【译】不合理的使用OAuth，导致账号被分分钟登录

□ / 2017-11-08 09:34:00 / 浏览数 5003 技术文章 技术文章 顶(0) 踩(0)

原文在此->Signing-Into-Billion-Mobile-Apps-Effortlessly-With-OAuth20。
译文在不改变原文的基础进行了部分调整。
术语：

- implicit flow

摘要

主流的身份提供商（IdP）使用OAuth2.0协议来支持单点登录服务。由于此协议最初设计用于满足第三方网站的授权需求，所以在使用OAuth来支持移动应用程序（app）身
USA'16 [3]，CCS'14 [2]和ACSAC'15
[5]提到的所有攻击都需要与受害者进行交互。例如通过恶意应用程序网络窃听等。但我们发现了第三方app开发人员不合理的使用OAuth导致的一种影响甚广的新型攻击手法
，Google或新浪）提供的基于OAuth2.0的身份验证服务。结果令人震惊：这些应用程序平均有41.21%容易受到新的攻击。我们已经向受影响的IdP汇报了我们的发现，并收

1. 介绍

由于第三方网站采用的基于OAuth2.0的单点登录（SSO）服务广受用户喜爱，最近，许多主流身份提供商（IdP）（如Facebook，Google和Sina）已经将OAuth2.0协议进

由于上述的问题，我们进一步对使用了顶级IdP服务的第三方移动应用程序进行了测试，发现了一种影响非常广的基于OAuth2.0的SSO服务的问题。这个问题的本质非常普遍
1毫不费力地登录受害者的移动应用程序帐户，而无需欺骗或与受害者进行交互，例如通过恶意应用程序或网络窃听等等。就目前来说我们的攻击是基于Android平台进行展

2. 背景

第三方移动应用的SSO服务细节涉及到四个部分：

- a. 第三方移动应用程序的后端服务器（app Server）
- b. IdP的后端服务器（IdP Server）
- c. 第三方移动应用（App）
- d. IdP的移动应用（IdP App）

OAuth的最终目的是让IdP服务器（IdP Server）向app Server发出身份证明，比如access token。通过access token，app
Server可以获取到由IdP服务器管理的用户信息，并进一步根据该信息识别用户并授权登录。

2.1 移动平台上的OAuth 2.0协议流程

图1描述了OAuth协议在网站和移动平台上实现的流程。为简单起见，我们首先介绍移动端OAuth实现的协议流程，然后指出与Web站点SSO服务的差异。请注意，由于OA

1. 用户访问app，并尝试通过IdP进行登录。app通过手机操作系统（Android）提供的安全通道将应用信息（如包名，签名和请求的权限等）发送给IdP的客户端。
2. 通过调用低级系统API，IdP客户端（IdP app）可以验证app的应用信息。如果信息无误，那么IdP客户端（IdP app）会向IdP服务器（IdP Server）发送授权请求。
3. IdP Server收到请求后，会比较来自IdP客户端（IdP app）的授权请求的信息和由第三方移动应用开发者预先注册的信息。如果相同，则IdP服务器（IdP Server）将通过自己的IdP客户端（IdP App）向第三方手机客户端（app）发出access token（AT）和可选的用户信息。
4. IdP app通过安全通道将access token（AT）返回给app（第三方客户端应用程序）。
5. 第三方客户端应用程序（app）将AT发送到其后端服务器（app Server）。
6. 第三方后端服务器（app Server）调用由IdP提供的重要安全SSO-API来调试access token。
7. 在验证access token的有效性之后，IdP服务器（IdP Server）会向第三方应用服务器（app Server）发送授权信息，同时指明access token发给哪个app。
8. 只有在授权信息正确的情况下，第三方应用服务器（app Server）才能通过access token获取用户数据。
9. IdP服务器（IdP Server）返回与access token关联的用户信息。
10. 通过用户信息，第三方应用程序服务器（app Server）可以识别用户并授权登录。

