

ModelArts

User Guide (Lite Server)

Issue 01
Date 2024-11-19



Copyright © Huawei Cloud Computing Technologies Co., Ltd. 2024. All rights reserved.

No part of this document may be reproduced or transmitted in any form or by any means without prior written consent of Huawei Cloud Computing Technologies Co., Ltd.

Trademarks and Permissions



HUAWEI and other Huawei trademarks are the property of Huawei Technologies Co., Ltd.

All other trademarks and trade names mentioned in this document are the property of their respective holders.

Notice

The purchased products, services and features are stipulated by the contract made between Huawei Cloud and the customer. All or part of the products, services and features described in this document may not be within the purchase scope or the usage scope. Unless otherwise specified in the contract, all statements, information, and recommendations in this document are provided "AS IS" without warranties, guarantees or representations of any kind, either express or implied.

The information in this document is subject to change without notice. Every effort has been made in the preparation of this document to ensure accuracy of the contents, but all statements, information, and recommendations in this document do not constitute a warranty of any kind, express or implied.

Huawei Cloud Computing Technologies Co., Ltd.

Address: Huawei Cloud Data Center Jiaoxinggong Road
Qianzhong Avenue
Gui'an New District
Gui Zhou 550029
People's Republic of China

Website: <https://www.huaweicloud.com/intl/en-us/>

Contents

| | |
|---|-----------|
| 1 Before You Start..... | 1 |
| 1.1 Using Lite Server..... | 1 |
| 1.2 High-Risk Operations..... | 3 |
| 1.3 Mapping Between Compute Resources and Image Versions..... | 4 |
| 2 Enabling Lite Server Resources..... | 10 |
| 3 Configuring Lite Server Resources..... | 19 |
| 3.1 Configuration Process..... | 19 |
| 3.2 Configuring the Network..... | 20 |
| 3.3 Configuring the Storage..... | 23 |
| 3.4 Configuring the Software Environment..... | 26 |
| 3.4.1 Configuring the Software Environment on the NPU Server..... | 27 |
| 3.4.2 Configuring the Software Environment on the GPU Server..... | 41 |
| 4 Using Lite Server Resources..... | 52 |
| 4.1 PyTorch GPU Training and Inference Guide for GPT-2..... | 52 |
| 5 Managing Lite Server Resources..... | 62 |
| 5.1 Viewing Lite Server Details..... | 62 |
| 5.2 Starting or Stopping the Lite Server..... | 63 |
| 5.3 Synchronizing the Lite Server Status..... | 63 |
| 5.4 Changing Lite Server OS..... | 64 |
| 5.5 Monitoring Lite Server Resources..... | 67 |
| 5.5.1 Using CES to Monitor Lite Server Resources..... | 67 |
| 5.5.2 Using DCGM to Monitor Lite Server Resources..... | 70 |
| 5.6 Collecting and Uploading NPU Logs..... | 73 |
| 5.7 Releasing Lite Server Resources..... | 76 |

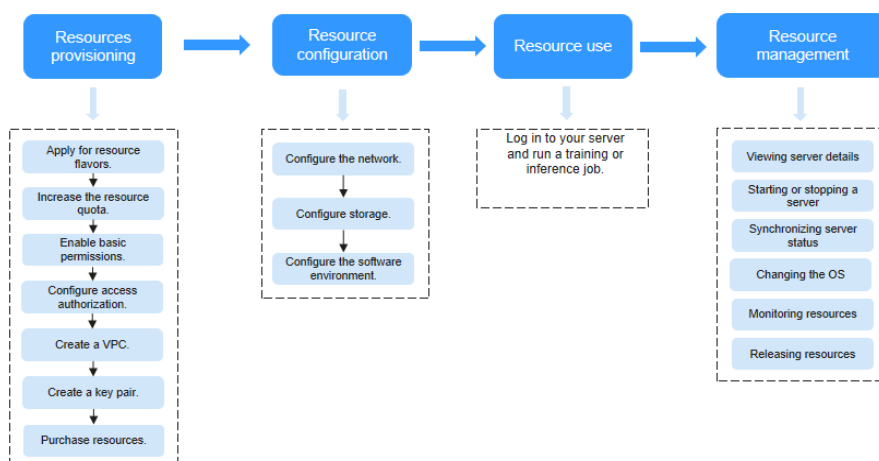
1 Before You Start

1.1 Using Lite Server

ModelArts Lite Server provides various xPU bare metal servers (BMSs), allowing you to install and deploy third-party software such as AI frameworks and applications as user **root**. To create a BMS, you only need to configure the specifications, image, network, and key pairs.

This document helps you understand how to use Lite Server.

Figure 1-1 Process for using Lite Server



1. Purchasing resources

Purchase resources first.

- a. Contact the customer manager to determine a resource solution. Apply for the required resource specifications since certain specifications are restricted.
- b. The default resources provided by Huawei Cloud, such as elastic cloud server (ECS), elastic IP (EIP), and Scalable File Service (SFS), may not

- meet your requirements. Submit a service ticket to increase resource quota.
 - c. Grant the required basic permissions to the IAM user.
 - d. ModelArts needs to access other dependent services. Create an agency for ModelArts.
 - e. When purchasing resources, you need to select a virtual private cloud (VPC) for network communication. You can use an existing VPC or create one.
 - f. When you use key pairs to log in to the BMS, use an existing key pair or create one.
 - g. Purchase resources on the ModelArts console.
2. Configuring resources
Configure the network, storage, and software environment.
 3. Using resources
Log in to the server for model training and inference. For details, see [Using Lite Server Resources](#).
 4. Managing resources
Lite Server allows you to start, stop, and switch OSs. You can manage resources on the ModelArts console.

