

密级状态：绝密( ) 秘密( ) 内部资料( ) 公开( ☒ )

## **RK3399\_LINUX\_SDK\_V2.01\_20180521**

(技术部, 第三系统产品部)

文件状态：  [ ] 草稿  [ <input checked="" type="checkbox"/> ] 正式发布  [ ] 正在修改	当前版本：	V2.0
	作 者：	Caesar Wang
	完成日期：	2018-05-17
	审 核：	Eddie Cai
	完成日期：	2018-05-17

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)

## 文档修改记录

日期	修订版本	修订内容	修改人	核定人
2017-01-16	V1.0	初始版本	Guochun Huang	Yuanbin Lan
2017-02-27	V1.1	增加 linux pc 下载工具	Guochun Huang	Yuanbin Lan
2017-06-08	V1.2	U-boot release branch	Guochun Huang	Yuanbin Lan
2018-04-08	V1.3	Android 和 Linux 的 U-boot 合并	Caesar Wang	Eddie Cai
2018-04-11	V1.4	SDK 获取说明	Caesar Wang	Eddie Cai
2018-04-18	V1.5	修改一些错词和仓库地址更改	Caesar Wang	Eddie Cai
2018-05-17	v2.0	1. buildroot 和 debian 文档合二为一 2. 增加 ssh 公钥说明	Caesar Wang	Eddie Cai
2018-05-21	v2.01	修复编译 debian 的命令	Caesar Wang	Eddie Cai

# 目录

1 概述.....	4
2 主要支持功能.....	4
3 SDK 获取说明.....	4
4 SDK 编译说明.....	6
4.1 Uboot 编译.....	6
4.2 Kernel 编译步骤.....	7
4.3 Buildroot 及 App 编译.....	7
4.4 Debian rootfs 编译.....	8
4.4.1 Building base debian system by ubuntu-build-service from linaro.....	8
4.4.2 Building rk-debian rootfs.....	8
4.4.3 Creating the ext4 image(linaro-rootfs.img).....	8
5 刷机说明.....	9
5.1 Windows 刷机说明.....	9
5.2 Linux 刷机说明.....	10
5.3 系统分区说明.....	11
6 Secure CRT 的参数设置.....	12
7 RK3399_Linux 工程目录介绍.....	13
8 RK3399 SDK 固件.....	14
9 SSH 公钥操作说明.....	15
9.1 SSH 公钥生成.....	15
9.2 使用 key-chain 管理密钥.....	15
9.3 多台机器使用相同 SSH 公钥.....	16
9.4 一台机器切换不同 SSH 公钥.....	17
9.5 密钥权限管理.....	18
9.6 Git 权限申请说明.....	18

# 1 概述

本 SDK 是基于 Linux 系统，内核基于 kernel 4.4，适用于 RK3399 挖掘机以及基于其上所有 linux 产品开发。

本 SDK 支持 VPU 硬解码、GPU 3D、Wayland 显示、QT 等功能。具体功能调试和接口说明，请阅读工程目录 docs/下文档。

## 2 主要支持功能

功能	模块名
数据通信	Wi-Fi、以太网卡、USB、SDCARD
应用程序	图库、设置、视频、音频、视频播放

## 3 SDK 获取说明

SDK 通过瑞芯微代码服务器对外发布或者从 Github 开源网站上获取。其编译开发环境，参考第 4 节 SDK 编译说明。

### 获取 SDK 方法一: 从瑞芯微代码服务器获取源码

获取 RK3399 Linux 软件包，需要有一个帐户访问 Rockchip 提供的源代码仓库。客户向瑞芯微技术窗口申请 SDK，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[第 9 节 SSH 公钥操作说明](#)。

RK3399\_LINUX\_SDK 下载命令如下：

```
repo init --repo-url ssh://git@www.rockchip.com.cn/repo/rk/tools/repo -u  
ssh://git@www.rockchip.com.cn/linux/rk/platform/manifests -b linux -m rk3399_linux_release.xml
```

