Intent IntentFilter参考<http://www.jianshu.com/p/a2f826064e29>

http://www.techotopia.com/index.php/Android\_Broadcast\_Intents\_and\_Broadcast\_Receivers

直接运行F:\CTF\SDK\platform-tools或者C:\下已经环境变量Path好的adb

adb devices就会显示已经attached的设备或者模拟器

如果蓝叠开启之后一直显示不出来就

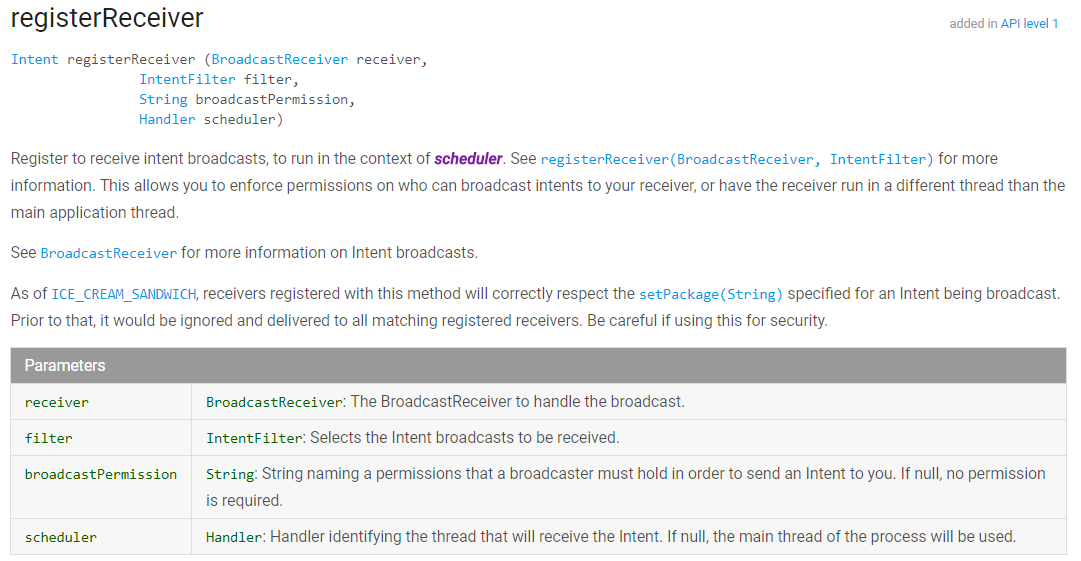
adb kill-server

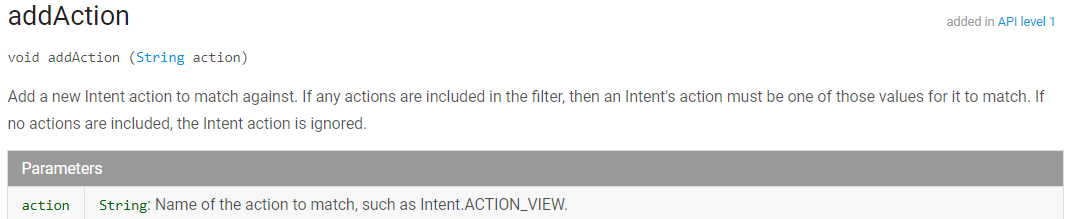
adb shell（当只attached有一个设备时直接连接）

或者指定设备

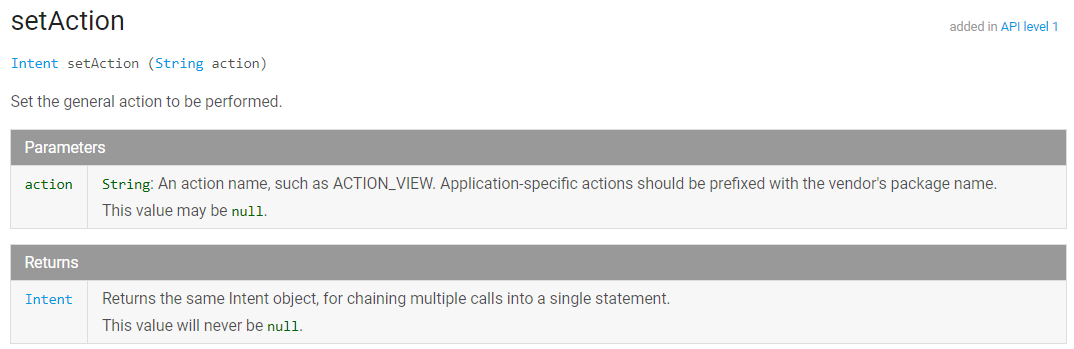
adb –s emulator-5554 shell

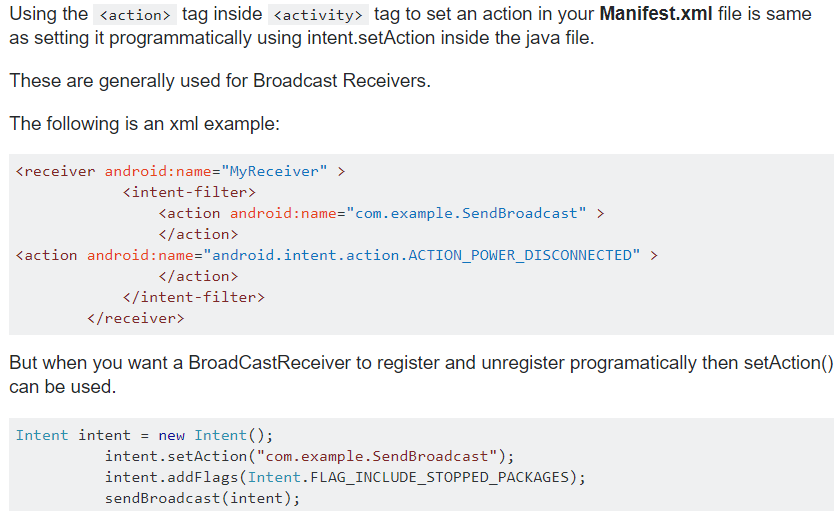
IntentFilter:





Intent:





通过命令行执行adb shell am broadcast发送广播通知。

adb shell am broadcast 后面的参数有：

[-a <ACTION>]

[-d <DATA\_URI>]

[-t <MIME\_TYPE>]

[-c <CATEGORY> [-c <CATEGORY>] ...]

[-e|--es <EXTRA\_KEY> <EXTRA\_STRING\_VALUE> ...]

[--ez <EXTRA\_KEY> <EXTRA\_BOOLEAN\_VALUE> ...]

[-e|--ei <EXTRA\_KEY> <EXTRA\_INT\_VALUE> ...]

[-n <COMPONENT>]

[-f <FLAGS>] [<URI>]

例如：

adb shell am broadcast -a com.android.test --es test\_string "this is test string" --ei test\_int 100 --ez test\_boolean true

说明：蓝色为key，红色为alue，分别为String类型，int类型，boolean类型

为什么不能在第三方的emulator上运行android\_server。因为android\_server是基于ARM的，而类似Bluestack的第三方emulator是采用了专有的LayerCake(夹心蛋糕)技术，将ARM转换成X86，所以速度比较快

You cannot debug native code with those emulators as they aren't arm based. The emulator from google will however work.

For example BlueStacks uses a proprietary technology called LayerCake which allows apps to run at near native speeds. This is possible since LayerCake "converts" ARM to x86 and this is the reason for its speed.

建议在cmd里做，不要在powershell里面做

安装apk——adb –s serial install 拖动apk形成路径，安装完成之后apk会保留在

