转自合天智汇：

作者如何利用 xss 漏洞 shua 盒子 rank 的

前言：

标题真不真，继续往下看，为什么要作者选择这个提交平台，一是作者真菜，二是好过吧，只要是存储 xss 就给过

本篇是作者的 xss 备忘录阶段小结，所以读者很有福，能得到大量的姿势（小菜姿势，哈哈哈）

现目前容易且主流的漏洞，作者认为有三种，一是框架漏洞，二是逻辑，三是 xss

ok，进入我们的主题

1.xss 基础知识和分类

2.xss 姿势绕过

3.xss 工具测试

4.xss 防御

5.总结

一.

好像没什么写的。。。

1.dom

2.反射

3.存储

这里推荐一本书《web 前端黑客技术揭秘》

作者这里写的常见基础知识，也是里面的笔记（过审了，再补吧，也写不出来多少）

二.

网上有很多姿势，作者先写一些自己印象深的，再拿出自己常用的

那些什么编码，各种事件替换，就不写了，推荐几篇文章，有兴趣可以自己阅读下

姿势：（xss 的攻击） https://www.cnblogs.com/black-humor/p/7810920.html
https://blog.csdn.net/qq_29277155/article/details/51320064

https://blog.csdn.net/kimqcn4/article/details/52813902 （理解实战）
http://netsecurity.51cto.com/art/201301/376873.htm

我们来挖小商城的 xss，因为商城的 xss 输入点多

第一步：

搜索框判断：

onfocus=alert1autofocus 那两个是反引号，实测可用，在不使用‘”( )的情况下，

但是一般都要闭合 value 的 "

“ onfocus=alert('1') autofocus ”

"onclick="alert(1)

"><!--

%22><!-- （有趣的现象，url 编码能绕过简单的反斜杠）

见招拆招了，一般都不会费尽心思过滤什么特殊的字符，都是直接函数过滤

真的过滤了，继续看

1.输入：

onclick onbblclick onmousedown onmouseup onmouseover onmousemove onmouseout onkeypress onkeydown onkeyup onabort onbeforeunload onerror onload onmove onresize onscroll onstop onupload onblur onchange onfocus onreset onsubmit onbounce onfinish onstart onbeforecopy onbeforecut onbeforeeditfocus onbeforepaste onbeforeupdate oncontextmenu oncopy oncut ondrag ondragdrop ondragend ondragenter ondragleave ondragover ondragstart ondrop onlosecapture onpaste onselect onselectstart onafterupdate oncellchange ondataavailable ondatasetchanged ondatasetcomplete onerrorupdate onrowenter onrowexit onrowsdelete onrowsinserted onafterprint onbeforeprint onfilterchange onhelp onpropertychange onreadystatechange

事件判断一波，博主两次遇到一个商城 cms，漏了两个事件 onuoload（页面关闭时触发），onbblclick(双击时触发)

2.常用的判断一波：

onfocus iframe src script onclick confirm document onmouseover javascript onsubmit onerror onload onscroll onstart onblur onhashchange embed action mario formaction background posters data code oncut onstart vbsonload expression location onfocusout onfocusin onmouseover style alert

这都是常用的字符

3.特殊字符判断一波

~!@#$%^&*()_+-={}|:"<>?[]\;',./

一般< > ' " 都过滤的了，有时也会漏下 ‘ ，作者也不是很清楚为什么

第二步：

如果这简单的搜索框都做的很好，那么自行考虑要不要继续挖掘

作者遇到一个，注册登陆后，它是一个选择题，都过滤了的，但正确选项那个答案没有过滤

"><!-- （资料填写处，收货地址处）

12"></textarea> <img src=x onerror=eval(atob(''))><!-- （留言处）

一般就这两个就够了

第三步

说这么多，来两个简单的实战吧（难的，自己也菜的抠脚）

第一个，反射：

http://www.xx.com (就不打码了，就一个简单的反射 xss)

