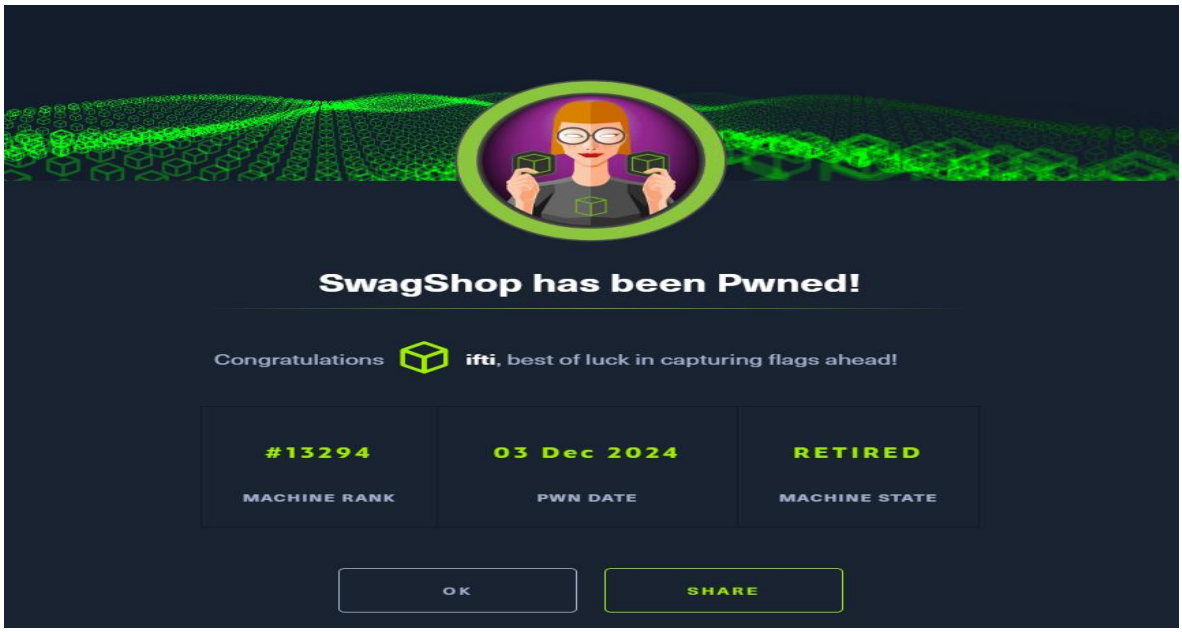
Executive Summary

The **SwagShop** machine exploits vulnerabilities in a Magento eCommerce server. Initial access was gained through an exploit using **SQL injection**, which allowed the creation of a user account in the admin panel. Further exploitation involved uploading a PHP reverse shell through product customization features. Privilege escalation was achieved by abusing sudo permissions to execute commands as **root** via the vi editor. This highlights critical issues, including insecure web applications, poor input validation, and misconfigured sudo privileges.



Contents

Reconnaissance

Nmap Scan

The enumeration began with **Nmap** to identify open ports, services, and operating system details.

Command:

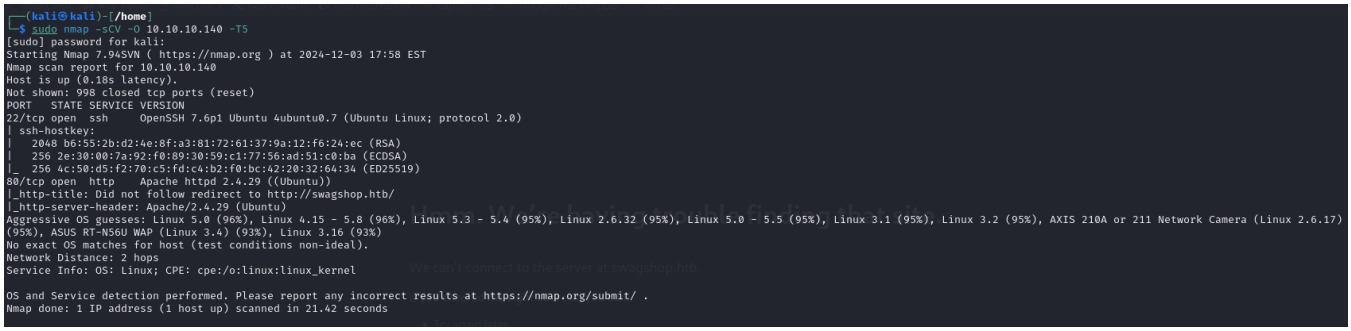
sudo nmap -sCV -O 10.10.10.140 -T5

Options Explained:

- -sC: Default script scan.
- -sV: Version detection.
- -O: OS detection.
- -T5: Aggressive and fast scanning.

