Penetration Test Report

Target: Startup (THM)

CONFIDENTIAL DOCUMENT

Contains security vulnerabilities and proofs of concept.

Prepared by:
Pedro Antunes
Junior Penetration Tester Student

Date:
December 13, 2025

Contents

1 Executive Summary

1.1 Overview

The objective of this assessment was to identify potential security weaknesses in the target infrastructure. The assessment was conducted using a black-box approach, simulating an external attacker with no prior knowledge of the system.

1.2 Key Findings

During the engagement, a critical misconfiguration was identified within the network's file sharing service (FTP).

This vulnerability allows unauthenticated attackers to log in anonymously and upload malicious files to the server. By exploiting this flaw, it was possible to execute arbitrary code and gain complete administrative control (Root Access) over the system.

1.3 Risk Summary

The following table summarizes the identified vulnerabilities by severity:

primaryColor	
Critical	2
High	1
Medium	0
Low	0

2 Scope and Methodology

2.1 Scope

- **Target Name:** Startup
- **IP Address:** 10.81.179.141
- **Authorized by:** TryHackMe Education

2.2 Methodology

The assessment followed standard penetration testing phases:

1. **Reconnaissance:** Information gathering and service enumeration.
2. **Vulnerability Analysis:** Identification of potential weaknesses.
3. **Exploitation:** Controlled execution of attacks to verify vulnerabilities.
4. **Post-Exploitation:** Assessment of the impact and privilege escalation.

3 Technical Findings

3.1 1. Anonymous FTP Upload Leading to RCE

CRITICAL

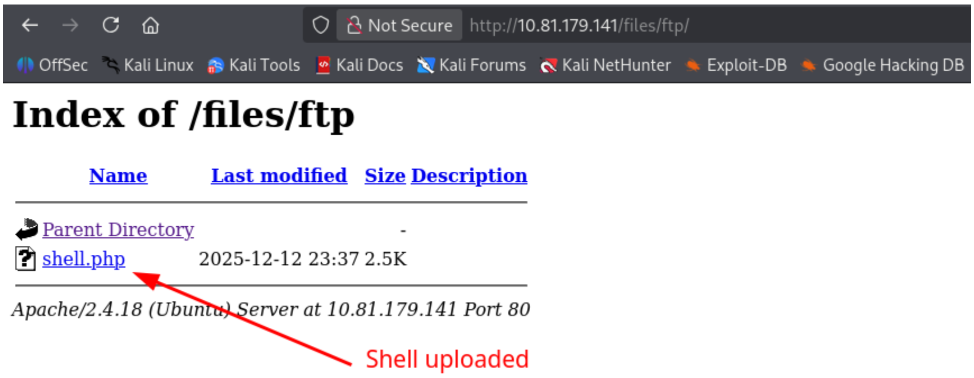
CVSS Score (Estimated): 9.8
Component: FTP (Port 21) / HTTP (Port 80)

Description

The FTP server is configured to allow "Anonymous" login with write permissions. Additionally, the directory used by the FTP server is also accessible via the Web Server. This allows an attacker to upload a malicious PHP script via FTP and execute it via the browser to gain Remote Code Execution (RCE).

Proof of Concept (PoC)

1. Logged in to FTP (Port 21) using the username `anonymous` and any password.
2. Uploaded a PHP Reverse Shell file named `shell.php`.
3. Navigated to `http://10.81.179.141/files/shell.php` in the browser.
4. A reverse shell connection was established, granting access as the `www-data` user.



Index of /files/ftp

Name	Last modified	Size	Description
Parent Directory		-	
shell.php	2025-12-12 23:37	2.5K	

Apache/2.4.18 (Ubuntu) Server at 10.81.179.141 Port 80

Shell uploaded

Figure 1: Uploading the shell via FTP

Remediation

Disable anonymous write access on the FTP server. Ensure that files uploaded via FTP are not stored in a directory executable by the web server.

3.2 2. Sensitive Data Exposure (Cleartext Credentials)

HIGH

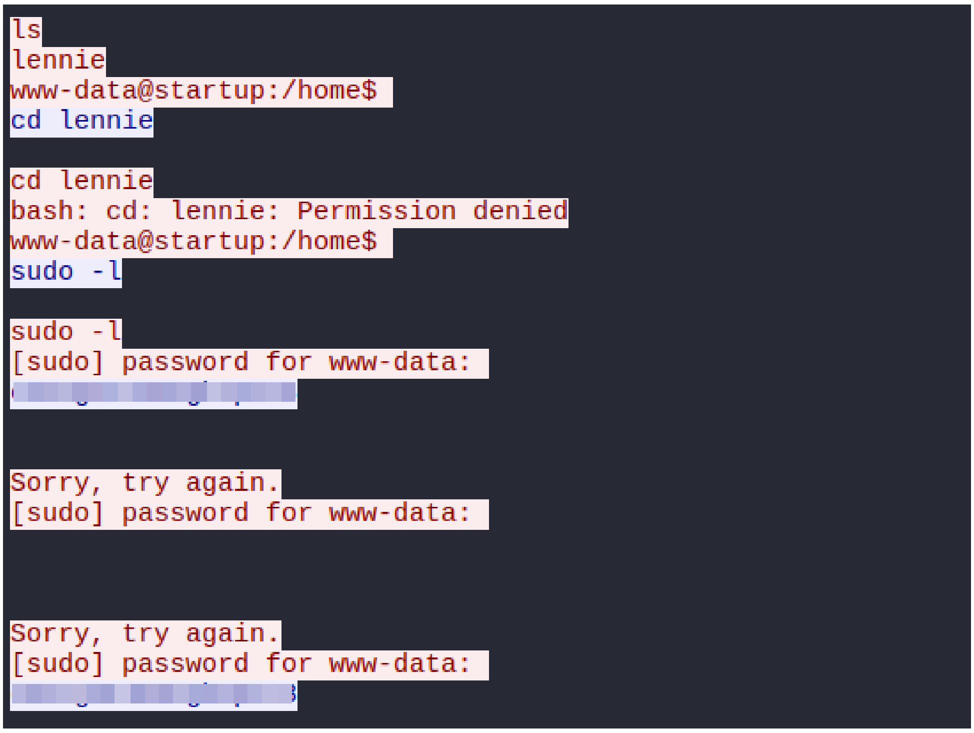
CVSS Score (Estimated): 7.5
Component: File System / Information Disclosure

