Interested in learning more about security?

SANS Institute InfoSec Reading Room

Linux RootKits For Beginners - From Prevention to Removal

One day while reading a mail list for the Linux Users Group in my hometown I discovered a call for help. It was a posting from a novice Linux user with a disturbing issue. While doing some routine checks on a Linux system, he found a user that had been added to the system with the user id of 0 (root). His first thought was that it might be a rootkit. He wanted to know what he could do to verify it was a rootkit and how to remove it from the system. He further asked for suggestions on preventative measures to ensure thi...

Linux RootKits For Beginners – From Prevention to Removal

Jeromey Hannel

GIAC Security Essentials Certification
Practical Assignment Version 1.4b

January 23, 2003

Linux RootKits For Beginners – From Prevention to Removal

Introduction

One day while reading a mail list for the Linux Users Group in my hometown I discovered a call for help. It was a posting from a novice Linux user with a disturbing issue.

While doing some routine checks on a Linux system, he found a user that had been added to the system with the user id of 0 (root). His first thought was that it might be a rootkit. He wanted to know what he could do to verify it was a rootkit and how to remove it from the system. He further asked for suggestions on preventative measures to ensure this kind of attack does not reoccur.

That situation prompted me to write this paper to an understanding of rootkits and its effects. This paper will also discuss how to monitor for a rootkit, and the steps that need to be taken to remove one.

What is a Rootkit?

According to http://www.whatis.com: "A rootkit is a collection of tools (programs) that a hacker uses to mask intrusion and obtain administrator-level access to a computer or computer network".

Rootkits are commonly written for variations of the Unix operating systems including Linux, BSD, and SunOs, but have also been written for the Windows operating systems as well. This paper will focus on rootkits written for the Linux operating systems.

There are many different versions of rootkits that perform basically the same function. Well known Linux rootkits include LRK, tOrn, and Adore and some Windows Rootkits include NTROOT, NTKap, and Nullsys. More information as well as copies of these files can be found on Packet Storm's site.

http://packetstormsecurity.nl/UNIX/penetration/rootkits/

Not only are rootkits designed to hide the presence of an attacker; they are also used to gain future administrator-level (root) access, launch distributed denial of service (ddos), or obtain financial or confidential information.

Because rootkits are designed to hide the presence of an attacker, it is necessary to understand how a rootkit functions.

When a rootkit is installed, it overwrites many commands used on a daily basis such as *ls*, *ps*, or *netstat*. By overwriting such commands, the intrusion can be masked from the administrators.

Take the following scenario; an intruder just gained access to a system through a recently published vulnerability in an un-patched application. The intruder then launches the rootkit installer. This rootkit just over wrote the ls command. By doing this, any time the trojaned ls command is ran, all files installed by the rootkit are hidden from view. This rootkit also installed a trojaned version of the ps command ultimately hiding the sniffer, backdoor, and log parsing processes from view. The ls and ps commands are not the only commands that can get over written by the rootkit. The following is a list of common commands that can be overwritten by a rootkit.

netstat: A useful tool used to display information about current network connections, routing tables, and interface statistics. Netstat can be altered by rootkits to hide the connections made by the intruder to and from the system.

du: A command used to display file space usage. Much like the ls command, du makes hierarchical directory tree but displays how much disk space each file or directory is using. The du command is trojaned mainly to hide files and directories installed by the rootkit.

find: Used to find files in a directory hierarchy. By altering the find command, intruders make it harder for administrators to search for known files installed by rootkits. Much like the ls and du commands, find is trojaned to hide the presence of rootkit files.

ifconfig: Used to configure and display information about network interfaces. If a sniffer is installed and running, the network interface is placed in promiscuous mode. Placing an interface in promiscuous mode enables the network interface to intercept and read packets on the network. Ifconfig is most commonly altered to conceal the evidence of an interface in promiscuous mode thus hiding the presence of a sniffer or password grabber.

inetd (xinetd): A super server designed to start programs that provide Internet services. (x)inetd then spawns the appropriate server to accept the connections. Many rootkits add their applications to the configuration file causing rootkit services to be spawned when a specific port is accessed. This is done to hide the process from administrators until the attacker calls it.

killall: A command used to stop processes. Killall is trojaned in most rootkits so administrators cannot stop certain processes that have been installed by the rootkit.

login: A daemon that is used when signing onto a system. The login daemon can be modified to document all usernames and passwords typed into the system. This documented list can be saved to a directory to be accessed for later use, sent to another system, or displayed on an alternative source such as a chat server or news group.

lsof: A command that is used to list open files. The lsof command is overwritten to hide any file or process that is open by the rootkit.

