**第一部分 ADB抓取日志**

**整体目标：**

1）抓取android日志，并存储到本地（PC 还是 手机）

2）尤其是针对APP的所有请求和用户行为，以及各种数据

3）从中分析例如用户名/口令/载荷等字段，并还原用户行为，进行行为重现/行为分析

4）后续有两种手段：获取用户名+口令 或者 接管APP，修改载荷

**思路或工具：**

1）libdvm.so或libart.so探针，即 修改后重新 编译生成的Dalvik虚拟机（部分编译）

源码目录： Dalvik / vm / dalvik

输出目录： out / host / linux-x86 / lib / libdvm.so

2）得到的身份认证信息(数据)可能经过加密或者码制变换，利用des开源解析工具

或者手动测试码制

3）虚拟机可能做不通，建议真机。但真机(非Nexus)无法下系统，虚拟机拿数据是否？

**基本指令：**

1）adb start-server + adb kill-server 开启/关闭服务器

2）adb devices 查看当前连接的设备；adb shell可以进入安卓系统(类Linux)命令行

3）抓取日志

adb logcat -v time 1>.\\logcat.log 输出时间+日志，重定向到当前目录下的logcat.log 文件（.\\logcat.log）中

adb logcat -b main -v time>app.log 打印应用程序的log

adb logcat -b radio -v time>radio.log 打印射频相关的log,SIM STK也会在里面

adb logcat -b events -v time 打印系统事件的log,如触屏事件

adb shell cat /proc/kmsg 内核日志的抓取

4）Log.v 的调试颜色为黑色的，任何消息都会输出

Log.d的输出颜色是蓝色的，仅输出debug调试

Log.i的输出为绿色，一般提示性的消息information

Log.w的意思为橙色，可以看作为warning警告

Log.e为红色，显示错误信息

**当前进展（Windows）：**

1）ADB配置：利用ADB（android develop bridge）可以输出日志信息到本地，原理在于 以电脑为客户端，手机为服务器端，调用手机命令行

<http://jingyan.baidu.com/article/eb9f7b6d6fae86869364e88c.html>

经测试（仅充电 或者 存储设备 等模式并不影响。。）

---------------------------------------------------------------------------------------------------------------------------------

当前问题：

在C:/adb安装了本工具，但是adb logcat ... 后 显示“找不到设备”，网上分析是 找不到驱动，其实都是瞎扯

解决方案：

最后一种方法：<http://www.cnblogs.com/yanzisucheng/p/4369266.html>

路径为C:\Users\fujia\.android

---------------------------------------------------------------------------------------------------------------------------------

当前问题：

若是驱动问题，在设备管理器（我的电脑右击，第一个就是）中会有黄色感叹号

解决方案：

1.尝试在设备管理器中修改，更新驱动

2.如果不行的话，下载豌豆荚等手机助手，会自动安装驱动，后面卸载即可

---------------------------------------------------------------------------------------------------------------------------------

当前问题：

在 adb devices中有显示，但是为offline,而且 adb logcat 还是没有效果

解决方案：

<http://jingyan.baidu.com/article/e8cdb32b435ce637052bad14.html>

android 版本与adb版本不兼容 更新adb版本至1.0.32

---------------------------------------------------------------------------------------------------------------------------------

---------------------------------------------------------------------------------------------------------------------------------

2）日志抓取：通过 adb logcat -v time 1>.\\logcat.log 终于有内容写入了，格式如下：

01-18 14:22:53.500 D/PowerManager( 481): AlarmManager [acquire]

