目录

前言 1.1

Xcode+iOSOpenDev概览 1.2

app可调试 1.3

- 为何要 1.3.1
- 底层机制 1.3.2
- 如何实现 1.3.3
  - unc0ver越狱 1.3.3.1
  - XinaA15越狱 1.3.3.2
  - 重签名加权限 1.3.3.3
  - 用插件实现 1.3.3.4

调试方式 1.4

- Attach挂载 1.4.1
- Spawn孵化 1.4.2

效果举例 1.5

附录 1.6

- 查看进程PID 1.6.1
- 进程csflags定义 1.6.2
- 参考资料 1.6.3

iOS逆向调试：Xcode+iOSOpenDev

- 最新版本：`v1.0`
- 更新时间：`20240127`

简介

介绍iOS逆向的动态调试期间的，常用的好用的调试方式：Xcode+iOSOpenDev。主要包括先是概览；再介绍为何要实现可调式debuggable；以及对应底层机制；以及如何实现可调试，包括用unc0ver、XinaA15等工具越狱后自带可调试、用codesign加上权限重签名、用jailbreakd_client加上权限等，以及用插件XcodeRootDebug等；然后介绍具体的调试方式，包括Attach挂载模式和Spawn孵化模式；然后给出具体的调试的例子，包括Xcode汇编代码、函数调用堆栈、hook代码加断点等等效果；最后加上附录、包括如何查看进程PID。

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_debug_xcode_iosopendev: iOS逆向调试：Xcode+iOSOpenDev

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逆向调试：Xcode+iOSOpenDev book.crifan.org
- iOS逆向调试：Xcode+iOSOpenDev crifan.github.io

离线下载阅读

- iOS逆向调试：Xcode+iOSOpenDev PDF
- iOS逆向调试：Xcode+iOSOpenDev ePub
- iOS逆向调试：Xcode+iOSOpenDev Mobi

版权和用途说明

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

Xcode+iOSOpenDev概览

关于iOS逆向中的动态调试，之前已有完整的成套的教程：

iOS逆向开发：动态调试

而此处是单独介绍其中的，最常用的，最好用的部分：

用 `Xcode+iOSOpenDev` 实现GUI图形方式去调试

- 对比
  - 和**command line=命令行**方式的debugserver+lldb调试方式相比
    - **GUI=图形界面**方式的调试： `Xcode+iOSOpenDev`
      - 图形界面中，调试更加直观
        - Xcode中加函数断点
        - Xcode中给hook代码加断点
        - Xcode中查看和调试汇编代码
- 前提
  - （app/二进制文件等）调试目标必须是：**可调试=Debuggable**的

app可调试debuggable

- 什么是：app可调试debuggable
  - = 可调试debuggable
  - = （GUI的Xcode 和 命令行的debugserver）可以调试 = 有权限调试 任意的目标（app、二进制的 进程）
    - 包括
      - Xcode可以调试任意app、二进制、进程
      - debugserver(+lldb)可以调试任意app、二进制、进程

为何要可调试debuggable

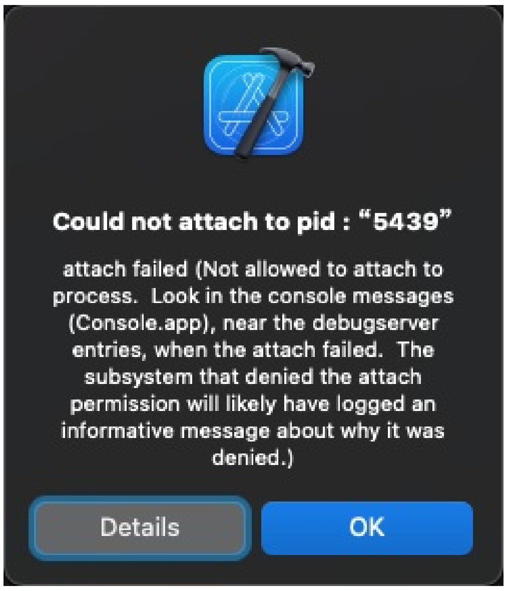
iOS逆向期间，尝试用Xcode去调试iPhone中的目标（app或二进制）时，如果目标不可调试，则会报错：

`Not allowed to attach to process`

所以被调试目标（app或二进制），必须：可调试= `debuggable`，才可以顺利调试。

报错举例

```
Could not attach to pid : “5439”

attach failed (Not allowed to attach to process.  Look in the console messages (Console
.app), near the debugserver entries, when the attach failed.  The subsystem that denied
 the attach permission will likely have logged an informative message about why it was
denied.)
```


Could not attach to pid : “5439”

attach failed (Not allowed to attach to process. Look in the console messages (Console.app), near the debugserver entries, when the attach failed. The subsystem that denied the attach permission will likely have logged an informative message about why it was denied.)

- Details

◦

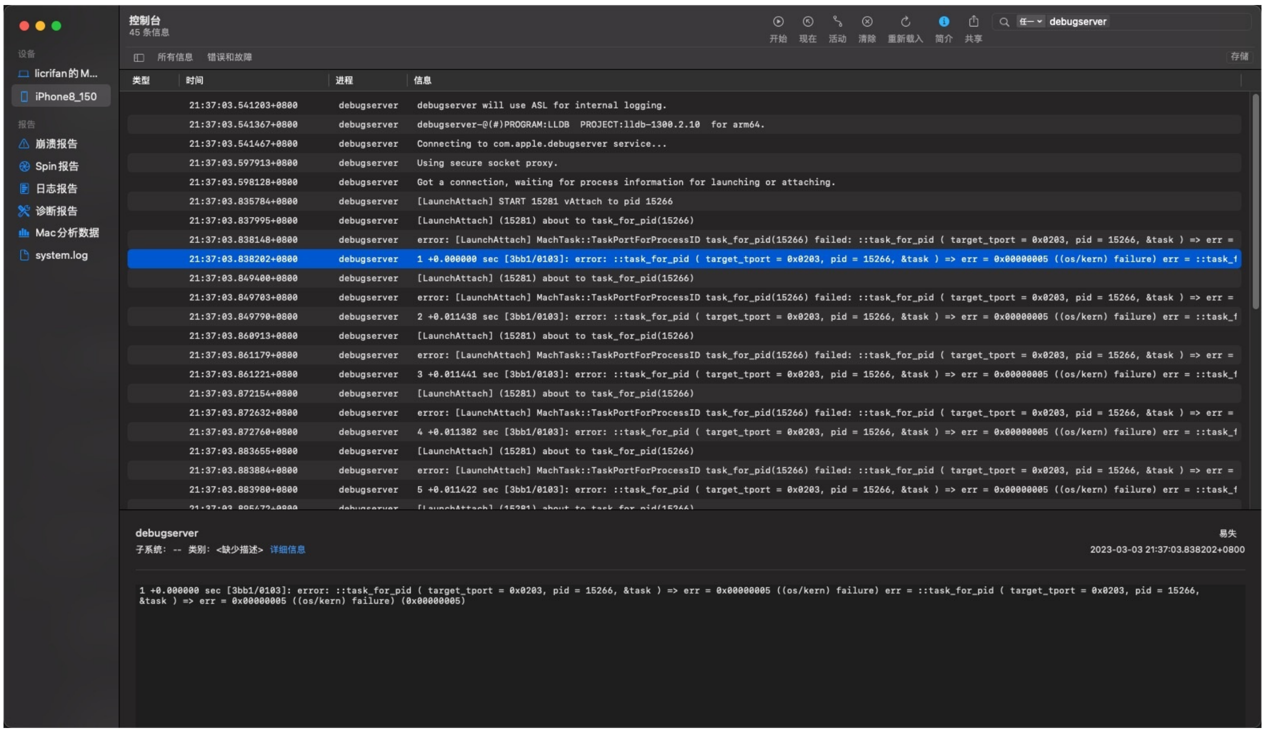
```
Details

Could not attach to pid : “5439”
Domain: IDEDebugSessionErrorDomain
Code: 3
Failure Reason: attach failed (Not allowed to attach to process.  Look in the console m
essages (Console.app), near the debugserver entries, when the attach failed.  The subsy
stem that denied the attach permission will likely have logged an informative message a
bout why it was denied.)
User Info: {
    DVTErrorCreationDateKey = "2023-03-02 09:53:45 +0000";
    DVTRadarComponentKey = 855031;
    IDERunOperationFailingWorker = DBGLLDBLauncher;
    RawUnderlyingErrorMessage = "attach failed (Not allowed to attach to process.  Look
 in the console messages (Console.app), near the debugserver entries, when the attach f
ailed.  The subsystem that denied the attach permission will likely have logged an info
rmative message about why it was denied.)";
}
--

Analytics Event: com.apple.dt.IDERunOperationWorkerFinished : {
    "device_model" = "iPhone10,1";
    "device_osBuild" = "15.0 (19A346)";
    "device_platform" = "com.apple.platform.iphoneos";
    "launchSession_schemeCommand" = Run;
    "launchSession_state" = 1;
    "launchSession_targetArch" = arm64;
    "operation_duration_ms" = 105483;
    "operation_errorCode" = 3;
    "operation_errorDomain" = IDEDebugSessionErrorDomain;
    "operation_errorWorker" = DBGLLDBLauncher;
    "operation_name" = IDEiPhoneRunOperationWorkerGroup;
    "param_consoleMode" = 0;
    "param_debugger_attachToExtensions" = 0;
    "param_debugger_attachToXPC" = 1;
    "param_debugger_type" = 3;
    "param_destination_isProxy" = 0;
    "param_destination_platform" = "com.apple.platform.iphoneos";
    "param_diag_MainThreadChecker_stopOnIssue" = 0;
    "param_diag_MallocStackLogging_enableDuringAttach" = 0;
    "param_diag_MallocStackLogging_enableForXPC" = 0;
    "param_diag_allowLocationSimulation" = 0;
    "param_diag_gpu_frameCapture_enable" = 3;
    "param_diag_gpu_shaderValidation_enable" = 0;
    "param_diag_gpu_validation_enable" = 1;
    "param_diag_memoryGraphOnResourceException" = 0;
    "param_diag_queueDebugging_enable" = 1;
    "param_diag_runtimeProfile_generate" = 0;
    "param_diag_sanitizer_asan_enable" = 0;
    "param_diag_sanitizer_tsan_enable" = 0;
    "param_diag_sanitizer_tsan_stopOnIssue" = 0;
    "param_diag_sanitizer_ubsan_stopOnIssue" = 0;
    "param_diag_showNonLocalizedStrings" = 0;
    "param_diag_viewDebugging_enabled" = 1;
    "param_diag_viewDebugging_insertDylibOnLaunch" = 1;
    "param_install_style" = 2;
    "param_launcher_UID" = 2;
    "param_launcher_allowDeviceSensorReplayData" = 0;
    "param_launcher_kind" = 0;
    "param_launcher_style" = 99;
    "param_launcher_substyle" = 256;
    "param_runnable_appExtensionHostRunMode" = 0;
    "param_runnable_type" = 0;
    "param_testing_launchedForTesting" = 0;
    "param_testing_suppressSimulatorApp" = 0;
    "param_testing_usingCLI" = 0;
    "sdk_canonicalName" = "iphoneos15.2";
    "sdk_osVersion" = "15.2";
    "sdk_variant" = iphoneos;
}
--


System Information

macOS Version 11.7.3 (Build 20G1116)
Xcode 13.2.1 (19586) (Build 13C100)
Timestamp: 2023-03-02T17:53:45+08:00
```

2024-01-27 12:02:15

可调试debuggable的底层机制和原理

`task_for_pid` 函数

根据前面的报错细节中的 `Look in the console messages (Console.app)` 所示，去查看 `Console.app` = `控制台` 中日志，往往可以看到相关错误细节：



```
21:37:03.541203+0800   debugserver   debugserver will use ASL for internal logging.
21:37:03.541367+0800   debugserver   debugserver-@(#)PROGRAM:LLDB  PROJECT:lldb-1300.2.10  for arm64.
21:37:03.541467+0800   debugserver   Connecting to com.apple.debugserver service...
21:37:03.597913+0800   debugserver   Using secure socket proxy.
21:37:03.598128+0800   debugserver   Got a connection, waiting for process information for launching or attaching.
21:37:03.835784+0800   debugserver   [LaunchAttach] START 15281 vAttach to pid 15266
21:37:03.837995+0800   debugserver   [LaunchAttach] (15281) about to task_for_pid(15266)
21:37:03.838148+0800   debugserver   error: [LaunchAttach] MachTask::TaskPortForProcessID task_for_pid(15266) failed: ::task_for_pid ( target_tport = 0x0203, pid = 15266, &task ) => err =
21:37:03.838202+0800   debugserver   1 +0.000000 sec [3bb1/0103]: error: ::task_for_pid ( target_tport = 0x0203, pid = 15266, &task ) => err = 0x00000005 ((os/kern) failure) err = ::task_f
21:37:03.849400+0800   debugserver   [LaunchAttach] (15281) about to task_for_pid(15266)
21:37:03.849703+0800   debugserver   error: [LaunchAttach] MachTask::TaskPortForProcessID task_for_pid(15266) failed: ::task_for_pid ( target_tport = 0x0203, pid = 15266, &task ) => err =
21:37:03.849790+0800   debugserver   2 +0.011438 sec [3bb1/0103]: error: ::task_for_pid ( target_tport = 0x0203, pid = 15266, &task ) => err = 0x00000005 ((os/kern) failure) err = ::task_f
21:37:03.860913+0800   debugserver   [LaunchAttach] (15281) about to task_for_pid(15266)
21:37:03.861179+0800   debugserver   error: [LaunchAttach] MachTask::TaskPortForProcessID task_for_pid(15266) failed: ::task_for_pid ( target_tport = 0x0203, pid = 15266, &task ) => err =
21:37:03.861221+0800   debugserver   3 +0.011441 sec [3bb1/0103]: error: ::task_for_pid ( target_tport = 0x0203, pid = 15266, &task ) => err = 0x00000005 ((os/kern) failure) err = ::task_f
21:37:03.872154+0800   debugserver   [LaunchAttach] (15281) about to task_for_pid(15266)
21:37:03.872632+0800   debugserver   error: [LaunchAttach] MachTask::TaskPortForProcessID task_for_pid(15266) failed: ::task_for_pid ( target_tport = 0x0203, pid = 15266, &task ) => err =
21:37:03.872760+0800   debugserver   4 +0.011382 sec [3bb1/0103]: error: ::task_for_pid ( target_tport = 0x0203, pid = 15266, &task ) => err = 0x00000005 ((os/kern) failure) err = ::task_f
21:37:03.883655+0800   debugserver   [LaunchAttach] (15281) about to task_for_pid(15266)
21:37:03.883884+0800   debugserver   error: [LaunchAttach] MachTask::TaskPortForProcessID task_for_pid(15266) failed: ::task_for_pid ( target_tport = 0x0203, pid = 15266, &task ) => err =
21:37:03.883980+0800   debugserver   5 +0.011422 sec [3bb1/0103]: error: ::task_for_pid ( target_tport = 0x0203, pid = 15266, &task ) => err = 0x00000005 ((os/kern) failure) err = ::task_f
21:37:03.885472+0800   debugserver   [LaunchAttach] (15281) about to task_for_pid(15266)
```

```
debugserver                                                                                易失
子系统： --  类别： <缺少描述>  详细信息                        2023-03-03 21:37:03.838202+0800

1 +0.000000 sec [3bb1/0103]: error: ::task_for_pid ( target_tport = 0x0203, pid = 15266, &task ) => err = 0x00000005 ((os/kern) failure) err = ::task_for_pid ( target_tport = 0x0203, pid = 15266,
&task ) => err = 0x00000005 ((os/kern) failure) (0x00000005)
```

```
默认	21:37:03.838148+0800	debugserver	error: [LaunchAttach] MachTask::TaskPortForProcessID task_for_pid(15266) failed: ::task_for_pid ( target_tport = 0x0203, pid = 15266, &task ) => err = 0x00000005 ((os/kern) failure)
默认	21:37:03.838202+0800	debugserver	1 +0.000000 sec [3bb1/0103]: error: ::task_for_pid ( target_tport = 0x0203, pid = 15266, &task ) => err = 0x00000005 ((os/kern) failure) err = ::task_for_pid ( target_tport = 0x0203, pid = 15266, &task ) => err = 0x00000005 ((os/kern) failure) (0x00000005)
```

核心错误

- debugserver
  - error: [LaunchAttach] MachTask::TaskPortForProcessID task_for_pid(15266) failed: ::task_for_pid (target_tport = 0x0203, pid = 15266, &task) => err = 0x00000005 ((os/kern) failure)

即：

- 核心原因
  - `task_for_pid()` 返回失败 -》无法获取task
- 解决办法
  - 能想办法给被调试的app加上 `task_for_pid-allow` （和 `get-task-allow` ）的权限entitlement

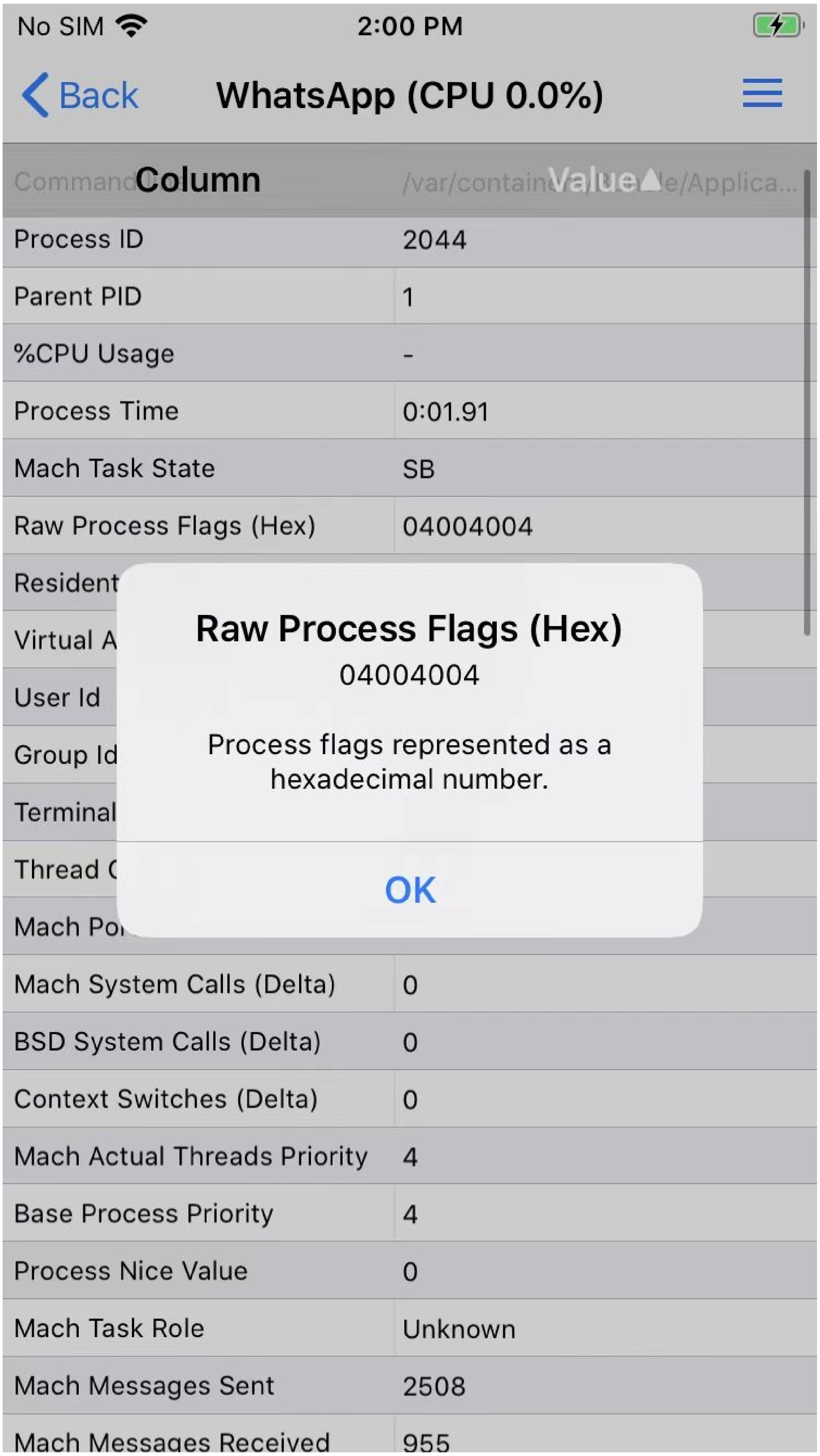
即可

        - 但是无法直接修改 `debugserver` 去加权限
            - 因为这个 `/Developer/usr/bin/debugserver`
                - 最早来源是： `DeveloperDiskImage.dmg`
            - 是挂载到 `Ramdisk` 中 -> 是只读的
                - 所以无法简单的直接修改
        - 所以只能用其他办法
            - 详见后续内容
                - 如何实现

`csflags` 中的 `CS_GET_TASK_ALLOW`

而决定进程函数 `task_for_pid()` 返回是否成功，再底层决定因素是：进程的 `csflags` 中的 `CS_GET_TASK_ALLOW`

- 举例
    - `WhatsApp` 的进程的flag
        -



Column	Value
Command	/var/contain…le/Applica...
Process ID	2044
Parent PID	1
%CPU Usage	-
Process Time	0:01.91
Mach Task State	SB
Raw Process Flags (Hex)	04004004
Resident	
Virtual A	
User Id	
Group Id	
Terminal	
Thread C	
Mach Po	
Mach System Calls (Delta)	0
BSD System Calls (Delta)	0
Context Switches (Delta)	0
Mach Actual Threads Priority	4
Base Process Priority	4
Process Nice Value	0
Mach Task Role	Unknown
Mach Messages Sent	2508
Mach Messages Received	955

Raw Process Flags (Hex)
04004004

Process flags represented as a hexadecimal number.

