目录

iOS逆向开发：签名和权限

- 最新版本： `v1.0`
- 更新时间： `20230715`

简介

介绍iOS逆向期间涉及到的codesign代码签名和entitlement权限等相关内容。先是给概览，再是entitlement权限，包括什么是entitlement权限，如何用ldid和codesign查看权限、以及常见的权限，比如get-task-allow、task_for_pid-allow、platform-application等等；接着介绍如何更改权限；然后介绍codesign代码签名，先介绍什么是codesign，然后介绍如何用codesign命令行工具查看签名信息，以及如何用ldid和codesign重新签名；接着介绍常用工具，包括ldid、codesign、security；接着介绍常见的问题，包括运行时崩溃killed、任意进程可调试，包括XinaA15自带支持和自己手动重签名、安装app时的签名问题、MonkeyDev调试时崩溃等，最后附录中贴上参考资料。

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_codesign_ent: iOS逆向开发：签名和权限

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：签名和权限 book.crifan.org
- iOS逆向开发：签名和权限 crifan.github.io

离线下载阅读

- iOS逆向开发：签名和权限 PDF
- iOS逆向开发：签名和权限 ePub
- iOS逆向开发：签名和权限 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

签名和权限概览

iOS逆向期间，在各种操作时，常会涉及到：

- `codesign` = `代码签名` = `sign` = `签名`
  - 安装ipa时
  - 重签名时
- `entitlement` = `权限`
  - 加权限再重签名时

相关内容，此处去整理相关心得。

entitlement权限

什么是entitlement权限

- `entitlement` = 权限
  -

查看权限

- 查看权限
  - 常用工具：`codesign`、`ldid`
  - 具体命令
    - `codesign`
      - 二进制文件

        ```
        codesign -d --entitlements - BinaryFile
        ```

        - 举例

          ```
          codesign -d --entitlements - debugserver
          codesign -d --entitlements - amsaccountsd_orig
          ```

      - app

        ```
        codesign -d --entitlements - xxx.app
        ```

        - 举例

          ```
          codesign -d --entitlements - Preferences.app
          ```

    - `ldid`

      ```
      ldid -e BinaryFile
      ```

      - 举例

        ```
        ldid -e debugserver
        ldid -e amsaccountsd
        ldid -e Preferences.app/Preferences
        ```

常见权限

- 注：iOS的app中，各种entitlement权限，都可以在官网文档中找到
  - Entitlements | Apple Developer Documentation

get-task-allow

task_for_pid-allow

platform-application

run-unsigned-code

proc_info-allow

seatbelt-profiles

com.apple.security相关

network相关

之前是遇到：

给debugserver重签名之前，要删除原entitlement中的：

- `com.apple.security.network.server` 和 `com.apple.security.network.client`
  - 目的：防止后续lldb调试报错Failed to get connection from a remote gdb process

官网解释

- com.apple.security.network.server
  - A Boolean value indicating whether your app may listen for incoming network connections.
- com.apple.security.network.client
  - A Boolean value indicating whether your app may open outgoing network connections.

com.apple.security.application-groups

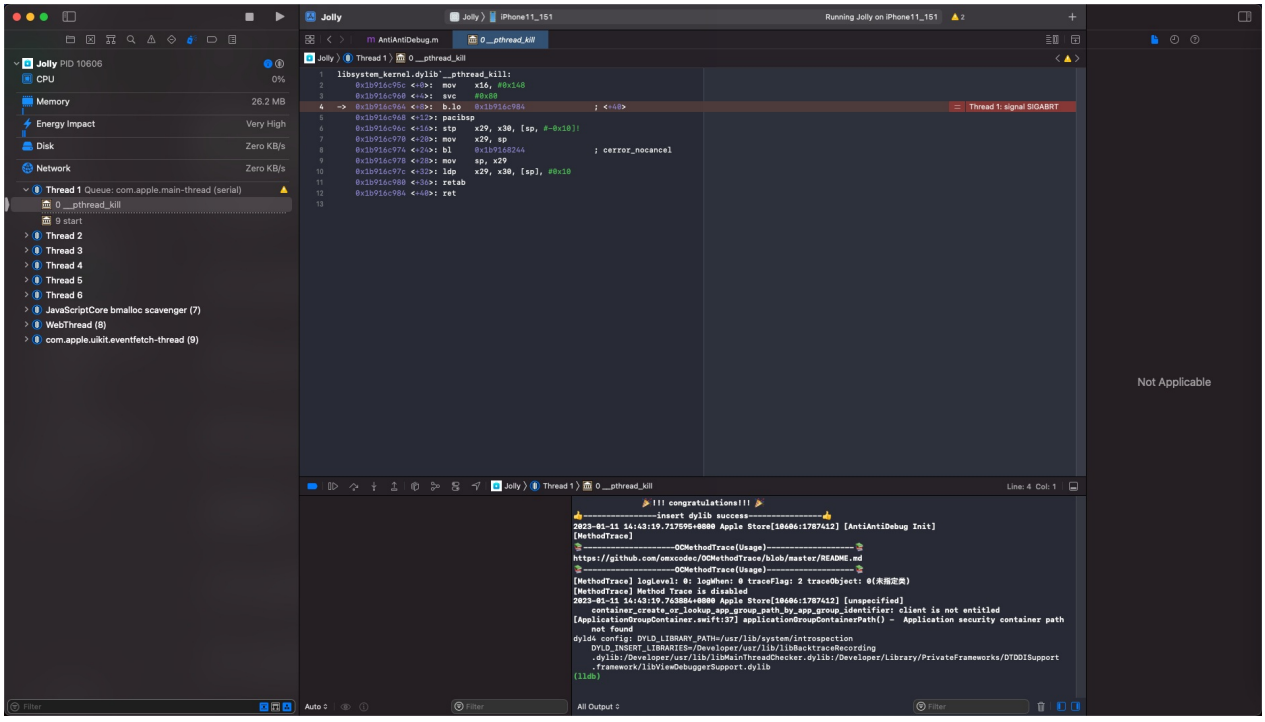
iOS的app：`Apple Store`，内置权限是：

```
    <key>com.apple.security.application-groups</key>
    <array>
        <string>group.com.apple.store.Jolly</string>
    </array>
```

如果（比如想MonkeyDev重新签名安装砸壳后的ipa后，其内部重签名过程会导致）去除了`com.apple.security.application-groups`的entitlement权限

则后续app运行会报`container_create_or_lookup_app_group_path_by_app_group_identifier`的错：

```
2023-01-11 14:43:19.763884+0800 Apple Store[10606:1787412] [unspecified] container_create_or_lookup_app_group_path_by_app_group_identifier: client is not entitled
[ApplicationGroupContainer.swift:37] applicationGroupContainerPath() -  Application security container path not found
```



官网解释

- com.apple.security.application-groups
  - Key
    - com.apple.security.application-groups
  - Type
    - Array of strings
  - Discussion

- App groups allow multiple apps produced by a single development team to access shared containers and communicate using interprocess communication (IPC). Apps may belong to one or more app groups.
- For iOS, format the identifier as follows:

```
group. group name
```

- For macOS:

```
team identifier . group name
```

- Apps within an app group share access to a group container. For more information about container creation, location, and deletion, see containerURLForSecurityApplicationGroupIdentifier:.
- Apps within a group can communicate with other members in the group using IPC mechanisms including Mach IPC, POSIX semaphores and shared memory, and UNIX domain sockets. In macOS, use app groups to enable IPC communication between two sandboxed apps, or between a sandboxed app and a non-sandboxed app.
- App groups also act as keychain access groups. For more information about the relationship between app groups and keychain access groups, see Sharing Access to Keychain Items Among a Collection of Apps.
- To add this entitlement to your app, enable the App Groups capability in Xcode, and add the groups your app belongs to.

更改权限

codesign

- 附录
  - 其他
    - 重签名工具
      - vtky/resign: XCode Project to resign .ipa files

什么是codesign代码签名

iOS的app，做了很多安全防护，其中之一就是：`codesign` = `代码签名`

加了代码签名，可以防止未经授权的app程序运行。只有通过签名校验的app才能正常运行。

增强了iOS系统的安全性，对应的，增加了iOS逆向的难度。

- app内部就包含 `Codesignature` 签名信息
    -

查看签名

- 查看代码签名
  - 常用工具：`codesign`
  - 具体命令

```
codesign -vv -d iOSBinaryFile
codesign -vv -d xxx.App
```

举例

jtool2_orig

```
➜  jtool2 codesign -vv -d jtool2_orig
Executable=/Users/crifan/dev/dev_tool/reverse_security/iOS/jtool2/jtool2/jtool2_orig
Identifier=jtool2.arm64e
Format=Mach-O universal (x86_64 arm64e arm64)
CodeDirectory v=20001 size=6446 flags=0x0(none) hashes=193+5 location=embedded
jtool2_orig: no signature
Info.plist=not bound
TeamIdentifier=not set
Sealed Resources=none
Internal requirements count=0 size=12
```

debugserver

```
crifan@licrifandeMacBook-Pro~/dev/dev_root/iosReverse/AppleStore/dynamicDebug/debugse
rver_lldb codesign -d --entitlements - debugserver
Executable=/Users/crifan/dev/dev_root/iosReverse/AppleStore/dynamicDebug/debugserver_ll
db/debugserver
debugserver: no signature
qqJ <?xml version="1.0" encoding="UTF-8"?>
```

- 说明
  - `no signature`：表示没有代码签名

Preferences

Preferences.app

```
➜  Preferences.app pwd
/Users/crifan/dev/dev_root/iosReverse/AppleStore/Preferences_app/ipa/Payload/Preference
s.app
➜  Preferences.app cd ..
➜  Payload ll
```

```
total 0
drwxr-xr-x@ 50 crifan  staff   1.6K  3  7 14:46 Preferences.app
➜  Payload codesign -vv -d Preferences.app
Executable=/Users/crifan/dev/dev_root/iosReverse/AppleStore/Preferences_app/ipa/Payload
/Preferences.app/Preferences
Identifier=com.apple.Preferences
Format=app bundle with Mach-O thin (arm64)
CodeDirectory v=20400 size=1326 flags=0x2(adhoc) hashes=31+7 location=embedded
Signature=adhoc
Info.plist entries=39
TeamIdentifier=not set
Sealed Resources version=2 rules=9 files=1
Internal requirements count=0 size=12
```

Preferences.app/Preferences

```
➜  Payload codesign -vv -d Preferences.app/Preferences
Executable=/Users/crifan/dev/dev_root/iosReverse/AppleStore/Preferences_app/ipa/Payload
/Preferences.app/Preferences
Identifier=com.apple.Preferences
Format=app bundle with Mach-O thin (arm64)
CodeDirectory v=20400 size=1326 flags=0x2(adhoc) hashes=31+7 location=embedded
Signature=adhoc
Info.plist entries=39
TeamIdentifier=not set
Sealed Resources version=2 rules=9 files=1
Internal requirements count=0 size=12
```

Apple Store.app

```
➜  425AB24D-6133-4247-AF3A-7D7AE10B3FFC pwd
/Users/crifan/dev/dev_root/iosReverse/AppleStore/fromiPhone11/AppleStore_official_inite
d/1st/Bundle/425AB24D-6133-4247-AF3A-7D7AE10B3FFC
➜  425AB24D-6133-4247-AF3A-7D7AE10B3FFC ll
total 16
drwxr-xr-x@ 70 crifan  staff   2.2K  1 31 10:54 Apple Store.app
-rw-r--r--@  1 crifan  staff   654B  1 31 10:55 BundleMetadata.plist
-rw-r--r--@  1 crifan  staff   1.4K  1 31 10:55 iTunesMetadata.plist

➜  425AB24D-6133-4247-AF3A-7D7AE10B3FFC codesign -vv -d Apple\ Store.app
Executable=/Users/crifan/dev/dev_root/iosReverse/AppleStore/fromiPhone11/AppleStore_off
icial_inited/1st/Bundle/425AB24D-6133-4247-AF3A-7D7AE10B3FFC/Apple Store.app/Apple Stor
e
Identifier=com.apple.store.Jolly
Format=app bundle with Mach-O thin (arm64)
CodeDirectory v=20500 size=2596 flags=0x0(none) hashes=35+7 location=embedded
Signature size=4390
Authority=Apple iPhone OS Application Signing
Authority=Apple iPhone Certification Authority
Authority=Apple Root CA
Info.plist entries=59
TeamIdentifier=APPLECOMPUTER
Sealed Resources version=2 rules=33 files=2422
```

```
Internal requirements count=1 size=104
```

重新签名

- `重新代码签名=重签名`
  - 何时需要去重签名
    - app的代码签名已过期
    - 想要更改app的权限
      - 比如
        - 加上可被调试相关entitlement权限后，重新签名打包，使其可调试
- 如何重新签名
  - 常用工具：`codesign`、`ldid`
  - 具体命令+举例
    - `codesign`

```
codesign -f -s - --entitlements debugserver.entitlements debugserver
```

      - 参数说明
        - `-f` == `--force`：`强制` 覆盖已有签名
        - `-s -`
          - `-s` == `--sign`
          - `-`：= `single letter` = `dash`，表示 `ad-hoc signing`
        - `debugserver.entitlements`：（经修改后加了进程可调试的的）新的权限文件
        - `debugserver`：目标文件=现有的二进制文件
    - `ldid`

```
ldid -Sdebugserver.entitlements debugserver
```

      - 参数说明
        - `-Sdebugserver.entitlements` = `-S` 不带空格的加上 `newEntitlementFile`
        - `debugserver`：目标文件=现有的二进制文件

常用工具

codesign

心得

security+codesign -> codesign

- 之前用：`security` + `codesign ...`

```
security find-identity -p codesigning
codesign --force --sign CDDC8C0D2F3A79EB17C183E14F799F75815E294E --entitlements deb
ugable_entitlement.xml --timestamp none --generate-entitlement-der debugserver_resi
gn
```

- 现在改用：`codesign ... -s - ...`

```
codesign --force --sign - --entitlements debugable_entitlement.xml --timestamp none
 --generate-entitlement-der debugserver_resign
```

