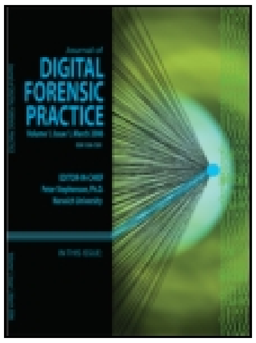

Journal of Digital Forensic Practice

ISSN: 1556-7281 (Print) 1556-7346 (Online) Journal homepage: http://www.tandfonline.com/loi/udfp20

Describing and Categorizing Disk-Avoiding Anti-Forensics Tools

Aaron Smith

To cite this article: Aaron Smith (2007) Describing and Categorizing Disk-Avoiding Anti-Forensics Tools, Journal of Digital Forensic Practice, 1:4, 309-313, DOI: 10.1080/15567280701418155

To link to this article: http://dx.doi.org/10.1080/15567280701418155

Published online: 15 Jun 2007.

Journal of Digital Forensic Practice, 1:309–313, 2006
Copyright © Taylor & Francis Group, LLC
ISSN: 1556-7281 print / 1556-7346 online
DOI: 10.1080/15567280701418155

ARTICLE

Describing and Categorizing Disk-Avoiding Anti-Forensics Tools

Aaron Smith
College of Technology, Purdue University, West Lafayette, IN 47907, USA

ABSTRACT Disk-avoiding anti-forensics tools are now being used to prevent current forensics tools from detecting their activities. These new tools must be described and categorized in order for forensics investigators to be aware of and therefore able to detect the tools and collect the information they produce. This article builds upon existing categories used to classify anti-forensics methods, such as evidence source elimination and data contraception, and provides useful information for understanding the current and anticipated trends in anti-forensics.

KEYWORDS anti-forensics, tools, evidence elimination, classification, data contraception

DESCRIBING AND CATEGORIZING DISK-AVOIDING ANTI-FORENSICS TOOLS

Having a common language between professionals in a discipline is important to foster good communication and build upon the work of others. The field of computer forensics is wrestling with this common language. In his 2006 DFRWS presentation on anti-forensics, Harris [1] defines anti-forensics as "any attempt to compromise the availability or usefulness of evidence to the forensics process" (p. 2). He elaborates that "compromising evidence availability includes any attempts to prevent evidence from existing, hiding existing evidence or otherwise manipulating evidence to ensure that it is no longer within reach of the investigator" (p. 2). Usefulness may be compromised by obliterating the evidence itself or by destroying its integrity. While it may be difficult to come to a complete agreement on whether a program or tool is an anti-forensics tool, for the purposes of this article, Harris's [1] definition will be used with the addition of the idea of intent. Therefore, for the purposes of this article, an anti-forensics tool is a program or tool that *is used with the intent* to compromise the availability or usefulness of evidence to the forensics process. Harris [1] also integrates the anti-forensic method classifications proposed by

Aaron Smith received his MS in Technology from Purdue University, and also maintains a CISSP certification. He is employed as a Network Engineer at Brotherhood Mutual Insurance in the Network Communication and Security Department. His current interests center around play dates with his daughter Aliyah and dinner with his wife Julie.

Peron and Legary [2] and Rogers [3] into a single scheme. He defines the following four categories: evidence destruction, evidence source elimination, evidence hiding, and evidence counterfeiting. Disk-avoiding anti-forensics tools fall into the category of evidence source elimination.

EVIDENCE SOURCE ELIMINATION AND HIDING EVIDENCE

Evidence source elimination is defined by Harris [2] as "neutralizing evidentiary sources (p. 3)." Peron and Legary [3] define it as preventing the creation of relevant data. Common methods that are employed to eliminate source evidence are the use of rootkits, turning off logging, bypassing system monitoring or accounting functions, and data wiping. While the definition of evidence source elimination is somewhat problematic due to the fact that a lack of evidence itself could be recognized as evidence, the category nicely encompasses techniques that *attempt* to eliminate evidence or prevent it from being created in the first place.

Another technique closely related to evidence source elimination is hiding evidence. Using inconspicuous hiding places such as slack space, hiding data within another file (such as with steganography), or using encryption are examples of this method. There are other common anti-forensics methods, but evidence source elimination and data hiding both generally involve subverting collection of evidence stored on disk media. Forensic investigators have an advantage when it comes to disk media. There are several investigative tools and techniques that have been created and refined over the years for acquiring and processing information stored on disks. One way that attackers are now trying to take back the advantage is by using new tools and techniques that avoid touching storage media.

There are some specific reasons why an attacker may want to avoid touching disk media. The first reason is to avoid detection by security systems such as anti-virus, intrusion detection/prevention, and other similar software. The second is to cover their tracks when an intrusion is discovered. To achieve these goals, attackers are employing some specific strategies and tools to subvert the forensics process.