OK

        - 含义解释
            - `0x04000000 = CS_PLATFORM_BINARY`
            - `0x00004000 = CS_ENTITLEMENTS_VALIDATED`
            - `0x00000004 = CS_GET_TASK_ALLOW`
        - 具体定义详见
            - 进程csflags定义
- 说明
    - 不过，目前也发现过：iOS进程中csflags中有CS_GET_TASK_ALLOW也还是无法被调试
    - 所以底层相关细节，暂时不是完全清楚，待深究。

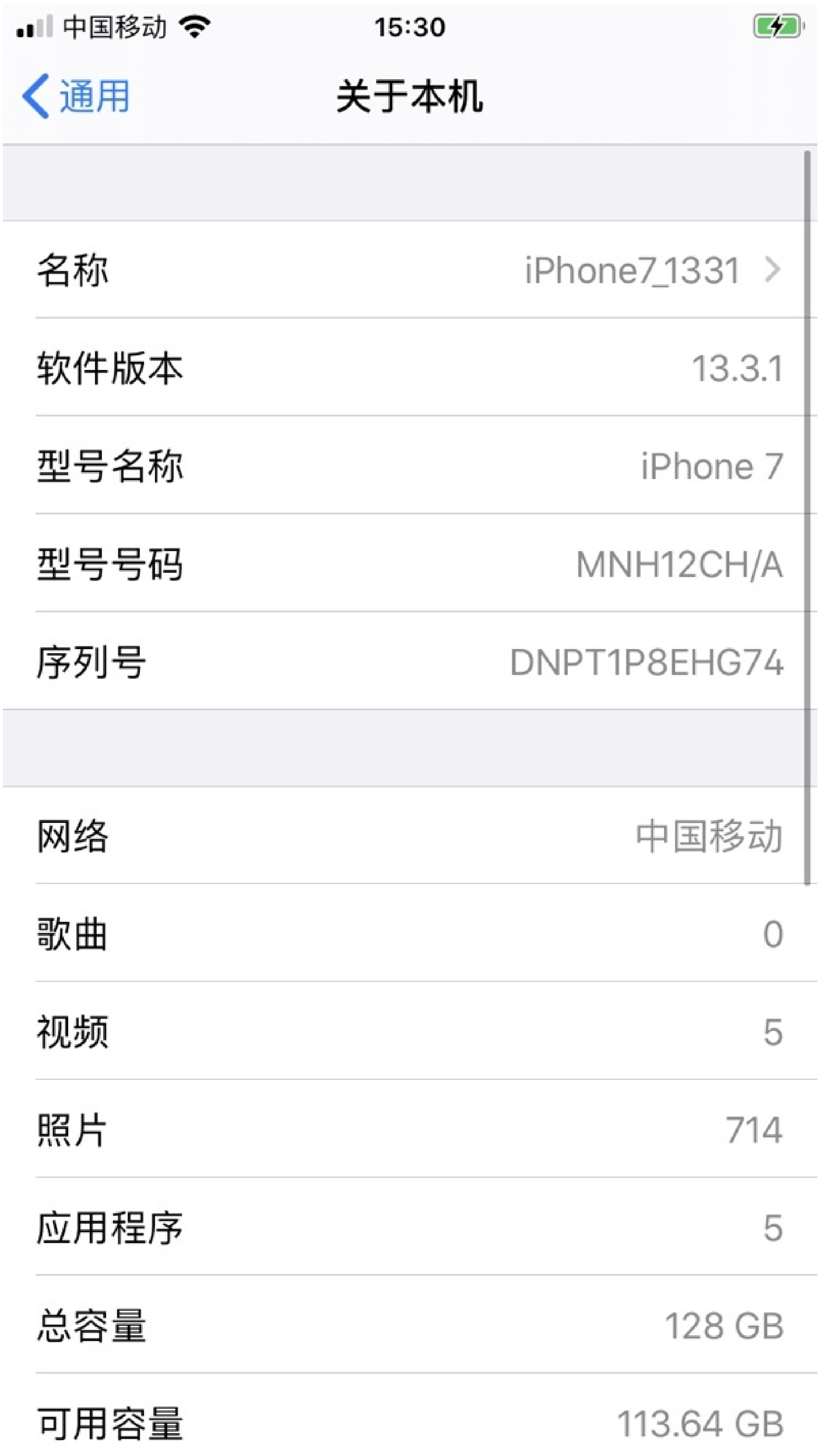
如何实现可调试debuggable

- 如何实现app可调试Debuggable
    - 概述
        - 全局的
            - 越狱工具越狱后，系统全局自动支持app可调式
                - 部分机型用unc0ver越狱后，自带：app可调式
                - A12+的机型，用XinaA15越狱后，自带：app可调式
            - 借助于插件XcodeRootDebug实现
        - 针对特定app的
            - （用codesign）给app二进制文件重新签名，加上可调试权限

下面详细解释：

用unc0ver越狱自动实现可调试

- 部分机型用unc0ver越狱后的自带app可调试
    - 说明
        - 实测部分机型
            - `iOS 13.3.1` 的 `iPhone7`

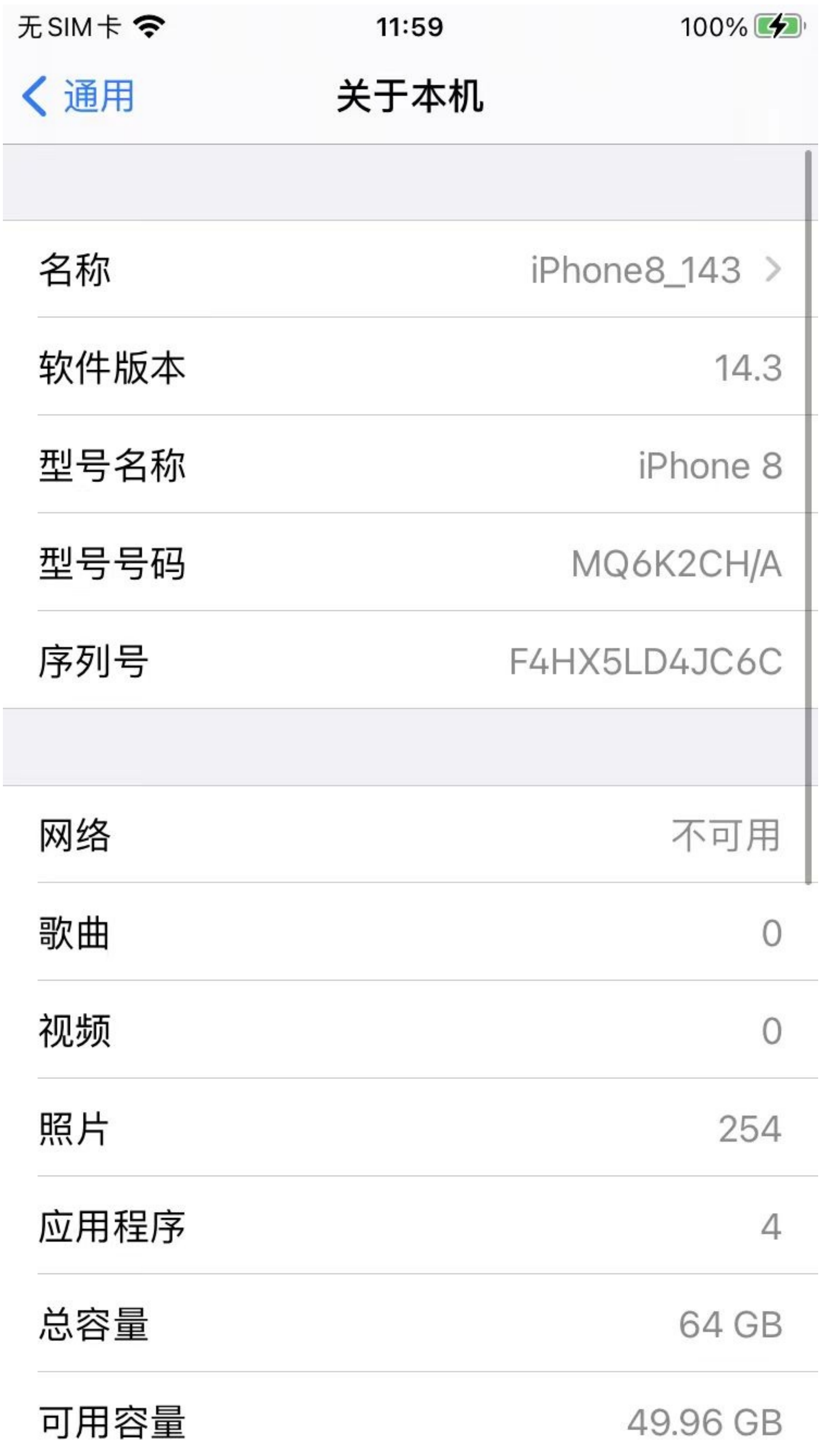


中国移动 15:30

通用 关于本机

名称	iPhone7_1331 >
软件版本	13.3.1
型号名称	iPhone 7
型号号码	MNH12CH/A
序列号	DNPT1P8EHG74
网络	中国移动
歌曲	0
视频	5
照片	714
应用程序	5
总容量	128 GB
可用容量	113.64 GB

- `iOS 14.3 的 iPhone8`

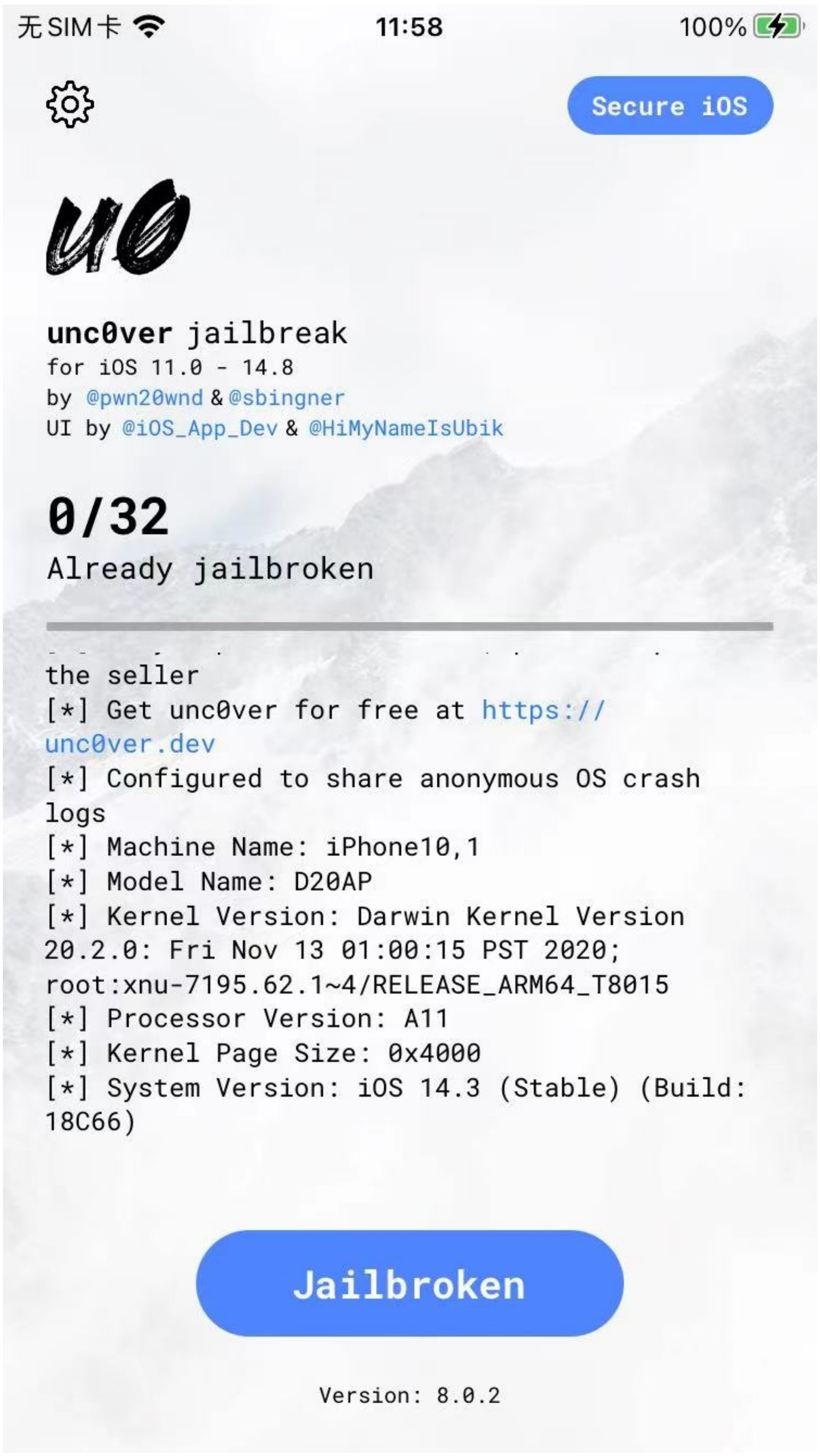
- 

无SIM卡 11:59 100%

通用 关于本机

名称	iPhone8_143
软件版本	14.3
型号名称	iPhone 8
型号号码	MQ6K2CH/A
序列号	F4HX5LD4JC6C
网络	不可用
歌曲	0
视频	0
照片	254
应用程序	4
总容量	64 GB
可用容量	49.96 GB

- 在用 `unc0ver`
  - 最新版 `v8.0.2`



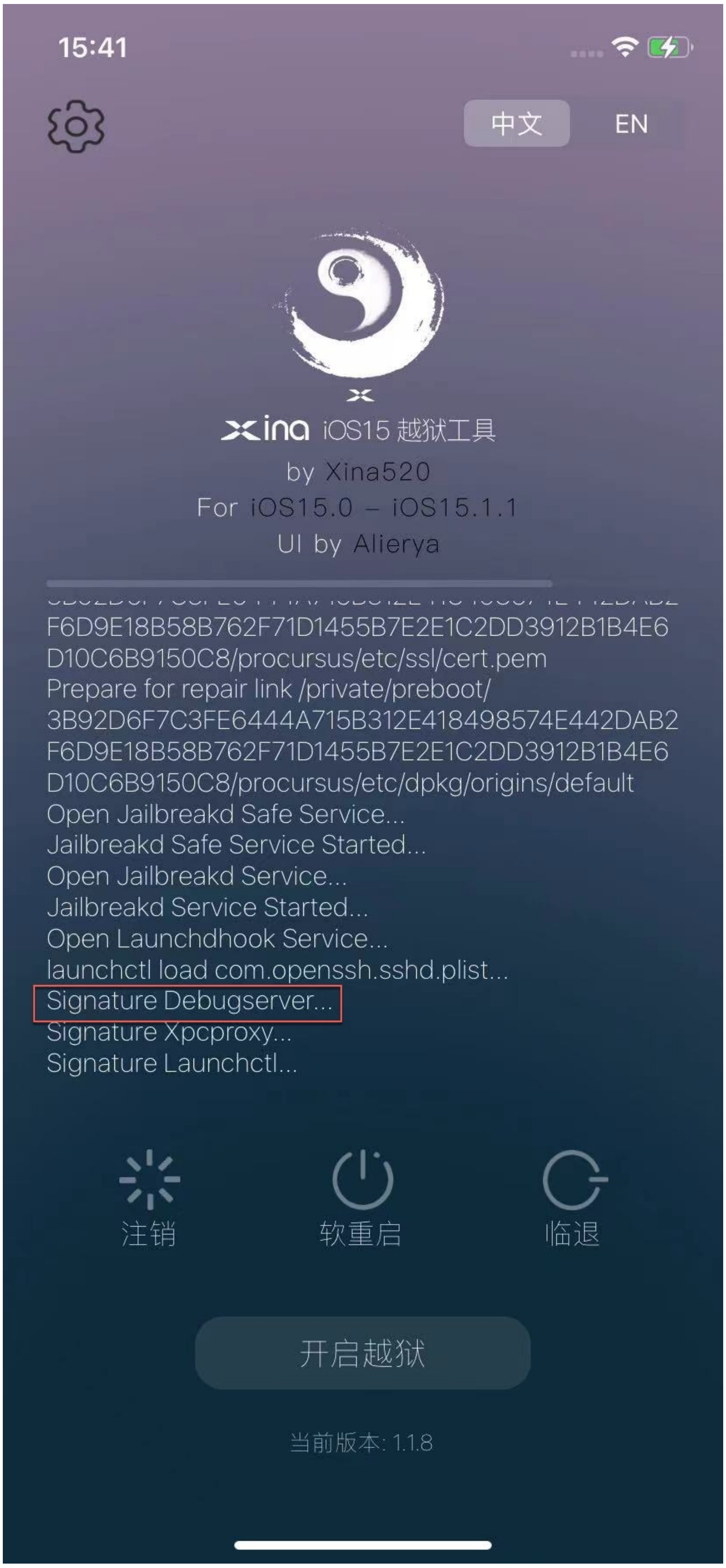
```
unc0ver jailbreak
for iOS 11.0 - 14.8
by @pwn20wnd & @sbingner
UI by @iOS_App_Dev & @HiMyNameIsUbik

0/32
Already jailbroken

the seller
[*] Get unc0ver for free at https://
unc0ver.dev
[*] Configured to share anonymous OS crash
logs
[*] Machine Name: iPhone10,1
[*] Model Name: D20AP
[*] Kernel Version: Darwin Kernel Version
20.2.0: Fri Nov 13 01:00:15 PST 2020;
root:xnu-7195.62.1~4/RELEASE_ARM64_T8015
[*] Processor Version: A11
[*] Kernel Page Size: 0x4000
[*] System Version: iOS 14.3 (Stable) (Build:
18C66)
```

- 越狱后，确认是自动拥有全局的 `app可调试`
- 优点
  - 无需额外操作，即可支持可调试

- 缺点
  - 之前unc0ver越狱的iPhone，好像不是自动支持可调试?

