# OWASP Juice Shop 三星难度 writeup

原创

neversec 于 2018-12-13 11:31:19 发布 859 收藏

分类专栏： CTF 信息安全 文章标签： ctf 靶场 juice shop writeup 信息安全



本文链接：https://blog.csdn.net/github_35912913/article/details/84985033







## 前言

本writeup所有题目基于**OWASP Juice shop V7.0.2**，靶场更新较快，后续有新的题目会接着更新。本版本靶场三星难度题目如下：

- Blockchain Tier 1 区块链第一关
- Forged Feedback 伪造反馈
- Forgotten Sales Backup 被遗忘的销售备份
- Login Bender 登录Bender
- Login Jim 登录Jim
- Payback Time 投资回收期
- Product Tampering 商品篡改
- Reset Jim's Password 重置Jim的密码
- Upload Size & Upload Type
- XSS Tier 2 XSS第二关
- XSS Tier 3 XSS第三关
- XXE Tier 1 XXE第一关

点此看环境搭建和一星难度wp
点此看二星难度wp

## 三星难度

### Blockchain Tier 1 区块链第一关

> Learn about the Token Sale before its official announcement. 在官方宣布前了解一下Token销售

看到题目一怔，是跟区块链比特币有关呢，仔细看了几遍hint。。这个题目暂时搁置了一会儿。

后续在做其他题目的时候看到了 http://192.168.209.146:3001/dist/juice-shop.min.js 这个js文件，搜索`token`发现里面存在题干关键的`Token Sale`这个词。关键代码如下：

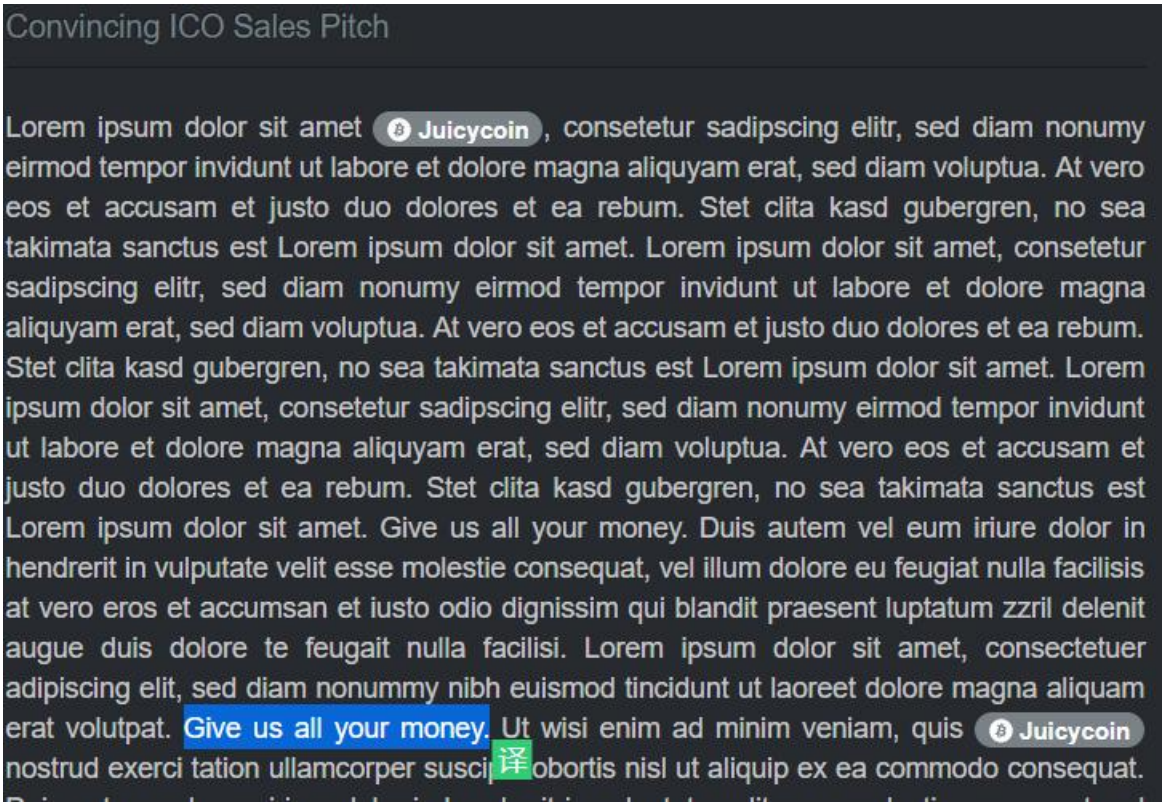
```
}), e.when("/access_token=:accessToken", {
    templateUrl: "views/OAuth.html",
    controller: "OAuthController"
}), e.when("/" +
function() {
    var e = Array.prototype.slice.call(arguments),
        t = e.shift();
    return e.reverse().map(function(e, n) {
        return String.fromCharCode(e - t - 45 - n)
    }).join("")
}(25, 184, 174, 179, 182, 186) + 36669..toString(36).toLowerCase() +
function() {
    var e = Array.prototype.slice.call(arguments),
        t = e.shift();
    return e.reverse().map(function(e, n) {
        return String.fromCharCode(e - t - 24 - n)
    }).join("")
}(13, 144, 87, 152, 139, 144, 83, 138) + 10..toString(36).toLowerCase(), {
    templateUrl: "views/TokenSale.html",
    controller: "TokenSaleController"
}), e.otherwise({
    redirectTo: "/search"
})
```

