CTF PHP代码审计 正则表达式 弱类型比较

原创

baynk 于 2020-03-22 18:38:06 发布 673 收藏 1

分类专栏： # HustWhCTF Writeup 文章标签： 正则表达式 弱类型比较 PHP代码审计

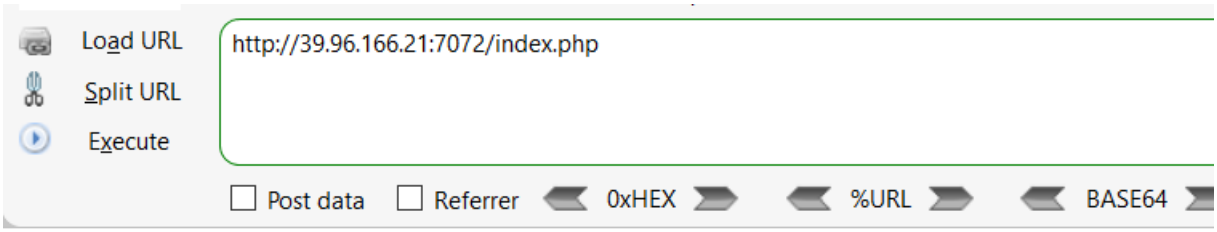
本文链接：https://blog.csdn.net/u014029795/article/details/105032422

HustWhCTF Writeup 专栏收录该内容

9 篇文章 0 订阅

订阅专栏

一进来就是PHP正则表达式




```php
<?php
highlight_file(__FILE__);
include 'flag.php';
if  ("POST" == $_SERVER['REQUEST_METHOD'])
{
    $password = $_POST['password'];
    if (0 >= preg_match('/^[[:graph:]]{12,}$/', $password))
    {
        echo 'Wrong Format';
        exit;
    }
    while (TRUE)
    {
        $reg = '/([[:punct:]]+|[[:digit:]]+|[[:upper:]]+|[[:lower:]]+)/';
     if (6 > preg_match_all($reg, $password, $arr))
            break;
        $c = 0;
        $ps = array('punct', 'digit', 'upper', 'lower');
        foreach ($ps as $pt)
        {
            if (preg_match("/[[:$pt:]]+/", $password))
                $c += 1;
        }
        if ($c < 3) break;
       if ("42"==intval($password)&&"42"!=$password) echo $flag;
        else echo 'Wrong password';
        exit;
    }
```

```
}
```


这个比较难的地方在于正则表达式里面的符号代表什么含义，网上去找了下资料。

POSIX	Description	ASCII	Unicode	Shorthand	Java
[:alnum:]	Alphanumeric characters	[a-zA-Z0-9]	[\p{L&}\p{Nd}]		\p{Alnum}
[:alpha:]	Alphabetic characters	[a-zA-Z]	\p{L&}		\p{Alpha}
[:ascii:]	ASCII characters	[\x00-\x7F]	\p{InBasicLatin}		\p{ASCII}
[:blank:]	Space and tab	[\t]	[\p{Zs}\t]	\h	\p{Blank}
[:cntrl:]	Control characters	[\x00-\x1F\x7F]	\p{Cc}		\p{Cntrl}
[:digit:]	Digits	[0-9]	\p{Nd}	\d	\p{Digit}
[:graph:]	Visible characters (i.e. anything except spaces, control characters, etc.)	[\x21-\x7E]	[^\p{Z}\p{C}]		\p{Graph}
[:lower:]	Lowercase letters	[a-z]	\p{Ll}		\p{Lower}
[:print:]	Visible characters and spaces (i.e. anything except control characters, etc.)	[\x20-\x7E]	\P{C}		\p{Print}
[:punct:]	Punctuation and symbols.	[!"#$%&'()*+, \-./:;<=>?@ [\\\]^_`{|}~]	[\p{P}\p{S}]		\p{Punct}
[:space:]	All whitespace characters, including line breaks	[\t\r\n\v\f]	[\p{Z}\t\r\n\v\f]	\s	\p{Space}
[:upper:]	Uppercase letters	[A-Z]	\p{Lu}		\p{Upper}
[:word:]	Word characters (letters, numbers and underscores)	[A-Za-z0-9_]	[\p{L}\p{N}\p{Pc}]	\w	
[:xdigit:]	Hexadecimal digits	[A-Fa-f0-9]	[A-Fa-f0-9]		\p{XDigit}

