目录

[客户端应用测试 1](#_Toc523306621)

[1.1 二进制程序保护 1](#_Toc523306622)

[1.2 程序完整性校验 2](#_Toc523306623)

[1.3 符号表信息泄露 4](#_Toc523306624)

[1.4 程序数据存储安全 5](#_Toc523306625)

[1.5 动态调试检测 7](#_Toc523306626)

[服务器端应用测试 8](#_Toc523306627)

[1.6 证书强校验检测 8](#_Toc523306628)

[1.7 测试说明 11](#_Toc523306629)

客户端应用测试

本次测试环境：

客户端：luckin coffee(v1.8.0).ipa

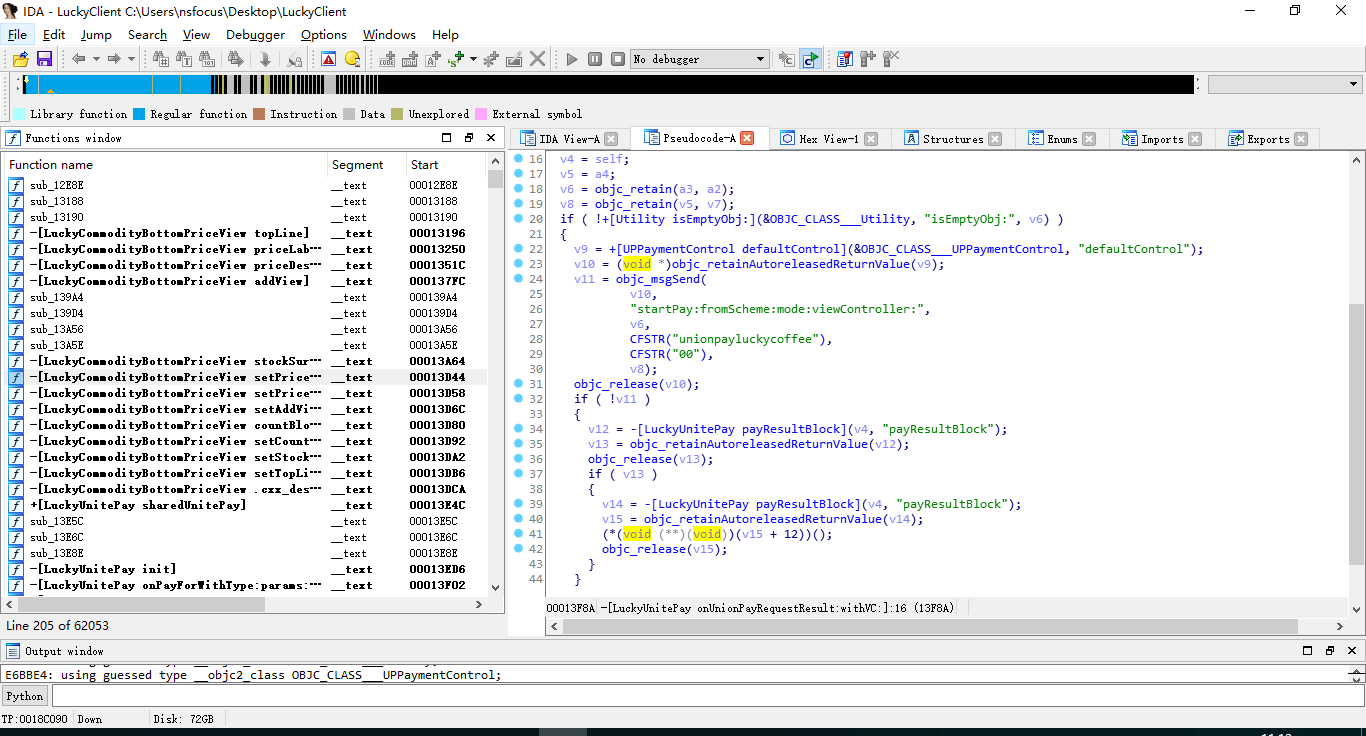
版本：V1.8.0

* 1. 二进制程序保护

|  |  |
| --- | --- |
| 涉及组件 | 无 |
| 漏洞级别 | 中危（S1） |
| 漏洞类型 | 信息泄露 |
| 影响 | 编译好的可执行文件可以容易地被Hopper或者IDA这类软件反编译。在没有进行函数名和逻辑混淆的时候，则容易被反汇编得到关键逻辑进而进行破解。 |
| 修复方案 | 利用iOS ClassGuard可以进行类名混淆，利用LLVM-obfuscator进行相关的逻辑混淆。 |