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

```
git clone ssh://git@www.rockchip.com.cn/repo/rk/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，开发者可以通过这种方式，获得 SDK 代码的初始压缩包，该压缩包解压得到的源码，与通过 repo 下载的源码是一致的。

以 rk3399\_linux\_sdk\_v2.0\_20180517.tgz 为例，拷贝到该初始化包后，通过如下命令可检出源码：

```
mkdir rk3399
tar xvf rk3399_linux_sdk_v2.0_20180517.tgz -C rk3399
cd rk3399
.repo/repo/repo sync -l
.repo/repo/repo sync
```

后续开发者可根据 Fae 窗口定期发布的更新说明，通过“.repo/repo/repo sync”命令同步更新。

### 获取 SDK 方法二：从 Github 开源网站获取

下载 repo 工具

```
git clone https://github.com/rockchip-linux/repo.git
```

建立 rk3399 linux 工作目录

```
mkdir rk3399_linux
```

进入 rk3399 linux 工作目录

```
cd rk3399_linux/
```

初始化 repo 仓库

```
../repo/repo init --repo-url=https://github.com/rockchip-linux/repo -u
```

```
https://github.com/rockchip-linux/manifests -m rk3399_linux_release.xml -b master
```

同步下载整个工程：

```
../repo/repo sync
```

## 4 SDK 编译说明

### Ubuntu 16.04 系统:

编译 **Buildroot** 环境搭建所依赖的软件包安装命令如下:

```
sudo apt-get install repo git-core gitk git-gui gcc-arm-linux-gnueabi-hf u-boot-tools device-tree-compiler gcc-aarch64-linux-gnu mtools parted libudev-dev libusb-1.0-0-dev python-linaro-image-tools linaro-image-tools autoconf autotools-dev libsigsegv2 m4 intltool libdrm-dev curl sed make binutils build-essential gcc g++ bash patch gzip bzip2 perl tar cpio python unzip rsync file bc wget libncurses5 libqt4-dev libglib2.0-dev libgtk2.0-dev libglade2-dev cvs git mercurial rsync openssh-client subversion asciidoc w3m dlatex graphviz python-matplotlib
```

编译 **Debian** 环境搭建所依赖的软件包安装命令如下:

```
sudo apt-get install repo git-core gitk git-gui gcc-arm-linux-gnueabi-hf u-boot-tools device-tree-compiler gcc-aarch64-linux-gnu mtools parted libudev-dev libusb-1.0-0-dev python-linaro-image-tools linaro-image-tools gcc-4.8-multilib-arm-linux-gnueabi-hf gcc-arm-linux-gnueabi-hf libssl-dev gcc-aarch64-linux-gnu g++ conf autotools-dev libsigsegv2 m4 intltool libdrm-dev curl sed make binutils build-essential gcc g++ bash patch gzip bzip2 perl tar cpio python unzip rsync file bc wget libncurses5 libqt4-dev libglib2.0-dev libgtk2.0-dev libglade2-dev cvs git mercurial rsync openssh-client subversion asciidoc w3m dlatex graphviz python-matplotlib
```

### Ubuntu 17.04 系统:

除了上面外还需如下依赖包:

```
sudo apt-get install lib32gcc-7-dev g++-7 libstdc++-7-dev
```

## 4.1 Uboot 编译

进入工程 u-boot 目录下执行 make.sh 来获取 rk3399\_loader\_v1.09.112.bin trust.img uboot.img:

rk3399 挖掘机开发板: `./make.sh evb-rk3399`

rk3399 Firefly 开发板: `./make.sh firefly-rk3399`

编译后生成文件在 u-boot 目录下:

```
u-boot/
├── rk3399_loader_v1.09.112.bin
├── trust.img
└── uboot.img
```

## 4.2 Kernel 编译步骤

进入工程目录根目录执行以下命令自动完成 kernel 的编译及打包：

rk3399 挖掘机开发板：

```
cd kernel  
make ARCH=arm64 rockchip_linux_defconfig  
make ARCH=arm64 rk3399-sapphire-excavator-linux.img -j12
```

rk3399 Firefly 开发板：