在有了上面那个的基础后就比较容易了，分段来说明。

```
if  ("POST" == $_SERVER['REQUEST_METHOD'])
```

需要用`POST`方式提交。

```
if (0 >= preg_match('/^[[:graph:]]{12,}$/', $password))
    {
        echo 'Wrong Format';
        exit;
    }
```

这里只需要有连续`12`个以上的非控制字符就可以，比如`空格`，`tab`等。

```
$reg = '/([[:punct:]]+|[[:digit:]]+|[[:upper:]]+|[[:lower:]]+)/';
     if (6 > preg_match_all($reg, $password, $arr))
            break;
```

这里有点难理解，这里指的是将`$password`分成连续的`符号punct`或者`数字digit`或者`大写upper`或者`小写lower`能分成6段以上即可。

PHP preg_match_all() 函数

PHP 正则表达式(PCRE)

preg_match_all 函数用于执行一个全局正则表达式匹配。

语法

```
int preg_match_all ( string $pattern , string $subject [, array &$matches [, int $flags = PREG_PATTERN_ORDER [, int $offs
et = 0 ]]] )
```

搜索 subject 中所有匹配 pattern 给定正则表达式的匹配结果并且将它们以 flag 指定顺序输出到 matches 中。

在第一个匹配找到后, 子序列继续从最后一次匹配位置搜索。

参数说明:

- $pattern: 要搜索的模式，字符串形式。
- $subject: 输入字符串。
- $matches: 多维数组，作为输出参数输出所有匹配结果, 数组排序通过flags指定。
- $flags：可以结合下面标记使用(注意不能同时使用PREG_PATTERN_ORDER和 PREG_SET_ORDER):
    1. PREG_PATTERN_ORDER: 结果排序为$matches[0]保存完整模式的所有匹配, $matches[1] 保存第一个子组的所有匹配，以此类推。
    2. PREG_SET_ORDER: 结果排序为$matches[0]包含第一次匹配得到的所有匹配(包含子组)， $matches[1]是包含第二次匹配到的所有匹配(包含子组)的数组，以此类推。
    3. PREG_OFFSET_CAPTURE: 如果这个标记被传递，每个发现的匹配返回时会增加它相对目标字符串的偏移量。
- offset: 通常， 查找时从目标字符串的开始位置开始。可选参数offset用于 从目标字符串中指定位置开始搜索(单位是字节)。

返回值

返回完整匹配次数（可能是0），或者如果发生错误返回FALSE。

这个函数也能分割出数组，只是有点小奇怪的是`$arr`这个变量没看到，可能是在`flag.php`里面有吧。。。

接着看下一段。

```
$ps = array('punct', 'digit', 'upper', 'lower');
        foreach ($ps as $pt)
        {
            if (preg_match("/[[:$pt:]]+/", $password))
                $c += 1;
        }
        if ($c < 3) break;
```

在密码里面至少要看到`符号`，`数字`，`大写`，`小写`至少出现三种。

```
       if ("42"==intval($password)&&"42"!=$password) echo $flag;
        else echo 'Wrong password';
        exit;
```

这里就是典型的弱类型比较了，只要满足上面的要求即可，随便测试了下`42.Hell0World!`就可以了。

```php
<?php
$password = '42.Hell0World!';
if ("42"==intval($password)&&"42"!=$password) echo "ok";
?>
```

ok


```php
<?php
highlight_file(__FILE__);
include 'flag.php';
if  ("POST" == $_SERVER['REQUEST_METHOD'])
{
    $password = $_POST['password'];
    if (0 >= preg_match('/^[[:graph:]]{12,}$/', $password))
    {
        echo 'Wrong Format';
        exit;
    }
    while (TRUE)
    {
        $reg = '/([[:punct:]]+|[[:digit:]]+|[[:upper:]]+|[[:lower:]]+)/';
     if (6 > preg_match_all($reg, $password, $arr))
            break;
        $c = 0;
        $ps = array('punct', 'digit', 'upper', 'lower');
        foreach ($ps as $pt)
        {
            if (preg_match("/[[:$pt:]]+/", $password))
                $c += 1;
        }
        if ($c < 3) break;
       if ("42"==intval($password)&&"42"!=$password) echo $flag;
        else echo 'Wrong password';
        exit;
    }
}
```

WHCTF{Regular_ weak_type_matching_bypass}

成功拿到`flag`，这里多说一下，不是非得用`科学计数法`才行，我看所有的`writeup`里面都写的`42.0000000e+1`这种。。。