HACK THE BOX

Penetration Test Report

Machine:	CodePartTwo
Candidate Name:	Senan Abdullayev
Customer:	Hack The Box
Date:	January 31, 2026
Version:	1.0

CONFIDENTIAL

Contents

1 Statement of Confidentiality

This report contains confidential information regarding the security posture of CodePartTwo. The information contained herein is intended for the internal use of CodePartTwo and should not be disclosed to unauthorized third parties.

2 Engagement Contacts

Name	Role	Email
Senan Abdullayev	Penetration Tester	tester@htb.com
CodePartTwo Admin	System Owner	admin@codeparttwo.htb

3 Executive Summary

3.1 Approach

A grey-box penetration test was conducted against the CodePartTwo infrastructure. The assessment focused on identifying vulnerabilities in the web application and underlying operating system that could lead to unauthorized access, data leakage, or privilege escalation. The testing methodology followed industry standards, including the OWASP Top 10 and PTES (Penetration Testing Execution Standard).

3.2 Scope

The scope of this assessment was limited to the following host:

- **Target IP:** 10.129.14.149
- **Hostname:** CodePartTwo

3.3 Assessment Overview and Recommendations

The assessment revealed critical security flaws that resulted in full system compromise (Root Access).

The primary entry point was a **Remote Code Execution (RCE)** vulnerability in the web application's code execution feature. This was caused by a known vulnerability (CVE-2024-28397) in the `js2py` library (version 0.74), which allowed an unauthenticated attacker to escape the Python sandbox and execute arbitrary system commands.

Following initial access, lateral movement was achieved by recovering credentials from a local SQLite database (`users.db`). The passwords were stored using weak MD5 hashing, allowing them to be cracked easily. Finally, **Privilege Escalation** to Root was achieved by exploiting a logic flaw in a custom backup utility (`npbackup-cli`) running with sudo privileges.

Key Recommendations:

1. **Patch the Web Application:** Remove the `js2py` implementation or strictly sandbox the environment. Ensure user input is never passed to `eval()` or similar functions.
2. **Strengthen Password Policies:** Migrate from MD5 hashing to a secure algorithm like Argon2 or bcrypt for password storage.
3. **Fix Sudo Configuration:** Remediate the argument parsing logic in `npbackup-cli` to prevent flag bypasses and restrict the binary from loading external backends.

4 Network Penetration Test Assessment Summary

4.1 Summary of Findings

ID	Title	Severity	Status
FIND-01	Remote Code Execution via Sandbox Escape (js2py)	**Critical**	Open
FIND-02	Privilege Escalation via Argument Injection	**Critical**	Open
FIND-03	Weak Password Hashing (MD5) & Reuse	**High**	Open
FIND-04	Source Code Disclosure	**Medium**	Open

5 Internal Network Compromise Walkthrough

5.1 Detailed Walkthrough

5.1.1 1. Reconnaissance & Enumeration

The assessment began with an Nmap scan, which identified OpenSSH on port 22 and a Gunicorn web server on port 8000. Directory fuzzing using `ffuf` revealed a `/download` endpoint. Accessing this endpoint provided the full source code of the application (`app.zip`). This allowed for a white-box analysis of the underlying Python code (app.py), revealing the use of the `js2py` library and a hardcoded secret key.

5.1.2 2. Initial Access (RCE)

Analysis of app.py revealed a /run_code endpoint that accepted JavaScript code and executed it using `js2py`. Although `js2py.disable_pyimport()` was active, the library version (0.74) was vulnerable to a sandbox escape (CVE-2024-28397).

By traversing the Python object hierarchy using `Object.getOwnPropertyNames({})`, access was gained to the subprocess.Popenclass. This allowed the execution of arbitrary system commands. A reverse shell payload was crafted and executed, granting access as the appuser.

Payload Used:

```
var obj = Object.getOwnPropertyNames({}).__class__.__base__;
var subs = obj.__subclasses__();
// Loop to find Popen class and execute bash
cls(['bash', '-c', 'echo [BASE64_SHELL] | base64 -d | bash'], ...);
```

5.1.3 3. Lateral Movement

Post-exploitation enumeration revealed a SQLite database at `instance/users.db`. Querying this database exposed user credentials stored as unsalted MD5 hashes.