```
cd kernel  
make ARCH=arm64 rockchip_linux_defconfig  
make ARCH=arm64 rk3399-firefly-linux.img -j12
```

编译后在 kernel 目录生成 boot.img,这个 boot.img 就是包含 kernel 的 Image 和 DTB。

## 4.3 Buildroot 及 App 编译

```
cd buildroot && make rockchip_rk3399_defconfig && cd .. && ./build_all.sh  
&& ./mkfirmware.sh buildroot
```

在执行完命令./mkfirmware.sh buildroot 后，即会打包 rootfs.img 到 rockimg/目录下。

**备注：**

若需要编译单个模块或者第三方应用，需对交叉编译环境进行配置。

交叉编译工具位于 buildroot/output/host/usr 目录下，需要将工具的 bin/目录和 aarch64-rockchip-linux-gnueabi/bin/目录设为环境变量，在顶层目录执行自动配置环境变量的脚本（只对当前控制台有效）：

```
source envsetup.sh
```

输入命令查看：

```
aarch64-linux-gcc --version
```

此时会打印出以下 log 即标志为配置成功：

```
aarch64-linux-gcc.br_real (Buildroot 2016.08.1-00150-gc031b95) 5.4.0
```

系统编译：

执行“./build\_all.sh”，其会自动找到系统中的“rk\_make\_first.sh”和“rk\_make.sh”，然后执行编译命令。如果只需要编译单模块，可以进入到模块目录下，执行“rk\_make.sh”或“rk\_make\_first.sh”命令。如果需要增加应用模块,可以参照增加“rk\_make.sh”或“rk\_make\_first.sh”来编译自己的应用。

## 4.4 Debian rootfs 编译

先进入 rootfs/目录

```
cd rootfs/
```

### 4.4.1 Building base debian system by ubuntu-build-service from linaro

```
sudo apt-get install binfmt-support qemu-user-static live-build
```

```
sudo dpkg -i ubuntu-build-service/packages/*
```

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

```
ARCH=armhf ./mk-base-debian.sh
```

编译完成会在 rootfs/生成: linaro-stretch-alip-xxxxx-1.tar.gz (xxxxx 表示生成时间戳)。

### 4.4.2 Building rk-debian rootfs

```
ARCH=armhf ./mk-rootfs.sh 或 VERSION=debug ARCH=armhf ./mk-rootfs-stretch.sh
```

(开发阶段推荐使用后面带 debug)。

### 4.4.3 Creating the ext4 image(linaro-rootfs.img)

