# 《并行计算》实验指导书

# 实验1：基于华为云的并行环境搭建和使用

## 实验1.1华为云实验环境说明

### 实验目的

1．了解华为云环境的使用过程。

2．掌握利用云环境搭建小型集群环境的过程。

### 实验内容

实验步骤：

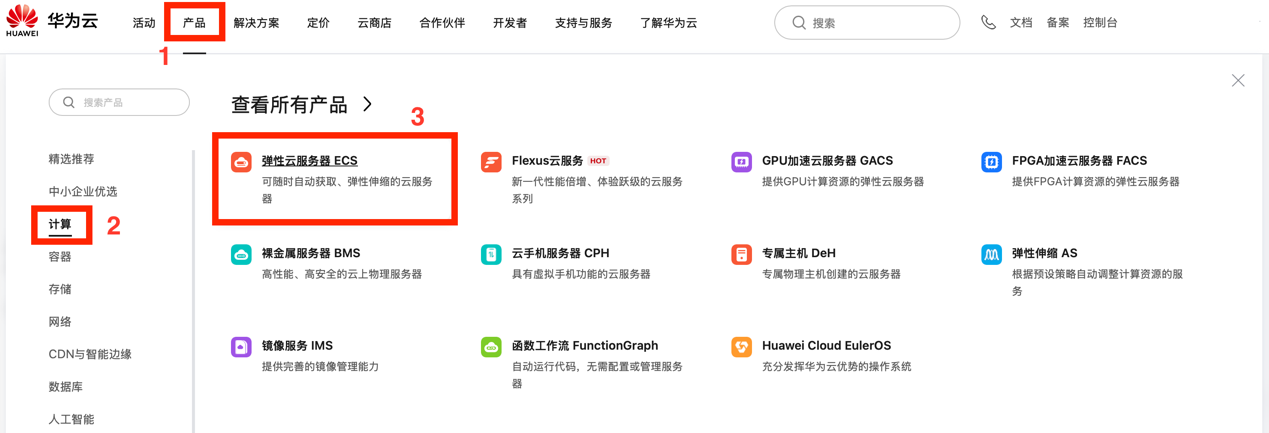
* 步骤一：登录华为云。

1. 打开浏览器，前往“华为云”官网：<https://www.huaweicloud.com>
2. 点击右上角登录按钮，输入用户名和密码。



* 步骤二：购买“弹性云服务器”