- **User:** marco
- **Hash:** 5f4dcc3b5aa765d61d8327deb882cf99 (MD5)
- **Cracked Password:** sweetangelbabylove

These credentials allowed for SSH access as the user marco

5.1.4 4. Privilege Escalation to Root

The user `marco`had sudo privileges to run `/usr/local/bin/npbackup-cli` without a password. Analysis of the script revealed a flawed blocklist implementation for the `--external-backend-binary` flag.

```
if '--external-backend-binary' in sys.argv:
    print("Error...")
```

This check was bypassed by using the `=` syntax (`--external-backend-binary=/tmp/script.sh`), preventing the flag from appearing as a standalone element in `sys.argv`. A compliant configuration file (`np.conf`) was created to satisfy the tool's requirements (Version 3.0.1, valid repo structure). A malicious backend script (`rootme.sh`) was created to set the SUID bit on `/bin/bash`. Executing the backup tool with these parameters triggered the backend script, resulting in Root access.

6 Remediation Summary

6.1 Short Term

- **FIND-01:** Disable the /run_code endpoint immediately.
- **FIND-02:** Remove the npbackup-cli entry from the sudoers file for user marco
- **FIND-03:** Force a password reset for all users.

6.2 Medium Term

- **FIND-01:** If the code execution feature is business-critical, implement it using isolated containers (e.g., Docker) rather than relying on library-level sandboxes like js2py .
- **FIND-03:** Upgrade the database schema to use a secure password hashing algorithm (e.g., bcrypt with a work factor of 12 or higher).

6.3 Long Term

- Implement a rigorous code review process to catch logic flaws such as the Python argument parsing bypass.
- Conduct regular penetration tests.

7 Technical Findings Details

7.1 Remote Code Execution via Sandbox Escape (js2py)

Severity	**Critical**
CVSS	**9.8 (Critical)**
Impact	**System Compromise, Data Exfiltration**

Description:

The application uses `js2py` (v0.74) to execute user-supplied JavaScript. This version contains a known vulnerability (CVE-2024-28397) that allows attackers to bypass the `disable_pyimport` restriction by accessing the Python `object` base class via JavaScript reflection (`Object.getOwnPropertyNames`). This provides access to `subprocess.Popen`, enabling the execution of arbitrary OS commands.

Remediation:

Do not use `js2py` for untrusted code execution. Use a proper isolation mechanism such as Docker or gVisor.

7.2 Cleartext Credentials & Weak Hashing (MD5)

Severity	**High**
CVSS	**7.5 (High)**
Impact	**Credential Theft, Lateral Movement**

Description:

User passwords are stored in the `users.db` SQLite database using the MD5 hashing algorithm without a salt. MD5 is cryptographically broken and allows for rapid cracking of passwords using rainbow tables or brute-force attacks.

Remediation:

Implement a strong hashing algorithm such as Argon2id or bcrypt. Ensure all existing passwords are reset and re-hashed.

7.3 Privilege Escalation via Argument Injection (npbackup-cli)

Severity	**Critical**
CVSS	**8.8 (High)**
Impact	**Privilege Escalation to Root**

Description:

The custom backup script (`npbackup-cli`) attempts to block the dangerous `--external-backend-binary` flag by checking for its presence in `sys.argv` using a list membership check. This logic is flawed and can be bypassed by appending the value with an equals sign (e.g., `--flag=value`). Since the tool runs as root via sudo, this allows unprivileged users to execute arbitrary binaries with root privileges.

Remediation:

Use Python's `argparse` library to handle arguments securely or remove the capability to load external binaries entirely.

8 Appendix A: Host & Service Discovery

8.1 A.1 Host Information

- **IP Address:** 10.129.14.149
- **OS:** Linux (Ubuntu)

8.2 A.2 Service Scan Results

Port	Protocol	Service	Version
22	TCP	SSH	OpenSSH 8.2p1
8000	TCP	HTTP	Gunicorn 20.0.4

8.3 A.3 Compromised Users

Username	Type	Method	Notes
app	Service Account	RCE	Accessed via web exploit.
marco	User	Credential Stuffing	Password cracked from database.
root	Administrator	Sudo Exploit	Exploited npbackup-cli .