2012

第一届
全国网络与信息安全防护峰会

对话·交流·合作

CONVERSATION · COMMUNICATION · COOPERATION

XDef 2012
对话·交流·合作

OAUTH SECURITY

张天琪(pnig0s_小P)
知道创宇

About:Me

对话·交流·合作

张天琪 ID:pnig0s(小P)

Security Researcher @ KNOWNSEC 知道创宇

Core Member @ FREEBUF.COM WHERE THERE IS SHELL .THERE IS A WAY 关注黑客与极客

xKungFoo2012 Speaker

Focus On:

Web安全研究

核心产品后台引擎研发

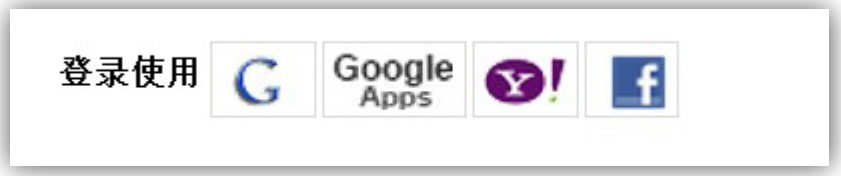
Fond Of:

渗透测试

大数据分析处理

Who Use OAuth?

对话·交流·合作



这仅仅是冰山一角，OAuth授权机制目前在国内得到了相当广泛的应用，包括各大电商，SNS交友，连锁酒店，招聘，旅游，微博，邮件系统等各类网站。

一，OAUTH那点事儿

OAuth三两言

OAuth(开放授权)是一种授权(Authorisation)协议而非认证协议(Authentication)

