**Android PackageManagerService**

**Android Permission Model**

date:2018/06/11

Context

ContextImpl

ContextWrapper

Service

ContextThemeWrapper

Activity

frameworks/base/core/java/android/app/ContextImpl.java

|  |
| --- |
| public PackageManager getPackageManager() { |
|  | if (mPackageManager != null) { |
|  | return mPackageManager; |
|  | } |
|  |  |
|  | IPackageManager pm = ActivityThread.getPackageManager(); |
|  | if (pm != null) { |
|  | // Doesn't matter if we make more than one instance. |
|  | return (mPackageManager = new ApplicationPackageManager(this, pm)); |
|  | } |
|  |  |
|  | return null; |
|  | } |

public abstract class PackageManager

{

}

class ContextImpl extends Context{

private PackageManager mPackageManager;

}

final class ApplicationPackageManager extends PackageManager{

private final IPackageManager mPM;

public List<PackageInfo> getInstalledPackages(int flags){

mPM.getInstalledPackages(flags,lastKey)

}

}

ActivityThread.getPackageManager();

public final class ActivityThread{

static volatile IPackageManager sPackageManager;

public static IPackageManager getPackageManager(){

if (sPackageManager != null) return sPackageManager;

IBinder b = ServiceManager.getService(“package”)

sPackageManager = IPackageManager.Stub.asInterface(b);

}

}

public interface IPackageManager extends android.os.IInterface{

public static abstract class Stub extends android.os.Binder implements IPackageManager{

public static IPackageManager asInterface(android.os.IBinder){

}

}

}

PackageManagerService.java

frameworks/base/services/java

package com.android.server.pm;

public class PackageManagerService extends IPackageManager.Stub{

}

<http://www.520monkey.com/archives/856>

# Android Permission Model

UID : Android在安装一个应用程序，会为它分配一个 uid，我们普通应用程序的uid是从 10000 开始分配 （见 Process.FIRST\_APPLICATION\_UID ）。在上面的定义代码块的最后一行有写到， 0-10000系统进程的 uid 。

GID : 对于普通应用程序来说， gid 等于 uid 。由于每个应用程序的 uid 和 gid 都不相同， 因此不管是 native 层还是 java 层都能够达到保护私有数据的作用 。

GIDS: gids 是由框架在 Application 安装过程中生成，与 Application 申请的具体权限相关。 如果 Application 申请的相应的 permission 被 granted ，而且 中有对应的 gid s ， 那么 这个 Application 的 gids 中将 包含这个 gids

1 mContext.getPackageManager().getPackageGids(app.info.packageName);

2

3

权限记录在什么地方？

安装权限

/data/system/packages.xml

/data/system/appops.xml

/data/system/users/0/runtime-permissions.xml

源码分析

权限定义

frameworks/base/core/res/AndroidManifest.xml

frameworks/base/core/res/AndroidManifest.xml

Android 6.0.1

frameworks/base/core/java/android/app/AppOpsManager.java

public class AppOpsManager{

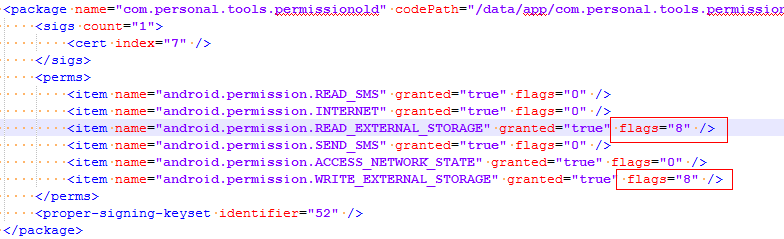
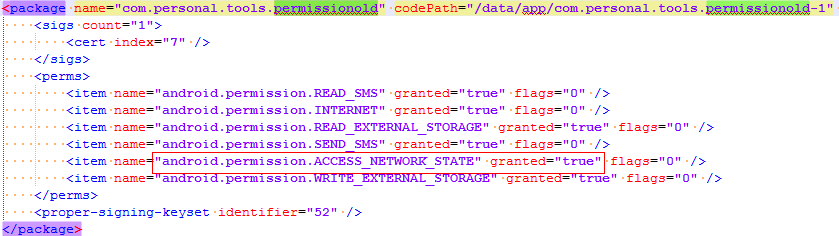
public static final int OP\_READ\_EXTERNAL\_STORAGE = 59;

public static final int OP\_WRITE\_EXTERNAL\_STOREAGE = 60;

