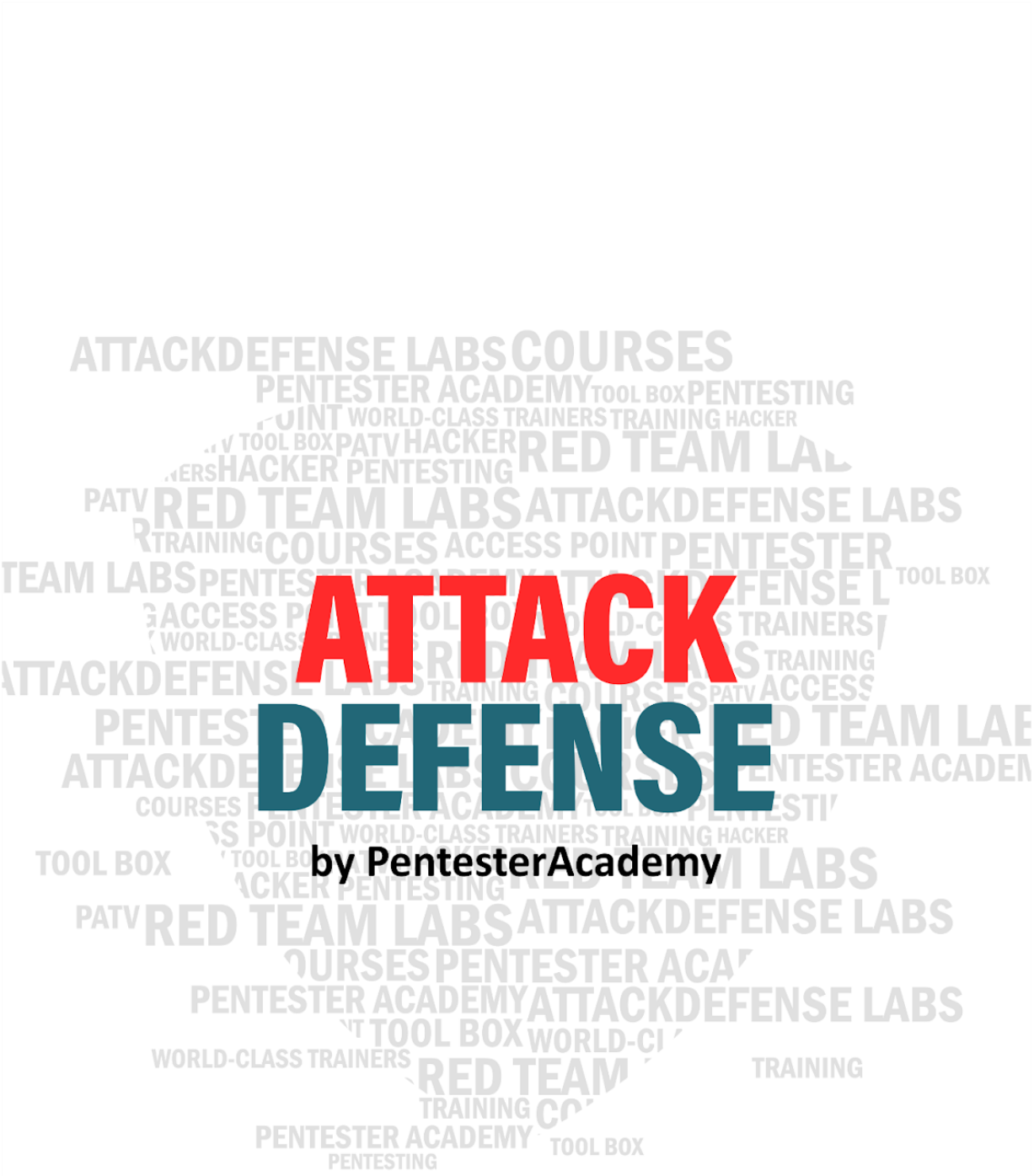

ATTACK
DEFENSE
by PentesterAcademy

Name	Exposed Claim
URL	https://attackdefense.com/challengedetails?cid=1348
Type	REST: JWT Basics

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.

Step 1: Check the IP address of the machine.