1. 点击上方导航条中的“产品”
2. 选择左侧列表中的“计算”
3. 点击主界面中的 “弹性云服务器ECS”
4. 购买





若提示需要实名认证，则先完成实名认证再进行后续操作。



* 步骤三：配置服务器

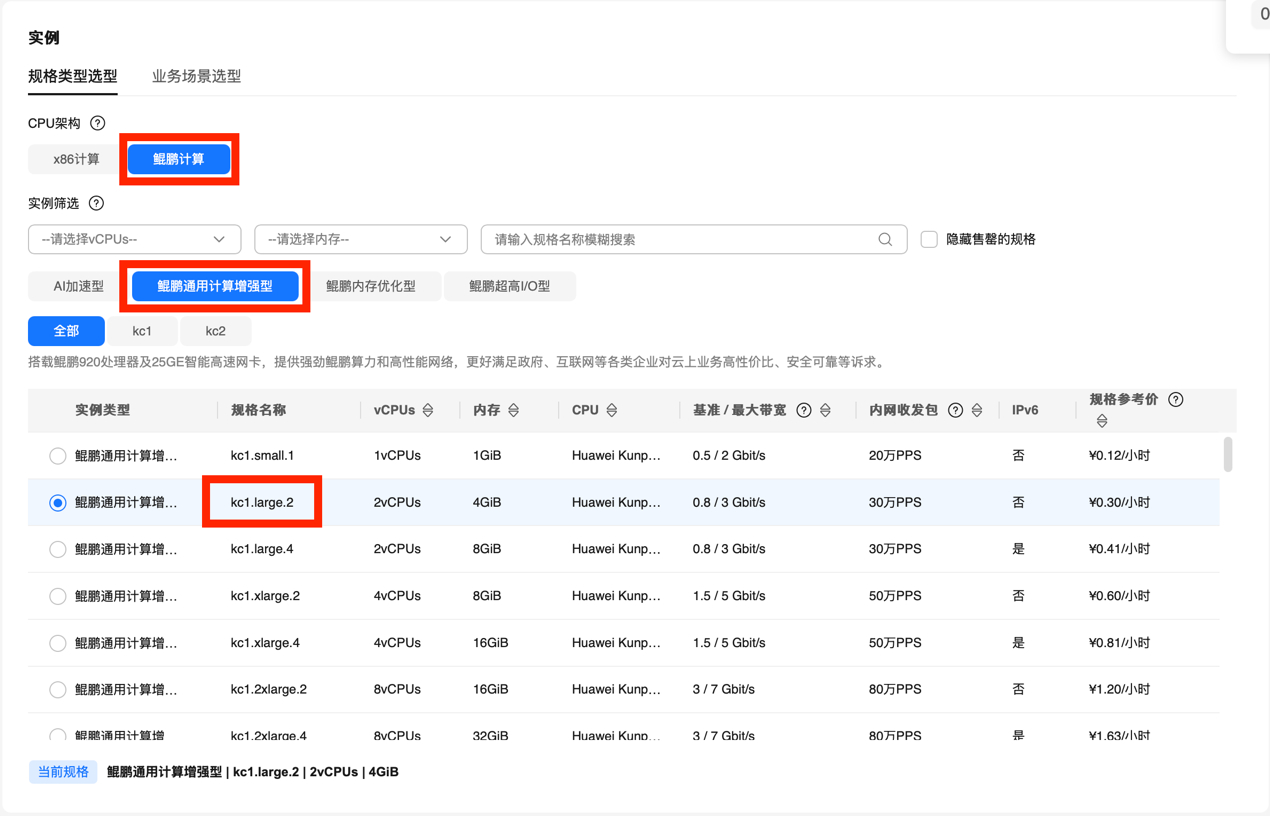
按照下表进行购买，一共购买3台。

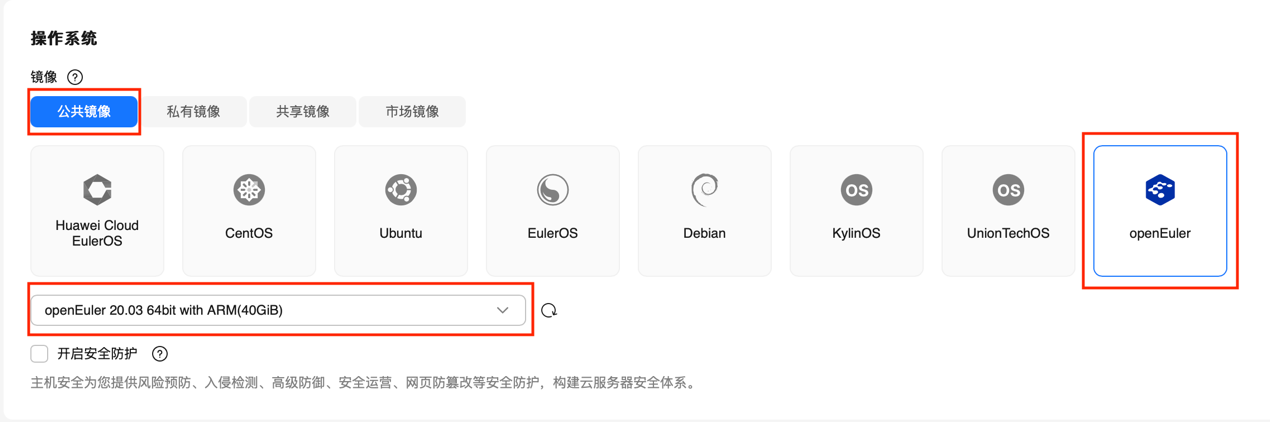
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 计费模式 | 区域 | CPU架构 | 规格 | 镜像 | 系统盘 |
| 按需计费 | 华北-北京四 | 鲲鹏计算 | kc1.large.2 | 公共镜像：openEuler 20.03 | 至少40GB |

