

Technical White Paper

July 22, 2003 www.TechPathways.com

Christopher L. T. Brown, CISSP

clbrown@techpathways.com

Why Low Level Disk Auditing is as Important as Virus Scanning

Abstract

Security conscious system administrators will agree that a comprehensive plan for monitoring and auditing a network can be as important as securing the network in the first place. This white paper is intended to discuss today's audit trends and outline why auditing plans that are considered the most comprehensive today may fail to alert you to crucial information about compromised systems.

Background

In addition to comprehensive intrusion detection and prevention systems, one of the best ways to stay on top of network security is to implement good system logging practices at the host level and follow up with regularly scheduled comprehensive security audits.

The benefits to comprehensive security audits are hard to ignore. In addition to reducing a company's exposure to litigation by allowing administrators to act on security breaches early, audits can prevent the outbreak of costly wide spread additional security breaches. Based on the 2003 Computer Crime and Security Survey published by the Computer Security institute the average loss due to system penetration from an outsider was \$56,212.00. The average loss due to sabotage of data networks was \$214,521.00. These losses occurred even though over 70% of the companies surveyed utilized intrusion detection Systems. Preventing or reducing the effect of either incident category can result in substantial cost savings.

A truly comprehensive security audit should include audits of administrative procedures, technical configuration and the physical controls aspects of the organization. This paper focuses on the technical aspects of audits to host systems.

Comprehensive system audit checkpoints may include:

1. Checking Administrator accounts and insuring no default passwords are utilized
2. Checking user accounts for poor passwords
3. Checking user, group, and object share permissions (Access Control List)
4. Checking for dormant accounts

5. Checking for suspicious files
6. Reviewing overall system configuration
7. Checking all ports to insure only proper ports are open
8. Reviewing Audit logs
9. Checking for Viruses
10. Checking for Trojans
11. Checking hosts patch level
12. Checking hosts for known vulnerabilities

As you might suspect, conducting comprehensive system audits can be very time consuming and need to be automated as much as possible. Over the years automated security audit tools have matured and helped system administrators keep the bad guys at bay. While these automated products may not cover everything needed in all situations they have greatly reduced the time required to accomplish high-level audits and provide a necessary adjunct to in-depth manual audits.

To accomplish the comprehensive security audit checkpoints previously outlined, automated audit tools generally fall into one of two familiar categories; Hostile Code Scanners or Configuration Scanners.

Hostile Code Scanners

- Virus Scanners
- Trojan Scanners

Configuration Scanners

- Vulnerability Scanners
- System Configuration Scanners

Hostile code scanning applications use the host operating systems file I/O to extract a segment of each file to be scanned. Once this segment of the systems file is extracted, it is then compared against a database of previously identified hostile [Trojan, Virus] file segments called signatures.

Did you know? While some intrusion detection systems utilize signature databases they are focused on detecting hostile code as it traverses the network or host file system and are considered a monitoring tool rather than an auditing tool. Even the best intrusion detection system will allow new viruses, worms, and Trojans to be installed in your system until they are identified and their signatures can be developed and installed. Since new threats are being identified every day, chances are relatively high that one or more have gotten through your intrusion detection system. Comprehensive auditing is the only way to eliminate these threats and should be a key component to any overall security plan.

Configuration scanners often use a similar approach to hostile code scanners by maintaining a database of preferred configurations or known vulnerabilities. The configuration scanner also utilizes the host operating systems file and network I/O to compare current settings to the database of known vulnerabilities and undesirable configurations.

Another notable approach to configuration scanners is to create a set of cryptographic file hashes for “known good” system files then compare the snap-shot of “known good” hashes to the cryptographic hashes of the current files during the audit process. In this case if any

file has changed it should be suspect. This approach is sometimes referred to as a tripwire approach.

Did you know? A Cryptographic hash is an algorithm used to produce fixed-length character sequences based on input of arbitrary length. Any given input always produces the same output, called a hash. If any input bit changes, the output hash will change significantly and in a random manner. Additionally there is no way the original input can be derived from the hash. Two of the most commonly used hashing algorithms are MD5 and SHA1.

The common component in all of the audit approaches we have discussed so far is that the audit tool [scanner] uses programming calls to a local systems file or network I/O which cannot be trusted during audits.

Unfortunately today's hackers can easily affect a host at a much deeper level than merely replacing files to cover their tracks and set up services. Hackers achieve this deeper infection by installing one of the widely available "Kernel Mode rootkits". These rootkits are implemented as device drivers in Windows platforms and LKM's (Loadable Kernel Modules) in Linux.

To better understand "Kernel Mode rootkits" let's take a look at the basic principles of the security kernel architecture used in Windows NT/2000/XP platform design. Microsoft divides the operating system into two modes;

User Mode - This is where all general applications operate. General applications and subsystems for Win32, Win16 and POSIX (Portable Operating System Interface) are all run in this mode.

Kernel Mode – This mode is a trusted mode of operation for system services and device operations or access. All requests by user mode applications are brokered through Windows NT Executive Services within the Kernel Mode. This includes checking security ACL's (Access Control List) and allowing access to file I/O and attached devices.