**验证过程：**

用IDA pro对程序进行反编译，发现其中存在大量可读性强的函数名称和函数逻辑，下图可以看到清晰的函数名和支付信息。



1可见清晰的函数名和支付信息

**测试结果：**

中风险。

**安全建议：**

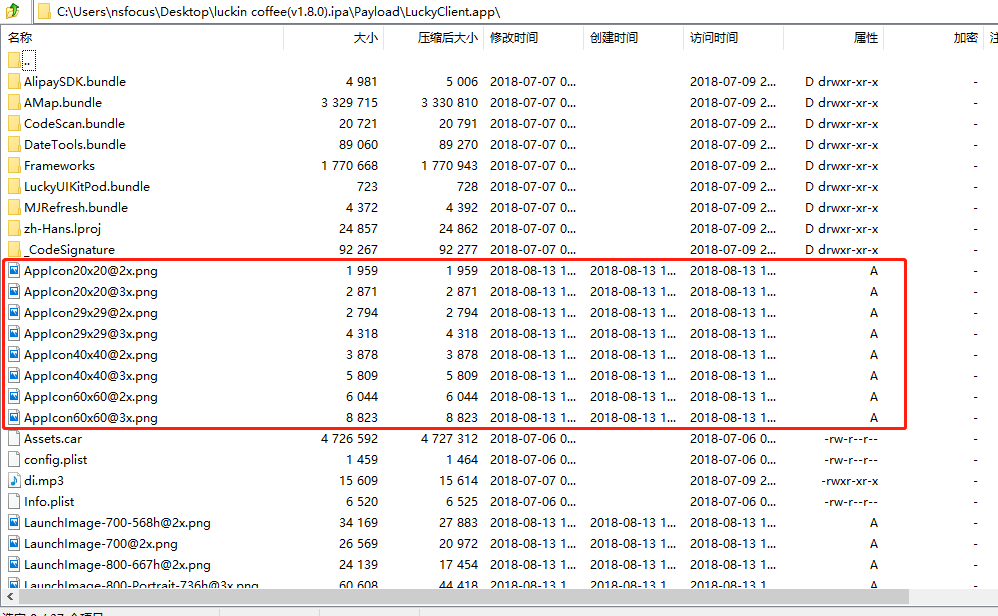
利用iOS ClassGuard可以进行类名混淆，利用LLVM-obfuscator进行相关的逻辑混淆。LLVM参数如下：

* Instructions substitution (–mllvm –sub)
* Bogus control flow (–mllvm –bcf)
* Control flow flattening (–mllvm –fla)
  1. 程序完整性校验

|  |  |
| --- | --- |
| 涉及组件 | 无 |
| 漏洞级别 | 低危（S1） |
| 漏洞类型 | 完整性校验 |
| 影响 | 测试客户端是否对自身完整性校验。客户端程序如果没有自校验机制的话，攻击者有可能通过篡改客户端程序，显示钓鱼信息欺骗用户，窃取用户的隐私信息。 |
| 修复方案 | 1.可以通过检测info.plist中是否存在SignerIdentity判断是否被篡改；  2.若要进行文件对比需要联网，并需要在发布之后得到新的时间/哈希并进行录入，在录入之前应打开debug开关，保证用户可以正常运行，若采用固定的值，则会导致程序无法运行（苹果修改图片和程序）；  3.可以通过检测cryptid的值来检测是否被篡改，篡改过cryptid的值为0 |

