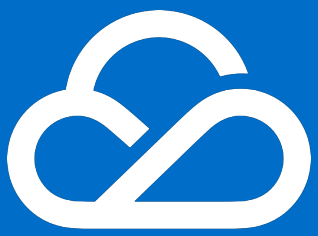
腾讯云移动安全

接入指引

产品文档



腾讯云

【版权声明】

【商标声明】

【服务声明】

本文档意在向客户介绍腾讯云全部或部分产品、服务的当时的整体概况，部分产品、服务的内容可能有所调整。您所购买的腾讯云产品、服务的种类、服务标准等应由您与腾讯云之间的商业合同约定，除非双方另有约定，否则，腾讯云对本文档内容不做任何明示或模式的承诺或保证。

文档目录

文档声明 2
游戏反外挂 C#接入指引 4
游戏反外挂 C 接入指引 12
游戏反外挂 Java接入指引 24

游戏反外挂 C#接入指引

1. 准备工作

接入安全SDK，开发者需要完成以下步骤：

1. 根据游戏运行平台和支持的CPU架构将SDK动态库拷贝到指定工程目录
2. 根据游戏id和用户登录信息调用SDK接口函数
3. 验证SDK接入是否正确

安全SDK在开发语言为C/C++的Android系统下接入需要的相关文件有以下:

tp2.cs
tp2.jar (Android)
libtersafe2.so (Android)

需要申请的权限:

```
<uses-permission android:name="android.permission.ACCESS_WIFI_STATE"/>
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
<uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE" />
<uses-permission android:name="android.permission.GET_TASKS" />
<uses-permission android:name="android.permission.INTERNET" />
```

SDK接口函数：

初始化接口 Tp2SdkInitEx
用户登录接口 Tp2UserLogin
前后台切换接口 Tp2SetGamestatus

2. 添加SDK文件到工程

2.1 添加文件

（1）将sdk\android\c#目录下的tp2.cs放在工程的Assets目录下

（2）将sdk\android\c#目录下的tp2.jar放在工程的Assets\Plugins\Android目录下

（3）多CPU：

以Unity5.0为例，如果游戏支持Android多cpu架构(目前只支持arm-v7a和x86)，将sdk\android\c#\lib目录下的armeabi和x86文件夹下的libtersafe2.so分别拷贝到以下目录:

Assets/Plugins/Android/libs/armeabi-v7a/

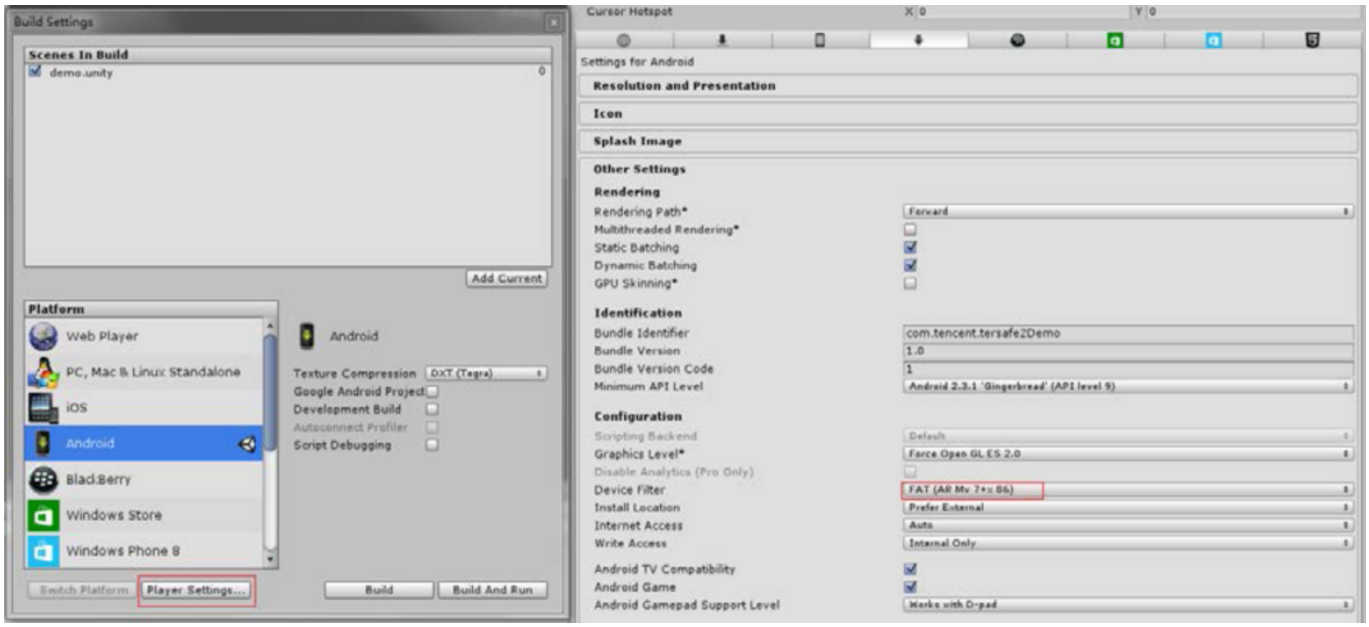
Assets/Plugins/Android/libs/x86/

（4）单CPU架构：

如果游戏只支持arm-v7，以Unity4.5为例，将SDK提供的tp2.jar和armeabi-v7a目录下的libtersafe2.so两个文件放在/Assets/Plugins/Android/目录下即可

2.2 工程属性设置

多cpu架构支持时选择[File] -> [Build settings] -> [Player Settings] -> [Other Settings] -> [Device Filter] -> [FAT(ARMv7+x86)]



3. SDK接口调用

3.1 初始化函数

函数原型

void Tp2SdkInitEx (int gameId, string appKey);

参数说明

参数	是否必须	说明
gameId	是	由腾讯云官网分配的game_id
appKey	是	由腾讯云官网分配game_key，与game_id对应

gameId和appKey在腾讯云官网（xxxxxxxxxxx）注册完新游戏后自动生成
返回值：0表示调用成功

3.2 设置用户信息

函数原型

void Tp2UserLogin (int accountType, int worldId, String openId, String roleId);

参数说明

参数	标题2
account_type	与运营平台相关的帐号类型，参考下文的TssSdkEntryId填写
world_id	用户游戏角色的大区信息
open_id	用户唯一身份标识，可自定义字符串。（和处罚相关必须填写）

参数	标题2
role_id	区分用户创建的不同角色的标识

account_type默认QQ平台填1，微信平台填2，其他平台填99。国内外主流帐号登录平台可参考以下数值填写
。

enum TssSdkEntryId
{
ENTRY_ID_QZONE = 1, // QQ
ENTRY_ID_MM = 2, // 微信
ENTRT_ID_FACEBOOK = 3, // facebook
ENTRY_ID_TWITTER = 4, // twitter
ENTRY_ID_LINE = 5, // line
ENTRY_ID_WHATSAPP = 6, // whatsapp
ENTRY_ID_OTHERS = 99, // 其他平台
};

world_id由游戏自定义，如果游戏没有分区可填0。
role_id用于区分同一帐号同一分区下的不同角色，如果没有角色区分可填“”。
open_id由所在运营平台分配，用于唯一区分用户。
返回值：0表示调用成功

3.3 设置游戏状态

函数原型

void Tp2SetGamestatus (Tp2Status status);

参数说明

参数	说明
status	前台 Tp2Status. FRONTEND

参数	说明
	后台 Tp2Status. BACKEND

枚举类型

```
public enum Tp2Status
{
FRONTEND = 1, // 前台
BACKEND = 2 // 后台
}
```

返回值：0表示调用成功

3.4 调用时机

（1）Tp2SdkInitEx在游戏启动的第一时间调用，参数为游戏id和appKey信息。更早时机调用安全接口函数可以更安全的保护游戏进程。
（2）Tp2UserLogin在游戏获取到用户授权的登录信息后调用，如果游戏有设置大区id和角色id，则在获取大区id和角色id之后再调用Tp2UserLogin函数。在游戏过程中，如果出现断线重连，用户注销重新登录等需要重新获取用户登录信息的情况，也需要再次调用该函数。传递的参数为用户的帐号信息，可自定义。
（3）Tp2SetGamestatus在游戏切换前后台调用。当游戏从后台切换到前台运行时调用Tp2SetGamestatus接口，设置参数为Tp2Status. FRONTEND。当游戏切从前台切换到后台时传递参数为Tp2Status. BACKEND。SDK部分功能在切换到后台时停止运行，因此该接口将影响SDK功能是否正常。

