弯道超车

几 乎 没 有 防 护 的 i O S 逆 向 体 验

—— r0ysue

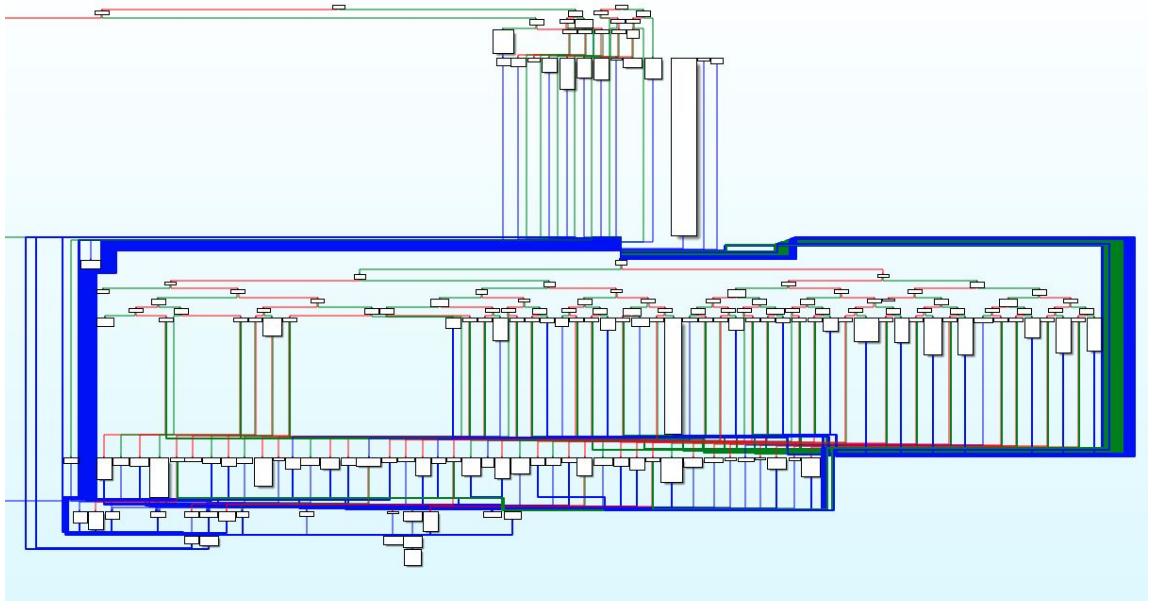
简介

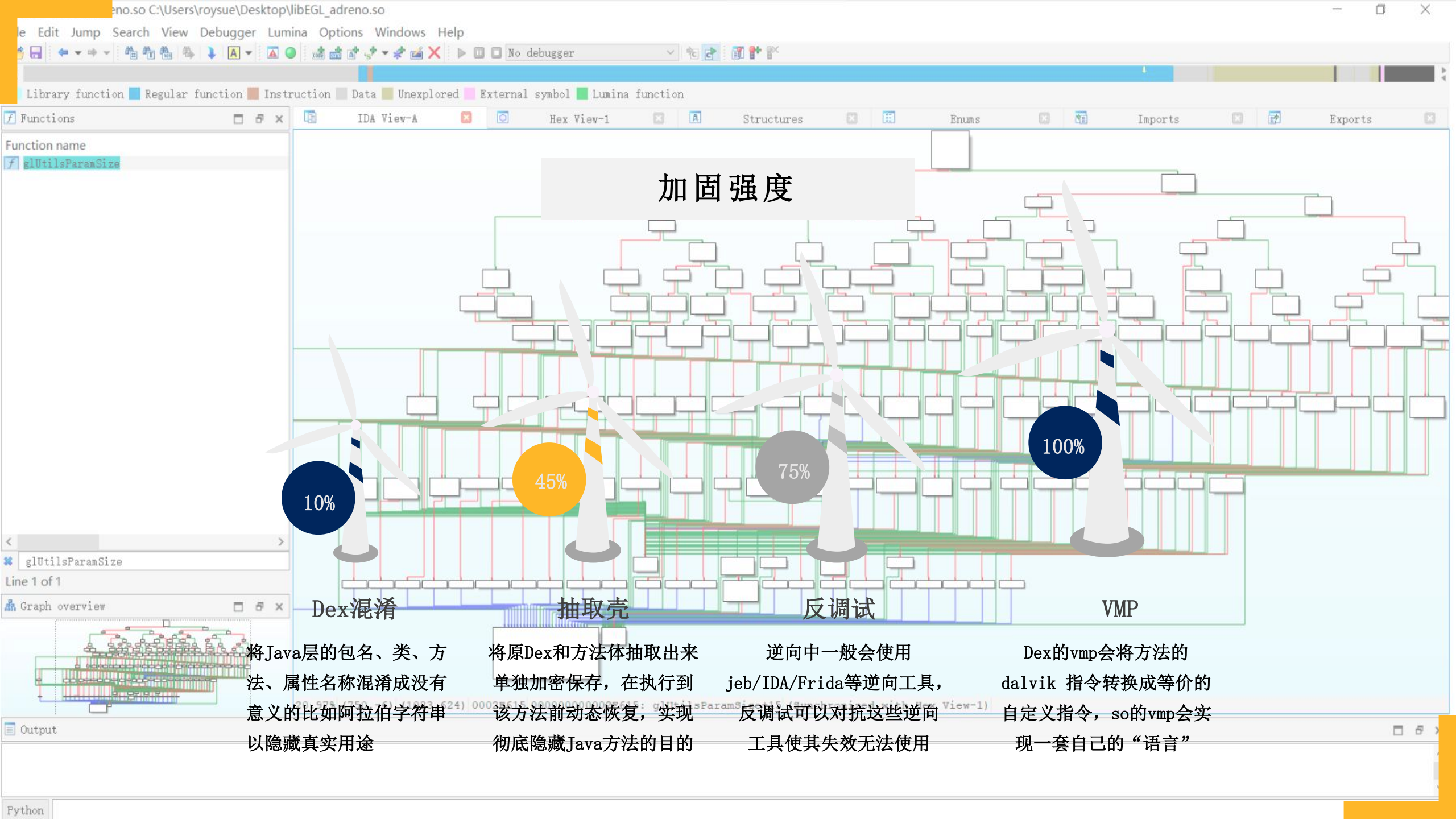
CONTENT

PART 01

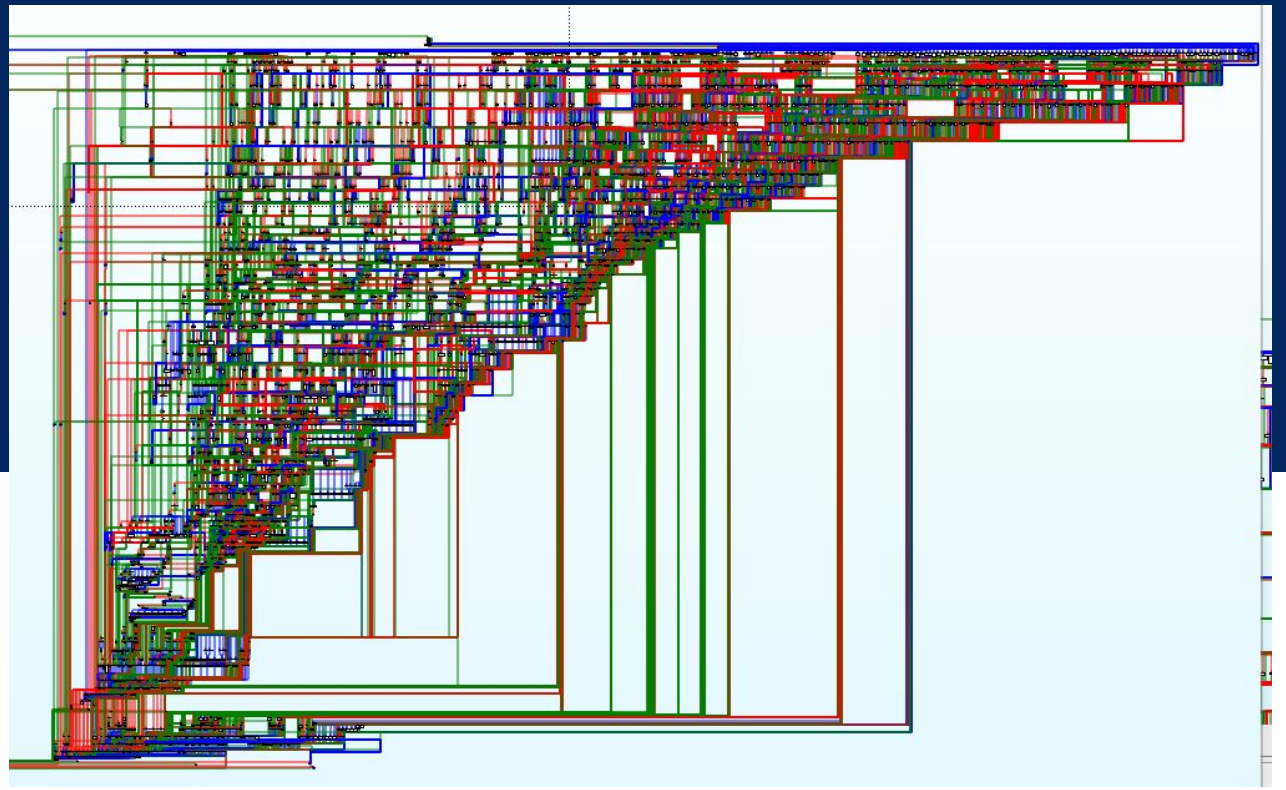
安卓开源→
加固白热化！

由于安卓的完全开源和去中心策略，加固厂商针对安卓系统推出了全系列的加固产品，包括且不限于一二三代壳、花指令、强混淆、反调试、虚拟机等！



