Hiding Data, Forensics, and Anti-Forensics

Delving into the digital warrens for concealing data.

Data hiding has been with us as long as there have been digital computers and networks. Some readers of this column might be old enough to remember data hiding on tracks above 80 of the ubiquitous 5-1/4-inch double-sided, double-density floppy disks in the late 1970s. It was not uncommon to store a program key on the upper regions of the disk for copy protection of PC software. The simplicity of this scheme was elegant: the DOS operating system would only recognize the first 80 tracks, so the program key would be lost during any DOS copy procedure. This became one of the more common techniques of data hiding in the early microcomputer era, although its effectiveness was short-lived because applications programs could access the out-of-standard tracks directly by bypassing the OS function calls and accessing disk controller directly. This gave rise to a cottage industry of copy-protection-defeating (aka "pirating") software as bitsmiths quickly developed controller-based copy software that rendered this form of out-of-standard copy protection obsolete. Now of only historical interest, data hiding techniques such as this led to more sophisticated approaches that remain with us today.

Similar strategies exist for data hiding over networks. Covert channeling is a case in point. Two popular covert channeling techniques, protocol bending and packet crafting, share the same out-of-standard approach to concealing data as the PC data hiding example given previously.

Protocol bending involves the use of a network protocol for some unintended purpose. Typically, this involves embedding data in TCP/IP packets in unexpected places (akin to the higher-level tracks in the floppy disk example). A time-worn tactic is covert channeling over Internet Control Message Protocol (ICMP) packets, for example, by using the ICMP options field in each packet to convey applications-layer covert data. Since ICMP was created to transmit

PEP MONTSERRAT

Currently, the forensically interesting dimension of physical data hiding involves the techniques that take advantage of the physical characteristics of formatted storage media to hide data.

command and control information between network appliances such as network destination unreachable, source quenching, echoes (pings) and their replies, there is no expectation that applications-layer data will be included in ICMP packets.

As a result, most firewalls and intrusion detection/prevention systems don't inspect ICMP packets. This is where the protocol is 'bent,' which results in the establishment of a covert channel between network endpoints that lies under the radar of any network administration tool that assumes that ICMP packets will all conform to IETF specifications. Perhaps the most widely known ICMP covert channel tool is Loki, a program named after the contriver of mischief in Norse mythology. In the absence of exhaustive packet analysis, Loki traffic looks like any other routine ICMP request-reply pattern for pings, source quenching, and so forth, while in fact these ICMP packets are transmitting covert data. Another popular protocol bender is Reverse WWW Shell, which uses a form of protocol bending called shell shoveling over HTTP.

Covert channeling via protocol benders deploys protocols in non-standard and perhaps nefarious ways. Contrasted with protocol benders are covert channeling tools that use packet crafting to embed data in the actual packet headers themselves. Covert_TCP and NUSHU are two such examples. Covert_TCP uses active channeling where it generates its own packet train to create the channel. On the other hand, NUSHU is a passive channeler that piggybacks on packets transmitted to the TCP/IP stack by other applications. The covert effect is the same.

So there you have it: a summary of data hiding from hiding application data on either storage media or TCP/IP packets in places where the standards suggest it doesn't belong. Part one of this two-part column deals with physical data hiding on disk file systems.

Physical Data Hiding

Physically hidden data is a special case of dark data (aka data dark matter). Information technologists speak of cyberspace, the Internet, and corporate intranets as mostly "dark," in that they contain large amounts of undiscovered, concealed, misplaced, missing, or hidden data. By some accounts, dark data is an order of magnitude larger than light data (data that is known, linked, observed, recovered, retrievable).

Covert data may be thought of as a small subset of dark data. There are many categories of covert data. Encryption produces dark data in the sense that while the existence of the data isn't hidden, its content is only readable and usable to those who have the correct decryption key. Steganography produces dark data that is typically buried within light data (for example, a non-perceptible digital watermark buried within a digital photograph). Both are illustrations of intentional concealment. They share this characteristic with physical data hiding.

Currently, the forensically interesting dimension of physical data hiding involves the techniques that take advantage of the physical characteristics of formatted storage media to hide data. One early attempt to do this was illustrated by Camouflage (camouflage.unfiction.com) that hid data in the area between the logical end-of-file and the end of the associated cluster in which the file was placed (called file slack or slack space). Though primitive, hiding data in file slack has the dual advantage that the host or carrier file is unaffected while the hidden data is transparent to the host OS and file managers. The disadvantage is that the hidden message is easily recovered with a basic disk editor.