```
./mk-image.sh
```

此时会生成 rootfs/linaro-rootfs.img.

回到工程根目录, 打包完整固件. 运行 `./mkfirmware.sh debian` 生成所有固件在 `rockimg/`目录下。



## 5 刷机说明

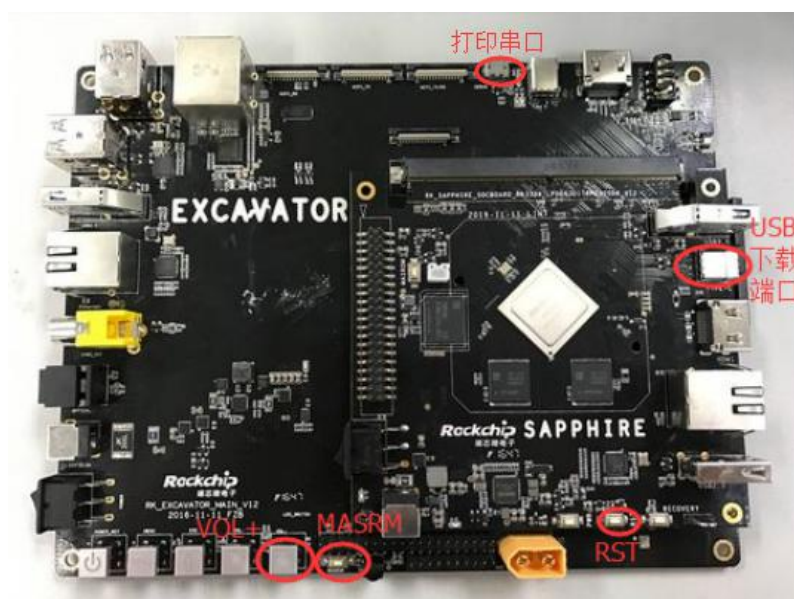


图 1 RK3399 挖掘机

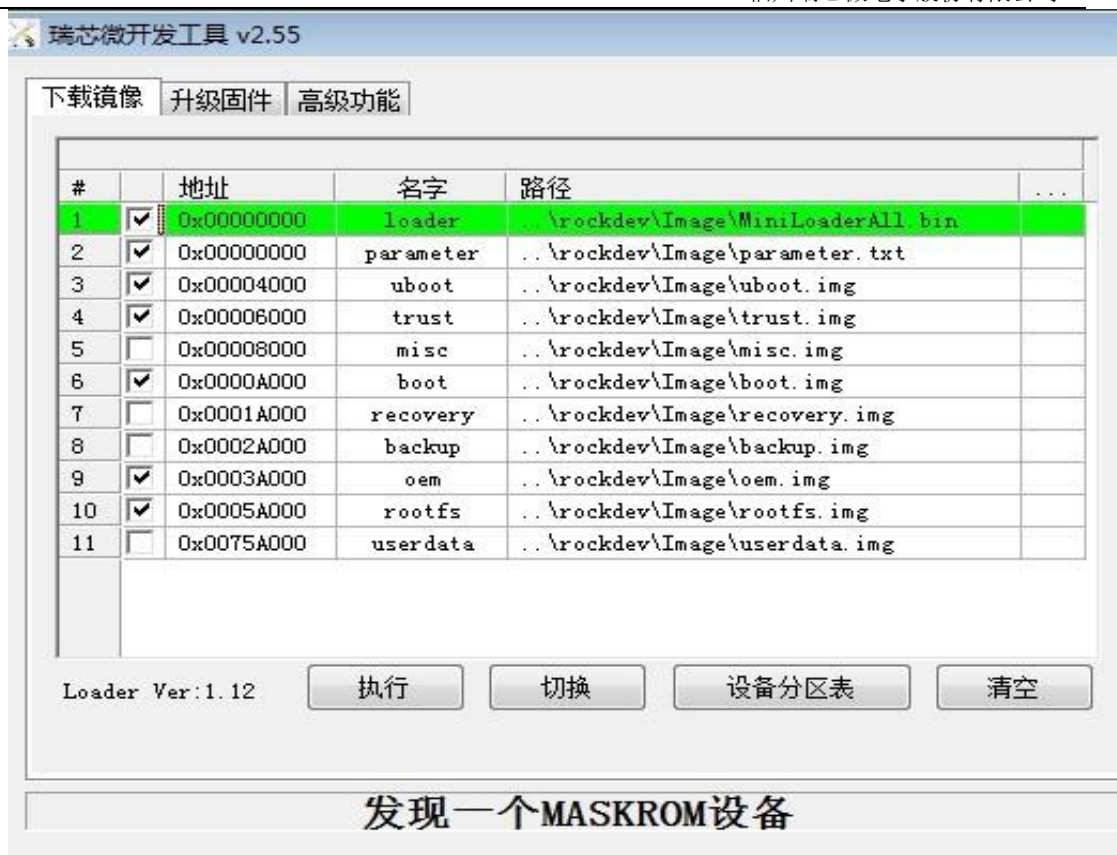
### 5.1 Windows 刷机说明

SDK 提供 Windows 烧写工具(工具版本需要 V2.55 或以上), 工具位于工程根目录:

tools/

└─ windows/AndroidTool

如下图, 编译生成相应的固件后, 设备烧写需要进入 MASKROM 烧写模式, 连接好 usb 下载线后, 按住按键“MSROM”不放并按下复位键“RST”后松手, 就能进入 MASKROM 模式, 加载编译生成固件的相应路径后, 点击“执行”进行烧写, 也可以按“recovery”按键不放并按下复位键“RST”后松手进入 loader 模式进行烧写, 下面是 MASKROM 模式的 分区偏移及烧写文件。(Note: WIndow PC 需要在管理员权限运行工具才可执行)

图 2 烧写工具 **AndroidTool.exe**

注：烧写前，需安装最新 USB 驱动，驱动详见：

[tools/windows/DriverAssitant\\_v4.6.zip](#)

## 5.2 Linux 刷机说明

Linux 下的烧写工具位于 `tools/linux` 目录下(Linux\_Upgrade\_Tool 工具版本需要 **V1.33** 或以上)，请确认你的板子连接到 `maskrom/loader rockusb`。比如编译生成的固件在 `rockimg` 目录下，升级命令如下：

