

密级状态：绝密() 秘密() 内部() 公开(✓)

Security Class: Top-Secret () Secret () Internal () Public (✓)

RK3326_ANDROID9.0_SDK 发布说明

RK3326_ANDROID9.0_SDK_Release_Instruction

(技术部, 第二系统产品部)

(Technical Department, R & D Dept. II)

文件状态:	当前版本:	V1.3
Status:	Current Version:	
[] 正在修改	作 者:	卞金晨
[] Modifying	Author:	Bian Jinchen
[✓] 正式发布	完成日期:	2019-05-29
[✓] Released	Finish Date:	
	审 核:	黄祖芳/吴良清
	Auditor:	Huang Zufang/Wu Liangqing
	完成日期:	2019-05-29
	Finish Date:	

福州瑞芯微电子股份有限公司

Fuzhou Rockchip Electronics Co., Ltd

(版本所有, 翻版必究)

(All rights reserved)

版本历史 Revision History

版本号 Version no.	作者 Author	修改日期 Revision Date	修改说明 Revision description	备注 Remark
Beta_V0.0.1	卞金晨 Bian Jinchen	2018.09.25	Beta 测试版 Beta version	
V1.0	卞金晨 Bian Jinchen	2018.11.15	发布正式版 Release official version	
V1.1	吴惊晨 Wu Jingchen	2019.01.30	支持 px30 Support PX30	
V1.2	吴惊晨 Wu Jingchen	2019.03.01	烧写工具规范 Update flashing tool version	
V1.3	吴惊晨 Wu Jingchen	2019.05.29	更新 GO 版本 lunch 选项 Update lunch option of GO version	

目 录 Contents

1	概述 Overview	1
2	主要支持功能 Main Functions	1
3	SDK 获取说明 SDK Acquisition	1
3.1	获取 SDK Acquire SDK	1
3.2	补充说明 Additional remarks	3
4	SDK 编译说明 SDK compilation instruction	4
4.1	JDK 安装 JDK installation	4
4.2	编译模式 Compilation mode	5
4.3	代码编译 Code compilation	5
4.3.1	uboot 编译步骤 uboot compiling steps	5
4.3.2	kernel 编译步骤 kernel compiling steps.....	6
4.3.3	Android 编译及固件生成步骤 Android compiling and image build steps	6
4.4	刷机说明 Flashing instruction	8
附录 A	编译开发环境搭建 Compiling and development environment setup	10
附录 B	SSH 公钥操作说明 SSH public key operation instruction	14
附录 B-1	SSH 公钥生成 SSH public key generation	14
附录 B-2	使用 key-chain 管理密钥 Use key-chain to manage the private key	15
附录 B-3	多台机器使用相同 ssh 公钥 Multiple devices use the same ssh public key	16
附录 B-4	一台机器切换不同 ssh 公钥 Switch different ssh public keys on one device	17
附录 B-5	密钥权限管理 Private key authority management	18
附录 B-6	Git 权限申请说明 Git authority application instruction	18

1 概述 Overview

本 SDK 是基于谷歌 Android9.0 64bit 系统，适配瑞芯微 RK3326 芯片的软件包，适用于 RK3326 平台开发的产品。

This SDK is compatible with Rockchip RK3326 chipset software package based on Google Android9.0 64bit system. It is suitable for the product development based on RK3326 platform.

2 主要支持功能 Main Functions

参数 Parameter	模块名 Module Names
数据通信 Data communication	Wi-Fi、USB 以太网卡、USB、SDCARD Wi-Fi、USB Ethernet card、USB、SDCARD
应用程序 Application	Launcher3、APK 安装器、浏览器、计算器、日历、相机、闹钟、下载、电子邮件、资源管理器、GMS 应用、音乐、录音、设置、视频播放器 Launcher3, APK installer, browser, calculator, calendar, camera, clock, download, e-mail, resource manager, GMS application, music, audio recorder, setting, video player

3 SDK 获取说明 SDK Acquisition

3.1 获取 SDK Acquire SDK

SDK 通过瑞芯微代码服务器对外发布。其编译开发环境，参考[附录 A 编译开发环境搭建](#)。

SDK is released through Rockchip code server. Please refer to [Appendix A](#) to setup the compiling and development environment.

客户向瑞芯微技术窗口申请 SDK，需同步提供 SSH 公钥进行服务器认证授权，获得授权后即可同步代码。关于瑞芯微代码服务器 SSH 公钥授权，请参考[附录 B SSH 公钥操作说明](#)。

Customers apply SDK from Rockchip FAE contact, and will be able to sync code after obtaining the server certificate authorization with SSH public key. For more details about Rockchip code server SSH public key authorization, please refer to [Appendix B SSH public key operation instruction](#).