  - 区别：`-` = `短横线`：表示`ad-hoc signing`，无需再去用`security`去寻找当前可用签名了
  - 可简写为

```
codesign -f -s - --entitlements debugserver.entitlements debugserver_resign
```

codesign的help语法

codesign(1) [osx man page]

```
CODESIGN(1)                          BSD General Commands Manual
       CODESIGN(1)

NAME

     codesign -- Create and manipulate code signatures

SYNOPSIS

     codesign -s identity [-i identifier] [-r requirements] [-fv] [path ...]
     codesign -v [-R requirement] [-v] [path|pid ...]
     codesign -d [-v] [path pid ...]
     codesign -h [-v] [pid ...]

DESCRIPTION

     The codesign command is used to create, check, and display code signatures, as wel
l as inquire into the dynamic status of signed code in the
     system.

     codesign requires exactly one operation option to determine what action is to be p
erformed, as well as any number of other options to modify
     its behavior. It can act on any number of objects per invocation, but performs the
 same operation on all of them.

     codesign accepts single-character (classic) options, as well as GNU-style long opt
ions of the form --name and --name=value. Common options
     have both forms; less frequent and specialized options have only long form.  Note
that the form --name value (without equal sign) will not
     work as expected on options with optional values.

OPTIONS

     The options are as follows:

     --all-architectures
         When verifying a code signature on code that has a universal ("fat") Mach-O bi
nary, separately verify each architecture contained.
         This is the default unless overridden with the -a (--architecture) option.

     -a, --architecture architecture
         When verifying or displaying signatures, explicitly select the Mach-O architec
ture given. The architecture can be specified either by
         name (e.g. i386) or by number; if by number, a sub-architecture may be appende
d separated by a comma.  This option applies only to
         Mach-O binary code and is ignored for other types.  If the path uses the Mach-
O format and contains no code of the given architec-
         ture, the command will fail.  The default for verification is --all-architectu
```

```
res, to verify all architectures present.  The default
         for display is to report on the native architecture of the host system.  When
signing, codesign will always sign all architectures
         contained in a universal Mach-O file.

     --bundle-version version-string
         When handling versioned bundles such as frameworks, explicitly specify the ver
sion to operate on. This must be one of the names in
         the "Versions" directory of the bundle.  If not specified, codesign uses the b
undle's default version.  Note that most frameworks
         delivered with the system have only one version, and thus this option is irrel
evant for them.  There is currently no facility for
         operating on all versions of a bundle at once.

     -d, --display
         Display information about the code at the path(s) given. Increasing levels of
verbosity produce more output.  The format is designed
         to be moderately easy to parse by simple scripts while still making sense to h
uman eyes.  In addition, the -r, --file-list,
         --extract-certificates, and --entitlements options can be used to retrieve add
itional information.

     -D, --detached filename
         When signing, designates that a detached signature should be written to the sp
ecified file. The code being signed is not modified and
         need not be writable.  When verifying, designates a file containing a detached
 signature to be used for verification. Any embedded
         signature in the code is ignored.

     --deep  When signing a bundle, specifies that nested code content such as helpers,
 frameworks, and plug-ins, should be recursively signed in
         turn. Beware that all signing options you specify will apply, in turn, to such
 nested content.
         When verifying a bundle, specifies that any nested code content will be recurs
ively verified as to its full content. By default, ver-
         ification of nested content is limited to a shallow investigation that may not
 detect changes to the nested code.
         When displaying a signature, specifies that a list of directly nested code sho
uld be written to the display output. This lists only
         code directly nested within the subject; anything nested indirectly will requi
re recursive application of the codesign command.

     --detached-database
         When signing, specifies that a detached signature should be generated as with
the --detached option, but that the resulting signature
         should be written into a system database, from where it is made automatically
available whenever apparently unsigned code is vali-
         dated on the system.
         Writing to this system database requires elevated process privileges that are
not available to ordinary users.

     -f, --force
         When signing, causes codesign to replace any existing signature on the path(s)
 given. Without this option, existing signatures will
         not be replaced, and the signing operation fails.
```

```
     -h, --hosting
         Constructs and prints the hosting chain of a running program. The pid argument
s must denote running code (pids etc.) With verbose
         options, this also displays the individual dynamic validity status of each ele
ment of the hosting chain.

     -i, --identifier identifier
         During signing, explicitly specify the unique identifier string that is embedd
ed in code signatures. If this option is omitted, the
         identifier is derived from either the Info.plist (if present), or the filename
 of the executable being signed, possibly modified by
         the --prefix option.  It is a very bad idea to sign different programs with th
e same identifier.

     -o, --options flag,...
         During signing, specifies a set of option flags to be embedded in the code sig
nature. The value takes the form of a comma-separated
         list of names (with no spaces). Alternatively, a numeric value can be used to
directly specify the option mask (CodeDirectory flag
         word). See OPTION FLAGS below.

     -P, --pagesize pagesize
         Indicates the granularity of code signing. Pagesize must be a power of two.  C
hunks of pagesize bytes are separately signed and can
         thus be independently verified as needed.    As a special case, a pagesize of
zero indicates that the entire code should be signed and
         verified as a single, possibly gigantic page. This option only applies to the
main executable and has no effect on the sealing of
         associated data, including resources.

     -r, --requirements requirements
         During signing, indicates that internal requirements should be embedded in the
 code path(s) as specified. See "specifying require-
         ments" below.  Defaults will be applied to requirement types that are not expl
icitly specified; if you want to defeat such a default,
         specify "never" for that type.
         During display, indicates where to write the code's internal requirements. Use
 -r- to write them to standard output.

     -R, --test-requirement requirement
         During verification, indicates that the path(s) given should be verified again
st the code requirement specified. If this option is
         omitted, the code is verified only for internal integrity and against its own
designated requirement.

     -s, --sign identity
         Sign the code at the path(s) given using this identity. See SIGNING IDENTITIES
 below.

     -v, --verbose
         Sets (with a numeric value) or increments the verbosity level of output. Witho
ut the verbose option, no output is produced upon suc-
         cess, in the classic UNIX style.  If no other options request a different acti
on, the first -v encountered will be interpreted as
         --verify instead (and does not increase verbosity).
```

```
     -v, --verify
         Requests verification of code signatures.    If other actions (sign, display,
etc.) are also requested, -v is interpreted to mean
         --verbose.

     --continue
         Instructs codesign to continue processing path arguments even if processing on
e fails.  If this option is given, exit due to opera-
         tional errors is deferred until all path arguments have been considered. The e
xit code will then indicate the most severe failure
         (or, with equal severity, the first such failure encountered).

     --dryrun
         During signing, performs almost all signing operations, but does not actually
write the result anywhere. Cryptographic signatures are
         still generated, actually using the given signing identity and triggering any
access control checks normally, though the resulting
         signature is then discarded.

     --entitlements path
         When signing, take the file at the given path and embed its contents in the si
gnature as entitlement data. If the data at path does
         not already begin with a suitable binary ("blob") header, one is attached auto
matically.
         When displaying a signature, extract any entitlement data from the signature a
nd write it to the path given. Use "-" to write to
         standard output.  By default, the binary "blob" header is returned intact; pre
fix the path with a colon ":" to automatically strip it
         off.  If the signature has no entitlement data, nothing is written (this is no
t an error).

     --extract-certificates prefix
         When displaying a signature, extract the certificates in the embedded certific
ate chain and write them to individual files. The
         prefix argument is appended with numbers 0, 1, ... to form the filenames, which
 can be relative or absolute. Certificate 0 is the
         leaf (signing) certificate, and as many files are written as there are certifi
cates in the signature. The files are in ASN.1 (DER)
         form.  If prefix is omitted, the default prefix is "codesign" in the current d
irectory.

     --file-list path
         When signing or displaying a signature, codesign writes to the given path a li
st of files that may have been modified as part of the
         signing process. This is useful for installer or patcher programs that need to
 know what was changed or what files are needed to make
         up the "signature" of a program. The file given is appended-to, with one line
per absolute path written. An argument of "-" (single
         dash) denotes standard output.  Note that the list may be somewhat pessimistic
 - all files not listed are guaranteed to be unchanged
         by the signing process, but some of the listed files may not actually have cha
nged.  Also note that changes may have been made to
         extended attributes of these files.

     --ignore-resources
         During static validation, do not validate the contents of the code's resources
```

```
.  In effect, this will pass validation on code whose
         resources have been corrupted (or inappropriately signed). On large programs,
it will also substantially speed up static validation,
         since all the resources will not be read into memory.  Obviously, the outcome
of such a validation should be considered on its mer-
         its.

     --keychain filename
         During signing, only search for the signing identity in the keychain file spec
ified. This can be used to break any matching ties if
         you have multiple similarly-named identities in several keychains on the user'
s search list.  Note that the standard keychain search
         path is still consulted while constructing the certificate chain being embedde
d in the signature.
         Note that filename will not be searched to resolve the signing identity's cert
ificate chain unless it is also on the user's keychain
         search list.

     --prefix string
         If no explicit unique identifier is specified (using the -i option), and if th
e implicitly generated identifier does not contain any
         dot (.) characters, then the given string is prefixed to the identifier before
 use. If the implicit identifier contains a dot, it is
         used as-is. Typically, this is used to deal with command tools without Info.pl
ists, whose default identifier is simply the command's
         filename; the conventional prefix used is com.domain. (note that the final dot
 needs to be explicit).

     --preserve-metadata=list
         When re-signing code that is already signed, reuse some information from the o
ld signature.  If new data is specified explicitly, it
         is preferred.  You still need to specify the -f (--force) option to enable ove
rwriting signatures at all.    If this option is absent,
         any old signature has no effect on the signing process.
         This option takes a comma-separated list of names, which you may reasonably ab
breviate:

         identifier      Preserve the signing identifier (--identifier) instead of gene
rating a default identifier.

         entitlements    Preserve the entitlement data (--entitlements).

         resource-rules  Preserve and reuse the resource rules (--resource-rules).

         requirements    Preserve the internal requirements (--requirements option), in
cluding any explicit Designated Requirement. Note that
                 all internal requirements are preserved or regenerated as a whole; you
 cannot pick and choose individual elements
                 with this option.
         For historical reasons, this option can be given without a value, which preser
ves all of these values as presently known. This use is
         deprecated and will eventually be removed; always specify an explicit list of
preserved items.

     --resource-rules filename
         During signing, this option overrides the default rules for identifying and co
```

```
llecting bundle resources and nested code to be sealed
         into the signature. The argument is the path to a property list (plist) file c
ontaining scanning and qualification instructions. See
         the code signing documentation for details.

     --timestamp [=URL]
         During signing, requests that a timestamp authority server be contacted to aut
henticate the time of signing. The server contacted is
         given by the URL value.  If this option is given without a value, a default se
rver provided by Apple is used.  Note that this server
         may not support signatures made with identities not furnished by Apple.  If th
e timestamp authority service cannot be contacted over
         the Internet, or it malfunctions or refuses service, the signing operation wil
l fail.
         If this option is not given at all, a system-specific default behavior is invo
ked.  This may result in some but not all code signa-
         tures being timestamped.
         The special value none explicitly disables the use of timestamp services.

OPERATION

     In the first synopsis form, codesign attempts to sign the code objects at the path
(s) given, using the identity provided. Internal
     requirements and entitlements are embedded if requested. Internal requirements not
 specified may be assigned suitable default values.
     Defaulting applies separately to each type of internal requirement.  If an identif
ier is explicitly given, it is sealed into all path(s).
     Otherwise, each path derives its identifier independently from its Info.plist or p
athname.  Code nested within bundle directories (as identi-
     fied from the applicable Resource Rules) must already be signed or the signing ope
ration will fail, unless the --deep option is given, in
     which case any unsigned nested code will be recursively signed before proceeding,
using the same signing options and parameters. If the
     --force option is given, any existing top-level signature is replaced, subject to
any --preserve-metadata options also present. Combining the
     --force and --deep options results in forcible replacement of all signatures withi
n the target bundle, as long as the resource rules identify
     them.

     In the second synopsis form, codesign verifies the code signatures on all the path
(s) given. The verification confirms that the code at those
     path(s) is signed, that the signature is valid, and that all sealed components are
 unaltered. If a requirement is given, each path is also
     checked against this requirement (but see DIAGNOSTICS below).  If verbose verifica
tion is requested, the program is also checked against its
     own designated requirement, which should never fail for a properly signed program.

     If a path begins with a decimal digit, it is interpreted as the process id of a ru
nning process in the system, and dynamic validation is per-
     formed on that process instead.  This checks the code's dynamic status and just en
ough static data to close the nominal security envelope.
     Add at least one level of verbosity to also perform a full static check.

     In the third synopsis form, codesign displays the contents of the signatures on th
e path(s) given. More information is displayed as the ver-
     bosity level increases.  This form may not completely verify the signatures on the
```

```
 path(s); though it may perform some verification steps in
     the process of obtaining information about the path(s).  If the -r path option is
given, internal requirements will be extracted from the
     path(s) and written to path; specify a dash "-" to write to standard output. If th
e code does not contain an explicit designated requirement,
     the implied one will be retrieved and written out as a source comment.  If the --e
ntitlements path option is given, embedded entitlement data
     will be extracted likewise and written to the file specified.

     In the fourth synopsis form, codesign constructs the hosting path for each pid giv
en and writes it, one host per line, to standard output.
     The hosting path is the chain of code signing hosts starting with the most specifi
c code known to be running, and ending with the root of
     trust (the kernel). If the --verbose option is given, the dynamic validity status
of each host is also displayed, separated from the path by
     a tab character.  Note that hosting chains can at times be constructed for invalid
 or even unsigned code, and the output of this form of the
     codesign command should not be taken as a statement of formal code validity. Only
codesign --verify can do that; and in fact, formal verifi-
     cation constructs the hosting chain as part of its operation (but does not display
 it).

SIGNING IDENTITIES

     To be used for code signing, a digital identity must be stored in a keychain that
is on the calling user's keychain search list.  All key-
     chain sources are supported if properly configured. In particular, it is possible
to sign code with an identity stored on a supported smart
     card.  If your signing identity is stored in a different form, you need to make it
 available in keychain form to sign code with it.
     If the --keychain argument is used, identity is only looked-for in the specific ke
ychain given. This is meant to help disambiguate references
     to identities.  Even in that case, the full keychain search list is still consulte
d for additional certificates needed to complete the signa-
     ture.

     The identity is first considered as the full name of a keychain identity preferenc
e.  If such a preference exists, it directly names the
     identity used.  Otherwise, the identity is located by searching all keychains for
a certificate whose subject common name (only) contains the
     identity string given. If there are multiple matches, the operation fails and no s
igning is performed; however, an exact match is preferred
     over a partial match.  These comparisons are case sensitive.  Multiple instances o
f the exactly same certificate in multiple keychains are
     tolerated as harmless.

     If identity consists of exactly forty hexadecimal digits, it is instead interprete
d as the SHA-1 hash of the certificate part of the desired
     identity.    In this case, the identity's subject name is not considered.

     Both identity preferences and certificate hashes can be used to identify a particu
lar signing identity regardless of name. Identity prefer-
     ences are global settings for each user and provide a layer of indirection. Certif
icate hashes are very explicit and local. These choices,
     combined with what is placed into Xcode project and target build variables and/or
script settings, allows for very flexible designation of
```

```
     signing identities.

     If identity is the single letter "-" (dash), ad-hoc signing is performed.    Ad-ho
c signing does not use an identity at all, and identifies
     exactly one instance of code. Significant restrictions apply to the use of ad-hoc
signed code; consult documentation before using this.

     codesign will attempt to embed the entire certificate chain documenting the signin
g identity in the code signature it generates, including
     any intermediate certificates and the anchor certificate. It looks for those in th
e keychain search list of the user performing the signing
     operation. If it cannot generate the entire certificate chain, signing may still s
ucceed, but verification may fail if the verifying code
     does not have an independent source for the missing certificates (from its keychai
ns).

SPECIFYING REQUIREMENTS

     The requirement(s) arguments (-r and -R) can be given in various forms. A plain te
xt argument is taken to be a path to a file containing the
     requirement(s).  codesign will accept both binary files containing properly compil
ed requirements code, and source files that are automati-
     cally compiled before use.  An argument of "-" requests that the requirement(s) ar
e read from standard input.  Finally, an argument that
     begins with an equal sign "=" is taken as a literal requirements source text, and
is compiled accordingly for use.

OPTION FLAGS

     When signing, a set of option flags can be specified to change the behavior of the
 system when using the signed code. The following flags are
     recognized by codesign; other flags may exist at the API level. Note that you can
specify any valid flags by giving a (single) numeric value
     instead of a list of option names.

     kill     Forces the signed code's kill flag to be set when the code begins executi
on.  Code with the kill flag set will die when it becomes
          dynamically invalid. It is therefore safe to assume that code marked this way
, once validated, will have continue to have a valid
          identity while alive.

     hard     Forces the signed code's hard flag to be set when the code begins executi
on.  The hard flag is a hint to the system that the code
          prefers to be denied access to resources if gaining such access would invalid
ate its identity.

     host     Marks the code as capable of hosting guest code. You must set this option
if you want the code to act as a code signing host, con-
          trolling subsidiary ("guest") code. This flag is set automatically if you spe
cify an internal guest requirement.

     expires  Forces any validation of the code to consider expiration of the certifica
tes involved. Code signatures generated with this flag will
          fail to verify once any of the certificates in the chain has expired, regardl
ess of the intentions of the verifier. Note that this
          flag does not affect any other checks that may cause signature validation to
```

```
fail, including checks for certificate revocation.

     Note that code can set the hard and kill flags on itself at any time. The signing
options only affect their initial state. Once set by any
     means, these flags cannot be cleared for the lifetime of the code. Therefore, spec
ifying such flags as signing options guarantees that they
     will be set whenever the signed code runs.

     If the code being signed has an Info.plist that contains a key named CSFlags, the
value of that key is taken as the default value for the
     options. The value of CSFlags can be a string in the same form as the --options op
tion, or an integer number specifying the absolute numeric
     value. Note however that while you can abbreviate flag names on the command lines,
 you must spell them out in the Info.plist.

EXAMPLES

     To sign application Terminal.app with a signing identity named "authority":
       codesign -s authority Terminal.app

     To sign the command-line tool "helper" with the same identity, overwriting any exi
sting signature, using the signing identifier
     "com.mycorp.helper", and embedding a custom designated requirement
       codesign -f -s authority --prefix=com.mycorp. -r="designated => anchor /tmp/foo"
 helper

     To verify the signature on Terminal.app and produce some verbose output:
       codesign --verify --verbose Terminal.app

     To verify the dynamic validity of process 666:
       codesign --verify 666

     To display all information about Terminal.app's code signature:
       codesign --display --verbose=4 Terminal.app

     To extract the internal requirements from Terminal.app to standard output:
       codesign --display -r- Terminal.app

DIAGNOSTICS

     codesign exits 0 if all operations succeed. This indicates that all codes were sig
ned, or all codes verified properly as requested. If a
     signing or verification operation fails, the exit code is 1. Exit code 2 indicates
 invalid arguments or parameters. Exit code 3 indicates
     that during verification, all path(s) were properly signed but at least one of the
m failed to satisfy the requirement specified with the -R
     option.

     For verification, all path arguments are always investigated before the program ex
its.  For all other operations, the program exits upon the
     first error encountered, and any further path arguments are ignored, unless the --
continue option was specified, in which case codesign will
     defer the failure exit until after it has attempted to process all path arguments
in turn.

SIGNING ATOMICITY
```

```
     When a signing operation fails for a particular code, the code may already have be
en modified in certain ways by adding requisite signature
     data. Such information will not change the operation of the code, and the code wil
l not be considered signed even with these pieces in place.
     You may repeat the signing operation without difficulty.  Note however that a prev
ious valid signature may have been effectively destroyed if
     you specified the -f option.
     If you require atomicity of signing stricter than provided by codesign, you need t
o make an explicit copy of your code and sign that.

ENVIRONMENT

     If the CODESIGN_ALLOCATE environment variable is set, it identifies a substitute c
odesign_allocate tool used to allocate space for code sig-
     natures in Mach-O binaries. This is used by Xcode SDK distributions to provide arc
hitectural support for non-native platforms such as
     iPhones.  The system will not accept such substitutes unless they are specially si
gned (by Apple).

FILES

     /var/db/DetachedSignatures  System-wide database of detached code signatures for u
nsigned code.

SEE ALSO

     csreq(1), xcodebuild(1), codesign_allocate(1)

HISTORY

     The codesign command first appeared in Mac OS 10.5.0 (Leopard).

BUGS

     Some options only apply to particular operations, and codesign ignores them (witho
ut complaining) if you specify them for an operation for
     which they have no meaning.

     The --preserve-metadata option used to take no value, and varied across releases in
 what exactly it preserved. The ensuing confusion is still
     with you if you need to make backward-compatible scripts.

     The dual meaning of the -v option, indicating either verbosity or verification, co
nfuses some people. If you find it confusing, use the unam-
     biguous long forms --verbose and --verify instead.

NOTES

     The Xcode build system invokes codesign automatically if the CODE_SIGN_IDENTITY bu
ild variable is set.  You can express any combination of
     codesign options with additional build variables there.

     codesign is fundamentally a shell around the code signing APIs, and performs nothi
ng of the underlying work.  Replacing it with older or
     newer versions is unlikely to have a useful effect.
```

```
    codesign has several operations and options that are purposely left undocumented in
 this manual page because they are either experimental
    (and subject to change at any time), or unadvised to the unwary.  The interminably
 curious are referred to the published source code.

BSD
                                    May 7, 2011                                    BSD
```

