DOM 型 XSS 攻击实战

1. Low 级别 DOM 型 XSS 攻击实战

步骤 1:设置安全级别为 Low,点击 XSS(DOM)按钮,进入 DOM 型 XSS 攻击页面。发现是个选择框,随便选择一个选项,提交发现选择的参数会携带在 URL 中,说明页面提交方式为 GET,如图 1.1。

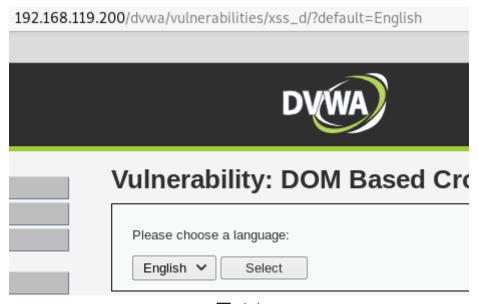


图 1.1

步骤 2: 在选择框附近点击右键选择 查看网页源代码 ,发现选择框中的内容使用了 document.write 的方式来输出,说明页面的 XSS 方式为 DOM型,如图 1.2。

```
/: DOM Basec × http://192.168.119.200/dvwa ×
             i view-source:http://192.168.119.200/dvwa/vulnerabilities/xss_d/?default=English
l 🌜 Getting Started
 Please choose a language:
 <form name="XSS" method="GET">
  <select name="default">
    <script>
      if (document.location.href.indexOf("default=") >= 0) {
        var lang = document.location.href.substring(document.location.hr
       document.write("<option value='" + lang + "'>" + decodeURI(lang) + '
       document.write("<option value='' disabled='disabled'>----</option
      }
      document.write("<option value='English'>English</option>");
      document.write("<option value='French'>French</option>");
      document.write("<option value='Spanish'>Spanish</option>");
      document.write("<option value='German'>German</option>");
    </script>
```

图 1.2

步骤 3:在 URL 后直接加入攻击脚本 <script>alert(document.cookie)</script>,发现可以直接弹窗,如图 1.3。

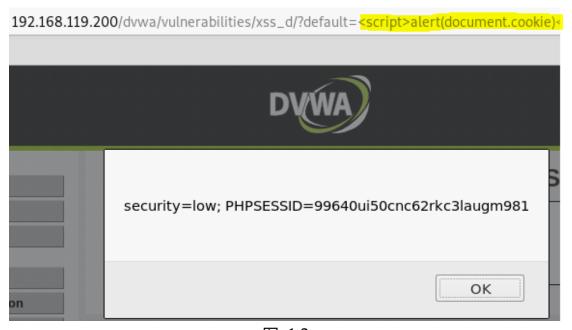
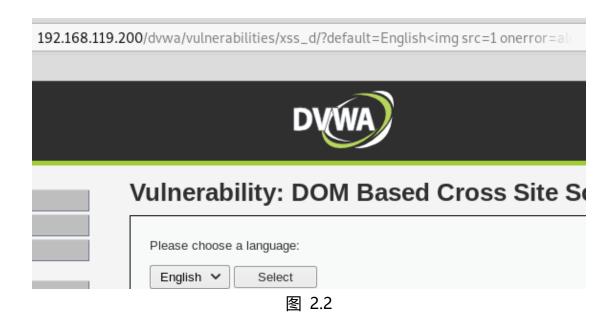


图 1.3

2. Medium 级别 DOM 型 XSS 攻击实战

步骤 1:设置安全级别为 Medium,进入 DOM 型 XSS 攻击页面。查看页面源码,发现存在代码,如果发现提交的内容中含有 <script,则替换为 English,如图 2.1,所以 <script> 标签无法使用。我们的思路与前面一样,使用 标签来进行攻击,在 URL 后加入 ,发现仍无法弹窗,如图 2.2。

图 2.1



步骤 2: 使用浏览器的开发者工具查看前端源码,发现脚本内容被输出到了 <option > 标签的 value 值中,导致无法执行,如图 2.3。



图 2.3

步骤 3: 根据上一步的发现,我们需要构造 </option> 标签和 </select> 标签来闭合语法,使脚本能够被执行,在 URL 后加入 ></option> </select> ,可以弹窗,如图 2.4。



图 2.4

3. High 级别 DOM 型 XSS 攻击实战

步骤 1:设置安全级别为 High,进入 DOM 型 XSS 攻击页面,查看页面源码,发现做了白名单过滤机制,提交的变量必须是 French、English、German、Spanish 中的一个,才允许提交,否则就会转换为空值,如图 3.1。

```
Damn Vulnerable Web Application (DVWA) v1.10 *Development*Source :: Damn Vulnerable Web Applica... 🍵 🏻 📵
                                                                                       ... ♥ ☆ 目
 ① 192.168.119.200/dvwa/vulnerabilities/view_source.php?id=xss_d&security=high
 <?php
 // Is there any input?
 if (array_key_exists("default", $_GET) && !is_null ($_GET['default'])) {
  #White list the allowable languages
  switch($_GET['default']){
   case "French":
    case "English":
    case "German":
    case "Spanish":
     #ok
     break:
    default:
     header("location:?default=English");
dmin
                                                                                     View Source View H
```

图 3.1

步骤 2: 考虑到 JS 是前端脚本,我们只需要在正常提交的值后用 # 把后面的攻击脚本注释掉,脚本内容就不会提交至服务器,也就可以通过白名单验证。但脚本仍然会在本地浏览器执行。在 URL 后加入如下脚本内容 #<script>alert(document.cookie)</script>,注意 # 前有一个空格,如图 3.2,可以弹窗。

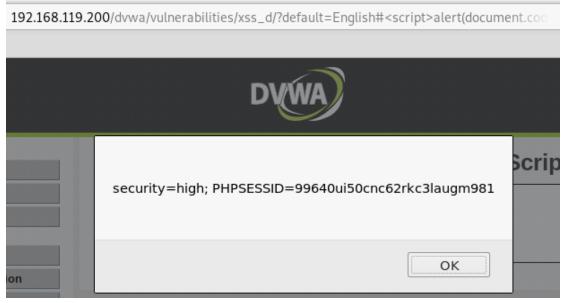


图 3.2