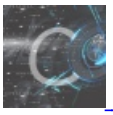
【PHP代码审计】基础入门漏洞

原创

3Ss安全前线 于 2021-06-15 15:01:24 发布 57 收藏 1

分类专栏： PHP代码审计 文章标签： 渗透测试

本文链接：https://blog.csdn.net/weixin_44978149/article/details/117921551



文章目录

PHP安全相关配置

php一些安全相关的配置如下：

配置名称	配置说明
register_globals	全局变量注册开关，为on时会将用户提交的参数注册成全局变量
allow_url_include	远程文件包含，为on时可直接包含远程文件
magic_quotes_gps	过滤相关，为on时会在get、post、cookie变量的特殊符号加入反斜杠，但不会过滤$_SERVER变量
magic_quotes_runtime	过滤相关，会对数据库或者文件中获取的数据进行过滤，特殊字符前加反斜杠

配置名称	配置说明
magic_quotes_sybase	过滤相关，为on时会覆盖magic_quotes_gps配置，转义空字符，把'转换成"
safe_mode	为on时所有文件操作函数都会受到限制例如include()、unlink()等，执行命令函数会提示错误例如system()、popen()
open_basedir	用来限制php只能访问哪些目录
disable_functions	禁止敏感函数的使用，把dl()函数也要加载到禁止列表，因为攻击者可以用dl()函数来加载自定义php扩展图片disable_functions指令的限制

用户可控变量

$_SERVER
$_GET
$_POST
$_COOKIE
$_REQUEST
$_FILES
$_ENV
$_HTTP_COOKIE_VARS
$_HTTP_ENV_VARS
$_HTTP_GET_VARS
$_HTTP_POST_FILES
$_HTTP_POST_VARS
$_HTTP_SERVER_VARS

常见漏洞

SQL注入

SQL注入成因就是在一些调用数据库增删改查的操作中没有对用户输入进行过滤，导致用户可以构造恶意语句拼接到这些操作中从而导致可控的SQL语句被执行。

风险函数

insert、delete、update、select

漏洞利用

漏洞代码如下：

```php
<?php
$con = mysql_connect("localhost","root","root");
if (!$con){die('Could not connect: ' . mysql_error());}
mysql_select_db("test", $con);
$id = stripcslashes($_REQUEST[ 'id' ]);
$query = "SELECT * FROM users WHERE id = $id ";
$result = mysql_query($query)or die('<pre>'.mysql_error().'</pre>');
while($row = mysql_fetch_array($result))
{
 echo $row['0'] . " " . $row['1'];
 echo "<br />";
}
echo "<br/>";
echo $query;
mysql_close($con);
?>
```

从上述代码中可以看到此处是我们可控的变量且此处的变量会拼接到数据库语句中，并且没有任何过滤

```php
$id = stripcslashes($_REQUEST[ 'id' ]);
$query = "SELECT * FROM users WHERE id = $id ";
```

从而可在此文件下构造判断是否存在SQL注入的语句:
http://127.0.0.1/sql-injection.php?id=1 and 1=1
http://127.0.0.1/sql-injection.php?id=1 and 1=2
判断可能存在SQL注入后使用order by之类的判断字段数，union select查询表名、列名等方法，建议使用SQLMAP一把梭，此处不多赘述。

防御方法

对用户输入的特殊字符进行严格过滤
使用参数化查询（PreparedStatement），避免将未经过滤的输入直接拼接到SQL查询语句中
Web应用中用于连接数据库的用户与数据库的系统管理员用户的权限有严格的区分（如不能执行drop等），并设置Web应用中用于连接数据库的用户不允许操作其他数据库。
设置Web应用中用于连接数据库的用户对Web目录不允许有写权限

文件包含漏洞

文件包含漏洞产生于程序需要包含某指定文件，这个过程作为用户可控且没有过滤便产生了文件包含漏洞，可以用于包含图片马、包含源码、远程包含webshell等操作。

风险函数

include、include_once、require、require_once、 show_source、highlight_file、readfile、file_get_contents、fopen、file

