OWASP MOBILE TOP 10 （BETA）

鸣谢

感谢 SecZone 自 2014 年 Mobile TOP 10 项目成立以来，对该项目持续的跟进、翻译、研究、分享。

同时，我们也对该项目的主要翻译人员王颉表示感谢。

Mobile TOP 10 项目支持单位：

目录

鸣谢 2
OWASP Mobile Top 10 简介 5
M1：平台使用不当 7
1.1. 概述 7
1.2. 明显特征 7
1.3. 风险 8
1.4. 举例 8
M2：不安全的数据存储 9
2.1. 概述 9
2.2. 明显特征 10
2.3. 风险 10
2.4. 举例 (空) 10
M3：不安全的通信 11
3.1. 概述 11
3.2. 明显特征 11
3.3. 风险 11
3.4. 举例 12
M4：不安全的身份验证 13
4.1. 概述 13
4.2. 明显特征 13
4.3. 风险 14
4.4. 举例 14
M5：加密不足 16
5.1. 概述 16
5.2. 明显特征 16
5.3. 风险 16
5.4. 举例 17
M6：不安全的授权 18

6.1. 概述 18
6.2. 明显特征 18
6.3. 风险 18
6.4. 举例 18
M7：客户端代码质量问题 19
7.1. 概述 19
7.2. 明显特征 19
7.3. 风险 19
7.4. 举例 19
M8：代码篡改 20
8.1. 概述 20
8.2. 明显特征 20
8.3. 风险 20
8.4. 举例 20
M9：逆向工程 21
9.1. 概述 21
9.2. 明显特征 21
9.3. 风险 21
9.4. 举例 21
M10：无关的功能 22
10.1. 概述 22
10.2. 明显特征 22
10.3. 风险 22
10.4. 举例 22

OWASP Mobile Top 10 简介

M1-平台使用不当

这个类别包括平台功能的滥用，或未能使用平台的安全控制。它可能包括 Android 的意图（intent）、平台权限、TouchID 的误用、密钥链（KeyChain）、或是移动操作系统中的其他一些安全控制。有几种方式使移动应用程序能受到这类风险。

M2-不安全的数据存储

这个新的类别是《2014 年版十大移动安全威胁》中 M2 和 M4 的组合。这个类别包括不安全的数据存储和非故意的数据泄漏。

M3-不安全的通信

这个类别包括不健全的握手通信过程、SSL 版本的不正确使用、脆弱协议、敏感信息的明文传输，等等。

M4-不安全的身份验证

这个类别包括对终端用户身份验证或坏的会话管理的意见。这可以包括：

- 当被要求时，没有对所有用户进行身份识别。
- 当被要求时，没有保持对用户身份的确认。
- 会话管理中的漏洞。

M5-加密不足

代码使用加密技术对敏感信息资产进行加密。然而，加密技术的应用在某种程度上是不足的。需要注意的是，任何与 TLS 或 SSL 有关的内容调整至 M3 中。此外，如果应用程序在它应当使用加密技术时而没有成功使用，该类问题可能属于 M2。本类别是在尝试使用加密技术时，却又没有成功使用的问题。

M6-不安全的授权

这个类别包括任何失败的授权行为（例如：在客户端的授权决策、强迫浏览等。）。它有别于身份验证问题（例如：设备注册、用户标识等）。

如果应用程序在需要的时候没有验证用户的身份 （例如：当访问要求需经过身份验证和授予权限时，授予匿名用户访问某些资源或服务的权限），那就是一起身份验证失败事件，而不是授权失败事件。

M7-客户端代码质量问题

这个类别曾经是“通过不可信的输入做出安全决定”，是我们较少使用的类别之一。这将包括全部的移动客户端代码级别开发问题。这不同于服务器端的编码错误。本类别包括例如缓冲区溢出、 字符串格式漏洞以及其他不同类型的代码级错误，而这些错误的解决方法是重写在移动设备中运行的某些代码。

M8-代码篡改

本类别包括二进制修补、 本地资源修改、 方法钩用、方法调整和动态内存修改。

一旦应用程序交付至移动设备，代码和数据资源就都存放在那里。攻击者要么可以直接修改代码、动态修改内存中的内容、更改或替换应用程序使用的系统API，要么可以修改应用程序中的数据和资源。这可以为攻击者提供颠覆本软件用户的使用预期或是获得金钱利益的直接方法。