Results:

- **Port 80 (HTTP)**: Apache 2.4.18 running a Magento eCommerce server.
- Other open ports were also discovered but were not explored further.



Enumeration

Web Application Inspection

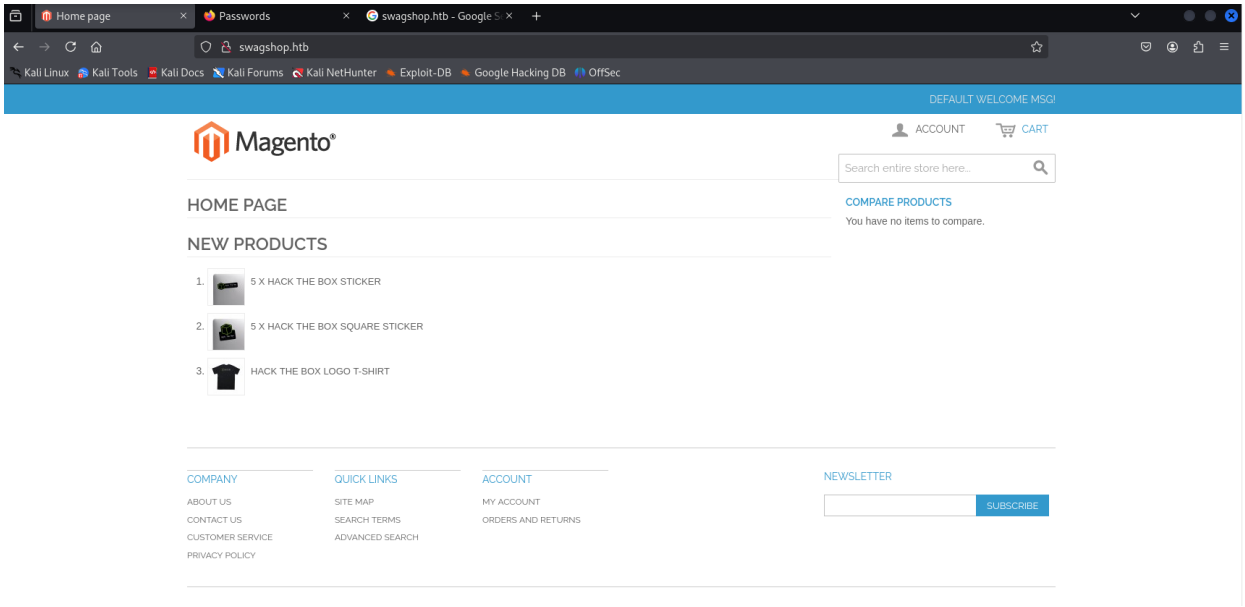
The web server was accessible via port 80. After adding the domain to /etc/hosts as swagshop.htb, visiting the site revealed a **Magento eCommerce application**.

Steps:

1. Add domain mapping:

echo "10.10.10.140 swagshop.htb" | sudo tee -a /etc/hosts

2. Visit http://swagshop.htb/ in the browser.



Magento was identified as the application running on the server. A Google search revealed multiple known exploits for this version.

Exploitation

SQL Injection to Add Admin User

An exploit for Magento allowed the injection of a user (faceless) into the database via SQL injection. This exploit was executed, and the user credentials were set as follows:

- **Username**: faceless
- **Password**: faceless

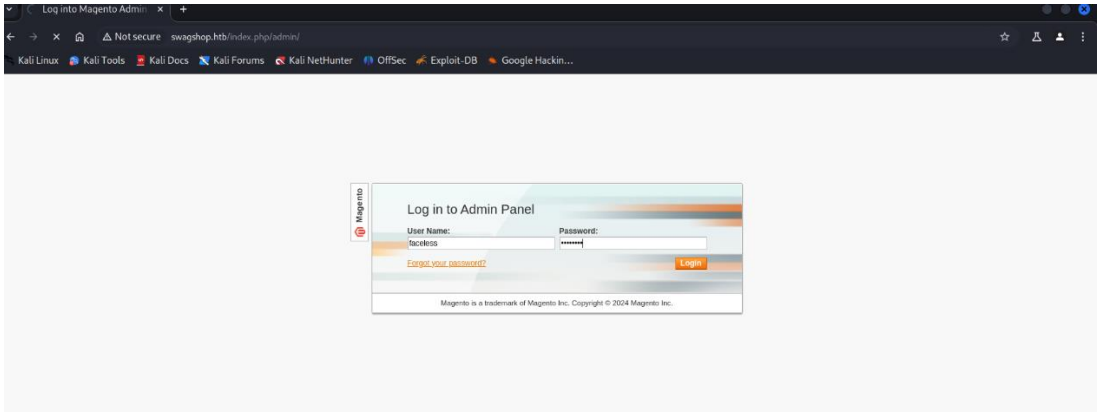
```
┌──(kali㉿kali)-[/home]
└─$ sudo nano exploitSwag.py

┌──(kali㉿kali)-[/home]
└─$ python3 exploitSwag.py
  File "/home/exploitSwag.py", line 27
    print "WORKED"
    ^^^^^^^^^^^^^^
SyntaxError: Missing parentheses in call to 'print'. Did you mean print( ... )?

┌──(kali㉿kali)-[/home]
└─$ python2 exploitSwag.py
WORKED
Check http://swagshop.htb/index.php/admin with creds faceless:faceless

┌──(kali㉿kali)-[/home]
└─$
```

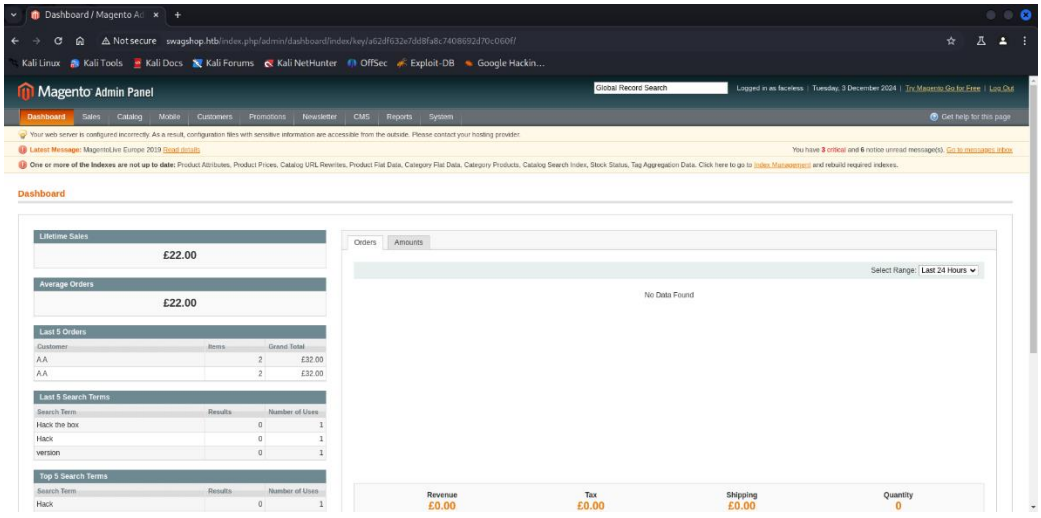
The exploit was used to access the admin panel at:

http://swagshop.htb/index.php/admin/

Outcome:

- Successful login to the admin panel.