```
sudo ./upgrade_tool ul      rockimg/MiniLoaderAll.bin
sudo ./upgrade_tool di -p   rockimg/parameter.txt
sudo ./upgrade_tool di -u   rockimg/uboot.img
sudo ./upgrade_tool di -t   rockimg/trust.img
sudo ./upgrade_tool di -b   rockimg/boot.img
sudo ./upgrade_tool di -oem rockimg/oem.img
sudo ./upgrade_tool di -rootfs rockimg/rootfs.img
sudo ./upgrade_tool rd
```

## 5.3 系统分区说明

**默认分区说明 (下面是挖掘机分区参考):**

Number	Start (sector)	End (sector)	Size	Code	Name
1	16384	24575	4096K	0700	uboot
2	24576	32767	4096K	0700	trust
3	32768	40959	4096K	0700	misc
4	40960	106495	32.0M	0700	boot
5	106496	172031	32.0M	0700	recovery
6	172032	237567	32.0M	0700	backup
7	237568	368639	64.0M	0700	oem
8	368640	3514367	3584M	0700	rootfs
9	7708672	15269854	3691M	0700	userdata

uboot 分区: 烧写 uboot 编译出来的 uboot.img.

trust 分区: 烧写 uboot 编译出来的 trust.img.

misc 分区: 烧写 misc.img。给 recovery 使用.

boot 分区: 烧写 kernel 编译出来的 boot.img.

recovery 分区: 烧写 recovery.img.

backup 分区: 预留，暂时没有用。后续跟 android 一样作为 recovery 的 backup 使用

oem 分区: 给厂家使用，存放厂家的 app 或数据。只读。代替原来音箱的 data 分区。挂载在/oem 目录.

rootfs 分区: 存放 buildroot 或者 debian 编出来的 rootfs.img,只读.

userdata 分区:存放 app 临时生成的文件或者是给最终用户使用。可读写，挂载在/userdata 目录下.

注意:此版本 misc/recovery/backup 分区都先预留. recovery 机制还没完全开启，功能还在完善中.

## 6 Secure CRT 的参数设置

利用 Secure CRT 软件打印调试信息 log，需要对串口参数进行设置，具体设置细节如下图：

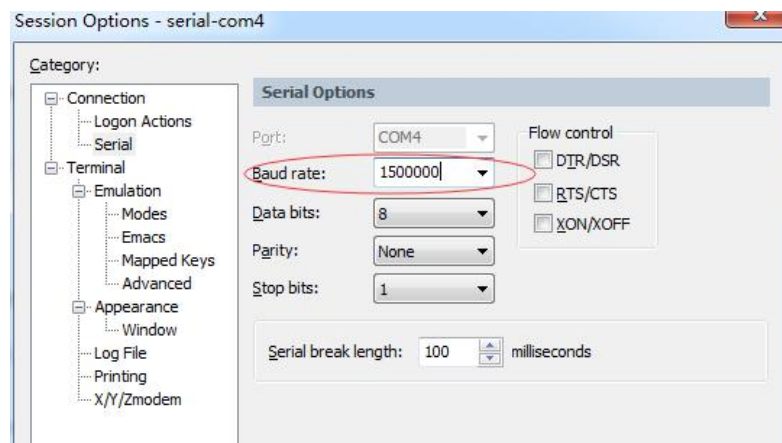


