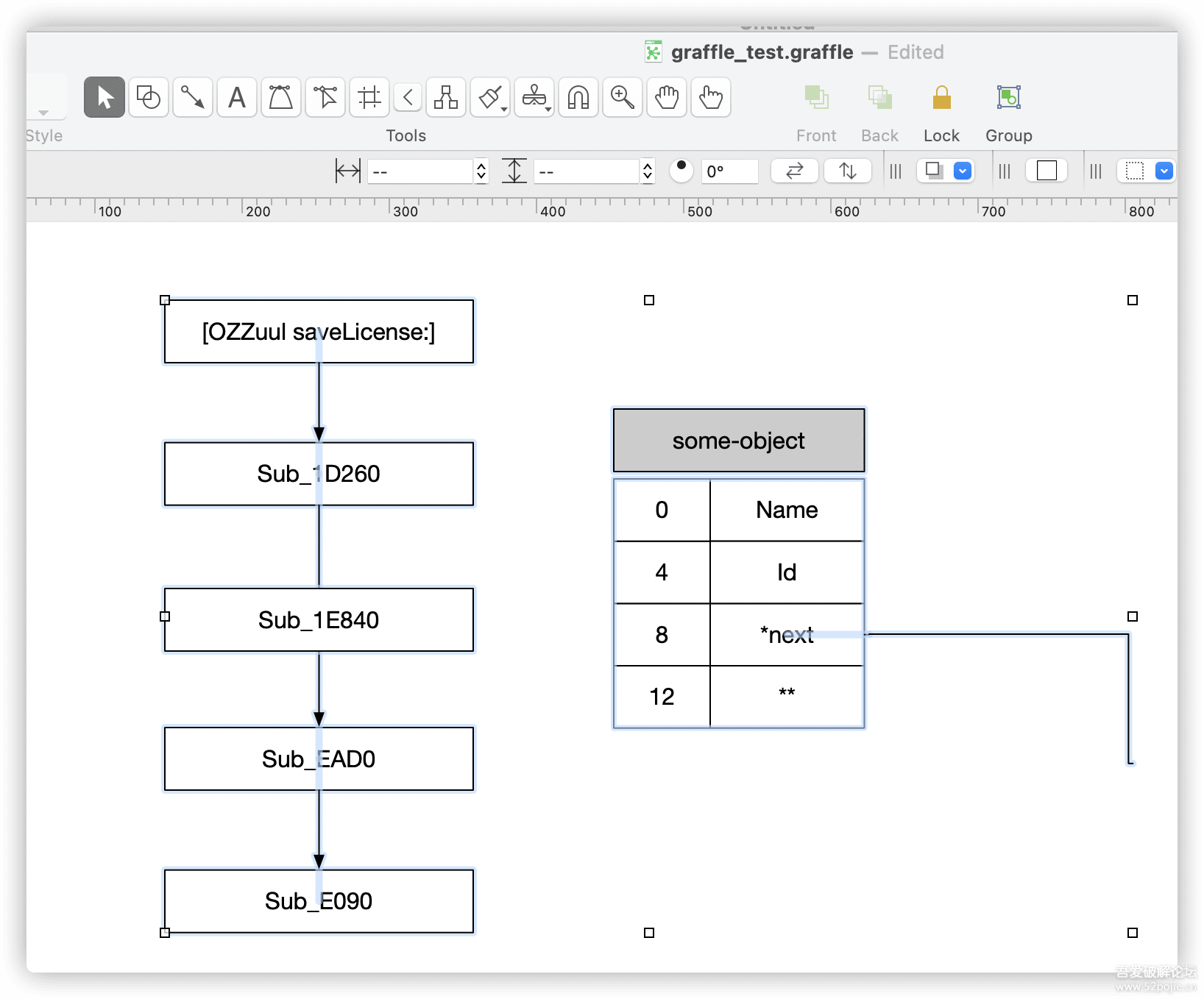
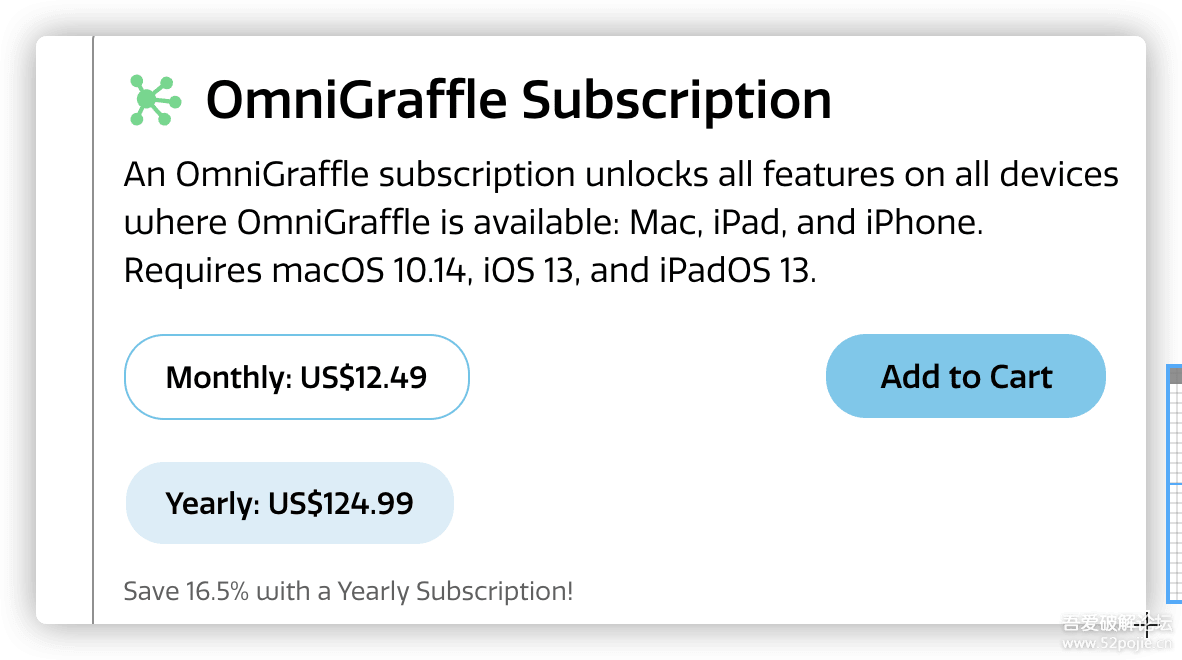
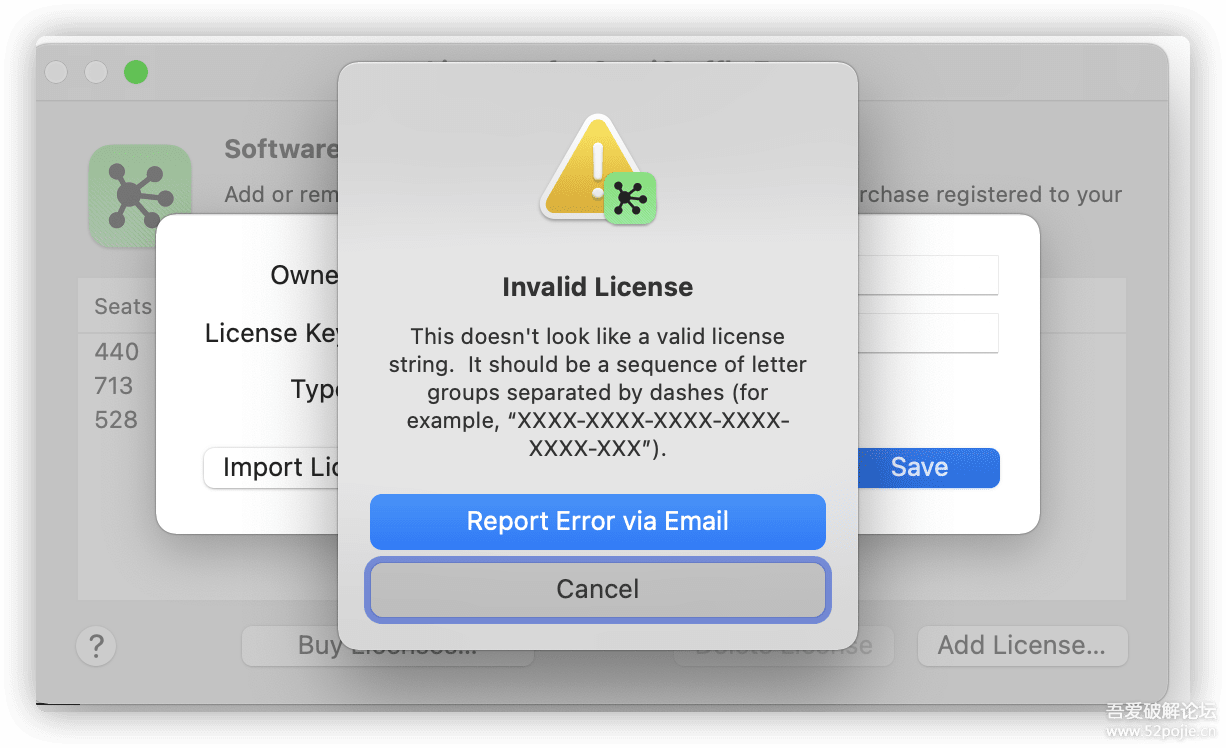
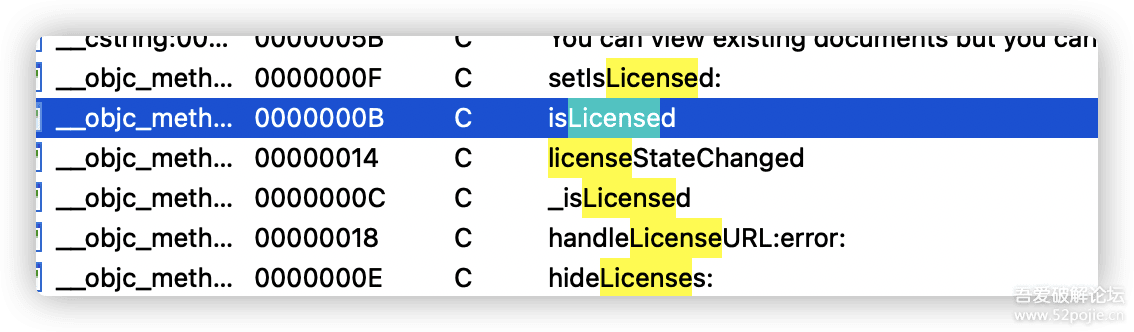
