White Paper

Detecting System Intrusions

Prepared on January 15, 2013 by:

Demyo Inc. is one hundred percent IT security oriented company with headquarters in Miami, Florida, USA.

Demyo Inc. delivers comprehensive penetration testing, vulnerability assessment, incident response, and compliance audit services just to name a few. Find out more at:

www.demyo.com

info@demyo.com

Tel. +1 201 665 6666

Miami, Florida, USA

Introduction

First things first, detecting system intrusion its not the same as Intrusion Detection System/Intrusion Prevention System (IDS/IPS). We want to detect system intrusion once attackers passed all defensive technologies in the company, such as IDS/IPS mentioned above, full packet capture devices with analysts behind them, firewalls, physical security guards, and all other preventive technologies and techniques. Many preventing technologies are using blacklisting [1] most of the time, and thus that's why they fail. Blacklisting is allowing everything by default, and forbidding something that is considered to be malicious. So for attacker it is a challenge to find yet another way to bypass the filter. It is so much harder to circumvent a whitelisting system.

Monitoring Key Files In The System

What are key files on the server? In Linux machine it will be /etc/passwd, /etc/shadow just to mention a few.

Lets take a look at example of /etc/shadow file:

```
# cat /etc/shadow

root:$6$OFny79f/$LC5hcqZXNYKachPKheRh5WkeTpa/zO3y8OX3EUHrFkrFQAdLU
TKwGjLPSdZ9uhwJQ9GmChLvbhPRbPw7lDTg90:15231:0:99999:7:::
daemon:x:15204:0:99999:7:::
bin:x:15204:0:99999:7:::
sys:x:15204:0:99999:7:::
www-data:15204:0:99999:7:::
<snip>
pulse:*:15204:0:99999:7:::
rtkit:*:15204:0:99999:7:::
festival:*:15204:0:99999:7:::
postgres:!:15204:0:99999:7:::
apache:$6$LqrWIgqp$jdq1exB2GiBFgLL9kDlDkks30azWBJ1/mDU.to84mHn6nmzU
zV7iHiMXK7rVm8.plMmaNKg9Yyu7ryw00r5VX.:15452:0:99999:7:::
```

What is wrong whit it? If you take a look at users list in this file you will notice that apache user has a hash value to it. Typically apache service never has any hash associated to it. If there is a hash for a use in this file that means this user has a password associated with it and is able to login via SSH. What happen here is hacker made a brand new account and is trying to masquerade with a valid system user/process.

One of the ways to monitor changes in the file system is to implement LoggedFS. This particular file system logs everything that happens on inside the files system. It is easily configurable via XML files to fit your needs [2].

Example of LoggedFS configuration file:

```
<?xml version="1.0" encoding="UTF-8"?>

<loggedFS logEnabled="true" printProcessName="true">
 <includes>
  <include extension=".*" uid="*" action=".*" retname=".*"/>
 </includes>
 <excludes>
  <exclude extension=".*\.bak$" uid="*" action=".*" retname="SUCCESS"/>
  <exclude extension=".*" uid="1000" action=".*" retname="FAILURE"/>
  <exclude extension=".*" uid="*" action="getattr" retname=".*"/>
 </excludes>
</loggedFS>
```

This configuration can be used to log everything except it if concerns a *.bak file, or if the uid is 1000, or if the operation is getattr.

Files Integrity

File integrity monitoring (FIM) is an internal control or process that performs the act of validating the integrity of operating system and application software files using a verification method between the current file state and the known, good baseline. This comparison method often involves calculating a known cryptographic checksum of the file's original baseline and comparing with the calculated checksum of the current state of the file. Other file attributes can also be used to monitor integrity.

Generally, the act of performing file integrity monitoring is automated using internal controls such as an application or process. Such monitoring can be performed randomly, at a defined polling interval, or in real-time.

Security Objectives

Changes to configurations, files, and file attributes across the IT infrastructure are common, but hidden within a large volume of daily changes can be the few that impact file or configuration integrity. These changes can also reduce security posture and in some cases may be leading indicators of a breach in progress. Values monitored for unexpected changes to files or configuration items include:

- Credentials
- Privileges and Security Settings
- Content
- Core attributes and size
- Hash values
- Configuration values [3]

Many open-source and commercial software products are available that perform file integrity monitoring:

- CimTrak
- OSSEC
- Samhain
- Tripwire
- Qualys
- nCircle
- Verisys
- AIDE [4]

nCircle file integrity monitor panel is in figure 3.1

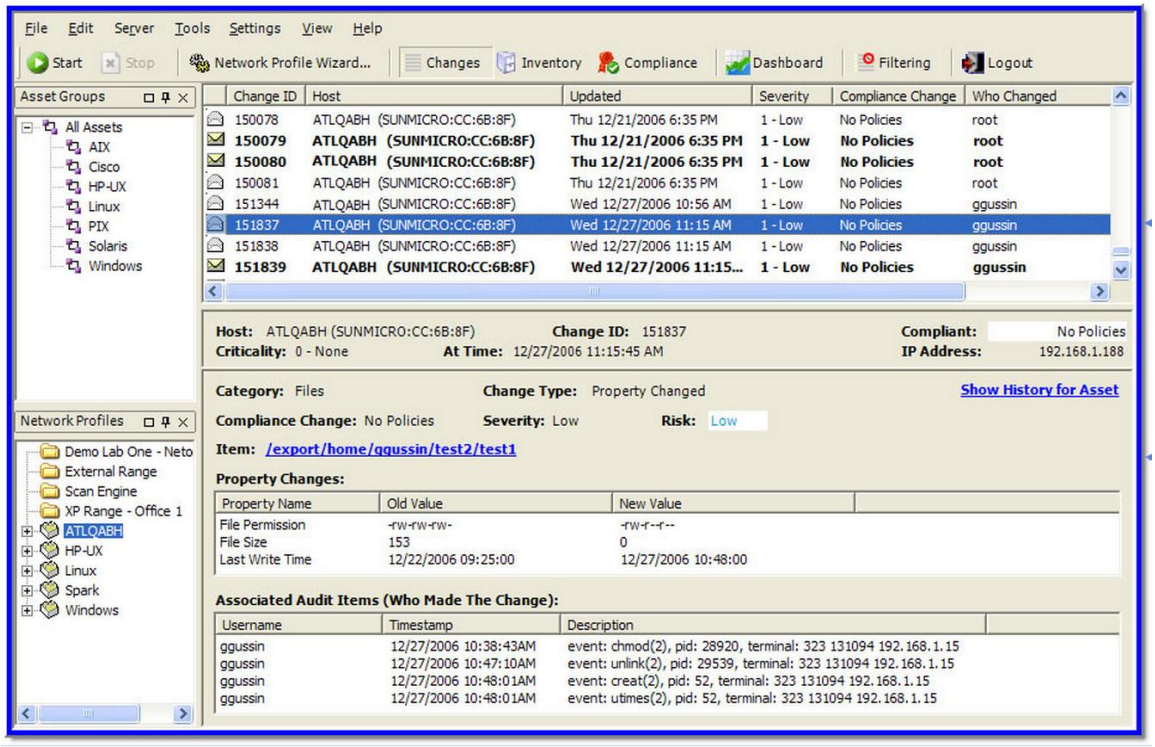

Figure 3.1

There Is Something Very Wrong Here

One bit or one symbol in the output may make the difference between war and peace, friend and foe, compromised and clean system. Lets take a look at example below, what is very wrong in the figure 3.2 screenshot?

