**《Web渗透测试》实验报告**

|  |  |  |
| --- | --- | --- |
| **组号 8** | | |
| **姓名** | **学号** | **专业** |
| **王峥** | **2211267** | **信息安全** |
| **李佳璐** | **2211985** | **信息安全** |
| **王承铃** | **2213906** | **信息安全** |
| **陈恩宝** | **2210977** | **信息安全** |

**完成时间：2024年7月14日**

1. **实验目的**

通过本实验，掌握web渗透测试技巧，对所学习的nmap扫描技术、sql注入、主机漏洞等对目标进行信息收集及漏洞分析，需要找到flag，并对实验过程截图。

注意：flag可能在数据库或者靶机某个用户桌面等地方。思路通过web漏洞找找数据库，并找到后台登录，找文件上传漏洞getshell，找系统漏洞提权等。

1. **小组分工**

视分工和贡献得分

|  |  |  |
| --- | --- | --- |
| 组员 | 分工 | 详细描述所做工作和贡献 |
| 王峥 | 进行漏洞挖掘并提出修复建议与报告整理 | 主要在后台挖掘数据库漏洞获取密码并利用navicat连接数据库夺旗与文本解析漏洞破解，并给出相关漏洞修复建议，整理报告 |
| 李佳璐 | 进行漏洞挖掘并提出修复建议 | 主要获取后台管理账号密码，挖掘文件上传漏洞以及系统漏洞提权并夺旗，同时给出相应的漏洞修复建议 |
| 陈恩宝 | 进行漏洞挖掘并夺旗 | 主要进行sql注入获取后台密码夺旗并挖掘数据库漏洞与利用夺旗 |
| 王承铃 | 扫描靶机并使用Nessus了解web | 主要使用nmap扫描网段并找到靶机，使用Nessus对web进行初步了解，并给出漏洞修复建议 |

1. **实验内容**

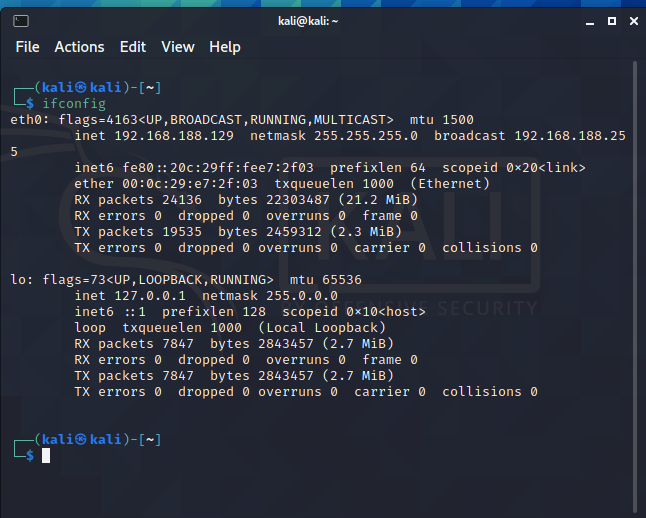
某公司的官方网站，现在需要进行上线前的渗透测试，发现问题并提出整改建议。视发现的漏洞以及修复建议是否合理得分。（注意：该系统在内网192.168.188.0/24，配置网络以及恢复win7服务器快照到attack状态，再进行测试。）



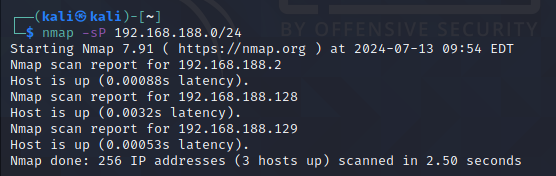
1. **实验步骤**
2. **利用nmap扫描靶机**

启动kali和win7，kali为操作机，win7为靶机。

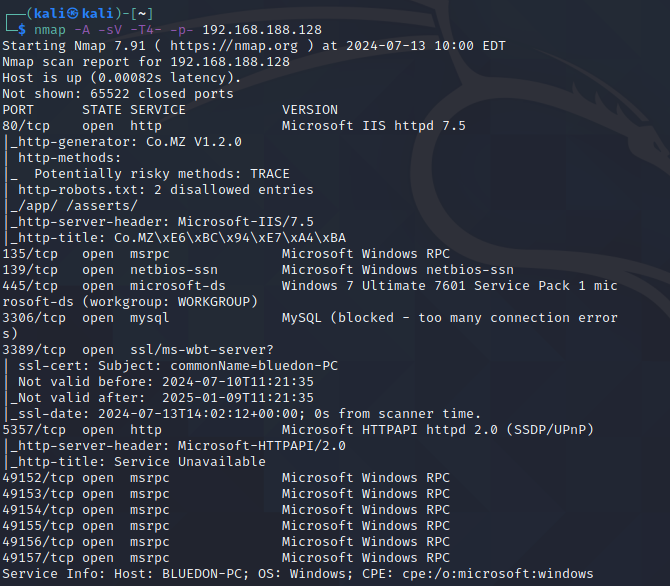
首先用ifconfig查询本机IP，得到本机IP为192.168.188.129。