M9-逆向工程

本类别包含对核心二进制代码的分析，以确定它的源代码、 库文件、 算法和其他资产。比如：IDA Pro、Hopper、 otool 和其他二进制检验工具，使攻击者能洞察到应用程序内部的工作原理。这可用于在应用程序中发现其他漏洞，并可揭露有关后端服务器、加密常数、密码以及知识产权的信息。

M10-无关的功能

通常，开发人员不会打算将隐藏地后门程序功能或其他内部开发安全控件发布到生产环境中。例如：开发人员可能在一个混合应用程序中无意包含了一个作为注释的密码。另一个例子包括在测试阶段禁用了双因子身份验证。

M1：平台使用不当

1.1. 概述

这个类别包括平台功能的滥用，或未能使用平台的安全控制。它可能包括 Android 的意图（intent）、平台权限、 TouchID 的误用、 密钥链（KeyChain）、或是移动操作系统中的其他一些安全控制。

有几种方式使移动应用程序能受到这类风险。

1.1.1. 违反了已公布的开发准则

所有平台（如：Android、iOS、Windows Phone）都针对安全提供了开发准则 。如果一个应用程序与厂商推荐的最佳实践相矛盾，那么它将暴露给这种风险。例如：有关于如何使用 iOS 密钥链或如何在 Android 上安全的导出服务的准则。不遵循这些准则的应用程序将会受到这种风险。

1.1.2. 违反了惯例或常见的实践

并不是所有的最佳实践都在厂商的开发准则中编纂。在某些情况下，移动应用程序中一些实际的最佳实践，在移动应用程序中是很常见的。

1.1.3. 无意误用

一些应用程序意图执行正确的操作，但实际上得到的部分应用存在错误。这可能是一个简单的 bug，如：在一个 API 调用上设置错误标志，或可能是对保护机制是如何工作的错误理解。

关于平台上有关权限模型的错误属于此类别。例如：如果应用程序请求太多的权限或错误的权限，都最好被分在本类中。

1.2. 明显特征

此类别中风险的决定性特征是：平台（iOS、Android、 Windows Phone 等）提供一项被记录的、充分理解的功能或能力。应用程序无法使用这种能力，或者没有正确地使用它。这不同于其他的十大移动风险，因为严格说来，设计和应用不是应用程序开发人员的问题。

1.3. 风险

这类风险的影响很难形容。移动平台提供许多不同的服务，从身份验证，到安全的数据存储，再到安全的网络通信。没有正确地使用平台的一些部分，因此，可能造成数据暴露、允许连接到不受信的主机、或实现欺诈付款。移动应用程序的隐私和权限也是平台的领域。因此，未能恰当使用平台的功能，可能会使最终用户暴露隐私。

1.4. 举例

由于存在多个平台，且每个平台都有成千上万项 API，本节的举例仅触及到一些皮毛。

1.4.1. 替代密钥链的应用程序本地存储

iOS 密钥链是一个对应用程序和系统数据的安全存储设备。在 iOS 中，应用程序应该使用它来存储任何具有安全特征的小型数据（如：会话密钥、密码、设备登记数据等）。一个常见的错误，是将这些数据存放在应用程序的本地存储中。存放在应用程序本地存储中的数据，可以在没有被加密的 iTunes 备份中使用（如：在用户的计算机上）。对于某些应用程序，将数据暴露是不恰当的。

M2：不安全的数据存储

2.1. 概述

这个新的类别是 2014 年版《OWASP 十大移动风险》中 M2 和 M4 的组合。本类别包括“不安全的数据存储”和“非故意的数据泄漏”。

不安全的数据存储，包括但不限于以下方面：

- SQL 数据库；
- 日志文件；
- XML 数据中存储 OU 名单文件；
- 二进制数据存储；
- Cookie 存储；
- SD 卡；
- Cloud 同步。

非故意的数据泄漏，包括但不限于以下方面：

- 操作系统；
- 开发框架；
- 编译器环境；
- 新硬件设备。

这显然离不开开发者的知识。特别是在移动开发过程中，最常见没有记录的、或记录在案的内部处理，包括：

- 操作系统缓存数据、图像、按键、日志和缓冲区的方式；
- 开发框架缓存数据、图像、按键、日志和缓冲区的方式；
- 数据、分析、社交或实现框架缓存数据、图像、按键、日志和缓冲区的方式与次数。