ldid

- ldid
  - 作用
    - 查看和设置entitlement权限
    - 伪签名
  - 源码
    - saurik
      - git://git.saurik.com/ldid.git
      - https://gitlab.com/opensource-saurik/ldid
    - other fork
      - https://github.com/xerub/ldid

基本用法

- dumps the binary's entitlements

```
ldid -e <binary>
```

- sets the binary's entitlements, where ent.xml is the path to an entitlements file

```
ldid -Sent.xml <binary>
```

- pseudo-signs a binary with no entitlements

```
ldid -S <binary>
```

举例

- if you wanted to add the entitlements in a file on your desktop named Entitlements.xml to debugserver in your current directory, you would run

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

心得

二进制有几种架构，ldid就输出几份权限信息

之前已知，查看现有entitlement权限，有2种方式：

- ldid

```
ldid -e debugserver
```

- codesign

```
codesign -d --entitlements - debugserver
```

不过其中有个细节，后来才彻底明白：

- 有时候ldid会输出多份权限信息
  - 原因：二进制中有几个arch，即几种架构，即二进制是FAT，包含多个架构，包含了几个架构，ldid就输出几份权限信息
  - 对比：即使对于FAT的多个架构，`codesign`也只输出1份entitlement权限信息

举例：

```
➜  jtool2 pwd
/Users/crifan/dev/dev_tool/reverse_security/iOS/jtool2/jtool2
➜  jtool2 ll
total 10696
-rw-r--r--@ 1 crifan  staff    28K  2 10  2020 WhatsNew.txt
-rwxr-xr-x@ 1 crifan  staff   329K 12 22  2020 disarm
-rwxr-xr-x@ 1 crifan  staff   2.5M  7  3 09:36 jtool2
-rw-r--r--@ 1 crifan  staff   321B  7  3 09:35 jtool2.entitlements
-rwxr-xr-x@ 1 crifan  staff   2.4M  7  3 09:33 jtool2_orig
-rw-r--r--@ 1 crifan  staff    15K 10 22  2022 matchers.txt

➜  jtool2 file jtool2
jtool2: Mach-O universal binary with 3 architectures: [x86_64:Mach-O 64-bit executable
x86_64] [arm64:Mach-O 64-bit executable arm64] [arm64e]
jtool2 (for architecture x86_64):    Mach-O 64-bit executable x86_64
jtool2 (for architecture arm64):    Mach-O 64-bit executable arm64
jtool2 (for architecture arm64e):    Mach-O 64-bit executable arm64e
```

即3份架构`x86_64`、`arm64`、`arm64e`的`FAT`多架构的`jtool2`，则此处`ldid`就会输出3份entitlement权限信息：

```
➜  jtool2 ldid -e jtool2
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE plist PUBLIC "-//Apple//DTD PLIST 1.0//EN" "http://www.apple.com/DTDs/Propert
yList-1.0.dtd">
<plist version="1.0">
<dict>
    <key get-task-allow /key>
    <true/>
    <key task_for_pid-allow /key>
    <true/>
    <key run-unsigned-code /key>
    <true/>
</dict>
</plist>
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE plist PUBLIC "-//Apple//DTD PLIST 1.0//EN" "http://www.apple.com/DTDs/Propert
yList-1.0.dtd">
<plist version="1.0">
<dict>
    <key get-task-allow /key>
    <true/>
```

```
    key task_for_pid-allow /key
    true/
    key run-unsigned-code /key
    true/
/dict
/plist
?xml version="1.0" encoding="UTF-8"?
 DOCTYPE plist PUBLIC "-//Apple//DTD PLIST 1.0//EN" "http://www.apple.com/DTDs/Propert
yList-1.0.dtd"
plist version="1.0"
dict
    key get-task-allow /key
    true/
    key task_for_pid-allow /key
    true/
    key run-unsigned-code /key
    true/
/dict
/plist
```

与之对比，codesign则始终只输出1份entitlement权限信息：

```
➜  jtool2 codesign -d --entitlements - jtool2
Executable /Users/crifan/dev/dev_tool/reverse_security/iOS/jtool2/jtool2/jtool2
[Dict]
    [Key] get-task-allow
    [Value]
        [Bool] true
    [Key] run-unsigned-code
    [Value]
        [Bool] true
    [Key] task_for_pid-allow
    [Value]
        [Bool] true
```

ldid的help语法

```
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.

     -Acputype: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.
```

```
iPhoneDevWiki            October 8, 2021          iPhoneDevWiki
```

security

常见问题

运行崩溃killed

iOS（或Mac中）的app，由于签名方面问题，运行时出现崩溃问题，且提示Killed，可能的情况有多种：

debugserver运行崩溃killed

- 现象：用 `ldid` 重签名后的debugserver在 `iOS 15.0` 的 `iPhone8` 上运行出现崩溃提示killed

```
iPhone8-150:/usr/bin root# /usr/bin/debugserver --version
zsh: killed     /usr/bin/debugserver --version
```

- 原因：代码签名方面的问题
  - 查看崩溃日志ips，可以看到是代码签名问题
    - `type EXC_CRASH signal SIGKILL CODESIGNING`
  - 对比
    - `iOS 15` 之前可以用 `ldid` 重签名 -> 运行不会崩溃
    - 但 `iOS 15` 之后会崩溃
      - iOS 15之后，应该是内部做了签名方面的校验和判断
- 解决办法：要改用 `codesign` 重签名

```
codesign -f -s - --entitlements debugable_entitlement.xml debugserver
```

  - 注：核心差异应该是， `-s -` 会用上当前的签名

由此有了心得：

ldid换codesign

- iOS 15之前

```
ldid -Sdebugserver.entitlements debugserver
```

  - 可以正常运行
- iOS 15之后

```
ldid -Sdebugserver.entitlements debugserver
```

  - 运行崩溃：`Killed 9``
    - 解决办法：换 `codesign`

```
codesign -f -s - --entitlements debugserver.entitlements debugserver
```

即：

- 给debugserver添加entitlement权限（后，再放回iPhone中使用）
  - `< iOS 15` ：可以用 `ldid`

```
ldid -Sdebugable_entitlement.xml debugserver
```

- `>= iOS 15`
  - 不能用ldid
    - 否则就会运行报错killed
      - 查看崩溃日志ips，可以看到是：代码签名问题
        - `type EXC_CRASH signal SIGKILL CODESIGNING`
  - 要用 `codesign` （估计是其中带了 `signature` ）

```
codesign -f -s - --entitlements debugable_entitlement.xml debugserver
```

