支付宝 Windows Phone 应用安全开发规范

本文档适用于 Windows Phone 7 应用的安全开发。

序号	版本号	修订日期	修订概述	修订人	审核人	备注
1	V1.0	2012-06-29	初稿	白开心		
2						
3						

1. Windows Phone 安全模型概述

为了解决移动设备中应用的数据和通信安全问题，微软为 Windows Phone 7 设计了一个安全模型，它包括一个创新的安全架构和一套数据保护策略。

1.1 Chamber

Windows Phone OS 7 的安全模型是建立在基于隔离和最小权限原则（isolation & least privilege principle）基础上，它引入了 chamber（房间）的概念。chamber 是用来实施应用隔离和最小权限原则的抽象概念。

每个 chamber 提供了一个安全边界，通过配置，一个进程可以在这个隔离的安全边界内运行。每个 chamber 都是通过一个策略系统来定义和实现的。而一个指定的 chamber 的安全策略定义了运行在该 chamber 中的进程能够使用操作系统的哪种能力（比如网络、电话、麦克风等）。

Chamber 有四种类型，如图 1.1 所示。其中三种拥有固定的权限集合（fixed permission set），而第四种是能力－驱动类型（capabilities-diriven），运行在这种 chamber 中的应用必须在安装和运行时获取到所需要的 capabilites。对这四种 chamber 概述如下。

图 1.1 Windows Phone Chamber 安全模型

(一) TCB Chamber（Trusted Computing Base）

TCB chamber 拥有最大的权限，允许进程对 wp7 的所有资源进行无限制的访问，并能够修改、实施安全策略。内核和内核模式的驱动、控制软件安装和文件访问的服务运行在 TCB chamber 中，只有微软的签名应用能够添加到该类 chamber 中运行。

(二) ERC（Elevated Rights Chamber）

EPC 能够访问除安全策略外的所有资源，为其他应用提供功能的服务、用户模式的驱动和服务运行在 ERC chamber 中，只有微软的签名应用能够添加到该类 chamber 运行。

(三) SRC（Standard Rights Chamber）

SRC 是为预装应用（包括微软的预装应用和 OEM 应用）准备的默认的 chamber，所有不在设备范围内运行的服务也都运行在 SRC 中，比如 Microsoft

Outlook Mobile 2010。

(四) LPC（Least Privileged Chamber）

LPC 是所有非微软应用默认的 chamber，LPCs 是利用 capabilities 来配置的，这些能力会在应用的 WMAppManifest.xml 中列出，它默认定义了一个最小的访问权限集合，但它能够利用 capabilities 进行动态扩展。

应用访问一个资源时，内核会根据应用/用户的 SID 和一个策略数据库来进行安全策略检查，看是否允许此次访问。

1.2 Capability

一个 capability 是指在 windows 手机上使用与用户隐私、安全、花费、商业等相关资源（包括地理位置信息、摄像头、麦克风、网络、传感器等）时所拥有的访问令牌；如果没有相应的权限时，系统会返回 UnauthorizedAccessException。

一个应用所需要的能力（capabilities）需要在应用的 WMAppManifest.xml 文件中明确定义，然后在应用安装时请求获取那些受限的 API 的访问权限，才能执行相应的操作，应用的权限在运行时不能被提升。

1.3 Sandbox

沙箱是对应用运行时所需的资源进行限制和隔离的模块的统称，它的目的是让不受信任的应用运行在特定的受限环境中，避免它能够访问到隔离区之外的资源。

Windows Phone 中的每个应用：

1) 运行在自己隔离的 chamber 中，除了系统默认给予应用的最小访问权利集合（比如访问 isolated storage），它只具有 WMAppManifest.xml 中声明的能力。
2) 拥有自己的内存空间和隔离的存储文件（isolated storage file），其它应用无法访问。
3) 除了通过 cloud，不能通过别的方式与手机上的其他应用通信。
4) 除微软外，任何其他公司开发并通过微软手机市场发布的应用在不能在后台运行。当用户将一个应用切换到其他应用后，前者就会被关闭，并保存应用状态。这保证了当用户不再使用一个应用时，它不能在后台使用关键的资源或者与网络上的服务通信。

