Second Penetration Test

Contents

Setup

1. Make sure backend branch is on the 'develop' branch and fully up to date with latest version ('git pull')
    a. Run 'python appserver.py' to start running the backend (It is running on port 8000)
2. Make sure frontend branch is likewise, on 'develop' and fully up to date with latest version
    a. Run 'npm start' to start running the frontend (It is running on port 3000)
3. Make sure there are no compile errors, and if there is, fix the compile errors and try and start again
4. Open Kali Linux in a VM
    a. I am using VMware Workstation 16 Player
    b. I am using 'Kali 2021 x64 Cusomized by zSecurity v1.0.4 2'

Initial Scans

Netdiscover

What it does:

This command is used to scan all the IP addresses on the network. Using this command, I am able to then determine what IP addresses link to what devices (printers, other computers, etc.). It is vague at times, and you need to use your intuition with what device belongs to what IP address given. For these tests, it is not needed so much as I already know the IP addresses, it was just done for formalities.

Command used:

`netdiscover -r 192.168.1.0/24`

Nmap

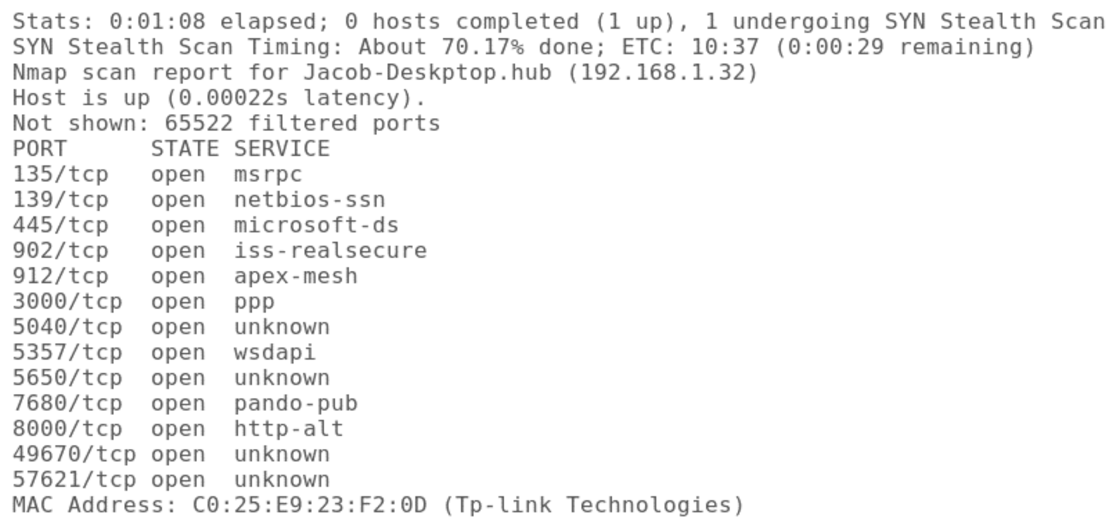
What it does:

This command is used to scan for all the open ports on a given IP address. It can be time consuming, which is why I avoid using the '-A' parameter so that it initially gives me just the basic data to start off with. Again, I know the ports beforehand, however, there may be other, open ports, that are unknown to me and need investigation. The second command we specify all the open ports found in the first command and use the '-A' parameter to search and gain even more data on each port. As seen in the below screenshot, there is a bunch of information for each port that needs further looking into. It is worth noting that some ports reference Microsoft when it is a Python server. And also, worth noting some of these ports may just have appeared because it is running on a Windows computer and in a live environment, wouldn't be open or have that information.