OAuth协议最大的进步是能够使第三方在不用获得目标网站账密的情况下使用目标网站的用户资源

随着开放式REST风格API的广泛使用，使得OAuth协议被应用的越来越广泛

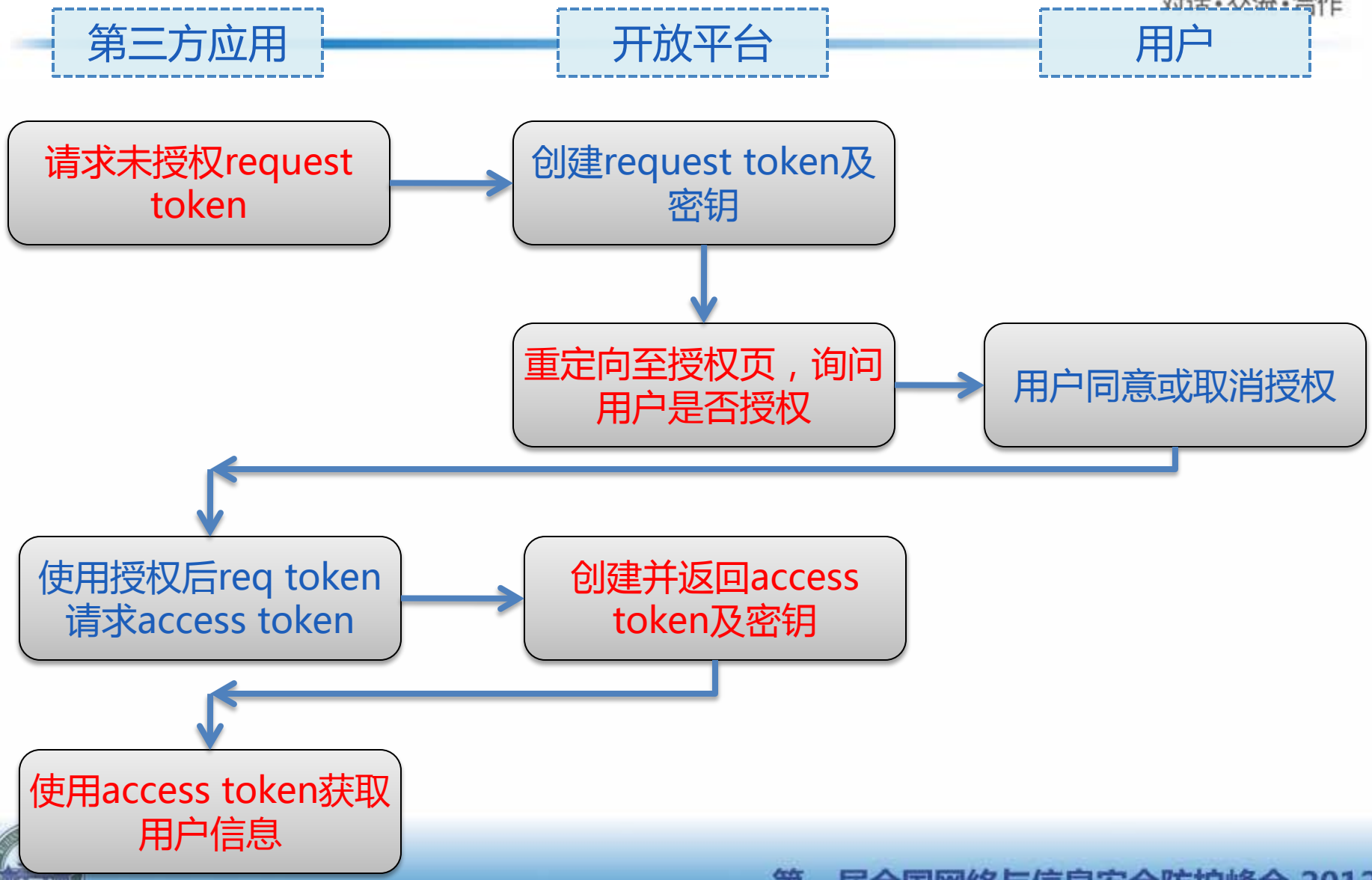
对于用户:免去了繁琐的注册过程，降低了注册成本，提高了用户体验

对于消费方:简化自身会员系统的同时又能够带来更多的用户和流量。

对于服务提供者:围绕自身进行开发，增加用户粘性。

二，OAUTH1.0&SECURITY ISSUE

OAuth1.0 授权流程

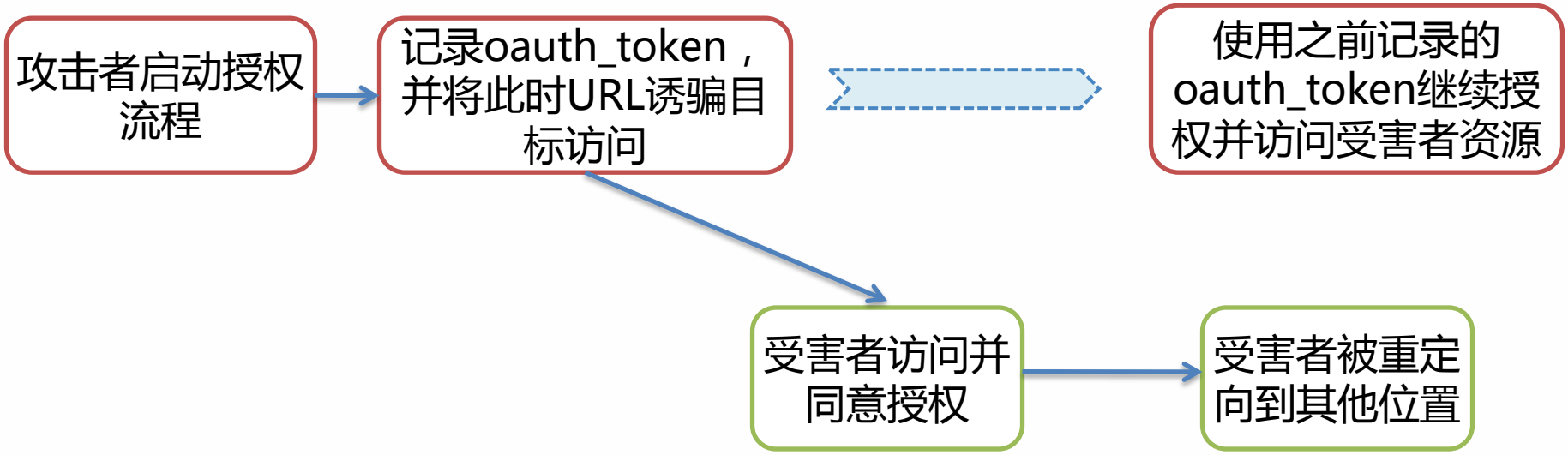


OAuth1.0安全问题

对话·交流·合作

Session Fixation Attack(会话固话攻击)：

由于1.0版本中没有对oauth_callback进行检查和限制，也没有任何机制保证整个授权流程只由一个人完成，因此造成了该安全问题，导致攻击者可以访问目标的用户资源。



诱骗URL：http://provider/auth/oauth_token=123456
&oauth_callback=http://evil.com/
攻击者伪造URL：http://consumer/callback/oauth_token=123456

关联案例：
WooYun-2010-00781《Sina 微博OAuth 存在session fixation attack漏洞 》

OAuth1.0安全问题

- **OAuth1.0a版本中已经将此问题修复：**
  - oauth_callback在请求未授权request token时即传递给平台方并参与签名计算，从而避免攻击者篡改oauth_callback回调地址。
  - 平台方获得用户授权后重定向用户到第三方应用时，返回oauth_verifier，它会被用在第三方申请Access Token的过程中，是一个随机的无法猜测的值。

三，OAUTH2.0&SECURITY ISSUE

OAuth2.0

OAuth2.0提供了多种授权流程：

- **Authorization Code授权:**适用于有Server端配合的应用
- **Implicit Grant授权:**适用于无Server端配合的应用