1.4 应用开发安全及发布审核

微软提供多种数据加密和 hash 方法（包括 AES、HMACSHA1、RSA、SHA256 等）来保证数据存储的安全，并内置各认证中心的 SSL 根证书用来支持 ssl 通信。

不幸的是，微软并没有包含内置、自动的密钥管理，或者是提供一种能够安全存储密钥的方法。

为保证应用的安全，微软会对在 App Hub 上线的应用进行审核。

（1）开发者资质审核

开发者必须首先在微软注册一个 Windows Live ID，然后在 App Hub 进行账户激活，整个注册过程需要对用户身份信息和银行卡信息检查。

（2）应用审核

应用进入 windows phone market 市场需要经过严格的审核流程，包括对开发者身份的 review，应用的功能和内容的检查、证书测试、应用签名等。

当一个应用被提交给 marketplace 时，检测服务会利用 MSIL（Microsoft Intermediate Language）代码分析技术来检测应用所需的能力。另外，所有应用的代码都利用开发者的数字证书进行了签名，证书是由 VeriSign 给每个开发者生成的；没有签名的应用不能在 windows phone 7 上运行。

2.防止敏感数据泄露

本章提到的敏感数据的定义、录入、传输、展示等，请参考《SofaMVC 安全编码规范》第 5 章。

http://doc.alipay.net/pages/viewpage.action?pageId=11255676。

2.1 数据的存储、输入、展示

2.1.1 敏感数据的加密存储

在客户端存储敏感数据（比如登录密码、长时效的 token、数据库连接字符串、交易数据、推送的重要消息）时，需要保证数据的保密性和完整性，防止关键数据被泄露或篡改，整体原则如下。

1) 尽量少的在客户端存储关键数据，将这些数据保存在服务端；如果必须保存，使用强加密算法加密存储（AES、3DES、RSA 等，避免使用弱加密算法 DES 或进行简单编码）。
2) 对称加密算法（AES、3DES 等）使用的密钥不得硬编码在源码中，防止安装包被人逆向反编译后直接获取。
3) 在客户端保存密码或 token 时，需对密码、token 进行 hash 后保存，或者用服务端公钥加密存储。Token 具有一定的时效性，避免使用长时效 token。
4) 用户的敏感数据禁止开放给第三方使用，如有开放需求，需要安全、风控、合规同学审核通过。
5) 使用第 3 方提供的库文件需要经过严格的测试，并检查是否存在恶意行为。

Windows Phone 应用通常采用以下三种方式在手机的 Isolated Storage 上存取数据。

1) 利用 IsolatedStorageSettings 类来存取键值对（可序列化的数据）；
2) 利用 IsolatedStorageFile 类来创建、删除、读写文件和文件夹；
3) 利用 LINQ to SQL 在本地数据库中存取关系型数据。

Windows Phone 中对数据的加解密和散列可以通过 DPAPI（Data Protecction API）来完成。通常是利用 System.Security.Cryptography.（ProtectedData、Rijndael 、AesManaged、TripleDES、SHA1、HMACSHA256、CodePlex-RSACrypto 等类）来实现，其中 ProtectedData、Rijndael 在加密和解密时不需要显式的提供密钥；RSACrypto 可用于非对称的加解密，数字签名及验签。

具体使用时可参考以下示例。

1) ProtectedData-http://msdn.microsoft.com/en-us/library/hh487164%28v=vs.92%29.aspx。
2) Rijndael-http://msdn.microsoft.com/zh-cn/library/system.security.cryptography.rijndael%28v=VS.90%29.aspx。
3) AES-http://robtiffany.com/dont-forget-to-encrypt-your-windows-phone-7-data/。
4) 3DES-http://msdn.microsoft.com/en-us/library/system.security.cryptography.rfc2898derivebytes.aspx。
5) SHA1-http://msdn.microsoft.com/en-us/library/system.security.cryptography.sha1.aspx。
6) HMACSHA256-http://msdn.microsoft.com/en-us/library/system.security.cryptography.hmacsha256%28v=VS.95%29.aspx。
7) RSA-http://www.dustinhorne.com/post/Asymmetric-Encryption-and-Signing-with-RSA-in-Silverlight.aspx。

