

# 目录

前言	1.1
iOS逆向举例概览	1.2
静态分析	1.3
动态调试	1.4
静态分析+动态调试	1.5
WhatsApp	1.5.1
StringAppend_100314974	1.5.1.1
逆向过程	1.5.1.1.1
前后对比和结果	1.5.1.1.2
附录	1.6
参考资料	1.6.1

# iOS逆向：举例详解

- 最新版本: v0.8
- 更新时间: 20240224

## 简介

通过具体的例子，详细介绍iOS逆向的过程，以便于更好的理解如何进行iOS逆向的开发。具体例子有，WhatsApp中的StringAppend\_100314974函数。

## 源码+浏览+下载

本书的各种源码、在线浏览地址、多种格式文件下载如下：

### HonKit源码

- [crifan/ios\\_re\\_detail\\_example: iOS逆向：举例详解](#)

### 如何使用此HonKit源码去生成发布为电子书

详见：[crifan/honkit\\_template: demo how to use crifan honkit template and demo](#)

## 在线浏览

- [iOS逆向：举例详解 book.crifan.org](#)
- [iOS逆向：举例详解 crifan.github.io](#)

## 离线下载阅读

- [iOS逆向：举例详解 PDF](#)
- [iOS逆向：举例详解 ePUB](#)
- [iOS逆向：举例详解 Mobi](#)

## 版权和用途说明

此电子书教程的全部内容，如无特别说明，均为本人原创。其中部分内容参考自网络，均已备注了出处。如发现有侵权，请通过邮箱联系我 [admin 艾特 crifan.com](mailto:admin@crifan.com)，我会尽快删除。谢谢合作。

各种技术类教程，仅作为学习和研究使用。请勿用于任何非法用途。如有非法用途，均与本人无关。

## 鸣谢

感谢我的老婆陈雪的包容理解和悉心照料，才使得我 crifan 有更多精力去专注技术专研和整理归纳出这些电子书和技术教程，特此鸣谢。

## 其他

### 作者的其他电子书

本人 crifan 还写了其他 150+ 本电子书教程，感兴趣可移步至：

[crifan/crifan\\_ebook\\_readme: Crifan的电子书的使用说明](#)

### 关于作者

关于作者更多介绍，详见：

[关于CrifanLi李茂 – 在路上](#)

crifan.org, 使用[署名4.0国际\(CC BY 4.0\)协议](#)发布 all right reserved, powered by Gitbook最后更新：  
2024-02-24 16:05:58

# iOS逆向举例概览

iOS逆向涉及内容较多，各种工具和过程，对于新手来说，即使看了些教程，也不能透彻的理解。

此处通过具体的详细的例子，去解释如何进行iOS逆向的过程。希望对你增加iOS逆向的理解有帮助。

crifan.org, 使用[署名4.0国际\(CC BY 4.0\)协议](#)发布 all right reserved, powered by Gitbook最后更新：  
2024-02-24 14:48:01

# 静态分析

crifan.org, 使用[署名4.0国际\(CC BY 4.0\)协议](#)发布 all right reserved, powered by Gitbook最后更新：  
2024-02-24 14:37:12

# 动态调试

crifan.org, 使用[署名4.0国际\(CC BY 4.0\)协议](#)发布 all right reserved, powered by Gitbook最后更新:  
2024-02-24 14:37:12

# 静态分析+动态调试

要实现iOS逆向的静态分析和动态调试，需要用到很多相关工具：

- 静态分析
  - [IDA](#)
    - 分析函数代码逻辑，尤其是通过伪代码和汇编
- 动态调试
  - [Xcode + lldb](#)
  - [Xcode+iOSOpenDev](#)

crifan.org, 使用[署名4.0国际\(CC BY 4.0\)协议](#)发布 all right reserved, powered by Gitbook最后更新：  
2024-02-24 14:49:47

# iOS逆向WhatsApp

此处介绍iOS逆向[WhatsApp](#)的iOS版本相关例子。

crifan.org, 使用[署名4.0国际\(CC BY 4.0\)协议](#)发布 all right reserved, powered by Gitbook最后更新:  
2024-02-24 14:43:52

## StringAppend\_100314974

此处以iOS的app WhatsApp 的二进制 WhatsApp 中的 `StringAppend_100314974` 函数为例，详细介绍，如何进行iOS逆向的静态分析和动态调试。

### 背景

对于一般的函数，没有特殊处理，比如混淆的话，往往是：

直接通过IDA伪代码，即可看出函数大体逻辑，甚至是完全看懂逻辑的所有细节

而此处要举例的函数，则是：

iOS的ObjC代码，但是底层涉及到iOS的Swift语言实现的，所以底层很难直接看到是什么类和函数

此处详细介绍，如何用各种工具，去一点点分析和调试，最终才能一步步的，彻底的搞懂代码逻辑。

### 目标

搞懂 WhatsApp 中的 `StringAppend_100314974` 函数的逻辑。

crifan.org, 使用[署名4.0国际\(CC BY 4.0\)协议](#)发布 all right reserved, powered by Gitbook最后更新：

2024-02-24 15:37:01

## 逆向过程

现在开始具体说明：

WhatsApp中的函数，二进制内函数偏移量地址是： 0x100314974

最开始，用IDA打开并自动分析后，去查看IDA伪代码是：

- sub\_100314974

```
__int64 sub_100314974()
{
    __QWORD v0; // x20
    __int64 v1; // x8
    __int64 v2; // x9
    __int64 v3; // x22
    __int64 v4; // x23
    __int64 v5; // x21
    __int64 v6; // x8
    __int64 v7; // x19
    __int64 v8; // x22
    __int64 v9; // x21
    __int64 v10; // x21
    __int64 v12; // [xsp+0h] [xbp-40h]
    __int64 v13; // [xsp+8h] [xbp-38h]

    v1 = v0[2];
    v12 = *(__QWORD *) (v1 + 16);
    v13 = *(__QWORD *) (v1 + 24);
    swift_bridgeObjectRetain(v13);
    String.append(_)(124LL, 0xE100000000000000LL);
    v2 = v0[3];
    v3 = *(__QWORD *) (v2 + 16);
    v4 = *(__QWORD *) (v2 + 24);
    swift_bridgeObjectRetain(v13);
    String.append(_)(v3, v4);
    swift_bridgeObjectRelease(v5);
    swift_bridgeObjectRetain(v13);
    String.append(_)(124LL, 0xE100000000000000LL);
    swift_bridgeObjectRelease(v13);
    v6 = v0[4];
    v7 = *(__QWORD *) (v6 + 16);
    v8 = *(__QWORD *) (v6 + 24);
    swift_bridgeObjectRetain(v9);
    String.append(_)(v7, v8);
    swift_bridgeObjectRelease(v10);
    return v12;
}
```

```

Function name: sub_100314974
1 _int64 sub_100314974()
2 {
3     _QWORD v0; // x0
4     _int64 v1; // x8
5     _int64 v2; // x9
6     _int64 v3; // x10
7     _int64 v4; // x11
8     _int64 v5; // x12
9     _int64 v6; // x13
10    _int64 v7; // x14
11    _int64 v8; // x22
12    _int64 v9; // x23
13    _int64 v10; // x24
14    _int64 v11; // [xsp+0h] [xbp-40h]
15    _int64 v12; // [xsp+0h] [xbp-30h]
16
17    v1 = v0[2];
18    v2 = *(QWORD *)v1 + 16;
19    v13 = *(QWORD *)v1 + 24;
20    swift_bridgeObjectRetain(v13);
21    String.append_(v13, v4);
22    swift_bridgeObjectRetain(v13);
23    v3 = *(QWORD *)v2 + 16;
24    swift_bridgeObjectRetain(v3);
25    String.append_(v3, v4);
26    swift_bridgeObjectRetain(v3);
27    String.append_(v3)124LL, 0xE100000000000000LL;
28    swift_bridgeObjectRelease(v3);
29
30    v1 = v0[4];
31    v2 = *(QWORD *)v1 + 16;
32    v7 = *(QWORD *)v2 + 16;
33    swift_bridgeObjectRetain(v7);
34    swift_bridgeObjectRetain(v7);
35    String.append_(v7, v8);
36    swift_bridgeObjectRelease(v10);
37    return v12;
38 }

Line 40481 of 300503
Output
10440Lb0: using guessed type _int64 __fastcall SWIFT_BridgeObjectRetain(_QWORD);
100314974: variable 'v0' is possibly undefined
100314974: variable 'v5' is possibly undefined
100314974: variable 'v6' is possibly undefined
100314974: variable 'v7' is possibly undefined
100314974: variable 'v8' is possibly undefined
100314974: variable 'v9' is possibly undefined
100314974: variable 'v10' is possibly undefined
100314974: variable 'v11' is possibly undefined
100314974: variable 'v12' is possibly undefined
100314974: variable 'v13' is possibly undefined
100314974: variable 'v14' is possibly undefined
100314974: variable 'v15' is possibly undefined
100314974: variable 'v16' is possibly undefined
100314974: variable 'v17' is possibly undefined
100314974: variable 'v18' is possibly undefined
100314974: variable 'v19' is possibly undefined
100314974: variable 'v20' is possibly undefined
100314974: variable 'v21' is possibly undefined
100314974: variable 'v22' is possibly undefined
100314974: variable 'v23' is possibly undefined
100314974: variable 'v24' is possibly undefined
100314974: variable 'v25' is possibly undefined
100314974: variable 'v26' is possibly undefined
100314974: variable 'v27' is possibly undefined
100314974: variable 'v28' is possibly undefined
100314974: variable 'v29' is possibly undefined
100314974: variable 'v30' is possibly undefined
100314974: variable 'v31' is possibly undefined
100314974: variable 'v32' is possibly undefined
100314974: variable 'v33' is possibly undefined
100314974: variable 'v34' is possibly undefined
100314974: variable 'v35' is possibly undefined
100314974: variable 'v36' is possibly undefined
100314974: variable 'v37' is possibly undefined
100314974: variable 'v38' is possibly undefined
Caching 'Functions'... ok
Caching 'Functions'... ok
Caching 'Functions'... ok

```

很明显，最开始的默认的IDA伪代码看不出此处函数的代码逻辑 =不知道函数具体是做什么的

然后，经过分析和调试，慢慢搞懂了代码逻辑，再回来给IDA伪代码中去优化改名：

- 函数名改名
  - sub\_100314974 -> StringAppend\_100314974
- 变量名改名
  - v1 -> universeInfo
  - v12 -> userInfoNameStringObjPart1
  - v13 -> userInfoNameStringObjPart2
  - v2 -> experimentInfo
  - v6 -> bucketInfo
  - ...
- 加上注释

