep up with the ever changing spyware and host software flora.

⁶Troj/Canary information, <http://www.sophos.com/virusinfo/analyses/trojcanary.html>.

4.2 Do these activities intrude on user privacy?

It is not possible for us to give a good answer to this question. Although other researchers classify components as spyware, and some of them describe the components' activities in depth, we lack the necessary confirmation regarding this property from our own investigation. However, several apparently well-made investigations presented on the Internet reach the same conclusions, which are also presented by respected organisations like SANS. Therefore, with a varying degree of certainty, we say that at least `Sentry.exe` is definitely spyware.

Sometimes it is obvious if a component fits the description of spyware, but generally it is not a trivial decision. In order to make an informed evaluation, it is necessary to perform quite technical investigations that require knowledge in computer communication and preferably also in how the Win32 platform works. Furthermore, even if two separate researchers get the same results, their opinion on categorising a component as either adware or spyware, or possibly both, may differ.

Moreover, the adware and spyware domain is rapidly changing. Anti-adware and anti-spyware tools and firewall blacklists make ad/spyware authors update or modify their software, and creating new ways of doing business. Placing ad banners on top of existing web page banners is just one example. Software bundles may be added or removed between versions, or existing bundles might be updated. It is therefore not certain that a P2P tool that used to contain e.g. spyware still does. Maybe the principle of "innocent until proven otherwise" should be applied. It certainly should if any proactive measures are going to be possible because loose accusations against the software companies will not solve the spyware problem.

4.3 How can an investigation method that discovers privacy intrusive components be constructed?

The method used in this investigation was based on the idea that as many relevant properties of the system as possible should be measured. In this context, relevant properties are those that are possible for any software to affect. Examples of such properties are file system contents and changes in content, network communication and Windows registry settings. Properties that would be less interesting to monitor are e.g. mean CPU load, overall memory usage and disk I/O. These properties can merely imply activity but not give any detailed information about installed spyware and its activities.

When looking upon the two parts of the investigation, data collection and data analysis, we see that the method in general was simple and sufficient. However, during analysis it became obvious that more data should have been collected. As it was constructed, the investigation method failed to provide means for associating processes to network data. A tool with the ability to perform that kind of association could greatly improve the extraction of interesting network data from the gathered. A well-performed investigation would mean several days of continuous packet dumping, creating extremely large amounts of data to analyse. Without good tool support such investigations would clearly

not be feasible. Moreover, an open source, well-documented tool, could possibly help users and anti-spyware organisations keep up with spyware authors.

4.4 Planning of analysis vs. analysis results

During the initial planning of the investigation we assumed that we would be able to analyse all collected data. However, due to the massive amount of gathered data, over 620 MB, that really was captured we had to restrict our analysis. We simply couldn't analyse all data. The following data was actually analysed:

Type	Description	P2P tool
File list data	Complete analysis	Morpheus & iMesh
"	Removal survival	All tools
"	Verify Ad-aware	All tools
Registry data	Autostart functionality	All tools
Network data	Network statistics	All tools
"	Data analysis	BearShare ¹

Table 4.1: Analysed data.

(1) Only data captured during install and 30 min run was analysed.

Among the various types of data that we captured, the list file data proved to be very useful⁷. This data allowed us to verify other types of data such as registry and Ad-aware data. No data that we captured was really useless. However, the network packet data was really hard to analyse due to the lack of proper analysing tools.

4.5 Correctness of collected data

Our list file data proved to be correct and accurate. We did not verify this data since we knew that it was accurate from prior tests of the method⁸. The data collected by Ad-aware also proved to be correct. We verified all files that Ad-aware identified with our list file data and no strange results were found. There were no problems regarding our registry data either. However, we had some problem with the log files that ZoneAlarm produced. We could not correlate all captured network data. The reason for this is unknown and we think it is quite strange. The network packet dump also had problems. It seems that we stopped packet dumping too early on some occasions. Therefore the end of certain TCP sessions got excluded from the packet dump. This proved to be a problem when trying to recreate some of the sessions (using the TCP stream analysis in Ethereal) since the end of these were missing.

