Group 2: Phase 3 Report

Lockwood Topping, Eric Chapman, Tre Germany, Joseph Daher, & Manasa Mutpur
Department of Cybersecurity, Kennesaw State University
CYBR 7910: Capstone in Cybersecurity Practicum
Dr. Zhigang Li
December 2, 2024

Project Status Update

Since the last report, we have validated that all our server's security configurations are functioning correctly while ensuring the server remains operational. A final restore point of the server was created before engaging in the attack phase. On November 19, 2024, we received the target server's IP address, 10.96.33.31, and commenced reconnaissance activities. Our reconnaissance confirmed the absence of HTTPS enforcement, exposure of directory listings, and the continued use of default WordPress and root credentials. Following the discovery phase, we leveraged various penetration testing tools, including Burpsuite, Hydra, and custom-crafted wordlists, to conduct targeted attacks on the WordPress administrative portal and SSH access. Ultimately, we identified multiple critical vulnerabilities in the target system. Additionally, on our system, we logged many attack attempts but our defensive configuration proved effective and no successful penetration occurred. When we got the report from the other team, they mentioned the main vulnerability being that some of our software, specifically apache, was showing as outdated, which was the vulnerability we had decided to leave for them to find.

Our project plan has not changed since the last update. We completed the project ahead of schedule due to the time buffer incorporated into our weekly timeline. Following the rapid and successful penetration of the other team's server, we dedicated the remaining time to compiling and refining our work for a 15-minute presentation. To ensure quality, we assigned time limits for each team member's portion, allowed each person to record and perfect their part of the video, and then combined the recordings into a cohesive final product. Our proactive planning helped us avoid potential setbacks from the Thanksgiving holiday, which ultimately had no impact on our timeline.

Vulnerability Analysis & Penetration Testing Report

Prepared For: Group 1 Server (10.96.33.31)
Prepared By: Group 2
Date: November 20, 2024

1. Executive Summary

This report details the vulnerabilities discovered during the penetration testing exercise conducted on the target server at **10.96.33.31**. Multiple critical vulnerabilities were identified, including the use of default credentials, lack of HTTPS enforcement, and publicly accessible sensitive directories. Exploitation of these weaknesses allowed for complete compromise of the target system, including root-level access, administrative control over WordPress, and full access to the MariaDB database. Immediate remediation is recommended to prevent future exploitation.

2. Objectives

- Assess the security posture of the target server.
- Identify vulnerabilities that could lead to unauthorized access or data breaches.
- Test the server's resistance to various attack techniques.
- Provide actionable recommendations to enhance security.

3. Methodology

The testing process followed a structured approach, which included:

1. **Reconnaissance**: Gathering information about the target system.
2. **Exploitation**: Attempting to gain unauthorized access using identified vulnerabilities.
3. **Post-Exploitation**: Verifying the extent of access and extracting critical data.

4. Reconnaissance Phase

4.1 Phishing Attack (see Figure A1 in Appendix A)

- A phishing email spoofing the professor was sent to request changed credentials.
- Result: **Unsuccessful**; no response was received.

4.2 Web Enumeration (see Figure A2 in Appendix A)

- The website did not enforce HTTPS, allowing unsecured communication.
- Sensitive directories, such as /home/wp-includes, were publicly accessible.
- Default WordPress credentials (username: ksuitg5, password: Kennesaw123!) successfully authenticated the WordPress admin portal.

4.3 Port Scanning (see Figure A3 in Appendix A)

- **Open Ports Identified**:
  - Port 22 (SSH): Running OpenSSH 8.0.
  - Port 80 (HTTP): Running Apache 2.4.37.
  - Port 3306 (MySQL/MariaDB): Running MySQL 5.5.5.
- A detailed Nmap scan confirmed the versions of these services.

4.4 Vulnerability Scanning (see Figures A4, A5, A6 in Appendix A)

- Tools used: **searchsploit, dirb**, **nikto**, **wpscan**, and **curl**.
- Results:
    - Many exploitations were tied to the current versions of the server's listed services via searchsploit
    - Directory listings successfully enumerated using dirb
    - No significant additional vulnerabilities were identified via nikto or wpscan.
    - WordPress API exposed plugin versions and usernames, revealing the admin account.

5. Exploitation Phase

5.1 WordPress Admin Portal (see Figure A7 in Appendix A)

- Tools used: **Burpsuite**, **Hydra**, and custom wordlists.
- Default credentials (username: ksuitg5, password: Kennesaw123!) provided access to the admin portal.

5.2 SSH Access (see Figure A8 in Appendix A)

- Credentials discovered:
    - **Username**: root
    - **Password**: cyberadmin01
- Tools used: **Hydra** with custom wordlists.
- Result: Full root access, enabling unrestricted control of the server.

5.3 MariaDB Database (see Figures A9, A10 in Appendix A)

- Default credentials discovered:
    - **Username**: admin
    - **Password**: pass
- Direct access via port 3306 failed due to misconfiguration.
- Exploitation: An SSH tunnel was established to bypass the restriction, enabling Hydra to crack credentials.
- Access to the wp_users table revealed the hashed password for the WordPress admin account, which was cracked using **John the Ripper**.

6. Findings

6.1 Critical Vulnerabilities

- **Default Credentials**:
    - SSH (root: cyberadmin01)
    - WordPress (ksuitg5: Kennesaw123!)
    - MariaDB (admin: pass)
- **Unsecured Web Traffic**: No HTTPS enforcement.
- **Public Directory Listings**: Sensitive directories exposed.
- **Outdated Software**: Apache 2.4.37, OpenSSH 8.0, and WordPress plugins were not updated.
- **Exposed WordPress API**: Plugins and usernames disclosed without authentication.

6.2 Post-Exploitation Risks

- Full server compromise, including root and database access.
- No firewall or WAF to prevent brute force attacks.

6.3 Unsuccessful Attempt

- Phishing attack did not yield results, likely due to unchanged default credentials.

7. Recommendations

7.1 Credential Management

- Change all default credentials immediately.
- Enforce strong password policies (minimum length, complexity, and expiration).
- Implement multi-factor authentication (MFA) for WordPress and SSH.
- Create non-administrative users for WordPress, MariaDB, and SSH

7.2 Network Security

- Enforce HTTPS using a valid SSL certificate.
- Deploy a Web Application Firewall (WAF) to mitigate brute force and other automated attacks.
- Close unnecessary ports, particularly **3306**, unless required for specific operations.

7.3 System Updates

- Regularly update all server software, including Apache, WordPress, and OpenSSH.
- Audit and update WordPress plugins and themes to their latest secure versions.

7.4 File and Directory Protections

- Disable public directory listings in Apache configuration.
- Apply restrictive file permissions to sensitive directories and files.

8. Conclusion

The target server exhibited multiple critical vulnerabilities that allowed for complete system compromise. These weaknesses underscore the importance of basic security hygiene, including password management, software updates, and network segmentation. By addressing the outlined recommendations, the server's resilience against future attacks can be significantly improved.