RK3326_ANDROID9.0_SDK 下载地址如下：

RK3326_ANDROID9.0_SDK download address is as below:

```
repo init
--repo-url=ssh://git@www.rockchip.com.cn:2222/repo-release/tools/repo.git -u ssh://git@www.rockchip.com.cn:2222/Android_pie_stable/platform/rk3326/manifests.git -m RK3326_Android_Pie_release.xml
```

如果需要包含 GMS 包的 SDK(需要开通权限), 使用如下地址:

If need SDK with GMS which requires additional authority, use the address as below:

```
repo init
--repo-url=ssh://git@www.rockchip.com.cn:2222/repo-release/tools/repo.git -u ss
h://git@www.rockchip.com.cn:2222/Android_pie_stable/platform/rk3326/manifests
.git -m RK3326_Android_Pie_Express_release.xml
```

注, repo 是 google 用 Python 脚本写的调用 git 的一个脚本, 主要是用来下载、管理 Android 项目的软件仓库, 其下载地址如下:

Note: repo is a script invoking git developed by Google using Python script, and mainly used to download, manage Android project software lib. The download address is as below:

```
git clone ssh:// git@www.rockchip.com.cn:2222/repo-release/tools/repo
```

为方便客户快速获取 SDK 源码, 瑞芯微技术窗口通常会提供对应版本的 SDK 初始压缩包。

以 Rk3326_Android_Pie_release_20180925.tar.gz 为例, 拷贝到该初始化包后, 通过如下命令可检出源码:

Rockchip FAE contact usually will provide the initial compressed package of the corresponding version SDK in order to help customers acquire SDK source code quickly. Take Rk3326_Android_Pie_release_20180925.tar.gz as an example, you can sync the source code through below command after copy the initial package:

```
mkdir RK3326
tar zxvf Rk3326_Android_Pie_release_20180925.tar.gz -C RK3326
cd RK3326
.repo/repo/repo sync -l
.repo/repo/repo sync
```

3.2 补充说明 Additional remarks

Android9.0 SDK 已不再支持 UMS 功能, 平台设备皆使用合并分区;

Android9.0 SDK does not support UMS function any more, all the devices use combined partition.

Android9.0 SDK 已支持全盘加密功能;

Android9.0 SDK already supports full disk encryption function.

Android9.0 SDK 已支持 Verified boot 2.0 (avb)的功能。

Android9.0 SDK already supports Verified boot 2.0 (avb) function.

4 SDK 编译说明 SDK compilation instruction

4.1 JDK 安装 JDK installation

Android9.0 系统编译依赖于 JAVA 8。编译之前需安装 OpenJDK。

Android9.0 system compiling is dependent on JAVA 8. Need to install OpenJDK before compiling.

安装命令如下。

Install command is as below:

```
sudo apt-get install openjdk-8-jdk
```

配置 JAVA 环境变量，例如，安装路径为/usr/lib/jvm/java-8-openjdk-amd64，可在终端执行如下命令配置环境变量。

Configure JAVA environment variable, for example, if the install path is /usr/lib/jvm/java-8-openjdk-amd64, it is able to execute below command to configure environment variable at the termination.

```
export JAVA_HOME=/usr/lib/jvm/java-8-openjdk-amd64
export PATH=$JAVA_HOME/bin:$PATH
export CLASSPATH=.:$JAVA_HOME/lib:$JAVA_HOME/lib/tools.jar
```

4.2 编译模式 **Compilation mode**

SDK 默认以 userdebug 模式编译。

SDK default compiling mode is userdebug.

使用 adb 时,需要先执行 adb root 使 shell 获取 root 权限,进而执行其它像 adb remount、adb push 等操作,其中 adb remount 前要先 disable-verity, 执行 adb disable-verity

When using adb, first need to execute adb root to make shell acquire root authority, and then execute other operations such as adb remount, adb push, and so on. Need to first execute adb disable-verity for disable-verity before adb remount.

4.3 代码编译 **Code compilation**

4.3.1 uboot 编译步骤 **uboot compiling steps**

```
cd u-boot  
make clean  
make mrproper  
./make.sh rk3326 适用于 RK3326 芯片 suitable for RK3326 chipset  
./make.sh px30 适用于 px30 芯片 suitable for PX30 chipset
```

编译完, 会生成

After compilation, it will generate:

trust.img

rk3326_loader_v1.10.112.bin 适用于 RK3326 芯片 suitable for RK3326 chipset

px30_loader_v1.10.112.bin 适用于 px30 芯片 (保证 u_boot 目录只有一个 bin)
suitable for PX30 chipset (ensure u_boot directory only has one bin)

uboot.img

三个文件。

three files.