对数据库中要存储的全部数据都加密时，只要在数据库连接字符串中添加 Password='password'字段（如下代码示例）即可，数据库将被 AES-128 加密算法加密，密码会采用 SHA-256 进行 hash。当然连接字符串本身也需要使用前面所描述算法进行加密和解密，以防止密码泄露。

```
// Create the data context, specify the database file location and password
ToDoDataContext db = new ToDoDataContext ("Data Source='isostore:/ToDo.sdf'; File Mode = read only; Max Database Size = 256; Max Buffer Size = 1024; Password='secure_password'");
```

2.1.2 数据输入与展示

1) 敏感信息录入时使用安全的方式录入（如手机软键盘等），防止信息泄露。
2) 客户端需要对用户输入的数据进行合法性检查并过滤，主要检查内容包括数据的长度、类型、是否包含非法字符（如空格、'、"、<、>、.、-等）。
3) 对输入内容中包含有上述特殊字符的可编码后展示；
4) 对敏感信息的展示采用部分截断展示的方式，比如信用卡号、身份证号等只展示前 6 后 4 位。

关于敏感数据的输入、展示可参考《支付宝敏感信息及日志打印规范》。

2.1.3 日志输出

开发者可以在开发环境中利用 Debug.WriteLine()、WCF+log4net、NLog 等方式输出日志信息用于调试，但在发布上线的安装包中，应该将调试输出关闭，避免在日志中输出账户认证、交易过程中的敏感数据。

2.2 网络传输安全

Windows Phone 支持应用通过 http、socket 的方式与服务端通信。

http 通信时可利用 WebClient、HttpWebRequest/HttpWebResponse 等类来实现数据的同/异步请求和接收，为保证数据传输过程中的安全（包括数据的机密性和完整性），客户端与服务端之间的通信建议采用 https 协议。实现时也非常简单，客户端代码中只需要将连接 URL 中协议部分设置为 https 即可。HttpWebRequest myReq = (HttpWebRequest) WebRequest.Create("https://www.xxx.com/")。

除 https 协议本身提供的身份认证、数据加密、签名的保护机制外，建议对一些重要数据（如用户的密码、交易订单等）进行业务层的数据加密和签名。

由于目前微软官方暂不支持 windows phone 的 SSL socket（SslStream）通信方式，如需采用 socket 方式传输关键业务数据或推送敏感消息，可暂时采用第三方的 SSL 库来实现（http://www.eldos.com/sbb/net-ssl.php）。

2.3 Web 服务安全

Windows Phone 应用可通过 WebClient、HttpWebRequest/HttpWebResponse、Client proxies 等方式访问服务端的基于 http 的 web 服务和数据服务。

当客户端应用连接到一个恶意的 web 服务时，可能存在以下安全风险：

1) 如果客户端采用 http 协议（未对服务端身份做验证和数据加密）传输敏感数据给 web 服务，可能造成信息泄露；
2) 恶意服务返回的响应数据（包括 XML、JSON、RSS、SOAP 等格式的恶意数据）可能导致客户端应用崩溃或者遭受 DoS（比如解析 XML 文档中的恶意嵌套或者复杂的内容模型）。

因此客户端应用与远程服务连接时需要注意：采用 https 通信协议；在用户未明确授权的情况下，避免与不受信任的 web 服务进行通信（比如一些 RSS 订阅）。

另外服务端需要：

1) 对外开放的重要的 web 服务和数据服务，首先需要对调用方的身份进行身份认证，避免服务被未授权访问；
2) 服务端应用做好自身的安全防范措施（包括业务规则限制、时效和地域限制等，或者接入应用层安全防护产品 hummock 或 RDS），防止遭受 CC 攻击（如账户、密码的暴力破解、大批量洗号、恶意注册等）。

3.防止逆向破解

3.1 安装包被逆向破解风险

如果 windows phone 应用在编译时没有做代码混淆处理就直接上线，应用的 xap 包在被恶意攻击者反编译和调试后，可能会暴露应用的数据传输、加密、存储等关键代码或核心业务逻辑，然后被用于进一步的攻击。

3.1.1 反编译过程

将一个.xap 安装包逆向破解获取到源码（C#、VB.net），可以利用现有的工具（比如 Reflector、Windows Phone App Analyzer、JustDecompile 等）来完成，以 Reflector 为例，主要步骤如下。