Mac M2 Max中jtool2运行崩溃Killed

- 现象

brew安装的官网的jtool2

```
brew install --cask jtool2
```

运行时报错：`Unrecoverable CT signature issue, bailing out`

```
默认     17:41:55.432539+0800    kernel    Checking in with amfid for DER jtool2.arm64e
默认     17:41:55.433342+0800    kernel    AMFI: /Users/crifan/dev/dev_tool/reverse_security/iOS/jtool2/jtool2_old/jtool2 doesn't have DER entitlements and will not work in a future release
默认     17:41:55.433439+0800    kernel    AMFI: '/Users/crifan/dev/dev_tool/reverse_security/iOS/jtool2/jtool2_old/jtool2' has no CMS blob?
默认     17:41:55.433454+0800    kernel    AMFI: '/Users/crifan/dev/dev_tool/reverse_security/iOS/jtool2/jtool2_old/jtool2': Unrecoverable CT signature issue, bailing out.
默认     17:41:55.433480+0800    kernel    AMFI: Releasing transmuted blob for jtool2.arm64e - private 94x
默认     17:41:55.433516+0800    kernel    proc 77535: load code signature error 4 for file "jtool2"
默认     17:41:55.434075+0800    kernel    exec_mach_imgact: not running binary "jtool2" built against preview arm64e ABI
默认     17:41:55.434268+0800    kernel    ASP: Security policy would not allow process: 77535, /Users/crifan/dev/dev_tool/reverse_security/iOS/jtool2/jtool2_old/jtool2
```

- 原因：签名方面问题
- 解决过程

用codesign去签名

```
codesign -f -s - --entitlements jtool2.entitlements jtool2
```

虽然能解决：`Unrecoverable CT signature issue, bailing out` 的签名问题，会仍会出现其他签名方面问题：

```
unsuitable CT policy 0 for this platform/device, rejecting signature.
```

```
  默认    09:46:52.624806+0800    kernel    AMFI: '/Users/crifan/dev/dev_tool/reverse_se
curity/iOS/jtool2/jtool2/jtool2' is adhoc signed.
  默认    09:46:52.624815+0800    kernel    AMFI: '/Users/crifan/dev/dev_tool/reverse_se
curity/iOS/jtool2/jtool2/jtool2': unsuitable CT policy 0 for this platform/device, reje
cting signature.
  默认    09:46:52.624854+0800    kernel    proc 13558: load code signature error 4 for
file "jtool2"
  默认    09:46:52.625370+0800    kernel    exec_mach_imgact: not running binary "jtool2"
 built against preview arm64e ABI
  默认    09:46:52.625580+0800    kernel    ASP: Security policy would not allow process
: 13558, /Users/crifan/dev/dev_tool/reverse_security/iOS/jtool2/jtool2/jtool2
```

-> 最终解决办法：

- 不用brew安装官网的jtool2 = 卸载掉原先安装的版本

  ```
  brew uninstall jtool2
  ```

- 换装其他版本的jtool2

  ```
  brew install --no-quarantine excitedplus1s/repo/jtool2
  ```

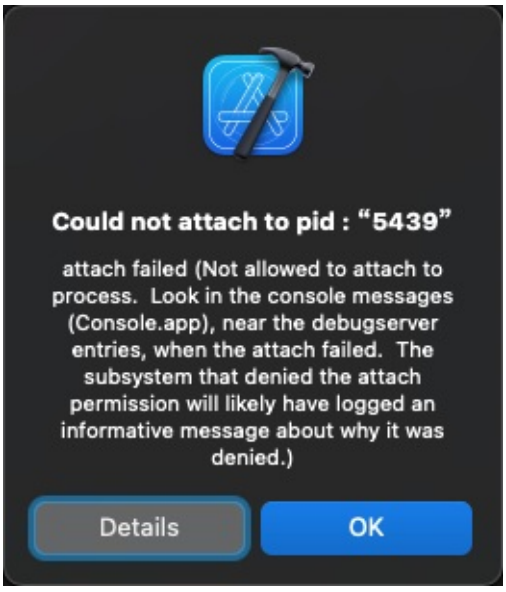
任意进程可调试

iOS逆向期间：

- 想要实现：任意进程可调试
  - 否则，Xcode去调试，就会报错：

```
Could not attach to pid : xxx

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



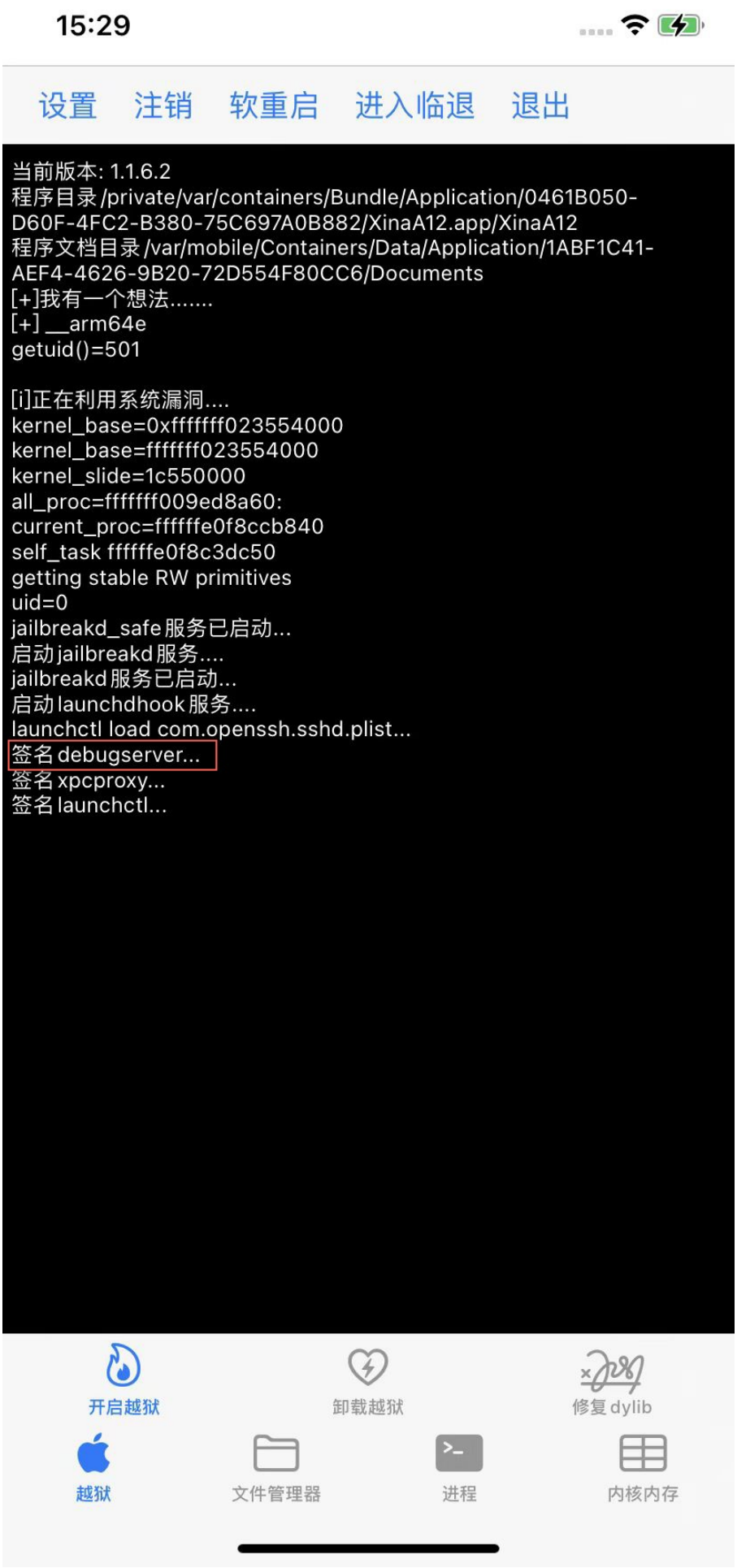
- 手段
  - 直接用 `XinaA15` 越狱（如果是A12+的iPhone的话）
    - 注：XinaA15自带支持所有进程可调试
  - 自己手动去操作：去给单个二进制文件**重新签名**，加可调试权限
    - 核心思路：导出原有entitlement权限，加上新的可调试权限（和其他权限微调），再用codesign去重签名，再写回iPhone

XinaA15自带支持所有进程可调试

XinaA15开启越狱期间，有个日志提示：