3.5 示例代码

```
void Awake ()
{
Tp2Sdk.Tp2SdkInitEx(8888, "d5ab8dc7ef67ca92e41d730982c5c602");
}
// 用户登录后调用
void Start ()
{
```

```
int accountType = (int)Tp2Entry.ENTRY_ID_QZONE ; /*帐号类型*/
int worldId = 100; /*大区id*/
string openId = "B73B36366565F9E02C752"; /*用户id*/
string roleId = "paladin"; /*角色id*/
Tp2Sdk.Tp2UerLogin(accountType， worldId， openId， roleId);
}
// 游戏切换前后台调用
void OnApplicationPause (bool pause)
 {
if (pause)
 {
Tp2Sdk.Tp2SetGamestatus(Tp2Status. BACKEND); // 切换到后台
}
else
{
Tp2Sdk.Tp2SetGamestatus(Tp2Status. FRONTEND); // 切换回前台
}
}
```

4. 验证SDK接入是否正确

1.将安卓手机通过usb数据线连接windows电脑。连接成功后，使用windows的命令行工具，登录到android adb控制台，如图：

```
C:\Users\Administrator>adb shell
shell@hwp7:/ $
```

2.敲入cd /sdcard回车，再敲入mkdir sdk回车，用于创建/sdcard/sdk目录。其中，如果目录已经存在，则系统会提示mkdir failed for /sdcard/sdk，File exists，继续下一步：

```
shell@hwp7:/ $ cd /sdcard
cd /sdcard
shell@hwp7:/sdcard $ mkdir sdk
mkdir sdk
shell@hwp7:/sdcard $
```

3.cd /sdcard/sdk进入目录，echo>enable.log创建enable.log空文件 ：

```
shell@hwp7:/sdcard/sdk $ echo >enable.log
echo >enable.log
```

有的机型可能无法访问shell创建的目录下的文件，这种情况请切换为root用户更改/sdcard/sdk目录权限或更换手机

```
shell@hwp7:/sdcard $ su
su
root@hwp7:/mnt/shell/emulated/0 # chmod -R 777 /sdcard/sdk
chmod -R 777 /sdcard/sdk
root@hwp7:/mnt/shell/emulated/0 #
```

4.运行游戏并登录用户，查看/data/data/log目录会生成日志文件tp2.log和tlog.log，如图：

```
shell@hwp7:/sdcard/sdk $ ls -l
ls -l
-rwxrwx--- root     sdcard_r        1 2016-04-20 22:37 enable.log
-rwxrwx--- root     sdcard_r     3324 2016-04-20 22:33 tlog.log
-rwxrwx--- root     sdcard_r     4151 2016-04-20 22:33 tp2.log
```

如果没有生成日志，请检查/sdcard/sdk和enable.log是否有读写权限。少部分机型无法读写这个目录，可更换机型测试或将/sdcard/sdk改为/data/data/log(需要root)。

注：enable.log只用于测试使用。

5.打开tp2.log文件，检查日志中是否包含三

个接口（native）信息

tp2_sdk_init_ex，tp2_setuserinfo，setgamestatus以及jar包版本号jar_ver

。以上条件必须都满足才能正确运行安全SDK。setgamestatus:1表示当前进程运行在前台，setgamestatus:2表示当前进程运行在后台。

请测试app切换前后台，查看接口调用是否正确。除了接口调用，还要检查用户信息(userinfo)是否填写正确。

```
root@hwG750-T01:/sdcard/sdk # cat tp2.log
[17:41:04] tp2_sdk_init_ex, ver:1.6.0
[17:41:04] {
[17:41:04]   app_id:8888
[17:41:04]   app_key:d5ab8dc7ef67ca92e41d730982c5c602
[17:41:04] }
[17:41:04] tp2_setuserinfo
[17:41:04] {
[17:41:04]   account_type:1
[17:41:04]   world_id:101
[17:41:04]   open_id:CF086A77B355AD8CFFEA6B94337EDFE4
[17:41:04]   role_id:Paladin
[17:41:04] }
[17:41:05] setgamestatus:1
[17:41:05] jar_ver:1.6.0
[17:41:05] jar_ns:!!com.tencent.tersafe2.util.JNCTask
[17:41:05] VER:1.6.0(Android), 20161230
[17:41:05] {
[17:41:05] }
[17:41:15] setgamestatus:2
[17:41:20] setgamestatus:1
```

6.打开tlog.log可查看安全SDK发送的数据，如图

```
shell@hwp7:/sdcard/sdk $ cat tlog.log
cat tlog.log
010000014F10000000005B2E1E0D00002328000000000000000000000000000000
3635653664626136623200000901000000010000000001010001 0A00230000000
4645413642393433333745444645340000C500000001 00C5010A00090000000000
877C9C13CF18874A86921AFF049C7F0484E78957B75C6CDC8F52143DB3841C986
D9861D94B644DD1DBAFBF2DB1FBD0AA7C0B5E7954EEEDB3AF461078F28AB87325
44AD733953D57651AB8F8C68CE1BBEAC78D5CCF0CBAB67E980AA0AA173EDA3059
01000001840 10001C1045B2E1E0D00002328E9991757C69101001BC8958111FDA
3635653664626136623200001C1050001C08AA2F9010009C700000100000128 0
18323F2928CA8D86F74CF560972AF08C7A62409D80074C9F527CD9ACEA6319B47
288DFE10E7C8C9BFF676A9C8D249009398AD1493658515DE4A7E77E83834CCD72
759C2BC61CBF52339B9430D4287D6924F39727AA83C49280225D9EFB18CFA4ACF
F128901E90B0D67AC7F6BC41592E52AE6DE58BA3F28FCFA9A09A87B1D521C94FF
095125F34DEB24BFD02849B9B19EBBA3F7B67CA8DF4534814B13DB89
010000014F10000000005B2E1E0D00002328000000000000000000000000000000
3635653664626136623200000901000000010000000001010001 0A00230000000
4645413642393433333745444645340000C500000001 00C5010A00090000000000
7A7115C5709B182B6FF243C07E8B6E053EC92AB29619322FF76125CAAFBFC7827
EA90EDA330ECB75CD1ADB2FAF0103C50DAE4A43F0DA7C57724F07907FF25CC708
C33249AC56E17BA2DE213A05B74E2D6B9DF0C2EF9975D4ED13583E8A3889FC990
```

安全SDK除了在初始化时会上报一些进程基本信息，还会根据定期的安全扫描结果来发送数据，例如扫描到app证书签名不正确，内存被修改，外挂进程正在运行等信息。tlog.log记录SDK发送的数据（只在测试时生成），一般1小时的数据是20K左右，可以查看tlog.log的大小来统计安全SDK发送的数据量。

游戏反外挂 C 接入指引

1. 准备工作

接入安全SDK，开发者需要完成以下步骤：

（1）根据游戏运行平台和支持的CPU架构将SDK动态库拷贝到指定工程目录
（2）根据用户登录信息调用SDK接口函数
（3）验证SDK接入是否正确

安全SDK在开发语言为C/C++的Android系统下接入需要的相关文件有以下:

tp2.jar
tp2_sdk.h
tss_sdt.h,tss_sdt_ex.h（安全数据类型选接，接入教程见《SDK安全数据类型接入教程C++）
libtersafe2.so

需要申请的权限：

```
<uses-permission android:name="android.permission.ACCESS_WIFI_STATE"/>
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
<uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE" />
<uses-permission android:name="android.permission.GET_TASKS" />
<uses-permission android:name="android.permission.INTERNET" />
```