- **Resource Owner Password Credentials授权**:使用用户名密码进行授权
- 此外还有一种Refresh Token获取Access Token的方式

OAuth2.0与OAuth1.0对比：

- Request Token在2.0中不再使用
- 取消所有签名，整个授权流程使用HTTPS确保安全性
- 授权流程大大简化，安全性有所提高
- 与1.0不兼容，作为一个全新的协议

OAuth2.0 ：Resource Owner Password Credentials

请求

https://open.xxx.com/oauth2/access_token?
client_id=1111111&client_secret=f7404ko9s748728313
&grant_type=password&username=xxx&password=xxx

该授权方式只需一步，用户在client端输入自己的用户名及密码，并同意授权后，资源提供方会直接返回access token给第三方使用。

安全问题：

这种授权方式通常需要授权提供方对第三方应用有着高度信任。
恶意应用使用该方式授权，可以导致暴力破解资源提供方的用户账密。

OAuth2.0： Implicit Grant授权流程

Implicit Grant 是Client端的授权流程，无需Server端配合，一步授权。
Access token被暴露在客户端，是2.0中最不安全的授权流程。

请求

https://api.weibo.com/oauth2/authorize?client_id=2970752
1&redirect_uri=http://www.xxx.com/connect/sinaweibo
&response_type=token

http://www.xxx.com/connect/sinaweibo#access_token=2.0
0xH12gCg2yCPD545628d9ce4RDZYD&remind_in=1274
263&expires_in=1274263

响应

整个授权流程结束，授权服务器一般会将如下字段返回给第三方应用：
access_token:用于请求用户资源
expires_in:过期时间
refresh_token:用于刷新access token
scope:请求的权限