找到本机IP后，通过nmap扫描同一区域内存活的所有主机的IP地址，发现靶机的IP地址为192.168.188.128。



接着用nmap扫描靶机开放的端口和服务，发现靶机开放的端口里有http端口——80端口，这是本实验主要的突破口。

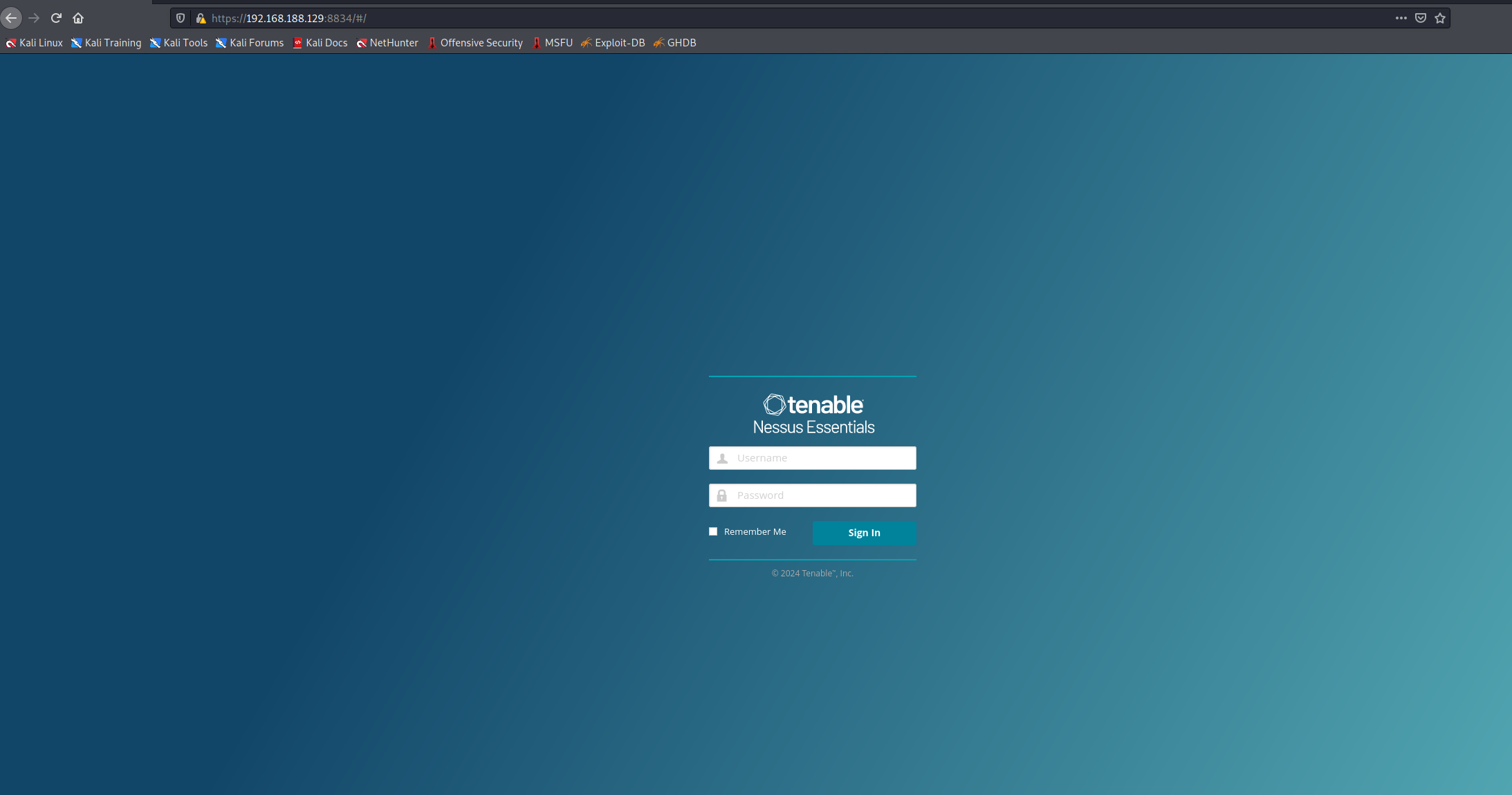


用浏览器访问80端口，可以成功进入到Co.MZ网站。

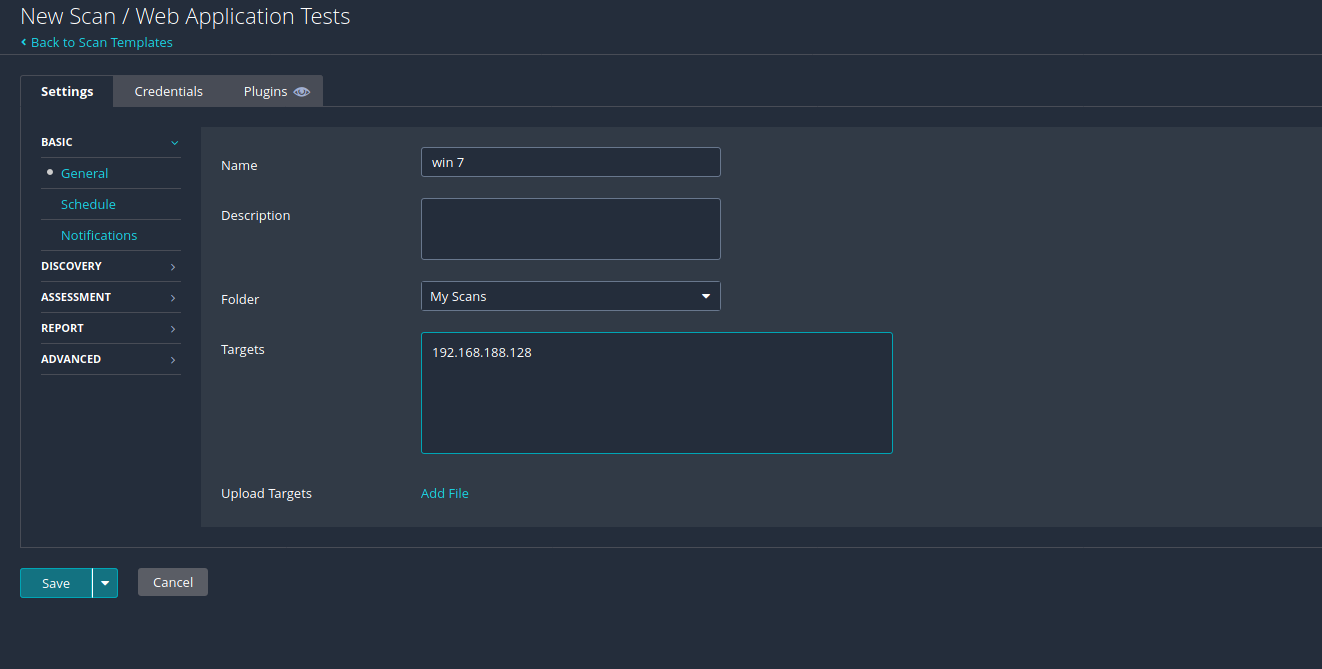


1. **通过Nessus初步对web进行了解**

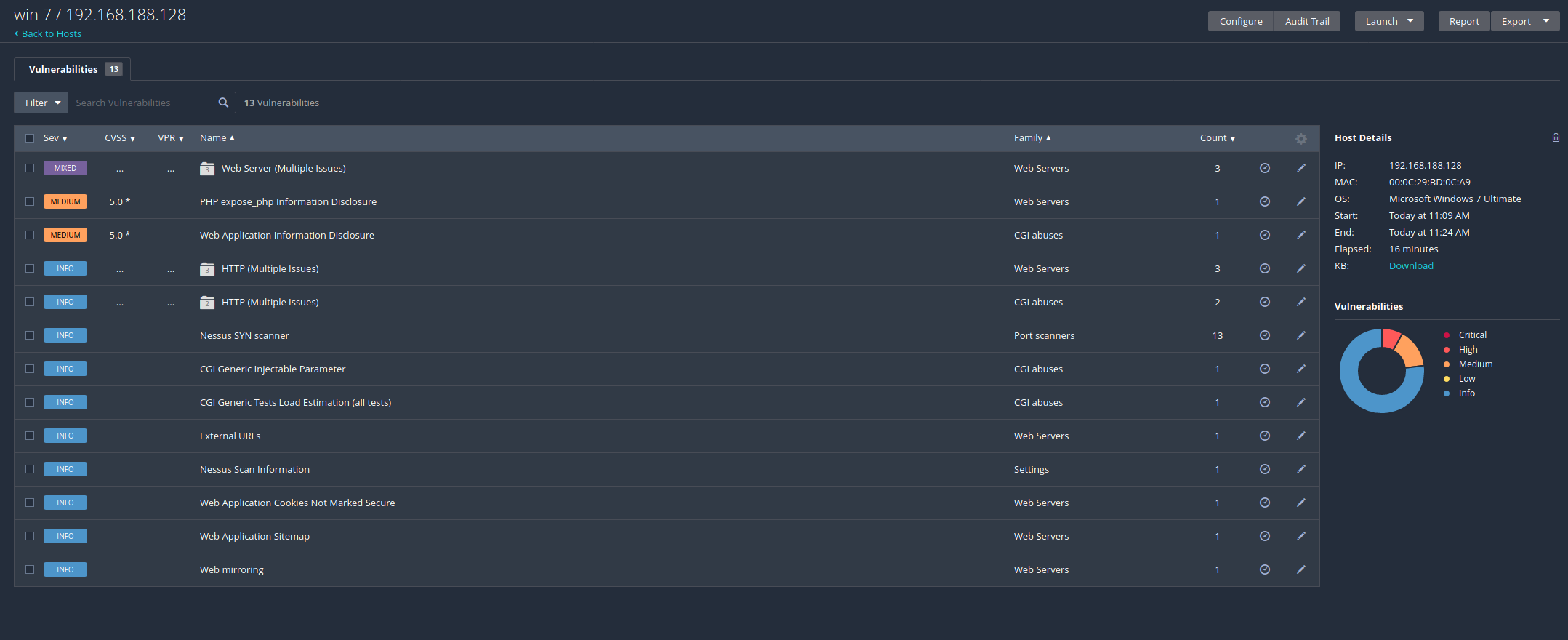
在kali中安装漏洞扫描工具Nessus，安装成功后用浏览器访问<https://192.168.188.129:8834>/#/即可进入。



登录进入，添加一个新的Web Application Tests扫描，此模板专注于扫描 Web 应用程序中的漏洞和安全风险。它会探测常见的 Web 漏洞，如跨站点脚本（XSS）、SQL 注入、敏感信息泄露等，并提供修复建议。扫描目标为靶机IP。