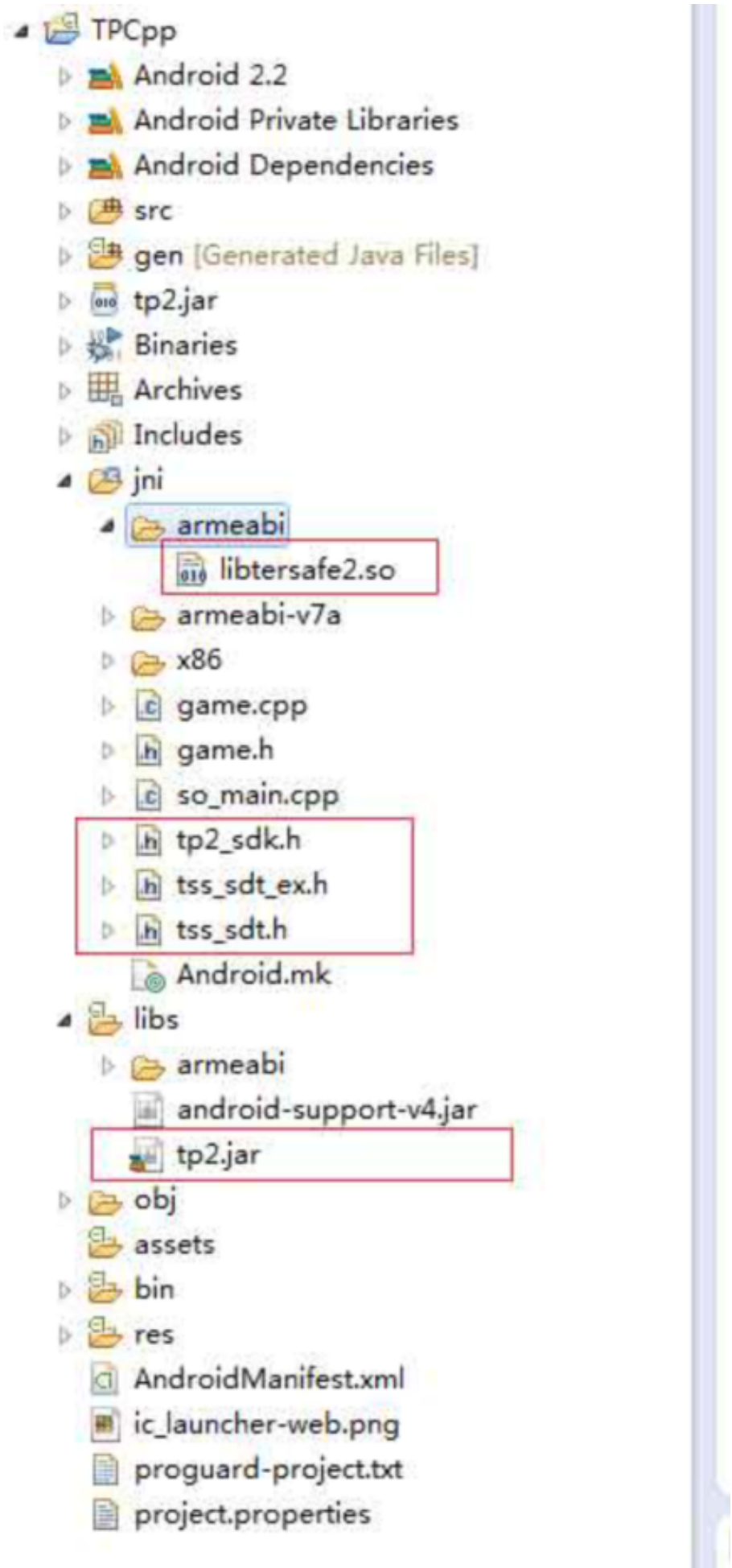
SDK接口函数：

初始化接口 tp2_sdk_init_ex
用户登录接口 tp2_setuserinfo
前后台切换接口 tp2_setgamestatus

2. 添加SDK文件到工程

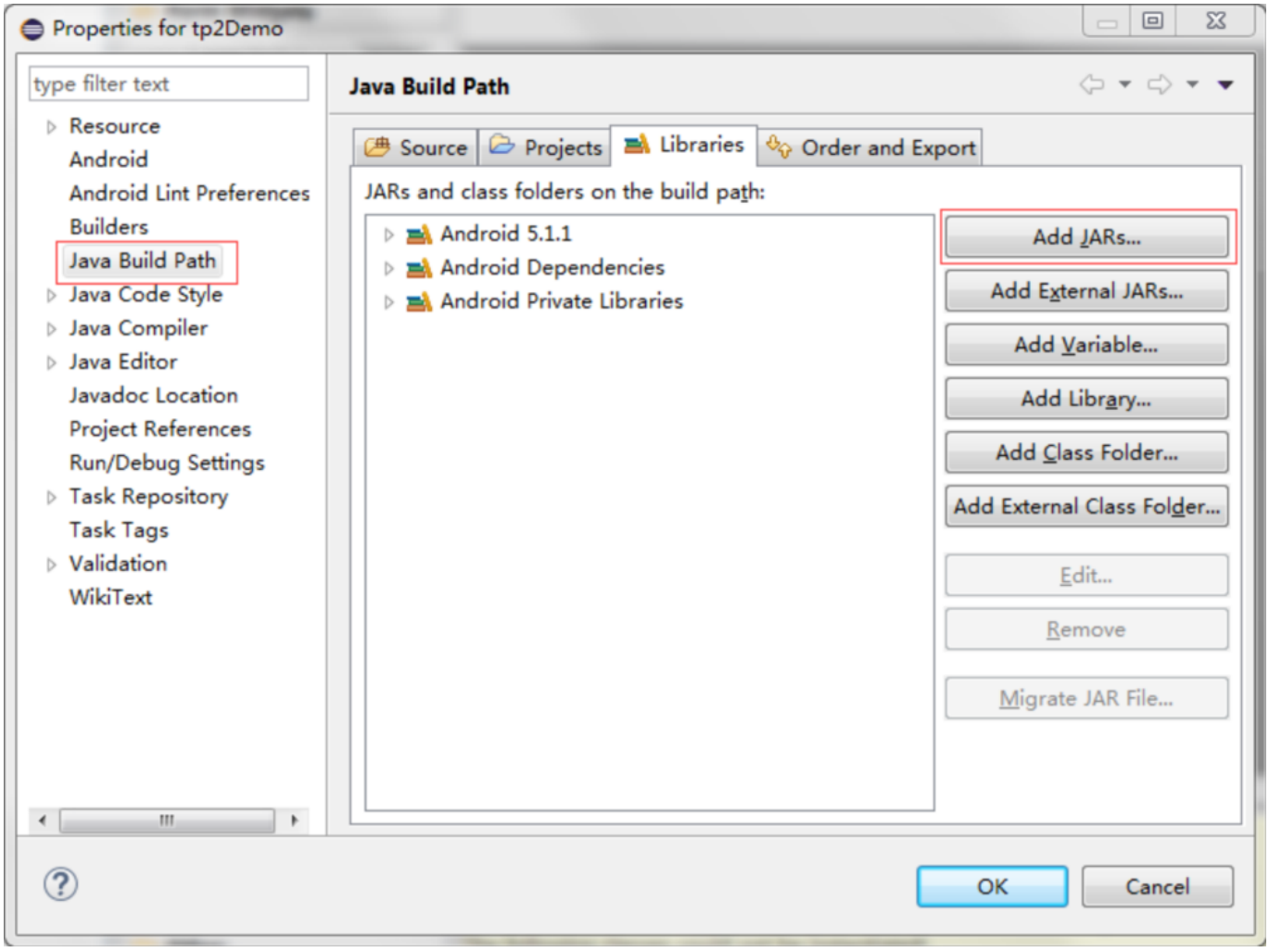
2.1 添加文件

（1）将sdk/android/c目录下的tp2.jar文件拷贝到android工程目录的libs目录下;
（2）将sdk/android/c目录下tp2_sdk.h文件拷贝到android工程目录的jni目录下;
（3）将sdt/c++目录下的tss_sdt.h和tss_sdt_ex.h文件拷贝到android工程目录的jni目录下（选接，接入教程见《C++安全数据类型接入教程.doc》）；
（4）将sdk/android/c/lib目录下以CPU架构命名的文件夹(包含libtersafe2.so文件)拷贝到android工程目录的jni/对应存so文件的目录，如jni/armeabi、jni/x86等, 对不支持的CPU架构体系不需要拷贝。