Not all rootkits were designed to overwrite commands. Rootkits such as Adore Knark were designed as a loadable kernel modules or LKM for short. LKM rootkits take advantage of system calls changing the behavior of the command without actually modifying the command. In laymen's terms, a system call is the way a user level application asks the operating system to perform a function for it. The system call table contains a list of each of the system calls.

Since rootkits are designed to hide their presence, any command affected by any rootkit should still function as it did before the intrusion.

Rootkit Prevention

Rootkit prevention as well as preventing any other type of attack starts at the basics of system security. A basic system security plan should include items such as firewalls, VPN's, as well as updating applications with vendor patches. By implementing basic security measures, administrators can assure a relatively secure network.

Security Basics

Firewall all networks.
It is always a good practice to make sure all networks are protected from the Internet by using a firewall. Keep in mind that a firewall is not the only preventative measure an administrator can take to deter intruders.

Know exactly what is running on all systems.
After a system is installed, take inventory of what is running. Turn off any services that are not needed. Perform periodic audits of system processes to ensure that no unauthorized applications are running.

Grant access to users that are needed to perform their jobs.
Never give a user more access to network services than he or she needs. Only grant permission to services that the user needs to perform his or her job and nothing more.

Enable secure communications such as VPNs and Secure Shell.
By implementing VPN's, administrators ensure that all data transmitted across Wide Area Network(s) (WAN) are encrypted. Secure Shell or SSH is an alternative to the telnet protocol. SSH encrypts all data transmitted during a session including usernames and password.

Keep up to date on all vendor updates.
Security and bug patches are frequently released for operating systems as well as applications. Keep systems up to date with the latest vendor releases that resolve security issues. Many attackers gain administrative level access by exploiting vulnerabilities in applications used everyday. It is good practice to belong to newsletters such as CERT or BUGTRAQ. These types of newsletters inform administrators of recent vulnerabilities, what systems are affected and where to go for patches.

Install host and network based intrusion detection systems.
Knowing what is going on within your network is very important. If an intruder tries to gain access, let your IDS application notify you.

Monitor all log files.
Monitoring log files will give administrators the upper hand since most system activities are logged. It is a good idea to automate this task using a log-checking program such as Logwatch or LogSentry. Monitoring log files will not notify administrators of a rootkit attack, however it will also inform them of unusual system activity such as successful and unsuccessful login attempts.

Implementing basic security precautions is the first step in keeping intruders off systems and preventing any kind of attack(s). An administrator can also prevent rootkit attacks by ensuring file security.

File Attribute Security
There are a few steps that an administrator can take to implement file security. One such way is to ensure that common files cannot be overwritten or changed. This can be accomplished by setting the immutable flag on important system files.

Using the Immutable Flag
The term immutable means unable to change. By setting the immutable flag on a file it cannot be changed, renamed, deleted, or even linked to.

To set the immutable flag on a file, use the "chattr" command found in most Linux distributions.

- **chattr +i <file>:** sets the immutable flag.
- **chattr -i <file>:** unsets the immutable flag.
- **lsattr <file>:** displays attributes set to a file.

Setting the immutable flag on some common files will cause most rootkits to fail since the file is marked as unchangeable. This procedure may not stop LKM rootkits as the kernel cannot be set as unchangeable. The list of files starting on page 2 is a good starting point when deciding which files to set the immutable flag on.

The immutable flag can only be set and unset by root. Remember that the flag can be removed as easily as it was added. Meaning should an attacker have access to a machine, he would be able to find the set flag then unset it. After the immutable flag is unset, then he would be able to install a rootkit. With this said, do not rely on the immutable flag as the only defense against rootkits.