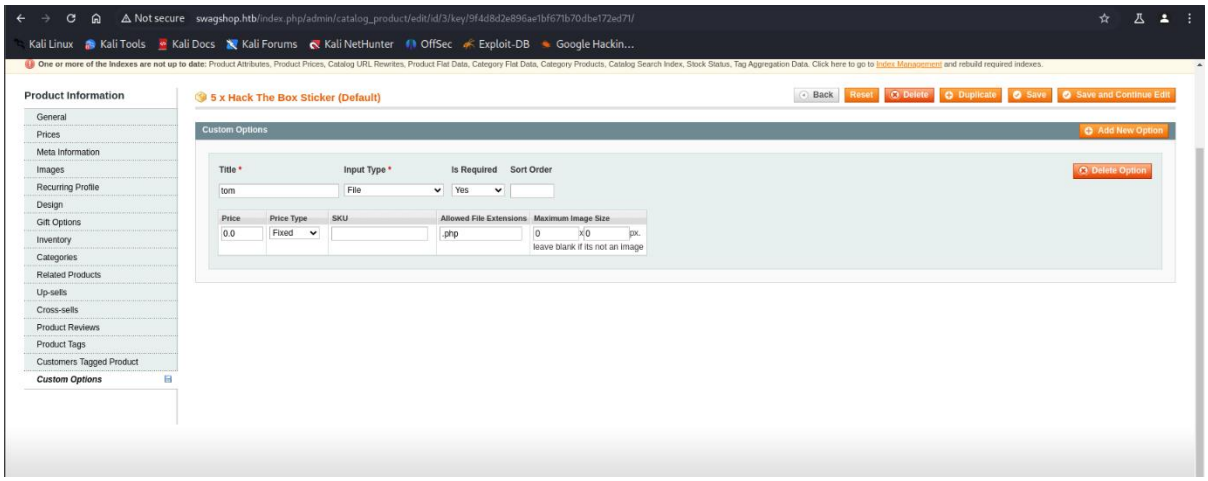


Uploading a PHP Reverse Shell

With access to the admin panel, a reverse shell was uploaded using the product customization feature.

Steps:

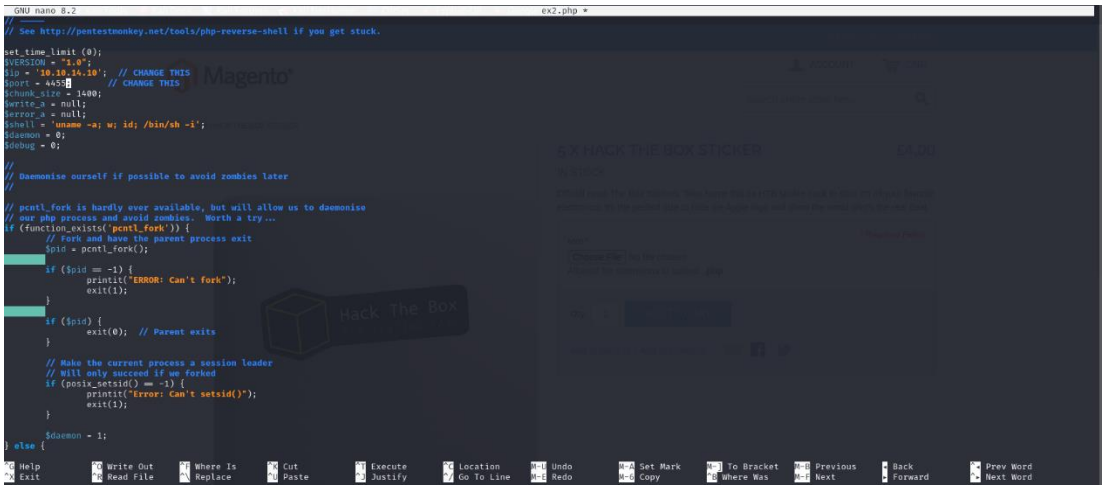
1. Navigate to **Catalog → Manage Products**.
2. Select the **Hack The Box Sticker** product.
3. Go to the **Custom Options** tab and add a new option:
    - **Title**: Custom Upload
    - **Input Type**: File
    - **File Extensions**: php
4. Save the changes.



Uploading the Reverse Shell:

1. Prepare a PHP reverse shell from this link.

Following is the commonly used php script for reverse shell. It was downloaded and locally stored as the ex2.php.



