Flag: flag{de3b1c3945c08107465c733353d3f1123de4aa92720a9e}

Used Tools:
➔ ping
➔ nmap -p- 10.20.73.1-255
➔ devtools
➔ hydra -L users.txt -P /usr/share/wordlists/rockyou.txt \ mysql://10.20.73.3
➔ sqlmap -u "http://10.20.73.3/php/recherche_old.php" \ --data="recherche=test" \ -p recherche \ --cookie="PHPSESSID=di3htsr0j40361hnd70l0cktf7" \ --dbs
➔ –11– \ --tables
➔ –11– \ --dump

Since flag3 is about database-hacking I did another ip/port scan and found that under one of the ips, the same one as the last flag, there was a port open for mysql. This usually means there's something database related there. I ran:

```
mysql -h 10.20.73.3 -u root -p
```

got prompted with a password, tried some standards but nothing worked. I found out about hydra that allows me to bruteforce quite a hefty amount of passwords with the rockyou.txt password list. Tried that for some time without any success. That meant I had to try something else, so I tried with the SQLI approach. I used Sqlmap to perform sql-injections without any results, and tried some more advanced checks without any result.

Another way of doing SQLI is by prompting the input fields. I used different parameters to alter the database call, like for instance the very simple and straightforward:

```
' OR '1'='1
```

But whatever I tried with I just could see a simple error message with wrong pass/user. However I found out in the post request that it does actually include the SQLI code in the response which was interesting. I then tried with different forms and finally under the old_resercher.php endpoint, I managed to get the previous command to list all users for me. This mean that the database for the user-info was vulnerable to SQLI.

Now I also realized that I probably used sqlmap not as it was supposed to. As after doing some research, I found out that if knowing that the input field for searching users was vulnerable to SQLI, I could simply run sqlmap and make it do the work for me by gathering the db. By taking the session-cookie from devtools under the post request `PHPSESSID=di3htsr0j40361hnd70l0cktf7` I could prompt the input file in the

`reserche_old.php` that requires a logged in user, and I also found the input field name in inspect element (`reserche`).

When running the command:

```
sqlmap -u "http://10.20.73.3/php/recherche_old.php" \
--data="recherche=test" \ -p recherche \
--cookie="PHPSESSID=di3htsr0j40361hnd70l0cktf7" \ --dbs
```

It showed me that the database information and its tables could be retrieved, I then added the command `--tables` to the previous command to check what was there. I found a table named flags which made it obvious to dump that one. So I ran the command again with `-T flags \ --dump` instead and received the response:

```
Flag=

flag_id | flag                                                    |
+---------+---------------------------------------------------------+
| 32f28e  | flag{de3b1c3945c08107465c733353d3f1123de4aa92720a9e}
```