4.3.2 kernel 编译步骤 kernel compiling steps

1)、RK3326 平板样机配置与编译如下:

The configuration and compilation of RK3326 tablet are as below:

使用的 dts 是: arch/arm64/boot/dts/rockchip/rk3326-863-lp3-v10-avb.dts

The corresponding dts is: arch/arm64/boot/dts/rockchip/rk3326-863-lp3-v10-avb.dts

```
cd kernel

make ARCH=arm64 rockchip_defconfig

make ARCH=arm64 rk3326-863-lp3-v10-avb.img 适用于 RK3326 芯片
suitable for RK3326 chipset

make ARCH=arm64 px30-evb-ddr3-v10-avb.img 适用于 px30 芯片 suitable
for PX30 chipset
```

4.3.3 Android 编译及固件生成步骤 Android compiling and image build steps

客户按实际编译环境配置好 JDK 环境变量后, 按照以下步骤配置完后, 执行 make 即可。

Customers configure JDK environment variable per below steps according to the actual compiling environment, and then execute make.

```
$ source build/envsetup.sh
```

```
$ lunch
```

```
You're building on Linux
```

```
Lunch menu... pick a combo:
```

1. aosp_arm-eng
2. aosp_arm64-eng
3. aosp_mips-eng
4. aosp_mips64-eng
5. aosp_x86-eng
6. aosp_x86_64-eng
7. rk3326_evb-userdebug
8. rk3326_evb-user
9. rk3326_mid-userdebug
10. rk3326_mid-user

11. rk3326_m2g-userdebug
12. rk3326_m2g-user
13. rk3326_32bit-userdebug
14. rk3326_32bit-user
15. rk3326_go-userdebug
16. rk3326_go-user

如果是平板: **If for tablet:**

Android Pie Go 版本, 选择 rk3326_go-userdebug, 输入对应序号 15。

Android Pie Go version, select rk3326_go-userdebug , input the corresponding serial number 15.

Android Pie Normal 版本, 选择 rk3326_m2g-userdebug, 输入对应序号 11。

Android Pie Normal version, select rk3326_m2g-userdebug, input the corresponding serial number 11.

Android Pie 32 位版本, 选择 rk3326_32bit-userdebug, 输入对应序号 13。

\$ make -j4

Android Pie 32bit version, select rk3326_32bit-userdebug, input the corresponding serial number 13.

注: Go 为 Google 专门针对低内存设备(1G DDR 及以下) 进行优化的 Android 版本; Normal 为传统的 GSM 版本, 适用于 2G DDR 及以上的设备.

Note: Go is the Android version optimized especially for low memory device (1G DDR or lower) by Google. Normal is the traditional GSM version which is suitable for the device with 2G DDR or higher.

完成编译后, 执行 SDK 根目录下的 mkimage.sh 脚本生成固件, 所有烧写所需的镜像将都会拷贝于 rockdev/Image-rk3326_mid 目录。

After compilation, execute mkimage.sh script under SDK root directory to generate images. All the mirror files required for flashing will be copied to rockdev/Image-rk3326_mid directory.

rockdev/Image-rk3326_mid

└─ boot.img

└─ kernel.img

```
└─ MiniLoaderAll.bin
└─ vbmeta.img
└─ dtbo.img
└─ misc.img
└─ oem.img
└─ parameter.txt
└─ pcba_small_misc.img
└─ pcba_whole_misc.img
└─ recovery.img
└─ resource.img
└─ system.img
└─ trust.img
└─ uboot.img
└─ vendor.img
```