Time 信息种类(VDIWE)/TAG PID MESSAGE

**进一步工作（Windows）：**

目标一：

看起来日志是杂乱无章的，如何解析日志（最终面向单个APP，所有行为）

相关资源：

1. [基本的实时日志 格式 + 种类 + 显示模式 + 过滤方法](http://www.cnblogs.com/JianXu/p/5468839.html)

adb logcat -v time -d > .\\logcatt1.log

adb logcat -v time -d -s TAG > .\\logcatt2.log

adb logcat -v time -d -s TAG:W > .\\logcatt3.log

adb logcat -c 清空缓存区

adb logcat -v time -b main -d -s TAG:W > .\\logcatt4.log

adb logcat -v time -b main -d -s TAG:W \*:S > .\\logcatt5.log

2. Main\_log : app及framework相关活动的log

Events\_log: AvtivityManager及PowerManager等事件相关log

Kernel\_log:驱动相关的log

Net\_log:网络相关的，网络链接情况，抓取网络包

Modern\_log:通话相关的，信号强弱，连接的哪个网络

源码中应该有对应路径

3. [分析实例一](http://blog.chinaunix.net/uid-20459533-id-3109996.html)，但感觉关系不大，这个是用logcat进行debug

[分析实例二](http://blog.csdn.net/ameyume/article/details/7667574)，区分了logcat的实时日志和其他各类日志，分门别类/不同路径

目标二：

在安卓开发过程中实时获取logcat日志信息（自己写的日志+系统日志）

相关思路：

1. APP源码：监听安卓程序的一个按钮或者其他，一旦点击按钮，事件触发，将各 种日志都打印出来

<http://www.jb51.net/article/79511.htm>

<http://www.jb51.net/article/43952.htm>

2.系统源码：在编译so文件的c/c++文件之前加入几行简单代码，并简单修改配置 文件 -> 编译

C++ ：<https://zhidao.baidu.com/question/1384240857848391780>

JAVA : <http://blog.sina.com.cn/s/blog_4a4f9fb50100x9x3.html>

<http://www.iteye.com/problems/85431>

[真机获得日志，不连接电脑 -- Java 源码](http://easion-zms.iteye.com/blog/981568)

<http://www.android100.org/html/201302/10/1406.html>

<http://blog.chinaunix.net/uid-20459533-id-3109996.html>

相关工具及资源：

[Android系统的日志系统及权限系统 -- java源码](http://www.cnblogs.com/androidNot/p/5613230.html)

[安卓日志分析（ppt）](http://wenku.baidu.com/link?url=y5e8o7H8GM47z6ygomAHJa9exQUVWzEaOcT9TSpUtgp3aob5XTYi52XTXmGXXAQtGowK2406IGSrq3eKThiJzEfcblFrFs8dTflL5x_cyDW )

**参考资源（基本原理）：**

1. [Android源码中java和c++的关系](http://blog.csdn.net/seniorwizard/article/details/7693573)

Java Native Interface (JNI) ：JAVA代码与其他代码相互调用的机制

Native层：都是C++ 应用层：虚拟机，都是JAVA类库

2. [dalvik虚拟机简要介绍与学习计划（系列4-5篇）](http://blog.csdn.net/luoshengyang/article/details/8852432)

即时编译（JIT）+ 垃圾回收（GC）+ Java本地方法调用（JNI）+ Java远程调试协议（JDWP）

[深入理解dalvik](http://www.infoq.com/cn/articles/android-in-depth-dalvik)

so是动态库，对应于Windows里面的dll，一个APP对应一个虚拟机+一个.so

3. [dalvik虚拟机在android架构中的位置](http://www.oschina.net/p/dalvik/)

[ART运行无缝替换Dalvik虚拟机](http://blog.csdn.net/luoshengyang/article/details/18006645)

|  |  |
| --- | --- |
| Dalvik | ART |
| Android4.4(Kitkat)同时包含，Android5.0完全淘汰Dalvik，改变了APP的根本运行机制 | |
| JIT(Just In Time) | AOT(Ahead Of Time) |
| libdvm.so | libart.so |

---------------------------------------------------------分界线--------------------------------------------------------------

上述完成工作就是利用adb logcat来抓取日志，当前实现了基本的日志抓取功能，不足在于：1）手动的话 要连电脑，自动的话怎么实现更高级？更科学

1. 对于抓取到的日志怎样分析，一是定位到特定目标，二是怎么重现该行为，三是完整性
2. 是否可以自己写脚本过滤日志，定位APP，例如 时间段过滤
3. Logcat在模拟器emulator环境是否可行

......