⁷Of course, the data gathered by Ad-aware was useful too.

⁸Prior our investigation we allowed about 20 students to perform a similar test as the one described in this paper. The main benefit from this was to test our techniques such as list file creation and the web surfing script.

4.6 Analysis tool proposal

To achieve this functionality, certain actions have to be taken. Current tools for the Win32 platform provide functionality to:

- Monitor and log processes and their properties in real time. Examples of tools that perform this are:
PMon, <http://www.sysinternals.com/ntw2k/freeware/pmon.shtml>
ProcessExplorer (no logging), <http://www.sysinternals.com/ntw2k/freeware/procexp.shtml>
- Monitor transport level network activity and log created sockets, their states, and which process created them, in real time. Examples of tools that perform this are:
TCPView (no logging), <http://www.sysinternals.com/ntw2k/source/tcpview.shtml>
TCPView Professional Edition (has logging functionality, commercial), <http://www.winternals.com/products/generalutilities/tcpviewpro.asp>
Netstat (no process information, not real time), part of Windows 2000/XP
- Monitor, and dump, network packets. Also, parse and filter packet dump files. Examples of tools that perform this are:
WinDump, <http://windump.polito.it>
Ethereal, <http://www.ethereal.com>

All collected data has to be timestamped because the tools monitor real time activity. If, for example, the same process executes several times it must be possible to make each execution unique. Preferably, process and socket state changes should also be timestamped.

The part of the output from tools described above that we are interested in are the log files. These logs are all text files, apart from the packet dump files that are binary files. They all contain measurements of different system properties. However, the files have in pairs common measured properties. Consider the following log file examples:

```
[timestamp, process_name, pid, proto, local_addr:port,  
remote_addr:port, state]
```

This could be the columns of a transport level network monitor log file.

```
[timestamp, src_addr:port, dst_addr:port, proto, data]
```

This could be typical columns in a network packet dump file.

```
[timestamp, process_name, pid, registry_op]
```

This could be the columns of a registry monitor log file.

```
[timestamp, process_name, pid, file_system_op]
```

This could be the columns of a file system monitor log file.

A brief description of the terminology above:

<code>timestamp</code>	The system time when the action occurred.
<code>pid</code>	Process ID number. Might be redundant if only <code>process_name</code> is used.
<code>proto</code>	Communication protocol, e.g. TCP, UDP, HTTP etc.
<code>local_addr</code>	The local IP address or host name. Same as <code>src_addr</code> .
<code>remote_addr</code>	The remote IP address or host name. Same as <code>dest_addr</code> .
<code>port</code>	TCP or UDP port.
<code>state</code>	Process or socket state.
<code>registry_op</code>	Windows registry operations (e.g. <code>Request</code> , <code>Path</code> , <code>Result</code>).
<code>file_system_op</code>	Windows file system operations (e.g. <code>Request</code> , <code>Path</code> , <code>Result</code>).

Assuming we start by looking at the network monitor log file, we get the combination of `timestamp`, `pid` and `local_addr:port`. Then we move on to the packet dump file and get the combination of `timestamp`, `src_addr:port`, `dst_addr:port` and `data`. By combining this information, preferably using a database (e.g. an SQL database or Berkeley DB), per-process network data filtering can be performed. Furthermore, including registry and file system activity in the same way would give the user ability to extract complete coverage of a process' activity during a single or multiple executions (i.e. when the process was in running state). The search abilities would, of course, depend on how the database was designed.

The hard part here is to collect the data because it has to be done on the Win32 platform (the Win32 API is much more complex than those of POSIX systems[5][10]). Once log files exist it is a matter of filtering the data and inserting it into a database to make it searchable. The entire analysis could be performed perfectly well on any other platform, preferably in a Unix system because of the extensive tool support. We do acknowledge that a number of tools have been ported to Win32 (e.g. in the Cygwin environment), but do not see any purpose in substituting a real Unix environment.