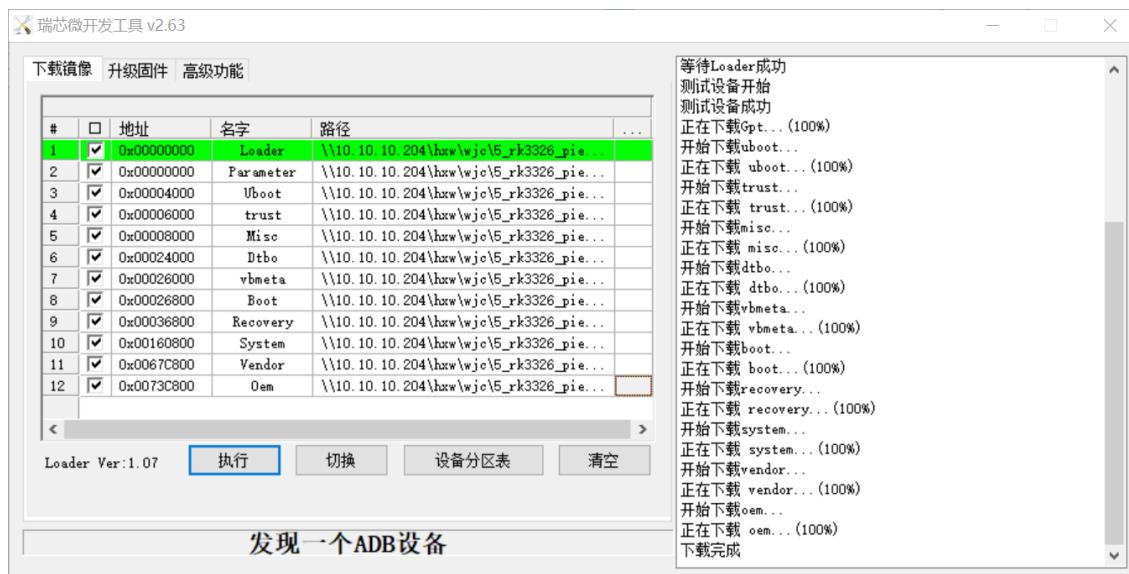
得到所有镜像文件后，为了方便烧写及量产，通常可手动将这些单独的镜像通过脚本打包成为 update.img。

After acquiring all the mirror files, usually you can manually use the script to package them as update.img which is convenient for flashing and MP.

4.4 刷机说明 **Flashing instruction**

SDK 提供烧写工具，如下图所示，版本为 2.63。编译生成相应的固件后，进入 loader 模式，即可进行刷机。对于已烧过其它固件的机器，请选择低格设备，擦除 idb，然后进行刷机。

SDK provides the flashing tool with version 2.63 shown as below picture. After compiling to generate images, enter loader mode, and then you can flash the images. For those devices with existing images, please select to format the device, erase idb, and then flash the images.



注：烧写工具必须使用 2.63 及以上版本的工具，量产工具使用 1.65 及以上版本；linux 下的烧写工具使用 1.34 及以上版本。

Note: The flashing tool version must be 2.63 or above, and the MP tool version must be 1.65 or above. Linux flashing tool version must be 1.34 or above.

附录 A 编译开发环境搭建 **Compiling and development environment setup**

1. Initializing a Build Environment

This section describes how to set up your local work environment to build the Android source files. You must use Linux or Mac OS; building under Windows is not currently supported.

For an overview of the entire code-review and code-update process, see Life of a Patch.

Note: All commands in this site are preceded by a dollar sign (\$) to differentiate them from output or entries within files. You may use the Click to copy feature at the top right of each command box to copy all lines without the dollar signs or triple-click each line to copy it individually without the dollar sign.

2. Choosing a Branch

Some requirements for the build environment are determined by the version of the source code you plan to compile. For a full list of available branches, see Build Numbers. You can also choose to download and build the latest source code (called master), in which case you will simply omit the branch specification when you initialize the repository.

After you have selected a branch, follow the appropriate instructions below to set up your build environment.

3. Setting up a Linux build environment

These instructions apply to all branches, including master.

The Android build is routinely tested in house on recent versions of Ubuntu LTS (14.04) and Debian testing. Most other distributions should have the required build tools available.

For Gingerbread (2.3.x) and newer versions, including the master branch, a 64-bit

environment is required. Older versions can be compiled on 32-bit systems.

Note: See Requirements for the complete list of hardware and software requirements, then follow the detailed instructions for Ubuntu and Mac OS below.

4. Installing the JDK

The master branch of Android in the Android Open Source Project (AOSP) comes with prebuilt versions of OpenJDK below prebuilts/jdk/ so no additional installation is required.

Older versions of Android require a separate installation of the JDK. On Ubuntu, use OpenJDK. See JDK Requirements for precise versions and the sections below for instructions.

For Ubuntu >= 15.04

Run the following:

```
sudo apt-get update
```

```
sudo apt-get install openjdk-8-jdk
```

For Ubuntu LTS 14.04

There are no available supported OpenJDK 8 packages for Ubuntu 14.04. The Ubuntu 15.04 OpenJDK 8 packages have been used successfully with Ubuntu 14.04. Newer package versions (e.g. those for 15.10, 16.04) were found not to work on 14.04 using the instructions below.