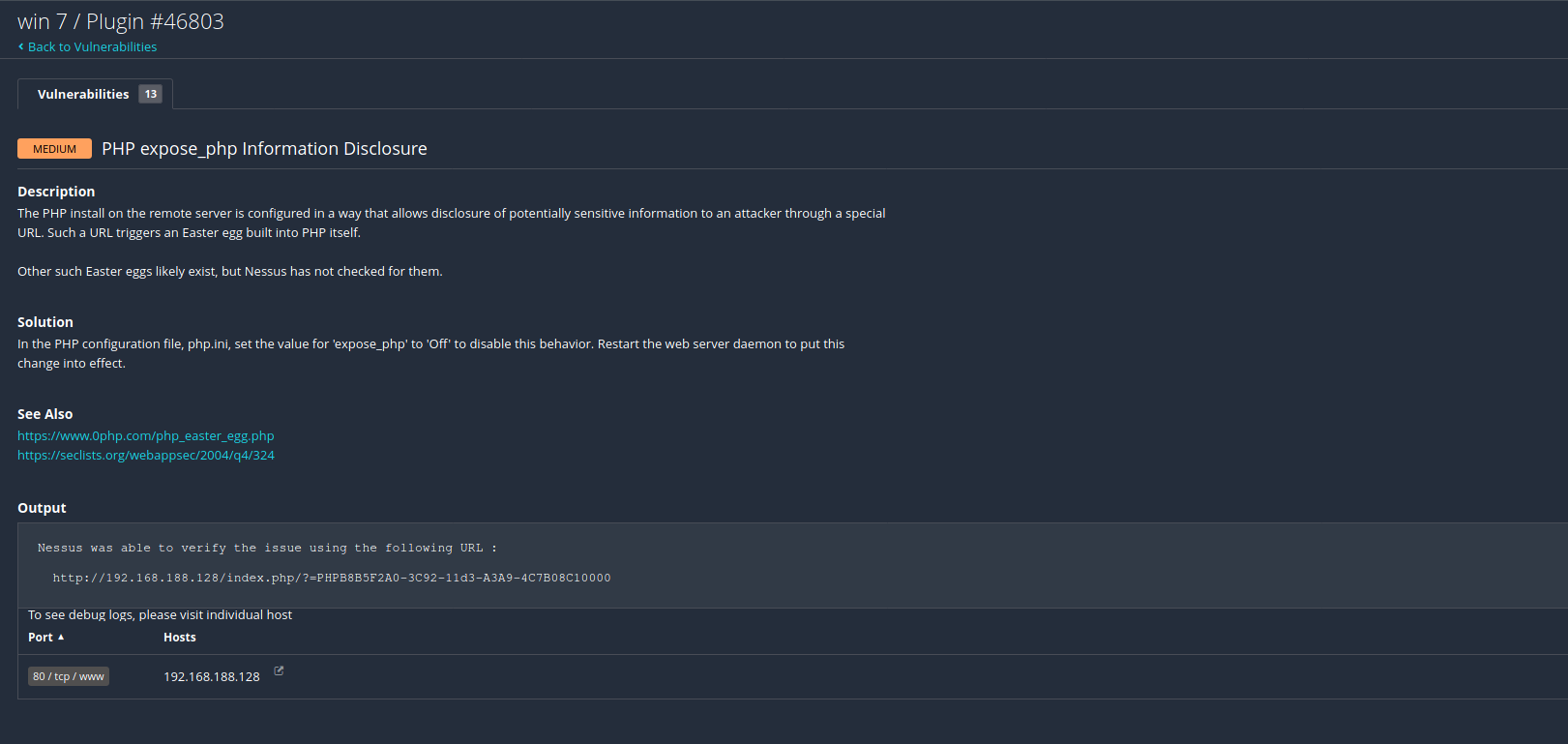


扫描结果如下：

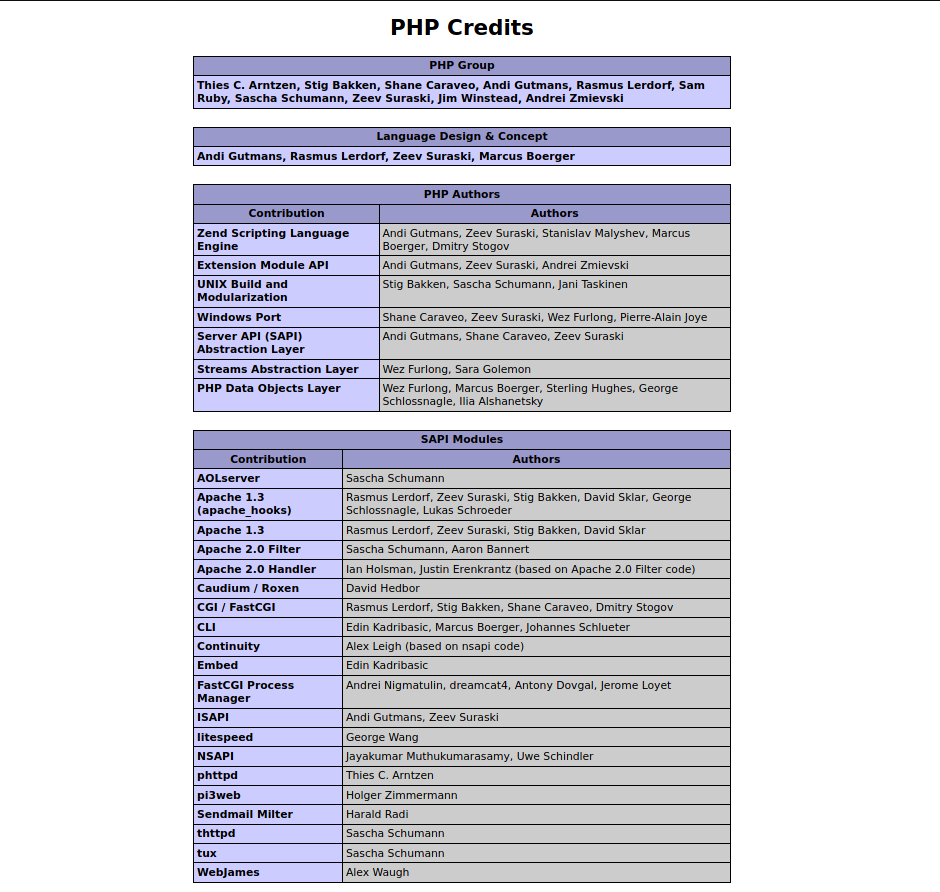


**PHP expose\_php Information Disclosure**

如扫描结果所示，靶机存在漏洞PHP expose\_php Information Disclosure，这是一种信息泄露漏洞，主要是因为在PHP配置中开启了expose\_php参数，导致服务器在响应HTTP请求时会向客户端披露服务器的PHP版本信息。黑客可以利用这个信息来进一步针对该服务器进行攻击。

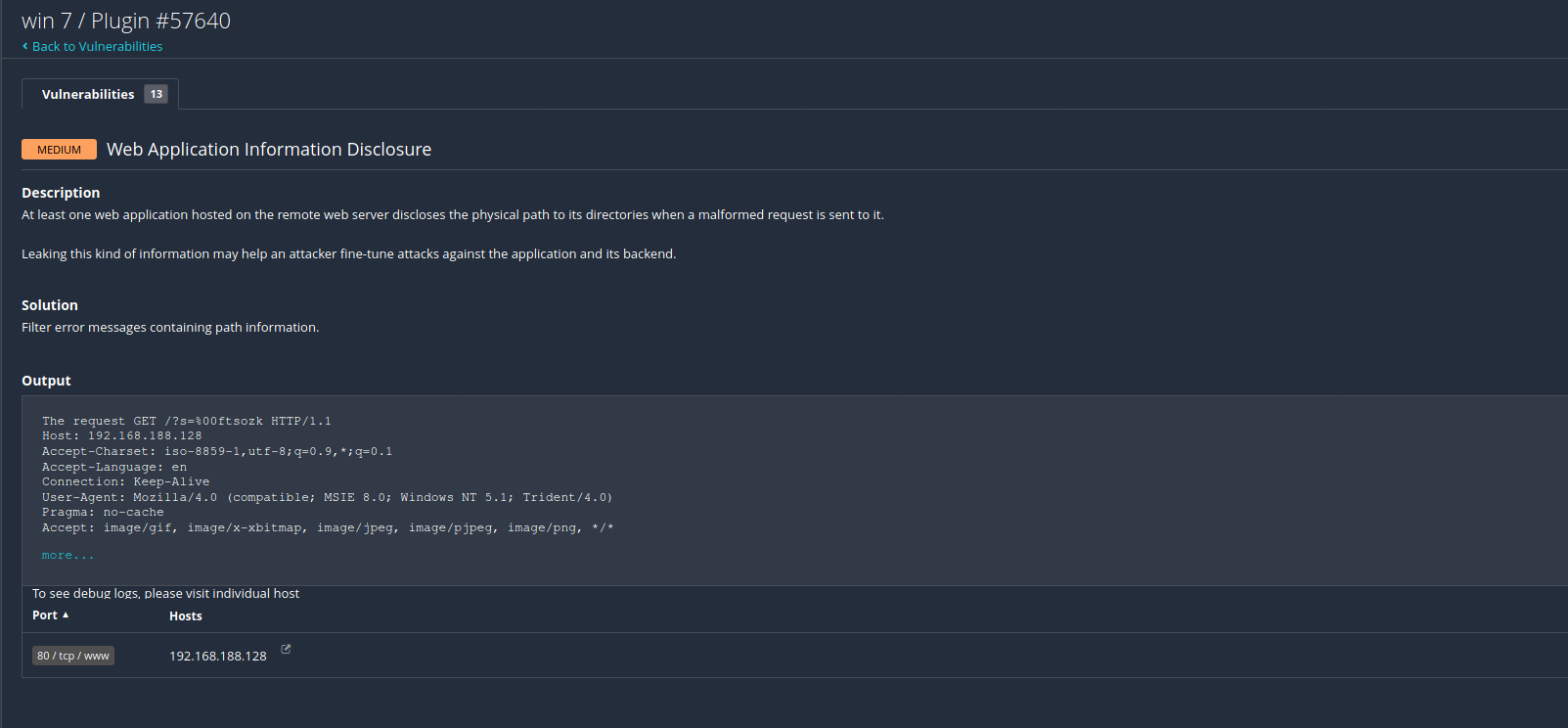


可以使用Nessus对该漏洞进行验证



**Web Application Information Disclosure**

如扫描结果所示，靶机存在漏洞Web Application Information Disclosure，这指的是Web应用程序中存在可能导致敏感信息泄露的漏洞。这可能包括但不限于配置文件、数据库凭据、用户信息、系统路径、错误信息等敏感数据被错误地暴露在Web应用程序的响应中，使攻击者可以利用这些信息来对系统进行攻击或者获取敏感信息。



1. **具体漏洞解析步骤**

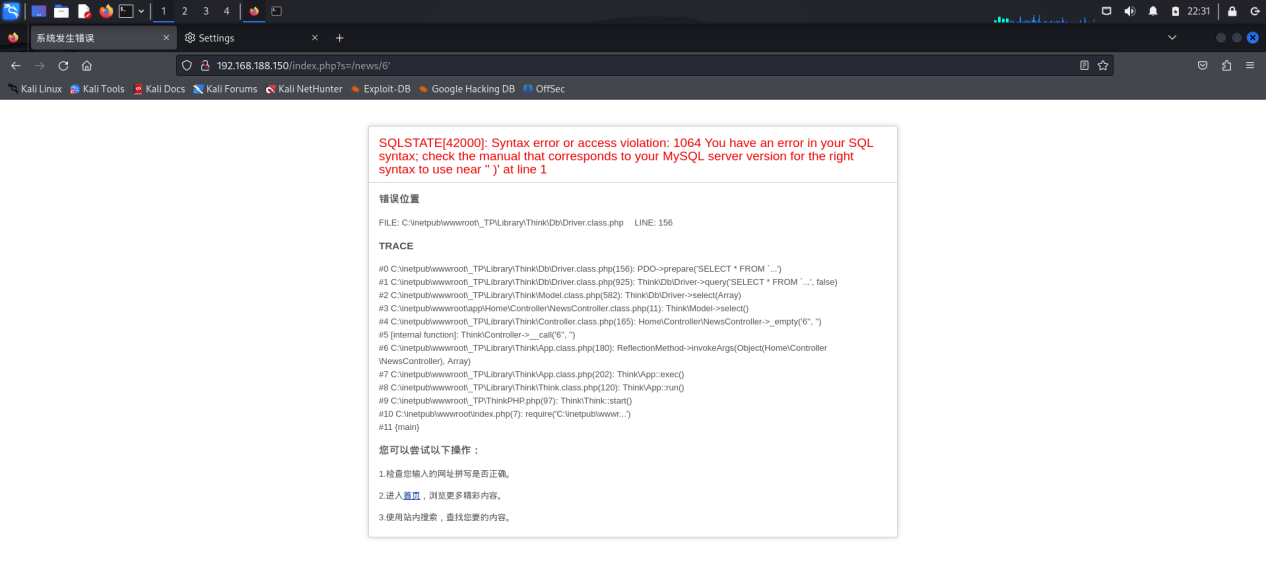
**步骤1：sql注入漏洞**

首先我们从前期的主机扫描等操作找到了对应进行渗透测试的网页，然后我们需要对其进行SQL注入漏洞的检测。

观察新闻页面的url，我们可以发现有参数传递，即“s=...”，猜测此处可能存在sql注入漏洞。



在发现可疑点之后我们需要验证是不是注入点，于是我们使用单引号法，即在参数最后添加一个“’”。



显示我们的sql语句缺失“）”闭合，于是我们尝试访问http://192.168.188.150/index.php?s=/news/6)--，发现页面正常。



接着进行 and 判断注入点，访问 http://192.168.188.150/index.php?s=/news/6) and 1=1--， 发现页面正常。



而访问 http://192.168.188.150/index.php?s=/news/6) and 1=2--时，返回首页，我们可以验证此处存在sql注入漏洞。