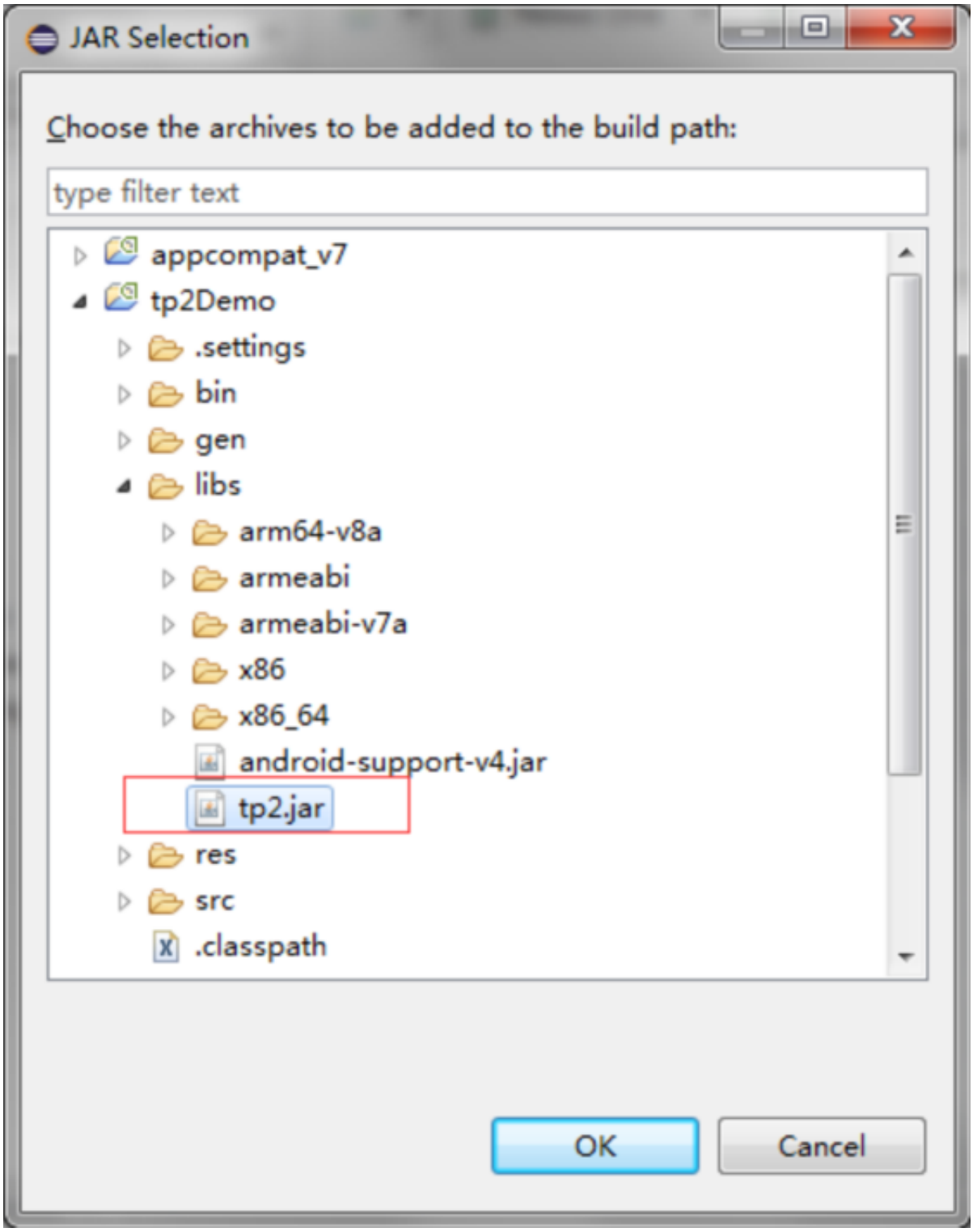

TPCpp
Android 2.2
Android Private Libraries
Android Dependencies
src
gen [Generated Java Files]
tp2.jar
Binaries
Archives
Includes
jni
armeabi
libtersafe2.so
armeabi-v7a
x86
game.cpp
game.h
so_main.cpp
tp2_sdk.h
tss_sdt_ex.h
tss_sdt.h
Android.mk
libs
armeabi
android-support-v4.jar
tp2.jar
obj
assets
bin
res
AndroidManifest.xml
ic_launcher-web.png
proguard-project.txt
project.properties

2.2 工程属性设置

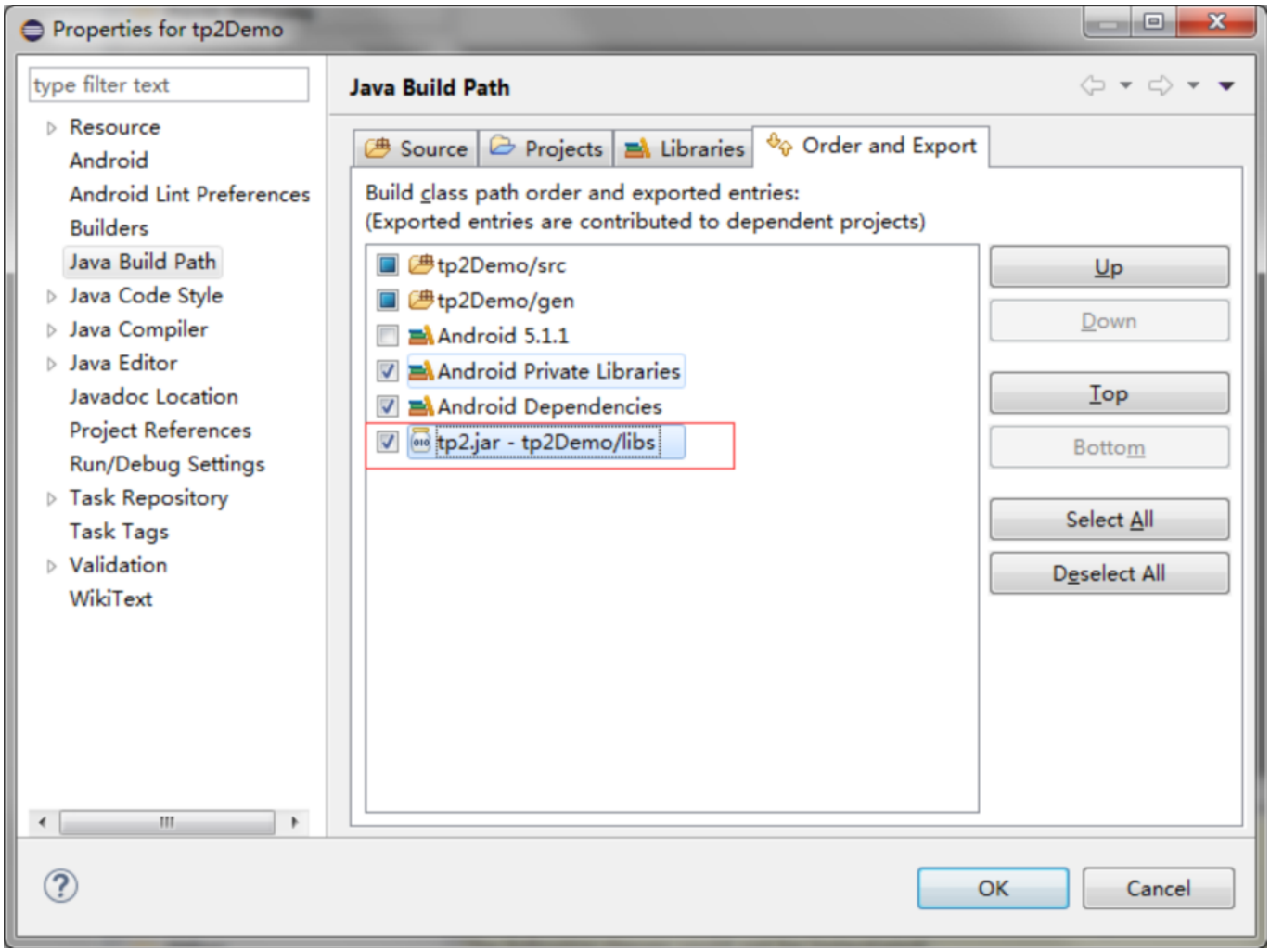
在Eclipse中左边的项目导航栏[Project Explorer]中选择游戏项目，点击鼠标右键，在弹出的菜单中选择[Properties]，选中Properties窗口左边导航栏中的[Java Build Path]选项，然后在[Library]中点击[add JARs]添加tp2.jar



选择已拷贝到工程目录的tp2.jar



添加tp2.jar后在[Order and Export]中选中tp2.jar



再接着，在android工程加载游戏so的地方，添加对libtersafe2.so的引用。

注意：libtersafe2.so的加载顺序，需要在游戏自己的so之前。

2.3 修改Android.mk

include $(CLEAR_VARS)
LOCAL_MODULE:=libtp2
LOCAL_SRC_FILES:=$(TARGET_ARCH_ABI)/libtersafe2.so
include $(PREBUILT_SHARED_LIBRARY)

在jin/android.mk中添加如下代码,用于加载libtersafe2.so

在jni/Android.mk中，游戏所在so节，添加如下代码，用于指示对libtp2的引用

LOCAL_SHARED_LIBRARIES:=libtp2

3. SDK接口调用

所需头文件

#include “tp2_sdk.h”

3.1 初始化接口

参数	是否必须	说明
game_id	是	由腾讯云官网分配的game_id
app_key	是	由腾讯云官网分配game_key，与game_id对应

gameId和appKey在腾讯云官网（xxxxxxxxxxxx）注册完新游戏后自动生成
返回值：0表示调用成功

3.2 用户登录接口

函数原型

int tp2_setuserinfo(int account_type, int world_id, string open_id, string role_id);