粘贴这段代码到控制台运行一波

```
> "/" + function() {
          var e = Array.prototype.slice.call(arguments)
            , n = e.shift();
          return e.reverse().map(function(e, t) {
              return String.fromCharCode(e - n - 45 - t)
          }).join("")
      }(25, 184, 174, 179, 182, 186) + 36669..toString(36).toLowerCase() + function()
          var e = Array.prototype.slice.call(arguments)
            , n = e.shift();
          return e.reverse().map(function(e, t) {
              return String.fromCharCode(e - n - 24 - t)
          }).join("")
      }(13, 144, 87, 152, 139, 144, 83, 138) + 10..toString(36).toLowerCase()
< "/tokensale-ico-ea"
```

访问这个地址就完成题目了，这道题的意思应该是掩饰了一下地址吧，没太懂

这个地址里面有一些关于区块链ICO白皮书之类的东西，看到这个调侃笑死我了哈哈哈




## Forged Feedback 伪造反馈

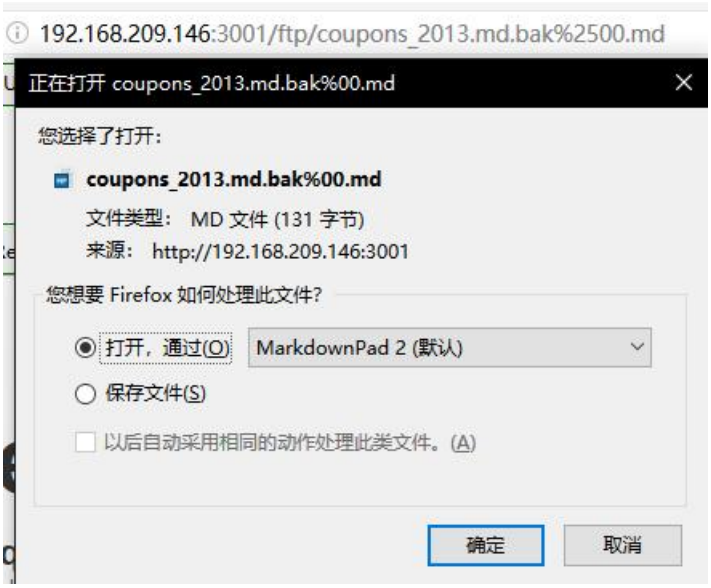
> Post some feedback in another users name. 用其他人的用户提交反馈

在http://192.168.209.146:3001/#/contact中提交评价的时候抓包改包，将`UserId`改成其它即可。

## Forgotten Sales Backup 被遗忘的销售备份

> Access a salesman's forgotten backup file. 进入到售货员遗忘的备份文件

在第一关的`Confidential Document 机密文件`题目中找到了一个ftp目录，并且该目录限定了只能查看 md 和 pdf 后缀的文件，在这里可以使用%00截断判断下载文件。其他文件同理。




## Login Bender 登录Bender

> Log in with Bender's user account.

