

目录

前言	1.1
静态分析概览	1.2
从ipa中找到二进制	1.3
静态分析内容	1.4
Mach-O工具	1.4.1
导出字符串资源	1.4.2
strings	1.4.2.1
举例	1.4.2.1.1
nm	1.4.2.2
导出头文件	1.4.3
class-dump	1.4.3.1
help语法	1.4.3.1.1
处理签名	1.4.4
ldid	1.4.4.1
help语法	1.4.4.1.1
codesign	1.4.4.2
分析代码逻辑	1.4.5
IDA	1.4.5.1
Hopper	1.4.5.2
静态分析实例	1.5
查看信息和导出字符串	1.5.1
相关工具	1.6
radare2	1.6.1
Cutter	1.6.1.1
举例	1.6.1.1.1
AwemeCore	1.6.1.1.1.1
静态分析心得	1.7
二进制中找不到类	1.7.1
附录	1.8
参考资料	1.8.1

iOS逆向开发：静态分析

- 最新版本: v1.2.9
- 更新时间: 20231022

简介

介绍iOS逆向开发期间的静态分析。先是静态分析的概览；接着是从ipa中找到要分析的二进制文件；再是静态分析主要涉及的内容，包括用Mach-O工具，用strings、nm、otool等导出字符串资源，用class-dump导出ObjC头文件，用IDA、Hopper等分析代码逻辑等等；以及相关工具，比如radare2以及GUI版Cutter等；然后给出一些相关实例；最后再整理一些相关的经验心得。

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_static_analysis: iOS逆向开发：静态分析](#)

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逆向开发：静态分析 book.crifan.org](#)
- [iOS逆向开发：静态分析 crifan.github.io](#)

离线下载阅读

- [iOS逆向开发：静态分析 PDF](#)
- [iOS逆向开发：静态分析 ePUB](#)
- [iOS逆向开发：静态分析 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：

2023-10-22 21:49:54

静态分析概览

iOS逆向过程中，在砸壳出ipa后，就是： 静态分析。

什么是静态分析

iOS逆向的目的，往往要搞清楚，app底层的某些功能和逻辑的具体实现机制和原理。

而搞懂底层逻辑，从过程角度来说，主要分：

- 静态分析：不运行程序的前提下，利用工具和手段，搞懂程序逻辑
- 动态调试：运行程序的前提下，动态运行期间，研究和调试程序的逻辑

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：
2022-10-21 20:48:31

从ipa中找到二进制

TODO:

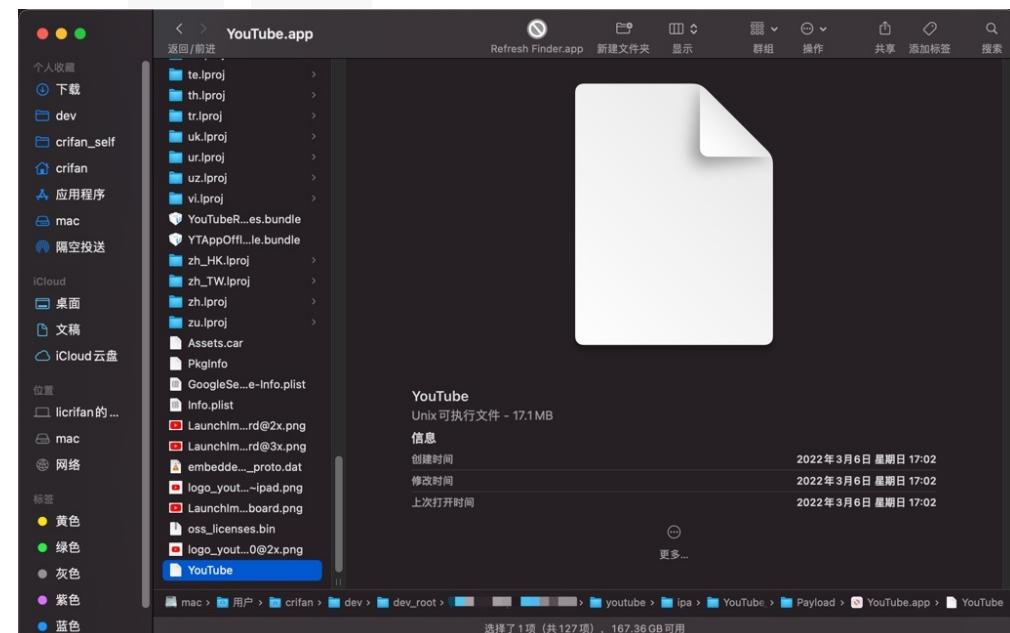
- 【未解决】静态分析抖音二进制寻找越狱检测手段
- 【已解决】如何从脱壳后的抖音的IPA文件中得到二进制文件
- 相关
 - 【已解决】越狱iPhone中抖音app的安装目录安装位置
 - 【已解决】研究iOS中app的目录的UUID类的值和app名称如何映射

作为iOS逆向的静态分析，其输入文件是iOS的app的二进制文件。

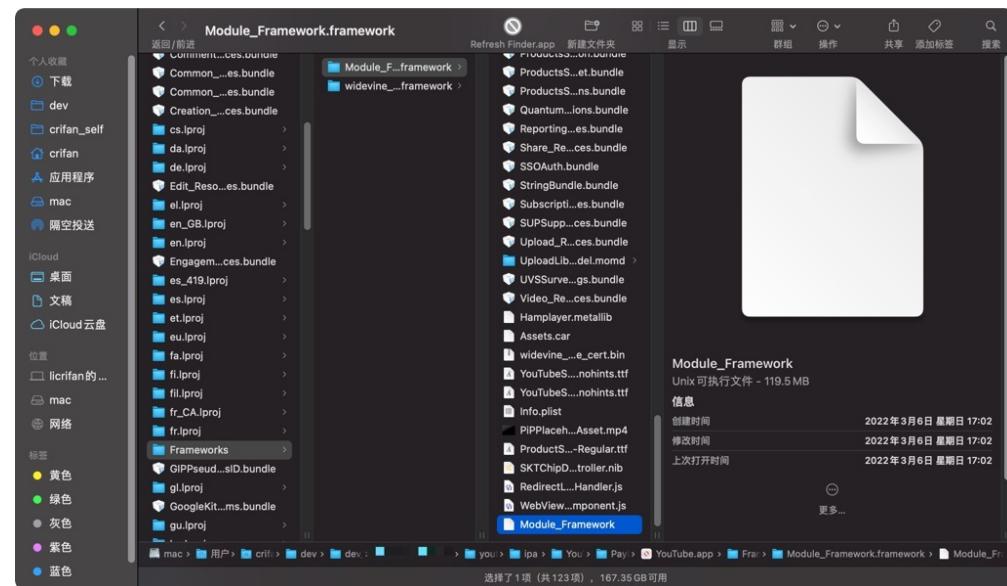
对应的就是，在前一步，[从app砸壳得到的ipa文件中](#)，找到对应的二进制文件，用于后续的静态分析。

比如：

- YouTube
 - v17.08.2
 - ipa解压后得到：`YouTube.app`
 - 入口二进制：`17MB+` 的 YouTube



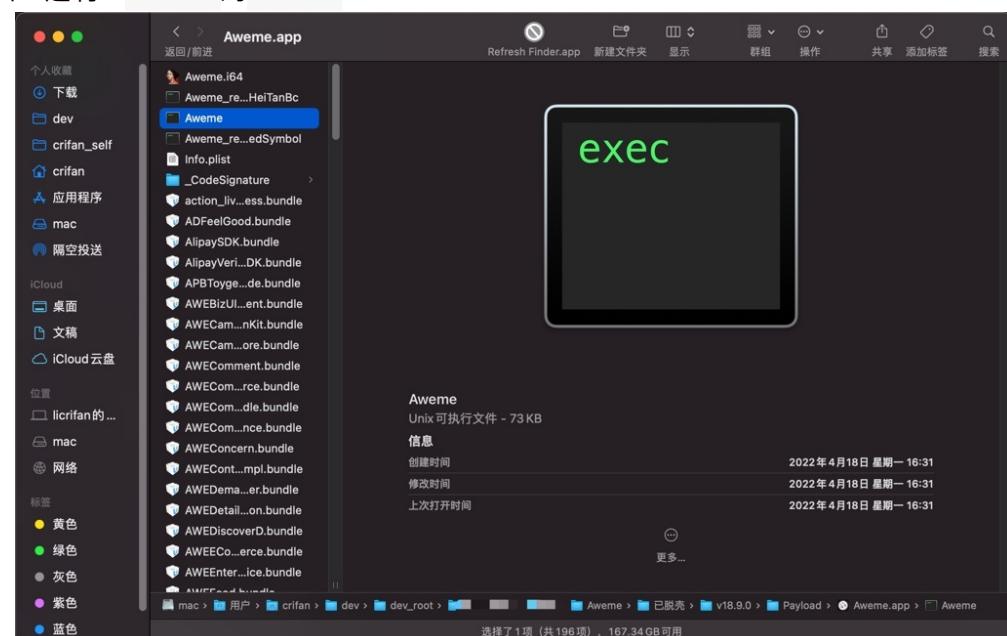
- 核心二进制：`100MB+` 的 Frameworks/Module_Framework.framework/Module_Framework



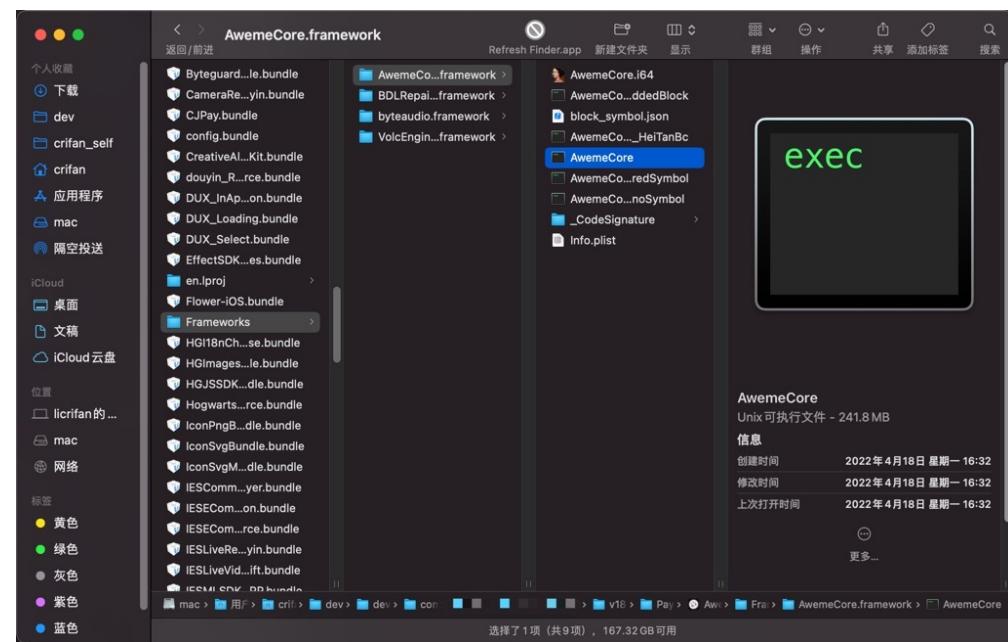
- 抖音

- v18.9.0

- ipa解压后得到: Aweme.app
 - 入口二进制: 70KB+ 的 Aweme



- 核心二进制: 240MB+ 的 Frameworks/AwemeCore.framework/AwemeCore



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:

2022-10-21 23:19:35

静态分析内容

iOS逆向的静态分析，主要针对的是：

- iOS的二进制格式： Mach-O
 - Mach-O 工具
 - 概述
 - 查看解析Mach-O
 - MachOView
 - rabin2
 - jtool2
 - otool
 - FAT瘦身
 - lipo
 - 详解
 - 独立子教程
 - 可执行文件格式：Mach-O
 - 其他处理
 - 导出字符串等资源：用于后续研究搜索函数、类等用途
 - nm
 - strings
 - 处理签名：重新签名避免运行崩溃
 - ldid
 - codesign
 - 导出头文件：研究类的具体函数和属性
 - class-dump
 - 分析代码逻辑：分析研究代码实现逻辑和其他各种细节
 - IDA
 - Hopper

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：
2023-10-04 22:37:40

Mach-O工具

- Mach-O 工具
 - 概述
 - 查看解析Mach-O
 - [MachOView](#)
 - [rabin2](#)
 - [jtool2](#)
 - [otool](#)
 - FAT瘦身
 - [lipo](#)
 - 详解
 - 独立子教程
 - [可执行文件格式：Mach-O](#)

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：

2023-10-20 09:26:54

导出字符串资源

TODO:

- 【未解决】静态分析符号函数字符串: Aweme
 - 【未解决】静态分析符号函数字符串: AwemeCore
-

iOS逆向期间，静态分析，往往涉及到：从二进制中导出字符串等信息和其他资源，供后续研究分析。

常用的导出字符串等资源的工具有：

- `strings`
- `nm`

注：之前的查看Mach-O格式信息的工具，也可以用来导致字符串等资源：

- `rabin2`
- `jtool2`
- `otool`

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：
2023-10-04 23:02:04

strings

用法

```
strings iOSBinaryFile > iOSBinaryFile_strings.txt
```

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：
2023-10-05 17:02:54

strings用法举例

MaskPro.dylib

→ DynamicLibraries strings MaskPro.dylib > MaskProDylib/MaskProDylib_strings.txt

输出：

```

MaskProDylib_strings.coffee — Mask
1 currentDevice
2 name
3 systemVersion
4 systemName
5 xWlxsPxExAux
6 identifierForVendor
7 UUIDString
8 sharedManager
9 advertisingIdentifier
10 dictionaryWithObjectsAndKeys:
11 URLWithString:
12 stringWithContentsOfURL:encoding:error:
13 hasPrefix:
14 deletedCharactersInRange:
15 length
16 substringToIndex:
17 dataUsingEncoding:
18 JSONObjectWithData:options:error:
19 objectForKeyedSubscript:
20 stringWithCString:encoding:
21 isEqualToString:
22 eKGEGRxxPxt
23 PHcNExxUJIxH
24 graGqxxPtxaoBY
25 xHvTCxxxxXVm
26 bundleWithPath:
27 load
28 numberWithInt:
29 setAppWirelessDataOption:forBundleIdentifier:completionHandler:
30 numberWithInt:
31 setAppCellularDataEnabled:forBundleIdentifier:completionHandler:
32 sharedInstance
33 setUsagePoliciesForBundle:cellular:wifi:
34 resolveNetworkProblemeForAppWithBundleId:
35 date
36 alloc
37 init
38 timeZoneWithName:
39 setTimeZone:
40 setDateformat:
41 stringFromDate:
42 defaultManager
43 attributesOfFilesystemForPath:error:

```

其中的：

```

eKGEGRxxPxt
PHcNExxUJIxH
graGqxxPtxaoBY
xHvTCxxxxXVm

```

很明显是：字符串加密的一种，手动替换或者hash成固定长度的字符串了。

zzzzHeiBaoLib.dylib

→ DynamicLibraries strings zzzzHeiBaoLib.dylib > zzzzHeiBaoLib_strings.txt

输出： zzzzHeiBaoLib_strings.txt

改名和提取核心字段后为： zzzzHeiBaoLib_strings_elite.txt

zzzzHeiBaoLib_strings_elite.txt — iPhone7

```

DynamicLibraries > zzzzHeiBaoLib_strings_elite.txt
1 system-file
2 system-file-time
3 system-file-time-nano
4 system-file-node
5 name
6 iPhone
7 boottime
8 boottime-nsec
9 fake.os.release
10 fake.os.kernver
11 fake.os.firmer
12 fake.device.model
13 fake.device.intername
14 fake.device.cpucore
15 fake.device.memory
16 fake.device.cache1
17 fake.device.cache2
18 ext.%@%
19 %@%@%
20 idfa
21 idfv
22 openudid
23 mach_msg
24 svc_call
25 com.bd.iphone.super
26 /private
27 dispatch
28 /private/var
29 /var
30 /private/etc/group
31 /etc/group
32 /private/etc/hosts
33 /etc/hosts
34 /System/Library/Caches/com.apple.dyld/dyld_shared_cache_arm64
35 ===== header : %p, executable_section : %p, size : %lu, %s, %
36 ===== start_address : %p, end_address : %p
37 ===== address : %p
38 %@2x
39 OS_object
40 Command <<<
41 Super Class
42 zh_CN
43 vvvv-MM-nd

```

行 1, 列 1 空格: 4 UTF-8 LF CoffeeScript Prettier

zzzzHeiBaoLib_strings.txt — iPhone7

```

DynamicLibraries > zzzzHeiBaoLib_strings_elite.txt
157 com.apple.unicode-data
158 com.apple.metadata-uid
159 com.apple.geod/.com.apple.mobile_container_manager.metadata.plist
160 geod-uid
161 CoreServices/SystemVersion.plist
162 com.apple.MobileGestalt.plist
163 apps
164 HeiBao
165 SystemVersion.plist
166 /Applications
167 /usr/share
168 /usr/libexec
169 /usr/include
170 /Library/Ringtones
171 /Library/Wallpaper
172 com.ss.iphone.ugc.Ame
173 com.zhiliaoapp.musically
174 f896afca-38b8-4648-a1c6-63ee5854b67c
175 com.jifen.kuan
176 CSToastPositionTop
177 CSToastPositionCenter
178 CSToastPositionBottom
179 CSToastTimerKey
180 CSToastDurationKey
181 CSToastPositionKey
182 CSToastCompletionKey
183 CSToastActiveToastViewKey
184 CSToastActivityViewKey
185 CSToastQueueKey
186 -----_CTServerConnectionCopyMobileIdentity-----%
187 com.tencent.xin
188 ---MicroMessengerAppDelegate application didFinishLaunchingWithOptions
189 ---
190 ----login with 62data : @@
191 /Library/WechatPrivate
192 /Library/WechatPrivate/wx.dat
193 xxxxxxxx----login with 62data : @@
194 apns
195 network.type
196 DeviceName
197 rkqlwPcRhwiX4gapPjanw
198 UserAssignedDeviceName
199 ghpAuGJlPoauWijdtPi7sQ

```

行 175, 列 5 (已选择4) 空格: 2 UTF-8 LF CoffeeScript Prettier

摘录部分内容：

- 疑似app
 - com.bd.iphone.super
 - com.apple.springboard
 - com.apple.geod/.com.apple.mobile_container_manager.metadata.plist
 - com.apple.MobileGestalt.plist
 - com.zhiliaoapp.musically

- com.jifen.qukan
- 趣头条
- com.tencent.xin
- com.kuaishou.nebula
- 快手极速版
- com.ss.iphone.ugc.Live
- com.xunmeng.pinduoduo
- com.soulapp.cn
- 抖音
- iPhone版
 - com.ss.iphone.ugc.Ame
 - com.ss.iphone.ugc.Aweme
 - com.ss.iphone.ugc.aweme.lite
- 注：安卓版是
 - com.ss.android.ugc.aweme
 - com.ss.android.ugc.aweme.lite
- com.dragon.read
- com.wemomo.momoappdemo1
- com.jiangjia.gif
- 系统属性
 - carrier.name
 - carrier.mcc
 - carrier.mnc
 - carrier.icc
 - network.addr
 - network.dstaddr
 - location.latitude
 - location.longitude
 - identifierForVendor
 - systemVersion
 - operatingSystemVersionString
 - operatingSystemVersion
 - carrierName
 - mobileCountryCode
 - mobileNetworkCode
 - isoCountryCode
 - allowsVOIP
- 疑似越狱检测路径相关
 - /
 - /Applications
 - /usr
 - /usr/share
 - /usr/libexec
 - /usr/include
 - /etc/hosts
 - /private

- /private/var
 - /private/var/containers/Bundle/Application/
 - /private/etc/group
- /Library
 - /Library/Ringtones
 - /Library/Wallpaper
- /System
 - /System/Library
 - /System/Library/Caches/com.apple.dyld/dyld_shared_cache_arm64
 - /System/Library/CoreServices/SystemVersion.plist
 - /System/Library/PreferenceBundles/VPNPreferences.bundle
 - /var
 - /var/containers/Shared/SystemGroup/systemgroup.com.apple.mobilegestaltcache/Library/Caches/com.apple.MobileGestalt.plist
- 黑豹本身有关
 - /var/mobile/Library/Preferences/com.heibao.fake_params.plist
 - /var/mobile/Library/Preferences/com.heibao.fake_prefs.plist
 - /var/mobile/Library/Preferences/com.heibao.vpn_config.plist
 - /var/mobile/Library/Preferences/com.heibao.scheme.plist
- 微信相关
 - /Library/WechatPrivate
 - /Library/WechatPrivate/wx.dat
- 反越狱检测相关
 - isJailbroken
 - btd_isJailBroken
 - isJailBreak
 - isJailBroken
 - HasInstallJailbreakPluginInvalidIAPPurchase
 - HasInstallJailbreakPlugin:
 - IsJailBreak
 - getJailbreakPath
 - getJailbreakRootDir
 - JailBroken
 - isDeviceJailBreak
 - isDeviceJailBroken
 - cannotPurchaseDueToJailbreakPlugin:

zorro.dylib

```
→ DynamicLibraries strings zorro.dylib > zorro_strings.txt
```

输出：

```

zorro_strings_elite.txt — iPhone7
DynamicLibraries > zorro_strings_elite.txt
zorro_strings_elite.txt
1 ![_Ew
2 +-----+
3 w`uN
4 M3@NQS
5 qKNJDuTAKLyCaH
6 UIApplicationDelegate
7 NSObject
8 CLLocationManagerDelegate
9 yjIMyUEGcrIuzoEk
10 zugZBMYNcOogFPwQ
11 __ARC Lite__
12 __ARC LiteIndexedSubscripting__
13 __ARC LiteKeyedSubscripting__
14 UAsqhyUshNiEHZN
15 NkMuJfExMIWnGKde
16 dbTOFdCBFDndTfk
17 UioJiZMecKPNRNU
18 init
19 MBcqt5Forhbv0qf:FfidxFkYSAxIYZC:
20 gDKTVxWEAItPasMv:FfidxFkYSAxIYZC:
21 CQuPqXs0dSSaTrAR:
22 dataUsingEncoding:
23 bytes
24 length
25 lowercaseString
26 characterAtIndex:
27 appendBytes:length:
28 isEqualToString:
29 numberWithBool:
30 arrayWithObjects:
31 initWithContentsOfFile:
32 objectForKey:
33 UTF8String
34 length
35 stringWithUTF8String:
36 getCString:maxLength:encoding:
37 bytes
38 dataWithBytesNoCopy:length:
39 dataWithContentsOfFile:options:error:
40 unarchiveObjectWithData:
41 mainBundle
42 bundleIdentifier
43 containsObject:

```

行 2, 列 11 空格: 4 UTF-8 LF CoffeeScript Prettier

结果分析：

- 反越狱检测的相关路径
 - /var/containers
 - /System/Library/Frameworks/
 - /var/containers/Bundle/Application/
 - /var/stash/
 - /private/var/db/stash/
 - /private/
 - /no/jailbreak
 - /System/Library/CoreServices/SystemVersion.plist

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:

2023-10-05 17:03:33

nm

用法

```
nm iOSBinaryFile > iOSBinaryFile_nm.txt
```

举例

MaskPro.dylib

```
→ DynamicLibraries nm MaskPro.dylib > MaskProDylib/MaskProDylib_nm.txt
```

输出：

```

MaskProDylib_nm.coffee — Mask
otool -l MaskPro.dylib > MaskProDylib/MaskProDylib_nm.txt
Library > MobileSubstrate > DynamicLibraries > MaskProDylib > MaskProDylib_nm.coffee

179 0003f518 S _y.192
180 0003f51c S _y.194
181 0003f520 S _y.196
182 0003f524 S _y.198
183 0003f534 S _y.20
184 0003f528 S _y.200
185 0003f538 S _y.22
186 0003f550 S _y.26
187 0003f554 S _y.28
188 0003f45c S _y.29
189 0003f558 S _y.30
190 0003f460 S _y.31
191 0003f55c S _y.32
192 0003f464 S _y.33
193 0003f560 S _y.34
194 0003f468 S _y.35
195 0003f438 S _y.374
196 0003f43c S _y.376
197 0003f440 S _y.378
198 0003f444 S _y.380
199 0003f448 S _y.382
200 U dyld_stub_binder
201
202 MaskPro.dylib (for architecture arm64):
203 U __CC_MD5
204 U __MGCopyAnswer
205 U __MSHookFunction
206 U __MSHookMessageEx
207 U __NSClassFromString
208 U __NSFileSystemFreeSize
209 U __NSHomeDirectory
210 U __OBJC_CLASS_$_ASIIdentifierManager
211 U __OBJC_CLASS_$_NSBundle
212 U __OBJC_CLASS_$_NSData
213 U __OBJC_CLASS_$_NSDate
214 U __OBJC_CLASS_$_NSDateFormatter
215 U __OBJC_CLASS_$_NSDictionary
216 U __OBJC_CLASS_$_NSFileManager
217 U __OBJC_CLASS_$_NSJSONSerialization
218 U __OBJC_CLASS_$_NSMutableData
219 U __OBJC_CLASS_$_NSMutableDictionary
220 U __OBJC_CLASS_$_NSMutableString
221 U __OBJC_CLASS_$_NSMutableURLRequest

```

MaskPro.dylib (for architecture armv7):

```

U __CC_MD5
U __MGCopyAnswer
U __MSHookFunction
U __MSHookMessageEx
U __NSClassFromString
U __NSFileSystemFreeSize
U __NSHomeDirectory
U __OBJC_CLASS_$_ASIIdentifierManager
U __OBJC_CLASS_$_NSBundle

```

```
0003f440 S _y.378
0003f444 S _y.380
0003f448 S _y.382
    U dyld_stub_binder

MaskPro.dylib (for architecture arm64):
    U _CC_MD5
    U _MGCopyAnswer
    U _MSHookFunction
    U _MSHookMessageEx
    U _NSClassFromString
    U _NSFileSystemFreeSize
    U _NSHomeDirectory
    U _OBJC_CLASS_$_ASIdentifierManager
...
```

导出头文件

iOS逆向的静态分析中，往往涉及到：从二进制文件中，导出ObjC的类的头文件，简称：导出头文件。

- 导出头文件的前提
 - iOS的app的二进制是已经**砸壳**后的 = 解密后的
 - 待确认：iOS的app的代码是ObjC的

一些经验和心得

- 关于其中：待确认：iOS的app的代码是ObjC的
 - ObjC由于底层机制（Runtime运行时的原因？），使得导出ObjC的类的头文件，成为可能
 - 而iOS的新的编程语言Swift，从底层机制上说，就无法导出Swift的类的头文件
 - 从而使得=导致：
 - 对于iOS的app的实现是ObjC和Swift混合都有的话，之前旧版本class-dump在导出时会报错，导致无法导出头文件
 - 需要找到新版，支持ObjC和Swift混合代码，才能继续导出（ObjC的类的）头文件
 - 从iOS正向防护角度，增加逆向导出头文件的难度：尽量把iOS的代码从ObjC改为Swift
 - 待确认：即可避免头文件被导出

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：
2022-11-08 11:36:19

class-dump

TODO:

- 【基本解决】砸壳抖音ipa后导出iOS抖音头文件

- class-dump
 - 一句话描述：iOS逆向中导出ObjC的头文件的常用工具
 - 用于处理 Objective-C 的 Mach-O 文件信息的命令行工具，可以导出类的定义、分组和协议。
 - command-line utility for examining the Objective-C segment of Mach-O files
 - 说明
 - 和 otool -ov 导出的信息是一样的
 - 但是显示为 Objective-C 定义，更易读
 - 原理
 - 利用了 Objective-C 语言的运行时的特性
 - 将存储在 Mach-O 文件中的头文件信息提取出来，并生成对应的 .h 文件
 - 用途
 - 查看闭源的应用、frameworks、bundles
 - 查看其中的头文件信息
 - 对比一个 APP 不同版本之间的接口变化
 - 通过导出不同版本的库的头文件的对比看出来
 - 对一些私有 frameworks 做些有趣的试验