pkg: /data/local/tmp/flagstore.apk里面

IDA调:

3个cmd:

1. 运行debug服务器

adb shell

su

cd data/local/tmp

./android\_server

1. 启动所要调试的Activity，app会弹出”Waitting for debugger”

adb shell

su

am start -D -n com.flagstore.ctf.flagstore/com.flagstore.ctf.flagstore.MainActivity

1. 设置端口转发，为了解决

“Could not connect to 127.0.0.1: 由于目标计算机积极拒绝，无法连接”

adb forward tcp:23946 tcp:23946

am start -D -n com.flagstore.ctf.flagstore/com.flagstore.ctf.flagstore.MainActivity

**IDA Pro调试出现"Could not connect to 127.0.0.1: 由于目标计算机积极拒绝，无法连接。"的解决方案**：可以尝试在执行adb forward tcp:23946 tcp:23946这行命令时，新建一个cmd窗口，输入这行口令，然后进行连接设备。

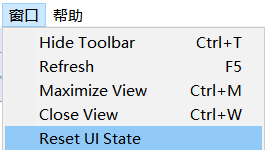
JEB2调：

am broadcast -a com.flagstore.ctf.INCOMING\_INTENT --es msg "OpenSesame"

常见问题：

1. Q:如果在attach的时候，可以显示设备但是无法显示process列表

A:Reset UI State然后重启jeb2即可



1. Q: ADB server didn’t ACK,ADB daemon启动失败

A：一般是由于5037这个端口被占用导致的

cmd>adb nodaemon server查看绑定出错信息

netstat –ano|findstr “5037”，最右一列为pid号

可以通过tasklist /fi “pid eq pid\_num” 来锁定进程名称来kill，或者直接根据上面的pid号在任务管理器里面杀

PS：我遇到这个问题是因为之前在刷机，“刷机工具”连接手机自动启动自带的adb导致端口冲突，进程里一直有一个“xadb”在循环启动，把“刷机软件”关了就好了

3. Q:如果遇到adb devices显示设备offline/如果设备之前adb devices查看是online，但是在开始start-attach的时候设备列表里没有显示任何设备或者显示offline

A:是由于电脑里面有多个adb导致冲突，新的adb会kill前面的adb从而导致设备offline。解决方案是：

如果我们在使用真机调试，那么使用的adb默认指定为系统环境变量Path里面的adb目录（比如C:\hkwany\adb.exe，那么环境变量里面添加C:\hkwany即可）；

如果我们使用的是模拟器emulator调试（我们这次用的是“逍遥模拟器”），那么在“逍遥模拟器”的文件目录D:\xiaoyao\MEmu下面其实都自带了adb.exe，并且模拟器设定的adb路径为这个，因此就会与系统原本的Path里原本的adb形成冲突，因此我们要将path里面的路径改成模拟器adb路径D:\xiaoyao\MEmu

PS：参考第2问题的回答，也可能是由于之前开启了刷机软件自带adb导致冲突，关闭“刷机软件”即可

4: Q:Attach的时候显示Could not attach to the target.