利用SQL注入即可，payload 为`email":"' or 1=1 limit 5,1 --","password":"123`，修改limit的序列即可

## Login Jim 登录Jim

> Log in with Jim's user account.

同上

## Payback Time 投资回收期

> Place an order that makes you rich. 下一个令你富有的订单。

商品加车漏洞。添加购物车的时候抓包，修改POST数据`{"ProductId":8,"BasketId":"10","quantity":1}`中的quantity为负数，加车后结账金额为负数。

Customer: mc.safesearch@juice-sh.op

Order #: e717403a57081ef60abfc4adf59ab088

3x Apple Juice (1000ml) ea. 1.99 = 5.97

1x Banana Juice (1000ml) ea. 1.99 = 1.99

-1000x OWASP Juice Shop CTF Girlie-Shirt ea. 22.49 = -22490

1x Carrot Juice (1000ml) ea. 2.99 = 2.99

Total Price: -22479.05

## Product Tampering 商品篡改

> Change the href of the link within the OWASP SSL Advanced Forensic Tool (O-Saft) product description into http://kimminich.de. 将owasp ssl高级取证工具（o-saft）产品描述中的链接的href更改为http://kimminich.de。

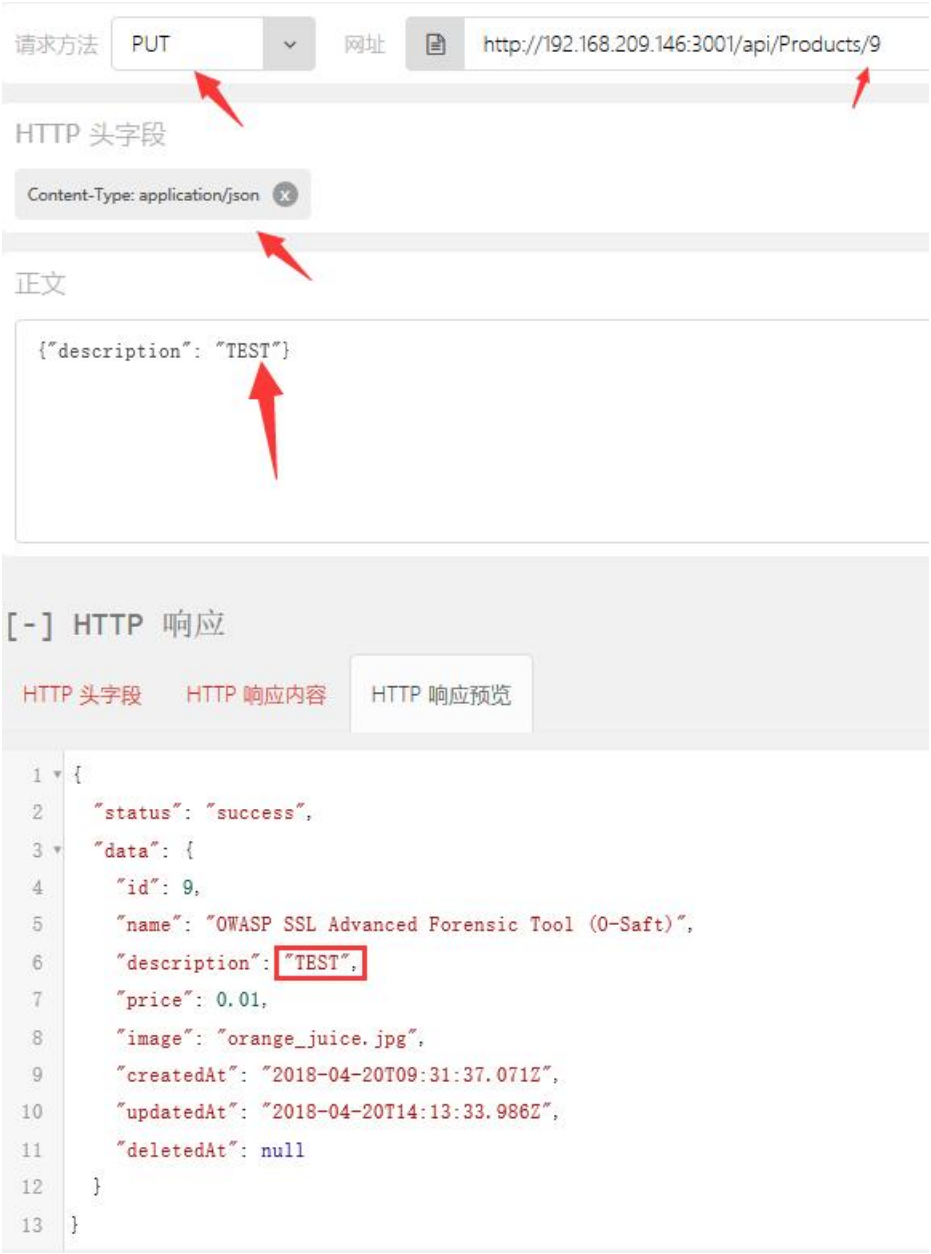
要改产品描述。访问首页，点击一个商品，在Chrome里面选择XHR请求，有几个个API接口，其中有一个包含描述信息的链接如下