LITERATURE REVIEW

Anti-forensics is a relatively new area of research. There are references to anti-forensics in the ACM journal going back several years but very few new articles that attempt to define anti-forensics or discuss anti-forensics tools. Many researchers acknowledge the grugq as one of the pioneers of research in anti-forensics. In 2002, the grugq [4] published a seminal work in Phrack entitled "Defeating forensic analysis on UNIX." This paper became the basis for several presentations at BlackHat conferences that he titled *The Art of Defiling*. It appears that a majority of anti-forensic research was "underground" until around 2004 or 2005. Research on anti-forensics tools began appearing more frequently in 2005. At BlackHat 2005, a conference paper by Foster and Liu ("Catch Me If You Can . . .") [5] detailed new tools that were designed to defeat popular forensics tools such as EnCase and The Coroner's Toolkit. These researchers relate detailed examples of how to use various anti-forensics tools. Their presentations are for the stated purpose of "finding opportunities to improve forensics." The Metasploit Project also began to devote some resources to anti-forensics research in 2005. At about the same time, Rogers [3] was describing these techniques in various presentations to the IT and forensics communities. His presentations focus on documenting the attacks against particular tools and informing the community of the growth and direction of anti-forensic tools and techniques.

In 2006, more research began to appear attempting to define, describe, and control anti-forensic methods. Henry [1] released *Anti-Forensics*, a white paper with detailed information on a wide spectrum of anti-forensics tools and techniques. Peron and Legary [2] also released a technical white paper detailing anti-forensic trends and underlying technical vulnerabilities that various tools and methods would or could exploit. Harris [1] also presented a paper at the 2006 DFRWS discussing how to define and control what he calls "The Anti-Forensics Problem."

Another important work in this area is Casey's [6] ACM article discussing sophisticated intrusions and examining the relationship that anti-forensic techniques have with national security concerns. His article, "Investigating sophisticated security breaches," relates how sophisticated intruders operate, weaknesses that they use, and some methods to overcome or prevent certain anti-forensics techniques.

Finally, Burdach [7] wrote a research paper discussing digital forensics of physical memory. In the paper, Burdach discussed some preliminary methods of investigating the physical memory of a compromised Linux machine. At BlackHat 2006, Burdach [8] expanded on his 2005 paper and included information on Windows memory analysis in his presentation entitled *Physical Memory Forensics*.

DISK-AVOIDING TOOLS AND TECHNIQUES

According to the grugq [4], the infrastructure of anti-forensics is built on the three strategies of data destruction, data hiding, and data contraception. Use of disk-avoiding tools would be categorized as a data contraception method. "Data contraception is the attempt to limit the quantity and quality of forensic evidence by keeping forensically valuable, or useful, data off the disk" [4]. There are two techniques used with data contraception. The first is to operate purely in memory and the second to use common utilities rather than custom-crafted tools. The first principle is of prime interest to this article. Data contraception methods would fall under Harris's [1] category of evidence source elimination.

Disk-avoiding tools (DA tools) have in common that they are used after a device has been compromised. Typically, when an attacker exploits a software vulnerability to take control of a device, he has the expectation of gaining a command interpreter or shell. Indeed, most public exploits have a payload that executes a command interpreter. However, there are certain limitations as noted by Skape and Turkulainen [9]. To avoid the limitations and to implement the principle of data contraception, attackers are changing the payload of exploits over to DA tools. The advantage to using DA tools in the payload stage of compromise is that the attacker may not need to leave any evidence on disk. They can immediately work in memory. Forensic analysis of memory is much less mature than disk forensics. Also, there is a much smaller pool of expertise and tools to examine memory than disk. Finally, most traces of evidence found in memory are eliminated when the device is powered off.

While there are no formal classifications of DA tools at the time of this writing, a slight adjustment to categories from Burdach's presentation at BlackHat USA 2006 [8] leads to a good start. These classifications are

- syscall proxying
- memory resident compiler/assemblers
- remote library injection
- direct kernel object manipulation (DKOM)
- LiveDistros

The categories may overlap slightly, but most DA tools available today can be classified easily under one of these labels. A more detailed description of each category and examples of tools within the category follows.

CATEGORIES AND EXAMPLES

Syscall Proxying

Invented in 2002 by Maximiliano Caceres, syscall proxying is a method where a local program transparently proxies a process's system call to a remote server. It provides "a real remote interface to the host's kernel. The goal is writing universal 'agents' to create all you can imagine locally but running it remotely. The best part of the syscall proxying technique is the attack tools are locally stored but remotely executed through the payload" [10]. Kernel trap calls are used by userland programs to access kernel functions. All versions of UNIX use syscalls; Windows and Cisco IOS use what Casek [10] calls "non-transparent syscalls." UNIX is the primary target for syscall proxying, although it can be done in different ways on Windows or IOS platforms. CORE Impact is a commercially available tool that falls under the category of syscall proxying.