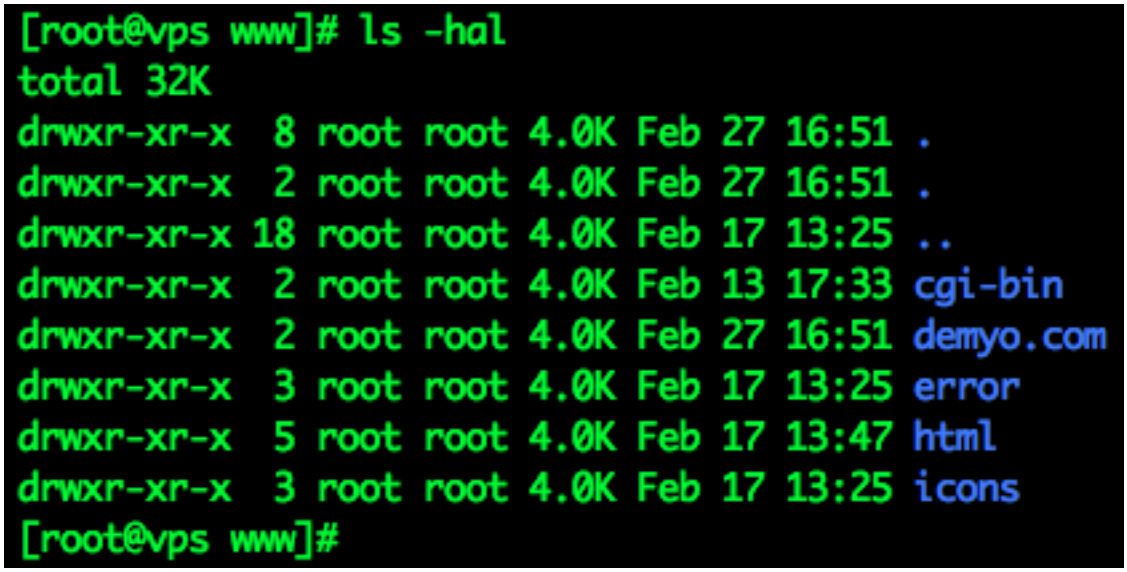

Figure 3.2

For those who don't see the wrong symbol here I will give you a hint. ls is a command to list files in directory, switch –h is for listing output in human readable format, i.e. megabytes will be megabytes and gigabytes will be gigabytes, not 1 073 741 824 bytes. Switch –l makes a list of files, once again to be easier readable by humans. Now we are coming to the main piece of information here, switch –a output will include directory entries whose names begin with a dot (.). A common hacker's technique is to hide within legit file names, or within somewhat legit names. In this case hacker has a directory on the system, which is named '. ' and this is the main issue here. In usual output you should see 1 single dotted directory, in this case we see 2 single dotted directories and it should pop big red flags in your head. We change to this hidden directory by issuing command cd '. '. Just make sure there is a space after dot.

So that's why we want to use ls –hal with switch 'a' all the time, because we want to see hidden directories and hidden files. It is pretty common to have these hidden directories in well known places, such as /root, /var/www, /home and others.

Additional Accounts On The System

Every account on the system should be accounted for. If there are accounts that nobody knows what they belong to that may mean system is compromised. Sometimes IT admins forget to disable old accounts for people who have left company, some of these accounts may be active for months and even years. This is unnecessary risk being introduced by poor IT administrators' management. A good practice is to disable employee's account before exit interview. After compromise hackers make new account on the server and try to mimic some legit accounts that should exist. An example of additional account DBNET is in figure 3.3

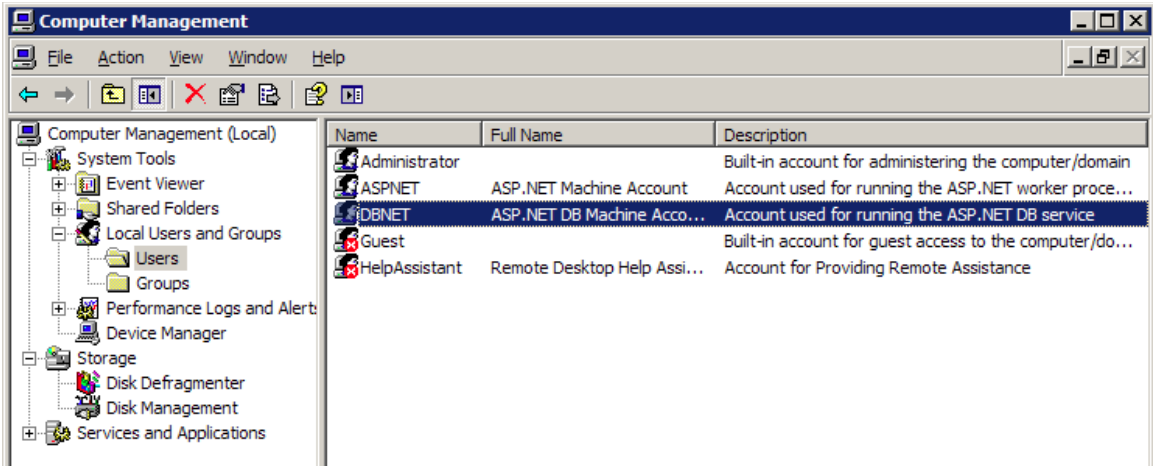

Figure 3.3

Time Stamps

A timestamp is a sequence of characters or encoded information identifying when a certain event occurred, usually giving date and time of day, sometimes accurate to a small fraction of a second. The term derives from rubber stamps used in offices to stamp the current date, and sometimes time, in ink on paper documents, to record when the document was received. A common example of this type of timestamp is a postmark on a letter. However, in modern times usage of the term has expanded to refer to digital date and time information attached to digital data. For example, computer files contain timestamps that tell when the file was last modified, and digital cameras add timestamps to the pictures they take, recording the date and time the picture was taken.

A timestamp is the time at which an event is recorded by a computer, not the time of the event itself. In many cases, the difference may be inconsequential: the time at which an event is recorded by a timestamp (e.g., entered into a log file) should be close to the time of the event.

The sequential numbering of events is sometimes called time stamping.

This data is usually presented in a consistent format, allowing for easy comparison of two different records and tracking progress over time; the practice of recording timestamps in a consistent manner along with the actual data is called time stamping.

Timestamps are typically used for logging events or in a sequence of events (SOE), in which case each event in the log or SOE is marked with a time stamp. In file systems, times tamp may mean the stored date/time of creation or modification of a file [5].

Lets say you have a lot of folders and executable files in C:/Windows/System32 directory, all of them pretty much match OS installation date and time, but there is one folder which does not match OS installation time. Could there be a problem? This executable might be just some additional software installed later on the system, or it also might be malware hiding in this directory. Windows malware just loves this folder! Folder was modified in different month than all others in figure 3.4