输入

">

输入

%22><!--

第二个，存储 xss

作者找了另一个网站当例子

1.搜索框反射

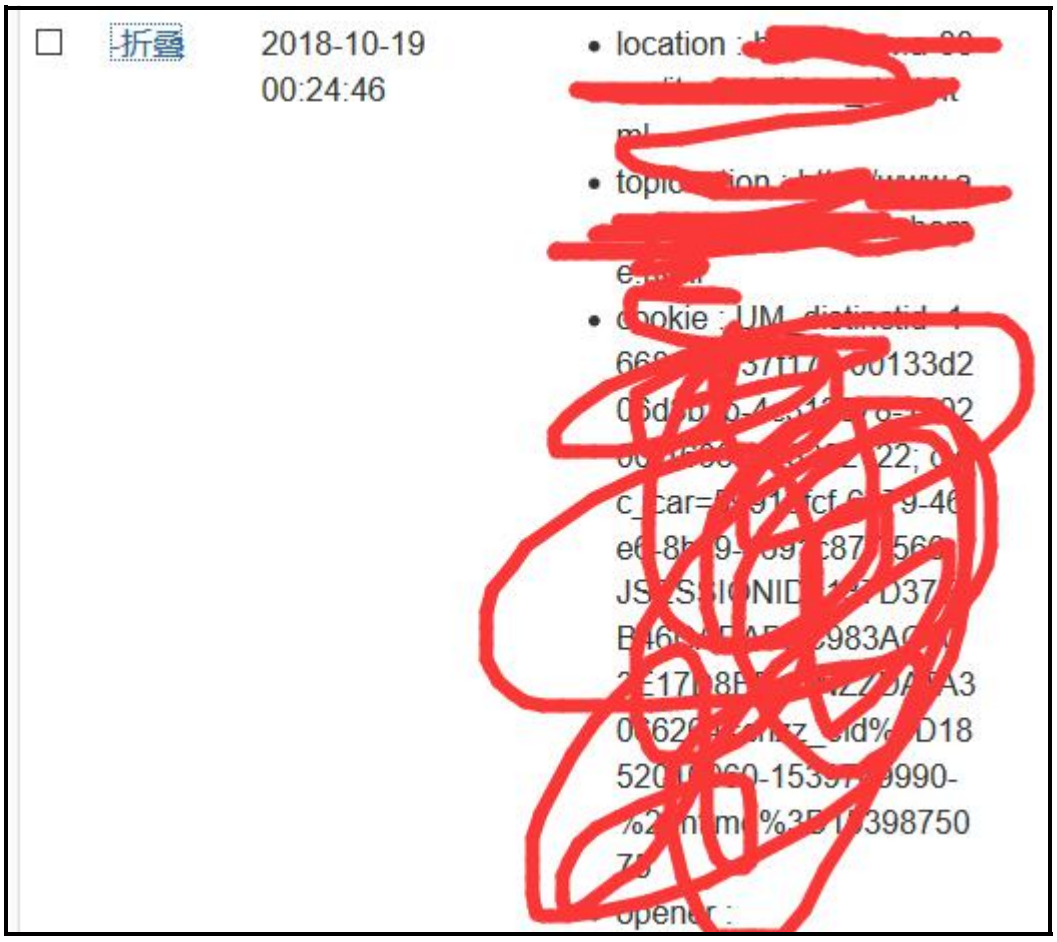
输入

"><!--

2.存在存储 xss

12"></textarea> <img src=x onerror=eval(atob(''))><!--

这里作者要说的是，这是我自己的cookie，到目前为止，还没有拿到管理的cookie



按着作者的方法，只要你有耐心和时间，rank 真的问题不大，有图有真相，好不好（作者也是刚上手的，只是小结下，想赚点稿费，毕竟这些小站的 xss，价值不大，还不如稿费）