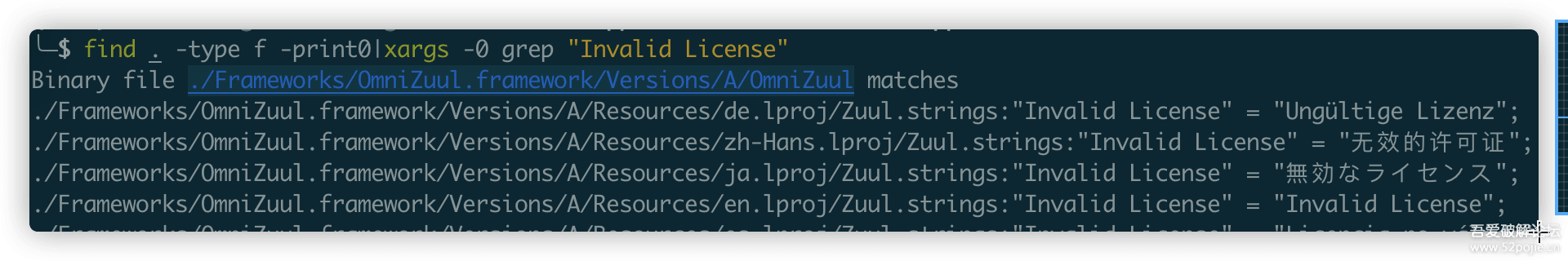
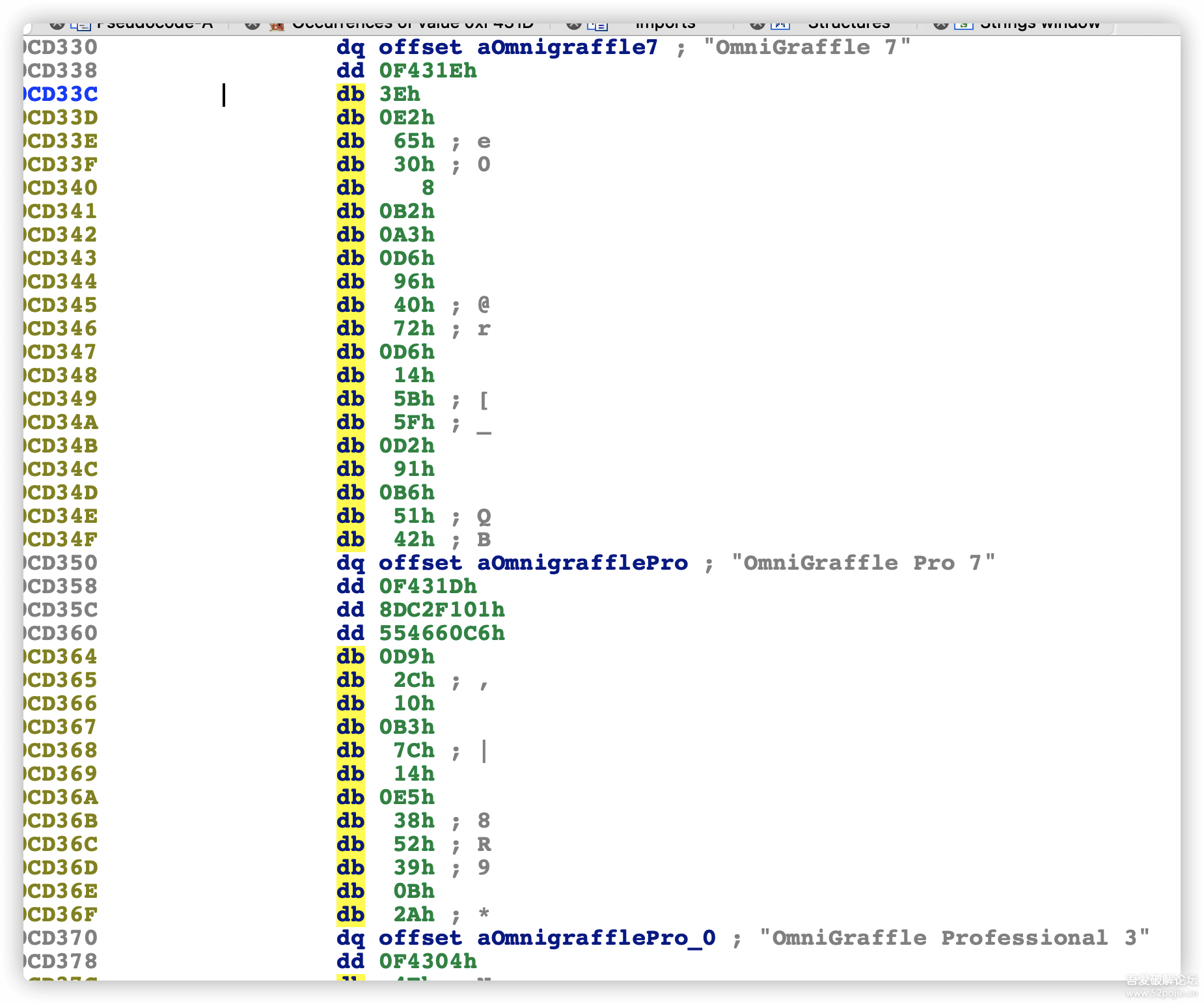
**逆向OmniGraffle并编写注册机[Python](附带注册机源码)**