}

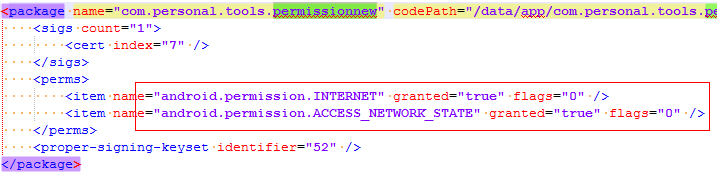
targetSdkVersion < 23

所有权限都放在packages.xml里

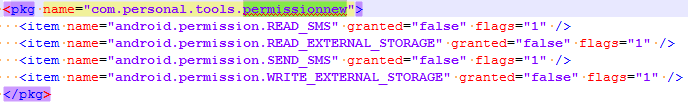


targetSdkVersion >= 23

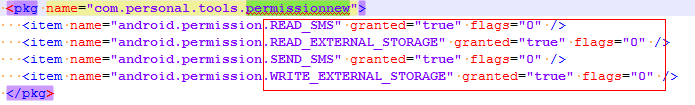
安装权限



运行时权限



运行时权限



## Application API

checkSelfPermission

shouldShowRequestPermissionRationale

requestPermissions

onRequestPermissionsResult

TargetSdkVersion >= 23

If your application is targeting an API level before 23 (Android M) then both:ContextCompat#checkSelfPermission and Context#checkSelfPermission doesn't work and always returns 0 (PERMISSION\_GRANTED). Even if you run the application on Android 6.0 (API 23).

It's not fully true that if you targeting an API level before 23 then you don't have to take care of permissions. If you targeting an API level before 23 then:

Android < 6.0: Everything will be ok.

Android 6.0: Application's run-time permissions will be granted by default (compatibility mode applies), but the user can change run-time permissions in Android Settings, then you may have a problem.

As I said in the 1st point, if you targeting an API level before 23 on Android 6.0 then ContextCompat#checkSelfPermission and Context#checkSelfPermission doesn't work. Fortunately you can use PermissionChecker#checkSelfPermission to check run-time permissions.

AppOps

/data/system/appops.xml

AppOpsManager

AppOpsSummary

AppOpsCategory

AppOpsService

mAppOpsService = new AppOpsService(new File(systemDir,”appops.xml”),mHandler)

Runtime Permission

app安装时权限默认是disable的

# PMS

PMS Source Code

<http://androidxref.com/6.0.1_r10/xref/frameworks/base/services/core/java/com/android/server/pm/PackageManagerService.java>

getPackageManager().getPackageGids(app.info.packageName)

PackageManager.java

<http://androidxref.com/6.0.1_r10/xref/frameworks/base/core/java/android/content/pm/PackageManager.java>

public abstract class PackageManager{

public static final int FLAG\_PERMISSION\_USER\_SET = 1 << 0;

public static final int FLAG\_PERMISSION\_USER\_FIXED = 1 << 1;

public static final int FLAG\_PERMISSION\_POLICY\_FIXED = 1 << 2;

public static final int FLAG\_PERMISSION\_REVOKE\_ON\_UPGRADE = 1 << 3;

public static final int FLAG\_PERMISSION\_SYSTEM\_FIXED = 1 << 4;

public static final int FLAG\_PERMISSION\_GRANTED\_BY\_DEFAULT = 1 << 5;

public static final int MASK\_PERMISSION\_FLAGS = 0xFF;

}

# PackageInstaller

<http://androidxref.com/6.0.1_r10/xref/packages/apps/PackageInstaller>

Permission.java

<http://androidxref.com/6.0.1_r10/xref/packages/apps/PackageInstaller/src/com/android/packageinstaller/permission/model/Permission.java>

public final class Permission{

private final String mName;

private final int mAppOp;

private Boolean mGranted;

private Boolean mAppOpAllowed;

private int mFlags

}

Public final class AppPermissionGroup implements Comparable<AppPermissionGroup>{

Private final PackageInfo mPackageInfo;

Private final String mName;

Private final ArrayMap<String,Permission> mPermissions = new ArrayMap<>();

}

// y

Public final class AppPermissions{

Private final ArrayList<AppPermissionGroup> mGroups = new ArrayList<>();

Private final LinkedHashMap<String,AppPermissionGroup> mNameToGroupMap =

New LinkedHashMap<>();