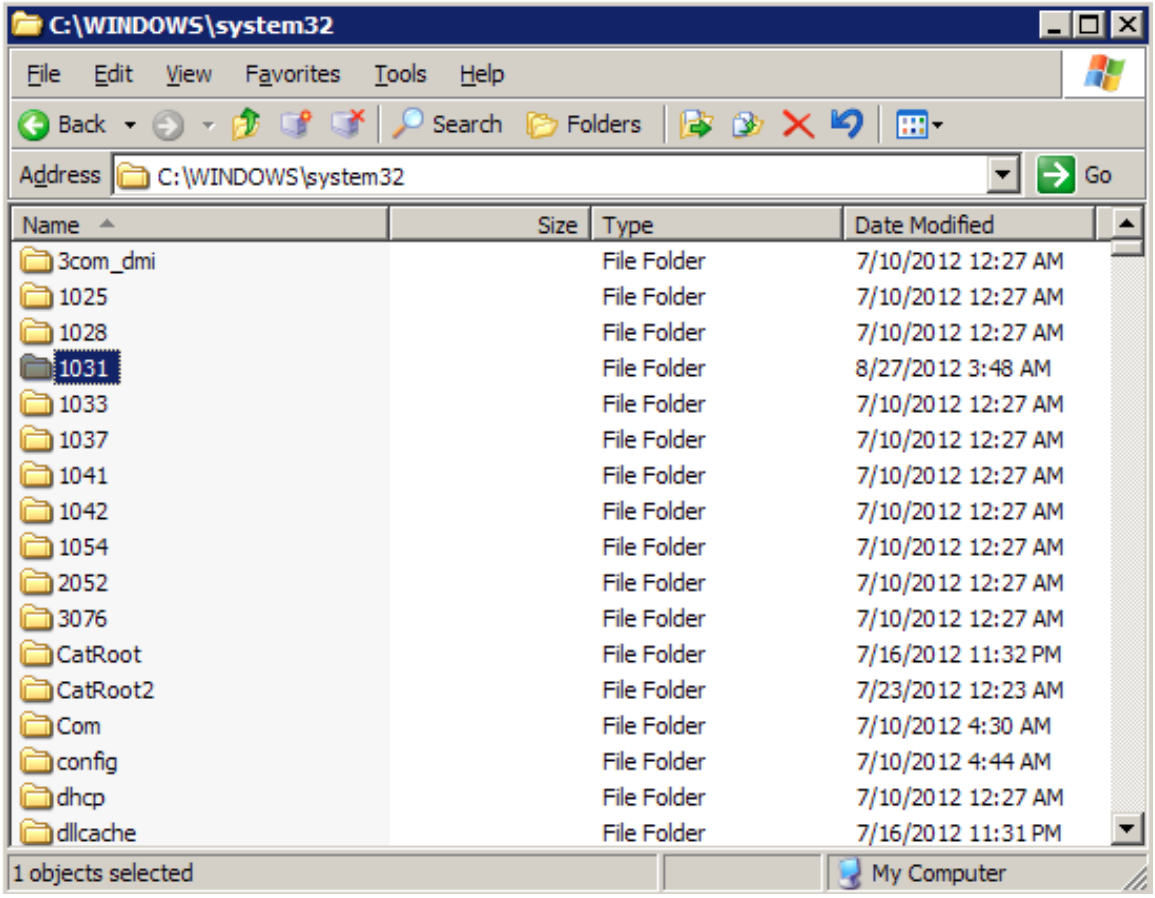

Figure 3.4

Hidden Files And Directories

A hidden file is a file that is not normally visible when examining the contents of the directory in which it resides. Likewise, a hidden directory is a directory that is normally invisible when examining the contents of the directory in which it resides.

A file is a named collection of related information that appears to the user as a single, contiguous block of data and that is retained in storage. Storage refers to computer devices or media that can retain data for relatively long periods of time (e.g., years or decades), such as hard disk drives (HDDs), CDROMs and magnetic tape; this contrasts with memory, which retains data only as long as the data is in use or the memory is connected to a power supply.

A directory (also sometimes referred to as a folder) can be conveniently viewed as a container for files and other directories. In Linux and other Unix-like operating systems, a directory is merely a special type of file that associates file names with a collection of metadata (i.e., data about the files). Likewise, a link is a special type of file that points to another file (which can be a directory). Thus, it is somewhat redundant to use phrases such as hidden files and directories; however, they are descriptive and convenient, and thus they are frequently used. More precise terms are hidden file system objects and hidden items.

Hidden items on Unix-like operating systems are easily distinguishable from regular (i.e., non-hidden) items because their names are prefixed by a period (i.e., a dot). In Unix-like operating systems, periods can appear anywhere within the name of a file, directory or link, and they can appear as many times as desired. However, usually, the only time that they have special significance is when used to indicate a hidden file or directory.

In the Microsoft Windows operating systems, whether a file system object is hidden or not is an attribute of the item, along with such things as whether the file is read-only and a system file (i.e., a file that is critical to the operation of the operating system). Changing the visibility of such items is accomplished using a multi-step procedure.

Unix-like operating systems provide a larger set of attributes for file system objects than do the Microsoft Windows operating systems, including a system of permissions, which control which user(s) have access to each such object for reading, writing and executing. However, whether objects are hidden or not is not among the attributes. Rather, it is merely a superficial property that is easily changed by adding or removing a period from the beginning of the object name.

Many operating systems and application programs routinely hide objects in order to reduce the chances of users accidentally damaging or deleting critical system and configuration files. Hiding objects can also be useful for reducing visual clutter in directories, and thereby making it easier for users to locate desired files and subdirectories.

Another reason to hide file system objects is to make them invisible to casual snoopers. Although it is a very simple matter to make hidden files and directories visible, the great majority of computer users are not even aware that such files and directories exist (nor need they be) [6].

0day Attacks

About 90 percent of all successful compromises are made via known flaws, so 0day attacks are not that common.

A zero-day attack or threat is an attack that exploits a previously unknown vulnerability in a computer application, meaning that the attack occurs on "day zero" of awareness of the vulnerability. This means that the developers have had

zero days to address and patch the vulnerability. 0day exploits (actual software that uses a security hole to carry out an attack) are used or shared by attackers before the developer of the target software knows about the vulnerability.

Attack Vectors

Malware writers are able to exploit zero-day vulnerabilities through several different attack vectors. Web browsers are a particular target because of their widespread distribution and usage. Attackers can also send e-mail attachments, which exploit vulnerabilities in the application opening the attachment. Exploits that take advantage of common file types are listed in databases like US-CERT. Malware can be engineered to take advantage of these file type exploits to compromise attacked systems or steal confidential data such as banking passwords and personal identity information.

Vulnerability Window

Zero-day attacks occur during the vulnerability window that exists in the time between when vulnerability is first exploited and when software developers start to develop and publish a counter to that threat.

For viruses, Trojans and other zero-day attacks, the vulnerability window typically follows this time line:

- The developer creates software containing an unknown vulnerability
- The attacker finds the vulnerability before the developer does
- The attacker writes and distributes an exploit while the vulnerability is not known to the developer
- The developer becomes aware of the vulnerability and starts developing a fix.

Measuring the length of the vulnerability window can be difficult, as attackers do not announce when the vulnerability was first discovered. Developers may not want to distribute data for commercial or security reasons. Developers also may not know if the vulnerability is being exploited when they fix it, and so may not record the vulnerability as a zero-day attack. However, it can be easily shown that this window can be several years long. For example in 2008 Microsoft confirmed vulnerability in Internet Explorer, which affected some versions that were released in 2001. The date the vulnerability was first found by an attacker is not known; however, the vulnerability window in this case could have been up to 7 years.

Discovery

A special type of vulnerability management process focuses on finding and eliminating zero-day weaknesses. This unknown vulnerability management lifecycle is a security and quality assurance process that aims to ensure the security and robustness of both in-house and third party software products by finding and fixing unknown (zero-day) vulnerabilities. The unknown vulnerability management process consists of four phases: analyze, test, report and mitigate.

Analyze: this phase focuses on attack surface analysis
Test: this phase focuses on fuzz testing the identified attack vectors
Report: this phase focuses on reproduction of the found issues to developers
Mitigate: this phase looks at protective measures explained below

Protection

Zero-day protection is the ability to provide protection against zero-day exploits. Zero-day attacks can also remain undetected after they are launched.

