DevKit插件使用实验手册

版本:1.0



华为技术有限公司

前言

## 简介

本书为编译插件和鲲鹏开发框架插件使用介绍，旨在帮助开发者快速掌握编译插件和鲲鹏开发插件的使用。

## 内容描述

本实验指导书主要内容如下：

编译插件的使用

鲲鹏开发框架插件的使用

环境说明

本实验环境面向搭建服务器版实验环境。实验环境包括openEuler操作系统、鲲鹏云服务器。

设备介绍

为了满足实验需要，建议每套实验环境采用以下配置

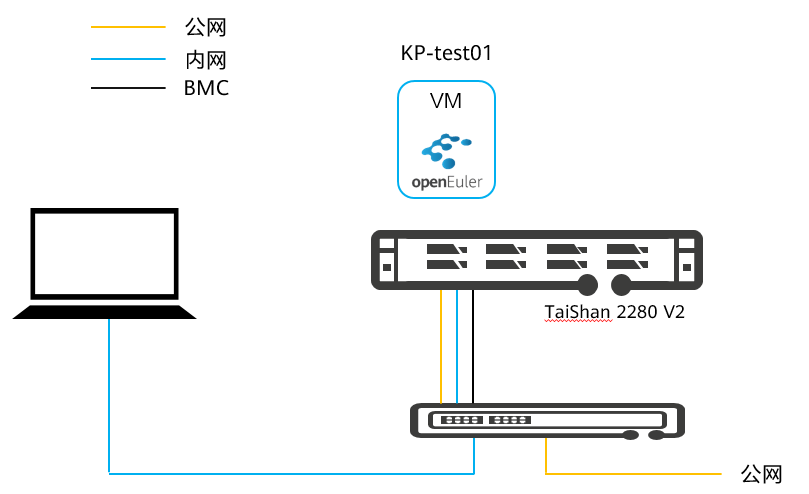
|  |  |  |  |
| --- | --- | --- | --- |
|  | 云主机名称 | 配置 | OS版本 |
| 鲲鹏云服务器 | kp-test01 | kc1.large.2 2vCPUs | 4GB | openEuler 20.03 LTS |

软件介绍

本实验用到的工具如下表，请提前准备好相关工具。

|  |  |
| --- | --- |
| 软件名称 | 使用说明 |
| putty | https://www.chiark.greenend.org.uk/~sgtatham/putty/latest.html |
| winScp | https://winscp.net/eng/download.php |

实验环境拓扑



openEuler基础操作实验环境 - 服务器版

提示：实验所需虚拟机均已配置完成，请按照授课老师安排登录各自的虚拟机完成实验操作。

目录

[前言 1](#_Toc93410794)

[简介 1](#_Toc93410795)

[内容描述 1](#_Toc93410796)

[1 DevKit插件使用实验手册 4](#_Toc93410797)

[1.1.1 关于本实验 4](#_Toc93410798)

[1.1.2 实验目的 4](#_Toc93410799)

[1.1.3 实验规划 4](#_Toc93410800)

[1.2 开发环境搭建 4](#_Toc93410801)

[1.2.1 购买云服务器 4](#_Toc93410802)

[1.2.2 环境登录验证 10](#_Toc93410803)

[1.2.3 安装vscode 12](#_Toc93410804)

[1.2.4 安装鲲鹏编译插件与开发框架插件 12](#_Toc93410805)

[1.3 编译插件IDE的使用 14](#_Toc93410806)

[1.3.1 实验介绍 14](#_Toc93410807)

[1.3.2 插件使用步骤 14](#_Toc93410808)

[1.3.3 实战操作 30](#_Toc93410809)

[1.4 鲲鹏开发框架插件IDE使用 30](#_Toc93410810)

[1.4.1 实验介绍 30](#_Toc93410811)

[1.4.2 插件使用步骤 33](#_Toc93410812)

[1.5 实验环境清理 46](#_Toc93410813)

[1.6 附录 47](#_Toc93410814)

[1.6.1 修改和删除目标服务器 47](#_Toc93410815)

[1.6.2 使用毕昇编译器， 48](#_Toc93410816)

[1.6.3 安装毕昇JDK 50](#_Toc93410817)

[1.6.4 鲲鹏gcc支持的操作系统 52](#_Toc93410818)

# DevKit插件使用实验手册

### 关于本实验

本实验介绍编译插件IDE和鲲鹏开发框架IDE的安装和使用。

### 实验目的

熟练使用编译插件IDE；

熟练使用鲲鹏开发框架IDE。

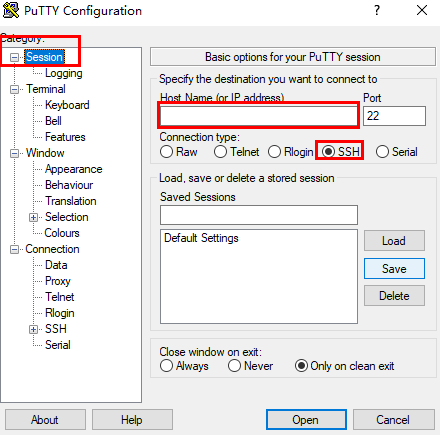
### 实验规划

本实验需要用到一台鲲鹏架构下装有openEuler操作系统的虚拟机，鲲鹏平台虚拟机openEuler配置为2vCPU | 4G RAM | 40G ROM ,还需要在本地机器安装vscode，以及插件安装包。

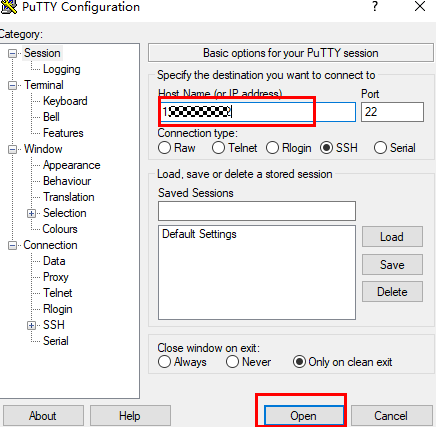
## 开发环境搭建

### 环境登录验证

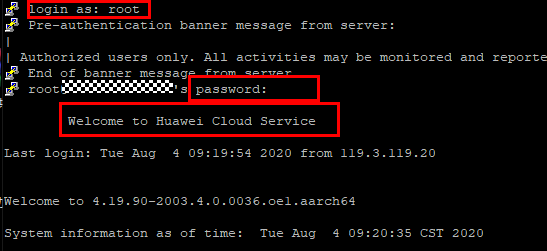
打开电脑上putty 工具，点击红色方框内的图标新建会话。



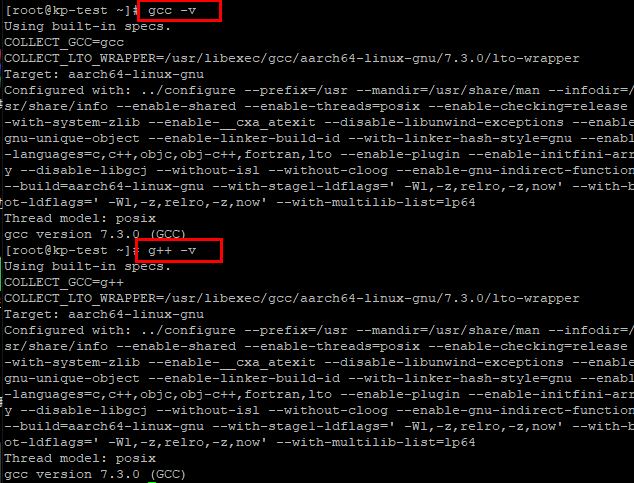
如图，红框填写为申请的弹性公网ip，点击open按钮。



进行身份验证设置，在login as输入用户名root，图中password输入购买ECS时设置的密码，点击确定, 出现Welcome to Huawei Cloud Service表示链接成功。



检查编译器是否安装，openEuler自带版本为gcc7.3.0，满足实验要求。输入命令gcc –v ;g++ -v。



### 安装vscode

安装地址：<https://code.visualstudio.com/>

Window: Dowmload for Window

Mac: Dowmload for Mac

Linux: Dowmload for Linux

### 安装鲲鹏编译插件与开发框架插件

鲲鹏开发套件是基于Visual Studio Code的一款扩展工具，编译插件是其中的一个子工具。编译插件即插即用，支持一键安装鲲鹏GCC编译器，以及鲲鹏平台远程调试能力，支持的功能特性如下：

* 一键式部署

支持从Visual Studio Marketplace下载并在线安装插件，同时支持一键部署服务端鲲鹏GCC编译器

* 编译调试

− 一键式安装鲲鹏GCC

− 可视化编译配置任务，一键式任务运行

− 远程单步调试C/C++代码

− 编译调试过程信息实时展示

− gtest框架用例树渲染及状态展示

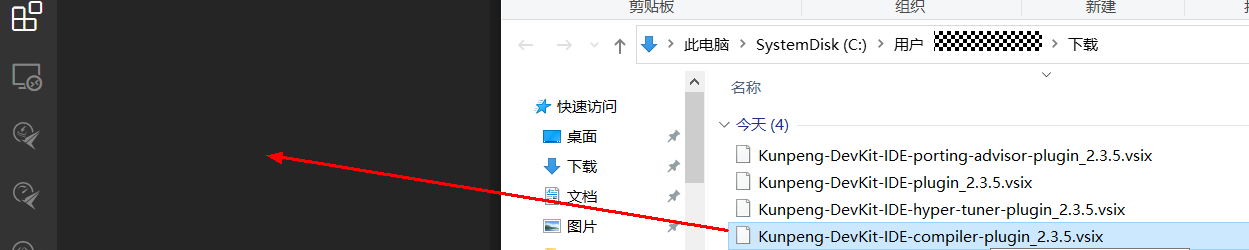
当前已在如下服务端操作系统上验证：

* CentOS 7.6
* openEuler 20.03（LTS）（鲲鹏GCC编译器）

鲲鹏编译插件安装方式如下：

打开Visual Studio Code扩展菜单，根据地址下载至本地后，拖入扩展菜单即可。

下载地址：https://obs-oa-zyl.obs.cn-north-4.myhuaweicloud.com/Kunpeng-DevKit-IDE-compiler-plugin\_2.3.5.vsix



打开本地PC上的Visual Studio Code，在左侧菜单栏中单击扩展图标。安装完成后，在左侧菜单栏中单击打开鲲鹏编译插件。

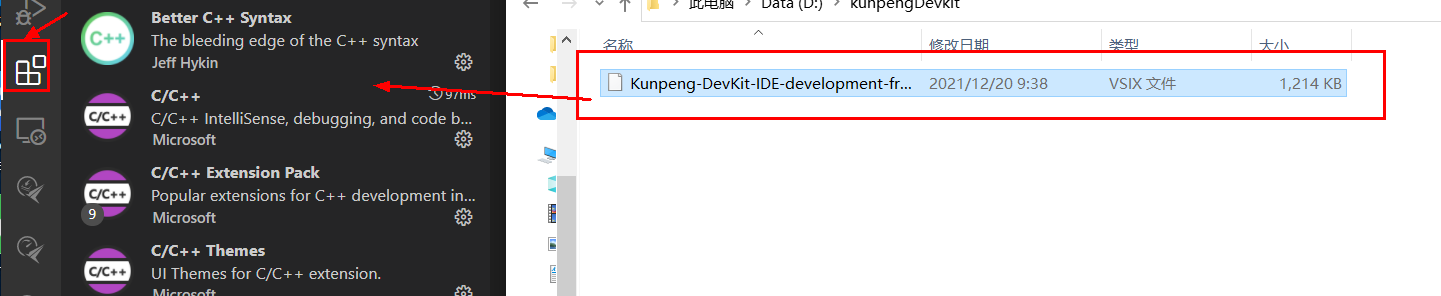
安装鲲鹏开发框架插件

鲲鹏开发套件是基于Visual Studio Code的一款扩展工具，鲲鹏开发框架插件是其中的一个子工具。鲲鹏开发框架插件即插即用，能够扫描代码文件中可使用鲲鹏加速库优化后的函数或汇编指令，生成可视化报告；编码时能够自动匹配鲲鹏加速库函数字典，智能提示、高亮、联想字典中可以替换的库和函数。关于鲲鹏开发框架的详细介绍请参见[鲲鹏开发框架](https://kunpeng.huawei.com/" \l "/developer/devkit/library" \o " )。