◦

然后才能慢慢搞懂，逐渐的彻底搞懂代码的全部逻辑

接着去解释中间是，如何：

- 静态分析
- 动态调试

的。

先去动态调试代码逻辑：

此时先要搞清楚：

传入的参数值：

且通过后续研究，比如查看IDA的汇编代码：

```
__text:0000000100314974 sub_100314974 ; CODE XREF: sub_100314
868+B4↑p
__text:0000000100314974
__text:0000000100314974 var_40 = 0x40
__text:0000000100314974 var_30 = 0x30
__text:0000000100314974 var_20 = 0x20
__text:0000000100314974 var_10 = 0x10
__text:0000000100314974 var_s0 = 0
__text:0000000100314974
__text:0000000100314974 SUB SP, SP, #0x50
__text:0000000100314974 STP X24, X23, [SP,#0x40+var_30]
__text:0000000100314974 STP X22, X21, [SP,#0x40+var_20]
__text:0000000100314974 STP X20, X19, [SP,#0x40+var_10]
__text:0000000100314974 STP X29, X30, [SP,#0x40+var_s0]
__text:0000000100314974 ADD X29, SP, #0x40
__text:0000000100314974 MOV X19, X20
__text:0000000100314974 LDR X8, [X20,#0x10]
__text:0000000100314974 LDP X9, X0, [X8,#0x10]
__text:0000000100314974 STP X9, X0, [SP,#0x40+var_40]
__text:0000000100314974 BL _swift_bridgeObjectRetain
__text:00000001003149A0 MOV X20, SP
__text:00000001003149A4 MOV W0, #0x7C ; ''
__text:00000001003149A8 MOV X1, #0xE100000000000000
__text:00000001003149AC BL _SSS6appendyySSF ; String.append(_)
nd(_)
__text:00000001003149B0 LDP X8, X21, [SP,#0x40+var_40]
__text:00000001003149B4 LDR X9, [X19,#0x18]
__text:00000001003149B8 LDP X22, X23, [X9,#0x10]
__text:00000001003149BC STP X8, X21, [SP,#0x40+var_40]
__text:00000001003149C0 MOV X0, X21
__text:00000001003149C4 BL _swift_bridgeObjectRetain
__text:00000001003149C8 MOV X20, SP
__text:00000001003149CC MOV X0, X22
__text:00000001003149D0 MOV X1, X23
__text:00000001003149D4 BL _SSS6appendyySSF ; String.append(_)
nd(_)
__text:00000001003149D8 LDP X20, X22, [SP,#0x40+var_40]
__text:00000001003149DC MOV X0, X21
__text:00000001003149E0 BL _swift_bridgeObjectRelease
```

```

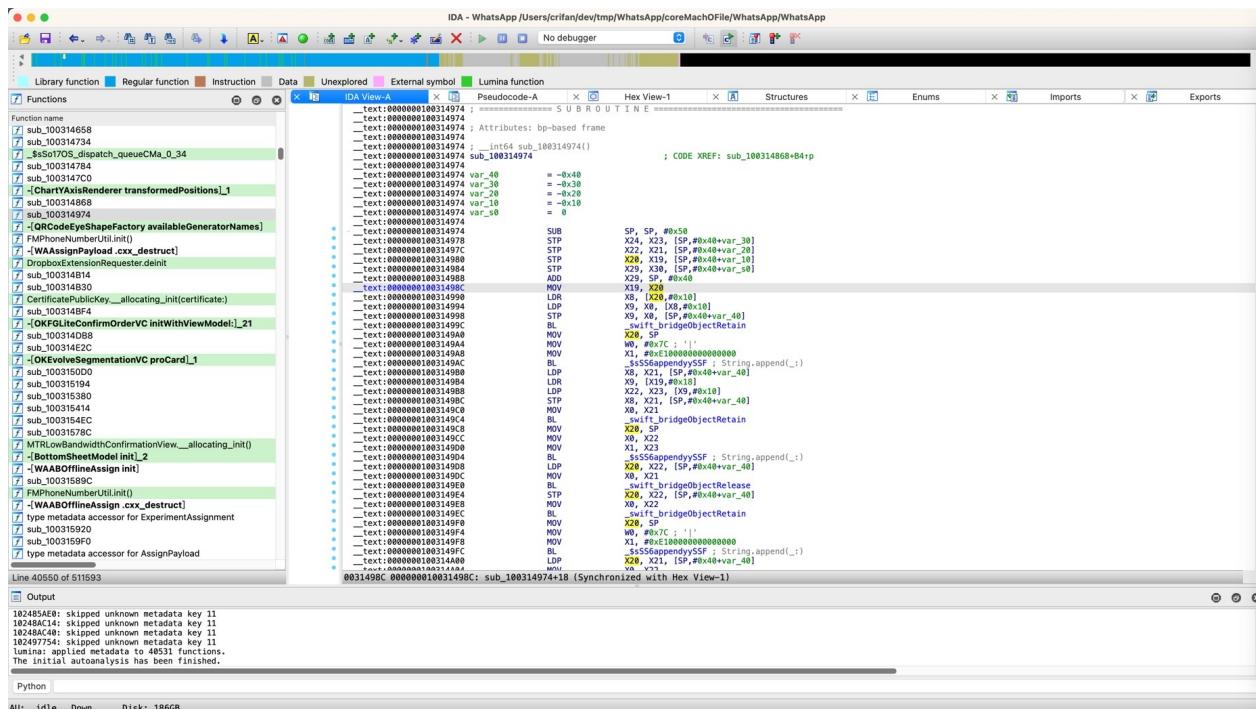
    __text:00000001003149E4      STP          X20, X22, [SP,#0x40+var_40]
    __text:00000001003149E8      MOV          X0, X22
    __text:00000001003149EC      BL           _swift_bridgeObjectRetain
    __text:00000001003149F0      MOV          X20, SP
    __text:00000001003149F4      MOV          W0, #0x7C ; '|'
    __text:00000001003149F8      MOV          X1, #0xE100000000000000
    __text:00000001003149FC      BL           _SSS6appendyySSF ; String.append

nd(_):
    __text:0000000100314A00      LDP          X20, X21, [SP,#0x40+var_40]
    __text:0000000100314A04      MOV          X0, X22
    __text:0000000100314A08      BL           _swift_bridgeObjectRelease
    __text:0000000100314A0C      LDR          X8, [X19,#0x20]
    __text:0000000100314A10      LDP          X19, X22, [X8,#0x10]
    __text:0000000100314A14      STP          X20, X21, [SP,#0x40+var_40]
    __text:0000000100314A18      MOV          X0, X21
    __text:0000000100314A1C      BL           _swift_bridgeObjectRetain
    __text:0000000100314A20      MOV          X20, SP
    __text:0000000100314A24      MOV          X0, X19
    __text:0000000100314A28      MOV          X1, X22
    __text:0000000100314A2C      BL           _SSS6appendyySSF ; String.append

nd(_):
    __text:0000000100314A30      LDP          X19, X20, [SP,#0x40+var_40]
    __text:0000000100314A34      MOV          X0, X21
    __text:0000000100314A38      BL           _swift_bridgeObjectRelease
    __text:0000000100314A3C      MOV          X0, X19
    __text:0000000100314A40      MOV          X1, X20
    __text:0000000100314A44      LDP          X29, X30, [SP,#0x40+var_s0]
    __text:0000000100314A48      LDP          X20, X19, [SP,#0x40+var_10]
    __text:0000000100314A4C      LDP          X22, X21, [SP,#0x40+var_20]
    __text:0000000100314A50      LDP          X24, X23, [SP,#0x40+var_30]
    __text:0000000100314A54      ADD          SP, SP, #0x50 ; 'P'
    __text:0000000100314A58      RET

__text:0000000100314A58 ; End of function sub_100314974

```



才搞清楚此处是特殊的：通过X20传递了参数的  
而不是普通的，通过X0、X1、X2等寄存器传递参数的  
然后去调试查看寄存器值：

```
(lldb) reg r x0 x1 x19 x20
x0 = 0x000000016d658c90
x1 = 0x00000002828d0780
x19 = 0x00000002828d0780
x20 = 0x00000002833c1140
(lldb) po $x20
MainAppLibrary.ExperimentAssignment
```

发现是：

- 类 MainAppLibrary.ExperimentAssignment

后续确认，是个：Swift的类，而不是普通的ObjC的类

如此，想要搞清楚，该类的具体细节，比如有哪些属性（字段）和函数等等，就无法直接查看到  
然后需要去通过分析才能找到。

经过：

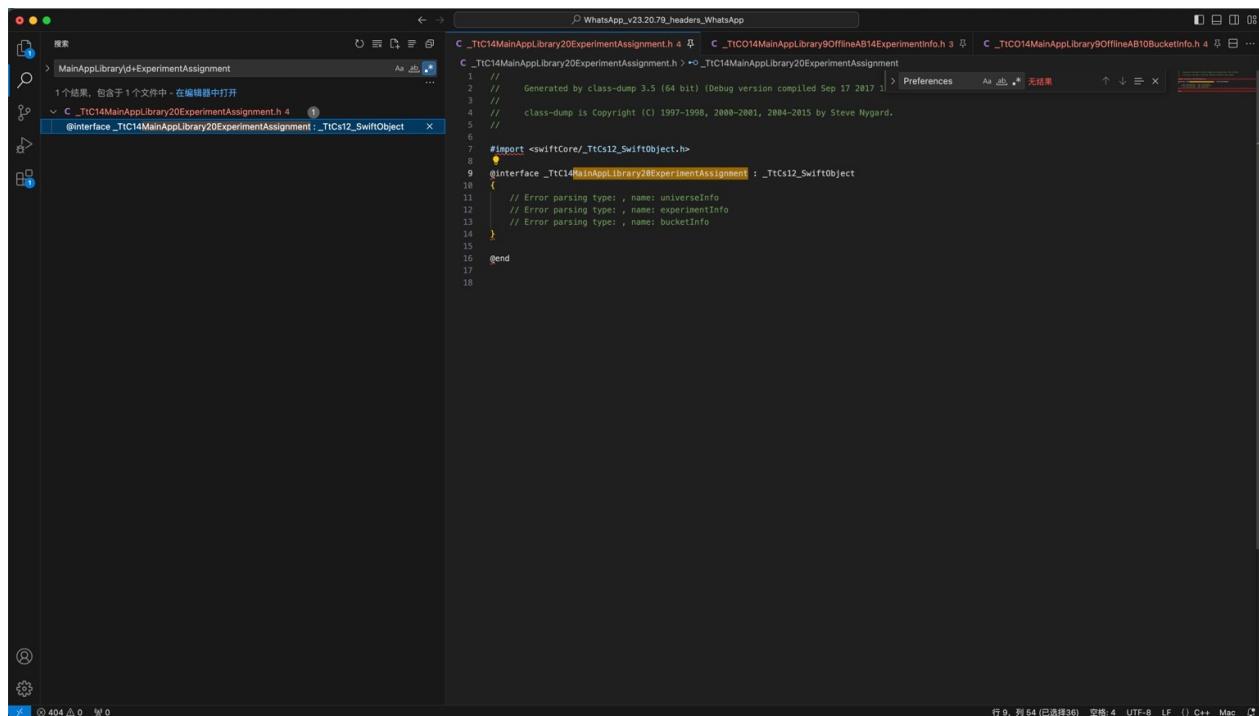
- 用class-dump导出的头文件
  - 详见
    - [class-dump · iOS逆向开发：静态分析](#)
- 找到类的字段=属性名称
- 导出的静态字符串等资源
  - 详见
    - [查看信息和导出字符串 · iOS逆向开发：静态分析](#)
- IDA中类的定义
  - 找到类的字段=属性，以及具体的偏移量定义

最后找到了：

之前导出的头文件中，搜：

```
MainAppLibrary\d+ExperimentAssignment
```

找到的：



- /Users/crifan/dev/dev\_root/iosReverse/WhatsApp/headers/WhatsApp\_v23.20.79\_headers\_WhatsApp/\_TtC14MainAppLibrary20ExperimentAssignment.h

```
//  
//     Generated by class-dump 3.5 (64 bit) (Debug version compiled Sep 17 2017 16:24:4  
8).  
//  
//     class-dump is Copyright (C) 1997-1998, 2000-2001, 2004-2015 by Steve Nygard.  
  
#import <swiftCore/_TtCs12_SwiftObject.h>  
  
@interface _TtC14MainAppLibrary20ExperimentAssignment : _TtCs12_SwiftObject  
{  
    // Error parsing type: , name: universeInfo  
    // Error parsing type: , name: experimentInfo  
    // Error parsing type: , name: bucketInfo  
}  
  
@end
```

-》能看出：

- 类 \_TtC14MainAppLibrary20ExperimentAssignment
  - 有3个属性
    - universeInfo
    - experimentInfo
    - bucketInfo

但是具体内部偏移量，不清楚。

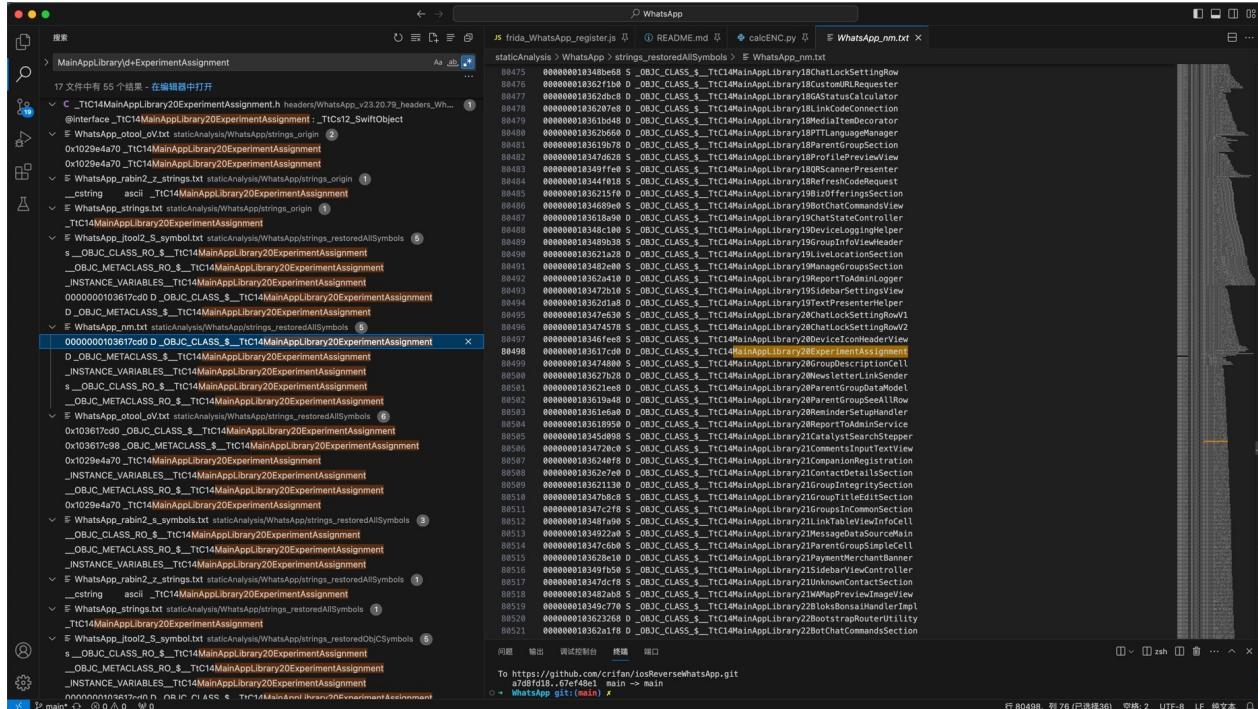
然后去找

- 类 \_TtC14MainAppLibrary20ExperimentAssignment

然后通过之前导出的静态字符串资源中，搜索

MainAppLibrary\d+ExperimentAssignment

而找到的：



- /Users/crfan/dev/dev\_root/iosReverse/WhatsApp/staticAnalysis/WhatsApp/strings\_restoredAllSymbols.ls/WhatsApp\_nm.txt

