

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

RK3399Pro_ANDROID8.1_SDK_ Release_Instruction

(Technical Department, R & D Dept. II)

Status: [] Modifying [<input checked="" type="checkbox"/>] Released	Version:	V1.01
	Author:	Zhou Weixin
	Date:	2018-12-15
	Auditor:	Wu Liangqing
	Date:	2018-12-15

Fuzhou Rockchips Electronics Co., Ltd

(All rights reserved)

Revision History

Version no.	Author	Revision Date	Revision description	Remark
V1.00	Zhou Weixin	2018.11.30	Initial version release	
V1.01	Zhou Weixin	2018.12.15	Add V11 evb compiling description	

Content

1	Overview.....	1
2	Main Functions.....	1
3	SDK Acquisition.....	1
3.1	How to get SDK.....	1
3.2	Additional remarks.....	2
4	SDK compilation instruction	3
4.1	JDK installation	3
4.2	Compilation mode.....	3
4.3	Code compilation.....	3
4.3.1	Uboot compiling steps.....	3
4.3.2	Kernel compiling steps	4
4.3.3	Android compiling and image build steps	4
4.4	Flashing instruction.....	5
	Appendix A Compiling and development environment setup	7
	Appendix B SSH public key operation instruction	11
	Appendix B-1 SSH public key generation.....	11
	Appendix B-2 Use key-chain to manage the private key	11
	Appendix B-3 Multiple devices use the same ssh public key	12
	Appendix B-4 Switch different ssh public keys on one device	13
	Appendix B-5 Private key authority management	14
	Appendix B-6 Git authority application instruction	14

1 Overview

This SDK is compatible with Rockchip RK3399Pro chipset software package based on Google Android8.1 64bit system. It is suitable for the product development based on RK3399Pro platform.

2 Main Functions

Parameter	Module Names
Data communication	Wi-Fi、USB Ethernet card、USB、SDCARD
Application	Launcher3、APK installer、browser、calculator、calendar、camera、clock、download、e-mail、resource manager、GMS application、music、audio recorder、setting、video player

3 SDK Acquisition

3.1 How to get SDK

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

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](#).

RK3399PRO_ANDROID8.1_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_oreo_stable/platform/rk3399pro/manifests.git
```

```
-m Rk3399pro_Android_Oreo_release.xml
```

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
```

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 [Rk3399Pro_Android8.1_SDK_V1.00_20181130.tar.gz](#) as an example, you can sync the source code through below command after copy the initial package:

```
mkdir rk3399pro
tar zxvf Rk3399Pro_Android8.1_SDK_V1.00_20181130.tar.gz -C rk3399pro
cd rk3399pro
.repo/repo/repo sync -l
.repo/repo/repo sync
```

Note: We may upgrade versions periodically to resolve some issues found in SDK. Customers can use the command `.repo/repo/repo sync` to upgrade the new versions periodically.

3.2 Additional remarks

Android8.1 SDK does not support UMS function any more, all the devices use combined partition.

Android8.1 SDK already supports full disk encryption function.

Android8.1 SDK already supports Verified boot function.

4 SDK compilation instruction

4.1 JDK installation

Android8.1 system compiling is dependent on JAVA 8. Need to install OpenJDK before compiling.

Install command is as below:

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

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 Compilation mode

SDK default compiling mode is userdebug.

While using adb, first need to execute adb root, adb disable-verity to close the verity feature of the system partition, then execute adb root, adb remount after reboot, and then execute push operation to debug.

4.3 Code compilation

4.3.1 Uboot compiling steps

```
cd u-boot
./make.sh rk3399pro
```

After compilation, it will generate trust.img, rk3399_loader_v1.15.115.bin and uboot.img the three files.

rk3399_loader_v1.15.115.bin is compatible with different DDR types and sizes.

The default running frequency is 800MHz.