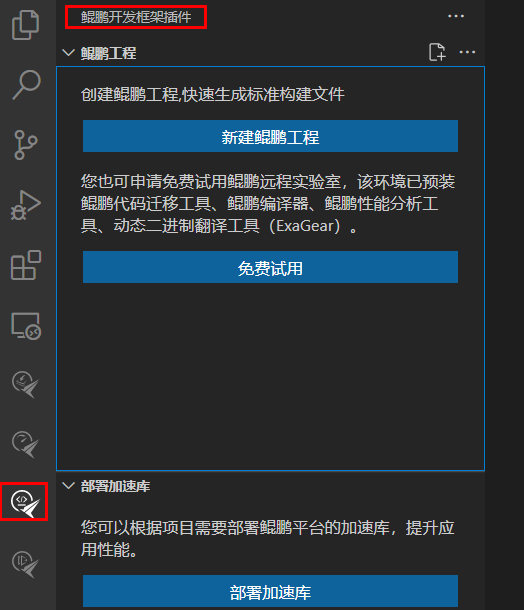
鲲鹏开发框架插件安装步骤如下：

下载链接地址：https://obs-oa-zyl.obs.cn-north-4.myhuaweicloud.com/Kunpeng-DevKit-IDE-development-framework-plugin\_2.3.5.vsix

打开Visual Studio Code扩展菜单，把下载好”Kunpeng-DevKit-IDE-development-framework-plugin\_2.3.5.vsix”工具拖入扩展区即可。



打开Visual Studio Code，在左侧菜单栏中显示鲲鹏开发框架插件。



## 编译插件IDE的使用

### 实验介绍

本章主要内容是以案例驱动，实践操作鲲鹏编译插件。

### 插件使用步

在桌面上新建一个文件夹codeC作为文件。最后把这一步，放在步骤一，一般是先开发文件夹，在配置服务器

选择vscode文件 –> 打开文件夹（选择codeC）

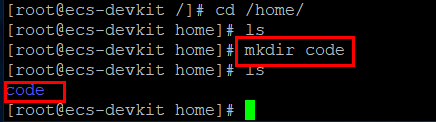
登录服务器创建存放代码目录，参照使用putty登录服务器

cd /

cd home

mkdir code

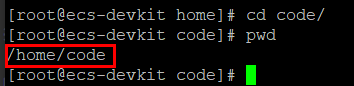
ls code



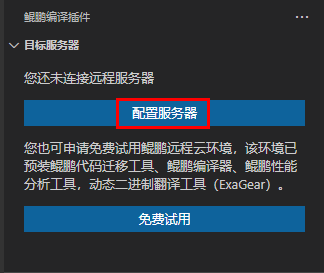
cd code

pwd

记录路径 /home/code

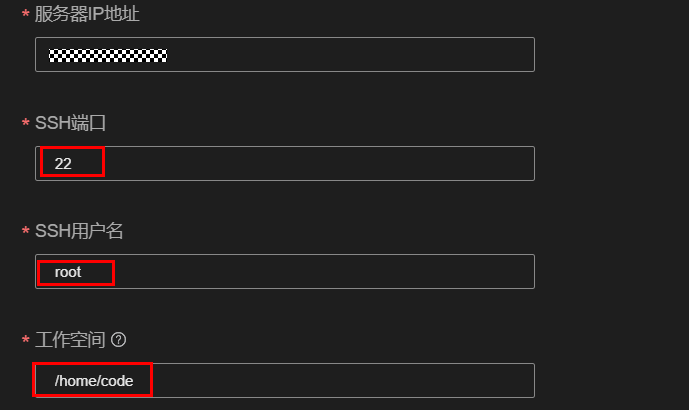


单击“目标服务器”模块中的“配置服务器”按钮，如图5-1所示。也可以单击，选择“配置远端服务器”，打开“添加目标服务器”页签。

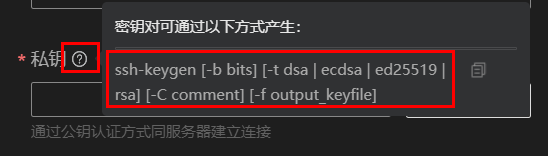


|  |  |
| --- | --- |
| 参数 | 说明 |
| 服务器IP地址 | 待连接的目标服务器IP地址。 |
| SSH端口 | 待连接的目标服务器SSH端口。 |
| SSH用户名 | 待连接的目标服务器SSH用户名。 |
| 工作空间 | 远程编译调试时，服务器上同步代码的存放路径。 |
| 如没在服务器配置公钥，是否需要工具自动上传 | 是  否  说明  选择“是”，则需要指定公钥文件及SSH密码。选择“否”，则需要确保服务器端已配置公钥。 |
| SSH密码 | 输入目标服务器的SSH密码。 |
| 公钥 | 导入id\_rsa公钥文件。  说明  密钥对可通过以下方式产生：ssh-keygen [-b bits] [-t dsa | ecdsa | ed25519 | rsa] [-C comment] [-f output\_keyfile]。 |
| 私钥 | 导入id\_rsa私钥文件。  说明  密钥对可通过以下方式产生：ssh-keygen [-b bits] [-t dsa | ecdsa | ed25519 | rsa] [-C comment] [-f output\_keyfile]。  可单击“私钥”后的问号，并在弹出的气泡提示中单击复制图标复制产生密钥对的命令。 |

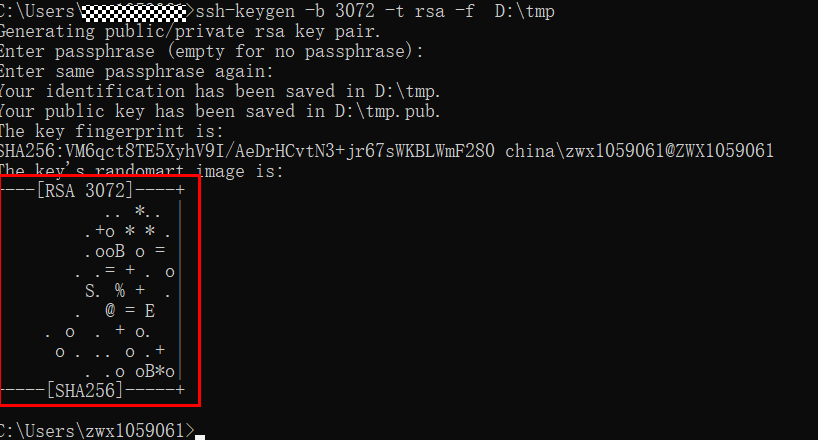
填写参数如下



生成私钥，点击问号的，



复制ssh-keygen命令至windos终端，回车



生成私钥与公钥会存放在D盘目录中，如下。

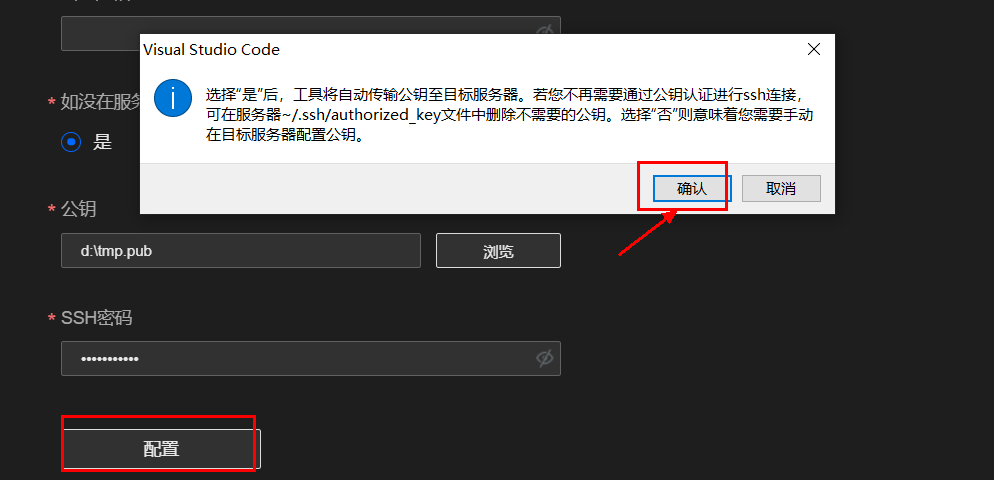


点击标签浏览，选择D盘中tmp文件，密码短语不填写即可，工具点击“是”为上传。公钥选择D盘中生成的公钥tmp.pub





单击“配置”



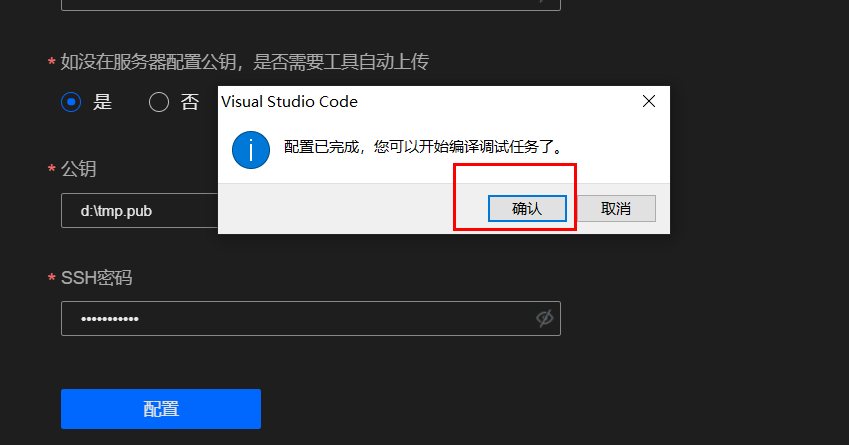
单击“确认”。

弹出消息，提示主机可信度不能确定。



单击“是”。

弹出提示框，提示“配置已完成，您可以开始编译调试任务了。”

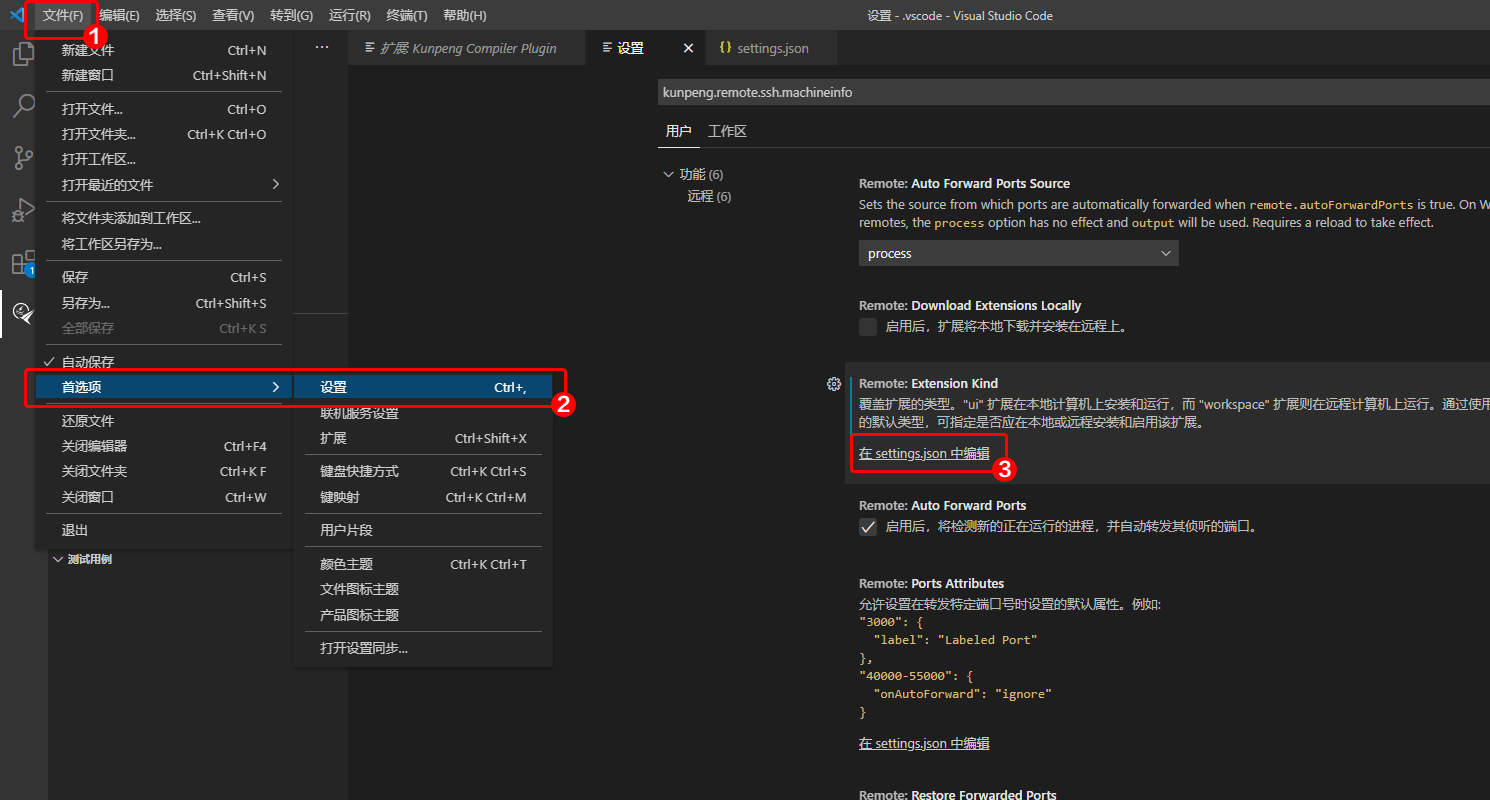


单击“确认”。

设置远程环境（setting.json）-可选

生成settings.json文件。

方法一：在VSCode上选择“文件 > 首选项 > 设置”，再选择“工作区”，并在输入框搜索“kunpeng.remote.ssh.machineinfo”, 单击“在settings.json中编辑”后生成settings.json文件。如图所示。



