

Forum:[Hack Forums](#)

Thread: [Almost Everything About Rootkits](#)

AoEMonk - June 12, 2011, 11:14 am

Almost Everything About Rootkits!

1 – What is a root kit? Spoiler ([Click to View](#)) A rootkit is a program. Rootkits come in all different shapes and styles, some more advance than others. Rootkits are basically programs that help attackers keep their position as root. Notice it's called a "rootkit". 'root' meaning the highest level of administration on *nix based systems and 'kit' meaning a collection of tools. Rootkits contain tools which help attackers hide their presence as well as give the attacker full control of the server or host continuously without being noticed.

Rootkits are usually installed on systems when they have been successfully compromised and the highest level of access has been given (usually root) Some rootkits refuse to be installed until the attacker has root access, due to read and write permission to certain files. Once the system has been successfully compromised and the attacker has root, he\she may then install the rootkit, allowing them to cover their tracks and wipe the log files.

A typical rootkit consists of the following utilities (Note: We will look at these in a lot more detail later on):

- Backdoor Programs – login backdoors, telnetd etc
- Packet Sniffers – Sniff network traffic such as FTP, TELNET, POP3
- Log-Wiping Utilities – Bash the logs to cover tracks
- DDoS Programs – Turn the box into a DDoS client
- IRC\Bots – Bots used to take over IRC channels
- Miscellaneous programs – May contain exploit, log editor

3 – Hackers or the Kids?

Spoiler ([Click to View](#)) Now the question you are probably asking yourself is "Is a rootkit a hackers tool or just another script kiddies tool?" (Well, you may not be thinking that, maybe I just suck and my psychology skills are as good as yours..) Well, the "Hacker Jargon" defines a rootkit and a script kiddies tool and to some extend he\she\they\IT is right (The jargon is always right)

Rootkits don't really require that much skill to run or use. Most rootkits can be compiled like this...

```
gcc t0rn.c -o rootkit
```

then...

```
./rootkit
```

(Now obviously when compiling all "hacker" tools you need to chose a name which disguises it's purpose, so rootkit would be a really stupid choice)

However there are some rootkits that require more skill to run and use. Some rootkits require you to edit the source code before it's compiled and some even need you to edit the iptables and kernel.

(Very advance ones, they used one at the "Black Hat Conference" in 2002)

So, rootkits are used by both hackers and script kiddies. I personally believe that a hacker would have to write his own rootkit to call himself a hacker not just run and use someone else code.

(However that's just my opinion, so don't hold me to that!)

5 – The Language rootkits are coded in

Spoiler ([Click to View](#)) Hmmm, well this isn't going to be short...

Most rootkits are coded in C or Assembly (Shell code). Most of the well-known rootkits are coded in C so the attacker can edit the source code to fit its target specification. (E.g. The logs files could be

stored in a different location)

6 – Different types of rootkits

Spoiler ([Click to View](#)) At the current time of writing there are 5 main types of rootkits.

Application rootkits – Established at the application layer

Kernel rootkits – Established at the kernel level (Core of any OS)

Firmware rootkits - Firmware rootkit implies use of creating a permanent illusion of rootkit malware. It can remain hidden in firmware as this is not checked for code integrity.

Virtualized rootkits - The lowest level of rootkits produced is virtualized rootkits. These rootkits function by the modification of the systems boot sequence, to be loaded instead of the original virtual machine monitor or operating system. A virtualized rootkit is able to intercept all hardware calls made by the guest operating system by loading the original operating system as Virtual Machine.

Library level rootkits - Library rootkits usually patch, hook, or supplant system cells with versions that seek to keep the attacker unknown.

When I say “established” this could be referred to of where exactly the rootkit hides. Now let's start of my looking at an application rootkit.

An application rootkit is basically a rootkit which “replaces” all the well know system binary files (ls, netstat, killall) with “fake” or “Trojaned” ones. The trojaned or fake system files will help hide the attackers presence, report false information to the system administrator and even provide a Backdoor for the attacker. To help you understand this more I have provided a list of all the typical system files, which are “replaced” to, help the attacker cover his or her tracks. The list was taken from “Rootkit: Attacker Undercover Tools” by Sailman Manap.

LIST START....

Programs replace to hide attacker presence.

“ls”, “find”, “du” – Trojaned system file will be able to hide attackers file, directory and stuff that have been brought into the system from being listing.

“ps”, “top”, “pidof” – All these programs are process monitor program. Trojaned program will hide attacker process from being listing.

“netstat” – netstat is used to check network activity such as open port, network connections establish and listening. Trojaned netstat will hide processes installed by attacker such as ssh daemon or other services.

“killall” – Trojaned “killall” will not be able to kill attacker process.

“ifconfig” – When sniffer is running PROMISC flag is set to the nic. “ifconfig” is a handy utility to set and to view setting of ethernet nic. Trojaned “ifconfig” will not display the PROMISC flag when sniffer is running. This is useful to hide sniffer from being detected.

“crontab” – Trojaned “crontab” will hide the attacker's crontab entry.

“tcpd”, “syslogd” – Trojanised “tcpd” and “syslog” will not log any connection made by attacker. “tcpd” also capable to bypass tcp wrapper enforcement.

LIST END

Hopefully, that would should have given you a better idea of what an Application is. Remember, this section has only be written so you can distinguish the differences between a “Application” rootkit and “Kernel” rootkit. Lets now take a look at a Kernel rootkit.

A Kernel rootkit is a rootkit that buries itself deep in the Kernel. This makes it extremely hard to detect and remove. Kernel rootkits are more advance then Application rootkits, A Kernel rootkit works by exploiting and manipulating Kernel capabilities. Now I don't really want to go in much more

detail on Kernel rootkits because they can get quite advance (Well, they ARE) we'll talk about them later in this post.