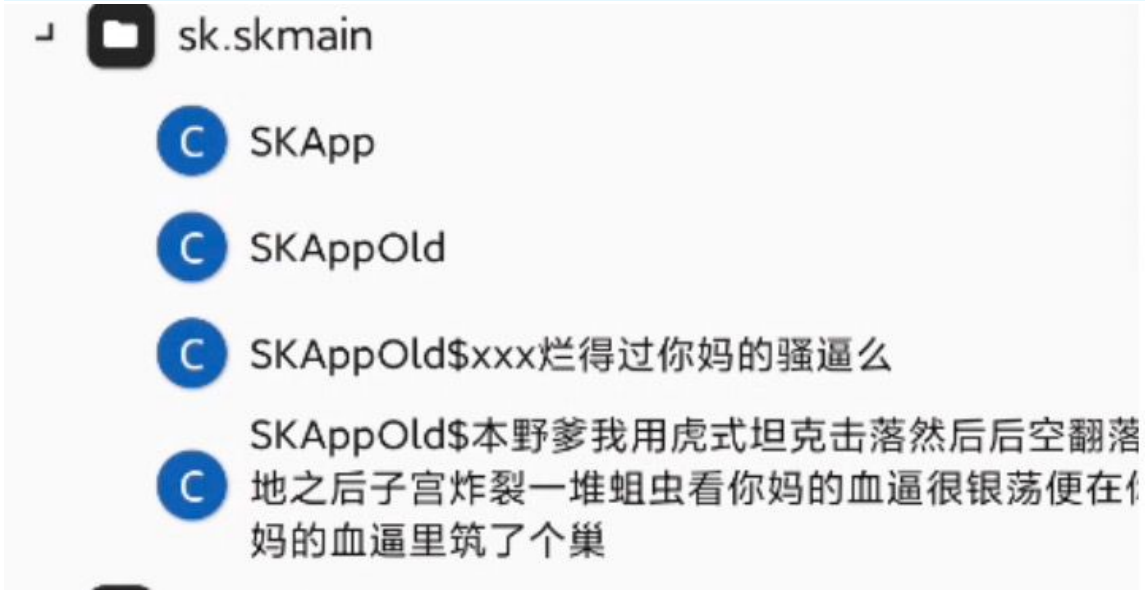

加固强度
10%
45%
75%
100%
Dex混淆
将Java层的包名、类、方法、属性名称混淆成没有意义的比如阿拉伯字符串以隐藏真实用途
抽取壳
将原Dex和方法体抽取出来单独加密保存，在执行到该方法前动态恢复，实现彻底隐藏Java方法的目的
反调试
逆向中一般会使用jeb/IDA/Frida等逆向工具，反调试可以对抗这些逆向工具使其失效无法使用
VMP
Dex的vmp会将方法的dalvik 指令转换成等价的自定义指令，so的vmp会实现一套自己的“语言”

由于开发和加固的不对等，简单的算法经过复杂的混淆可以达到爷娘都不认识的效果，比如可以把a=b+c混淆成右图所示的模样。逆向仨月终于还原出来a=b+c的时候精神是会崩溃的。



代码混淆、动态加载、Native开发、反调试/反反编译器、符号混淆/字符串加解密、花指令、SO加壳/方法体/动态释放、自定义linker、shellcode/armvmp

各种恶心
层出不穷
想得到的、想不到的
都能有！



新时代Frida“脚本小子”

基于JavaScript开发的Frida拥有速度快、范围广、效率高的特征，使用r0tracer可以一秒钟hook几万个类/方法/重载，日志中直接捞手机号即可定位关键函数！