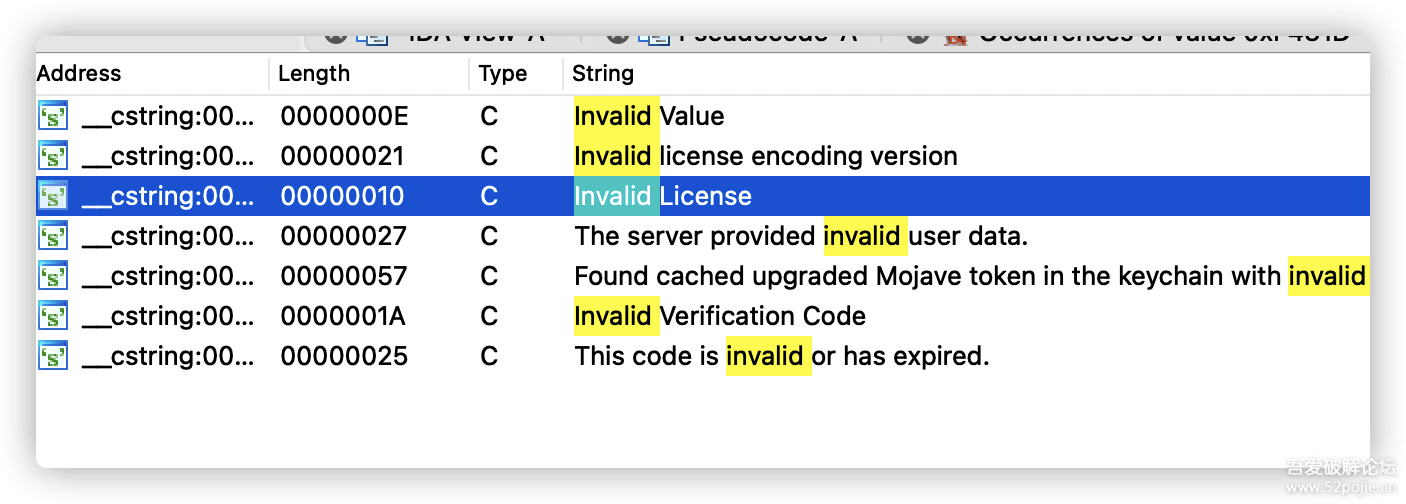
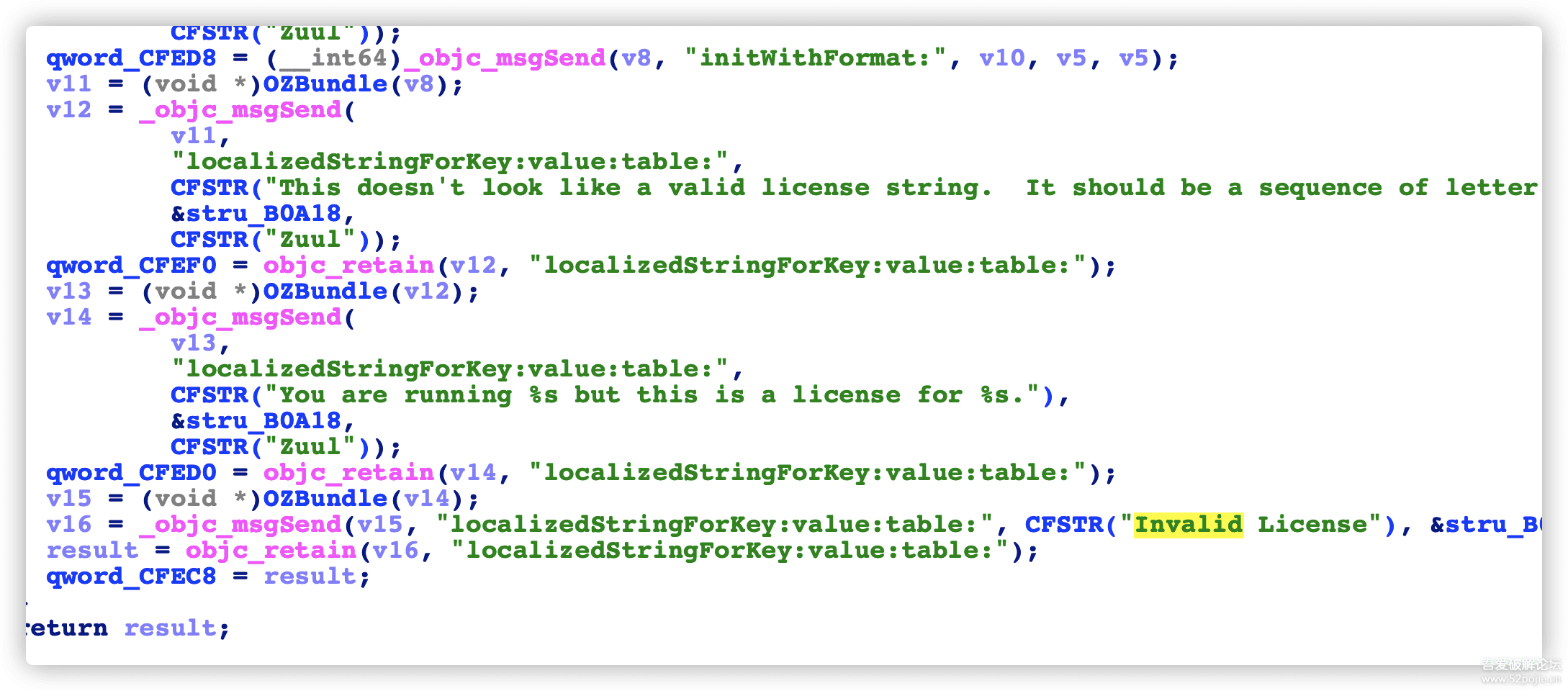
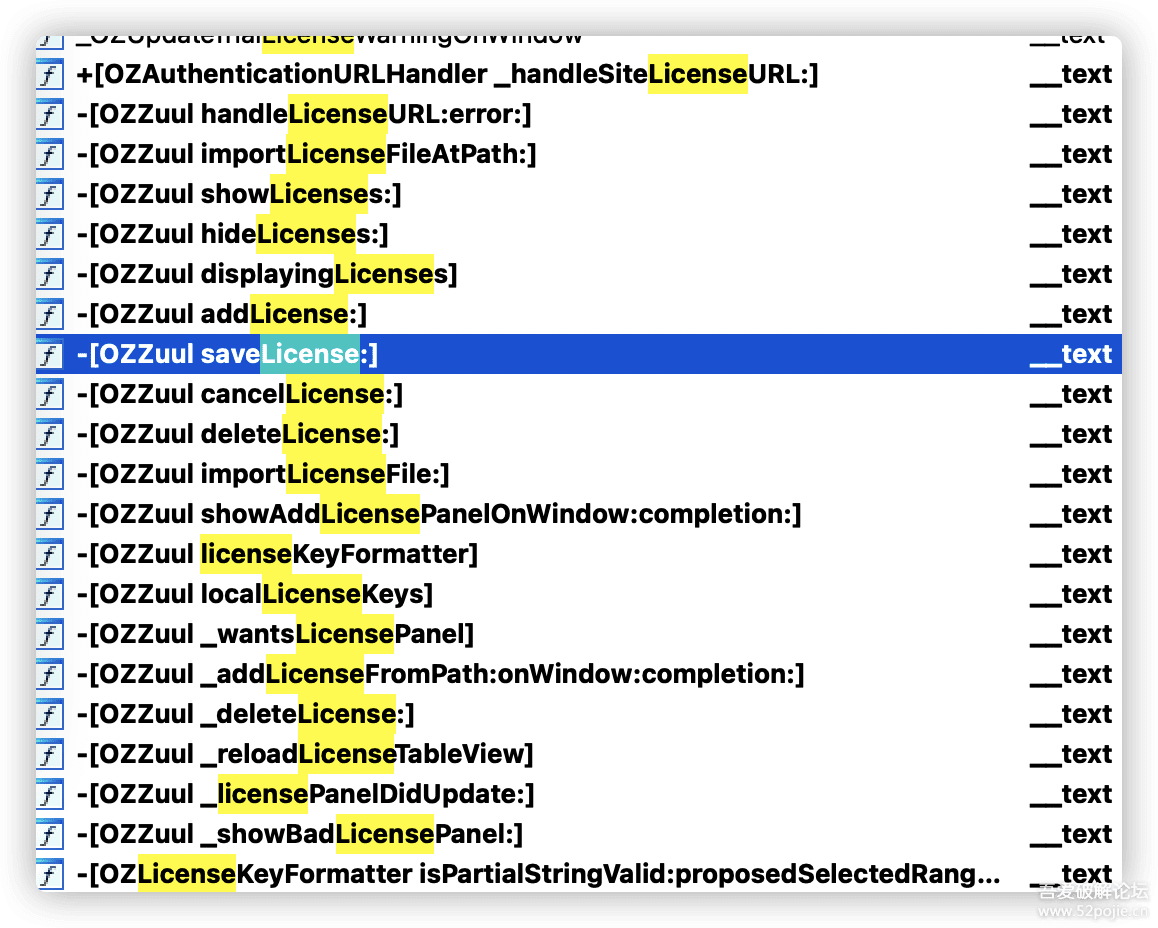
https://www.52pojie.cn/forum.php?mod=viewthread&tid=1445389