2. Modify the script to include the attacker's IP address and port.
3. Set up a **Netcat listener**:

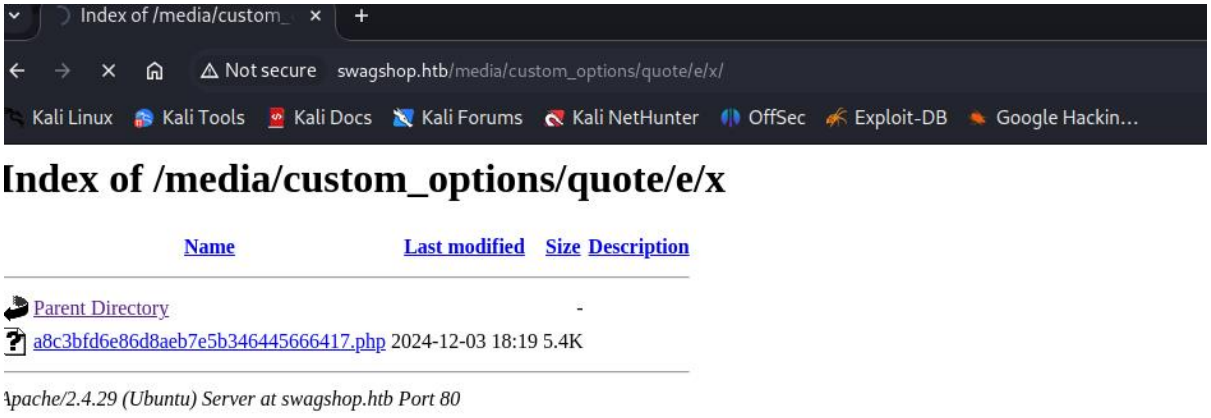
nc -nvlp 4455

4. Upload the shell via the product's custom upload field.

Access the Shell:

The shell was uploaded to the following location:

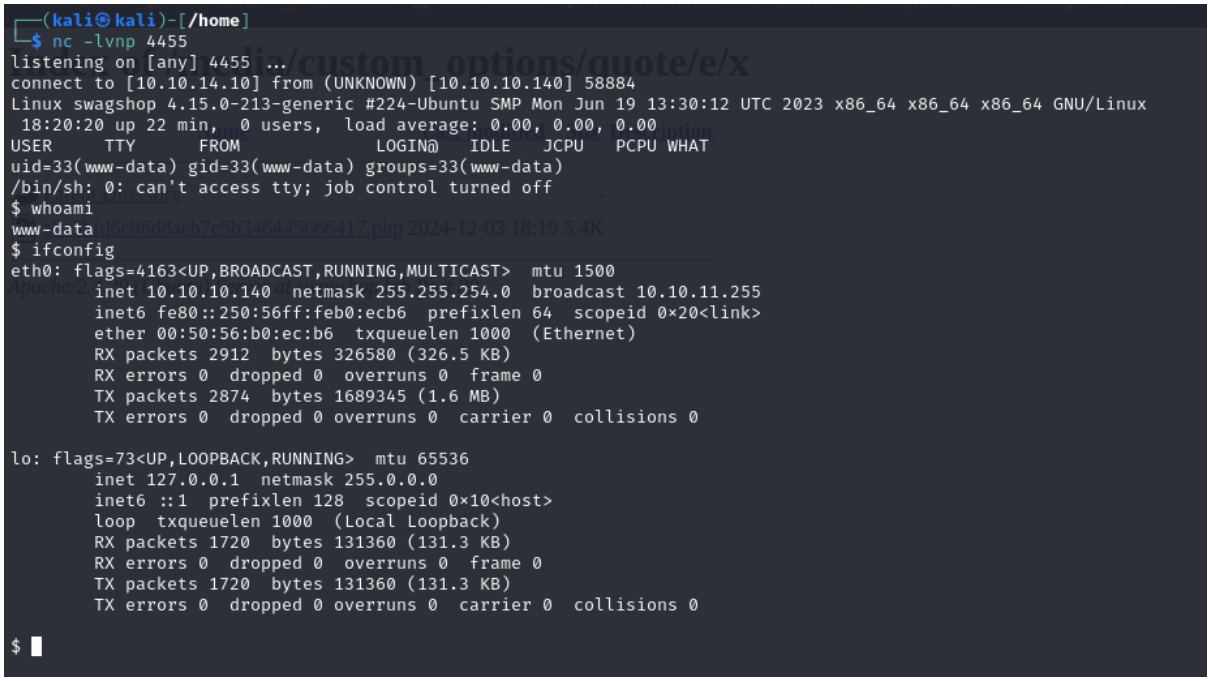
http://swagshop.htb/media/custom_options/quote/e/x/