版本和下载

- class-dump 版本
 - 概述
 - 推荐：支持Swift和ObjC混淆的版本
 - MonkeyDev 中的 class-dump
 - <https://github.com/AloneMonkey/MonkeyDev/blob/master/bin/class-dump>
 - 详解
 - 官网的版本
 - 安装包
 - [class-dump-3.5.dmg](#)
 - 相关资料
 - GitHub
 - nygard/class-dump: Generate Objective-C headers from Mach-O files.
 - <https://github.com/nygard/class-dump>
 - 官网
 - class-dump - Steve Nygard
 - <http://stevenyngard.com/projects/class-dump/>
 - 源码

- class-dump-3.5.tar.gz
 - 或: class-dump-3.5.tar.bz2
- 升级版=优化版: 支持swift和ObjC混淆
 - 有多个源
 - 比如
 - MonkeyDev 中的 class-dump
 - <https://github.com/AloneMonkey/MonkeyDev/blob/master/bin/class-dump>
 - 其他: 待研究
 - <https://github.com/0xed/class-dump>
 - swift分支=swift版本
 - <https://github.com/0xed/class-dump/tree/swift-binaries>

用法

典型用法:

```
./class-dump --arch <arch> -H -o <outputFolder> <inputBinaryFile>
```

- 参数说明
 - --arch : 指定架构
 - 最常见的值: arm64
 - 默认可以省略
 - 但如果是 FAT = 胖二进制 时, 则必须指定对应架构
 - -H : 输出头文件
 - -o : 输出目录

举例:

- AwemeCore


```
MonkeyDev/bin/class-dump --arch arm64 -H AwemeCore -o /Users/crifan/dev/DevRoot/iOS
Reverse/Aweme/class_dump_output
```
- Aweme


```
AloneMonkey/MonkeyDev/bin/class-dump --arch arm64 -H Aweme -o /Users/crifan/dev/Dev
Root/Aweme/classDumpResult/17.8.0/Aweme
```
- YouTube


```
class-dump --arch arm64 -H ./ipa/YouTube_17.08.2_dumped/Payload/YouTube.app/Youtub
e -o _
```
- MusicallyCore


```
./class-dump --arch arm64 -H ipa/Payload/TikTok.app/Frameworks/MusicallyCore.framework/MusicallyCore -o tiktok_headers_26.8.0
```

- 其他
 - class-dump AppKit
 - class-dump /System/Library/Frameworks/AppKit.framework
 - class-dump UIKit
 - class-dump
 /Developer/Platforms/iPhoneOS.platform/Developer/SDKs/iPhoneOS4.3.sdk/System/Library/Frameworks/UIKit.framework
 - class-dump UIKit and all the frameworks it uses
 - class-dump
 /Developer/Platforms/iPhoneOS.platform/Developer/SDKs/iPhoneOS4.3.sdk/System/Library/Frameworks/UIKit.framework -r --sdk-ios 4.3
 - class-dump UIKit (and all the frameworks it uses) from developer tools that have been installed in /Dev42 instead of /Developer
 - class-dump
 /Dev42/Platforms/iPhoneOS.platform/Developer/SDKs/iPhoneOS5.0.sdk/System/Library/Frameworks/UIKit.framework -r --sdk-root
 /Dev42/Platforms/iPhoneOS.platform/Developer/SDKs/iPhoneOS5.0.sdk

常见问题

TODO:

- 【记录】支持iOS的Swift和ObjC混编的class-dump
- 【已解决】class-dump导出Framework二进制AwemeCore报错： Cannot find offset for address in dataOffsetForAddress
- 【未解决】Mac中无法删除临时目录出现没有权限Operation not permitted
- 【已解决】砸壳后抖音ipa安装失败：DeviceNotSupportedByThinning

class-dump导出头文件是空的

- 现象：抖音的二进制 AwemeCore，用 class-dump 导出头文件是空的
- 原因：抖音内部代码应该是 Swift 和 ObjC 混编代码，原版 class-dump 只支持 ObjC 的，不支持 Swift 和 ObjC 混编，所以导出是空的。
- 解决办法：找支持Swift的版本的class-dump去导出，即可
 - 比如
 - MonkeyDev 中的 class-dump
 - <https://github.com/AloneMonkey/MonkeyDev/blob/master/bin/class-dump>
 - 相关命令


```
MonkeyDev/bin/class-dump --arch arm64 -H AwemeCore -o crifan/Aweme/class_dump_output
```

使用心得

从 Generated by class-dump 可以看出原始用到了 class-dump

之前从[WebDriverAgent](#)的源码中看到很多头文件的头部都有: Generated by class-dump

举例:

refer/[WebDriverAgent/PrivateHeaders/XCTest/XCTestDriver.h](#)

```
//  
//      Generated by class-dump 3.5 (64 bit).  
//  
//      class-dump is Copyright (C) 1997-1998, 2000-2001, 2004-2013 by Steve Nygard.  
//
```

-》说明这些文件都是通过 class-dump 从库文件中导出生成的。

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:
2023-10-21 17:29:19

class-dump的help语法

```
class-dump 3.5 (64 bit)
Usage: class-dump [options] <mach-o-file>

where options are:
-a                  show instance variable offsets
-A                  show implementation addresses
--arch <arch>     choose a specific architecture from a universal binary (ppc, ppc64,
i386, x86_64)
-C <regex>        only display classes matching regular expression
-f <str>           find string in method name
-H                  generate header files in current directory, or directory specified with -o
-I                  sort classes, categories, and protocols by inheritance (overrides -s)

-o <dir>          output directory used for -H
-r                  recursively expand frameworks and fixed VM shared libraries
-s                  sort classes and categories by name
-S                  sort methods by name
-t                  suppress header in output, for testing
--list-arches      list the arches in the file, then exit
--sdk-ios          specify iOS SDK version (will look in /Developer/Platforms/iPhoneOS.
platform/Developer/SDKs/iPhoneOS version .sdk
--sdk-mac          specify Mac OS X version (will look in /Developer/SDKs/MacOSX version
>.sdk
--sdk-root         specify the full SDK root path (or use --sdk-ios/--sdk-mac for a short cut)
```

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：
2023-10-21 17:29:23

处理签名

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：
2023-10-04 23:03:27

ldid

- ldid
 - 是什么：用于修改二进制的entitlement权限的工具
 - 作者：saurik
 - 其他作用
 - 对于二进制签名，生成SHA1和SHA256的hash值，以便于iPhone能够正常执行二进制
 - Cydia中叫做：Link Identity Editor
 - ldid用法
 - 查看/导出二进制的entitlement权限

```
ldid -e binary
```

- 设置二进制的entitlement权限

```
ldid -S ent.xml binary
```

- 注：-S 和文件名 ent.xml 中间没有空格！
- 举例
 - 使用对应的entitlement文件给 debugserver 重新签名

```
ldid -S/Users/your/Desktop/Entitlements.xml ./debugserver
```

- 伪签名二进制，无entitlement权限

```
ldid -S binary
```

- 源码
 - 官网
 - [git.saurik.com Git - ldid.git/summary](https://git.saurik.com/ldid.git/summary)
 - GitHub 的copy
 - [xerub/ldid: Unofficial fork from saurik git repository git://git.saurik.com/ldid.git \(github.com\)](https://github.com/xerub/ldid)
- 其他资料
 - [ldid - iPhone Development Wiki](#)

不同版本

- brew (安装的Cellar) 版本：
 - 位置：/usr/local/bin/ldid
 - 其是软链接
 - 实际位置是：/usr/local/Cellar/ldid/2.1.5/bin/ldid
 - 大小：814K
 - 文件格式：Mach-O 64-bit executable x86_64
- iOSOpenDev版本

- 位置: /opt/iOSOpenDev/bin/ldid
- 大小: 383K
- 文件格式:
 - (for architecture i386): Mach-O executable i386
 - (for architecture x86_64): Mach-O 64-bit executable x86_64

-> 结论:

- 优先用**brew**版本的 ldid
 - 尽量不要用iOSOpenDev的ldid

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:
2023-10-22 20:53:33

ldid的help语法

ldid的help语法

Mac中的（用brew安装的）ldid

```
"/usr/local/bin/ldid --version
usage: /usr/local/bin/ldid -S[entitlements.xml] <binary>
/usr/local/bin/ldid -e MobileSafari
/usr/local/bin/ldid -S cat
/usr/local/bin/ldid -Stfp.xml gdb
```

iPhone11中的：ldid

(XinaA15越狱后的，TrollStore安装出的？) iPhone11中的：ldid

```
iPhone11-151:~ root# which ldid
/var/bin/ldid
iPhone11-151:~ root# ldid --version
Usage: ldid [-Acputype:subtype] [-a] [-C[adhoc | enforcement | expires | hard |
    host | kill | library-validation | restrict | runtime]] [-D] [-d]
    [-Enum:file] [-e] [-H[sha1 sha256]] [-h] [-Iname]
    [-Kkey.p12 [-Upassword]] [-M] [-P[num]] [-Qrequirements.xml] [-q]
    [-r | -Sfile.xml | -s] [-u] [-arch arch_type] file ...
Common Options:
    -S[file.xml] Pseudo-sign using the entitlements in file.xml
    -Kkey.p12 Sign using private key in key.p12
    -Upassword Use password to unlock key.p12
    -M Merge entitlements with any existing
    -h Print CDHash of file

More information: 'man ldid'
```

ldid的man page

LDID(1)	FreeBSD General Commands Manual	LDID(1)
---------	---------------------------------	---------

NAME
ldid - Link Identity Editor

SYNOPSIS
`ldid [-aDdehMPqu] [-Acputype:subtype]
 [-C[adhoc | enforcement | expires |
 hard | host | kill | library-validation | restrict | runtime]]
 [-Kkey.p12] [-r | -Sfile | -s] [-Ttimestamp] file ...`

DESCRIPTION
`ldid` adds SHA1 and SHA256 hashes to a Mach-O `file` so that they can be run

on a system that has validation but not signature verification.

-a Print the CPU types and subtypes in hexadecimal.

-Acpctype:subtype

When used with -a, -D, -e, -h, -q, or -u, only act on the slice specified by cputype and subtype. cputype and subtype should both be integers.

-C[adhoc | enforcement | expires | hard | host | kill | library-validation | restrict | runtime]

Specify the option flags to embed in the code signature. See codesign(1) for details about these options.

-D Reset the cryptid.

-d Print the cryptid in the binaries if it exists. For compatibility reasons it also acts as -h, but this will be removed in the future.

-e Print the entitlements in each slice, or the slice specified by -A, to stdout.

-h Print information about the signature, such as hash types, flags, CDHash, and CodeDirectory version to stdout.

-Kkey.p12

Sign using the identity in key.p12. This will give the binary a valid signature so that it can be run on a system with signature validation. key.p12 must not have a password.

-M When used with -S, merge the new and existing entitlements instead of replacing the existing entitlements, this is useful for adding a few specific entitlements to a handful of binaries.

-P Mark the Mach-O as a platform binary.

-Qfile Embed the requirements found in file.

-q Print embedded requirements of the binaries.

-r Remove the signature from the Mach-O.

-S[file]

Pseudo sign the Mach-O binaries. If file is specified then the entitlements found in file will be embedded in the Mach-O.

-s Resign the Mach-O binaries while keeping the existing entitlements.

-Ttimestamp

When signing a dylib, set the timestamp to timestamp. timestamp should be an UNIX timestamp in seconds, if timestamp is a single dash ('-'), the timestamp will be set to a hash of the Mach-O header.

`-U` If the binary was linked against UIKit, then print the UIKit version that the Mach-O binaries was linked against.

EXAMPLES

The command:

```
ldid -S file
```

will fakesign `file` with no entitlements.

The command:

```
ldid -Cadhoc -K/path/to/key.p12 -Sent.xml file
```

will sign `file` using the key in `/path/to/key.p12` with the entitlements found in `ent.xml`, and mark it as an adhoc signature.

The command:

```
ldid -Sent.xml -M file
```

will add the entitlements in `ent.xml` to the entitlements already in `file`.

The command:

```
ldid -e file > ent.xml
```

will save the entitlements found in each slice of `file` to `ent.xml`.

