# 目录

# 安卓逆向：Root环境检测及绕过

- 最新版本：`v0.6.0`
- 更新时间：`20250123`

## 简介

安卓逆向期间涉及到的，root环境检测和绕过root环境检测相关的各种内容。

## 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/android_re_root_env_detect_bypass: 安卓逆向：Root环境检测及绕过

### 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

- 安卓逆向：Root环境检测及绕过 book.crifan.org
- 安卓逆向：Root环境检测及绕过 crifan.github.io

### 离线下载阅读

- 安卓逆向：Root环境检测及绕过 PDF
- 安卓逆向：Root环境检测及绕过 ePub
- 安卓逆向：Root环境检测及绕过 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

# 安卓root环境检测概览

安卓逆向期间，涉及到：

- 正向的：root环境检测
- 逆向的：绕过root环境检测 = 反root环境检测

其中如果是涉及到调试的部分，往往算另外一个子领域：

- 正向：禁止被人调试你的安卓app == 反调试
- 逆向：绕过禁止调试 == 绕过反调试 == 反反调试

下面详细介绍具体内容。

# 检测工具

# Hunter

# momo

# 检测内容

TODO：

- 安卓 反调试 环境检测点汇总 hook( 面具 xp ida frida )检测点 和 绕过策略_安卓 检查代码是否被hook-CSDN博客

---

部分要点：

绕过脚本参考:

- anti_fgets();
- anti_exit();
- anti_fork();
- anti_kill();
- anti_ptrace();

https://github.com/deathmemory/FridaContainer/blob/011c205d5ef1372b697ed04a66f69b8fd75680ed/utils/android/Anti.ts

签名校验之类:

https://github.com/axhlzy/ll2CppHookScripts/blob/862b39b695540008ff06ed2b15e1a1ec98b44fa3/Others/FTS/fts.js

系统：

- frida syscalls https://github.com/FrenchYeti/frida-syscall/blob/main/index.js
- frida syscalls https://github.com/AeonLucid/frida-syscall-interceptor

写的修改系统源码：

https://blog.csdn.net/weixin_42453905/article/details/122462984?spm=1001.2014.3001.5501

# Root文件列表

# Root文件列表检测

TODO：

- 把常见的root文件列表整理过来

# 绕过Root文件列表检测

- 思路：Frida或xposed去hook掉
  - 相关的常见的安卓中的root文件列表

# Frida

TODO：

- 【整理】Android安卓中Frida检测相关手段

---

- 正向：Frida的检测
  - = 反Frida调试
- 逆向：绕过Frida检测
  - = 过Frida检测
  - = 反Firda检测
  - = 反反Frida调试
  - = Frida反反调试

# Frida检测

- Frida检测
  - 要点
    - 检测更多的frida的特征字符串
    - 检测更深的hook特征
    - 尽量使用双进程保护
    - 尽量使用svc去获取数据
    - 保护检测线程不被干掉
- 资料
  - 常规检测参考
    - https://github.com/qtfreet00/AntiFrida
    - https://github.com/xxr0ss/AntiFrida
    - https://github.com/emanriquez/antifridaAndroid
  - 进阶检测参考
    - [原创]关于frida检测的一个新思路
    - 从inlinehook角度检测frida
    - https://github.com/TUGOhost/anti_Android
    - https://github.com/kumar-rahul/detectfridalib
    - 从Frida对SO的解析流程中寻找检测Frida的点

# 绕过Frida检测

- 绕过frida检测 = 反调试
  - 核心思路=核心的点
    - 把 `frida-server` 的文件名改掉
      - 举例
        - `frida-server-12.8.9-android-arm64` -> `fs1289amd64`
    - 改变端口号
      - 举例
        - `frida-server` 的默认端口是 `27042`，改为别的端口号
    - 通过Frida内存特征对maps中elf文件进行扫描匹配特征
    - 其他反调试思路
      - 想过这种反调试，得找到反调试在哪个so的哪里，nop掉创建 `check_loop` 线程的地方，或者 `nop` 掉kill自己进程的地方，都可以。也可以直接kill掉反调试进程
        - 举例
          - 别人就曾经遇到过这种情况，frida命令注入后，app调不起来，这时候用 `ps -e` 命令查看多一个反调试进程，直接kill掉那个进程后，app就起来了，这个app是使用的一个大厂的加固服务，这个进程就是壳的一部分
  - 常用方案
    - 使用魔改去特征的版本的frida
      - hluda
      - Florida
      - Froda ?