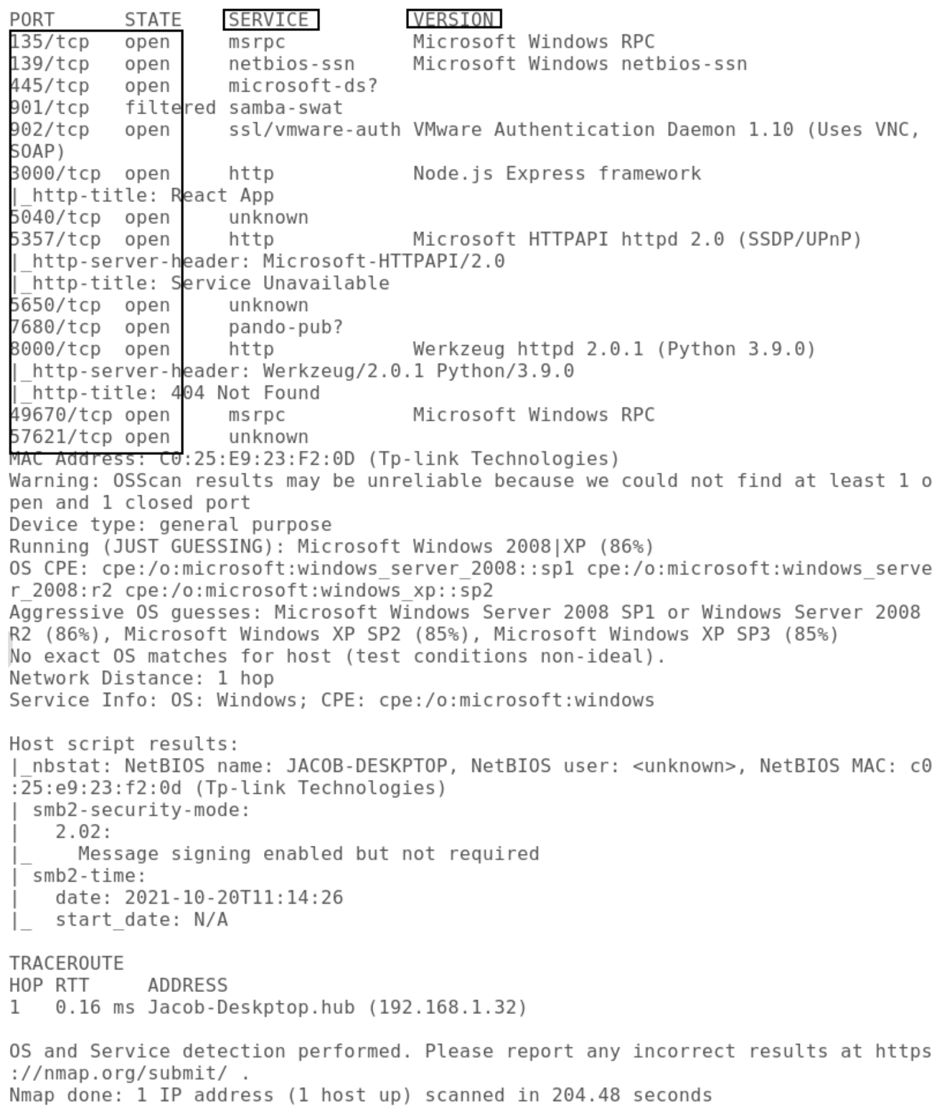
Commands used:

1. *`nmap -T4 -p- 192.168.1.32`*
2. *`nmap –T4 -A -p-135, 139, 445,902, 901,3000, 5040, 5357, 5650, 7680, 8000, 49670, 57621 192.168.1.32`*

