White Paper | April 2006

McAfee®
Proven Security™

Rootkits, Part 1 of 3: The Growing Threat

Table of Contents

Key Findings

1. In just three short years, the use of stealth techniques in malicious software (malware) has grown by more than 600 percent.
2. From 2000 to 2005, rootkit complexity grew by more than 400 percent, and year-over-year, Q1 2005 to 2006, complexity has grown by more than 900 percent.
3. The share of Linux-based techniques has gone from a high of roughly 71 percent of all malware stealth components in 2001 to a negligible number in 2005, while the number of Windows®-based stealth components has increased by 2,300 percent in the same time period.
4. The "open-source" environment, along with online collaboration sites and blogs, is largely to blame for the increased proliferation and complexity of rootkit components.
5. Malware authors find the Windows platform an attractive target not only because of its massive installed based but also because of the exciting technical challenge it presents with its many undocumented application programming interfaces (APIs).
6. Products employing stealth techniques are not necessarily rootkits by themselves, but the practice is aiding and abetting the spread of malware, helping it to spread further than it would otherwise.

Abstract

Rootkits are a pervasive and evasive threat to today's systems. Increasingly sophisticated stealth techniques make detecting rootkits and stopping the damage they cause a significant challenge. In this paper, we distinguish between stealth techniques that are simply strategies for concealing files, processes, and activities, and the term *rootkit*, which has come to be associated with malware that conceals its activities. Also, we look at the history of stealth and rootkit technologies to better understand how these threats evolved. We analyze the underlying motivations and technologies driving the growing use of rootkits, examine recent trends, and offer a perspective on the future of this relatively new form of malware.

A Brief History of Stealth Malware (a.k.a. Rootkits)

Originally, a rootkit was simply a collection of tools that enabled administrator-level access (also known as *root* access in the Unix world) to a computer or network. The term referred to a set of recompiled Unix tools, including *ps*, *netstat*, *ls*, and *passwd*. Because these same tools could be used by an attacker to hide any trace of intrusion, the term rootkit became associated with stealth. When these same strategies were applied to the Windows environment, the rootkit name transferred with them. Today, rootkit is a term commonly used to describe malware—such as Trojans, worms, and viruses—that actively conceals its existence and actions from users and other system processes.

The practice of hiding malware from the prying eyes of users and security products dates back to the very first PC virus, *Brain*,[1] which was released in 1986. *Brain* escaped detection by intercepting PC boot-sector interrogations and redirecting the read operations to elsewhere on the disk. Virus authors soon recognized that the longevity of any virus was critically dependent upon such stealth techniques when, in 1987, the *Lehigh* virus[2] was quickly contained after its release because it made no such attempt to hide its presence.

Malware authors continued to develop ever more complex DOS viruses throughout the late 1980s and early 1990s, adding innovative stealth techniques to mask detection. Two of the most common methods involved intercepting and replacing BIOS disk I/O interrupts for read (INT 13) calls with modified results, and continuously disinfecting a disk sector whenever a program — including a virus scanner — started up, then reinfecting the disk sector when the program execution concluded. The boot-sector virus *Tequila*,[3] released in 1991, and the file-infecting virus *1689 Stealth*,[4] introduced in 1993, used these techniques to hide their increased file lengths, concealing this otherwise obvious indicator of infection. Later DOS viruses intercept higher-level functions, such as DOS driver calls, to mask their presence or maintain infection.

The emergence of Windows in the mid-1990s brought with it immunity to DOS viruses and a brief dormancy for stealth innovation. Before they could devise ways to circumvent its defenses, virus authors first had to learn to use Windows' APIs and protected memory architecture. In late 2001, the lull in stealth activity ended with the appearance of the Trojans *NTRootkit*[5] and, later in 2003, *HackerDefender*.[6] These Trojans hooked the operating system at a very low level of function calls, allowing them to conceal their presence.

As the computing environment has evolved, so have stealth technologies. Deceptive naming conventions, network manipulation, and other techniques have been developed to hide malware in plain sight. Renaming an infected file so that it appears to be a legitimate system or user file is one of the simplest, yet most effective of these approaches. The Trojan *scvhost.exe* or *svehost.exe* could reside in the Windows *system32* directory along with the original file

named *svchost.exe*. Additionally, a Trojan named *svchost.exe* can execute from the *Windows* or *WINNT* directories. The close resemblance to the legitimate file names in the first case, and a user's lack of knowledge that the original file location should be in the Windows *system32* directory in the second, trick a user into believing the Trojan files can be trusted.

