IOS客户端通用测试

IOS客户端静态安全

2.1 程序完整性校验

测试项描述

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。

结果判定

—修改图片后正常运行则存在风险；

—若修改图片后无法显示修改后的图片，需要重新确认图片是否修改正确，如果真的修改之后正常运行且不显示修改后图片，则为安全；

—若修改图片后程序运行直接闪退，则应该是改版本iOS系统校验了资源。

风险评级

视泄露数据的情况而定

低风险——非由于苹果渠道控制，且代码签名防护，可利用性小

安全建议

针对内部应用:

- 通过对CodeResources读取资源文件原始hash，和当前hash进行对比，判断是否经过篡改，被篡改过的文件应从服务器重新请求资源文件进行替换；
- 可以通过检测info.plist中是否存在SianerIdentity判断是否被算改；
- 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

安全建议

针对APP Store 发布应用:

- 可以通过检测infoplist中是否存在SianerIdentity判断是否被篡改;
- 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);
- 可以通过检测的值来检测是否被篡改，篡改过cryptid的值为0

IOS客户端静态安全

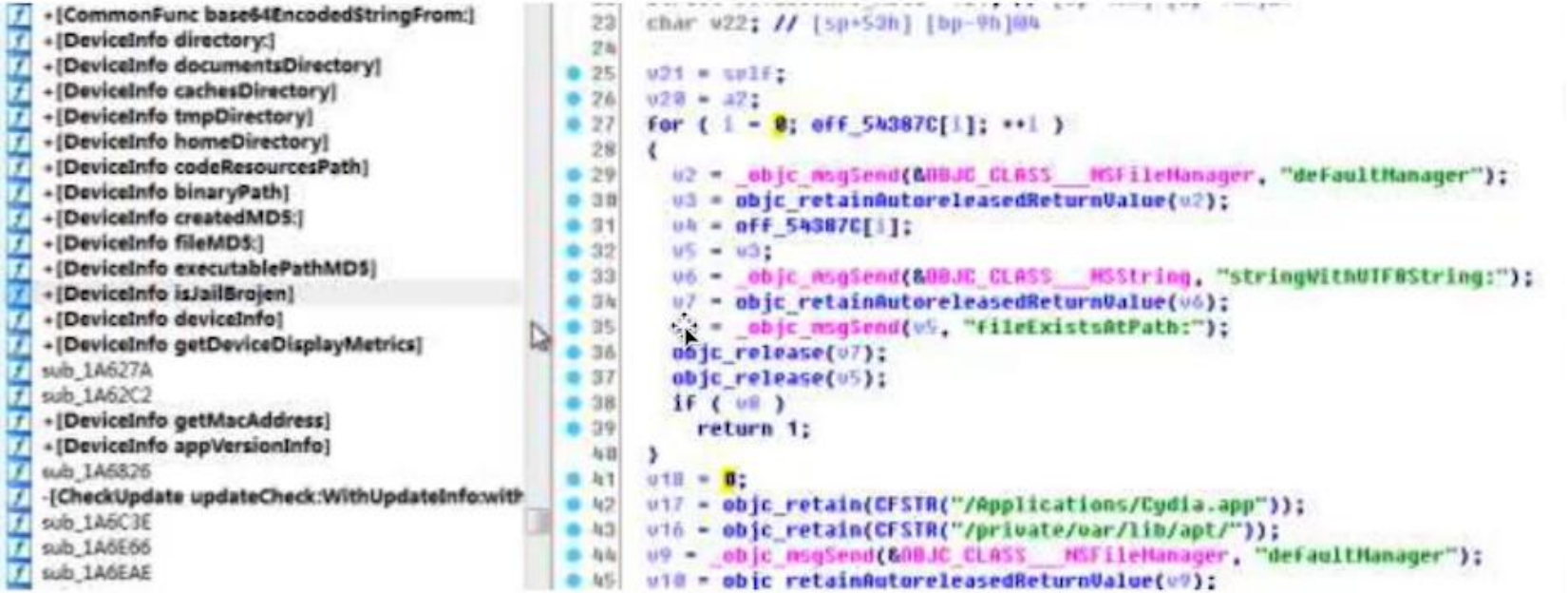
2.2 二进制程序保护

测试项描述

对程序进行反编译，检查其是否存在大量可读性强的函数名称和可见函数逻辑。

结果判定

如图所示，可以看到清晰的函数名和越狱检测方法:



风险评级

中风险—— 未进行类名混淆以及逻辑混淆

低风险——进行了大部分类名混淆，未进行逻辑混淆

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)

IOS客户端静态安全

2.3 符号表信息泄露

测试项描述

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。

测试步骤

1.对ipa文件进行解压缩，获得可执行程序

2.利用nm命令获取符号列表

结果判定

不安全，低风险。符号表中存在较多信息。

风险评级

低风险——只要有Symbols就是低风险

安全建议

在编译时候去除相关的符号表

谢谢观赏