三.xss 工具

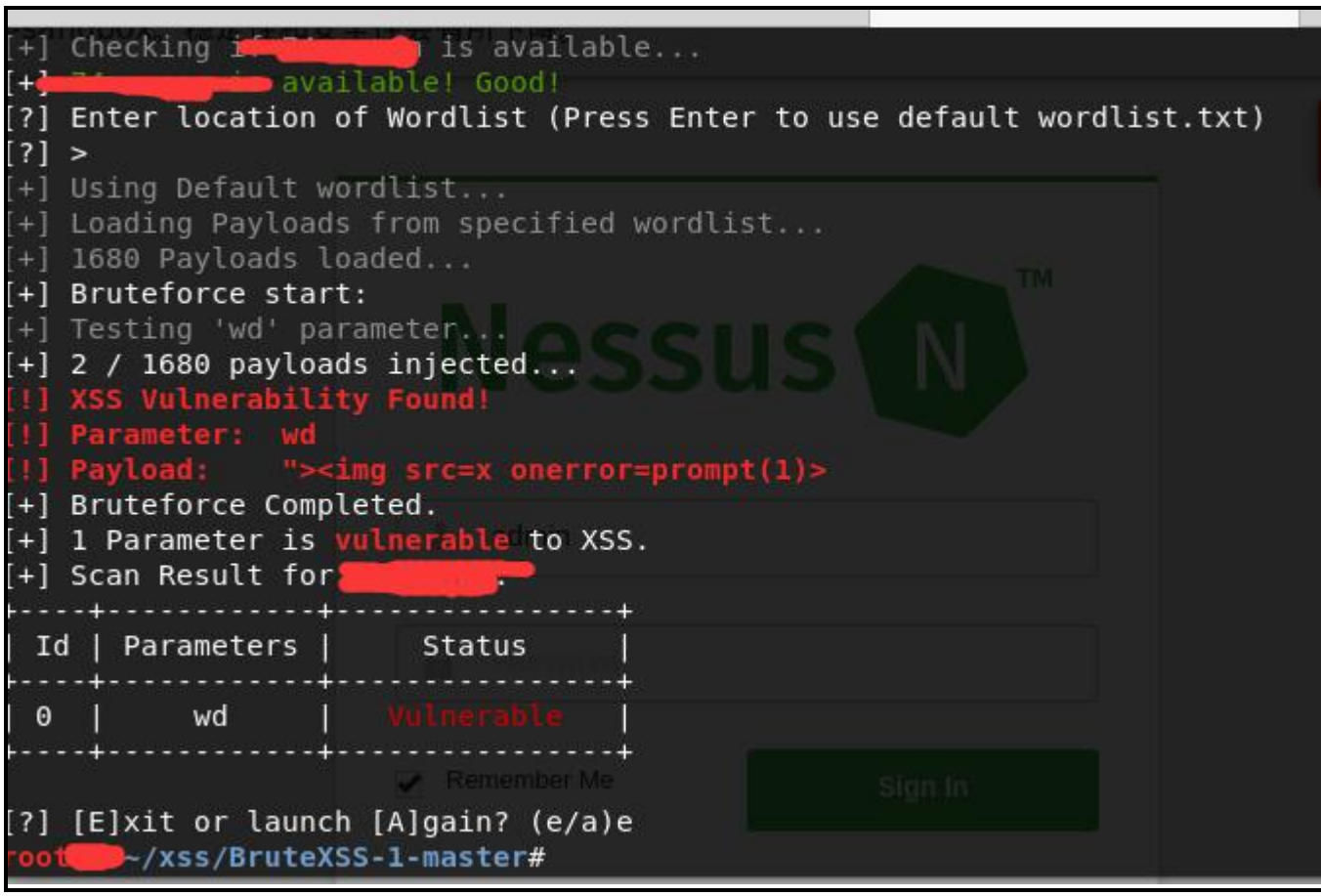
1.github 脚本

来自一位前辈的收集

- https://github.com/shawarkhanethicalhacker/BruteXSS （Cross-Site Scripting Bruteforcer）
-
-
- https://github.com/1N3/XSSTracer (A small python script to check for Cross-Site Tracing)
-
-
- https://github.com/0x584A/fuzzXssPHP (PHP 版本的反射型 xss 扫描)
-

- https://github.com/chuhades/xss_scan (批量扫描 xss 的 python 脚本）
-
- https://github.com/BlackHole1/autoFindXssAndCsrf (自动化检测页面是否存在 XSS 和 CSRF 漏洞的浏览器插件）
- 简单测试过，把其他的脚本 payload 转移到了 BruteXSS 里，其他的作者用起来效果不是很好（可能没有找到使用方法吧）

  就是上文的 prompt （有点小瑕疵就是，误报，误报后就停止了，还在思考解决）



2.

xsser，kali 自带的一款工具，由于作者没有研究过，就不扩展讨论了

四.xss 防御

作者的浅薄见解

很粗暴的方法：输入实体化，敏感词过滤

作者见过一个 cms 的过滤是这样的，on 开头过滤掉（现在想来应该没有过滤完），script，src，img，iframe，ifame 等敏感可引入词过滤，再加上输入实体化

五.总结

1.理解 xss 姿势的来源

2.了解现如今的 xss 常用方法

3.了解现在的 xss 防御机制

2018.10.19

----------by k----------