018--微信抢红包

- 第一天： 定位消息管理者
  - 动态调试：
    - 1、定位消息界面的控制器，通过Cycript打印当前控制器，定位到：BaseMsgContentViewController
    - 2、HOOK该控制器所有方法
      - 利用 logify.pl 脚本创建XM文件。原理就是HOOK每一个方法，打印调用者、方法名称、方法参数。
      - 删除不必要的方法，编译通过。
      - 通过发送消息，找到处理消息的相关方法！
    - 3、定位到消息接受方法（不止一个，找到接近的。）
      - addMessageNode: layout: addMoreMsg: （以下简称addMessageNode）
        - 该方法有可能是专门处理消息的。
        - 通过参数分析和消息有关
        - 根据需求。利用该方法找到消息管理者。
    - 4、查看addMessageNode函数调用栈
      - 下断点
        - 快速定位方法地址：通过LLDB，methods指令找到该方法的物理地址（每次运行都不一样）
      - 函数调用栈：由于没有恢复符号表的函数调用栈地址不准确
        - lldb查看
          - 通过 dis -s 查看汇编代码（该地址在函数内部），定位到改函数的真实地址。
        - 反汇编工具查看
          - 通过ida 或 hopper 将物理地址转成文件偏移地址：文件偏移地址 = 物理地址 - ASLR
          - 通过文件偏移地址，找到所属函数。快速定位到函数地址。
        - 恢复符号表重签名应用
          - 利用工具 restore-symbol 恢复MachO文件（砸壳后的）的符号表。
          - 利用MonkeyAPP重签名应用。
          - 直接断住addMessageNode，查看函数调用栈。
            - [BaseMsgContentLogicController DidAddMsg:]
            - [RoomContentLogicController DidAddMsg:]
            - [BaseMsgContentLogicController OnAddMsg:MsgWrap:]
            - [CMessageMgr MainThreadNotifyToExt:]
- 第二天：找到红包代码
  - 找到最适合抢红包的地方。
    - hook调试：
      - hook下面4个方法。

        - [BaseMsgContentLogicController DidAddMsg:]
        - [RoomContentLogicController DidAddMsg:]
        - [BaseMsgContentLogicController OnAddMsg:MsgWrap:]
        - [CMessageMgr MainThreadNotifyToExt:]
      - 找到MainThreadNotifyToExt
        - 在不需要进入聊天界面，只要收到新消息就会触发。
        - 但是调用次数太平凡。
        - 通过分析，该方法属于CMessageMgr对象。
    - 分析CMessageMgr
      - HOOK该对象的所有方法。
      - 收到信消息就来，过滤调用多次的方法。并且拥有相关消息的参数。
      - 最终定位onNewSyncAddMessage:方法
- 动态调试找到抢红包方法
  - 定位开红包界面。找到开红包按钮。并定位方法
- 静态分析抢红包方法
  - 分析 WCRedEnvelopesReceive OnOpenRedEnvelops】
  - 内部调用 WCRedEnvelopesReceiveHomeViewOpenRedEnvelopes
  - 分析　WCRedEnvelopesReceiveHomeViewOpenRedEnvelopes
  - 利用F5 伪代码提高分析效率。