00000000103617cd0 D \_OBJC\_CLASS\_\$\_TtC14MainAppLibrary20ExperimentAssignment

- /Users/crfan/dev/dev\_root/iosReverse/WhatsApp/staticAnalysis/WhatsApp/strings\_restoredAllSymbols.ls/WhatsApp\_otool\_oV.txt

```
000000001030a7aa8 0x103617cd0 _OBJC_CLASS_$_TtC14MainAppLibrary20ExperimentAssignment
isa          0x103617c98 _OBJC_METACLASS_$_TtC14MainAppLibrary20ExperimentAssignment
superclass 0x0 _OBJC_CLASS_$_TtCs12_SwiftObject

data        0x1030dfe79 Swift class

name        0x1029e4a70 _TtC14MainAppLibrary20ExperimentAssignment

ivars       0x1030dfe10 __OBJC_$_INSTANCE_VARIABLES__TtC14MainAppLibrary20ExperimentAssignment
```

找到了：

- OBJC\_CLASS\_\_\_\_TtC14MainAppLibrary20ExperimentAssignment

然后去IDA中看看：

- `OBJC_CLASS____TtC14MainAppLibrary20ExperimentAssignment`
  - 偏移量=地址: `0x103617cd0`

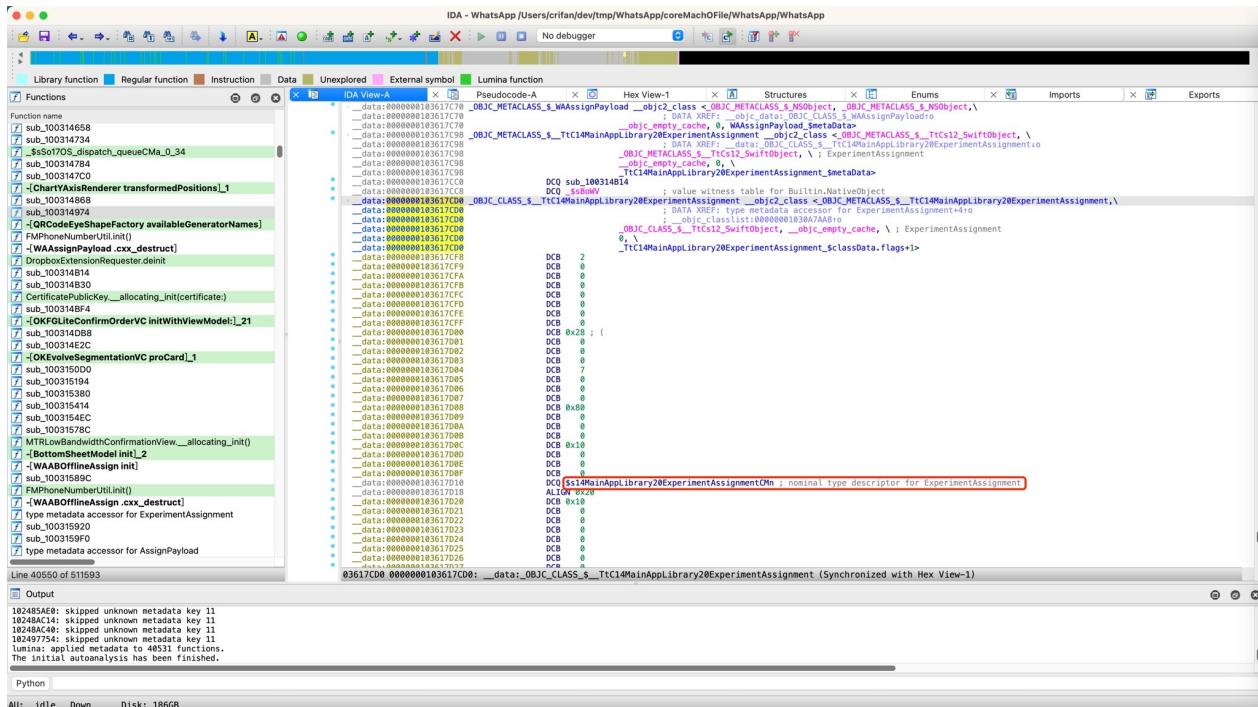
而找到

```

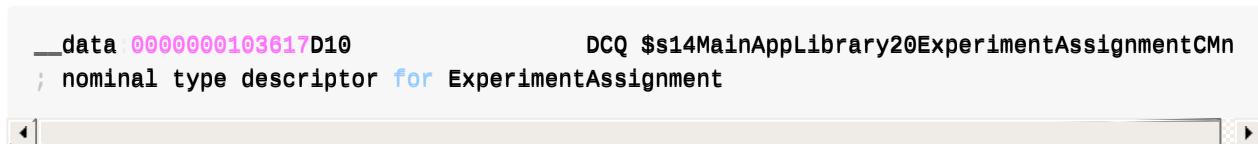
__data:0000000103617CD0 _OBJC_CLASS_<__TtC14MainAppLibrary20ExperimentAssignment __objc
2_class <_OBJC_METACLASS_<__TtC14MainAppLibrary20ExperimentAssignment,\

__data:0000000103617CD0 ; DATA XREF: type metad
ata accessor for ExperimentAssignment+410
__data:0000000103617CD0 ; __objc_classlist:0000
0001030A7AA8↑0
__data:0000000103617CD0 _OBJC_CLASS_<__TtCs12_SwiftObjec
t, __objc_empty_cache, \ ; ExperimentAssignment
__data:0000000103617CD0
__data:0000000103617CD0 _TtC14MainAppLibrary20Experiment
Assignment_classData.flags 1>
__data:0000000103617CF8 DCB 2
__data:0000000103617CF9 DCB 0
__data:0000000103617CFA DCB 0
__data:0000000103617CFB DCB 0
__data:0000000103617CFC DCB 0
__data:0000000103617CFD DCB 0
__data:0000000103617CFE DCB 0
__data:0000000103617CFF DCB 0
__data:0000000103617D00 DCB 0x28 ; (
__data:0000000103617D01 DCB 0
__data:0000000103617D02 DCB 0
__data:0000000103617D03 DCB 0
__data:0000000103617D04 DCB 7
__data:0000000103617D05 DCB 0
__data:0000000103617D06 DCB 0
__data:0000000103617D07 DCB 0
__data:0000000103617D08 DCB 0x80
__data:0000000103617D09 DCB 0
__data:0000000103617D0A DCB 0
__data:0000000103617D0B DCB 0
__data:0000000103617D0C DCB 0x10
__data:0000000103617D0D DCB 0
__data:0000000103617D0E DCB 0
__data:0000000103617D0F DCB 0
__data:0000000103617D10 DCQ <s14MainAppLibrary20ExperimentAssignmentCMn
; nominal type descriptor for ExperimentAssignment
__data:0000000103617D18 ALIGN 0x20
__data:0000000103617D20 DCB 0x10
__data:0000000103617D21 DCB 0

```



继而找到：



双击

- `$s14MainAppLibrary20ExperimentAssignmentCMn`

进去看到定义：

即：

- 类 `MainAppLibrary.ExperimentAssignment` 的具体定义

IDA - WhatsApp /Users/crifan/dev/tmp/WhatsApp/coreMachOFile/WhatsApp/WhatsApp

Functions Regular function Instruction Data Unexplored External symbol Lumina function

File name sub\_100314958 sub\_100314734 \$So7C0\$dispatch\_queueCMA\_0\_34 sub\_100314784 sub\_1003147C0 {-ChartAxisRenderer transformedPositions}\_1 sub\_100314829 {-QRCodeEyeShapeFactory availableGeneratorNames} FMPhoneNumberUtilInit() -WAAAssignPayload\_cxx\_destruct DropboxExtensionRequester\_deinit sub\_100314814 sub\_100314830 -CertificateKey\_\_allocating\_init(certificate) sub\_100314854 {-OKGluieConfirmOrderC initWithViewModel}\_21 sub\_100314D88 sub\_100314E2C {-KEvolveSegmentationVC proCard}\_1 sub\_1003150D0 sub\_100315104 sub\_100315380 sub\_100315414 sub\_1003154EC sub\_10031578C MTRLowBandwidthConfirmationView\_\_allocating\_init() {-BottomSheetModel [init]\_2 {-WAABofflineAssign\_init} sub\_100315920 sub\_1003159F0 type metadata accessor for AssignPayload

Line 40560 of 511593

Output

```

102485AE0: skipped unknown metadata key 11
10248AC14: skipped unknown metadata key 11
10249754: skipped unknown metadata key 11
102497754: skipped unknown metadata key 11
lumina: applied metadata to 40531 functions.
The initial autoanalysis has been finished.

```

Python

AU: idle Down Disk: 186GB

```

__const:0000000102606D27          DCB      0
__const:0000000102606D28 _OBJC_IVAR_S_TtC14MainAppLibrary20ExperimentAssignment.univer
seInfo DCD 0x10
__const:0000000102606D28          ; DATA XREF: __objc_co
nst:00000001030DFE18+0
__const:0000000102606D2C          ALIGN 0x10
__const:0000000102606D30 _OBJC_IVAR_S_TtC14MainAppLibrary20ExperimentAssignment.experi
mentInfo DCD 0x18
__const:0000000102606D30          ; DATA XREF: __objc_co
nst:00000001030DFE38+0
__const:0000000102606D34          ALIGN 8
__const:0000000102606D38 _OBJC_IVAR_S_TtC14MainAppLibrary20ExperimentAssignment.bucket
Info DCD 0x20
__const:0000000102606D38          ; DATA XREF: __objc_co
nst:00000001030DFE58+0
__const:0000000102606D3C          ALIGN 0x20
__const:0000000102606D40 aExperimentassi_0 DCB "ExperimentAssignment",0
__const:0000000102606D40          ; DATA XREF: __const:n
ominal type descriptor for ExperimentAssignment!o
__const:0000000102606D55          ALIGN 8
__const:0000000102606D58 ; nominal type descriptor for MainAppLibrary.ExperimentAssignm
ent
__const:0000000102606D58 s14MainAppLibrary20ExperimentAssignmentCMn ClassDescriptor <0
x80000050, s14MainAppLibraryMXM - ., \
__const:0000000102606D58          ; DATA XREF: __swift5_
types:0000000102C7D2AC+0
__const:0000000102606D58          ; __data:0000000103617
D10+0 ...
__const:0000000102606D58          aExperimentassi_0 - ., \ ; ty
pe metadata accessor for ExperimentAssignment ...
__const:0000000102606D58          ; s14MainAppLibrary20Experimen
tAssignmentCMA - ., \
__const:0000000102606D58          ; s14MainAppLibrary20Experimen
tAssignmentCMn.FieldDescriptor - ., \

```

```

__const:0000000102606D58          $14MainAppLibrary20Experiment
tAssignmentCMn.SuperclassType = .,\n
__const:0000000102606D58          2, 0xE, 4, 3>
__const:0000000102606D80          DCB  0xA
__const:0000000102606D81          DCB  0

```

此处其实就可以看出：

- MainAppLibrary.ExperimentAssignment == \_TtC14MainAppLibrary20ExperimentAssignment
  - [+0x10] = universeInfo
  - [+0x18] = experimentInfo
  - [+0x20] = bucketInfo

后续去（Xcode调试中的lldb调试界面）查看内存中的数据，就是可以对得上的：

```

(lldb) x/8gx $x20
0x2833c1140: 0x0000000105dbbcd0 0x0000000200000003
0x2833c1150: 0x00000002819ecea0 0x00000002819ecf00
0x2833c1160: 0x00000002833b9b00 0x0000000000000000
0x2833c1170: 0x00000001efe1f820 0x0000000000000000
(lldb) po 0x0000000105dbbcd0
MainAppLibrary.ExperimentAssignment
(lldb) po 0x00000002819ecea0
MainAppLibrary.OfflineAB.UniverseInfo
(lldb) po 0x00000002819ecf00
MainAppLibrary.OfflineAB.ExperimentInfo
(lldb) po 0x00000002833b9b00
MainAppLibrary.OfflineAB.BucketInfo

```

如此，即可推断和确认和计算出此处的值：

- MainAppLibrary.ExperimentAssignment == 0x00000002833c1140 == 0x2833c1140
  - [+0x10] = universeInfo
    - 0x00000002819ecea0
  - [+0x18] = experimentInfo
    - 0x00000002819ecf00
  - [+0x20] = bucketInfo
    - 0x00000002833b9b00

而至此，理论上，其实可以通过：

IDA汇编代码 + IDA伪代码

能看出：

```

v1 = v0[2];
v2 = v0[3];
v6 = v0[4];

```

==

—text:0000000100314990	LDR	X8, [X20,#0x10]
—text:00000001003149B4	LDR	X9, [X19,#0x18]
—text:0000000100314A0C	LDR	X8, [X19,#0x20]

其实就分别对应着上面的：

- [+0x10] = universeInfo
  - 0x00000002819ecea0
- [+0x18] = experimentInfo
  - 0x00000002819ecf00
- [+0x20] = bucketInfo
  - 0x00000002833b9b00