OmniGraffle是优秀的作图软件，尤其对于mac平台的逆向工程师来讲，是替代dia的好工具。dia虽然也讲有mac osx版本，但运行起来就是一个白板，怎么看起来都不对劲。相比Linux平台的Dia，OmniGraffle使用起来体验更优秀，  
  
例如在逆向时，进行函数调用分析、程序内存布局分析等，如下图。个人觉得实在是没有比Graffle更好的工具来做这件事情了。  
   
  
  
  
  
缺点就是有点贵：  
   
每年125美元。  
  
那么打开OmniGraffle以后，进行尝试注册：  
   
  
获取有效信息：Invalid License, 验证License的按钮为save按钮。  
  
1. 我们按照通常的过程尝试： 从Graffle的可执行文件中搜索Invalid License. 发现找不到哈。  
不过我们搜到了isLicensed等判断是否已经注册的函数。  
   
  
按照惯例，我们进入isLicensed函数，修改返回值，让它一直返回True。然后运行Graffle发现，这个软件运行起来但不现实窗口，且kill -9杀不死。各种杀不死。即便关机也会卡死关机进程。怀疑作者是防[破解](https://www.52pojie.cn" \t "_blank)做了一些奇怪的事情导致的。具体为啥没有分析。  
在强制关机几次以后，忍着心碎我决定不再爆破，还是看看注册逻辑吧。  
  
2. 还是从Invalid License入手：  
在`/Applications/OmniGraffle.app/Contents`目录中，用一下命令搜索：  
find . -type f -print0|xargs -0 grep "Invalid License"  
   
  
好的吧。可以看到实质上注册逻辑是在一个Framework文件里：./Frameworks/OmniZuul.framework/Versions/A/OmniZuul  
如果仔细逆向一下这个文件，会发现一个神奇的东西，所有Omni公司的注册逻辑都在这里面。使用的同一套注册算法，只是算法常量不同。而各个版本的算法常量在这个文件里是一个表格的形式存在：  
具体如下图：  
   