参数说明

参数	说明
account_type	与运营平台相关的帐号类型，参考下文的TssSdkEntryId填写

参数	说明
world_id	用户游戏角色的大区信息
open_id	用户唯一身份标识，可自定义字符串。（和处罚相关必须填写）
open_id	区分用户创建的不同角色的标识

account_type默认QQ平台填1，微信平台填2，其他平台填99。国内外主流帐号登录平台可参考以下数值填写。

```
enum TssSdkEntryId
{
ENTRY_ID_QZONE = 1, // QQ
ENTRY_ID_MM = 2, // 微信
ENTRT_ID_FACEBOOK = 3, // facebook
ENTRY_ID_TWITTER = 4, // twitter
ENTRY_ID_LINE = 5, // line
ENTRY_ID_WHATSAPP = 6, // whatsapp
ENTRY_ID_OTHERS = 99, // 其他平台
};
```

world_id由游戏自定义，如果游戏没有分区可填0。
role_id用于区分同一帐号同一分区下的不同角色，如果没有角色区分可填“”。
open_id由所在运营平台分配，用于唯一区分用户。
返回值：0表示调用成功

3.3 前后台切换接口

函数原型

```
int tp2_setgamestatus (int status);
```

参数	说明

参数	说明
status	前台 TP2_GAME_STATUS_FRONTEND 后台 TP2_GAME_STATUS_BACKEND

枚举类型

```
enum TP2GameStatus
{
TP2_GAME_STATUS_FRONTEND = 1, // 前台
TP2_GAME_STATUS_BACKEND = 2 // 后台
}
```

返回值：0表示调用成功

3.4 调用时机

（1）tp2_sdk_init_ex在游戏启动的第一时间调用，参数为游戏id和app_key信息。更早时机调用安全接口函数可以更安全的保护游戏进程。
（2）tp2_setuserinfo在游戏获取到用户授权的登录信息后调用，如果游戏有设置大区id和角色id，则在获取大区id和角色id之后再调用tp2_setuserinfo函数。在游戏过程中，如果出现断线重连，用户注销重新登录等需要重新获取用户登录信息的情况，也需要再次调用该函数。传递的参数为用户的帐号信息，可自定义。
（3）tp2_setgamestatus在游戏切换前后台调用。当游戏从后台切换到前台运行时调用Tp2SetGamestatus接口，设置参数为Tp2Status. FRONTEND。当游戏切从前台切换到后台时传递参数为Tp2Status. BACKEND。SDK部分功能在切换到后台时停止运行，因此该接口将影响SDK功能是否正常。

3.5 示例代码

```
void Start ()
{
// 游戏启动的第一时间调用
tp2_sdk_init_ex (8888, “a5ab8dc7ef67ca92e41d730982c5c602”);
// 用户登录时调用
int account_type = ENTRY_ID_QZONE; /*帐号类型*/
```

```
int world_id = 101; /*大区id*/
string open_id = "B73B36366565F9E02C752"; /*与平台相关的用户标识*/
string role_id = "paladin"; /*角色id*/
tp2_setuserinfo(account_type, world_id, open_id, role_id);
}

// 游戏切换到前台
void onResume ()
{
tp2_setgamestatus(TP2_GAME_STATUS_FRONTEND);
}

// 游戏切换到后台
void onPause ()
{
tp2_setgamestatus(TP2_GAME_STATUS_BACKEND);
}
```

4. 验证SDK接入是否正确

1 将安卓手机通过usb数据线连接windows电脑。连接成功后，使用windows的命令行工具，登录到android adb控制台，如图：

```
C:\Users\Administrator>adb shell
shell@hwp7:/ $
```

2 敲入cd /sdcard回车，再敲入mkdir sdk回车，用于创建/sdcard/sdk目录。其中，如果目录已经存在，则系统会提示mkdir failed for /sdcard/sdk，File exists，继续下一步。：

```
shell@hwp7:/ $ cd /sdcard
cd /sdcard
shell@hwp7:/sdcard $ mkdir sdk
mkdir sdk
shell@hwp7:/sdcard $
```

3 cd /sdcard/sdk进入目录，echo>enable.log创建enable.log空文件 ：

```
shell@hwp7:/sdcard/sdk $ echo >enable.log
echo >enable.log
```

有的机型可能无法访问shell创建的目录下的文件，这种情况请切换为root用户更改/sdcard/sdk目录权限或更换手机

```
shell@hwp7:/sdcard $ su
su
root@hwp7:/mnt/shell/emulated/0 # chmod -R 777 /sdcard/sdk
chmod -R 777 /sdcard/sdk
root@hwp7:/mnt/shell/emulated/0 #
```

4 运行游戏并登录用户，查看/data/data/log目录会生成日志文件tp2.log和tlog.log，如图：

```
shell@hwp7:/sdcard/sdk $ ls -l
ls -l
-rwxrwx--- root     sdcard_r        1 2016-04-20 22:37 enable.log
-rwxrwx--- root     sdcard_r     3324 2016-04-20 22:33 tlog.log
-rwxrwx--- root     sdcard_r     4151 2016-04-20 22:33 tp2.log
```

如果没有生成日志，请检查/sdcard/sdk和enable.log是否有读写权限。少部分机型无法读写这个目录，可更换机型测试或将/sdcard/sdk改为/data/data/log(需要root)。

注：enable.log只用于测试使用。

5

打开tp2.log文件，检查日志中是否包含三个

接口（native）信息

tp2_sdk_init_ex，tp2_setuserinfo，setgamestatus以及jar包版本号jar_ver

。以上条件必须都满足才能正确运行安全SDK。setgamestatus:1表示当前进程运行在前台，setgamestatus:2表示当前进程运行在后台。

请测试app切换前后台，查看接口调用是否正确。除了接口调用，还要检查用户信息(userinfo)是否填写正确。

```
root@hwG750-T01:/sdcard/sdk # cat tp2.log
[17:41:04] tp2_sdk_init_ex, ver:1.6.0
[17:41:04] {
[17:41:04]   app_id:8888
[17:41:04]   app_key:d5ab8dc7ef67ca92e41d730982c5c602
[17:41:04] }
[17:41:04] tp2_setuserinfo
[17:41:04] {
[17:41:04]   account_type:1
[17:41:04]   world_id:101
[17:41:04]   open_id:CF086A77B355AD8CFFEA6B94337EDFE4
[17:41:04]   role_id:Paladin
[17:41:04] }
[17:41:05] setgamestatus:1
[17:41:05] jar_ver:1.6.0
[17:41:05] jar_ns:!!com.tencent.tersafe2.util.JNCTask
[17:41:05] VER:1.6.0(Android), 20161230
[17:41:05] {
[17:41:05] }
[17:41:15] setgamestatus:2
[17:41:20] setgamestatus:1
```

6 打开tlog.log可查看安全SDK发送的数据，如图

```
shell@hwp7:/sdcard/sdk $ cat tlog.log
cat tlog.log
010000014F10000000005B2E1E0D00002328000000000000000000000000000000
3635653664626136623200000901000000010000000001010001 0A00230000000
4645413642393433333745444645340000C500000001 00C5010A00090000000000
877C9C13CF18874A86921AFF049C7F0484E78957B75C6CDC8F52143DB3841C986
D9861D94B644DD1DBAFBF2DB1FBD0AA7C0B5E7954EEEDB3AF461078F28AB87325
44AD733953D57651AB8F8C68CE1BBEAC78D5CCF0CBAB67E980AA0AA173EDA3059
01000001840 10001C1045B2E1E0D00002328E9991757C69101001BC8958111FDA
3635653664626136623200001C1050001C08AA2F9010009C700000100000128 0
18323F2928CA8D86F74CF560972AF08C7A62409D80074C9F527CD9ACEA6319B47
288DFE10E7C8C9BFF676A9C8D249009398AD1493658515DE4A7E77E83834CCD72
759C2BC61CBF52339B9430D4287D6924F39727AA83C49280225D9EFB18CFA4ACF
F128901E90B0D67AC7F6BC41592E52AE6DE58BA3F28FCFA9A09A87B1D521C94FF
095125F34DEB24BFD02849B9B19EBBA3F7B67CA8DF4534814B13DB89
010000014F10000000005B2E1E0D00002328000000000000000000000000000000
3635653664626136623200000901000000010000000001010001 0A00230000000
4645413642393433333745444645340000C500000001 00C5010A00090000000000
7A7115C5709B182B6FF243C07E8B6E053EC92AB29619322FF76125CAAFBFC7827
EA90EDA330ECB75CD1ADB2FAF0103C50DAE4A43F0DA7C57724F07907FF25CC708
C33249AC56E17BA2DE213A05B74E2D6B9DF0C2EF9975D4ED13583E8A3889FC990
```