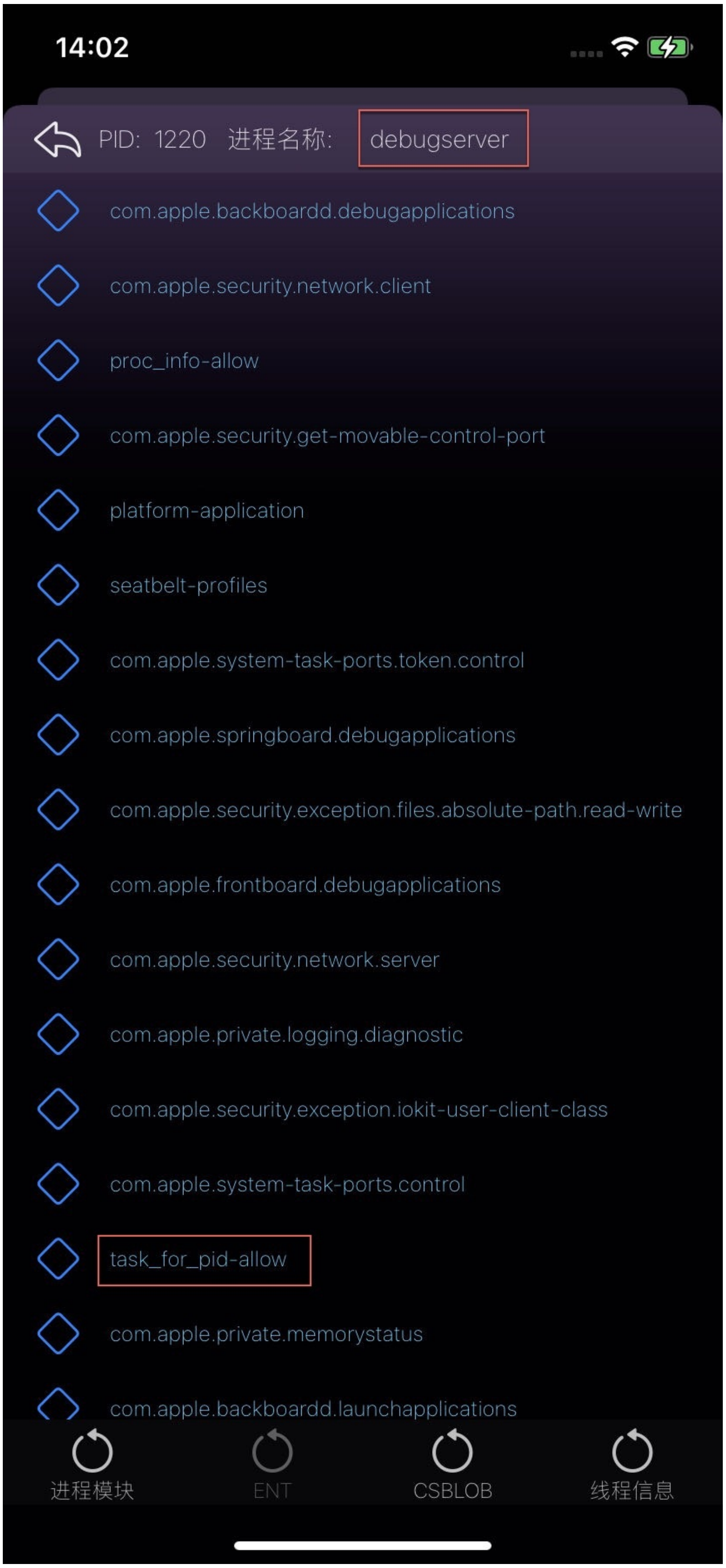
用XinaA15越狱自动实现可调试

- XinaA15越狱后：自带app可调试
    - 说明
        - 关键点
            - 越狱期间
                - `Signature Debugserver` = `重签名Debugserver`
                    -



```
F6D9E18B58B762F71D1455B7E2E1C2DD3912B1B4E6
D10C6B9150C8/procursus/etc/ssl/cert.pem
Prepare for repair link /private/preboot/
3B92D6F7C3FE6444A715B312E418498574E442DAB2
F6D9E18B58B762F71D1455B7E2E1C2DD3912B1B4E6
D10C6B9150C8/procursus/etc/dpkg/origins/default
Open Jailbreakd Safe Service...
Jailbreakd Safe Service Started...
Open Jailbreakd Service...
Jailbreakd Service Started...
Open Launchdhook Service...
launchctl load com.openssh.sshd.plist...
Signature Debugserver...
Signature Xpcproxy...
Signature Launchctl...
```

- 越狱后
    - 进程管理器中的 `debugserver` 的entitlement权限中，就有了： `task_for_pid-allow`
    -



14:02

PID: 1220 进程名称: debugserver

com.apple.backboardd.debugapplications
com.apple.security.network.client
proc_info-allow
com.apple.security.get-movable-control-port
platform-application
seatbelt-profiles
com.apple.system-task-ports.token.control
com.apple.springboard.debugapplications
com.apple.security.exception.files.absolute-path.read-write
com.apple.frontboard.debugapplications
com.apple.security.network.server
com.apple.private.logging.diagnostic
com.apple.security.exception.iokit-user-client-class
com.apple.system-task-ports.control
task_for_pid-allow
com.apple.private.memorystatus
com.apple.backboardd.launchapplications

        - -> 给debugserver加上可调试权限 -》 可以调试任意app/进程了
  - 缺点
    - 只适用于：A12+芯片的iPhone机型
      - 注：XinaA15只支持 `A12+` 的机型

重签名添加权限实现可调试

- 重签名添加权限实现可调试
    - 概述
        - 方式1：用codesign重新签名，加上可调试的entitlement权限
        - 方式2：用 `jailbreakd_client` 加可调试的权限

详解：

用codesign重签名加可调试权限

- 用codesign重新签名，加上可调试的entitlement权限
    - 优点
        - 此方式相对比较通用，适用于各种机型和系统
    - 缺点
        - 需要熟悉codesign签名和entitlement权限等细节
    - 举例
        - 给akd重签名，使其可调试
            - 从iPhone中导出akd

              ```
              scp root@192.168.1.22:/System/Library/PrivateFrameworks/AuthKit.framewor
              k/akd akd_origin
              ```

            - 导出adk的entitlement权限

              ```
              ldid -e akd_origin > akd_entitlements.xml
              ```

            - 修改entitlement，增加可调式相关权限
                - 修改 `akd_entitlements.xml`
                    - 加上权限
                        - `get-task-allow` =true
                        - `task_for_pid-allow` =true
                        - `run-unsigned-code` =true
                        - ->

                          ```
                              <key>get-task-allow</key>
                              <true/>
                              <key>task_for_pid-allow</key>
                              <true/>
                              <key>run-unsigned-code</key>
                              <true/>
                          ```

                    - 去掉权限
                        - seatbelt-profiles

                          ```
                              <key>seatbelt-profiles</key>
                              <array>
                                  <string>akd</string>
                          ```

```
    </array>
```

                - com.apple.security.network.client

```
    <key>com.apple.security.network.client</key>
    <true/>
```

            - 保存为 `akd_entitlements_debuggable.xml`
        - 用新的entitlement去重新签名

```
cp akd_origin akd_debuggable
codesign -f -s - --entitlements akd_entitlements_debuggable.xml akd_debu
ggable
```

        - 再把重签名后的akd放回越狱iPhone中

```
scp akd_debuggable root@192.168.1.22:/System/Library/PrivateFrameworks/A
uthKit.framework/akd
```

    - 详见
        - 手动重签名 · iOS逆向开发：签名和权限 (crifan.org)

用 `jailbreakd_client` 加可调试的权限

- 用 `jailbreakd_client` 加可调试的权限
    - 说明
        - `jailbreakd_client` 是部分越狱系统才有的工具，好像是
            - 用的coolstar系越狱（Electra或者Chimera）后，有对应文件

```
xia0:/chimera root# ls -la
total 1100
drwxr-xr-x  8 root wheel    256 Feb 26 13:19 ./
drwxr-xr-x 28 root wheel    896 Sep 16 17:44 ../
-rwxr-xr-x  1 root wheel 168736 Sep 17 10:21 inject_criticald*
-rwxr-xr-x  1 root wheel 207920 Sep 17 10:21 jailbreakd*
-rwxr-xr-x  1 root wheel 133840 Sep 17 10:21 jailbreakd_client*
-rwxr-xr-x  1 root wheel 167296 Sep 17 10:21 libjailbreak.dylib*
...
```

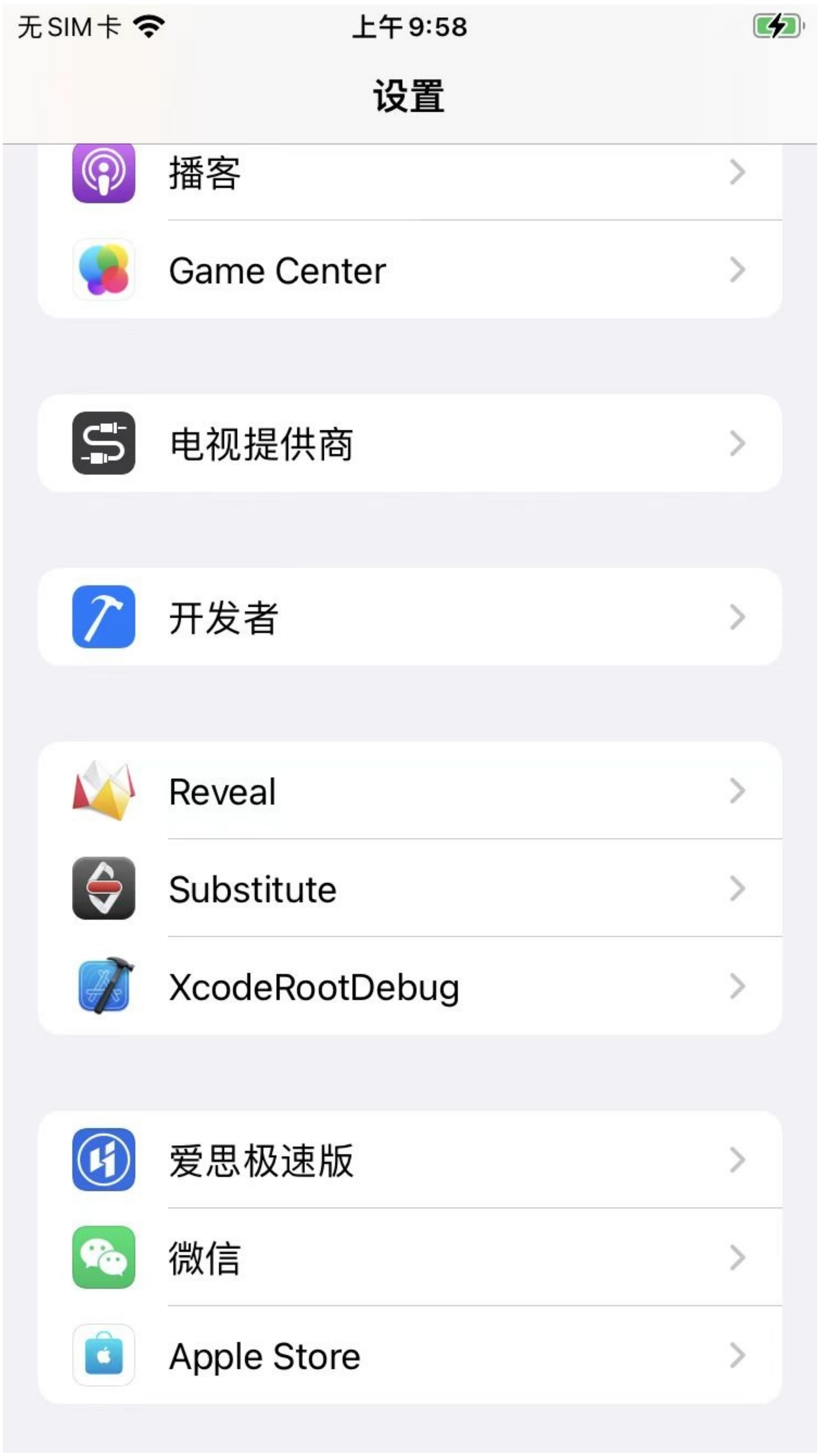
        - 基本用法

```
/electra/jailbreakd_client <PID> 1
```

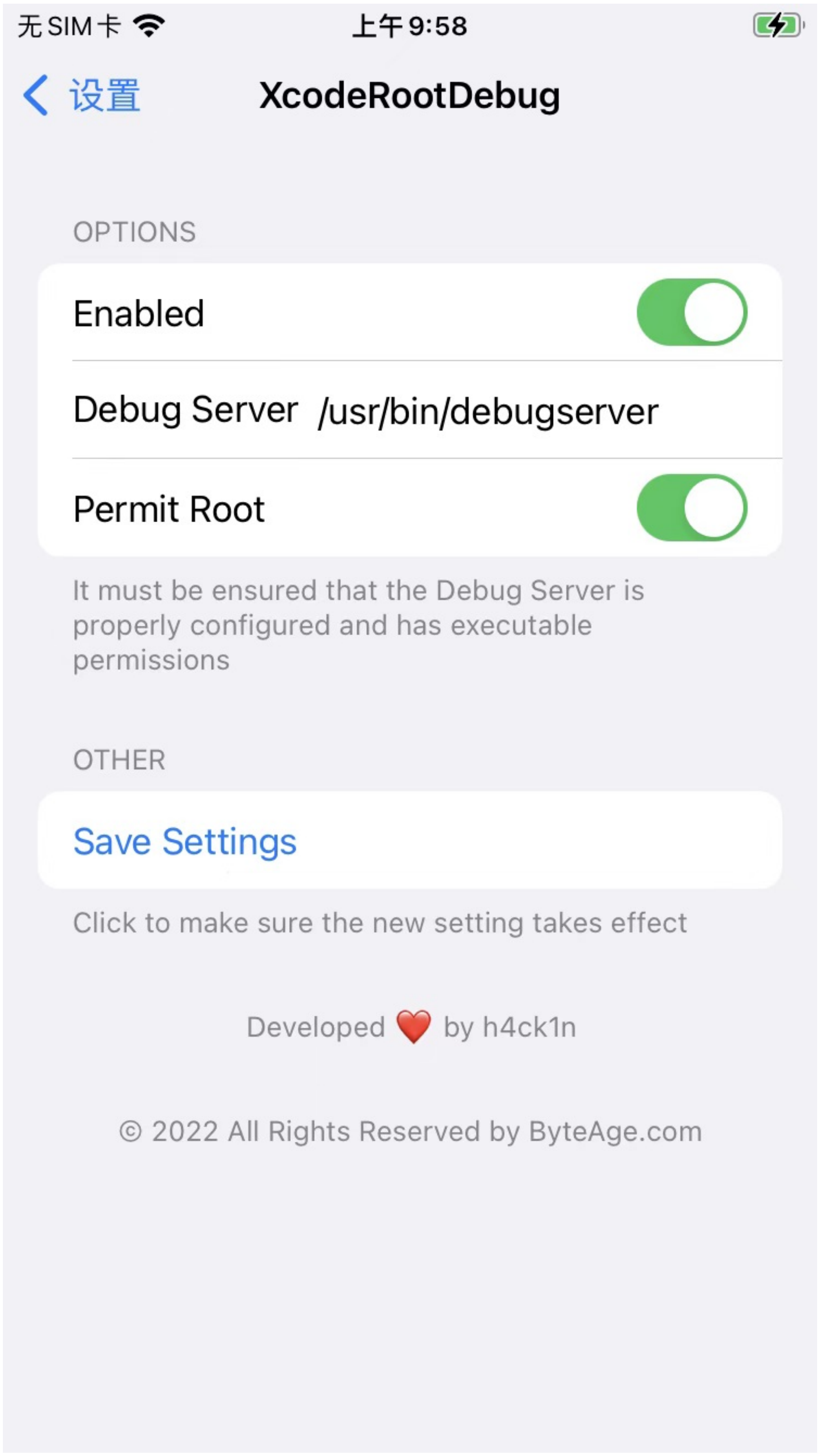
    - 暂未成功使用，之前的折腾详见
        - 【未解决】iOS逆向：寻找可用的jailbreakd_client用于给进程加可调试权限
        - 【未解决】iOS逆向：用jailbreakd_client给debugserver去加上entitle和platformize

借助插件实现可调式

- 借助插件实现可调式
    - XcodeRootDebug
        - Github
            - https://github.com/lemon4ex/XcodeRootDebug
        - Repo源
            - https://repo.byteage.com
        - 说明
            - 之前在iOS15中用过：不起效果
                - 不过对于 `< iOS 15` ，应该是可以工作的
        - 前提
            - 把重签名加了可调试权限的debugserver放到iPhone中的对应位置
                - 位置举例
                    - `/usr/bin/debugserver`
                - 关于如何重签名debugserver，详见
                    - 确保debugserver权限 · iOS逆向调试：debugserver+lldb (crifan.org)
        - 安装后
            - `设置` 中能看到 `XcodeRootDebug`

- 

- 进入后可以配置XcodeRootDebug的参数

- 

无SIM卡 上午9:58

设置 XcodeRootDebug

OPTIONS

Enabled: on

Debug Server /usr/bin/debugserver

Permit Root: on

It must be ensured that the Debug Server is properly configured and has executable permissions

OTHER

Save Settings

Click to make sure the new setting takes effect

Developed ❤️ by h4ck1n

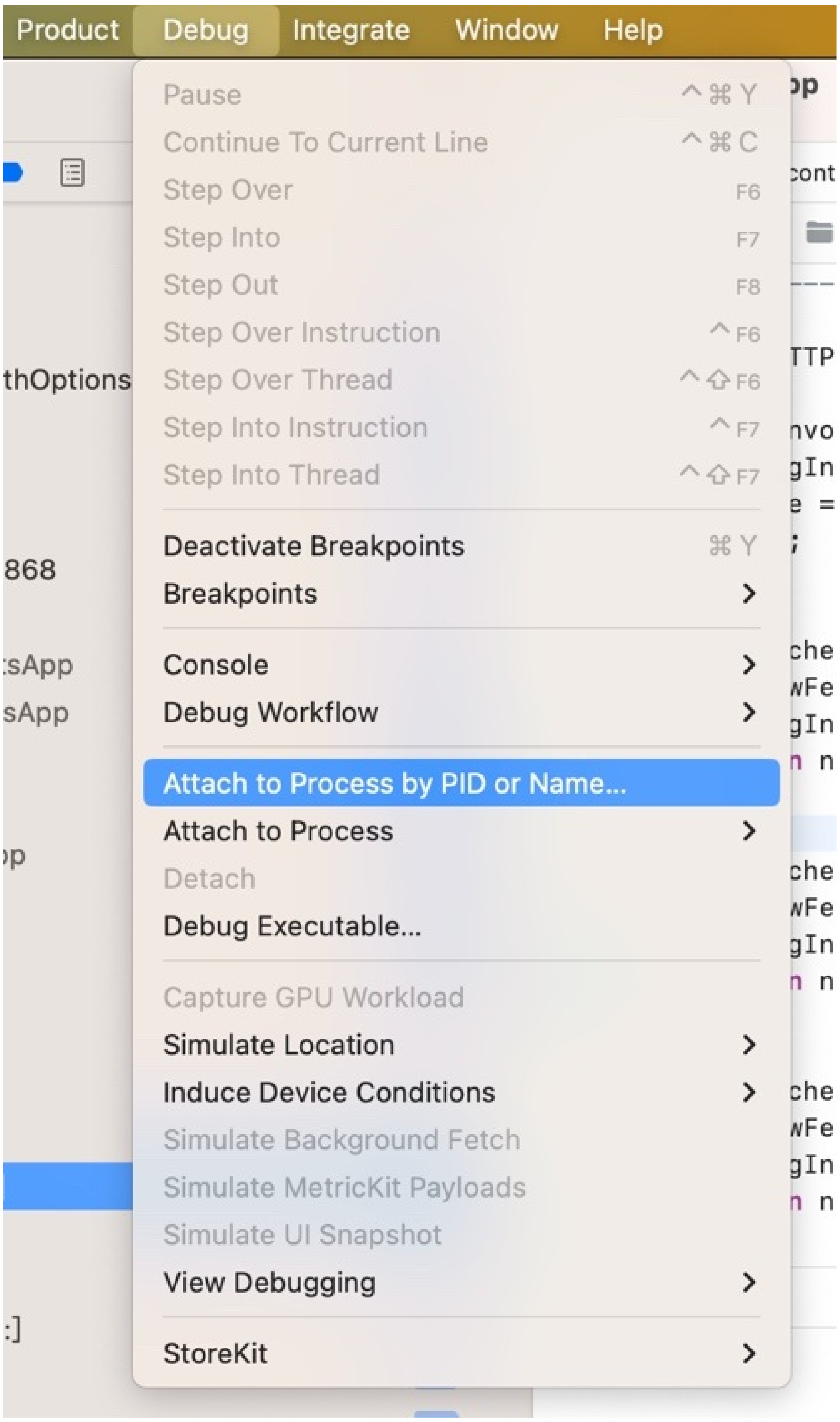
© 2022 All Rights Reserved by ByteAge.com