其中数据结构大致为： 

[C] 纯文本查看 复制代码

[?](https://www.52pojie.cn/forum.php?mod=viewthread&tid=1445389&extra=page%3D1%26filter%3Dtypeid%26typeid%3D377)

|  |  |
| --- | --- |
| 1  2  3  4  5 | 00000000 OmniCONSTTable  struc ; (sizeof=0x20, mappedto\_284)  00000000 AppName         dq ?  00000008 AppUid          dd ?  0000000C AppConst        db 20 dup(?)  00000020 OmniCONSTTable  ends |

这种形式。  
  
我们继续在Strings window 里面搜索Invalid License：  
  
   
  
然后定位到对应的函数：  
  
大致看一眼，就是非法licese报错的地方  
   
  
往上查找xref，一直找到：  
  
[OZZuul \_deleteLicense:] 函数。看来有那么一堆对License操作的函数哟。  
  
在函数窗口我们搜一下License：  
   
果然，有一个saveLicense函数。我们还记得一开始尝试输入License时，确认License的按钮名称为save。对应的处理函数就是它啦。  
注册逻辑判定就在这个函数里面。  
  
分析这个函数比较复杂，整个算法分散在很多函数调用里面。不过一点一点动态调试也比较容易弄清楚逻辑。  
详细的算法逻辑，大家可以看复原出的python源码哟。

[Python] 纯文本查看 复制代码

[?](https://www.52pojie.cn/forum.php?mod=viewthread&tid=1445389&extra=page%3D1%26filter%3Dtypeid%26typeid%3D377)