Forensic Report on Security Operations of Protected Web Server

Overview:

On November 21, 2024, the server was subjected to a series of attempted attacks targeting various services, including SSH and the WordPress website. The server is protected by several layers of security, including Cockpit, Wordfence, Wazuh, Fail2Ban, Snort, and secure system configurations. The attack attempts included SSH brute force attacks, local file inclusion (LFI), cross-site scripting (XSS), and directory traversal attacks. Despite the efforts, there were no successful penetrations into the system, and all malicious actions were blocked or logged for further analysis.

We also received a vulnerability report of our server from the other team. You may view the whole report in Appendix C.

1. Cockpit Logs: (see Figure B1 in Appendix B)

- At 4:11 PM on November 21, 2024, Cockpit logged multiple unsuccessful SSH login attempts using the root user, which had already been disallowed from logging in through SSH. The repeated login attempts were blocked automatically, with no successful login recorded except from the authorized personal IP address using an authorized SSH account.
- **Security Measures**: Root login was disabled for SSH access, effectively preventing any successful login from unauthorized users.

2. Wordfence Logs (WordPress Security Plugin): (see Figure B2 in Appendix B)

- Between 4:21:33 PM and 4:21:58 PM on November 21, Wordfence blocked 16 attacks from the IP address 172.27.14.115. The attacks were of the following types:
    - Directory Traversal in query string
    - Local File Inclusion (LFI) in query string
    - Cross-Site Scripting (XSS) in query string
- **Outcome**: Wordfence successfully blocked all malicious requests, and no login attempts to the wp-admin page were successful, confirming that the attackers could not gain access to the WordPress admin interface.
- **Security Measures**: Wordfence's WAF effectively mitigated the malicious queries targeting vulnerabilities in WordPress.

3. Wazuh Manager Logs (Intrusion Detection and Response): (see Figure B3 in Appendix B)

- Wazuh alerted to a **Reconnaissance attack**, correlating with the SSH attack attempts from the IP 172.27.14.115.
- **Fail2Ban** identified multiple failed SSH login attempts from the same IP and temporarily banned it for 10 minutes. After the unban, no further attempts from that IP were observed.
- **Security Measures**: Fail2Ban was effective in blocking the attacker's IP after repeated SSH login failures, preventing further brute force attempts.

4. System Logs: (see Figures B4, B5, B6 in Appendix B)

- **Secure Log**: The secure log documented the failed SSH login attempts from IP 172.27.14.115. These attempts were part of the brute force attack, which was blocked by Fail2Ban after several failed login attempts.
- **HTTPD Logs**:
  - **Access Log**: The access log showed web page curls originating from the other team's server (10.96.33.31), with a few requests from the server made to various pages on the site.
  - **Error Log**: An incorrect page request to /var/www/html/page was logged from the other team's server, indicating a possible scanning or testing activity.
- **ModSecurity Logs**: The modsec_audit.log confirmed that all malicious requests (including those blocked by Wordfence) were detected and blocked by ModSecurity, ensuring no harmful requests were processed by the server.
- **Firewall and MariaDB Logs**: There were no relevant entries in the firewall or MariaDB logs, indicating no successful intrusion attempts via these services.
- **Syslog and Snort IDS Logs**: The syslog contained alerts from Snort IDS/IPS, which logged attempted administrator privilege gains from multiple ports associated with the attacker's IP 172.27.14.115. These attempts were detected by the intrusion detection system but did not result in any successful exploitation.

5. Root History Logs:

- The /root/.bash_history file showed no additional commands executed during the attack period, indicating that the attackers did not gain root access or execute any commands on the system.

6. Overall Attack Analysis:

- The logs indicate multiple attack vectors, including SSH brute force attempts, local file inclusion (LFI), cross-site scripting (XSS), and directory traversal attacks. However, due to the layered security mechanisms in place (including root SSH access restrictions, Fail2Ban, Wordfence, Snort, and ModSecurity), all attacks were successfully blocked or mitigated.
- The attackers' IP 172.27.14.115 was consistently monitored and blocked by various tools, and no successful login, privilege escalation, or unauthorized command execution was recorded.

7. Response to the Vulnerability Report from the Other Team: (see Appendix C)

The vulnerability report provided by the other team outlined several findings from their penetration testing efforts. Below are their findings, followed by our responses:

- **Outdated Apache and SSH Versions:**
  - **Finding**: The report highlighted that the Apache server and SSH service were not updated to their latest versions.
  - **Response**: This vulnerability was deliberately left for the attacking team to identify, as part of the lab's learning objectives. To resolve this vulnerability, the

apache and openssh packages should be updated or upgraded to the current stable version.

- **Weak or Default Passwords for SSH:**
  - **Finding**: The team speculated that SSH may be configured with weak or default passwords, potentially increasing the risk of compromise.
  - **Response**: This speculation is incorrect. All usernames had been changed and passwords were intentionally created to be strong and further secured with multi-factor authentication (MFA), ensuring robust protection against brute force attacks.
- **Private IP Address Exposure:**
  - **Finding**: The report noted that private IP addresses were visible during the scans, indicating potential information leakage.
  - **Response**: The lab environment operates entirely within a private network. The exposure of private IP addresses is not a vulnerability in this context.
- **Firewall Configuration on Open Ports:**
  - **Finding**: The team mentioned that while the firewall was active, it did not filter certain open ports.
  - **Response**: While the firewall could be configured to filter additional traffic, the open ports were necessary for the lab's functionality and testing purposes. No malicious access was successful through these open ports during the testing period.
- **Leaked Information**
  - **Finding**: The team mentioned software versions and directories with file names were exposed on the server.
  - **Response**: This was unintentional, and can be corrected by modifying the .htaccess and wp-config files to further restrict which files are visible to unauthorized users.
- **General Observations and Commendations:**
  - Multi-factor authentication effectively secured the SSH and WordPress services.
  - Database configurations prevented SQL injection attacks.
  - Blocking ECHO pings was a smart move to limit reconnaissance capabilities.

8. Conclusion:

- **No successful penetration occurred**. The server's security mechanisms, including Fail2Ban, Wordfence, Snort, and Cockpit, effectively prevented the attackers from gaining unauthorized access or exploiting vulnerabilities.
- The attacker's activities were limited to reconnaissance and attempted exploitation, all of which were blocked or logged for further analysis.

Appendix A: Vulnerability Analysis Pictures

Figure A1

The custom phishing email used to prod for confidential credentials

[EXTERNAL] Password Changes

Zhigang Li <zli8@kennesaw.edu>
To Lockwood Topping

Reply | Reply A

We could not verify the identity of the sender. Click here to learn more.

Attention Class,

If you have made any changes to the default administrative SSH, MariaDB, and WordPress passwords on your server, please send an email with your Group number, server IP, and updated usernames/passwords to: CYBR7910.FA24.ProjectRecords@gmail.com.

This will help me monitor the progress of the attacks and defenses on your servers. If you have any questions, feel free to reach out through D2L or during my office hours.

Thank you!