Many techniques exist to limit the effectiveness of zero-day memory corruption vulnerabilities, such as buffer overflows. These protection mechanisms exist in contemporary operating systems such as Windows 7, Microsoft Windows Vista, Apple's Mac OS X, recent Oracle Solaris, Linux and possibly other Unix and Unix-like environments; Microsoft Windows XP Service Pack 2 includes limited protection against generic memory corruption vulnerabilities. Desktop and server protection software also exists to mitigate zero day buffer overflow vulnerabilities.

"Multiple layers" provides service-agnostic protection and is the first line of defense should an exploit in any one layer be discovered. An example of this for a particular service is implementing access control lists in the service itself, restricting network access to it via local server firewalling (i.e., IP tables), and then protecting the entire network with a hardware firewall. All three layers provide redundant protection in case a compromise in any one of them occurs.

The use of port knocking or single packet authorization daemons may provide effective protection against zero-day exploits in network services. However these techniques are not suitable for environments with a large number of users.

Whitelisting effectively protects against zero day threats. Whitelisting will only allow known good applications to access a system and so any new or unknown exploits are not allowed access. Although whitelisting is effective against zero-day attacks, an application "known" to be good can in fact have vulnerabilities that were missed in testing. To bolster its protection capability, it is often combined with other methods of protection such as host-based intrusion-prevention system or a blacklist of virus definitions, and it can sometimes be quite restrictive to the user.

Keeping the computer's software up-to-date is very important as well and it does help.

Users need to be careful when clicking on links or opening email attachments with images or PDF files from unknown users. This is how many cyber criminals deceive users, by pretending they are something they are not and gaining the user's trust.

Utilize sites with Secure Socket Layer (SSL), which secures the information being passed between the user and the visited site.

Ethics

Differing views surround the collection and use of zero-day vulnerability information. Many computer security vendors perform research on zero-day vulnerabilities in order to better understand the nature of vulnerabilities and their exploitation by individuals, computer worms and viruses. Alternatively, some vendors purchase vulnerabilities to augment their research capacity. While selling and buying these vulnerabilities is not technically illegal in most parts of the world, there is much controversy over the method of disclosure. A recent German decision to include Article 6 of the Convention on Cybercrime and the EU Framework Decision on Attacks against Information Systems may make selling or even manufacturing vulnerabilities illegal.

Most formal efforts follow some form of disclosure guidelines or the more recent OIS Guidelines for Security Vulnerability Reporting and Response. In general these rules forbid the public disclosure of vulnerabilities without notification to the developer and adequate time to produce a patch [7].

Good Known State

When attackers compromise a system, what is the very first thing they do? They install different backdoors, and as many as possible. So, if some backdoor was found on the system and it was deleted, it does not mean the system is clean. It is much safer to restore the system to a good known state; typically it is done via OS re-installation. Big companies typically have a gold image for their systems. They use gold image to quickly wipe any infected machine and reinstall OS with all its updates, and software at once. On Linux systems the software called System Imager is capable of doing many Linux installations at once.

System Imager is software that makes the installation of Linux to masses of similar machines relatively easy. It makes software distribution, configuration, and operating system updates easy, and can also be used for content distribution [8].

Monitoring Running Processes In The System

What is wrong on the running process list in the following Linux system in figure 3.5?

```
root@bt:~/. # ps aux
USER       PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
root         1  0.0  0.3   2844  1604 ?        Ss   Apr15   0:01 /sbin/init
root         2  0.0  0.0      0     0 ?        S    Apr15   0:00 [kthreadd]
<snip>
root     10962  0.0  0.0   2740   476 ?        S<   09:33   0:00 udevd --daemon
root     11550  0.0  0.0      0     0 ?        S    11:13   0:00 [kworker/0:2]
root     11567  0.0  0.0      0     0 ?        S<   11:15   0:00 [hci0]
root     11619  0.0  0.0      0     0 ?        S    11:18   0:00 [kworker/0:1]
root     11654  0.0  0.0      0     0 ?        S    11:23   0:00 [kworker/0:0]
root     11664  5.3  6.1  36092 31360 pts/1    S    11:24   0:00 ./httpd
root     11665  0.0  0.2   2764  1052 pts/1    R+   11:24   0:00 ps aux
root     12015  0.0  1.7  34800  8736 ?        S    Apr16   0:00 /usr/lib/notification-daemon/notification-daemon
```

Figure 3.5

Process ./httpd should catch a security professional eye. Dot slash at the beginning indicates it was launched locally from the directory. Processes on the servers typically are not launched locally from their directories. Attacker has launched a process and is trying to hide by renaming his software to legit looking software typically found on the server.

Files With Weird Names

Malware frequently make files with weird looking file names, and example in Windows system is in figure 3.6:

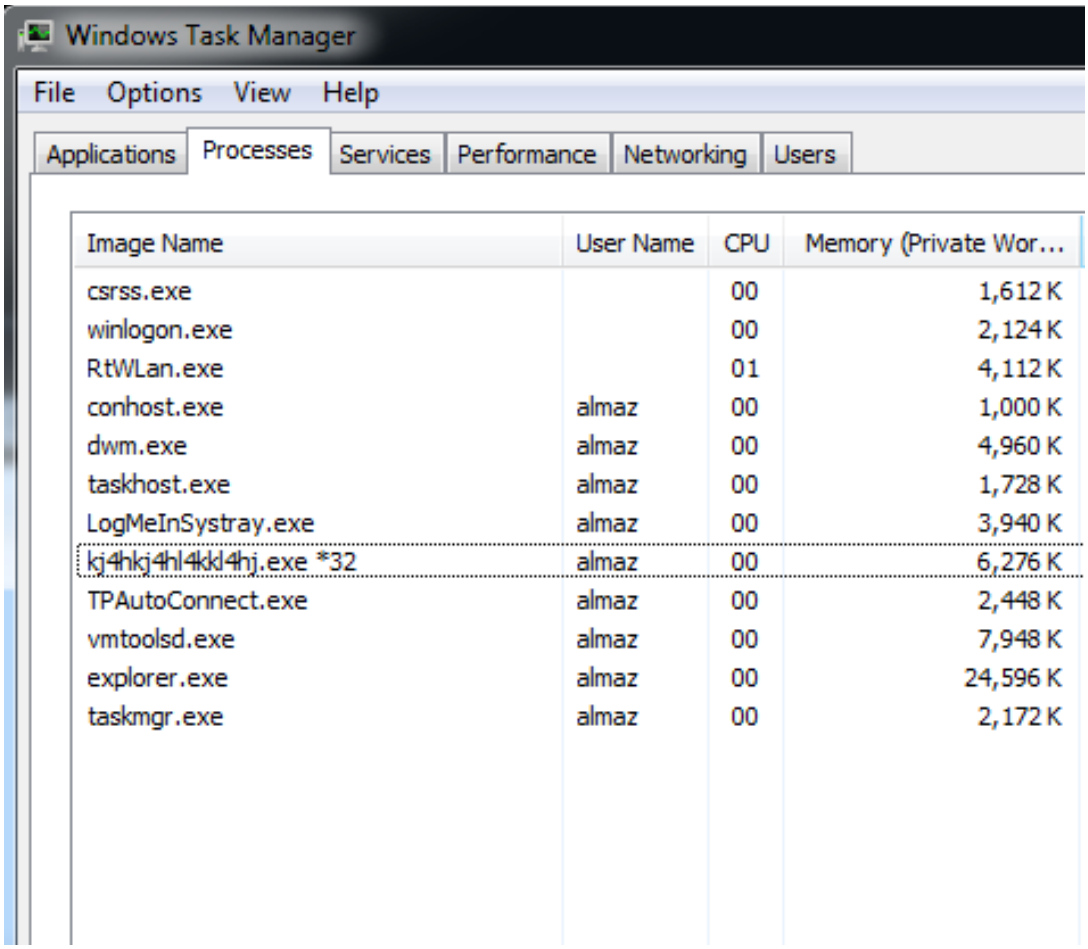

Figure 3.6

We see some file kj4hkj4hl4kkl4hj.exe is running in the memory. This should be a first indicator something funky is going on in the system. Windows updates create random named temporary folders and should not be confused with malware.