Command: ifconfig

```
root@attackdefense:~# ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.1.1.6  netmask 255.255.255.0  broadcast 10.1.1.255
        ether 02:42:0a:01:01:06  txqueuelen 0  (Ethernet)
        RX packets 1330  bytes 163506 (163.5 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 1345  bytes 4458367 (4.4 MB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

eth1: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.10.137.2  netmask 255.255.255.0  broadcast 192.10.137.255
        ether 02:42:c0:0a:89:02  txqueuelen 0  (Ethernet)
        RX packets 26  bytes 2004 (2.0 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 2749  bytes 5903898 (5.9 MB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 2749  bytes 5903898 (5.9 MB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

root@attackdefense:~#
```

The IP address of the machine is 192.10.137.2.

Therefore, the target REST API is running on 192.10.137.3, at port 1337.

Step 2: Checking the presence of the REST API.

Command: curl 192.10.137.3:1337

```
root@attackdefense:~# curl 192.10.137.3:1337
<!doctype html>

<html>
  <head>
    <meta charset="utf-8" />
    <meta http-equiv="X-UA-Compatible" content="IE=edge,chrome=1" />
    <title>Welcome to your Strapi app</title>
    <meta name="viewport" content="width=device-width, initial-scale=1" />
    <style>
      * {
        -webkit-box-sizing: border-box;
      }
```

The response reflects that Strapi CMS is running on the target machine.

Step 3: Getting the JWT Token for user elliot.

curl -H "Content-Type: application/json" -X POST -d '{"identifier": "elliot","password": "elliotalderson"}' http://192.10.137.3:1337/auth/local/ | jq

```
root@attackdefense:~# curl -H "Content-Type: application/json" -X POST -d '{"identifier": "elliot","password": "elliotalderson"}' ht
tp://192.10.137.3:1337/auth/local/ | jq
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100   486  100   433  100    53   1678    205 --:--:-- --:--:-- --:--:--  1883
{
  "jwt": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpZCI6MiwiZmxhZyI6IlRoaXMtSXMtVGhlLUZsYWctNzg3ZDFhMTYzNmNhIiwiaWF0IjoxNTcyOTgwNTkwL
CJleHAiOjE1NzU1NzI1OTB9.BOF5oKVn_FhqHAJdHJOD7RjtL_7qkxYlJ61xq4bbUpk",
  "user": {
    "username": "elliot",
    "id": 2,
    "email": "elliot@evilcorp.com",
    "provider": "local",
    "confirmed": 1,
    "blocked": null,
    "role": {
      "id": 2,
      "name": "Authenticated",
      "description": "Default role given to authenticated user.",
      "type": "authenticated"
    }
  }
}
root@attackdefense:~#
```