图 3 Secure CRT 参数设置

## 7 RK3399\_Linux 工程目录介绍

进工程目录下有 buildroot、buildroot、recovery、app、kernel、u-boot、device、docs、external 等目录。每个目录或其子目录会对应一个 git 工程，提交需要在各自的目录下进行。

- 1) app: 存放上层应用 app，主要是 Carmachin 和一些测试应用程序。
- 2) buildroot: 定制根文件系统
- 3) device/rockchip/rk3399: 存放开机初始化脚本，存放第三方库、bin、alsa/wifi 等配置文件；另还存放编译脚本，系统根目录的几个 sh 脚本都是在 repo sync 的时候，从这里拷贝出来的，所以若要提交修改的脚本，必须在 device/rockchip/rk3399 目录下进行。

build\_all.sh：编译所有第三方库和应用。

mkfirmware.sh: 打包最终烧写的固件。

envsetup.sh: 终端环境变量设置。

- 4) docs: 存放工程帮助文件。
- 5) external: 相关库，包括音频、视频、网络等。
- 6) kernel: kernel 代码。
- 7) prebuilts: 存放交叉编译工具链。
- 8) recovery: 存放 recovery 工程文件。
- 9) rkbin: 存放固件和工具。
- 10) rockimg: 存放编译输出固件
- 11) rootfs: Debian 根文件系统
- 12) tools: 存放一些常用工具。
- 13) u-boot: uboot 代码。

## 8 RK3399 SDK 固件

RK3399 EVB 和 Firefly 发布的 V2.0\_20180517 的固件如下[链接地址](#)下载  
(包含 Debian 和 Buildroot 的固件):.

<ftp://ftp.rock-chips.com>

user: linux\_rk3399

psw: odvAzX0qXe

[链接地址](#)包含如下内容:

RK3399 挖掘机(EVB):

RK3399\_Evb\_Buildroot\_V2.0\_20180517.zip 和 RK3399\_Evb\_Debian\_V2.0\_20180517.zip

RK3399 Firefly:

RK3399\_Firefly\_Buildroot\_V2.0\_20180517.zip 和 RK3399\_Firefly\_Debian\_V2.0\_20180517.zip

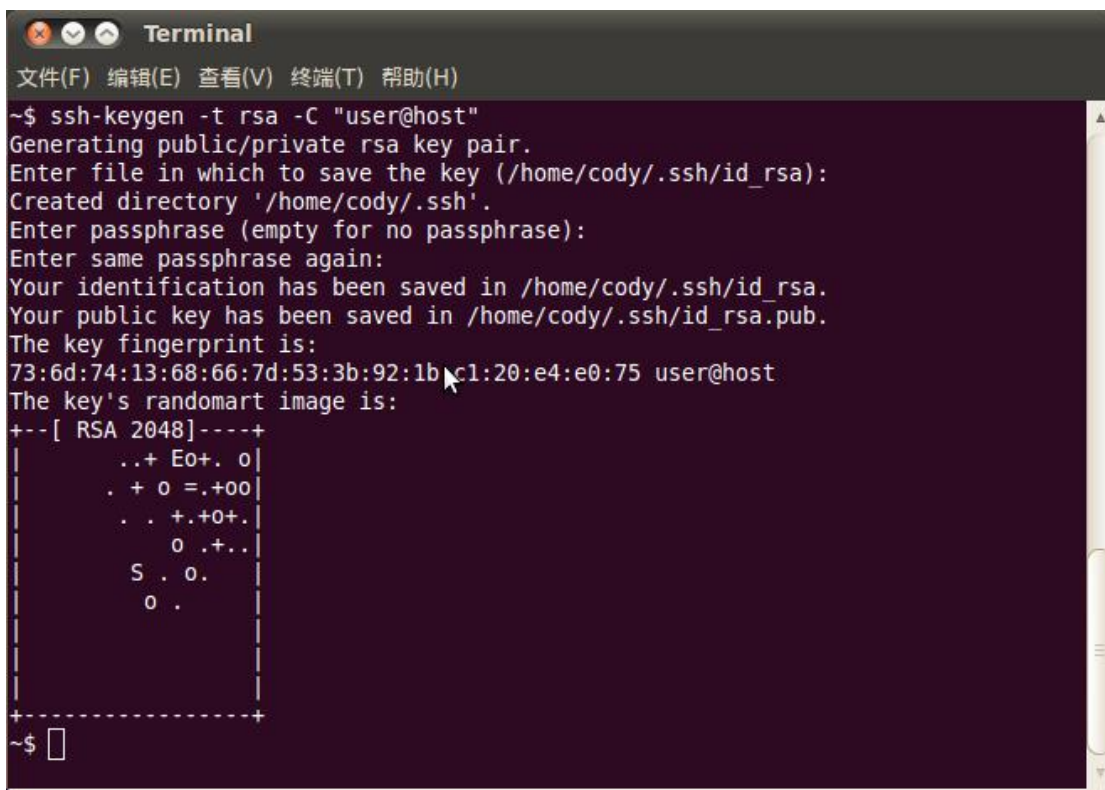
## 9 SSH 公钥操作说明

### 9.1 SSH 公钥生成

使用如下命令生成：

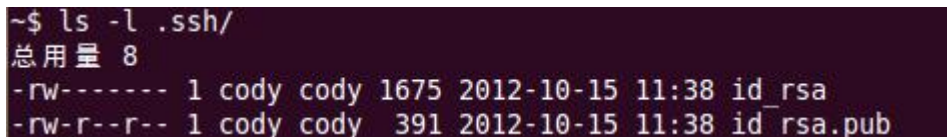
```
ssh-keygen -t rsa -C "user@host"
```

请将 **user@host** 替换成您的邮箱地址。