Chapter 5

Conclusion

5.1 Conclusion

We have in this paper presented a method for investigating spyware that are bundled with any type of software. By using this method we have investigated five of the most popular P2P tools on the Internet. Four out of these five tools were bundled with adware and/or spyware. To assist us in identifying spyware we used one of the most popular anti spyware tool, Ad-aware. During our investigation we found a number of suspicious components, among these two were overlooked by Ad-aware. Out of these two one was definitely spyware. The investigation we performed allowed us to test the investigation method in practice. We came to the conclusion that the method was a stable foundation on which future investigations could be developed. However, there were a number of problems concerning the tools used in some of the more central steps of the investigation. These problems are thoroughly discussed and from this discussion came also a number of improvements. Incorporating these improvements of the method should overcome these problems which allows more effective investigation to take place in the future.

5.2 Future work

Method improvement proposal

We propose that further efforts should be focused towards creating a new analysis tool. Its purpose would be to enable per-process tracking of network packet data. This is interesting because defining which processes to examine (i.e. finding process names) in a Win32 system is both easy and a good place to start an analysis. A detailed description of such a tool can be found in section 4.6.

Thoroughly investigate P2P tools

Further analysis of iMesh, Morpheus and Kazaa is probably an interesting prolongation of our investigation. By implementing the improvements described above such an investigation should be able to find and analyse the network traffic sent between spyware and their central servers. If such network traffic could

be linked to a certain process there could be no disagreements when classifying such components as spyware.

Licence agreement

We propose that someone with proper knowledge about licence agreements investigate exactly what are stated in licence agreements. It should be of special interest to see such an investigation combined with an technical investigation, like the one we presented in this paper.

Bibliography

- [1] Ross Anderson, *“Security Engineering - A Guide to Building Dependable Distributed Systems”*, John Wiley & Sons (2001)
- [2] Counterexploitation <http://www.cexx.org>
- [3] Zeinalipour-Yazti Demetrios. *“Exploiting the Security Weaknesses of the Gnutella Protocol”*, University of California (2002)
- [4] Dieter Gollmann, *“Computer Security”*, John Wiley & Sons (1999)
- [5] David G. Korn. *“Porting UNIX* to Windows NT”*, USENIX Conference Proceedings (1997)
- [6] Bo Leuf, *“Peer-to-Peer - Collaboration and Sharing over the Internet”*, Addison Wesley (2002)
- [7] Marshall Kirk McKusick et. al. *“The Design and Implementations of the 4.4BSD Operating System”*, Addison Wesley (1998)
- [8] Andy Oram, *“Peer-To-Peer - Harnessing the Power of Disruptive Technologies”*, O’Reilly (2001)
- [9] Bruce Schneier *“Secret & Lies - Digital Security in a Networked World”*, John Wiley & Sons (2000)
- [10] Charles P. Shelton et. al. *“Robustness Testing of the Microsoft Win32 API”*, IEEE International Conference on Dependable Systems and Networks (2000)
- [11] A. Silberschatz, P. Baer Galvin , G. Gagne *“Operating System Concepts 6th Edition”*, John Wiley & Sons (2002)
- [12] William Stallings, *“Local & Metropolitan Area Networks, 6th Edition”*, Prentice Hall (2000)
- [13] W. Richard Stevens, *“TCP/IP Illustrated Volume 1 - The Protocols”*, Addison Wesley (2000)
- [14] Andrew S. Tanenbaum, *“Computer Networks, 3rd Edition”*, Prentice Hall (1996)

Chapter 6

Appendices

6.1 Hardware specification

All computers were Dell OptiPlex GX260 and they included the following hardware.