Table 1-1 Terms

| Term | Description |
|----------|--|
| BMS | <p>Combining VM scalability with physical server performance, BMS provides dedicated cloud servers. These servers are designed to meet the demands of computing performance and data security for core databases, critical applications, high-performance computing (HPC), and big data.</p> <p>After you purchase Lite Server on the ModelArts console, a BMS corresponding to it will be created on the BMS console as Lite Server is a BMS. You can mount disks and bind EIPs on the BMS console.</p> |
| xPU | <p>Graphics processing unit (GPU) or neural network processing unit (NPU).</p> <ul style="list-style-type: none"> • GPUs are used to accelerate the training and inference of deep learning models. • NPUs are used to accelerate neural network computing. Compared with GPUs, NPUs feature higher efficiency and lower power consumption at neural network computing. |
| Key Pair | <p>You can log in to an elastic server using only an SSH key pair. Therefore, you do not need worry about password interception, cracking, and leakage.</p> <p>NOTE To ensure ECS security, private keys that are not managed by Huawei Cloud can be downloaded only once. Keep your downloaded private keys properly.</p> |

| Term | Description |
|------|---|
| VPC | A VPC is a logically isolated, configurable, and manageable virtual network. It helps improve the security of cloud resources and simplifies network deployment. Within your own VPC, you can create security groups and VPNs, configure IP address ranges, and specify bandwidth sizes by customizing security groups, VPNs, IP address ranges, and bandwidth. This simplifies network management. You can also customize access rules to control BMS access within a security group and across different security groups to enhance BMS security. |

1.2 High-Risk Operations

To avoid adverse impacts on ModelArts Lite Server, you must perform high-risk operations according to operation guides during the routine O&M.

Risky operations fall into three levels:

- High: Such operations may cause service failures, data loss, system maintenance failures, and system resource exhaustion.
- Medium: Such operations may cause security risks and reduce service reliability.
- Low: Such operations include high-risk operations other than those of a high or medium risk level.

Table 1-2 High-risk operations

| Object | Operation | Risk | Severity | Solution |
|--------|--|---|----------|--|
| OS | Upgrade or modify the OS kernel or driver. | The driver and kernel versions may not be compatible. As a result, the OS cannot be started or basic functions are unavailable. For example, apt-get upgrade is high-risk command. | High | Contact Huawei technical support. |

| Object | Operation | Risk | Severity | Solution |
|--------|---------------------|--|----------|--|
| | Switch or reset OS. | The EVS system ID is changed. As a result, the EVS system disk cannot be scaled out, and message "The order is expired. The capacity cannot be expanded. Renew the order." is displayed. | Medium | Mount an EVS or SFS disk for capacity expansion. |

1.3 Mapping Between Compute Resources and Image Versions

Lite Server provides multiple NPU and GPU images. You can learn about the supported images and details before purchasing them.

Images Supported by NPU Snt9 BMS

Image name: **ModelArts-Euler2.8_Aarch64_Snt9_C78**

Table 1-3 Image details

| Type | Version Details |
|-------------------|--|
| OS | EulerOS 2.0 (SP8) |
| Kernel version | 4.19.36-vhulk1907.1.0.h619.eulerosv2r8.aarch64 |
| Architecture type | aarch64 |
| mlnx-ofed-linux | 21.0.2 |

Images Supported by NPU Snt9B BMS

- Image name: EulerOS2.10-Arm-64bit-for-Snt9B-BareMetal-with-23.0.6-7.1.0.9.220-CANN7.0.1.5

Table 1-4 Image details

| Type | Version Details |
|------|-----------------|
| OS | EulerOS 2.10 |

| Type | Version Details |
|-----------------------|--|
| Kernel version | Linux 4.19.90-vhulk2211.3.0.h1543.eulerosv2r10.aarch64 |
| Architecture type | aarch64 |
| Firmware version | 7.1.0.9.220 |
| npu-driver | 23.0.6 |
| Ascend-cann-toolkit | 7.0.1.5 |
| cann-kernels | 7.0.1.5 |
| Ascend-mindx-toolbox | 5.0.1.1 |
| Docker | 24.0.7 |
| Ascend-docker-runtime | 5.0.1.1 |
| MindSpore Lite | 2.1.0-cp37-cp37m |
| Mpich | 3.2.1 |

- Image name: HCE2.0-Arm-64bit-for-Snt9B-BareMetal-with-23.0.6-7.1.0.9.220-CANN7.1.0.5

Table 1-5 Image details

| Type | Version Details |
|-----------------------|--|
| OS | HCE2.0 |
| Kernel version | Linux 5.10.0-60.18.0.50.r865_35.hce2.aarch64 |
| Architecture type | aarch64 |
| Firmware version | 7.1.0.9.220 |
| npu-driver | 23.0.6 |
| Ascend-cann-toolkit | 7.0.1.5 |
| cann-kernels | 7.0.1.5 |
| Ascend-mindx-toolbox | 5.0.1.1 |
| Docker | 18.09 |
| Ascend-docker-runtime | 5.0.1.1 |
| MindSpore Lite | 2.1.0-cp37-cp37m |

| Type | Version Details |
|-------|-----------------|
| Mpich | 4.1.3 |

Images Supported by GP Ant8 BMS

- Image name: Ubuntu-20.04-for-Ant8-with-RoCE-and-NVIDIA-525-CUDA-12.0-Uniagent

Table 1-6 Image details

| Type | Version Details |
|--------------------------|--------------------------------|
| OS | Ubuntu 20.04 server 64bit |
| Kernel version | 5.4.0-144-generic |
| Architecture type | x86 |
| Driver version | 525.105.17 |
| cuda | 12.0 |
| container-toolkit | 1.13.3-1 |
| fabricmanager | 525.105.17 |
| mlnx-ofed-linux | 5.8-2.0.3.1-ubuntu20.04-x86_64 |
| peer-memory-dkms | 1.2-0 |
| libnccl2 | 2.18.1 |
| nccl-test | v.2.13.6 |
| docker | 20.10.23 |
| RoCE route configuration | Yes |