2.2. 明显特征

它对你的移动应用程序、操作系统、平台和框架进行威胁建模分析是非常重要的，这样可以对应用程序处理的信息资产有所理解，并且可以知道 API 是如何处理这些资产的。关键是看它们如何处理以下类型的功能：

- URL 缓存（包括请求与响应）；
- 键盘按键缓存；
- 复制、粘贴缓冲区缓存；
- 应用程序后台处理；
- 日志；
- HTML5 数据存储；
- 浏览器 cookie 对象；
- 发往第三方的数据分析。

2.3. 风险

这可能会导致数据丢失，最好的情况是只丢失一个用户的数据，而最坏的情况是丢失许多用户的数据。它也可能导致以下技术影响：通过移动设备中所含的恶意软件、被篡改的应用程序或调查分析工具，提取应用程序中的敏感信息。

对业务影响的性质，高度依赖于被盗信息的性质。不安全的数据可能会导致以下业务影响：

- 身份信息被盗；
- 隐私泄露；
- 诈骗；
- 名誉受损；
- 违反外部的政策要求（如：PCI）；
- 材料丢失。

2.4. 举例 (空)

M3：不安全的通信

3.1. 概述

本类风险涵盖了从 A 点到 B 点之间不安全地获取数据的所有方面。它的组成包含了移动设备之间的通信、应用程序至服务器之间的通信、或者移动设备至其他之间的通信。这种风险包括移动设备可能使用的所有通信技术：TCP/IP、WiFi、蓝牙或 LE 蓝牙、NFC、音频、红外、GSM、3G、短信等。

所有有关 TLS 通信、NFC、蓝牙和 WiFi 的问题，都包含在这部分。

3.2. 明显特征

明显特征包括将一些敏感数据组包，并发送出或发送到设备中。一些有关敏感数据的例子包括：加密密钥、密码、用户的私人信息、账户信息、会话令牌、文件、元数据和二进制文件。敏感数据可以从服务器传递进入设备，可以从应用程序传递进入服务器，或者可能是设备和其他本地设备（如：NFC 终端或 NFC 卡）之间的数据通信。这种风险的明显特征是存在两个设备，且在它们之间传递一些数据。

如果数据被本地存储在设备中，那这就是不安全的数据。如果会话细节以安全的方式通信（如：通过强大的 TLS 连接），但会话标识符本身是坏的（或许是可猜到的、低熵等），那么这则是一个“不安全的身份验证”问题，而不是通信问题。

3.3. 风险

不安全的通信的一般风险有关于数据的完整性、数据的保密性和数据来源的完整性。如果数据在传递过程中是可以改变的，且在信息传递过程中没有检测到发生的变化（如：通过中间人攻击），那么这就是这种风险的一个很好的例子。如果保密数据可以被暴露、获取、或通过在通信过程中观察（如：窃听）而得出、或通过在通信过程中进行录制并在稍后进行攻击（如：离线攻击），这也是不安全的通信问题。没有正确安装和验证一个 TLS 连接（如：证书的检验、弱密码、其他 TLS 配置问题）都属于“不安全的通信”问题。

3.4. 举例

渗透测试人员经常在检查一个移动应用程序的安全通信时发现的常见情况：

3.4.1. 缺乏证书检查

移动应用程序和终端成功连接并执行一个 TLS 握手协议建立安全通道。然而，移动应用程序没有检查服务器提供的证书，且移动应用程序无条件地接受由服务器提供给它的任何证书。这破坏了在移动应用程序和终端之间的相互认证能力。移动应用程序通过 TLS 代理容易受到中间人攻击。

3.4.2. 脆弱的握手协议

移动应用程序和终端成功连接，并协商密码套件作为连接握手的一部分。客户与服务器成功地协商使用弱密码套件，导致加密强度不足，以至于可以很容易地被攻击者破解。这种方式危害了移动应用程序和终端之间信道的保密性。

3.4.3. 隐私信息泄露

移动应用程序通过不安全的信道传递个人身份信息到终端，而不是通过 SSL。这种方式危害了移动应用程序和终端之间有关隐私数据的保密性。

M4：不安全的身份验证

4.1. 概述

这个类别包括对终端用户身份验证或坏的会话管理的意见。这可以包括：

- 当被要求时，没有对所有用户进行身份识别；
- 当被要求时，没有保持对用户身份的确认；
- 会话管理中的漏洞。

身份验证问题是一个处置与身份验证有关的隐私问题的合理地方。例如：如果一个移动应用程序使用特定于设备的数据，如：IMEI、蓝牙 MAC 地址、或其他作为用户身份验证的硬件标识符，可以为应用程序开发人员或拥有者建立对隐私保护的期望。