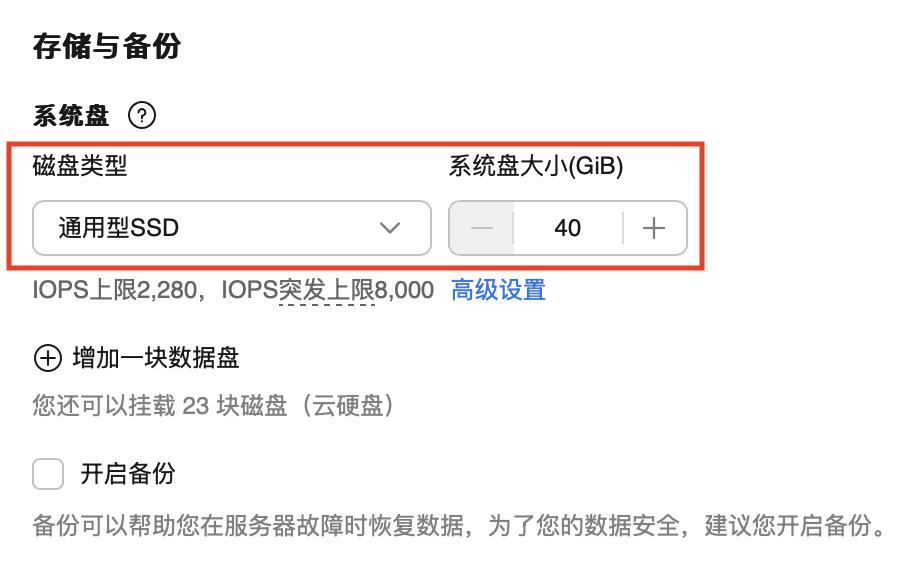
注意：一定要选择openEuler，否则安装软件包会有问题。

参考截图：

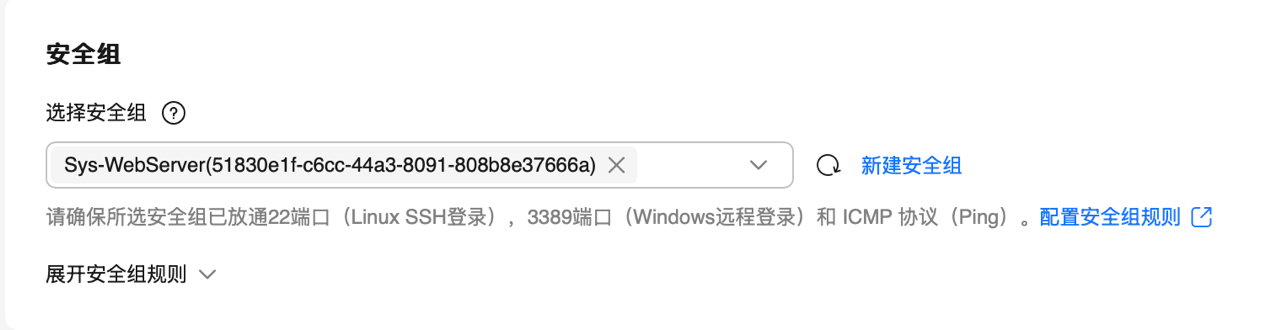














设置云服务器名称（ecs-hw）和密码（Parallel2022）。



填入购买量3台，点击“立即购买”、“同意并立即购买”。

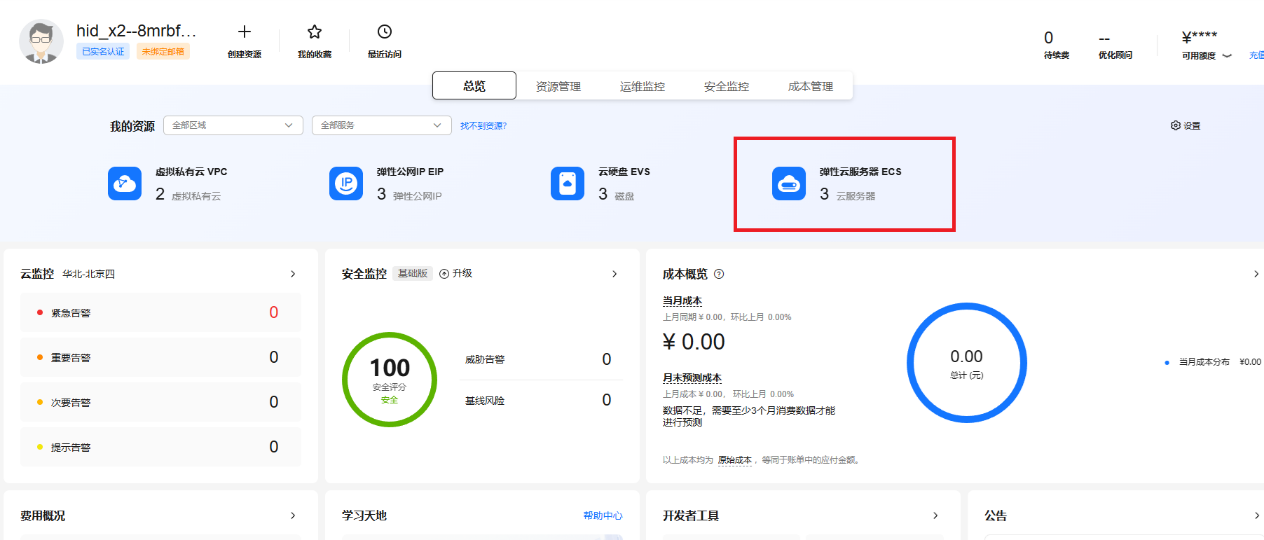


* 步骤四：查看云服务器虚拟机

1. 点击导航栏右上角“控制台”



1. 点击“弹性云服务器ECS”