速度 62%

效率 93%

反调试盛行

由于Frida效率实在太高，导致现在是个app都内置了frida反调试，只要用frida就会退出！

bypass很难

本来过反调试并不难，难的是过这些加固手段！简单的反调试经过加固后可以达到过滤95%脚本小子的效果！

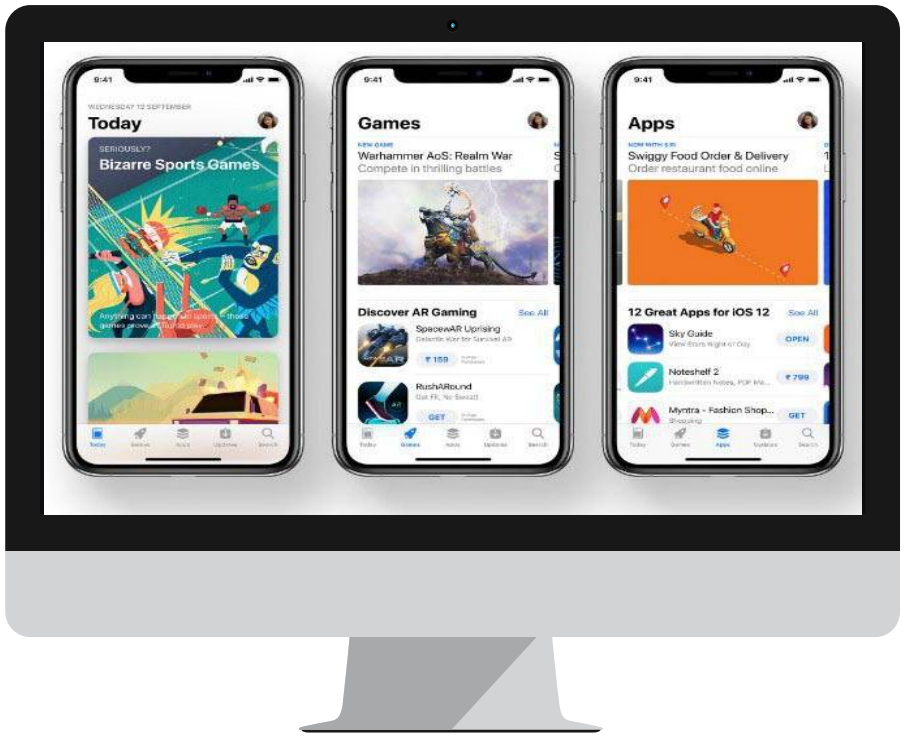
PART 02

封闭生态、蛮不讲理
→动辄下架/没有理由！

由于苹果大帝完全掌控了iOS应用分发的渠道，App想要抵达用户手中有且只有一条道路就是AppStore，由于摸不清楚苹果审核团队的脉门，导致应用不敢做额外的加固手段！

应用分发唯一途径


W2
MARVEL
HBO max
winnie
Lake


Today
Bizarre Sports Games
Games
Warhammer AoS: Realm War
Compete in thrilling battles
Discover AR Gaming
SpaceWAR Uprising
RushARound
Apps
Swiggy Food Order & Delivery
Order restaurant food online
12 Great Apps for iOS 12
Sky Guide
Noteshelf 2
Myntra - Fashion Shop...

- 源码封闭
- 生态封闭
- 证书校验
- 随时下架

防护手段

越狱检测

检测设备越狱之后产生的 文件/目录，

通过 NSFileManager、access、stat 、lstat、statfs、open、fopen 函数检测 文件/目录 的状态

Frida检测

检测 Frida 的相关文件同样使用了 NSFileManager、access、stat 、lstat、statfs、open、fopen 函数

代理检测

使用了 CFNetworkCopySystemProxySettings 函数检测代理状态。设置 connectionProxyDictionary 变量，可以防止 Burp、Charles 抓包工具抓包

调试检测

由常见的几种反调试函数：ptrace、syscall、sysctl 及其 “变种” 方式。检测 /Library/MobileSubstrate/DynamicLibraries 目录下的动态库。

