ROOTKITS: THE ENEMY WITHIN

While it was assumed in the past that viruses only targeted Windows, hackers targeting the FOSS world proved this wrong. A rootkit on a Linux distribution makes it vulnerable to programmatic and manual attacks, which calls for serious detection methods and removal mechanisms. As Linux is becoming a de-facto standard in data centres, it's crucial that sysadmins gain knowledge about rootkit problems and resolutions to secure their infrastructure. This article hopes to create the necessary awareness, and should be helpful to systems administrators and IT management teams.

In today's world, IT is used almost everywhere, to automate routine tasks and add value to our lives, right from purchasing stocks on the Internet to withdrawing money from an ATM. With its benefits, there are bound to be problems as well. Practically every computer user knows about the threat of viruses. Even the partially computer-literate learn the hard way that their data is not safe but can be corrupted, deleted or stolen — hence, good backups are essential.

While there are many anti-virus software available, there is another category of malicious software called rootkits, which are more dangerous by nature. These are created by some of the world’s smartest programmers, but unfortunately, with bad intentions, hence creating a problem for users.

Malicious software could run as an application, in user-space, or as part of the operating system. Rootkits are typically in the second category, making them powerful, dangerous, tough to detect, and very difficult to get rid of. An unknown person gaining control of your machine and working on it, probably at the same time you are using it, could be dangerous, and invades your privacy. Key-logger kits can steal passwords, credit card numbers, personal details, finance data stored in spreadsheets, confidential business data, etc.

Rootkits are essentially small collections of tools, utilities and scripts. The sole idea behind installing this collection is to be able to gain administrator-level access on the target machine, so that it can either be remotely controlled to harvest secret data, or used to attack other vulnerable machines, to replicate the rootkit on those, and subsequently “own” those systems too.

A rootkit typically contains a combination of network sniffers, log parsers, log-cleaning scripts, scripts to change user privileges, core systems utilities to detect IP addresses, netstat, utilities to kill running processes, scripts to hide code, and a zipped copy of the whole kit, for replication.

How rootkits differ from viruses

Let’s quickly look at the symptoms of both first.

Virus	**Rootkit**
Primarily runs as an application process	Primarily runs as a part of OS/kernel
Usually gains user level access	Gains root/admin level access
Does not open backdoors	Opens backdoors — viz., port, IP, etc.
Does not provide any remote access	Provides remote access to attacker
Fairly easy to detect and remove	Extremely difficult to detect and remove
Is meant to create nuisance and data damage	Is meant to cause privacy/data theft

As you can see in the above table, while the dividing line between viruses/Trojans and rootkits is becoming blurred (with the common purpose of causing data loss, or capturing/harvesting confidential data like user names, passwords, email addresses, etc), there are still some distinct differences between them. Although a virus normally runs in “stealth mode”, hiding itself by infecting executables and system files, it still typically runs as an application — which is why anti-virus software can detect and remove it. A Trojan, which is an advanced virus, is meant to hide in a more sophisticated fashion.

A rootkit, on the other hand, subverts part of the operating system to hide itself and gain the maximum control possible. Due to this, it is capable of monitoring as well as performing all activities on a system. It can act as a vehicle for other rootkits and viruses as well. Rootkits turn a computer into a remotely controllable victim, often also making it a spambot to send out unsolicited commercial email.

Infection/installation

Rootkits follow virus-like methods to compromise the system; however, since they need OS-level privileges, installation methods differ a bit. Typically, attackers try the following ways to get access to a system:

- With physical access to the system, attackers try guessing non-complex passwords. If it's possible to boot the computer from the attacker's media (CD, USB drive), then the attacker can also try various techniques to recover the root password from the installed OS on the hard disk.
- Attackers can remotely detect OS or network application vulnerabilities that have not been patched by security fixes, and run attacks against these.
- Over the Web, attackers use embedded scripts that sneak into a system via the browser.
- Many spyware software act as easy and sure-fire vehicles to propagate rootkits to systems connected to the Internet.

Usually, a rootkit is packaged as a compressed self-extracting zip file that extracts its contents once it gets on a system. Often, a small kit installer is also packaged, which ensures the health of an installation, gains administrative rights and does what's needed to run in stealth mode.

A few advanced rootkits are also capable of detecting anti-virus and anti-spyware software, and accordingly change their way of working, or modify their outputs, to avoid being detected. For example, some rootkits copy the kit tools to the root of the file system, but a simple directory listing does not show those files since the rootkit modifies the output of the OS directory listing commands.

Since attackers make every attempt to optimise the kit's code size, it takes little time to install and activate a rootkit. Rootkits are expected to spread as much as possible, and so almost all contain a distributable copy of themselves. When in action, they use all possible means to scan the local network and find other vulnerable systems.

If LAN security is not properly designed, gaining administrative rights on one system is often enough for a rootkit to easily propagate to other systems on the LAN. Modern kits are designed to detect Internet access, fetch the latest rootkit update, copy it to all infected machines, and then try to propagate over the Internet to other vulnerable or poorly configured systems.