}

[android.Manifest.**permission**. WRITE\_EXTERNAL\_STORAGE;](http://androidxref.com/6.0.1_r10/xref/frameworks/base/services/core/java/com/android/server/policy/PhoneWindowManager.java#1900)

android.Manifest.permission. WRITE\_EXTERNAL\_STORAGE

android.Manifest.permission. CAMERA

# Services

## Location

LocationManagerService

<http://androidxref.com/6.0.1_r10/xref/frameworks/base/services/core/java/com/android/server/LocationManagerService.java>

CM12 源码

requestLocationUpdates()

private int getAllowedResolutionLevel(int pid,int uid){

mContext.checkPermission(android.Manifest.permission.ACCESS\_FINE\_LOCATION,pid,uid)

}

## Camera

android.hardware.Camera

<http://androidxref.com/6.0.1_r10/xref/frameworks/base/core/java/android/hardware/Camera.java>

package android.hardware;

public class Camera{

Camera(int cameraId){

int err = cameraInitNormal(cameraId);

}

private int cameraInitNormal(int cameraId){

// cameraInitVersion(cameraId,

}

}

public static final int CAMERA\_HAL\_API\_VERSION\_NORMAL\_CONNECT = -2;

<http://androidxref.com/6.0.1_r10/xref/frameworks/base/core/jni/android_hardware_Camera.cpp>

static jint android\_hardware\_Camera\_native\_setup(

JNIEnv\* env,

jobject this,

jobject weak\_this,

jint cameraId,

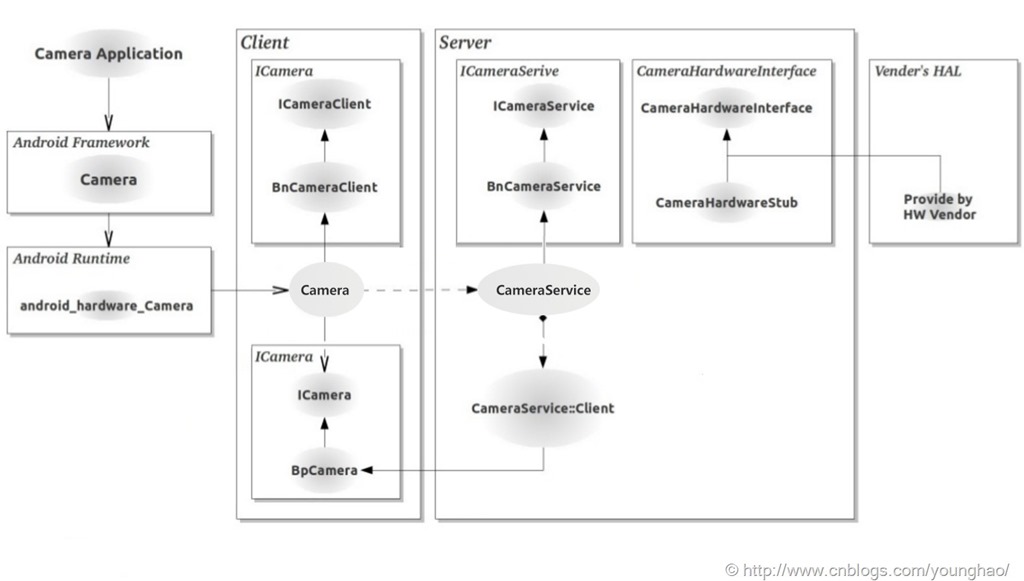
jint halVersion,

jstring clientPackageName

){

}

<http://androidxref.com/6.0.1_r10/xref/frameworks/av/services/camera/libcameraservice/api1/CameraClient.cpp>



# Native 自实现Binder Proxy通信

1 获取安装包列表

2 获取TOP Activity

adb shell dumpsys activity top

# ROM MI 7.0

PackageManagerServiceInjector

checkGetInstalledAppsPermission(int pid,int uid,String desc){

UserHandler.getAppId(uid)

}

public final class UserHandler{

public static final int PER\_USER\_RANGE = 100000

public static final int getAppId(int uid){

return uid % PER\_USER\_RANGE

}

}

public static String getPackageNameByPid(int pid){

ActivityManagerService manager;

mPidsSelfLocked.get()

}

sAppOpsService.noteOperation(int op,int uid,String packageName)

userId = 10224

# References

<https://blog.csdn.net/mhhyoucom/article/details/78794950>

<https://www.cnblogs.com/jiangz/p/3330245.html>

<https://blog.csdn.net/xiaosayidao/article/details/50450448>

<https://dzone.com/articles/depth-android-package-manager>

<http://www.vogella.com/tutorials/AndroidCamera/article.html>

<https://www.javatpoint.com/android-camera-tutorial>