Rootkits

A rootkit is a stealthy type of malicious software designed to hide the existence of certain processes or programs from normal methods of detection and enable continued privileged access to a computer. The term rootkit is a concatenation of "root" (the traditional name of the privileged account on Unix operating systems) and the word "kit" (which refers to the software components that implement the tool). The term "rootkit" has negative connotations through its association with malware.

Rootkit installation can be automated, or an attacker can install it once they've obtained root or Administrator access. Obtaining this access is either a result of direct attack on a system (i.e. exploiting a known vulnerability, password (either by cracking, privilege escalation, or social engineering)). Once installed it becomes

possible to hide the intrusion as well as to maintain privileged access. Like any software they can have a good purpose or a malicious purpose. The key is the root/administrator access. Full control over a system means that existing software can be modified, including software that might otherwise be used to detect or circumvent it.

Rootkit detection is difficult because a rootkit may be able to subvert the software that is intended to find it. Detection methods include using an alternative and trusted operating system, behavioral-based methods, signature scanning, difference scanning, and memory dump analysis. Removal can be complicated or practically impossible, especially in cases where the rootkit resides in the kernel; reinstallation of the operating system may be the only available solution to the problem. When dealing with firmware rootkits, removal may require hardware replacement, or specialized equipment. [9]

Kernel Level Rootkits

Kernel-mode rootkits run with the highest operating system privileges (Ring 0) by adding code or replacing portions of the core operating system, including both the kernel and associated device drivers. Most operating systems support kernel-mode device drivers, which execute with the same privileges as the operating system itself. As such, many kernel-mode rootkits are developed as device drivers or loadable modules, such as loadable kernel modules in Linux or device drivers in Microsoft Windows. This class of rootkit has unrestricted security access, but is more difficult to write. The complexity makes bugs common, and any bugs in code operating at the kernel level may seriously impact system stability, leading to discovery of the rootkit. One of the first widely known kernel rootkits was developed for Windows NT 4.0 and released in Phrack magazine in 1999 [10].

Kernel rootkits can be especially difficult to detect and remove because they operate at the same security level as the operating system itself, and are thus able to intercept or subvert the most trusted operating system operations. Any software, such as antivirus software, running on the compromised system is equally vulnerable. In this situation, no part of the system can be trusted.

A rootkit can modify data structures in the Windows kernel using a method known as direct kernel object modification (DKOM). This method can hook kernel functions in the System Service Descriptor Table (SSDT), or modify the gates between user mode and kernel mode, in order to cloak itself. Similarly for the Linux operating system, a rootkit can modify the system call table to subvert kernel functionality. It's not uncommon for a rootkit to create a hidden, encrypted file system in which it can hide other malware or original copies of files it has infected.

Operating systems are evolving to counter the threat of kernel-mode rootkits. For example, 64-bit editions of Microsoft Windows now implement mandatory signing of all kernel-level drivers in order to make it more difficult for untrusted code to execute with the highest privileges in a system.

Userland Rootkits

User-mode rootkits run in ring 3, along with other applications as user, rather than low-level system processes. They have a number of possible installation vectors to intercept and modify the standard behavior of application programming interfaces (APIs). Some inject a dynamically linked library (such as a .dll file on Windows, or a .dylib file on Mac OS X) into other processes, and are thereby able to execute inside any target process to spoof it; others with sufficient privileges simply overwrite the memory of a target application. Injection mechanisms include:

- Use of vendor-supplied application extensions. For example, Windows Explorer has public interfaces that allow third parties to extend its functionality
- Interception of messages
- Debuggers
- Exploitation of security vulnerabilities
- Function hooking or patching of commonly used APIs, for example, to mask a running process or file that resides on a file system

Rootkit Detection

There are a lot of software for rootkit searches meant to be run on live system. One of many examples would be software called “rootkit hunter” in figure 3.7 [11].

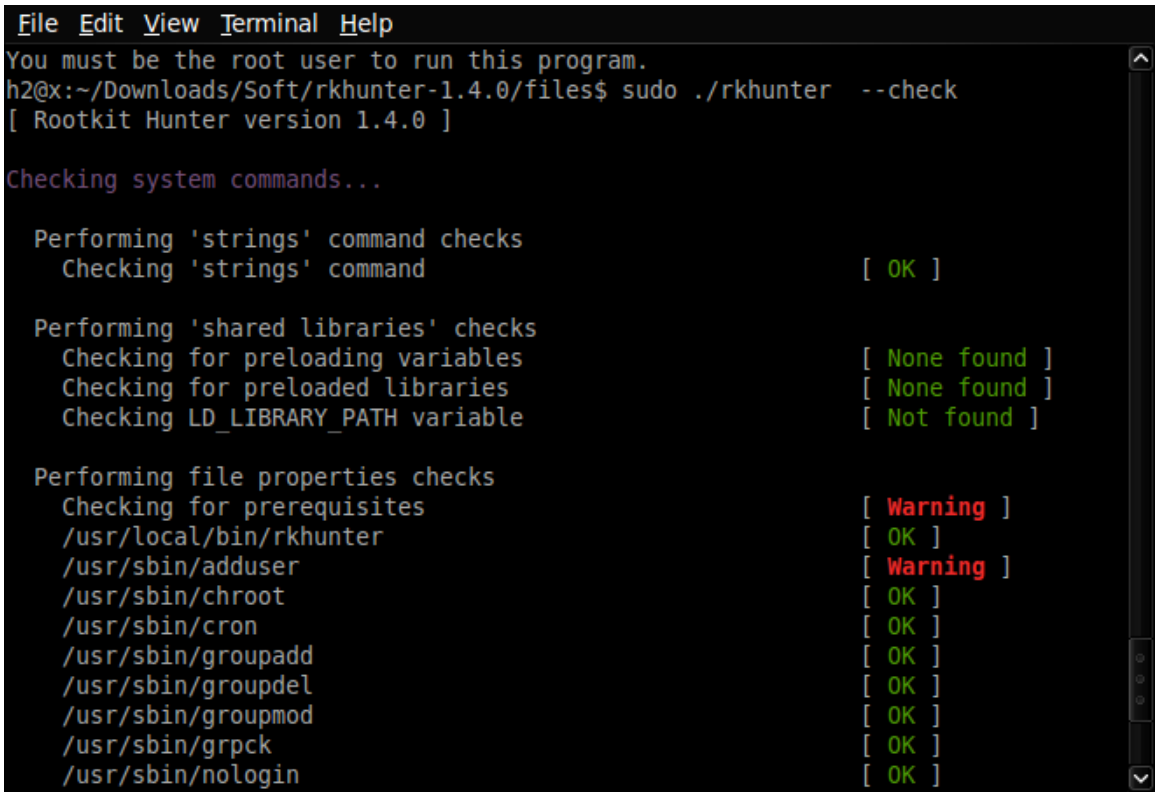

Figure 3.7

Low Hanging Fruit

Do you have to run faster than bear? Not necessarily, you just have to be running faster than your friend, so he will be eaten and not you. Do your systems have to be as secure as Pentagon computers with myriad of controls? Not necessarily, your system have to be more secure than your neighbor's and hopefully you will avoid trouble. Some other techniques to deter intrusions:

- Deterring intrusions by snow flaking (no two snowflakes are the same, so it takes more time to analyze particular system in order to gain access. Making them useless to be scanned with automatic tools). Example would be to move SSH port from default TCP/22 to TCP/31234. Some determined hacker will find it out pretty soon, but it will be an extra step for a script kiddie.
- Low hanging fruit is attacked most of the time, simply ignoring pings to the host will deter some hackers, as there are many more systems that reply to ping and it takes much less time to detect those live IPs and scan them for vulnerabilities [12].