SEE ALSO

[codesign\(1\)](#)

HISTORY

The `ldid` utility was written by Jay "Saurik" Freeman. iPhoneOS 1.2.0 and 2.0 support was added on April 6, 2008. `-S` was added on June 13, 2008. SHA256 support was added on August 25, 2016, fixing iOS 11 support. iOS 14 support was added on July 31, 2020 by Kabir Oberai. iOS 15 support was added on June 11, 2021.

codesign

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：
2023-10-04 23:03:40

分析代码逻辑

iOS逆向的静态分析期间，最大的最有用的，还是：分析代码逻辑

iOS逆向方面，常用的分析代码逻辑的工具有：

- IDA
- Hopper

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：

2022-10-21 21:14:50

IDA

TODO:

- 【记录】IDA分析脱壳后抖音ipa的二进制AwemeCore
- 【已解决】iOS底层函数：objc_enumerationMutation
- 【未解决】研究抖音越狱检测逻辑：____lldb_unnamed_symbol1841884即sub_11326A84

后已整理出独立教程，详见：

[逆向利器：IDA \(crifan.org\)](#)

iOS逆向之IDA使用心得

objc_enumerationMutation表示循环的逻辑

iOS逆向期间，IDA的伪代码中，有时候会看到：objc_enumerationMutation

其含义是：只是表示这段代码是循环遍历而已

代码举例：

```
v4 = (void *)objc_retain(v2->_allTrackRenderers);
v5 = v4;
v6 = objc_msgSend(v4, "countByEnumeratingWithState:objects:count:", &v24, &v28, 16LL);

if ( v6 )
{
    v7 = v6;
    v8 = *(__QWORD *)v25;
    do
    {
        v9 = 0LL;
        do
        {
            if ( *(__QWORD *)v25 != v8 )
                objc_enumerationMutation(v5);
            objc_msgSend( (void **)(((__QWORD *)v24 + 1) + 8 * v9), "terminate");
        }
        while ( v9 < (unsigned __int64)v7 );
        v7 = objc_msgSend(v5, "countByEnumeratingWithState:objects:count:", &v24, &v28, 1
6LL);
    }
    while ( v7 );
}
```

和：

```
do
{
    v34 = 0LL;
    ...
    v81 = v31;
    do
    {
        if (*(_QWORD *)v98 != v32)
            objc_enumerationMutation(v28);
        v86 = v34;
        v35 = (*_QWORD *)((((_QWORD *)v97 + 1) + 8 * v34));
        v85 = jmp_objc_msgSend_x19tox2(v26[83], objectForKeyedSubscript__);
        ...
    }
    while ( v86 + 1 < (unsigned __int64)v31 );
    v31 = objc_msgSend(v28, v73, &v97, &v109, 16LL);
}
while ( v31 );
```

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:
2023-10-06 22:30:08

Hopper

- Hopper = Hopper Disassembler
 - 概述
 - Hopper是常用的iOS逆向工具之一，主要用来静态分析iOS二进制的代码逻辑
 - 详解
 - [iOS逆向工具：Hopper](#)

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：
2023-10-08 10:45:49

静态分析实例

此处给出具体的实际的例子，来说，具体如何去静态分析，以及输出结果如何。

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：
2022-10-21 21:42:43

查看信息和导出字符串

TODO:

- 【记录】静态分析iOS的17.8.0旧版抖音
- 【记录】静态分析黑豹二进制HeiBao

此处对于静态分析中的，查看二进制信息和导出字符串等资源，给出实例，供参考。

通用成套做法

- 典型的成套的做法 = otool + nm + strings + jtool2 + rabin2 + ldid / codesign :

```
otool -l iOSBinaryFile > iOSBinaryFile_otool_l.txt
otool -ov iOSBinaryFile > iOSBinaryFile_otool_ov.txt

nm iOSBinaryFile > iOSBinaryFile_nm.txt

strings iOSBinaryFile > iOSBinaryFile_strings.txt

jtool2 -h iOSBinaryFile > iOSBinaryFile_jtool2_h_header.txt
jtool2 -l iOSBinaryFile > iOSBinaryFile_jtool2_l_list.txt
jtool2 -L iOSBinaryFile > iOSBinaryFile_jtool2_L_library.txt
jtool2 -S iOSBinaryFile > iOSBinaryFile_jtool2_S_symbol.txt

jtool2 --analyze iOSBinaryFile
mv iOSBinaryFile.ARM64.xxx-xxx-xxx-xxx-xxx iOSBinaryFile_jtool2_analyze.txt

rabin2 -I iOSBinaryFile > iOSBinaryFile_rabin2_I_identification.txt
rabin2 -i iOSBinaryFile > iOSBinaryFile_rabin2_i_imports.txt
rabin2 -E iOSBinaryFile > iOSBinaryFile_rabin2_E_exports.txt
rabin2 -l iOSBinaryFile > iOSBinaryFile_rabin2_l_libraries.txt
rabin2 -z iOSBinaryFile > iOSBinaryFile_rabin2_z_strings.txt
rabin2 -s iOSBinaryFile > iOSBinaryFile_rabin2_s_symbols.txt
rabin2 -S iOSBinaryFile > iOSBinaryFile_rabin2_S_sections.txt

# export entitlements
codesign -d --entitlements - iOSBinaryFile > iOSBinaryFile_codesign_entitlement.xml
# or
# ldid -e iOSBinaryFile > iOSBinaryFile_ldld_entitlement.xml

# export code sign info
#   for binary
codesign -vv -d iOSBinaryFile > iOSBinaryFile_codesign.txt
#   for app
codesign -vv -d xxx.App > iOSApp_codesign.txt
```

- 特殊：
 - 如果二进制是 FAT 格式= 胖二进制，那么对于 jtool2，要指定架构才能继续

```
export ARCH=arm64
jtool2 -h iOSBinaryFile > iOSBinaryFile_jtool2_h_header.txt
```

举例

HeiBao的dylib

对于二级制文件，此处是dylib的动态库：

```
→ DynamicLibraries pwd
/Users/crifan/dev/DevRoot/Aweme/exportFromiPhone/iPhoneX-137/Library/MobileSubstrate/DynamicLibraries
→ DynamicLibraries ll
total 152568
-rwxr-xr-x@ 1 crifan staff 6.2M 3 14 10:34 zzzzHeiBaoLib.dylib
-rw-r--r-- 1 crifan staff 68M 3 17 21:39 zzzzHeiBaoLib.164
```

去导出字符串等资源：

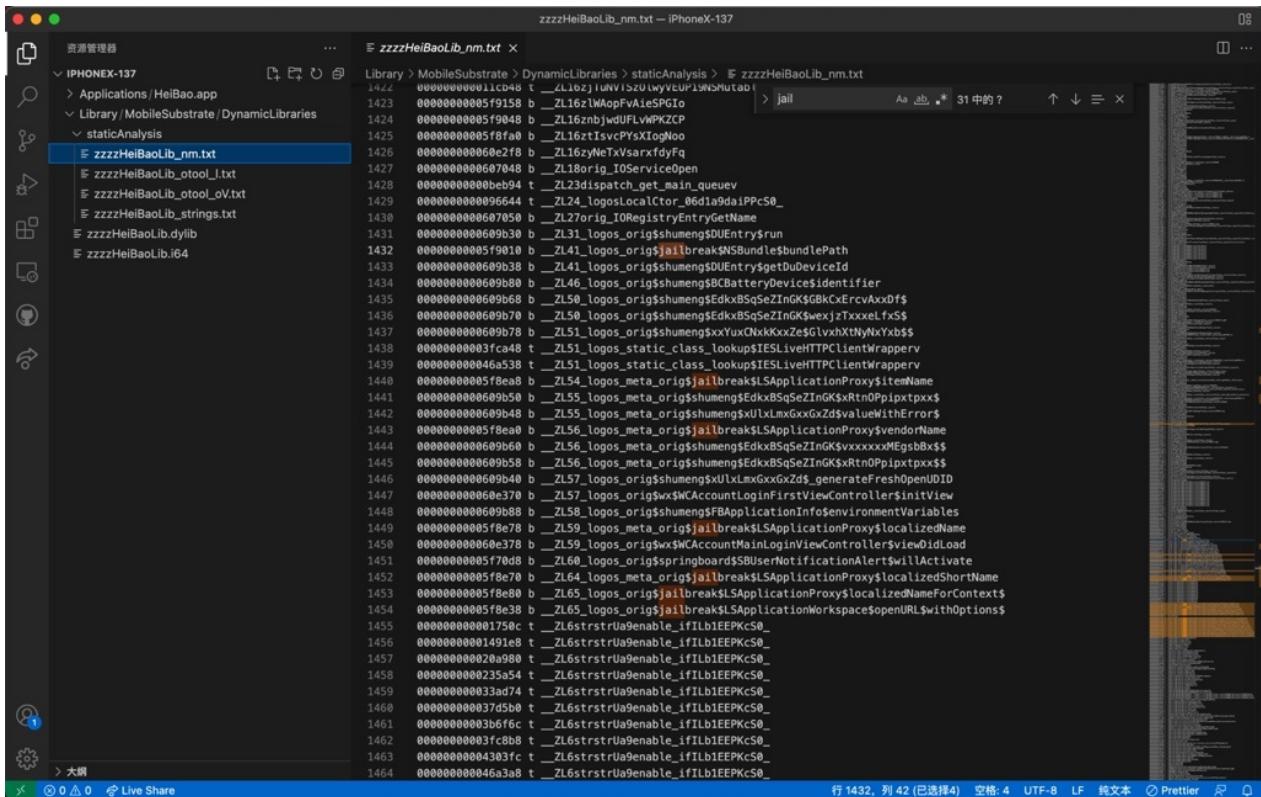
```
→ DynamicLibraries otool -l zzzzHeiBaoLib.dylib > HeiBaoLib_otool_l.txt
→ DynamicLibraries otool -oV zzzzHeiBaoLib.dylib > HeiBaoLib_otool_oV.txt
→ DynamicLibraries nm zzzzHeiBaoLib.dylib > HeiBaoLib_nm.txt
→ DynamicLibraries strings zzzzHeiBaoLib.dylib > HeiBaoLib_strings.txt

→ DynamicLibraries ll
total 153112
-rw-r--r-- 1 crifan staff 108K 3 21 10:00 HeiBaoLib_nm.txt
-rw-r--r-- 1 crifan staff 12K 3 21 10:00 HeiBaoLib_otool_l.txt
-rw-r--r-- 1 crifan staff 51K 3 21 10:00 HeiBaoLib_otool_oV.txt
-rw-r--r-- 1 crifan staff 89K 3 21 10:00 HeiBaoLib_strings.txt
-rwxr-xr-x@ 1 crifan staff 6.2M 3 14 10:34 zzzzHeiBaoLib.dylib
-rw-r--r-- 1 crifan staff 68M 3 17 21:39 zzzzHeiBaoLib.164
```

后续即可去分析和搜索想要研究的值了。

比如：

搜索越狱 jailbreak 相关内容：



抖音的AwemeCore

```
jtool2 -h ../../../../../已脱壳/v18.9.0/Payload/Aweme.app/Frameworks/AwemeCore.framework/AwemeCore
```

```
jtool2 -l ../../../../../已脱壳/v18.9.0/Payload/Aweme.app/Frameworks/AwemeCore.framework/AwemeCore > AwemeCore_jtool2_l_list.txt
jtool2 -L ../../../../../已脱壳/v18.9.0/Payload/Aweme.app/Frameworks/AwemeCore.framework/AwemeCore > AwemeCore_jtool2_L_library.txt
```

```
jtool2 -S ../../../../../已脱壳/v18.9.0/Payload/Aweme.app/Frameworks/AwemeCore.framework/AwemeCore > AwemeCore_jtool2_S.txt
```

```
jtool2 --analyze ../../../../../已脱壳/v18.9.0/Payload/Aweme.app/Frameworks/AwemeCore.framework/AwemeCore > AwemeCore_jtool2_analyze.txt
```

```
rabin2 -I ../../../../../已脱壳/v18.9.0/Payload/Aweme.app/Frameworks/AwemeCore.framework/AwemeCore
```

```
rabin2 -i ../../../../../已脱壳/v18.9.0/Payload/Aweme.app/Frameworks/AwemeCore.framework/AwemeCore > AwemeCore_rabin2_i.txt
```

```
rabin2 -E ../../../../../已脱壳/v18.9.0/Payload/Aweme.app/Frameworks/AwemeCore.framework/AwemeCore > AwemeCore_rabin2_E.txt
```

```
rabin2 -l ../../../../../已脱壳/v18.9.0/Payload/Aweme.app/Frameworks/AwemeCore.framework/AwemeCore > AwemeCore_rabin2_l.txt
```

```
rabin2 -z ../../../../../已脱壳/v18.9.0/Payload/Aweme.app/Frameworks/AwemeCore.framework/AwemeCore > AwemeCore_rabin2_z.txt
```

```
rabin2 -s ../../../../../已脱壳/v18.9.0/Payload/Aweme.app/Frameworks/AwemeCore.framework/AwemeCore > AwemeCore_rabin2_s.txt

rabin2 -S ../../../../../已脱壳/v18.9.0/Payload/Aweme.app/Frameworks/AwemeCore.framework/AwemeCore > AwemeCore_rabin2_S_section.txt
```

MaskPro.dylib

```
→ DynamicLibraries otool -l MaskPro.dylib > MaskProDylib/MaskProDylib_otool_l.txt
→ DynamicLibraries otool -ov MaskPro.dylib > MaskProDylib/MaskProDylib_otool_ov.txt
→ DynamicLibraries nm MaskPro.dylib > MaskProDylib/MaskProDylib_nm.txt
→ DynamicLibraries strings MaskPro.dylib > MaskProDylib/MaskProDylib_strings.txt
```

期间遇到 `FAT = 胖二进制` 的问题：

```
→ DynamicLibraries jtool2 -h MaskPro.dylib > MaskProDylib/MaskProDylib_jtool2_h_header.txt
Fat binary, little-endian, 2 architectures: armv7, arm64
Select an architecture setting the ARCH= environment variable
```

解决办法：

```
→ DynamicLibraries export ARCH=arm64
→ DynamicLibraries jtool2 -h MaskPro.dylib > MaskProDylib/MaskProDylib_jtool2_h_header.txt
```

继续：

```
→ DynamicLibraries jtool2 -l MaskPro.dylib > MaskProDylib/MaskProDylib_jtool2_l_list.txt
→ DynamicLibraries jtool2 -L MaskPro.dylib > MaskProDylib/MaskProDylib_jtool2_L_library.txt
→ DynamicLibraries jtool2 -S MaskPro.dylib > MaskProDylib/MaskProDylib_jtool2_S_symbol.txt
→ DynamicLibraries jtool2 --analyze MaskPro.dylib > MaskProDylib/MaskProDylib_jtool2_analyze.txt
...
```

继续：

```
→ DynamicLibraries rabin2 -I MaskPro.dylib > MaskProDylib/MaskProDylib_rabin2_I_identification.txt
→ DynamicLibraries rabin2 -i MaskPro.dylib > MaskProDylib/MaskProDylib_rabin2_i_imports.txt
→ DynamicLibraries rabin2 -E MaskPro.dylib > MaskProDylib/MaskProDylib_rabin2_E_exports.txt
→ DynamicLibraries rabin2 -l MaskPro.dylib > MaskProDylib/MaskProDylib_rabin2_l_libraries.txt
→ DynamicLibraries rabin2 -z MaskPro.dylib > MaskProDylib/MaskProDylib_rabin2_z_string
```

```
s.txt
→ DynamicLibraries rabin2 -s MaskPro.dylib > MaskProDylib/MaskProDylib_rabin2_s_symbol
s.txt
→ DynamicLibraries rabin2 -S MaskPro.dylib > MaskProDylib/MaskProDylib_rabin2_S_sectio
ns.txt
```