1. Download the .deb packages for 64-bit architecture from

old-releases.ubuntu.com:

[openjdk-8-jre-headless_8u45-b14-1_amd64.deb](#) with SHA256

0f5aba8db39088283b51e00054813063173a4d8809f70033976f83e214ab56c0

[openjdk-8-jre_8u45-b14-1_amd64.deb](#) with SHA256

9ef76c4562d39432b69baf6c18f199707c5c56a5b4566847df908b7d74e15849

[openjdk-8-jdk_8u45-b14-1_amd64.deb](#) with SHA256

6e47215cf6205aa829e6a0a64985075bd29d1f428a4006a80c9db371c2fc3c4c

2. Optionally, confirm the checksums of the downloaded files against the SHA256 string listed with each package above. For example, with the sha256sum tool:

```
sha256sum {downloaded.deb file}
```

3. Install the packages:

```
sudo apt-get update
```

Run dpkg for each of the .deb files you downloaded. It may produce errors due to missing dependencies:

```
sudo dpkg -i {downloaded.deb file}
```

To fix missing dependencies:

```
sudo apt-get -f install
```

Update the default Java version - optional

Optionally, for the Ubuntu versions above update the default Java version by running:

```
sudo update-alternatives --config javasudo update-alternatives --config javac
```

Note: If, during a build, you encounter version errors for Java, see Wrong Java version for likely causes and solutions.

Installing required packages (Ubuntu 14.04)

You will need a 64-bit version of Ubuntu. Ubuntu 14.04 is recommended.

```
sudo apt-get install git-core gnupg flex bison gperf build-essential zip curl
```

```
zlib1g-dev gcc-multilib g++-multilib libc6-dev-i386 lib32ncurses5-dev
x11proto-core-dev libx11-dev lib32z-dev ccache libgl1-mesa-dev libxml2-utils
xsltproc unzip
```

Note: To use SELinux tools for policy analysis, also install the python-networkx package. Note: If you are using LDAP and want to run ART host tests, also install the libnss-sss:i386 package.

5. Configuring USB Access

Under GNU/Linux systems (and specifically under Ubuntu systems), regular users can't directly access USB devices by default. The system needs to be configured to allow such access.

The recommended approach is to create a file `/etc/udev/rules.d/51-android.rules` (as the root user) and to copy the following lines in it. `<username>` must be replaced by the actual username of the user who is authorized to access the phones over USB.

```
# adb protocol on passion (Rockchip products)

SUBSYSTEM=="usb",                                ATTR{idVendor}=="2207",
ATTR{idProduct}=="0010", MODE="0600", OWNER="<username>"
```

Those new rules take effect the next time a device is plugged in. It might therefore be necessary to unplug the device and plug it back into the computer.

This is known to work on both Ubuntu Hardy Heron (8.04.x LTS) and Lucid Lynx (10.04.x LTS). Other versions of Ubuntu or other variants of GNU/Linux might require different configurations.