如此，即可去分别：

优化IDA伪代码，给变量重命名：

IDA View-A    Pseudocode-A    Pseudocode-B

```

1 int64 StringAppend_100314974()
2 {
3     _QWORD *v0; // x20
4     __int64 universeInfo; // x8
5     __int64 experimentInfo; // x9
6     __int64 v3; // x22
7     __int64 v4; // x23
8     __int64 v5; // x21
9     __int64 v6; // x8
10    __int64 v7; // x19
11    __int64 v8; // x22
12    __int64 v9; // x21
13    __int64 v10; // x21
14    __int64 v12; // [xsp+0h] [xbp-40h]
15    __int64 v13; // [xsp+8h] [xbp-38h]
16
17    universeInfo = v0[2];
18    v12 = *(_QWORD *) (universeInfo + 0x10);
19    v13 = *(_QWORD *) (universeInfo + 0x18);
20    ((void (*)(void)) swift_bridgeObjectRetain)();
21    String.append(_ :)(0x7CLL, 0xE100000000000000LL); // 0x7C = '|'
22    experimentInfo = v0[3];
23    v3 = *(_QWORD *) (experimentInfo + 0x10);
24    v4 = *(_QWORD *) (experimentInfo + 0x18);
25    swift_bridgeObjectRetain(v13);
26    String.append(_ :)(v3, v4);
27    swift_bridgeObjectRelease(v5);
28    swift_bridgeObjectRetain(v13);
29    String.append(_ :)(0x7CLL, 0xE100000000000000LL); // 0x7C = '|'
30    swift_bridgeObjectRelease(v13);
31    v6 = v0[4];
32    v
33    V Run to cursor F4
34    v
35    S Add execution trace
36    S Add breakpoint F2
37    S
38    r Synchronize with ►
39    C Copy ^C
40    R Rename Ivar... N
41    Set Ivar type... Y
42    Convert to struct *...
43    Create new struct type...
44    Map to another variable... =
45    Jump to xref... X
46    Edit comment...
47    Edit block comment...
48    Hide casts \
49
50 .formIndex(at1
51 _release(_QWORD
52 _retain(_QWORD
53 signPayload_experimentAssignments;

```

(100314A0C)

IDA - WhatsApp.i64 (WhatsApp) /Users/crifan/dev/dev\_root/iosReverse/WhatsApp.ipa/Payload/WhatsApp.app/WhatsApp.i64

Functions Library function Regular function Instruction Data Unexplored External symbol Lumina function

```

Function name
7 {-WAAssignPayload_cxx_destruct}
7 DropboxExtensionRequester.deinit
7 sub_100314B14
7 sub_100315A30
7 Cocos2dPublicKey__allocating_init(certificate)
7 sub_100314BF4
7 {-OKGLiteConfirmOrderVC initWithViewModel_21}
7 getWABOfflineAssignAssign_100314DB8
7 generateWABOfflineAssignAssign_100314E2C
7 {-WAABofflineAssign_assign}
7 sub_100315A40
7 sub_100315B04
7 sub_100315B30
7 sub_100315B44
7 sub_100315EC
7 sub_10031578C
7 MTRLowBandwidthConfirmationView__allocating_init()
7 {-BottomSheetModel_init_2}
7 {-WAABofflineAssign_init}
7 $sfpFindv17FMPhoneNumberUtilICAfcyfc_85_589C
7 FMPhoneNumberUtilInit()
7 {-WAABofflineAssign_cxx_destruct}
7 MainAppLibrary_ExperimentAssignment_init_5904
7 sub_100315920
7 sub_1003159F0
7 type metadata accessor for AssignPayload
7 type metadata accessor for ABOfflineAssign
7 outlined init with copy of Any?
7 outlined destroy of Any?
7 sub_100315B24
7 outlined destroy of Response
7 sub_100315B70
7 sub_100315B8B
7 sub_100315BC0
Line 40550 of 518612

```

Output

```

10:444C98: using guessed type __int64 __fastcall Array<__int64>(__int64) __WORD, __WORD;
10:444C98: using guessed type __int64 __fastcall swift_release(_QWORD);
10:444C94: using guessed type __int64 __fastcall swift_retain(_QWORD);
10:3617C30: using guessed type __int64 OBJS_IVAR_WAAssignPayload_experimentAssignments;
10:331454C: restored microcode from idb
10:314A5C: restored pseudocode from idb
10:444C98: using guessed type __int64 __fastcall Array<__int64>(__int64) __WORD, __WORD;
10:444C98: using guessed type __int64 __fastcall swift_bridgeToObjectRelease(_QWORD);
Python
AU: idle Down Disk: 194GB

```

IDA - WhatsApp.i64 (WhatsApp) /Users/crifan/dev/dev\_root/iosReverse/WhatsApp.ipa/Payload/WhatsApp.app/WhatsApp.i64

Functions Library function Regular function Instruction Data Unexplored External symbol Lumina function

```

Function name
7 {-WAAssignPayload_cxx_destruct}
7 DropboxExtensionRequester.deinit
7 sub_100314B14
7 sub_100315A30
7 Cocos2dPublicKey__allocating_init(certificate)
7 sub_100314BF4
7 {-OKGLiteConfirmOrderVC initWithViewModel_21}
7 getWABOfflineAssignAssign_100314DB8
7 generateWABOfflineAssignAssign_100314E2C
7 {-WAABofflineAssign_assign}
7 sub_100315A40
7 sub_100315B04
7 sub_100315B30
7 sub_100315B44
7 sub_100315EC
7 sub_10031578C
7 MTRLowBandwidthConfirmationView__allocating_init()
7 {-BottomSheetModel_init_2}
7 {-WAABofflineAssign_init}
7 $sfpFindv17FMPhoneNumberUtilICAfcyfc_85_589C
7 FMPhoneNumberUtilInit()
7 {-WAABofflineAssign_cxx_destruct}
7 MainAppLibrary_ExperimentAssignment_init_5904
7 sub_100315920
7 sub_1003159F0
7 type metadata accessor for AssignPayload
7 type metadata accessor for ABOfflineAssign
7 outlined init with copy of Any?
7 outlined destroy of Any?
7 sub_100315B24
7 outlined destroy of Response
7 sub_100315B70
7 sub_100315B8B
7 sub_100315BC0
Line 40550 of 518612

```

Output

```

10:444K68: using guessed type __int64 __fastcall Array<__int64>(__int64) __WORD, __WORD;
10:444K68: using guessed type __int64 __fastcall swift_release(_QWORD);
10:444C94: using guessed type __int64 __fastcall swift_retain(_QWORD);
10:3617C30: using guessed type __int64 OBJS_IVAR_WAAssignPayload_experimentAssignments;
10:331454C: restored microcode from idb
10:314A5C: restored pseudocode from idb
10:444C98: using guessed type __int64 __fastcall Array<__int64>(__int64) __WORD, __WORD;
10:444C98: using guessed type __int64 __fastcall swift_bridgeToObjectRelease(_QWORD);
10:444C98: using guessed type __int64 __fastcall swift_release(_QWORD);
10:444C94: using guessed type __int64 __fastcall swift_retain(_QWORD);
10:3617C30: using guessed type __int64 OBJS_IVAR_WAAssignPayload_experimentAssignments;
10:331454C: restored microcode from idb
10:314A5C: restored pseudocode from idb
10:444C98: using guessed type __int64 __fastcall Array<__int64>(__int64) __WORD, __WORD;
10:444C98: using guessed type __int64 __fastcall swift_bridgeToObjectRelease(_QWORD);
Python
AU: idle Down Disk: 194GB

```

此时，IDA伪代码就变成了：

```

__int64 StringAppend_100314974()
{
    __QWORD v0; // x20
    __int64 universeInfo; // x8
    __int64 experimentInfo; // x9
    __int64 v3; // x22
    __int64 v4; // x23
    __int64 v5; // x21
    __int64 bucketInfo; // x8
    __int64 v7; // x19
    __int64 v8; // x22
}

```

```

__int64 v9; // x21
__int64 v10; // x21
__int64 v12; // [xsp+0h] [xbp-40h]
__int64 v13; // [xsp+8h] [xbp-38h]

universeInfo = v0[2];
v12 = *(_QWORD*)(universeInfo + 0x10);
v13 = *(_QWORD*)(universeInfo + 0x18);
((void (*)(void))swift_bridgeObjectRetain)();
String.append(_)(0x7CLL, 0xE100000000000000LL); // 0x7C = '|'
experimentInfo = v0[3];
v3 = *(_QWORD*)(experimentInfo + 0x10);
v4 = *(_QWORD*)(experimentInfo + 0x18);
swift_bridgeObjectRetain(v13);
String.append(_)(v3, v4);
swift_bridgeObjectRelease(v5);
swift_bridgeObjectRetain(v13);
String.append(_)(0x7CLL, 0xE100000000000000LL); // 0x7C = '|'
swift_bridgeObjectRelease(v13);
bucketInfo = v0[4];
v7 = *(_QWORD*)(bucketInfo + 0x10);
v8 = *(_QWORD*)(bucketInfo + 0x18);
swift_bridgeObjectRetain(v9);
String.append(_)(v7, v8);
swift_bridgeObjectRelease(v10);
return v12;
}

```

-》很明显，代码逻辑，就慢慢的，逐渐的，更加清晰了。

而后续的逻辑，看起来，主要就是：

```

String.append(_)(0x7CLL, 0xE100000000000000LL); // 0x7C = '|'

String.append(_)(v3, v4);

String.append(_)(0x7CLL, 0xE100000000000000LL); // 0x7C = '|'

String.append(_)(v7, v8);

```

本来以为很简单，直接去：

找到具体的：原先字符串和拼接字符串，即可

之前调试已经知道，希望的最终的值是：

```

Swift.__SwiftDeferredNSArray 0x282642100 (
    dummy_aa_offline_rid_universe_ios dummy_aa_offline_rid_experiment_ios | control,
    hide_link_device_button_release_rollout_universe hide_link_device_button_release_roll
out_experiment | control,
    ios_prod_latam_tos_reg_universe ios_prod_latam_tos_reg_experiment | test
)

```

所以，此处第一轮，应该是：

- `dummy_aa_offline_rid_universe_ios|dummy_aa_offline_rid_experiment_ios|control,`

而去掉竖杠 |，则应该分别是：

- `dummy_aa_offline_rid_universe_ios`
- `dummy_aa_offline_rid_experiment_ios`
- `control`

所以就去调试

结果去调试时，却发现找不到我们要的字符串的值

和预想的逻辑对不上

比如，第一个，不是：

- `dummy_aa_offline_rid_universe_ios`

首先是：

```
0x102ab89a0 <+ 44 :  mov    x20, sp
0x102ab89a4 <+ 48 :  mov    w0, #0x7c
0x102ab89a8 <+ 52 :  mov    x1, # 0x1f00000000000000
0x102ab89ac <+ 56 :  bl     0x104c44824           ; symbol stub for: String.append
d(_:_)
```

此时值：

```
(lldb) reg r x0 x1
x0 = 0x0000000000000000
x1 = 0xe100000000000000
```

-》好像此处Swift函数 `String.append(_:_)` 中，`x0` 和 `x1`，就不是我们要的普通的字符串

找不到我们要的：`dummy_aa_offline_rid_universe_ios`

后来自己研究出了，看起来是放到了：`x20 = sp` 中了？

去看看，果然还是有字符串值的：

其次是，虽然有字符串值，但是也还不是我们要的值：

```
(lldb) reg r x20 sp
x20 = 0x000000016d658c20
sp = 0x000000016d658c20
(lldb) x/8gx 0x000000016d658c20
0x16d658c20: 0xd000000000000021 0x80000001051946f0
0x16d658c30: 0x0000000000000002 0x00000002833c1140
0x16d658c40: 0x0000000105da1628 0x0000000105dbbcd0
0x16d658c50: 0x00000002833c1140 0x00000002828d0780
(lldb) po 0x80000001051946f0
9223372041235285744
(lldb) x/s 0x80000001051946f0
```

```
0x1051946f0: "dummy_aa_offline_user_rid_ios"
```

即，此处：

- 打印出的字符串是：
  - dummy\_aa\_offline\_user\_rid\_ios
- 希望的字符串是
  - dummy\_aa\_offline\_rid\_universe\_ios

所以就怀疑，难道是，Swift的字符串拼接函数： `String.append(_:)`， 内部有额外的逻辑？

经过一番研究：

- 【未解决】iOS逆向WhatsApp：Swift函数`String.append(_:)`的字符串拼接的实现逻辑

没有发现啥特殊的逻辑。

正在一筹莫展之际，无意间发现别的帖子：

[Swift 里字符串（十）修改字符串 – huahuahu \(wordpress.com\)](#)

好像是Swift的String，比较特殊？

所以再转去研究：

Swift中的String的类的具体定义

- [String字符串 · iOS逆向：Swift逆向](#)

最终彻底搞懂了：

核心就一句话：

- Swift中(Native的)LargeString中：真正字符串的地址= `objectAddr + 0x20`

所以此处是：

```
(lldb) p/x 0x80000001051946f0 + 0x20
(unsigned long) 0x8000000105194710
(lldb) po 0x8000000105194710
9223372041235285776
(lldb) x/s 0x8000000105194710
0x105194710: "dummy_aa_offline_rid_universe_ios"
```