settings.json 配置说明

"kunpeng.remote.ssh.machineinfo": [

{

"label": "xxx",/\*服务器名称\*/

"ip": "xxx.xxx.xxx.xxx",/\*目标服务器的IP\*/

"port": "xxx",/\*目标服务器IP对应的端口\*/

"user": "xxx", /\*登录目标服务器的用户名\*/

"workspace": "xxx"/\*同步代码到服务器的路径\*/

},

{

//可同时配置的多台服务器

"label": "xxx",

"ip": "xxx.xxx.xxx.xxx",

"port": "xxx",

"user": "xxx", /

"workspace": "xxx"

}

]

代码案例

需在vscode中新建一个文件，编写以下代码，并命名成sinvalue.c

求正弦值，

#include <stdio.h>

#include <math.h>

#define LEN 100000

int main()

{

//数据初始化

double src[LEN] = {0};

double dst1[LEN] = {0};

double dst2[LEN] = {0};

for (int i = 0; i < LEN; i++) {

src[i] = i;

}

long t;

//使用系统数学库对向量中每个元素求正弦值

for (int i = 0; i < LEN; i++) {

dst1[i] = sin(src[i]);

}

for (int i = 0; i < LEN; i++) {

printf( "正弦值%d:",dst1[i]);

}

return 0;

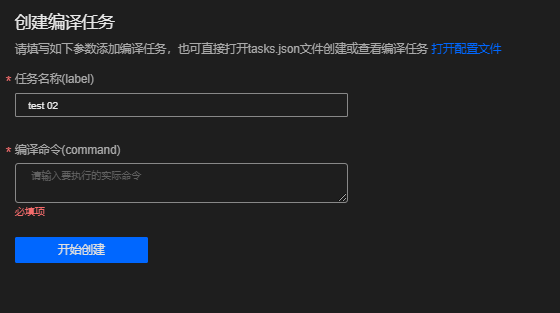
}

配置编译任务（tasks.json）

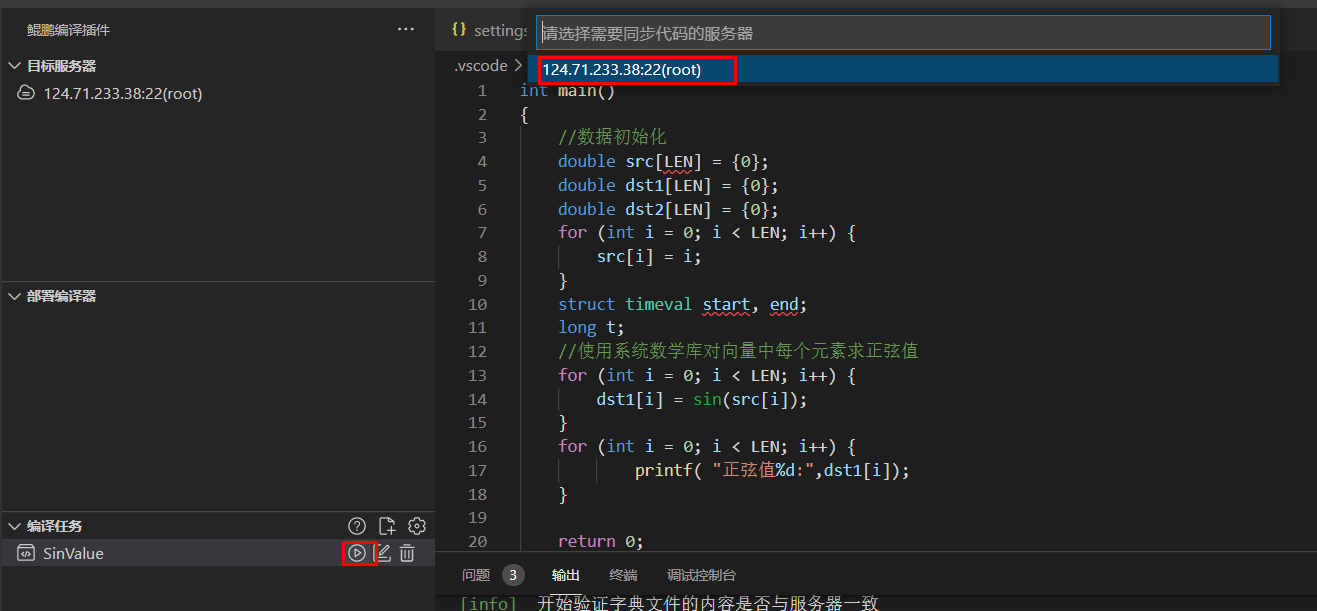
在“编译任务”区域，单击 ，打开“创建编译任务”页签。填写任务参数，并单击“开始创建”。如图所示。

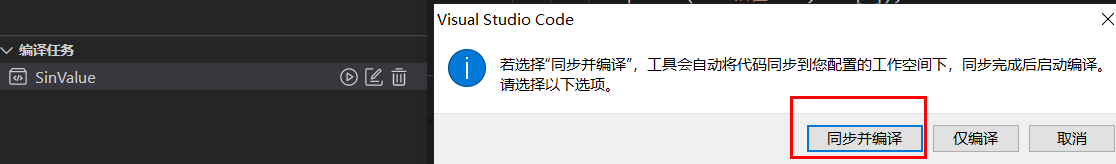
任务参数为：SinValue (根据自己工程情况实际编写)

编译命令: gcc sinvalue.c -o sinvalue -g -lm (根据自己工程情况实际编写)

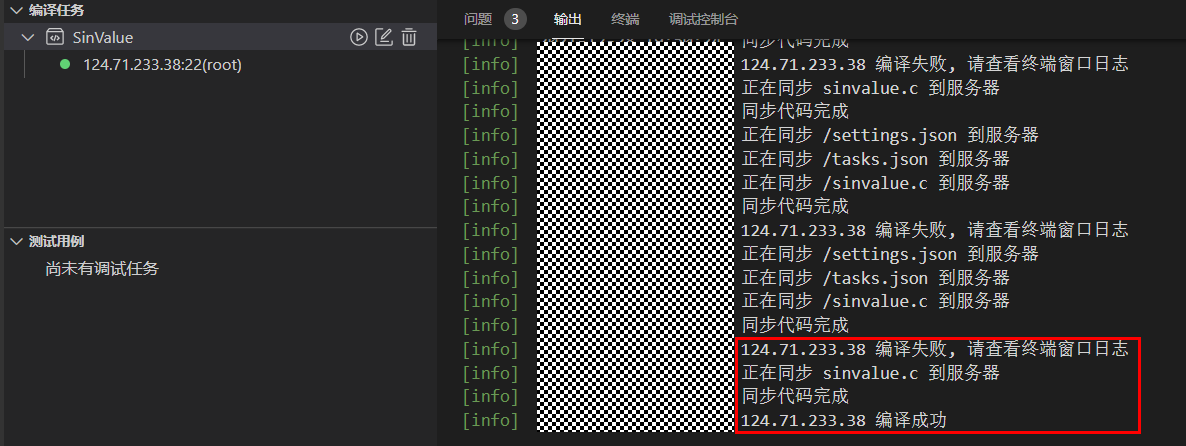


创建成功以后，并开始编译任务，编译会把代码同步至服务器。





显示成功

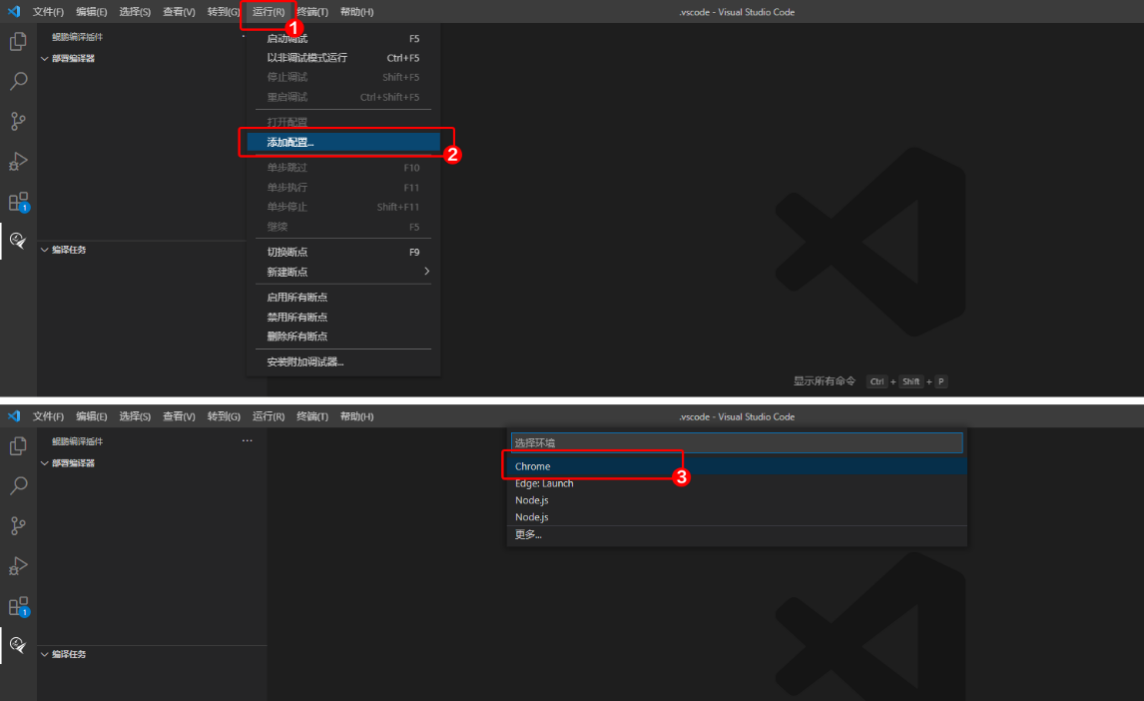


配置测试用例，本次采用方法三添加测试用例

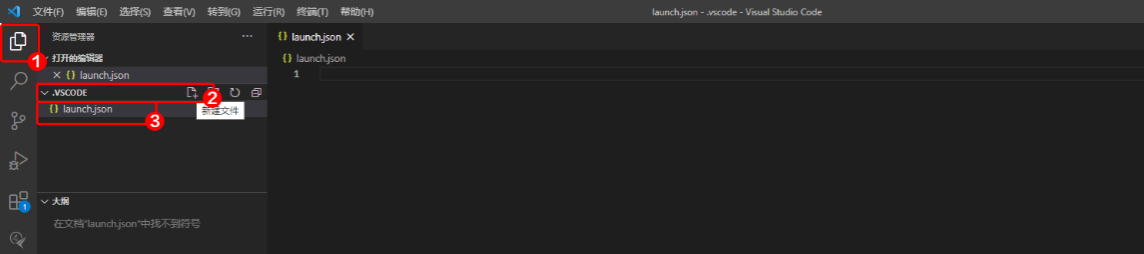
除上述快速获取测试用例以外，还提供以下三种方法。

生成launch.json文件。

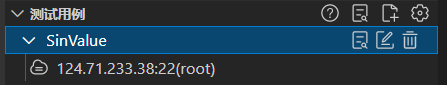
方法一：当vscode项目中没有launch.json文件时，选择“运行 > 添加配置 > Chrome”，生成launch.json文件。