漏洞利用

windows常用测试包含文件：

C:/boot.ini //查看系统版本
C:/Windows/System32/inetsrv/MetaBase.xml //IIS配置文件
C:/Windows/repair/sam //存储系统初次安装的密码
C:/Program Files/my/sqlmy.ini //Mysql配置
C:/Program Files/mysql/data/mysql/user.MYD //Mysql root
C:/Windows/php.ini //php配置信息
C:/Windows/my.ini //Mysql配置信息
C:/Windows/win.ini //Windows系统的一个基本系统配置文件

linux常用测试包含文件：
/root/.ssh/id_rsa
/root/.ssh/id_ras.keystore
/root/.ssh/known_hosts //记录每个访问计算机用户的公钥
/etc/passwd
/etc/shadow
/etc/my.cnf //mysql配置文件
/etc/httpd/conf/httpd.conf //apache配置文件
/root/.bash_history //用户历史命令记录文件
/root/.mysql_history //mysql历史命令记录文件
/proc/mounts //记录系统挂载设备
/porc/config.gz //内核配置文件
/var/lib/mlocate/mlocate.db //全文件路径
/porc/self/cmdline //当前进程的cmdline参数

漏洞代码：

```php
<?php
include($_GET['file'].".php");
?>
```

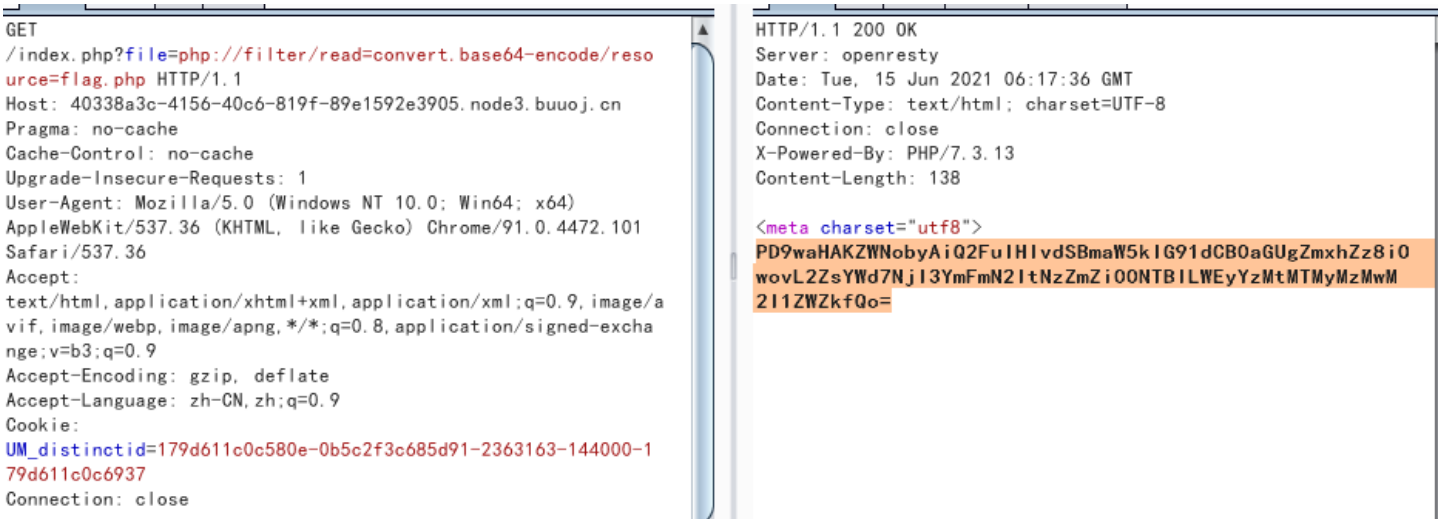
此处便可以构造一个值为file的get请求来包含传递的参数拼接后面的.php，包含一个文件
[ACTF2020 新生赛]Include 1 案例演示：
打开题目进入首页