图形用户界面, 文本, 应用程序, 电子邮件

描述已自动生成

* 步骤五：记录每台服务器私有IP

图形用户界面, 文本, 应用程序, 电子邮件

描述已自动生成

例如： ecs-hw-0001的私有IP为 192.168.0.154

ecs-hw-0002的私有IP为 192.168.0.203

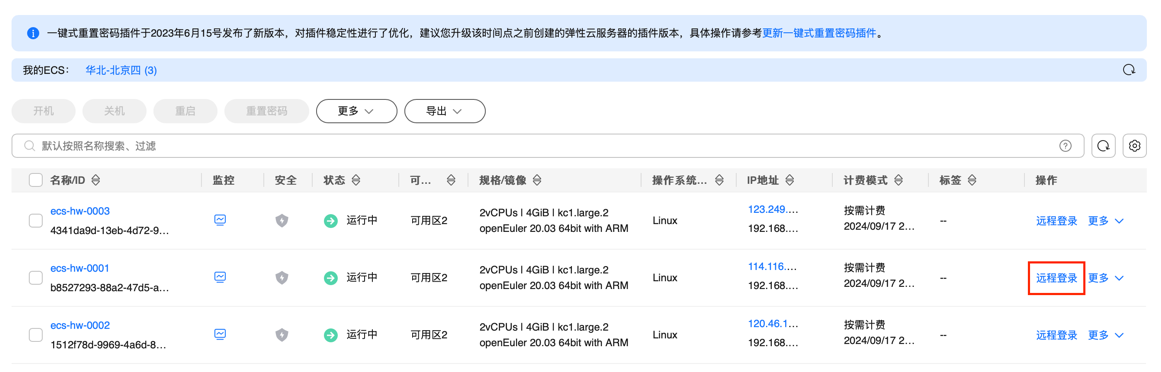
ecs-hw-0003的私有IP为 192.168.0.167

你的私有IP会和我的不一样，你自己要记清楚，以便后续步骤使用。

此处经常出错。

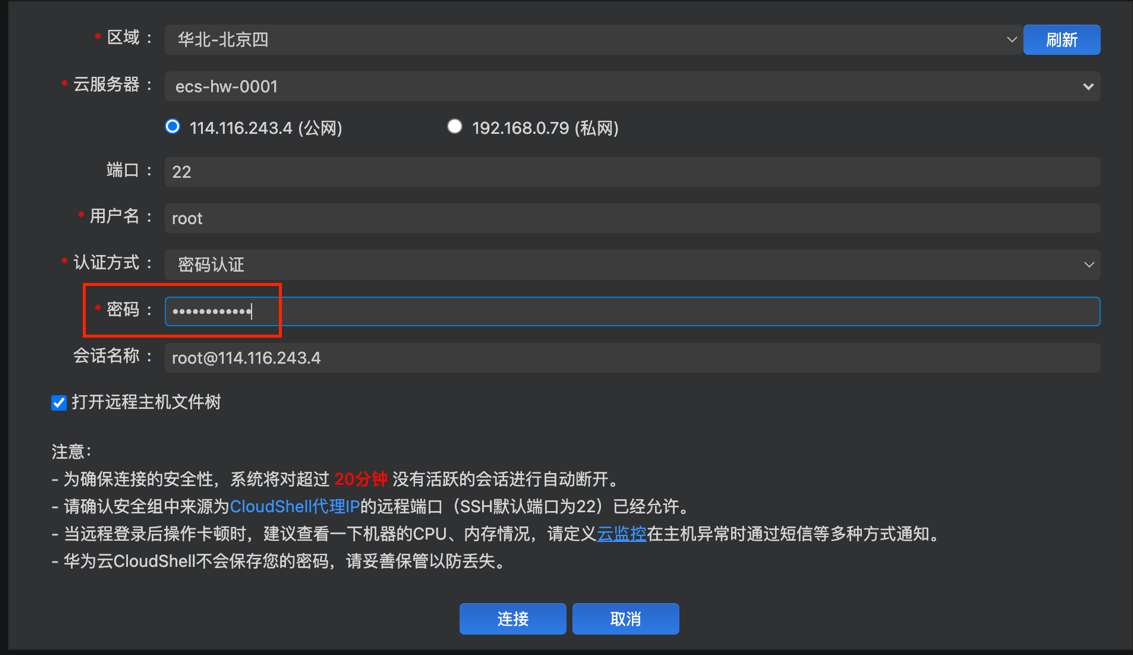
* 步骤六：远程登录ECS

1. 点击远程登录，建议选择CloudShell登录。





1. 用户名：root 密码：Parallel2022。输入后点击“连接”。



如果一直显示“连接远程服务器”，则使用其他浏览器尝试。

建议使用Chrome浏览器。

* 步骤七：创建用户

1. 在3台服务器上，均使用以下命令创建个人账户：

|  |
| --- |
| adduser pengjinjie // 添加用户。将zhangsan替换为你自己的名字  passwd pengjinjie // 为新用户设置密码。将zhangsan替换为你自己的名字  // 输入新密码，记清楚你自己的用户名和密码  usermod -aG wheel pengjinjie // 新用户加入wheel组，取得root权限。将zhangsan替换为你自己的名字 |

为避免出错，建议三台服务器上使用相同的新用户名和密码。

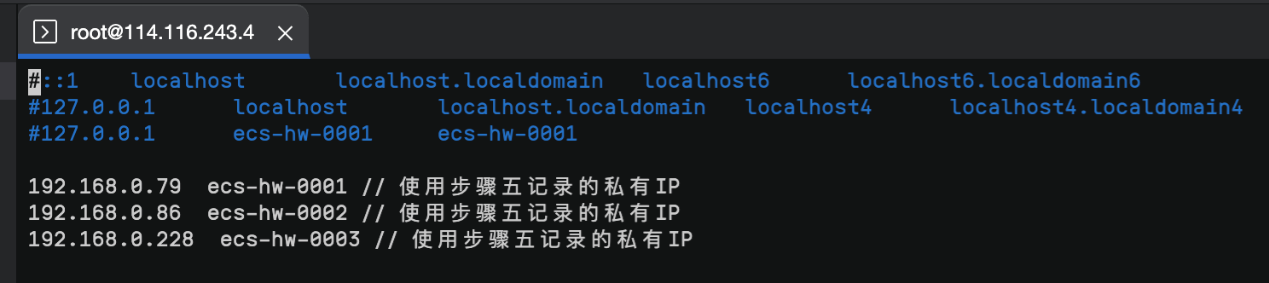
* 步骤八：免密配置

1. 在3台服务器上，使用以下命令查看/etc/hosts：

|  |
| --- |
| vim /etc/hosts |

1. 在3台服务器上，将文件中已有内容全部注释掉，并添加以下信息：

|  |
| --- |
| 192.168.0.154 ecs-hw-0001  192.168.0.203 ecs-hw-0002  192.168.0.167 ecs-hw-0003 |

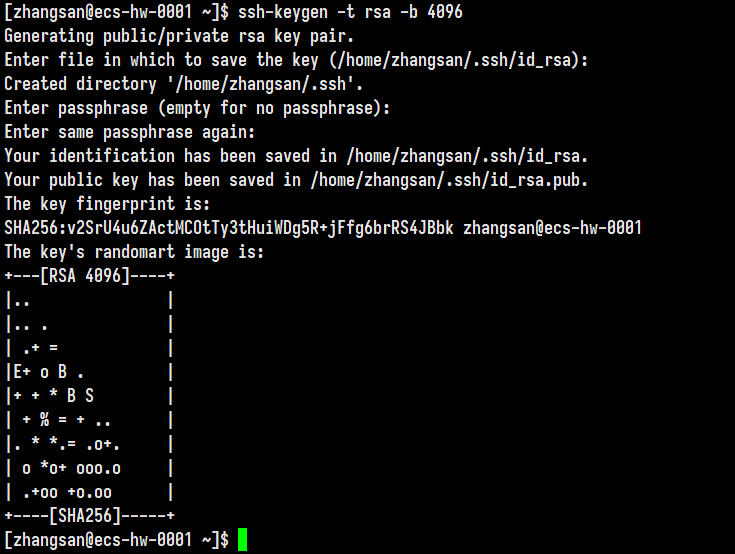


1. 在3台服务器上，均使用以下命令由root用户切换到新创建的个人账户：

|  |
| --- |
| su - pengjinjie // 替换为新创建的用户名 |

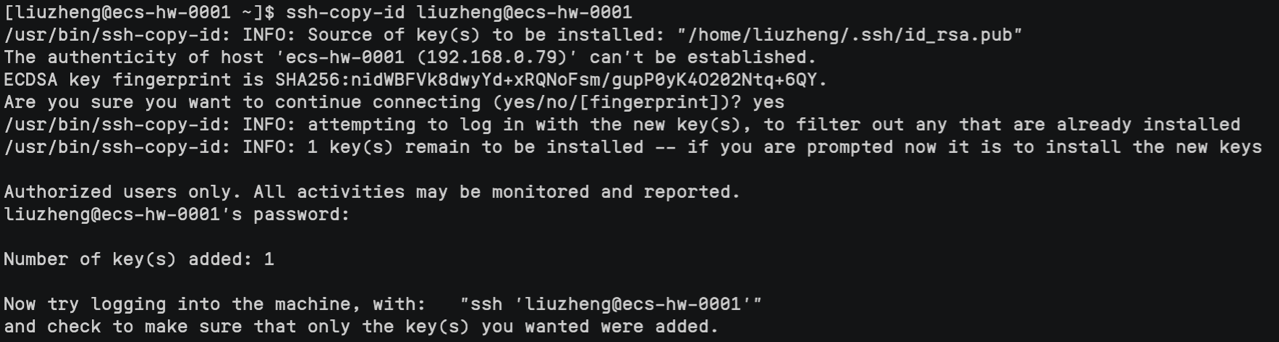
1. 在3台服务器上，均使用以下命令生成本地密钥:

|  |
| --- |
| ssh-keygen -t rsa -b 4096 // 按Enter跳过所有输入 |

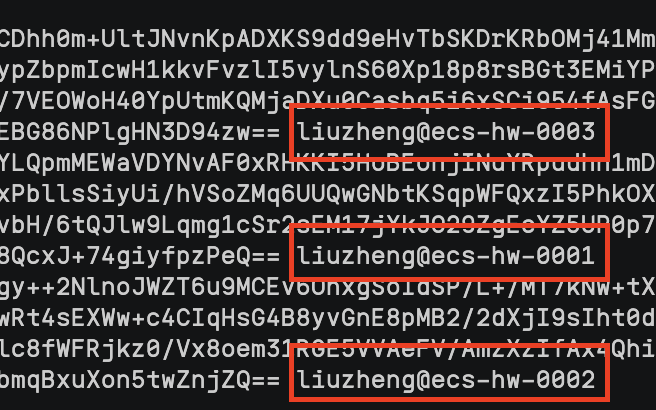


1. 在3台服务器上，均使用以下命令添加公钥至所有主机:

|  |
| --- |
| ssh-copy-id pengjinjie@ecs-hw-0001 // 将zhangsan替换为你的用户名  // 输入yes，然后输入目标服务器的用户密码  ssh-copy-id pengjinjie@ecs-hw-0002 // 将zhangsan替换为你的用户名  // 输入yes，然后输入目标服务器的用户密码  ssh-copy-id pengjinjie@ecs-hw-0003 // 将zhangsan替换为你的用户名  // 输入yes，然后输入目标服务器的用户密码 |



可以使用 more ~/.ssh/authorized\_keys 查看是否copy成功：



每台服务器上都应该有0001、0002、0003，顺序可能不同。

* 步骤九：配置MPI

1. 在3台服务器上，使用以下命令安装依赖包：

|  |
| --- |
| sudo yum -y install gcc-gfortran --nogpgcheck  sudo yum -y install mpich mpich-devel mpich-doc --nogpgcheck |

1. 在3台服务器上，使用以下命令配置环境变量:

|  |
| --- |
| vim ~/.bashrc |

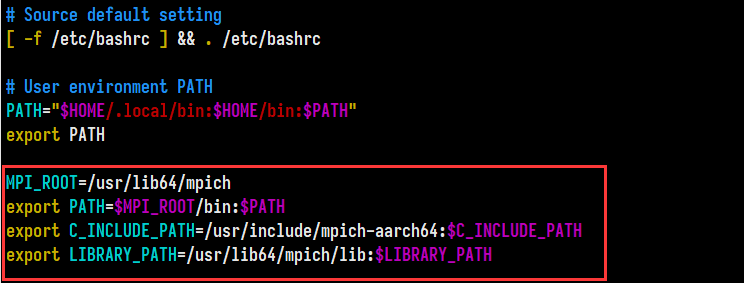
1. 在文件结尾添加并保存：

MPI\_ROOT=/usr/lib64/mpich

export PATH=$MPI\_ROOT/bin:$PATH

export C\_INCLUDE\_PATH=/usr/include/mpich-aarch64:$C\_INCLUDE\_PATH

export LIBRARY\_PATH=/usr/lib64/mpich/lib:$LIBRARY\_PATH



1. 执行source命令，使之生效。

|  |
| --- |
| source ~/.bashrc |

## 实验1.2 并行环境下程序的编译和运行

### 实验目的

1. 通过在华为鲲鹏上编译运行并行程序，掌握集群环境并行计算的配置以及加深对并行计算的了解.

### 实验内容

实验步骤：

以下步骤均在ecs-hw-0001上changtian用户下执行。

su - changtian

* 步骤一：创建程序源码

1. 创建hello目录存放该程序的所有文件, 并进入hello目录:

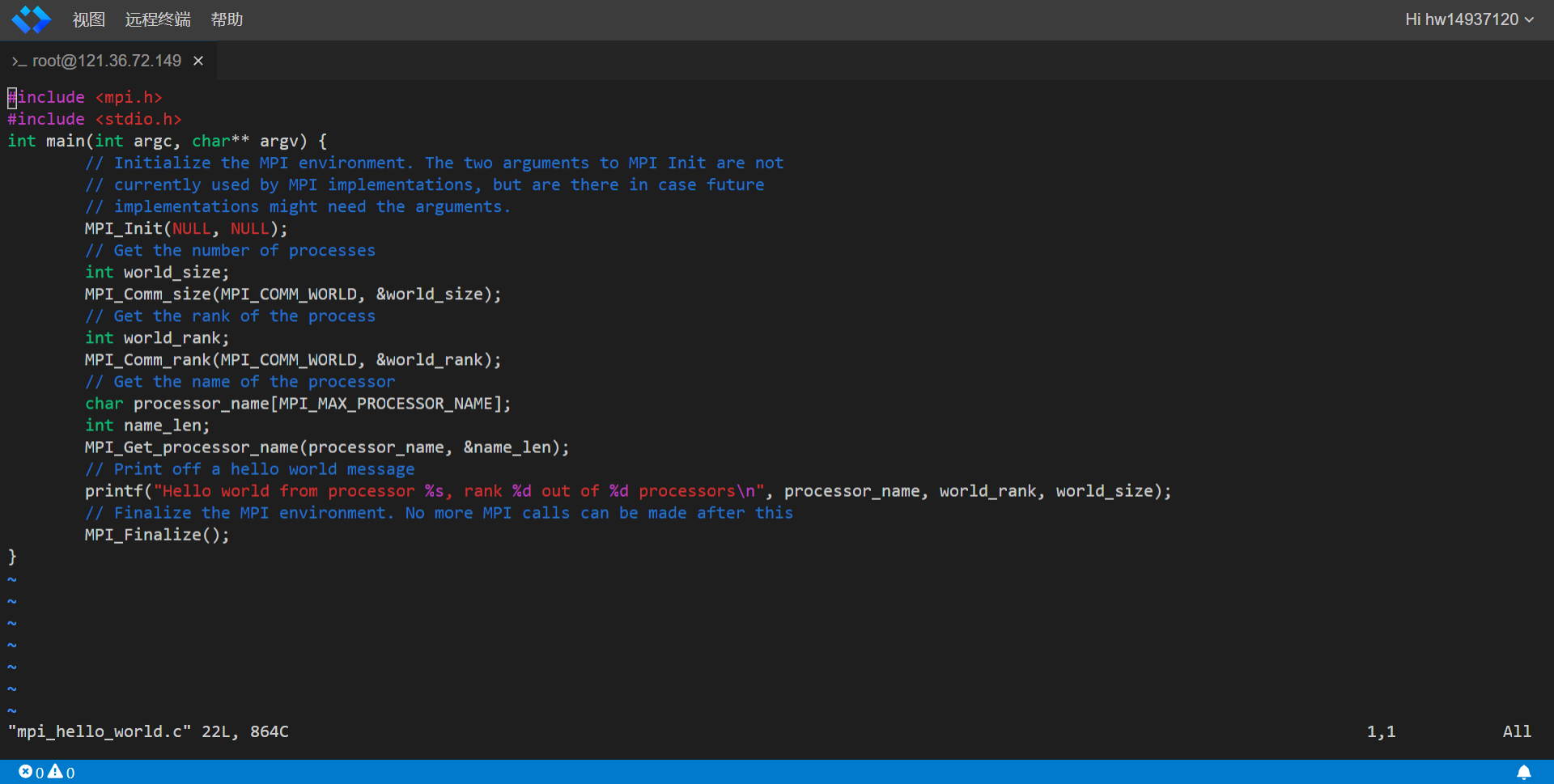
|  |
| --- |
| mkdir /home/pengjinjie/hello  cd /home/pengjinjie/hello |

1. 创建示例程序源码mpi\_hello\_world.c

|  |
| --- |
| vim mpi\_hello\_world.c |

代码内容如下：

|  |
| --- |
| #include <mpi.h>  #include <stdio.h>  int main(int argc, char\*\* argv) {  // Initialize the MPI environment. The two arguments to MPI Init are not  // currently used by MPI implementations, but are there in case future  // implementations might need the arguments.  MPI\_Init(NULL, NULL);  // Get the number of processes  int world\_size;  MPI\_Comm\_size(MPI\_COMM\_WORLD, &world\_size);  // Get the rank of the process  int world\_rank;  MPI\_Comm\_rank(MPI\_COMM\_WORLD, &world\_rank);  // Get the name of the processor  char processor\_name[MPI\_MAX\_PROCESSOR\_NAME];  int name\_len;  MPI\_Get\_processor\_name(processor\_name, &name\_len);  // Print off a hello world message  printf("Hello world from processor %s, rank %d out of %d processors\n", processor\_name, world\_rank, world\_size);  // Finalize the MPI environment. No more MPI calls can be made after this  MPI\_Finalize();  } |



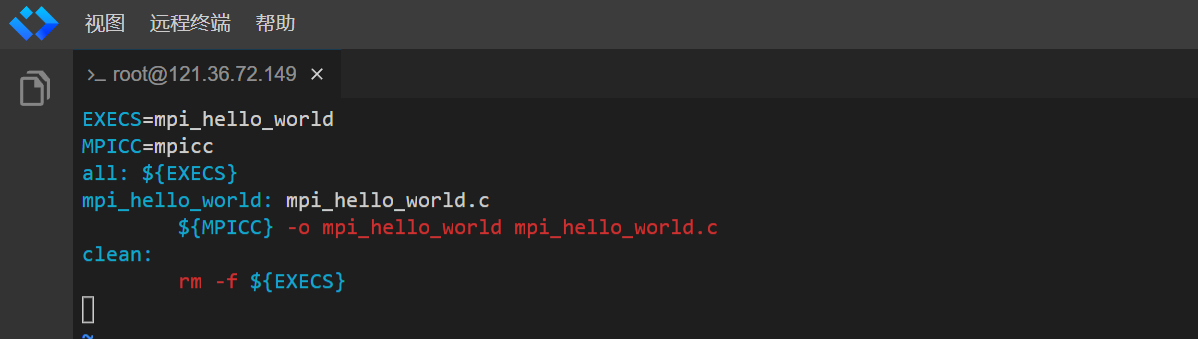
* 步骤二：创建makefile

1. 执行以下命令，创建makefile

|  |
| --- |
| vim makefile |

代码内容如下：

|  |
| --- |
| EXECS=mpi\_hello\_world  MPICC=mpicc  all: ${EXECS}  mpi\_hello\_world: mpi\_hello\_world.c  ${MPICC} -o mpi\_hello\_world mpi\_hello\_world.c  clean:  rm -f ${EXECS} |