调试方式

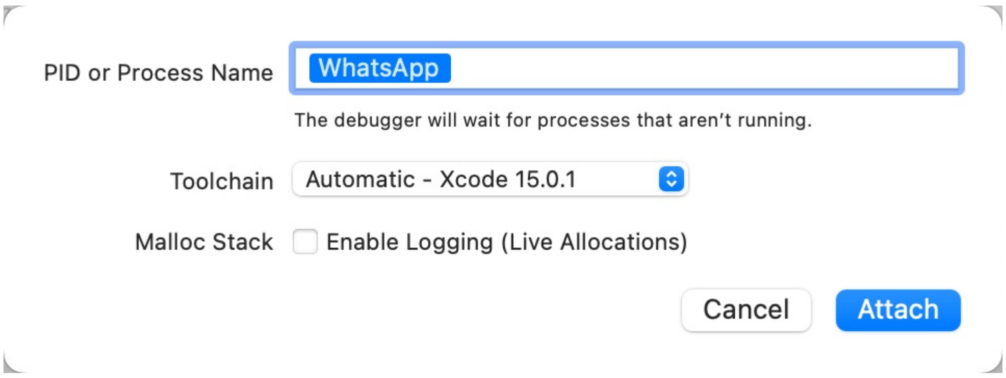
- 调试方式
    - 概述
        - Attach模式
            - 进程已运行：Xcode中Attach时输入Name或PID，即可调试
        - Spawn模式
            - 进程未运行：Xcode中Attach时输入Name，再去启动（app或二进制文件对应的）进程，即可调试

以Attach挂载模式启动调试

- Attach模式=挂载模式
    - 前提：已经启动=app正在运行
    - 方式1：手动输入PID或Name
        - 步骤： `Xcode` -> `Debug` -> `Attach to Process by PID or name` ->输入 `PID` 或 `name`
            - 图



- 方式1：Name
    - 举例： `WhatsApp`

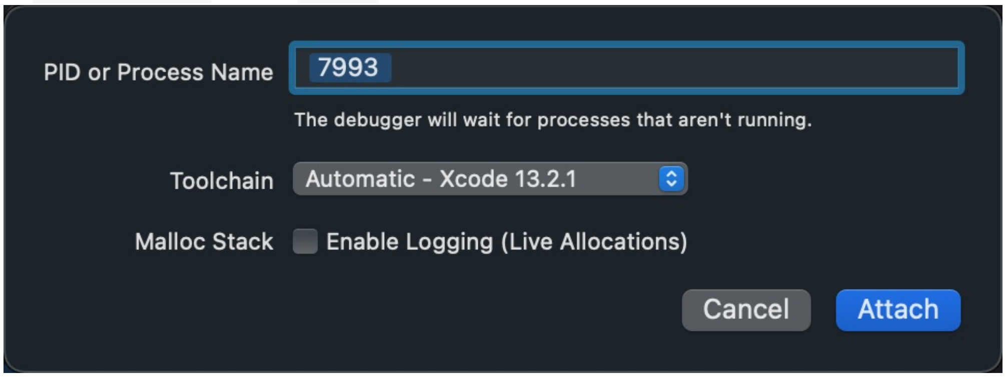
- 

PID or Process Name: WhatsApp

The debugger will wait for processes that aren't running.

Toolchain: Automatic - Xcode 15.0.1

Malloc Stack: Enable Logging (Live Allocations)

- 方式2：PID
  - 举例： `Preferences` 的PID= `7993`
    - 

PID or Process Name: 7993

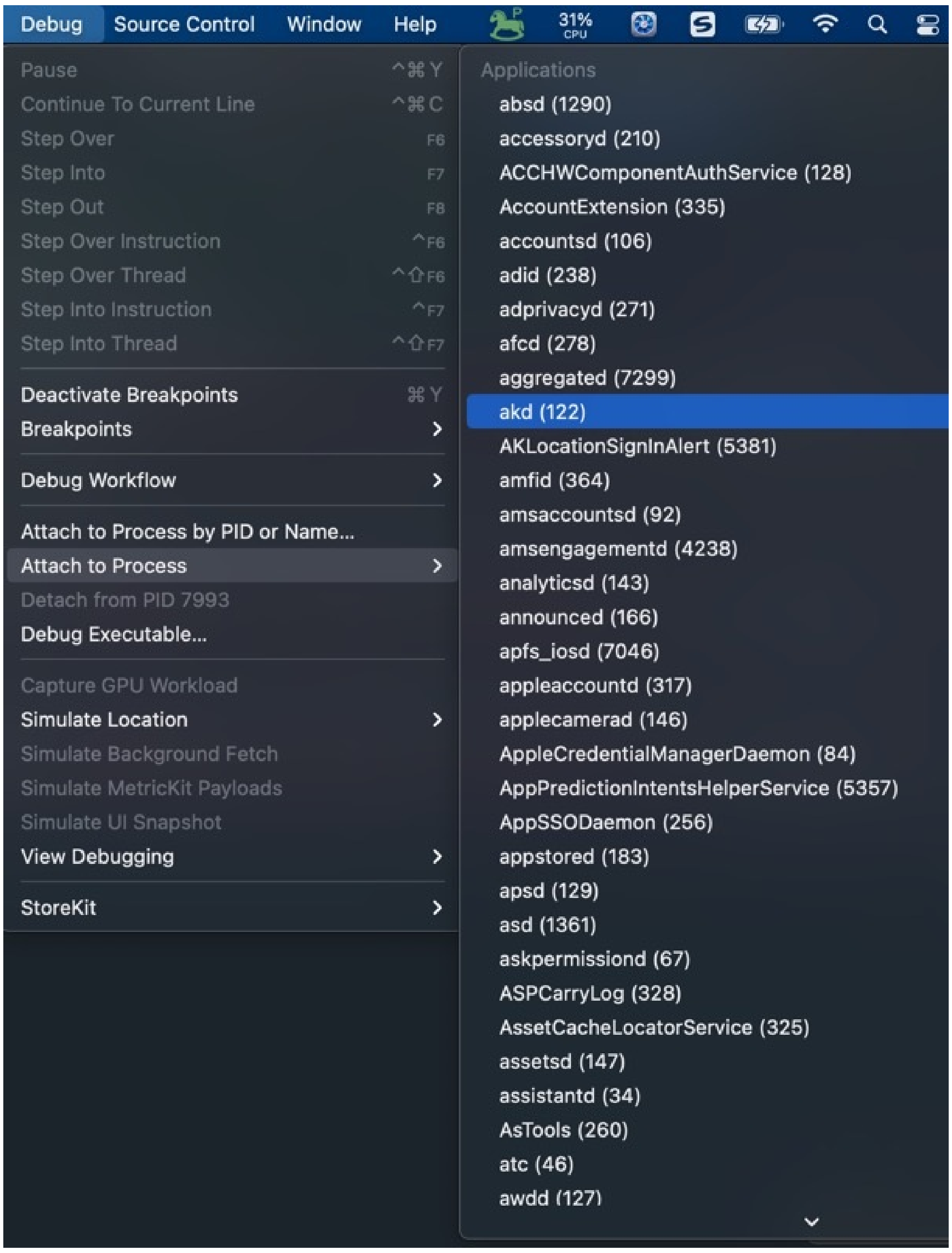
The debugger will wait for processes that aren't running.

Toolchain: Automatic - Xcode 13.2.1

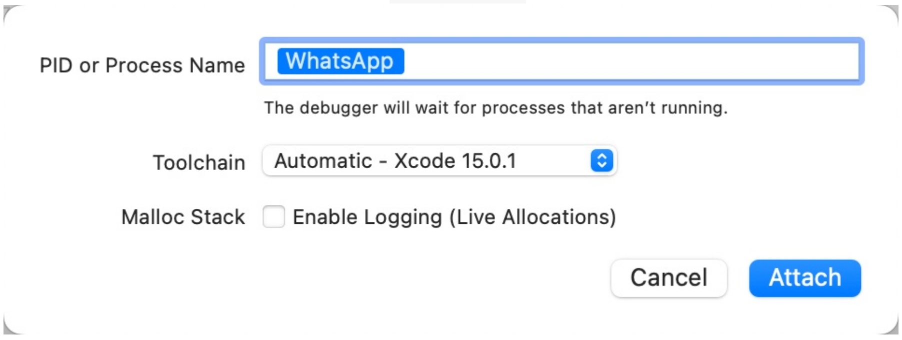
Malloc Stack: Enable Logging (Live Allocations)

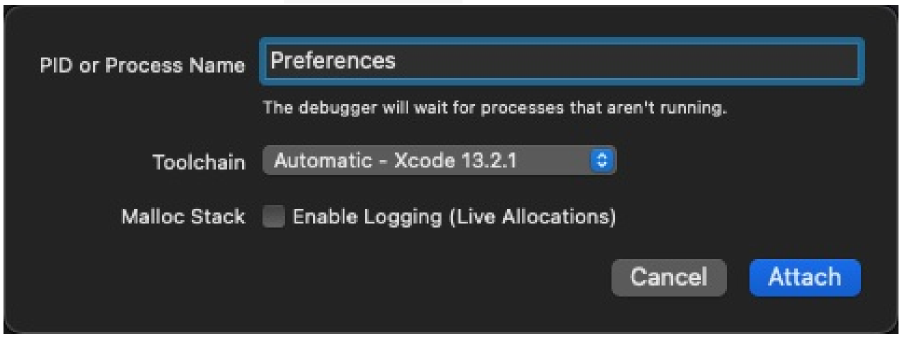
  - 说明
    - 关于如何查看进程PID，详见：查看进程PID

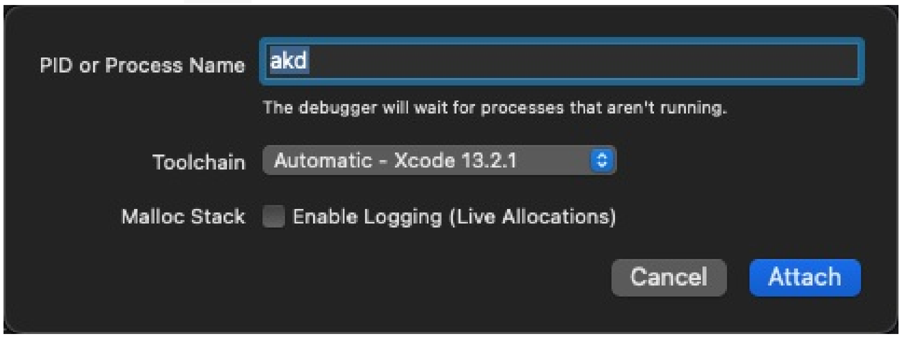
◦ 方式2：从进程列表中点击选择
  - 步骤： `Xcode` -> `Debug` -> `Attach to Process` ->等待（一会，即可）显示出进程的列表-》点击选择对应进程
  - 举例
    - akd

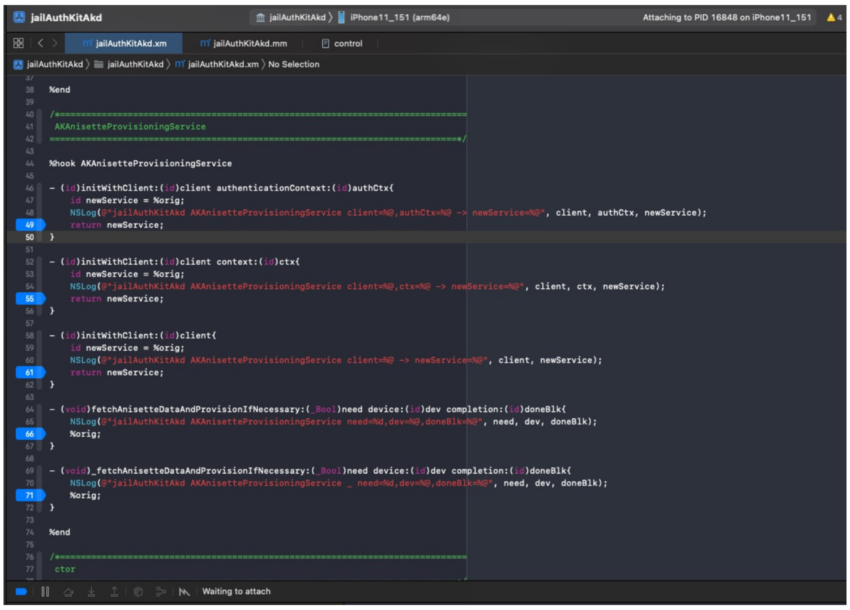
- 
Debug
Source Control
Window
Help
31% CPU
Pause ^⌘Y
Continue To Current Line ^⌘C
Step Over F6
Step Into F7
Step Out F8
Step Over Instruction ^F6
Step Over Thread ^⇧F6
Step Into Instruction ^F7
Step Into Thread ^⇧F7
Deactivate Breakpoints ⌘Y
Breakpoints
Debug Workflow
Attach to Process by PID or Name...
Attach to Process
Detach from PID 7993
Debug Executable...
Capture GPU Workload
Simulate Location
Simulate Background Fetch
Simulate MetricKit Payloads
Simulate UI Snapshot
View Debugging
StoreKit
Applications
absd (1290)
accessoryd (210)
ACCHWComponentAuthService (128)
AccountExtension (335)
accountsd (106)
adid (238)
adprivacyd (271)
afcd (278)
aggregated (7299)
akd (122)
AKLocationSignInAlert (5381)
amfid (364)
amsaccountsd (92)
amsengagementd (4238)
analyticsd (143)
announced (166)
apfs_iosd (7046)
appleaccountd (317)
applecamerad (146)
AppleCredentialManagerDaemon (84)
AppPredictionIntentsHelperService (5357)
AppSSODaemon (256)
appstored (183)
apsd (129)
asd (1361)
askpermissiond (67)
ASPCarryLog (328)
AssetCacheLocatorService (325)
assetsd (147)
assistantd (34)
AsTools (260)
atc (46)
awdd (127)

以Spawn孵化模式启动调试

- Spawn模式=孵化模式
  - 用途：
    - 用于app启动阶段时的相关逻辑
      - 举例
        - 调试反越狱检测
        - 调试app首次启动初始化相关过程
          - 比如app中只初始化一次的相关代码逻辑
            - 举例
              - WhatsApp中的初始化创建WAUUID、WAFBUUID等相关逻辑
  - 前提：Xcode中可以通过name找到被调试的app或二进制
  - 步骤：
    - Xcode中
      - 操作： `Xcode` -> `Debug` -> `Attach to Process by PID or name` ->输入 `name`
        - 说明：Name是app或二进制文件的name
        - 举例
          - WhatsApp这个app的主入口二进制： `WhatsApp`
            - 
              PID or Process Name: WhatsApp
              The debugger will wait for processes that aren't running.
              Toolchain: Automatic - Xcode 15.0.1
              Malloc Stack: Enable Logging (Live Allocations)

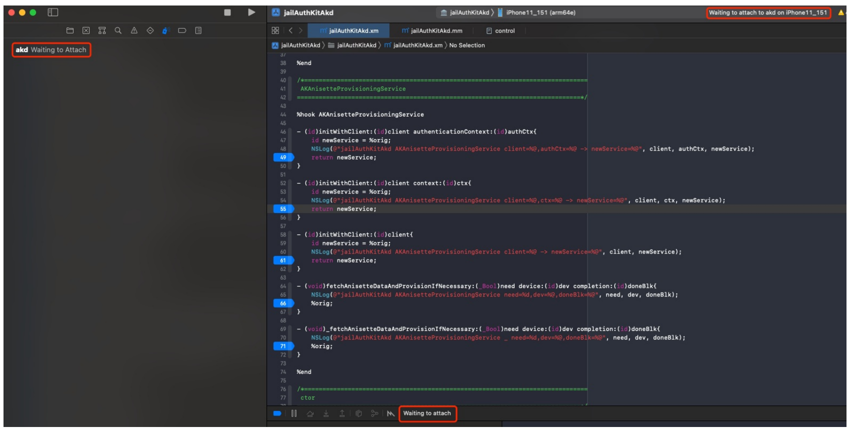
          - Preferences的二进制名称： `Preferences`
            - 
              PID or Process Name: Preferences
              The debugger will wait for processes that aren't running.
              Toolchain: Automatic - Xcode 13.2.1
              Malloc Stack: Enable Logging (Live Allocations)

          - akd的二进制名称： `akd`
            - 
              PID or Process Name: akd
              The debugger will wait for processes that aren't running.
              Toolchain: Automatic - Xcode 13.2.1
              Malloc Stack: Enable Logging (Live Allocations)

- Xcode中会显示：正在等待挂载
  - 对于PID：`Attaching to PID xxx`
    - 举例
      - PID=16848
        - 

jailAuthKitAkd
jailAuthKitAkd 〉 iPhone11_151 (arm64e)
Attaching to PID 16848 on iPhone11_151
jailAuthKitAkd.xm
jailAuthKitAkd.mm
control
jailAuthKitAkd 〉 jailAuthKitAkd 〉 jailAuthKitAkd.xm 〉 No Selection
Waiting to attach

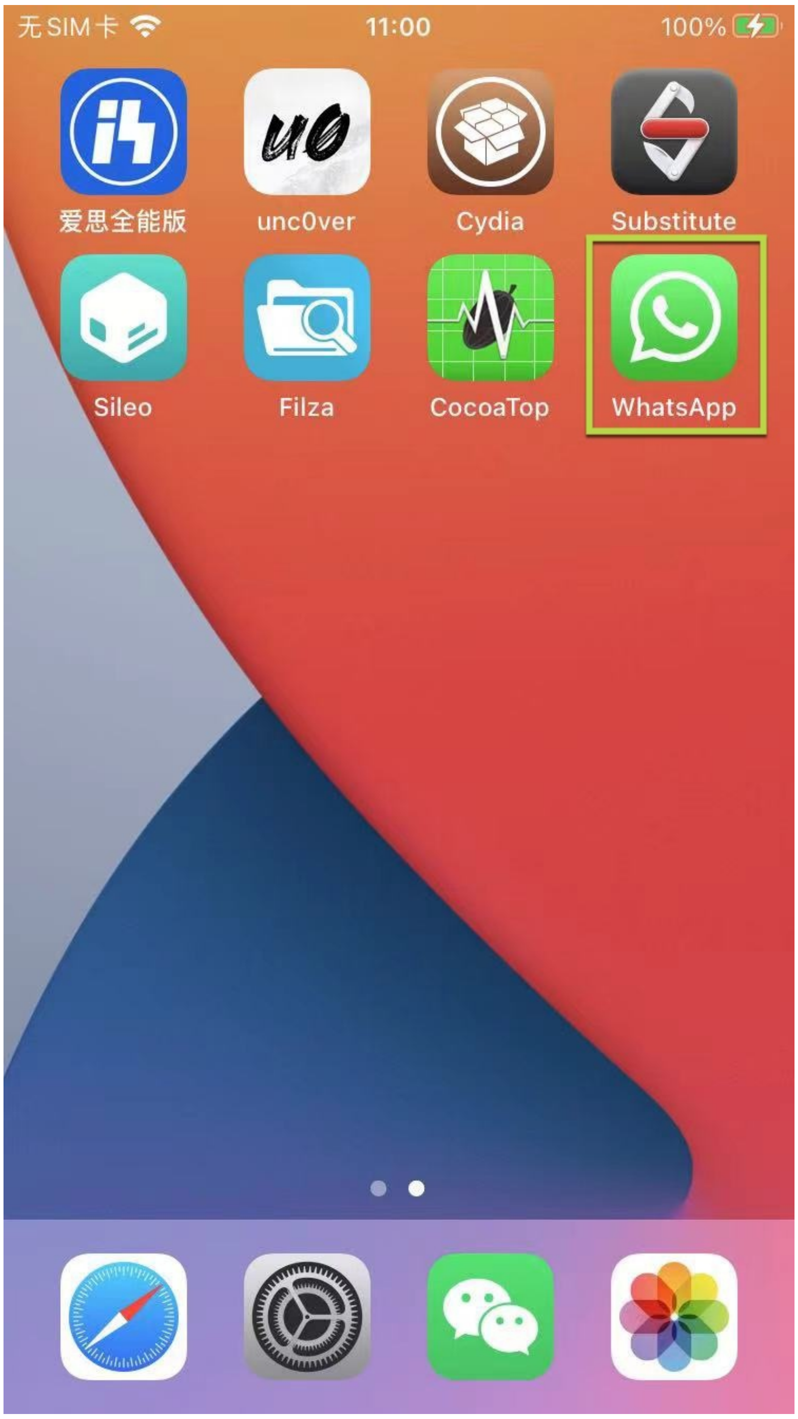
```
%end

/*=============================================================================
 AKAnisetteProvisioningService
=============================================================================*/

%hook AKAnisetteProvisioningService

- (id)initWithClient:(id)client authenticationContext:(id)authCtx{
    id newService = %orig;
    NSLog(@"jailAuthKitAkd AKAnisetteProvisioningService client=%@,authCtx=%@ -> newService=%@", client, authCtx, newService);
    return newService;
}

- (id)initWithClient:(id)client context:(id)ctx{
    id newService = %orig;
    NSLog(@"jailAuthKitAkd AKAnisetteProvisioningService client=%@,ctx=%@ -> newService=%@", client, ctx, newService);
    return newService;
}

- (id)initWithClient:(id)client{
    id newService = %orig;
    NSLog(@"jailAuthKitAkd AKAnisetteProvisioningService client=%@ -> newService=%@", client, newService);
    return newService;
}

- (void)fetchAnisetteDataAndProvisionIfNecessary:(_Bool)need device:(id)dev completion:(id)doneBlk{
    NSLog(@"jailAuthKitAkd AKAnisetteProvisioningService need=%d,dev=%@,doneBlk=%@", need, dev, doneBlk);
    %orig;
}

- (void)_fetchAnisetteDataAndProvisionIfNecessary:(_Bool)need device:(id)dev completion:(id)doneBlk{
    NSLog(@"jailAuthKitAkd AKAnisetteProvisioningService _ need=%d,dev=%@,doneBlk=%@", need, dev, doneBlk);
    %orig;
}

%end

/*=============================================================================
 ctor
```