```
签名debugserver
```

-


15:29
设置
注销
软重启
进入临退
退出
当前版本: 1.1.6.2
程序目录/private/var/containers/Bundle/Application/0461B050-D60F-4FC2-B380-75C697A0B882/XinaA12.app/XinaA12
程序文档目录/var/mobile/Containers/Data/Application/1ABF1C41-AEF4-4626-9B20-72D554F80CC6/Documents
[+]我有一个想法.....
[+] __arm64e
getuid()=501
[i]正在利用系统漏洞....
kernel_base=0xffffff023554000
kernel_base=ffffff023554000
kernel_slide=1c550000
all_proc=ffffff009ed8a60:
current_proc=ffffffe0f8ccb840
self_task ffffffe0f8c3dc50
getting stable RW primitives
uid=0
jailbreakd_safe服务已启动...
启动jailbreakd服务....
jailbreakd服务已启动...
启动launchdhook服务....
launchctl load com.openssh.sshd.plist...
签名debugserver...
签名xpcproxy...
签名launchctl...
开启越狱
卸载越狱
修复dylib
越狱
文件管理器
进程
内核内存

估计就是指的是：

- 给debugserver重签名，使得其支持：所有进程可调试

得到我们要的效果：

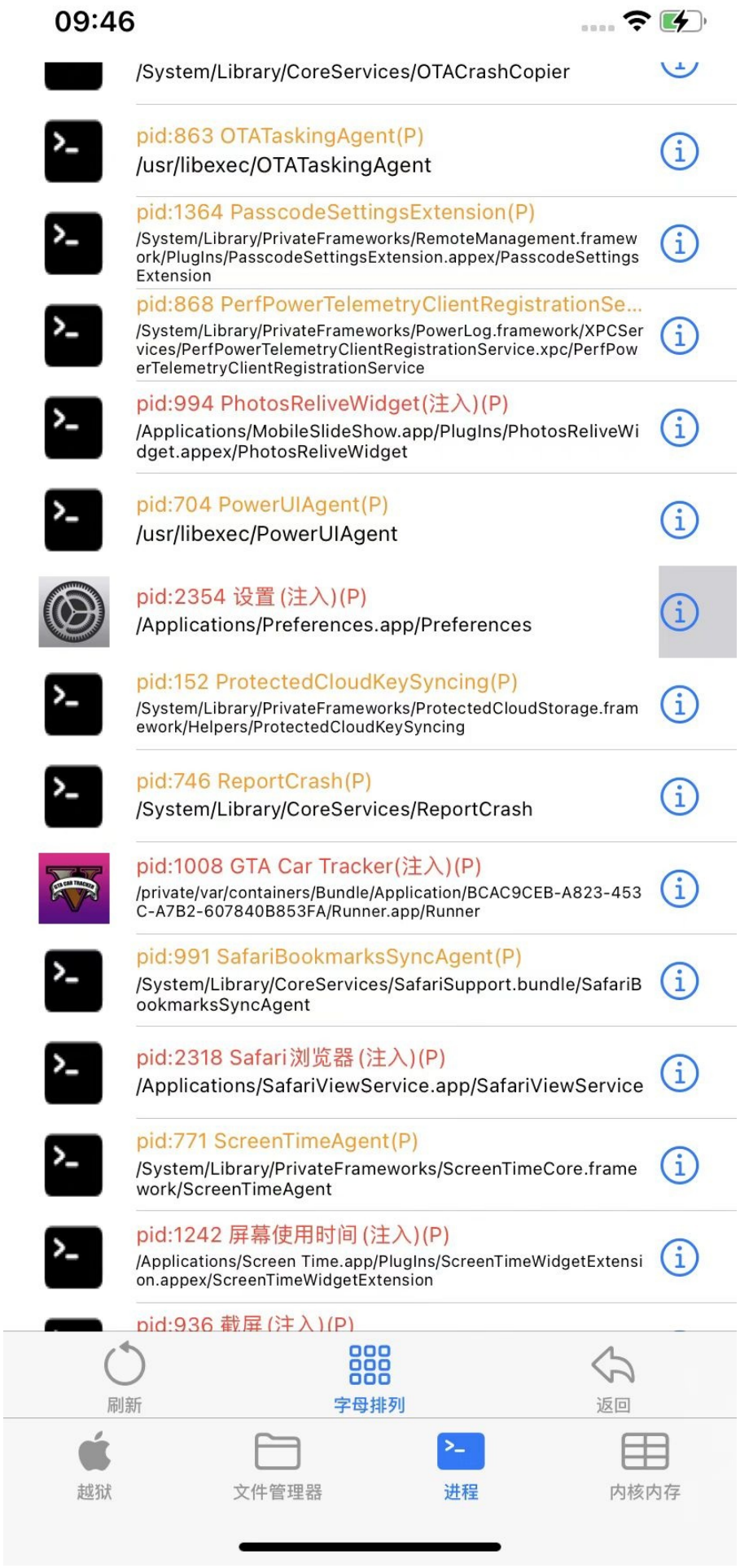
XinaA15越狱后，任何进程均可调试。

举例

`设置 = Preferences`

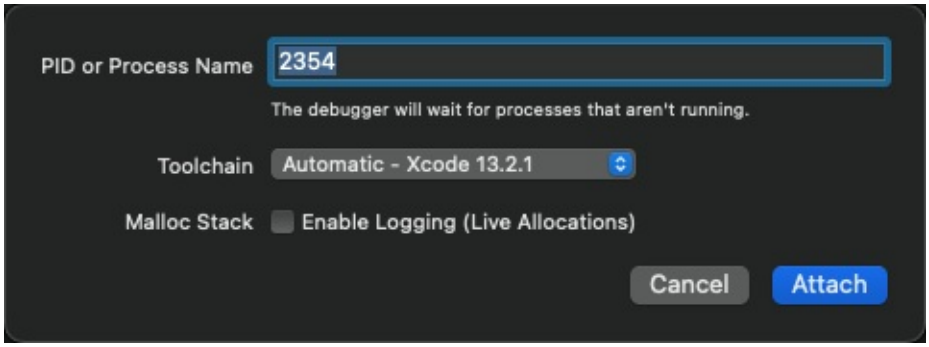
iOS内置app：`设置 = Preferences`

- 已注入
  -

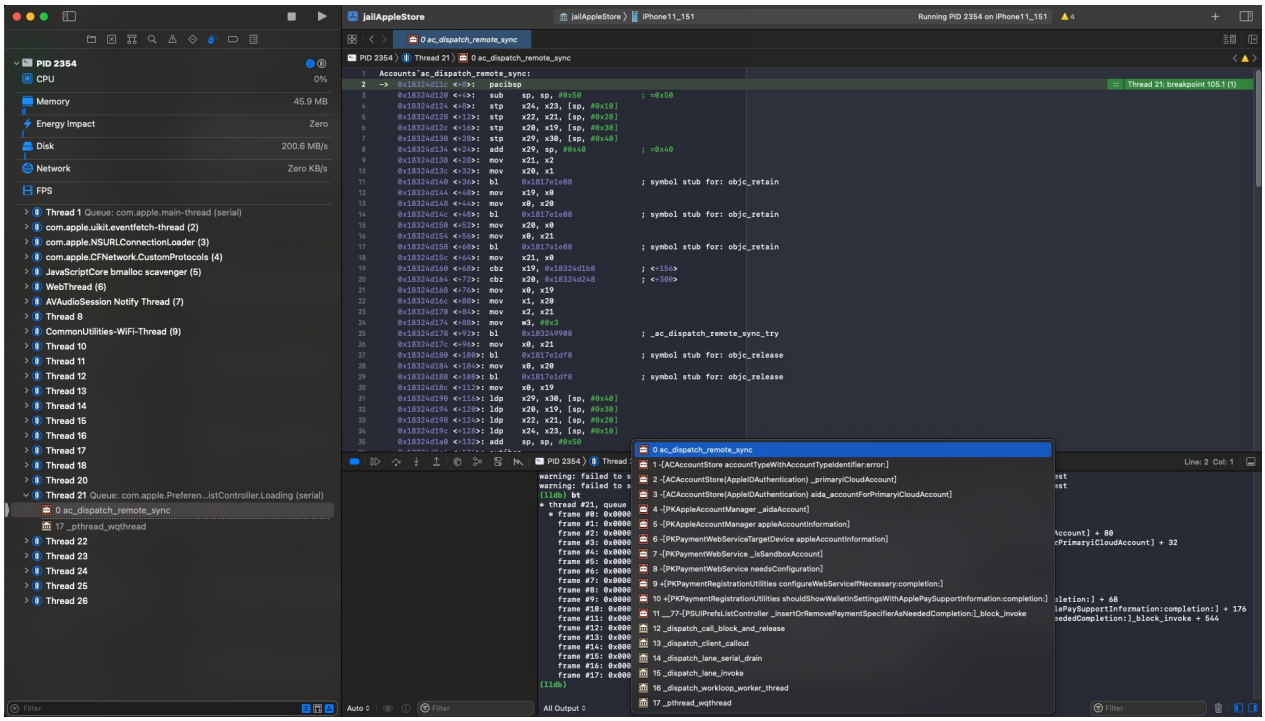

09:46
/System/Library/CoreServices/OTACrashCopier
pid:863 OTATaskingAgent(P)
/usr/libexec/OTATaskingAgent
pid:1364 PasscodeSettingsExtension(P)
/System/Library/PrivateFrameworks/RemoteManagement.framework/PlugIns/PasscodeSettingsExtension.appex/PasscodeSettingsExtension
pid:868 PerfPowerTelemetryClientRegistrationSe...
/System/Library/PrivateFrameworks/PowerLog.framework/XPCServices/PerfPowerTelemetryClientRegistrationService.xpc/PerfPowerTelemetryClientRegistrationService
pid:994 PhotosReliveWidget(注入)(P)
/Applications/MobileSlideShow.app/PlugIns/PhotosReliveWidget.appex/PhotosReliveWidget
pid:704 PowerUIAgent(P)
/usr/libexec/PowerUIAgent
pid:2354 设置 (注入)(P)
/Applications/Preferences.app/Preferences
pid:152 ProtectedCloudKeySyncing(P)
/System/Library/PrivateFrameworks/ProtectedCloudStorage.framework/Helpers/ProtectedCloudKeySyncing
pid:746 ReportCrash(P)
/System/Library/CoreServices/ReportCrash
pid:1008 GTA Car Tracker(注入)(P)
/private/var/containers/Bundle/Application/BCAC9CEB-A823-453C-A7B2-607840B853FA/Runner.app/Runner
pid:991 SafariBookmarksSyncAgent(P)
/System/Library/CoreServices/SafariSupport.bundle/SafariBookmarksSyncAgent
pid:2318 Safari浏览器 (注入)(P)
/Applications/SafariViewService.app/SafariViewService
pid:771 ScreenTimeAgent(P)
/System/Library/PrivateFrameworks/ScreenTimeCore.framework/ScreenTimeAgent
pid:1242 屏幕使用时间 (注入)(P)
/Applications/Screen Time.app/PlugIns/ScreenTimeWidgetExtension.appex/ScreenTimeWidgetExtension
pid:936 截屏 (注入)(P)
刷新
字母排列
返回
越狱
文件管理器
进程
内核内存

- 进程PID是 `2354`

Xcode中去通过PID `2354` 去调试Preferences：



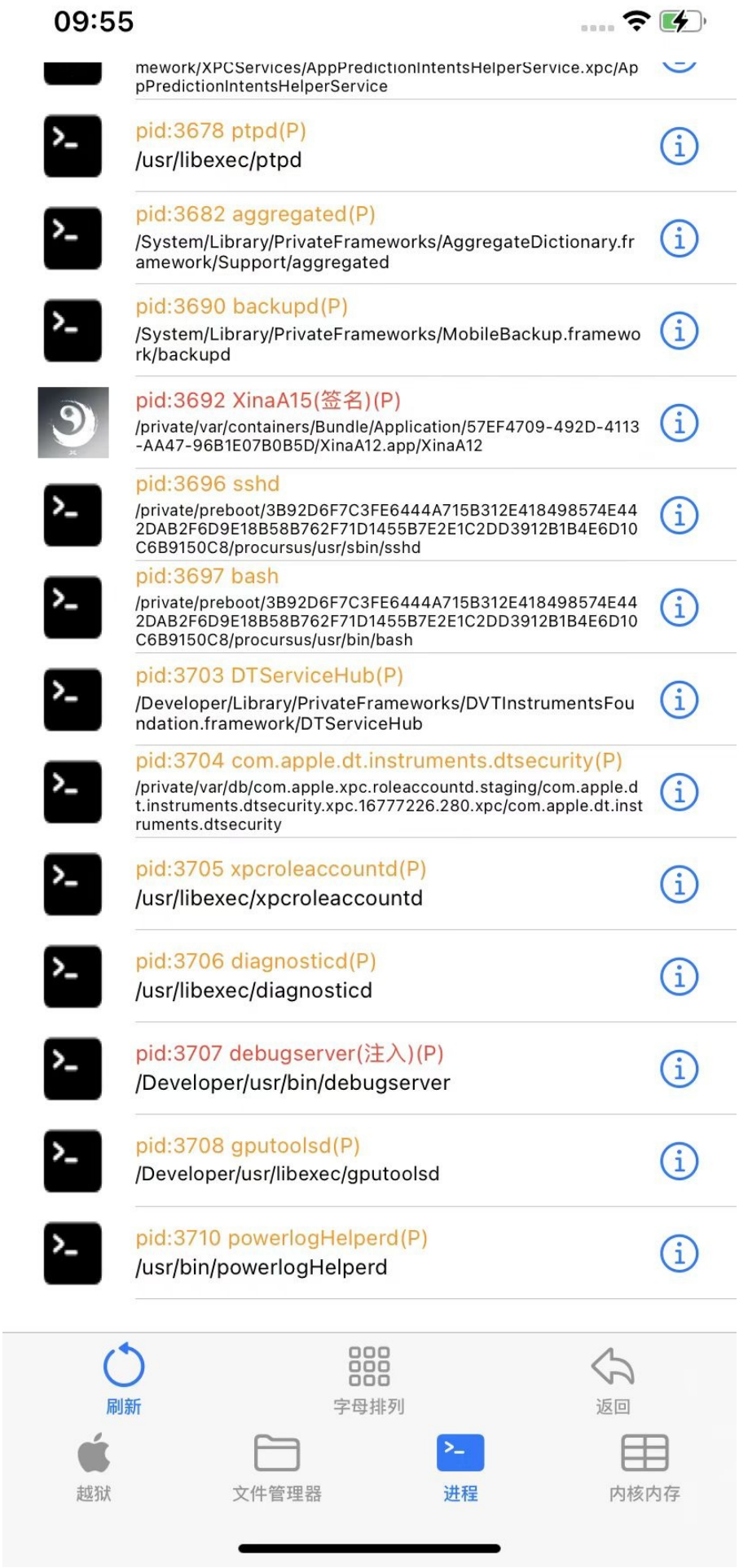
即可顺利调试Preferences：



debugserver

`XinaA15` 中的 `debugserver` 的进程：

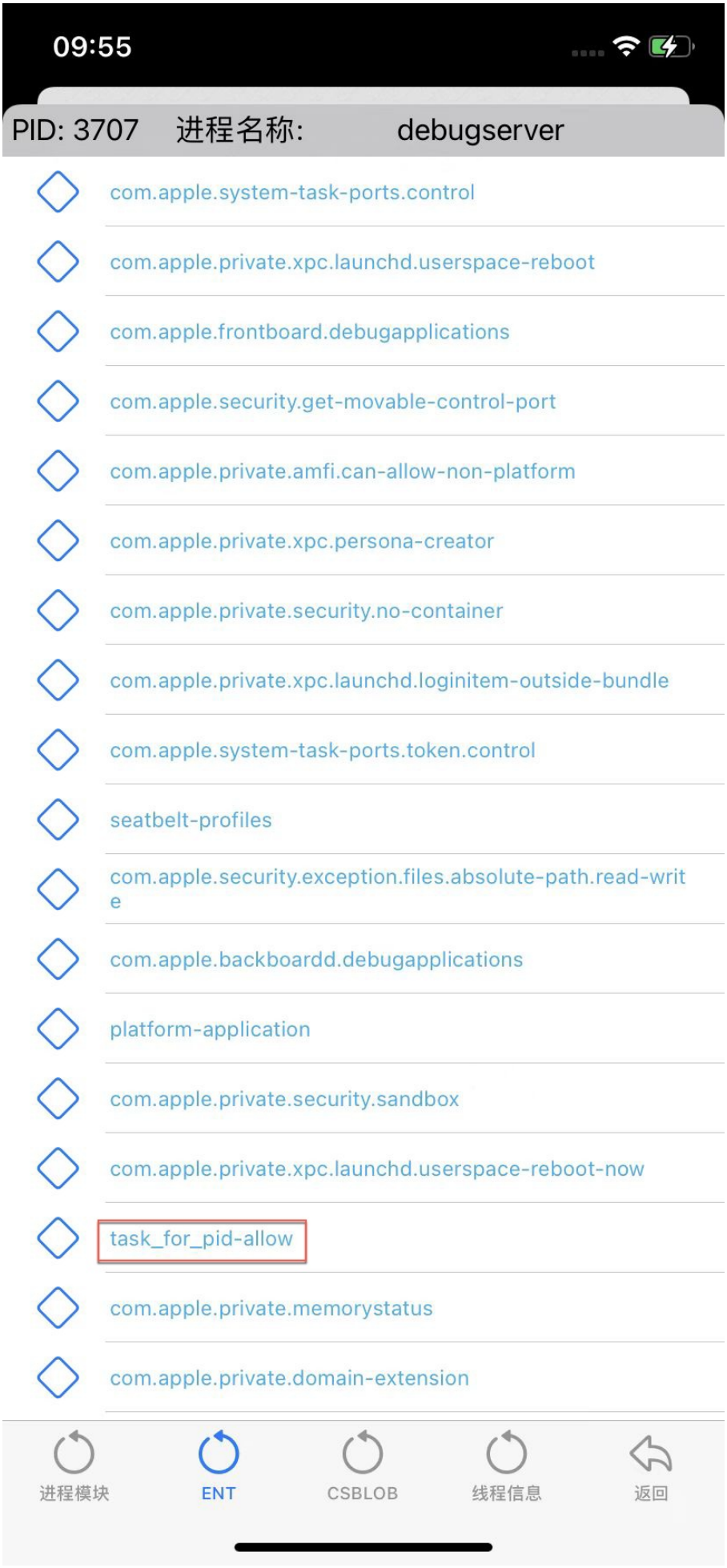
- debugserver：显示已被 `(注入)(P)`
  -


09:55
mework/XPCServices/AppPredictionIntentsHelperService.xpc/AppPredictionIntentsHelperService
pid:3678 ptpd(P)
/usr/libexec/ptpd
pid:3682 aggregated(P)
/System/Library/PrivateFrameworks/AggregateDictionary.framework/Support/aggregated
pid:3690 backupd(P)
/System/Library/PrivateFrameworks/MobileBackup.framework/backupd
pid:3692 XinaA15(签名)(P)
/private/var/containers/Bundle/Application/57EF4709-492D-4113-AA47-96B1E07B0B5D/XinaA12.app/XinaA12
pid:3696 sshd
/private/preboot/3B92D6F7C3FE6444A715B312E418498574E442DAB2F6D9E18B58B762F71D1455B7E2E1C2DD3912B1B4E6D10C6B9150C8/procursus/usr/sbin/sshd
pid:3697 bash
/private/preboot/3B92D6F7C3FE6444A715B312E418498574E442DAB2F6D9E18B58B762F71D1455B7E2E1C2DD3912B1B4E6D10C6B9150C8/procursus/usr/bin/bash
pid:3703 DTServiceHub(P)
/Developer/Library/PrivateFrameworks/DVTInstrumentsFoundation.framework/DTServiceHub
pid:3704 com.apple.dt.instruments.dtsecurity(P)
/private/var/db/com.apple.xpc.roleaccountd.staging/com.apple.dt.instruments.dtsecurity.xpc.16777226.280.xpc/com.apple.dt.instruments.dtsecurity
pid:3705 xpcroleaccountd(P)
/usr/libexec/xpcroleaccountd
pid:3706 diagnosticd(P)
/usr/libexec/diagnosticd
pid:3707 debugserver(注入)(P)
/Developer/usr/bin/debugserver
pid:3708 gputoolsd(P)
/Developer/usr/libexec/gputoolsd
pid:3710 powerlogHelperd(P)
/usr/bin/powerlogHelperd
刷新
字母排列
返回
越狱
文件管理器
进程
内核内存

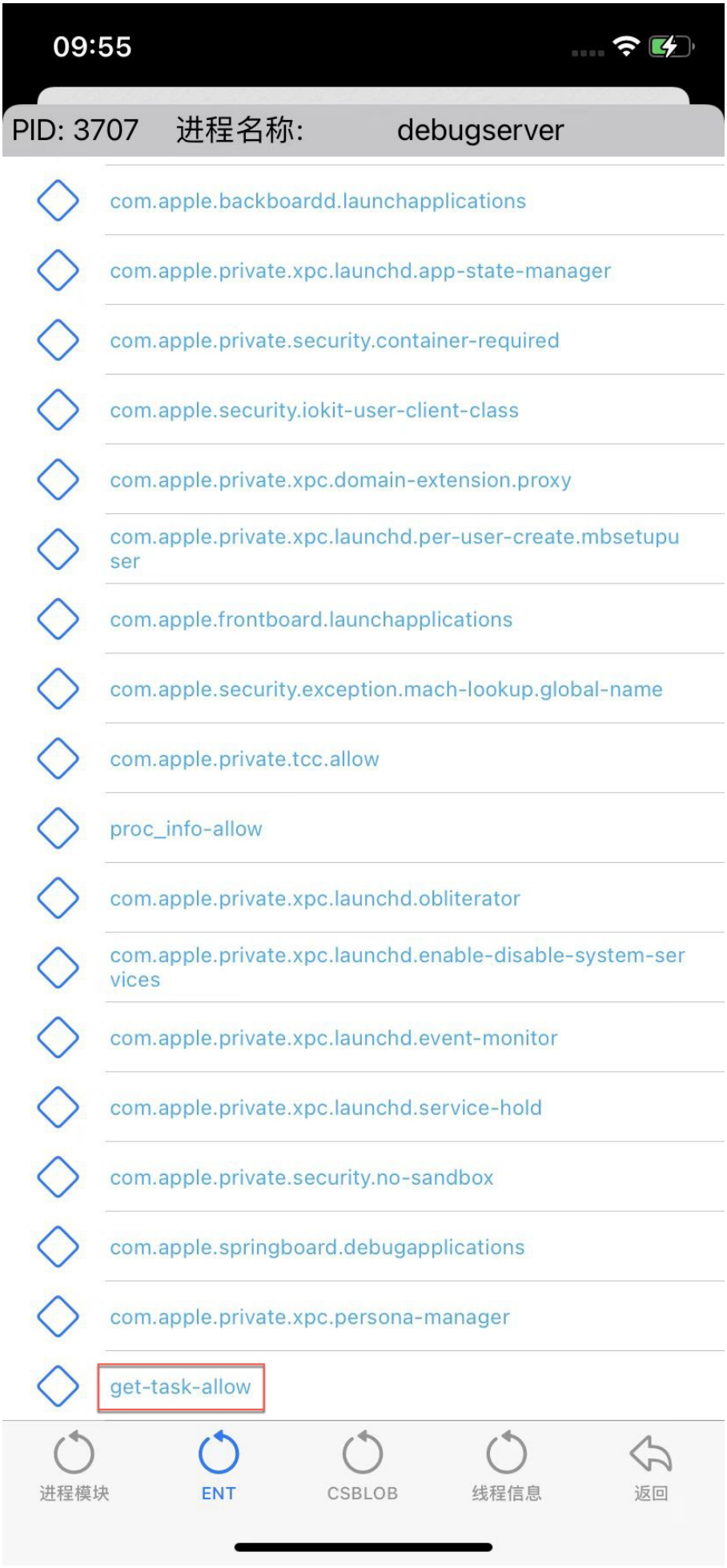
- 点击弹框可以查看到
  - 已注入了库：`xxx/procursus/usr/lib/TweakInject.dylib`
    -


09:55
PID: 3707 进程名称: debugserver
0x102240000/Developer/usr/bin/debugserver
0x102804000/private/preboot/3B92D6F7C3FE6444A715B312E418498574E442DAB2F6D9E18B58B762F71D1455B7E2E1C2DD3912B1B4E6D10C6B9150C8/procursus/usr/lib/TweakInject.dylib
0x102bac000/Developer/Library/PrivateFrameworks/DVTInstrumentsFoundation.framework/DVTInstrumentsFoundation
0x1027e8000/Developer/usr/lib/libsysmon.dylib
0x102820000/Developer/Library/PrivateFrameworks/ConditionInducer.framework/ConditionInducer
0x10299c000/Developer/Library/PrivateFrameworks/DTXConnectionServices.framework/DTXConnectionServices
0x10291c000/Developer/Library/PrivateFrameworks/DVTInstrumentsUtilities.framework/DVTInstrumentsUtilities
0x1aa087000/System/Library/PrivateFrameworks/CoreSymbolication.framework/CoreSymbolication
0x181a9c000/System/Library/Frameworks/CoreFoundation.framework/CoreFoundation
0x18e933000/System/Library/Frameworks/IOKit.framework/Versions/A/IOKit
0x19a23a000/usr/lib/libc++.1.dylib
0x1bbde6000/usr/lib/libSystem.B.dylib
0x19a194000/usr/lib/libobjc.A.dylib
0x1817f7000/usr/lib/libicucore.A.dylib
0x19a29e000/usr/lib/libc++abi.dylib
0x1f1c36000/usr/lib/system/libcache.dylib
0x1f1c3c000/usr/lib/system/libcommonCrypto.dylib
0x1f1c49000/usr/lib/system/libcompiler_rt.dylib
进程模块
ENT
CSBLOB
线程信息
返回

- 去查看 `ENT` = `entitlement` = `权限` ，发现已经有我们希望的值了
  - `task_for_pid-allow`
    -


09:55
PID: 3707 进程名称: debugserver
com.apple.system-task-ports.control
com.apple.private.xpc.launchd.userspace-reboot
com.apple.frontboard.debugapplications
com.apple.security.get-movable-control-port
com.apple.private.amfi.can-allow-non-platform
com.apple.private.xpc.persona-creator
com.apple.private.security.no-container
com.apple.private.xpc.launchd.loginitem-outside-bundle
com.apple.system-task-ports.token.control
seatbelt-profiles
com.apple.security.exception.files.absolute-path.read-write
com.apple.backboardd.debugapplications
platform-application
com.apple.private.security.sandbox
com.apple.private.xpc.launchd.userspace-reboot-now
task_for_pid-allow
com.apple.private.memorystatus
com.apple.private.domain-extension
进程模块
ENT
CSBLOB
线程信息
返回

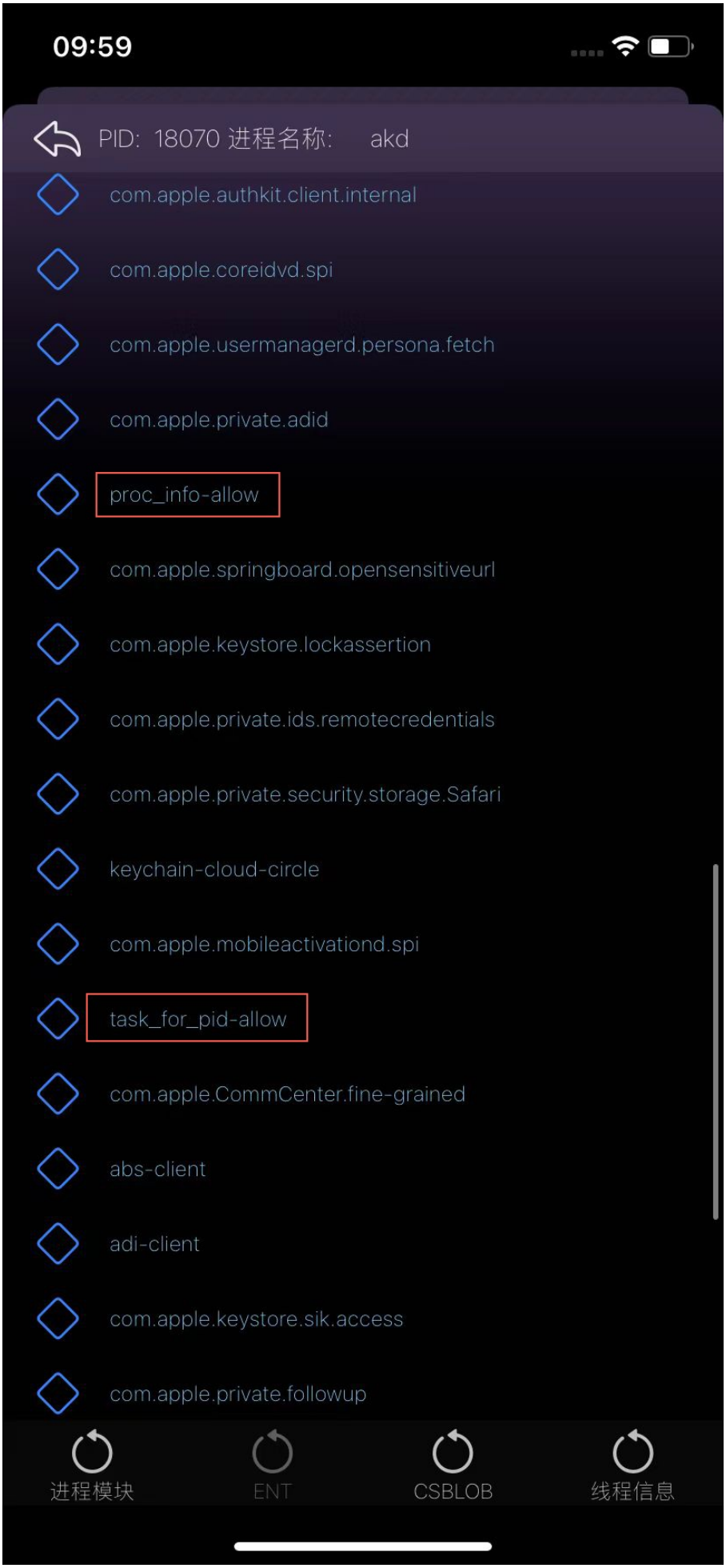
- `get-task-allow`
  -


09:55
PID: 3707 进程名称: debugserver
com.apple.backboardd.launchapplications
com.apple.private.xpc.launchd.app-state-manager
com.apple.private.security.container-required
com.apple.security.iokit-user-client-class
com.apple.private.xpc.domain-extension.proxy
com.apple.private.xpc.launchd.per-user-create.mbsetupuser
com.apple.frontboard.launchapplications
com.apple.security.exception.mach-lookup.global-name
com.apple.private.tcc.allow
proc_info-allow
com.apple.private.xpc.launchd.obliterator
com.apple.private.xpc.launchd.enable-disable-system-services
com.apple.private.xpc.launchd.event-monitor
com.apple.private.xpc.launchd.service-hold
com.apple.private.security.no-sandbox
com.apple.springboard.debugapplications
com.apple.private.xpc.persona-manager
get-task-allow
进程模块
ENT
CSBLOB
线程信息
返回

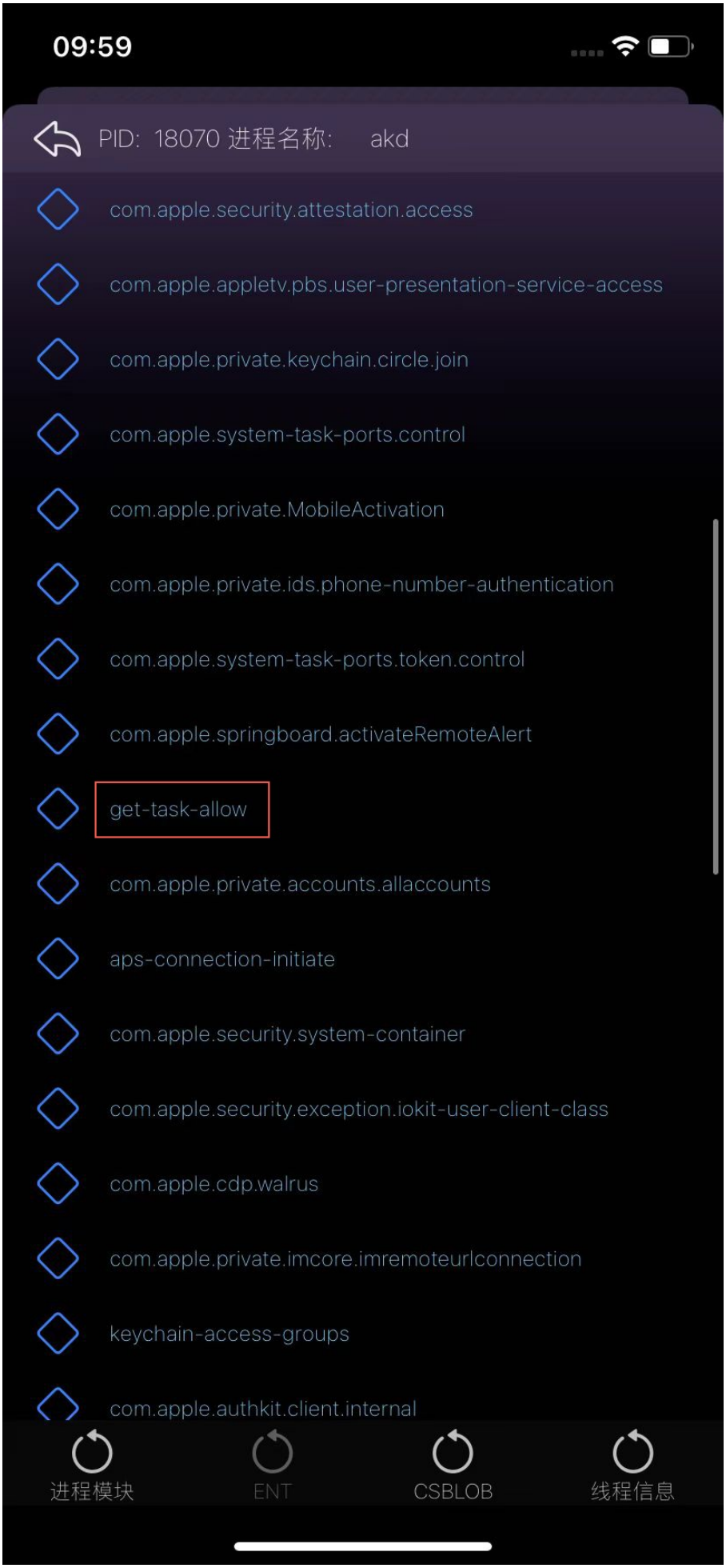
akd

`XinaA15` 中的 `akd` 的进程：

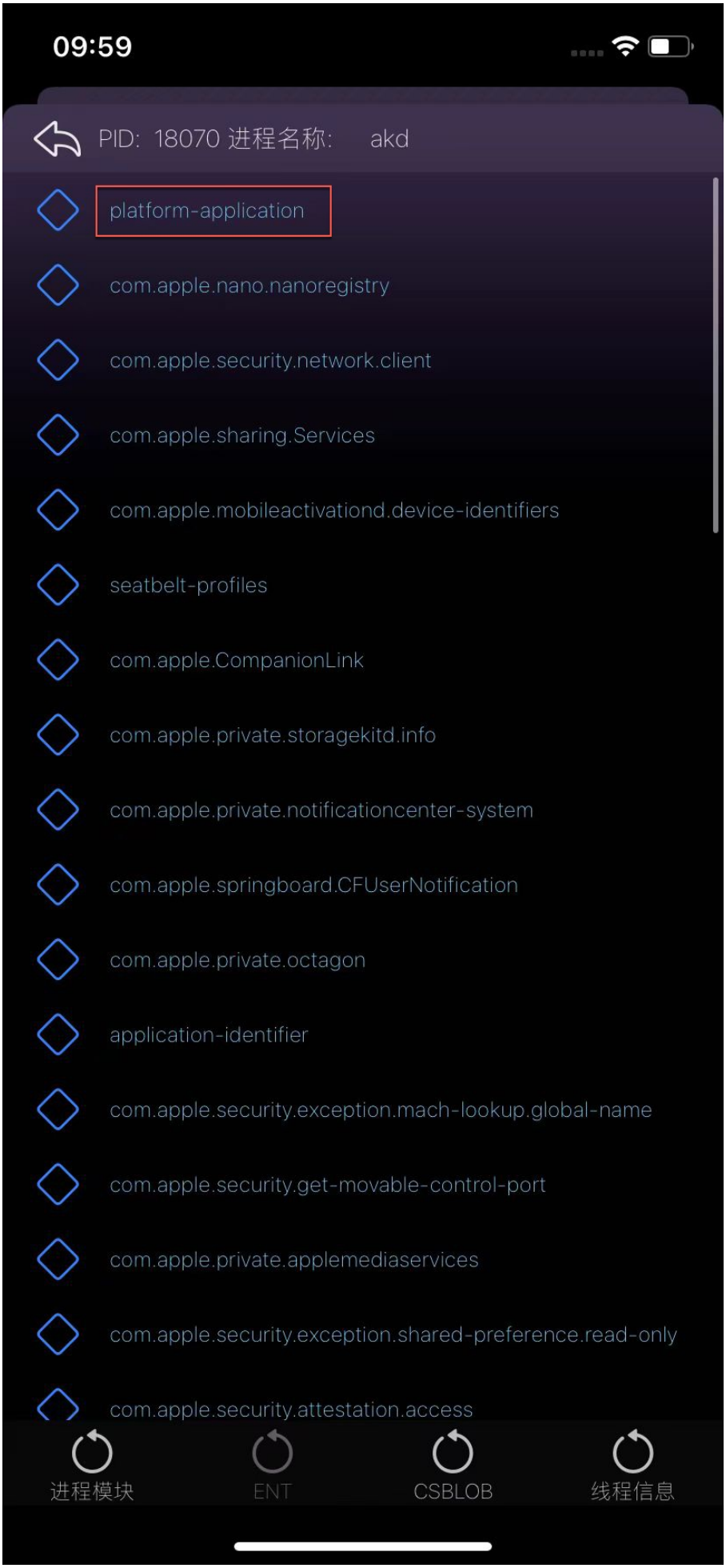
- 去查看 `ENT` = `entitlement` = `权限` ，发现已经有我们希望的值了
  - `task_for_pid-allow` + `proc_info-allow`
    -


09:59
PID: 18070 进程名称: akd
com.apple.authkit.client.internal
com.apple.coreidvd.spi
com.apple.usermanagerd.persona.fetch
com.apple.private.adid
proc_info-allow
com.apple.springboard.opensensitiveurl
com.apple.keystore.lockassertion
com.apple.private.ids.remotecredentials
com.apple.private.security.storage.Safari
keychain-cloud-circle
com.apple.mobileactivationd.spi
task_for_pid-allow
com.apple.CommCenter.fine-grained
abs-client
adi-client
com.apple.keystore.sik.access
com.apple.private.followup
进程模块
ENT
CSBLOB
线程信息

- `get-task-allow`
  -


09:59
PID: 18070 进程名称: akd
com.apple.security.attestation.access
com.apple.appletv.pbs.user-presentation-service-access
com.apple.private.keychain.circle.join
com.apple.system-task-ports.control
com.apple.private.MobileActivation
com.apple.private.ids.phone-number-authentication
com.apple.system-task-ports.token.control
com.apple.springboard.activateRemoteAlert
get-task-allow
com.apple.private.accounts.allaccounts
aps-connection-initiate
com.apple.security.system-container
com.apple.security.exception.iokit-user-client-class
com.apple.cdp.walrus
com.apple.private.imcore.imremoteurlconnection
keychain-access-groups
com.apple.authkit.client.internal
进程模块
ENT
CSBLOB
线程信息

- `platform-application`
  -


09:59
PID: 18070 进程名称: akd
platform-application
com.apple.nano.nanoregistry
com.apple.security.network.client
com.apple.sharing.Services
com.apple.mobileactivationd.device-identifiers
seatbelt-profiles
com.apple.CompanionLink
com.apple.private.storagekitd.info
com.apple.private.notificationcenter-system
com.apple.springboard.CFUserNotification
com.apple.private.octagon
application-identifier
com.apple.security.exception.mach-lookup.global-name
com.apple.security.get-movable-control-port
com.apple.private.applemediaservices
com.apple.security.exception.shared-preference.read-only
com.apple.security.attestation.access
进程模块
ENT
CSBLOB
线程信息

手动给单个二进制文件重签名

后来发现另外方式也暂时可以实现：任意进程可调试

对于要调试的文件：

- 服务进程=二进制文件
  - 比如
    - `AuthKit` 的daemon服务进程的二进制文件：`akd`
- app内的核心二进制文件
  - 比如
    - iOS系统内置app：`设置` = `Preferences` 的 `Preferences.app/Preferences`

举例

akd

- （1）从越狱iPhone中导出原始的 `akd`

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

- （2）导出 `akd` 中原始的entitlement权限