Antivirus Software

The biggest fear for malware is antivirus engine on the system. Antivirus can detect attack, but it might be too late already. AV is based on signatures in the files. Hackers bypass signature detection by encrypting their executables in unique ways. Every executable is encrypted in unique way and AV engines are always losing by being late into the game of detection. If your AV engine fires – that means malware managed to slip by your IDS/IPS solution into the network and/or system.

Homegrown Intrusion Detection.

In order to defeat a hacker you have to think as a hacker. Lets take a look what is a robots.txt file in web server. This file sits in the root of a web page, for example www.mywebpage.com/robots.txt and provides information to search engines what should be cached, what should be skipped, how frequently crawling has to be done, etc. Lets say you have sensitive files in directory called "reports". This directory can be excluded from search engines crawlers and will not end up in search results. Other files and directories such as /private/, /adminpanel/, /phpmyadmin/ should be excluded from search engine results. This technique looks great so far, but a little more experienced attacker will take a look at robots.txt file and see what you don't want him to know!

Incorrect robots.txt implementation	**Correct robots.txt implementation**
Disallow: /adminpanel/ Disallow: /phpmyadmin/ Disallow: /backup/ Disallow: /uploads/	Move all sensitive directories into one directory called for example /private/ and disallow this directory: Disallow: /private/

A little customized robots.txt file would look like this:

```
User-Agent: *
Disallow: /private/
Allow: /
User-Agent: hacker
Disallow: /please/go/to/an/easier/target/
```

It would give attacker some clue that this is probably not the easiest target, and hopefully he will move to an easier one. Needles to say it will not push away targeted attack [13]. So, if you have somebody trying to access non existing directory "/please/go/to/an/easier/target/" on the server it should give you a clue who is interested in your website.

Full Packet Capture Devices

Sometimes it is easier to detect intrusion on the wire, i.e. by monitoring ingress and egress traffic. We have to be aware of out of band communications, for example communication that come to the corporate network via GSM signals. These communications do not go through border routers of the company, and thus cannot be inspected via this technology.

Packet capture appliance is a standalone device that performs packet capture. Packet capture appliances may be deployed anywhere on a network, however, most commonly are placed at the entrances to the network (i.e. the internet connections) and in front of critical equipment, such as servers containing sensitive information.

In general, packet capture appliances capture and record all network packets in full (both header and payload), however, some appliances may be configured to capture a subset of a network's traffic based on user-definable filters. For many applications, especially network forensics and incident response, it is critical to conduct full packet capture, though filtered packet capture may be used at times for specific, limited information gathering purposes.

Deployment

The network data that a packet capture appliance captures depends on where and how the appliance is installed on a network. There are two options for deploying packet capture appliances on a network. One option is to connect the appliance to the SPAN port (port mirroring) on a network switch or router. A second option is to connect the appliance inline, so that network activity along a network route traverses the appliance (similar in configuration to a network tap, but the information is captured and stored by the packet capture appliance rather than passing on to another device).

When connected via a SPAN port, the packet capture appliance may receive and record all Ethernet/IP activity for all of the ports of the switch or router.

When connected inline, the packet capture appliances captures only the network traffic traveling between two points, that is, traffic that passes through the cable to which the packet capture appliance is connected.

There are two general approaches to deploying packet capture appliances: centralized and decentralized.

Centralized
With a centralized approach, one high-capacity, high-speed packet capture appliance connects to data-aggregation point. The advantage of a centralized approach is that with one appliance you gain visibility over the network's entire traffic. This approach, however, creates a single point of failure that is a very attractive target for hackers; additionally, one would have to re-engineer the network to bring traffic to appliance and this approach typically involves high costs.

Decentralized
With a decentralized approach you place multiple appliances around the network, starting at the point(s) of entry and proceeding downstream to deeper network segments, such as workgroups. The advantages include: no network re-configuration required; ease of deployment; multiple vantage points for incident response investigations; scalability; no single point of failure – if one fails, you have the others; if combined with electronic invisibility, this approach practically eliminates the danger of unauthorized access by hackers; low cost. Cons: potential increased maintenance of multiple appliances.

In the past, packet capture appliances were sparingly deployed, oftentimes only at the point of entry into a network. Packet capture appliances can now be deployed more effectively at various points around the network. When conducting incident response, the ability to see the network data flow from various vantage points is indispensable in reducing time to resolution and narrowing down which parts of the network ultimately were affected. By placing packet capture appliances at the entry point and in front of each work group, following the path of a particular transmission deeper into the network would be simplified and much quicker. Additionally, the appliances placed in front of the workgroups would show intranet transmissions that the appliance located at the entry point would not be able to capture.

Capacity
Packet capture appliances come with capacities ranging from 500 GB to 32 TB and more. Only a few organizations with extremely high network usage would have use for the upper ranges of capacities. Most organizations would be well served with capacities from 1 TB to 4 TB.

A good rule of thumb when choosing capacity is to allow 1 GB per day for heavy users down to 1 GB per month for regular users. For a typical office of 20 people with average usage, 1 TB would be sufficient for about 1 to 4 years.

Features

Filtered vs. Full Packet Capture

Full packet capture appliances capture and record all Ethernet/IP activity, while filtered packet capture appliances captured only a subset of traffic, based on a set of user-definable filters, such as IP address, MAC address or protocol. Unless using the packet capture appliance for a very specific, narrow purpose covered by the filter parameters, it is generally best to use full packet capture appliances or otherwise risk missing vital data. Particularly when using a packet capture for network forensics or cyber security purposes, it is paramount to capture everything because any packet not captured on the spot is a packet that is gone forever. It is impossible to know ahead of time the specific characteristics of the packets or transmissions needed, especially in the case of an advanced persistent threat (APT). APTs and other hacking techniques rely for success on network administrators not knowing how they work and thus not having solutions in place to counteract them. Most APT attacks originate from Russian and China.

Encrypted vs. Unencrypted Storage

Some packet capture appliances encrypt the captured data before saving it to disk, while others do not. Considering the breadth of information that travels on a network or Internet connection and that at least a portion of it could be considered sensitive, encryption is a good idea for most situations as a measure to keep the captured data secure. Encryption is also a critical element of authentication of data for the purposes of data/network forensics.

Sustained Capture Speed vs. Peak Capture Speed

The sustained captured speed is the rate at which a packet capture appliance can capture and record packets without interruption or error over a long period of time. This is different from the peak capture rate, which is the highest speed at which a packet capture appliance can capture and record packets. The peak capture speed can only be maintained for short period of time, until the appliance's buffers fill up and it starts losing packets. Many packet capture appliances share the same peak capture speed of 1 Gbps, but actual sustained speeds vary significantly from model to model.

Permanent vs. Overwritable Storage

A packet capture appliance with permanent storage is ideal for network forensics and permanent record-keeping purposes because the data captured cannot be overwritten, altered or deleted. The only drawback of permanent storage is that eventually the appliance becomes full and requires replacement. Packet capture appliances with overwritable storage are easier to manage because once they reach capacity they will start overwriting the oldest captured data with the new, however, network administrators run the risk of losing important capture data when it gets overwritten. In general, packet capture appliances with overwrite capabilities are useful for simple monitoring or testing purposes, for which a permanent record is not necessary. Permanent recording is a must for network forensics information gathering.