OAuth2.0： Implicit Grant安全问题

Client_id与redirect_uri没有做校验或信任域范围过大，配合信任域下的XSS，即可劫持用户access token。

关联案例：

WooYun-2012-05804《人人网Oauth 2.0授权可导致用户access_token泄露》
漏洞详情：
1、登陆人人网
2、访问该地址
http://graph.renren.com/oauth/grant?client_id=cd271e3051444285b8a18f1211a095cd&redirect_uri=http://zone.ku6.com/u/17958620&response_type=token
3、跳转到存在xss的酷6地址
http://zone.ku6.com/u/17958620
2步中的人人那个地址是用来授权第三方的，response_type=token的授权请求只需要提供应用的client_id以及该应用申请时所填写的回跳地址redirect_uri，但是人人网并没有对redirect_uri进行严格检查，如果该redirect_uri域下存在xss漏洞，则可以诱导用户授权并劫持该用户的access_token
WooYun-2012-12683《百度开放平台oauth授权接口可以劫持access_token》
WooYun-2012-12689 《QQ互联开放平台oauth授权接口可以劫持access_token》

OAuth2.0： Implicit Grant安全问题

攻击者启动正常授权流程 → 伪造回调地址诱骗目标访问 → 从伪造页面中解析access token

伪造回调地址诱骗目标访问 → 目标访问 → 被跳转到存在XSS的页面 →

OAuth2.0： Implicit Grant安全问题

越权API访问

授权服务器通常会给予第三方应用一些基础功能API的权限，如果用到受限API，需要指定scope进行申请。若access token没有与应用的appkey绑定，可导致越权访问受限API：

https://graph.qq.com/oauth2.0/authorize?response_type=token&client_id=28&redirect_uri=http://open.z.qq.com/success.jsp&scope=get_user_info,add_one_blog

从回调地址的fragment部分中获得access token，此时已经被授权了高级权限接口。

关联案例：
WooYun-2012-12462《可用QQ登录平台发表空间日志等高权限操作》

OAuth2.0：Authorization Code 授权流程

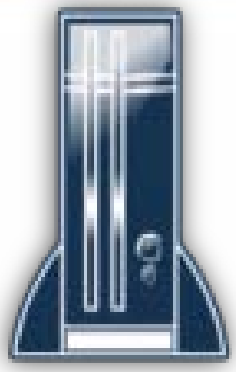
第三方应用

跳转到授权页面请求code[GET]

用户同意授权, 返回code[GET]

第三方利用code请求access token[POST]

以响应体的形式返回access token[POST]



