# 破冰手机云勘大师DC-5500C技术预研报告

目录

[破冰手机云勘大师DC-5500C技术预研报告 1](#_Toc5851)

[1. 文档介绍 2](#_Toc29905)

[1.1 编写目的 2](#_Toc26587)

[1.2 文档范围 2](#_Toc17513)

[1.3 读者对象 2](#_Toc14447)

[1.4 参考资料 3](#_Toc32575)

[2. 背景介绍 3](#_Toc16982)

[3. 预研目标 3](#_Toc29158)

[4. 预研成果 3](#_Toc31337)

[4.1 C++开发环境 3](#_Toc22075)

[4.2 Python开发环境 3](#_Toc9199)

[a. 源码安装包下载 3](#_Toc1892)

[b. 编译安装 4](#_Toc5700)

[c. 配置默认python环境 4](#_Toc3100)

[4.3 基础依赖库 4](#_Toc6304)

[4.4 第三方工具 6](#_Toc26782)

[4.5 编译命令备忘 6](#_Toc6386)

## 文档介绍

### 1.1 编写目的

提前调研发现国产化研发过程中将会遇到的问题，并将解决方案进行总结整理，以帮助后期研发人员能更快更顺畅完成研发工作。

### 1.2 文档范围

1 对产品研发过程中设计到的相关技术进行预先研究。

2 当业务需求需要变更时追加可能涉及的技术研究成果。

3 本文档当前版本主要阐述破冰版手机研发大师DC-5500C于国产化平台及国产银河麒麟操作系统上研发过程中的一些技术点。

### 1.3 读者对象

本项目的设计、开发人员。

### 1.4 参考资料

CMake官网文档： https://cmake.org/cmake/help/v3.5/

Boost官网：https://www.boost.org/doc/libs/1\_35\_0/libs/python/doc/index.html

Redis官网：<https://redis.io/>

[linux下VscodeC++环境配置](https://www.cnblogs.com/ange-plus-plus/p/11647648.html)：<https://www.cnblogs.com/ange-plus-plus/p/11647648.html>

Linux下aria2的安装与配置：https://www.cnblogs.com/LiQingsong/p/10293142.html

## 背景介绍

破冰手机云勘大师DC-5500C是手机云勘察大师DC-5500国产化版本，需要于国产机器及国产系统银河麒麟ARM64上确保和DC-5500的基本功能，并能稳定运行。初版云取证仅需支持微信和支付宝的云数据提取。

## 预研目标

在国产化装备银河麒麟系统上完成DC-5500C所设计的基础库与工具的编译、安装、测试等工作，并确保其能顺利运行云取证相关功能。

## 预研成果

预先调研过程均基于国产化平台和银河麒麟操作系统完成，以下预研成果仅适用预研平台。

### C++开发环境

由于银河麒麟自带G++开发环境,无需特别安装，为提升开发效率此处选用安装并配置VSCode用以C++代码的编写和调试。

#### VSCode的安装配置

详情请参考 [linux下VscodeC++环境配置](https://www.cnblogs.com/ange-plus-plus/p/11647648.html) ：

https://www.cnblogs.com/ange-plus-plus/p/11647648.html

### 4.2 Python开发环境

#### Python的安装

源码安装包下载

sudo apt-get http://xxxx.Python-3.6.8.tgz

tar -zxvf Python-3.6.1.tgz

cd Python-3.6.1/

编译安装

# ./configure --prefix=/usr/local/ --enable-shared CFLAGS=-fPIC

# prefix= 编译的时候用来指定程序存放路径

./configure --prefix=/usr/local/python3 --enable-shared CFLAGS=-fPIC

make

make altinstall

配置默认python环境

cp libpython3.6m.so.1.0 /usr/local/lib64/

cp libpython3.6m.so.1.0 /usr/lib/

cp libpython3.6m.so.1.0 /usr/lib64/

cd /usr/bin

mv python python.backup

ln -s /usr/local/bin/python3.6 /usr/bin/python

ln -s /usr/local/bin/python3.6 /usr/bin/python3

rm -rf /usr/bin/python2

ln -s /usr/bin/python2.7 /usr/bin/python2

#### Pycharm的安装配置

详情参考：[Linux系统下PyCharm的安装教程及激活教程！ - 值南针 - 博客园 (cnblogs.com)](https://www.cnblogs.com/honeynan/p/12313790.html)

