CSRF蠕虫存在的可能性分析

2018-7-21

CSRF蠕虫顾名思义就是利用CSRF技术进行传播的Web蠕虫，前段时间我看见这篇文章《译言CSRF蠕虫分析》说明了CSRF蠕虫存在的事实，译言网站（以下称这样的宿主为victim\_site）的这个CSRF蠕虫是由用户驱动的，蠕虫的代码都存放于另外一个网站上（比如worm\_site），在victim\_site上需要用户驱动的就一个链接http://worm\_site/，该链接指向CSRF蠕虫本身。

CSRF蠕虫如何就是一个蠕虫？我们写的代码其实没有表现出一个蠕虫本身要具有的所有性质。在这，最关键要解决的就是CSRF蠕虫的传播性，基于用户驱动的传播性（主动或者被动）。当它可以传播了，我们就可以考虑为它增加其它的功能，比如利用CSRF技术删除、编辑某些内容、增加新的内容、添加好友等等。

一、CSRF蠕虫的传播

比如我们要在一个SNS网站中传播CSRF蠕虫的链接：http://worm\_site/，当用户点击这个链接时就会触发CSRF蠕虫，初始时我们的蠕虫链接http://worm\_site/被提交在http://victim\_site/user=37这个用户的留言板上，当某登录的用户A点击这个蠕虫链接时，worm\_site会判断点击的来源地址，根据来源的地址本身或者来源地址对应的页面内容中的用户唯一标志来区分出该留言板属于的用户的所有好友信息。接着筛选出所有好友的有价值的信息，比如这里就是http://victim\_site/user=[user\_id]这个链接中的[user\_id]信息。根据批量获取的这个唯一的[user\_id]，CSRF蠕虫就可以借用用户A的权限循环发起POST型的CSRF攻击，此时蠕虫提交的表单类似如下代码：

<form action="http://victim\_site/post\_info.do" method="post">

<input type="text" name="title" value="hi" style="display:none!important;display:block;width=0;height=0" />

<input type="text" name="info" value="有趣的网站：http://worm\_site/。" style="display:none!important;display:block;width=0;height=0" />

<input type="text" name="user\_id" value="[user\_id]" style="display:none!important;display:block;width=0;height=0" />

</form>

该表单中的[user\_id]是这个SNS网站进行用户身份标志的值，假如我们的CSRF蠕虫不能获取这个值就无法正常传播。从这个例子中，我们可以看出蠕虫要传播就必须能够获取那些唯一值，然后利用获取到的唯一值进行传播。在一个SNS网站中，什么值是唯一的？比如用户id、用户昵称、用户email、用户session、用户个人页面地址。那么什么是CSRF蠕虫可以得到的？用户session仅通过CSRF显然得不到，其它的都可以得到。获取这些唯一值的意义是什么？CSRF蠕虫必须知道自己正在处理的是谁的信息，比如[user\_id]为37的好友列表页面地址：http://victim\_site/friends/user=37，这是唯一的，只有知道这样的唯一值，才能知道该唯一值对应的其它唯一信息。上面的表单代码我们在worm\_site打包为一个函数：

function post\_info(user\_id){

var id = user\_id;

create\_form with the id;

}

该函数通过唯一的[user\_id]动态生成并提交POST型的CSRF攻击表单。被攻击的目标就是[user\_id]对应的用户页面，CSRF蠕虫就是通过这样的方式传播开。

二、跨域获取数据的几种方式

上面提到的CSRF蠕虫传播必须面对的问题是如何获取各种必要的唯一值。这里有三种方式：服务端代理技术、Flash AS跨域请求技术、JSON HiJacking技术。

1、服务端代理技术：

译言CSRF蠕虫使用的就是这样的技术，利用服务端脚本获取到的referer值来判断来源地址，由于该referer值也许包含SNS网站中用户的唯一标志，比如译言的个人空间链接地址http://www.yeeyan.com/space/show/19076，19076就是该用户的唯一标志。蠕虫在服务端就可以根据这个唯一标志区分自己将要处理的数据，比如获取19076用户的好友信息，就可以操作这个地址http://www.yeeyan.com/space/friends/19076。

使用服务端代理技术优点是显而易见的，比如蠕虫代码、逻辑可以被很好的隐藏。缺点是在服务端发起的GET或POST请求无法跨域带上被攻击站点的本地Cookie或内存Cookie。这样CSRF蠕虫就只能通过referer里的唯一值来进行下一步攻击，而不能通过获取referer地址对应的页面内容里的由Cookie决定的唯一值，比如这样的地址http://www.yeeyan.com/space/showme，对每个登录的译言用户而言，链接地址一样，但是页面内容不一样，里面不一样的内容由用户的身份标志决定。使用服务端代理技术无法通过CSRF技术获取http://www.yeeyan.com/space/showme链接页面里不一样的用户唯一标志。

2、Flash AS跨域请求技术：

目标服务器下必须存在crossdomain.xml文件，且crossdomain.xml中的配置允许其他域的AS脚本进行跨域请求，如下：

<?xml version="1.0"?>

<cross-domain-policy>

<allow-access-from domain="\*" />

</cross-domain-policy>

那么worm\_site就可以使用AS脚本来发起跨域的GET请求，这是客户端发起的CSRF攻击，在请求时会带上本地Cookie或者内存Cookie，所以可以很方便地获取我们想要的页面内的唯一值。结合AS与服务端的通信技术、AS与JS的通信技术，CSRF蠕虫将更加的强大。关于详细的利用AS进行CSRF攻击的文章可以看这《CSRF with Flash》

3、JSON HiJacking技术：

我在这《JSON Hijacking的利用以及Web API安全》提到了如何利用JSON HiJacking获取用户隐私的方法，JSON HiJacking其实就是一个CSRF过程。由于<script>标签的跨域性，通过JSON HiJacking就有可能跨域获取目标服务器返回的JSON数据。而这些数据中就非常可能包含我们需要的唯一值。比如这个链接http://api.fanfou.com/private\_messages/inbox.json?callback=hijack&count=2，链接的内容如下：

hijack([{"id":1113357,"text":"第二条私信~","sender\_id":"evilcos","recipient\_id":"ycosxhack","created\_at":"Sun Nov 09 10:24:44 +0000 2008","sender\_screen\_name":"evilcos","recipient\_screen\_name":"余弦"}, {"id":1113354,"text":"第一条私信~","sender\_id":"evilcos","recipient\_id":"ycosxhack","created\_at":"Sun Nov 09 10:24:25 +0000 2008","sender\_screen\_name":"evilcos","recipient\_screen\_name":"余弦"}])

这段JSON内容中包含了我们需要的唯一值：ycosxhack、evilcos等。80sec的百度空间CSRF蠕虫使用了这个技术。

三、结论

除了上面介绍的三种跨域获取数据的方法，还有其他方法。在不同的Web2.0环境下，CSRF蠕虫细节不一样，不过各类原理是一样的。通过对 CSRF蠕虫传播原理的分析，我个人觉得许多广泛存在CSRF漏洞的Web2.0网站都面临着CSRF蠕虫的威胁，虽然在真实环境下编写CSRF蠕虫，现实往往是残酷的，不会那么顺利:(。Web2.0蠕虫由用户驱动，被动的、主动的，加上些社工技巧，这将很难防御。这篇文章分析的CSRF蠕虫是指站外 CSRF蠕虫，如果在站内，同域环境下且利用XSS技术，CSRF蠕虫就不再是单纯的CSRF了。