Enumeration (Linux OS)

Once you have a shell on a Linux system, the first task to perform to find a vulnerability in the system to escalate privileges is enumerating the targeted system. Without enumeration it is not possible to find the vulnerabilities that may lead to a higher permission account. There are two ways to enumerate a system: the manual way, and the automated way. The manual way is a slower process, but often more reliable. It involves issuing commands to retrieve system information, reading key files to learn more about the system, seeing the running processes in the system, and so on. In a general sense, learning about the system, and what can be done to escalate to a higher permission account/shell. The automated way involves importing or transferring scripts to the target machine that will essentially perform the same tasks as the manual way, but in high speed. It is often the first resource used for escalating privileges because of the extensive result list it generates.

Below it is a small list of commands used to enumerate a Linux OS:

$Hostname

$Uname -a

$Cat /proc/version (kernel version and GCC)

$cat /etc/passwd

$cat /etc/shadow

$cat /etc/issue (OS)

$ps (see running processes for current shell)(-A flag to view all)(axjf, aux)

$env (PATH variable may have compiler of scripting language could be used to run code on the system.

$sudo -l

$ls -la

$id (gen. overview of user's priv level and group membership)

$cat /etc/passwd (Discover users on the system)(Use this command to convert the output of this command to a list for brute force: $ cat /etc/passwd | cut -d ":" -f 1)(another way is: cutting the system or service users out of the list, since they would not be very helpful. We can do that by grepping for "home", as real users will most likely have their folders under the "home"directory. Like so: $cat /etc/passwd | grep home)

$history (show earlier commands)(maybe passwords and usernames could be stored in the history)

$ifconfig (see network interfaces)(compare to the $"ip route" results)

$ip route

$netstat (following initial check for existing interfaces and network routes) (several flags) (-tp {list connections with the service and PID info}; -l {for listening ports}; -i {show statistics}; -ano {[-a]-all sockets,[-n] do not resolve names, [-o] display timers})

$find

Find files:

- `find . -name flag1.txt`: find the file named “flag1.txt” in the current directory
- `find /home -name flag1.txt`: find the file names “flag1.txt” in the /home directory
- `find / -type d -name config`: find the directory named config under “/”
- `find / -type f -perm 0777`: find files with the 777 permissions (files readable, writable, and executable by all users)
- `find / -perm a=x`: find executable files
- `find /home -user frank`: find all files for user “frank” under “/home”
- `find / -mtime 10`: find files that were modified in the last 10 days
- `find / -atime 10`: find files that were accessed in the last 10 day
- `find / -cmin -60`: find files changed within the last hour (60 minutes)
- `find / -amin -60`: find files accesses within the last hour (60 minutes)
- `find / -size 50M`: find files with a 50 MB size

This command can also be used with (+) and (-) signs to specify a file that is larger or smaller than the given size.

```
┌──(root💀 TryHackMe)-[/home/alper]
└─# find / -size +100M
```

The example above returns files that are larger than 100 MB. It is important to note that the “find” command tends to generate errors which sometimes makes the output hard to read. This is why it would be wise to use the “find” command with “-type f 2>/dev/null” to redirect errors to “/dev/null” and have a cleaner output (below).

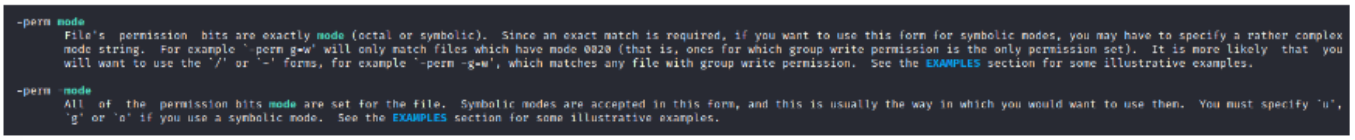
```
┌──(root💀 TryHackMe)-[/home/alper]
└─# find / -size +100M -type f 2>/dev/null
/usr/bin/burpsuite
/usr/lib/oracle/19.6/client64/lib/libociei.so
/usr/lib/jvm/java-11-openjdk-amd64/lib/modules
/usr/lib/firefox-esr/libxul.so
/proc/kcore

┌──(root💀 TryHackMe)-[/home/alper]
└─#
```

Folders and files that can be written to or executed from:

- `find / -writable -type d 2>/dev/null` : Find world-writeable folders
- `find / -perm -222 -type d 2>/dev/null` : Find world-writeable folders
- `find / -perm -o w -type d 2>/dev/null` : Find world-writeable folders

The reason we see three different "find" commands that could potentially lead to the same result can be seen in the manual document. As you can see below, the perm parameter affects the way "find" works.



```
-perm mode
       File's  permission  bits are exactly mode (octal or symbolic).  Since an exact match is required, if you want to use this form for symbolic modes, you may have to specify a rather complex
       mode string.  For example `-perm g=w' will only match files which have mode 0020 (that is, ones for which group write permission is the only permission set).  It is more likely  that  you
       will want to use the `/' or `-' forms, for example `-perm -g=w', which matches any file with group write permission.  See the EXAMPLES section for some illustrative examples.

-perm -mode
       All  of  the  permission bits mode are set for the file.  Symbolic modes are accepted in this form, and this is usually the way in which you would want to use them.  You must specify `u',
       `g' or `o' if you use a symbolic mode.  See the EXAMPLES section for some illustrative examples.
```

- `find / -perm -o x -type d 2>/dev/null` : Find world-executable folders

Find development tools and supported languages:

- `find / -name perl*`
- `find / -name python*`
- `find / -name gcc*`

Find specific file permissions:

Below is a short example used to find files that have the SUID bit set. The SUID bit allows the file to run with the privilege level of the account that owns it, rather than the account which runs it. This allows for an interesting privilege escalation path,we will see in more details on task 6. The example below is given to complete the subject on the "find" command.

- `find / -perm -u=s -type f 2>/dev/null`: Find files with the SUID bit, which allows us to run the file with a higher privilege level than the current user.

General Linux Commands

As we are in the Linux realm, familiarity with Linux commands, in general, will be very useful. Please spend some time getting comfortable with commands such as `find`, `locate`, `grep`, `cut`, `sort`, etc.

Automated tools include:

- LinPeas: https://github.com/carlospolop/privilege-escalation-awesome-scripts-suite/tree/master/linPEAS
- LinEnum: https://github.com/rebootuser/LinEnum[](https://github.com/rebootuser/LinEnum)
- LES (Linux Exploit Suggester): https://github.com/mzet-/linux-exploit-suggester

- Linux Smart Enumeration: https://github.com/diego-treitos/linux-smart-enumeration
- Linux Priv Checker: https://github.com/linted/linuxprivchecker