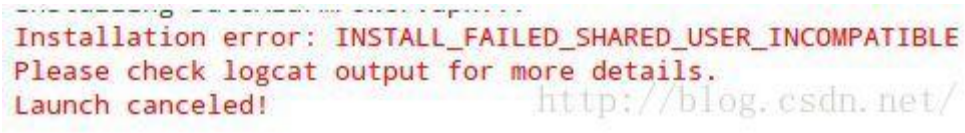
给第三方 apk 进行系统签名的几种方式

注：本文假设你已经拥有 Android 系统源码，且对 Android 源码有一定认识。

工作中有时会遇到一些 apk 签名不同，导致无法安装的问题。

场景一：

有一个第三方 apk（具有系统权限），无法安装在我们自己的 Android 机器上，提示以下错误，导致无法安装。



这是由于该 APK 具有系统权限，而系统签名与我们的 Android 设备系统签名不一致。Android 检测到系统签名不一致，由于安全因素考虑，就阻止安装了。

解决方法：使用自己的 Android 签名工具给 apk 重新签名。

(1) Android 的签名文件存放于系统源码的 build/target/product/security/目录下

该目录下有 media.pk8、media.x509.pem、platform.pk8、platform.x509.pem、shared.pk8、shared.x509.pem、testkey.pk8、testkey.x509.pem 等签名文件，不同的签名文件，对应不同的权限。Android 默认的签名文件为 testkey.pk8、testkey.x509.pem。

(2) Android 自带的签名工具为 signapk.jar， 可以在源码编译目录 out 中找到，具体路径为：out/host/linux-x86/framework/signapk.jar 以上 APK 具有系统权限，重新签名应该使用 platform 签名文件进行签名。

签名方法 :将对应权限的签名文件 platform.pk8、platform.x509.pem， 签名工具 signapk.jar，以及需要签名的 apk(假设 old.apk) 放到同一目录下 ,打开 linux 终端(windows cmd 也可以)，进入该目录，进行重新签名：

java -jar signapk.jar platform.x509.pem platform.pk8 old.apk new.apk

重新生成的 new.apk 就可以安装在我们的 Android 设备上了。

场景二：具有 apk 源码，同样是具备系统权限的，当我们将 apk 源码导入 Eclipse 中，使用 Run as --> Android application 编译安装 APK 时，Eclise 同样会提示场景一的错误信息，原因也是一样。我们同样可以将 Eclipse 生成的 apk 按照场景一的方法进行重新签名 ,再安装到我们的设备上。但是，有时可能我们会经常修改 apk 源码进行调试验证，如果每次都把 apk 拿出来进行重新签名，再安装，这样确实麻烦了一些。**Eclipse 是支持使用自己的系统签名工具进行 APK 打包签名的。**使用这种方法，可以快速而方便地对 APK 进行系统签名，并将其安装到我们的 Android 设备上。下面说一下具体做法：

步骤一：同样取源码目录 build\target\product\security 目录下的 platform.pk8 platform.x509.pem 放到某一个目录下

步骤二：进入该目录，生成 shared.priv.pem

openssl pkcs8 -in platform.pk8 -inform DER -outform PEM -out shared.priv.pem -nocrypt

步骤三：生成 pkcs12

openssl pkcs12 -export -in platform.x509.pem -inkey shared.priv.pem -out shared.pk12 -name androiddebugkey

Enter Export Password:

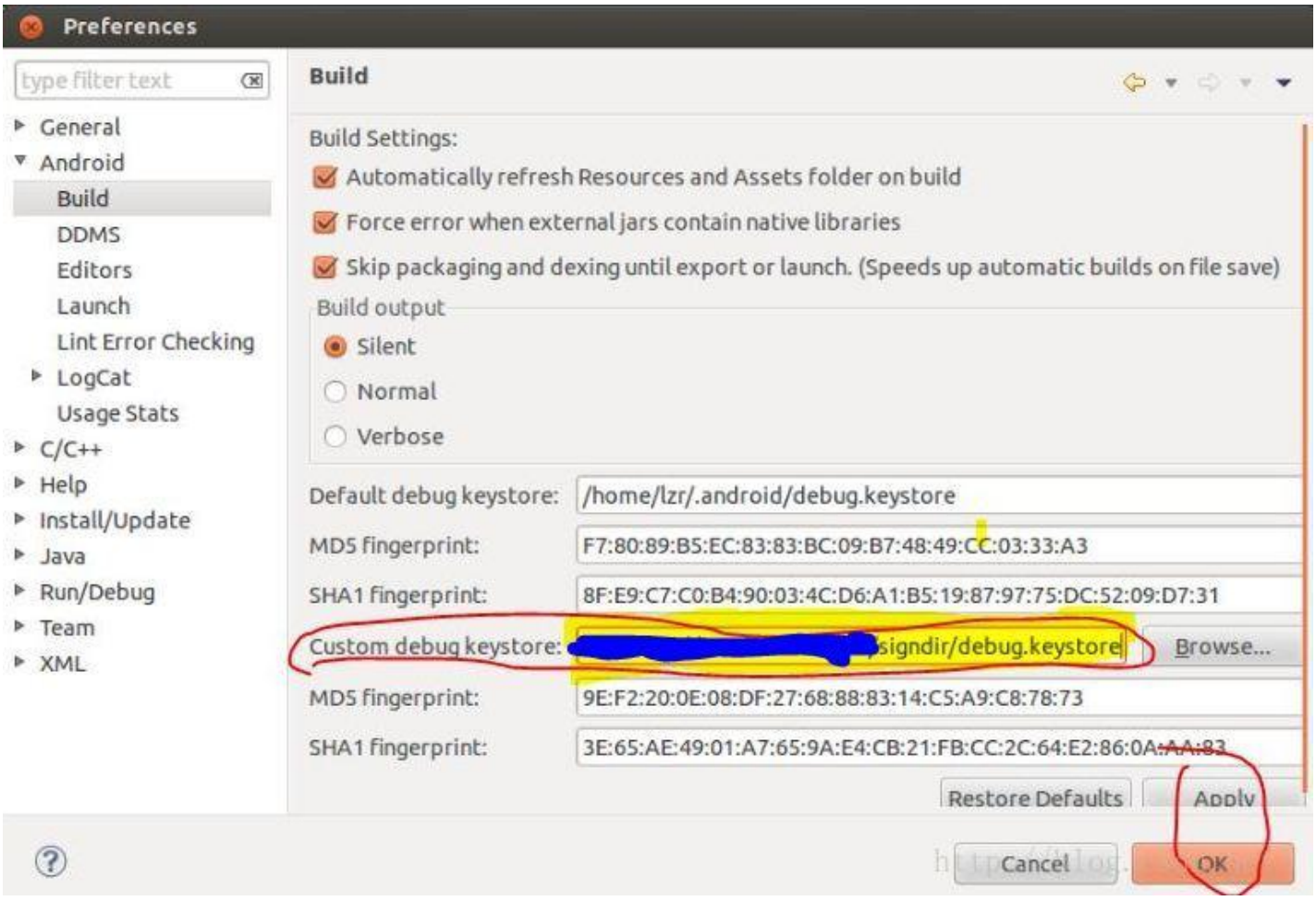
Verifying - Enter Export Password:

这里会提示输入密码，默认密码是 android，如是自己制作的 key，输入对应的密码。

步骤四：生成 debug.keystore，Eclipse 需要使用该 keystore.

keytool -importkeystore -deststorepass android -destkeypass android -destkeystore debug.keystore -srckeystore shared.pk12 -srcstoretype PKCS12 -srcstorepass android -alias androiddebugkey

步骤五：在 Eclipse 的 **Windows/Preferences/Android/Build** 中设置"Custom debug keystore"为刚才步骤四生成的 debug.keystore 即可直接 run 安装调试 apk.这样的话，就可以不用再去用 signapk.jar，如 java -jar signapk.jar platform.x509.pem platform.pk8 *.apk **.apk 进行签名了。



注：场景二其实也可以直接将 APK 源码放在 Android 系统源码的环境下用 make 来编译，需要编写 Android.mk，加入 **LOCAL_CERTIFICATE := platform**，

可以直接使用 mm 编译 apk，编译出来的 APK 同样可以顺利安装在我们自己的 Android 设备上。

一个简单的 Android.mk（APK 源码只包含 java 文件，不含 JNI 代码）文件可以参考以下写法，其中 XXX 修改为你的 apk 名字。

```
LOCAL_PATH:= $(call my-dir)

# DateAlarmPower.apk
include $(CLEAR_VARS)
LOCAL_MODULE_TAGS := optional

LOCAL_SRC_FILES := $(call all-java-files-under, src)

LOCAL_PACKAGE_NAME := xxx
LOCAL_CERTIFICATE := platform

include $(BUILD_PACKAGE)
```

进入该源码目录，执行 mm 命令，即可在 **out/target/product/$PRODUCT_NAME/system/app** 目录下生成 xxx.apk