PART 03

防护力度对比

Frida反调试

Github上开源检测库对比

安卓：检测Frida的命名空间/线程/内存读写权限/C库比对/字符串/maps遍历

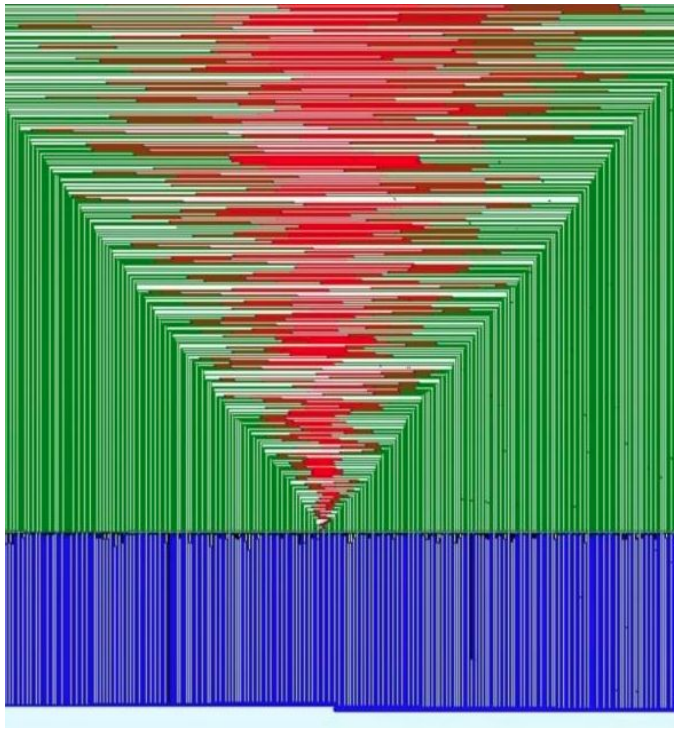
```
struct sockaddr_in addr;
addr.sin_family = AF_INET;          // AF_INET ipv4
addr.sin_addr.s_addr = inet_addr("127.0.0.1");
addr.sin_port = htons(27042);       // 终端输入 ns -l 27042

/**
    1 发送一个信号给服务端，在吗（ACK）
    2 服务端回一个（ACK）我在
    3 客服端，那我们就开始吧

    0：连接成功  -1：连接失败
 */
```

iOS:

- *文件检测*
- *端口检测*



01

系统调用

自己写汇编直接调用内核函数，防止直接hook libc来绕过反调试

02

函数重写

重写字符串比对、内存读写等操作，防止被frida最直接hook来绕过

在杜绝一切hook可能性的基础上，再使用最新的强混淆OLLVM，几乎无法逆向还原出原来的逻辑，使得hook绕过变得几乎不可能实现

企业壳

```
public final boolean onOptionsItemSelected(MenuItem a
    return x.z(0x1000006, new Object[]{this, arg3});
}

public final void setContentView(int arg3) {
    x.v(0x1000007, new Object[]{this, Integer.valueOf
}

public final void setContentView(View arg3) {
    x.v(0x1000008, new Object[]{this, arg3});
}

public final void setContentView(View arg3, ViewGroup
    x.v(0x1000009, new Object[]{this, arg3, arg4});
}

public final void setIntent(Intent arg3) {
    x.v(0x100000A, new Object[]{this, arg3});
}
```

企业服务、使命速达！

反调试、反抓包、反反编译、反逆向分析、
保护数据/资产、安全至上！

收费动辄七八位数的企业壳，集百十位巨佬十多年研发成果，
投资九位数研发的结晶，能给你一个脚本绕过了？！

企业壳
安卓

不允许加壳
iOS

同应用防护对比

小红书

iOS：

安卓：

爱奇艺

iOS：

安卓：

扶老二

iOS：

安卓：

得物

iOS：

安卓：

PART 04

r0cap/r0tracer → 支持iOS!

船新自动化逆向新体验!