The advent of the Internet brought new opportunities and capabilities to both attackers and defenders. For malware authors, it added new propagation vectors and masses of exploitable victims. For system defenders, it provided new means of real-time network detection with intrusion prevention system (IPS) devices and other traffic-monitoring equipment to watch for the telltale signs of malicious activity.

Today, effective stealth technology must hide or protect files, processes, and registry entries. Increasingly, malware must also hide its tracks by manipulating raw packets on the network, TCP/IP stack, TCP/IP protocol,[7] and BIOS[23] to evade some of the more advanced security technologies.

Rootkits, Malware, and Controversy

Malware comes in many forms. There are, however, explicit differences between viruses, Trojans, worms, and potentially unwanted programs (PUPs). Viruses, like their biological analogues, are self-replicating programs that can also steal confidential information, block system resources, destroy information, or perform other malicious acts. Trojans are programs that appear to be benign or even useful software applications on the surface, but harbor malicious code within. While Trojans are not self-replicating, they can cause an infected computer to download other malware that is. Worms are malware that replicate by spreading copies of themselves through a shared network, floppy drives, or even USB drives, often autonomously without human intervention. Although similar to Trojans and other malware in that they often steal confidential and private information, PUPs are distinct because they are installed and executed with the tacit approval of the user.

Stealth technology, however, is not the exclusive domain of malware. PUPs and commercial software applications are increasingly employing stealth technologies to prevent their removal. In April 2005, *Adware-Isearch*[9] was one of the first adware found to use stealth technology. Since then, several others have been discovered, including *Apropos*,[10] *Qoolaid*,[11] and *DigitalNames*,[12] all of which were reclassified as Trojans because they posed a significant threat to the user.

It is tempting to classify all software that employs stealth techniques as rootkits, but to do so would dilute the clarity and power of the description. McAfee® and others have adopted this point of view, classifying commercial software that employs stealth techniques as PUPs rather than rootkits. However, others in the security community argue that any use of stealth is unwarranted and is therefore worthy of the term rootkit and its negative connotation.[13,14,15] This somewhat academic debate erupted into a highly publicized controversy when Mark Russinovich posted a finding to his blog on October 31, 2005.[28] Sony BMG's digital rights-management software, *Extended Copy Protection (XCP)*, came under fire because it employed stealth technologies and made computers vulnerable to attack. In late 2005, McAfee AVERT™ Labs recognized the potential that this commercial application had to be used for malicious purposes and began classifying the program as a PUP.

XCP features a device driver implementation that includes kernel-level privileges. This driver hides digital rights-management files and processes so that the user cannot disable them and make illegal copies of music files. This protection was achieved by writing kernel code that would hide any file, folder, or process starting with the string "sys." Unfortunately, any malware similarly named with this same string would also be hidden from most virus scanners.

As might be expected, malware authors have seized this opportunity and are writing programs that use the Sony-installed software to hide their files. For example, variants of the *W32/Brepibot* worm,[17] which was reported in November 2005 and spread through Internet Relay Chat (IRC) channels, have been found that exploit this weakness.

This commercial application calls into question the utility of labeling all programs that employ stealth technologies as rootkits. If stealth is becoming a mainstream software practice, then the term *rootkits* is perhaps better served by reserving it exclusively for malware that employs stealth techniques. McAfee and others have adopted this position, and thus classify *XCP* simply as a PUP and not a rootkit.[18]

The controversy over stealth techniques continues today. On January 10, 2006, less than two months after the public disclosure of *XCP*, the storm struck at the heart of the anti-virus community with the revelation that Symantec had used stealth techniques in one of its products to hide a directory named *NProtect*.[19] While less of a potential security risk than *XCP*, *NProtect* still shields files in a directory hidden from anti-virus scanning. Malware authors could theoretically protect their files by placing them in the *NProtect* directory. Symantec defends its use of stealth techniques as an important tool in the fight against malware, but many in the community chastise it for adopting and legitimizing the very technologies that its adversaries are distributing and promoting.