4.3.2 Kernel compiling steps

If for evb_v10 board(green), kernel compiling method is as below:

```
cd kernel
make ARCH=arm64 rockchip_defconfig -j8
make ARCH=arm64 rk3399pro-evb-v10.img -j12
```

If for evb_v11 board(black), kernel compiling method is as below:

```
cd kernel
make ARCH=arm64 rockchip_defconfig -j8
make ARCH=arm64 rk3399pro-evb-v11.img -j12
```

4.3.3 Android compiling and image build steps

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

Android compiling:

```
source build/envsetup.sh
lunch rk3399pro-userdebug
make -j12
./mkimage.sh
```

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-xxx directory.

```
rockdev/Image-xxx
└─ boot.img
```

```
├── kernel.img
├── MiniLoaderAll.bin
├── 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
```

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. For the detailed package method, please refer to Chapter 8.5 of RK3399Pro_Android8.1_Software_Development_Guide_V1.00_20181130.

4.4 Flashing instruction

SDK provides the flashing tool with version 2.54 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.

瑞芯微开发工具 v2.54

下载镜像

升级固件

高级功能

#		地址	名字	路径	...
1	<input checked="" type="checkbox"/>	0x00000000	Loader	..\IMAGES\MiniLoaderAll bin	
2	<input checked="" type="checkbox"/>	0x00000000	Parameter	..\IMAGES\parameter.txt	
3	<input checked="" type="checkbox"/>	0x00002000	uboot	..\IMAGES\uboot.img	
4	<input checked="" type="checkbox"/>	0x00004000	trust	..\IMAGES\trust.img	
5	<input checked="" type="checkbox"/>	0x00006000	Misc	..\IMAGES\misc.img	
6	<input checked="" type="checkbox"/>	0x00008000	Resource	..\IMAGES\resource.img	
7	<input checked="" type="checkbox"/>	0x00010000	Kernel	..\IMAGES\kernel.img	
8	<input checked="" type="checkbox"/>	0x00020000	Boot	..\IMAGES\boot.img	
9	<input checked="" type="checkbox"/>	0x00030000	Recovery	..\IMAGES\recovery.img	
10	<input checked="" type="checkbox"/>	0x0018A000	System	..\IMAGES\system.img	
11	<input checked="" type="checkbox"/>	0x00592000	vendor	..\IMAGES\vendor.img	
12	<input checked="" type="checkbox"/>	0x00694000	oem	..\IMAGES\oem.img	

Loader Ver:1.12

执行

切换

设备分区表

清空

发现一个MASKROM设备

Note: The flashing tool version must be 2.54 or higher, and the MP tool version must be 1.6 or higher. Linux flashing tool version must be 1.31 or higher.

Appendix A 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

0f5aba8db39088283b51e00054813063173a4d8809f70033976f83e214ab56c

0

[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>

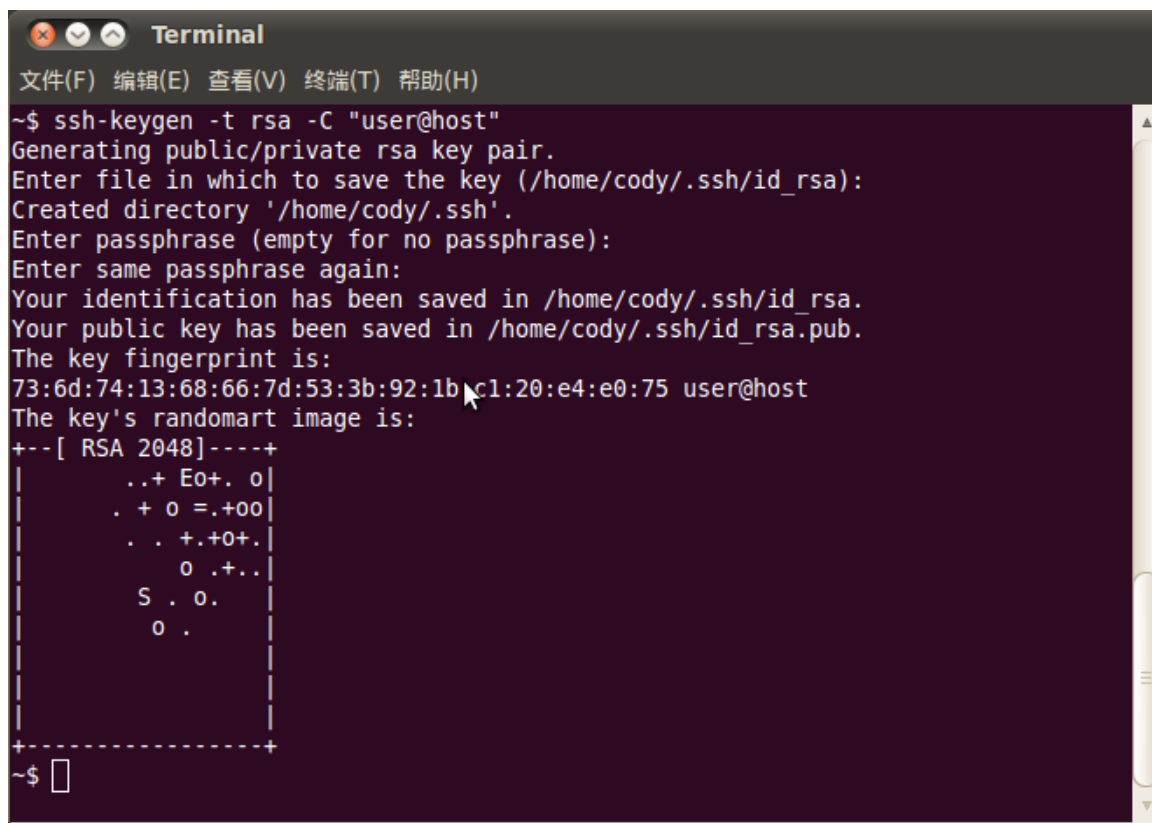
Appendix B SSH public key operation instruction