Memory Resident Compiler/Assemblers

Memory resident compiler/assemblers are used when an attacker wants to send remote code fragments from a remote device to the compiler/assembler residing in the memory of the local (compromised) device. This technique allows tools to be compiled for the compromised platform, but, more importantly, to be compiled on the fly in memory (inside a hijacked process) so as not to leave a trace on the local disk. The most well-known and probably the first implementation was MOSDEF [11], written by Dave Aitel.

Remote Library Injection

Remote library injection occurs when a library is loaded into memory without any disk activity. In *Remote Library Injection*, Skape and Turkulainen [9] state:

> Library Injection is the process by which a dynamically linked library is injected, or forcibly loaded, into a process' address space. Once loaded, the library exists like any other standard library in that its initialization routines are called and its exported symbols can be resolved through the platform's symbol resolution interfaces. In addition, the loading process resolves all of the library's dependencies, much like the process taken when an application is launched. This provides the library with all the tools commonly exposed to an executable. In short, an injected library has the same amount of flexibility associated with an executable and is capable of running in the context of an existing process. (p. 4)

However, unlike executing an application, some methods of library injection are not externally noticeable without special tools. The difficulty in detection makes this technique ideal for anti-forensics use. The most well-known example of a library injection tool is the Meterpreter component of the Metasploit Framework. Meterpreter works by

> allowing developers to write their own extensions in the form of shared object (DLL) files that can be uploaded and injected into a running process on a target computer after exploitation has occurred. Meterpreter and all of the extensions that it loads are executed entirely from memory and never touch the disk, thus allowing them to execute under the radar of standard Anti-Virus detection. (p. 4)

Direct Kernel Object Manipulation (DKOM)

Direct kernel object manipulation (DKOM) is a method that allows an attacker to use drivers or loadable kernel modules to modify the memory associated with kernel objects such as those that represent a process' token [12]. One of the technical aspects that makes this technique possible is that Microsoft and other OS vendors typically only use two rings of privilege of the four available on Intel architecture. This leaves no separation between the kernel and third-party drivers. In practice, this means that the driver or LKM has access to kernel memory allowing any number of privileged activities including stealthy behavior. The FU rootkit is an example of a DKOM tool.

LiveDistros

The final category of DA tools is LiveDistros [13]. LiveDistro is a generic term for an operating system distribution that is executed upon boot, without installation on a hard drive. Typically, it is stored on bootable media such as a CD-ROM, DVD, or USB flash drive. The term "live" derives from the fact that it does not reside on a hard drive. Rather, it is "brought to life" upon boot without having to be physically installed onto a hard drive. A LiveDistro does not alter the current operating system or files unless the user specifically requests it—a feature that makes it ideal as a DA tool. The system returns to its previous state when the LiveDistro is ejected and the computer is rebooted. It does this by placing the files that typically would be stored on a hard drive into temporary memory, such as a RAM disk. In fact, a hard drive is not needed at all. While the above definition borrows heavily from Wikipedia [13], for a greater understanding of the history and usage of LiveDistros, the reader is directed to Pillay's [14] article in *Free Software Magazine* entitled "The magic of live CDs," Knopper's [15] original paper on Knoppix, and Schaumann's [16] white paper "Pondering live CDs" from netbsd.org.[1]

CONCLUSION

Disk-avoiding anti-forensics tools are implemented using the anti-forensics infrastructure strategy of data contraception. This technique falls under Harris's [1] evidence source elimination category. Disk-avoiding tools may be further grouped by type. Types of disk-avoiding tools discussed in this article include syscall proxying, memory resident compiler/assemblers, remote library injection, direct kernel object manipulation (DKOM), and LiveDistros.

DA tools represent not only a significant challenge to forensic investigators but also a shift in strategy for anti-forensics. Physical memory has become the new arena of choice for anti-forensics activity; thus, an important direction for new research is in the area of physical memory forensics as use of DA tools takes advantage of the lack of knowledge and experience in the area. By studying the methods and tools used by attackers, forensic investigators will be better prepared to recognize and counter the opponent.