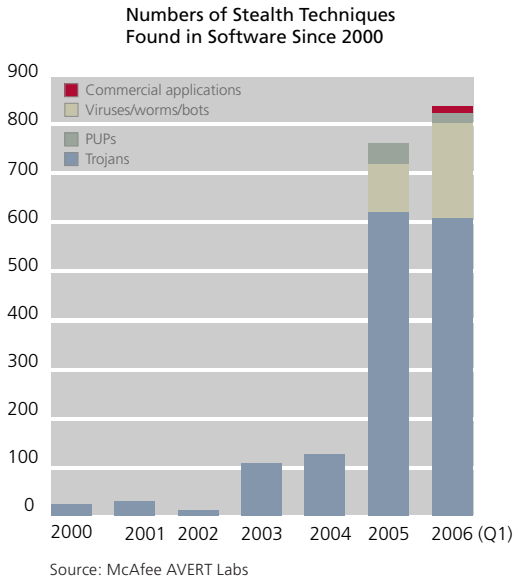
The disclosure that the Kaspersky Anti-Virus (KAV) engine[24] employed iStreams technology continued the debate regarding stealth techniques in commercial software. iStreams technology improves the KAV scanner's performance by storing the checksum of files that have already been scanned in NTFS alternate data streams (ADS). Kaspersky claimed that hiding the NTFS streams did not pose any threat, since it is the program's internal data and so will rebuild itself if overwritten by any malicious code or data.[20] Although this data-hiding technique did not pose any major security risk, it too was rebuked in the media.

Rootkit Technology Trends

In this section, we discuss the reasons behind the increase in rootkit adoption and diversity, the motivation driving rootkit writers, and the technological trends that will shape the future of rootkits.

Trend 1: Rootkits spread beyond Trojans to other forms of malware and PUPs

Over the past three years, the incidence rate of stealth technology in malware, PUPs, and commercial applications has more than sextupled. As Graph 1 shows, the use of stealth technologies was no longer the exclusive domain of Trojans in 2005, turning up in other forms of malware, as well as PUPs and commercial applications.

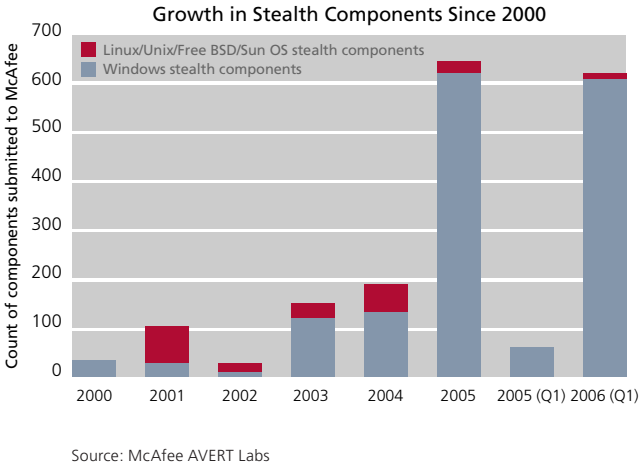


Graph 1: Growing prevalence of stealth techniques in software

The sudden rise of stealth technologies may be attributable to online collaborative research efforts. Web sites, such as *www.rootkit.com*, contain hundreds of lines of rootkit code. All of it, plus binary executables, are readily available for injection into malware. Several rootkits observed in the wild are directly borrowed or modified from the stealth technologies found on these Web sites. Some examples include *AFXrootkit*, *NTRootkit*, *FURootkit*, *He4Hook*, and *PWS-Progent*. Even worse, blog entries found on the sites sometimes go so far as to teach readers how to evade virus scan detection by compiling source code themselves, writing, "Now, that's not very creative, guys—you can compile source, right? Well, for what it's worth, that means virus scanners will look for that exact file, and a simple change here or there might make it slip by the ol' virus scanner. At least for a while."[13]

Trend 2: Rootkit sophistication is increasing

Collaboration does more than just spread stealth technologies. It also fosters the development of new and more sophisticated stealth techniques. One measure of complexity is the number of component files in a software package. For example, if a rootkit package named *a.exe* installs the files *b.exe*, *c.dll*, and *d.sys*, in which *d.sys* installs the rootkit's stealth component, the total number of components is counted as four. We make the assumption that *d.sys* is hiding or protecting other files in the package. Graph 2 illustrates the increasing complexity of rootkits over the past six years. The complexity of known rootkits increased by nearly 200 percent in 2005. By comparison, only 60 stealth components were submitted to McAfee AVERT in the first three months of 2005. This number rocketed to 612 components in the first three months of 2006, an increase of over 900 percent. While the number of samples submitted grew as well, the vast majority of that growth is attributable to the increasing sophistication of rootkit technology.