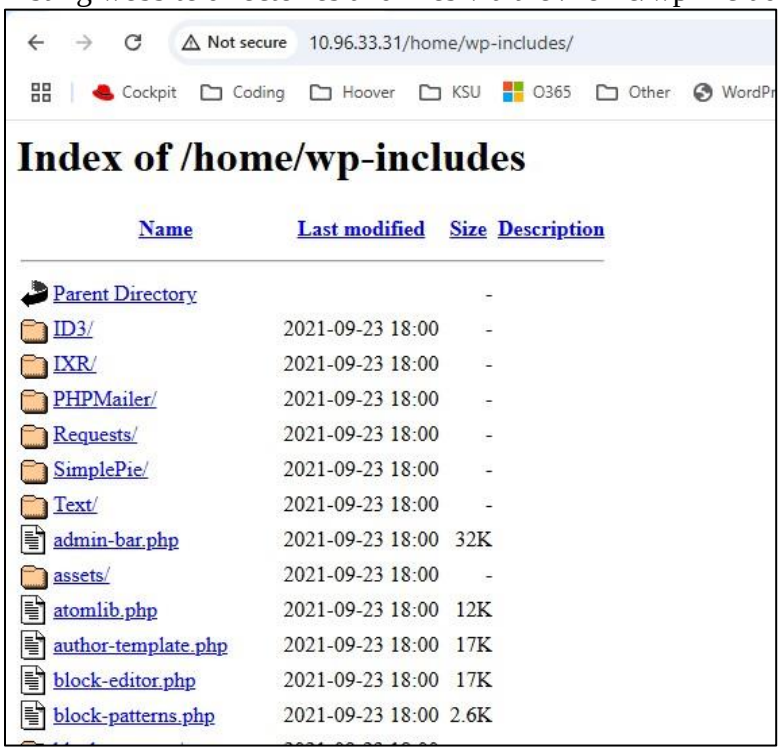
Zhigang Li, Ph.D.
Associate Professor & Assistant Chair
Department of Information Technology
College of Computing and Software Engineering
470-578-4481

[This email has been forwarded from D2L]

CAUTION: This email originated from outside of KSU. Use caution replying or supplying information, clicking links, or opening attachments. If you suspect the message is fraudulent, report it using the Phishing Ale
links.protection.outlook.com/?url=https%3A%2F%2Fuits.kennesaw.edu%2Focs%2Fphish-market%2Freport.php&data=05%7C02%7Cjtopping%40students.kennesaw.edu%7C84713244b0fb41a18efe08dd0979d4a4%
7C45f26ee5f134439ebc93e6c7e33d61c2%7C1%7C0%7C638677141078855986%7CUnknown%7CTWFpbGZsb3d8eyJFbXB0eU1hcGkiOnRydWUsIlYiOiIwLjAuMDAwMCIsIlAiOiJXaW4zMiIsIkFOIjoiTWFpbCIsIldUIjoyfQ%
7C&sdata=1ynsxbkYJCorjvr8Hl6YQwgKsQuanXd4qTsuQpptk1g%3D&reserved=0> or forward the message to abuse@kennesaw.edu<mailto:abuse@kennesaw.edu>

Figure A2

Listing website directories and files via the /home/wp-includes page



Not secure 10.96.33.31/home/wp-includes/

Cockpit | Coding | Hoover | KSU | O365 | Other | WordPr

Index of /home/wp-includes

Name	Last modified	Size	Description
Parent Directory		-	
ID3/	2021-09-23 18:00	-	
IXR/	2021-09-23 18:00	-	
PHPMailer/	2021-09-23 18:00	-	
Requests/	2021-09-23 18:00	-	
SimplePie/	2021-09-23 18:00	-	
Text/	2021-09-23 18:00	-	
admin-bar.php	2021-09-23 18:00	32K	
assets/	2021-09-23 18:00	-	
atomlib.php	2021-09-23 18:00	12K	
author-template.php	2021-09-23 18:00	17K	
block-editor.php	2021-09-23 18:00	17K	
block-patterns.php	2021-09-23 18:00	2.6K	