方法二：在VSCode上选择“资源管理器”，找到“.vscode”文件夹，单击右键选择“新建文件”， 输入文件名“launch.json”生成launch.json文件。如图所示。



方法三：编译任务下方，“测试用例”区域，单击 ，打开“添加测试用例”页签。填写任务参数，并单击“开始开始添加”，后获取测试用例。 



配置测试用例信息。

复制以下代码至新创建的launch.json文件中，修改以下代码中的测试用例名称、需要测试的二进制文件名称、需要测试的二进制文件的绝对路径和测试需要依赖的前置编译任务名称，如图所示。

{

// 使用 IntelliSense 了解相关属性。

// 悬停以查看现有属性的描述。

// 欲了解更多信息，请访问: https://go.microsoft.com/fwlink/?linkid=830387

"version": "0.2.0",

"configurations": [

{

"name": "xxx", /\*测试用例名称\*/

"type": "cppdbg",

"request": "launch",

"program": "./swm\_llt", /\*需要测试的二进制文件名称\*/

"args": [],

"stopAtEntry": true,

"cwd": "${command:remotessh.remoteworkspace}/xxx/xxx/xxx/swm\_llt",/\*需要测试的二进制文件的绝对路径\*/ 把/xxx/xxx/xxx/swm\_llt去掉，不然路径会重复

"externalConsole": true,

"MIMode": "gdb",

"pipeTransport": {

"pipeCwd": "",

"pipeProgram": "${command:remotessh.sshclientpath}",

"pipeArgs": [

"-p",

"${command:remotessh.remoteport}",

"${command:remotessh.remoteuser}@${command:remotessh.remoteip}"

],

"debuggerPath": "/usr/bin/gdb"

},

"sourceFileMap": {

"${command:remotessh.remoteworkspace}": "${workspaceRoot}"

},

"setupCommands": [

{

"description": "Enable pretty-printing for gdb",

"text": "-enable-pretty-printing",

"ignoreFailures": true

},

{

"text": "handle SIGPIPE nostop noprint pass",

"description": "ignore SIGPIPE",

"ignoreFailures": true

}

],

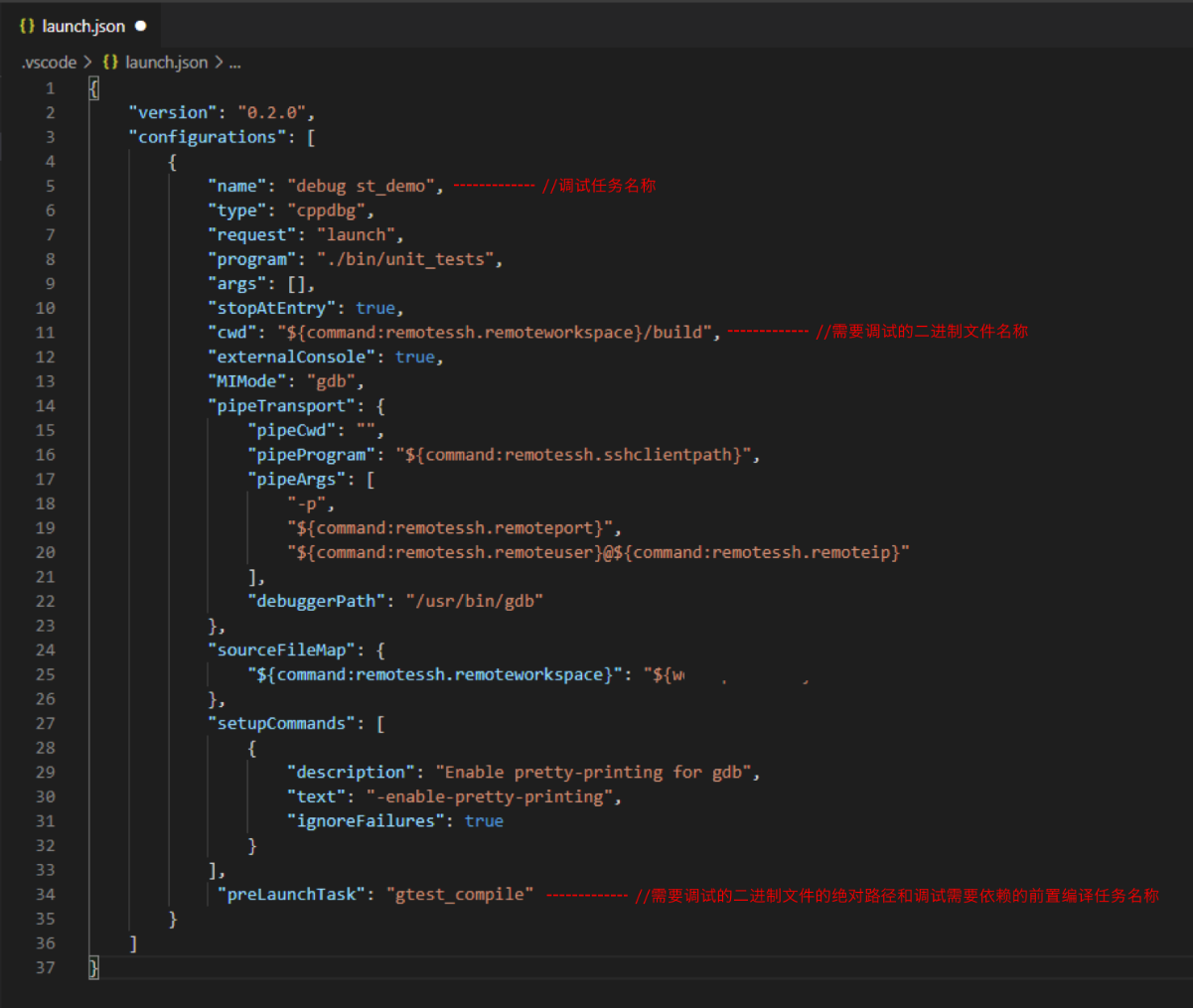
"preLaunchTask": "xxx" /\*测试需要依赖的前置编译任务名称\*/

}

]

