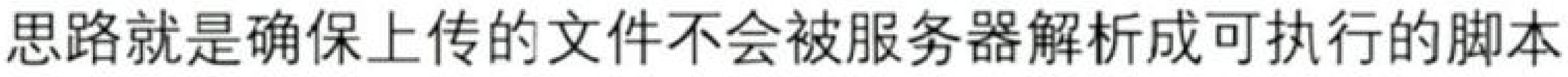
AWD（Attack With Defence）模式比赛技巧

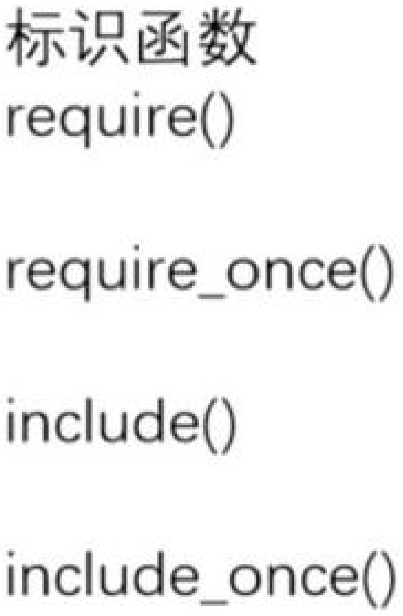
第5课 AWD中WEB向常见的题目类型-sql_server

文件上传



思路就是确保上传的文件不会被服务器解析成可执行的脚本

文件包含漏洞



标识函数

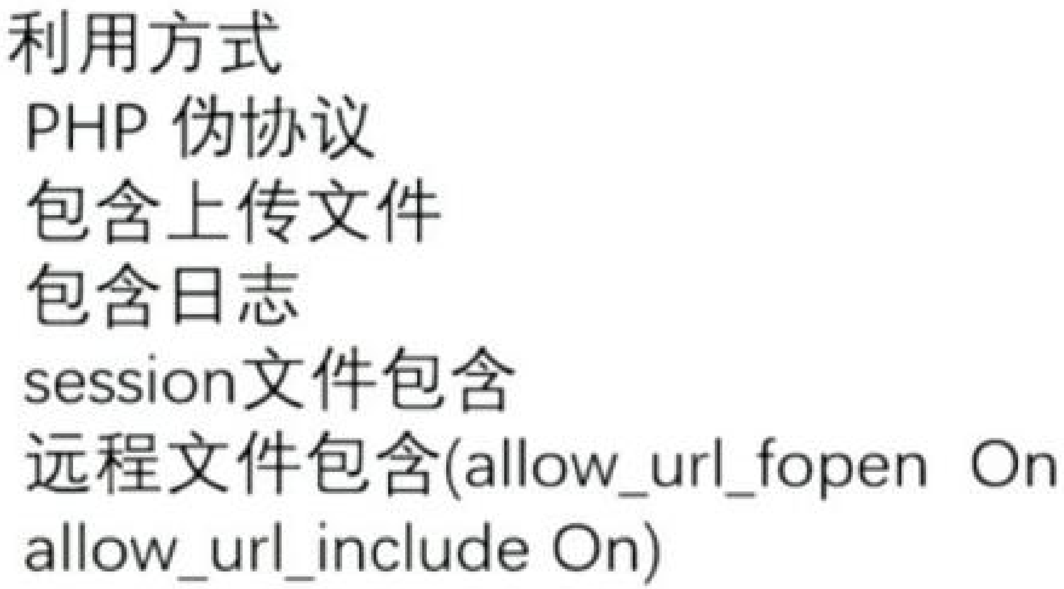
require()

require_once()

include()

include_once()