Screenshots:



```
Stats: 0:01:08 elapsed; 0 hosts completed (1 up), 1 undergoing SYN Stealth Scan
SYN Stealth Scan Timing: About 70.17% done; ETC: 10:37 (0:00:29 remaining)
Nmap scan report for Jacob-Deskptop.hub (192.168.1.32)
Host is up (0.00022s latency).
Not shown: 65522 filtered ports
PORT      STATE SERVICE
135/tcp   open  msrpc
139/tcp   open  netbios-ssn
445/tcp   open  microsoft-ds
902/tcp   open  iss-realsecure
912/tcp   open  apex-mesh
3000/tcp  open  ppp
5040/tcp  open  unknown
5357/tcp  open  wsdapi
5650/tcp  open  unknown
7680/tcp  open  pando-pub
8000/tcp  open  http-alt
49670/tcp open  unknown
57621/tcp open  unknown
MAC Address: C0:25:E9:23:F2:0D (Tp-link Technologies)
```



```
PORT      STATE    SERVICE        VERSION
135/tcp   open     msrpc          Microsoft Windows RPC
139/tcp   open     netbios-ssn    Microsoft Windows netbios-ssn
445/tcp   open     microsoft-ds?
901/tcp   filtered samba-swat
902/tcp   open     ssl/vmware-auth VMware Authentication Daemon 1.10 (Uses VNC,
SOAP)
3000/tcp  open     http           Node.js Express framework
|_http-title: React App
5040/tcp  open     unknown
5357/tcp  open     http           Microsoft HTTPAPI httpd 2.0 (SSDP/UPnP)
|_http-server-header: Microsoft-HTTPAPI/2.0
|_http-title: Service Unavailable
5650/tcp  open     unknown
7680/tcp  open     pando-pub?
8000/tcp  open     http           Werkzeug httpd 2.0.1 (Python 3.9.0)
|_http-server-header: Werkzeug/2.0.1 Python/3.9.0
|_http-title: 404 Not Found
49670/tcp open     msrpc          Microsoft Windows RPC
57621/tcp open     unknown
MAC Address: C0:25:E9:23:F2:0D (Tp-link Technologies)
Warning: OSScan results may be unreliable because we could not find at least 1 o
pen and 1 closed port
Device type: general purpose
Running (JUST GUESSING): Microsoft Windows 2008|XP (86%)
OS CPE: cpe:/o:microsoft:windows_server_2008::sp1 cpe:/o:microsoft:windows_serve
r_2008:r2 cpe:/o:microsoft:windows_xp::sp2
Aggressive OS guesses: Microsoft Windows Server 2008 SP1 or Windows Server 2008
R2 (86%), Microsoft Windows XP SP2 (85%), Microsoft Windows XP SP3 (85%)
No exact OS matches for host (test conditions non-ideal).
Network Distance: 1 hop
Service Info: OS: Windows; CPE: cpe:/o:microsoft:windows

Host script results:
|_nbstat: NetBIOS name: JACOB-DESKPTOP, NetBIOS user: <unknown>, NetBIOS MAC: c0
:25:e9:23:f2:0d (Tp-link Technologies)
| smb2-security-mode:
|   2.02:
|_    Message signing enabled but not required
| smb2-time:
|   date: 2021-10-20T11:14:26
|_  start_date: N/A

TRACEROUTE
HOP RTT     ADDRESS
1   0.16 ms Jacob-Deskptop.hub (192.168.1.32)

OS and Service detection performed. Please report any incorrect results at https
://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 204.48 seconds
```

Finding:

LOW Finding

- A lot of open ports on the scanned IP address could potentially equal more opportunities for unwanted access. This could be an issue, however, looking through each port, there wasn't anything I could find. However, I am new to penetration testing. It would be worth double checking the ports on the server and which ports are being opened, whether intentional or not.

Curl

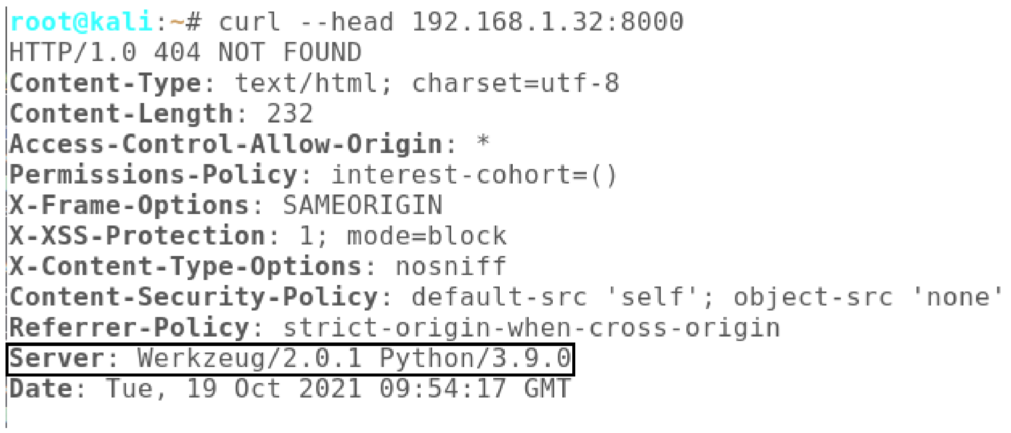
What it does:

The Curl command is used to receive information about a website, and can even be used to send and retrieve data from the server. In this instance, we are using it to retrieve the server header information

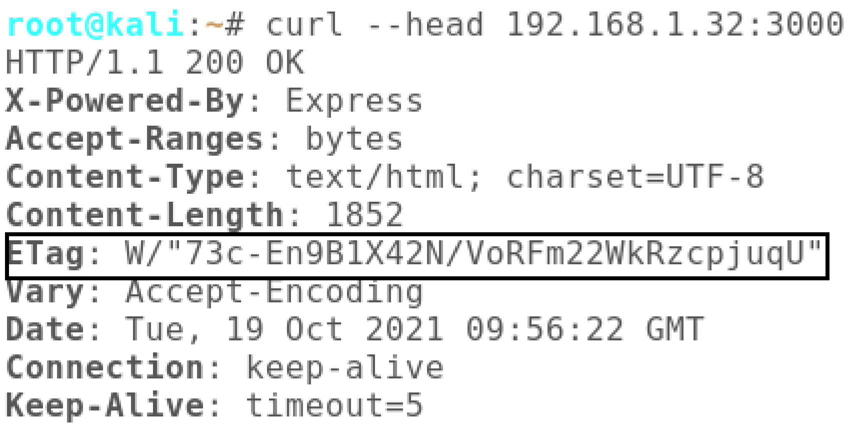
Commands:

1. *`Curl –head 192.1.32:8000`*
2. *`Curl –head 192.1.32:3000`*

Screenshots:



```
root@kali:~# curl --head 192.168.1.32:8000
HTTP/1.0 404 NOT FOUND
Content-Type: text/html; charset=utf-8
Content-Length: 232
Access-Control-Allow-Origin: *
Permissions-Policy: interest-cohort=()
X-Frame-Options: SAMEORIGIN
X-XSS-Protection: 1; mode=block
X-Content-Type-Options: nosniff
Content-Security-Policy: default-src 'self'; object-src 'none'
Referrer-Policy: strict-origin-when-cross-origin
Server: Werkzeug/2.0.1 Python/3.9.0
Date: Tue, 19 Oct 2021 09:54:17 GMT
```



```
root@kali:~# curl --head 192.168.1.32:3000
HTTP/1.1 200 OK
X-Powered-By: Express
Accept-Ranges: bytes
Content-Type: text/html; charset=UTF-8
Content-Length: 1852
ETag: W/"73c-En9B1X42N/VoRFm22WkRzcpjuqU"
Vary: Accept-Encoding
Date: Tue, 19 Oct 2021 09:56:22 GMT
Connection: keep-alive
Keep-Alive: timeout=5
```

Findings:

LOW Finding

- This is revealing server version and some other information here. We can see XSS is blocked but it is worth trying that attack at some point as well to test its effectiveness. Server information can be used to exploit certain versions of the server. It is important to keep the server updated to receive the latest bug patches and make it more difficult to exploit.
- This is also a low finding ETags on their own don't help attacks too much, other than the fact that a few different ETags can help provide some information that can be used elsewhere in the code. ETags are basically encrypted data from the server that can sometimes (not always) contain sensitive information about the server.