Figure A3

Using nmap to uncover the services running on the server's open ports

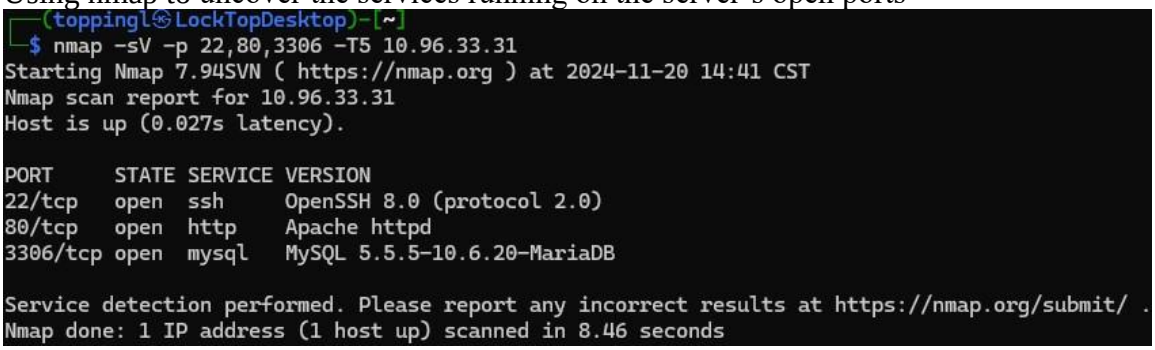


Figure A4

Using searchsploit to find known exploits for services running on the server

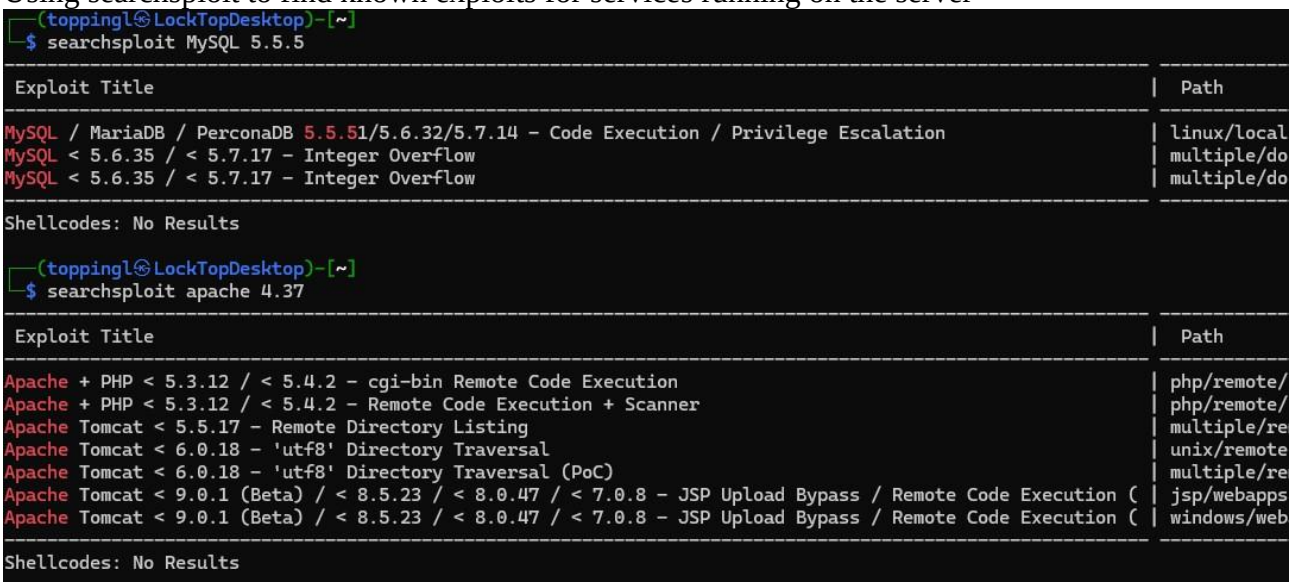


Figure A5

Using dirb to enumerate a list of potential directories and files on the server

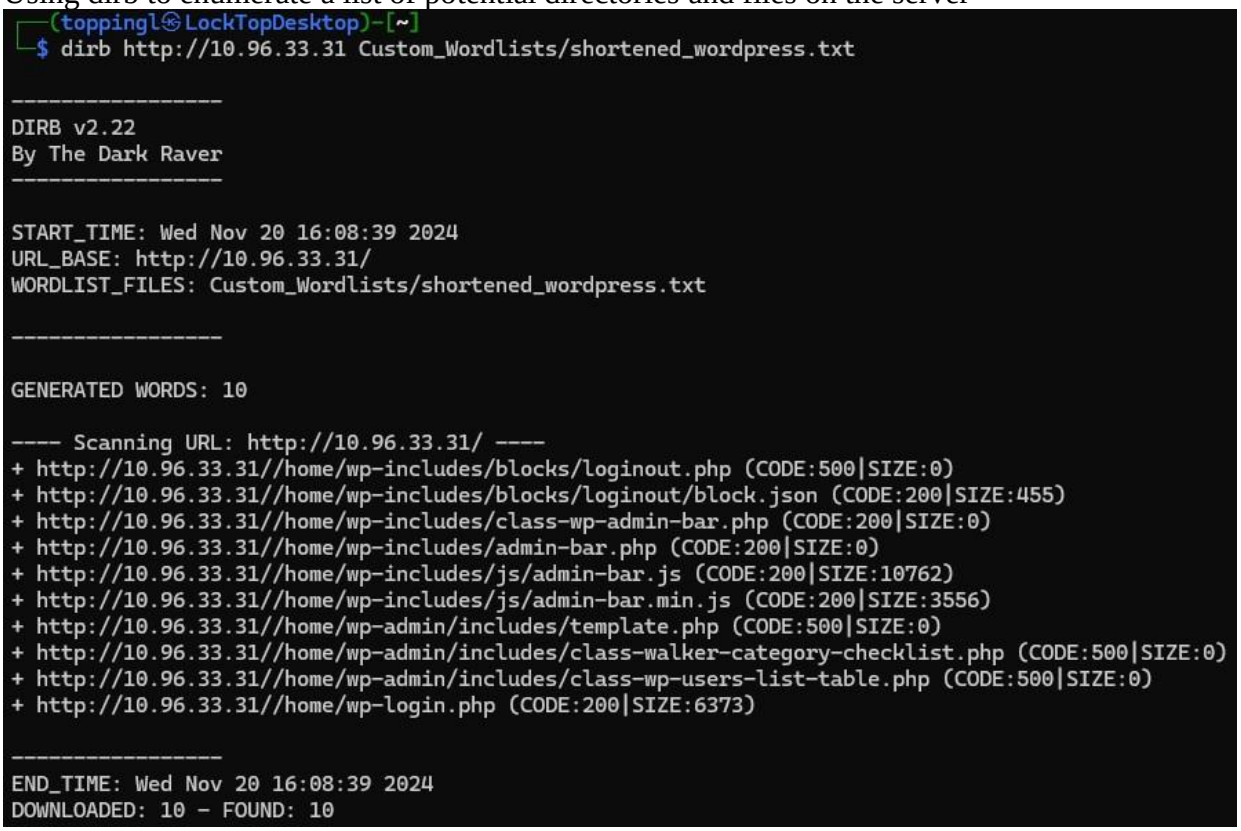


Figure A6

Using curl to interact with the WordPress API

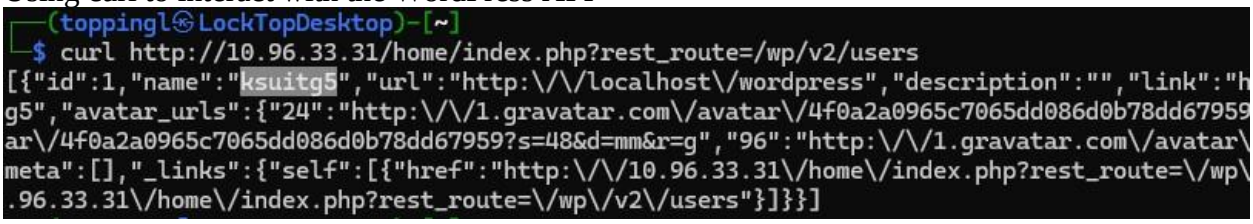


Figure A7
Cracking WordPress admin credentials using custom packets created in Burpsuite