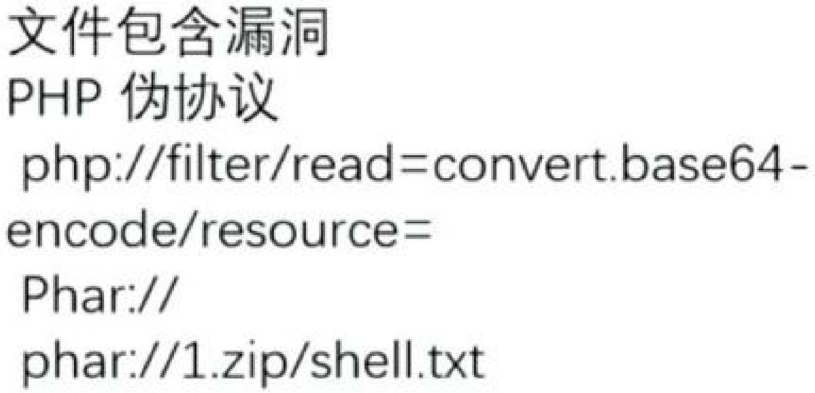
文件包含漏洞



利用方式

- PHP 伪协议
- 包含上传文件
- 包含日志
- session文件包含
- 远程文件包含(allow_url_fopen On allow_url_include On)

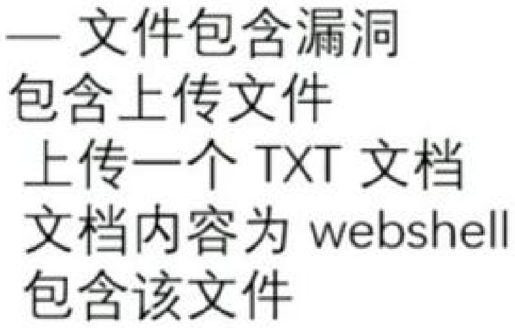
文件包含漏洞



文件包含漏洞

PHP 伪协议

```
php://filter/read=convert.base64-encode/resource=
Phar://
phar://1.zip/shell.txt
```