2.2 OpenID连接协议

由于OAuth2.0最初是为授权而设计的，为了适应认证需求，这就涉及了多次高延迟的请求，即图1（b）的步骤6到步骤9。为了更好地支持使用OAuth2.0进行身份验证（即
Connect（OIDC）协议[4]和拓展。具体而言，IdP服务器需要对用户信息进行数字签名。如图2所示，签名的用户信息以及原始的access
token，随后会被发送到app Server（第三方移动应用程序的后端服务器）。由于签名不能被攻击者篡改/伪造，app
Server现在可以通过签名直接识别用户。换句话说，app Server可以立即从签名中提取用户信息，而不需要进行高延迟的API调用。

2.3 网站OAuth实现的不同

从图1所示的协议实现流程图去看，网站和移动端的差异看起来似乎很简单，但实际上正是因为这种简单的不同，导致重要的安全结论并促使OAuth在移动平台上实现的复杂
（i）第三方web服务器（Client Server）
（ii）IdP的后端服务器（IdP Server）
（iii）终端用户的浏览器（Browser），要求能够支持第三方网站OAuth2.0的SSO。
而之前聊到过的移动端SSO交互过程，却涉及到了4个部分（a、b、c、d）。首先，（c）和（d）都在用户设备上运行，并且可能被篡改。其次，OAuth2.0协议标准并未涉

flow）。此外，典型的移动应用程序的客户端负责更多的消息交换，相反，在第三方网站（及其相应的web服务）的情况下，这些消息是由后端服务器管理。

3. 不同的错误实现形式

尽管平台之间差异巨大，但IdP没有提供明确的开发手册来降低OAuth用于移动端可能发生的隐患。所以，第三方移动应用程序开发人员早就犯了各种各样的错误，比如下面

1. 如图3（a）所示，当IdP服务器会返回用户信息和OAuthaccess token（例如，用户ID /电子邮件地址）时，许多第三方应用的后端服务器根据收到的用户信息来授权登录，而不验证收到的用户信息是否真的绑定到已发布的OAuth access token。
2. 图3（b）显示了Facebook和Google采用OpenID Connect协议的另一种情况。在这种情况下，IdP需要对用户信息进行数字签名，以便第三方应用的后端服务器可以通过签名验证来对用户进行鉴别。但是，某些第三方
3. 还有些第三方移动应用程序直接从其正在运行的移动设备获取用户信息，直接忽视IdP接收到的OAuth token。（例如，图4(a)所示的IdP服务器提供API，应用通过调用获取用户信息或图4(b)所示设备上有个账户系统存储了用户Google账户信息，可以用来支持SSO服务） token，第三方后端服务器无法验证返回的用户标识符是否绑定了OAuth access token。

4. 漏洞利用

因为第三方应用程序开发人员的不规范开发，攻击者可以利用受害者信息登录到存在问题的app，这一切只需要下面这些步骤：

1. 如图5所示，攻击者在自己的移动设备上启用了ssl的MITM代理（比如mitm-proxy），监控往来的网络流量。
2. 攻击者在自己的移动设备上安装了易受攻击的第三方应用程序。
3. 攻击者通过自己的IdP用户密码，用OAuth登录了易受攻击的移动应用程序。
4. 在步骤3触发的OAuth消息交换过程中，攻击者通过ssl的MITM代理将受害者的用户标识替换为自己的用户标识（IdP或电子邮件地址中的用户标识）。受害者的用户ID是
5. 由于第三方后端服务器直接使用客户端应用程序返回的用户身份证明来标识app用户，因此攻击者可以用受害者的身份登录app，并且在大多数情况下拥有完整的权限访

除了受SSL / HTTPS保护外，应对第三方移动应用程序的客户端与其后端服务器之间的消息交换进行加密或签名。否则的话，篡改IdP服务器返回的用户标识信息是很容易的。