References : <http://source.android.com/source/initializing.html>

附录 B SSH 公钥操作说明 SSH public key operation instruction

附录 B-1 SSH 公钥生成 SSH public key generation

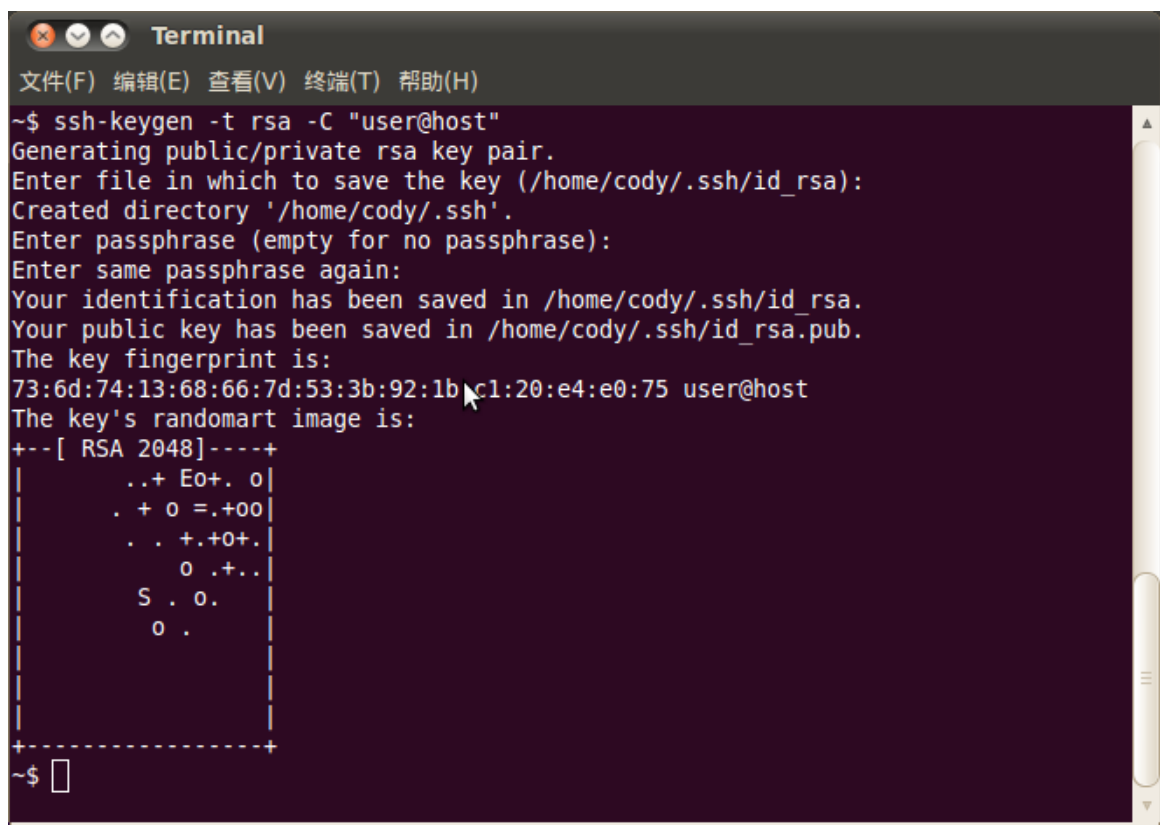
使用如下命令生成：

Use below command to generate:

```
ssh-keygen -t rsa -C "user@host"
```

请将 **user@host** 替换成您的邮箱地址。

Please replace **user@host** with your email address.



```
Terminal
文件(F) 编辑(E) 查看(V) 终端(T) 帮助(H)

~$ ssh-keygen -t rsa -C "user@host"
Generating public/private rsa key pair.
Enter file in which to save the key (/home/cody/.ssh/id_rsa):
Created directory '/home/cody/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/cody/.ssh/id_rsa.
Your public key has been saved in /home/cody/.ssh/id_rsa.pub.
The key fingerprint is:
73:6d:74:13:68:66:7d:53:3b:92:1b:c1:20:e4:e0:75 user@host
The key's randomart image is:
+---[ RSA 2048]-----+
|      .+. Eo+. 0|
|     . + 0 =.+00|
|    . . +.+0+. |
|      0 .+..   |
|     S . 0.    |
|      0 .      |
+-----+
~$
```

命令运行完成会在你的目录下生成 **key** 文件。

It will generate key file in your directory after the command executes successfully.

```
~$ ls -l .ssh/
总用量 8
-rw----- 1 cody cody 1675 2012-10-15 11:38 id_rsa
-rw-r--r-- 1 cody cody 391 2012-10-15 11:38 id_rsa.pub
```

请妥善保存生成的私钥文件 **id_rsa** 和密码，并将 **id_rsa.pub** 发邮件给 SDK 发布服务器的

管理员。

Please keep carefully the generated private key file `id_rsa` and password, and send `id_rsa.pub` to SDK release server admin through email.

附录 B-2 使用 **key-chain** 管理密钥 Use key-chain to manage the private key

推荐您使用比较简易的工具 `keychain` 管理密钥。

Recommend you use the simple tool `keychain` to manage the private key.

具体使用方法如下：

The detailed usage is as below:

1. 安装 `keychain` 软件包：

Install `keychain` software package:

```
$sudo aptitude install keychain
```

2. 配置使用密钥：

Configure to use the private key:

```
$vim ~/.bashrc
```

增加下面这行：

Add below command:

```
eval `keychain --eval ~/.ssh/id_rsa`
```

其中，`id_rsa` 是私钥文件名称。

Among which, `id_rsa` is the file name of the private key.

以上配置以后，重新登录控制台，会提示输入密码，只需输入生成密钥时使用的密码即可，若无密码可不输入。

Log in the console again after configuring as above, and it will prompt to input the password. Only need to input the password used for generating the private key if there is one.

另外，请尽量不要使用 `sudo` 或 `root` 用户，除非您知道如何处理，否则将导致权限以及密钥

管理混乱。

Besides, please avoid using sudo or root user unless you know how to deal with, otherwise it will case the authority and private key management problems.

附录 B-3 多台机器使用相同 **ssh** 公钥 Multiple devices use the same ssh public key

在不同机器使用，可以将你的 **ssh** 私钥文件 `id_rsa` 拷贝到要使用的机器的“`~/.ssh/id_rsa`”即可。

In order to use on different devices, you can copy ssh private key file `id_rsa` to the target device “`~/.ssh/id_rsa`”.