Mask的dylib

对于一个二进制，此处是一个动态库文件 `Mask.dylib`，想要导出字符串等资源，供后续分析。

- 典型的成套的做法 = `otool + nm + strings + jtool2 + rabin2` :

```
otool -l Mask.dylib > MaskDylib_otool_l.txt
otool -ov Mask.dylib > MaskDylib_otool_ov.txt

nm Mask.dylib > MaskDylib_nm.txt

strings Mask.dylib > MaskDylib_strings.txt

jtool2 -h Mask.dylib > MaskDylib_jtool2_h_header.txt
jtool2 -l Mask.dylib > MaskDylib_jtool2_l_list.txt
jtool2 -L Mask.dylib > MaskDylib_jtool2_L_library.txt
jtool2 -S Mask.dylib > MaskDylib_jtool2_S_symbol.txt
jtool2 --analyze Mask.dylib > MaskDylib_jtool2_analyze.txt

rabin2 -I Mask.dylib > MaskDylib_rabin2_I_identification.txt
rabin2 -i Mask.dylib > MaskDylib_rabin2_i_imports.txt
rabin2 -E Mask.dylib > MaskDylib_rabin2_E_exports.txt
rabin2 -l Mask.dylib > MaskDylib_rabin2_l_libraries.txt
rabin2 -z Mask.dylib > MaskDylib_rabin2_z_strings.txt
rabin2 -s Mask.dylib > MaskDylib_rabin2_s_symbols.txt
rabin2 -S Mask.dylib > MaskDylib_rabin2_S_sections.txt
```

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:

2023-07-10 11:06:27

相关工具

对于iOS逆向中的静态分析，其实还有其他一些相关工具，值得介绍。

- `radare2`：成套的逆向工具
 - 包含多个命令行的工具，前面提到的`rabin2`就是其中之一
- `Cutter`：`radare2`的GUI版本
- `Miasm`
 - 一句话介绍：用Python实现的逆向工程框架
 - Reverse engineering framework in Python
 - 用途：分析、修改、生成二进制程序
 - analyze / modify / generate binary programs
 - 特性
 - Opening / modifying / generating PE / ELF 32 / 64 LE / BE
 - Assembling / Disassembling X86 / ARM / MIPS / SH4 / MSP430
 - Representing assembly semantic using intermediate language
 - Emulating using JIT (dynamic code analysis, unpacking, ...)
 - Expression simplification for automatic de-obfuscation
 - 资料
 - GitHub
 - cea-sec/miasm: Reverse engineering framework in Python
 - <https://github.com/cea-sec/miasm>
 - 官网
 - Home — Miasm's blog
 - <https://miasm.re/blog/>

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：

2022-10-26 15:50:56

radare2

- radare2
 - 是什么：一个著名的开源逆向工程平台
 - Unix-like reverse engineering framework and commandline tools
 - 评价
 - 可谓是一大神器
 - 支持包括反汇编、分析数据、打补丁、比较数据、搜索、替换、虚拟化等等，同时具备超强的脚本加载能力，并且可以运行在几乎所有主流的平台

- 竞品

- IDA

- 截图

The screenshot shows the radare2 interface with several panes:

- [X] Registers [On]**: Shows CPU registers (rax, rcx, rdx, rsi, rdi, rbp, r8, r9, r10, r11, r12, r13, r14, r15) and memory offsets.
- [Cache] Off**: Shows assembly code for the current function, with various assembly instructions like jne, lea, mov, test, and cmp.
- [Cache] On**: Shows the list of functions found in the binary.
- [X] Disassembly [pd]**: Shows the assembly code for the current function, with annotations for comments, classes, and symbols.
- [Cache] Off**: Shows assembly code for the current function, with annotations for comments, classes, and symbols.
- [X] Functions [af1]**: Shows the list of functions found in the binary.
- [Cache] On**: Shows assembly code for the current function, with annotations for comments, classes, and symbols.
- [X] Entropy Fix**: Shows entropy analysis for strings in data sections.
- [X] Hexdump**: Shows the hex dump of the memory dump pane.
- [Cache] Off**: Shows assembly code for the current function, with annotations for comments, classes, and symbols.
- [X] agfb**: Shows a memory dump pane with a grid of hex values.

The screenshot shows the radare2 interface with several panes:

- [0x100001206]> pd 10**: Shows assembly dump of 10 bytes starting at 0x100001206.
- [0x100001206]> pd 4 @..0d**: Shows assembly dump of 4 bytes starting at 0x100001206 + 0xd.
- [0x100001206]> pd 4 @..225**: Shows assembly dump of 4 bytes starting at 0x100001206 + 0x225.
- [0x100001206]>**: Shows assembly dump of 4 bytes starting at 0x100001206.

```

    mov r14d, ec
    lea rax, [rbp-local_200]
    mov qword [rbp-local_201], rax
    test r14d, r14d
    jg 0x1000011a6
    f t
    0x1000011a1
    call sym.func.1000043ff
    v
    0x1000011a6
    lea rsi, [rip + 0x3943]
    xor edi, edi
    call sym.imp.setlocale
    mov r12d, 1
    mov edi, 1
    call sym.imp.isatty
    test eax, eax
    je 0x100001229
    t f
    0x100001229
    lea rdi, [rip + 0x38c1]
    call sym.imp.getenv
    test rax, rax
    je 0x100001248
    f t
    0x1000011c8
    mov dword [rip + 0x42fe], 0x50
    lea rdi, [rip + 0x3918]
    call sym.imp.getenv
    test rax, rax
    je 0x1000011f2
    f t
    0x10000123a
    mov rdi, rax
    call sym.imp atoi
    mov dword [rip + 0x4288], eax
    v
    0x1000011e3
    cmp byte [rax], 0
    je 0x1000011f2
    f t
    0x1000011e8
    mov rdi, rax
    call sym.imp atoi
    jmp 0x100001214
    v
    0x1000011f2
    lea rdx, [rbp-local_6]
    mov edi, 1
    mov esi, 0x40087468
    xor eax, eax
    call sym.imp.ioctl
    cmp eax, -1
    je 0x10000121a
    f t
  
```

- 支持平台

- Mac
- Windows
- Linux
- Solaris
- Android
- iOS
- Haiku

- 历史

- Radare project started as a forensics tool, a scriptable commandline hexadecimal editor able to open disk files but later support for analyzing binaries, disassembling code, debugging programs, attaching to remote gdb servers

- 功能: Radare is a portable reversing framework that can

- Disassemble (and assemble for) many different architectures
- Debug with local native and remote debuggers (gdb, rap, webui, r2pipe, winedbg, windbg)
- Run on Linux, *BSD, Windows, OSX, Android, iOS, Solaris and Haiku
- Perform forensics on filesystems and data carving

- Be scripted in Python, Javascript, Go and more
- Support collaborative analysis using the embedded webserver
- Visualize data structures of several file types
- Patch programs to uncover new features or fix vulnerabilities
- Use powerful analysis capabilities to speed up reversing
- Aid in software exploitation

- 特性

- Batch, commandline, visual and panels interactive modes
- Embedded webserver with js scripting and webui
- Assemble and disassemble a large list of CPUs
- Runs on Windows and any other UNIX flavour out there
- Analyze and emulate code with ESIL
- Native debugger and GDB, WINDBG, QNX and FRIDA
- Navigate ascii-art control flow graphs
- Ability to patch binaries, modify code or data
- Search for patterns, magic headers, function signatures
- Easy to extend and modify
- Commandline, C API, script with r2pipe in any language

- 包含工具

- rabin2

- 获取 ELF , PE , Mach-O , Java CLASS 文件的区段、头信息、导入导出表、字符串相关、入口点等等
 - 包括打印出二进制文件的系统属性、语言、字节序、框架、以及使用了哪些加固技术
- 支持多种格式的输出文件
- 截图

```
root@kali:~/study# rabin2 -I megabeets_0x1
arch      x86
binsz    6220
bintype   elf
bits      32
canary    false
class     ELF32
crypto    false
endian   little
havecode  true
intrp    /lib/ld-linux.so.2
lang      c
linenum   true
lsyms    true
machine   Intel 80386
maxopsz  16
minopsz  1
nx       false
os       linux
pcalign   0
pic      false
relocs   true
relro    partial
```

- radiff2 : 比较文件不同的
- rahash2 : 各种密码算法, hash算法集成
- rasim2 : 汇编和反汇编
- ragg2 : 开发shellcode工具(radare2自己编写的编译器)
- radare2 : 整合了所有工具

- 资料
 - 官网
 - radare
 - <https://rada.re/n/radare2.html>
 - GitHub
 - radareorg/radare2: UNIX-like reverse engineering framework and command-line toolset
 - <https://github.com/radareorg/radare2>
 - 教程
 - The Official Radare2 Book
 - <https://book.rada.re/index.html>

help帮助语法

```
$ radare2 -h
Usage: r2 [-ACdfLMnNqStuvwzX] [-P patch] [-p prj] [-a arch] [-b bits] [-i file]
           [-s addr] [-B baddr] [-m maddr] [-c cmd] [-e k-v] file|pid - | -- =
--           run radare2 without opening any file
-           same as 'r2 malloc://512'
=           read file from stdin (use -i and -c to run cmds)
-=          perform := command to run all commands remotely
-0          print \x00 after init and every command
-2          close stderr file descriptor (silent warning messages)
-a [arch]   set asm.arch
-A          run 'aaa' command to analyze all referenced code
-b [bits]   set asm.bits
-B [baddr]  set base address for PIE binaries
-c 'cmd..'  execute radare command
-C          file is host:port (alias for -c=http://%s/cmd/)
-d          debug the executable 'file' or running process 'pid'
-D [backend] enable debug mode (e cfg.debug=true)
-e k-v     evaluate config var
-f          block size = file size
-F [binplug] force to use that rbin plugin
-h, -hh    show help message, -hh for long
-H ([var])  display variable
-i [file]   run script file
-I [file]   run script file before the file is opened
-k [OS/kern] set asm.os (linux, macos, w32, netbsd, ...)
-l [lib]    load plugin file
-L          list supported IO plugins
-m [addr]   map file at given address (loadaddr)
-M          do not demangle symbol names
-n, -nn    do not load RBin info (-nn only load bin structures)
-N          do not load user settings and scripts
-q          quiet mode (no prompt) and quit after -i
-Q          quiet mode (no prompt) and quit faster (quickLeak true)
-p [prj]    use project, list if no arg, load if no file
-P [file]   apply rapatch file and quit
-r [rarun2] specify rarun2 profile to load (same as -e dbg.profile=X)
-R [rr2rule] specify custom rarun2 directive
-s [addr]   initial seek
-S          start r2 in sandbox mode
```

```
-t          load rabin2 info in thread
-u          set bin.filter=false to get raw sym/sec/cls names
-v, -V      show radare2 version (-V show lib versions)
-w          open file in write mode
-x          open without exec-flag (asm.emu will not work), See io.exec
-X          same as -e bin.usextr=false (useful for dyldcache)
-z, -zz     do not load strings or load them even in raw
```

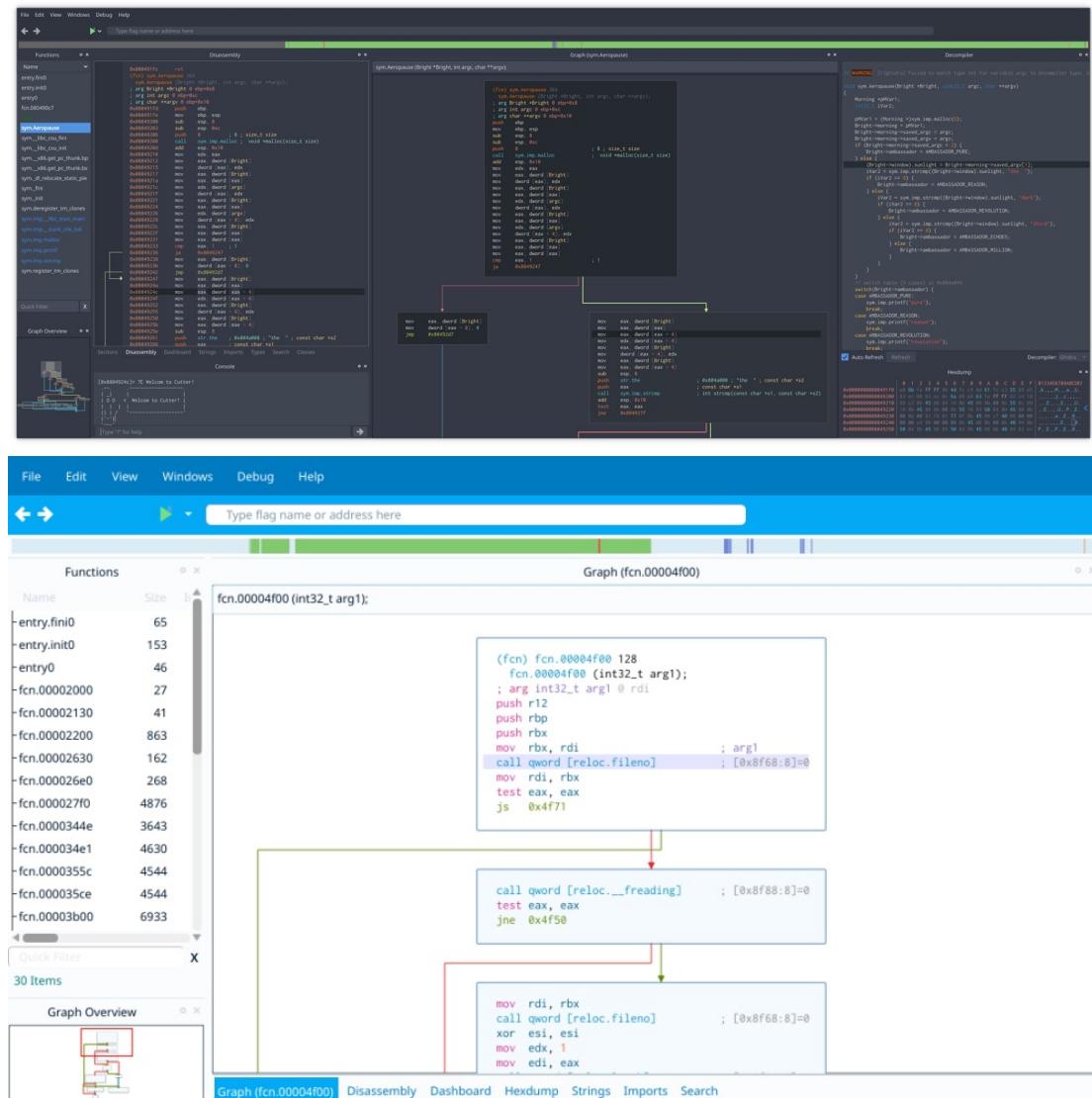
R2Pipe

- R2Pipe
 - 是什么： R2Pipe 是一个可以调用 radare2 的 Python 脚本库
 - 示例代码
 - <https://github.com/radareorg/radare2-r2pipe/tree/master/python/examples>