— 文件包含漏洞

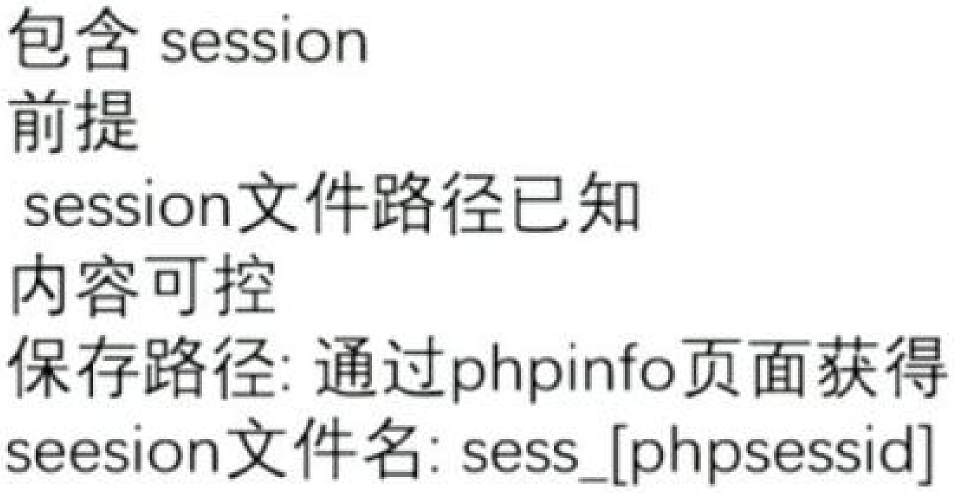
包含上传文件

上传一个 TXT 文档

文档内容为 webshell

包含该文件

文件包含漏洞



包含 session

前提

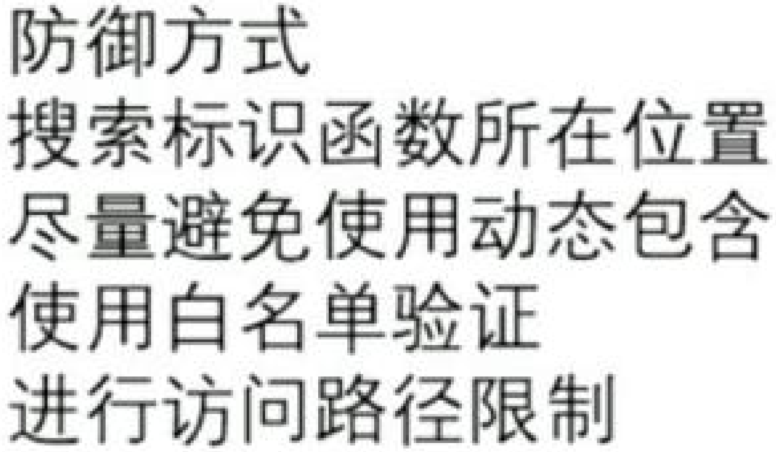
session文件路径已知

内容可控

保存路径: 通过phpinfo页面获得

seesion文件名: sess_[phpsessid]

文件包含漏洞



防御方式

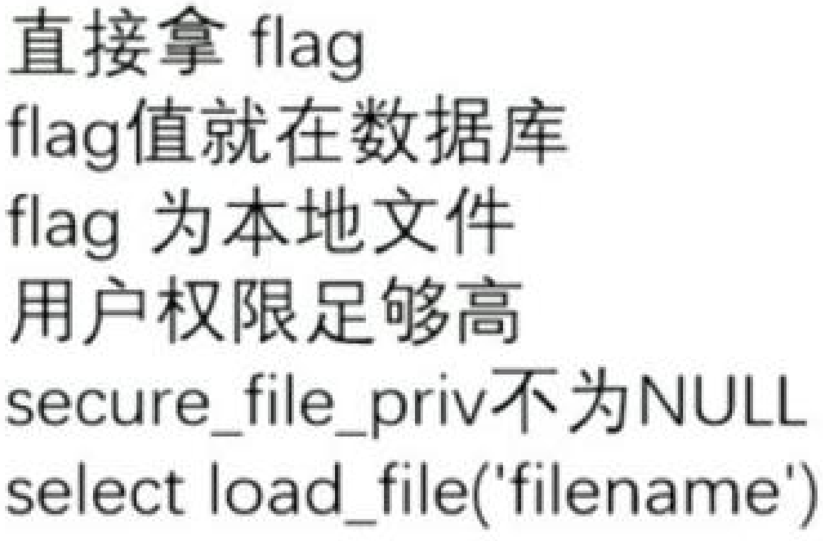
搜索标识函数所在位置

尽量避免使用动态包含

使用白名单验证

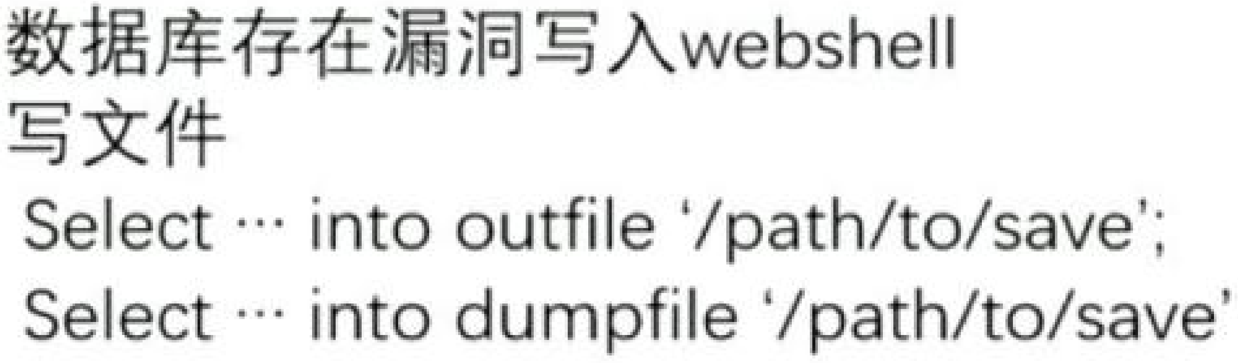
进行访问路径限制

数据库层面



直接拿 flag
flag值就在数据库
flag 为本地文件
用户权限足够高
secure_file_priv不为NULL
select load_file('filename')