Appendix B-1 SSH public key generation

Use below command to generate:

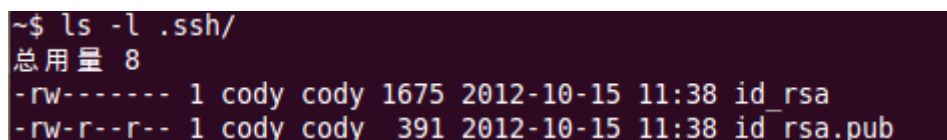
```
ssh-keygen -t rsa -C "user@host"
```

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:4c:1:20:e4:e0:75 user@host
The key's randomart image is:
+--[ RSA 2048 ]-----+
|      .+ Eo+. o|
|      . + o =.+oo|
|      . . +.+o+.|
|      o .+..|
|      S . o.|
|      o .|
+-----+
~$
```

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
```

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

Appendix B-2 Use key-chain to manage the private key

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

The detailed usage is as below:

1. Install keychain software package:

```
$sudo aptitude install keychain
```

2. Configure to use the private key:

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

Add below command:

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

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

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

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.

Appendix B-3 Multiple devices use the same ssh public key

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

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: 
```

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
```

Below error may occur while adding ssh private key.

Agent admitted failure to sign using the key

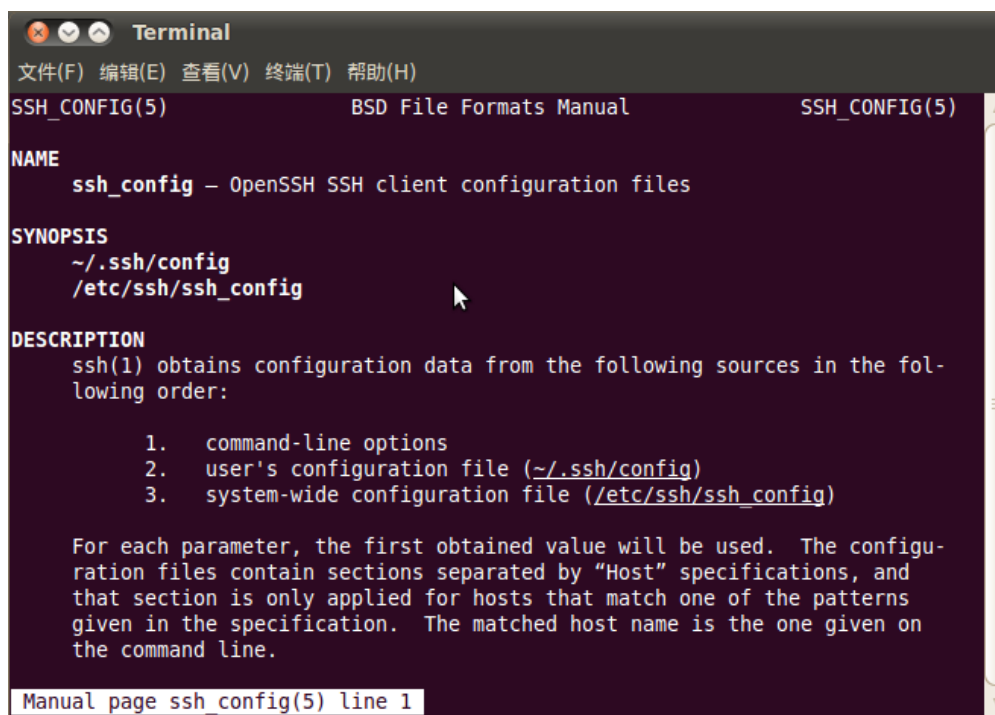
Input below command at console to fix it.