7 – Backdoors

Spoiler ([Click to View](#)) Most of today's (decent) rootkits contain "Backdoors". Now you should all know what a Backdoor is but just in case you didn't I will quickly give a brief explanation of all.

Backdoor – A program or script which allows an attacker to establish some form of privilege and remote communication without logging into the system. Backdoors are usually installed when the system has been successfully compromised and some form of exploit has been entailed. The advantage of installing a backdoor on a system means that the attacker doesn't have to keep using the same exploit over and over again. The disadvantage of installing a backdoor means at one point or another the system administrator will notice suspicious activity in his network traffic, if he or she were to run a port scanner such as Nmap (Coded by Fyodor <http://www.insecure.org>) he or she would soon uncover an open port and sooner or later remove the backdoor.

A typical example of a Windows backdoor is one entitled "Tini.exe" (Made by NTSecurity) This little program listens on port 7777 for incoming connections, once a connection has been established a remote command shell is executed for the attacker who establishes the connection. (Now as I have mentioned this t-file generally deals with *nix backdoors, so I don't really want to get side stepped talking about windows backdoors, exploits etc I thought I'd just mention tini.exe to give you a general idea of what a Backdoor consists of.

8 – Sniffers

Spoiler ([Click to View](#)) A lot of today's rootkits contain programs known as "Sniffers". What are Sniffers? (Also known as Packet Sniffers) Basically packet Sniffers are programs that are made to "Monitor" network traffic, TCP/IP or any other network device. I'm sure you know when you are browsing the Internet or playing online games "Packets" of data are going to and from your Computer. Attackers install Sniffers so they can capture valuable information which is floating to and from your computer.

What type of valuable information? Here is a list of what a Sniffer is capable of...

- Sniffing FTP passwords
- Sniffing Telnet passwords
- Sniffing Network passwords
- Sniffing POP3 passwords
- Capturing websites you have visited
- Sniffing Gateways
- Lots more

9 – Cleaners (Log Bashers)

Spoiler ([Click to View](#)) Ah, we come to something a lot simpler, Log Bashers (Also known as Log deleters, Log killers and Log Cleaners)

No matter what the title they all do the same thing. Delete system log files. System Administrators rely on logging as an extra form of security. Log files can keep track on who logged in last and at what type, what programs were run as that user was logged in etc etc. Therefore it is exceptionally important for the attacker to destroy ALL traces of log files. Now, some of you may be thinking:

"If all the Log files are deleted won't this give an indication to the system administrator that there box has been hacked?"

If you are thinking that, then you're dead right. Deleting the log files can sometimes be pretty stupid, the best way to get around the log files is to "edit" the entries by deleting your entries and filling in some false ones (Sometimes this requires root access, but if your running a log cleaner of a rootkit you should already have root

10 – Rootkit Extra Utilities

Spoiler ([Click to View](#)) I will try and keep this section short due to there isn't really that much to say. As you should know by now and as I have mentioned rootkits come in all shapes and styles. Some rootkits are well known for their advance log cleaner, others for their advance Backdoor and others for their advance, stealth hard to remove installation procedure. There are some rootkits which are well known for being SAR (Swiss Army Rootkits) basically, they are rootkits with average features plus a whole load of extra utilities such as Bots, DDoS, Extra scripts, Password crackers, Killer scripts etc

Rootkits that contain scripts that cause DDoS attacks are considered dangerous; if an attacker were to exploit 100's of servers and install such a rootkit those servers would then become "Zombies" they could launch DDoS attacks (SYN, PING, FINGER, UDP, TCP) against chosen targets. Rootkits are continuously being made more advance and extra utilities are being added on each time.

11 – Kernel Rootkits (More Detail)

Spoiler ([Click to View](#)) The best way to start of this section is talk about how Kernel rootkits actually work. Kernel rootkits work, basically by exploiting LKM. (Loadable Kernel Modules) LKM are used to load device drivers on a "as-needed" bases. LKM are usually only exploited so the attacker can perform malicious activity.

Kernel rootkits are way more dangerous than Application rootkits because instead of just replacing the basic binaries like "ls" and "netstat" they attack the kernel directly and manipulate system-calls like open() and read(). As we know application rootkits replace binaries, if the administrator was clever and analyzed the actual binaries which had been replaced they will realize the differences in size (e.g. the program could contain an extra 128 bytes) However, this wouldn't be possible with Kernel rootkits because instead of actually changing the size and structure of the program, they just change the way the program operates. For example programs like "ps" use an open system call "open()" and reads information from files in the directory /proc, where also the information about running processes is kept.

Credits

Spoiler ([Click to View](#)) I didn't make this tutorial. I just found it on the internet and made it a little better by adding different colours, made sections and took out some of the outdated stuff. I originally found it on Code

<https://slimshady90358.wordpress.com/2011/03/22/the-art-of-rootkits-everything-about-rootkits/>

Guess Who Is Back - June 12, 2011, 11:14 am

Very nice totorial + rated 5 star.

IDCUMF - June 12, 2011, 12:27 pm

Really clear and nice tutorial gave me a bit more than i know. Thank you ans 5 stared also.

Sonic Boom - June 12, 2011, 1:24 pm

Hey that is a really nice guide! Thanks for this contribution.

Anonymous.7654 - June 12, 2011, 1:40 pm

Thanks for Tutorial,very nice thread.

AoEMonk - June 13, 2011, 12:04 am

Thanks for the great feedback guys!