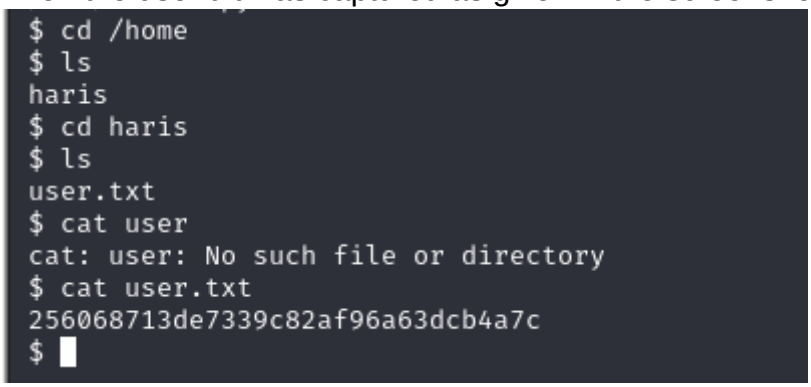
Triggering the reverse shell gave initial access to the system.

Outcome:

- Gained a shell with user privileges.



Then the user.txt was captured as given in the screenshot:



Privilege Escalation

Abusing Sudo Permissions

Running sudo -l revealed that the vi text editor could be executed with root privileges for files in /var/www/html/.

Command:

sudo -l

Result:

- Permission to execute:

(ALL) NOPASSWD: /usr/bin/vi /var/www/html/*

Exploitation:

1. Open a file in the /var/www/html/ directory with vi:

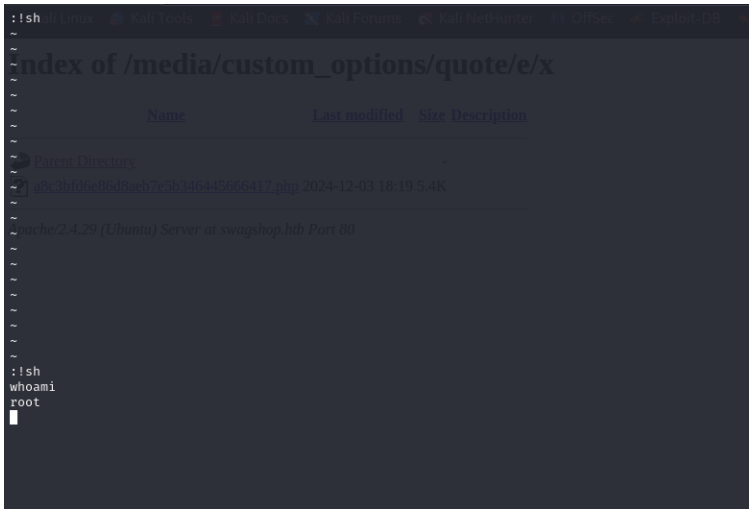
sudo vi /var/www/html/index.php

2. Escalate to a root shell:

:!bash

Outcome:

- Gained a root shell.



Post-Exploitation

- **Root Flag**:

Cleaning Up Evidence

1. Clear shell history:

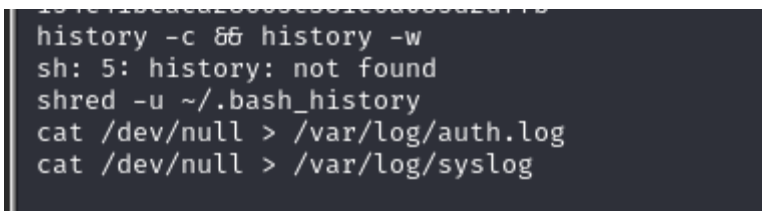
history -c && history -w

2. Remove logs:

shred -u ~/.bash_history

cat /dev/null > /var/log/auth.log

cat /dev/null > /var/log/syslog



Considerations/Mitigations

1. **Sanitize Input**:
    - Prevent SQL injection by using prepared statements and proper input validation.
2. **Restrict Admin Access**:
    - Limit access to the admin panel to trusted IP addresses.
3. **Disable File Uploads**:
    - Restrict or validate file uploads to prevent malicious content.
4. **Review Sudo Configurations**:
    - Remove unnecessary or overly permissive sudo rules.
5. **Update Software**:
    - Upgrade Magento and the underlying server software to the latest, patched versions.
6. **Enable Logging and Monitoring**:
    - Monitor web and server logs for suspicious activity.

Conclusion

The **SwagShop** machine demonstrated several security misconfigurations, including SQL injection vulnerabilities, unrestricted file uploads, and overly permissive sudo rules. Addressing these issues would significantly enhance security and reduce the risk of compromise.