如此，最终才彻底搞清楚了：

- Swift中字符串拼接函数： `String.append(_:)`， 内部没有特殊逻辑
  - 但是传入的参数，此处其实是多个来源
    - 2个寄存器：`x0` 和 `x1`
    - 其实保存的好像也是：Swift中的small string，用2个寄存器=2个64位保存对应的值
    - sp堆栈
    - 保存的是Swift的large string，也是2个64位的地址保存其值

综合起来就是：

- Swift的String.append(\_:)
- originSwiftString.append(toAppendSwiftString)
  - originSwiftString: 放在了sp堆栈中
    - 只不过也要2个64位地址保存: [sp]、[sp+0x08]
  - toAppendSwiftString: 放在了寄存器中
    - 只不过要2个寄存器保存: x0、x1

最终，才真正搞懂，此处后续字符串的拼接的逻辑。

注：

关于另外的几个类的属性字段的定义，详见

- /Users/crifan/dev/dev\_root/iosReverse/WhatsApp/WhatsApp\_v23.25.85/headers/headers\_WhatsApp\_v23.25.85\_WhatsApp\_paradiseduo\_dsdump/\_TtC7Catalog24VariantTypeBaseViewModel.h

```
...
0x0010381f460 _TtC014MainAppLibrary90fflineAB10BucketInfo : Swift._SwiftObject @rpath/libswiftCore.dylib
{
    0x0010  name (0x10)
    0x0020  size (0x8)
    0x0028  configList (0x8)
}

0x0010381f510 _TtC014MainAppLibrary90fflineAB14ConfigVariable : Swift._SwiftObject @rpath/libswiftCore.dylib
{
    0x0010  code (0x8)
    0x0018  name (0x10)
    0x0028  value (0x10)
}

0x0010381f5c0 _TtC014MainAppLibrary90fflineAB14ExperimentInfo : Swift._SwiftObject @rpath/libswiftCore.dylib
{
    0x0010  name (0x10)
    0x0020  startTime (0x8)
    0x0028  endTime (0x8)
    0x0030  bucketList (0x8)
    0x0038  userFilter (0x28)
}

...
0x0010381f730 _TtC014MainAppLibrary90fflineAB12UniverseInfo : Swift._SwiftObject @rpath/libswiftCore.dylib
{
    0x0010  name (0x10)
    0x0020  unit (0x10)
    0x0030  experimentList (0x8)
    0x0038  userFilter (0x28)
}
```



此处就不赘述了。

然后就可以去计算出此处拼接处的字符串了：

```
(lldb) x/8gx 0x00000002833c1140
0x2833c1140: 0x0000000105dbbcd0 0x0000000200000003
0x2833c1150: 0x00000002819ecea0 0x00000002819ecf00
0x2833c1160: 0x00000002833b9b00 0x0000000000000000
0x2833c1170: 0x00000001ef1f820 0x0000000000000000
(lldb) po 0x00000002819ecea0
MainAppLibrary.OfflineAB.UniverseInfo
(lldb) po 0x00000002819ecf00
MainAppLibrary.OfflineAB.ExperimentInfo
(lldb) po 0x00000002833b9b00
MainAppLibrary.OfflineAB.BucketInfo

(lldb) x/6gx 0x00000002819ecea0
0x2819ecea0: 0x0000000105dc0398 0x0000000200000003
0x2819eceb0: 0xd000000000000021 0x80000001051946f0
0x2819ecec0: 0x6469725f72657375 0xe800000000000000
(lldb) x/6gx 0x00000002819ecf00
0x2819ecf00: 0x0000000105dc0238 0x0000000200000003
0x2819ecf10: 0xd000000000000023 0x80000001051946a0
0x2819ecf20: 0x000000006316eff0 0x0000000065e82280
(lldb) x/6gx 0x00000002833b9b00
0x2833b9b00: 0x0000000105dc00e8 0x0000000200000003
0x2833b9b10: 0x006c6f72746e6f63 0xe700000000000000
0x2833b9b20: 0x00000000000001388 0x00000002833b9b90

(lldb) x/s 0x8000000105194710
0x105194710: "dummy_aa_offline_rid_universe_ios"
(lldb) x/s 0x80000001051946c0
0x1051946c0: "dummy_aa_offline_rid_experiment_ios"
(lldb) p/c 0x006c6f72746e6f63
(long) control\0
```

-》

- "dummy\_aa\_offline\_rid\_universe\_ios"
- "dummy\_aa\_offline\_rid\_experiment\_ios"
- "control"

由此得到最终拼接后的字符串：

- dummy\_aa\_offline\_rid\_universe\_ios|dummy\_aa\_offline\_rid\_experiment\_ios|control

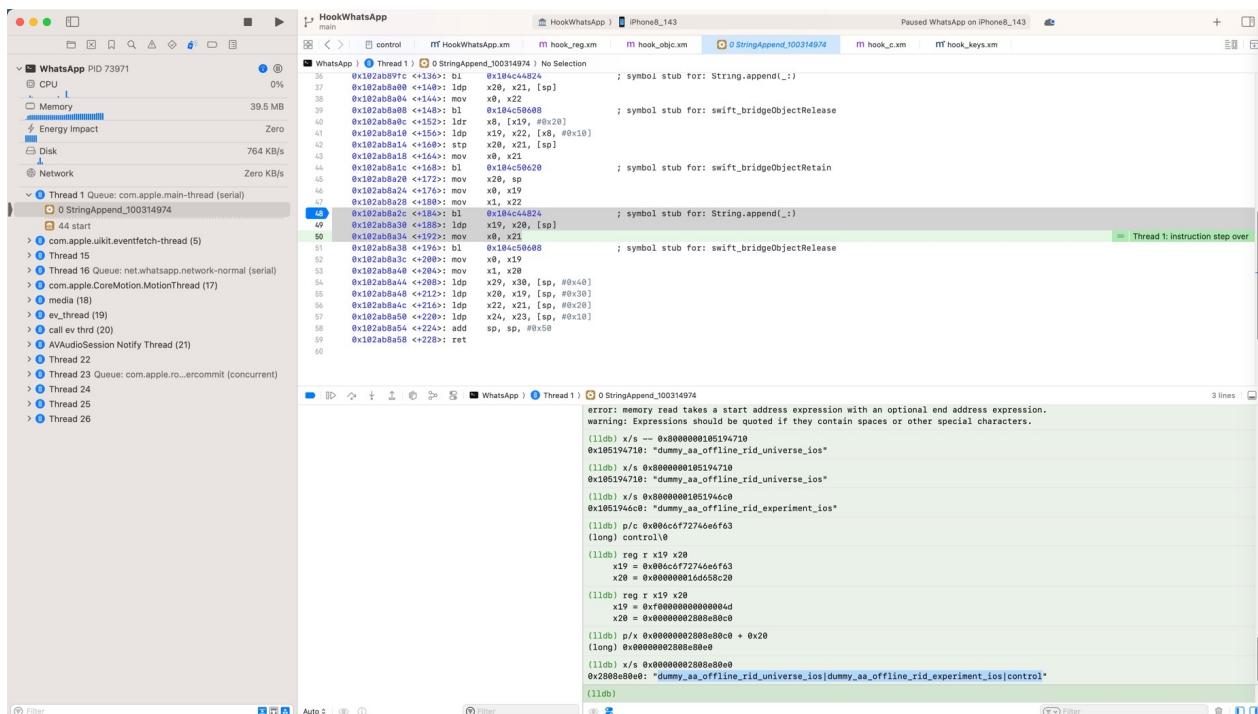
去调试确认和我们算出的值是一样的：

```
0x102ab8a2c < 184 : bl      0x104c44824 ; symbol stub for: String.append
d(_:_)
0x102ab8a30 < 188 : ldp      x19, x20, [sp]
```

```
0x102ab8a34 < 192 : mov      x0, x21
```

->

```
(lldb) reg r x19 x20
x19 = 0xf000000000000000d
x20 = 0x00000002808e80c0
(lldb) p/x 0x00000002808e80c0 + 0x20
(long) 0x00000002808e80e0
(lldb) x/s 0x00000002808e80e0
0x2808e80e0: "dummy_aa_offline_rid_universe_ios|dummy_aa_offline_rid_experiment_ios|control"
```



是一样的。

如此，继续去优化IDA伪代码：

去改名：

```
__int64 StringAppend_100314974()
{
    _QWORD curMainAppLibraryExperimentAssignment; // x20
    __int64 universeInfo; // x8
    __int64 experimentInfo; // x9
    __int64 experimentInfoNameStringObjPart1; // x22
    __int64 experimentInfoNameStringObjPart2; // x23
    __int64 v5; // x21
    __int64 bucketInfo; // x8
    __int64 bucketInfoNameStringObjPart1; // x19
    __int64 bucketInfoNameStringObjPart2; // x22
    __int64 v9; // x21
    __int64 v10; // x21
    __int64 userInfoNameStringObjPart1; // [xsp+0h] [xbp-40h]
```

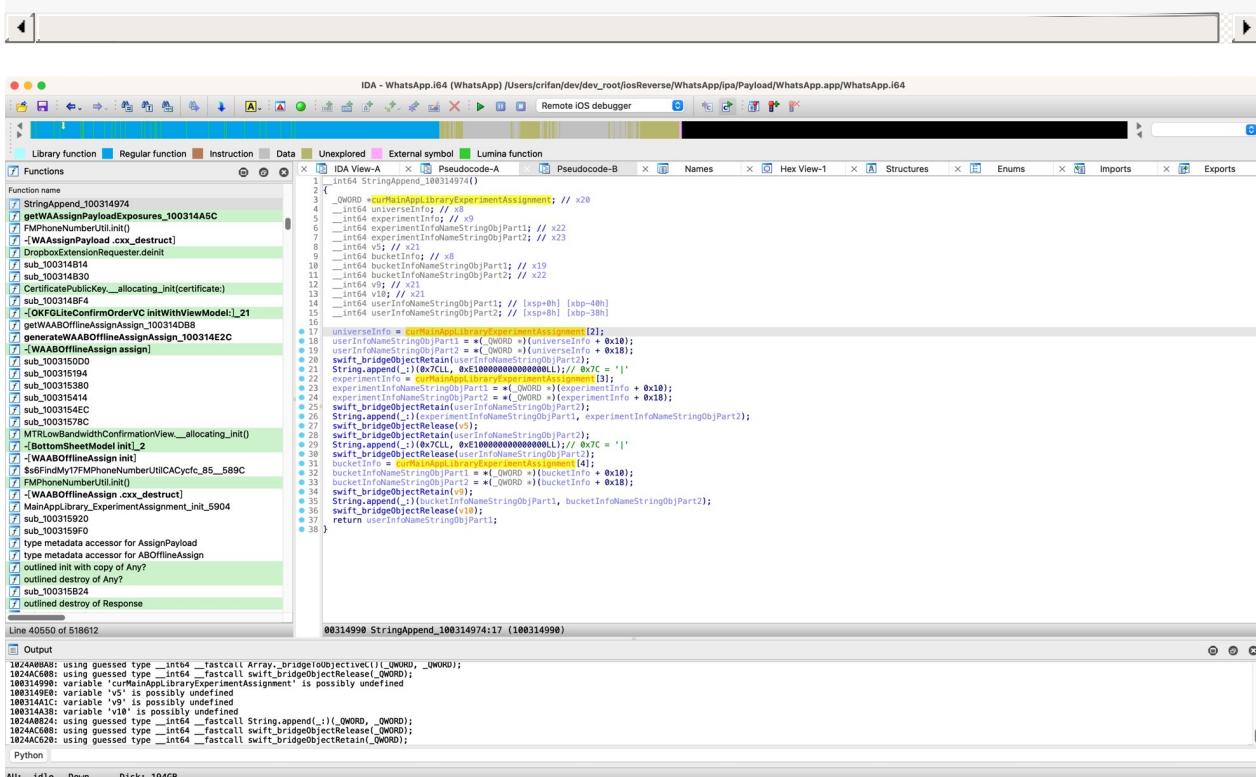
```

__int64 userInfoNameStringObjPart2; // [xsp+8h] [xbp-38h]

universeInfo = curMainAppLibraryExperimentAssignment[2];
userInfoNameStringObjPart1 = (*(_QWORD *)(&universeInfo + 0x10));
userInfoNameStringObjPart2 = (*(_QWORD *)(&universeInfo + 0x18));
swift_bridgeObjectRetain(userInfoNameStringObjPart2);
String.append(_)(0x7CLL, 0xE100000000000000LL); // 0x7C = '|'
experimentInfo = curMainAppLibraryExperimentAssignment[3];
experimentInfoNameStringObjPart1 = (*(_QWORD *)(&experimentInfo + 0x10));
experimentInfoNameStringObjPart2 = (*(_QWORD *)(&experimentInfo + 0x18));
swift_bridgeObjectRetain(userInfoNameStringObjPart2);
String.append(_)(experimentInfoNameStringObjPart1, experimentInfoNameStringObjPart2);

swift_bridgeObjectRelease(v5);
swift_bridgeObjectRetain(userInfoNameStringObjPart2);
String.append(_)(0x7CLL, 0xE100000000000000LL); // 0x7C = '|'
swift_bridgeObjectRelease(userInfoNameStringObjPart2);
bucketInfo = curMainAppLibraryExperimentAssignment[4];
bucketInfoNameStringObjPart1 = (*(_QWORD *)(&bucketInfo + 0x10));
bucketInfoNameStringObjPart2 = (*(_QWORD *)(&bucketInfo + 0x18));
swift_bridgeObjectRetain(v9);
String.append(_)(bucketInfoNameStringObjPart1, bucketInfoNameStringObjPart2);
swift_bridgeObjectRelease(v10);
return userInfoNameStringObjPart1;
}