Did you know? Early rootkits only replaced user mode applications such as "netstat", "dir", etc. By replacing "dir" a hacker could control the "dir" application output (set to not display certain files), but "dir" would still need to request all file I/O from a protected source in the Kernel Mode. It was these early rootkits that hashing and trusted binary schemes were designed to overcome.

The current approach to "Kernel Mode rootkits" is simple. If the goal is to hide a file or process, rather than replace "dir" or "netstat" why not replace the command user mode applications would call for information from in the Kernel? In the case of file I/O we need to replace the kernel mode I/O routine "ZWQUERYDIRECTORYFILE" In this approach not only will "dir" be able to hide hacker's files, but any other application which makes a call to the kernel mode I/O routine "ZWQUERYDIRECTORYFILE" will receive compromised

information. Hackers accomplish this by writing a Windows Device Driver that through a process called “Hooking” replaces trusted kernel mode I/O routine with their own. Of course the hacker’s routine only provides information they want users to see.

Did you know? The TK Worm developed and released by members of hacker group “Thr34t Krew” included the automated installation of a rootkit as part of its payload. During law enforcement investigation of Thr34t Krew and the TK Worm in early 2003 it was determined that over 18,000 servers were infected with the TK Worm and essentially rooted. This worm even patched the system after installing itself to avoid being replaced by other worms and avoid detection from system patch audits. While infection with known versions of the TK Worm may leave some traces, this is an example of wide spread automated rootkit installation. More worms of this nature are certain to follow. This worm is still spreading and some estimates believe it has now infected over 50,000 systems, eluding many current auditing tools.

The implication of these relatively new hacking techniques is that auditing files or comparing hash values of files on the system is useless because any hashes created on the system can not be trusted. The newly created local hashes would use local system file I/O and the user mode files most likely didn’t change anyway. Using trusted binaries running locally will not help for the same reasons.

Another important issue to consider is all that the auditing process on the suspect system, even when using trusted binaries from a CDROM, change almost every file’s last accessed time. If the audit process identifies an incident, tracking hacker’s actions becomes more difficult and these chances can raise authenticity issues in legal proceedings.

Low-Level Disk Auditing

One accepted way to detect a Kernel Mode rootkit is to reboot the suspected system in “Safe Mode”, then look around for anything that’s been hiding. Another way is to connect to the suspect system file shares from a trusted remote system (using the trusted remote system I/O and trusted binaries) then audit as before. In the first case taking the server offline during the auditing process is rarely an option. In both cases files last access times will be changed and the question may still remain “are the trusted binaries truly trusted?”

How do you audit a live system with trusted I/O and not change any last accessed times?

To answer this need Technology Pathways has introduced a new network enabled version of its computer forensics product, ProDiscover® IR which reads the live disks sector-by-sector then implements a read-only file system for auditing the suspect system. This process of low level disk reads and utilizing a separate, trusted read-only file system can truly reveal all the files on the disk, even if the system has been rooted by a Kernel Mode rootkit. ProDiscover® IR’s core features provide the system administrator the ability to audit systems in a *least-intrusive* manor preserving evidence for possible criminal or civil litigation. Among other techniques the investigation may include searching for known file hash values, search for

known file signatures, or searching files and disks for keywords. All search functions are provided for recovered deleted files and NTFS Alternate Data Streams. ProDiscover® IR is intended to be a key part of a comprehensive IR investigation including monitoring Intrusion Detection Systems, logging and auditing.

Did you know? Some administrators will suppose that if a rootkit could hook (replace) File I/O request they could simply hook the sector level read commands and foil the approach that applications such as ProDiscover® IR use. While this is true on the most basic level, hooking kernel sector read commands would have a trickle-down effect on all other kernel level file system operations and require a large amount of real-to-Trojanned sector mapping and/or specific sector placement for the rootkit and supporting files. This undertaking would not be a trivial task even for the most accomplished kernel mode rootkit author.

For quick audits the administrator need only utilize a PDServer™ CDROM or floppy and run “pdserver.exe” on the suspected system. Once PDServer™ is running, connect to the suspect system from the ProDiscover® IR client and add the suspected disk to a current project.

Did you know? Alternate Data Streams (ADS) were created as file attributes in the Windows NT file system to support Macintosh computer file system resource forks. Resource forks in the Macintosh file systems are separate files which contain information about the type of data in each file. The NT File System does not provide a native facility to view or detect the existence of ADS files, but can create and access these files. In recent years ADS files have been created on systems to hide data including executable program files from system administrators and security scanners including virus scanners.

If it turns out after an audit that the system had been compromised, full incident response procedures can be implemented that may include creating a bit-stream image of the suspect hard disk. In this case ProDiscover® IR will allow the administrator to create the image through any TCP/IP LAN or WAN in a secure channel encrypted with the 256 bit TwoFish encryption algorithm.

Conclusion

Comprehensive security auditing is a key component of any information technology risk management program. As the sophistication of the hackers increase, so must the tools to detect them. Using today’s hostile code scanners and configuration scanners which rely on the system file I/O is not enough to alert administrators to compromised systems.

It is also becoming more important to report the discovery of compromised systems to legal authorities. Recent laws require retailers to report any potential loss of credit information. The processes utilized in identifying compromised systems during audits and the capturing of evidentiary quality data can be critical to successful criminal or civil litigation. ProDiscover® IR provides administrators with solutions to implement and properly manage the technical aspects to the corporate audit process.