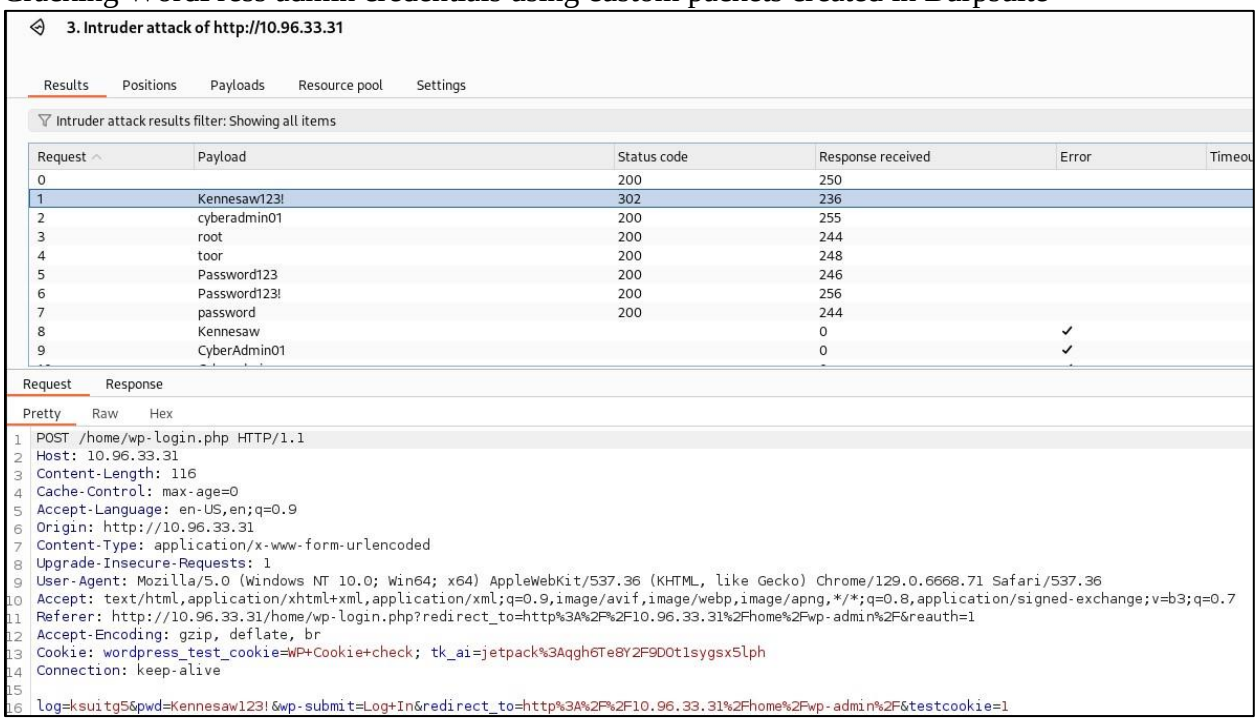


Figure A8
Using hydra to crack ssh credentials

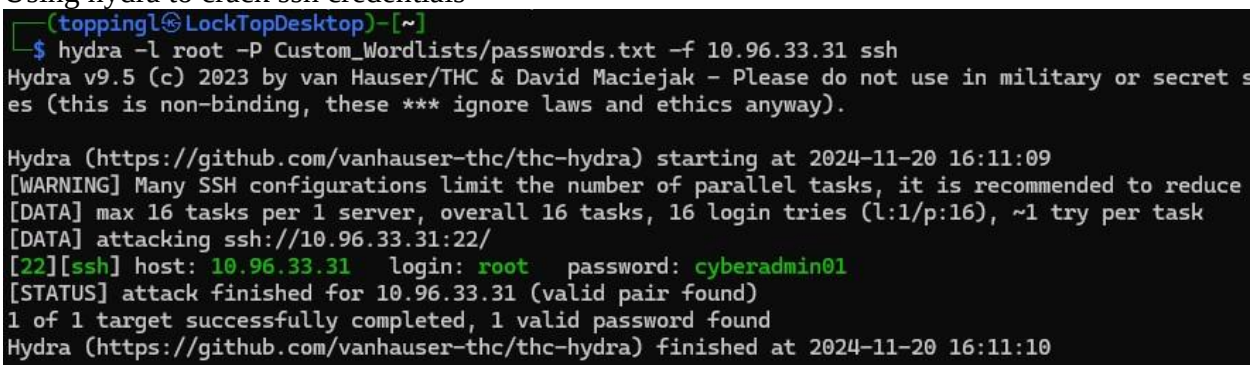


Figure A9
Using hydra and an ssh shell to crack the MariaDB user account

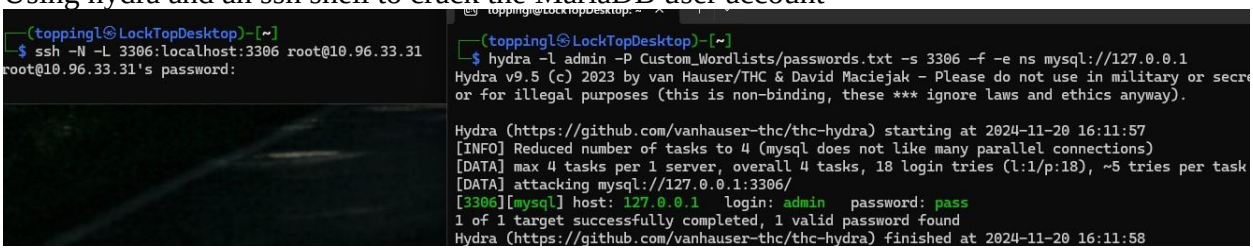
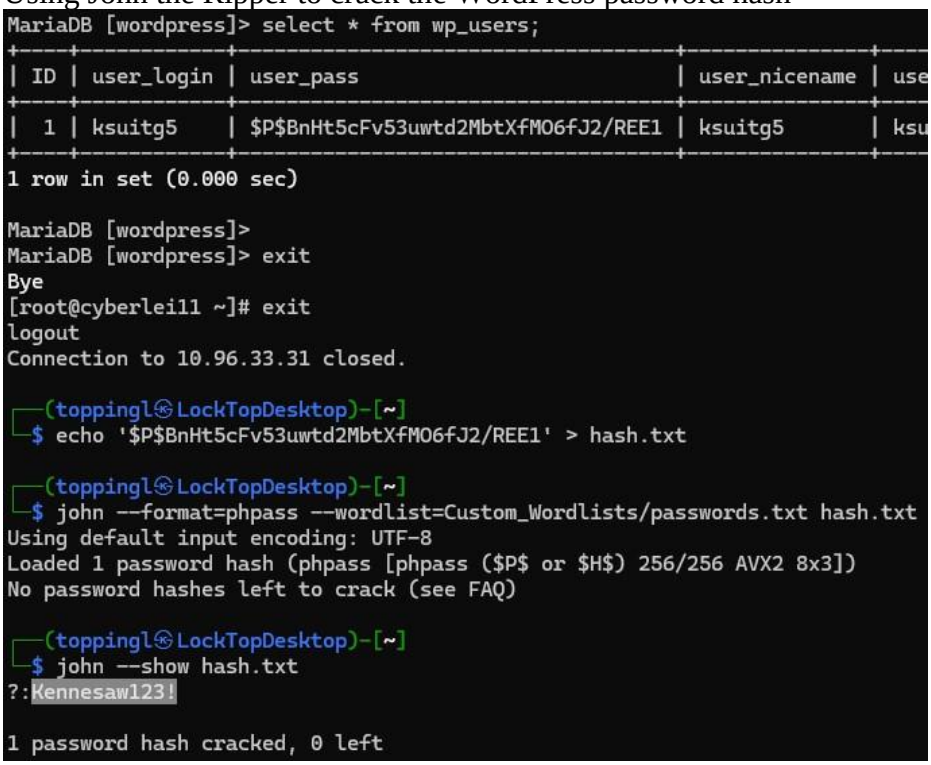