Data Security

Since packet capture appliances capture and store a large amount of data on network activity, including files, emails and other communications, they could, in themselves, become attractive targets for hacking. A packet capture appliance deployed for any length of time should incorporate security features, to protect the recorded network data from access by unauthorized parties. If deploying a packet capture appliance introduces too many additional concerns about security, the cost of securing it may outweigh the benefits. The best approach would be for the packet capture appliance to have built-in security features. These security features may include encryption, or methods to "hide" the appliance's presence on the network. For example, some packet capture appliances feature "electronic invisibility", that is, have a stealthy network profile by not requiring or using IP nor MAC addresses.

Though on the face of it connecting a packet capture appliance via a SPAN port appears to make it more secure, the packet capture appliance would ultimately still have to be connected to the network in order to allow management and data retrieval. Though not accessible via the SPAN link, the appliance would be accessible via the management link.

Despite the benefits, a packet capture appliance's remote access feature presents a security issue that could make the appliance vulnerable. Packet capture appliances that allow remote access should have a robust system in place to protect it against unauthorized access. One way to accomplish this is to incorporate a manual disable, such as a switch or toggle that allows the user to physically disable remote access. This simple solution is very effective, as it is doubtful that a hacker would have an easy time gaining physical access to the appliance in order to flip a switch.

A final consideration is physical security. All the network security features in the world are moot if someone is simply able to steal the packet capture appliance or make a copy of it and have ready access to the data stored on it. Encryption is one of the best ways to address this concern, though some packet capture appliances also feature tamperproof enclosures [14].

Out Of Band Attack Vectors

What is the weakest link in any corporation? The answer is people. People fall into social engineering attacks; people bring "forgotten" USB sticks and CDs from bathrooms/parking lots and plug them into their computers just out of curiosity. People bring their own devices from home and connect to corporate networks. BYOD or Bring Your Own Device is a big pain for IT administrators to manage. It also introduces additional risk, because employee's own devices might already be backdoored or infected and by connecting these devices to corporate network employees are introducing a new risk. Social engineering attack with lost CD - figure 3.8

Figure 3.8

Demyo power strip is a full-blown Linux based OS with many penetration testing tools preinstalled, it looks like innocent power surge/strip, but has Wi-Fi, Ethernet and Bluetooth installed inside. Once connected to the power outlet it immediately calls back home via GSM 3g modem and establishes connection. Once connected penetration testers can use it as a jump box to do further penetration testing inside the LAN of the corporation [15]. Demyo power strip is shown in figure 3.9

Figure 3.9

How to prevent employees bringing “lost CDs” and “lost USB sticks” from parking lots and plugging them into their machines? A strong policy should be in place disallowing connecting non-approved hardware to workstations. It is not enough just to write a policy and consider the job to be done. Policy has to be enforced and most importantly policy has to be understood by employees. There is no way rules can be followed if they are not understood. Another way to minimize risk is to provide security awareness training to employees explaining typical social engineering attacks and how not to fall for them.

Security Awareness Training

Security awareness is the knowledge and attitude members of an organization possess regarding the protection of the physical and, especially, information assets of that organization. Many organizations require formal security awareness training for all workers when they join the organization and periodically thereafter, usually annually.

Topics covered in security awareness training include:

The nature of sensitive material and physical assets they may come in contact with, such as trade secrets, privacy concerns and government classified information
Employee and contractor responsibilities in handling sensitive information, including review of employee nondisclosure agreements
Requirements for proper handling of sensitive material in physical form, including marking, transmission, storage and destruction
Proper methods for protecting sensitive information on computer systems, including password policy and use of two-factor authentication

Other computer security concerns, including malware, phishing, social engineering, etc.
Workplace security, including building access, wearing of security badges, reporting of incidents, forbidden articles, etc.
Consequences of failure to properly protect information, including potential loss of employment, economic consequences to the firm, damage to individuals whose private records are divulged, and possible civil and criminal penalties
Being security aware means you understand that there is the potential for some people to deliberately or accidentally steal, damage, or misuse the data that is stored within a company's computer systems and throughout its organization. Therefore, it would be prudent to support the assets of the institution (information, physical, and personal) by trying to stop that from happening.

According to the European Network and Information Security Agency, 'Awareness of the risks and available safeguards is the first line of defense for the security of information systems and networks.'

'The focus of Security Awareness consultancy should be to achieve a long-term shift in the attitude of employees towards security, whilst promoting a cultural and behavioral change within an organization. Security policies should be viewed as key enablers for the organization, not as a series of rules restricting the efficient working of your business. '[16]

Data Correlation
Data correlation is a technique used in information security to put all pieces together and come up with some meaningful information. For example if you see in Linux system SSH connections coming in all day long, and after 200 tries to login in there is a successful login after all. What does it tell you? It should be a good starting point to suggest a brute force attack is going on with a success at the end. All technologies help to find out intrusions, however technologies do not find intrusions, people do. Appliances and sensors are typically good about finding bad events, but good events can combine into bad one as well. How is it possible you would ask? Lets outline a simple scenario where human makes determination about compromise.

Lets say there is a company with many employees which travel a lot around the globe. Company is doing a good job by implementing various control systems, various logging systems, this company also uses RFID enabled cards for its employees in order to track who is coming and leaving its offices. All data is collected and pushed to SIEM [17] engine to do correlation between events and logs. One morning 2 seemingly good events come into SIEM. First event is user john VPN connection is established from overseas to corporate office. Second event is user john RFID badge being scanned at the entrance to the corporate office. Well both events are pretty standard and are harmless when taken separately, but then combined together they reveal something weird. How can user john VPN in from overseas and get a physical entrance to the office at the same time? The answer is

one of two: either VPN credentials are compromised, or his employee card is used by somebody else to enter the office. Figure 3.10 shows how 2 good things can create 1 bad thing when combined.

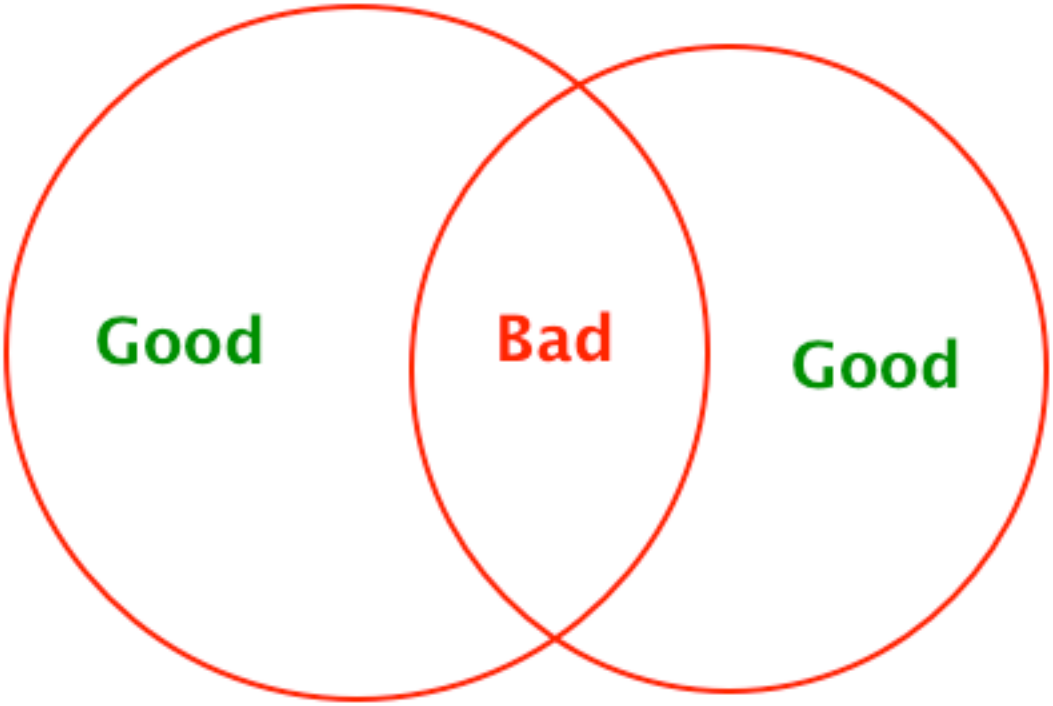

Figure 3.10

SIEM

Security Information and Event Management (SIEM) solutions are a combination of the formerly disparate product categories of SIM (security information management) and SEM (security event manager). SIEM technology provides real-time analysis of security alerts generated by network hardware and applications. SIEM solutions come as software, appliances or managed services, and are also used to log security data and generate reports for compliance purposes.