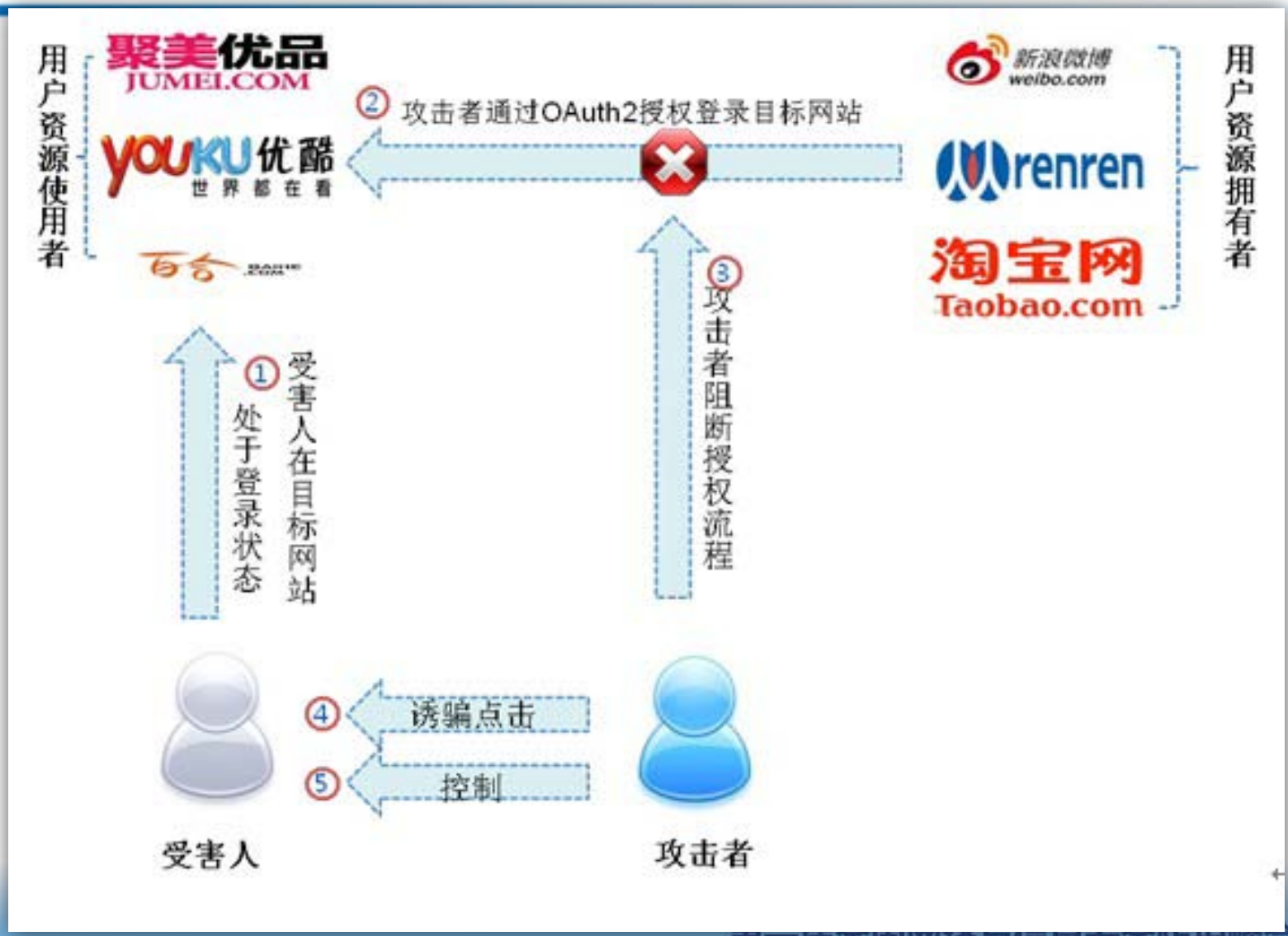
授权服务器

https://api.weibo.com/oauth2/authorize?client_id=1111111&redirect_uri=http://www.xxx.com/connect/sinaweibo?&response_type=code

第三方应用

https://api.weibo.com/oauth2/access_token?client_id=1111111&client_secret=l29dgp03m2&grant_type=authorization_code&redirect_uri=http://www.xxx.com/connect/sinaweibo?&code=1sk42lf0s

OAuth2.0：Authorization Code CSRF



测试案例展示：

对话·交流·合作

测试场景：攻击者要通过新浪微博劫持目标的360主站及360浏览器帐号

1，目标(victim_01)处于登录状态：

2,目标当前未绑定任何第三方帐号：

测试案例展示：

3,攻击者(attacker_01)通过新浪微博登录360：

4,点击授权后阻止页面自动跳转：

测试案例展示：

5，诱骗目标访问：
(以下页面真实攻击时不会在受害方出现)