安全SDK除了在初始化时会上报一些进程基本信息，还会根据定期的安全扫描结果来发送数据，例如扫描到app证书签名不正确，内存被修改，外挂进程正在运行等信息。tlog.log记录SDK发送的数据（只在测试时生成），一般1小时的数据是20K左右，可以查看tlog.log的大小来统计安全SDK发送的数据量。

游戏反外挂 Java接入指引

1. 准备工作

接入安全SDK，开发者需要完成以下步骤：

1. 根据游戏运行平台和支持的CPU架构将SDK动态库拷贝到指定工程目录
2. 根据游戏id和用户登录信息调用SDK接口函数
3. 验证SDK接入是否正确

安全SDK在开发语言为C/C++的Android系统下接入需要的相关文件有以下:

tp2.jar
libtersafe2.so

需要申请的权限

```
<uses-permission android:name="android.permission.ACCESS_WIFI_STATE"/>
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
<uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE" />
<uses-permission android:name="android.permission.GET_TASKS" />
<uses-permission android:name="android.permission.INTERNET" />
```

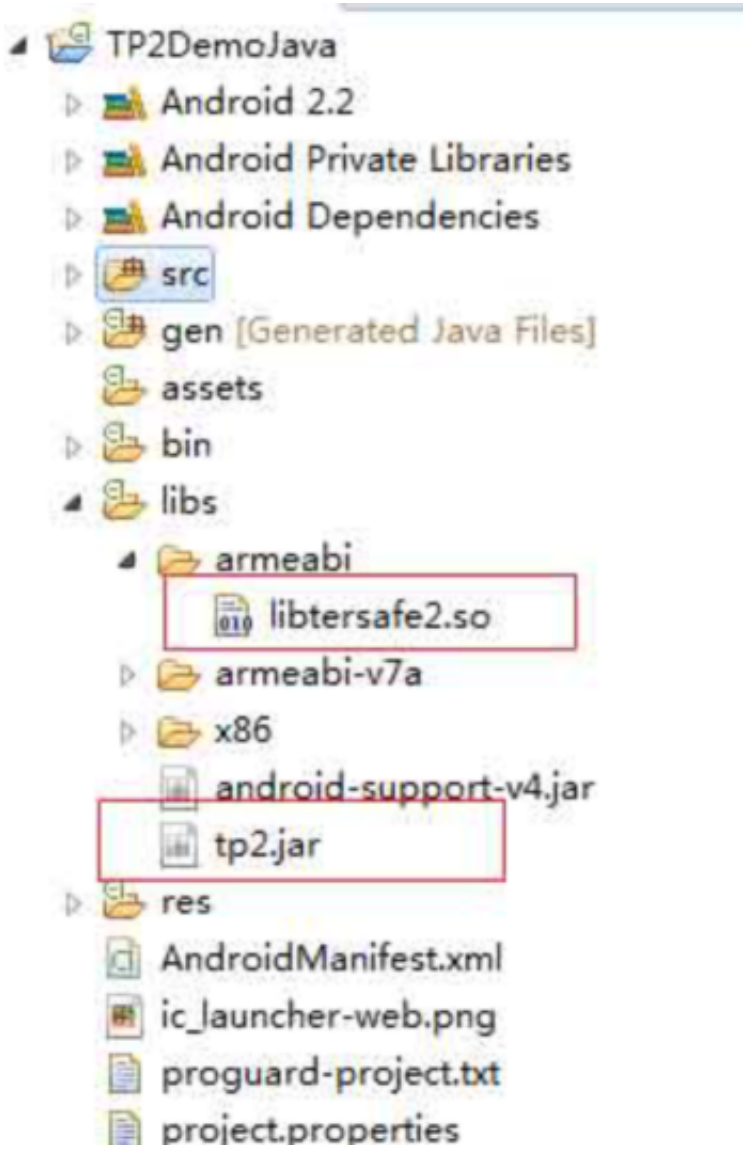
SDK接口函数：

初始化接口 initEx
用户登录接口 onUserLogin
前台切换到后台接口 onAppPause
后台切换到前台接口 onAppResume

2. 添加SDK文件到工程

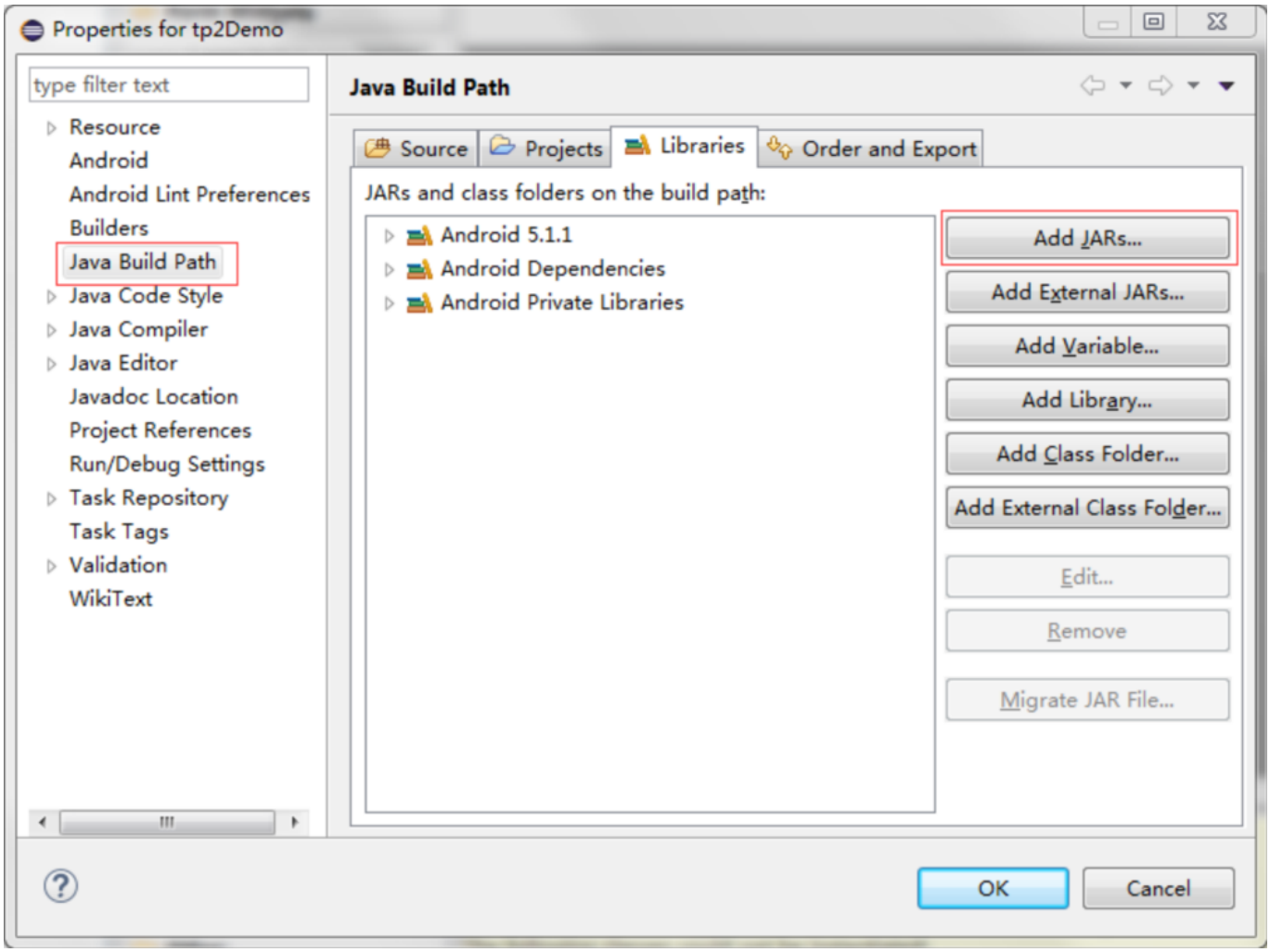
2.1 添加文件

（1）将sdk/android/c目录下的tp2.jar文件拷贝到android工程目录的libs目录下;
（2）将sdk/android/java/lib目录下以CPU架构命名的文件夹(包含libtersafe2.so文件)拷贝到android工程目录的libs目录下,对不支持的CPU架构体系不需要拷贝.