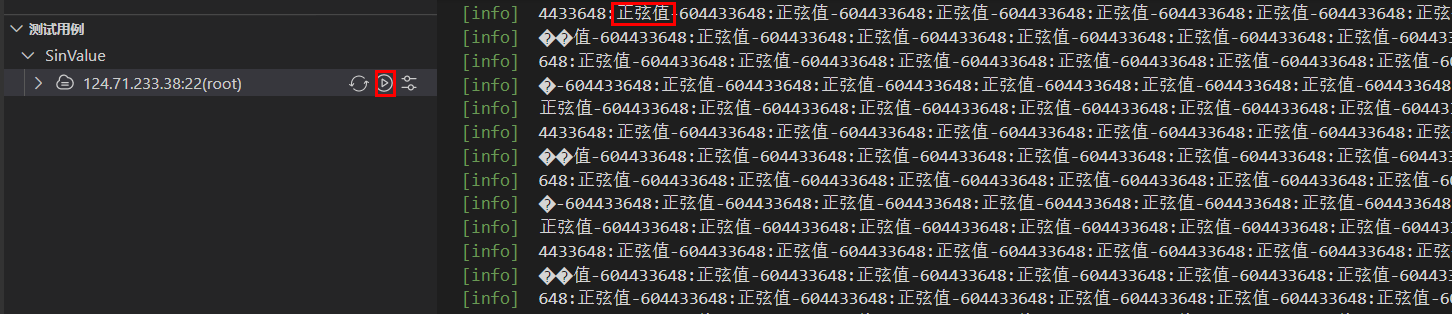
}

配置样例如下：



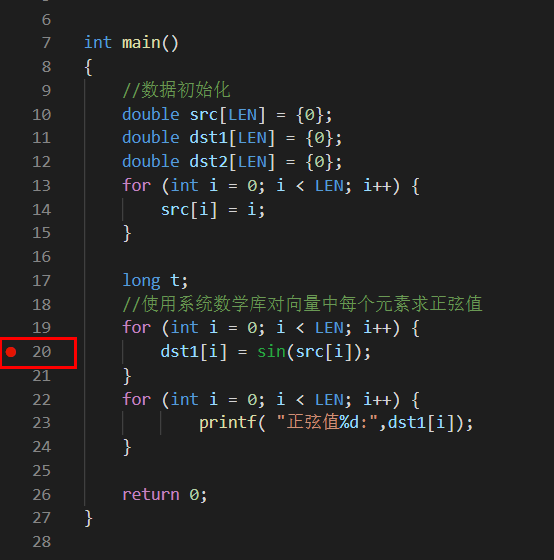
运行测试用列

点击运行，成功如下



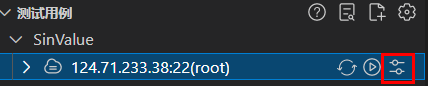
启动调试任务

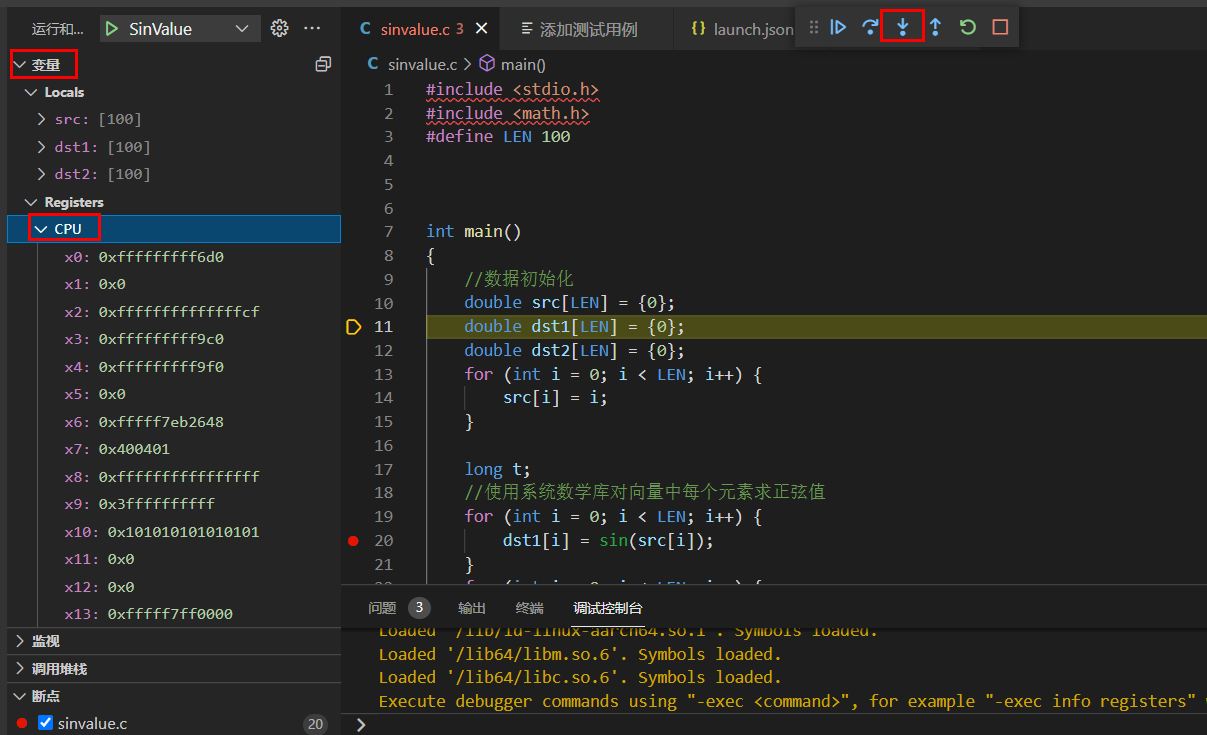
根据具体情况在工程中打断点。



选择（Debug All）开启断点调试前，

需安装gdb,使用yum -y install gdb

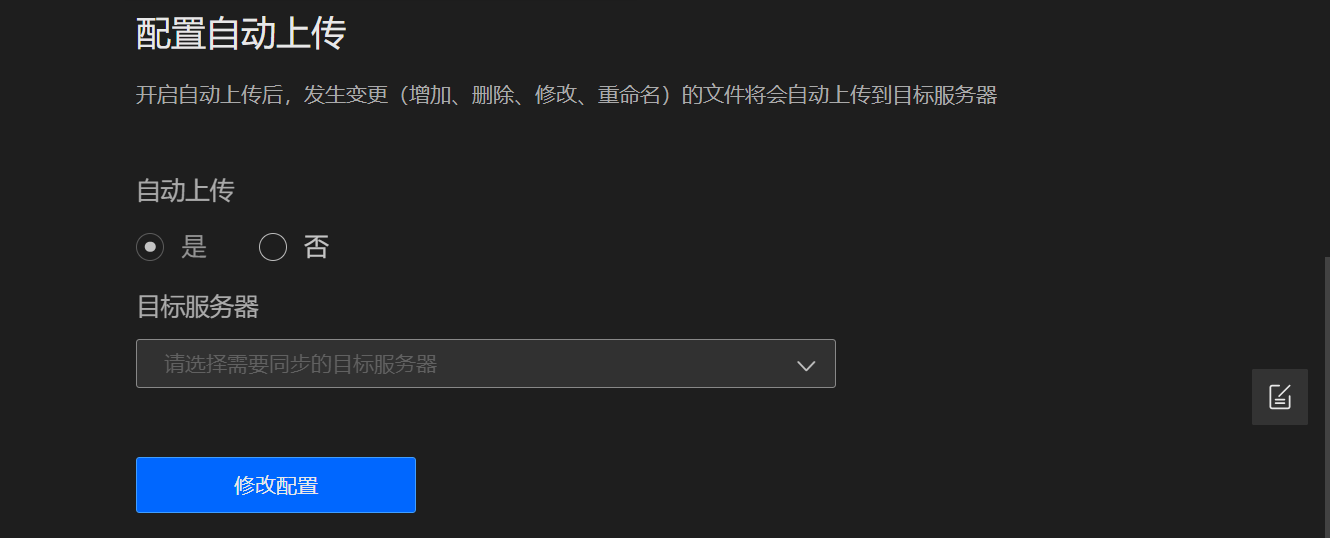




自动同步

在“目标服务器”区域，单击，选择“设置自动同步功能”。

打开“设置自动同步”页签，配置自动上传如所示。



单击“修改设置”。

单击“是”。

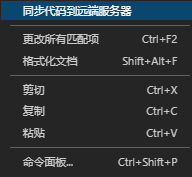
界面增加目标服务器下拉选择框。

选择目标服务器。

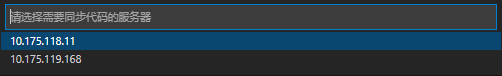
单击“保存设置”。

手动同步

在编译界面上单击右键，然后选择“同步代码到远端服务器”，



选择目标服务器



### 实战操作

使用编译插件编写一个小程序，编译通过，并能成功运行。

## 鲲鹏开发框架插件IDE使用

### 实验介绍

鲲鹏开发框架插件即插即用，能够扫描代码文件中可使用鲲鹏加速库优化后的函数或汇编指令，生成可视化报告；编码时能够自动匹配鲲鹏加速库函数字典，智能提示、高亮、联想字典中可以替换的库和函数。

目前，鲲鹏开发框架插件支持以下加速库：

|  |  |  |
| --- | --- | --- |
| 类别 | 名称 | 说明 |
| 系统库 | glibc-patch | 对内存、字符串、锁等接口基于华为鲲鹏920处理器微架构特点进行了加速优化 |
| hyperscan | 基于鲲鹏微架构优势，使用鲲鹏指令加速正则表达式的编译、扫描性能 |
| AvxToNeon | AvxToNeon是一款接口集合库。使用鲲鹏加速指令对传统平台定义的gcc Intrinsic内建函数进行适配，以支持使用了传统平台Intrinsic接口的应用能平滑迁移到鲲鹏平台 |
| 压缩 | gzip | 基于gzip-1.10，通过数据预取、循环展开、CRC指令替换等方法，来提升其在鲲鹏计算平台上的压缩和解压缩速率，尤其对文本类型文件的压缩及解压具有更明显的性能优势 |
| zstd | 基于zstd-1.4.4，通过使用NEON指令、内联汇编、代码结构调整、内存预取、指令流水线排布优化等方法，实现zstd在鲲鹏计算平台上压缩和解压性能的提升 |
| snappy | KAEzip是鲲鹏加速引擎的压缩模块，使用鲲鹏硬加速模块实现deflate算法，结合无损用户态驱动框架，提供高性能gzip/zlib格式压缩接口 |
| KAEzip | KAEzip是鲲鹏加速引擎的压缩模块，使用鲲鹏硬加速模块实现deflate算法，结合无损用户态驱动框架，提供高性能gzip/zlib格式压缩接口 |
| 加解密 | KAECrypto | 使用鲲鹏硬加速模块实现RSA/SM3/SM4/DH/MD5/AES算法，结合无损用户态驱动框架，提供高性能对称加解密、非对称加解密算法能力，兼容openssl1.1.1a及其之后版本，支持同步&异步机制 |
| 媒体 | x265-patch | 使用鲲鹏加速指令对x265开源库的转码核心代码进行重构，在保证原有接口及算法不变情况下，在鲲鹏平台下能获得较好的转码性能。针对ffmpeg视频转码场景，对x265的转码底层算子使用鲲鹏向量指令进行加速优化，提高转码性能 |
| HW265 | HW265视频编码器是符合H.265/HEVC视频编码标准、基于鲲鹏处理器NEON指令加速的华为自研H.265视频编码器。HW265支持四个预设编码档位可选，对应不同编码速度的应用场景，码率控制支持平均比特率模式（ABR）和恒定QP模式（CQP），功能涵盖直播、点播等各个场景，整体性能优于目前的主流开源软件 |
| HMPP | 鲲鹏超媒体性能库HMPP（Hyper Media Performance Primitives）包括向量缓冲区的分配与释放、向量初始化、向量数学运算与统计学运算、向量采样与向量变换、滤波函数、变换函数（快速傅里叶变换），支持IEEE 754浮点数运算标准，支持鲲鹏平台下使用 |
| X264 | X264是采用GPL授权的视频编码免费软件，主要功能实现H.264/MPEG-4 AVC的视频编码 |
| 数据库 | KML\_FFT | KML\_FFT基于鲲鹏架构，通过向量化、算法改进，对快速离散傅里叶变换进行了深度优化，使得快速傅里叶变换接口函数的性能有大幅度提升 |
| KML\_BLAS | KML\_BLAS基于鲲鹏架构，通过向量化、数据预取、编译优化、数据重排等手段，对BLAS的计算效率进行了深度挖掘，使得BLAS接口函数的性能逼近理论峰值 |
| KML\_SPBLAS | KML\_SPBLAS基于鲲鹏架构，充分利用鲲鹏的指令集和架构特点，开发了高性能稀疏矩阵运算库，提升HPC和大数据解决方案业务性能 |
| KML\_MATH | KML\_MATH通过周期函数规约、算法改进等手段，提供了基于鲲鹏处理器性能提升较大的函数实现 |
| KML\_VML | KML\_VML通过NEON指令优化、内联汇编等方法，对输入数据进行向量化处理，充分利用了鲲鹏架构下的寄存器特点，实现了在鲲鹏处理器上的性能提升 |
| KML\_LAPACK | KML\_LAPACK通过分块、求解算法组合、多线程、BLAS接口优化等手段，基于鲲鹏架构对LAPACK的计算效率进行了优化，实现了在鲲鹏处理器上的性能提升 |
| KML\_SVML | KML\_SVML通过Neon指令优化、内联汇编等方法，对输入向量进行批量处理，充分利用了鲲鹏架构下的寄存器特点，实现了在鲲鹏服务器上的性能提升 |
| 存储 | Smart Prefetch | Smart Prefetch（智能预取），创新性地采用高速缓存盘配合高效的预取算法，提升系统存储IO性能，进而提升上述解决方案中存储IO密集型场景的整体性能。 |
| SPDK | SPDK全称Storage Performance Development Kit（高性能存储开发包），SPDK的目标是通过使用网络技术、处理技术和存储技术来提升效率和性能。通过运行为硬件设计的软件，SPDK已经证明很容易达到每秒钟数百万次IO读取，通过使用许多处理器核心和许多NVMe驱动去存储，而不需要额外卸载硬件。 |
| ISA-L | ISA-L全称Intelligent Storage Acceleration Library，是提供RAID、纠删码、循环冗余检查、密码散列和压缩的高度优化的函数。 |
| 网络 | XPF | XPF（Extensible Packet Framework）加速库是鲲鹏自研加速库，XPF自研功能模块，在OVS（Open vSwitch）软件内部实现了一个智能卸载引擎模块 |
| DPDK | DPDK全称Data Plane Development Kit，为用户空间高效的数据包处理提供数据平面开发工具集，包括库函数和驱动 |