r0capture

5k star

抓包必备

万千用户亲身体验、真实强大有效！

无视证书

绕过握手

证书只在SSL通信的初始化期间需要，r0cap工作在数据传输时

协议通杀

基础设备

r0cap工作在网络框架底层，所有的上层协议传输的必经之地！

框架通杀

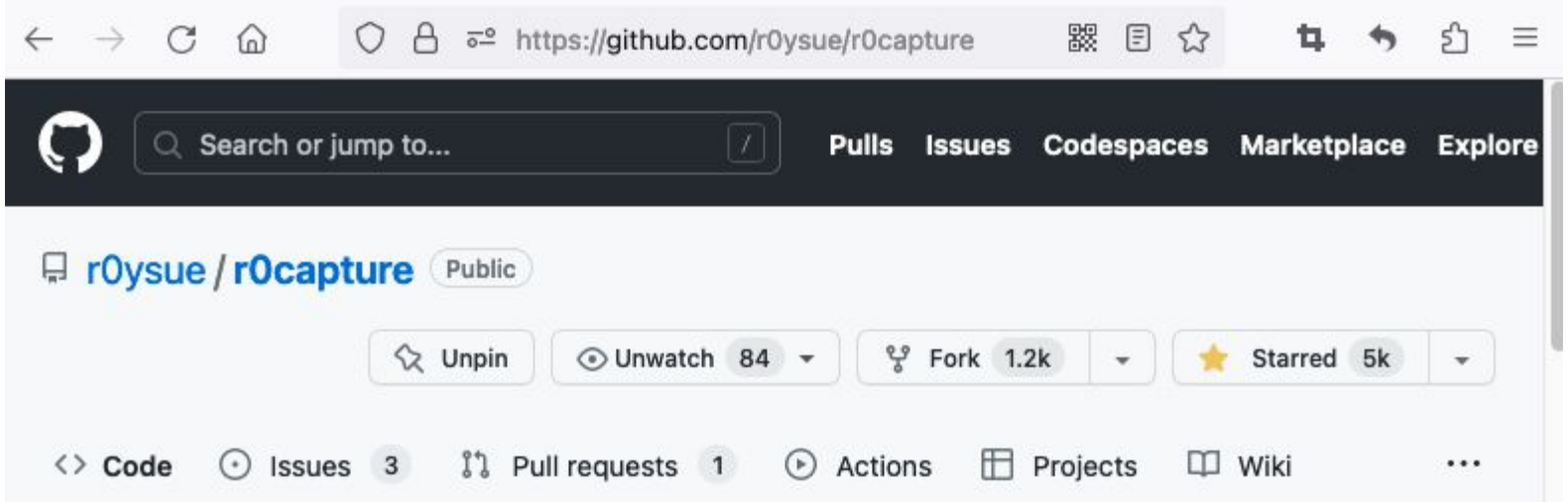
系统底层

r0cap工作在系统框架底层，上层应用调用sdk的必经之地！

无视加固

Frida自洽

使用Frida直接注入进程内存寻找符号转储流量！此时可执行文件已经是解密的状态



5k star!

没有抓不到的包！

系统API → 定位加解密 → 展示文字图片视频

```
File  Actions  Edit  View  Help
root@VXIDr0ysue: ~
        java.util.concurrent.ThreadPoolExecutor$Worker.run(ThreadPoolExecutor.java:636)
        java.lang.Thread.run(Thread.java:764)

(agent) [8y4wbshznyg] Arguments com.android.org.conscrypt.ConscryptFileDescriptorSocket$SSLInputStream.read
([object Object], (none), 8192)
(agent) [8y4wbshznyg] Return Value: 8192
(agent) [8y4wbshznyg] Called com.android.org.conscrypt.ConscryptFileDescriptorSocket$SSLInputStream.read([B
, int, int)
(agent) [8y4wbshznyg] Backtrace:
        com.android.org.conscrypt.ConscryptFileDescriptorSocket$SSLInputStream.read(Native Method)
        com.android.okhttp.okio.Okio$2.read(Okio.java:136)
        com.android.okhttp.okio.AsyncTimeout$2.read(AsyncTimeout.java:211)
        com.android.okhttp.okio.RealBufferedSource.read(RealBufferedSource.java:50)
        com.android.okhttp.internal.http.Http1xStream$FixedLengthSource.read(Http1xStream.java:393)
        com.android.okhttp.okio.RealBufferedSource$1.read(RealBufferedSource.java:371)
        com.google.android.exoplayer2.upstream.u.b(DefaultHttpDataSource.java:5)
        com.google.android.exoplayer2.upstream.u.a(DefaultHttpDataSource.java:38)
        com.google.android.exoplayer2.upstream.s.a(DefaultDataSource.java:31)
        com.google.android.exoplayer2.upstream.o.read(DataSourceInputStream.java:6)
        com.google.android.exoplayer2.upstream.o.read(DataSourceInputStream.java:3)
        javax.crypto.CipherInputStream.getMoreData(CipherInputStream.java:123)
        javax.crypto.CipherInputStream.read(CipherInputStream.java:249)
        com.google.android.exoplayer2.source.hls.d.a(Aes128DataSource.java:12)
        com.google.android.exoplayer2.upstream.e0.a(StatsDataSource.java:7)
        c.b.a.a.g1.d.a(DefaultExtractorInput.java:18)
        c.b.a.a.g1.d.a(DefaultExtractorInput.java:2)
        c.b.a.a.g1.y.g0.b(TsExtractor.java:11)
        c.b.a.a.g1.y.g0.a(TsExtractor.java:38)
        com.google.android.exoplayer2.source.hls.l.a(HlsMediaChunk.java:42)
        com.google.android.exoplayer2.source.hls.l.i(HlsMediaChunk.java:5)
        com.google.android.exoplayer2.source.hls.l.a(HlsMediaChunk.java:35)
        com.google.android.exoplayer2.upstream.b0$d.run(Loader.java:4)
        java.util.concurrent.ThreadPoolExecutor.runWorker(ThreadPoolExecutor.java:1162)
```