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：
2022-10-23 15:17:26

Cutter

- Cutter
 - 一句话描述：(radare2的fork版) Rizin的GUI版本
 - Free and Open Source RE Platform powered by radare2
 - Cutter is the official UI for radare2 for Linux, macOS and Windows, it's written in C++ and uses the Qt
 - 支持多平台
 - Linux
 - Mac
 - Windows
 - 实现细节
 - C++ 语言写的
 - 前端：QT
 - 截图



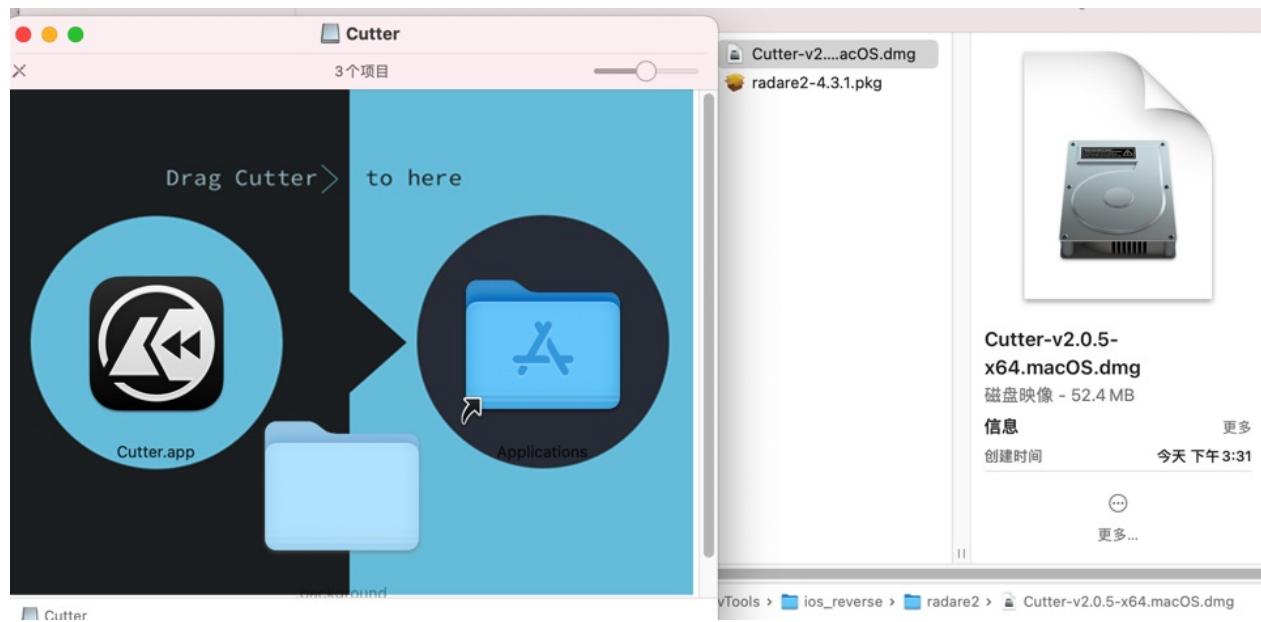
- 特点
 - 内置Ghidra decompiler

- 无需额外安装Java
 - 用C++实现的，目的是性能更好
- 核心功能和特点
 - 开源 Open Source
 - Completely FREE and licensed under GPLv3
 - Decompiler
 - Native integration of Ghidra's decompiler in Cutter releases
 - Graph View
 - Fully featured graph view as well as mini-graph for fast navigation
 - Debugger
 - Multiplatform native and remote debugger for dynamic analysis
 - Disassembly
 - Linear disassembly view
 - Hex Editor
 - View and modify any file with a rich and powerful Hex View
 - Python Scripting Engine
 - Quickly write python scripts to automate tasks
 - Plugins
 - Use Native or Python plugins to extend Cutter's core functionality
 - Binary Patching
 - Add, remove and modify bytes and instructions
 - Emulation
 - Great for automation, crypto algorithms and malware analysis
 - Theme Editor
 - Fully featured theme editor for easy and user-friendly customization of Cutter
 - Modern & Customizable UI
 - Built using Qt C++ and design best practices
 - Integrated Radare2 Console
 - Multi Language
 - Binary Searching
 - Types & Structs
 - Syntax Highlighting
 - STDIO Redirection
 - Remote Debugging
 - Kernel Debug
 - Graph Overview
- 资料
 - 官网
 - Cutter
 - <https://cutter.re>
 - GitHub
 - radareorg/cutter: Free and Open Source Reverse Engineering Platform powered by radare2
 - <https://github.com/rizinorg/cutter>
 - rizinorg/rizin: UNIX-like reverse engineering framework and command-line toolset.
 - <https://github.com/rizinorg/rizin>

安装Cutter

Cutter官网点击Download，此处下载到：（之前某旧版本）

Cutter-v2.0.5-x64.macOS.dmg



双击，即可安装。

打开后，即可看到主界面：

- Cutter欢迎界面

- - Cutter的关于

◦

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：
2023-10-07 14:46:01

Cutter使用举例

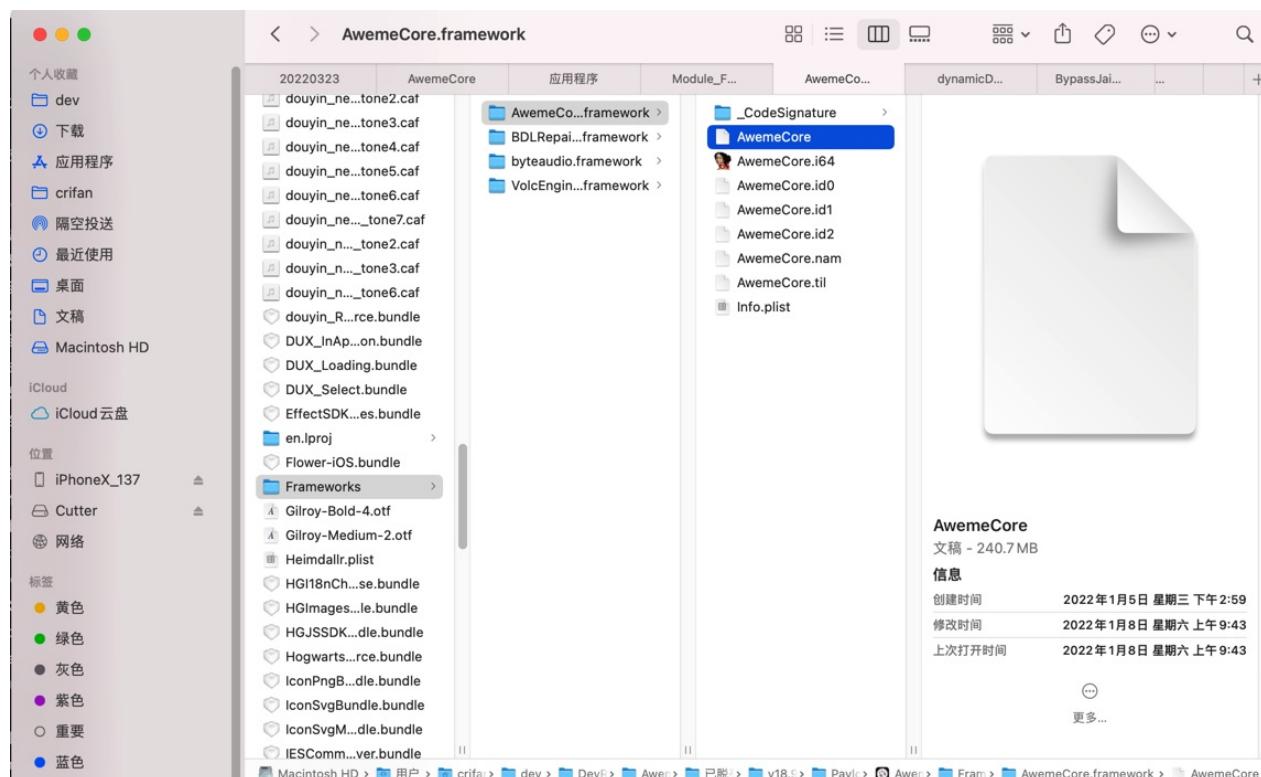
crifan.org, 使用[署名4.0国际\(CC BY 4.0\)协议](#)发布 all right reserved, powered by Gitbook最后更新：
2023-10-07 14:39:50

Cutter使用举例：AwemeCore

打开页面：



把要逆向的文件 AwemeCore 拖动过去：



加载选项:



点击展开 高级选项 看看:



各个参数选项：

- `architecture` : 支持很多类型

- - bits

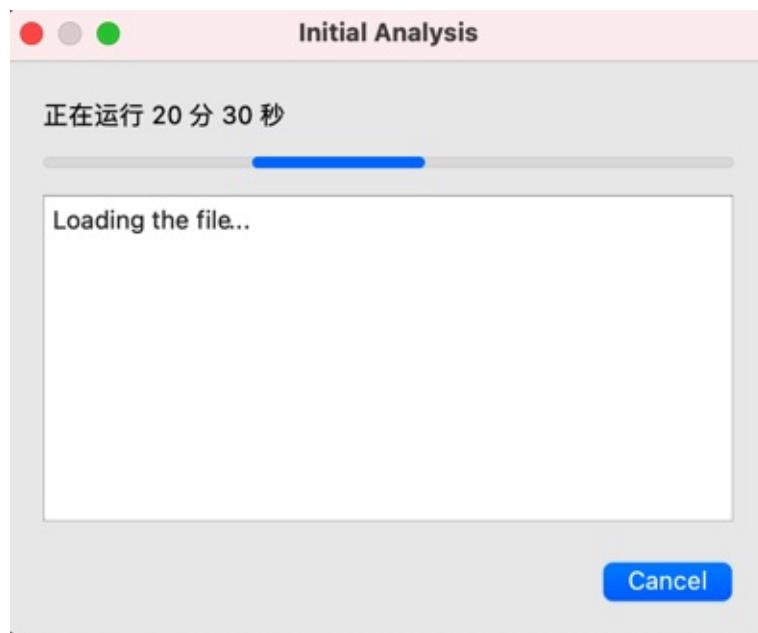
- - Endianness

- - kernel

- - format

◦

开始加载和分析：



运行了几个小时，都没结束。效率太低。那算了，最终放弃。

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：
2023-10-07 14:53:29

静态分析心得

此处整理iOS逆向期间的静态分析反面的心得。

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:
2023-10-21 10:31:18

二进制中找不到类

当前二进制找不到函数时，找别的二进制看看：

概述：

- iOS逆向期间，如果当前二进制中找不到某个类和函数时，可以去试试：找其他二进制
 - 以及可以从加载动态库依赖中，确认是否是依赖于别的二进制
 - 对应的 Load Command 中是否有 LC_LOAD_DYLIB，去加载（同一个iOS的app内部的其他的）二进制文件
 - 对应的 library 中是否有别的二进制文件
 - 以及二进制查看函数
 - 一个是 U = Undefined：是当前二进制未定义的，需要从别的二进制导入的
 - 一个是 D = Defined：是当前二进制实现的类和函数

WhatsApp找不到WAURLQueryItem，但别的二进制SharedModules中可以找到

iOS的app WhatsApp 中的主要二进制是：

- WhatsApp

◦

去逆向期间，发现 IDA 伪代码中：

IDA - WhatsApp /Users/crifan/dev/dev_root/iosReverse/WhatsApp/ipa/Payload/WhatsApp.app/WhatsApp

Library function Regular function Instruction Data Unexplored External symbol Lumina function

Functions

```

1 Functions
2 sub_10224E900
3 [-WAFRegistrationToken randomToken]
4 [-WAFRegistrationToken fromEncodedData:]
5 [+WAFRegistrationToken fromWKKeychainTokenData:]
6 [+WAFRegistrationToken fromEncodedData oldEncoding...]
7 [-WAPreparedRegistrationURL initWithBaseUrlString...]
8 [-WAPreparedRegistrationURL urlWithRegistrationToke...]
9 [-WAPreparedRegistrationURL urlWithTokenArray:]
10 [-WAPreparedRegistrationURL _destruct]
11 [-WAFRegistrationURLBuilder initWithBaseURL:lan...]
12 [-WAFRegistrationURLBuilder enableProxyingWithProxy...]
13 [-WAFRegistrationURLBuilder disableProxying...]
14 [-WAFRegistrationURLBuilder clientLogRequestURLWith...]
15 [-WAFRegistrationURLBuilder clientLogRequestURLWith...]
16 [-WAFRegistrationURLBuilder sameDeviceCheckRequest...]
17 [-WAFRegistrationURLBuilder verifyItemWithNameValue...]
18 [-WAFRegistrationURLBuilder registerDomainProbe...]
19 [-WAFRegistrationURLBuilder twoFactorVerifyRequest...]
20 [-WAFRegistrationURLBuilder twoFactorResetViaEmail...]
21 [-WAFRegistrationURLBuilder twoFactorResetWithWipeT...]
22 [-WAFRegistrationURLBuilder cxx_destruct]
23 [+WAFRegistrationURLBuilder encodedWAUUID:]
24 [+WAFRegistrationURLBuilder registrationDomainProbe...]
25 [+WAFRegistrationURLBuilder preChatAPIPropURLWith...]
26 [+WAFRegistrationURLBuilder preChatAPIPropURLWith...]
27 [+WAFRegistrationURLBuilder sameDeviceCheckURL]
28 [+WAFRegistrationURLBuilder twoFactorAuthenticationU...]
29 [+WAFRegistrationURLBuilder registrationDomainProbe...]
30 [+WAFRegistrationURLBuilder clientLogURL]
31 [-WABProperties hideTableRowWithHandler...]
32 [-WAProfileMyaboutTableRowDidUpdate...]
33 [-WAProfileMyaboutTableRow updateAbout...]
34 [-WAProfileMyaboutTableRow userContext]
35 [-WAABProperties cross_viewcontroller_preloader_scr...]
36 [-WAABProperties enable_backup_token_for_registerati...]
37 [-WAABProperties hide_link_device_button_enabled!]

Line 330980 of 511693