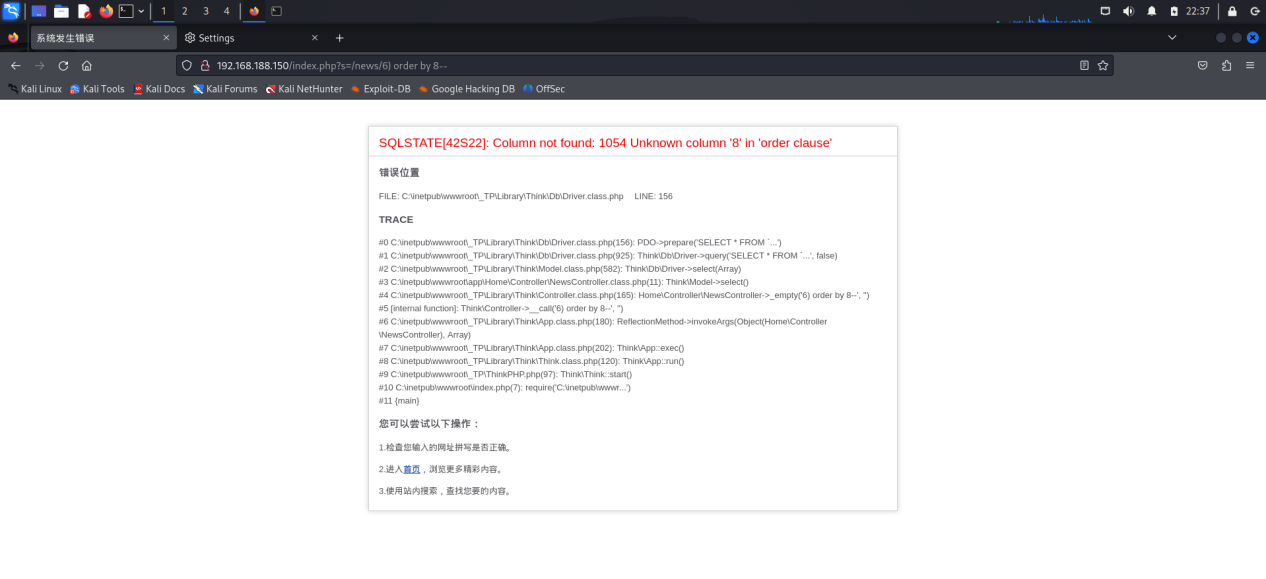
在判断存在sql注入漏洞之后我们就需要破解数据库，首先我们要知道字段的回显顺序。

通过 order by 猜测字段数，发现访问 http://192.168.188.150/index.php?s=/news/6)

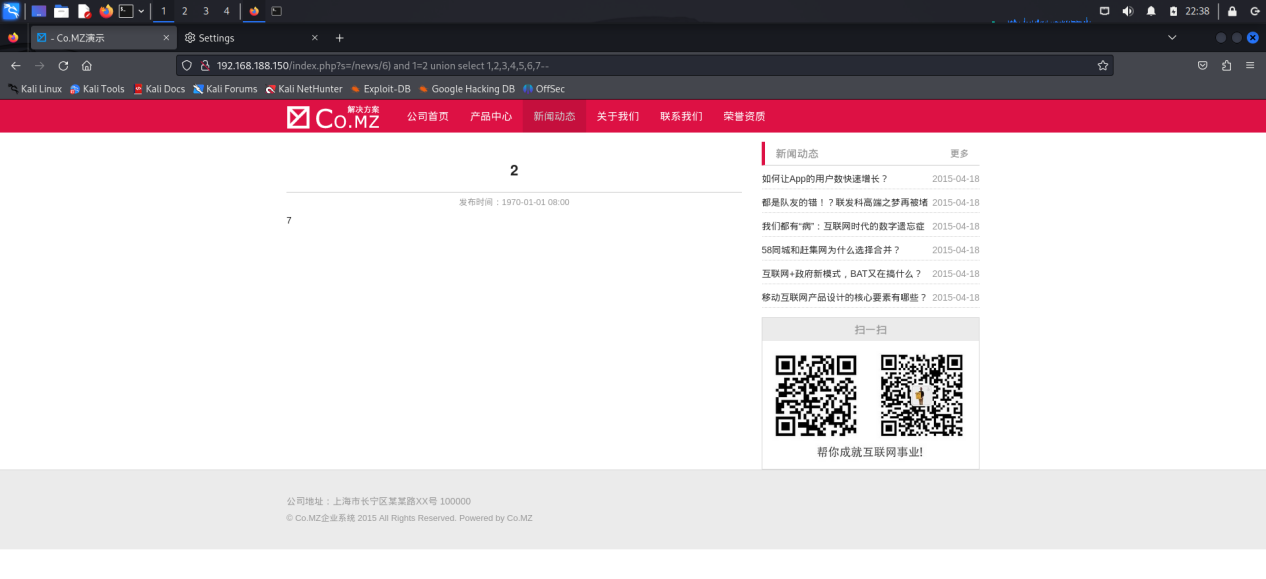
order by 7--时，新闻界面正常显示。



而访问 http://192.168.188.150/index.php?s=/news/6) order by 8 时，出现报错，说明字段数为 7。



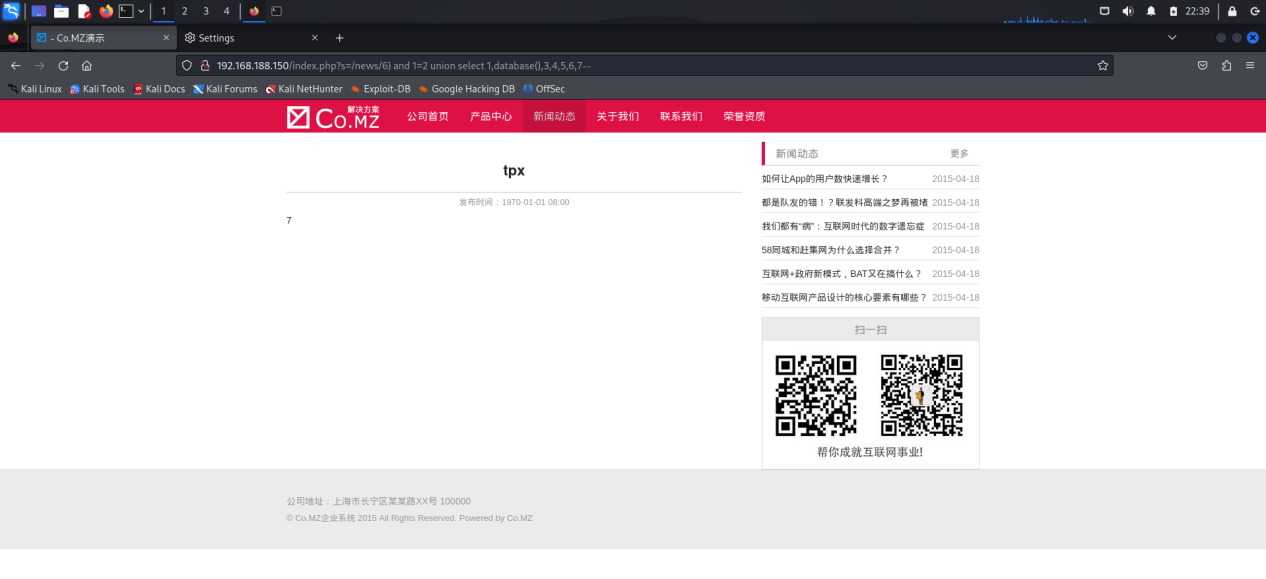
通过访问http://192.168.188.150/index.php?s=/news/6)and 1=2 unoin select 1,2,3,4,5,6,7--，可以确定回显位为 2 和 7。



确定回显顺序之后我们就可以尝试破解数据库名字了。

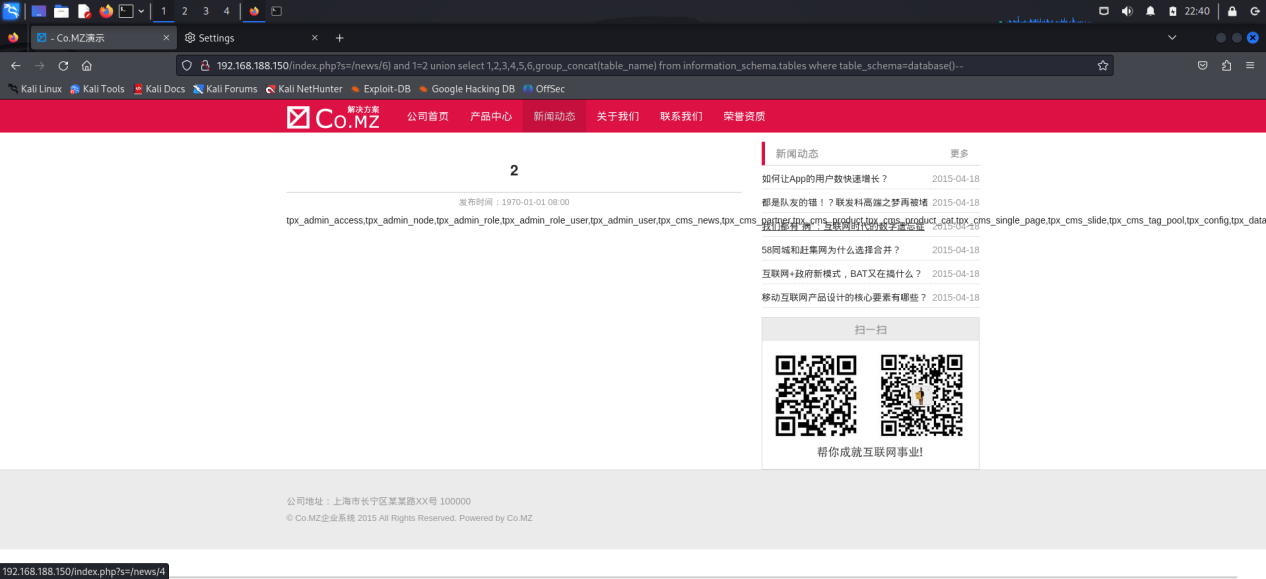
利用函数 database()，通过访问 http://192.168.188.150/index.php?s=/news/6) and 1=2