It is important to remember that following basic security guidelines is always good practice. After security measures are in place, administrators should begin to implement detection and monitoring practices on their systems.

Rootkit Detection and Monitoring

How can an administrator monitor for something that has been designed to hide its existence?

Basically there are applications that have been written to assist administrators to monitor system activities. Such applications also assist administrators in detecting rootkits.

Tripwire and AIDE
Both Tripwire and AIDE are utilities used to monitor the integrity of files. They both create a secure password protected database of file and directory attributes that is used to compare against the current files and directories for changes. They use MD5 to check the integrity of the file. MD5 sums are a hash function that transforms a string of data of any length into a shorter fixed-length value. It is believed that no two strings of data will have the same MD5 value.

Output of an md5sum against netstat and ls on a Linux RedHat 7.1 system

```
dc1961b6ce3ff6d6fe2c89c8603f4985  ls
30286974e55bb9f9e82f93cc44c39492  netstat
```

Notice the MD5 value is not the same for these commands. If files being monitored are modified in any way, Tripwire and AIDE will notify administrators of the change.

Both applications have been widely accepted in the security industry. I recommend learning more about it and using it throughout your enterprise. Tripwire is available in both commercial and GNU licensing. AIDE is also available in GNU licensing.

```
http://www.tripwire.com              commercial license
http://www.tripwire.org              gnu license
http://www.cs.tut.fi/~rammer/aide.html
```

RedHat Package Manager
Much like Tripwire and AIDE, Redhat Package Mangager (rpm) can be used to verify the checksum of installed applications. Not only does it verify the checksum, it can also verify other file discrepancies such as permissions and file size. The rpm command with the –V option is used to check the signature of any installed rpm package. Using the –Va option will check all installed packages.

Running "rpm –V util-linux" on a test system returned the following is the result:

```
.......T c /etc/fdprm
.......T c /etc/pam.d/chfn
.......T c /etc/pam.d/chsh
S.5....T c /etc/pam.d/login
```

Using the excerpt from the man page found below, you could interpret the above results of the rpm command. Based on the results, it can be determined that mTime (T) has changed on all files. The mTime is the file modification date and time. It can also be said that the file size (S) and MD5 checksum (5) of the login file has also changed. With this information an administrator would be able to research why the files have changed.

From the RPM man page:

```
S file Size differs
M Mode differs (includes permissions and file type)
5 MD5 sum differs
D Device major/minor number mis-match
L readLink(2) path mis-match
U User ownership differs
G Group ownership differs
T mTime differs
```

Another application called Chkrootkit can be used to assist administrators in monitoring for rootkit signatures.

Chkrootkit
Chkrootkit is a shell script that checks system binaries for rootkit modification. It can also detect some well-known LKM rootkits. Chkrootkit is written and maintained by Neslon Murilo and can be found at www.chkrootkit.org.

Using the following command files, chkrootkit searches for common files and directories that rootkits place on the system.

- awk
- cut
- echo
- egrep
- find
- head

- id
- ls
- netstat
- ps
- strings
- sed
- uname

Chkrootkit also checks for hidden processes by checking the output of *ps* with the */proc* directory. Many rootkits may run sniffers and backdoors as hidden processes. On a system that is very busy, this process may generate false positives because the system will have a process that finishes running before the compare can complete.

Chkrootkit also verifies that the network interfaces are not in promiscuous mode. Promiscuous mode allows a network device to intercept and read each network packet that arrives in its entirety. This mode provides the sniffer program all packets for analysis.

Chkrootkit has the ability to use command files from an alternate path. With this option, administrators can place files from the above list to a write protected file system, compact disk, or floppy diskette. If these files cannot be copied out, setting the immutable flag can assure files do not get overwritten. This ensures that chkrookit is using uninfected files. Using infected files defeats the purpose of utilizing this application.

As mentioned before, chkrootkit monitors and detects for well-known LKM rootkits. In addition, utilizing commands such as lsmod and kstat will aide administrators monitoring against these types of rootkits.