```

Output

```

1030A8368: using guessed type __CFString cfstr_CellularStrength;
1035EE908: using guessed type __objc_z_class OBJC_CLASS_WAPreparedRegistrationURL;

```

Python

A: idle Down Disk: 292GB

```

id __cdecl [-WAFRegistrationURLBuilder verificationCodeRequestURLWithBaseURL method mcc
mnc jailbroken context oldPhoneNumber silentPushNotifRegCode cellularStrength]()

    WAResponseBuilder self,
    SEL a2,
    id a3,
    id a4,
    id a5,
    id a6,
    bool a7,
    id a8,
    id a9,
    id a10,
    id a11)

{
    ...
}

v46 = objc_msgSend_queryItemWithName_value_(OBJC_CLASS_WAURLQueryItem, v43, CFSTR("reason"), CFSTR("jailbroken"));

```

其中有：

- `OBJC_CLASS_WAURLQueryItem`
 - iOS的类：`WAURLQueryItem`

但是，此处WhatsApp的主要二进制 `WhatsApp` 中，竟然找不到 `WAURLQueryItem`

具体现象是：

WhatsApp 发现类 `WAURLQueryItem` 是未定义，需要额外引入的，且另外需要加载的库文件有 `SharedModules.framework/SharedModules`

- WhatsApp

(1) 导出的头文件中

- 搜不到 WAURLQueryItem

◦

(2) 导出的字符串等资源中

- 能搜到 WAURLQueryItem 的字眼

◦

具体搜索到的内容有

```
WhatsApp_jtool2_S_symbol.txt:  
8954          U_OBJC_CLASS_$_WATooltipView  
8955          U_OBJC_CLASS_$_WATooltipView  
8956:         U_OBJC_CLASS_$_WAURLQueryItem  
8956          U_OBJC_CLASS_$_WAUUID  
8957          U_OBJC_CLASS_$_WAUUID  
  
WhatsApp_nm.txt:  
8954          U_OBJC_CLASS_$_WATooltipView  
8955          U_OBJC_CLASS_$_WAURLQueryItem  
8956          U_OBJC_CLASS_$_WAUUID
```

```
WhatsApp_otool_oV.txt:
1050534 0000000103417970 0x0 _OBJC_CLASS_$_NSURLComponents
1050535: 0000000103417978 0x0 _OBJC_CLASS_$_WAURLQueryItem
1050536 0000000103417980 0x0 _OBJC_CLASS_$_WACacheQueue

WhatsApp_rabin2_i_imports.txt:
8957 8953 ----- NONE OBJC_CLASS WATooltipView
8958: 8954 ----- NONE OBJC_CLASS WAURLQueryItem
8959 8955 ----- NONE OBJC_CLASS WAUUID
```

- 具体分析：
 - WhatsApp_jtool2_S_symbol.txt 中的
 - 8956: U _OBJC_CLASS_\$_WAURLQueryItem
 - 相关含义: U=Undefined=未定义
 - WhatsApp_rabin2_i_imports.txt 中的
 - 8958: 8954 ----- NONE OBJC_CLASS WAURLQueryItem
 - 中有WAURLQueryItem: 表示WAURLQueryItem是属于（从外部）import进来的
- -> 表示此类: WAURLQueryItem, 是从外部导入import进来的, 当前二进制 (WhatsApp) 中是没有此类的具体实现的

(3) 后记: 在依赖的导入的Library库中, 也能找到, 是依赖于: 库 SharedModules 的

具体现象是:

导出的字符串资源文件中, 可以找到:

```
13 个结果 - 8 文件

WhatsApp_jtool2_L_library.txt:
20      /System/Library/Frameworks/CoreBluetooth.framework/CoreBluetooth (compatibility version 1.0.0, current version 1.0.0)
21      /System/Library/Frameworks/CoreBluetooth.framework/CoreBluetooth (compatibility version 1.0.0, current version 1.0.0)
22:      @rpath/SharedModules.framework/SharedModules (compatibility version 1.0.0, current version 1.0.0)
22      @rpath/WARCDManager.framework/WARCDManager (compatibility version 0.0.0, current version 0.0.0)
23      @rpath/WARCDManager.framework/WARCDManager (compatibility version 0.0.0, current version 0.0.0)

WhatsApp_jtool2_l_list.txt:
88  LC 32: LC_LOAD_DYLIB           /System/Library/Frameworks/CoreBluetooth.framework/CoreBluetooth
89: LC 33: LC_LOAD_DYLIB           @rpath/SharedModules.framework/SharedModules
90  LC 34: LC_LOAD_WEAK_DYLIB     @rpath/WARCDManager.framework/WARCDManager
...
.

WhatsApp_otool_l.txt:
Load command 33
    cmd LC_LOAD_DYLIB
    cmdsize 72
    name @rpath/SharedModules.framework/SharedModules (offset 24)
```

```
time stamp 2 Thu Jan 1 08:00:02 1970
current version 1.0.0
compatibility version 1.0.0

...
WhatsApp_rabin2_l_libraries.txt:
21 /System/Library/Frameworks/CoreBluetooth.framework/CoreBluetooth
22: @rpath/SharedModules.framework/SharedModules
23 @rpath/WARCDManager.framework/WARCDManager

...
```

具体解释是：

- WhatsApp_jtool2_L_library.txt
 - 22: @rpath/SharedModules.framework/SharedModules (compatibility version 1.0.0, current version 1.0.0)
 - 表示要，当前二进制 WhatsApp 加载时，所依赖的，还要额外加载的库是
 - SharedModules.framework/SharedModules
- WhatsApp_jtool2_l_list.txt
 - 89: LC 33: LC_LOAD_DYLIB @rpath/SharedModules.framework/SharedModules
 - 表示还要额外加载一个 Dylib=动态库文件： SharedModules.framework/SharedModules
- WhatsApp_otool_l.txt
 - Load command 33 : 是有一个Load Command
 - cmd LC_LOAD_DYLIB : command类型是要加载 Dylib =动态库
 - name @rpath/SharedModules.framework/SharedModules (offset 24) : 具体要加载的库文件是： SharedModules.framework/SharedModules

SharedModules 中找到了类 WAURLQueryItem 的定义

最后是从同一个iOSapp： WhatsApp 的：

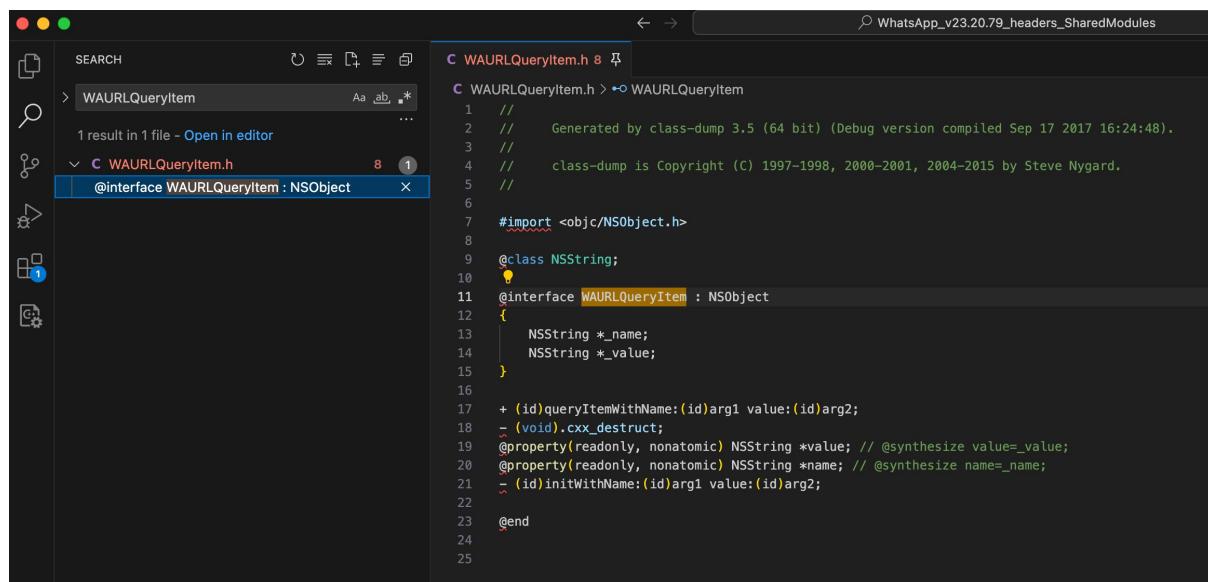
- 另外一个核心二进制 SharedModules

◦

中找到了：类 WAURLQueryItem 的具体实现

具体现象是：

(1) SharedModules的导出头文件中，能搜到：类WAURLQueryItem



The screenshot shows the Xcode interface with a search result for 'WAURLQueryItem'. The search bar at the top has 'WAURLQueryItem' typed into it. Below the search bar, there is a list of results: '1 result in 1 file - Open in editor'. Underneath this, there is a tree view showing a single file: 'C WAURLQueryItem.h'. The file content is displayed in the main editor area, showing the class definition:

```
// Generated by class-dump 3.5 (64 bit) (Debug version compiled Sep 17 2017 16:24:48).
// class-dump is Copyright (C) 1997-1998, 2000-2001, 2004-2015 by Steve Nygard.

#import <objc/NSSObject.h>

@class NSString;

@interface WAURLQueryItem : NSObject
{
    NSString *_name;
    NSString *_value;
}

+ (id)queryItemWithName:(id)arg1 value:(id)arg2;
- (void).cxx_destruct;
@property(readonly, nonatomic) NSString *value; // @synthesize value=_value;
@property(readonly, nonatomic) NSString *name; // @synthesize name=_name;
- (id)initWithName:(id)arg1 value:(id)arg2;
@end
```

(2) 以及SharedModules导出的字符串资源中，也能找到：类WAURLQueryItem

SharedModules_tool2_S_symbol.txt

```

搜索: WAURLQueryItem
7 个文件中有 25 个结果 - 在编辑器中打开
SharedModules_tool2_S_symbol.txt
0000000001a1ed20 D _OBJC_CLASS_$_WAURLQueryItem
SharedModules_nm.txt
0000000001a18b0 D _OBJC_METACLASS_$_WAURLQueryItem
00000000000895a01 t [WAURLQueryItem cxx_destruct]
00000000000895a01 t [WAURLQueryItem initWithNibName:value]
00000000000895a01 t [WAURLQueryItem name]
00000000000895a01 t [WAURLQueryItem value]
0000000001a1ed20 S _OBJC_CLASS_$_WAURLQueryItem
0000000001913b424 s _OBJC_IVAR_$_WAURLQueryItem_name
0000000001913b428 s _OBJC_IVAR_$_WAURLQueryItem_value
0000000001a018b0 S _OBJC_METACLASS_$_WAURLQueryItem
SharedModules_tool2_E_exports.txt
000000000161fafc8 0x10ed20 _OBJC_CLASS_$_WAURLQueryItem
0xa018b0 _OBJC_METACLASS_$_WAURLQueryItem
name 0x1427163 WAURLQueryItem
name 0x1427163 WAURLQueryItem
0000000001912f340 0x1a1ed20 _OBJC_CLASS_$_WAURLQueryItem
00000000019137518 0x1a1ed20 _OBJC_CLASS_$_WAURLQueryItem
SharedModules_rabin2_E_exports.txt
FUNC 0 _OBJC_CLASS_$_WAURLQueryItem
_OBJC_METACLASS_$_WAURLQueryItem
SharedModules_rabin2_S_symbols.txt
FUNC 0 _OBJC_CLASS_$_WAURLQueryItem
_OBJC_METACLASS_$_WAURLQueryItem
SharedModules_rabin2_L_strings.txt
_cstring ascii B1@0@"WAURLQueryItem"
__objc_classname ascii WAURLQueryItem
SharedModules_strings.txt
B1@0@"WAURLQueryItem"
SaySo4WAURLQueryItemCG
WAURLQueryItem

```

行 2881, 列 48 (已选择14) 空格: 2 UTF-8 LF 纯文本

SharedModules_ctool_lo.txt

```

搜索: WAURLQueryItem
7 个文件中有 25 个结果 - 在编辑器中打开
SharedModules_tool2_S_symbol.txt
0000000001a1ed20 D _OBJC_CLASS_$_WAURLQueryItem
0000000001a18b0 D _OBJC_METACLASS_$_WAURLQueryItem
00000000000895a01 t [WAURLQueryItem cxx_destruct]
00000000000895a01 t [WAURLQueryItem initWithNibName:value]
00000000000895a01 t [WAURLQueryItem name]
00000000000895a01 t [WAURLQueryItem value]
0000000001a1ed20 S _OBJC_CLASS_$_WAURLQueryItem
0000000001913b424 s _OBJC_IVAR_$_WAURLQueryItem_name
0000000001913b428 s _OBJC_IVAR_$_WAURLQueryItem_value
0000000001a018b0 S _OBJC_METACLASS_$_WAURLQueryItem
SharedModules_tool2_E_exports.txt
000000000161fafc8 0x10ed20 _OBJC_CLASS_$_WAURLQueryItem
0xa018b0 _OBJC_METACLASS_$_WAURLQueryItem
name 0x1427163 WAURLQueryItem
name 0x1427163 WAURLQueryItem
0000000001912f340 0x1a1ed20 _OBJC_CLASS_$_WAURLQueryItem
00000000019137518 0x1a1ed20 _OBJC_CLASS_$_WAURLQueryItem
SharedModules_rabin2_E_exports.txt
FUNC 0 _OBJC_CLASS_$_WAURLQueryItem
_OBJC_METACLASS_$_WAURLQueryItem
SharedModules_rabin2_S_symbols.txt
FUNC 0 _OBJC_CLASS_$_WAURLQueryItem
_OBJC_METACLASS_$_WAURLQueryItem
SharedModules_rabin2_L_strings.txt
_cstring ascii B1@0@"WAURLQueryItem"
__objc_classname ascii WAURLQueryItem
SharedModules_strings.txt
B1@0@"WAURLQueryItem"
SaySo4WAURLQueryItemCG
WAURLQueryItem