- Image name: Ubuntu-20.04-for-Ant8-with-RoCE-and-NVIDIA-515-CUDA-11.7-Uniagent (CN North-Ulanqab1, CN North-Beijing4, and CN North-Ulanqab-Auto1)

Table 1-7 Image details

| Type | Version Details |
|-------------------|---------------------------|
| OS | Ubuntu 20.04 server 64bit |
| Kernel version | 5.4.0-144-generic |
| Architecture type | x86 |
| Driver version | 515.105.01 |

| Type | Version Details |
|--------------------------|--------------------------------|
| cuda | 11.7 |
| container-toolkit | 1.13.3-1 |
| fabricmanager | 515.105.01-1 |
| mlnx-ofed-linux | 5.8-2.0.3.1-ubuntu20.04-x86_64 |
| peer-memory-dkms | 1.2-0 |
| libnccl2 | 2.14.3 |
| nccl-test | v2.13.6 |
| docker | 20.10.23 |
| RoCE route configuration | Yes |

Images Supported by GP Vnt1 BMS

NOTE

For Vnt1, the specifications in CN North-Beijing4, CN North-Beijing1, and CN East-Shanghai1 are the same. However, the product configuration and release time differ. As a result, the images cannot be shared.

- Image name: Ubuntu-18.04-for-BareMetal-Vnt1-p3-with-NVIDIA-470-CUDA-11.4-Uniagent (only for CN North-Beijing1, CN North-Beijing4, and CN South-Guangzhou1)

Table 1-8 Image details

| Type | Version Details |
|-------------------|--------------------------------|
| OS | Ubuntu 18.04 server 64bit |
| Kernel version | 4.15.0-45-generic |
| Architecture type | x86 |
| Driver version | 470.182.03 |
| cuda | 11.4 |
| container-toolkit | 1.15.0.-1 |
| mlnx-ofed-linux | 5.7-1.0.2.1-ubuntu18.04-x86_64 |
| libnccl2 | 2.10.3-1 |
| nccl-test | v2.13.9 |
| docker | 24.0.2 |

- Image name: Ubuntu-18.04-for-BareMetal-Vnt1-p6-with-NVIDIA-470-CUDA-11.4-Uniagent (only for CN East-Shanghai1)

Table 1-9 Image details

| Type | Version Details |
|-------------------|--------------------------------|
| OS | Ubuntu 18.04 server 64bit |
| Kernel version | 4.15.0-45-generic |
| Architecture type | x86 |
| Driver version | 470.182.03 |
| cuda | 11.4 |
| container-toolkit | 1.15.0.-1 |
| mlnx-ofed-linux | 5.7-1.0.2.1-ubuntu18.04-x86_64 |
| libnccl2 | 2.10.3-1 |
| nccl-test | v2.13.9 |
| docker | 24.0.2 |

- Image name: Euler2.9-X86-for-Vnt1-BareMetal (only for CN North-Beijing4 and CN East-Shanghai1)

Table 1-10 Image details

| Type | Version Details |
|-------------------|-------------------|
| OS | EulerOS 2.9 64bit |
| Architecture type | x86 |

- Image name: CentOS-7.9-64bit-for-BareMetal-Vnt1-with-NVIDIA-515-CUDA-11.7-Uniagent (only for CN North-Beijing1, CN North-Beijing4, and CN South-Guangzhou1)

Table 1-11 Image details

| Type | Version Details |
|-------------------|------------------|
| OS | CentOS 7.9 64bit |
| Architecture type | x86 |

Images supported by GPU Ant1 BMS

- Image name: Ubuntu-20.04-x86-for-Ant1-BareMetal-with-RoCE-and-GP-515-CUDA-11.7-AIGC (only for CN North-Beijing4 and CN North-Ulanqab 1)

Table 1-12 Image details

| Type | Version Details |
|--------------------------|---|
| OS | Ubuntu 20.04 server 64bit |
| Architecture type | x86 |
| RoCE route configuration | Automatic configuration is not supported. You need to configure it yourself after the image is created. |

- Image name: Ubuntu-20.04-x86-for-Ant1-BareMetal-with-RoCE-and-NVIDIA-525-CUDA-12.0-AIGC (only for CN North-Ulanqab1)

Table 1-13 Image details

| Type | Version Details |
|--------------------------|---|
| OS | Ubuntu 20.04 server 64bit |
| Architecture type | x86 |
| RoCE route configuration | Automatic configuration is not supported. You need to configure it yourself after the image is created. |