union select 1,database(),3,4,5,6,7--，得到库名为 tpx。

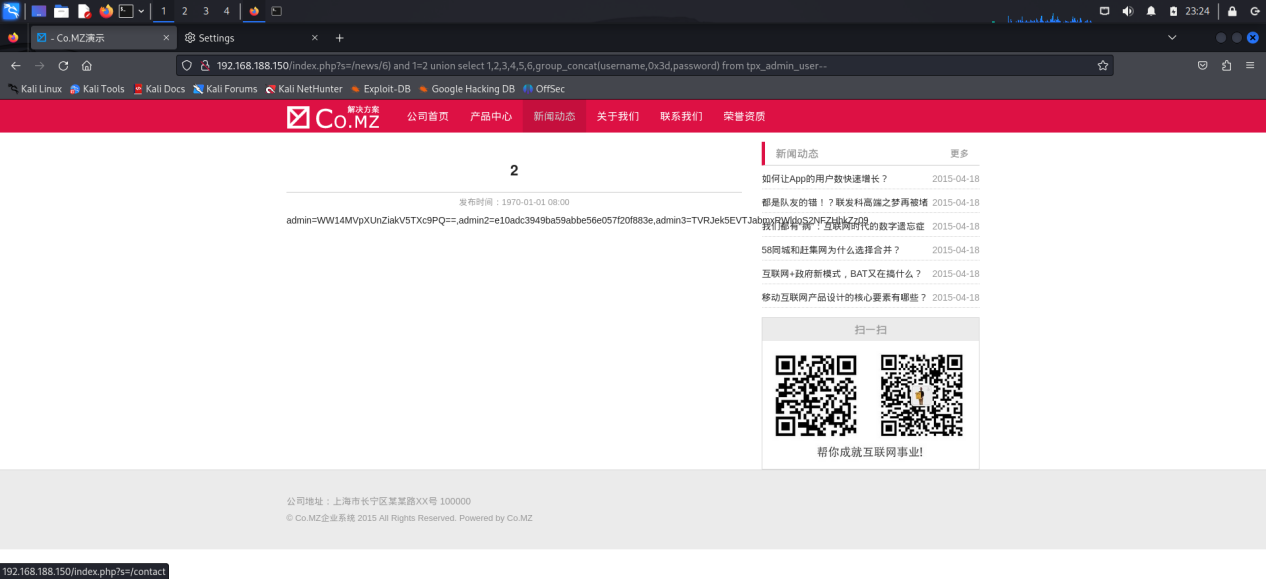


众所周知，information\_schema 数据库跟 performance\_schema 一样，都是 MySQL 自带的信息数据库。其中 performance\_schema 用于性能分析，而 information\_schema 用于存储数据库元数据(关于数据的数据)，例如数据库名、表名、列的数据类型、访问权限等。所以我们先通过访问information\_schema 数据库里的数据看是否能够得到有用的信息。

通过访问http://192.168.188.150/index.php?s=/news/6) and 1=2 union select1,2,3,4,5,6,group\_concat(table\_name) from information\_schema.tables where table\_schema=database()--，得到所有表名，其中发现表 tpx\_admin\_user。



在得知表名后，我们可以尝试破解这个表中的各个属性判断是否有有用到信息。通过访问 192.168.188.150/index.php?s=/news/6) and 1=2 union select 1,2,3,4,5,6,group\_concat(username,0x3d,password) from tpx\_admin\_user--，得到账号和密码，账号：admin，密码：WW14MVpXUnZiakV5TXc9PQ==