# 相关源码

## AntiFrida

https://github.com/qtfreet00/AntiFrida

核心代码：

```
void *check_loop(void *) {
    int fd;
    char path[256];
    char perm[5];
    unsigned long offset;
    unsigned int base;
    long end;
    char buffer[BUFFER_LEN];
    int loop = 0;
    unsigned int length = 11;
    //"frida:rpc"的内存布局特征
    unsigned char frida_rpc[] =
            {
                    0xfe, 0xba, 0xfb, 0x4a, 0x0a, 0xca, 0x7f, 0xfb,
                    0xdb, 0xea, 0xfe, 0xdc
            };
    for (unsigned char &m : frida_rpc) {
        unsigned char c = m;
        c = ~c;
        c ^= 0xb1;
        c = (c >> 0x6) | (c << 0x2);
        c ^= 0x4a;
        c = (c >> 0x6) | (c << 0x2);
        m = c;
    }
    //开始检测frida反调试循环
    LOGI("start check frida loop");
    while (loop < 10) {
```

```
        fd = wrap_openat(AT_FDCWD, "/proc/self/maps", O_RDONLY, 0);
        if (fd > 0) {
            while ((read_line(fd, buffer, BUFFER_LEN)) > 0) {
                // 匹配frida-server和frida-gadget的内存特征
                if (sscanf(buffer, "%x-%lx %4s %lx %*s %*s %s", &base, &end, perm, &offset, path) !=
                    5) {
                    continue;
                }
                if (perm[0] != 'r') continue;
                if (perm[3] != 'p') continue;
                if (0 != offset) continue;
                if (strlen(path) == 0) continue;
                if ('[' == path[0]) continue;
                if (end - base <= 1000000) continue;
                if (wrap_endsWith(path, ".oat")) continue;
                if (elf_check_header(base) != 1) continue;
                if (find_mem_string(base, end, frida_rpc, length) == 1) {
                    //发现其内存特征
                    LOGI("frida found in memory!");
#ifndef DEBUG
                    //杀掉自己的进程
                    wrap_kill(wrap_getpid(),SIGKILL);
#endif
                    //退出
                    break;
                }
            }
        } else {
            LOGI("open maps error");
        }
        wrap_close(fd);
        //休息三秒，进入下一个检查循环，也就是这个反调试一共会运作30秒，30秒后结束
        loop ++;
        sleep(3);
    }
    return nullptr;
}
void anti_frida_loop() {
    pthread_t t;
    //创建一个线程，执行反调试工作
    if (pthread_create(&t, nullptr, check_loop, (void *) nullptr) != 0) {
        exit(-1);
    };
    pthread_detach(t);
}
```

# hluda

- 定制版Frida=魔改版Frida：hluda == strongR-frida-android
  - Github代码
    - https://github.com/hzzheyang/strongR-frida-android
      - hzzheyang/strongR-frida-android: An anti detection version frida-server for android. (github.com)
        - 由于工作太忙，长时间没搞这个了，所以借用了一下Ylarod的patch，以后这里没更新的话，可以去Ylarod看看
  - 核心二进制
    - hluda-server
      - 直接下载
        - https://github.com/hzzheyang/strongR-frida-android/releases
          - Releases · hzzheyang/strongR-frida-android (github.com)
      - 自己编译
        - [原创]手动编译Hluda Frida Server-Android安全-看雪-安全社区|安全招聘|kanxue.com
  - 相关
    - Fork自 = 原始仓库
      - https://github.com/CrackerCat/strongR-frida-android
        - CrackerCat/strongR-frida-android: An anti detection version frida-server for android. (github.com)
          - 但是原始仓库：没有release编译好的，可下载的文件
    - 其他工具
      - https://github.com/Exo1i/MagiskHluda
        - Exo1i/MagiskHluda: Run a more undetectable frida-server on boot using magisk (github.com)
          - Florida on Boot: Unleash Florida's Power with Enhanced Undetectability
          - This a magisk module based on Florida which is a more undetectable version of Frida.
          - It basically sets up Florida to start on boot for you.
        - -》直接安装即可用

# 详细介绍

hluda-server-15.1.21 是一个经过魔改的版本的 Frida-server，主要针对 x86-64 和 arm64 架构的设备。Frida 是一款强大的动态代码插桩工具，它允许开发者在运行时对应用程序进行修改、调试和逆向工程。这个“魔改版”主要是为了绕过某些应用或系统对于 Frida 的检测机制，以实现更加隐蔽的功能。