2 Enabling Lite Server Resources

Figure 2-1 Flowchart for enabling resources

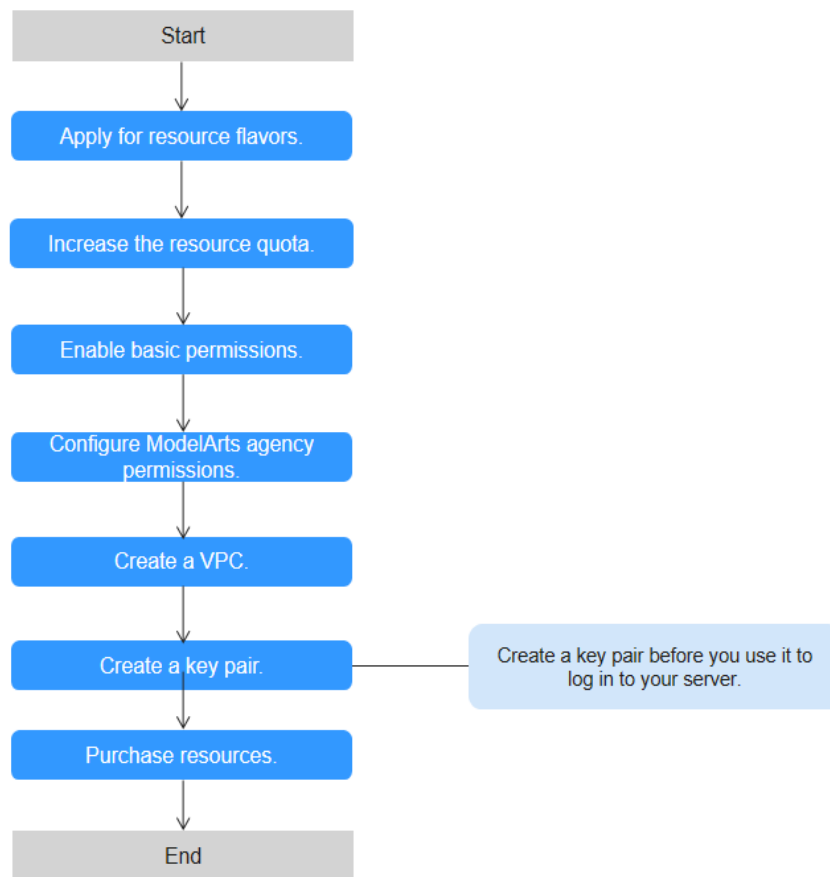


Table 2-1 Enabling resources

| Phase | Task |
|--------------|---------------------------------------|
| Preparations | 1. Apply for resource specifications. |
| | 2. Increase the resource quota. |

| Phase | Task |
|----------------------------------|---|
| | 3. Enable basic permissions. |
| | 4. Configure an agency authorization for ModelArts. |
| | 5. Create a VPC. |
| | 6. (Optional) Create a key pair. (This step is not required if password login is used.) |
| Purchasing Lite Server resources | 7. Purchase a resource pool on the ModelArts console. |

Step 1: Applying for Resource Specifications

Contact Huawei Cloud customer manager to determine a resource solution for Lite Server. Apply for the required resource specifications. If there is no customer manager, submit a service ticket.

Step 2: Increasing Resource Quota

The resources required by the server may exceed the default resources (such as ECS, EIP, and SFS) provided by Huawei Cloud. In this case, you need to increase the resource quota.

Step 1 Log in to Huawei Cloud console.

Step 2 Hover over **Resources** from the top navigation bar and choose **My Quotas**.

Step 3 Click **Increase Quota** in the upper right corner, fill in the materials, and submit a service ticket.

NOTE

Increase the quota before purchasing and provisioning the resource, ensuring it exceeds the resource's requirements.

----End

Step 3: Enabling Basic Permissions

To enable basic permissions, log in to the management console as the administrator account and assign the basic permissions (such as ModelArts FullAccess, BMS FullAccess, ECS FullAccess, VPC FullAccess, VPC Administrator, and VPC Endpoint Administrator) required by server to IAM users.

Step 1 Log in to the IAM console.

Step 2 In the navigation pane on the left, choose **User Groups**, and then click **Create User Group** in the upper right corner.

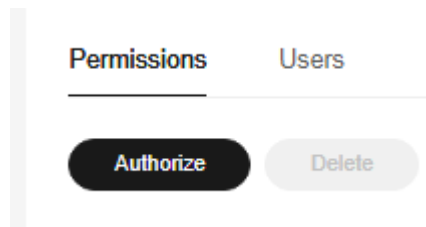
Step 3 Enter the user group name and click **OK**.

Step 4 Click **Manage User** in the **Operation** column and add the target users.

Step 5 Click the user group name to go to the group details page.

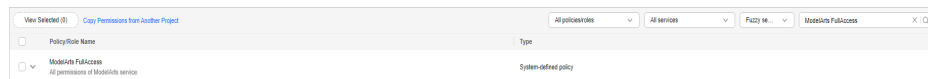
Step 6 In the **Permissions** tab, click **Authorize**.

Figure 2-2 Assigning permissions



Step 7 Enter **ModelArts FullAccess** in the search box and select **ModelArts FullAccess**.

Figure 2-3 ModelArts FullAccess



Use the same method to select **BMS FullAccess**, **ECS FullAccess**, **VPC FullAccess**, **VPC Administrator**, and **VPC Endpoint Administrator**. **Server Administrator** and **DNS Administrator** are dependent policies and are automatically selected.

Step 8 Click **Next** and set **Scope** to **All resources**.

Step 9 Click **OK**.

----End

Step 4: Configuring an Agency Authorization for ModelArts

ModelArts accesses other dependent services. Therefore, you need to configure an agency authorization for ModelArts. For details, see [Assigning Permissions to Individual Users for Using ModelArts](#).