```
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。



```
~$ 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。

### 9.2 使用 key-chain 管理密钥

推荐您使用比较简易的工具 keychain 管理密钥。

具体使用方法如下：



### 1. 安装 keychain 软件包:

```
$sudo aptitude install keychain
```

### 2. 配置使用密钥:

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

增加下面这行:

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

其中, id\_rsa 是私钥文件名称。

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。

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。

## 9.3 多台机器使用相同 SSH 公钥

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。

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。

```
~/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。

```
~$ 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。

```
Agent admitted failure to sign using the key
```

在 console 输入如下命令即可解决。

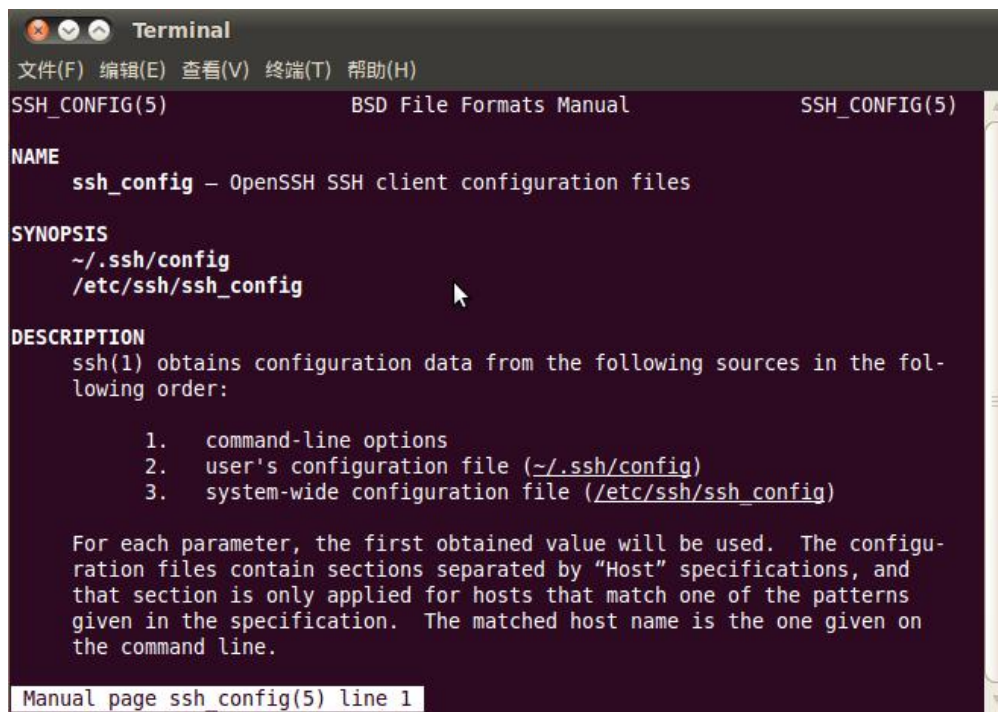
```
ssh-add ~/.ssh/id_rsa
```



## 9.4 一台机器切换不同 SSH 公钥

可以参考 ssh\_config 文档配置 SSH。