## Frida-server

- Frida-server 是 Frida 的一部分，它是一个可以在目标设备上运行的代理服务器，与宿主机上的 Frida 客户端配合工作。
- Frida-server 可以注入 JavaScript 代码到目标应用的进程中，允许在运行时分析和修改应用的行为。
- 在 Android 平台上，通常通过 ADB (Android Debug Bridge) 将 Frida-server 推送到设备上，并启动服务，然后在宿主机上连接并执行脚本。

## 魔改（Magic Modification）

- 魔改版 hluda-server 去除了 Frida 的一些特征码，以避免被反调试和反病毒软件检测到。
- 这种修改可能包括但不限于重写签名验证、去除特定字符串、混淆代码结构等，以提高隐蔽性。
- 使用魔改版可以降低被目标应用或者安全防护系统识别为恶意软件的风险。

## x86-64 和 arm64 支持

- x86-64 版本适用于基于 x86 或 x86_64 架构的设备，如某些模拟器或特定的 Android 设备。
- arm64 版本则针对基于 ARMv8 架构的 64 位设备，这是现代 Android 设备的主流架构。

- 提供这两个版本的 hluda-server 意味着它能广泛地覆盖不同硬件平台的用户。

## 逆向工程和调试

- Frida 的核心功能是用于逆向工程，开发者可以使用它来获取应用内部的工作原理，查找漏洞，或测试安全措施。
- 由于 hluda-server 能避开检测，它特别适用于那些对调试有严格限制或者有自我保护机制的应用。

## 安全风险

- 魔改版工具虽然提高了隐蔽性，但可能会被滥用，例如进行非法的恶意活动，因此使用时应确保遵守法律法规。
- 对于普通用户，未经许可的设备调试可能侵犯隐私，因此在使用 Frida 或 hluda-server 时，务必确保拥有对目标应用的合法访问权限。

## 使用方法

- 安装 hluda-server 之前，确保设备已开启调试模式，且 ADB 已正确配置。
- 通过 ADB 将对应的 hluda-server 安装包推送到设备上，并按照官方文档的指示启动服务。
- 在宿主机上安装 Frida 客户端，通过连接 hluda-server 实现远程控制和脚本执行。

总结来说，hluda-server-15.1.21 是一个为绕过检测而优化的 Frida-server 版本，适用于多平台，尤其适合那些需要避开反调试机制的场景，如安全研究、应用测试和逆向工程。然而，使用此类工具时必须注意合法性和道德问题，以避免不必要的法律风险。

# Florida

- Florida
  - Github
    - https://github.com/Ylarod/Florida
      - Ylarod/Florida: 基础反检测 frida-server / Basic anti-detection frida-server
        - Follow FRIDA upstream to automatic patch and build an anti-detection version of frida-server for android.
        - 跟随 FRIDA 上游自动修补程序，并为 Android 构建反检测版本的 frida-server
        - 基础反检测 frida-server / Basic anti-detection frida-server

# Froda

- 定制版Frida=魔改版Frida：froda
  - https://github.com/xyxdaily/frida-detect-protect
    - xyxdaily/frida-detect-protect (github.com)
      - froda（课程相关，暂时未公开）
      - 首先这里感谢hluda的开源定制，因为froda是基于hluda，然后再定制了一些额外的字符串（frida:rpc等），以及线程名，绕过了常见的字符串检测。

# Xposed

- 检测Xposed
  - 细节：检测各种常用插件

# adb

- adb检测
  - 调用相关系统接口?
- 绕过adb的检测
  - 思路:
    - Frida或xposed去hook掉，相关的接口调用

# 其他

# 附录

下面列出相关参考资料。

# 参考资料


- 【整理】安卓中绕过Frida检测相关资料
-
- bill-bil/20210123-200754-467: 抖音数据采集Frida进阶：脱壳、自动化、高频问题 (github.com)
- 安卓 反调试 环境检测点汇总 hook( 面具 xp ida frida )检测点 和 绕过策略_安卓 检查代码是否被hook-CSDN博客
- [原创]手动编译Hluda Frida Server-Android安全-看雪-安全社区|安全招聘|kanxue.com
- hluda，绕过frida检测_libmsaoaidsec.so资源-CSDN文库
- hluda-server-15.1.21(魔改版frida-server)，里面包含x86-64和arm64版本_frida魔改资源-CSDN文库
- Detecting Frida the “Right” way. It’s very common for an application to… | by Aimar Sechan Adhitya | Medium
-