1) 首先下载要逆向破解的安装包（或者利用 ISETool.exe 将已安装到手机的应用拷贝到电脑中），比如 AlipayClient.xap。
2) 然后利用 winrar、Winzip 等解压缩工具将 AlipayClient.xap 解压缩到任一目录中，会解压出 WMAppManifest.xml、AlipayClient.dll、Paylib.dll、DH.Scrypt.dll 等文件。
3) 打开 Reflector，选择 File->Open Assembly，在弹出的对话框中选中第 2 步中解压出来的.dll 动态库文件，将其导入到 reflector 中。
4) 在左边栏选中要查看的类文件，比如 AlipayClient.dll 中的 AlipayClient.util.HttpManager，即可在右边展示框中查看到该类反编译后的 C#源码，如图 3.1 所示。

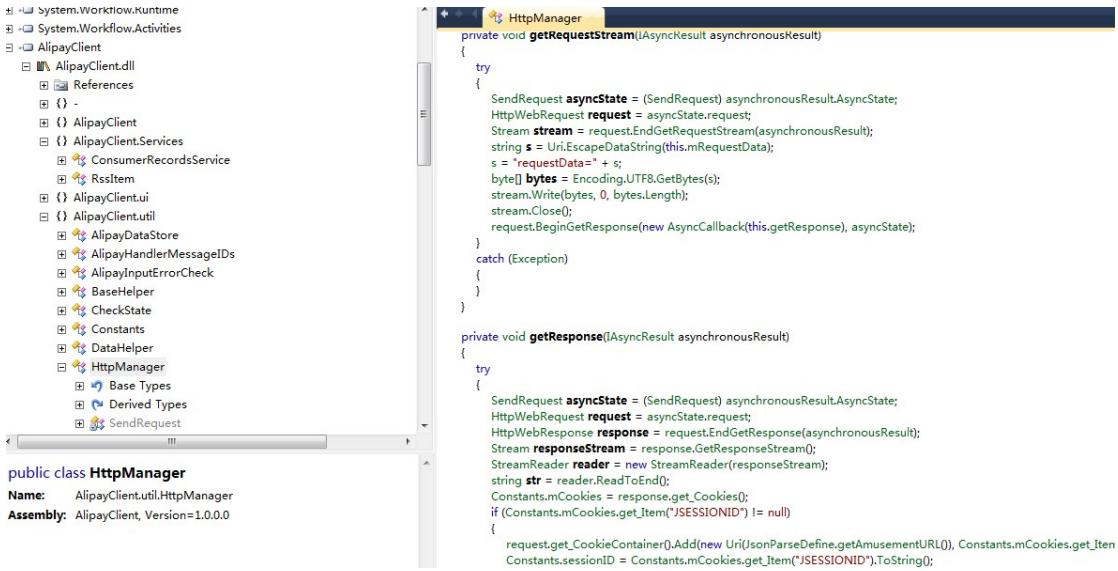

图 3.1 未进行代码混淆过的客户端反编译后的代码展示

3.1.2 调试过程

逆向分析一个 xap 包功能的另外一个方法是分析应用的动态行为，通常是通过对应用调试获取信息。

我们可以利用 XAPspy 工具（http://www.sensepost.com/labs/tools/poc/xapspy）对 windows phone 应用进行调试。

1) 下载解压后运行 XAPspy.exe，点击“Browerse”按钮，选中要调试的 xap 包，如图 3.2 所示。
2) 点击“Deploy”按钮，应用会启动模拟器，然后将 xap 包安装到模拟器中；
3) 在模拟器中运行要调试的应用，进行各种业务操作。此时正在运行的应用会在控制台输出此时调用的方法和变量名（这是因为安装包本身已经被 XAPspy 修改过，并重新打包签名）。
4) 调试完毕后，点击 XDE-Monitor 的“File->Save as”，将整个调试过程中输出的时间、应用包名、方法、变量等信息导出到文件，进行进一步分析。