会话层：SSL

表示层：HTTP

应用层：自定义视频流解密

应用层：播放解密后的流媒体

```
File  Actions  Edit  View  Help
root@VXIDr0ysue: ~
android.net.LocalSocketImpl$SocketInputStream
android.os.ParcelFileDescriptor$AutoCloseInputStream
com.android.org.conscrypt.ConscryptFileDescriptorSocket$SSLIn
com.android.org.conscrypt.OpenSSLBIOInputStream
java.io.BufferedInputStream
java.io.ByteArrayInputStream
java.io.DataInputStream
java.io.FileInputStream
java.io.FileInputStream$UseManualSkipException
java.io.FilterInputStream
java.io.InputStream
java.io.InputStreamReader
java.io.ObjectInputStream
java.io.ObjectInputStream$BlockDataInputStream
java.io.ObjectInputStream$HandleTable
java.io.ObjectInputStream$HandleTable$HandleList
java.io.ObjectInputStream$PeekInputStream
java.io.ObjectInputStream$ValidationList
java.io.PushbackInputStream
java.io.SequenceInputStream
java.lang.ProcessBuilder$NullInputStream
java.lang.UNIXProcess$ProcessPipeInputStream
java.net.SocketInputStream
java.util.zip.CheckedInputStream
java.util.zip.GZIPInputStream
java.util.zip.GZIPInputStream$1
java.util.zip.InflaterInputStream
java.util.zip.ZipFile$ZipFileInflaterInputStream
java.util.zip.ZipFile$ZipFileInputStream
javax.crypto.CipherInputStream
sun.security.util.DerInputStream
root@VXIDr0ysue:~#
root@VXIDr0ysue:~#
```