或者直接使用mpicc命令进行编译，编译命令如下：

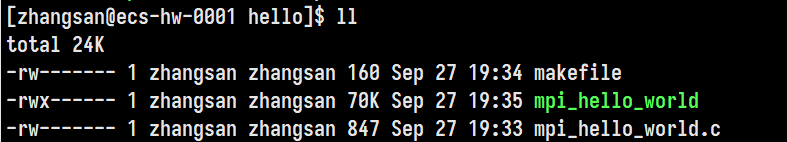
|  |
| --- |
| mpicc mpi\_hello\_world.c -o mpi\_hello\_world |

* 步骤三：源码编译

1. 执行以下命令，进行编译

|  |
| --- |
| cd /home/pengjinjie/hello  make |

编译的文件列表如下：



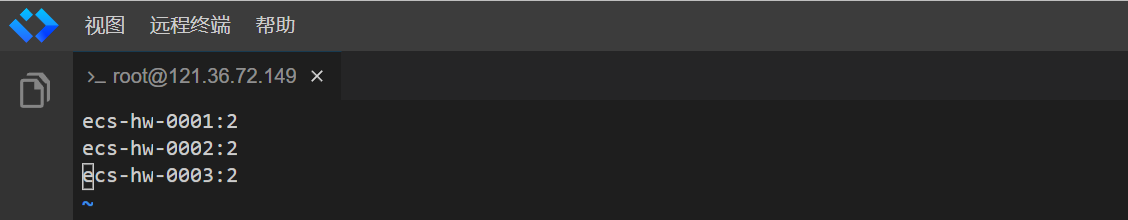
* 步骤四：建立主机配置文件

1. 执行以下命令，建立主机配置文件

|  |
| --- |
| vim /home/pengjinjie/hello/config |

添加以下内容：

|  |
| --- |
| ecs-hw-0001:2  ecs-hw-0002:2  ecs-hw-0003:2 |



* 步骤五：将目录拷贝到其他节点

|  |
| --- |
| scp -r /home/pengjinjie/hello/ pengjinjie@ecs-hw-0002:/home/pengjinjie/  scp -r /home/pengjinjie/hello/ pengjinjie@ecs-hw-0003:/home/pengjinjie/ |

注意修改用户名

* 步骤六：运行监测

1. 执行以下命令，查看运行结果（只需要在ecs-hw-0001上执行）

|  |
| --- |
| mpirun -np 6 -f /home/pengjinjie/hello/config /home/pengjinjie/hello/mpi\_hello\_world |

注意修改用户名

**文本

描述已自动生成**

**要求：**查看结果，并截图保存。

通过上述代码运行，可以看出，编写的hello-wolrd程序可以在华为鲲鹏上运行，程序在集群之间并行计算处理。

## 实验1.3 PI程序的编译和运行

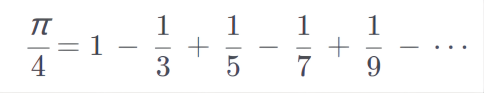
### 实验目的

1. 集群环境中并行程序编译实战，了解在不同进程下程序的性能变化，体会并行计算的优势。

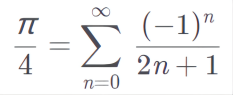
### 实验内容

* 程序原理介绍：

在数学领域，π的莱布尼茨公式说明



右边的展式是一个无穷级数，被称为莱布尼茨级数，这个级数收敛到π/4。它通常也被称为格雷戈里-莱布尼茨级数，用以纪念莱布尼茨同时代的天文学家兼数学家詹姆斯·格雷戈里。使用求和符号可记作：



利用该公式，可以对π进行求解，项数越多，π值的求解越精确。

* 串行代码（pi\_c.c）如下：

|  |
| --- |
| /\* File: pi\_c.c  \*  \* Purpose: Estimates pi using the Leibniz formula (in C).  \*  \*  \* Compile: gcc -o pi\_c pi\_c.c -lm  \* Run: ./pi\_c  \*/  #include <stdio.h>  #include <math.h>  #include <time.h>  /\* We define pi here so we can check and see how accurate our computation is. \*/  #define PI 3.141592653589793238462643  int main(int argc, char \*\*argv) {  int intervals;  printf("Number of intervals: ");  fflush(stdout);  scanf("%d", &intervals);  clock\_t time1 = clock();  int i;  double sum, total = 0;  for (i = 0; i < intervals; ++i) {  total += pow(-1, i) / (2 \* i + 1);  }  clock\_t time2 = clock();  total = total \* 4;  printf("Result: %.10lf\n", total);  printf("Accuracy: %.10lf\n", PI - total);  printf("Time: %.10lf\n", (double)(time2 - time1)/CLOCKS\_PER\_SEC);  } |

* 并行代码（MPI实现，pi\_mpi.c）如下：

|  |
| --- |
| /\* File: pi\_mpi.c  \*  \* Purpose: Estimates pi using the Leibniz formula parallelized with MPI  \*  \* Compile: mpicc -o pi\_mpi pi\_mpi.c -lm  \* Run: mpirun -np <number of processes> ./pi\_mpi  \*/  #include <stdio.h>  #include <math.h>  #include <mpi.h>  /\* We define pi here so we can check and see how accurate our computation is. \*/  #define PI 3.141592653589793238462643  int main(int argc, char \*\*argv) {  MPI\_Init(&argc, &argv);  int processes, pe;  MPI\_Comm\_size(MPI\_COMM\_WORLD, &processes);  MPI\_Comm\_rank(MPI\_COMM\_WORLD, &pe);  /\* Let's prompt for the number of intervals. We'll broadcast whatever  \* process 0 reads to the other processes. We could use command line  \* arguments instead, but then there'd be no reason to broadcast! \*/  int intervals;  if (pe == 0) {  printf("Number of intervals: ");  fflush(stdout);  scanf("%d", &intervals);  }  double time1 = MPI\_Wtime();  MPI\_Bcast(&intervals, 1, MPI\_INT, 0, MPI\_COMM\_WORLD);  int count = intervals / processes;  int start = count \* pe;  int end = count \* pe + count;  int i;  double subtotal, total = 0;  for (i = start; i < end; ++i) {  subtotal += pow(-1, i) / (2 \* i + 1);  }  MPI\_Reduce(&subtotal, &total, 1, MPI\_DOUBLE, MPI\_SUM,  0, MPI\_COMM\_WORLD);  double time2 = MPI\_Wtime();  if (pe == 0) {  total = total \* 4;  printf("Result: %.10lf\n", total);  printf("Accuracy: %.10lf\n", PI - total);  printf("Time: %.10lf\n", time2 - time1);  }  MPI\_Finalize();  } |