The ability to hide data on computer storage media is a byproduct of the system and peripheral architectures. If all storage were bit-addressable at the OS level, there would be no place to hide data, hence no physical concealment. But for efficiency considerations, system addressability must be at more abstract levels (typically words in primary, and blocks in secondary). Such abstractions create digital warrens where data may go unnoticed or, in some cases, be inaccessible.

Disk Drives and Digital Warrens

Where on a disk can data possibly be hidden? A formatted hard drive may be thought of as a logical structure mapped onto a physical medium. The logical structure consists of partitions, file systems, files, records, fields, and so forth. The physical structure consists of disks, cylinders, tracks, clusters, and sectors. The absence of 1:1 mappings between the logical and the physical realms creates the digital warrens for concealed data.

This has several implications. For example, applications software and OSs typically interface with the logical structure. If data was concealed on a disk, the typical user would never know it.

More frightening, however, is that modern computer forensics tools are not designed to uncover all digital warrens. They typically focus on those disk areas that have already been observed to hold concealed data. This presents a major problem for law enforcement, because the more sophisticated hacker, criminal, or terrorist could take advantage of the disk warrens that are not easily found by current forensics tools. So, from a security and forensics point of view it is wise to approach the problem of data hiding from the point of view of what's possible rather than what's already known.

For example, computer vendors commonly create two reserved areas when they format new computer hard disks: a Host Protected Area (HPA) for their proprietary software and data, and a Device Configuration Overlay (DCO) area for disk metadata. You've probably noticed the abundance of software that bears the name of the manufacturer that came with your computer for management, updating, and diagnostic functions. The manufacturer wants this software available to the user, and wants to make it difficult for the user to delete it. Such software would typically fit in the HPA.

Access to of these areas by an OS is prohibited by the disk controller. This is the modern-day analogue to the track 81–82 copy protection scheme that was described in the first paragraph of this column. The hack in this case would be to write a program at a low enough level to access the disk controller, and then hide the data in the HPA or DCO—not difficult at all if one knows the physical boundaries and boots to a non-host OS. Even within the OS, it's possible to reassign these areas to OS control, change the contents, and then reassign them to the HPA or DCO. This is an example of a hiding method that takes advantage of what is more or less a "physical" feature of the drive architecture. So, with a little sophistication one could bury covert data in either the HPA or DCO where it would be concealed from even the OS.

Further down in the disk hierarchy, we have the disk partition. Modern OSs allow the administrator to redefine the number and sizes of disk partitions with any number of commercial and shareware utilities. It is fairly common to place the OS (that is seldom modified) on a partition by itself and place all applications on another partition. In this manner, rigorous configuration changes to the applications software would be unlikely to affect the OS.

Therein lies another opportunity to conceal data. Because the logical partition may not fit perfectly within the physical subdivisions of the disk, partition slack results. Partition slack is the area between the end of a logical partition and the end of the physical block the partition falls within. As with the HPA/DCO example, this partition slack space is unusable to applications and the OS. Extended partitions exacerbate the problem by enabling a multitude of embedded logical partitions, each one of which contains a digital warren of 62 sectors.

If the partitions in aggregate do not use up all of the available disk space, volume slack results. One