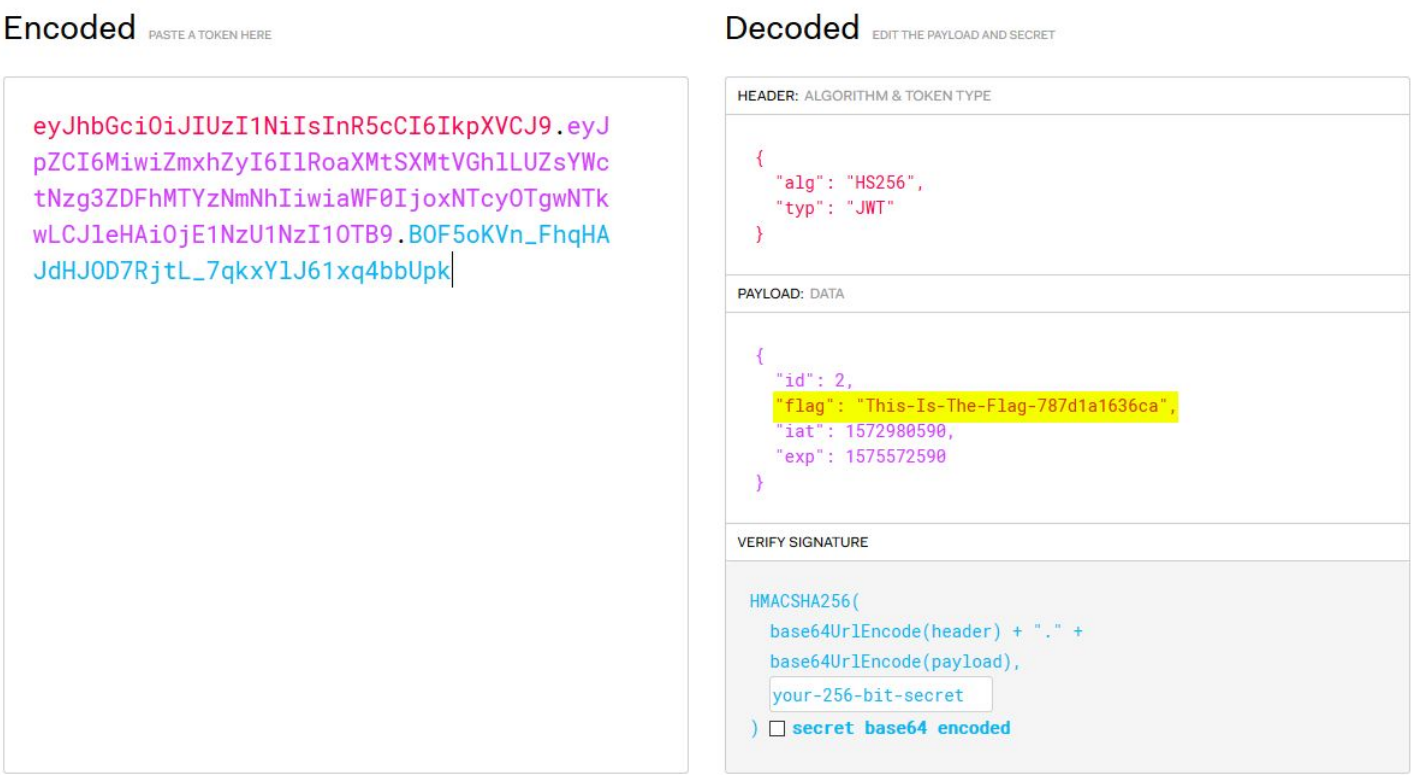
The response contains the JWT Token for the user.

JWT Token:
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpZCI6MiwiZmxhZyI6IlRoaXMtSXMtVGhlLUZsYWctNzg3ZDFhMTYzNmNhIiwiaWF0IjoxNTcyOTgwNTkwLCJleHAiOjE1NzU1NzI1OTB9.BOF5oKVn_FhqHAJdHJOD7RjtL_7qkxYlJ61xq4bbUpk

Step 4: Decoding the token payload.

The token payload could be decoded either using a free service like https://jwt.io or using the base64 utility.

Alternative 1: Using https://jwt.io



Paste the token in the "Encoded" section and the "Decoded" section contains the decoded header and payload.

The payload field contains the flag: This-Is-The-Flag-787d1a1636ca

Alternative 2: Using base64 utility.

The payload part of the JWT token is the middle part, that is, eyJpZCI6MiwiZmxhZyI6IlRoaXMtSXMtVGhlLUZsYWctNzg3ZDFhMTYzNmNhIiwiaWF0IjoxNTcyOTgwNTkwLCJleHAiOjE1NzU1NzI1OTB9

Use the following command to decode the token payload:

Command:
echo -n eyJpZCI6MiwiZmxhZyI6IlRoaXMtSXMtVGhlLUZsYWctNzg3ZDFhMTYzNmNhIiwiaWF0IjoxNTcyOTgwNTkwLCJleHAiOjE1NzU1NzI1OTB9 | base64 -d; echo

```
root@attackdefense:~# echo -n eyJpZCI6MiwiZmxhZyI6IlRoaXMtSXMtVGhlLUZsYWctNzg3ZDFhMTYzNmNhIiwiaWF0I
joxNTcyOTgwNTkwLCJleHAiOjE1NzU1NzI1OTB9 | base64 -d; echo
{"id":2,"flag":"This-Is-The-Flag-787d1a1636ca","iat":1572980590,"exp":1575572590}
root@attackdefense:~#
```

Note: The echo command is added to the above command to display the decoded payload in its own line.

The payload contains the flag field having the value "This-Is-The-Flag-787d1a1636ca".

Flag: This-Is-The-Flag-787d1a1636ca

References:

1. Strapi Documentation (https://strapi.io/documentation)
2. JWT debugger (https://jwt.io/#debugger-io)