---------------------------------------------------------分界线--------------------------------------------------------------

**第二部分 android源代码编译**

**整体目标：**

1）安卓系统代码层面 修改/抓取日志，为logcat做准备

2）APP源码层面 修改/抓取日志，为logcat做准备

3）构建emulator实验环境 ubuntu14.04 + android 4.4.4

**思路或工具：**

1）需要先配环境（基本就是linux系统安装，安卓系统代码和内核代码下载，系统编 译环境配置，编译系统+内核代码）

2）下真机 -- 编译安卓源码下真机，突然明白就是“刷机”

大概思路：编译后的一堆img,在同一路径下，直接fastboot即可（机型？）

具体说明：对照官方rom包，验证了本思路，因为./flash-all.sh完成的即为本工作 <https://www.zhihu.com/question/19824640>

**当前进展（Linux）：**

1）配环境 -- Linux (Ubuntu 最好双系统 + 空间足够大 100G左右)

<http://blog.csdn.net/liu1075538266/article/details/51272398>

较为完整的 配环境 + 下载android源代码 + 下载驱动 + 编译 + 下真机

1.对照AOSP（android opensource project）（安卓版本 -- Ubuntu版本+软硬件）

<http://source.android.com/source/requirements.html>

要编译android4.4.x (Kitkat) 版本的源码 Linux操作系统为64位Ubuntu 14.04，磁盘空间预留100G，JDK版本为1.6，make 版本3.81，python 版 本2.7.6，Git1.3及以上（这里获取源码不用Git+repo的方法，故暂不需要）

2. 安装JDK <http://blog.csdn.net/jiangwei0910410003/article/details/37996723>

3. 对照官网安装 其他依赖包

<http://source.android.com/source/initializing.html>

<http://blog.csdn.net/jiangwei0910410003/article/details/37988637>

4. 对源码和驱动版本的一些准备

<http://source.android.com/source/build-numbers.html>

第一列是build代码，我们根据build码查找对应设备的驱动程序，并下载； 第二列是分支（branch）代码，不同的分支会对应不同的设备；

第三列是Android版本代号，这里我们关注的是4.4系列的版本（KitKat）； 第四列是对应的设备名称，值得注意的是，同一设备即便是不同运营商发布 的也可能使用不同分支的源码，例如android-4.4.4\_r2就只针对上述运营 商的定制版本的分支，而r1就支持其余的Nexus 5设备

5. 源码的下载：

1）官网上推荐Git+repo的方式，但是<https://android.googlesource.com/> 在 正常情况下无法访问，因此我们不采取官方提供的方式获取源码

2）到 百度云网盘 上面找对应版本源代码

注意：不要在Windows中解压，用图形化工具解压会缺少内容，linux解压

7z格式解压 <http://www.educity.cn/linux/1241489.html>

Linux压缩<http://blog.csdn.net/jiangwei0910410003/article/details/37997899>

6. Ccache编译的加速 <http://blog.chinaunix.net/uid-29140689-id-4033900.html>

结合两个文档的介绍，综合一下

7. 驱动的下载即脚本执行 <http://www.jb51.net/article/53924.htm>

8. 开始编译 make -j4(电脑核数的二倍)

---------------------------------------------------------------------------------------------------------------------------------

问题一：unstable问题（原因是系统的源 版本过低或以失效）

解决方案：打开文件，原来的源删掉，新的源粘贴过来。其中extra..注释掉

<http://blog.csdn.net/lvchaoshun/article/details/52245936>

<http://wiki.ubuntu.org.cn/%E6%A8%A1%E6%9D%BF:14.04source>

---------------------------------------------------------------------------------------------------------------------------------问题二：collect2：Id terminated with signal 9错误 --> swap/RAM分配不足