http://i.360.cn/oauth/bind?a=dobind&c=Sina&f=&destUrl=&type=

6，目标访问URL，劫持成功：

测试案例展示：

7，由于360浏览器同样支持第三方登录：

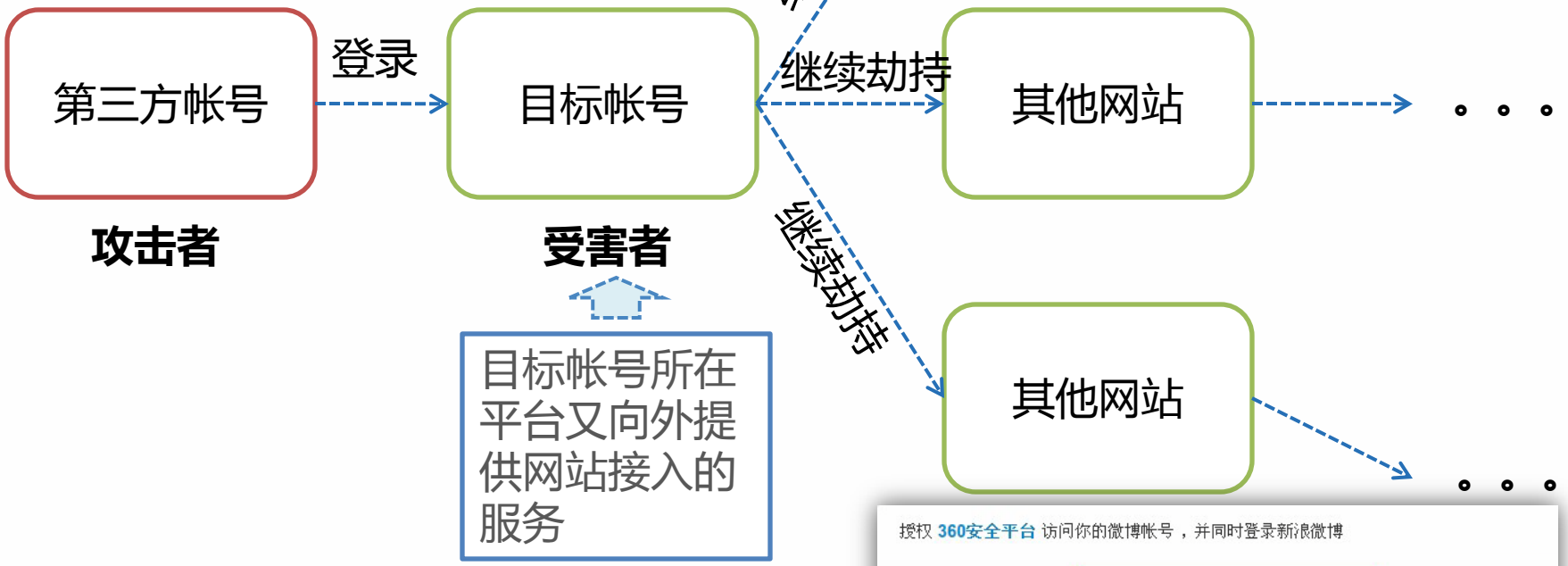
测试案例展示：

对话·交流·合作

8，成功劫持目标360浏览器的账户：

OAuth2.0：Authorization Code CSRF递归劫持

模拟场景：
已经通过CSRF劫持了用户的某个帐号



有时会通过forcelogin来强制用户输入第三方账密信息，而改为false即可直接读取session登录。

OAuth2.0：Authorization Code Replay Attack

对话·交流·合作

授权码重放攻击：

Authorization Code通常会随着access token过期而过期。规范中建议过期时间为60到80分钟，且要保证授权码仅可用一次。而很多资源提供方为了降低授权成本，没有严格按照规范实施，导致该安全问题。

http://magru.net/users/auth/facebook/callback?code=AQDxwQebPEVVzmlfnFIZ0U11gCOVFUxz5MDcgvmWEnXwsbGF

获取目标授权码方式：

- 日志:GET “/auth/sinaweibo/callback?code=...”
- 嗅探：回调地址位于Client端，不会强制使用HTTPS
- XSS：配合目标网站同域下的XSS可以实现劫持code

```
<iframe/src="https://www.facebook.com/dialog/permissions.request?app_id=159836&display=page&next=http://ori.net/users/callback&response_type=code"
name="refcontainer" onload="alert(refcontainer.document.referrer)"></iframe>
```

四，OAUTH安全STYLE

资源提供方：

- 对client_id和回调地址做严格校验
- 获取access token的code仅能使用一次
- 尽量避免直接读取当前用户session进行绑定

资源使用方：

- 使用Authorization Code方式进行授权
- 授权过程使用state随机哈希，并在服务端进行判断
- 尽量使用HTTPS保证授权过程的安全性

关于我们

对话·交流·合作

我们的需求和期待

我们能输出什么？

我们的成果分享及交流途径

通过官方博客：http://blog.knownsec.com/

通过官方微博： @知道创宇

Thanks！

weibo.com/pnigos

twitter.com/pnig0s