### 4.3 基础依赖库

#### Boost库的编译

**下载源码包**

<https://www.boost.org/users/download/>

此处选择 boost\_1\_69\_0

**编译安装**

解压boost安装包，并进入到首层目录。

tar -zxvf boost\_1\_69\_0.tar.gz

执行配置，选择完全安装。此处要指定python3.6的位置，不同的linux系统有所不同，可以通过命令：which python3来查找路径。

./bootstrap.sh --with-libraries=all --with-python=/usr/local/bin/python3.6m

配置project-config.jam，编辑python相关目录，执行：

vim project-config.jam

打开文件后编辑 using python 字段，如下：

using python : 3.6 : /usr/bin/python3.6m : /usr/include/python3.6m : /usr/lib/python3.6 ;

sudo ./b2

#sudo ./b2 install --with-python include="/usr/include/python3.6m" --prefix=/home/zjw/boost

sudo ./b2 install include="/usr/local/include/python3.6m" --prefix=/home/zjw/boost

**测试**

#include <string>

#include <boost/python.hpp>

using namespace std;

using namespace boost::python;

struct World{

void set(string msg) { this->msg = msg; }

string greet() { return msg; }

string msg;

};

//特别注意下面的模块名hello同将来引入Python的模块名、编译完成的文件名，三者必须相同

BOOST\_PYTHON\_MODULE(hello){

class\_<World>("World")

.def("greet", &World::greet)

.def("set", &World::set)

;

}

#### Log4cplus

tar -zxvf log4cplus-1.2.1.tar.gz

# 设置安装目录 --enable-static 生成静态库

./configure --prefix=/user/local/log4cplus --enable-static

make

make install

#### OpenSSL的编译

**源码下载**

去https://www.openssl.org网站下载openssl源码 或用git命令从github上下载 git clone <https://github.com/openssl/openssl.git>

**解压**

tar -xzf openssl-1.1.0c.tar.gz

**进入解压目录，设置Openssl 安装路径，( --prefix )参数为欲安装之目录，执行如下命令：**

./config --prefix=/usr/local/openssl

执行命令./config -t

执行make命令，编译Openssl，编译需要等待一定的时间。

执行make install，安装 Openssl，安装也需要一定的时间。

安装完成后,openssl 会被安装到/usr/local/openssl 目录，包括头文件目录

进入lib后，libssl.a和libcrypto.a则为编译后的库文件

mv openssl-1.1.1.tar.gz ./openssl

gunzip openssl-1.1.1.tar.gz

tar -xvf openssl-1.1.1.tar

./config --prefix=/usr/local/openssl

make

make test

make install

cd /usr/local/openssl

ls

### 4.4 第三方工具

#### Redis

**下载源码包**

<http://download.redis.io/releases/redis-4.0.10.tar.gz>

**解压**

tar -zxvf redis-4.0.10.tar.gz

**进入redis源码，直接可以编译且安装**

make && make install

#### Aria2

**下载源码包**

[https://github.com/aria2/aria2/releases](https://github.com/aria2/aria2/releases" \t "https://www.cnblogs.com/LiQingsong/p/_blank)

**解压并开始编译安装**

tar zxvf aria2-1.33.0.tar.gz && cd aria2-1.33.0 && mkdir build && cd build && ../configure && make && make install

**创建目录与配置文件**

# 这一步需要切换到root下进行

# 需要 su 一下

cd /etc/ && mkdir aria2 && cd aria2 && touch aria2c.conf && touch aria2.session

### 4.5 编译命令备忘

#### CMake

CMakeLists文件编写语法规则详解：

https://blog.csdn.net/zhanghm1995/article/details/80902807