如果使用不安全的信道进行身份验证而导致欺骗、重放或其他针对身份验证的攻击，那么这也属于这个类别的范畴。在移动应用程序中不恰当的存储密码，属于“不安全的数据存储”问题；以明文传输密码，则属于“不安全的通信”问题。例如：如果攻击者通过观察一个用户在服务中的注册行为，而洞察到用户的数据，并将其用于随后的注册行为中，这就是属于本类别的一个身份验证问题。

1.1. 会话的问题

在以前版本的《OWASP 十大移动风险》中，有不同类别的会话管理问题。而在本版本中，所有这些问题都集中在这里，除非它们有另一个更好的位置。

会话的问题包括：

- 可预见的会话标识符；
- 用户登出失败；
- 会话寿命风险（会话有效时间太长、会话在跨多个信道中有效）；
- 会话固定。

4.2. 明显特征

身份验证问题的关键特征是，代表用户的令牌可以在某种程度上受到损害。另

外，应用程序可能带有在所有需要身份认证的地方进行身份认知的机制。例如：没有进行用户识别的机制、没有识别或注册设备的方式、没有验证微服务真实性的方式等。

4.3. 风险

一个有关身份验证的常见风险是向未识别身份的用户暴露数据或提供服务。匿名用户可以调用 Web 服务（如：计划旅行路线、将项目放入购物篮、产生图像），而 Web 服务的本意是只允许已注册的或已识别身份的用户执行该操作。

4.4. 举例

4.4.1. 没用的可猜测的标识符

如果一个应用程序使用了一些小的、非递减的整数来标识用户，它将会明显受到攻击，如：欺骗。

4.4.2. 匿名服务终端

在移动应用程序中的一个常见的问题是，当未经身份验证的请求发送至服务移动应用程序的服务终端时，服务终端向未经身份验证的请求提供了信息。想象一个会员忠诚度计划，其为每个会员编号关联了一些忠诚度积分。如果微服务向某个特定的会员反馈提供了折扣代码，但没有事先验证请求发起人的身份，则可能导致滥用（如：攻击者检查所有会员编号寻找有价值的折扣代码）。

该问题的解决方法是安全地验证用户的身份，并在显示会员数据时要求身份验证。

4.4.3. 使用私有数据作为身份信息

在欧洲和英国，IMEI 和其他硬件地址数据是私有数据，必须按照对待用户电话号码或家庭住址的方式同样对待。使用硬件标识符作为用户标识符的移动应用程序，可能是传输、存储和使用这些标识符的方式，但这并不符合适用的隐私保护法律。

该问题的解决方法是为用户使用一个独特的、不可预测的标识符，且与任何私有数据无关。另外，应用程序只要在对待私有数据时使用了合理的策略与技术保障安全性，则可以使用这些硬件地址。

4.4.4. 仅客户端登出

应用程序有时并没有真正的从应用程序的服务器端登出。应用程序很少删除会话标识符和其他的登录数据。这种脆弱的注销方式可以将会话的保留时间远超出其预期的时间。被观察（如：通过窃听）到的会话可以被克隆到另一个设备或植入一个 Web 浏览器中并继续使用，即使移动应用程序认为它已经退出登录了。

M5：加密不足

5.1. 概述

代码使用加密技术对敏感信息资产进行加密。然而，加密技术的应用在某种程度上是不足的。需要注意的是，任何与 TLS 或 SSL 有关的内容调整至“不安全的通信”类别中。此外，如果应用程序在它应当使用加密技术时而没有成功使用，该类问题可能属于“不安全的数据存储”类别中。本类别是在尝试使用加密技术时，却又没有成功使用的问题。

5.2. 明显特征

本类包含的常见问题为：

- 弱密码；
- 短密钥；
- 错误类型的加密算法（如：对称加密算法比不对称加密算法适用）；
- 众所周知的密码分析攻击漏洞（如：选择明文攻击）；
- 已知明文攻击；
- 差的密钥选择（如：可预测的随机性）。