|  |  |
| --- | --- |
| 001  002  003  004  005  006  007  008  009  010  011  012  013  014  015  016  017  018  019  020  021  022  023  024  025  026  027  028  029  030  031  032  033  034  035  036  037  038  039  040  041  042  043  044  045  046  047  048  049  050  051  052  053  054  055  056  057  058  059  060  061  062  063  064  065  066  067  068  069  070  071  072  073  074  075  076  077  078  079  080  081  082  083  084  085  086  087  088  089  090  091  092  093  094  095  096  097  098  099  100  101  102  103  104  105  106  107  108  109  110  111  112  113  114  115  116  117  118  119  120  121  122  123  124  125  126  127  128  129  130  131  132  133  134  135  136  137  138  139  140  141  142  143  144  145  146  147  148  149  150  151  152  153  154  155  156  157  158  159  160  161  162  163  164  165  166  167  168  169  170  171  172  173  174  175  176  177  178  179  180  181  182  183  184  185  186  187  188  189  190  191  192  193  194  195  196  197  198  199  200  201  202  203  204  205  206  207  208  209  210  211  212  213  214  215  216  217  218  219  220  221  222  223  224  225  226  227  228  229  230  231  232  233  234 | #!/usr/bin/python  # coding=utf-8    #   Author    :   Simon Huang  #   Time      :   5/20/21 6:49 PM    import os, sys  import random  import time  import hashlib    log256\_26 = [2, 4, 6, 7]    APP\_VERSION\_LIST = {"OmniFocus v3": 0xe,                      "OmniFocus Pro v3": 0xf,                      "OmniGraffle v7": 9,                      "OmniGraffle Pro v7": 8,                      "OmniOutliner Essentials v5": 0xc,                      "OmniOutliner Pro v5": 0xd,                      "OmniPlan v3": 6,                      "OmniPlan Pro v3": 7                      }    APP\_UID\_LIST = [      1000205, 1000216, 1000215, 1000200, 1000211, 1000210,      1000219, 1000220, 1000221, 1000222, 1000212, 1000209,      1000224, 1000223, 1000228, 1000226, 1000214, 1000208,      1000207, 1000197  ]    APP\_CONST\_LIST = [      0x8F, 0x0E0, 0x1F, 0x81, 0x36, 0x28, 0x9B, 0x0D1, 0x45, 0x0A3, 0x1E, 0x0C7, 0x0B2, 0x56, 0x0BE, 0x0E7, 0x42, 0x96,      0x7E, 0x0E9, 0x0DD, 0x18, 0x56, 0x0E8, 0x80, 0x0CF, 0x8E, 0x0E5, 0x0BF, 0x0FF, 0x0B1, 0x0BB, 0x39, 0x63, 0x0C0,      0x57, 0x0C6, 0x0EA, 0x0D1, 0x0B5, 0x0C0, 0x0A2, 0x0F2, 0x0E4, 0x0CD, 0x0E9, 0x0FF, 0x39, 0x59, 0x0F, 0x79, 0x0DB,      0x0D7, 0x29, 0x0A7, 0x68, 0x0D4, 0x68, 0x48, 0x16, 0x0, 0x15, 0x3A, 0x0A9, 0x17, 0x0D, 0x5, 0x6E, 0x75, 0x2F, 0x54,      0x0AD, 0x0B6, 0x72, 0x0E5, 0x0E4, 0x0EB, 0x9C, 0x0C7, 0x75, 0x0BA, 0x72, 0x0F2, 0x6, 0x99, 0x8B, 0x4D, 0x4B, 0x46,      0x17, 0x0CD, 0x0A, 0x0E1, 0x0E0, 0x82, 0x88, 0x0FB, 0x27, 0x0D8, 0x0B7, 0x87, 0x0D6, 0x87, 0x4B, 0x91, 0x9, 0x0A6,      0x54, 0x32, 0x0E6, 0x0A0, 0x48, 0x0F2, 0x0E4, 0x7E, 0x0E, 0x82, 0x9E, 0x17, 0x0A7, 0x35, 0x0CF, 0x0EC, 0x5C, 0x0C1,      0x0A7, 0x0F1, 0x0C9, 0x23, 0x0F4, 0x21, 0x96, 0x0C3, 0x0A7, 0x60, 0x0E0, 0x27, 0x0EF, 0x0A4, 0x17, 0x0B4, 0x33,      0x52, 0x3C, 0x1F, 0x0DA, 0x11, 0x0C7, 0x12, 0x32, 0x0A2, 0x1A, 0x0B0, 0x6F, 0x0F9, 0x3F, 0x89, 0x8, 0x0C, 0x6C, 0x1,      0x0F1, 0x0C2, 0x8D, 0x0C6, 0x60, 0x46, 0x55, 0x0D9, 0x2C, 0x10, 0x0B3, 0x7C, 0x14, 0x0E5, 0x38, 0x52, 0x39, 0x0B,      0x2A, 0x3E, 0x0E2, 0x65, 0x30, 0x8, 0x0B2, 0x0A3, 0x0D6, 0x96, 0x40, 0x72, 0x0D6, 0x14, 0x5B, 0x5F, 0x0D2, 0x91,      0x0B6, 0x51, 0x42, 0x4C, 0x0BF, 0x8A, 0x0BE, 0x0A1, 0x7D, 0x2F, 0x6, 0x36, 0x0B9, 0x0FD, 0x26, 0x81, 0x0C3, 0x0E8,      0x90, 0x55, 0x7E, 0x8D, 0x7, 0x0B0, 0x2A, 0x8B, 0x0B5, 0x37, 0x0BE, 0x0C1, 0x85, 0x0EC, 0x8, 0x2C, 0x0FE, 0x0B9,      0x2F, 0x0B0, 0x0FC, 0x42, 0x6F, 0x61, 0x0BC, 0x6F, 0x81, 0x0D1, 0x92, 0x13, 0x68, 0x0A3, 0x0DE, 0x75, 0x0D4, 0x0F6,      0x0CC, 0x46, 0x69, 0x52, 0x4D, 0x53, 0x7B, 0x91, 0x0B7, 0x39, 0x17, 0x7E, 0x0A3, 0x6, 0x7, 0x8F, 0x47, 0x8E, 0x88,      0x61, 0x0C0, 0x0F6, 0x0C9, 0x64, 0x7E, 0x4B, 0x0D0, 0x89, 0x0B3, 0x79, 0x89, 0x2E, 0x3C, 0x17, 0x3E, 0x0BB, 0x0C4,      0x0E9, 0x0DB, 0x68, 0x91, 0x15, 0x1F, 0x0F2, 0x0B4, 0x47, 0x2B, 0x7, 0x63, 0x9B, 0x0A2, 0x71, 0x0E9, 0x0E6, 0x3A,      0x1B, 0x5C, 0x0F6, 0x5, 0x0F8, 0x3F, 0x8B, 0x6E, 0x8E, 0x0D6, 0x8C, 0x0B2, 0x90, 0x0F7, 0x0A5, 0x0D, 0x2, 0x0B8,      0x0AC, 0x0E3, 0x0F2, 0x0C6, 0x82, 0x0C5, 0x1E, 0x8E, 0x4C, 0x64, 0x0EE, 0x9A, 0x8, 0x5D, 0x77, 0x0D5, 0x28, 0x7E,      0x72, 0x0B4, 0x58, 0x0BA, 0x0DD, 0x0F5, 0x0B2, 0x48, 0x6C, 0x8, 0x0ED, 0x0D0, 0x3C, 0x0, 0x7F, 0x0FD, 0x0D5, 0x1C,      0x73, 0x91, 0x0CB, 0x4E, 0x24, 0x0B3, 0x0F6, 0x0FB, 0x0E5, 0x90, 0x4A, 0x0A8, 0x86, 0x22, 0x79, 0x0F6, 0x1E, 0x0E2,      0x0A1, 0x18, 0x0E4, 0x0A3, 0x17, 0x0AB, 0x0B7, 0x99, 0x0C4, 0x8F, 0x67, 0x0F2, 0x0FD, 0x7B, 0x0BF, 0x4E, 0x4B, 0x71,      0x0D7, 0x0C0, 0x29, 0x4C  ]      def ascii26str(data\_bytes: bytes):      data\_length = len(data\_bytes)      sepchar = ''        if data\_length == 0:          return sepchar        tmpi = 0      tmpv = 0        data\_cursor = 0        buffer = [0] \* 48      buffer\_cursor = 0      sbuff = [0] \* 8        restult\_str = ''        while data\_length > 0:          data\_length -= 1          tmpv = ((data\_bytes[data\_cursor] | (tmpv << 8) & 0xffffffff)) & 0xffffffffffffffff            if tmpi == 3:              sbuff = [0] \* 8                j = logv = 7  # log256\_26[3]              sbuff\_i = 0                while j > 0:                  rdx = (((0x4EC4EC4EC4EC4EC5 \* tmpv) >> 64) >> 3) & 0xffffffffffffffff                  eax = (rdx \* 5) & 0xffffffff                  eax = (eax \* 5) & 0xffffffff                  eax = (eax + rdx) & 0xffffffff                    sbuff[7 - j] = (tmpv - eax) & 0xff                  tmpv = rdx                  j -= 1                i = 0              buffer\_cursor = 0              while i != -7:                  buffer[buffer\_cursor] = (sbuff[7 + i - 1] + 65) & 0xff                  buffer\_cursor += 1                  i -= 1                # bufutf8 =  utf-8.encode(buf[:7]              # restult\_str += (buffer[:7]).encode('utf-8')              i = 0              while buffer[i] != 0:                  restult\_str += chr(buffer[i])                  i += 1              tmpv = 0              tmpi = 0            else:              tmpi += 1          data\_cursor += 1        if tmpi > 0:          sbuff = [0] \* 8          sbuff\_i = 0          tmpi1v = tmp1v = log256\_26[tmpi - 1]            while tmp1v > 0:              rdx = (((0x4EC4EC4EC4EC4EC5 \* tmpv) >> 64) >> 3) & 0xffffffffffffffff              eax = (rdx \* 5) & 0xffffffff              eax = (eax \* 5) & 0xffffffff              eax = (eax + rdx) & 0xffffffff                sbuff[sbuff\_i] = (tmpv - eax) & 0xff              sbuff\_i += 1              tmpv = rdx              tmp1v -= 1            buffer\_cursor = 0          negtmp1v = -log256\_26[tmpi - 1]          p = 0          while p != negtmp1v:              buffer[buffer\_cursor] = (sbuff[tmpi1v + p - 1] + 65) & 0xff              buffer\_cursor += 1              p -= 1            i = 0          while buffer[i] != 0:              restult\_str += chr(buffer[i])              i += 1        return restult\_str      def createSerial(username="Simon Huang", napp\_id=8, dbgRandList=None):      if username is None or len(username.strip()) == 0:          username = "Simon Huang"        joinedName = ''      for achr in username:          if achr.isalpha() or achr.isnumeric():              joinedName += achr      joinedNameLen = len(joinedName)        # random.seed(time.time())        randvalue = random.randint(0, 0xff)      if dbgRandList:          randvalue = dbgRandList[0]      md2 = [0] \* 32      i = 0      while i != 4:          md2[i + 5] = randvalue          randvalue = random.randint(0, 0xff)          if dbgRandList:              randvalue = dbgRandList[i + 1]          i += 1        md2[9] = randvalue        randvalue = random.randint(0, 0xff)      if dbgRandList:          randvalue = dbgRandList[5]        md2[10] = randvalue & 3      md2[11] = 0xff      md2[12] = 0xff        len8data = md2[5:5 + 8]      len8str = ascii26str(len8data)        len8str\_04 = len8str[0:0 + 4]      len8str\_44 = len8str[4:4 + 4]      len8str\_84 = len8str[8:8 + 4]      len8str\_122 = len8str[12:12 + 2]        formatedLic = "%s-%s-%s-%s" % (len8str\_04, len8str\_44, len8str\_84, len8str\_122)      idLicNameStr = "%u%s%s" % (APP\_UID\_LIST[napp\_id], formatedLic, joinedName)      idLicNameStrUtf8 = idLicNameStr.encode('utf8')      idLicNameBytes = idLicNameStrUtf8        md = list(hashlib.sha1(idLicNameBytes).digest())        appConstOffset = 20 \* napp\_id      j = 0      while j != 20:          md[j] ^= APP\_CONST\_LIST[appConstOffset + j]          j += 1        k = 0      while k != 5:          md[k] ^= md[k + 15] ^ md[k + 10] ^ md[k + 5]          k += 1        i = 0      while i < 5:          md2[i] = md[i]          i += 1        if joinedNameLen > 0:          n = 0          while n != 13:              md2[n] = (~(md2[n] ^ ord(joinedName[(n % joinedNameLen)]))) & 0xff              n += 1        md2len13d = md2[:13]      md2len13dStr = ascii26str(md2len13d)      insert\_pos = [24, 20, 16, 12, 8, 4]      for apos in insert\_pos:          md2len13dStr = md2len13dStr[:apos] + "-" + md2len13dStr[apos:]        serial = md2len13dStr[:-1]      return (username, serial)      if \_\_name\_\_ == '\_\_main\_\_':      username = input("Please input your username:")      for app in APP\_VERSION\_LIST:          appUid = APP\_VERSION\_LIST[app]          license = createSerial(username=username, napp\_id=appUid)          print(license, '\tfor\t', app) |