    - 说明
      - 关于如何查看进程PID，详见：查看进程PID
  - 对于Name：`Waiting to attach to xxx`
    - 举例
      - akd
        - 

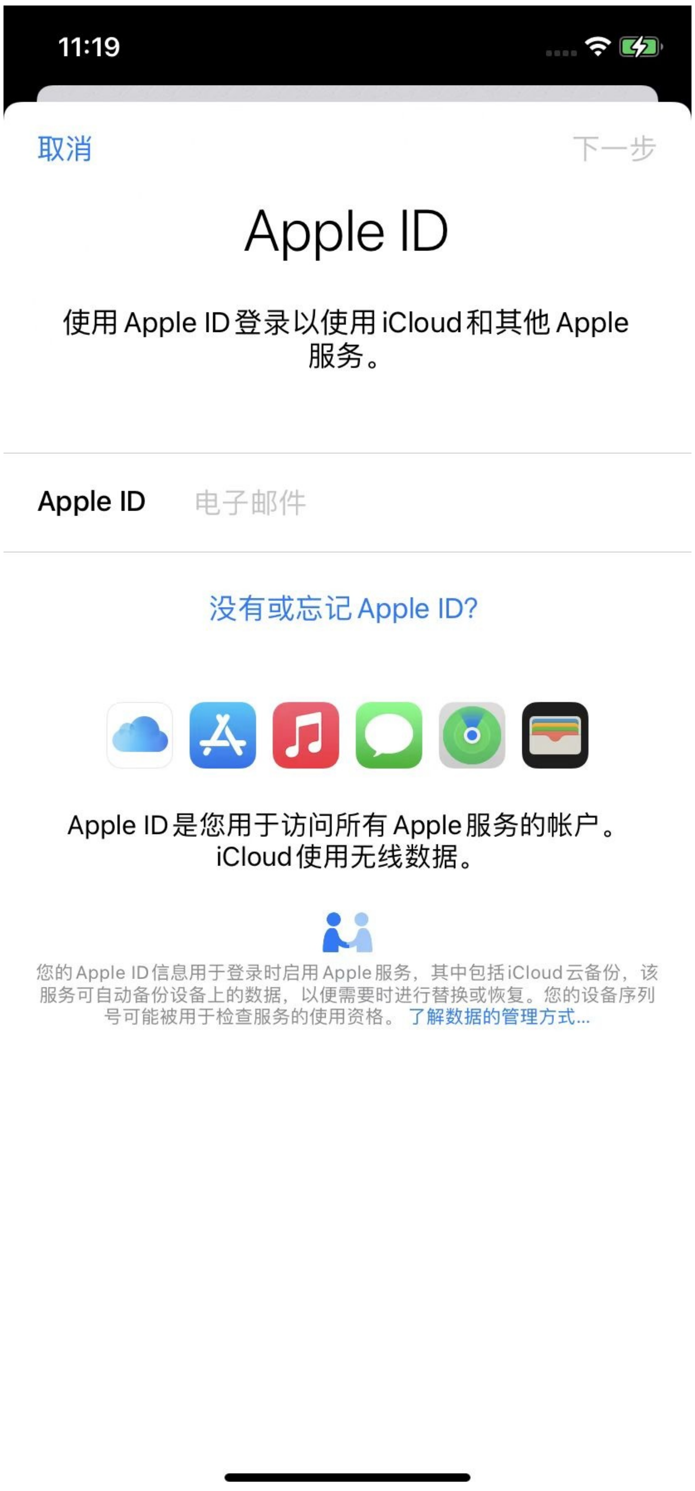
akd Waiting to Attach
jailAuthKitAkd
jailAuthKitAkd 〉 iPhone11_151 (arm64e)
Waiting to attach to akd on iPhone11_151
jailAuthKitAkd.xm
jailAuthKitAkd.mm
control
jailAuthKitAkd 〉 jailAuthKitAkd 〉 jailAuthKitAkd.xm 〉 No Selection
Waiting to attach

```
%end

/*=============================================================================
 AKAnisetteProvisioningService
=============================================================================*/

%hook AKAnisetteProvisioningService

- (id)initWithClient:(id)client authenticationContext:(id)authCtx{
    id newService = %orig;
    NSLog(@"jailAuthKitAkd AKAnisetteProvisioningService client=%@,authCtx=%@ -> newService=%@", client, authCtx, newService);
    return newService;
}

- (id)initWithClient:(id)client context:(id)ctx{
    id newService = %orig;
    NSLog(@"jailAuthKitAkd AKAnisetteProvisioningService client=%@,ctx=%@ -> newService=%@", client, ctx, newService);
    return newService;
}

- (id)initWithClient:(id)client{
    id newService = %orig;
    NSLog(@"jailAuthKitAkd AKAnisetteProvisioningService client=%@ -> newService=%@", client, newService);
    return newService;
}

- (void)fetchAnisetteDataAndProvisionIfNecessary:(_Bool)need device:(id)dev completion:(id)doneBlk{
    NSLog(@"jailAuthKitAkd AKAnisetteProvisioningService need=%d,dev=%@,doneBlk=%@", need, dev, doneBlk);
    %orig;
}

- (void)_fetchAnisetteDataAndProvisionIfNecessary:(_Bool)need device:(id)dev completion:(id)doneBlk{
    NSLog(@"jailAuthKitAkd AKAnisetteProvisioningService _ need=%d,dev=%@,doneBlk=%@", need, dev, doneBlk);
    %orig;
}

%end

/*=============================================================================
 ctor
```

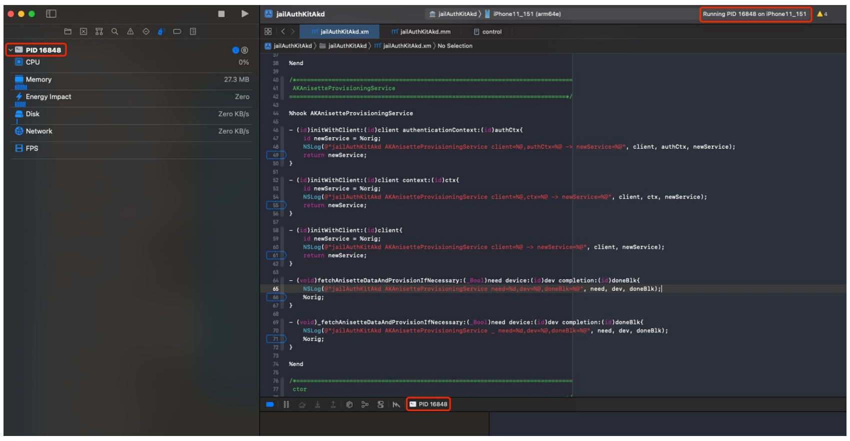
- iPhone中：
  - 确保进程运行
    - （手动点击）启动app
      - 点击app桌面图标
        - 举例
          - WhatsApp

- 

- 或：以某种方式，触发二进制服务进程加载
  - 举例
    - akd
      - 点击iOS系统的 `设置` = `Preferences` =包名 `com.apple.Preferences`
        - 内部会触发XPC的服务，去请求到 `akd` ，从而触发启动 `akd` 进程
          - 比如，Apple ID账号登录

- 
  11:19
  取消
  下一步
  Apple ID
  使用 Apple ID 登录以使用 iCloud 和其他 Apple 服务。
  Apple ID 电子邮件
  没有或忘记 Apple ID?
  Apple ID 是您用于访问所有 Apple 服务的帐户。iCloud 使用无线数据。
  您的 Apple ID 信息用于登录时启用 Apple 服务，其中包括 iCloud 云备份，该服务可自动备份设备上的数据，以便需要时进行替换或恢复。您的设备序列号可能被用于检查服务的使用资格。了解数据的管理方式...
- 说明：
  - akd完整路径：`/System/Library/PrivateFrameworks/AuthKit.framework/akd`

- Xcode中：
  - 说明：需要稍等一段时间，才能挂载上
    - 一般都很快：大概几秒
    - 偶尔：很多秒（之后，才能挂载上，注意此时要多等一会）
  - 即可挂载成功
    - 具体现象

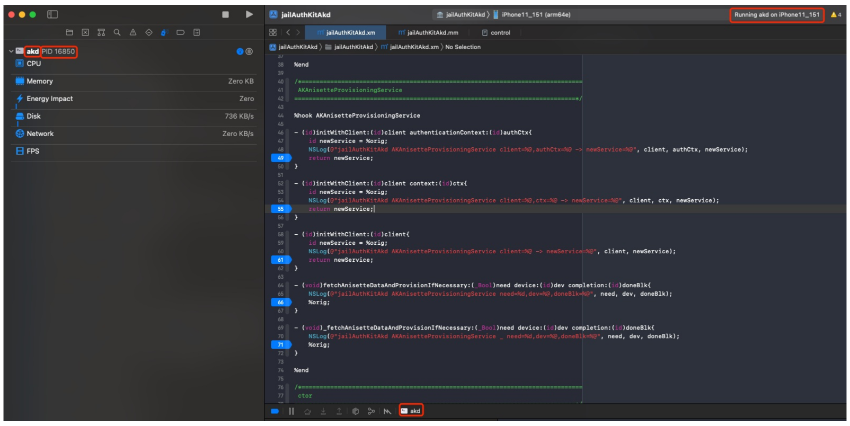
- 描述
  - 右下角：调试窗口：显示出进程PID值/Name
  - 右上角：状态信息显示：Running PID值/Name on iPhone
  - 左上角：Debug Navigator中显示出PID值/Name和当前硬件信息：CPU、Memory等等
- 图
  - 对于PID
    - 

```
PID 16848
CPU                0%
Memory             27.3 MB
Energy Impact      Zero
Disk               Zero KB/s
Network            Zero KB/s
FPS
```

```
jailAuthKitAkd    jailAuthKitAkd 〉 iPhone11_151 (arm64e)    Running PID 16848 on iPhone11_151    ⚠ 4
jailAuthKitAkd.xm    jailAuthKitAkd.mm    control
jailAuthKitAkd 〉 jailAuthKitAkd 〉 jailAuthKitAkd.xm 〉 No Selection
```

```
%end

/*==============================================================================
 AKAnisetteProvisioningService
==============================================================================*/

%hook AKAnisetteProvisioningService

- (id)initWithClient:(id)client authenticationContext:(id)authCtx{
    id newService = %orig;
    NSLog(@"jailAuthKitAkd AKAnisetteProvisioningService client=%@,authCtx=%@ -> newService=%@", client, authCtx, newService);
    return newService;
}

- (id)initWithClient:(id)client context:(id)ctx{
    id newService = %orig;
    NSLog(@"jailAuthKitAkd AKAnisetteProvisioningService client=%@,ctx=%@ -> newService=%@", client, ctx, newService);
    return newService;
}

- (id)initWithClient:(id)client{
    id newService = %orig;
    NSLog(@"jailAuthKitAkd AKAnisetteProvisioningService client=%@ -> newService=%@", client, newService);
    return newService;
}

- (void)fetchAnisetteDataAndProvisionIfNecessary:(_Bool)need device:(id)dev completion:(id)doneBlk{
    NSLog(@"jailAuthKitAkd AKAnisetteProvisioningService need=%d,dev=%@,doneBlk=%@", need, dev, doneBlk);
    %orig;
}

- (void)_fetchAnisetteDataAndProvisionIfNecessary:(_Bool)need device:(id)dev completion:(id)doneBlk{
    NSLog(@"jailAuthKitAkd AKAnisetteProvisioningService _ need=%d,dev=%@,doneBlk=%@", need, dev, doneBlk);
    %orig;
}

%end

/*==============================================================================
 ctor
```

```
PID 16848
```

  - 对于Name
    - 

```
akd PID 16850
CPU
Memory             Zero KB
Energy Impact      Zero
Disk               736 KB/s
Network            Zero KB/s
FPS
```

```
jailAuthKitAkd    jailAuthKitAkd 〉 iPhone11_151 (arm64e)    Running akd on iPhone11_151    ⚠ 4
jailAuthKitAkd.xm    jailAuthKitAkd.mm    control
jailAuthKitAkd 〉 jailAuthKitAkd 〉 jailAuthKitAkd.xm 〉 No Selection
```

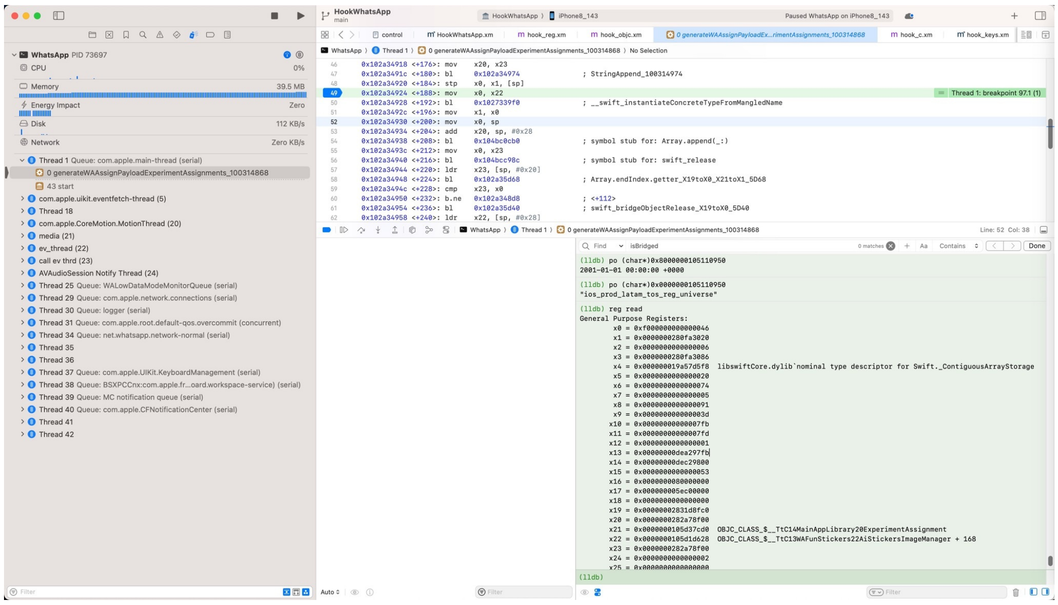
```
%end

/*==============================================================================
 AKAnisetteProvisioningService
==============================================================================*/

%hook AKAnisetteProvisioningService

- (id)initWithClient:(id)client authenticationContext:(id)authCtx{
    id newService = %orig;
    NSLog(@"jailAuthKitAkd AKAnisetteProvisioningService client=%@,authCtx=%@ -> newService=%@", client, authCtx, newService);
    return newService;
}

- (id)initWithClient:(id)client context:(id)ctx{
    id newService = %orig;
    NSLog(@"jailAuthKitAkd AKAnisetteProvisioningService client=%@,ctx=%@ -> newService=%@", client, ctx, newService);
    return newService;
}

- (id)initWithClient:(id)client{
    id newService = %orig;
    NSLog(@"jailAuthKitAkd AKAnisetteProvisioningService client=%@ -> newService=%@", client, newService);
    return newService;
}

- (void)fetchAnisetteDataAndProvisionIfNecessary:(_Bool)need device:(id)dev completion:(id)doneBlk{
    NSLog(@"jailAuthKitAkd AKAnisetteProvisioningService need=%d,dev=%@,doneBlk=%@", need, dev, doneBlk);
    %orig;
}

- (void)_fetchAnisetteDataAndProvisionIfNecessary:(_Bool)need device:(id)dev completion:(id)doneBlk{
    NSLog(@"jailAuthKitAkd AKAnisetteProvisioningService _ need=%d,dev=%@,doneBlk=%@", need, dev, doneBlk);
    %orig;
}

%end

/*==============================================================================
 ctor
```

```
akd
```

- 然后即可正常（Spawn方式去）调试

效果举例

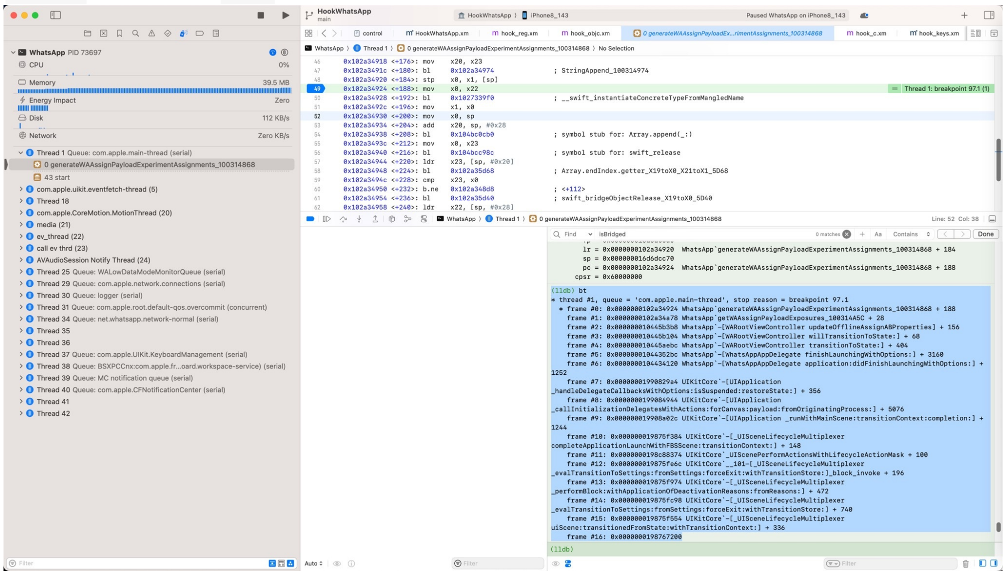
此处举例说明iOS逆向的 `Xcode+iOSOpenDev` 的图形界面调试的效果：