在使用错误的私钥会出现如下提示，请注意替换成正确的私钥。

Below hint will show up if using the wrong private key. Please replace with the correct private key.

```
~/tmp$ git clone git@172.16.10.211:rk292x/mid/4.1.1_r1
Initialized empty Git repository in /home/cody/tmp/4.1.1_r1/.git/
The authenticity of host '172.16.10.211 (172.16.10.211)' can't be established.
RSA key fingerprint is fe:36:dd:30:bb:83:73:e1:0b:df:90:e2:73:e4:61:46.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '172.16.10.211' (RSA) to the list of known hosts.
git@172.16.10.211's password: 
```

添加正确的私钥后，就可以使用 **git** 克隆代码，如下图。

After adding the correct private key, you can use git to clone code, shown as below picture:

```
~$ cd tmp/
~/tmp$ git clone git@172.16.10.211:rk292x/mid/4.1.1_r1
Initialized empty Git repository in /home/cody/tmp/4.1.1_r1/.git/
The authenticity of host '172.16.10.211 (172.16.10.211)' can't be established.
RSA key fingerprint is fe:36:dd:30:bb:83:73:e1:0b:df:90:e2:73:e4:61:46.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '172.16.10.211' (RSA) to the list of known hosts.
remote: Counting objects: 237923, done.
remote: Compressing objects: 100% (168382/168382), done.
Receiving objects: 9% (21570/237923), 61.52 MiB | 11.14 MiB/s
```

添加 **ssh** 私钥可能出现如下提示错误。

Below error may occur while adding ssh private key:

Agent admitted failure to sign using the key

在 console 输入如下命令即可解决。

Input below command at console can fix it.

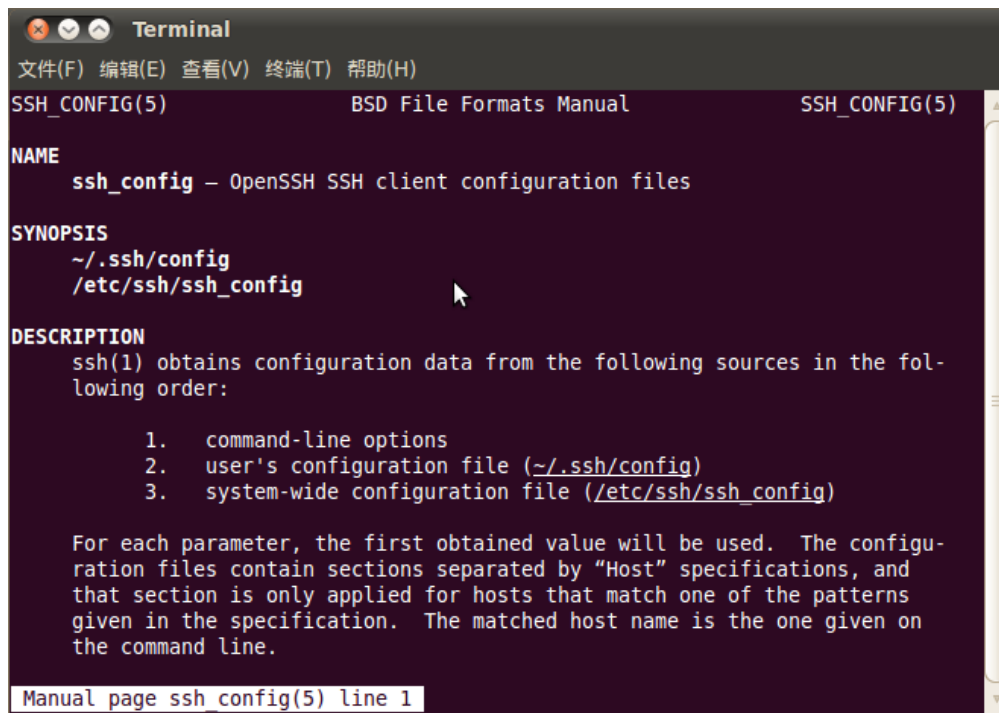
```
ssh-add ~/.ssh/id_rsa
```

附录 B-4 一台机器切换不同 **ssh** 公钥 Switch different ssh public keys on one device

可以参考 ssh_config 文档配置 ssh。

You can refer to ssh_config document to configure ssh.

```
~$ man ssh_config
```