```



如此，其实代码逻辑已经很清楚了。

不过另外想到了：

如果IDA此处能识别：

- 类MainAppLibrary\+ExperimentAssignment

就好了？就可以自动识别出：具体属性和字段了

而去试了试，前面搜到的，也确认IDA有的：

- 类名：\_TtC14MainAppLibrary20ExperimentAssignment

即给此处变量更改类型type为：

- \_TtC14MainAppLibrary20ExperimentAssignment\*

操作步骤：

```

1 _int64 StringAppend_100314974()
2 {
3     _QWORD *curMainAppLibraryExperimentAssignment; // x20
4     _int64 universeInfo; // x8
5     _int64 experimentInfo; // x9
6     _int64 experimentInfoNameStringObjPart1; // x19
7     _int64 experimentInfoNameStringObjPart2; // x22
8     _int64 v5; // x21
9     _int64 bucketInfo; // x8
10    _int64 bucketInfoNameStringObjPart1; // x19
11    _int64 bucketInfoNameStringObjPart2; // x22
12    _int64 v9; // x21
13    _int64 v10; // x21
14    _int64 userInfoNameStringObjPart1; // [xsp+]
15    _int64 userInfoNameStringObjPart2; // [xsp+]
16
17    universeInfo = curMainAppLibraryExperimentAssignment;
18    userInfoNameStringObjPart1 = *( _QWORD * )( universeInfo );
19    userInfoNameStringObjPart2 = *( _QWORD * )( userInfoNameStringObjPart1 );
20    swift_bridgeObjectRetain( userInfoNameStringObjPart1 );
21    String.append( _ : ) ( 0x7CLL, 0xE100000000000000 );
22    experimentInfo = curMainAppLibraryExperimentAssignment;
23    experimentInfoNameStringObjPart1 = *( _QWORD * )( experimentInfo );
24    experimentInfoNameStringObjPart2 = *( _QWORD * )( experimentInfoNameStringObjPart1 );
25    swift_bridgeObjectRetain( userInfoNameStringObjPart2 );
26    String.append( _ : )( experimentInfoNameStringObjPart1 );
27    swift_bridgeObjectRelease( v5 );
28    swift_bridgeObjectRetain( userInfoNameStringObjPart2 );
29    String.append( _ : )( 0x7CLL, 0xE100000000000000 ); // 0x7C = '|'
30    swift_bridgeObjectRelease( userInfoNameStringObjPart2 );
31    bucketInfo = curMainAppLibraryExperimentAssignment[ 4 ];
32    bucketInfoNameStringObjPart1 = *( _QWORD * )( bucketInfo + 0x10 );
33    bucketInfoNameStringObjPart2 = *( _QWORD * )( bucketInfo + 0x18 );
34    swift_bridgeObjectRetain( v9 );
35    String.append( _ : )( bucketInfoNameStringObjPart1, bucketInfoNameStringObjPart2 );
36    swift_bridgeObjectRelease( v10 );
37    return userInfoNameStringObjPart1;
38 }

```

IDA - WhatsApp.i64 (WhatsApp) /Users/crifan/dev\_root/iosReverse/WhatsApp/ipa/Payload/WhatsApp.app/WhatsApp.i64

Remote iOS debugger

Unexplored External symbol Lumina function

```

1 int64 StringAppend_100314974()
2 {
3     _QWORD *curMainAppLibraryExperimentAssignment; // x20
4     _int64 universeInfo; // x8
5     _int64 experimentInfo; // x9
6     _int64 experimentInfoNameStringObjPart1; // x
7     _int64 experimentInfoNameStringObjPart2; // x
8     _int64 v5; // x21
9     _int64 bucketInfo; // x8
10    _int64 bucketInfoNameStringObjPart1; // x19
11    _int64 bucketInfoNameStringObjPart2; // x22
12    _int64 v9; // x21
13    _int64 v10; // x21
14    _int64 userInfoNameStringObjPart1; // [xsp+0h]
15    _int64 userInfoNameStringObjPart2; // [xsp+0h] Lwp_30h
16
17    universeInfo = curMainAppLibraryExperimentAssignment[2];
18    userInfoNameStringObjPart1 = *( _QWORD *) (universeInfo + 0x10);
19    userInfoNameStringObjPart2 = *( _QWORD *) (universeInfo + 0x18);
20    swift_bridgeObjectRetain(userInfoNameStringObjPart2);
21    String.append(_);(0x7CALL, 0xE100000000000000L); // 0x7C = '!'
22    experimentInfo = curMainAppLibraryExperimentAssignment[3];
23    experimentInfoNameStringObjPart1 = *( _QWORD *) (experimentInfo + 0x10);
24    experimentInfoNameStringObjPart2 = *( _QWORD *) (experimentInfo + 0x18);
25    swift_bridgeObjectRetain(userInfoNameStringObjPart2);
26    String.append(_);
27    swift_bridgeObjectRelease(v5);
28    swift_bridgeObjectRetain(userInfoNameStringObjPart2);
29    String.append(_);(0x7CALL, 0xE100000000000000L); // 0x7C = '!'
30    swift_bridgeObjectRelease(userInfoNameStringObjPart2);
31    bucketInfo = curMainAppLibraryExperimentAssignment[4];
32    bucketInfoNameStringObjPart1 = *( _QWORD *) (bucketInfo + 0x10);
33    bucketInfoNameStringObjPart2 = *( _QWORD *) (bucketInfo + 0x18);
34    swift_bridgeObjectRetain(v9);
35    String.append(_);(bucketInfoNameStringObjPart1, bucketInfoNameStringObjPart2);
36    swift_bridgeObjectRelease(v10);
37    return userInfoNameStringObjPart1;
38 }

```

从：

- `_QWORD *curMainAppLibraryExperimentAssignment`

改为：

- `_TtC14MainAppLibrary20ExperimentAssignment *curMainAppLibraryExperimentAssignment`

IDA - WhatsApp.i64 (WhatsApp) /Users/crifan/dev\_root/iosReverse/WhatsApp/ipa/Payload/WhatsApp.app/WhatsApp.i64

Remote iOS debugger

Unexplored External symbol Lumina function

```

1 int64 StringAppend_100314974()
2 {
3     _QWORD *curMainAppLibraryExperimentAssignment; // x20
4     _int64 universeInfo; // x8
5     _int64 experimentInfo; // x9
6     _int64 experimentInfoNameStringObjPart1; // x
7     _int64 experimentInfoNameStringObjPart2; // x
8     _int64 v5; // x21
9     _int64 bucketInfo; // x8
10    _int64 bucketInfoNameStringObjPart1; // x19
11    _int64 bucketInfoNameStringObjPart2; // x22
12    _int64 v9; // x21
13    _int64 v10; // x21
14    _int64 userInfoNameStringObjPart1; // [xsp+0h]
15    _int64 userInfoNameStringObjPart2; // [xsp+0h] Lwp_30h
16
17    universeInfo = curMainAppLibraryExperimentAssignment[2];
18    userInfoNameStringObjPart1 = *( _QWORD *) (universeInfo + 0x10);
19    userInfoNameStringObjPart2 = *( _QWORD *) (universeInfo + 0x18);
20    swift_bridgeObjectRetain(userInfoNameStringObjPart2);
21    String.append(_);(0x7CALL, 0xE100000000000000L); // 0x7C = '!'
22    experimentInfo = curMainAppLibraryExperimentAssignment[3];
23    experimentInfoNameStringObjPart1 = *( _QWORD *) (experimentInfo + 0x10);
24    experimentInfoNameStringObjPart2 = *( _QWORD *) (experimentInfo + 0x18);
25    swift_bridgeObjectRetain(userInfoNameStringObjPart2);
26    String.append(_);(experimentInfoNameStringObjPart1, experimentInfoNameStringObjPart2);
27    swift_bridgeObjectRelease(v5);
28    swift_bridgeObjectRetain(userInfoNameStringObjPart2);
29    String.append(_);(0x7CALL, 0xE100000000000000L); // 0x7C = '!'
30    swift_bridgeObjectRelease(userInfoNameStringObjPart2);
31    bucketInfo = curMainAppLibraryExperimentAssignment[4];
32    bucketInfoNameStringObjPart1 = *( _QWORD *) (bucketInfo + 0x10);
33    bucketInfoNameStringObjPart2 = *( _QWORD *) (bucketInfo + 0x18);
34    swift_bridgeObjectRetain(v9);
35    String.append(_);(bucketInfoNameStringObjPart1, bucketInfoNameStringObjPart2);
36    swift_bridgeObjectRelease(v10);
37    return userInfoNameStringObjPart1;
38 }

```

00314974 StringAppend\_100314974:3 (100314974)

IDA中可见，的确自动识别成，类的属性=字段=成员的引用了：

IDA - WhatsApp.i64 (WhatsApp) /Users/crifan/dev/dev\_root/iosReverse/WhatsApp/ipa/Payload/WhatsApp.app/WhatsApp.i64

Library function Regular function Instruction Data Unexplored External symbol Lumina function

Functions IDA View-A Pseudocode-A Pseudocode-B Names Hex View-1 Structures Enums Imports Exports

```

Function name
1 StringAppend_100314974
2 _TtC14MainAppLibrary20ExperimentAssignment *curMainAppLibraryExperimentAssignment; // <x20>
3 int64 universeInfo; // <x8>
4 int64 experimentInfo; // <x8>
5 int64 experimentInfoNameStringObjPart1; // <x22>
6 int64 experimentInfoNameStringObjPart2; // <x23>
7 int64 v5; // <x21>
8 int64 bucketInfo; // <x8>
9 int64 bucketInfoNameStringObjPart1; // <x21>
10 int64 bucketInfoNameStringObjPart2; // <x22>
11 int64 v9; // <x21>
12 int64 v10; // <x21>
13 int64 userInfoNameStringObjPart1; // [xsp+0h] [xsp-40h]
14 int64 userInfoNameStringObjPart2; // [xsp+0h] [xsp-38h];
15 int64 v11; // <x21>
16 universeInfo = *(QWORD *)curMainAppLibraryExperimentAssignment->universeInfo;
17 userInfoNameStringObjPart1 = *(QWORD *)(&universeInfo + 0x10);
18 userInfoNameStringObjPart2 = *(QWORD *)(&universeInfo + 0x18);
19 swift_bridgeObjectRetain(userInfoNameStringObjPart1);
20 String.append(_)(0x7CLL, 0xE100000000000000LL); // 0x7C = '|'
21 experimentInfoNameStringObjPart1 = *(QWORD *)(&experimentInfo + 0x10);
22 experimentInfoNameStringObjPart2 = *(QWORD *)(&experimentInfo + 0x18);
23 swift_bridgeObjectRetain(userInfoNameStringObjPart1);
24 String.append(_)(0x7CLL, 0xE100000000000000LL); // 0x7C = '|'
25 experimentInfoNameStringObjPart1 = *(QWORD *)(&experimentInfo + 0x10);
26 experimentInfoNameStringObjPart2 = *(QWORD *)(&experimentInfo + 0x18);
27 swift_bridgeObjectRetain(userInfoNameStringObjPart1);
28 String.append(_)(0x7CLL, 0xE100000000000000LL); // 0x7C = '|'
29 bucketInfoNameStringObjPart1 = *(QWORD *)(&bucketInfo + 0x10);
30 bucketInfoNameStringObjPart2 = *(QWORD *)(&bucketInfo + 0x18);
31 swift_bridgeObjectRetain(userInfoNameStringObjPart1);
32 String.append(_)(0x7CLL, 0xE100000000000000LL); // 0x7C = '|'
33 bucketInfoNameStringObjPart1 = *(QWORD *)(&bucketInfo + 0x10);
34 swift_bridgeObjectRetain(userInfoNameStringObjPart1);
35 String.append(_)(0x7CLL, 0xE100000000000000LL); // 0x7C = '|'
36 swift_bridgeObjectRelease(v10);
37 return userInfoNameStringObjPart1;
38 }

```

Line 40550 of 518612 00314974 StringAppend\_100314974:3 (100314974)

Output

```

10244080: using guessed type __int64 _fastcall Array_BridgeObjective((QWORD, QWORD);
10244080: using guessed type __int64 _fastcall swift_bridgeObjectRelease(QWORD);
100314980: variable 'curMainAppLibraryExperimentAssignment' is possibly undefined
100314980: variable 'v5' is possibly undefined
100314980: variable 'v10' is possibly undefined
100314980: variable 'v11' is possibly undefined
102440824: using guessed type __int64 _fastcall String.append(_)(QWORD, QWORD);
102440828: using guessed type __int64 _fastcall swift_bridgeObjectRelease(QWORD);
102440832: using guessed type __int64 _fastcall swift_bridgeObjectRetain(QWORD);