```

行 444383, 列 58 (已选择14) 空格: 4 UTF-8 LF 纯文本

25 个结果 - 7 文件

SharedModules_jtool2_S_symbol.txt:

```

2880 0000000001952928 D _OBJC_CLASS_$_WATrustSignalsDataFetcher
2881: 0000000001a1ed20 D _OBJC_CLASS_$_WAURLQueryItem
2882 0000000001a1ede8 D _OBJC_CLASS_$_WAUUID

6993 0000000001a26170 D _OBJC_METACLASS_$_WATrustSignalsDataFetcher
6994: 0000000001a018b0 D _OBJC_METACLASS_$_WAURLQueryItem
6995 0000000001a1ecf8 D _OBJC_METACLASS_$_WAUUID

```

SharedModules_nm.txt:

```

25025 00000000000ac75c t -[WATrustSignalsDataFetcher init]

```

```

25026: 00000000000b89ab0 t -[WAURLQueryItem cxx_destruct]
25027: 00000000000b89a04 t -[WAURLQueryItem initWithName:value:]
25028: 00000000000b89aa8 t -[WAURLQueryItem name]
25029: 00000000000b89aac t -[WAURLQueryItem value]
25030: 00000000000ee293c t -[WAUUID cxx_destruct]

92477 00000000001952928 S _OBJC_CLASS_$_WATrustSignalsDataFetcher
92478: 00000000001a1ed20 S _OBJC_CLASS_$_WAURLQueryItem
92479 00000000001a1ede8 S _OBJC_CLASS_$_WAUUID

101272 00000000001a261b8 d _OBJC_IVAR_$_WATrustSignalsDataFetcher.workQueue
101273: 0000000000193b424 s _OBJC_IVAR_$_WAURLQueryItem._name
101274: 0000000000193b428 s _OBJC_IVAR_$_WAURLQueryItem._value
101275 0000000000193dc64 s _OBJC_IVAR_$_WAUUID._creationDate

112634 00000000001a26170 D _OBJC_METACLASS_$_WATrustSignalsDataFetcher
112635: 00000000001a018b0 S _OBJC_METACLASS_$_WAURLQueryItem
112636 00000000001a1ecf8 S _OBJC_METACLASS_$_WAUUID

SharedModules_otool_oV.txt:
444381      baseProperties 0x0
444382: 0000000000161afc8 0x1a1ed20 _OBJC_CLASS_$_WAURLQueryItem
444383:     isa          0x1a018b0 _OBJC_METACLASS_$_WAURLQueryItem
444384:     superclass   0x0 _OBJC_CLASS_$_NSObject

444393      layout map    0x02
444394:     name        0x1427163 WAURLQueryItem
444395:     baseMethods  0x1822628

444442      ivarLayout   0x0
444443:     name        0x1427163 WAURLQueryItem
444444:     baseMethods  0x18225c0

672642 0000000000192f338 0x1a02120 _OBJC_CLASS_$_WASashedMessage
672643: 0000000000192f340 0x1a1ed20 _OBJC_CLASS_$_WAURLQueryItem
672644 0000000000192f348 0x1a16aa8 _OBJC_CLASS_$_WAPBPeerMessage

676826 000000000019375f0 0x1a01ba8 _OBJC_CLASS_$_WALogWriter
676827: 000000000019375f8 0x1a1ed20 _OBJC_CLASS_$_WAURLQueryItem
676828 00000000001937600 0x1a01928 _OBJC_CLASS_$_WAFuture

SharedModules_rabin2_E_exports.txt:
34621 34617 0x01952928 0x01952928 GLOBAL FUNC 0           _OBJC_CLASS_$_WATrustSignalsD
ataFetcher
34622: 34618 0x01a1ed20 0x01a1ed20 GLOBAL FUNC 0           _OBJC_CLASS_$_WAURLQueryItem
34623 34619 0x01a1ede8 0x01a1ede8 GLOBAL FUNC 0           _OBJC_CLASS_$_WAUUID

38734 38730 0x01a26170 0x01a26170 GLOBAL FUNC 0           _OBJC_METACLASS_$_WATrustSign
alsDataFetcher
38735: 38731 0x01a018b0 0x01a018b0 GLOBAL FUNC 0           _OBJC_METACLASS_$_WAURLQueryI
tem
38736 38732 0x01a1ecf8 0x01a1ecf8 GLOBAL FUNC 0           _OBJC_METACLASS_$_WAUUID

SharedModules_rabin2_s_symbols.txt:
34621 34617 0x01952928 0x01952928 GLOBAL FUNC 0           _OBJC_CLASS_$_WATrustSignalsD
ataFetcher

```

```

34622: 34618 0x01a1ed20 0x01a1ed20 GLOBAL FUNC 0      _OBJC_CLASS_$_WAURLQueryItem
34623  34619 0x01a1ede8 0x01a1ede8 GLOBAL FUNC 0      _OBJC_CLASS_$_WAUUID

38734  38730 0x01a26170 0x01a26170 GLOBAL FUNC 0      _OBJC_METACLASS_$_WATrustSign
alsDataFetcher
38735: 38731 0x01a018b0 0x01a018b0 GLOBAL FUNC 0      _OBJC_METACLASS_$_WAURLQueryI
tem
38736  38732 0x01a1ecf8 0x01a1ecf8 GLOBAL FUNC 0      _OBJC_METACLASS_$_WAUUID

SharedModules_rabin2_z_strings.txt:
18365 13101 0x0114de06 0x0114de06 34 35 5._TEXT.__cstring      ascii    mms-
base-task//finish/finishing%@
18366 13101 0x0114de06 0x0114de06 34 35 5._TEXT.__cstring      ascii    mms-
base-task//finish/finishing%@
18367: 13102 0x0114de29 0x0114de29 24 25 5._TEXT.__cstring      ascii    B16@?
?0@"WAURLQueryItem"8

18367 13103 0x0114de42 0x0114de42 7 8 5._TEXT.__cstring      ascii    _nc_
cat
18368 13103 0x0114de42 0x0114de42 7 8 5._TEXT.__cstring      ascii    _nc_
cat

114694 2073 0x0142715e 0x0142715e 4 5 12._TEXT.__objc_classname ascii    HTML
114695 2073 0x0142715e 0x0142715e 4 5 12._TEXT.__objc_classname ascii    HTML
114696: 2074 0x01427163 0x01427163 14 15 12._TEXT.__objc_classname ascii    WAUR
LQueryItem
114696 2075 0x01427172 0x01427172 13 14 12._TEXT.__objc_classname ascii    Data
Detection
114697 2075 0x01427172 0x01427172 13 14 12._TEXT.__objc_classname ascii    Data
Detection

SharedModules_strings.txt:
17521  mms-base-task//finish/finishing%@
17522: B16@?0@"WAURLQueryItem"8
17523  _nc_cat

112316  ySo11WAMessageIDCSo0A0C_G
112317: SaySo14WAURLQueryItemCG
112318  SDySo11WADeviceJIDC

114720  HTML
114721: WAURLQueryItem
114722  DataDetection

```

具体解释是：

- SharedModules_jtool2_S_symbol.txt
 - 中的
 - 2881: 0000000001a1ed20 D _OBJC_CLASS_\$_WAURLQueryItem
 - 6994: 0000000001a018b0 D _OBJC_METACLASS_\$_WAURLQueryItem
 - 其中：
 - D= Defined = 已定义 = 有定义
 - _OBJC_CLASS_ = 是ObjC的class类

- `_OBJC_METACLASS_` = 是ObjC的meta class=元类
- `SharedModules_nm.txt`
 - 25027: 0000000000b89a04 t -[WAURLQueryItem initWithNibName:value:]
 - `t= Text symbol, local (static)` = 本地的static的文本符号
 - 函数名: `-[WAURLQueryItem initWithNibName:value:]`
 - 92478: 0000000001a1ed20 S `_OBJC_CLASS_$_WAURLQueryItem`
 - `S= Section symbol, global` = 全局的节的符号, 此处ObjC的类WAURLQueryItem
 - 101273: 000000000193b424 s `_OBJC_IVAR_$_WAURLQueryItem._name`
 - `t= Text symbol, local (static)` = 本地的static的文本符号
 - ObjC类WAURLQueryItem的ivar变量名`_name`
 - 112635: 0000000001a018b0 S `_OBJC_METACLASS_$_WAURLQueryItem`
 - `S= Section symbol, global` = 全局的节的符号, 此处是ObjC的MetaClass元类WAURLQueryItem
- `SharedModules_otool_oV.txt`
 - 444382: 000000000161afc8 0x1a1ed20 `_OBJC_CLASS_$_WAURLQueryItem`
 - 是有ObjC的类: WAURLQueryItem
- `SharedModules_rabin2_E_exports.txt`
 - 中的
 - 34622: 34618 0x01a1ed20 0x01a1ed20 GLOBAL FUNC 0 `_OBJC_CLASS_$_WAURLQueryItem`
 - 38735: 38731 0x01a018b0 0x01a018b0 GLOBAL FUNC 0 `_OBJC_METACLASS_$_WAURLQueryItem`
 - 表示:
 - 有ObjC的普通类WAURLQueryItem 和ObjC的元类WAURLQueryItem, 要导出export (供别处调用)
- `SharedModules_rabin2_s_symbols.txt`
 - 中的
 - 34622: 34618 0x01a1ed20 0x01a1ed20 GLOBAL FUNC 0 `_OBJC_CLASS_$_WAURLQueryItem`
 - 38735: 38731 0x01a018b0 0x01a018b0 GLOBAL FUNC 0 `_OBJC_METACLASS_$_WAURLQueryItem`
 - 表示
 - 有global全局的类: ObjC普通类和元类: WAURLQueryItem的 s=symbol符号
- `SharedModules_rabin2_z_strings.txt`
 - 114696: 2074 0x01427163 0x01427163 14 15 12. __TEXT.__objc_classname ascii WAURLQueryItem
 - 表示: WAURLQueryItem是个Text节 (Section) 的ObjC的类名 `__objc_classname`
- `SharedModules_strings.txt`
 - 中的
 - 17522: B16@?0@"WAURLQueryItem"8
 - 112317: SaySo14WAURLQueryItemCG
 - 114721: WAURLQueryItem
 - 表示
 - 有类WAURLQueryItem相关的字符串 (类名) 和函数名等内容

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：
2022-03-17 20:39:28

参考资料

- 【已解决】class-dump导出砸壳后抖音ipa的头文件为空
- 【记录】用jtool查看抖音二进制信息
- 【记录】静态分析Mask的动态库：MaskPro.dylib
- 【记录】静态分析Mask的动态库：Mask.dylib
- 【记录】静态分析黑豹动态库zzzzHeiBaoLib.dylib
- 【记录】用radare2查看抖音二进制信息
- 【记录】用rabin2查看抖音AwemeCore二进制的信息
- 【已解决】Mac M2 Max中安装和使用jtool2
- 【规避解决】Mac M2 Max中jtool2运行崩溃：killed
- 【记录】用strings查看dylib库中包含的字符串
- 【记录】研究黑豹dylib文件zzzzHeiBaoLib.dylib中字符串和反越狱相关内容
- 【记录】研究佐罗dylib文件zorro.dylib中字符串和反越狱相关内容
- 【记录】用Cutter查看分析抖音AwemeCore二进制文件信息
- 【未解决】iOS逆向WhatsApp: -[WARegistrationURLBuilder verificationCodeRequestURLWithBaseURL:method:mcc:mnc:jailbroken:context:oldPhoneNumber:silentPushNotifRegCode:cellularStrength:]
- 【已解决】iOS逆向WhatsApp：寻找类WAURLQueryItem的定义
- 【已解决】nm输出的函数符号类型T、S、U、s的含义
- 【已解决】iOS逆向WhatsApp：静态分析WhatsApp二进制
- 【整理】Mac中2个版本的Idid：brew版和iOSOpenDev版
- 【已解决】修改iOS二进制权限工具：Idid
-
- iOS逆向开发
- iOS逆向开发：砸壳ipa
-
- OS Internals: (newosxbook.com)
- IOS逆向初探 · 浮萍's Blog (fuping.site)
- iOS逆向之Reveal、Hopper、MachOView等逆向工具的安装使用 - 简书 (jianshu.com)
- Mach-O Programming Topics Introduction
- IOS逆向初探 · 浮萍's Blog (fuping.site)
- iOS逆向之Reveal、Hopper、MachOView等逆向工具的安装使用 - 简书 (jianshu.com)
- pboke/Class-Decompile: Class Decompile is a python script for Hopper Disassembler. This script can export pseudo code of the classes. (github.com)
- 【OSG】jtool - Taking the O out of otool(1)-iOS安全-看雪论坛-安全社区|安全招聘|bbs.pediy.com
- JTool2 - Taking the O out of otool - squared (newosxbook.com)
- Program Sections - The Official Radare2 Book
- Strings - The Official Radare2 Book
- Libraries - The Official Radare2 Book
- Symbols - The Official Radare2 Book
- Exports - The Official Radare2 Book
- Imports - The Official Radare2 Book
- Entrypoint - The Official Radare2 Book

- [File Identification - The Official Radare2 Book](#)
- [Rabin2 - The Official Radare2 Book](#)
- [Introduction - The Official Radare2 Book](#)
- [git.saurik.com Git - ldid.git/summary](#)
- [xerub/ldid: Unofficial fork from saurik git repository git://git.saurik.com/ldid.git \(github.com\)](#)
-

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:
2023-10-22 20:48:09