The acronyms SEM, SIM and SIEM have been used interchangeably, though there are differences in meaning and product capabilities. The segment of security management that deals with real-time monitoring, correlation of events, notifications and console views is commonly known as Security Event Management (SEM). The second area provides long-term storage, analysis and reporting of log data and is known as Security Information Management (SIM).

The term Security Information Event Management (SIEM), describes the product capabilities of gathering, analyzing and presenting information from network and security devices; identity and access management applications; vulnerability

management and policy compliance tools; operating system, database and application logs; and external threat data. A key focus is to monitor and help manage user and service privileges, directory services and other system configuration changes; as well as providing log auditing and review and incident response.

SIEM Capabilities

- Data Aggregation: SIEM/LM (log management) solutions aggregate data from many sources, including network, security, servers, databases, applications, providing the ability to consolidate monitored data to help avoid missing crucial events.
- Correlation: looks for common attributes, and links events together into meaningful bundles. This technology provides the ability to perform a variety of correlation techniques to integrate different sources, in order to turn data into useful information.
- Alerting: the automated analysis of correlated events and production of alerts, to notify recipients of immediate issues.
- Dashboards: SIEM/LM tools take event data and turn it into informational charts to assist in seeing patterns, or identifying activity that is not forming a standard pattern.
- Compliance: SIEM applications can be employed to automate the gathering of compliance data, producing reports that adapt to existing security, governance and auditing processes.
- Retention: SIEM/SIM solutions employ long-term storage of historical data to facilitate correlation of data over time, and to provide the retention necessary for compliance requirements.

Other Weird Stuff On The System

What are other symptoms of possible system compromise? Some examples below:

- Log files are missing completely. Why there are no log files?

Script kiddies delete logs whereas hackers modify them by taking out only their IP addresses, their commands and manipulations with system.

- Network interface is in promiscuous mode

In computer networking, promiscuous mode is a mode for a wired network interface controller (NIC) or wireless network interface controller (WNIC) that causes the controller to pass all traffic it receives to the central processing unit (CPU) rather than passing only the frames that the controller is intended to receive. This mode is normally used for packet sniffing that takes place on a router or on a computer connected to a hub (instead of a switch) or one being part of a WLAN. The mode is also required for bridged networking for hardware virtualization.

In IEEE 802 networks such as Ethernet, token ring, and IEEE 802.11, and in FDDI, each frame includes a destination Media Access Control address (MAC address). In non-promiscuous mode, when a NIC receives a frame, it normally drops it unless the frame is addressed to that NIC's MAC address or is a broadcast or multicast frame. In promiscuous mode, however, the card allows all frames through, thus allowing the computer to read frames intended for other machines or network devices.

Many operating systems require super user privileges to enable promiscuous mode. A non-routing node in promiscuous mode can generally only monitor traffic to and from other nodes within the same broadcast domain (for Ethernet and IEEE 802.11) or ring (for token ring or FDDI). Computers attached to the same network hub satisfy this requirement, which is why network switches are used to combat malicious use of promiscuous mode. A router may monitor all traffic that it routes.

Promiscuous mode is often used to diagnose network connectivity issues. There are programs that make use of this feature to show the user all the data being transferred over the network. Some protocols like FTP and Telnet transfer data and passwords in clear text, without encryption, and network scanners can see this data. Therefore, computer users are encouraged to stay away from insecure protocols like telnet and use more secure ones such as SSH.

Detection

As promiscuous mode can be used in a malicious way to sniff on a network, one might be interested in detecting network devices that are in promiscuous mode. In promiscuous mode, some software might send responses to frames even though they were addressed to another machine. However, experienced sniffers can prevent this (e.g., using carefully designed firewall settings).

An example is sending a ping (ICMP echo request) with the wrong MAC address but the right IP address. If an adapter is operating in normal mode, it will drop this frame, and the IP stack never sees or responds to it. If the adapter is in promiscuous mode, the frame will be passed on, and the IP stack on the machine (to which a MAC address has no meaning) will respond as it would to any other ping. The sniffer can prevent this by configuring his firewall to block ICMP traffic [18].

- Immutable files on the system that cannot be deleted, find those with lsattr command

lsattr is a command-line program for listing the attributes on a Linux second extended file system. It is also a command to display attributes of devices on an AIX operating system. Some malware puts +i flag on its own executable, so you cannot delete it, even if you are root.

- Mysterious open ports and services

All open ports and running services should be accounted for. For example if there is a service running, but its not clear what it does, or why is it running – an investigation should be launched [19].

Summary

As we outlined above there are so many ways to detect system intrusions and so many ways to hide them. What is the proper way to analyze suspect system then?

The proper sequence is:

1. Memory dump and analysis. Hackers are getting smart these days; they stay in memory as long as possible. Why? Because they know forensics will be done on the HDD itself, but if they stay in memory it requires better skill to do memory

analysis. Some companies just pull the plug from the power and network and do HDD forensics analysis. This is wrong, because as soon as you pull the power plug – half of the goodies are gone…

2. Selective HDD files analysis (we make HDD image first, and work from the copy). Depending on the machine role on the network it might be an overkill to do full blown forensic analysis. In some situations partial forensic examination is enough.

3. Full HDD analysis if needed (we make HDD image first, and work from the copy)

References

1.Whitelisting vs blacklisting - http://bit.ly/RNxEHO
2.LoggedFS - http://loggedfs.sourceforge.net/
3.File Integrity Monitoring - https://en.wikipedia.org/wiki/File_integrity_monitoring
4.AIDE - http://aide.sourceforge.net/
5.Timestamps - https://en.wikipedia.org/wiki/Timestamp
6.Hidden files - http://www.linfo.org/hidden_file.html
7.0day attacks - https://en.wikipedia.org/wiki/Zero-day_attack
8.SystemImager - http://sourceforge.net/projects/systemimager/
9.Rootkit - https://en.wikipedia.org/wiki/Rootkit
10.Phrack - http://phrack.org/
11.Rootkit hunter - http://rkhunter.sourceforge.net/
12.What is vulnerability - http://bit.ly/PFCWCh
13.Targeted attack - http://bit.ly/MTjLVv
14.Full Packet Capture - https://en.wikipedia.org/wiki/Packet_Capture_Appliance
15.Demyo power strip – http://www.demyo.com
16.Security Awareness - https://en.wikipedia.org/wiki/Security_awareness
17.SIEM - https://en.wikipedia.org/wiki/Siem
18.Promiscuous mode - https://en.wikipedia.org/wiki/Promiscuous_mode
19.Intrusion Detection - http://bit.ly/OCB7UU