```
ldid -e akd
```

此处输出内容是：

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE plist PUBLIC "-//Apple//DTD PLIST 1.0//EN" "http://www.apple.com/DTDs/PropertyList-1.0.dtd">
<plist version="1.0">
<dict>
    <key>abs-client</key>
    <string>143531244</string>
    <key>adi-client</key>
    <string>572356293</string>
    <key>application-identifier</key>
    <string>com.apple.akd</string>
    <key>aps-connection-initiate</key>
    <true/>
    <key>com.apple.CommCenter.fine-grained</key>
    <array>
        <string>spi</string>
        <string>identity</string>
    </array>
    <key>com.apple.CompanionLink</key>
    <true/>
    <key>com.apple.accounts.idms.fullaccess</key>
    <true/>
```

```
    <key>com.apple.appletv.pbs.user-presentation-service-access</key>
    <true/>
    <key>com.apple.authkit.client.internal</key>
    <true/>
    <key>com.apple.cdp.walrus</key>
    <true/>
    <key>com.apple.coreidvd.spi</key>
    <true/>
    <key>com.apple.keystore.lockassertion</key>
    <true/>
    <key>com.apple.keystore.sik.access</key>
    <true/>
    <key>com.apple.mobileactivationd.device-identifiers</key>
    <true/>
    <key>com.apple.mobileactivationd.spi</key>
    <true/>
    <key>com.apple.nano.nanoregistry</key>
    <true/>
    <key>com.apple.private.MobileActivation</key>
    <array>
        <string>RequestActivationState</string>
    </array>
    <key>com.apple.private.accounts.allaccounts</key>
    <true/>
    <key>com.apple.private.adid</key>
    <true/>
    <key>com.apple.private.applemediaservices</key>
    <true/>
    <key>com.apple.private.aps-connection-initiate</key>
    <true/>
    <key>com.apple.private.associated-domains</key>
    <true/>
    <key>com.apple.private.followup</key>
    <true/>
    <key>com.apple.private.ids.messaging</key>
    <array>
        <string>com.apple.private.alloy.anisette</string>
    </array>
    <key>com.apple.private.ids.phone-number-authentication</key>
    <true/>
    <key>com.apple.private.ids.remotecredentials</key>
    <true/>
    <key>com.apple.private.imcore.imremoteurlconnection</key>
    <true/>
    <key>com.apple.private.keychain.circle.join</key>
    <true/>
    <key>com.apple.private.notificationcenter-system</key>
    <array>
        <dict>
            <key>identifier</key>
            <string>com.apple.AuthKit</string>
        </dict>
    </array>
    <key>com.apple.private.octagon</key>
    <true/>
    <key>com.apple.private.security.storage.Safari</key>
```

```
    <true/>
    <key>com.apple.private.storagekitd.info</key>
    <true/>
    <key>com.apple.private.usernotifications.bundle-identifiers</key>
    <array>
        <string>com.apple.Preferences</string>
    </array>
    <key>com.apple.security.attestation.access</key>
    <true/>
    <key>com.apple.security.exception.mach-lookup.global-name</key>
    <array>
        <string>com.apple.SharedWebCredentials</string>
        <string>com.apple.fairplayd.versioned</string>
        <string>com.apple.mobile.keybagd.xpc</string>
        <string>com.apple.mobileactivationd</string>
    </array>
    <key>com.apple.security.exception.shared-preference.read-only</key>
    <array>
        <string>com.apple.nanobuddy</string>
        <string>com.apple.pairedsync</string>
    </array>
    <key>com.apple.security.network.client</key>
    <true/>
    <key>com.apple.security.system-container</key>
    <true/>
    <key>com.apple.sharing.Services</key>
    <true/>
    <key>com.apple.springboard.CFUserNotification</key>
    <true/>
    <key>com.apple.springboard.activateRemoteAlert</key>
    <true/>
    <key>com.apple.springboard.opensensitiveurl</key>
    <true/>
    <key>com.apple.symptom_diagnostics.report</key>
    <true/>
    <key>com.apple.usermanagerd.persona.background</key>
    <true/>
    <key>com.apple.usermanagerd.persona.fetch</key>
    <true/>
    <key>fairplay-client</key>
    <string>508119322</string>
    <key>keychain-access-groups</key>
    <array>
        <string>apple</string>
        <string>com.apple.akd</string>
        <string>com.apple.cfnetwork</string>
        <string>com.apple.akd.pcsauth</string>
    </array>
    <key>keychain-cloud-circle</key>
    <true/>
    <key>platform-application</key>
    <true/>
    <key>seatbelt-profiles</key>
    <array>
        <string>akd</string>
    </array>
```

```
</dict>
</plist>
```