- 方便查看和调试
  - Xcode
    - Xcode中查看汇编代码
      - 

```
WhatsApp PID 73697
CPU                0%
Memory             39.5 MB
Energy Impact      Zero
Disk               112 KB/s
Network            Zero KB/s
Thread 1 Queue: com.apple.main-thread (serial)
    0 generateWAAssignPayloadExperimentAssignments_100314868
    43 start
com.apple.uikit.eventfetch-thread (5)
Thread 18
com.apple.CoreMotion.MotionThread (20)
media (21)
ev_thread (22)
call ev thrd (23)
AVAudioSession Notify Thread (24)
Thread 25 Queue: WALowDataModeMonitorQueue (serial)
Thread 29 Queue: com.apple.network.connections (serial)
Thread 30 Queue: logger (serial)
Thread 31 Queue: com.apple.root.default-qos.overcommit (concurrent)
Thread 34 Queue: net.whatsapp.network-normal (serial)
Thread 35
Thread 36
Thread 37 Queue: com.apple.UIKit.KeyboardManagement (serial)
Thread 38 Queue: BSXPCCnx:com.apple.fr...oard.workspace-service) (serial)
Thread 39 Queue: MC notification queue (serial)
Thread 40 Queue: com.apple.CFNotificationCenter (serial)
Thread 41
Thread 42
```

```
HookWhatsApp  main    HookWhatsApp 〉 iPhone8_143    Paused WhatsApp on iPhone8_143
control | HookWhatsApp.xm | hook_reg.xm | hook_objc.xm | 0 generateWAAssignPayloadEx...rimentAssignments_100314868 | hook_c.xm | hook_keys.xm
WhatsApp 〉 Thread 1 〉 0 generateWAAssignPayloadExperimentAssignments_100314868 〉 No Selection

0x102a34918 <+176>: mov    x20, x23
0x102a3491c <+180>: bl     0x102a34974             ; StringAppend_100314974
0x102a34920 <+184>: stp    x0, x1, [sp]
0x102a34924 <+188>: mov    x0, x22                                    Thread 1: breakpoint 97.1 (1)
0x102a34928 <+192>: bl     0x1027339f0             ; __swift_instantiateConcreteTypeFromMangledName
0x102a3492c <+196>: mov    x1, x0
0x102a34930 <+200>: mov    x0, sp
0x102a34934 <+204>: add    x20, sp, #0x28
0x102a34938 <+208>: bl     0x104bc0cb0             ; symbol stub for: Array.append(_:)
0x102a3493c <+212>: mov    x0, x23
0x102a34940 <+216>: bl     0x104bcc98c             ; symbol stub for: swift_release
0x102a34944 <+220>: ldr    x23, [sp, #0x20]
0x102a34948 <+224>: bl     0x102a35d68             ; Array.endIndex.getter_X19toX0_X21toX1_5D68
0x102a3494c <+228>: cmp    x23, x0
0x102a34950 <+232>: b.ne   0x102a348d8             ; <+112>
0x102a34954 <+236>: bl     0x102a35d40             ; swift_bridgeObjectRelease_X19toX0_5D40
0x102a34958 <+240>: ldr    x22, [sp, #0x28]

WhatsApp 〉 Thread 1 〉 0 generateWAAssignPayloadExperimentAssignments_100314868    Line: 52  Col: 38
```

```
(lldb) po (char*)0x8000000105110950
2001-01-01 00:00:00 +0000

(lldb) po (char*)0x0000000105110950
"ios_prod_latam_tos_reg_universe"

(lldb) reg read
General Purpose Registers:
        x0 = 0xf000000000000046
        x1 = 0x0000000280fa3020
        x2 = 0x0000000000000006
        x3 = 0x0000000280fa3086
        x4 = 0x000000019a57d5f8  libswiftCore.dylib`nominal type descriptor for Swift._ContiguousArrayStorage
        x5 = 0x0000000000000020
        x6 = 0x0000000000000074
        x7 = 0x0000000000000005
        x8 = 0x0000000000000091
        x9 = 0x000000000000003d
       x10 = 0x00000000000007fb
       x11 = 0x00000000000007fd
       x12 = 0x0000000000000001
       x13 = 0x00000000dea297fb
       x14 = 0x00000000dec29800
       x15 = 0x0000000000000053
       x16 = 0x0000000080000000
       x17 = 0x000000005ec00000
       x18 = 0x0000000000000000
       x19 = 0x00000002831d8fc0
       x20 = 0x0000000282a78f00
       x21 = 0x0000000105d37cd0  OBJC_CLASS_$__TtC14MainAppLibrary20ExperimentAssignment
       x22 = 0x0000000105d1d628  OBJC_CLASS_$__TtC13WAFunStickers22AiStickersImageManager + 168
       x23 = 0x0000000282a78f00
       x24 = 0x0000000000000002
       x25 = 0x0000000000000000
(lldb)
```

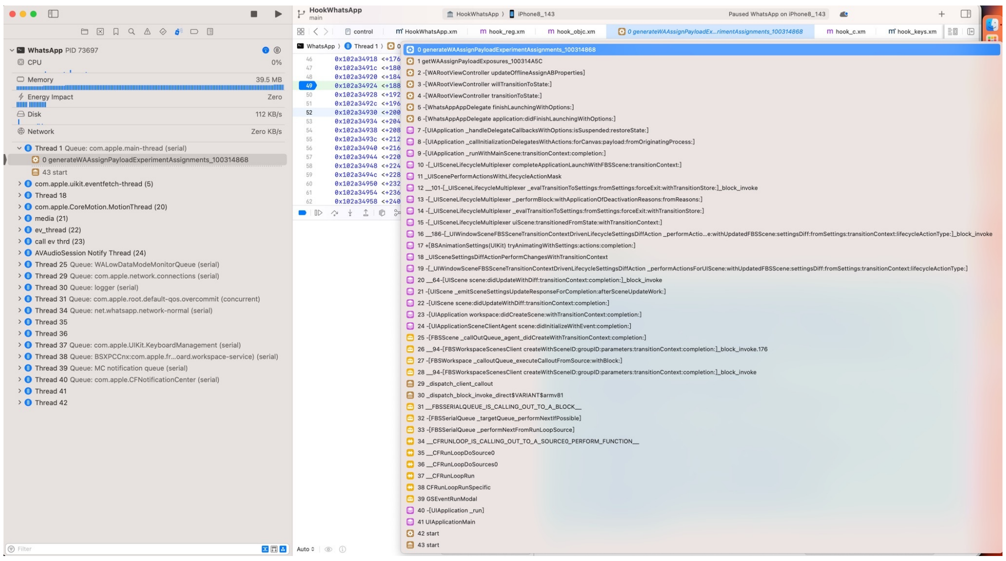
- 查看函数调用堆栈
  - 除了 `lldb` **命令行中的** `bt`
    - 

```
WhatsApp PID 73697
CPU                0%
Memory             39.5 MB
Energy Impact      Zero
Disk               112 KB/s
Network            Zero KB/s
Thread 1 Queue: com.apple.main-thread (serial)
    0 generateWAAssignPayloadExperimentAssignments_100314868
    43 start
com.apple.uikit.eventfetch-thread (5)
Thread 18
com.apple.CoreMotion.MotionThread (20)
media (21)
ev_thread (22)
call ev thrd (23)
AVAudioSession Notify Thread (24)
Thread 25 Queue: WALowDataModeMonitorQueue (serial)
Thread 29 Queue: com.apple.network.connections (serial)
Thread 30 Queue: logger (serial)
Thread 31 Queue: com.apple.root.default-qos.overcommit (concurrent)
Thread 34 Queue: net.whatsapp.network-normal (serial)
Thread 35
Thread 36
Thread 37 Queue: com.apple.UIKit.KeyboardManagement (serial)
Thread 38 Queue: BSXPCCnx:com.apple.fr...oard.workspace-service) (serial)
Thread 39 Queue: MC notification queue (serial)
Thread 40 Queue: com.apple.CFNotificationCenter (serial)
Thread 41
Thread 42
```

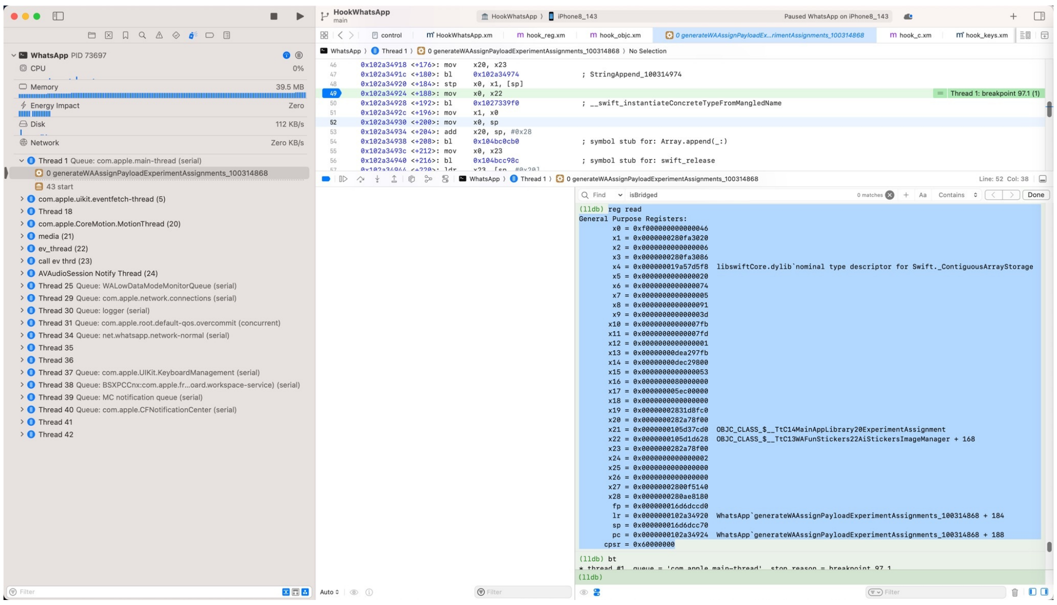
```
HookWhatsApp  main    HookWhatsApp 〉 iPhone8_143    Paused WhatsApp on iPhone8_143
control | HookWhatsApp.xm | hook_reg.xm | hook_objc.xm | 0 generateWAAssignPayloadEx...rimentAssignments_100314868 | hook_c.xm | hook_keys.xm
WhatsApp 〉 Thread 1 〉 0 generateWAAssignPayloadExperimentAssignments_100314868 〉 No Selection

0x102a34918 <+176>: mov    x20, x23
0x102a3491c <+180>: bl     0x102a34974             ; StringAppend_100314974
0x102a34920 <+184>: stp    x0, x1, [sp]
0x102a34924 <+188>: mov    x0, x22                                    Thread 1: breakpoint 97.1 (1)
0x102a34928 <+192>: bl     0x1027339f0             ; __swift_instantiateConcreteTypeFromMangledName
0x102a3492c <+196>: mov    x1, x0
0x102a34930 <+200>: mov    x0, sp
0x102a34934 <+204>: add    x20, sp, #0x28
0x102a34938 <+208>: bl     0x104bc0cb0             ; symbol stub for: Array.append(_:)
0x102a3493c <+212>: mov    x0, x23
0x102a34940 <+216>: bl     0x104bcc98c             ; symbol stub for: swift_release
0x102a34944 <+220>: ldr    x23, [sp, #0x20]
0x102a34948 <+224>: bl     0x102a35d68             ; Array.endIndex.getter_X19toX0_X21toX1_5D68
0x102a3494c <+228>: cmp    x23, x0
0x102a34950 <+232>: b.ne   0x102a348d8             ; <+112>
0x102a34954 <+236>: bl     0x102a35d40             ; swift_bridgeObjectRelease_X19toX0_5D40
0x102a34958 <+240>: ldr    x22, [sp, #0x28]

WhatsApp 〉 Thread 1 〉 0 generateWAAssignPayloadExperimentAssignments_100314868    Line: 52  Col: 38
```

```
        lr = 0x0000000102a34920  WhatsApp`generateWAAssignPayloadExperimentAssignments_100314868 + 184
        sp = 0x000000016d6dcc70
        pc = 0x0000000102a34924  WhatsApp`generateWAAssignPayloadExperimentAssignments_100314868 + 188
      cpsr = 0x60000000

(lldb) bt
* thread #1, queue = 'com.apple.main-thread', stop reason = breakpoint 97.1
  * frame #0: 0x0000000102a34924 WhatsApp`generateWAAssignPayloadExperimentAssignments_100314868 + 188
    frame #1: 0x0000000102a34a78 WhatsApp`getWAAssignPayloadExposures_100314A5C + 28
    frame #2: 0x000000010445b3b8 WhatsApp`-[WARootViewController updateOfflineAssignABProperties] + 156
    frame #3: 0x000000010445b104 WhatsApp`-[WARootViewController willTransitionToState:] + 68
    frame #4: 0x000000010445aebc WhatsApp`-[WARootViewController transitionToState:] + 404
    frame #5: 0x00000001044352bc WhatsApp`-[WhatsAppAppDelegate finishLaunchingWithOptions:] + 3160
    frame #6: 0x0000000104434120 WhatsApp`-[WhatsAppAppDelegate application:didFinishLaunchingWithOptions:] +
1252
    frame #7: 0x00000001990829a4 UIKitCore`-[UIApplication
_handleDelegateCallbacksWithOptions:isSuspended:restoreState:] + 356
    frame #8: 0x0000000199084944 UIKitCore`-[UIApplication
_callInitializationDelegatesWithActions:forCanvas:payload:fromOriginatingProcess:] + 5076
    frame #9: 0x000000019908a02c UIKitCore`-[UIApplication _runWithMainScene:transitionContext:completion:] +
1244
    frame #10: 0x000000019875f384 UIKitCore`-[_UISceneLifecycleMultiplexer
completeApplicationLaunchWithFBSScene:transitionContext:] + 148
    frame #11: 0x0000000198c88374 UIKitCore`_UIScenePerformActionsWithLifecycleActionMask + 100
    frame #12: 0x000000019875fe6c UIKitCore`__101-[_UISceneLifecycleMultiplexer
_evalTransitionToSettings:fromSettings:forceExit:withTransitionStore:]_block_invoke + 196
    frame #13: 0x000000019875f974 UIKitCore`-[_UISceneLifecycleMultiplexer
_performBlock:withApplicationOfDeactivationReasons:fromReasons:] + 472
    frame #14: 0x000000019875fc98 UIKitCore`-[_UISceneLifecycleMultiplexer
_evalTransitionToSettings:fromSettings:forceExit:withTransitionStore:] + 740
    frame #15: 0x000000019875f554 UIKitCore`-[_UISceneLifecycleMultiplexer
uiScene:transitionedFromState:withTransitionContext:] + 336
    frame #16: 0x0000000198767200
(lldb)
```

  - 还可以GUI**图形界面**中查看：

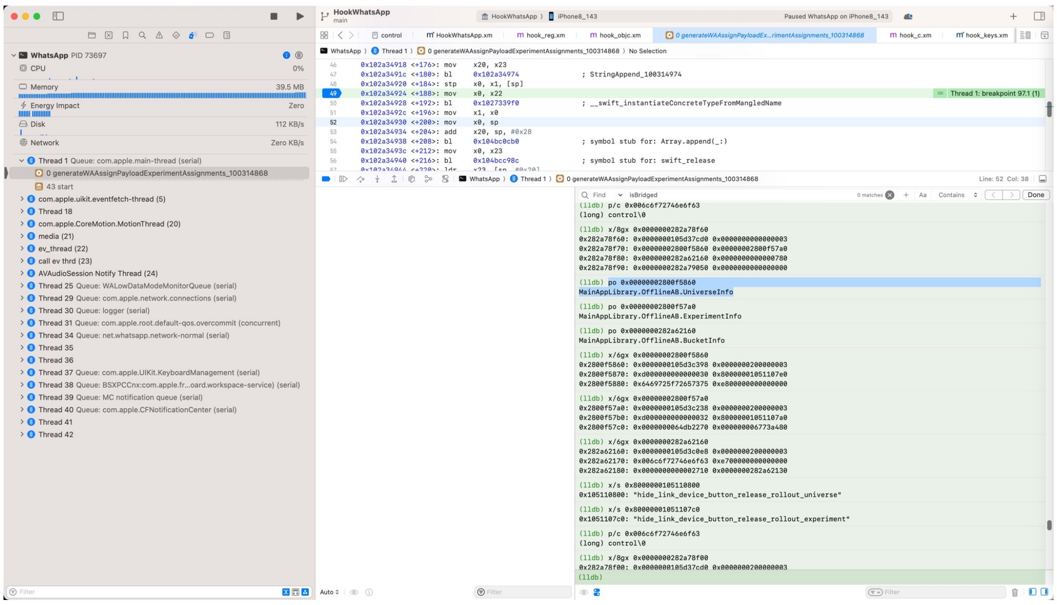
- ■ 

```
0 generateWAAssignPayloadExperimentAssignments_100314868
1 getWAAssignPayloadExposures_100314A5C
2 -[WARootViewController updateOfflineAssignABProperties]
3 -[WARootViewController willTransitionToState:]
4 -[WARootViewController transitionToState:]
5 -[WhatsAppAppDelegate finishLaunchingWithOptions:]
6 -[WhatsAppAppDelegate application:didFinishLaunchingWithOptions:]
7 -[UIApplication _handleDelegateCallbacksWithOptions:isSuspended:restoreState:]
8 -[UIApplication _callInitializationDelegatesWithActions:forCanvas:payload:fromOriginatingProcess:]
9 -[UIApplication _runWithMainScene:transitionContext:completion:]
10 -[_UISceneLifecycleMultiplexer completeApplicationLaunchWithFBSScene:transitionContext:]
11 _UIScenePerformActionsWithLifecycleActionMask
12 __101-[_UISceneLifecycleMultiplexer _evalTransitionToSettings:fromSettings:forceExit:withTransitionStore:]_block_invoke
13 -[_UISceneLifecycleMultiplexer _performBlock:withApplicationOfDeactivationReasons:fromReasons:]
14 -[_UISceneLifecycleMultiplexer _evalTransitionToSettings:fromSettings:forceExit:withTransitionStore:]
15 -[_UISceneLifecycleMultiplexer uiScene:transitionedFromState:withTransitionContext:]
16 __186-[_UIWindowSceneFBSSceneTransitionContextDrivenLifecycleSettingsDiffAction _performActio...e:withUpdatedFBSScene:settingsDiff:fromSettings:transitionContext:lifecycleActionType:]_block_invoke
17 +[BSAnimationSettings(UIKit) tryAnimatingWithSettings:actions:completion:]
18 _UISceneSettingsDiffActionPerformChangesWithTransitionContext
19 -[_UIWindowSceneFBSSceneTransitionContextDrivenLifecycleSettingsDiffAction _performActionsForUIScene:withUpdatedFBSScene:settingsDiff:fromSettings:transitionContext:lifecycleActionType:]
20 __64-[UIScene scene:didUpdateWithDiff:transitionContext:completion:]_block_invoke
21 -[UIScene _emitSceneSettingsUpdateResponseForCompletion:afterSceneUpdateWork:]
22 -[UIScene scene:didUpdateWithDiff:transitionContext:completion:]
23 -[UIApplication workspace:didCreateScene:withTransitionContext:completion:]
24 -[UIApplicationSceneClientAgent scene:didInitializeWithEvent:completion:]
25 -[FBSScene _callOutQueue_agent_didCreateWithTransitionContext:completion:]
26 __94-[FBSWorkspaceScenesClient createWithSceneID:groupID:parameters:transitionContext:completion:]_block_invoke.176
27 -[FBSWorkspace _calloutQueue_executeCalloutFromSource:withBlock:]
28 __94-[FBSWorkspaceScenesClient createWithSceneID:groupID:parameters:transitionContext:completion:]_block_invoke
29 _dispatch_client_callout
30 _dispatch_block_invoke_direct$VARIANT$armv81
31 __FBSSERIALQUEUE_IS_CALLING_OUT_TO_A_BLOCK__
32 -[FBSSerialQueue _targetQueue_performNextIfPossible]
33 -[FBSSerialQueue _performNextFromRunLoopSource]
34 __CFRUNLOOP_IS_CALLING_OUT_TO_A_SOURCE0_PERFORM_FUNCTION__
35 __CFRunLoopDoSource0
36 __CFRunLoopDoSources0
37 __CFRunLoopRun
38 CFRunLoopRunSpecific
39 GSEventRunModal
40 -[UIApplication _run]
41 UIApplicationMain
42 start
43 start
```

- lldb中
  - 查看寄存器变量
    - ■ 

```
0x102a34918 <+176>: mov   x20, x23
0x102a3491c <+180>: bl    0x102a34974               ; StringAppend_100314974
0x102a34920 <+184>: stp   x0, x1, [sp]
0x102a34924 <+188>: mov   x0, x22                                Thread 1: breakpoint 97.1 (1)
0x102a34928 <+192>: bl    0x1027339f0               ; __swift_instantiateConcreteTypeFromMangledName
0x102a3492c <+196>: mov   x1, x0
0x102a34930 <+200>: mov   x0, sp
0x102a34934 <+204>: add   x20, sp, #0x28
0x102a34938 <+208>: bl    0x104bc0cb0               ; symbol stub for: Array.append(_:)
0x102a3493c <+212>: mov   x0, x23
0x102a34940 <+216>: bl    0x104bcc98c               ; symbol stub for: swift_release
```