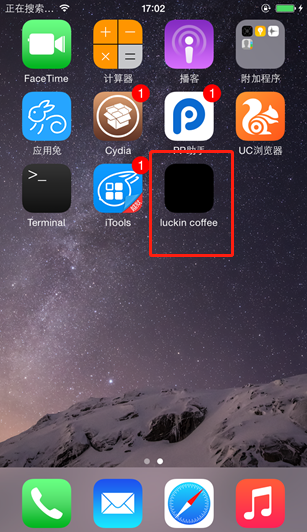
**验证过程：**

1. 用压缩包格式打开luckin coffee(v1.8.0).ipa，编辑修改其中的图标图片。



修改安装包中的图标

1. 修改后，安装该ipa包，发现安装成功，且能成功运行，但是应用图标无法正常显示。



3图标无法正常显示

**测试结果：**

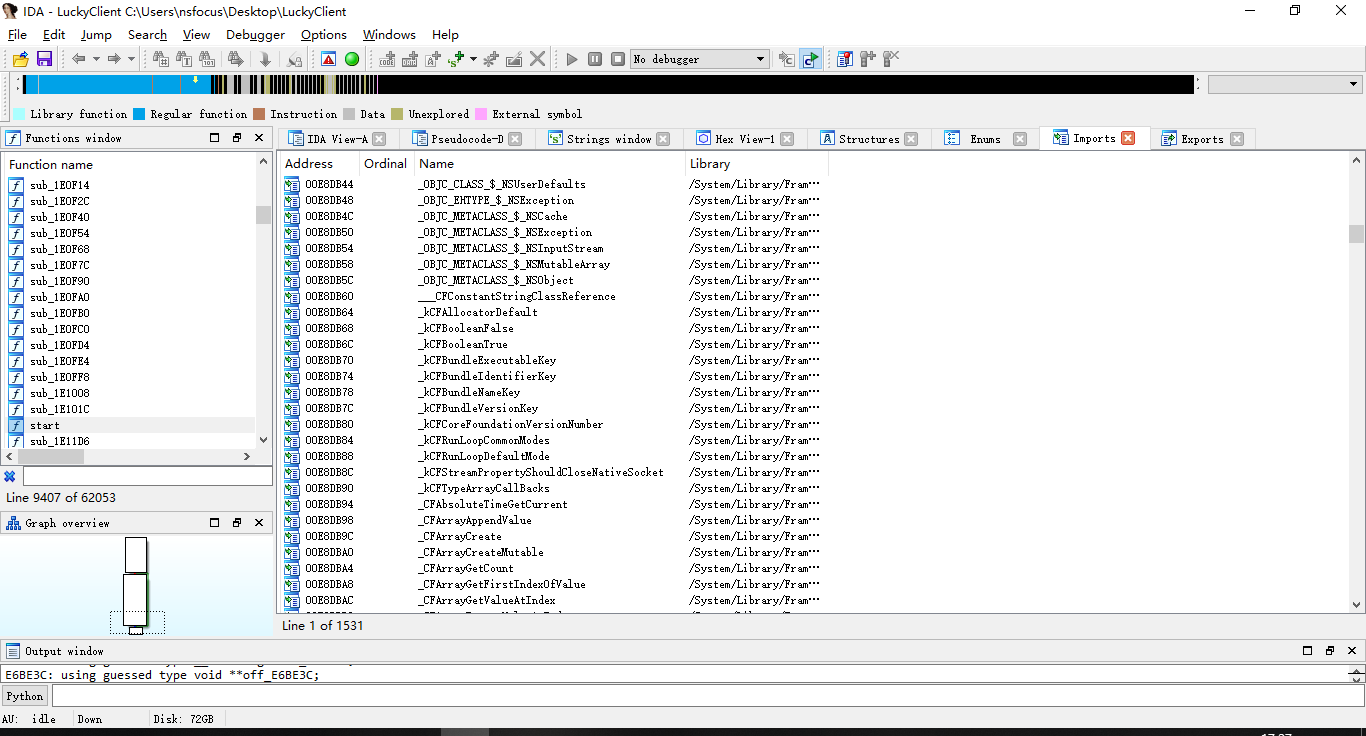
低风险。

* 1. 符号表信息泄露

|  |  |
| --- | --- |
| 涉及组件 | 无 |
| 漏洞级别 | 低危（S1） |
| 漏洞类型 | 信息泄露 |
| 影响 | 符号表是逆向工程中的兵家必争之地。有效的符号表能够极大地方便反汇编和逆向工作的进行，在发布应用的时候未清除掉相关的符号则会辅助逆向分析。  如果客户端与服务器之间的通信加密协议实现不当，攻击者将有机会对当前网络环境中其他合法用户的通信内容进行窃听甚至篡改。 |
| 修复方案 | 1.使用strip命令，但是只能去除部分表，效果如下，还是存在部分符号表，但是大部分敏感函数名已经被去除：  2.在编译时候去除相关的符号表 |

