瑞星之剑逆向分析

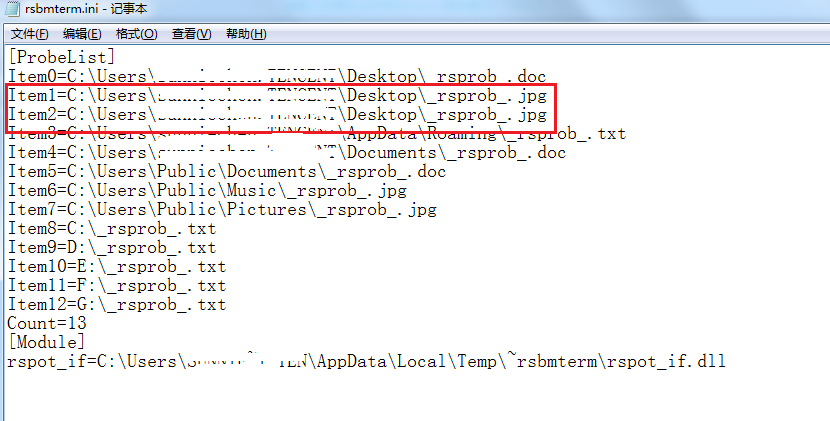
1. **官网说明：**

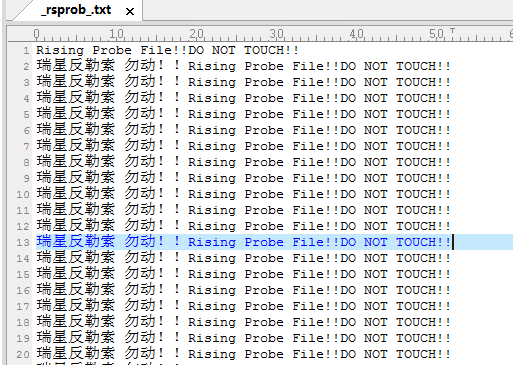


1. **运行过程：**
2. 释放文件rsbmterm.ini至系统根目录，在固定路径内创建隐藏文件\_rsprob\_.txt、\_rsprob\_.jpg、\_rsprob\_.doc

Rsbmterm.ini配置路径包括desktop、appdata\roaming、documets、music、pictures，及各个磁盘根目录。

如下图：





rsprob.txt、rsbmterm.ini 均有自保护，禁止修改及删除，如有进程尝试修改\删除该文件，则弹出告警：



1. 检测是否存在Rising Anti Blackmail Service服务，如无则注册该服务

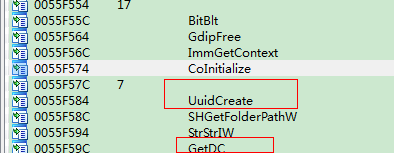


1. rsbmterm自带更新功能，运行后先检测是否有更新，并释放资源文件
2. RSPOT\_IF.DLL到临时目录
3. RSNDISPOT(-X64).SYS到%system32%\driver
4. RSPORT(-X64).sys
5. **模块分析**  
   **1）RSPOT\_IF.DLL**