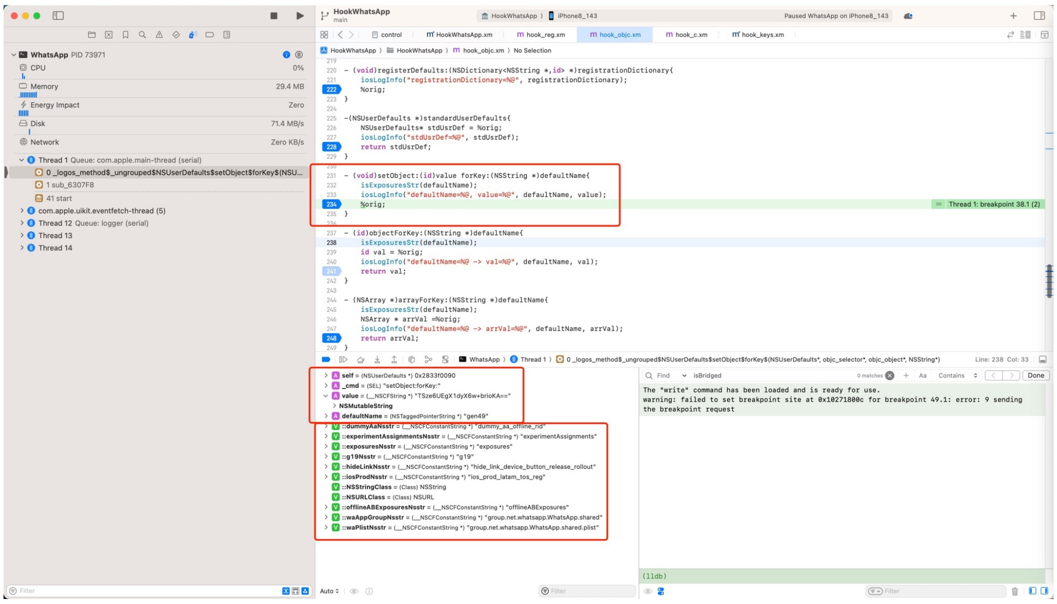
```
(lldb) reg read
General Purpose Registers:
        x0 = 0xf000000000000046
        x1 = 0x0000000280fa3020
        x2 = 0x0000000000000006
        x3 = 0x0000000280fa3086
        x4 = 0x000000019a57d5f8  libswiftCore.dylib`nominal type descriptor for Swift._ContiguousArrayStorage
        x5 = 0x0000000000000020
        x6 = 0x0000000000000074
        x7 = 0x0000000000000005
        x8 = 0x0000000000000091
        x9 = 0x000000000000003d
       x10 = 0x00000000000007fb
       x11 = 0x00000000000007fd
       x12 = 0x0000000000000001
       x13 = 0x00000000dea297fb
       x14 = 0x00000000dec29800
       x15 = 0x0000000000000053
       x16 = 0x0000000080000000
       x17 = 0x000000005ec00000
       x18 = 0x0000000000000000
       x19 = 0x00000002831d8fc0
       x20 = 0x0000000282a78f00
       x21 = 0x0000000105d37cd0  OBJC_CLASS_$__TtC14MainAppLibrary20ExperimentAssignment
       x22 = 0x0000000105d1d628  OBJC_CLASS_$__TtC13WAFunStickers22AiStickersImageManager + 168
       x23 = 0x0000000282a78f00
       x24 = 0x0000000000000002
       x25 = 0x0000000000000000
       x26 = 0x0000000000000000
       x27 = 0x00000002800f5140
       x28 = 0x0000000280ae8180
        fp = 0x000000016d6dccd0
        lr = 0x0000000102a34920  WhatsApp`generateWAAssignPayloadExperimentAssignments_100314868 + 184
        sp = 0x000000016d6dcc70
        pc = 0x0000000102a34924  WhatsApp`generateWAAssignPayloadExperimentAssignments_100314868 + 188
      cpsr = 0x60000000

(lldb) bt
* thread #1, queue = 'com.apple.main-thread', stop reason = breakpoint 97.1
(lldb)
```

  - 用po查看变量的值
    - ■ 

```
(lldb) p/c 0x006c6f72746e6f63
(long) control\0

(lldb) x/8gx 0x0000000282a78f60
0x282a78f60: 0x0000000105d37cd0 0x0000000000000003
0x282a78f70: 0x00000002800f5860 0x00000002800f57a0
0x282a78f80: 0x0000000282a62160 0x0000000000000780
0x282a78f90: 0x0000000282a79050 0x0000000000000000

(lldb) po 0x00000002800f5860
MainAppLibrary.OfflineAB.UniverseInfo

(lldb) po 0x00000002800f57a0
MainAppLibrary.OfflineAB.ExperimentInfo

(lldb) po 0x0000000282a62160
MainAppLibrary.OfflineAB.BucketInfo

(lldb) x/6gx 0x00000002800f5860
0x2800f5860: 0x0000000105d3c398 0x0000000200000003
0x2800f5870: 0xd000000000000030 0x80000001051107e0
0x2800f5880: 0x6469725f72657375 0xe800000000000000

(lldb) x/6gx 0x00000002800f57a0
0x2800f57a0: 0x0000000105d3c238 0x0000000200000003
0x2800f57b0: 0xd000000000000032 0x80000001051107a0
0x2800f57c0: 0x0000000064db2270 0x000000006773a480

(lldb) x/6gx 0x0000000282a62160
0x282a62160: 0x0000000105d3c0e8 0x0000000200000003
0x282a62170: 0x006c6f72746e6f63 0xe700000000000000
0x282a62180: 0x0000000000002710 0x0000000282a62130

(lldb) x/s 0x8000000105110800
0x105110800: "hide_link_device_button_release_rollout_universe"

(lldb) x/s 0x80000001051107c0
0x1051107c0: "hide_link_device_button_release_rollout_experiment"

(lldb) p/c 0x006c6f72746e6f63
(long) control\0

(lldb) x/8gx 0x0000000282a78f00
0x282a78f00: 0x0000000105d37cd0 0x0000000200000003
(lldb)
```

- 对于hook代码

- 用iOSOpenDev写插件代码，加断点，实时调试，触发断点后的）效果
  - 能看到各种参数值和当前变量的值，甚至包括全局变量的值



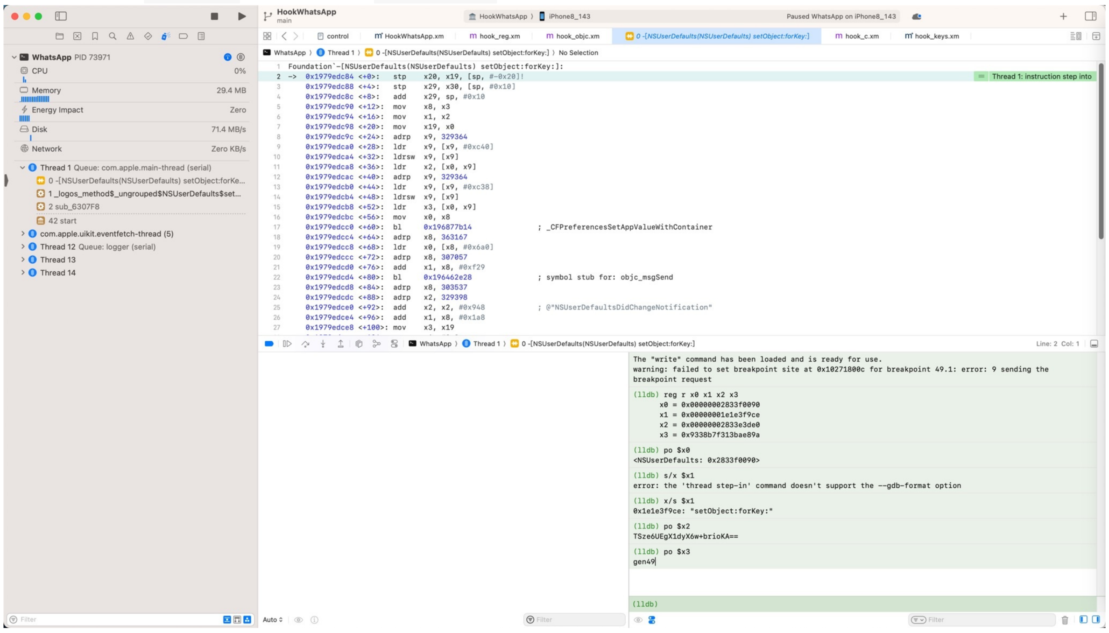
```
- (void)registerDefaults:(NSDictionary<NSString *,id> *)registrationDictionary{
    iosLogInfo("registrationDictionary=%@", registrationDictionary);
    %orig;
}

-(NSUserDefaults *)standardUserDefaults{
    NSUserDefaults* stdUsrDef = %orig;
    iosLogInfo("stdUsrDef=%@", stdUsrDef);
    return stdUsrDef;
}

- (void)setObject:(id)value forKey:(NSString *)defaultName{
    isExposuresStr(defaultName);
    iosLogInfo("defaultName=%@, value=%@", defaultName, value);
    %orig;
}

- (id)objectForKey:(NSString *)defaultName{
    isExposuresStr(defaultName);
    id val = %orig;
    iosLogInfo("defaultName=%@ -> val=%@", defaultName, val);
    return val;
}

- (NSArray *)arrayForKey:(NSString *)defaultName{
    isExposuresStr(defaultName);
    NSArray * arrVal =%orig;
    iosLogInfo("defaultName=%@ -> arrVal=%@", defaultName, arrVal);
    return arrVal;
}
```

WhatsApp 〉 Thread 1 〉 0 _logos_method$_ungrouped$NSUserDefaults$setObject$forKey$(NSUserDefaults*, objc_selector*, objc_object*, NSString*)

```
self = (NSUserDefaults *) 0x2833f0090
_cmd = (SEL) "setObject:forKey:"
value = (__NSCFString *) "TSze6UEgX1dyX6w+brioKA=="
  NSMutableString
defaultName = (NSTaggedPointerString *) "gen49"
::dummyAaNsstr = (__NSCFConstantString *) "dummy_aa_offline_rid"
::experimentAssignmentsNsstr = (__NSCFConstantString *) "experimentAssignments"
::exposuresNsstr = (__NSCFConstantString *) "exposures"
::g19Nsstr = (__NSCFConstantString *) "g19"
::hideLinkNsstr = (__NSCFConstantString *) "hide_link_device_button_release_rollout"
::iosProdNsstr = (__NSCFConstantString *) "ios_prod_latam_tos_reg"
::NSStringClass = (Class) NSString
::NSURLClass = (Class) NSURL
::offlineABExposuresNsstr = (__NSCFConstantString *) "offlineABExposures"
::waAppGroupNsstr = (__NSCFConstantString *) "group.net.whatsapp.WhatsApp.shared"
::waPlistNsstr = (__NSCFConstantString *) "group.net.whatsapp.WhatsApp.shared.plist"
```

```
The "write" command has been loaded and is ready for use.
warning: failed to set breakpoint site at 0x10271800c for breakpoint 49.1: error: 9 sending
the breakpoint request

(lldb)
```

- 对于hook代码的 `%orig` ，（多次） `Ctrl+F7` 后，可以进入对应的Xcode汇编代码

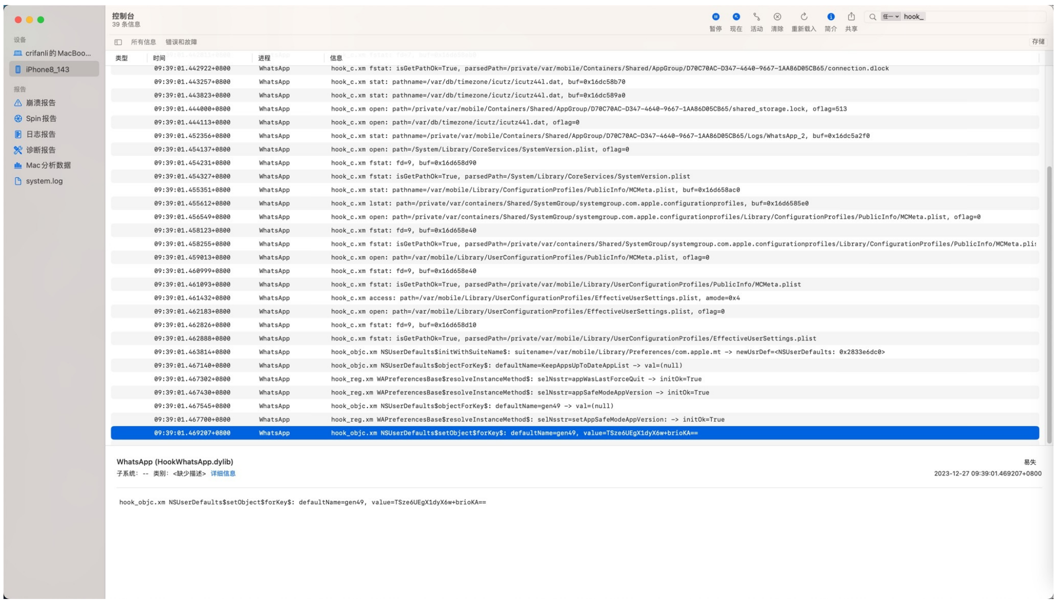


WhatsApp 〉 Thread 1 〉 0 -[NSUserDefaults(NSUserDefaults) setObject:forKey:] 〉 No Selection

```
Foundation`-[NSUserDefaults(NSUserDefaults) setObject:forKey:]:
->  0x1979edc84 <+0>:   stp    x20, x19, [sp, #-0x20]!
    0x1979edc88 <+4>:   stp    x29, x30, [sp, #0x10]
    0x1979edc8c <+8>:   add    x29, sp, #0x10
    0x1979edc90 <+12>:  mov    x8, x3
    0x1979edc94 <+16>:  mov    x1, x2
    0x1979edc98 <+20>:  mov    x19, x0
    0x1979edc9c <+24>:  adrp   x9, 329364
    0x1979edca0 <+28>:  ldr    x9, [x9, #0xc40]
    0x1979edca4 <+32>:  ldrsw  x9, [x9]
    0x1979edca8 <+36>:  ldr    x2, [x0, x9]
    0x1979edcac <+40>:  adrp   x9, 329364
    0x1979edcb0 <+44>:  ldr    x9, [x9, #0xc38]
    0x1979edcb4 <+48>:  ldrsw  x9, [x9]
    0x1979edcb8 <+52>:  ldr    x3, [x0, x9]
    0x1979edcbc <+56>:  mov    x0, x8
    0x1979edcc0 <+60>:  bl     0x196877b14               ; _CFPreferencesSetAppValueWithContainer
    0x1979edcc4 <+64>:  adrp   x8, 363167
    0x1979edcc8 <+68>:  ldr    x0, [x8, #0x6a0]
    0x1979edccc <+72>:  adrp   x8, 307057
    0x1979edcd0 <+76>:  add    x1, x8, #0xf29
    0x1979edcd4 <+80>:  bl     0x196462e28               ; symbol stub for: objc_msgSend
    0x1979edcd8 <+84>:  adrp   x8, 303537
    0x1979edcdc <+88>:  adrp   x2, 329398
    0x1979edce0 <+92>:  add    x2, x2, #0x948            ; @"NSUserDefaultsDidChangeNotification"
    0x1979edce4 <+96>:  add    x1, x8, #0x1a8
    0x1979edce8 <+100>: mov    x3, x19
```

```
The "write" command has been loaded and is ready for use.
warning: failed to set breakpoint site at 0x10271800c for breakpoint 49.1: error: 9 sending the
breakpoint request
(lldb) reg r x0 x1 x2 x3
      x0 = 0x00000002833f0090
      x1 = 0x00000001e1e3f9ce
      x2 = 0x00000002833e3de0
      x3 = 0x9338b7f313bae89a
(lldb) po $x0
<NSUserDefaults: 0x2833f0090>
(lldb) s/x $x1
error: the 'thread step-in' command doesn't support the --gdb-format option
(lldb) x/s $x1
0x1e1e3f9ce: "setObject:forKey:"
(lldb) po $x2
TSze6UEgX1dyX6w+brioKA==
(lldb) po $x3
gen49

(lldb)
```

- 辅助
  - Console.app=控制台
    - 另外，hook代码中加上的 `os_log` 日志打印，还可以单独在 `Console.app` 中看到：

- 

```
09:39:01.442922+0800  WhatsApp  hook_c.xm fstat: isGetPathOk=True, parsedPath=/private/var/mobile/Containers/Shared/AppGroup/D70C70AC-D347-4640-9667-1AA86D05CB65/connection.dlock
09:39:01.443257+0800  WhatsApp  hook_c.xm stat: pathname=/var/db/timezone/icutz/icutz44l.dat, buf=0x16dc58b70
09:39:01.443823+0800  WhatsApp  hook_c.xm stat: pathname=/var/db/timezone/icutz/icutz44l.dat, buf=0x16dc589a0
09:39:01.444000+0800  WhatsApp  hook_c.xm open: path=/private/var/mobile/Containers/Shared/AppGroup/D70C70AC-D347-4640-9667-1AA86D05CB65/shared_storage.lock, oflag=513
09:39:01.444113+0800  WhatsApp  hook_c.xm open: path=/var/db/timezone/icutz/icutz44l.dat, oflag=0
09:39:01.452356+0800  WhatsApp  hook_c.xm stat: pathname=/private/var/mobile/Containers/Shared/AppGroup/D70C70AC-D347-4640-9667-1AA86D05CB65/Logs/WhatsApp_2, buf=0x16dc5a2f0
09:39:01.454137+0800  WhatsApp  hook_c.xm open: path=/System/Library/CoreServices/SystemVersion.plist, oflag=0
09:39:01.454231+0800  WhatsApp  hook_c.xm fstat: fd=9, buf=0x16d658d90
09:39:01.454327+0800  WhatsApp  hook_c.xm fstat: isGetPathOk=True, parsedPath=/System/Library/CoreServices/SystemVersion.plist
09:39:01.455351+0800  WhatsApp  hook_c.xm stat: pathname=/var/mobile/Library/ConfigurationProfiles/PublicInfo/MCMeta.plist, buf=0x16d658ac0
09:39:01.455612+0800  WhatsApp  hook_c.xm lstat: path=/private/var/containers/Shared/SystemGroup/systemgroup.com.apple.configurationprofiles, buf=0x16d6585e0
09:39:01.456549+0800  WhatsApp  hook_c.xm open: path=/private/var/containers/Shared/SystemGroup/systemgroup.com.apple.configurationprofiles/Library/ConfigurationProfiles/PublicInfo/MCMeta.plist, oflag=0
09:39:01.458123+0800  WhatsApp  hook_c.xm fstat: fd=9, buf=0x16d658e40
09:39:01.458255+0800  WhatsApp  hook_c.xm fstat: isGetPathOk=True, parsedPath=/private/var/containers/Shared/SystemGroup/systemgroup.com.apple.configurationprofiles/Library/ConfigurationProfiles/PublicInfo/MCMeta.pli
09:39:01.459013+0800  WhatsApp  hook_c.xm open: path=/var/mobile/Library/UserConfigurationProfiles/PublicInfo/MCMeta.plist, oflag=0
09:39:01.460999+0800  WhatsApp  hook_c.xm fstat: fd=9, buf=0x16d658e40
09:39:01.461093+0800  WhatsApp  hook_c.xm fstat: isGetPathOk=True, parsedPath=/private/var/mobile/Library/UserConfigurationProfiles/PublicInfo/MCMeta.plist
09:39:01.461432+0800  WhatsApp  hook_c.xm access: path=/var/mobile/Library/UserConfigurationProfiles/EffectiveUserSettings.plist, amode=0x4
09:39:01.462183+0800  WhatsApp  hook_c.xm open: path=/var/mobile/Library/UserConfigurationProfiles/EffectiveUserSettings.plist, oflag=0
09:39:01.462826+0800  WhatsApp  hook_c.xm fstat: fd=9, buf=0x16d658d10
09:39:01.462888+0800  WhatsApp  hook_c.xm fstat: isGetPathOk=True, parsedPath=/private/var/mobile/Library/UserConfigurationProfiles/EffectiveUserSettings.plist
09:39:01.463814+0800  WhatsApp  hook_objc.xm NSUserDefaults$initWithSuiteName$: suitename=/var/mobile/Library/Preferences/com.apple.mt -> newUsrDef=<NSUserDefaults: 0x2833e6dc0>
09:39:01.467140+0800  WhatsApp  hook_objc.xm NSUserDefaults$objectForKey$: defaultName=KeepAppsUpToDateAppList -> val=(null)
09:39:01.467302+0800  WhatsApp  hook_reg.xm WAPreferencesBase$resolveInstanceMethod$: selNsstr=appWasLastForceQuit -> initOk=True
09:39:01.467430+0800  WhatsApp  hook_reg.xm WAPreferencesBase$resolveInstanceMethod$: selNsstr=appSafeModeAppVersion -> initOk=True
09:39:01.467545+0800  WhatsApp  hook_objc.xm NSUserDefaults$objectForKey$: defaultName=gen49 -> val=(null)
09:39:01.467700+0800  WhatsApp  hook_reg.xm WAPreferencesBase$resolveInstanceMethod$: selNsstr=setAppSafeModeAppVersion: -> initOk=True
09:39:01.469207+0800  WhatsApp  hook_objc.xm NSUserDefaults$setObject$forKey$: defaultName=gen49, value=TSze6UEgX1dyX6w+brioKA==
```

```
WhatsApp (HookWhatsApp.dylib)                                                易失
子系统: -- 类别: <缺少描述> 详细信息                              2023-12-27 09:39:01.469207+0800

hook_objc.xm NSUserDefaults$setObject$forKey$: defaultName=gen49, value=TSze6UEgX1dyX6w+brioKA==
```