在IdP客户端应用程序（例如Facebook的应用程序）应用证书锁定的情况下，如果攻击者通过MIMT代理篡改了IdP服务器发送给其客户端应用程序的消息，那该消息不会被 SDK（通常是由OAuth2.0第三方移动应用程序广泛使用）将自动降级，然后通过内置webview浏览器进行OAuth身份验证。对于常见的内置浏览器来说，webview不支持

除了IdP不支持基于webview的OAuth授权情况，还有一些其他的情况。对于这些情况下的IdP来说，攻击者可以使用现成的工具，如SSLUnpinning（如果他们使用原生的A

5. 现实的惨状

我们研究了由三家顶级IdP（即新浪，Facebook和Google）提供的基于OAuth2.0的API，这三家IdP支持全球许多第三方移动应用的SSO服务。如表1所示，这些IdP的注册 [1]最近的调查，以51%的SSO用户采用率进行保守估计的话，截至撰写本文时，有超过10亿的不同类型的移动应用账号容易受本文所讲述的攻击。

攻击者通过exp登录受害者手机应用账号，并且大多数情况下他们是拥有完整的权限，能够访问受害者的隐私，尽管这些信息由被黑app的服务器管理。单纯是针对表2中列出

6. 建议

我们的研究已经展示了这个问题的危害性，对于第三方开发来说在实现或使用基于OAuth2.0的服务时应采取如下措施进行补救：

1. IdP应该提供基于OAuth2.0的SSO API更清晰、更侧重安全性的使用准则。
2. app的后端服务器不应该信任任何信息，即使信息被app或IdP的app签了名。最好只相信来自IdP服务器的信息。
3. IdP不应依赖全球用户标识符来进行第三方应用程序认证/授权，而应根据每个移动应用程序发布私人用户标识符。事实上，自2014年5月以来，Facebook已经采用了这种
4. IdP应对第三方移动应用程序进行更加全面的安全测试，特别是通过OAuth2.0或其他类似协议（如OpenID Connect（OIDC）协议）实施单点登录服务这块。

7. 结论

本文中，我们已经确定了一个以前未知的漏洞，攻击者无需交互就能利用这个漏洞劫持受害者的移动应用账户。我们已经检查了美国Top200的app和中国Android app情况，当然这些app都是使用了三家顶级IdP的OAuth2.0授权服务。同时我们展示了这些流行应用程序在多大程度上会受到这种新漏洞的攻击。我们的发现表明，各方追

8. 引用

[1] “Social login continues strong adoption,” 2014. [Online]. Available:

http://janrain.com/blog/social-login-continues-strong-adoption/

[2] E. Y. Chen, Y. Pei, S. Chen, Y. Tian, R. Kotcher, and P. Tague, “OAuth demysti□ed for mobile application developers,” in Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security. ACM, 2014.

[3] C. Eric, P. Tague, R. Kotcher, S. Chen, Y. Tian, and Y. Pei, “1000 ways to die in mobile OAuth,” in BlackHat USA, 2016.

[4] N. Sakimura, J. Bradley, M. Jones, B. de Medeiros, and C. Mortimore, “OpenID Connect core 1.0,” The OpenID Foundation, p. S3, 2014.

[5] H. Wang, Y. Zhang, J. Li, H. Liu, W. Yang, B. Li, and D. Gu, “Vulnerability assessment of OAuth implementations in Android applications,” in Proceedings of the 31st Annual Computer Security Applications Conference. ACM, 2015.

[6] Q. Ye, G. Bai, K. Wang, and J. S. Dong, “Formal analysis of a Single Sign-On protocol implementation for Android,” in 20th International Conference on Engineering of Complex Computer Systems, ICECCS 2015, 2015.

上一篇：利用Ms17-10获取Window... 下一篇：有木有办法发post的时候不带or...

1. 0 条回复
   - 动动手指，沙发就是你的了！