枚举系统Socket框架

并批量自动化hook

r0tracer

https://github.com/r0ysue/r0tracer

枚举所有类加载器

由于Java有加固/热补丁/动态加载，有些类是后续加载的，所以报找不到类时需要切类加载器（安卓&iOS）

枚举所有方法

Java使用反射枚举所有方法，so和ObjC根据符号名搜索方法的符号的内存地址进行attach（安卓&iOS）

枚举所有类

根据关键字直接搜刮内存中的所有符号进行hook（安卓&iOS）

输出对象所有属性

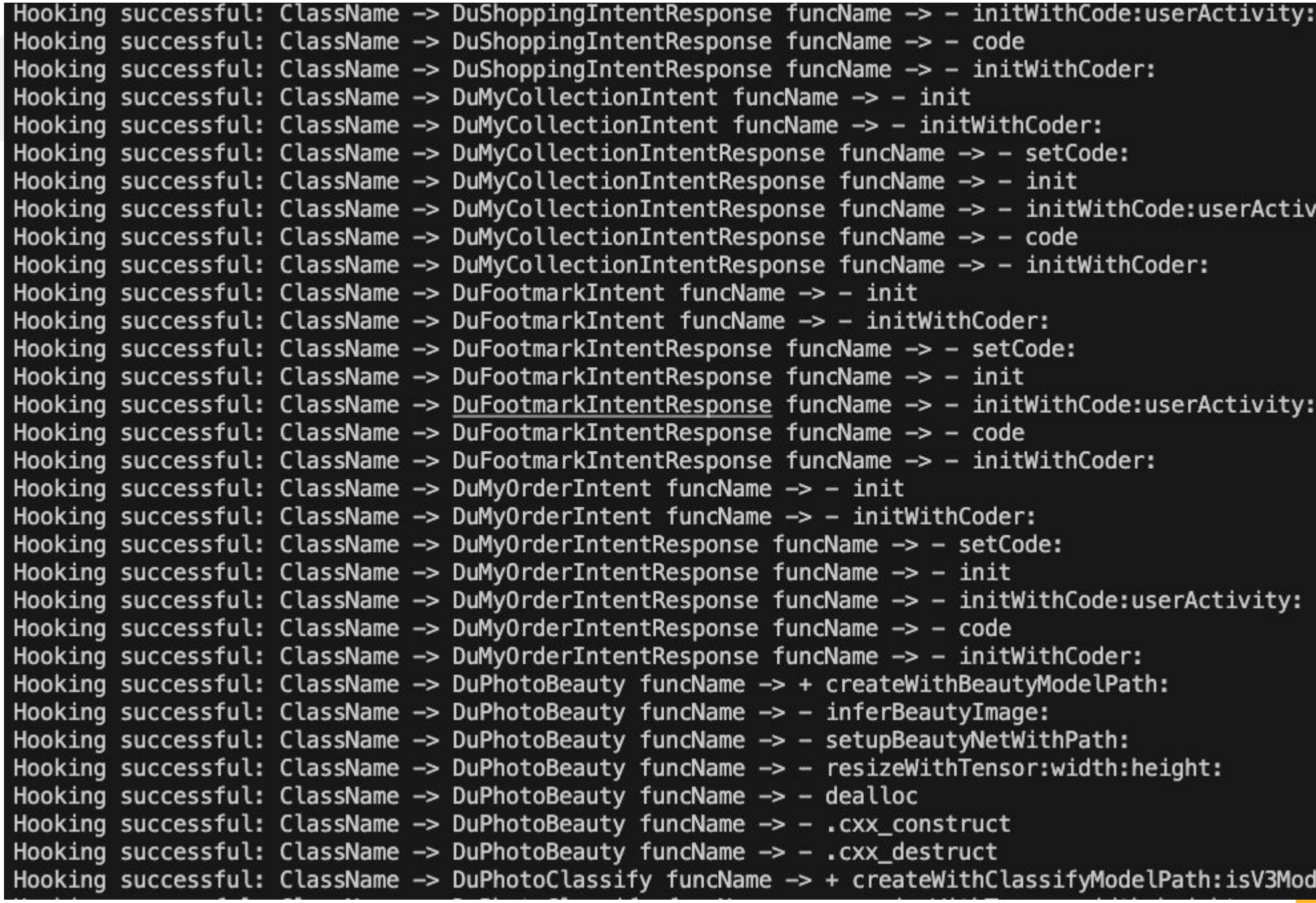
Java反射获取所有属性的值，ObjC通过$ivars获取属性的值，同时输出所有参数、调用栈、返回值，实现日志中直接搜电话号码的效果

运行时数据直接
定位关键函数

一秒钟hook
一万个方法！

参数里直接搜手机号

即可定位关键函数地址



运行时数据直接
定位关键函数

一秒钟hook
一万个方法！

参数里直接搜手机号

即可定位关键函数地址

userinfo1.txt

```
557  Inspecting Fields: => true => class com.chanson.business.model.BasicUserInfoBean
558  int     advantages => 1 => 1
559  int     age => 31 => 31
560  java.lang.String    annualIncome => 0 => "0"
561  int     appointment => 2 => 2
562  java.lang.String    avatar => https://cdn.kela77.com/images/public/default.png?x-oss-process=style/original
563  int     await => 0 => 0
564  int     birthday => 631196894 => 631196894
565  java.lang.String    bubbleText => 最近电影不错，一起看？ => "最近电影不错，一起看？"
566  java.lang.String    constellation => 摩羯座 => "摩羯座"
567  int     currentState => 1 => 1
568  java.lang.String    distance => 0.00km => "0.00km"
569  int     disturb => 0 => 0
570  int     education => 1 => 1
571  int     figure => 2 => 2
572  int     getAlong => 3 => 3
573  java.lang.String    gpsCity => 上海市 => "上海市"
574  boolean     hiddenSocial => false => false
575  java.lang.String    iconArea => 国权路 => "国权路"
576  java.lang.String    inviteType => 通过系统发放的邀请码加入克拉恋人 => "通过系统发放的邀请码加入克拉恋人"
577  boolean     isGoddess => undefined => undefined
578  boolean     isReal => undefined => undefined
579  boolean     isVip => undefined => undefined
580  java.lang.String    job => 8 => "8"
581  java.lang.String    nickname => anonymou => "anonymou"
582  java.lang.String    onlineTime => online => "online"
583  boolean     sendUnlock => true => true
584  int     sex => 2 => 2
585  int     vipLabel => 0 => 0
586  [native     function h() {
587      [native code]
588  } => undefined => undefined
589
```

三克油

---- vxid：r0ysue ----

上海电特安科技有限公司