LSMOD
Lsmod is a utility used to list modules loaded into the kernel. An administrator should become familiar with the output of this command. The following is a sample output of lsmod command ran on a RedHat 7.2 system. The following output displays the module, module size, use count, and the list of referring modules.

```
3c59x                  28424       1
cdrom                  31936       0     (autoclean) [sr_mod]
sr_mod                 16056       10    (autoclean)
ext3                   64768       2
usb-uhci               24324       0     (unused)
usbcore                71072       1     [usb-uhci]
ext3                   64768       2
jbd                    47892       2     [ext3]
aic7xxx                128256      3
sd_mod                 12832       6
scsi_mod               104800      4     [sr_mod aha1542 aic7xxx
sd_mod]
```

Keeping track of all modules that are loaded into the kernel is a good step to monitoring against LKM rookits.

It is important to remember that the lsmod command may be manipulated into hiding the rootkit module by the rootkit module. There is however another command that can be run to list what modules are loaded into the kernel. This command is called cat.

Cat is a command that concatenates files and displays them to screen or some other output. Using the cat command against /proc/modules will display the same information as lsmod.

KSTAT
To assist in the detection of LKM rootkits, a program has been written by the softproject group called Kernel Security Therapy Anti Trolls (KSTAT). KSTAT has been written for both the 2.2.x and 2.4.x kernels.

Running kstat with the –s option will display information about the system call table (`sys_call_table`). Kstat will display discrepancies in the memory address results.

The following is an example of a kstat -s warning. It indicates that the memory address in the system call table has changed thus indicating an LKM rootkit.

```
sys_fork      0xc4051428 WARNING! Should be at 0xc0108c88
sys_write     0xc4051590 WARNING! Should be at 0xc01269b8
sys_close     0xc405163c WARNING! Should be at 0xc01264a4
sys_kill      0xc40514d0 WARNING! Should be at 0xc011060c
sys_mkdir     0xc405172c WARNING! Should be at 0xc012e540
sys_clone     0xc405147c WARNING! Should be at 0xc0108ca4
sys_getdents  0xc40512a4 WARNING! Should be at 0xc013022c
```

It is strongly recommend that all the above applications be run automatically on a nightly basis through cron and the results be emailed to the IT staff. This will ensure that the applications are run thus minimizing the effects should such an attack happen.

If a rootkit still finds its way onto a system, using all of the above applications should indicate what rootkit is installed on the system giving administrators the upper hand in removing the rootkit.

A lesson that should be taken from this paper when trying to remove a rootkit from a system is it is never known the effects of the rootkit. Should an administrator try and remove the rootkit he may miss something important allowing the intruder to gain future access to the machine. It is necessary to thoroughly analyze the entire system after a rootkit compromise. A system may never be the same after this kind of attack.

Should an administrator decide to try and resolve the issue, the following section will assist him in such a task.

Rootkit Removal

Many articles on rootkits and rootkit removal state that the first step to removing a rootkit is to disconnect the infected system from the network completely. CERT also recommends removing the system from the network. It is my belief that before removing the system from the network, the first step should be to evaluate the situation.

If the system is mission critical and is taken off the network, how much money will the company or client lose? This is a very important question that every administrator should ask before taking the system off the network.

Taking the system off the network would remove any active sessions open by an attacker causing the intruder to be disconnected. Since most rootkits also run a sniffer or password grabber, disconnecting the system from the network will ensure that any communication to the outside world will fail.

If a system is mission critical and time is of the essence to get the system back up and running, I would recommend replacing the infected hard drive in order to get the system back into production. Keeping the infected hard drive will give administrators the ability to look at the attack in greater detail without compromising the productivity of the business. If the hard drive cannot be replaced, use the dd or mkisofs command to make an iso image of the entire system. After an image is made of the hard drive, then begin formatting and reinstalling the operating system and application software. Having some sort of backup of an infected system will assist with any legal proceedings that may occur. The backup or backup image can also be used for training other employees on intrusions and rootkits.

When reviewing the compromised system, next the administrator needs to find out the extent of the rootkit damage. Utilize all the applications mentioned in the paper to begin searching for the rootkit. Using these applications, an administrator can find out what rootkit may be on the system. Remember running chkrootkit at this time may fail unless the administrator has placed the proper commands in a read only location. If an administrator has not properly setup chkrookit the administrator can use the RPM command to get the original commands placed back on the system.