-> 如此即可实现：用Xcode去方便的、深入的，调试每一行代码，以及搞懂hook代码和汇编代码的底层逻辑和细节。

附录

下面列出相关参考资料。

如何查看进程PID

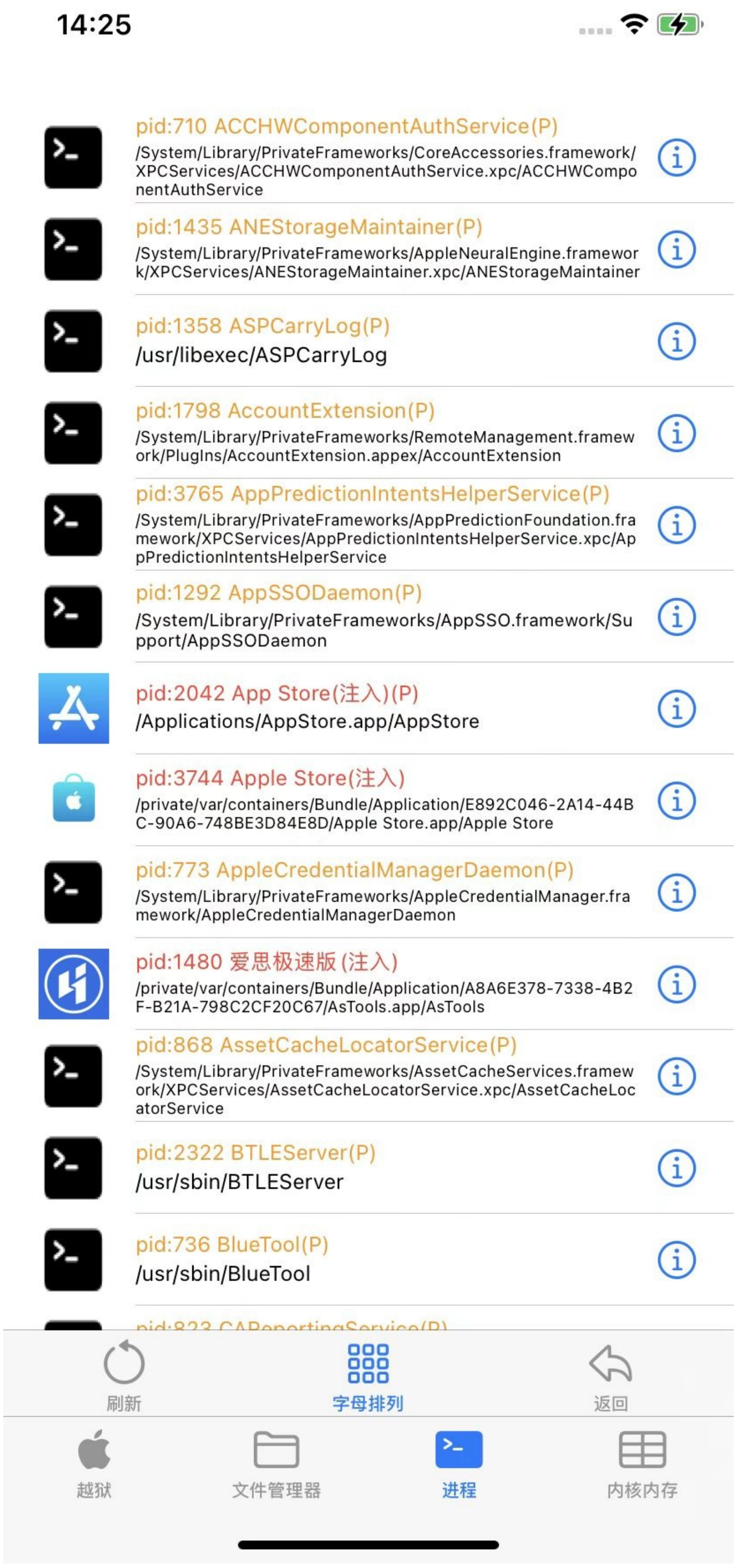
- 如何查看进程PID
  - 用 `ssh` 中 `ps` 查看PID
    - 举例
      - akd

        ```
        iPhone11-151:~ root# ps -A | grep akd
          125 ??         0:36.25 /System/Library/PrivateFrameworks/AuthKit.frame
        work/akd
          352 ??         0:01.53 /var/bin/jailbreakd_safe
          353 ??         0:07.82 /var/bin/jailbreakd
        14933 ttys001    0:00.01 grep akd
        ```

        -> akd的PID是 `125`
  - 用 `frida-ps` 查看PID

    ```
    crifan@licrifandeMacBook-Pro ~/dev/dev_src/ios_reverse/dumpdecrypted/stefaness
    er/dumpdecrypted ╲ master ● frida-ps -U | grep Store
    2042   App Store
    3744   Apple Store
    1559   TrollStore
    ```

    -> `Apple Store` 的PID是 `3744`
  - 用越狱后的工具查看
    - `XinaA15` -> `进程`
      -



14:25

pid:710 ACCHWComponentAuthService(P)
/System/Library/PrivateFrameworks/CoreAccessories.framework/XPCServices/ACCHWComponentAuthService.xpc/ACCHWComponentAuthService

pid:1435 ANEStorageMaintainer(P)
/System/Library/PrivateFrameworks/AppleNeuralEngine.framework/XPCServices/ANEStorageMaintainer.xpc/ANEStorageMaintainer

pid:1358 ASPCarryLog(P)
/usr/libexec/ASPCarryLog

pid:1798 AccountExtension(P)
/System/Library/PrivateFrameworks/RemoteManagement.framework/PlugIns/AccountExtension.appex/AccountExtension

pid:3765 AppPredictionIntentsHelperService(P)
/System/Library/PrivateFrameworks/AppPredictionFoundation.framework/XPCServices/AppPredictionIntentsHelperService.xpc/AppPredictionIntentsHelperService

pid:1292 AppSSODaemon(P)
/System/Library/PrivateFrameworks/AppSSO.framework/Support/AppSSODaemon

pid:2042 App Store(注入)(P)
/Applications/AppStore.app/AppStore

pid:3744 Apple Store(注入)
/private/var/containers/Bundle/Application/E892C046-2A14-44BC-90A6-748BE3D84E8D/Apple Store.app/Apple Store

pid:773 AppleCredentialManagerDaemon(P)
/System/Library/PrivateFrameworks/AppleCredentialManager.framework/AppleCredentialManagerDaemon

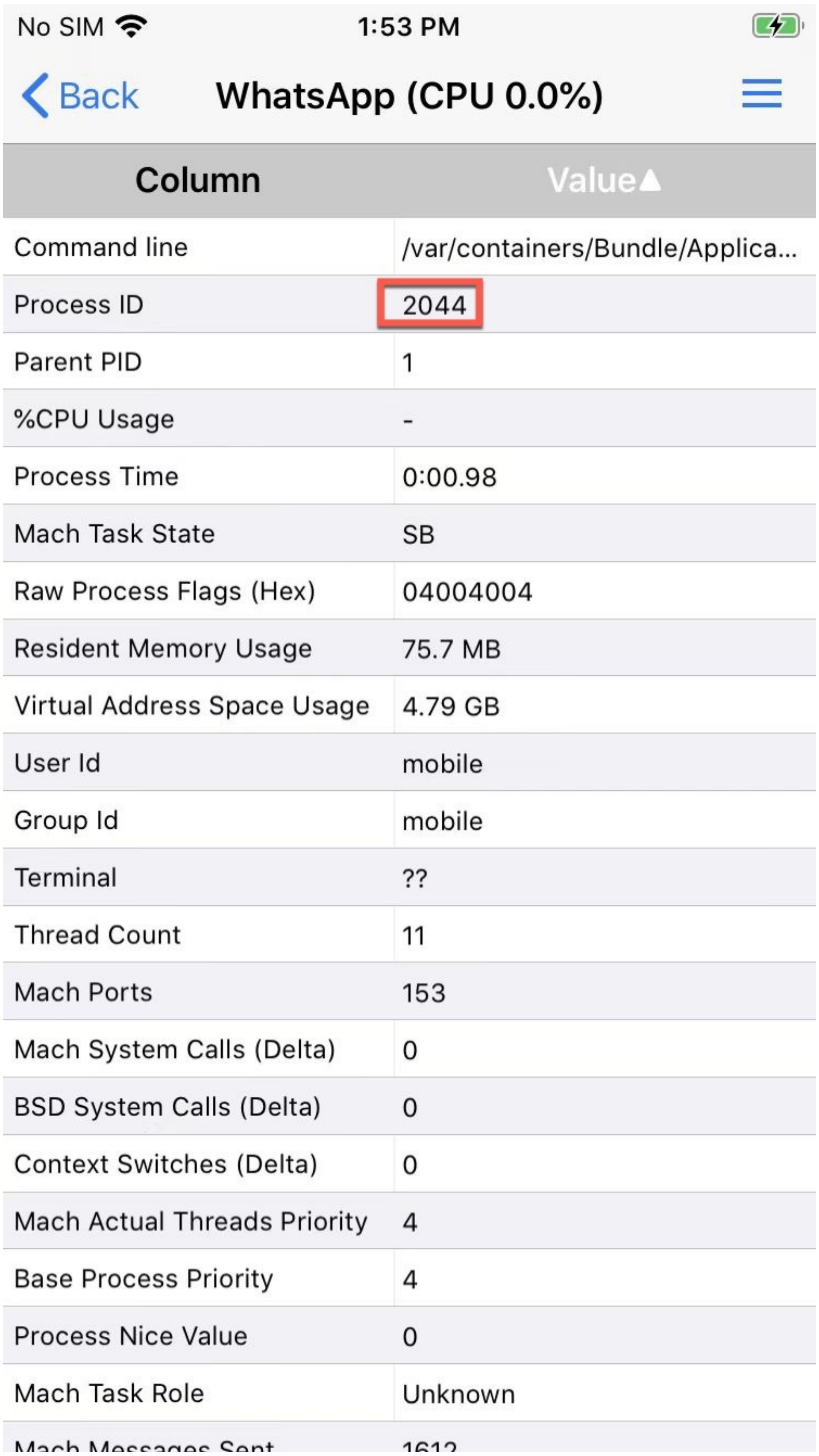
pid:1480 爱思极速版 (注入)
/private/var/containers/Bundle/Application/A8A6E378-7338-4B2F-B21A-798C2CF20C67/AsTools.app/AsTools

pid:868 AssetCacheLocatorService(P)
/System/Library/PrivateFrameworks/AssetCacheServices.framework/XPCServices/AssetCacheLocatorService.xpc/AssetCacheLocatorService

pid:2322 BTLEServer(P)
/usr/sbin/BTLEServer

pid:736 BlueTool(P)
/usr/sbin/BlueTool

- CocoaTop64



Column	Value▲
Command line	/var/containers/Bundle/Applica...
Process ID	2044
Parent PID	1
%CPU Usage	-
Process Time	0:00.98
Mach Task State	SB
Raw Process Flags (Hex)	04004004
Resident Memory Usage	75.7 MB
Virtual Address Space Usage	4.79 GB
User Id	mobile
Group Id	mobile
Terminal	??
Thread Count	11
Mach Ports	153
Mach System Calls (Delta)	0
BSD System Calls (Delta)	0
Context Switches (Delta)	0
Mach Actual Threads Priority	4
Base Process Priority	4
Process Nice Value	0
Mach Task Role	Unknown
Mach Messages Sent	1612

进程的csflags的定义

- 来源
  - cs_blobs.h (apple.com)
  - codesign.h (apple.com)
- 常用定义

```
#define CS_GET_TASK_ALLOW           0x00000004  /* has get-task-allow entitlement */
#define CS_ENTITLEMENTS_VALIDATED   0x00004000  /* code signature permits restricted en
titlements */
#define CS_PLATFORM_BINARY          0x04000000  /* this is a platform binary */
#define CS_DEBUGGED                 0x10000000  /* process is currently or has previous
ly been debugged and allowed to run with invalid pages */
```

- 全部定义

```
#ifndef _KERN_CODESIGN_H_
#define _KERN_CODESIGN_H_

/* code signing attributes of a process */
#define CS_VALID                    0x00000001  /* dynamically valid */
#define CS_ADHOC                    0x00000002  /* ad hoc signed */
#define CS_GET_TASK_ALLOW           0x00000004  /* has get-task-allow entitlement */
#define CS_INSTALLER                0x00000008  /* has installer entitlement */

#define CS_FORCED_LV                0x00000010  /* Library Validation required by Harde
ned System Policy */
#define CS_INVALID_ALLOWED          0x00000020  /* (macOS Only) Page invalidation allow
ed by task port policy */

#define CS_HARD                     0x00000100  /* don't load invalid pages */
#define CS_KILL                     0x00000200  /* kill process if it becomes invalid */

#define CS_CHECK_EXPIRATION         0x00000400  /* force expiration checking */
#define CS_RESTRICT                 0x00000800  /* tell dyld to treat restricted */

#define CS_ENFORCEMENT              0x00001000  /* require enforcement */
#define CS_REQUIRE_LV               0x00002000  /* require library validation */
#define CS_ENTITLEMENTS_VALIDATED   0x00004000  /* code signature permits restricted en
titlements */
#define CS_NVRAM_UNRESTRICTED       0x00008000  /* has com.apple.rootless.restricted-nv
ram-variables.heritable entitlement */

#define CS_RUNTIME                  0x00010000  /* Apply hardened runtime policies */

#define CS_ALLOWED_MACHO            (CS_ADHOC | CS_HARD | CS_KILL | CS_CHECK_EXPIRATION
| \
                                     CS_RESTRICT | CS_ENFORCEMENT | CS_REQUIRE_LV | CS_
RUNTIME)

#define CS_EXEC_SET_HARD            0x00100000  /* set CS_HARD on any exec'ed process */
```

```
#define CS_EXEC_SET_KILL            0x00200000  /* set CS_KILL on any exec'ed process */

#define CS_EXEC_SET_ENFORCEMENT     0x00400000  /* set CS_ENFORCEMENT on any exec'ed process */
#define CS_EXEC_INHERIT_SIP         0x00800000  /* set CS_INSTALLER on any exec'ed process */

#define CS_KILLED                   0x01000000  /* was killed by kernel for invalidity */
#define CS_DYLD_PLATFORM            0x02000000  /* dyld used to load this is a platform binary */
#define CS_PLATFORM_BINARY          0x04000000  /* this is a platform binary */
#define CS_PLATFORM_PATH            0x08000000  /* platform binary by the fact of path (osx only) */

#define CS_DEBUGGED                 0x10000000  /* process is currently or has previously been debugged and allowed to run with invalid pages */
#define CS_SIGNED                   0x20000000  /* process has a signature (may have gone invalid) */
#define CS_DEV_CODE                 0x40000000  /* code is dev signed, cannot be loaded into prod signed code (will go away with rdar://problem/28322552) */
#define CS_DATAVAULT_CONTROLLER     0x80000000  /* has Data Vault controller entitlement */

#define CS_ENTITLEMENT_FLAGS        (CS_GET_TASK_ALLOW | CS_INSTALLER | CS_DATAVAULT_CONTROLLER | CS_NVRAM_UNRESTRICTED)

/* executable segment flags */
#define CS_EXECSEG_MAIN_BINARY          0x1     /* executable segment denotes main binary */
#define CS_EXECSEG_ALLOW_UNSIGNED       0x10    /* allow unsigned pages (for debugging) */
#define CS_EXECSEG_DEBUGGER             0x20    /* main binary is debugger */
#define CS_EXECSEG_JIT                  0x40    /* JIT enabled */
#define CS_EXECSEG_SKIP_LV              0x80    /* OBSOLETE: skip library validation */
#define CS_EXECSEG_CAN_LOAD_CDHASH      0x100   /* can bless cdhash for execution */
#define CS_EXECSEG_CAN_EXEC_CDHASH      0x200   /* can execute blessed cdhash */
...
```

参考资料

- 手动重签名 · iOS逆向开发：签名和权限 (crifan.org)
- unc0ver
- XinaA15
- iOS逆向开发：动态调试
- debugserver+lldb
- 确保debugserver权限 · iOS逆向调试：debugserver+lldb (crifan.org)
-
- 【记录】Xcode用debugserver调试报错Not allowed to attach to process的核心原因：task_for_pid()调用失败
- 【未解决】Xcode调试Preferences报错：Could not attach to pid attach failed Not allowed to attach to process
- 【已解决】用新版1.1.8的XinaA15重新越狱
- 【记录】看看新版1.1.8的XinaA15各个功能和界面
- 【未解决】iOS逆向：如何查看进程详细信息
- 【已解决】iOS逆向进程可调试：给akd加上可调试的entitlement权限
- 【未解决】iOS逆向：用插件XcodeRootDebug实现debugserver允许调试任意进程
- 【整理】Xcode去Attach挂载调试app或二进制：通过PID或进程名
- 【未解决】iOS逆向：XCode去Attach调试启动时间有快有慢
- 【未解决】iOS逆向akd：动态调试
- 【已解决】iOS逆向akd：手动启动akd服务进程
- 【已解决】iOS逆向akd：用Xcode直接去调试akd的进程
- 【已解决】iOS逆向：查看越狱iPhone中Apple Store进程
- 【记录】用CocoaTop查看WhatsApp进程的flag属性
- 【未解决】iOS逆向：寻找可用的jailbreakd_client用于给进程加可调试权限
- 【未解决】iOS逆向：用jailbreakd_client给debugserver去加上entitle和platformize
- 【未解决】iOS逆向：iPhone中如何修改DeveloperDiskImage.dmg中的/Developer/usr/bin/debugserver
- 【记录】Xcode用debugserver调试报错Not allowed to attach to process的核心原因：task_for_pid()调用失败
- 【未解决】越狱iPhone中允许Xcode调试任意进程=
- 【未解决】iOS逆向：借助Frida实现任意目标进程可调试
- 【未解决】iOS逆向：允许进程被调试的csflags中的CS_GET_TASK_ALLOW
- 【已解决】iOS中进程的flag定义
- 【未解决】iOS进程中csflags中有CS_GET_TASK_ALLOW也还是无法被调试
- 【未解决】iOS逆向：iPhone中如何修改Ramdisk中的/Developer/usr/bin/debugserver
- 【未解决】Xcode调试Preferences报错：Could not attach to pid attach failed Not allowed to attach to process
- 【已解决】Xcode调试报错：Failed to start remote service com.apple.debugserver on device
- 【已解决】Xcode调试抖音报错：Failed to start remote service com.apple.debugserver on device Please check your connection to your device
- 【已解决】Xcode调试iPhoneX中抖音报错：Failed to start remote service com.apple.debugserver on device

- 【已解决】XCode调试iPhone报错：Failed to start remote service com.apple.debugserver on device
-