http://192.168.209.146:3001/api/Products/1?d=Fri Apr 20 2018 d后面跟的是时间参数，不加的话好像是返回最新修改日期



```
{
  "status": "success",
  "data": {
    "id": 1,
    "name": "Apple Juice (1000ml)",
    "description": "The all-time classic.",
    "price": 1.99,
    "image": "apple_juice.jpg",
    "createdAt": "2018-04-20T09:31:37.068Z",
    "updatedAt": "2018-04-20T09:31:37.068Z",
    "deletedAt": null
  }
}
```


如果API接口使用了PUT或者DELETE这些危险的HTTP方法的话，数据就很容易被修改。这里试了一下PUT请求，结果是可以搞定的。使用火狐插件 RESTclient，记得加HTTP头指定类型为 json，注意要使用 `{"description": "TEST"}` 而不能是 `{"data": {"description": "TEST"}}`




所以最后的payload为`{"description": "<a href=\"http://kimminich.de\"></a>"}`

## Reset Jim's Password 重置Jim的密码

> Reset Jim's password via the Forgot Password mechanism with the original answer to his security question. 通过忘记密码来重置Jim的密码，要求回答他的安全问题

首先通过登陆的SQL注入`{"email":"' or 1=1 limit 1,1 --","password":"123"}`搞到邮箱号， jim@juice-sh.op。问的安全问题是`Your eldest siblings middle name?`，这道题没法儿社工啊。

这道题懵逼了，查了别人的解题说是美国的社工？提到JIM会想起 *James T. Kirk* ？反正我搜JIM只能搜到 *Jim Parsons* ( *Sheldon* 扮演者233)

所以正确的解答路径是

1. 想到 *James_T._Kirk*，到Wiki上搜他https://en.wikipedia.org/wiki/James_T._Kirk
2. 上面写着Jim有个Brother叫 *George Samuel Kirk*
3. 所以安全问题的答案就是 *Samuel*
4. 重置密码，收工。

## Upload Size & Upload Type

> Upload a file larger than 100 kB.Upload a file that has no .pdf extension. 上传一个文件大于100kb，上传一个文件不用pdf扩展名

两道题在一块儿写。文件上传跟前面说到的上传XML是同一个点，位于投诉页面。尝试提交，查看XHR请求，很明显有个接口 http://192.168.209.146:3001/file-upload 是用于文件上传的。