**验证过程：**

用IDA pro检查程序的符号表，发现存在较多信息。



4符号表中存在较多信息

**测试结果：**

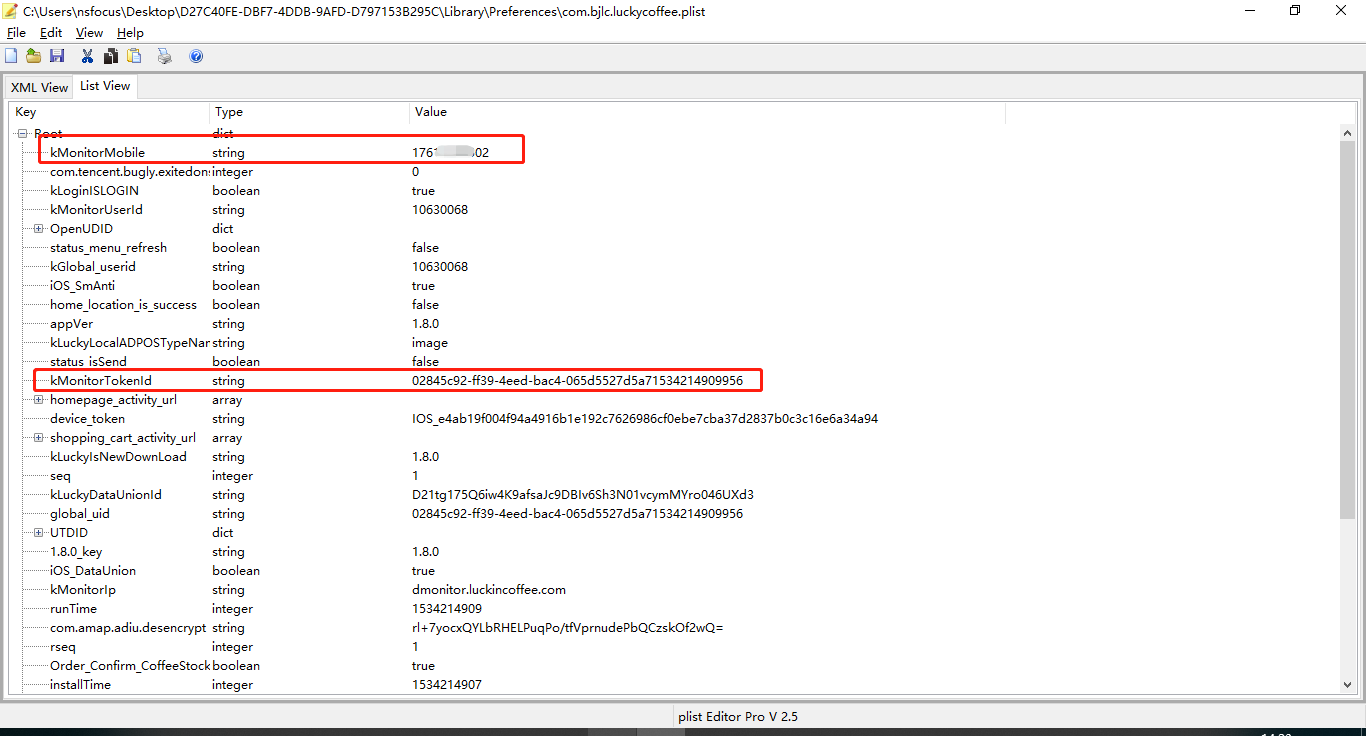
低风险。

* 1. 程序数据存储安全

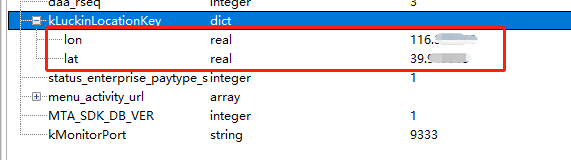
|  |  |
| --- | --- |
| 涉及组件 | 无 |
| 漏洞级别 | 低危（S1） |
| 漏洞类型 | 信息泄露 |
| 影响 | 由于设备可以进行越狱，且越狱之后数据不会丢失，则导致本地存储敏感信息将造成风险。 |
| 修复方案 | 对敏感信息文件进行独立加密，并解密读取即可 |

**验证过程：**

先运行程序，进行大量操作来填充数据；进入设备文件系统，在目录：/var/mobile/Containers/Data/Application/<UUID>中查看文件，发现在/Library/Preferences/com.bjlc.luckycoffee.plist中明文存储了用户的手机号、TokenId、地理位置等敏感信息。