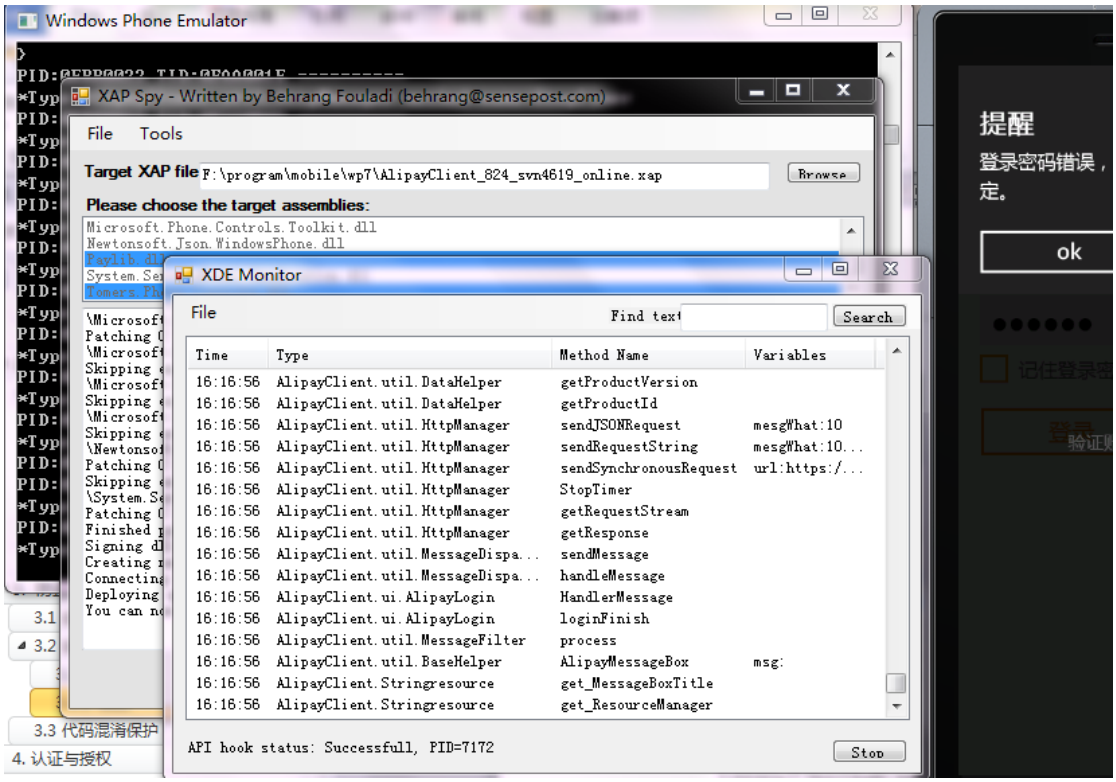

图 3.2 利用 XAPspy 调试 windows phone 客户端

3.2 代码混淆保护

为了增加安装包被逆向破解的难度，在应用发布到 App Hub 之前，需要对应用进行混淆。

代码混淆工具可采用 PreEmptive 公司的 Dotfuscator，该软件在 2011 年 12 月之前是微软和 PreEmptive Solutions 公司签订协议，专门为 Windows Phone 开发者提供的一个代码混淆工具，用户注册和下载地址为 http://www.preemptive.com/windowsphone7.html。