鲲鹏开发框架插件支持的功能特性如下：

* 智能联想

Coding时自动联想鲲鹏加速库优化后的相关函数

* 函数搜索

支持鲲鹏加速库函数的代码定义跳转、函数搜索

* 语法高亮

Coding时高亮鲲鹏加速库优化后的相关函数

* 加速分析

支持工程和文件扫描，识别出可以用鲲鹏加速库替换的函数

* 字典管理

支持加速库函数字典管理，可线上（自动）和线下更新

* 自动化构建工程

只需要在新建鲲鹏工程页面进行简单的输入和选择，便可以实现自动化构建工程

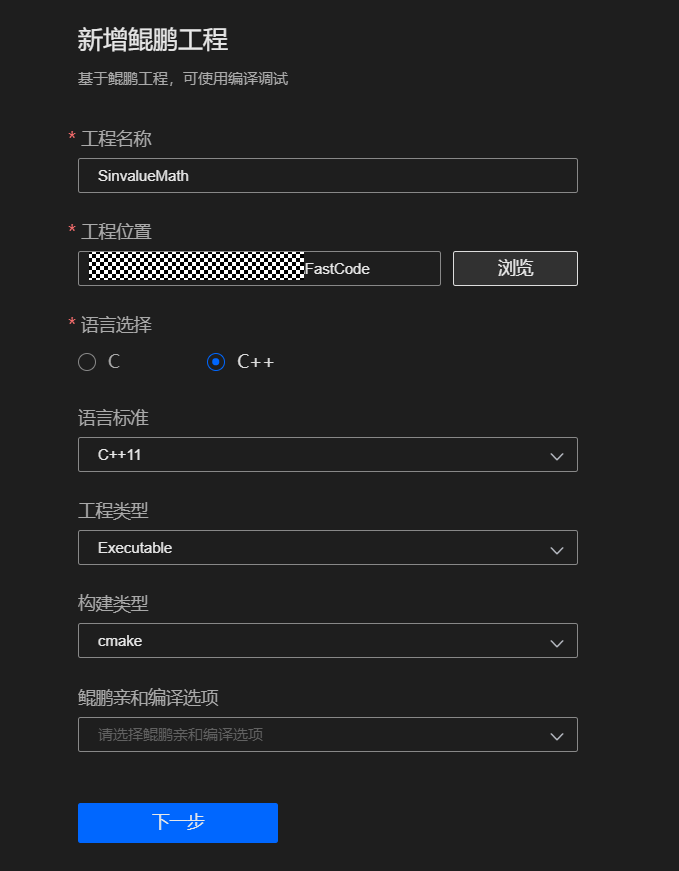
### 插件使用步骤

点击vscode上的鲲鹏开发框架插件，新建文件夹为FastCode

单击左上角“鲲鹏工程”区域 ，配置以下参数，创建鲲鹏工程。

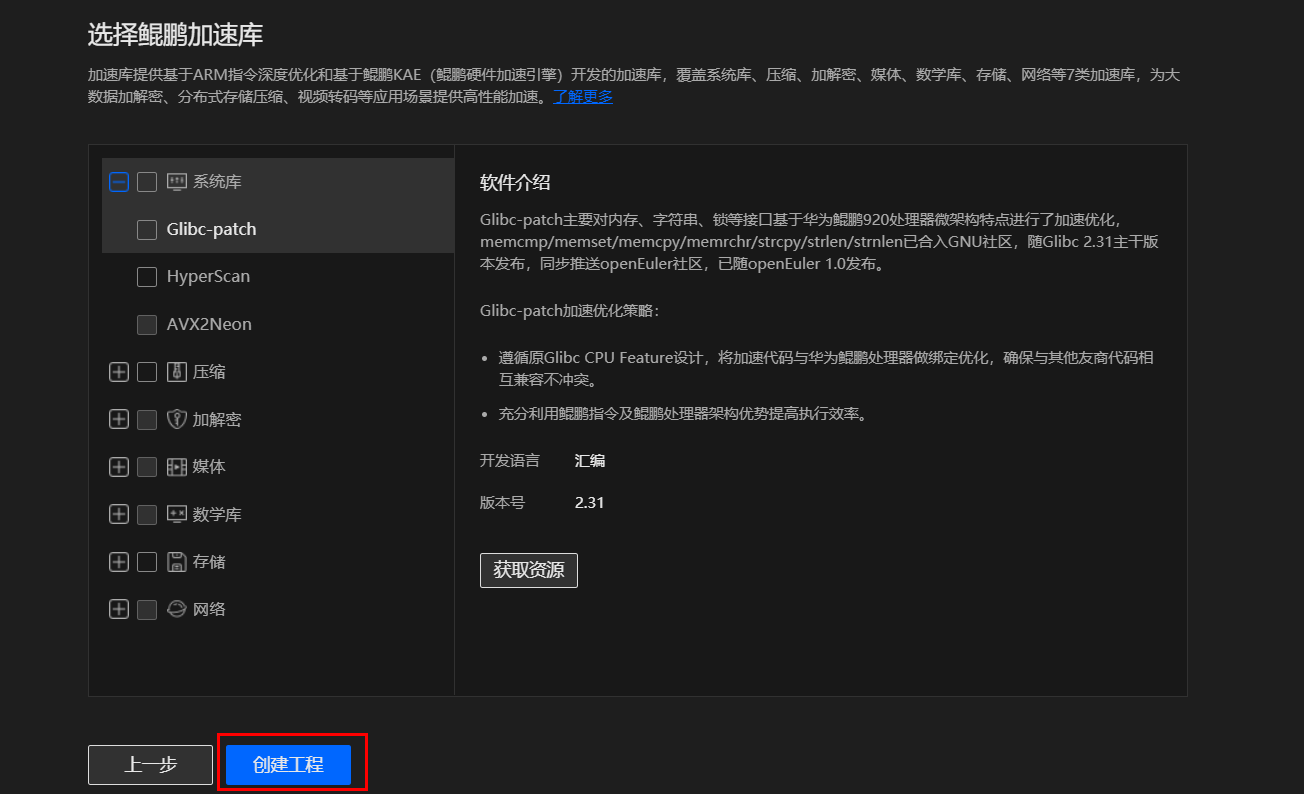
工程名称：SinvalueMath（根据实际情况创建即可）

工程位置选择刚新建的FastCode文件夹。



|  |  |
| --- | --- |
| 参数 | 说明 |
| 工程名称 | 创建的鲲鹏工程名称。  说明  工程名称只能由字母、数字、“.”、“-”、“+”、“()”和“\_”组成，长度为1~64个字符且不能以“.”开头。 |
| 工程位置 | 鲲鹏工程存储路径。 |
| 语言选择 | 选择工程所使用的开发语言。  C语言  C++语言 |
| 语言标准 | 选择工程使用的开发语言标准。  C语言：  C90标准  C99标准  C11标准  C++语言：  C++98标准  C++11标准  C++14标准  C++17标准  C++20标准 |
| 工程类型 | Executable  Libray  static  shared |
| 构建类型 | make  cmake |
| 鲲鹏亲和编译选项 | 全选  -march=armv8.2-a  -march=armv8-a  -mtune=tsv110 |

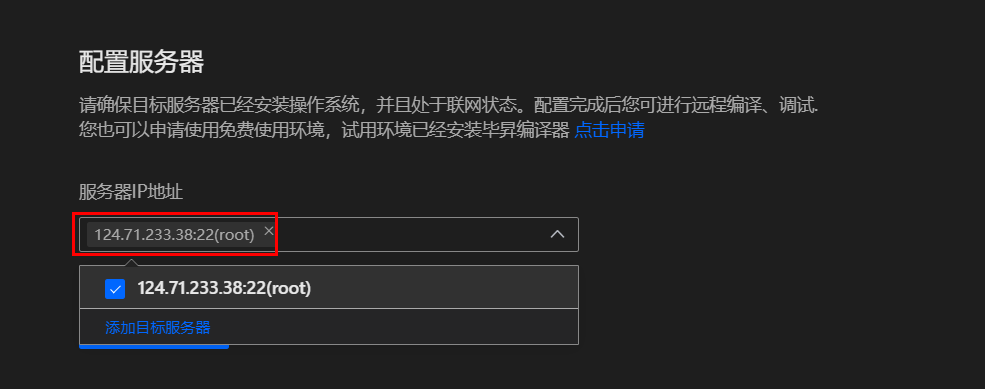
单击“下一步”，进入“选择鲲鹏开发框架”页面。选择所需的加速库。



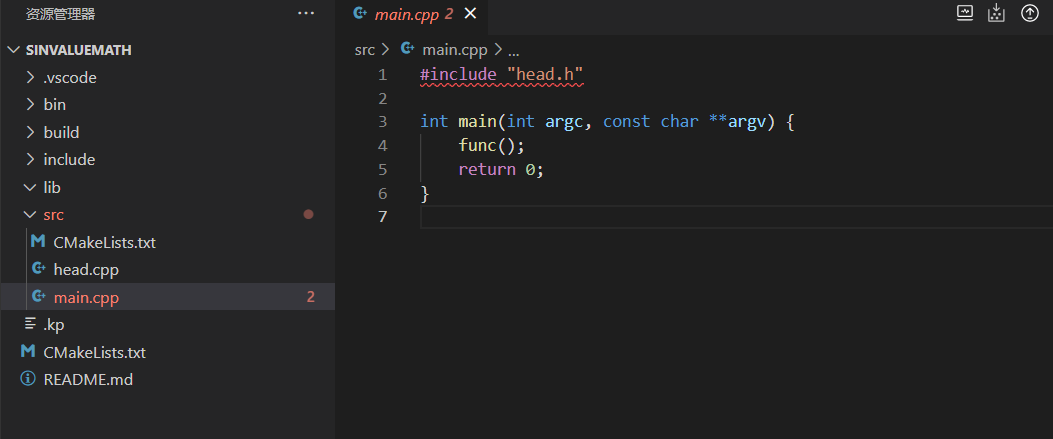
单击“创建工程”，进入加速库下载确认页面。单击“是”进行下载，单击“否”取消下载。



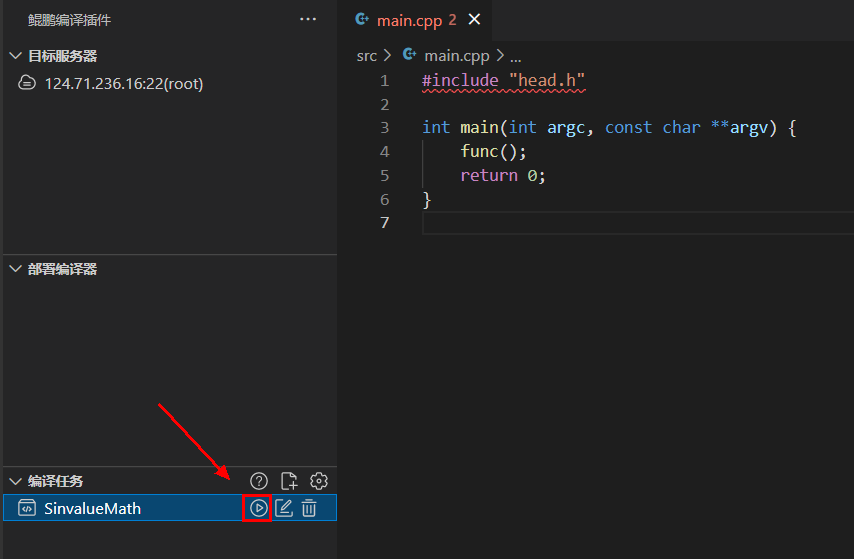
鲲鹏工程创建完成完成后，单击“配置服务器”，进入配置服务器页面，配置IP。



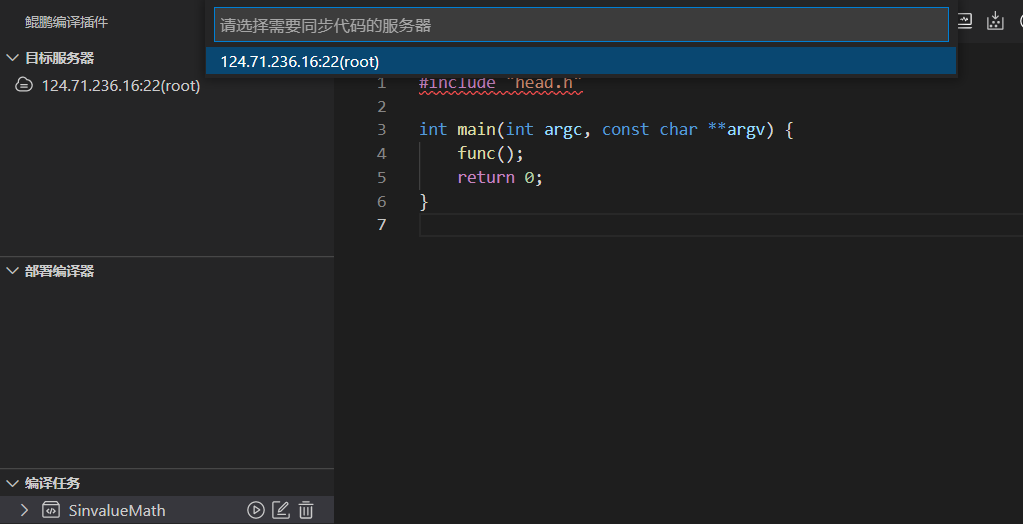
查看鲲鹏工程



验证鲲鹏工程是否正常编译执行,进入编译插件，选择编译插件，点击运行。



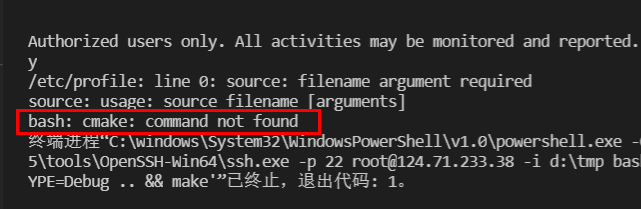
同步代码至服务器



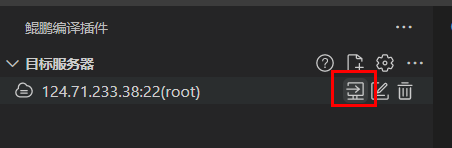
选择同步并编译



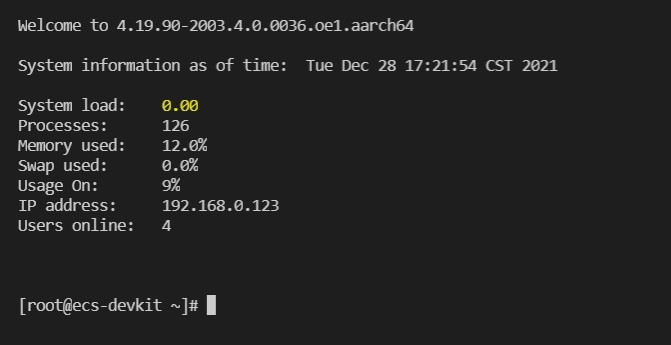
若在编译过程中出现错误如下错误，没有安装cmake,连接服务器安装cmake。（无跳过）