7明文存储手机号和tokenid



8明文存储用户的地理位置信息

**测试结果：**

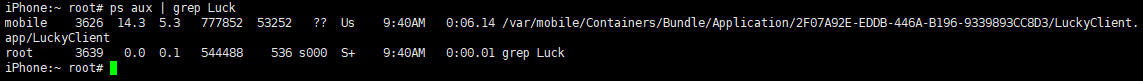
低风险。

* 1. 动态调试检测

|  |  |
| --- | --- |
| 涉及组件 | 无 |
| 漏洞级别 | 低危（S1） |
| 漏洞类型 | 中间人 |
| 影响 | 程序如果没有反动态调试，攻击者能够使用GDB、LLDB对程序进行动态调试，修改运行时数据，影响程序正常运行甚至获取程序逻辑。 |
| 修复方案 | 使用ptrace阻止附加。 |

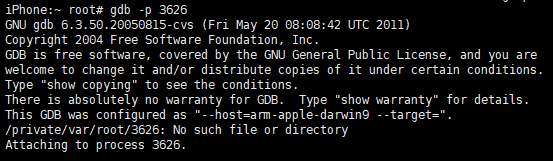
**验证过程：**

1. 打开luckin coffee Ios客户端程序后，用SSH连接iPhone,运行ps aux | grep （程序名称），获取进程pid；



获取进程pid