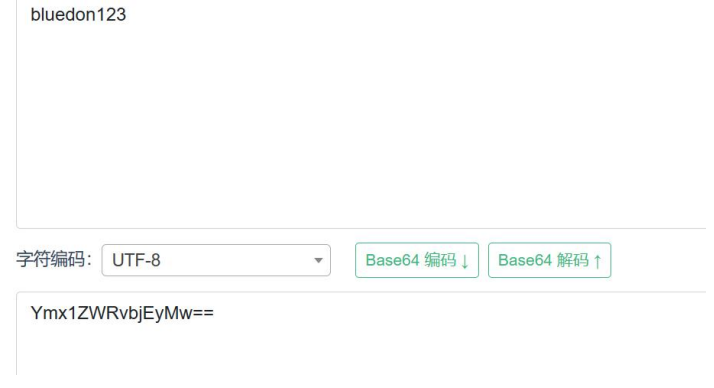


尝试后台登录，发现登录失败，说明这是一个加密后的密码，通常网页加密使用的是base64编码，于是经过 base64 在线工具进行解码。

经过 base64 在线工具第一次解码，发现仍然为加密后的数据，推测可能为多次加密

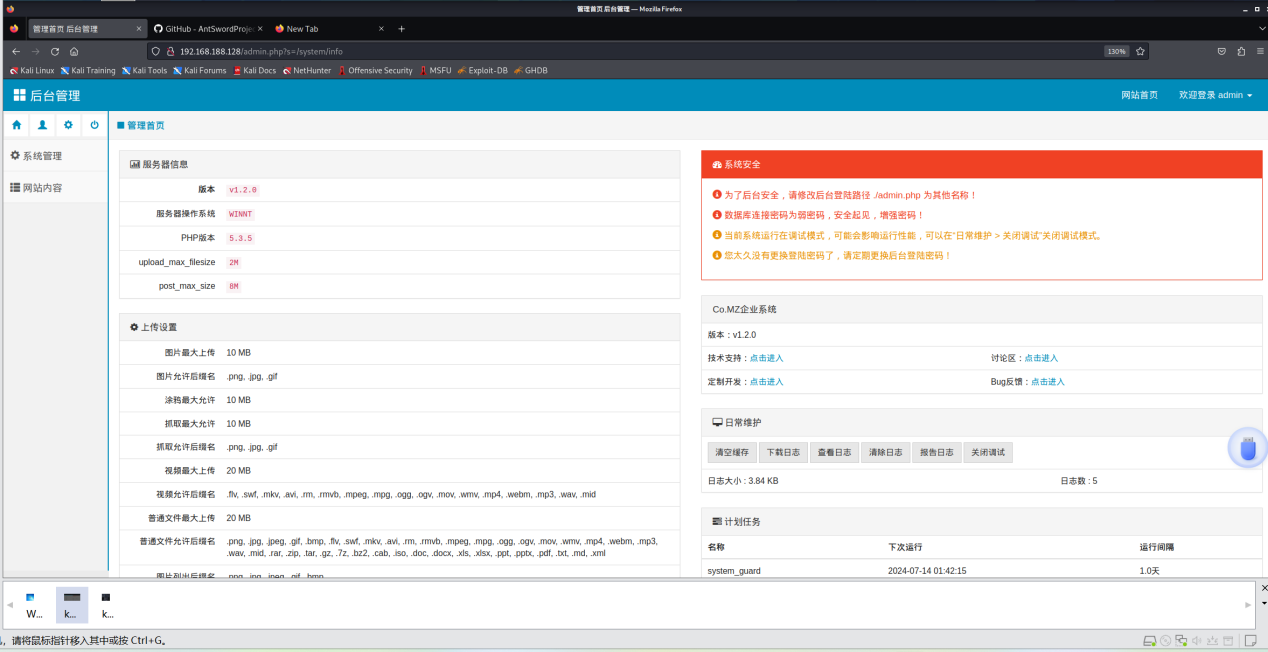


第二次解密后，得到解密后的登录密码为 bluedon123。



**步骤2：后台登录与数据库破解**

通过前期sql注入我们获取了用户名和密码，顺利进入后台管理



在系统管理-数据库-数据库操作中，我们输入sql语句：

Select \* from mysql.user ，找到数据库的密码



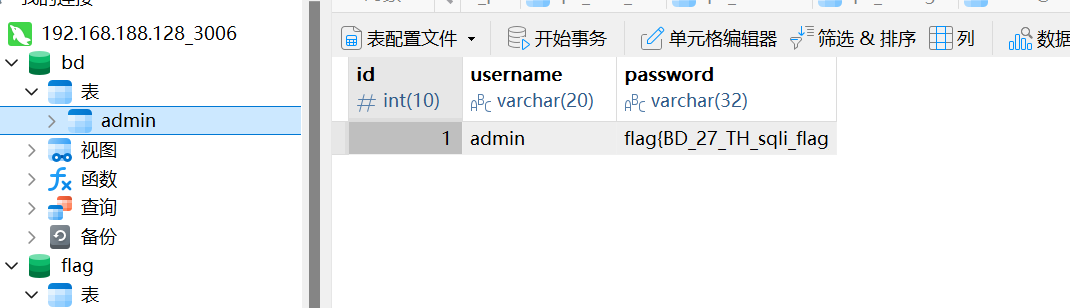
密码经过md5的破解，我们发现是123456

在navicat上我们连接上相应的数据库

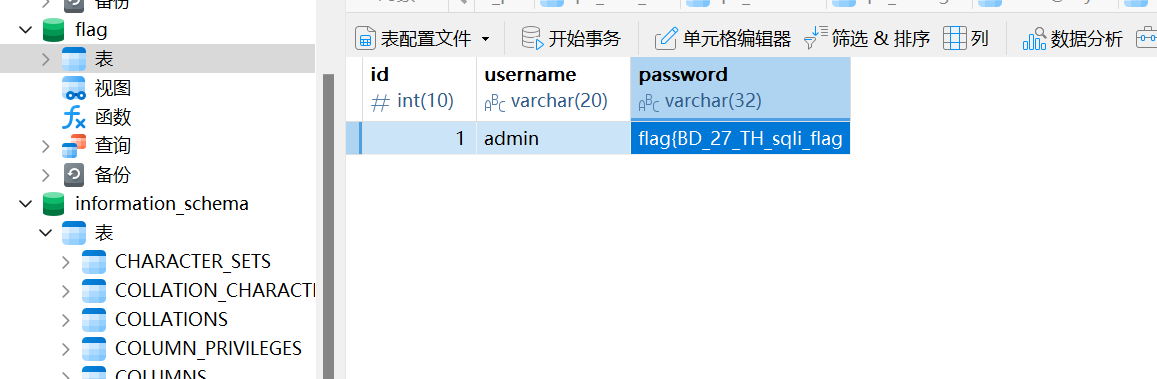


连接成功后，我们在数据库中发现了相应的flag

分别在db数据库admin表和flag数据库中flag表中发现：

Flag{BD\_27\_TH\_sqli\_flag\_kgesfa}  


flag{BD\_27\_TH\_sqli\_flag\_kgesfa}\



**步骤3：文件解析漏洞**

创建后缀名为.Png的文件，文件内容如下：



将文件通过logo进行上传



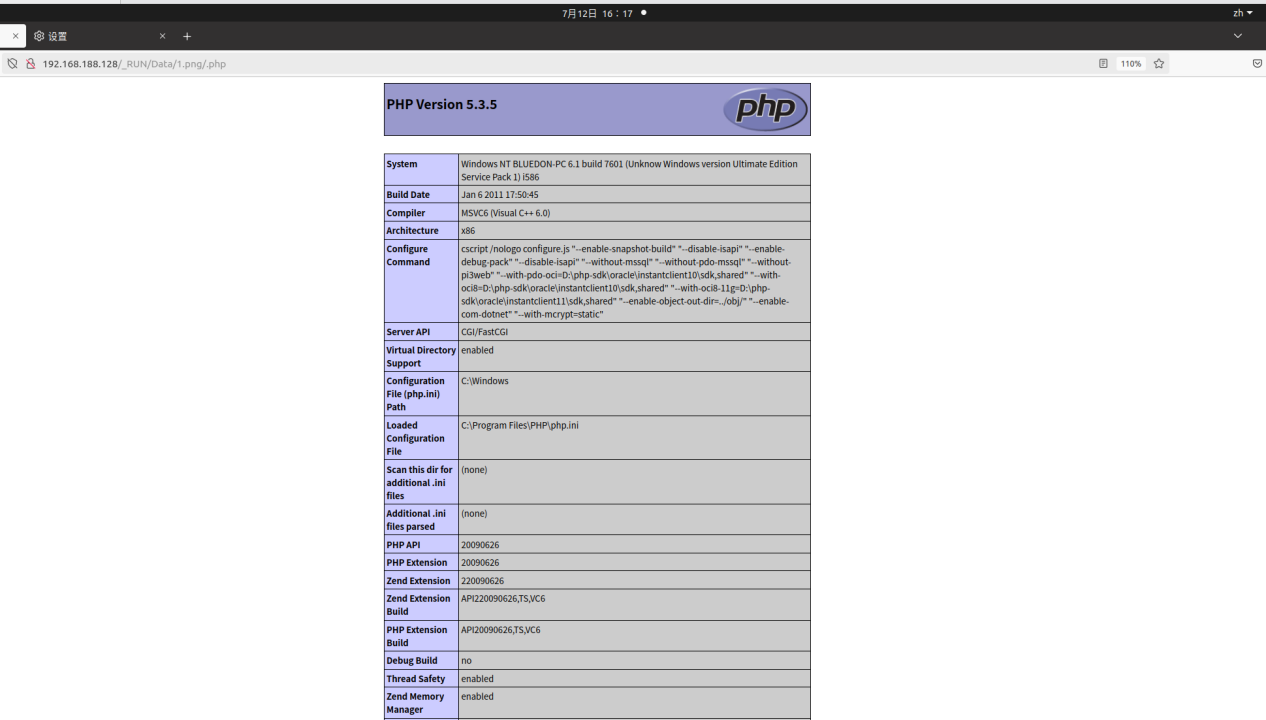
访问文件，发现无法解析



利用IIS7.5文件解析漏洞的特性进行访问

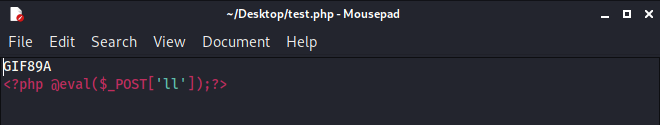
<http://192.168.188.128/_RUN/Data/1.png/.php>