解决方案： <http://blog.csdn.net/q35126156/article/details/53514187>

最后一步提示权限不够，直接打开文本修改就好

---------------------------------------------------------------------------------------------------------------------------------

---------------------------------------------------------------------------------------------------------------------------------

9.经过超过10小时编译（这里未用到ccache..）,编译成功（Nexus Ham版本的）

10.下机有风险，先编译一个模拟器版本（lunch -> 1，之前真机lunch时选6对应)

1）在Linux中启动模拟器：

1.1 在源代码目录下直接输入emulator，默认会启动自带模拟器

重启或关闭之后，先要.（即source） Build/envsetup.sh + lunch之后才能找 到emulator指令

1.2 在源代码目录下输入完整指令

emulator –kernel  ./prebuilt/android-arm/kernel/ kernel-qemu-armv7 – sysdir./out/target/product/generic –system system.img –data userdata.img –ramdiskramdisk.img

额外输入指令

emulator -partition-size 500 还可以设置启动的内存大小

2）在Windows中启动模拟器（前提：安装Android SDK）

...

11.Linux系统下尝试下真机

1）Cd到多个img的目录之上 ==> ham或者genetic文件夹下

---------------------------------------------------------------------------------------------------------------------------------问题一：adb reboot bootloader 时找不到设备

解决方法：<http://blog.csdn.net/fireness/article/details/50060477>

---------------------------------------------------------------------------------------------------------------------------------问题二：adb reboot bootloader 时显示no permission

解决方法：<https://zhidao.baidu.com/question/589239941.html>

---------------------------------------------------------------------------------------------------------------------------------

问题三：fastboot时，显示waiting for device

解决办法：在指令前增加 sudo，给予root权限（否则权限不够）

---------------------------------------------------------------------------------------------------------------------------------问题四：fastboot时， [error: neither -p product specified nor ANDROID\_PRODUCT\_OUT set](http://blog.csdn.net/hejianhua1/article/details/50311715)

解决办法：<http://blog.csdn.net/hejianhua1/article/details/50311715> Note: -p 在 -w后面

---------------------------------------------------------------------------------------------------------------------------------

---------------------------------------------------------------------------------------------------------------------------------

最终结果：发现确实 非Nexus设备不能下载原生系统

**参考资源（基本原理）：**

1.Build 结果的目录结构

所有的编译产物都将位于 /out目录下，该目录下主要有以下几个子目录：

/out/host/：该目录下包含了针对主机的 Android开发工具的产物。即 SDK中的各种工 具，例如：emulator，adb，aapt等。

/out/target/common/：该目录下包含了针对设备的共通的编译产物，主要是 Java应用 代码和 Java 库。

/out/target/product/<product\_name>/：包含了特定设备的编译结果以及平台相 关 的 C/C++库和二进制文件。其中，<product\_name>是具体目标设备的名称。

/out/dist/：包含了为多种分发而准备的包，通过“make disttarget”将文件拷贝到该目录， 默认的编译目标不会产生该目录。

2..Build 生成的镜像文件

Build 的产物中最重要的是三个镜像文件，它们都位 于 /out/target/product/<product\_name>/目录下。

system.img：包含了 Android OS的系统文件，库，可执行文件以及预置的应用 程序，将被挂载为根分区。

ramdisk.img：在启动时将被 Linux内核挂载为只读分区，它包含了 /init文件和 一些配 置文件。它用来挂载其他系统镜像并启动 init进程。

userdata.img：将被挂载为 /data，包含了应用程序相关的数据以及和用户相关的数据。

3. [android系统 源代码阅读指南](http://blog.csdn.net/watermusicyes/article/details/38531537)