NOTE

1. Knoppix and BartPE are two commonly used LiveDistros.

REFERENCES

1. R. Harris, "Arriving at an Anti-Forensics Consensus: Examining How to Define and Control the Anti-Forensics Problem," 2006 Proceedings of the DFRWS, August 14, 2006.
2. C. Peron and M. Legary, "Digital Anti-Forensics: Emerging Trends in Data Transformation Techniques." (2006) [cited 16 September 2006]. Available from http://layerone.info/2006/presentations/Anti-Forensics-LayerOne Paul_Henry.pdf#search=%22anti-forensics%22
3. M. Rogers, "Anti-Forensics." (2005). [cited 16 September 2006]. Available from http://www.cyberforensics.purdue.edu/docs/ Lockheed.ppt
4. Grugq, "Defeating Forensic Analysis on Unix." Phrack 0x0b, no. 0x3b (2002): Phile #0x06 of 0x12
5. J. Foster and V. Liu, "Catch Me, If You Can …" (2005) [cited 5 October 2006]. Available from http://www.metasploit.com/ projects/antiforensics/BH2005-Catch_Me_If_You_Can.ppt
6. E. Casey, "Investigating Sophisticated Security Breaches." Communications of the ACM 49, no. 2 (2006), pp. 48–55.
7. M. Burdach, "Digital Forensics of the Physical Memory." (2005) [cited 28 October 2006]. Available from http://forensic.seccure.net/pdf/mburdach_digital_forensics_of_physical_memory.pdf
8. M. Burdach, "Physical Memory Forensics." (2006) [cited 28 October 2006]. Available from http://strony.aster.pl/forensics/pdf/mburdach_ physical_memory_forensics_bh06.pdf
9. Skape and J. Turkulainen, "Remote Library Injection." (2004) [cited 28 October 2006]. Available from http://www.nologin.org/Downloads/ Papers/remote-library-injection.pdf
10. Casek, "Syscall Proxying Fun and Applications." (2005) [cited 27 October 2006]. Available from http://events.ccc.de/congress/2005/ fahrplan/events/553.en.html.
11. D. Aitel, "MOSDEF." (2003) [cited 28 October 2006]. Available from http://pacsec.jp/psj03/en/2–4dave-MOSDEF.ppt
12. J. Butler, "VICE – Catch the Hookers!" (2004) [cited 28 October 2006]. Available from http://www.blackhat.com/presentations/bh-usa-04/bh-us-04-butler/bh-us-04-butler.pdf
13. Wikipedia, "LiveDistro." (2006) [cited 28 October 2006]. Available from http://en.wikipedia.org/wiki/LiveDistro
14. H. Pillay, "The Magic of Live CDs." (2005) [cited 2 December 2006]. Available from http://www.freesoftwaremagazine.com/ articles/live_cds
15. K. Knopper, "Building a Self-Contained Auto-Configuring Linux System on an iso9660 Filesystem." (2000) [cited 2 December 2006]. Available from http://www.knopper.net/knoppix-info/ knoppix-als2000-paper.pdf
16. J. Schaumann, "Pondering Live CDs." (2006) [cited 2 December 2006]. Available from http://www.netbsd.org/˜jschauma/ nblivecds.pdf
17. E. Cole, "Evolution of the Sploit." (2005) [cited 28 October 2006]. Available from http://www.issa-nova.org/Documents/ ArchivePresentations/Evolu-tion%20of%20the%20Sploit%20April% 202005. ppt
18. Grugq, "The Art of Defiling." (2003) [cited 15 October 2006]. Available from http://www.blackhat.com/presentations/bh-asia-03/bh-asia-03-grugq/bh-asia-03-grugq.pdf
19. Grugq, "FIST! FIST! FIST! Its All in the Wrist: Remote Exec." Phrack 0x0b, no. 0x3e (2004): Phile #0x08 of 0x10.
20. P. Henry, "Anti-Forensics." (2006). [cited 16 September 2006]. Available from http://layerone.info/2006/presentations/Anti-Forensics-LayerOnePaul_Henry.pdf#search=%22anti-forensics%22
21. Kdm, "NTIllusion: A Portable Win32 Userland Rootkit." Phrack 0x0b, no. 0x3e (2004): Phile #0x0c of 0x10.
22. V. Liu and P. Stach, "Defeating Forensic Analysis." (2006) [cited 16 September 2006]. Available from http://www.metasploit.com/ projects/antiforensics/CEIC2006-Defeating_Forensic_Analysis.pdf
23. Pluf, Ripe, "Advanced Anti-Forensics: SELF." Phrack 0x0b, no. 0x3f (2005): Phile #0x0b of 0x14.
24. G. Richard III and V. Roussev, "Next-Generation Digital Forensics." Communications of the ACM 49, no. 2 (2006), pp. 76–80.
25. Skape, "Metasploit's Meterpreter." (2005) [cited 27 October 2006]. Available from http://www.nologin.org/Downloads/Papers/ meterpreter.pdf
26. Spoonm, Skape, "Beyond EIP." (2005) [cited 24 October 2006]. Available from http://www.blackhat.com/presentations/bh-usa-05/ bh-us-05-spoonm.pdf