Step 5: Creating a VPC

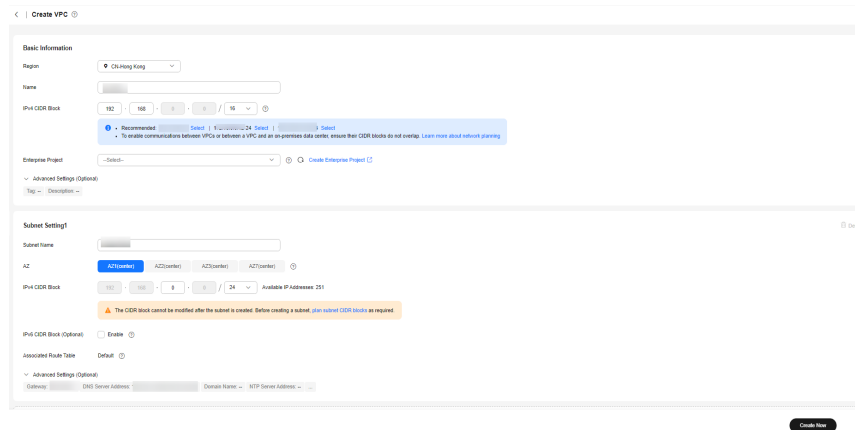
To create a VPC, you need to log in to the management console as the administrator.

Step 1 Log in to the management console.

Step 2 In the service list on the left, choose **Networking** > **Virtual Private Cloud**.

Step 3 Click **Create VPC** in the upper right corner, configure the parameters, and click **Create Now**. For details about the parameters, see [Creating a VPC and Subnet](#).

Figure 2-4 Creating a VPC



----End

Step 6: Creating a Key Pair

NOTE

Key pairs are not required for password login.

- Step 1** Log in to the ModelArts console.
- Step 2** In the navigation pane on the left, choose **AI Dedicated Resource Pools > Elastic Servers**.
- Step 3** Click **Buy Dedicated AI Server**.
- Step 4** Click **Create Key Pair**.
- Step 5** On the displayed page, select **I have read and agree to *Key Pair Service Disclaimer***, and click **OK**. Then, save the key pair to the local host.

----End

Step 7: Purchasing Resources

[Mapping Between Compute Resources and Image Versions](#) lists the supported BMS images.

- Step 1** Log in to the ModelArts console.
- Step 2** In the navigation pane on the left, choose **AI Dedicated Resource Pools > Elastic Servers**.
- Step 3** Click **Buy Dedicated AI Server** and configure the parameters.

Table 2-2 Basic parameters

| Parameter | Description |
|---------------|---|
| Resource Type | <ul style="list-style-type: none"> • BMS: A BMS features both the scalability of Elastic Cloud Servers (ECSs) and high performance of physical servers, providing dedicated servers on the cloud for you and your enterprise. • ECS: ECS provides secure, scalable, on-demand compute resources, enabling you to flexibly deploy applications and workloads. |
| Billing Mode | <p>Select Pay-per-use or Yearly/Monthly.</p> <ul style="list-style-type: none"> • Yearly/Monthly Yearly/Monthly is a prepaid billing mode. You pay in advance for a subscription term, and in exchange, you get a discounted rate. Yearly/Monthly billing is a good option for long-term, stable services. • Pay-per-use Pay-per-use is a postpaid billing mode. You are charged for how long you use each ECS. You can purchase or delete such an ECS at any time. |
| Region | Select a region near you to ensure the lowest latency possible. After the resources are purchased, you can switch the region in the upper left corner of the console to view the resources. |
| AZ | <p>A standalone data center with an independent network and power supply. When deploying resources, consider your applications' requirements on disaster recovery (DR) and network latency.</p> <ul style="list-style-type: none"> • For high DR capability, deploy resources in different AZs within the same region. • For lower network latency, deploy resources in the same AZ. <p>If CloudPond is used, you can view the corresponding edge AZ. Edge AZs deploy cloud infrastructure and services at customer premises. In scenarios where there are high requirements on application access latency, local data retention, and local system interaction, edge AZs ensure easy deployment to the local environment. For details about CloudPond, see What Is CloudPond?</p> |

Table 2-3 Parameters for configuring specifications

| Parameter | Description |
|-------------|---|
| Server Name | Server name, which can contain 1 to 64 characters, including letters, digits, hyphens (-), and underscores (_). |

| Parameter | Description |
|------------------|--|
| CPU Architecture | <p>CPU architecture of the resource type, which can be x86 or Arm. Select the CPU architecture based on the required flavors. If GPUs are used, select x86. If NPUs are used, select Arm. The flavors vary by region. The actual flavors are displayed on the console.</p> <p>NOTE If no flavor is available on the page, contact Huawei Cloud technical support to apply for the flavor.</p> |
| System Disk | <p>This parameter is displayed only when you select a flavor that supports mounting. After an ECS is created, you can mount a data disk to the ECS or expand the capacity of the system disk on the ECS. The recommended value is not smaller than 100 GB.</p> |

Table 2-4 Image parameters

| Parameter | Description |
|-----------|--|
| Image | <ul style="list-style-type: none"> Public image A public image is a standard OS image provided by the system and is available to all users. It contains an OS and pre-installed public applications, such as the SDI iNIC driver, bms-network-config (a network configuration program), and Cloud-Init (an initialization tool). If you need other applications or software, configure them on the new BMSs. ModelArts provides images, which support multiple OSs, built-in drivers and software for AI scenarios, and preset custom ModelArts OS optimization components. For details about supported images, see Mapping Between Compute Resources and Image Versions. Private image A private image is created from an external image file or a BMS and is available only to the user who created it. It contains an OS, preinstalled public applications, and the user's personal applications. You can select a private image to save your time from repeatedly configuring servers. |