1. 运行gdb –p （进程pid），成功附加该进程；



gdb附加进程

1. 用gdb的info mach-regions命令读取进程内存信息。



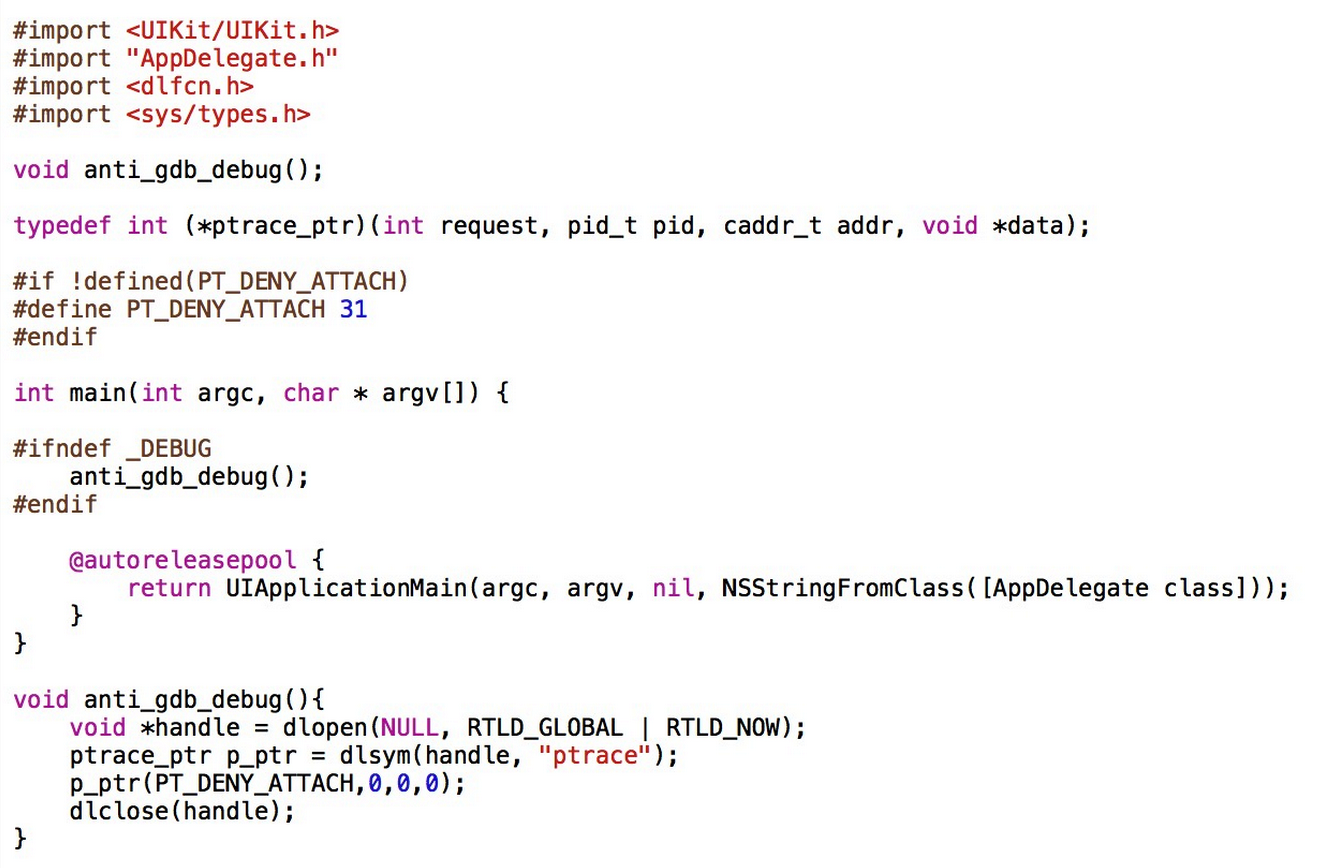
12读取进程内存信息

**测试结果：**

低风险。

**安全建议：**

使用ptrace阻止附加。



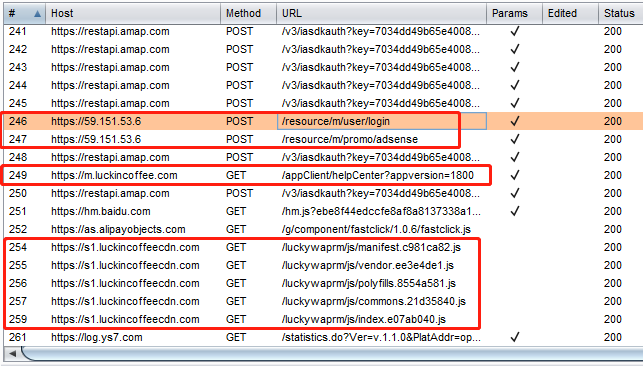
服务器端应用测试

* 1. 证书强校验检测

|  |  |
| --- | --- |
| 涉及组件 | 无 |
| 漏洞级别 | 低危（S1） |
| 漏洞类型 | 中间人 |
| 影响 | 客户端可能存在忽略服务端证书校验的安全漏洞，对服务器没有校验或者没有在校验错误时候进行错误提示等。导致攻击者可通过伪造证书等手法进行攻击获取。 |
| 修复方案 | 1.单向证书校验  2.双向校验方案 |