Figure A10

Using John the Ripper to crack the WordPress password hash



Appendix B: Forensic Report Pictures

Figure B1
SSH failure logs showing in Cockpit

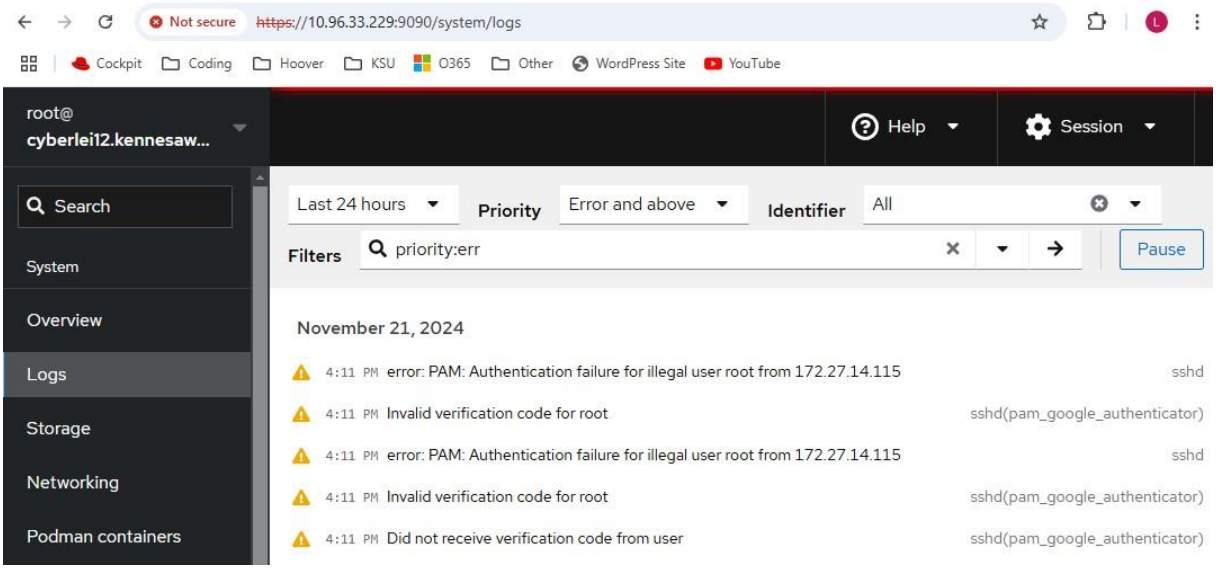


Figure B2
Failed WordPress attack showing in Wordfence

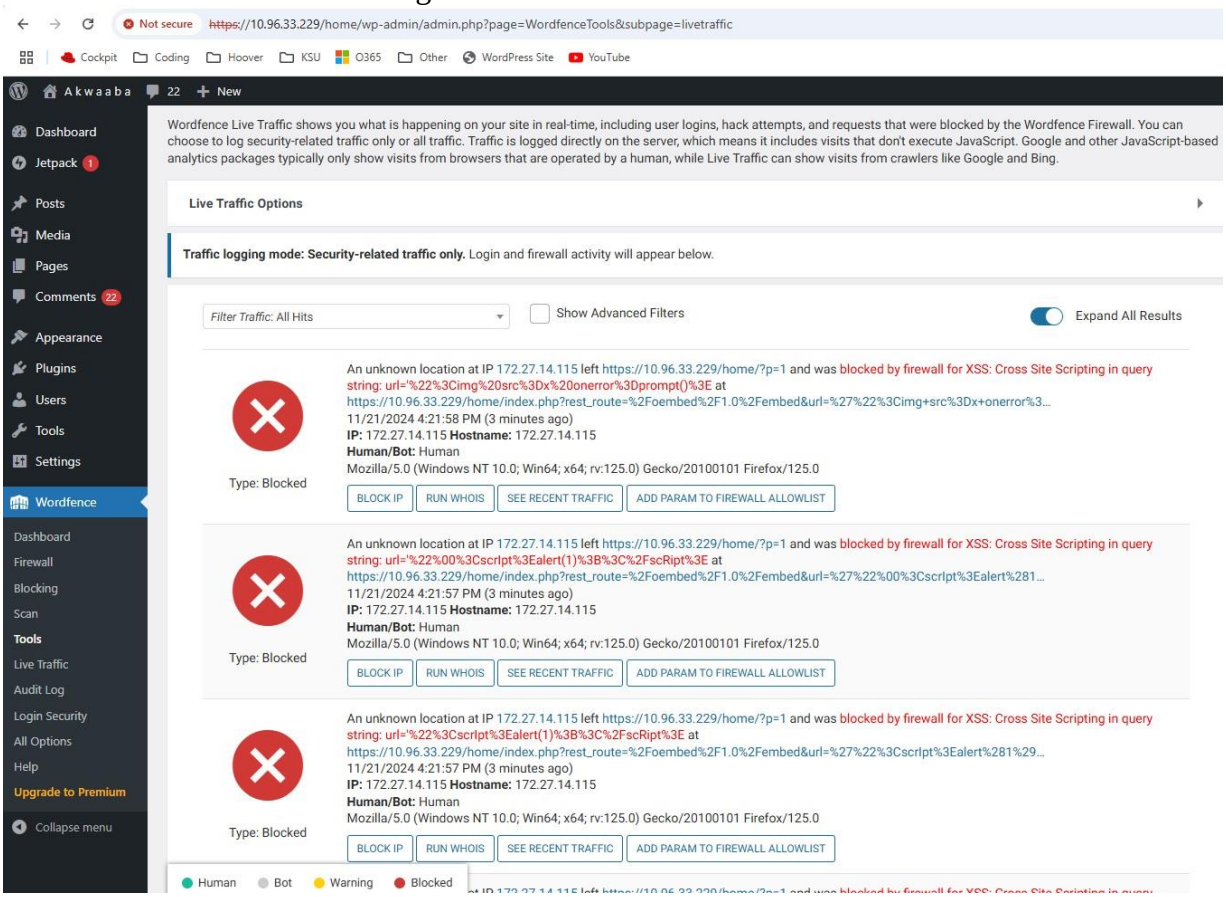


Figure B3

Reconnaissance and Fail2Ban alerts showing in Wazuh Manager

```
[root@cyberleil2 ~]# grep Reconnaissance /var/ossec/logs/alerts/alerts.json |  tail -n 1 | jq -c
{"timestamp":"2024-11-22T12:10:56.275-0500","rule":{"level":10,"description":"Multiple web server 400 error codes from same source ip.","id":"31151","mitre"
:{"id":["T1595.002"],"tactic":["Reconnaissance"],"technique":["Vulnerability Scanning"]},"frequency":14,"firedtimes":621,"mail":false,"groups":["web","acces
slog","web_scan","recon"],"pci_dss":["6.5","11.4"],"gdpr":["IV_35.7.d"],"nist_800_53":["SA.11","SI.4"],"tsc":["CC6.6","CC7.1","CC8.1","CC6.1","CC6.8","CC7.2
","CC7.3"]},"agent":{"id":"000","name":"cyberleil2.kennesaw.edu"},"manager":{"name":"cyberleil2.kennesaw.edu"},"id":"1732295456.7011077","previous_output":"
172.27.12.178 - - [22/Nov/2024:12:10:55 -0500] \"GET http://aws.cirt.net/metadata/instance?api-version=2017-08-01 HTTP/1.1\" 404 196 \"-\" \"Mozilla/5.0 (Wi
ndows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/74.0.3729.169 Safari/537.36\"\n172.27.12.178 - - [22/Nov/2024:12:10:55 -0500] \"GET
 http://aws.cirt.net/metadata/v1.json HTTP/1.1\" 404 196 \"-\" \"Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/74.
0.3729.169 Safari/537.36\"\n172.27.12.178 - - [22/Nov/2024:12:10:55 -0500] \"GET http://aws.cirt.net/openstack/latest HTTP/1.1\" 404 196 \"-\" \"Mozilla/5.0
 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/74.0.3729.169 Safari/537.36\"\n172.27.12.178 - - [22/Nov/2024:12:10:54 -0500] \
"GET http://aws.cirt.net/computeMetadata/v1/project/ HTTP/1.1\" 404 196 \"-\" \"Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Ge
cko) Chrome/74.0.3729.169 Safari/537.36\"\n172.27.12.178 - - [22/Nov/2024:12:10:54 -0500] \"GET http://169.254.169.254/openstack/latest HTTP/1.1\" 404 196 \
"-\" \"Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/74.0.3729.169 Safari/537.36\"\n172.27.12.178 - - [22/Nov/2024
:12:10:54 -0500] \"GET http://aws.cirt.net/latest/meta-data/ HTTP/1.1\" 404 196 \"-\" \"Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML,
 like Gecko) Chrome/74.0.3729.169 Safari/537.36\"\n172.27.12.178 - - [22/Nov/2024:12:10:54 -0500] \"GET http://169.254.169.254/computeMetadata/v1/project/ H
TTP/1.1\" 404 196 \"-\" \"Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/74.0.3729.169 Safari/537.36\"\n172.27.12.1
78 - - [22/Nov/2024:12:10:54 -0500] \"GET http://rancher-metadata/2015-07-25/ HTTP/1.1\" 400 226 \"-\" \"-\"\n172.27.12.178 - - [22/Nov/2024:12:10:54 -0500]
 \"GET http://169.254.169.254/latest/dynamic/instance-identity/document HTTP/1.1\" 404 196 \"-\" \"Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537
.36 (KHTML, like Gecko) Chrome/74.0.3729.169 Safari/537.36\"\n172.27.12.178 - - [22/Nov/2024:12:10:54 -0500] \"GET http://169.254.169.254/latest/meta-data/ 