此处先用burp截断，点击上传，提示


burp处没有收到请求，说明是前端校验，因此用改包、写脚本等方法直接post到上传接口处即可。

扩展名绕过可以用`%2500`截断，也可以用如上的方法。

```
------WebKitFormBoundaryAwr09qjMg7ICDiX1
Content-Disposition: form-data; name="file"; filename="test.pdf%2500.js"
Content-Type: text/xml


------WebKitFormBoundaryAwr09qjMg7ICDiX1--
```

## XSS Tier 2 XSS第二关

> Perform a persisted XSS attack with bypassing a client-side security mechanism. 绕过客户端安全机制执行存储型 XSS攻击。

xss的输入点其实不多，但是最容易想到的是用户名，注册一个用户。改包修改邮箱为xss语句

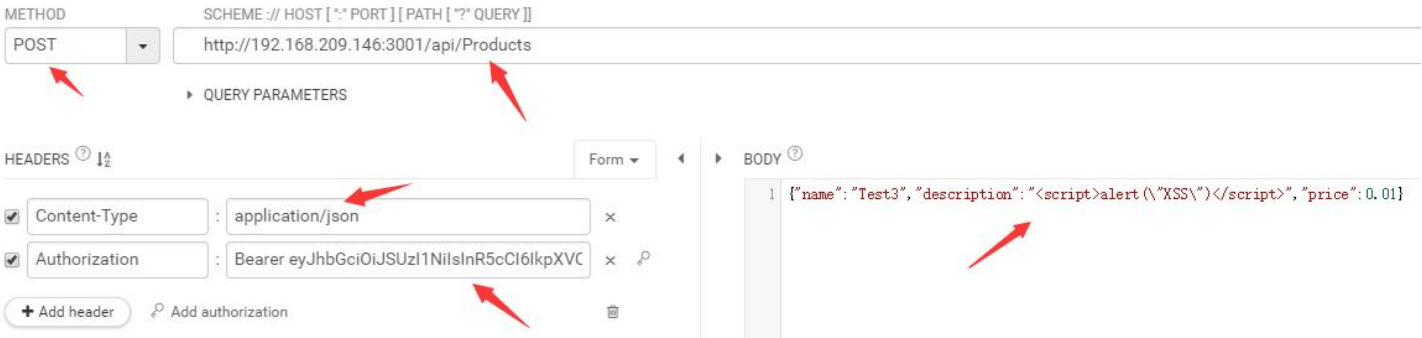
`php {"email":"<script>alert(\"xss3\")</script>","password":"123123","passwordRepeat":"123123","securityQuestion":{"id":1,"question":"Your eldest siblings middle name?<script>alert(/xss/)<script>","createdAt":"2018-04-21T02:42:19.878Z","updatedAt":"2018-04-21T02:42:19.878Z"},"securityAnswer":"123123"}` 这个xss能在管理员页面弹出 http://192.168.209.146:3001/#/administration，商品评论绕过不了，无法xss。

## XSS Tier 3 XSS第三关

> Perform a persisted XSS attack with without using the frontend application at all. 执行存储型 XSS攻击，根本不使用前端应用程序。

前文我们说道发现了一个API接口http://192.168.209.146:3001/api/Products/9，并且这里可以修改描述信息，这里存在一个存储型的xss。使用 payload `{"description":"<script>alert(\"xss\")</script>"}` 就阔以了，然而计分板没有显示我完成xss。

后来发现 api/Products 接口可以POST，这里POST一个新的商品也是可以XSS攻击的。注意需要用管理员的 Authorization 进行验证，否则会报401错误。




Response

**201 Created**


## XXE Tier 1 XXE第一关

> Retrieve the content of C:\Windows\system.ini or /etc/passwd from the server. 用XXE攻击搞到服务器的/etc/passwd

构造xml文件如下

```
<?xml version="1.0" encoding="utf-8"?>

<!ENTITY xxe SYSTEM "file:///etc/passwd" >]>
<root>
<name>&xxe;</name>
</root>
```上传到投诉页面的上传点就过了，不过暂时不知道输出点哪儿
**update**: 使用docker搭建时，XXE的关卡都不可用