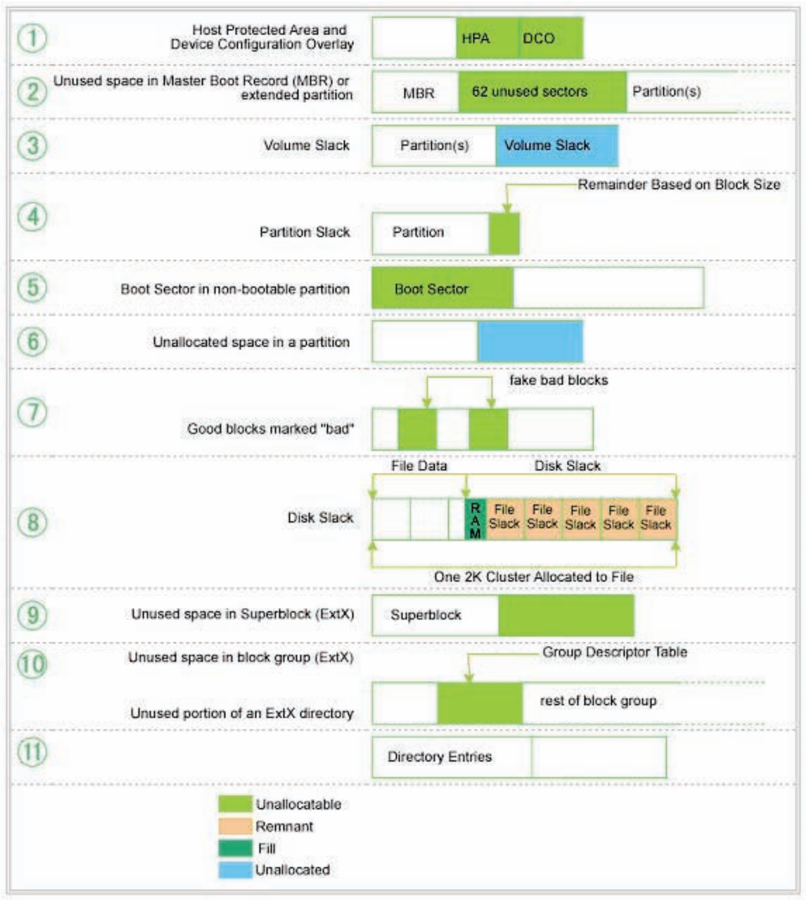

1
Host Protected Area and Device Configuration Overlay
HPA
DCO
2
Unused space in Master Boot Record (MBR) or extended partition
MBR
62 unused sectors
Partition(s)
3
Volume Slack
Partition(s)
Volume Slack
4
Remainder Based on Block Size
Partition Slack
Partition
5
Boot Sector in non-bootable partition
Boot Sector
6
Unallocated space in a partition
7
fake bad blocks
Good blocks marked "bad"
8
File Data
Disk Slack
Disk Slack
RAM
File Slack
File Slack
File Slack
File Slack
File Slack
One 2K Cluster Allocated to File
9
Unused space in Superblock (ExtX)
Superblock
10
Unused space in block group (ExtX)
Group Descriptor Table
Unused portion of an ExtX directory
rest of block group
11
Directory Entries
Unallocatable
Remnant
Fill
Unallocated

Figure 1. Covert data warrens on disk drives (source: H. Berghel, D. Hoelzer, and M. Sthultz, Data hiding tactics for Windows and Unix file systems; www.berghel.net/publications/data_hiding/data_hiding.php).

could easily create a multitude of partitions, load one with covert data and then delete it. Since deleting the partition does not delete the data but only the reference to it by the OS, the data remains, located beyond the reach of applications and OS.

Down further still, is routine disk slack. It would be very unusual to find files that are exactly as long as the sector/cluster sequence in which they are stored. At the sector level, any unused part of a partially filled sector is padded with either data from memory (RAM slack) or null characters (sector slack). After the padded sector, any remaining, unused sectors are simply ignored (file slack). Once again, the OS and applications have no access to this space, which follows the end of an active file but is within the allocated sectors and clusters. Figure 1 lists 11 data warrens on file systems that are typically unobservable. Variations on this theme are endless.

Forensic Implications

Without question, the most frightening side effect of these digital warrens is the inability of modern forensic tools to easily recover the data. With workstations now shipping with RAID five stacks and terabytes of disk space, manual investigation of hard drives at the byte level is simply not viable.

In a sense, we've been living in a fool's paradise because today's crooks and criminals seldom take extraordinary measures to conceal data. Most of the forensics work in law enforcement that I'm aware of involves very basic data recovery techniques with a few popular forensics tools. However, it would be unwise to expect this to continue, as miscreants and their misdeeds become more sophisticated.

For simplicity, I will illustrate the principle of covert data hiding on a hard disk with a simple example based on the old FAT 16 format. The relevant design consideration is a 1:1 mapping between the entries in the file allocation table and the physical clusters on the disk. For example, a FAT entry of hex 0000 indicates the corresponding cluster on the disk is free for use. A hex value of 0002-FFEF is a pointer to the next cluster on the disk that is part of a file. Hex FFF7 indicates a bad cluster that has been culled so that it can't be reallocated.

You guessed it, our simple example will involve changing some entries in the FAT from "free" to "bad," and then storing data on the bad clusters. In our

URL Pearls

Out-of-standard disk copying software has passed into the digital dustbin. Trade magazines of the late 1970s and early 1980s would reveal widespread use of such software among computer enthusiasts of that era.

Loki (www.phrack.org) has been a premier covert channeling tool for Unix systems for many years. Although it is widely associated with ICMP, in principle it could use any protocol that is unlikely to be subjected to close inspection by network security appliances. Unless the packets are analyzed, the Loki transmission looks innocuous (for example, an ICMP ping request or a UDP DNS query). Loki can encrypt all data for additional stealth, and swap between ICMP and UDP on the fly. For further details on the ICMP and UDP packet formats, see our Packet Pal Primer at www.berghel.net/resources/packetpal/index.php.