通过如下命令，配置当前用户的 ssh 配置。

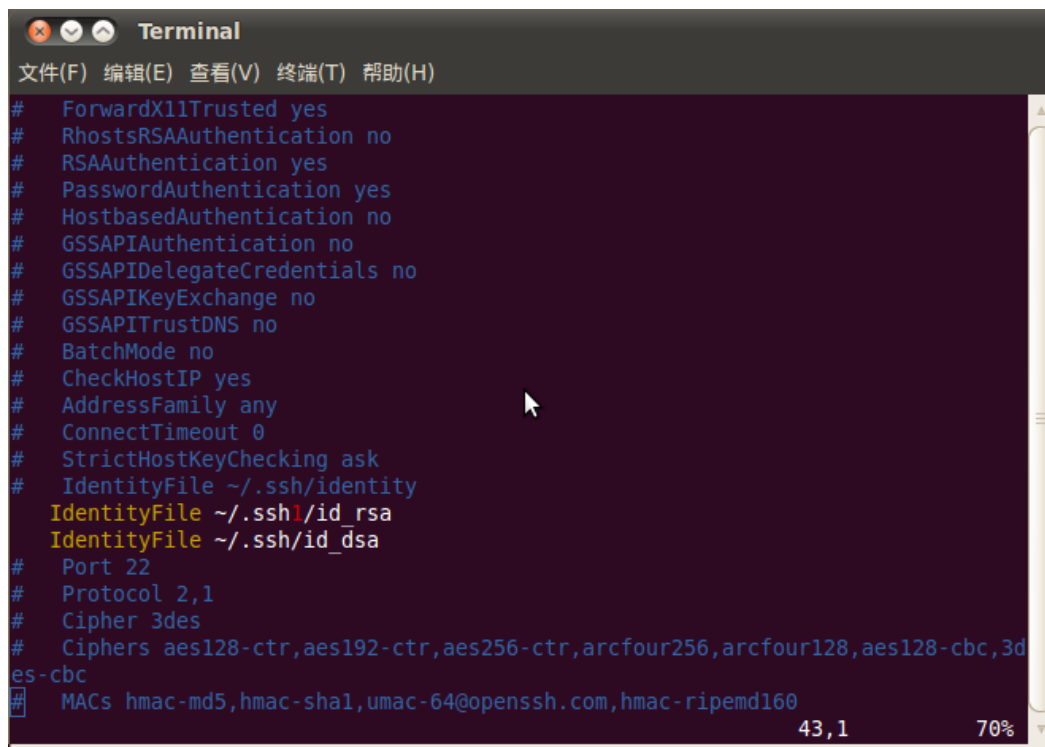
Use below commands to configure ssh for current user.

```
~$ cp /etc/ssh/ssh_config ~/.ssh/config
```

```
~$ vi ~/.ssh/config
```

如图，将 ssh 使用另一个目录的文件 “~/.ssh1/id_rsa” 作为认证私钥。通过这种方法，可以切换不同的的密钥。

As below picture, identify another directory ssh file “~/.ssh1/id_rsa” as certificate private key. In this way, you can switch different private keys.



```
Terminal
文件(F) 编辑(E) 查看(V) 终端(T) 帮助(H)
# ForwardX11Trusted yes
# RhostsRSAAuthentication no
# RSAAuthentication yes
# PasswordAuthentication yes
# HostbasedAuthentication no
# GSSAPIAuthentication no
# GSSAPIDelegateCredentials no
# GSSAPIKeyExchange no
# GSSAPITrustDNS no
# BatchMode no
# CheckHostIP yes
# AddressFamily any
# ConnectTimeout 0
# StrictHostKeyChecking ask
# IdentityFile ~/.ssh/identity
IdentityFile ~/.ssh/id_rsa
IdentityFile ~/.ssh/id_dsa
# Port 22
# Protocol 2,1
# Cipher 3des
# Ciphers aes128-ctr,aes192-ctr,aes256-ctr,arcfour256,arcfour128,aes128-cbc,3des-cbc
# MACs hmac-md5,hmac-sha1,umac-64@openssh.com,hmac-ripemd160
43,1 70%
```

附录 B-5 密钥权限管理 Private key authority management

服务器可以实时监控某个 key 的下载次数、IP 等信息，如果发现异常将禁用相应的 key 的下载权限。

The server can real-time monitor the information for the specific key such as the download times, IP etc. If there is abnormal case, it will prohibit the download authority of the corresponding key.

请妥善保管私钥文件。并不要二次授权与第三方使用。

Please keep carefully the private key file. DO NOT re-authorize it to the third party.

附录 B-6 Git 权限申请说明 Git authority application instruction

参考上述章节，生成公钥文件，发邮件至 fae@rock-chips.com，申请开通 SDK 代码下载权限。

Refer to above chapters, generate the public key file, and send email to

fae@rock-chips.com applying for SDK code download authority.