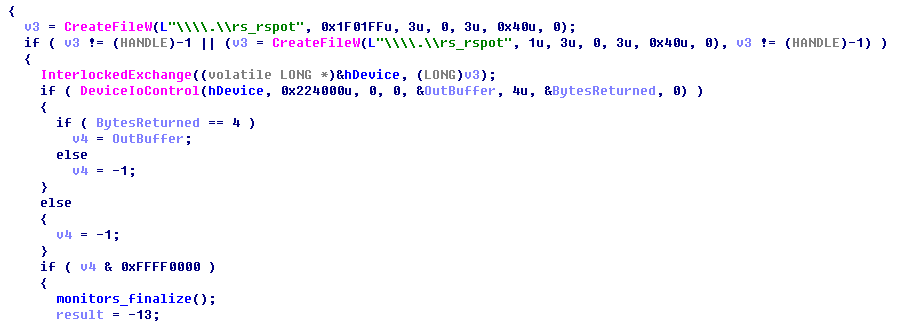
该模块集成了驱动加载和监控规则初始化更新的逻辑。

这块逻辑比较多，大概就是加载驱动做内核监控和监控规则的获取更新，推测是需要联网验证UUID来下发基于机器学习的规则，这个程序的逻辑不是重点规则是核心内容；

初始运行基本没什么监控规则，可能是通过机器生成的UUID来获取更新服务器端的规则更新。

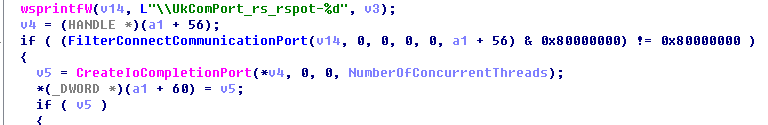
  
**详细逻辑：**

monitors\_init初始化加载驱动

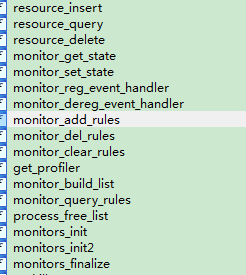


Sub\_100013F0

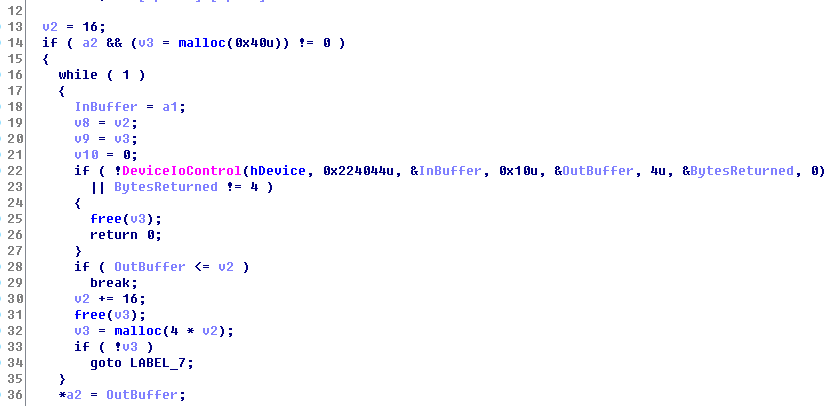
创建IoCompletePort做内核监控



**监控规则的更新**



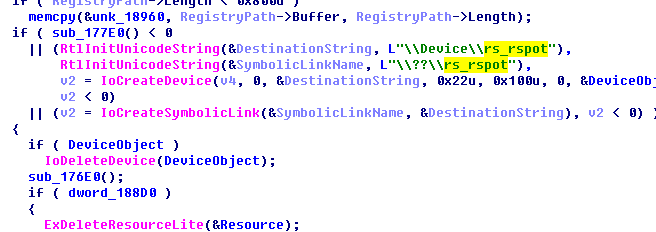
monitor\_build\_list



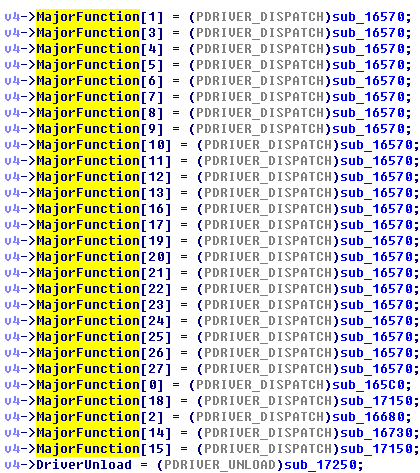
规则建立的逻辑是下发给驱动由驱动来负责完成，add和del等等都是通过DeviceIoControl下发命令字给由驱动完成

1. **RSPOT.sys驱动**

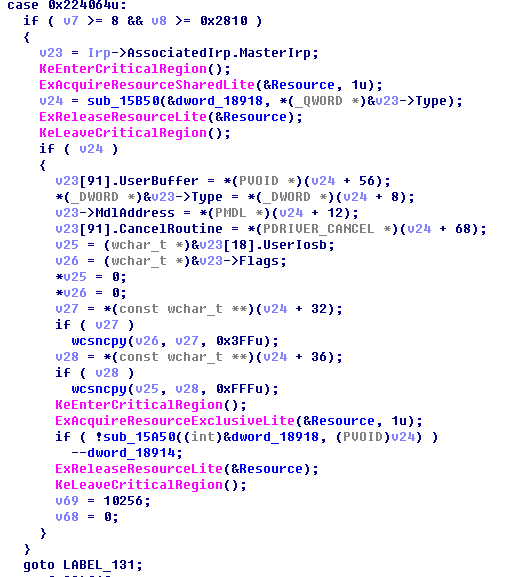
**创建设备加载驱动**



Dll下发的命令字实现函数在这里负责



主要的规则检测逻辑MajorFunction[14]有检测逻辑的实现，比如



1. **RSNDISPOT.SYS**

比较简单，就是做了下进程和线程加载的监控

**模块分析结论：**

参考官方工具说明：基于机器学习的生成特征内存指令检测。

但本次分析过程，**并未看到基于机器学习规则的创建过程和data文件**，程序里的比较关键的函数monitor\_build\_list也是更新IoCotrol命令字，**应该就是普通的基于IoControl命令字的驱动防护。**