Table 2-5 Network parameters

| Parameter | Description |
|-----------|--|
| VPC | <p>The VPC should be the same as that of other cloud services, such as MapReduce Service (MRS) and Cloud Container Engine (CCE) for network interaction.</p> |
| Subnet | <p>Select a subnet of the current VPC.</p> |

| Parameter | Description |
|----------------|---|
| IPv6 Network | <p>IPv6 is available when it is supported by the current subnet, flavors, and images.</p> <p>Ensure that IPv6 has been enabled. For details about enabling IPv6, see Creating a Subnet for the VPC.</p> <p>The requirements on flavors and images differ. Refer to the information displayed on the console.</p> |
| RoCE Network | <p>When GPUs of series A are used, you need to configure the RoCE network to use the RoCE NICs on the hardware during distributed training.</p> <p>This parameter is related to the selected flavors. RoCE network will not be available if any flavor does not support RoCE.</p> <p>If RoCE is supported but has not been created, click Create RoCE.</p> <p>If RoCE is supported and has been created, select an existing one. Repeated creation is not supported.</p> |
| Security Group | <p>A security group is a collection of access control rules for ECSs that have the same security requirements and that are mutually trusted within a VPC.</p> |

Table 2-6 Management parameters

| Parameter | Description |
|------------|--|
| Login Mode | <p>Key pair is recommended because it features higher security than Password. If you select Password, ensure that the password meets complexity requirements to prevent malicious attacks.</p> <ul style="list-style-type: none"> Key pair A key pair is used for BMS login authentication. You can select an existing key pair, or click Create Key Pair to create one. <p>NOTE If you use an existing key pair, ensure that you have one.</p> <ul style="list-style-type: none"> Password The initial password is used for authentication. You can log in to the BMS using the username and its initial password. <p>If the BMS runs Linux, you can use username root and its initial password to log in to the BMS. If the BMS runs Windows, you can use username Administrator and its initial password to log in to the BMS. Password complexity must meet the following requirements:</p> <ul style="list-style-type: none"> Contains 8 to 26 characters. Contains at least three types of the following characters: uppercase letters, lowercase letters, digits, and special characters (!@\$%^_-=+[]{};./?). The password should be different from the username or the username spelled backwards. Cannot contain root, administrator, or their reverse. |

Table 2-7 Advanced settings

| Parameter | Description |
|--------------------|--|
| Enterprise Project | <p>This parameter is available only if you have enabled enterprise projects or your account is an enterprise account. You can contact your service manager to enable this function</p> <p>An enterprise project groups cloud resources, so you can manage resources and members by project. The default project is default.</p> <p>Select an enterprise project from the drop-down list. For details about enterprise projects, see Enterprise Management User Guide.</p> |

Table 2-8 Purchase parameters

| Parameter | Description |
|-----------|---|
| Quantity | You can purchase multiple instances simultaneously, with a value between 1 and 10 . Each instance generates a separate order, which must be paid for individually. |

Step 4 Click **Create Now**.

Step 5 Pay for the order.

 **NOTE**

Each instance generates a separate order, which must be paid for individually.

Step 6 Once paid, the resource will be created in 20 to 60 minutes. Wait until the resource is created.

 **NOTE**

- If ModelArts fails to create an elastic server, there are multiple possible causes. The following provides several possibilities for quick troubleshooting.
 - Insufficient resources: Switch to the BMS page and check whether the specifications to be purchased are sold out. If so, there are no resources of this flavor. In this case, contact the customer manager to obtain resources and purchase again.
 - Insufficient quota: Check whether the resource quota of the account is sufficient. If the resource quota of the account, including the number of cores and RAM, is insufficient, the creation will fail. In this case, apply for a quota before purchasing the resource.
 - Internal BMS error: Check whether there is an internal BMS error. If yes, submit a service ticket to BMS to locate and rectify the fault.
- If a container is used or shared by multiple users, you should restrict the container from accessing the OpenStack management address (169.254.169.254) to prevent host machine metadata acquisition. For details, see [Forbidding Containers to Obtain Host Machine Metadata](#).

----End

3 Configuring Lite Server Resources

3.1 Configuration Process

After enabling Lite Server resources, you need to complete the configurations as follows.