tips

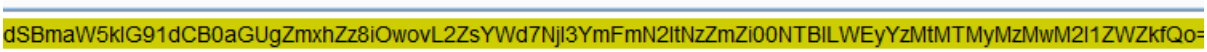
查看源代码发现点击tips会包含flag.php

包含后如下图，基本可以猜到要读这个flag.php

使用伪协议读源码：php://filter/read=convert.base64-encode/resource=flag.php



进行base64解密读出flag



防御方法

针对文件包含漏洞，建议采用“白名单”的方式，限制允许包含的文件范围。

命令执行

在一些需要调用系统命令的函数中没有进行过滤和限制导致可以用&、|等符号拼接命令进行执行反弹shell等操作。

风险函数

system、exec、passthru、``、shell_exec、popen、proc_open、pcntl_exec

漏洞利用

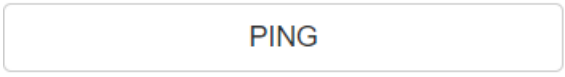
漏洞代码如下：

```php
<?php
$target=$_REQUEST['ip'];
$cmd = shell_exec('ping '.$target);
echo "<pre>{$cmd}</pre>";
?>
```

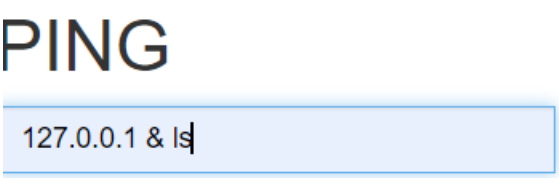
可以看到我们通过$_REQUEST方法提交的ip参数被带入shell_exec函数进行执行，后面可以直接接&、|进行命令执行。

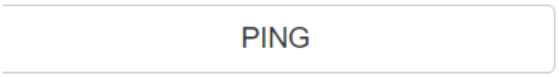
[ACTF2020 新生赛]Exec 1案例演示：

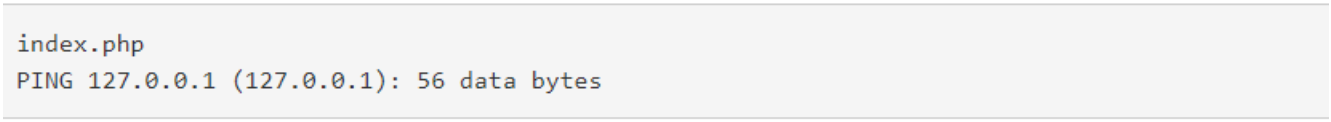
访问该题目



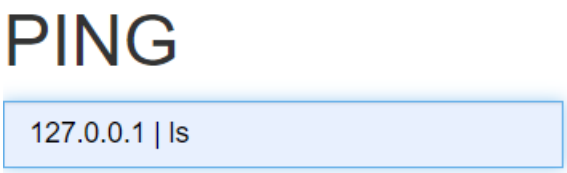
可以发现是一个执行ping命令的功能，尝试拼接ls进行执行命令

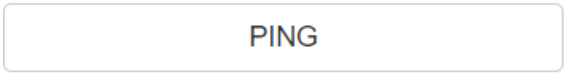


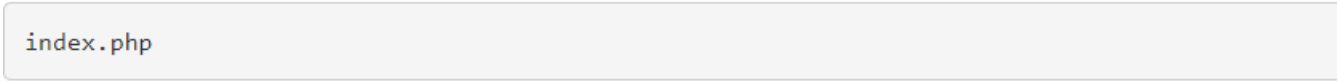




也可以尝试&&、|、||等查看具体区别







防御方法

黑名单：过滤特殊字符或替换字符
白名单：只允许特殊输入的类型/长度