本类的所有问题都有一个共同的质量问题：应用程序试图保护数据，且保护机制可能在一定程度上有效。但是，总能出于一个或多个原因发现保护机制不足。

5.3. 风险

使用加密技术保护的数据可能会被暴露。典型的结果为：要么是数据的保密性得到了保护，要么是数据的完整性得到了保护。

机密性风险可能包括对数据的离线暴力破解攻击，或者是对加密数据的推理攻击。这种风险意味着攻击者获得了一些或所有受保护的信息。

完整性风险可能包括重放攻击（如果一个密码签名可以伪造的话）。如果代码签名可以被伪造，可能会得到不安全的代码。可猜测的或可预测的加密令牌则可能导致虚假的交易。

5.4. 举例

这里有几个可以被归在这一类问题的例子。

5.4.1. 可预测的密钥

想象一个移动应用程序在数据发送给服务器端以前，使用了强对称加密算法（如：AES-256）对数据进行加密。如果加密的密钥很容易确定，这将是不足的加密技术。也许加密密钥仅仅是用户的数字标识符 userid 或其他一些容易被猜到的标识符。

解决这个问题的方法将涉及到生成一个强密钥，并安全地在服务器和客户端之间进行通信，或者使用非对称加密技术。

5.4.2. 容易伪造的完整性检查

想象一个移动应用程序将数据（如：游戏的高得分、用户的地理位置等）发布到中央服务器。想象一下，移动应用程序在请求中添加了一个用户数据的哈希值作为一个完整性检查的参数。这个哈希值对复制是微不足道的，所以攻击者可以很轻松的伪造一个合法的更新。攻击者只需对假冒的数据计算得到相同的哈希值。

解决这个问题的方法涉及到生成独一无二的签名，且难以伪造。

M6：不安全的授权

6.1. 概述

这个类别包括任何失败的授权行为（如：在客户端的授权决策、强迫浏览等。）。它有别于身份验证问题（如：设备注册、用户标识等。）。

如果应用程序在需要的时候没有验证用户的身份（如：当访问要求需经过身份验证和授予权限时，授予匿名用户访问某些资源或服务的权限），那就是一起身份验证失败事件，而不是授权失败事件。

6.2. 明显特征

[空]

6.3. 风险

典型风险涉及对未经授权的用户授予访问权限。用户可执行他们本不能执行的创建、读取、更新、删除（CRUD）操作。他们可以调用服务，或使用凭据本没有授予他们的服务。

6.4. 举例

6.4.1. 基于客户端的授权决定

如果移动应用程序知道用户的权限级别，并根据用户的等级提供适当的菜单和选项，这是正常的。但是，如果服务器在没有验证用户的权限或等级的情况下响应和执行这些请求，那么这就是一个问题。如果服务器毫无保留的信任移动程序代码，并根据用户的权限级别生成相应的请求，这就是一个典型的不安全授权。

6.4.2. 身份验证替代授权

为移动应用程序服务的 Web 应用程序终端，可能在身份验证通过后允许访问资源，且想访问多久就访问多久。想象一个基于 REST 的微服务，它为一个给定的会员账号提供会员忠诚度积分。如果该服务需要某种身份验证，但不是验证用户的身份，而是被授权查看积分额度，那么这就是失败的授权。

M7：客户端代码质量问题

7.1. 概述

这个类别曾经是“通过不可信的输入做出安全决定”，是我们较少使用的类别之一。这将包括全部的移动客户端代码级别开发问题。这不同于服务器端的编码错误。本类别包括例如缓冲区溢出、 字符串格式漏洞以及其他不同类型的代码级错误，而这些错误的解决方法是重写在移动设备中运行的某些代码。

这不同于“平台使用不当”问题，因为它通常指向编程语言本身（如：Java，Swift，面向对象的 C，JavaScript）。C 语言的缓冲区溢出、或在 Webview 移动应用程序中基于 DOM 的 XSS，都是代码质量的问题。

7.2. 明显特征

这种风险的主要特点是，代码在移动设备上执行，且代码需要根据相当本地化的方式进行修改。最高级别的风险是要求更改代码。但在代码质量事件中，风险往往来自于使用了错误的 API、使用了不安全的 API、使用了不安全的语言结构或其他一些代码级的问题。重要的是：这不是在服务器上运行的代码。这个风险是在移动设备本身执行有害代码。