Dotfuscator 主要提供了对 MSIL 中间代码的变量（包括变量/类/方法名）重

命名、无用信息移除、控制流混淆、字符串加密、水印、篡改通知、产品监控及统计等功能。

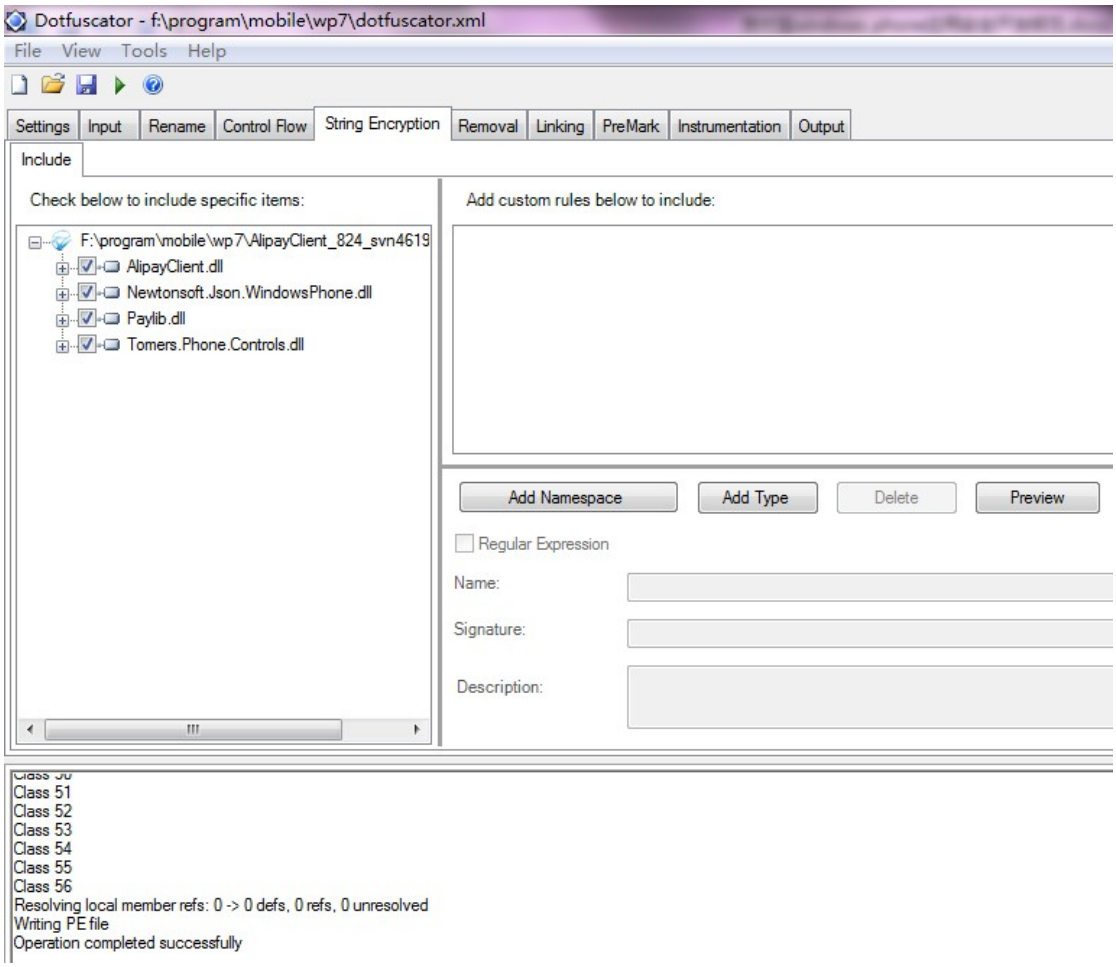

图 3.3 Dotfuscator 混淆器主界面

Dotfuscator 安装完成后，打开软件主界面的 Input 选项卡，导入将要混淆的 xap 包，然后分别在 Rename、Control Flow、String Encryption、Removal 等选项卡中按照 userguide.pdf 第 45 页进行设置，如图 3.3 所示（该用户使用手册可以在安装目录下找到—C:\Program Files\PreEmptive Solutions\Dotfuscator Windows Phone Edition 4.9\docs，或者在线查看，http://www.preemptive.com/images/stories/dotfuscator_documentation/Dotfuscator/webframe.html）。