```

AU: idle Down Disk: 194GB

另外，再去给：

- universeInfo
- experimentInfo
- bucketInfo

也分别去修改类型为对应的类名：

IDA - WhatsApp.i64 (WhatsApp) /Users/crifan/dev/dev\_root/iosReverse/WhatsApp/ipa/Payload/WhatsApp.app/WhatsApp.i64

Unexplored External symbol Lumina function

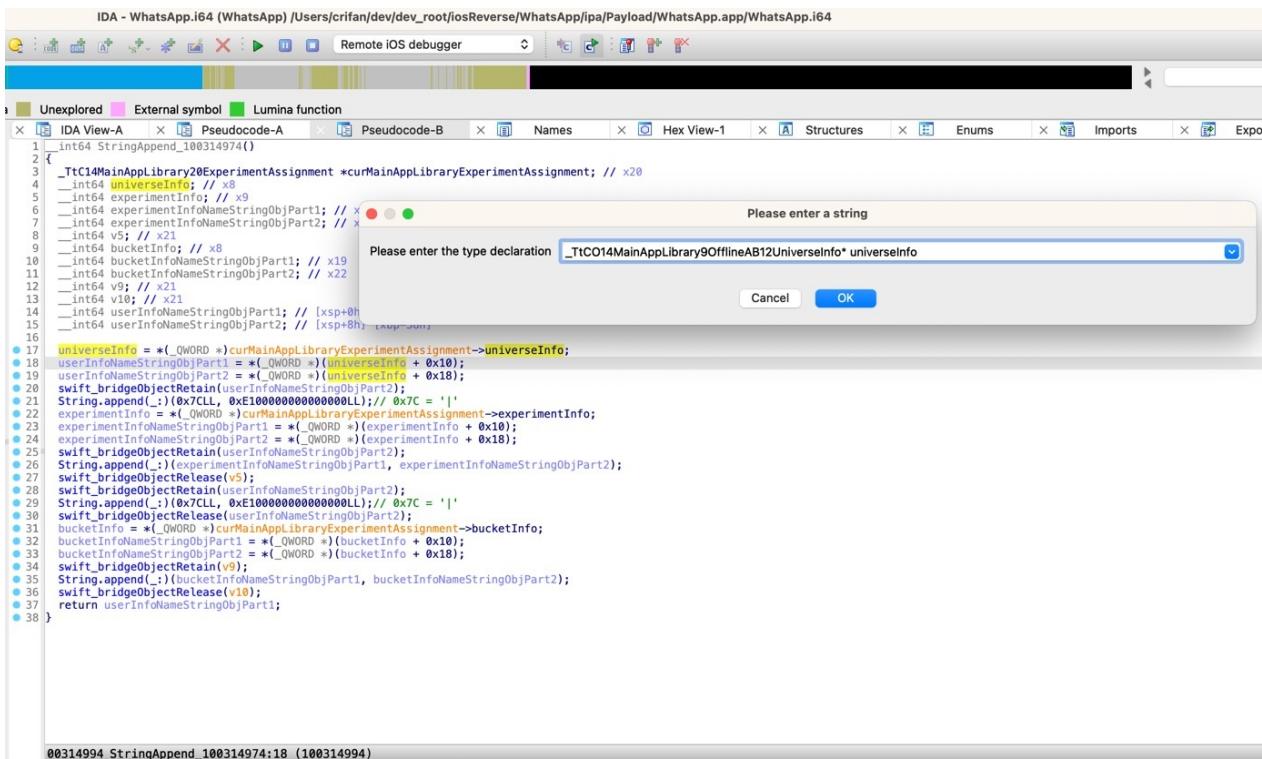
Functions IDA View-A Pseudocode-A Pseudocode-B Names Hex View-1 Structures Enums Imports Exports

```

1 int64 StringAppend_100314974()
2 {
3     _TtC14MainAppLibrary20ExperimentAssignment *curMainAppLibraryExperimentAssignment; // <x20>
4     int64 universeInfo; // <x8>
5     int64 experimentInfo; // <x8>
6     int64 experimentInfoNameStringObjPart1; // <x22>
7     int64 experimentInfoNameStringObjPart2; // <x23>
8     int64 v5; // <x21>
9     int64 bucketInfo; // <x8>
10    int64 bucketInfoNameStringObjPart1; // <x21>
11    int64 bucketInfoNameStringObjPart2; // <x22>
12    int64 v9; // <x21>
13    int64 v10; // <x21>
14    int64 userInfoNameStringObjPart1; // [xsp+0h]
15    int64 userInfoNameStringObjPart2; // [xsp+0h] [xsp-40h];
16    universeInfo = *(QWORD *)curMainAppLibraryExperimentAssignment->universeInfo;
17    userInfoNameStringObjPart1 = *(QWORD *)(&universeInfo + 0x10);
18    userInfoNameStringObjPart2 = *(QWORD *)(&universeInfo + 0x18);
19    swift_bridgeObjectRetain(userInfoNameStringObjPart1);
20    Please enter the type declaration _int64 universeInfo
21    String.append(_)(0x7CLL, 0xE100000000000000LL); // 0x7C = '|'
22    experimentInfoNameStringObjPart1 = *(QWORD *)(&experimentInfo + 0x10);
23    experimentInfoNameStringObjPart2 = *(QWORD *)(&experimentInfo + 0x18);
24    swift_bridgeObjectRetain(userInfoNameStringObjPart1);
25    String.append(_)(0x7CLL, 0xE100000000000000LL); // 0x7C = '|'
26    bucketInfoNameStringObjPart1 = *(QWORD *)(&bucketInfo + 0x10);
27    swift_bridgeObjectRetain(userInfoNameStringObjPart1);
28    String.append(_)(0x7CLL, 0xE100000000000000LL); // 0x7C = '|'
29    bucketInfoNameStringObjPart1 = *(QWORD *)(&bucketInfo + 0x10);
30    swift_bridgeObjectRetain(userInfoNameStringObjPart1);
31    String.append(_)(0x7CLL, 0xE100000000000000LL); // 0x7C = '|'
32    bucketInfoNameStringObjPart1 = *(QWORD *)(&bucketInfo + 0x10);
33    swift_bridgeObjectRetain(userInfoNameStringObjPart1);
34    String.append(_)(0x7CLL, 0xE100000000000000LL); // 0x7C = '|'
35    swift_bridgeObjectRelease(v10);
36    swift_bridgeObjectRelease(v10);
37    return userInfoNameStringObjPart1;
38 }

```

- `_int64 universeInfo -> _TtC014MainAppLibrary90offlineAB12UniverseInfo* universeInfo`



优化后的代码：

```

_int64 StringAppend_100314974()
{
    _TtC14MainAppLibrary20ExperimentAssignment *curMainAppLibraryExperimentAssignment; // x20
    _TtCO14MainAppLibrary9OfflineAB12UniverseInfo* universeInfo; // x8
    _TtCO14MainAppLibrary9OfflineAB14ExperimentInfo* experimentInfo; // x9
    _int64 experimentInfoNameStringObjPart1; // x22
    _int64 experimentInfoNameStringObjPart2; // x23
    _int64 v5; // x21
    _TtCO14MainAppLibrary9OfflineAB10BucketInfo* bucketInfo; // x8
    _int64 bucketInfoNameStringObjPart1; // x19
    _int64 bucketInfoNameStringObjPart2; // x22
    _int64 v9; // x21
    _int64 v10; // x21
    _int64 userInfoNameStringObjPart1; // [xsp+0h] [xbp-40h]
    _int64 userInfoNameStringObjPart2; // [xsp+8h] [xbp-38h]

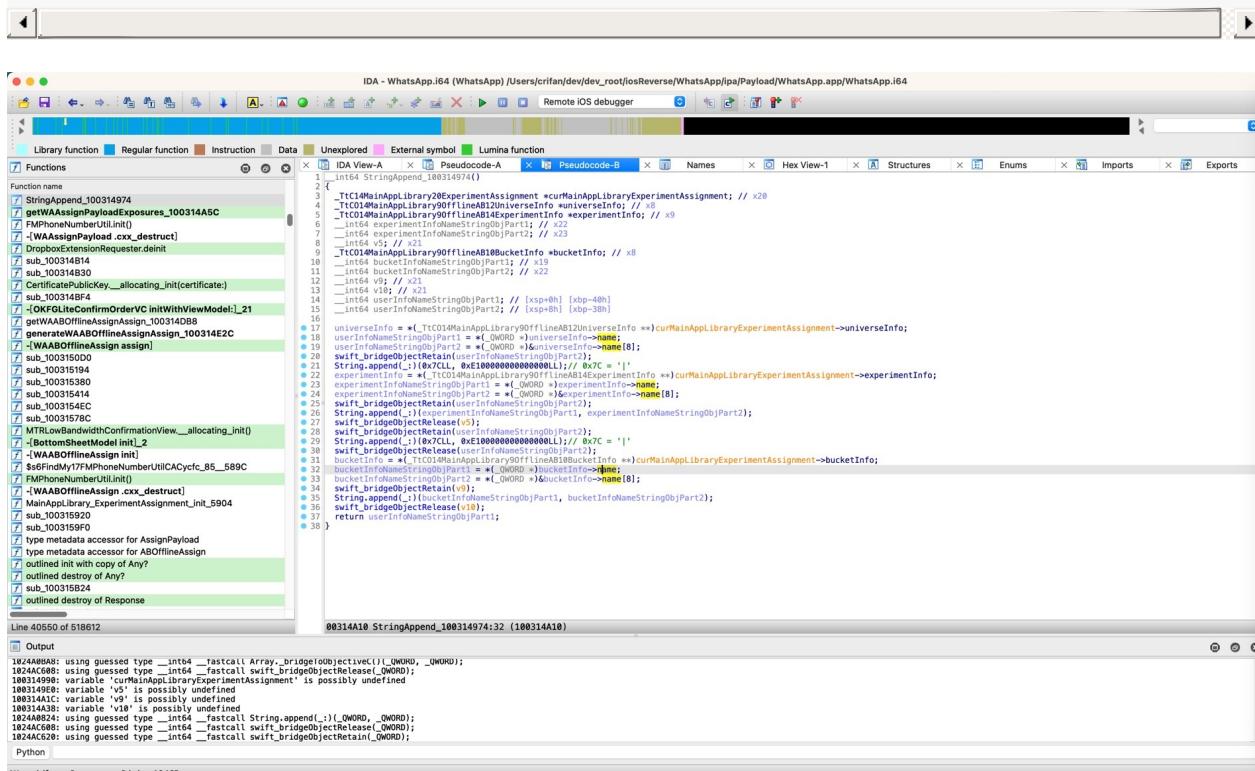
    universeInfo = (*(_TtCO14MainAppLibrary9OfflineAB12UniverseInfo**)curMainAppLibraryExperimentAssignment->universeInfo);
    userInfoNameStringObjPart1 = (*(_QWORD*)universeInfo->name);
    userInfoNameStringObjPart2 = (*(_QWORD*)universeInfo->name[8]);
    swift_bridgeObjectRetain(userInfoNameStringObjPart2);
    String.append(_)(0x7CLL, 0xE100000000000000LL); // 0x7C = '|'
    experimentInfo = (*(_TtCO14MainAppLibrary9OfflineAB14ExperimentInfo**)curMainAppLibraryExperimentAssignment->experimentInfo);
    experimentInfoNameStringObjPart1 = (*(_QWORD*)experimentInfo->name);
    experimentInfoNameStringObjPart2 = (*(_QWORD*)experimentInfo->name[8]);
    swift_bridgeObjectRetain(userInfoNameStringObjPart2);
    String.append(_)(experimentInfoNameStringObjPart1, experimentInfoNameStringObjPart2);
}

```

```

swift_bridgeObjectRelease(v5);
swift_bridgeObjectRetain(userInfoNameStringObjPart2);
String.append(_)(0x7CLL, 0xE100000000000000LL); // 0x7C = '|'
swift_bridgeObjectRelease(userInfoNameStringObjPart2);
bucketInfo = (*(_TtC014MainAppLibrary9OfflineAB10BucketInfo **)curMainAppLibraryExperimentAssignment->bucketInfo;
bucketInfoNameStringObjPart1 = (*(_QWORD *)bucketInfo->name;
bucketInfoNameStringObjPart2 = (*(_QWORD *)&bucketInfo->name[8]);
swift_bridgeObjectRetain(v9);
String.append(_)(bucketInfoNameStringObjPart1, bucketInfoNameStringObjPart2);
swift_bridgeObjectRelease(v10);
return userInfoNameStringObjPart1;
}

```



crifan.org, 使用署名4.0国际(CC BY 4.0)协议发布 all right reserved, powered by Gitbook最后更新：  
2024-02-24 15:55:56