Nessus

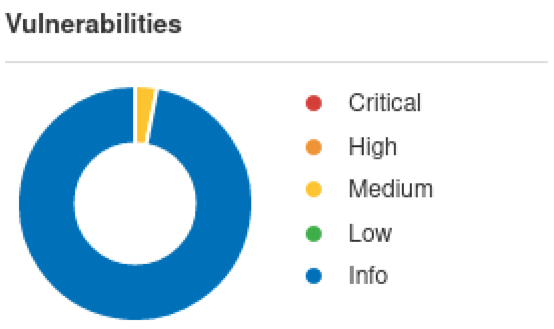
What it does:

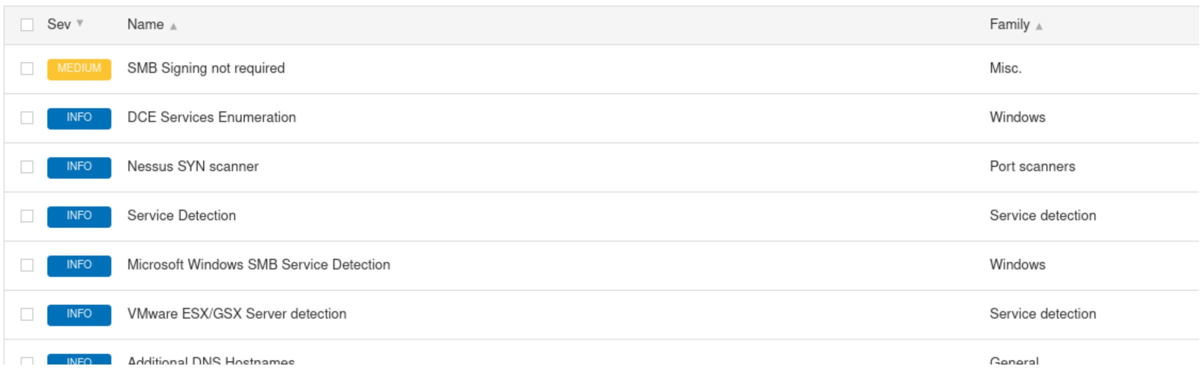
Nessus is an open-source network vulnerability scanner that uses the Common Vulnerabilities and Exposures architecture for easy cross-linking between compliant security tools. In fact, Nessus is one of the many vulnerability scanners used during vulnerability assessments and penetration testing engagements, including malicious attacks. The below command is used to start the Nessus server on the VM and then you can navigate to 'https://kali:8834/' and then perform the scans.

Commands:

`/binsystemctl start nessusd.service`

Screenshots





Sev	Name	Family
MEDIUM	SMB Signing not required	Misc.
INFO	DCE Services Enumeration	Windows
INFO	Nessus SYN scanner	Port scanners
INFO	Service Detection	Service detection
INFO	Microsoft Windows SMB Service Detection	Windows
INFO	VMware ESX/GSX Server detection	Service detection
INFO	Additional DNS Hostnames	General

Findings:

Only one MEDIUM issue found with Nessus scan. This issue is again not as relevant due to not using SMB signing for Python server, as it is more relevant for Microsoft Server or for a Samba server.

Nikto

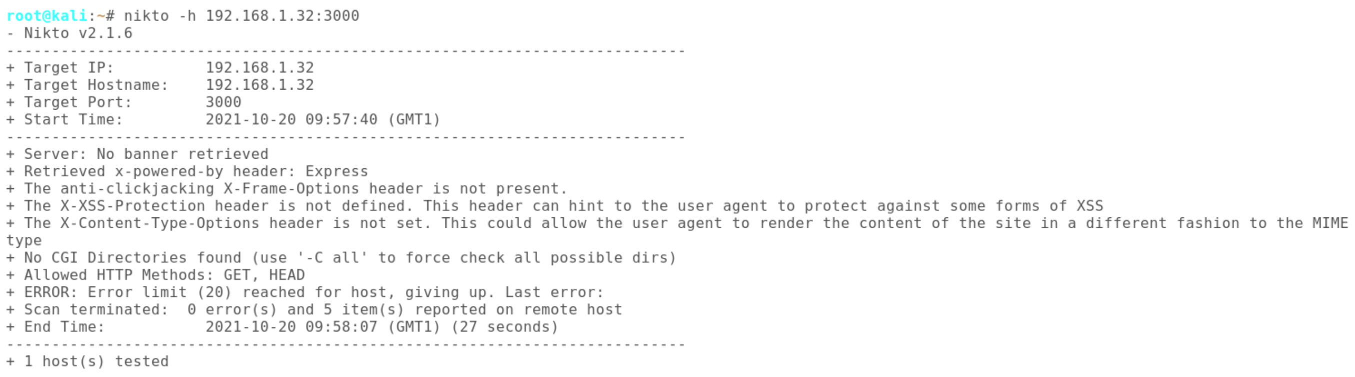
What it does:

The Nikto command is used to examine a web server to find potential problems and security vulnerabilities. As seen in the below screenshot, there are some potential vulnerabilities for the port 3000 and 8000 that would need further investigating and potential defence against. It is worth noting that nothing substantial was found through surface level investigating.

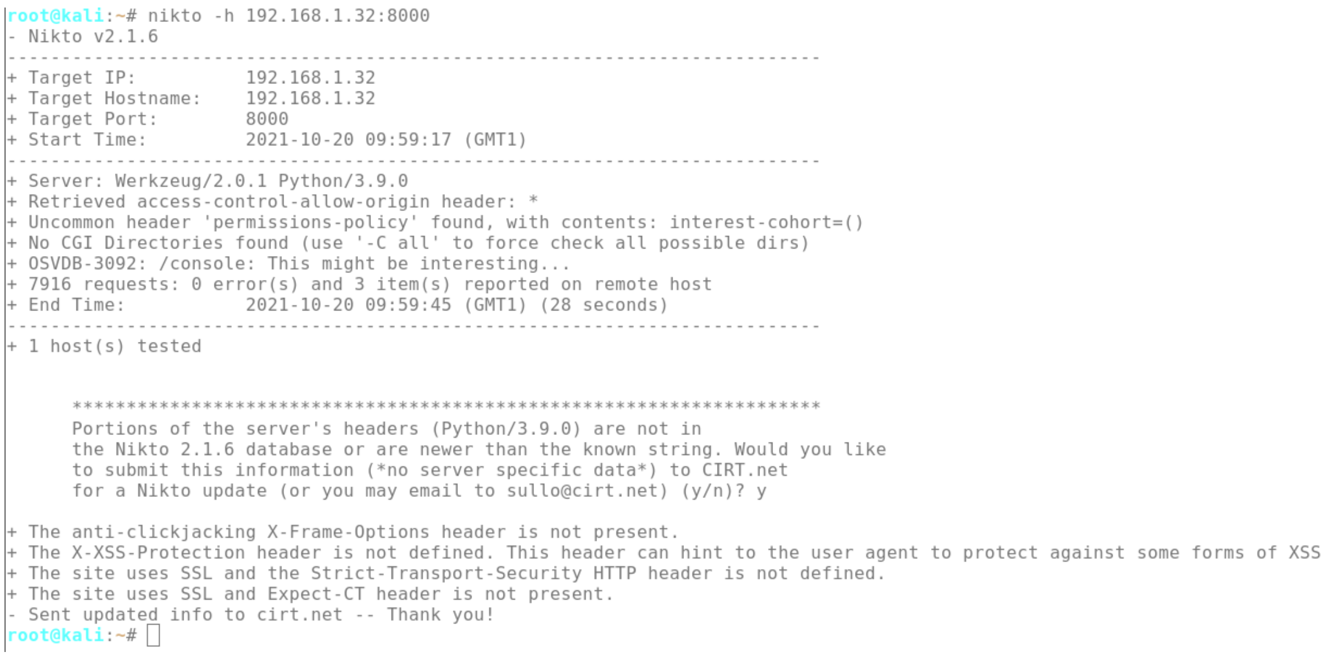
Commands

'nikto -h 192.168.1.32:3000'