7.3. 风险

有害代码可以允许攻击者利用业务逻辑，或绕过设备上执行的安全控制。代码级的错误可能以意外的方式暴露敏感数据。

7.4. 举例

如果一个应用程序有一个字符串格式漏洞或类似漏洞，它可以在用户界面上显示敏感信息（如：加密密钥、会话标识、或 API 密钥），而这些敏感信息本应只存在于应用程序中的 RAM 中。

M8：代码篡改

8.1. 概述

本类别包括二进制修补、本地资源修改、方法钩用、方法调整和动态内存修改。

一旦应用程序交付至移动设备，代码和数据资源就都存放在那里。攻击者要么可以直接修改代码、动态修改内存中的内容、更改或替换应用程序使用的系统 API，要么可以修改应用程序中的数据和资源。这可以为攻击者提供颠覆本软件用户的使用预期或是获得金钱利益的直接方法。

8.2. 明显特征

- 对应用程序包的直接修改；
- 系统 API 的重定向或更换；
- 对后端服务器的直接攻击。

8.3. 风险

这类攻击可以改变应用程序的隐性或显性逻辑。这可能用来破坏或绕过在应用程序内的购买或 license 许可，导致应用程序开发人员失去合法的收入来源。它也可以被用来复制应用程序形成恶意变种，以在合法应用程序中植入恶意软件的载荷。此外，该方法还可以用来改变或中断向应用程序后端服务器的网络流量，或将应用程序的通信重定向至攻击者的服务器。

8.4. 举例

有许多假冒的应用程序含有恶意的载荷，且存在于多个应用程序商店中。许多假冒的应用软件是由复制和修改底层核心的二进制代码和资源、再重新封装形成。有条件的跳转可以检测应用程序内的在线购买是否成功，从而使不付钱但想获得内容的终端用户无法成功。

M9：逆向工程

9.1. 概述

本类别包含对核心二进制代码的分析，以确定它的源代码、 库文件、 算法和其他资产。比如：IDA Pro、Hopper、 otool 和其他二进制检验工具，使攻击者能洞察到应用程序内部的工作原理。这可用于在应用程序中发现其他漏洞，并可揭露有关后端服务器、加密常数、密码以及知识产权的信息。

9.2. 明显特征

- 泄漏源代码；
- Intel 导致的代码篡改。

9.3. 风险

通过逆向工程，攻击者可以枚举或绕过业务逻辑、绕过安全控制、促进源代码盗用和篡改代码。如果源代码不是模糊的，那么就变得很容易让攻击者或对手复制应用程序。攻击者可以重新打包应用程序，并通过各种方式发布给公众。这涉及到一个高的业务风险：收入损失、品牌损害、或假冒的应用程序副本。这些假冒的副本也可以便于网络钓鱼或凭据盗窃。

9.4. 举例

考虑一个 Android 的银行应用程序。APK 文件可以很容易地使用 7zip、WinRAR、WinZip、gunzip 软件提取到。一旦被提取，攻击者就拥有了各类清单文件、信息资产、源代码，最重要的是 classes.dex 文件。

然后使用 Dex 至 Jar 的转换器，攻击者可以很容易地将其转换为 Jar 文件。下一步，Java 反编译器（如：JDgui）将为你提供代码。

M10：无关的功能

10.1. 概述

通常，开发人员不会打算将隐藏地后门程序功能或其他内部开发安全控件发布到生产环境中。例如：开发人员可能在一个混合应用程序中无意包含了一个作为注释的密码。另一个例子包括在测试阶段禁用了双因子身份验证。

10.2. 明显特征

这个风险的明显特征是，在应用程序中启用了在发布时并不打算发布的功能。

10.3. 风险

通过这些额外的能力，有窃取敏感数据或访问未经授权功能的高风险。如果一个银行应用程序植入有恶意代码，它不仅会窃取客户信息和资金，而且还损害了银行的声誉。这种恶意代码将启动一个传出连接，这将被视为真正的连接，而不是恶意的连接。尽管拥有各种安全控制，这对银行通过在移动设备上的应用程序检测恶意活动，是非常困难的。

另一个例子是保险应用程序。如果开发人员将恶意代码片段注入任何保险应用程序中，那么，窃取包含所有个人信息在内的客户数据是有很可能的。想象一下，如果这段恶意程序正在监视你的支付交易。

10.4. 举例