登录服务器的终端

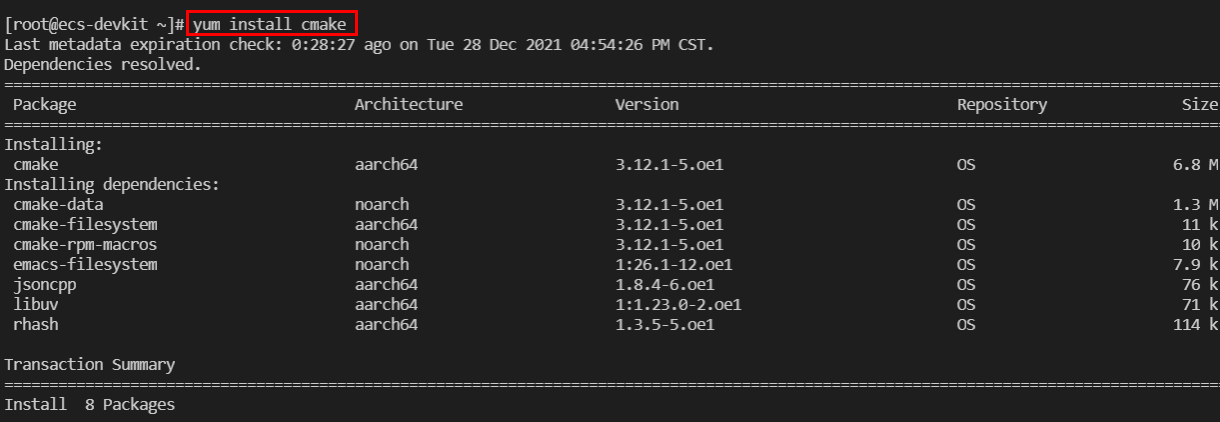


登录成功

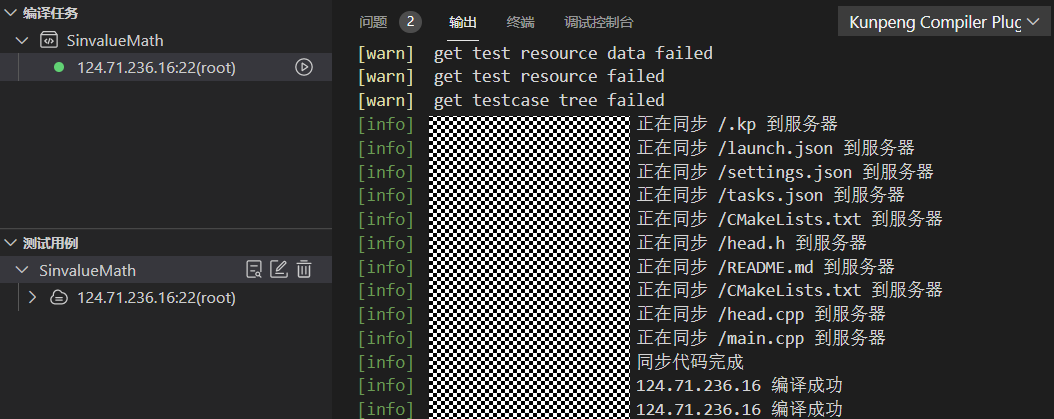


安装cmake,

yum install cmake



重新编译执行。

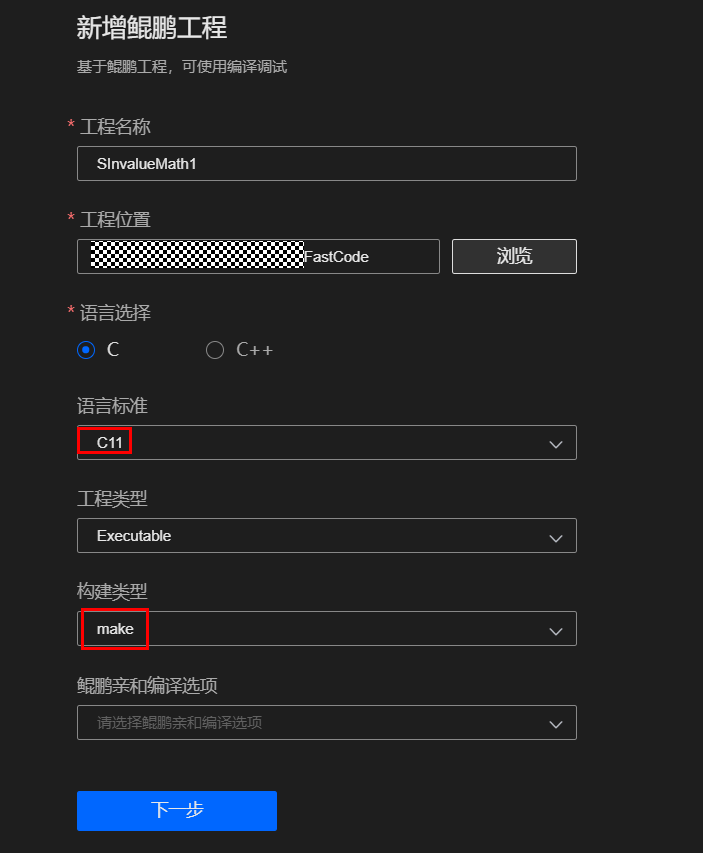


获取测试用例



输出HI kunpeng！表示成功

新建鲲鹏工程，语言c



代码案例

添加代码至新建工程main.c中

求正弦值，使用系统库与KML\_VML加速库

#include <stdio.h>

#include <sys/time.h>

#include <math.h>

#include "kvml.h"

#define LEN 100000

int main()

{

//数据初始化

double src[LEN] = {0};

double dst1[LEN] = {0};

double dst2[LEN] = {0};

for (int i = 0; i < LEN; i++) {

src[i] = i;

}

struct timeval start, end;

long t;

gettimeofday(&start, NULL);

//使用系统数学库对向量中每个元素求正弦值

for (int i = 0; i < LEN; i++) {

dst1[i] = sin(src[i]);

}

gettimeofday(&end, NULL);

//记录计算时间

t = 1000000 \* ( end.tv\_sec - start.tv\_sec ) + end.tv\_usec - start.tv\_usec;

printf("Calculate Time without KML\_VML: %ld us\n", t);

gettimeofday(&start, NULL);

//使用KML\_VML提供的函数计算向量中每个元素的正弦值

vdsin(LEN, src, dst2);

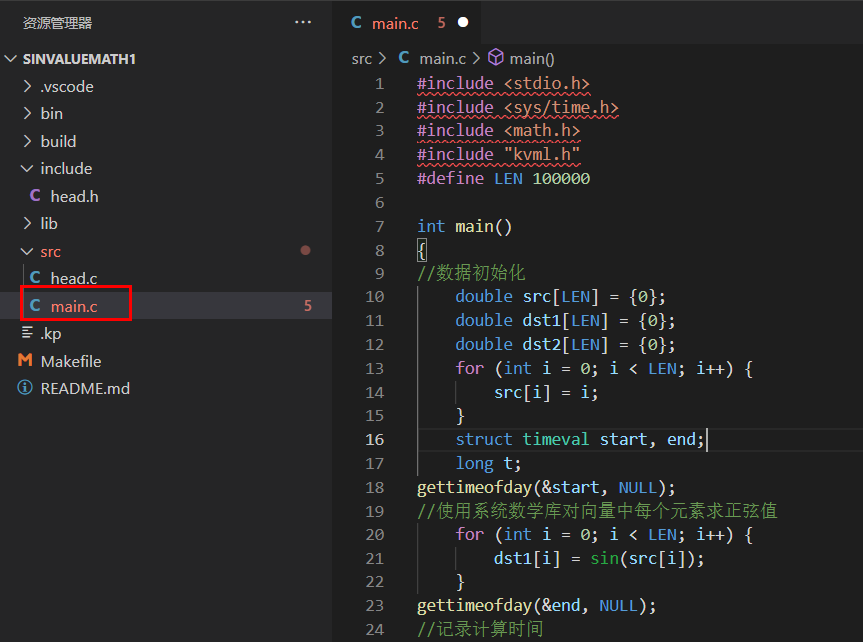
gettimeofday(&end, NULL);

t = 1000000 \* ( end.tv\_sec - start.tv\_sec ) + end.tv\_usec - start.tv\_usec;

printf("Calculate Time with KML\_VML: %ld us\n", t);

return 0;

}



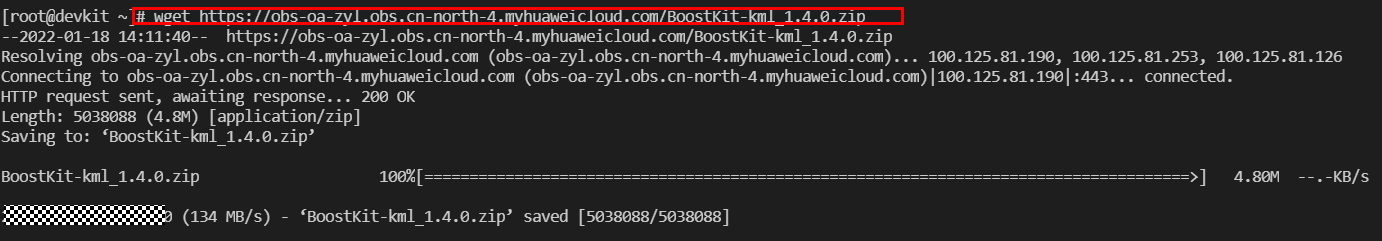
安装加速库

登录目标服务器



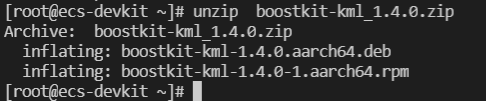
安装加速库

wget https://obs-oa-zyl.obs.cn-north-4.myhuaweicloud.com/BoostKit-kml\_1.4.0.zip



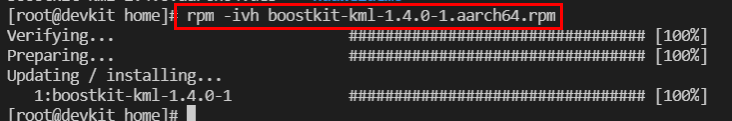
解压

unzip BoostKit-kml\_1.4.0.zip



安装KML

rpm -ivh boostkit-kml-1.4.0-1.aarch64.rpm



安装结束后，系统在环境变量LD\_LIBRARY\_PATH中自动添加lib文件夹所在目录即 “/usr/local/kml/lib”

删除 /etc/profile文件的环境变量

export LD\_LIBRARY\_PATH=/usr/local/kml/lib/kvml/multi:$LD\_LIBRARY\_PATH

export LD\_LIBRARY\_PATH=/usr/local/kml/lib/kspblas/multi:$LD\_LIBRARY\_PATH

执行source命令或重新登陆终端让环境变量生效。

source /etc/profile

查看环境变量LD\_LIBRARY\_PATH是否包含KML的安装路径“/usr/local/kml/lib”。

env | grep LD\_LIBRARY\_PATH

鲲鹏工程中编写Makefile文件。

src = ./src

bin = ./bin

cc = gcc

CXXFLAGS = -std=c99

LIBRARY = -fPIC -shared

INCLUDE = -I /usr/local/kml/include

LD\_PATH = -L /usr/local/kml/lib/kvml/single

default: all

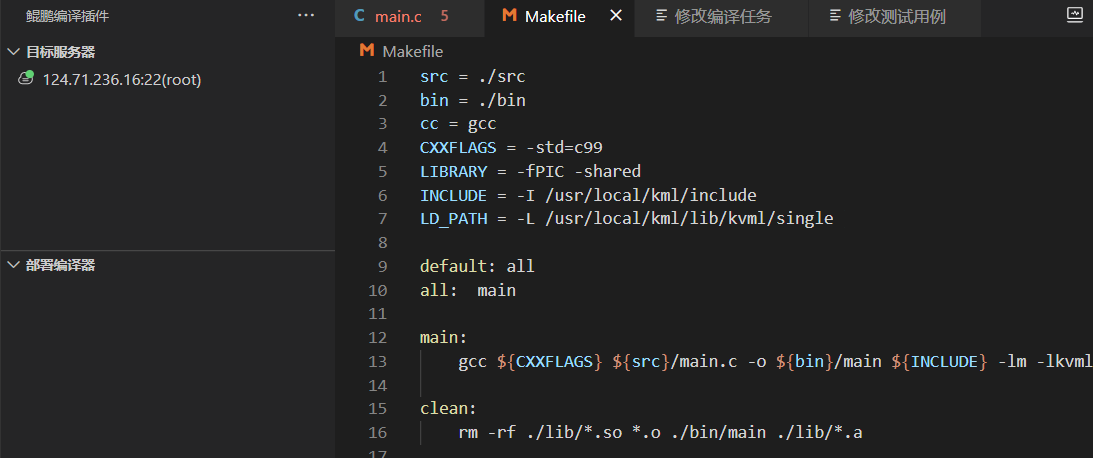
all: main

main:

gcc ${CXXFLAGS} ${src}/main.c -o ${bin}/main ${INCLUDE} -lm -lkvml ${LD\_PATH}

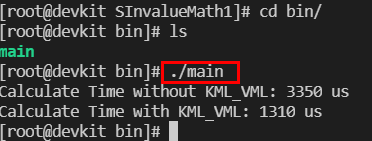
clean:

rm -rf ./lib/\*.so \*.o ./bin/main ./lib/\*.a

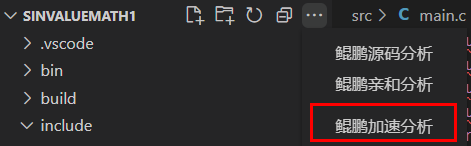


编译（参考步骤三）进入到当前生成目录中执行

./main



加速分析

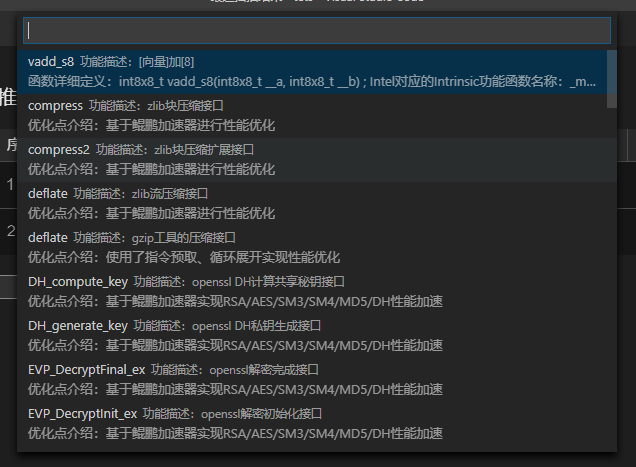
单击“鲲鹏加速分析”后，插件会分析工程里面依赖加速库的函数，并将扫描的结果在可视化面板以及VSCode的问题栏展示。点击问题栏中的函数可以跳转到函数方法，点击可视化面板中中的“查看”可以跳转到函数方法。点击“下载”会跳转到函数所在的源码仓库地址。