数据库层面

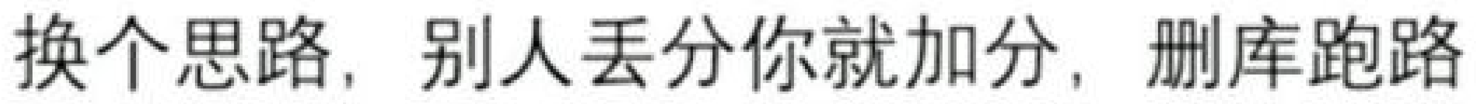


数据库存在漏洞写入webshell

写文件

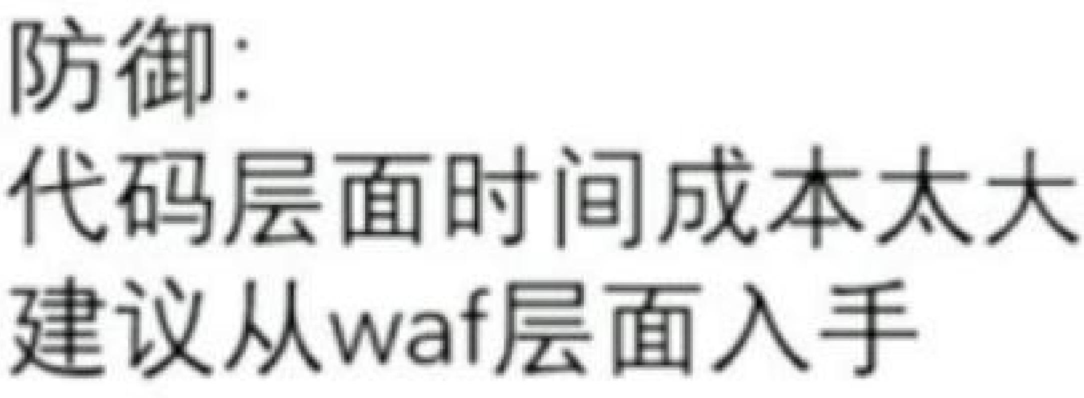
```
Select … into outfile ‘/path/to/save’;
Select … into dumpfile ‘/path/to/save’
```

数据库层面



换个思路，别人丢分你就加分，删库跑路

数据库层面

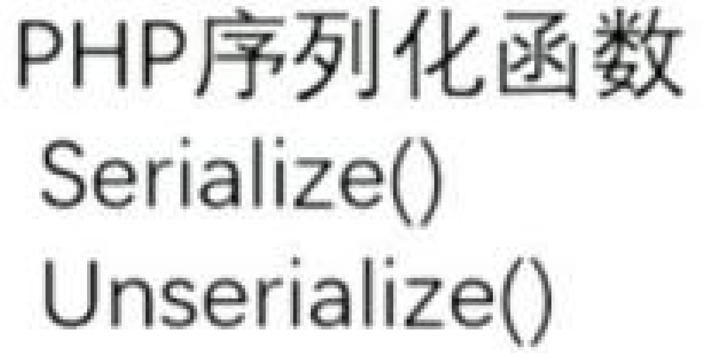


防御:

代码层面时间成本太大

建议从waf层面入手

反序列化攻击

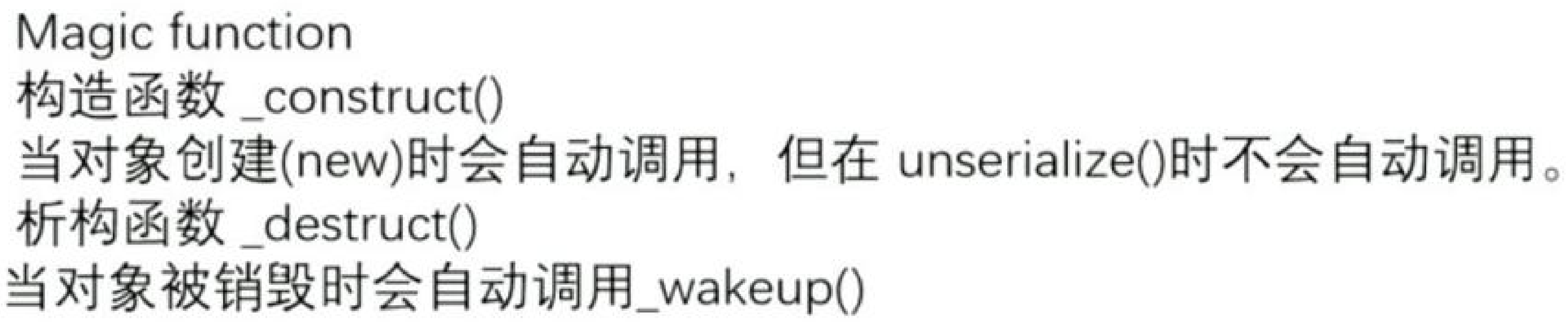


PHP序列化函数

Serialize()

Unserialize()

反序列化攻击



Magic function

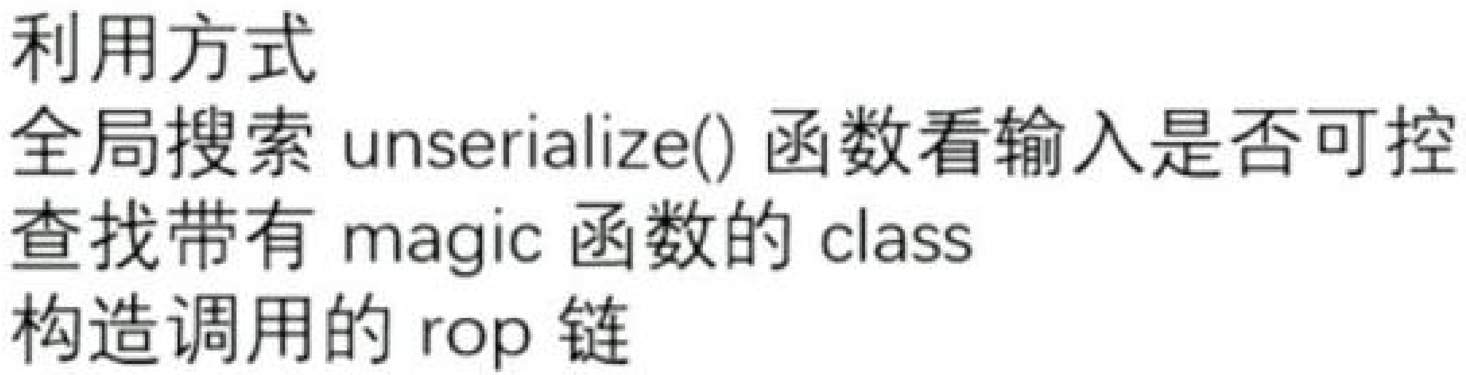
构造函数 _construct()

当对象创建(new)时会自动调用，但在 unserialize()时不会自动调用。

析构函数 _destruct()

当对象被销毁时会自动调用_wakeup()

反序列化攻击



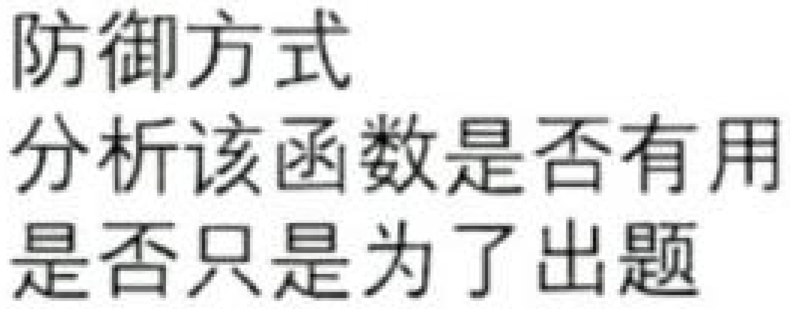
利用方式

全局搜索 unserialize() 函数看输入是否可控

查找带有 magic 函数的 class

构造调用的 rop 链

反序列化攻击



防御方式
分析该函数是否有用
是否只是为了出题

谢谢观赏