Description

A network capture file (`suspicious.pcapng`) was found in a directory accessible to the web user. Analysis of this file revealed unencrypted network traffic containing the plaintext password for the user `lennie`.

Proof of Concept (PoC)

1. Located the file `suspicious.pcapng` in the incidents directory.
2. Transferred the file to the attacker machine and analyzed it using Wireshark.
3. Followed a TCP stream which revealed a login attempt containing the password `c****3` (Redacted).
4. Used these credentials to log in via SSH as user `lennie`.



```
ls
lennie
www-data@startup:/home$
cd lennie

cd lennie
bash: cd: lennie: Permission denied
www-data@startup:/home$
sudo -l

sudo -l
[sudo] password for www-data:
[redacted]


Sorry, try again.
[sudo] password for www-data:



Sorry, try again.
[sudo] password for www-data:
[redacted]
```

Figure 2: Wireshark analysis revealing the password

Remediation

Remove sensitive log or capture files from the system immediately after use. Ensure all internal authentication traffic occurs over encrypted channels (SSH/HTTPS) to prevent credential sniffing.

3.3 3. Privilege Escalation via Insecure Script Permissions

CRITICAL

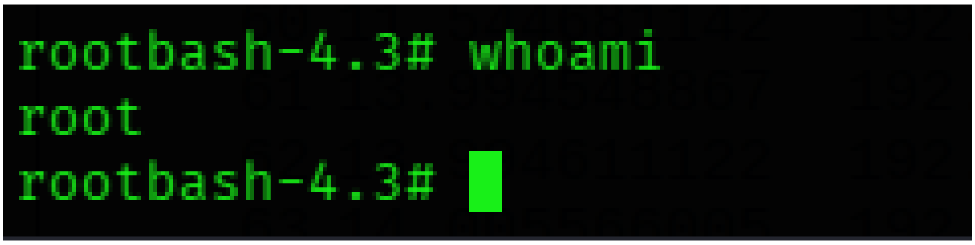
CVSS Score (Estimated): 8.8
Component: Linux Permissions / Cron Job

Description

A script named `planner.sh` is executed periodically the root user. This script calls another script, `/etc/print.sh`, which is writable by the user `lennie`. This allows a low-privileged user to inject malicious code into `print.sh`, which is then executed with Root privileges.

Proof of Concept (PoC)

1. Identified that `planner.sh` runs as root.
2. Verified that `/etc/print.sh` was writable by the current user.
3. Modified `/etc/print.sh` to execute a reverse shell command.
4. Execute the script `planner.sh`.
5. Received a connection back providing full `root` access.



```
rootbash-4.3# whoami
root
rootbash-4.3# 
```

Figure 3: Root access obtained

Remediation

Change the ownership of `/etc/print.sh` to root and remove write permissions for other users (`chmod 700`).

A tua conclusão está aceitável, mas foca-se demasiado apenas no FTP.

Lembra-te: Tu conseguiste Root (Administrador Máximo). Se o relatório disser apenas "Corrijam o FTP", o cliente pode pensar que o resto do sistema (a parte do utilizador Lennie e os scripts) está segura, o que é mentira.

Uma conclusão profissional deve resumir a Cadeia de Ataque completa.

Aqui tens uma versão melhorada, mais completa e com um inglês mais polido para o teu relatório final: Código Melhorado para a Conclusão

Substitui a tua secção de conclusão por esta: Snippet de código

4 Conclusion

The penetration test against the **Startup** system revealed critical security deficiencies spanning multiple layers of the infrastructure.

While the initial compromise was caused by a misconfigured **FTP server** allowing anonymous uploads, the complete takeover of the system (Root Access) was only possible due to a chain of additional failures:

- **Data Leakage:** Sensitive network capture files containing passwords were left readable by low-privileged users.
- **Weak Permissions:** Critical system scripts were writable by non-root users, allowing for Privilege Escalation.

Immediate attention is required to disable anonymous FTP access, remove sensitive files from public directories, and enforce strict file permissions on administrative scripts.

It is highly recommended to perform a re-test after the remediation process to verify that all fixes have been effectively implemented.