Another approach to covert channeling is the reverse WWW shell (aka, shoveling shell) developed by van Hauser in the late 1990s (www.megasecurity.org/Sources/rwwwshell-1_6_perl.txt). Like Loki, the reverse WWW shell requires a server daemon to be running on the server. The daemon submits outbound HTTP requests for commands from an external computer. The intruder's command is contained within the HTTP response. The command is executed on the compromised computer, and the results are subsequently shoveled to the intruder via a stream of outbound HTTP packets. The HTTP traffic that contains the covert data appears to the network to be routine Web surfing.

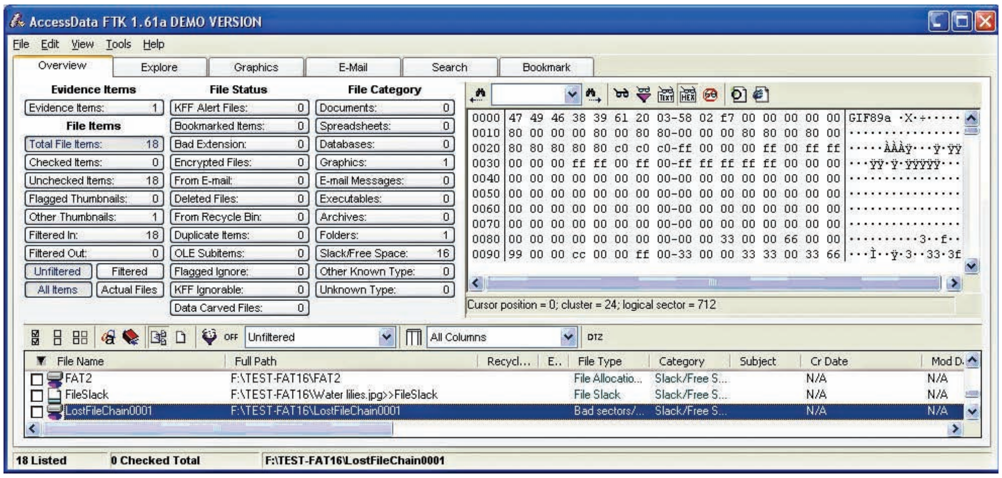
Where Loki and the reverse WWW shell establish the covert channel over an embedded protocol by means of protocol bending, other techniques exist for establishing a covert channel by means of packet crafting. Craig Rowland's Covert_TCP (www.securityfocus.com/tools/1475) and Joanna Rutkowska's NUSHU (http://invisiblethings.org/papers/passive-covert-channels-linux.pdf) are two such examples.

Covert_TCP creates "active channels," that is, the daemon actually generates packets with data buried in either the ID field of the IP packet or the Sequence or Acknowledgment Number fields of the TCP packet (see www.berghel.net/resources/packetpal/index.php). By contrast, NUSHU creates "passive channels" by embedding the data in the SEQ and ACK fields of existing packets by adding an offset (data value) to the existing sequence number. The sequence of offset values is the covert data. The daemon just has to remember to subtract that offset from the returned sequence number to fool the application. Nushu, incidentally, means "woman's writing." It is a apparently a secret language developed by Chinese women. "Traditional Chinese culture is male-centered and forbids girls from any kind of formal education, so Nushu was developed in secrecy over hundreds of years in the Jiangyong county of Hunan province"; see www.crystalinks.com/nushu.html.

Dark data/digital dark matter is usually used in some search or indexing context. See Paul Chin's 2005 summary of dark data within intranets (www.intranetjournal.com/articles/200507/pij_07_07_05a.html) or the recent discussion on the PC Forum blog where Yahoo's Jeff Weiner estimates that 99% of the world's collective knowledge is dark data: blogs.zdnet.com/BTL/?p=2715.

Cryptography, steganography, and digital watermarking have been extensively reported in the professional literature, so a Web search will provide millions of links.

For readers interested in file carving and disk wiping, consult my August 2006 column. For a more thorough treatment of the topic, see "Data hiding tactics for Windows and Unix file systems," by H. Berghel, D. Hoelzer, and M. Sthultz at www.berghel.net/publications/data_hiding/data_hiding.php, and the February 2006 section of *Communications* on next-generation cyber forensics. **C**