文件被成功解析

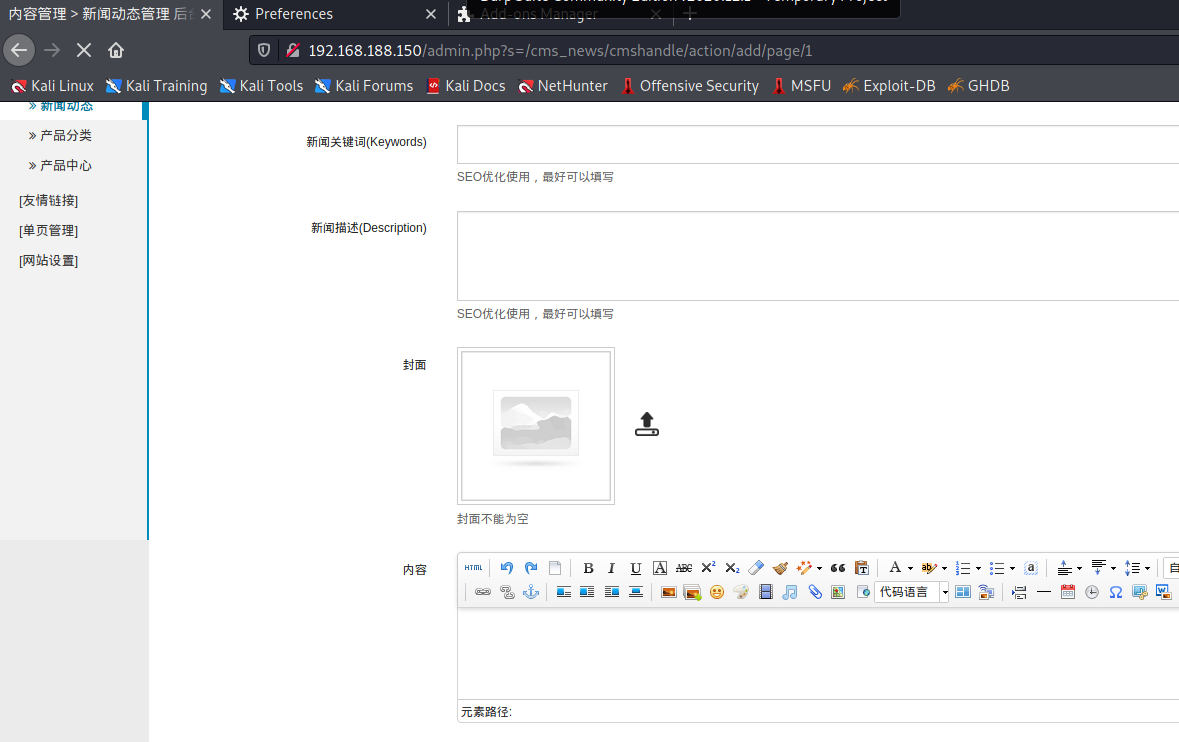


**步骤4：文件上传漏洞**

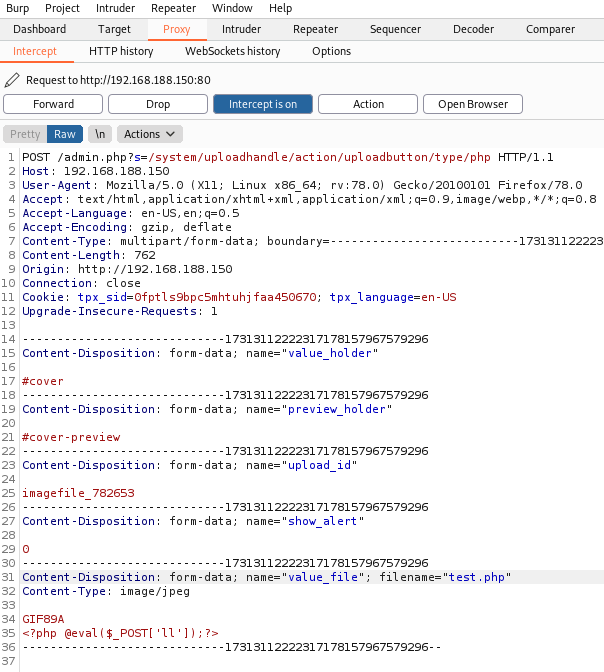
编写一句话代码并更改后缀名为.jpg



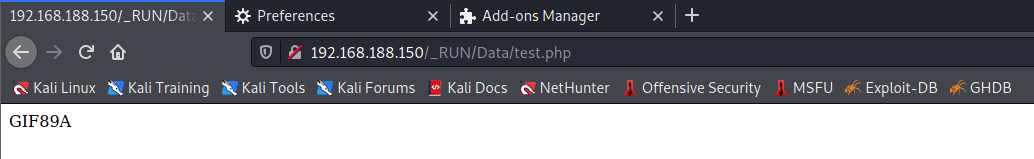
查找可能存在文件上传漏洞的位置



上传该图片文件并抓包，更改文件类型和后缀名后forword

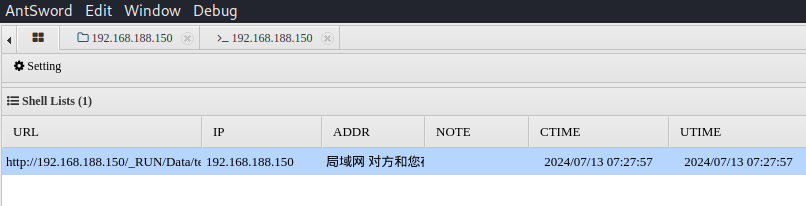


复制图片路径并查找

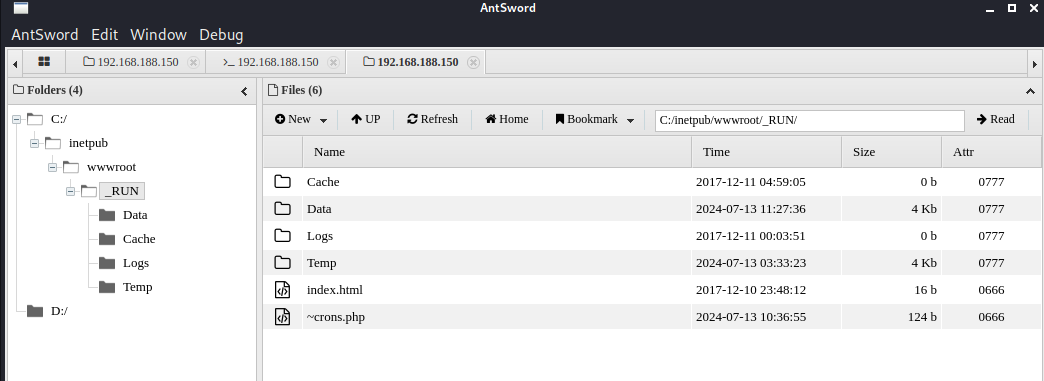


可见上传成功

连接蚁剑，并添加该URL

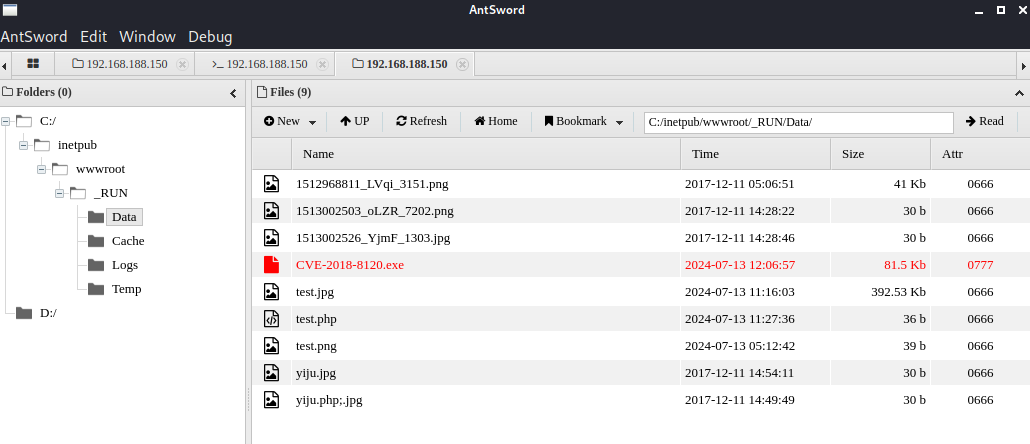


查看文件管理

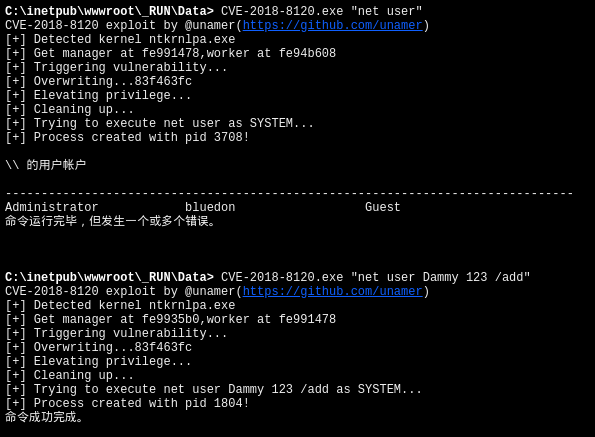


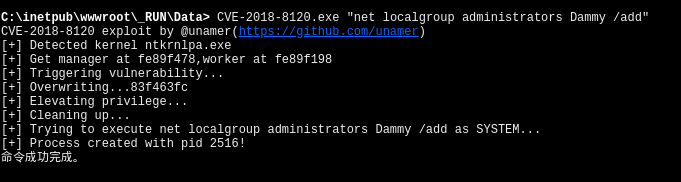
可见之前上传的文件和该网页内其他文件，实现了文件上传漏洞。

**步骤5：系统提权**

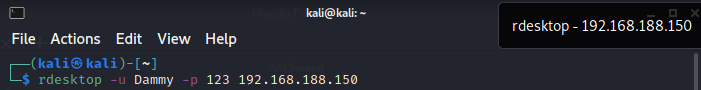
利用CVE-2018-8120进行提权，将工具上传到服务器上

使用其针对靶机添加新用户（Dammy \ 123），并将其添加至管理员组中

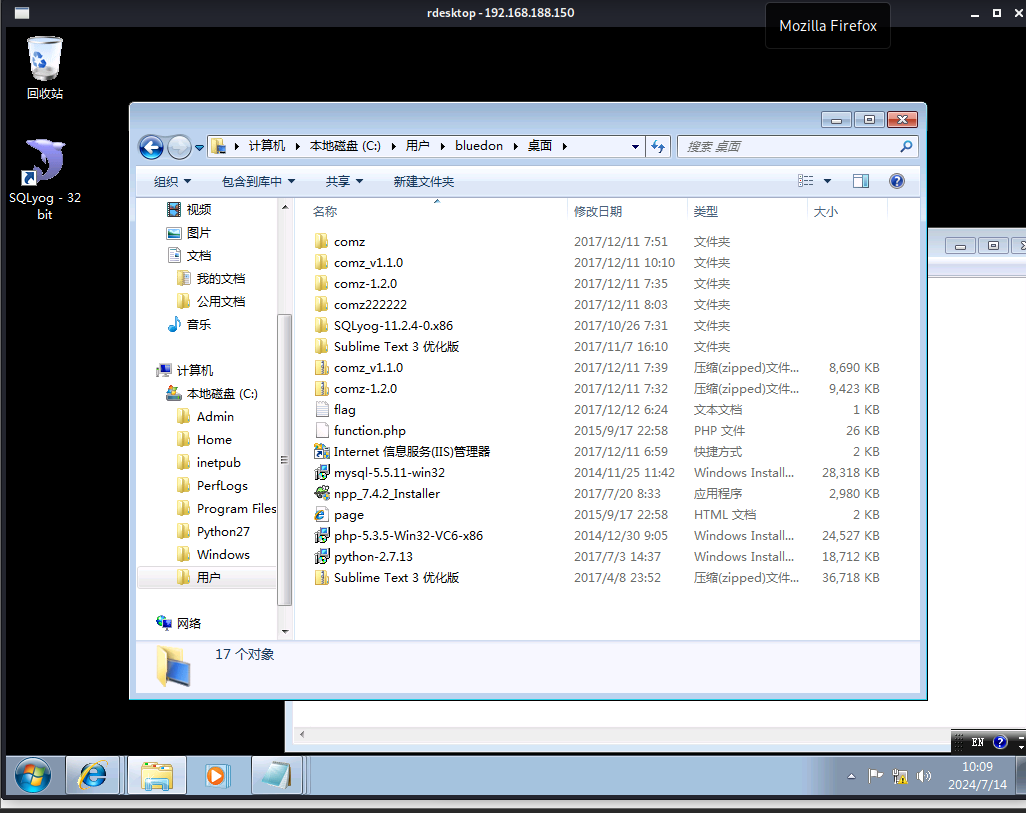


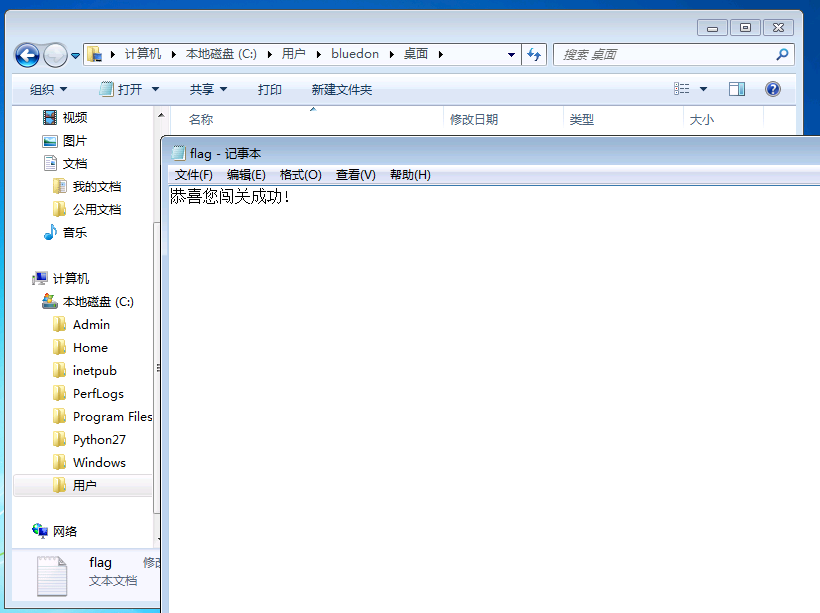


进行远程连接



进入C盘查看所有用户，在用户bluedon的桌面获得flag！





1. **修复建议**

* PHP expose\_php Information Disclosure：

1.关闭expose\_php参数：在PHP配置文件（php.ini）中，找到expose\_php参数并将其设置为Off。这样可以防止服务器在响应HTTP请求时泄露PHP版本信息。