设置完成后，点击绿色的“Build”按钮，Dotfuscator 就会按照设置进行代码混淆，最终会输出一个混淆过的 xap 包。

我们可以利用 3.2.1 节使用的 Reflector 重新对混淆过的 xap 包进行逆向反编译，可以看到混淆后的代码如图 3.4 所示。

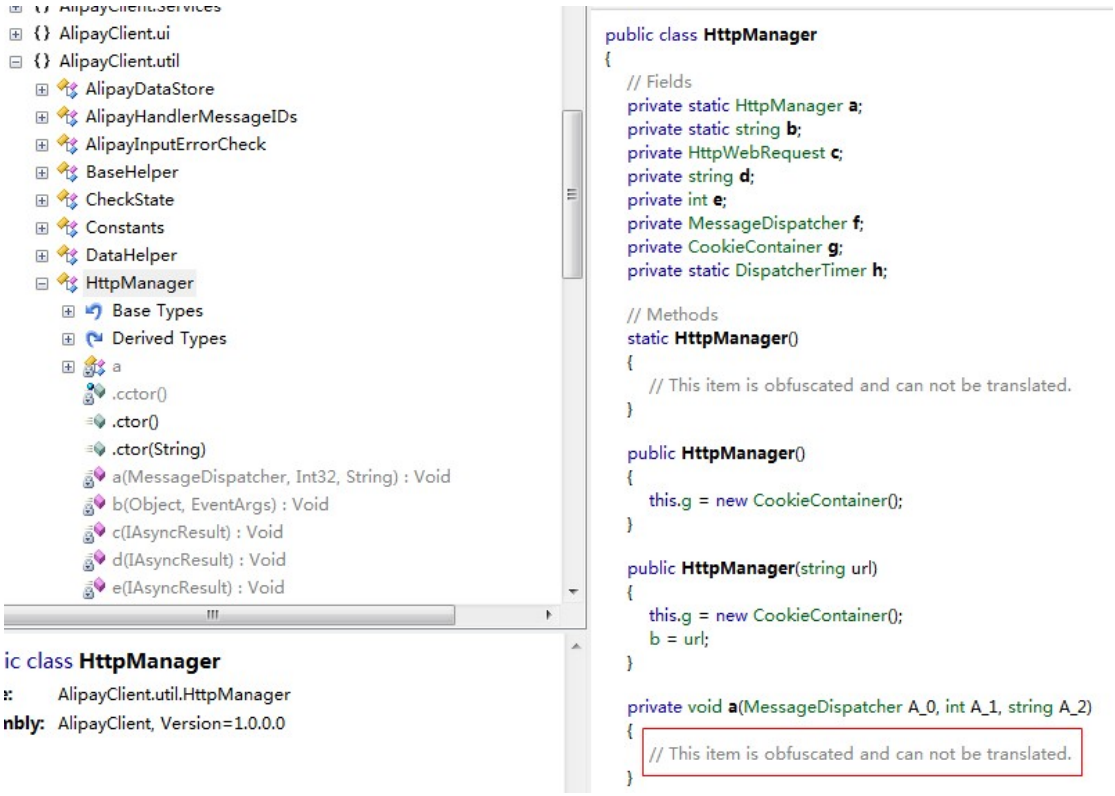

图 3.4 反编译 Dotfuscator 混淆过的 xap 包结果展示

请测试同学对代码混淆过的应用进行功能测试，避免因代码混淆导致正常功能不可用。

4.应用的发布审核

wp7 应用的发布主要是通过 App Store 渠道。开发者 ID 一旦被盗取，攻击者将可以替换我们发布的原始文件为其他恶意应用。

1) 手机客户端软件的发布，需要有专人负责保管开发者 ID（backup）。
2) ID 强密码。
3) 为了防止应用程序被篡改，禁止使用第三方 ID 发布应用程序。
4) 应用发布前需抄送安全部门（白开心）一份进行安全测试和审核。

5.参考资料及工具

以下列出本文档用到的一些参考资料及工具软件。

1) Reflector－http://www.reflector.net/
2) donetReactor－http://www.eziriz.com/dotnet_reactor.htm
3) Crypto Obfuscator For .Net－
   http://www.ssware.com/cryptoobfuscator/obfuscator-net.htm
4) dotPeek1.0－ http://www.jetbrains.com/decompiler/
5) JustDecompile－http://www.telerik.com/products/decompiler.aspx

6) Crypto Obfuscator－
http://www.ssware.com/cryptoobfuscator/obfuscator-net.htm
7) Silverlight security－
http://msdn.microsoft.com/en-us/library/cc972657(v=vs.95).aspx
8) Network－
http://msdn.microsoft.com/en-us/library/gg521151%28v=vs.92%29.aspx
9) Isolated Storage－
http://msdn.microsoft.com/en-us/library/bdts8hk0%28v=vs.95%29.aspx
10) SSLStream－
http://msdn.microsoft.com/en-us/library/system.net.security.sslstream.aspx
11) Code Obfuscation－
http://www.martani.net/2010/12/windows-phone-7-why-you-should.html