A:原因是你开了DDMS(Dalvik Debug Monitor Service)：

如果你在Eclipse里面打开的Android工程，就是因为开启了eclipse/android studio/

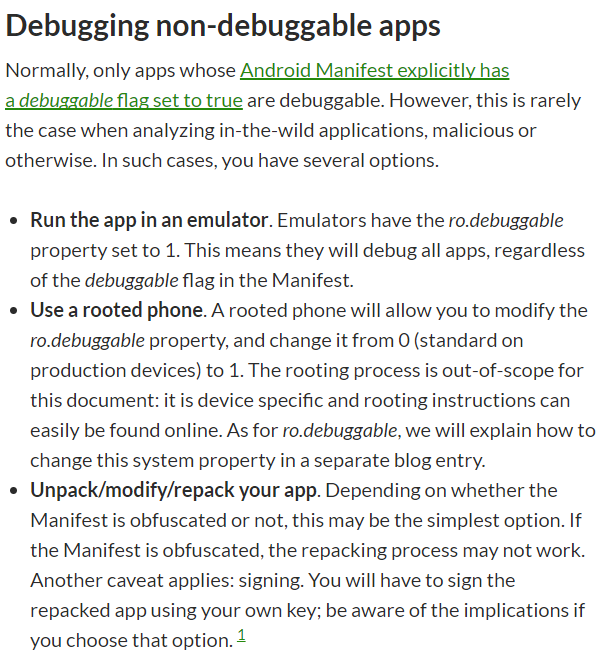
ddms导致jeb不能附加指定的应用,关闭DDMS应用即可

如果你在Android Studio软件打开的Android工程，打开Tools/Android/Android Device Monitor，在弹出的对话框里面就可以看到DDMS了

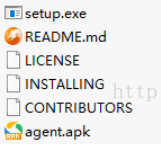
所以，关闭Eclipse和Android Studio两个软件即可

5： Q:其他问题（Debugging non-debuggable apps）

A:调试问题还要先要安装插件ideasmali，然后要在Manifest中加入android:debuggable="true",重新打包安装，才能调试（见下图）



drozer监听: https://russtone.io/2017/02/14/bssidessf-2017-flagreceiver/



Android端：

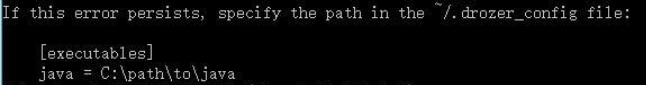
adb install agent.apk安装agent.apk，右下角打开，可以看见绑定了31415这个端口

电脑端：

安装setup.exe，生成drozer目录，命令行到该目录下运行命令drozer

一定要在该目录下(F:\CTF\drozer\drozer)执行命令，否则，即便你设置了环境变量，许多drozer内置的功能所必要的文件由于可能在子目录下而导致list之类的drozer功能命令无法使用！！！

Drozer提示找不到java路径:Could not find java. Please ensure that it is installed and on your path，会紧接着提示你要在你的个人目录下生成.drozer\_config文件



也就是在我的这台电脑上C:\Users\hkwany下

win7之后好像都不可以直接命名，会显示“一定要有文件名”

可以先创建1.txt，然后在命令行里面rename 1.txt .drozer\_config

.drozer\_config的内容为

[executables]

java=C:\Program Files\Java\jdk1.8.0\_121\bin\java.exe

java=C:\Program Files\Java\jdk1.8.0\_121\bin\javac.exe

指定一下jdk

进入dz>环境

启动另一个cmd

设置端口转发:

adb forward tcp:31415 tcp:31415 #adb forward tcp:local\_port tcp:remote\_port

adb -s 79841b2d forward tcp:31415 tcp:31415（指定serial number）

之后drozer界面会显示selecting + 设备

下面开始连接agent

本机环境：

dz>drozer console connect

真实设备（同一局域网）环境：指定IP

drozer console connect –server 192.168.1.104(手机IP)

dz>list 可以查看所有模块

我们要使用app.broadcast.sniff这个广播嗅探模块

run app.broadcast.sniff --action "com.flagstore.ctf.OUTGOING\_INTENT"

指定某个action\_name的Intent，开启广播监听

然后重新开启一个cmd，对其进行发送广播：

am broadcast -a com.flagstore.ctf.INCOMING\_INTENT --es msg "OpenSesame"

点button就可以看到flag

常见问题：