Earlier in the paper, I discussed the usage of RPM to monitor for file integrity changes. At this point an administrator will want to utilize this command to find out exactly what packages will need to be reinstalled. By reinstalling the infected applications, chkrootkit will now function. It is recommended to start installing the

following packages. Please note that the following packages may be RedHat specific and may be different in other Linux distrobutions.

Fileutils: contains common file management utilities such as ls, cp, chown, chgrp, and du.

Procps: contains system and process monitoring utilities such as ps, top, and w.

Net-tools: contain basic networking tools such as ifconfig and netstat

Findutils: contains the find command that will assist locating files on a system.

Binutils: contains the stings command that is used with chkrookit to examine binary files.

When removing and restoring infected files, try not to restore files from a backup tape unless it can be guaranteed that files on the tape have not been altered. It is recommended that the files being restored come from the original copies of the distribution installation media.

After restoring commonly used commands, use them to find what files and processes are on the system. Use the ps command to show rootkit processes. After finding these processes, use the killall command to stop them. After they have stopped, next step is to find out how they got started. Look in the init scripts directory on the system. RedHat systems init scripts are found in /etc/rc.d/init.d/.

As mentioned before, some rootkits are designed as a Loadable Kernel Module (LKM). Because these rootkits are loaded using kernel modules, these modules can be removed from the kernel and then deleted from the system. To remove a module from the kernel, use the rmmod command. Remember that LKM rootkits can change the behavior of a command, which means if an administrator tries to remove a module by running rmmod, the module may not be removed. The sure fire way to make sure that a kernel module is no longer loaded is to reinstall the kernel from a backup or the distribution media.

After the rootkit is cleaned up and there is no trace of it, find out how the intruder gained entry. Maybe your intrusion detection program needs to be evaluated or your patches have to be installed sooner.

Conclusion

In conclusion, it is a good practice to secure your system from intruders, but it is important to know the effects of the intrusion. Protecting your systems requires more than a single tool. It also requires pre-planning when installing new systems, maintaining backups, keeping up-to-date with the latest system and/or security patches, as well as performing regular audits. As long as there is a possibility of system vulnerabilities, security will continue to challenge IT professionals.

References:

Altunergil, O. (2001, December 14). Understanding Rootkits. Retrieved from the World Wide Web on October 5, 2002: http://linux.oreillynet.com/pub/a/linux/2001/12/14/rootkit.html

Brumley, D. (1999, September). Rootkits - How Intruders Hide. Retrieved from the World Wide Web on October 5, 2002: http://www.theorygroup.com/Theory/rootkits.html

Steps for Recovering from a UNIX or NT System Compromise. (2000, April 17). Retrieved from the World Wide Web on November 3, 2002: http://www.cert.org/tech_tips/root_compromise.html

Drake, J. (2001, October 12). How to tell if your Linux box has been cracked. Retrieved from the World Wide Web on October 18, 2002: http://www.linuxworld.com/site-stories/2001/1012.cracked.html

Galitz, G. (2001). Rootkits: Hiding a Successful System Compromise. Retrieved from the World Wide Web on November 19, 2002: http://www.iwar.org.uk/comsec/resources/root-berkeley/rootkit.htm

Miller, T. (2000). Detecting Loadable Kernel Modules. Retrieved from the World Wide Web on January 1, 2003: http://www.incident-response.org/LKM.htm

Project: Tripwire: Summary. (2002, February 28). Retrieved from the World Wide Web on October 4, 2002: http://sourceforge.net/projects/tripwire

Prosise, C. and Shah, S, (2001, January 25). At the root of rootkits. Retrieved from the World Wide Web on October 15, 2002: http://builder.cnet.com/webbuilding/0-7532-8-4561014-1.html?tag=st.bl.7532.edt.7532-8-4561014

Somer, L. (2002, February 11). Unix rootkits and backdoors. Retrieved from the World Wide Web on October 15, 2002: http://packetstormsecurity.nl/UNIX/penetration/rootkits/