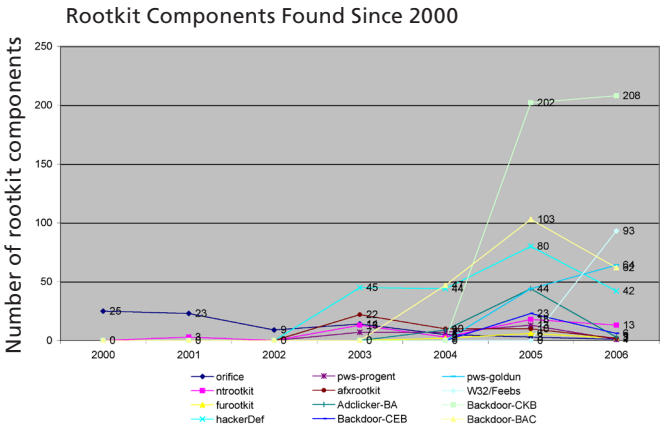


Graph 2: Distribution of stealth techniques across software families

Trend 3: Embedded Windows rootkits become dominant

Most of the rootkit activity observed today are targeted at the Windows platform. After peaking at nearly 70 percent in 2001, Linux-based stealth activity is now negligible. While Linux-based rootkits will almost certainly remain, Windows-based rootkits will clearly dominate the landscape for the foreseeable future, due primarily to the popularity of Windows and the technical challenge presented by the many undocumented Windows APIs.

Graph 3 illustrates the growth of "pure-form" rootkits on Windows,[F1] and the trend toward embedding them in other malware categories. First observed in 2001, *NTRootkit* and its variants were present in the wild until 2005. The rootkits *HackerDefender*, *AFXRootkit*, and *PWS-Progent* first appeared in 2003. *HackerDefender* has shown strong growth in recent years, while variants of *AFXRootkit* and *PWS-Progent* were detected as recently as the end of 2005. Relatively advanced rootkits, such as *FURootkit*, are among the most prevalent today. Stealth technologies embedded with varying forms of malware, such as *Backdoor-CEB*, *AdClicker-BA*, *W32/Feebs*, *Backdoor-CTV*, *Qoolaid*, *PWS-LDPinch*, *Opanki.worm*, and *W32/Sdbot.worm*, have also been found in recent months.



Graph 3: Increasing numbers of components in rootkits

Trend 4: Rootkit attack vectors found in both illegitimate and legitimate software

The versatility of stealth technologies has driven their spread into nearly every known malware attack vector. Their popularity has even convinced commercial software vendors to begin employing them in their products. As seen in Figure 1, stealth technology injection vectors now span the spectrum of software delivery methods from exploits that require no user interaction to user-installed, trusted applications. Some examples of well-known rootkits and their attack vectors demonstrate this broad coverage. *Backdoor-BAC* has been distributed via spam e-mails, Trojan downloaders, and direct exploits.[22] *HackerDefender*[6] is generally distributed via spam, bots, direct exploits, and peer-to-peer file-sharing applications. Some observed rootkits are downloaded via mass-mailing worms to create complex blended attacks. This was the case with *Backdoor-CEB*,[16] which was downloaded by *W32/MyDoom.bb*,[29] *W32/MyDoom.bc*,[30] and *W32/MyDoom.d*.[31] *FURootkit*[32,33] seems to be a favorite choice for bots such as *SDbot*[34] and *Opanki*[35] because of its highly complex structure and easy deployment. *W32/Feebs*[36] was among the first rootkits observed to spread through e-mail attachments, as well as peer-to-peer networks.

Bundled software that distributes adware is another common source of stealth technologies, usually in the form of PUPs. One such sample submitted to McAfee was named *build2.exe* and was detected as *Adware-Isearch*.[9] It bundled a desktop search utility and Firefox plug-in together and created icons that promoted the anti-spyware program spywareavenger (*www.spywareavenger.com*) and the anti-virus program VirusHunter (*www.virushunter.com*). However, the bundle also contained an adware program, *aBetterIntrnt*, which downloaded a utility. That utility then installed several more adware programs on the machine. Finally, *Adware-Isearch* dropped a kernel-mode rootkit to protect all of the newly installed files from being removed by either the user, an anti-virus scanner, or an anti-spyware scanner.[21]

[F1] "Pure-form" represents proof-of-concept rootkits, including *N trootkit* and *FURootkit*, that display innovative rootkit techniques.