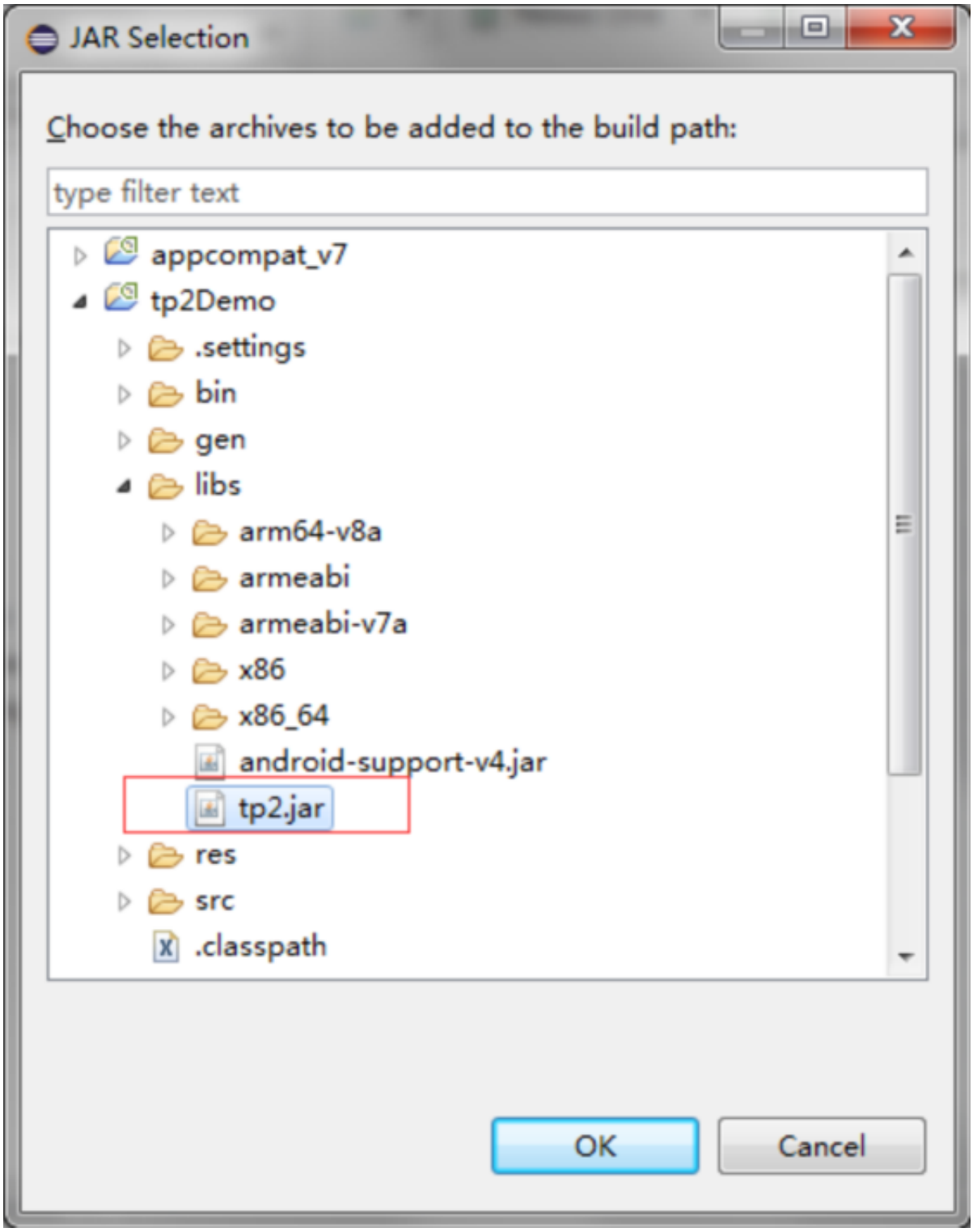


2.2 工程属性设置

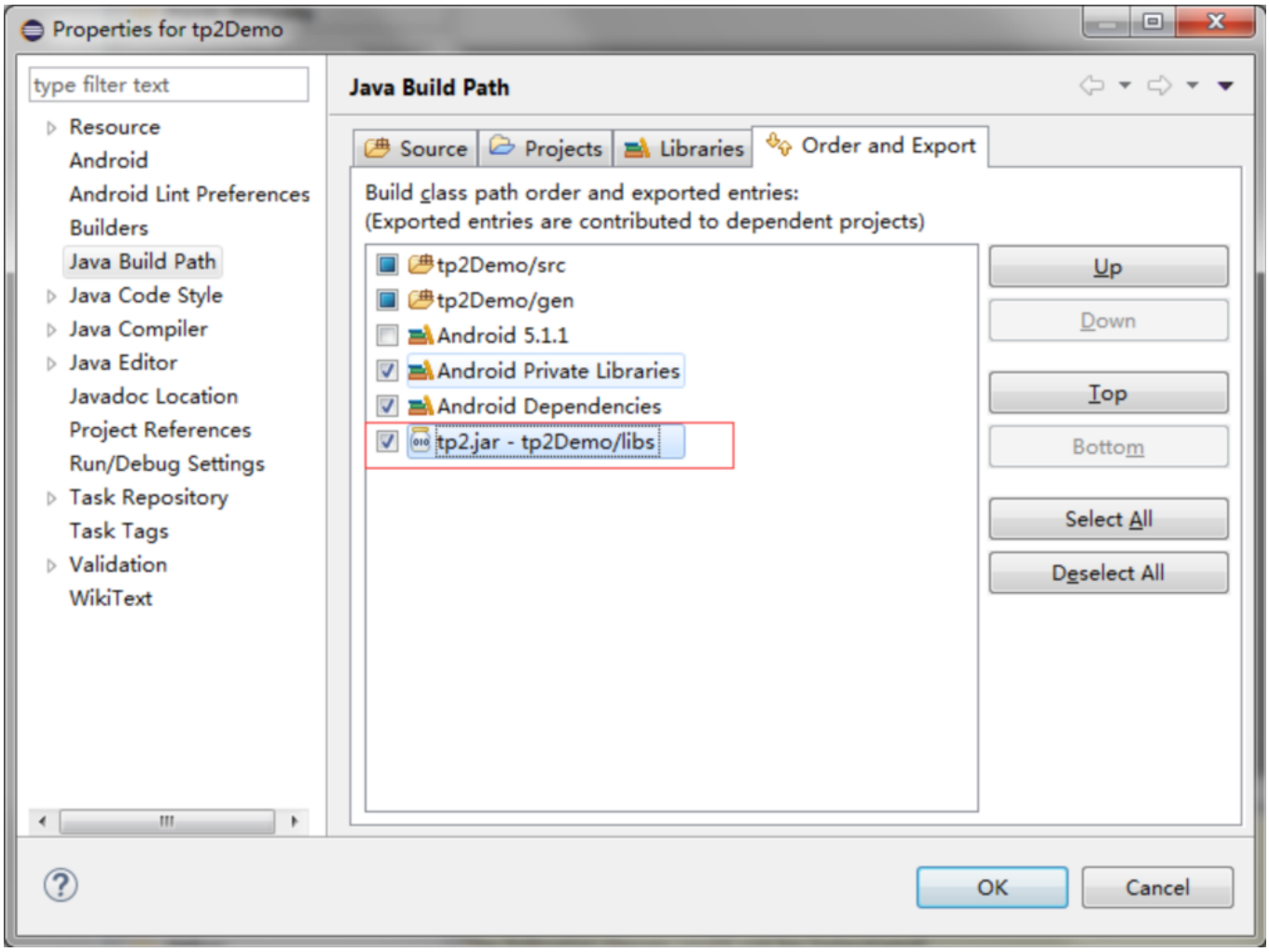
在Eclipse中左边的项目导航栏[Project Explorer]中选择游戏项目，点击鼠标右键，在弹出的菜单中选择[Properties]，选中Properties窗口左边导航栏中的[Java Build Path]选项，然后在[Library]中点击[add JARs]添加tp2.jar



选择已拷贝到工程目录的tp2.jar



添加tp2.jar后在[Order and Export]中选中tp2.jar



clean工程并重新编译

3. SDK接口调用

导入包

import com.tencent.tersafe2.TP2Sdk;

3.1 初始化函数

函数原型

public static int initEx(int gameId, String appKey);

参数说明

参数	是否必须	说明
gameId	是	由腾讯云官网分配的game_id
appKey	是	由腾讯云官网分配game_key，与game_id对应

gameId和appKey在腾讯云官网（xxxxxxxxxxxx）注册完新游戏后自动生成
返回值：0表示调用成功

3.2 用户登录接口

函数原型

public static native int onUserLogin(int accountType, int worldId, String openId, String roleId);