HTTP/1.1\" 404 196 \"-\" \"Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/74.0.3729.169 Safari/537.36\"\n172.27.12.
178 - - [22/Nov/2024:12:10:54 -0500] \"GET /shopdbtest.asp HTTP/1.1\" 404 196 \"-\" \"Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, l
ike Gecko) Chrome/74.0.3729.169 Safari/537.36\"","full_log":"172.27.12.178 - - [22/Nov/2024:12:10:55 -0500] \"GET http://aws.cirt.net/hetzner/v1/metadata/pr
ivate-networks HTTP/1.1\" 404 196 \"-\" \"Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/74.0.3729.169 Safari/537.3
6\"","decoder":{"name":"web-accesslog"},"data":{"protocol":"GET","srcip":"172.27.12.178","id":"404","url":"http://aws.cirt.net/hetzner/v1/metadata/private-n
etworks"},"location":"/var/log/httpd/access_log"}
[root@cyberleil2 ~]# grep fail2ban /var/ossec/logs/alerts/alerts.json | grep -iv unbanned | tail -n 1 | jq -c
{"timestamp":"2024-11-22T10:29:24.962-0500","rule":{"level":6,"description":"Fail2Ban: IP banned","id":"100002","firedtimes":33,"mail":false,"groups":["loca
l","syslog","fail2bansyslog","authentication_failed"]},"agent":{"id":"000","name":"cyberleil2.kennesaw.edu"},"manager":{"name":"cyberleil2.kennesaw.edu"},"i
d":"1732289364.108169","full_log":"Nov 22 15:29:22 cyberleil2.kennesaw.edu dbus-daemon[969]: [system] Successfully activated service 'org.freedesktop.timeda
tel'","predecoder":{"program_name":"dbus-daemon","timestamp":"Nov 22 15:29:22","hostname":"cyberleil2.kennesaw.edu"},"decoder":{},"location":"journald"}
```

Figure B4

Bad ssh attempts showing in /var/log/secure

```
[root@cyberleil2 ~]# grep sshd /var/log/secure | tail -n 15
Nov 21 17:11:18 cyberleil2 sshd[416921]: Failed keyboard-interactive/pam for invalid user root from 172.27.14.115 port 59636 ssh2
Nov 21 17:11:18 cyberleil2 sshd[416921]: Postponed keyboard-interactive for invalid user root from 172.27.14.115 port 59636 ssh2 [preauth]
Nov 21 17:11:21 cyberleil2 sshd[416936]: pam_unix(sshd:auth): authentication failure; logname= uid=0 euid=0 tty=ssh ruser= rhost=172.27.14
Nov 21 17:11:21 cyberleil2 sshd[416921]: Postponed keyboard-interactive/pam for invalid user root from 172.27.14.115 port 59636 ssh2 [preau
Nov 21 17:11:23 cyberleil2 sshd(pam_google_authenticator)[416936]: Dummy password supplied by PAM. Did OpenSSH 'PermitRootLogin <anything 
other config block this login?
Nov 21 17:11:23 cyberleil2 sshd(pam_google_authenticator)[416936]: Dummy password supplied by PAM. Did OpenSSH 'PermitRootLogin <anything 
other config block this login?
Nov 21 17:11:23 cyberleil2 sshd(pam_google_authenticator)[416936]: Invalid verification code for root
Nov 21 17:11:25 cyberleil2 sshd[416921]: error: PAM: Authentication failure for illegal user root from 172.27.14.115
Nov 21 17:11:25 cyberleil2 sshd[416921]: Failed keyboard-interactive/pam for invalid user root from 172.27.14.115 port 59636 ssh2
Nov 21 17:11:25 cyberleil2 sshd[416921]: Postponed keyboard-interactive for invalid user root from 172.27.14.115 port 59636 ssh2 [preauth]
```