2. 重启Web服务器：在修改php.ini文件后，需要重启Web服务器以使更改生效。不同的服务器软件有不同的重启命令，例如对于Apache服务器，可能需要运行sudo service apache2 restart（基于Debian系）或sudo systemctl restart httpd（基于Red Hat系）

3. 检查.htaccess文件：如果无法直接修改php.ini文件，或者想要额外的安全层，可以在.htaccess文件中添加规则来阻止expose\_php相关的查询字符串

4. 更新和维护：确保PHP安装保持最新，以包含最新的安全修复和更新。定期检查更新和安全补丁，并及时应用。

* Web Application Information Disclosure：

1.限制错误信息泄露：确保Web应用程序在生产环境中不会向用户公开敏感错误信息，可以通过正确配置Web服务器和应用程序来限制错误信息的泄露。

2.加强访问控制：检查Web应用程序的访问控制机制，确保用户只能访问他们具有权限的资源，避免未授权访问敏感信息。

3.定期漏洞扫描和安全审计：定期进行漏洞扫描和安全审计，及时发现并修复可能导致信息泄露的漏洞。

4.数据加密：对Web应用程序中传输的敏感数据进行加密处理，确保数据在传输和存储过程中得到保护

5. 输入验证与过滤：实施严格的输入验证，确保所有用户输入都经过验证，以防止恶意数据进入系统。这包括检查输入的长度、类型和格式。使用白名单过滤，只允许已知安全的输入通过，拒绝所有未知或不安全的输入。

6. 禁用详细错误消息显示： 确保在生产环境中禁用详细错误消息显示，不要向用户公开网站的内部错误信息。可以在Web服务器配置中设置自定义错误页面，用简洁的错误提示替代详细的错误信息。

* Sql注入修复建议：

1.使用预编译语句：将SQL查询与数据分离，避免了将用户输入直接嵌入到SQL语句中的风险。使用预编译语句时，SQL引擎会先解析SQL语句，而后再绑定参数，从而防止SQL注入。

2.使用ORM框架：自动处理SQL语句的查询，减少手写SQL语句的机会，从而降低SQL注入的风险。

3.对输入进行验证和过滤：对用户输入的数据进行严格的验证和过滤，确保输入符合预期格式，不含有SQL控制字符或其他特殊字符。

4.将SQL查询封装在存储过程中，避免在应用程序中直接使用动态SQL语句，存储过程可以在数据库端进行参数化查询，从而提高安全性。

* 数据库漏洞修复建议

1.强化数据库访问权限：强化身份验证和授权采用**强密码策略从而**确保数据库用户账户使用强密码策略，包括使用长密码、混合使用大小写字母、数字和特殊字符。同时使用最小权限原则，即仅授予用户执行其任务所需的最小权限。例如，应用程序账号应仅具有读写权限，不应有删除或管理权限。同时限制方位范围，采取IP白名单来配置数据库服务器仅允许特定IP地址或IP范围的访问，阻止来自未知或不可信来源的连接，并使用防火墙和网络安全组来限制数据库服务器的访问范围。

2.及时更新数据库软件和补丁：定期检查数据库供应商的安全公告，及时更新数据库软件和补丁以修复已知漏洞。如果可能，配置自动更新以确保数据库系统始终运行最新和最安全的版本。

3.加强后台访问控制： 限制访问，限制**IP ，**配置后台管理界面只允许来自特定IP地址的访问，并使用**多因素身份验证（MFA）**增加安全性，结合使用密码和其他身份验证方式（如短信验证码、认证应用或硬件令牌）。建立**访问日志记录，**记录所有管理界面的访问日志，监控异常的登录尝试和活动并做到**实时监控，**检测和响应可疑活动。

### 4. 加密敏感数据：**采用静态数据加密，**对存储在数据库中的敏感数据进行加密，例如使用AES、RSA等强加密算法。以及**传输层加密，即**使用TLS/SSL协议加密客户端与数据库服务器之间的数据传输。同时，加强**安全的密钥管理，**确保加密密钥安全存储和管理，定期更换密钥，并限制对密钥的访问。

* 文件上传漏洞修复建议

1. 服务器配置：

（1）将上传目录和上传文件设置为不可执行， 杜绝脚本执行。

（2）应保证服务器安全，避免文件解析漏洞。

2. 在服务端对上传文件进行检查：

（1）使用白名单控制上传文件类型，即只允许指定扩展名的文件上传。

（2）对上传文件后缀与MIME Type进行匹配校验， 对文件头信息与文件后缀进行匹配校验。

（3）对单个文件大小和总文件数进行限制， 避免拒绝服务攻击。

（4）对文件名进行输入校验，显示时进行输出编码。

3. 文件存储：

（1）上传文件应保存在指定路径下。

（2）对上传文件进行随机数重命名，避免文件被覆盖。

（3）设置上传文件路径， 使用户不能轻易访问自己上传的文件 。

（4） 文件应尽量保存在内容服务器或web目录外部，避免通过web应用直接访问上传的文件。

4. 对于图片文件进行二次渲染、压缩， 避免图片写马。

5. 校验失败后，记录错误日志信息，内容至少包括时间、用户、IP、操作内容、校验失败的参数及参数内容等。

* 利用CVE-8-8120提权修复建议

1. 安装补丁程序：Microsoft已经发布了修复此漏洞的安全补丁。确保操作系统上安装了最新的安全更新。你可以通过Windows Update或者Microsoft Update Catalog手动下载和安装相关补丁。
2. 更新操作系统：确保操作系统及时更新到最新版本。较新的操作系统版本通常会包含安全性改进和修复漏洞的更新。
3. 监控和审计：在安装了补丁后，建议监控系统以检测是否有异常行为或未经授权的提权尝试。定期审计系统日志可以帮助发现潜在的安全问题。
4. 限制用户权限：为了减少潜在攻击的影响，用户应该以最低权限原则操作系统。管理员权限应该仅在需要时才授予。
5. 检查系统是否受CVE-2018-8120等各种漏洞的影响，并及时更新内核以修复该漏洞。定期检查和更新系统是确保服务器安全的重要措施。

**六、个人心得**

**王峥**：通过这次web渗透测试的练习，我第一次完整地体验了漏洞挖掘与利用的过程，同时也不断扩充了我的知识库，包括包括操作系统、网络协议、编程语言等方面的知识。同时我也深刻的感受到漏洞挖掘的不易，需要有极强的耐心去完成，中间会遇到很多问题，也会有很多报错，甚至会不停的变换思路，这更需要我能更熟练的使用安全工具。同时我们也在网上进行了很多自主学习，发现了用Nessus去探索漏洞，帮助我们先全面的认识靶机。同时我们在搜集方法同时也发现很多技术更新迭代很快，这也需要我们不断保持学习和跟进行业的最新动态，同时加强对网络安全的认识也是十分重要。希望通过在未来的学习

，我能够不断精进自己的技能，为国家的网络安全事业贡献自己的力量。

**李佳璐：**在本次web渗透测试中，我学习了许多web渗透技巧。通过sql注入获取管理后台账号和密码，查询其中可能存在的文件上传漏洞，并通过抓包获得文件管理，再通过提权获得靶机的管理权限，并通过远程连接查看所有用户内容，并成功获取到了flag。在整个过程中，不断试错、不断收获新技巧和新方法，对与web渗透有了更直接的感受，可见存在系统漏洞的极大危险性，对方可通过渗透技巧直接提权，掌握系统的所有用户信息。现下的系统在不断更新过程中安全性已经提高了很多，但所谓“道高一尺，魔高一丈”，应时刻注意，谨防安全漏洞，及时发现，及时修复，防止其可能会带来的不可估量的风险和损失。

**王承铃：**在进行这次实验的过程中，我学到了很多关于Web渗透测试的技巧和方法。通过使用nmap扫描技术，我可以快速有效地获取目标系统的信息，识别潜在的漏洞和攻击面。同时，学习了如何使用漏洞扫描工具Nessus，能够利用Nessus对靶机进行扫描，发现靶机漏洞。另外，学习了利用SQL注入、文件解析、文件上传等技术对目标系统进行渗透测试的相关知识。通过模拟攻击和漏洞利用，我深刻认识到在现实世界中，安全漏洞可能会给企业带来巨大的风险和损失。因此，从渗透测试中提出的整改建议是至关重要的，可以帮助企业及时修复漏洞，提高系统的安全性和稳定性。通过这次实验，我意识到了Web渗透测试的重要性，以及对安全威胁和漏洞进行及时发现和解决的必要性。我将继续学习和提升自己在渗透测试领域的技能，为保护网络安全和数据安全贡献自己的一份力量。

**陈恩宝：**在本次实验中，对于web渗透的基本技巧及基本知识进行了进一步地学习，实践，使我更好地掌握了web渗透的基本技巧和方法，如主机的扫描和发现，SQL注入的流程，漏洞扫描等等。在这个过程中，使我明白了发现漏洞的重要性，渗透测试揭示了系统中存在的各种漏洞和安全风险。这不仅仅是技术问题，更是组织安全的重要组成部分。给我带来了安全意识的提升，通过测试，意识到安全意识的重要性，不仅仅是技术人员，整个团队都需要理解并采取相应的安全措施。并且让我意识到了实际攻击和防御的差距，测试过程中可能发现实际系统的防御措施和理论上的安全最佳实践之间存在差距。这需要更加有效地实施和更新安全策略。最后使我明白了合作和沟通的重要性，渗透测试是一个需要团队合作和有效沟通的过程。安全团队之间的良好沟通可以帮助更快速地解决问题和改进安全性。