Evidence Items		File Status		File Category	
Evidence Items:	1	KFF Alert Files:	0	Documents:	0
File Items		Bookmarked Items:	0	Spreadsheets:	0
Total File Items:	18	Bad Extension:	0	Databases:	0
Checked Items:	0	Encrypted Files:	0	Graphics:	1
Unchecked Items:	18	From E-mail:	0	E-mail Messages:	0
Flagged Thumbnails:	0	Deleted Files:	0	Executables:	0
Other Thumbnails:	1	From Recycle Bin:	0	Archives:	0
Filtered In:	18	Duplicate Items:	0	Folders:	1
Filtered Out:	0	OLE Subitems:	0	Slack/Free Space:	16
		Flagged Ignore:	0	Other Known Type:	0
		KFF Ignorable:	0	Unknown Type:	0
		Data Carved Files:	0		

```
0000 47 49 46 38 39 61 20 03-58 02 f7 00 00 00 00 00 GIF89a ·X·÷·····
0010 80 00 00 00 80 00 80 80-00 00 00 80 80 00 80 00 ················
0020 80 80 80 80 80 c0 c0 c0-ff 00 00 00 ff 00 ff ff ·····ÀÀÀÿ···ÿ·ÿÿ
0030 00 00 00 ff ff 00 ff 00-ff ff ff ff ff 00 00 00 ···ÿÿ·ÿ·ÿÿÿÿÿ···
0040 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 ················
0050 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 ················
0060 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 ················
0070 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 ················
0080 00 00 00 00 00 00 00 00-00 00 33 00 00 66 00 00 ··········3··f··
0090 99 00 00 cc 00 00 ff 00-33 00 00 33 33 00 33 66 ···Ì··ÿ·3··33·3f
```

Cursor position = 0; cluster = 24; logical sector = 712

File Name	Full Path	Recycl...	E..	File Type	Category	Subject	Cr Date	Mod D.
FAT2	F:\TEST-FAT16\FAT2			File Allocatio...	Slack/Free S...		N/A	N/A
FileSlack	F:\TEST-FAT16\Water lilies.jpg>>FileSlack			File Slack	Slack/Free S...		N/A	N/A
LostFileChain0001	F:\TEST-FAT16\LostFileChain0001			Bad sectors/...	Slack/Free S...		N/A	N/A

18 Listed   0 Checked Total   F:\TEST-FAT16\LostFileChain0001

Figure 2. File carver analysis of covert data on "bad" clusters (source: H. Berghel, D. Hoelzer, and M. Sthultz, Data hiding tactics for Windows and Unix file systems; www.berghel.net/publications/data_hiding/data_hiding.php).

case, we modified the FAT to show clusters 24–29 as bad, and then stored a GIF file on those clusters. The OS sees the clusters as bad and won't access them, so the data is covert from the OS point of view. But suppose we look at this forensically.

A mainstay of modern forensics tools is a file carver. File carvers attempt to reconstruct the disk contents without using the OS's meta-level information. Figure 2 shows the result of looking the example disk with a modern file carver. It can be observed that the file carver ignored the file's "magic number" identifier that revealed it as a GIF graphics file, and simply reported the clusters as a lost file fragment, that is, it saw something there, but didn't look to see what it was so it, erroneously, assumed that it must have been data residue from a broken file allocation chain. This is analogous to network administration ignoring the contents of the options field of an ICMP packet.

Conclusion

It isn't a question of whether covert data is being hidden on hard drives of unsuspecting users, but what is being hidden and for what purposes. Well-funded hackers, criminals, and terrorists are already hiding the data, while law enforcement tries to catch up with the latest tactic of the day. Challenged by resource limitations, law enforcement personnel must rely on the technical community to help provide solutions and motivate vendors to pay closer attention to such potential security breaches.

To make matters worse, anti-forensic tools have been developed that are becoming as sophisticated as the forensics tools they seek to defeat. To illustrate, the Metasploit project (www.metasploit.com/projects/antiforensics) has developed three tools that are devastating for automated forensic analysis tools:

Timestamp provides complete editing capabilities of the NTFS timestamp rendering the timestamps recovered by forensics tools unreliable in court),

Slacker is an automated tool for storing files in slack space, and to appear in the near future,

Transmogrify is a tool to defeat file signature analysis.

The importance of this burgeoning art of anti-forensics can not be overstated. Imagine the impact on law enforcement if fingerprint evidence was unreliable and iris scans could be easily spoofed. In many ways, anti-forensics is scarier than network hacking. It offers the triple threat of hiding covert data, manipulating system data to exonerate a criminal, and planting system data to implicate an innocent party—without leaving behind telltale evidence. **C**

Hal Berghel is associate dean of the Howard R. Hughes College of Engineering at the University of Nevada-Las Vegas, the director of the Center for Cybersecurity Research (ccr.i2.nscee.edu), and co-director of the Identity Theft and Financial Fraud Research and Operations Center (www.it.ffroc.org).