Storage

Java

StorageManagerService

Native

Vold、sdcard

目前研究的结果是carmera进程被赋予了write的view，通过adb shell 进入手机发现write view 的mode mask 组没有赋予everybody 组对sdcard进行写的权限。

**[next action]**

  Write view 目前是由sdcard进行进行创建的，所以需要ssd storage 的owner来check write view是否允许组everybody gid 能够对sdcard 进行读写操作。

  目前来看Johnson.sh  的这笔patch就是保证write view 能够保证everybody gid能够对sdcard具有读写的权限。

**[分析过程]**

**1.对进程来说目前使用三个不同的fuse view来管理进程对sdcard的读写权限：**

/mnt/runtime/default - 对所有的应用、root 命名空间（adb 和其他系统组件）可见，而无需任何权限

/mnt/runtime/read - 对有 READ\_EXTERNAL\_STORAGE 权限的应用可见。

/mnt/runtime/write - 对有 WRITE\_EXTERNAL\_STORAGE 权限的应用可见。

**2.view mode的赋予流程：**

//app 进程启动

StorageManagerInternalImpl--->getExternalStorageMountMode

PackageManagerService----->new StorageManagerInternal.ExternalStorageMountPolicy()---->getMountMode

ForkAndSpecializeCommon----->MountEmulatedStorage

//app权限动态改变时候

PermissionManagerService---->grantRuntimePermission

StorageManagerService----->onExternalStoragePolicyChanged----->remountUidExternalStorage

VolumeManager---->remountUid

**3.OCL的default、write view的mode mask 和gid 如下：**

在没有”86684240eb5753bb97c2cfc93d1d25fa1870f8f1” 这笔patch之前app通过WRITE\_MEDIA\_STORAGE permission获得了default view 同时也包含有group sdcard\_rw,

所以可以对sdcard进行write。

**Default view：**

htc\_ocluhljapan:/mnt/runtime/default # ls -l

total 132

drwx**rwx**--x 1 root sdcard\_rw     131072 1969-12-31 19:00 582E-C944     **----->组sdcard\_rw是可以对sdcard(582E-C944)进行读写的。**

**write view：**

htc\_ocluhljapan:/mnt/runtime/write # ls -l

total 132

drwx**r-x**r-x 1 root everybody 131072 1969-12-31 19:00 582E-C944    **----->组everybody只有对sdcard(582E-C944)进行读的权限。**

**android\_filesystem\_config.h**

#define AID\_EVERYBODY 9997 **/\* shared between all apps in the same profile \*/**

**//camera 进程的grops中是包含everybody gid**

   archermind@archermind:/work$ adb shell cat /proc/4769/status

Name:      m.htc.cameragep

Groups:     1015 1023 3003 **9997** 20043 50043

**4.对google patch的解读：**

86684240eb5753bb97c2cfc93d1d25fa1870f8f1：

Media process should run with "write" access.

The WRITE\_MEDIA\_STORAGE permission had inadvertently been giving apps

the "default" view of storage.  This had worked for a long since,

since we also gave them the "sdcard\_rw" permission, but a recent

security patch broke this for secondary users.

Apps holding this permission should have been mounted "write" all

along, and relied on that view to access storage devices.  This also

means they no longer need the "sdcard\_rw" GID.

按照这个解释是目前多媒体进程只要有write view就可以了。

之前WRITE\_MEDIA\_STORAGE这个permission不光给app赋予default view同时也给了sdcard\_rw这个group的permission。

[code analysis]

//view 创建的过程：

[system/core/sdcard/sdcard.cpp#182](http://axr.htc.com/source/xref/P_StockUI/system/core/sdcard/sdcard.cpp" \l "182)

Official document:

<https://developer.android.com/training/articles/scoped-directory-access>

Official sample code:

<https://github.com/googlesamples/android-ScopedDirectoryAccess/blob/master/Application/src/main/java/com/example/android/scopeddirectoryaccess/ScopedDirectoryAccessFragment.java>

It is designed for accessing to specific or entire external storage directories, such as SD cards

It need to grant permission to each default folder first.

The following code is an example of how to open SD card ‘s DCIM directory.

StorageManager sm = (StorageManager)getSystemService(Context.STORAGE\_SERVICE);

List<StorageVolume> sv = sm.getStorageVolumes(); // get all storage for (SrorageVolume volume : sv) {

if (volume.isPrimary()) {

// don’t need to open primary storage

continue;

}

// ask for permission

Intent intent = volume.createAccessIntent(Environment.DCIM);

startActivityForResult(intent, request\_code);

break;

}