- CPU: Inter Pentium 4 1800 MHz
- Chipset: Inter 845G
- RAM: 256 MB SDRAM
- HDD: Seagate Barracuda 80 GB
- HDD Controller: Promise Ultra 100 TX/2
- Graphics card: Intel onboard
- CD: Samsung CD-ROM 48x
- NIC: 3Com EtherLink 10/100 Mbps

6.2 Application base specification

- Microsoft Windows 2000 Pro. 5.00.2195
- Microsoft Windows 2000 Service Pack 3
- Microsoft Windows 2000 hotfixes (all pre SP 4)
 - Q322842
 - Q322913
 - Q323172
 - Q324096
 - Q324380
 - Q326830
 - Q326886
 - Q327269
 - Q328145
 - Q328523
 - Q329115
 - Q329834
- Cygwin 1.3.12
- Windows Commander 5.0
- WinRAR 3.10.65
- Microsoft Baseline Security Analyzer
- OpenOffice.org 1.0.1
- Exam Diff Pro 3.0
- gVim 6.1
- Ad-aware 5.83.2930
- jv16 PowerTools 1.1
- ZoneAlarm 3.1.395
- ZoneLog 1.15
- VisualZone 5.7.0.2909
- Eraser 5.5.4.1
- WinPcap 3.0
- Ethereal 0.9.8
- History Inspector 5.0.5
- History Reader 5.2.0
- O&O Defrag 2000 Freeware Edition
- Installation files for our P2P tools

6.3 Web surfing script

The Cygwin command shell is started and the working directory is changed to the Internet Explorer program folder by executing the following commands:

```
cd c:  
cd "Program Files"  
cd "Internet Explorer"
```

Then the web surfing script `wscript.sh` is executed by the command `./wscript.sh`. The script head contains a variable definition: `t` is the number of seconds for every Internet Explorer process to stay alive. The next line contains the first actual web browsing activity. This, and all following lines have the same composition:

```
./iexplore <URL> & (a=$! ; sleep $t ; kill $a)
```

In the first part of the command, we see the execution of `iexplore.exe` (Internet Explorer) with an URL as command line argument. Then follows a shell command sequence:

```
(a=$! ; sleep $t ; kill $a)  
  
a=$!      assigns the PID of the started iexplore.exe to the  
          variable a.  
sleep $ t puts the shell to sleep for t seconds.  
kill $a   kills the process with the PID a (which is the  
          iexplore.exe started 60 seconds ago).
```

These three commands are executed sequentially by the shell (which is bash), but concurrently with `iexplore.exe`.

The list below shows all the URLs that were (sequentially) passed as arguments to `iexplore.exe` processes. Note that several levels of some web sites are visited by following links from the start page (i.e. `index.html`) and its sub-pages, thus simulating human browsing.

```
http://www.amazon.com
```

```
http://www.amazon.com/exec/obidos/ilm-redirect/102-7127654-0504145?append-uid=no&path=http://www.amazon.com/exec/obidos/tg/browse/-/909656/ref=ilm_rc_414422/102-7127654-0504145&message=414422,nov-14-2002-todays-deals-gateway-top-center,2&hmac=9DEBA4BCFE2370FD7200C0ED8101315A97792F4B
```

```
http://www.amazon.com/exec/obidos/tg/feature/-/414395/ref=td_b_as_dvdplay/102-7127654-0504145
```

```
http://www.amazon.com/exec/obidos/tg/detail/-/B0000659UN/qid=br=1-3/ref=br_lf_gw_3//102-7127654-0504145?v=glance
```

```
http://www.amazon.com/exec/obidos/tg/detail/-/B0000659UN/ref=pm_dp_ln_e_5/102-7127654-0504145?v=glance&s=electronics&vi=accessories&me=ATVPDKIKXODER
```

```
http://www.amazon.com/exec/obidos/tg/detail/http://www.gamespy.com/previews/december02/csczpc/index3.shtml-/B00007EPJ6/ref=segway_tn_left/102-7127654-0504145
```

```
http://www.amazon.com/exec/obidos/tg/detail/-/B00007EPJ6//102-7127654-0504145?v=glance&s=el
```

electronics&vi=pictures#more-pictures

http://www.amazon.com/exec/obidos/tg/browse/-/1065836/ref=e_hp_tn_6/102-7127654-0504145