系统的讲解了阅读android源代码的步骤 并罗列了众多链接资源

**其他版本：**

1.编译安卓源码 -- 详细步骤

1) Ubuntu 12.04 + Android 4.4.4\_r1/4.4.4\_r2/4.0.3\_r6/2.3\_x/5.1.1\_r6/5.1.1\_r12

<http://blog.csdn.net/u011422742/article/details/52068327>

虚拟机编译源码 + 模拟器 运行（模）+ 部分编译 + 内置app

2) Ubuntu 16.04 + Android 6.0.1

<http://blog.csdn.net/fuchaosz/article/details/51487585> 编译Android源码（模）

<http://blog.csdn.net/fuchaosz/article/details/52473660> 下真机各种详细分析

上一篇的基础上，进行补充，包含更为详细的 配环境+下载源代码+基础知识

3) Ubuntu 11.10 + Android 4.0.1

<http://blog.csdn.net/hens007/article/details/7406452> android环境建立

<http://blog.csdn.net/hens007/article/details/7465574> 编译源代码

4) Ubuntu 11.10 + Android 4.0

<http://blog.csdn.net/flydream0/article/details/7036156>

<http://blog.csdn.net/flydream0/article/details/7046612>

2. 安卓源码与系统源码 -- 区别

1) <http://www.tuicool.com/articles/RjeEZb> 安卓源码的版本与分支详解

2) <http://www.cnblogs.com/andrew-wang/archive/2012/11/05/2755222.html>

安卓系统和Linux内核的关系详解

3) <http://blog.sina.com.cn/s/blog_b47e52490102uztf.html> 安卓与内核源码区别

<http://blog.csdn.net/flydream0/article/details/7070392> 下载并编译内核代码

内附 -- 下载android4.0源码 + 编译android4.0源码

4) <http://blog.csdn.net/c15522627353/article/details/51151156>

编译源码和内核代码 归纳总结

<http://www.linuxidc.com/Linux/2016-01/127291.htm> 问题汇总（参考）

3. 下真机（与第一个相同。。属于简略版->缺少源代码下载等，重点在下真机）

<http://blog.csdn.net/u011422742/article/details/52253637>

源码版本 - - > build驱动下载 - - > 驱动生成/安装脚本 - - > 编译 - - >刷机

---------------------------------------------------------分界线--------------------------------------------------------------

新思路 -- 直接修改系统源码，重新编译后下到真机

整体思路：

1. 配环境 + 下载系统源码/内核源码 + 编译成功 + 下真机
2. 上一阶段可能会花费很长时间，各种不同版本（Linux + JDK + android + 驱动 + 内核）

最好可以综合调试一些

1. 修改源码代码 + 重新编译 + 下真机

在实验室机器上发现：

如果选中 完全模式安装Linux,会修改系统启动设置，无法安装Windows

---------------------------------------------------------分界线--------------------------------------------------------------

**第三部分 抓取数据**

**整体目标：**

1. 抓取APP相关的各类数据，构建本地存储
2. 区分数据：验证信息（用户名 口令 验证码 认证链等）+ 业务信息（内容）
3. 配合抓取到的日志，实现 行为重现/分析/APP缺陷/本地因素最小集等

**思路或工具：**

1. 本地数据：

[词典demo获取本地数据](http://www.jianshu.com/p/bc02c809772a)

安卓获取程序/DB路径 <http://www.cnblogs.com/bluestorm/p/3688937.html>

安卓本地数据存储 <http://www.jianshu.com/p/3d73366093c3>

2.网络数据：

1）APP抓包工具Fiddler

Fiddler基本原理 <https://my.oschina.net/u/587105/blog/322952>

Fiddler实例 <http://www.cnblogs.com/softlin/p/4966709.html>

<https://my.oschina.net/jhao104/blog/605963>

<http://www.cnblogs.com/wanghaixing/p/6262367.html>

2）Python爬虫

<https://my.oschina.net/jhao104/blog/606922>

**参考资源：**