Figure 3-1 Flowchart for configuring resources

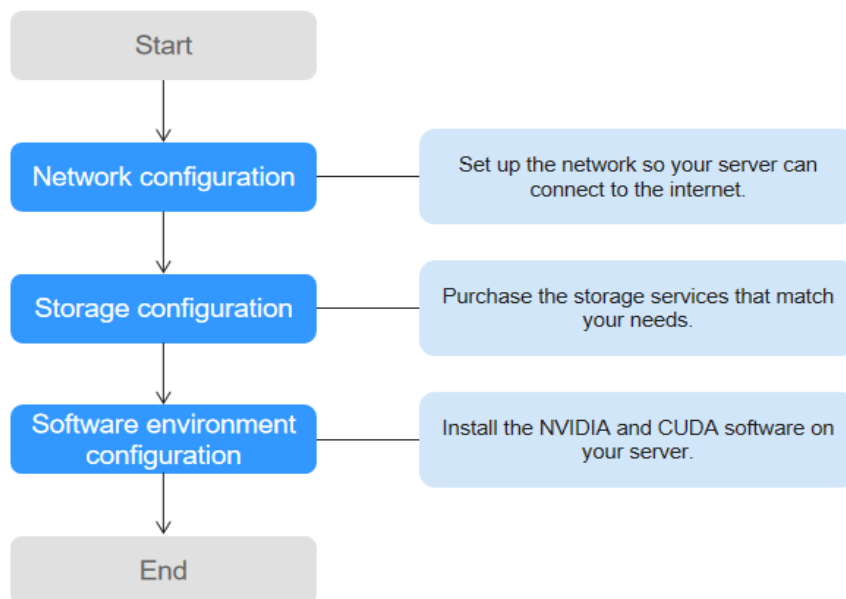


Table 3-1 Resource configuration process

| Step | Task | Description |
|------|--|--|
| 1 | Configuring the Network | Configure the network so that the Lite Server can communicate with the Internet. Before configuring the storage and software environment, ensure that the server can access the network. |
| 2 | Configuring the Storage | Data disks need to be mounted to the server to store data files. Currently, SFS, Object Storage Service (OBS), and EVS are supported. |
| 3 | Configuring the Software Environment | The software to be pre-installed varies among images. View the installed software by referring to Mapping Between Compute Resources and Image Versions . If the software pre-installed on the Lite Server cannot meet service requirements, you can configure the required software environment on the server. |

3.2 Configuring the Network

Configure the network so that Lite Server can communicate with the Internet. Network configuration involves the following two scenarios:

- **Binding an EIP to a Single Server:** Bind an EIP to a single server. The server exclusively uses network resources.
- **Binding an EIP to Multiple Servers:** An EIP is configured for a VPC. All servers in the VPC can access the Internet through the EIP and share network resources.

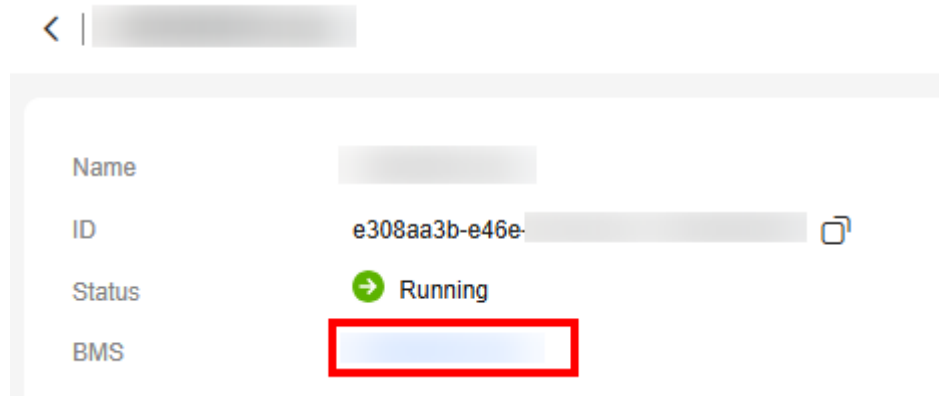
Binding an EIP to a Single Server

Step 1 Log in to the ModelArts management console.

Step 2 In the navigation pane on the left, choose **AI Dedicated Resource Pools > Elastic Servers**.

Step 3 Click the server name to go to the details page. On the displayed page, click the BMS name.

Figure 3-2 BMS

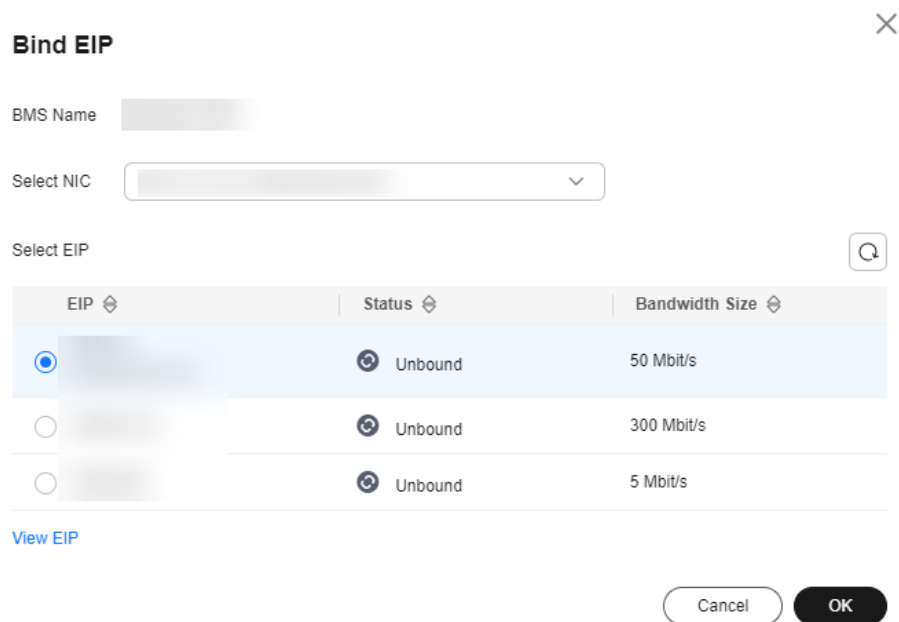


Step 4 Click the **EIPs** tab and then click **Bind EIP**.

The **Bind EIP** dialog box is displayed.

Select the EIP to be bound and click **OK**.

Figure 3-3 Binding an EIP



NOTE

Only one EIP can be bound to a NIC.

----End

Binding an EIP to Multiple Servers

NOTE

All servers must be deployed in the same VPC which does not have a NAT gateway or default route.

Step 1 Buy an EIP.

1. Log in to Huawei Cloud console.
2. In the service list on the left, choose **Networking > Elastic IP**.
3. Click **Buy EIP**.
4. Retain the default settings and click **Next**.
5. Confirm the configurations and select "I have read and agree to the *Elastic IP Service Statement*".
 - If you set **Billing Mode** to **Pay-per-Use**, click **Submit**.
 - If you set **Billing Mode** to **Yearly/Monthly**, click **Pay Now**.On the payment page, confirm the order information, and click **Pay**.