函数搜索

单击编辑区右上角的 ，在搜索框中输入搜索关键字，然后单击function函数会跳转头文件定义，单击Intrinsic函数会打开Intrinsic的帮助文档地址。



## 实验环境清理

清理ECS资源。

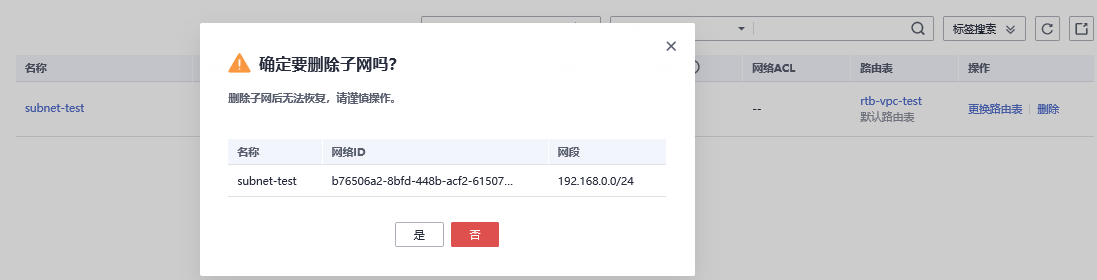
返回ECS控制台，选择kp-test云主机，然后点击“更多>删除。

在弹出的对话框中勾选“释放云服务器绑定的弹性公网IP地址”和“删除云服务器挂载的数据盘”，然后点击“是”，删除ECS。



清理安全组、子网、EPC。



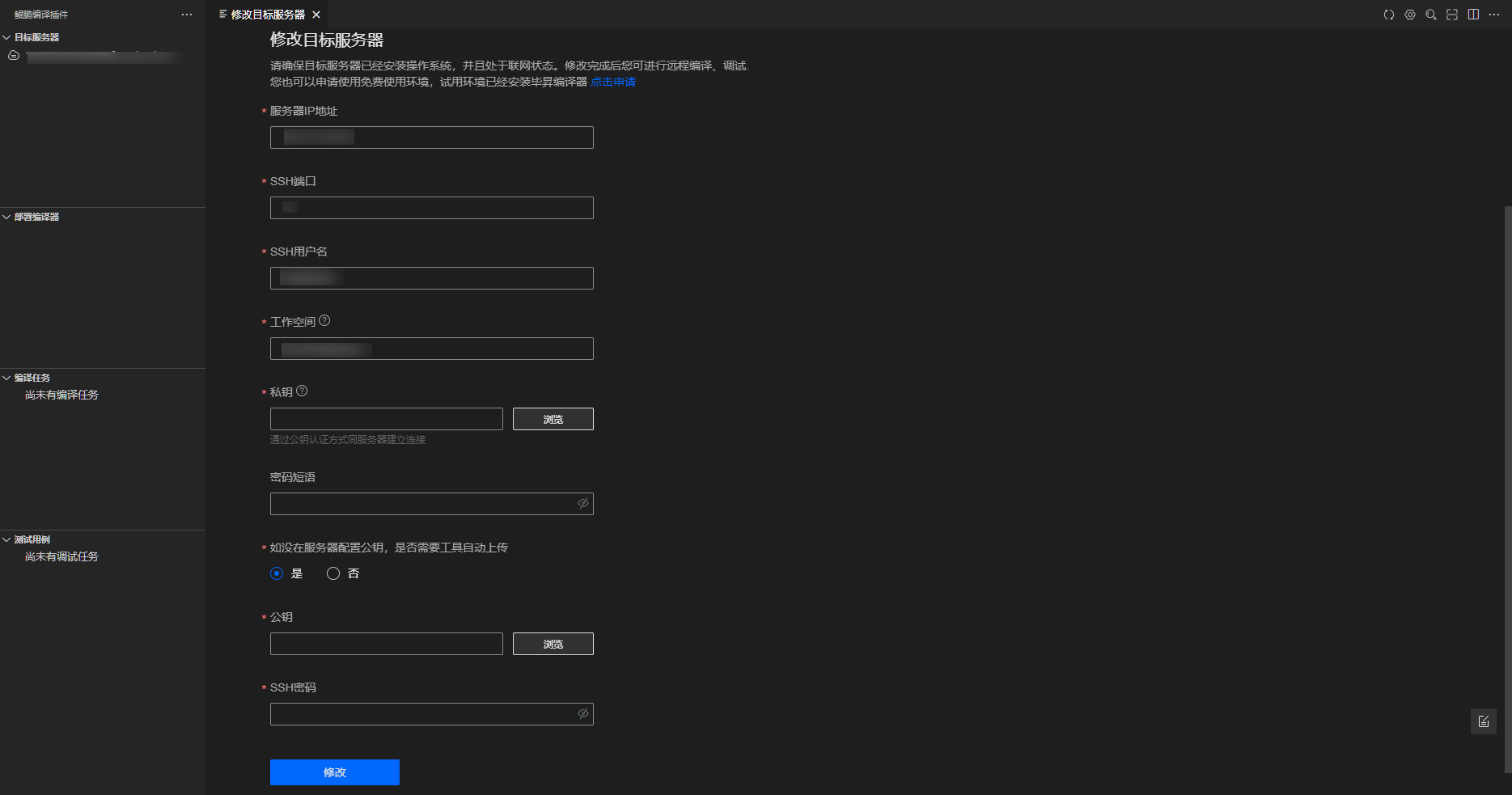




## 附录

### 修改和删除目标服务器

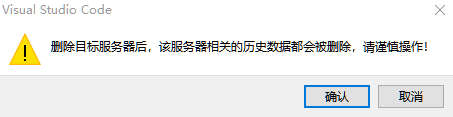
选中目标服务器列表中的目标服务器，单击右侧 图标，进入修改目标服务器页面。



修改目标服务器参数，步骤 3 修改完成后，单击“修改”，完成对目标服务器的修改。

单击选中目标服务器列表中的目标服务器，单击右侧的。

在弹出的选择框中，单击确认，完成删除目标服务器。



### 使用毕昇编译器，

则需要先在服务端安装毕昇编译器。毕昇编译器基于开源LLVM开发，并进行了优化和改进，同时将flang作为默认的Fortran语言前端编译器，是针对鲲鹏平台的高性能编译器。

当前仅以下操作系统下支持安装毕昇编译器：

* openEuler 20.03（LTS）
* openEuler 21.03（LTS）
* CentOS 7.6
* Ubuntu 18.04
* Ubuntu 18.04.2 LTS
* Ubuntu 20.04 LTS
* Ubuntu 20
* Kylin Linux Advanced Server V10 (Azalea)
* UOS Server Euler 20

在左侧菜单栏单击  > 部署编译器，进入部署页面。

配置目标服务器参数后，单击“开始部署”，一键式安装毕昇编译器。



|  |  |
| --- | --- |
| 参数 | 说明 |
| SSH端口 | 待安装毕昇编译器的服务器SSH端口。 |
| SSH用户名 | 登录待安装毕昇编译器的服务器的SSH用户名。  说明  如果想通过普通用户安装，需要满足一些条件，具体参照5.3.3 如何通过普通用户安装编译器。 |
| 选择连接方式 | 密码认证或密钥认证。 |
| SSH密码 | 输入SSH密码。 |
| 私钥 | 导入id\_rsa私钥文件。 |
| 密码短语 | （可选）输入生成密钥设置的密码短语。 |

安装过程中需要：

* 再次输入服务器用户密码
* 输入服务器root用户密码毕昇编译器

### 安装毕昇JDK

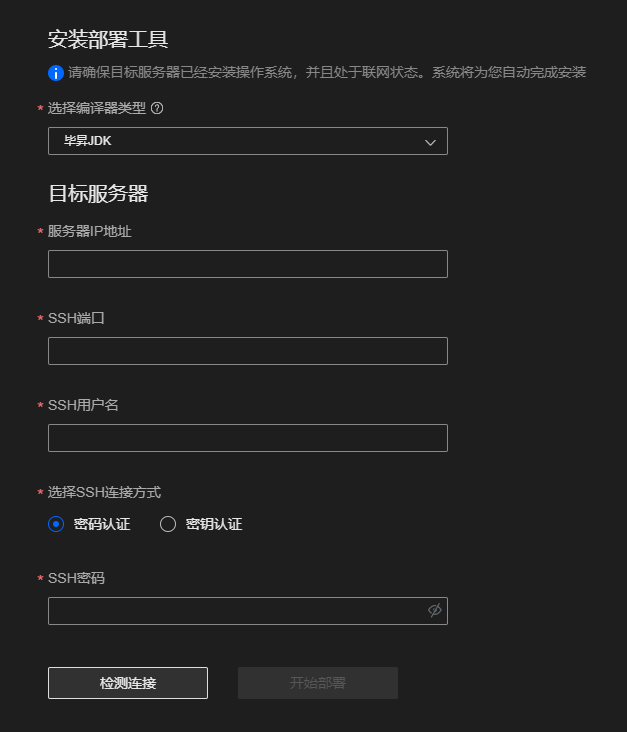
如果您需要使用毕昇JDK，则需要先在服务端安装毕昇JDK。毕昇JDK基于OpenJDK开发，是一个高性能、可用于生产环境的OpenJDK发行版，它积累了大量使用场景和Java开发者反馈的问题和诉求，解决了业务实际运行中遇到的多个问题，并在ARM架构上进行了性能优化。

当前仅以下操作系统下支持安装毕昇JDK：

* openEuler 20.03（LTS）
* openEuler 21.03（LTS）
* CentOS 7.6
* Ubuntu 18.04
* Ubuntu 18.04.2 LTS
* Ubuntu 20.04 LTS
* Ubuntu 20
* Kylin Linux Advanced Server V10 (Azalea)
* UOS Server Euler 20

在左侧菜单栏单击  > 部署编译器，进入部署页面。

配置目标服务器参数后，单击“开始部署”，一键式安装毕昇JDK



|  |  |
| --- | --- |
| 参数 | 说明 |
| 服务器IP地址 | 待安装毕昇JDK的服务器IP地址。 |
| SSH端口 | 待安装毕昇JDK的服务器SSH端口。 |
| SSH用户名 | 登录待安装毕昇JDK的服务器的SSH用户名。  说明  如果想通过普通用户安装，需要满足一些条件，具体参照5.3.3 如何通过普通用户安装编译器。 |
| 选择连接方式 | 密码认证或密钥认证。 |
| SSH密码 | 输入SSH密码。 |
| 私钥 | 导入id\_rsa私钥文件。 |
| 密码短语 | （可选）输入生成密钥设置的密码短语。 |

安装过程中需要：

* 再次输入服务器用户密码
* 输入服务器root用户密码毕昇编译器

### 鲲鹏gcc支持的操作系统

当前仅以下操作系统下支持安装鲲鹏GCC编译器：

* openEuler 20.03（LTS）
* openEuler 21.03（LTS）
* CentOS 7.6
* Ubuntu 18.04
* Ubuntu 18.04.2 LTS
* Ubuntu 20.04 LTS
* Ubuntu 20
* Kylin Linux Advanced Server V10 (Azalea)
* UOS Server Euler 20