参数说明

参数	标题2
account_type	与运营平台相关的帐号类型，参考下文的TssSdkEntryId填写
worldId	用户游戏角色的大区信息
openId	用户唯一身份标识，可自定义字符串。（和处罚相关必须填写）
roleId	区分用户创建的不同角色的标识

account_type默认QQ平台填1，微信平台填2，其他平台填99。国内外主流帐号登录平台可参考以下数值填写
。

```
enum TssSdkEntryId
{
ENTRY_ID_QZONE = 1, // QQ
ENTRY_ID_MM = 2, // 微信
```

ENTRT_ID_FACEBOOK = 3, // facebook
ENTRY_ID_TWITTER = 4, // twitter
ENTRY_ID_LINE = 5, // line
ENTRY_ID_WHATSAPP = 6, // whatsapp
ENTRY_ID_OTHERS = 99, // 其他平台
};

world_id由游戏自定义，如果游戏没有分区可填0。
role_id用于区分同一帐号同一分区下的不同角色，如果没有角色区分可填“ ”。
open_id由所在运营平台分配，用于唯一区分用户。
返回值：0表示调用成功

3.3 前台切换到后台接口

函数原型

int onAppPause ();

程序由前台切换到后台，表明游戏当前为非活动状态。
返回值：0表示调用成功

3.4 后台切换到前台接口

函数原型
程序由后台切换到前台，表明游戏当前为活动状态。
返回值：0表示调用成功

3.5 调用时机

（1）TP2Sdk.initEx在游戏启动的第一时间调用，参数为游戏id和appKey信息。更早时机调用安全接口函数可以更安全的保护游戏进程。
（2）TP2Sdk.onUserLogin在游戏获取到用户授权的登录信息后调用，如果游戏有设置大区id和角色id，则在

获取大区id和角色id之后再调用TP2Sdk.onUserLogin函数。在游戏过程中，如果出现断线重连，用户注销重新登录等需要重新获取用户登录信息的情况，也需要再次调用该函数。传递的参数为用户的帐号信息，可自定义。

（3）TP2Sdk.onAppPause在游戏从前台切换到后台时调用。

（4）TP2Sdk.onAppResume在游戏从后台切换到前台时调用。

3.6 示例代码

```
public void onCreate()
 {
//游戏启动的第一时间调用
TP2Sdk. initEx(9000, “d5ab8dc7ef67ca92e41d730982c5c602”);
int accountType = ENTRYID.ENTRY_ID_QZONE; /*帐号类型*/
int worldId = 1; /*大区id*/
String openId = "B73B36366565F9E02C752"; /*与平台相关的用户标识*/
String roleId = "paladin"; /*角色id*/
//用户登录游戏时调用
TP2Sdk.onUserLogin(accountType, worldId, openId, roleId);
}
// 游戏切换到后台
public void onPause()
 {
super.onResume();
TP2Sdk.onPause();
}
// 游戏切换到前台
public void onResume()
{
super.onResume();
TP2Sdk.onResume();
}
```

4. 验证SDK接入是否正确

1.将安卓手机通过usb数据线连接windows电脑。连接成功后，使用windows的命令行工具，登录到android adb控制台，如图：

```
C:\Users\Administrator>adb shell
shell@hwp7:/ $
```

2.敲入cd /sdcard回车，再敲入mkdir
sdk回车，用于创建/sdcard/sdk目录。其中，如果目录已经存在，则系统会提示mkdir failed for /sdcard/sdk，File exists，继续下一步：

```
shell@hwp7:/ $ cd /sdcard
cd /sdcard
shell@hwp7:/sdcard $ mkdir sdk
mkdir sdk
shell@hwp7:/sdcard $
```

3.cd /sdcard/sdk进入目录，echo>enable.log创建enable.log空文件 ：

```
shell@hwp7:/sdcard/sdk $ echo >enable.log
echo >enable.log
```

有的机型可能无法访问shell创建的目录下的文件，这种情况请切换为root用户更改/sdcard/sdk目录权限或更换手机

```
shell@hwp7:/sdcard $ su
su
root@hwp7:/mnt/shell/emulated/0 # chmod -R 777 /sdcard/sdk
chmod -R 777 /sdcard/sdk
root@hwp7:/mnt/shell/emulated/0 #
```

4.运行游戏并登录用户，查看/data/data/log目录会生成日志文件tp2.log和tlog.log，如图：

```
shell@hwp7:/sdcard/sdk $ ls -l
ls -l
-rwxrwx--- root     sdcard_r        1 2016-04-20 22:37 enable.log
-rwxrwx--- root     sdcard_r     3324 2016-04-20 22:33 tlog.log
-rwxrwx--- root     sdcard_r     4151 2016-04-20 22:33 tp2.log
```

如果没有生成日志，请检查/sdcard/sdk和enable.log是否有读写权限。少部分机型无法读写这个目录，可更换机型测试或将/sdcard/sdk改为/data/data/log(需要root)。

注：enable.log只用于测试使用。

5.打开tp2.log文件，检查日志中是否包含三

个接口（native）信息

tp2_sdk_init_ex，tp2_setuserinfo，setgamestatus以及jar包版本号jar_ver

。以上条件必须都满足才能正确运行安全SDK。setgamestatus:1表示当前进程运行在前台，setgamestatus:2表示当前进程运行在后台。

请测试app切换前后台，查看接口调用是否正确。除了接口调用，还要检查用户信息(userinfo)是否填写正确。

```
root@hwG750-T01:/sdcard/sdk # cat tp2.log
[17:41:04] tp2_sdk_init_ex, ver:1.6.0
[17:41:04] {
[17:41:04]   app_id:8888
[17:41:04]   app_key:d5ab8dc7ef67ca92e41d730982c5c602
[17:41:04] }
[17:41:04] tp2_setuserinfo
[17:41:04] {
[17:41:04]   account_type:1
[17:41:04]   world_id:101
[17:41:04]   open_id:CF086A77B355AD8CFFEA6B94337EDFE4
[17:41:04]   role_id:Paladin
[17:41:04] }
[17:41:05] setgamestatus:1
[17:41:05] jar_ver:1.6.0
[17:41:05] jar_ns:!!com.tencent.tersafe2.util.JNCTask
[17:41:05] VER:1.6.0(Android), 20161230
[17:41:05] {
[17:41:05] }
[17:41:15] setgamestatus:2
[17:41:20] setgamestatus:1
```

6.打开tlog.log可查看安全SDK发送的数据，如图

```
shell@hwp7:/sdcard/sdk $ cat tlog.log
cat tlog.log
010000014F10000000005B2E1E0D00002328000000000000000000000000000000
3635653664626136623200000901000000010000000001010001 0A00230000000
4645413642393433333745444645340000C500000001 00C5010A00090000000000
877C9C13CF18874A86921AFF049C7F0484E78957B75C6CDC8F52143DB3841C986
D9861D94B644DD1DBAFBF2DB1FBD0AA7C0B5E7954EEEDB3AF461078F28AB87325
44AD733953D57651AB8F8C68CE1BBEAC78D5CCF0CBAB67E980AA0AA173EDA3059
01000001840100 01C1045B2E1E0D00002328E9991757C69101001BC8958111FDA
3635653664626136623200000 1C1050001C08AA2F9010009C700000100000128 0
18323F2928CA8D86F74CF560972AF08C7A62409D80074C9F527CD9ACEA6319B47
288DFE10E7C8C9BFF676A9C8D249009398AD1493658515DE4A7E77E83834CCD72
759C2BC61CBF52339B9430D4287D6924F39727AA83C49280225D9EFB18CFA4ACF
F128901E90B0D67AC7F6BC41592E52AE6DE58BA3F28FCFA9A09A87B1D521C94FF
095125F34DEB24BFD02849B9B19EBBA3F7B67CA8DF4534814B13DB89
010000014F10000000005B2E1E0D00002328000000000000000000000000000000
3635653664626136623200000901000000010000000001010001 0A00230000000
4645413642393433333745444645340000C500000001 00C5010A00090000000000
7A7115C5709B182B6FF243C07E8B6E053EC92AB29619322FF76125CAAFBFC7827
EA90EDA330ECB75CD1ADB2FAF0103C50DAE4A43F0DA7C57724F07907FF25CC708
C33249AC56E17BA2DE213A05B74E2D6B9DF0C2EF9975D4ED13583E8A3889FC990
```

安全SDK除了在初始化时会上报一些进程基本信息，还会根据定期的安全扫描结果来发送数据，例如扫描到app证书签名不正确，内存被修改，外挂进程正在运行等信息。tlog.log记录SDK发送的数据（只在测试时生成），一般1小时的数据是20K左右，可以查看tlog.log的大小来统计安全SDK发送的数据量。