```
~$ man ssh_config
```

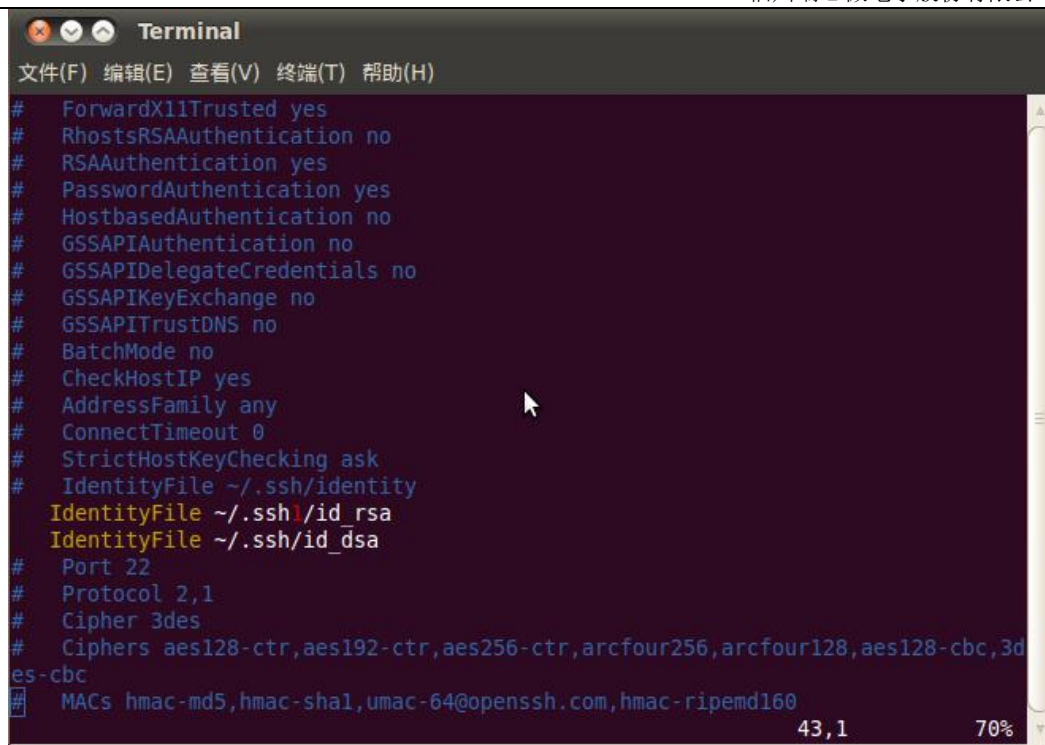


通过如下命令，配置当前用户的 SSH 配置。

```
~$ 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。

A terminal window titled "Terminal" with a menu bar containing "文件(F)", "编辑(E)", "查看(V)", "终端(T)", and "帮助(H)". The terminal displays a list of SSH configuration options, each preceded by a hash symbol (#). The options and their values are: 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, and MACs hmac-md5,hmac-sha1,umac-64@openssh.com,hmac-ripemd160. The terminal has a dark background and a light-colored scrollbar on the right. The bottom right corner shows "43,1" and "70%".

```
# 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
```

## 9.5 密钥权限管理

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。

请妥善保管私钥文件。并不要二次授权与第三方使用。

## 9.6 Git 权限申请说明

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