- （3）编辑原先的签名=entitlement=权限，加上可调试（和部分的其他的entitlement的的调整）
  - 主要处理逻辑是
    - 加上几个新的权限
      - `get-task-allow`

        ```
        <key>get-task-allow</key>
        <true/>
        ```

      - `task_for_pid-allow`

        ```
        <key>task_for_pid-allow</key>
        <true/>
        ```

      - `run-unsigned-code`

        ```
        <key>run-unsigned-code</key>
        <true/>
        ```

    - 以及去掉不需要的权限
      - `seatbelt-profiles`

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

      - `com.apple.security.network.client`

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

最后变成：

`akd_debuggable.entitlements`

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE plist PUBLIC "-//Apple//DTD PLIST 1.0//EN" "http://www.apple.com/DTDs/Propert
yList-1.0.dtd">
<plist version="1.0">
<dict>
    <key>abs-client</key>
    <string>143531244</string>
    <key>adi-client</key>
    <string>572356293</string>
    <key>application-identifier</key>
    <string>com.apple.akd</string>
    <key>aps-connection-initiate</key>
    <true/>
    <key>com.apple.CommCenter.fine-grained</key>
```

```
    <array>
        <string>spi</string>
        <string>identity</string>
    </array>
    <key>com.apple.CompanionLink</key>
    <true/>
    <key>com.apple.accounts.idms.fullaccess</key>
    <true/>
    <key>com.apple.appletv.pbs.user-presentation-service-access</key>
    <true/>
    <key>com.apple.authkit.client.internal</key>
    <true/>
    <key>com.apple.cdp.walrus</key>
    <true/>
    <key>com.apple.coreidvd.spi</key>
    <true/>
    <key>com.apple.keystore.lockassertion</key>
    <true/>
    <key>com.apple.keystore.sik.access</key>
    <true/>
    <key>com.apple.mobileactivationd.device-identifiers</key>
    <true/>
    <key>com.apple.mobileactivationd.spi</key>
    <true/>
    <key>com.apple.nano.nanoregistry</key>
    <true/>
    <key>com.apple.private.MobileActivation</key>
    <array>
        <string>RequestActivationState</string>
    </array>
    <key>com.apple.private.accounts.allaccounts</key>
    <true/>
    <key>com.apple.private.adid</key>
    <true/>
    <key>com.apple.private.applemediaservices</key>
    <true/>
    <key>com.apple.private.aps-connection-initiate</key>
    <true/>
    <key>com.apple.private.associated-domains</key>
    <true/>
    <key>com.apple.private.followup</key>
    <true/>
    <key>com.apple.private.ids.messaging</key>
    <array>
        <string>com.apple.private.alloy.anisette</string>
    </array>
    <key>com.apple.private.ids.phone-number-authentication</key>
    <true/>
    <key>com.apple.private.ids.remotecredentials</key>
    <true/>
    <key>com.apple.private.imcore.imremoteurlconnection</key>
    <true/>
    <key>com.apple.private.keychain.circle.join</key>
    <true/>
    <key>com.apple.private.notificationcenter-system</key>
    <array>
```