Most recently, stealth technologies have spread through commercial software vectors, that is, trusted programs—as seen with the distribution of *XCP*.[18] Most users unknowingly permitted the installation of *XCP*'s stealth technology on their systems because they trusted the application and wanted to listen to copyrighted music on Sony CDs. In doing so, however, they created serious security vulnerabilities.

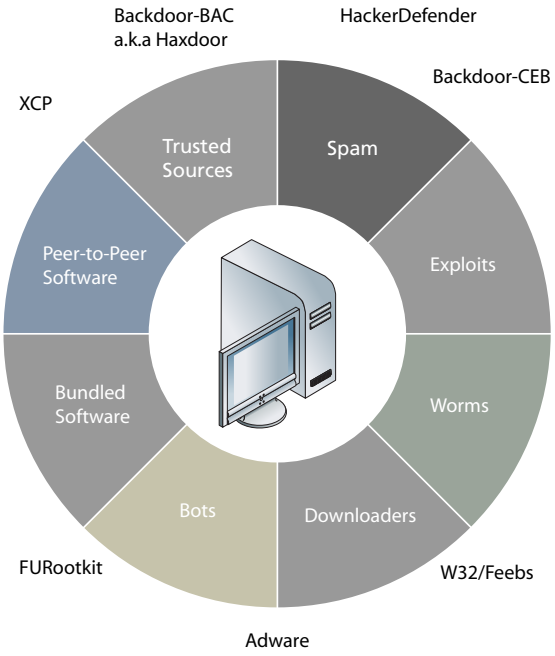


Figure 1: Attack-vector diversity—Channels used by rootkits to propagate

Trend 5: Embedding stealth technology becomes easier

With the availability of rootkit code and stealth-creation kits, malware authors can easily hide processes, files, and registries, without detailed knowledge of the target operating system. Figure 2 illustrates just how easy this is: the stealth-creation kit *Nuclear Rootkit*'s user interface simply requires a file or directory name and, with a click, uses various stealth techniques to create custom binary code that hides the file or directory, as well as ports, processes, and registry entries.

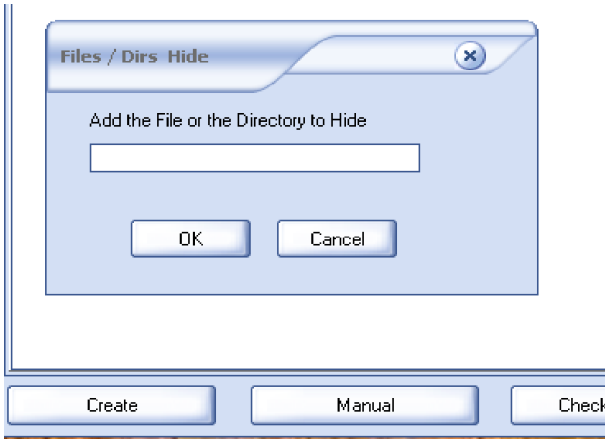


Figure 2: Nuclear Rootkit's simplistic user interface

Although the stealth-creation kit shown above is freely available for download, anyone can also buy highly complex and custom stealth-creation kits, such as *A-311 Death* and the gold edition of *HackerDefender*, for prices that range from about $200 to $2,000. The implications of this easy access were highlighted by *Backdoor-BAC*'s (alias *Haxdoor* and *A-311 Death*) phishing success. The Trojan was able to gather thousands of bank personal identification numbers (PINs), passwords, and other sensitive information for its author.[22]

Motivated by financial rewards and faced with relatively inexpensive start-up costs, hackers and malware authors continue to write new rootkits that evade detection by anti-virus scanners and other security products. The online collaboration of these malefactors presents a significant challenge to the security community as the increasing sophistication of their malware makes it ever harder to prevent, detect, and remove these malicious programs.

The Future

In 2004, McAfee recorded approximately 15,000 Trojans, out of which only 0.87 percent were Windows rootkits. In 2005, McAfee saw approximately 30,000 Trojans, but this time rootkits comprised a much more significant chunk, nearly two percent, corresponding to a nominal growth rate of almost 400 percent. Across all malware and PUPs, McAfee has seen a more than 900 percent year-over-year increase in rootkit components submitted in the first quarter of 2006.

