Hail Mary CSE331 Project 1

Authors

- Ian Peitzsch
  - Modified syscalls
- Mathews Thankachan
  - Made /proc/hidePid writing system
- Xinhang Xie
  - Made backdoor

About

This project is a rootkit in the form of a loadable kernel module for Linux Kernel version 2.6.38.8 and has been tested on Ubuntu 11.04 32-bit version. This rookit assumes the attacker has already managed to gain root access to computer. Once loaded, the rootkit does the following tasks:

- Hides specific files and directories from showing up when a user does `ls` and similar commands
- Modifies the /etc/passwd and /etc/shadow file to add a backdoor account while returning the original contents of the files (pre-attack) when a normal user requests to see the file
- Hides specific processes from the process table when a user does a `ps`
- Give the ability to a malicious process to elevate its uid to 0 (root) upon demand

Github Repo

https://github.com/irpeitzsch/cse331

How to Use and How it Works

Loading and Unloading

After getting into the directory, run `./insertRootkit.sh`. This will make the rootkit.ko kernel object, load rootkit.ko into the kernel, and changes the permissions of `/proc/hidePid` to be writable. `insertRootkit.sh` can be modified to make `sudo insmod rootkit.ko` also take in optional parameter, in the form of `sudo insmod rootkit.ko pass=<passwd> shad=<shadow> ending=<end> magicNum=<num>` where:

- `passwd` is a string the user wants to be written to `/etc/passwd`
- `shadow` is a string the user wants to be written to `/etc/shadow`
- `end` is a string the user wants to use to signify what files to hide

- `num` is an `int` the user wants to use to hack setuid All of these parameters have default values, so they are not necessary to run. The rootkit can then be unloaded by doing `sudo rmmod rootkit`.

Hide Specific Files

This rootkit modifies the `getdents64` syscall which is used by `ls` and other similar functions to find entries in a directory. The modified syscall checks the name of each entry to see if it contains the string, specified by `ending`. If so, then that entry is not included in the returned list of entries. As shown above, `ending` can be changed to whatever the user wants it to be. e.g. By default `ending` is set to ".evil" so any file name that contains ".evil" will not show up by a call to ls.

Back Door

On initialization, this rootkit reads the contents of `/etc/passwd` and `/etc/shadow` into buffers, and writes the values of `pass` and `shad` to each file, respectively. `pass` and `shad` are strings that contain information like username, uid, hashed password, etc. in the normal formats for both `/etc/passwd` and `/etc/shadow`. As stated above, both `pass` and `shad` can be modified to the user's preferece. Since the original contents of each file are stored in buffers, the `read` syscall has also been modified to return the respective buffer if one of those files is read. e.g. By default, `pass = "boogyman:x:2000:1000:boogyman,,,:/home/boogyman:/bin/bash\n"` and `shad = "boogyman:$6$0/RGDCY4$RC0GMeLMGdzmTDw./9af6cr1Rc/zkh06uE3KCKLgPTiWIn0PDheGG8qMN4TEqQ61ke3PunJ2QHbcEMh4RyjoA/:18233:0:99999:7:::\n"`. These are equivalent to a user named "boogyman" with password "password" (super secure).

Hide Specific Processes

On initialization, this rootkit creates a file `/proc/hidePid` which has had its write function rewritten to add written values to a buffer. The rootkit then also modifies the `getdents` syscall, which is called by `ps`, to get not include entries that contain the any of the contents of this buffer in the return, similar to how we hide files. To use this function, simply do `sudo echo "pid" > /proc/hidePid` where `pid` is the process ID number of the process you would like to hide.

Elevate UID

This rootkit modifies the `setuid32` syscall, which is used by the `setuid(uid_t uid)` method defined in C to change a process's uid. The modified syscall checks if the value passed to it is equal to a number `magicNum`. If so, then the rootkit elevates the process's uid to root, otherwise the rootkit executes the original syscall. This `magicNum` field can be set by the user, as stated above. e.g. By default `magicNum = 55555`, so calling `setuid(55555)` will elevate a process's uid to 0.

Testing

Hiding Files and Processes

We tested this rootkit for hiding files and processes by doing

```
ls
ps
./insertRootkit.sh
ls
sudo echo "pid" > /proc/hidePid
ps
sudo rmmod rootkit
```

Where `pid` is changed to a known pid that we wanted to hide and there existed a file that had the default of `.evil` somewhere in its name.

Back Door

We tested the back door by doing

```
sudo cat /etc/shadow
sudo cat /etc/passwd
./insertRootkit.sh
sudo cat /etc/shadow
sudo cat /etc/passwd
su boogyman
```

And then entered the password `password` for the `su` command since that is the default password for the default username `boogyman`.

Elevate UID

We made a test program `testSetuid.c`. We tested the functionality of our rootkit by doing

```
gcc testSetuid.c
./insertRootkit.sh
./a.out
sudo rmmod rootkit
```

This runs `testSetuid.c` which prints its uid from before calling `setuid` with `magicNum` and then prints its uid after calling, which is 0.

References

- http://tldp.org/LDP/lkmpg/2.6/html/lkmpg.html
- https://www.oreilly.com/library/view/linux-device-drivers/0596000081/ch03s04.html
- http://man7.org/linux/man-pages/man2/getdents.2.html
- https://www.kernel.org/doc/Documentation/security/credentials.txt