<http://www.bestbuy.com>

<http://www.bestbuy.com/movies/default.asp?m=270>

<http://www.bestbuy.com/movies/productInfo.asp?e=11184948&m=270&cat=&scat=>

<http://www.bestbuy.com/movies/bestsellers.asp?b=0&st=3&m=270&cat=1756>

<http://www.bestbuy.com/movies/productinfo.asp?e=11187874&m=270&cat=1756&scat=>

<http://www.gamespy.com>

<http://www.gamespy.com/previews/december02/zeldagcn/>

<http://www.gamespy.com/previews/december02/csczpc/>

<http://www.gamespy.com/previews/december02/csczpc/screenshots.shtml>

<http://www.gamespy.com/previews/december02/csczpc/index2.shtml>

<http://www.gamespy.com/previews/december02/csczpc/index3.shtml>

<http://www.cdnw.com>

<http://www.cdnwabc.com/cgi-bin/mserver/pagename=/RP/CDN/FIND/discography.html/ArtistID=SHAKIRA/from=re6:x:cdn:120302ac1>

<http://www.cdnwabc.com/cgi-bin/mserver/SID=318829227/pagename=/RP/CDN/FIND/album.html/artistid=SHAKIRA/itemid=1461851>

http://www.cdnwabc.com/cgi-bin/mserver/SID=318829227/pagename=/RP/TOP100/index.html/clickID=tn_sttp_img

http://www.cdnwabc.com/cgi-bin/mserver/SID=318829227/pagename=/RP/AO_NR/new_rel.html/clickID=tn_stnw_img

<http://astalavista.box.sk>

<http://astalavista.box.sk/cgi-bin/robot?srch=windows+2000+crack>

<http://neworder.box.sk/>

<http://neworder.box.sk/codebox.links.php?sid=&&key=hack-nt>

<http://neworder.box.sk/codebox.links.php?sid=&&key=explwin>

http://neworder.box.sk/codebox.click.php?id=25973&url=http%3A%2F%2Fnewdata.box.sk%2Fmike%2Fcrashie_form.htm

<http://neworder.box.sk/codebox.links.php?&key=dvd>

<http://www.cnn.com>

<http://www.cnn.com/cnnsi/>

http://sportsillustrated.cnn.com/inside_game/don_banks/news/2002/12/13/banks_insider

<http://sportsillustrated.cnn.com/basketball/>

http://sportsillustrated.cnn.com/basketball/news/2002/12/13/hornets_lakers_ap

[http://ar.atwola.com/link/93118740/html?badsc=BOL8Lb517KLCbWPZuUdyKh7Ha9iin6hODRQqYdzwVPhlED2xjUI3wHjHgn6yEsECYQYqnPZ7XGzoekHqr5Q4YFLHYrud-Q3ZBR4_N2nJJej3Y\\$](http://ar.atwola.com/link/93118740/html?badsc=BOL8Lb517KLCbWPZuUdyKh7Ha9iin6hODRQqYdzwVPhlED2xjUI3wHjHgn6yEsECYQYqnPZ7XGzoekHqr5Q4YFLHYrud-Q3ZBR4_N2nJJej3Y$)

<http://216.24.232.109/freelx3500ht.asp>

http://www.consumer.philips.com/global/b2c/ce/catalog/product.jhtml?divId=0&groupId=AUDIO&catId=HOMETHEATERSYS&subCatId=DVDHOMETHEATER&productId=LX3500D_17&country=us&language=en