## 逆向前后对比

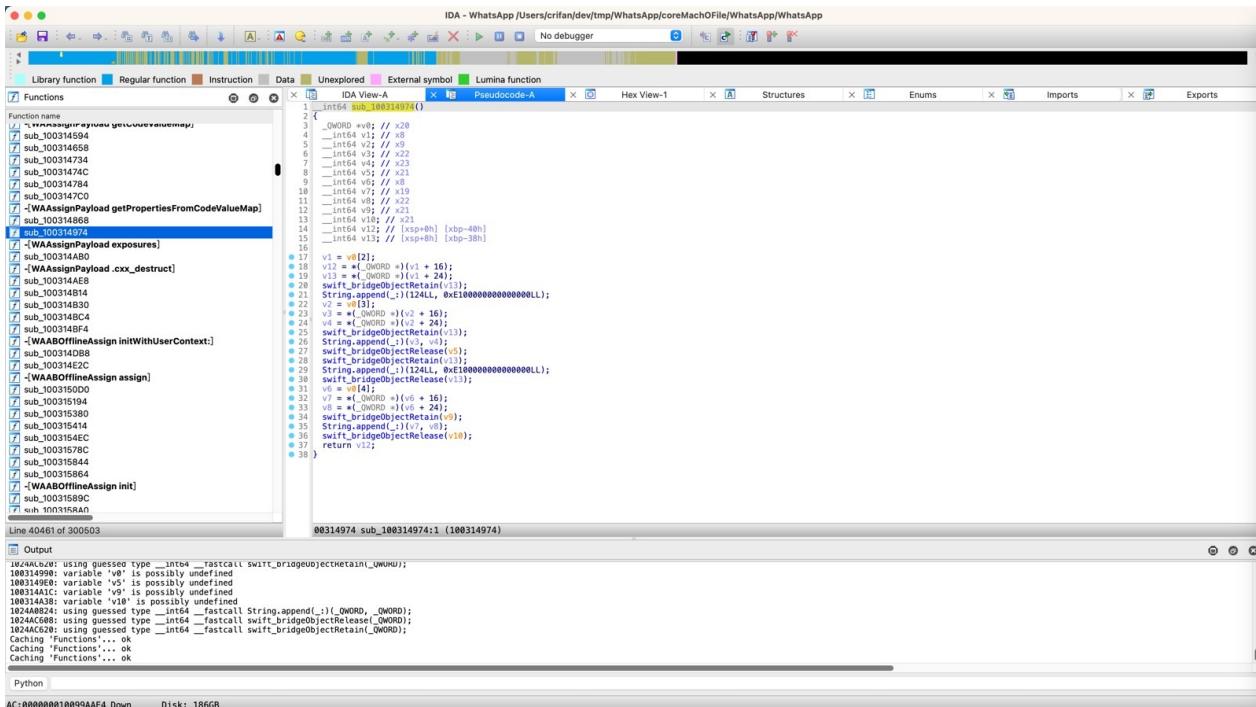
### 逆向之前

#### IDA伪代码

最初的IDA的相关伪代码，优化之前，是：

```
__int64 sub_100314974()
{
    _QWORD v0; // x20
    __int64 v1; // x8
    __int64 v2; // x9
    __int64 v3; // x22
    __int64 v4; // x23
    __int64 v5; // x21
    __int64 v6; // x8
    __int64 v7; // x19
    __int64 v8; // x22
    __int64 v9; // x21
    __int64 v10; // x21
    __int64 v12; // [xsp+0h] [xbp-40h]
    __int64 v13; // [xsp+8h] [xbp-38h]

    v1 = v0[2];
    v12 = *(__QWORD *) (v1 + 16);
    v13 = *(__QWORD *) (v1 + 24);
    swift_bridgeObjectRetain(v13);
    String.append(_)(124LL, 0xE100000000000000LL);
    v2 = v0[3];
    v3 = *(__QWORD *) (v2 + 16);
    v4 = *(__QWORD *) (v2 + 24);
    swift_bridgeObjectRetain(v13);
    String.append(_)(v3, v4);
    swift_bridgeObjectRelease(v5);
    swift_bridgeObjectRetain(v13);
    String.append(_)(124LL, 0xE100000000000000LL);
    swift_bridgeObjectRelease(v13);
    v6 = v0[4];
    v7 = *(__QWORD *) (v6 + 16);
    v8 = *(__QWORD *) (v6 + 24);
    swift_bridgeObjectRetain(v9);
    String.append(_)(v7, v8);
    swift_bridgeObjectRelease(v10);
    return v12;
}
```



## 逆向后

### IDA伪代码

逆向后，经过分析，优化之后的：IDA伪代码：

```
__int64 StringAppend_100314974()
{
    _TtC14MainAppLibrary20ExperimentAssignment curMainAppLibraryExperimentAssignment; // x20
    _TtC014MainAppLibrary90offlineAB12UniverseInfo universeInfo; // x8
    _TtC014MainAppLibrary90offlineAB14ExperimentInfo experimentInfo; // x9
    __int64 experimentInfoNameStringObjPart1; // x22
    __int64 experimentInfoNameStringObjPart2; // x23
    __int64 v5; // x21
    _TtC014MainAppLibrary90offlineAB10BucketInfo bucketInfo; // x8
    __int64 bucketInfoNameStringObjPart1; // x19
    __int64 bucketInfoNameStringObjPart2; // x22
    __int64 v9; // x21
    __int64 v10; // x21
    __int64 userInfoNameStringObjPart1; // [xsp+0h] [xbp-40h]
    __int64 userInfoNameStringObjPart2; // [xsp+8h] [xbp-38h]

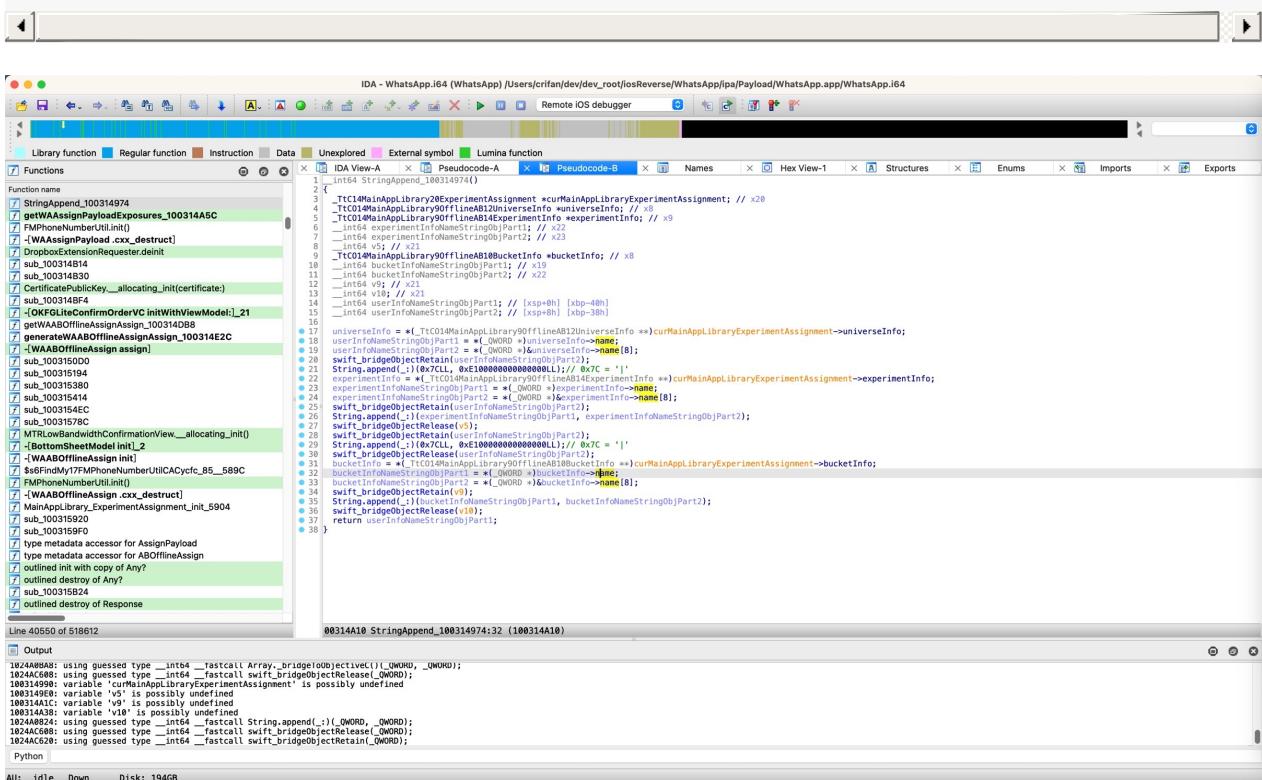
    universeInfo = (*(_TtC014MainAppLibrary90offlineAB12UniverseInfo **))curMainAppLibraryExperimentAssignment.universeInfo;
    userInfoNameStringObjPart1 = (*(_QWORD *))universeInfo->name;
    userInfoNameStringObjPart2 = (*(_QWORD *))universeInfo->name[8];
    swift_bridgeObjectRetain(userInfoNameStringObjPart2);
    String.append(_)(0x7CLL, 0xE100000000000000LL); // 0x7C = '|'
    experimentInfo = (*(_TtC014MainAppLibrary90offlineAB14ExperimentInfo **))curMainAppLibraryExperimentAssignment.experimentInfo;
    experimentInfoNameStringObjPart1 = (*(_QWORD *))experimentInfo->name;
```

```

experimentInfoNameStringObjPart2 = (*(_QWORD *) experimentInfo->name[8]);
swift_bridgeObjectRetain(userInfoNameStringObjPart2);
String.append(_*)(experimentInfoNameStringObjPart1, experimentInfoNameStringObjPart2);

swift_bridgeObjectRelease(v5);
swift_bridgeObjectRetain(userInfoNameStringObjPart2);
String.append(_*)(0x7CLL, 0xE100000000000000LL); // 0x7C = '|'
swift_bridgeObjectRelease(userInfoNameStringObjPart2);
bucketInfo = (*(_TtC014MainAppLibrary90offlineAB10BucketInfo **)curMainAppLibraryExperimentAssignment->bucketInfo;
bucketInfoNameStringObjPart1 = (*(_QWORD *)bucketInfo->name;
bucketInfoNameStringObjPart2 = (*(_QWORD *)bucketInfo->name[8]);
swift_bridgeObjectRetain(v9);
String.append(_*)(bucketInfoNameStringObjPart1, bucketInfoNameStringObjPart2);
swift_bridgeObjectRelease(v10);
return userInfoNameStringObjPart1;
}

```



## 代码逻辑

经过分析和调试，以及代码优化后，就真正的，完全的，彻底的搞懂了代码逻辑。

然后就可以分析和整理出函数的逻辑了：

- `StringAppend_100314974`
  - 输入： `MainAppLibrary.ExperimentAssignment` 的实例
  - 处理过程
    - 获取 `MainAppLibrary.ExperimentAssignment` 的 `universeInfo`
    - 再获取 `universeInfo` 的 `name`
      - 其是 Swift 的 (Native) Large String
    - 去拼接上： `"|"`

- 其是个 Swift 的 small string
  - 继续类似逻辑
    - 获取 MainAppLibrary.ExperimentAssignment 的 experimentInfo
      - 再获取 experimentInfo 的 name
        - 其是 Swift 的 Large String
      - 拼接 " | "
    - 获取 MainAppLibrary.ExperimentAssignment 的 bucketInfo
      - 再获取 bucketInfo 的 name
        - 其是 Swift 的 Small String
  - 最终得到拼接后的最终的字符串
- 处理逻辑概述：用 universeInfo、experimentInfo、bucketInfo 的 name，中间加上 |，拼接后的字符串
  - 输出=返回值：
    - 最终拼接后的字符串
    - 举例
      - "dummy\_aa\_offline\_rid\_universe\_ios|dummy\_aa\_offline\_rid\_experiment\_ios|control"

如此，实现了：

通过静态分析（IDA的汇编代码和伪代码、class-dump导出的头文件、导出的字符串等资源）和动态调试（Xcode、lldb、iOSOpenDev插件hook代码等），加上此处特定的，Swift的String的Append函数以及String的内部类型和逻辑（String分Small String和Large String），最终才彻底搞懂代码逻辑，以及去优化IDA伪代码为，最终的代码，人类能看懂的代码。

crifan.org, 使用[署名4.0国际\(CC BY 4.0\)协议](#)发布 all right reserved, powered by Gitbook最后更新：  
2024-02-24 15:50:30

## 附录

下面列出相关参考资料。

crifan.org, 使用[署名4.0国际\(CC BY 4.0\)协议](#)发布 all right reserved, powered by Gitbook最后更新：  
2024-02-24 14:28:43

## 参考资料

- 【整理】iOS逆向心得：举例说明如何静态分析和动态调试出最终完整的代码逻辑
- 【未解决】iOS逆向WhatsApp：Swift函数String.append(\_:)的字符串拼接的实现逻辑
- 【已解决】iOS逆向：Swift中String字符串的内部结构和逻辑
- 【已解决】iOS逆向WhatsApp：类MainAppLibrary.OfflineAB.UniverseInfo
- 【已解决】iOS逆向WhatsApp：类MainAppLibrary.OfflineAB.ExperimentInfo
- 【已解决】iOS逆向WhatsApp：类MainAppLibrary.OfflineAB.BucketInfo
- 
- [String字符串 · iOS逆向：Swift逆向](#)
- 
- [Swift 里字符串（十）修改字符串 – huahuahu \(wordpress.com\)](#)
- 

crifan.org, 使用[署名4.0国际\(CC BY 4.0\)协议](#)发布 all right reserved, powered by Gitbook最后更新：  
2024-02-24 15:54:10