* 串行代码编译命令：

|  |
| --- |
| gcc -o pi\_c pi\_c.c -lm |

* 并行代码编译命令：

|  |
| --- |
| mpicc -o pi\_mpi pi\_mpi.c -lm |

**要求：**

1. 在集群环境中分别编译和执行串行代码和并行代码，记录执行时间并截图保存。
2. 逐步增加并行代码的进程数，看看时间如何变化，并截图保存。

su - pengjinjie

mkdir /home/pengjinjie/pi

cd /home/pengjinjie/pi

vim pi\_c.c

gcc -o pi\_c pi\_c.c -lm

vim /home/pengjinjie/pi/config

ecs-hw-0001:2

ecs-hw-0002:2

ecs-hw-0003:2

scp -r /home/pengjinjie/pi/ pengjinjie@ecs-hw-0002:/home/pengjinjie/

scp -r /home/pengjinjie/pi/ pengjinjie@ecs-hw-0003:/home/pengjinjie/

./pi\_c 1000 1000000 1000000000

mkdir /home/pengjinjie/pi2

cd /home/pengjinjie/pi2

vim pi\_mpi.c

mpicc -o pi\_mpi pi\_mpi.c -lm

vim /home/pengjinjie/pi2/config

ecs-hw-0001:2

ecs-hw-0002:2

ecs-hw-0003:2

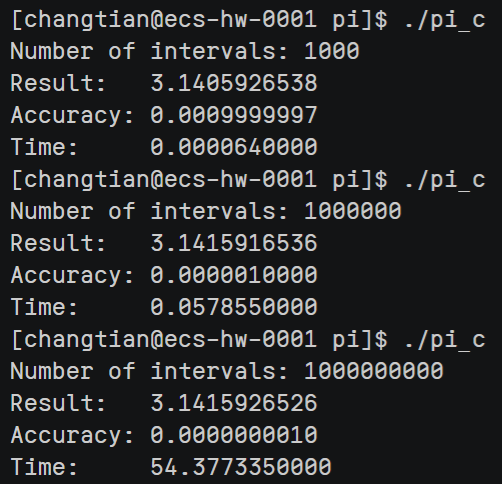
scp -r /home/pengjinjie/pi2/ pengjinjie@ecs-hw-0002:/home/pengjinjie/

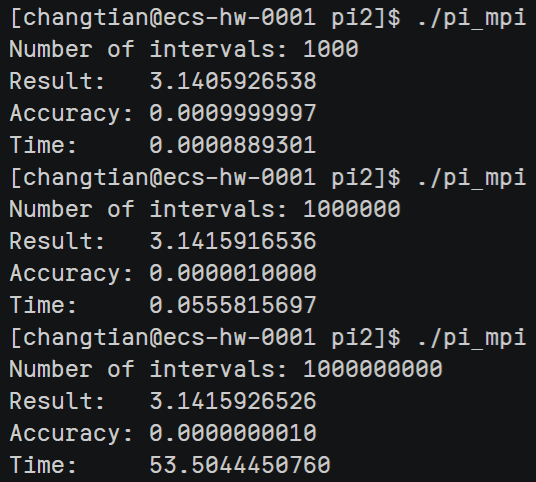
scp -r /home/pengjinjie/pi2/ pengjinjie@ecs-hw-0003:/home/pengjinjie/

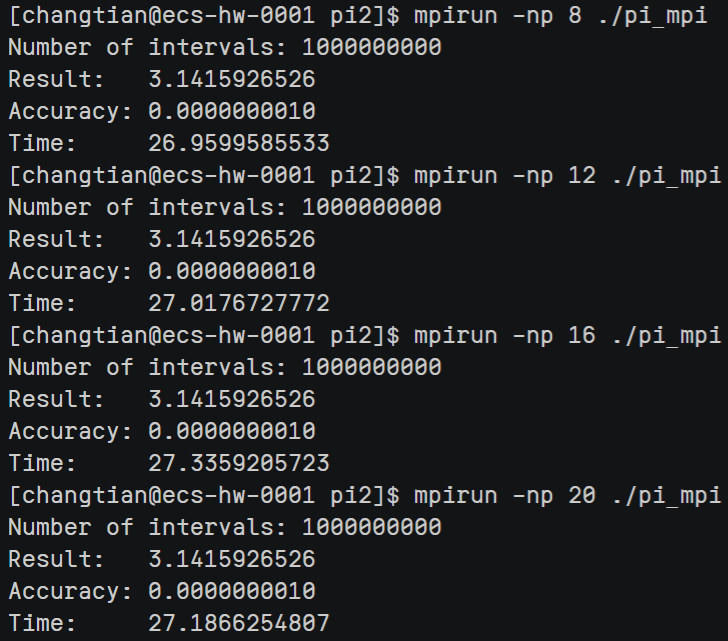
mpirun -np 20 ./pi\_mpi

注意：实验结束后，要删除虚拟机，以免继续扣费。









### 实验评分标准

一、课堂表现（10分）

二、实验结果（50分）

三、实验报告（40分）

### 思考题

思考题1：简述华为云环境下集群的搭建过程。

思考题2：集群之间如果彼此不配置信任秘钥，程序能否正常运行？

思考题3：hello-world程序多次运行，结果一样吗？

思考题4：PI并行代码，增加更多的进程数（超过每个机器的CPU核数），性能如何变化？