Detection

As mentioned earlier, detecting a rootkit is a seriously difficult job, which requires administrators to go an extra mile when compared to detecting viruses and Trojans. While a few rootkits can be blocked by the latest anti-virus tools, most can still bypass these. Since the rootkit becomes part of the operating system, seemingly rudimentary methods such as booting from an emergency disk or USB drive are very useful in loading a fresh and known "good copy" of the OS, to run a detection tool without interference from the rootkit. Even then, many detection tools only detect, and still cannot remove these, which requires a manual cleanup process.

Detection tools need to adopt a different methodology rather than simple file checks or process checks, because if the rootkit is active, it may fool these legacy detection methods or algorithms. A better approach is to take a snap of different views of a system, and compare those with a baseline snap captured when the detection tool was first installed on a newly-built system.

Newer tools use artificial-intelligence algorithms to detect variations in the system's status, and hence don't require any baseline benchmark. There are various detection algorithms, such as signature-based, which are similar to detecting a virus, or integrity-based, wherein files, processes and kernel modules are checked for their binary integrity. There is another effective method, where a memory dump is taken by the tool, and then parsed for anomalies, signatures or trends which are either directly or symptomatically related to a rootkit.

There are many commercial and open source rootkit detection software; let's take a look at a few popular tools.

McAfee and Symantec provide some protection against letting rootkits get installed, and also provide some level of detection. However, detecting rootkits needs dedicated specialised tools.

In the FOSS world, a famous tool called chkrootkit takes precedence over other tools. It can perform detailed binary checks, file modification validations, and check kernel modules. It works smoothly over a wide range of Linux distributions, making it a must-have tool in the administrator's toolbox.
Similarly, Tripwire is an important open source tool, which can perform extensive MD5 hash checking, and detect anomalies such as back-door process checks and other local exploits.
Rootkit Hunter is another such famous tool — an elaborate script capable of checking multiple rootkits. It can also check for wrong file permissions and kernel modules, which can be incorporated into a daily scanning job. Upon arrival of a

new rootkit, an efficient systems administrator can study it and write a script to detect it.
It is important to note that malware creators often study these detection mechanisms too, and make necessary changes in the newer versions of their rootkits, to make them undetectable by detection tools.

Rootkit examples and the damage they cause

Similar to viruses, there are unfortunately a lot of rootkits for Windows, for commercial-grade Red Hat Enterprise Linux, as well as all open source distributions. In many of these, since the kernel does not change drastically over a period of time, it’s often simple for attackers to write kits that would easily propagate.

Although there are a few hundred dangerous rootkits impacting the FOSS world, we will look at just a few commonly found ones.

Let’s start by mentioning the LRK kit first, because it is one of the oldest, and still active (first detected in 1997, but still found today on vulnerable systems). It has multiple versions, and is known to install very commonly used binaries such as `netstat`, `linsniffer`, `inetd`, `ifconfig`, etc.
Knark is a kit that is very difficult to detect, as it resides entirely in the kernel. It is very notorious because once active, it hides ports, files and processes from the administrator.

Beastkit is of a fairly new variety, written to target Red Hat distributions, while another dangerous variety, called Illogic, is known to start a process on port 901, which lets attackers telnet into the system and do practically anything that an administrator can with physical access.

It is worth mentioning the rootkits that have caused significant damage to Linux systems: Sneakin, Kitko, Ajakit and Devil. All these kits are capable of using advanced techniques, such as detecting the OS and modifying the kernel on the fly, port NATing to decipher information, key-logging to steal passwords and sneakily emailing them to the attacker's email address, optimising actions to take least amount of system resources, etc. These kits are also known to spread quickly, and convert infected machines into zombies that participate in spreading the infection further.

As noted earlier, rootkits also open backdoors in the form of ports, IP addresses or kernel processes, which allow attackers to take complete control of the system remotely, even over the Internet.

There are a few "friendly" rootkits that work with each other. If a rootkit trying to invade a system sees that there is already a friendly kit present, it helps by updating older binaries to make it stronger. A few rootkits are known to act as gateways to install viruses and Trojans on a LAN; the rootkit goes in first, makes room, sets user privileges, takes control of the file system, disables any anti-virus that's running, opens backdoors for viruses, and goes into stealth mode.

A new generation of rootkits is known to install fake SSL certificates to try deciphering HTTPS communications to gain credit card information, which is usually sent over a secure channel. There are some kits that use compromised systems for man-in-the-middle attacks on other systems' communications, which makes it difficult for cyber security investigators to find the actual attacker.

Designing a continuously improving security infrastructure is very essential to stop the spread of, and damage caused by rootkits. The latest intrusion-detection systems (IDS) *can* stop rootkits effectively at the doorstep. Senior IT management should be aware of these serious problems, and take aggressive action to secure their networks.

Source : http://www.opensourceforu.com/2011/09/rootkits-the-enemy-within/