注意其中一部分加密算法在./Frameworks/OmniFoundation.framework/Versions/A/OmniFoundation文件中。  
需要对该文件进行逆向一下哦。  
  
该文件运行效果如下；  
Please input your username:52pojie  
('52pojie', 'JPJH-WWIL-CLRN-PLBM-DMTO-CEBD-MTO')         for         OmniFocus v3  
('52pojie', 'ESKK-XEBH-SQGS-PHGR-NKMV-YEBN-KMV')         for         OmniFocus Pro v3  
('52pojie', 'JKOT-VBFL-OMTL-YHJT-DZUI-SEBD-ZUI')         for         OmniGraffle v7  
('52pojie', 'FWIY-MAJD-NVKT-DMIL-YYIL-MEBY-YIL')         for         OmniGraffle Pro v7  
('52pojie', 'HOUP-JOKL-CRZN-BPMM-AVWK-GEBA-VWK')         for         OmniOutliner Essentials v5  
('52pojie', 'KVUS-WLAD-XZCU-FBHB-PZTY-WEBP-ZTY')         for         OmniOutliner Pro v5  
('52pojie', 'JBKN-GJPB-XJDQ-RQDV-FQPU-WEBF-QPU')         for         OmniPlan v3  
('52pojie', 'DCAR-ZCSC-ATQZ-NICH-QRBH-IEBQ-RBH')         for         OmniPlan Pro v3