**验证过程：**

安装证书，关闭SSL Kill Switch，设置代理并进行抓包，发现程序无报错，可正常抓包。



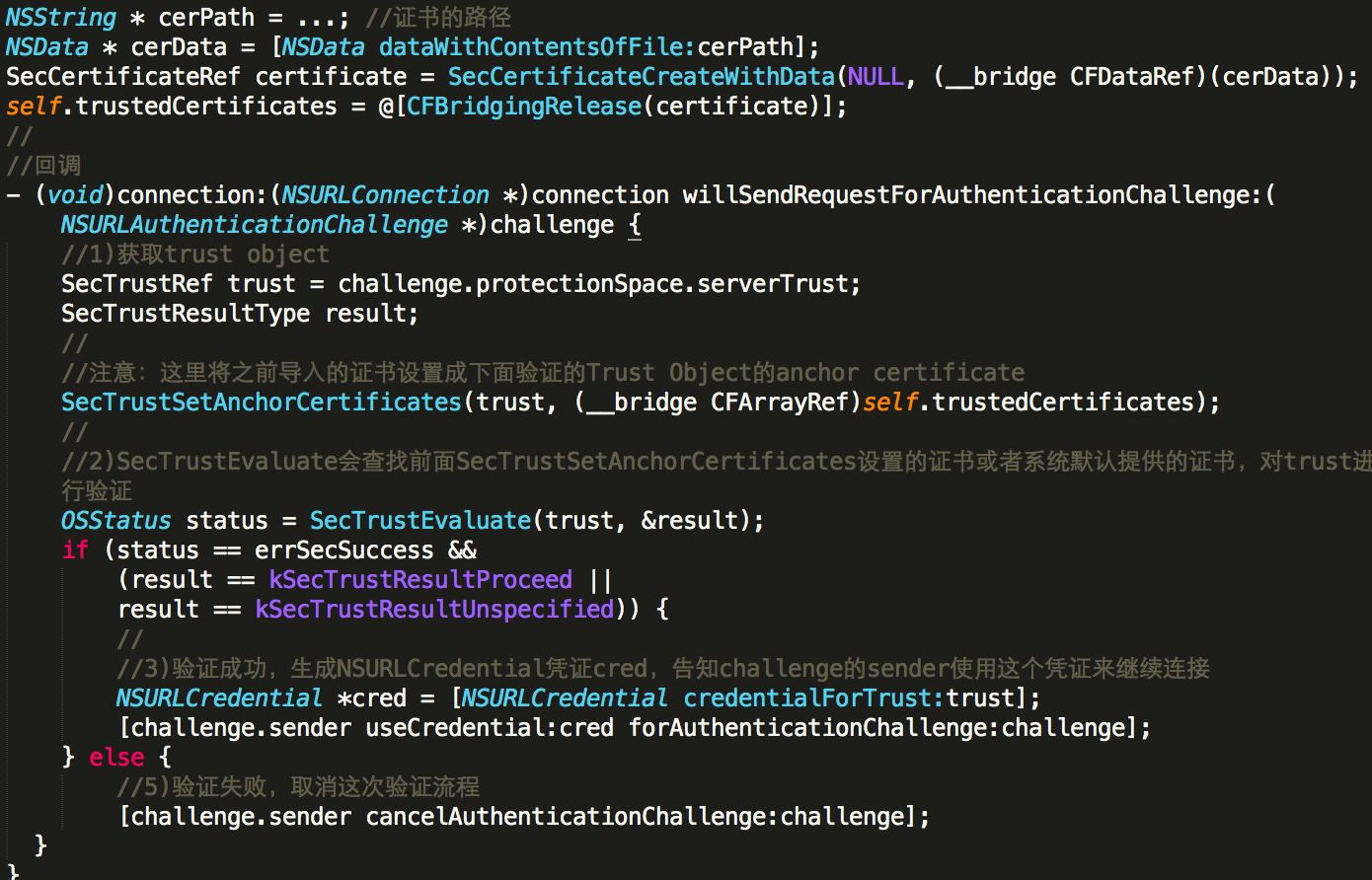
20正常抓取https数据包

**测试结果：**

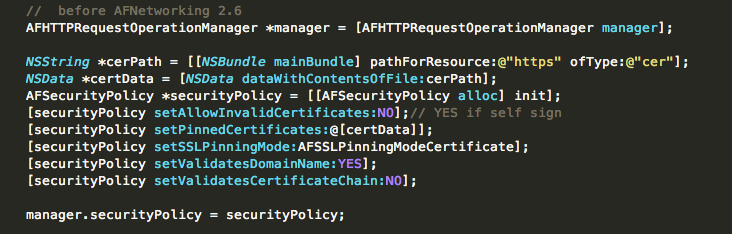
低风险。

**修复方案：**

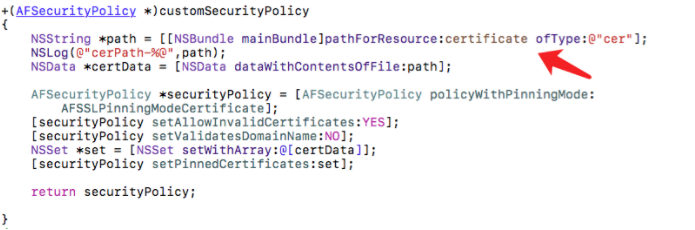
* 方案一、单向证书校验（NSURLConnection为例）



* 方案二、单向证书校验（AFNetworking为例）



对于AFNetworking3.0及之后：



需要在请求方法中进行调用：

[manager setSecurityPolicy:[self customSecurityPolicy]];

* 方案三、双向校验方案（以tomcat为例）

只是需要在Tomcat配置中加入校验

<!-- root是根证书（二级根证书也可以）、server.p12是服务器证书，两者可以一样 -->

<Connector port="8443" protocol="org.apache.coyote.http11.Http11Protocol" SSLEnabled="true"

maxThreads="150" scheme="https" schemeecure="true"

keystoreType="PKCS12" keystoreFile="server.p12" keystorePass="1234"

truststoreType="JKS" truststoreFile="root.jks" truststorePass="password"

clientAuth="false" sslProtocol="TLS" />

* 1. 测试说明

输入验证和客户服务功能（对应域名http://multimedia.gateway.ucarinc.com:2008）同样存在输入未校验的问题，该问题在Android客户端测试报告中已经有详细体现，这里不再赘述。