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

Appendix B-4 Switch different ssh public keys on one device

You can refer to ssh_config document to configure ssh.

```
~$ man ssh_config
```

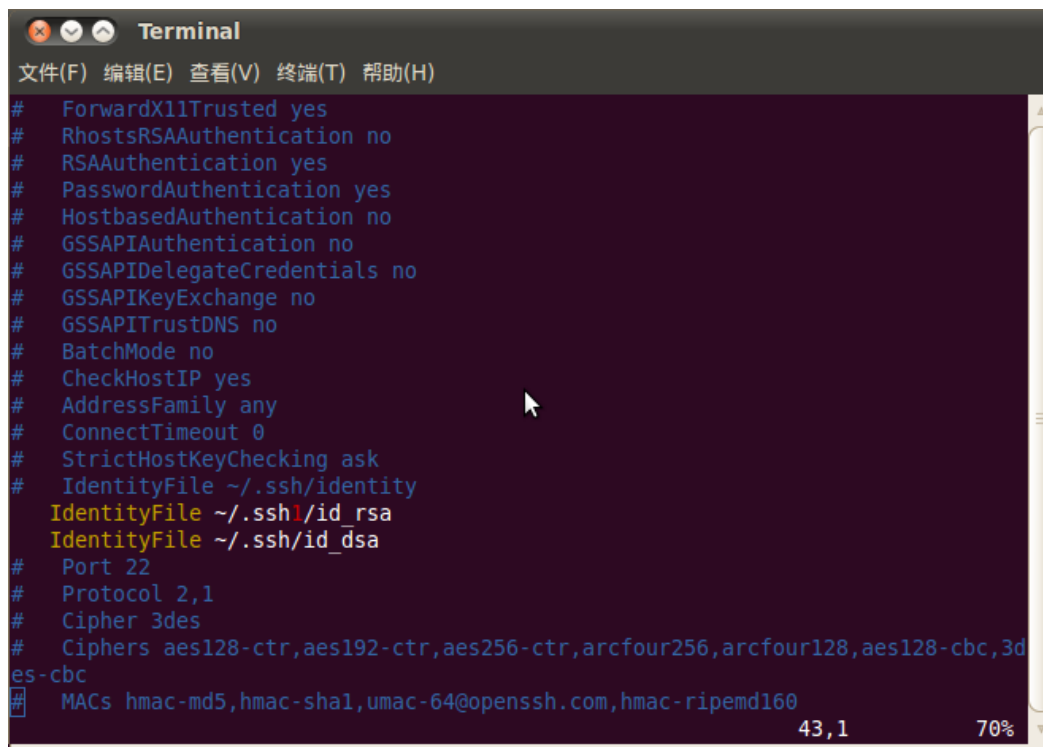


Use below commands to configure ssh for current user.

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

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


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 ~/.ssh1/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%

```

Appendix B-5 Private key authority management

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.

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

Appendix B-6 Git authority application instruction

Refer to above chapters, generate the public key file, and send email to fae@rock-chips.com applying for SDK code download authority.