Screenshots:



```
root@kali:~# nikto -h 192.168.1.32:3000
- Nikto v2.1.6
---------------------------------------------------------------------------
+ Target IP:          192.168.1.32
+ Target Hostname:    192.168.1.32
+ Target Port:        3000
+ Start Time:         2021-10-20 09:57:40 (GMT1)
---------------------------------------------------------------------------
+ Server: No banner retrieved
+ Retrieved x-powered-by header: Express
+ The anti-clickjacking X-Frame-Options header is not present.
+ The X-XSS-Protection header is not defined. This header can hint to the user agent to protect against some forms of XSS
+ The X-Content-Type-Options header is not set. This could allow the user agent to render the content of the site in a different fashion to the MIME
type
+ No CGI Directories found (use '-C all' to force check all possible dirs)
+ Allowed HTTP Methods: GET, HEAD
+ ERROR: Error limit (20) reached for host, giving up. Last error:
+ Scan terminated:  0 error(s) and 5 item(s) reported on remote host
+ End Time:           2021-10-20 09:58:07 (GMT1) (27 seconds)
---------------------------------------------------------------------------
+ 1 host(s) tested
```



```
root@kali:~# nikto -h 192.168.1.32:8000
- Nikto v2.1.6
---------------------------------------------------------------------------
+ Target IP:          192.168.1.32
+ Target Hostname:    192.168.1.32
+ Target Port:        8000
+ Start Time:         2021-10-20 09:59:17 (GMT1)
---------------------------------------------------------------------------
+ Server: Werkzeug/2.0.1 Python/3.9.0
+ Retrieved access-control-allow-origin header: *
+ Uncommon header 'permissions-policy' found, with contents: interest-cohort=()
+ No CGI Directories found (use '-C all' to force check all possible dirs)
+ OSVDB-3092: /console: This might be interesting...
+ 7916 requests: 0 error(s) and 3 item(s) reported on remote host
+ End Time:           2021-10-20 09:59:45 (GMT1) (28 seconds)
---------------------------------------------------------------------------
+ 1 host(s) tested


      *********************************************************************
      Portions of the server's headers (Python/3.9.0) are not in
      the Nikto 2.1.6 database or are newer than the known string. Would you like
      to submit this information (*no server specific data*) to CIRT.net
      for a Nikto update (or you may email to sullo@cirt.net) (y/n)? y

+ The anti-clickjacking X-Frame-Options header is not present.
+ The X-XSS-Protection header is not defined. This header can hint to the user agent to protect against some forms of XSS
+ The site uses SSL and the Strict-Transport-Security HTTP header is not defined.
+ The site uses SSL and Expect-CT header is not present.
- Sent updated info to cirt.net -- Thank you!
root@kali:~# 
```

SQL Injection

What is it:

Basic SQL injection was tried on both the login screen and the profile creation screen without any effect. We also tried using the kali tool called 'SQLMap' to try automated SQL injections in which it failed. I am an amateur at this currently and therefore this might need further investigating. The below command was used for the login screen and profile creation. And this following command was run in the terminal to test out the tool 'SQLMap'

Commands:

1. Select * from ? where ? like '%'
2. *Python3 sqlmap.py -u 192.168.1.32 –batch –banner*

Protection against SQL Injection:

To protect against SQL attacks, it is best to use forms on all of your inputs and validate every input from the user. It is also important to remove and not allow certain characters and be careful in how you pass the data to the server to process. A good philosophy should be to not trust any input from the user as you never know what they will input and even if it isn't an SQL injection attack, it may still cause errors or bugs

XSS Attack

These attacks were also tried, using the Kali tool 'Beef' which was run in the browser. This is also used to pass unvalidated data to the server in which it is processed and can cause unwanted effects, such as changing the style formatting of the frontend, to passing malicious scripts to the server, in which can get run when certain calls are made. These can have a large array of consequences.

Protections against XSS Attacks

As noted in SQL injection, it is important to never trust any input from the user and validate everything the user passes in and use forms. Something some people may miss during development is double checking the URL and data passed in through that. Malicious code could be passed as parameters through the URL bar and cause unwanted effects. Double check your error pages and make sure to validate inputs through the URL bar.