<http://www.download.com/>

<http://download.com.com/3101-2001-0-1.html?tag=dir>

<http://download.com.com/3000-2166-10152861.html?tag=list>

http://download.com.com/3150-2166-0.html?tag=stbc_gp &<http://www.gamestop.com/product.asp?product%5Fid=645022>

http://download.com.com/redirect?pid=10172014&merid=81695&mfgid=81695<ype=dl_elite_gen&lop=link&edId=3&siteId=4&oId=3150-2166-10172014&ontId=2166&destUrl=%2F3000-2166-10172014.html&tag=lst-0-1

<http://www.gamespot.com>

http://chkpt.zdnet.com/chkpt/gsprmo_all_qkl/gamespot.com/gamespot/features/all/holiday2002/

http://chkpt.zdnet.com/chkpt/gmbyr_gsholiday2002_466245/www.mysimon.com/cobrand/gamespot/isrch/index.jhtml?c=computergames&pgid=shop&InputTitle=Battlefield+1942&InputKeyword=&InputPlatform=PC&InputPub=&InputPriceMax=&key=&pid=GS466245&_ttag=gamespot.gsfeature

http://www.mysimon.com/lead/rlogo/redirect.jhtml?_ttag=gamespot.gsfeature&m=5267&c=computergames&url=http%3A%2F%2Fwww.gamestop.com%2Fdefault.asp%3Faffid%3D4965&order=5&sort=_&_ttag=gamespot.gsfeature

<http://www.gamestop.com/prodhttp://www.casinoonnet.com/home.htm?GoPage=FAQ&lang=en&S=063601081039884153&SR=268768&OS=063601081039884153&SR=268768&flag=No&un=trueuct.asp?product%5Fid=645022>

<http://www.casino-on-net.com/>

<http://www.casinoonnet.com/home.htm?GoPage>Aboutus&lang=en&S=063601081039884153&SR=268768&OS=063601081039884153&SR=268768&flag=No&un=truehttp://www.casinoonnet.com/home.htm?GoPage=FAQ&lang=en&S=063601081039884153&SR=268768&OS=063601081039884153&SR=268768&flag=No&un=true>

6.4 P2P tool investigation work list

ls-files and registry dump files are created twice in every step, in the beginning and at the end. Although the collected data is slightly redundant, the purpose is to minimize the number of differences found when comparing ls or registry files.

6.4.1 Installation

1. create ls-file `installation_ls_1.txt`
2. dump the registry to the file `installation_reg_1.txt`
3. check the registry with RegCleaner
4. start WinDump
5. run the installation file (exit the tool if it starts running after installation)
6. stop WinDump
7. dump the registry to the file `installation_reg_2.txt`
8. check the registry with RegCleaner
9. create ls-file `installation_ls_2.txt`
10. search for installed ad/spyware with Ad-aware (but do **not** remove any found items)

6.4.2 Running the tool (30 min/100 min)

1. create ls-file `run{30,100}_ls_1.txt`
2. dump the registry to the file `run{30,100}_reg_1.txt`
3. check the registry with RegCleaner
4. start WinDump
5. if this is the 100 minutes run, start the web surfing script and execute the search queries.
6. run the P2P tool
7. stop WinDump
8. dump the registry to the file `run{30,100}_reg_2.txt`
9. check the registry with RegCleaner
10. create ls-file `run{30,100}_ls_2.txt`
11. search for installed ad/spyware with Ad-aware (but do **not** remove any found items)

6.4.3 Removal

1. create ls-file `removal_ls_1.txt`
2. dump the registry to the file `removal_reg_1.txt`
3. check the registry with RegCleaner
4. start WinDump
5. remove the P2P tool using the utility `Add/Remove Programs` in the `Control Panel`
6. stop WinDump
7. dump the registry to the file `removal_reg_2.txt`
8. check the registry with RegCleaner
9. create ls-file `removal_ls_2.txt`
10. search for installed ad/spyware with Ad-aware and remove any found items

Typeset by L^AT_EX.