Figure B5

Curl requests and bad page requests showing in /var/log/httpd/ logs

```
[root@cyberleil2 ~]# grep 10.96.33.31 /var/log/httpd/access_log | tail
10.96.33.31 - - [21/Nov/2024:23:50:52 -0500] "GET / HTTP/1.1" 301 229 "-" "curl/7.61.1"
10.96.33.31 - - [21/Nov/2024:23:50:52 -0500] "GET / HTTP/1.1" 301 229 "-" "curl/7.61.1"
10.96.33.31 - - [21/Nov/2024:23:50:52 -0500] "GET / HTTP/1.1" 301 229 "-" "curl/7.61.1"
10.96.33.31 - - [21/Nov/2024:23:50:52 -0500] "GET / HTTP/1.1" 301 229 "-" "curl/7.61.1"
10.96.33.31 - - [21/Nov/2024:23:50:52 -0500] "GET / HTTP/1.1" 301 229 "-" "curl/7.61.1"
10.96.33.31 - - [21/Nov/2024:23:50:52 -0500] "GET / HTTP/1.1" 301 229 "-" "curl/7.61.1"
10.96.33.31 - - [21/Nov/2024:23:50:52 -0500] "GET / HTTP/1.1" 301 229 "-" "curl/7.61.1"
10.96.33.31 - - [21/Nov/2024:23:50:52 -0500] "GET / HTTP/1.1" 301 229 "-" "curl/7.61.1"
10.96.33.31 - - [21/Nov/2024:23:50:52 -0500] "GET / HTTP/1.1" 301 229 "-" "curl/7.61.1"
10.96.33.31 - - [21/Nov/2024:23:51:43 -0500] "\n" 400 226 "-" "-"
[root@cyberleil2 ~]# grep 10.96.33.31 /var/log/httpd/error_log | tail
[Thu Nov 21 17:44:09.390809 2024] [ssl:info] [pid 205137:tid 139863722551040] [client 10.96.33.31:51026] AH01964: Connection to child 19 established (serve
[Thu Nov 21 17:44:09.392517 2024] [ssl:info] [pid 205137:tid 139863722551040] [client 10.96.33.31:51026] AH02008: SSL library error 1 in handshake (server 
[Thu Nov 21 17:44:09.392545 2024] [ssl:info] [pid 205137:tid 139863722551040] [client 10.96.33.31:51026] AH01998: Connection closed to child 19 with aborti
[Thu Nov 21 17:45:44.142166 2024] [ssl:info] [pid 216672:tid 139864267814656] [client 10.96.33.31:51324] AH01964: Connection to child 196 established (serv
[Thu Nov 21 17:45:44.144480 2024] [core:info] [pid 216672:tid 139864267814656] [client 10.96.33.31:51324] AH00128: File does not exist: /var/www/html/page
[root@cyberleil2 ~]#
```

Figure B6

Snort IDS alerts showing in syslog

```
[root@cyberleil2 ~]# grep snort /var/log/messages | tail
Nov 21 18:02:14 cyberleil2 snort[1157]: [1:30790:8] "SERVER-WEBAPP Java ClassLoader access attempt" [Classification: Attempted Administrator Privilege Gain] [Priority: 1] {TCP} 172.27.14.115:62095 -> 10.96.33.229:80
Nov 21 18:02:14 cyberleil2 snort[1141]: [1:30790:8] "SERVER-WEBAPP Java ClassLoader access attempt" [Classification: Attempted Administrator Privilege Gain] [Priority: 1] {TCP} 172.27.14.115:62095 -> 10.96.33.229:80
Nov 21 18:02:14 cyberleil2 snort[1157]: [1:30790:8] "SERVER-WEBAPP Java ClassLoader access attempt" [Classification: Attempted Administrator Privilege Gain] [Priority: 1] {TCP} 172.27.14.115:62095 -> 10.96.33.229:80
Nov 21 18:02:14 cyberleil2 snort[1157]: [1:30790:8] "SERVER-WEBAPP Java ClassLoader access attempt" [Classification: Attempted Administrator Privilege Gain] [Priority: 1] {TCP} 172.27.14.115:62094 -> 10.96.33.229:80
Nov 21 18:02:14 cyberleil2 snort[1141]: [1:30790:8] "SERVER-WEBAPP Java ClassLoader access attempt" [Classification: Attempted Administrator Privilege Gain] [Priority: 1] {TCP} 172.27.14.115:62095 -> 10.96.33.229:80
Nov 21 18:02:14 cyberleil2 snort[1141]: [1:30790:8] "SERVER-WEBAPP Java ClassLoader access attempt" [Classification: Attempted Administrator Privilege Gain] [Priority: 1] {TCP} 172.27.14.115:62094 -> 10.96.33.229:80
Nov 21 18:02:14 cyberleil2 snort[1157]: [1:30790:8] "SERVER-WEBAPP Java ClassLoader access attempt" [Classification: Attempted Administrator Privilege Gain] [Priority: 1] {TCP} 172.27.14.115:62093 -> 10.96.33.229:80
Nov 21 18:02:14 cyberleil2 snort[1141]: [1:30790:8] "SERVER-WEBAPP Java ClassLoader access attempt" [Classification: Attempted Administrator Privilege Gain] [Priority: 1] {TCP} 172.27.14.115:62093 -> 10.96.33.229:80
Nov 21 18:02:14 cyberleil2 snort[1157]: [1:30790:8] "SERVER-WEBAPP Java ClassLoader access attempt" [Classification: Attempted Administrator Privilege Gain] [Priority: 1] {TCP} 172.27.14.115:62096 -> 10.96.33.229:80
Nov 21 18:02:14 cyberleil2 snort[1141]: [1:30790:8] "SERVER-WEBAPP Java ClassLoader access attempt" [Classification: Attempted Administrator Privilege Gain] [Priority: 1] {TCP} 172.27.14.115:62096 -> 10.96.33.229:80
```

Appendix C: Vulnerability Report from Other Team

Vulnerability analysis/penetration testing report

During the vulnerability analysis and penetration testing of the other groups VM (10.96.33.229), we used multiple tools in order to find what vulnerabilities were present. Examples of scanning tools that were used in this analysis report are ZAP, Metasploit, NMAP, Nikito, and TRACE. Throughout running the scans, it was shown that the team has done an outstanding job securing their server.

Whether it was a simple login attack that showed the team configured multi-factor authentication to blocking SQL injection attacks. We also attempted to send out a DOS attack by flooding the server with multiple requests, but not too many because we do not want to affect the server, only wanting to show that this can be an issue. The group implemented a block on the ECHO so that we were not able to tell if the server was up and running which was a very smart move by them. One thing I did notice however was that they do have a firewall set up, but it is not filtering anything in the ports that are being used by the group, so that could potentially be a vulnerability where it can be filtered for precautionary reasons. Configuration wise, we were able to find that the apache server was not updated to its latest version. Not having the latest version of services running could potentially lead to attacks by not having the latest versions of software.

On port 22 which is used by SSH, our scanning tools showed multiple items that could be listed as vulnerabilities. Weak or default passwords: If the SSH service is configured with weak or default passwords, it can be easily compromised. Outdated SSH versions: Older versions of SSH may contain known vulnerabilities that can be exploited by attackers. Insecure SSH configurations: Improper SSH configurations, such as allowing root login or using weak ciphers, can increase the risk of attacks. Information leaks seemed to be available as well. Running through the scans, the scans were able to show private IP addresses, software versions, and directories with file names as well.

With the listed vulnerabilities above, it still comes to show how secure the other team's server was. The group showed great technical skills from implementing a MFA, blocking ECHO pings, securing the database so SQL injection attacks are not possible, and many more solutions. As always, there are still some small; things to consider securing so that the server can be as secure as possible, but with all of what the group has done, this can still be classified as a secure server! Below this will show the screenshots captured to show what has been done and found during our analysis for further proof as well.