Although a newer version of Windows (Vista) is expected soon, any drop in malware activity that might accompany its release—comparable, for example, to the lull witnessed with the release of Windows 95—would not be expected until widespread adoption had taken place. Thus, with the ease of deployment and growing popularity of rootkits among malware authors, we can predict that, in the coming two to three years, the growth of rootkits for the current

Windows architecture will reach an annual rate of at least 650 percent and that new and more cunning techniques will likely be introduced. In our next paper, "Rootkits Part 2 of 3: Technical Primer," we will analyze the evolving stealth techniques in detail and forecast the future of rootkit technology. In the third and final paper of the series, we will look at how current and future rootkits can be defeated using practical security strategies.

Works Cited

1. Brain, *http://vil.nai.com/vil/content/v_221.htm*
2. Lehigh, *http://vil.nai.com/vil/content/v_705.htm*
3. Tequila, *http://vil.nai.com/vil/content/v_98230.htm*
4. 1689 Stealth, *http://vil.nai.com/vil/content/v_98083.htm*
5. NTRootkit, *http://vil.nai.com/vil/content/v_99877.htm*
6. HackerDefender, *http://vil.nai.com/vil/content/v_100035.htm*
7. Greg Hoglund and James Butler, *Subverting the Windows Kernel: Rootkits*
8. Adware-SaveNow, *http://vil.mcafeesecurity.com/vil/content/v_100836.htm*
9. Adware-Isearch, *http://vil.nai.com/vil/content/v_133320.htm*
10. Apropos, *http://vil.nai.com/vil/content/v_137345.htm*
11. Qoolaid, *http://vil.nai.com/vil/content/v_126149.htm*
12. DigitalNames, *http://vil.nai.com/vil/content/v_135063.htm*
13. Greg Hoglund, "State of the Rootkit Address," *http://rootkit.com/blog.php?newsid=336*
14. *When's a Rootkit Not a Rootkit? In Search of Definitions, http://www.eweek.com/article2/0,1895,1913083,00.asp*
15. *Some Rootkits Are Worse Than Others, http://www.eweek.com/article2/0,1895,1910240,00.asp*
16. Backdoor-CEB, *http://vil.nai.com/vil/content/v_131340.htm*
17. W32/Brepibot, *http://vil.nai.com/vil/content/v_133091.htm*
18. XCP, *http://vil.nai.com/vil/content/v_136855.htm*
19. "Symantec Caught in Norton 'Rootkit' Flap" *http://www.eweek.com/article2/0,1895,1910077,00.asp*
20. "No Rootkit in Kaspersky Anti-Virus," *http://www.viruslist.com/en/weblog?weblogid=177727537*
21. Re; Sunbelt Software Distribution Inc. ("SunBelt"); Classification of iDownload.com's isearch Toolbar Product, *http://www.sunbelt-software.com/blog/sunbelt_idownload.pdf*
22. Backdoor-BAC.gen, *http://vil.nai.com/vil/content/v_138676.htm*
23. *Implementing and Detecting an ACPI BIOS Rootkit*, John Heasman
24. "Symantec, Kaspersky Criticized for Cloaking Software," *http://www.pcworld.com/news/article/0,aid,124365,00.asp*
25. Adclicker-BA, *http://vil.mcafeesecurity.com/vil/content/v_128301.htm*
26. AFXrootkit, *http://vil.nai.com/vil/content/v_102335.htm*
27. Adware-Elitebar, *http://vil.nai.com/vil/content/v_133782.htm*
28. Mark's Sysinternals Blog: Rootkits in Commercial Software, *http://www.sysinternals.com/blog/2006/01/rootkits-in-commercial-software.html*
29. W32/MyDoom.bb, *http://vil.nai.com/vil/content/v_131856.htm*
30. W32/MyDoom.bc, *http://vil.nai.com/vil/content/v_131860.htm*
31. W32/MyDoom.bd, *http://vil.nai.com/vil/content/v_131861.htm*
32. FURootkit, *http://vil.nai.com/vil/content/v_127131.htm*
33. FURootkit description, *http://rootkit.com/project.php?id=12*
34. SDBot, *http://vil.nai.com/vil/content/v_100454.htm*
35. Opanki.worm, *http://vil.nai.com/vil/content/v_133397.htm*
36. W32/Feebs, *http://vil.nai.com/vil/content/v_137971.htm*

McAfee, Inc. 3965 Freedom Circle, Santa Clara, CA 95054, 888.847.8766, www.mcafee.com

6-cor-root-001-0406