```
        <dict>
            <key>identifier</key>
            <string>com.apple.AuthKit</string>
        </dict>
    </array>
    <key>com.apple.private.octagon</key>
    <true/>
    <key>com.apple.private.security.storage.Safari</key>
    <true/>
    <key>com.apple.private.storagekitd.info</key>
    <true/>
    <key>com.apple.private.usernotifications.bundle-identifiers</key>
    <array>
        <string>com.apple.Preferences</string>
    </array>
    <key>com.apple.security.attestation.access</key>
    <true/>
    <key>com.apple.security.exception.mach-lookup.global-name</key>
    <array>
        <string>com.apple.SharedWebCredentials</string>
        <string>com.apple.fairplayd.versioned</string>
        <string>com.apple.mobile.keybagd.xpc</string>
        <string>com.apple.mobileactivationd</string>
    </array>
    <key>com.apple.security.exception.shared-preference.read-only</key>
    <array>
        <string>com.apple.nanobuddy</string>
        <string>com.apple.pairedsync</string>
    </array>
    <key>com.apple.security.system-container</key>
    <true/>
    <key>com.apple.sharing.Services</key>
    <true/>
    <key>com.apple.springboard.CFUserNotification</key>
    <true/>
    <key>com.apple.springboard.activateRemoteAlert</key>
    <true/>
    <key>com.apple.springboard.opensensitiveurl</key>
    <true/>
    <key>com.apple.symptom_diagnostics.report</key>
    <true/>
    <key>com.apple.usermanagerd.persona.background</key>
    <true/>
    <key>com.apple.usermanagerd.persona.fetch</key>
    <true/>
    <key>fairplay-client</key>
    <string>508119322</string>
    <key>keychain-access-groups</key>
    <array>
        <string>apple</string>
        <string>com.apple.akd</string>
        <string>com.apple.cfnetwork</string>
        <string>com.apple.akd.pcsauth</string>
    </array>
    <key>keychain-cloud-circle</key>
    <true/>
```

```
    <key>platform-application</key>
    <true/>
    <key>get-task-allow</key>
    <true/>
    <key>task_for_pid-allow</key>
    <true/>
    <key>run-unsigned-code</key>
    <true/>
</dict>
</plist>
```

- （4）用新的entitlement去重新签名

注：不要用 `ldid`，否则 `iOS 15` 之后，`ldid` 重签名后的二进制运行会崩溃 `segment fault, killed 9`，因为是内部只有entitlement权限，没有codesign签名

要用codesign去重新签名：

```
codesign -f -s - --entitlements akd_debuggable.entitlements akd
```

- 输出

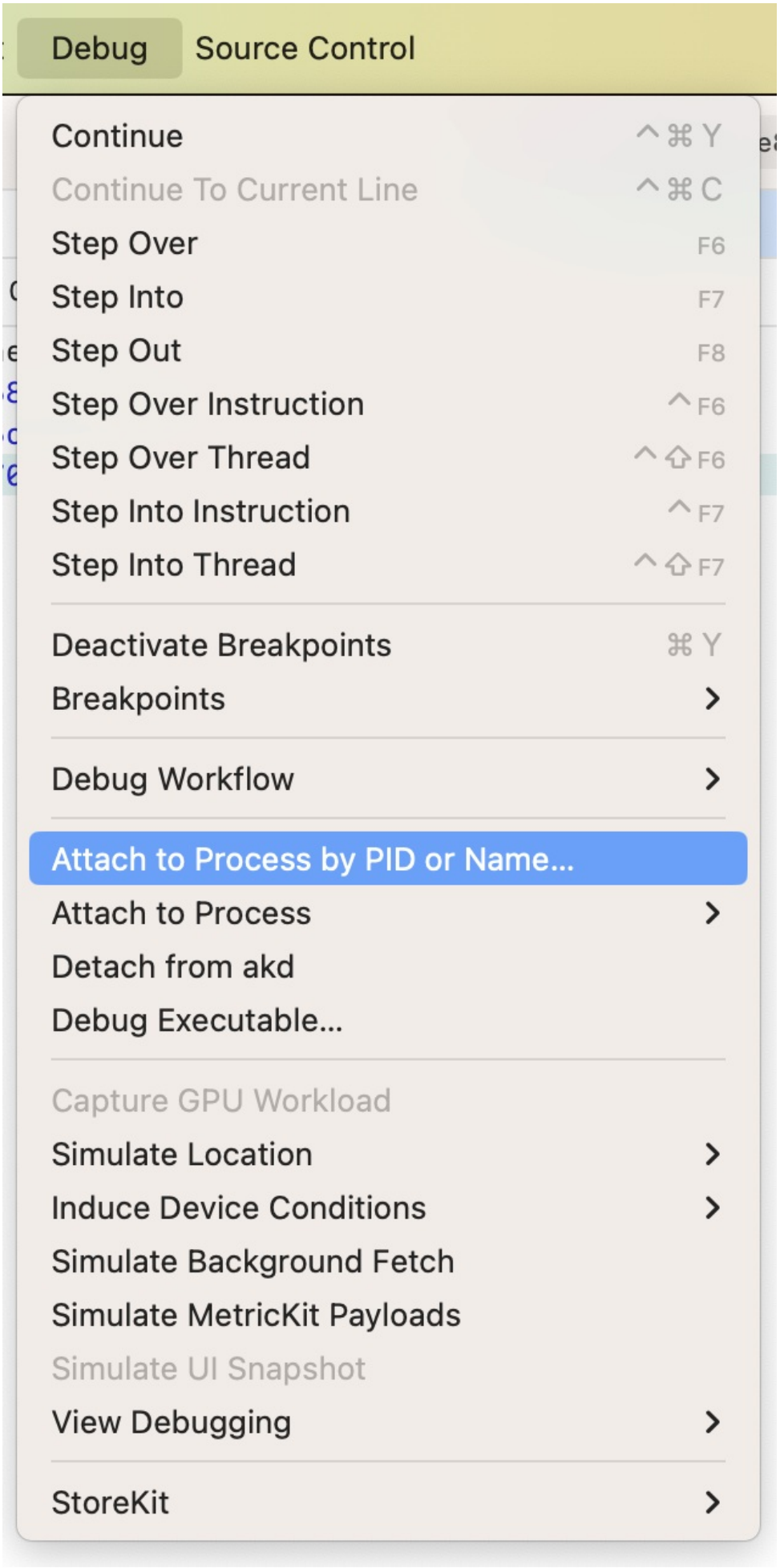
  ```
  akd: replacing existing signature
  ```

- （5）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/AuthKit.framework/akd
```

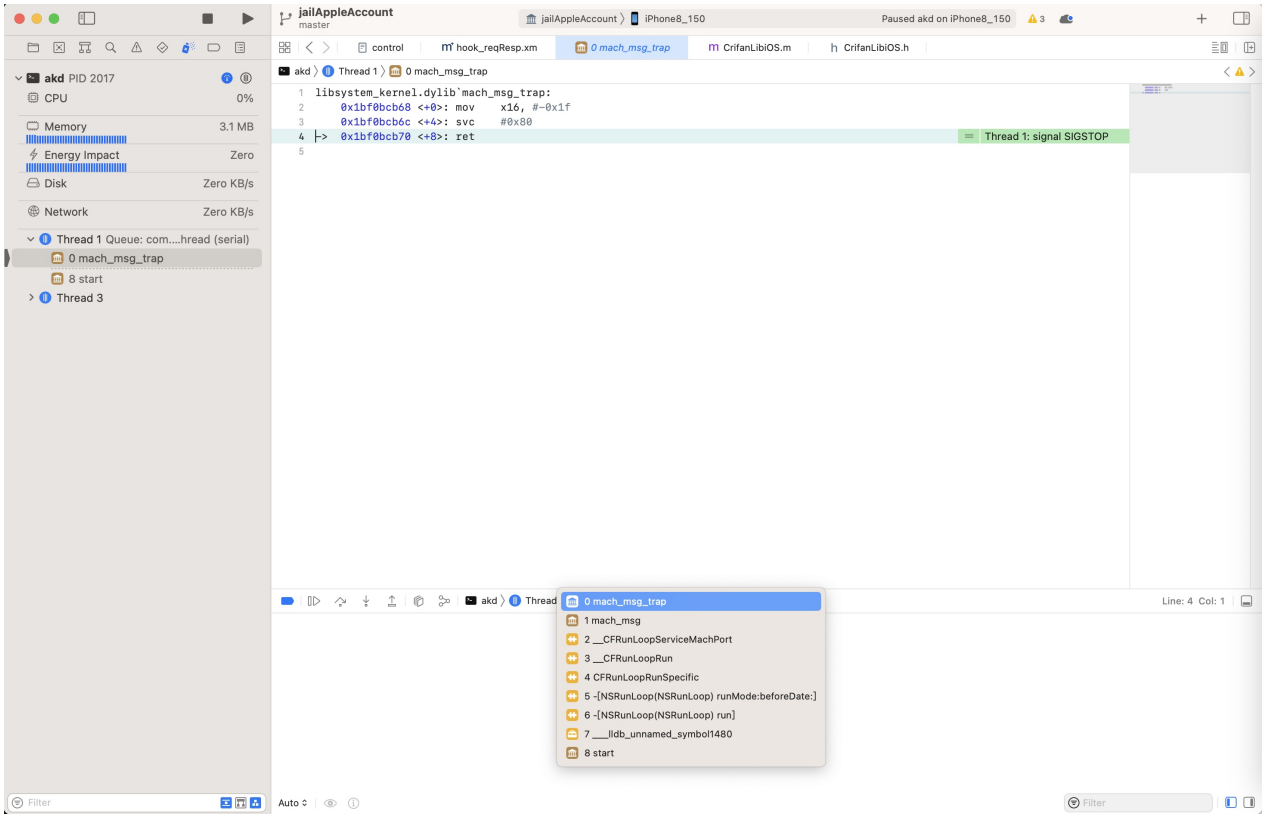
之后即可用Xcode去：

- `Xcode -> Debug -> Attach to Process by PID or Name -> PID or Process Name：akd -> Attach`
  -


Debug
Source Control
Continue ^⌘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 akd
Debug Executable...
Capture GPU Workload
Simulate Location
Induce Device Conditions
Simulate Background Fetch
Simulate MetricKit Payloads
Simulate UI Snapshot
View Debugging
StoreKit

-

从而实现Xcode可以调试此处 `akd` 进程了：



`Preferences` = 设置

核心思路是：

- 导出原有的iPhone中的二进制
  - 此处的：`/Applications/Preferences.app/Preferences`
- （用 `ldid` ）导出原先：`entitlement` = 权限

```
ldid -e Preferences > Preferences_entitlement.plist
```

- 重新编辑签名文件
  - 加上之前的3个属性
    - `get-task-allow`
    - `task_for_pid-allow`

    - `run-unsigned-code`
- （用 `codesign` ）重新签名

```
codesign -f -s - --entitlements Preferences_entitlement_debuggable.plist Preferences
```

- 写回 `iPhone`

如此，Preferences就可以被调试了 -> Xcode就可以去Attach挂载方式去调试 `Preferences` 了

`amsaccountsd`

Xcode中想要调试 `AppleMediaServices.framework` 的daemon进程： `amsaccountsd`

然后也按照上述步骤去操作：

- 导出原始的二进制

```
scp root@192.168.2.13:/System/Library/PrivateFrameworks/AppleMediaServices.framework/amsaccountsd .
```

- 导出原有的entitlement权限

```
amsaccountsd > amsaccountsd_entitlement_orig.xml
```

- 改动权限
    - 新增3个可调试属性

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

    - 注：没有 `seatbelt-profiles` 、 `com.apple.security.network.client` ，所以无需移除
    - 保存为新文件： `amsaccountsd_entitlement_debuggable.xml`
- 重新签名

```
codesign -f -s - --entitlements amsaccountsd_entitlement_debuggable.xml amsaccountsd
```

    - 再改名为： `amsaccountsd_debuggable`
- 拷贝回去

```
scp amsaccountsd_debuggable root@192.168.2.13:/System/Library/PrivateFrameworks/AppleMediaServices.framework/amsaccountsd
```

安装app时的签名问题

- 早期：iOS的app，默认只有7天的签名有效期
  - 7天后，签名生效，需要：重新签名=续签
    - 关于重签名=续签，当时有很多工具
- 现在：有了各种更加高效的ipa安装工具，已可以自动规避或解决重签名问题
  - 比如
    - TrollStore

MonkeyDev调试时崩溃

附录

下面列出相关参考资料。

参考资料

- 【已解决】debugserver加了权限后放回iPhone8无法运行而崩溃报错killed
- 【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
- 【已解决】iOS逆向：jtool2在Mac M2 Max中运行崩溃AMFI Unrecoverable CT signature issue, bailing out
- 【规避解决】iOS逆向：jtool2在Mac M2 Max中运行崩溃AMFI unsuitable CT policy 0 for this platform/device, rejecting signature
- 【已解决】解决debugserver崩溃问题：用codesign重新加权限保留原签名
- 【未解决】XinaA15是如何重签名debugserver而加上权限task_for_pid-allow能调试任意进程的
- 【未解决】iOS逆向：用jailbreakd_client给debugserver去加上entitle和platformize
- 【未解决】iOS逆向获取进程中csflags中CS_GET_TASK_ALLOW：权限proc_info-allow
- 【已解决】iOS逆向Apple账号：Xcode实时调试Preferences设置
- 【已解决】iOS逆向进程可调试：给akd加上可调试的entitlement权限
- 【已解决】iOS逆向：给amsaccountsd重签名以使得Xcode可以调试amsaccountsd
- 【未解决】Xcode调试Preferences报错：Could not attach to pid attach failed Not allowed to attach to process
- 【未解决】iOS中debugserver报connection错：查看entitlement时输出no signature
-
- lldb+debugserver · iOS逆向开发：动态调试 (crifan.org)
- Reverse Engineering iOS Mobile Apps (nowsecure.com)
- owasp-mastg/Document/0x06c-Reverse-Engineering-and-Tampering.md at master · OWASP/owasp-mastg · GitHub
- man page codesign section 1
- vtky/resign: XCode Project to resign .ipa files
- Cycript Tricks - iPhone Development Wiki
- hta-r04a-hacking-ios-on-the-run-using-cycript.pdf (huihoo.com)
- ios_code_signing_examples.sh
- Entitlements | Apple Developer Documentation
- com.apple.security.network.server
- com.apple.security.network.client
-