What's chkrootkit? (2002, September 16). Retrieved from the World Wide Web on October 15, 2002: http://www.chkrootkit.org

Windows NT/2K Rootkits. Retrieved from the World Wide Web on December 31, 2002: http://www.rootkit.com/softwaremap/trove_list.php?form_cat=22&discrim=22

Last Updated: August 21st, 2017

Upcoming SANS Training

Click Here for a full list of all Upcoming SANS Events by Location

SANS Tampa - Clearwater 2017	Clearwater, FLUS	Sep 05, 2017 - Sep 10, 2017	Live Event
SANS Network Security 2017	Las Vegas, NVUS	Sep 10, 2017 - Sep 17, 2017	Live Event
SANS Dublin 2017	Dublin, IE	Sep 11, 2017 - Sep 16, 2017	Live Event
SANS Baltimore Fall 2017	Baltimore, MDUS	Sep 25, 2017 - Sep 30, 2017	Live Event
SANS Copenhagen 2017	Copenhagen, DK	Sep 25, 2017 - Sep 30, 2017	Live Event
SANS London September 2017	London, GB	Sep 25, 2017 - Sep 30, 2017	Live Event
Data Breach Summit & Training	Chicago, ILUS	Sep 25, 2017 - Oct 02, 2017	Live Event
Rocky Mountain Fall 2017	Denver, COUS	Sep 25, 2017 - Sep 30, 2017	Live Event
SANS SEC504 at Cyber Security Week 2017	The Hague, NL	Sep 25, 2017 - Sep 30, 2017	Live Event
SANS Oslo Autumn 2017	Oslo, NO	Oct 02, 2017 - Oct 07, 2017	Live Event
SANS DFIR Prague 2017	Prague, CZ	Oct 02, 2017 - Oct 08, 2017	Live Event
SANS Phoenix-Mesa 2017	Mesa, AZUS	Oct 09, 2017 - Oct 14, 2017	Live Event
SANS October Singapore 2017	Singapore, SG	Oct 09, 2017 - Oct 28, 2017	Live Event
SANS AUD507 (GSNA) @ Canberra 2017	Canberra, AU	Oct 09, 2017 - Oct 14, 2017	Live Event
Secure DevOps Summit & Training	Denver, COUS	Oct 10, 2017 - Oct 17, 2017	Live Event
SANS Tysons Corner Fall 2017	McLean, VAUS	Oct 14, 2017 - Oct 21, 2017	Live Event
SANS Tokyo Autumn 2017	Tokyo, JP	Oct 16, 2017 - Oct 28, 2017	Live Event
SANS Brussels Autumn 2017	Brussels, BE	Oct 16, 2017 - Oct 21, 2017	Live Event
SANS Berlin 2017	Berlin, DE	Oct 23, 2017 - Oct 28, 2017	Live Event
SANS San Diego 2017	San Diego, CAUS	Oct 30, 2017 - Nov 04, 2017	Live Event
SANS Seattle 2017	Seattle, WAUS	Oct 30, 2017 - Nov 04, 2017	Live Event
SANS Gulf Region 2017	Dubai, AE	Nov 04, 2017 - Nov 16, 2017	Live Event
SANS Miami 2017	Miami, FLUS	Nov 06, 2017 - Nov 11, 2017	Live Event
SANS Amsterdam 2017	Amsterdam, NL	Nov 06, 2017 - Nov 11, 2017	Live Event
SANS Milan November 2017	Milan, IT	Nov 06, 2017 - Nov 11, 2017	Live Event
Pen Test Hackfest Summit & Training 2017	Bethesda, MDUS	Nov 13, 2017 - Nov 20, 2017	Live Event
SANS Paris November 2017	Paris, FR	Nov 13, 2017 - Nov 18, 2017	Live Event
SANS Sydney 2017	Sydney, AU	Nov 13, 2017 - Nov 25, 2017	Live Event
SANS San Francisco Fall 2017	OnlineCAUS	Sep 05, 2017 - Sep 10, 2017	Live Event
SANS OnDemand	Books & MP3s OnlyUS	Anytime	Self Paced