Step 2 Buy a public NAT gateway.

1. Log in to Huawei Cloud console.
2. In the service list on the left, choose **Network > NAT Gateway**.
3. Click **Buy Public NAT Gateway**.
4. Set **VPC**, **Subnet**, and **Billing Mode**. Retain the default settings for other parameters and click **Next**.
5. On the displayed page, confirm the information and click **Submit**.
 - If you set **Billing Mode** to **Pay-per-Use**, click **Submit**.
 - If you set **Billing Mode** to **Yearly/Monthly**, click **Pay Now**.On the payment page, confirm the order information, and click **Pay**.

NOTE

The VPC and subnet must be that of the server.

Step 3 Add an SNAT rule.

SNAT translates private IP addresses into EIPs, allowing servers in a VPC to share an EIP to access the Internet securely and efficiently.

1. On the **Public NAT Gateways** page, click the name of the created NAT gateway.
2. In the **SNAT Rules** tab, click **Add SNAT Rule**.
3. In the displayed dialog box, configure the SNAT rule as follows:
 - **Scenario**: Select **VPC**.
 - **Subnet**: Choose an existing subnet.
 - **EIP**: Select a created EIP.
4. Click **OK**.

Step 4 Configure a DNAT rule.

By adding a DNAT rule, the servers in a VPC can access services using SSH. For each server, a port corresponds to a DNAT rule, and one port can be mapped to only one EIP.

1. In the **DNAT Rules** tab, click **Add DNAT Rule**.
2. In the displayed dialog box, configure the DNAT rule as follows:
 - **Scenario:** Select **VPC**.
 - **Port Type:** Select **Specific port**.
 - **Protocol:** Select **TCP**.
 - **EIP:** Select a created EIP.
 - **Outside Port:** The recommended value ranges from **20000** to **30000**. Ensure that the port number is unique.
 - **Instance Type:** Click **Server** and select a server.
 - **NIC:** Select a NIC.
 - **Inside Port:** Choose **22**.
3. Click **OK**.

----End

3.3 Configuring the Storage

Currently, SFS, OBS, and EVS are supported for the server. The following table describes the differences of storage solutions. For details about how to configure local disks, see [Physical Machine Environment Configuration](#).

Table 3-2 Comparison among SFS, OBS, and EVS

| Dimension | SFS | OBS | EVS |
|-----------|---|---|---|
| Concept | SFS provides on-demand high-performance file storage, which can be shared by multiple cloud servers. SFS is similar to a remote directory for Windows or Linux OSs. | OBS provides massive, secure, reliable, and cost-effective data storage for users to store data of any type and size. | EVS provides scalable block storage that features high reliability, high performance, and a variety of specifications for cloud servers to meet service requirements in different scenarios. An EVS disk is similar to a hard disk on a PC. |

| Dimension | SFS | OBS | EVS |
|--------------------|--|---|---|
| Data storage logic | SFS stores files and organizes them in a directory hierarchy. | OBS stores data as objects with metadata and unique identifiers. You can upload files directly to OBS. The system can generate metadata for files, or you can customize the metadata for files. | EVS stores binary data and cannot store files directly. To store files on an EVS disk, you need to format the file system first. |
| Access method | SFS file systems can be accessed only after being mounted to ECSs or BMSs through NFS or CIFS. You need to specify a network address or map it to a local directory for access. | Accessible through the Internet or Direct Connect (DC). You need to specify the bucket address for access and use transmission protocols such as HTTP and HTTPS. | EVS disks can be used and accessed from applications only after being attached to ECSs or BMSs and formatted. |
| Scenario | High-performance computing (HPC), media processing, file sharing, content management, and web services NOTE HPC: High bandwidth is required for shared file storage, such as gene sequencing and image rendering. | Big data analysis, static website hosting, online video on demand (VoD), gene sequencing, and intelligent video surveillance | HPC, enterprise core cluster applications, enterprise application systems, and development and testing NOTE HPC: High-speed and high-IOPS storage is required, such as industrial design and energy exploration. |
| Capacity | PB | EB | TB |
| Latency | 3–10 ms | 10 ms | Sub-millisecond |
| IOPS/TPS | 10,000 for a single file system | Tens of millions | 128,000 for a single disk |
| Bandwidth | GB/s | TB/s | MB/s |
| Data sharing | Yes | Yes | Yes |
| Remote access | Yes | Yes | No |

| Dimension | SFS | OBS | EVS |
|--------------------|-----|-----|---|
| Online editing | Yes | No | Yes |
| Used independently | Yes | Yes | No (EVS must work with BMS to store files.) |

Using SFS for Storage

Use SFS Turbo file system. SFS Turbo provides high-performance file storage on demand. It features high reliability and availability. It can be elastically expanded and performs better as its capacity grows. The service is suitable for a wide range of scenarios.

- Step 1** Create a file system on the SFS console. For details, see [Creating an SFS Turbo File System](#). File systems and ECSs in different AZs of the same region can communicate with each other. Therefore, ensure that SFS Turbo and the server are in the same region.
- Step 2** Mount the file system to the server. For details, see [Mounting an NFS File System to ECSs \(Linux\)](#).
- Step 3** Set automatic mounting upon restart on the server to prevent mounting loss. For details, see [Mounting a File System Automatically](#).

----End

Using OBS for Storage

Use the combination of parallel file systems and obsutil. The parallel file system provided by OBS is an optimized high-performance file semantic system with millisecond-level access latency, TB/s-level bandwidth, and million-level IOPS. You can use obsutil, a command line tool for accessing OBS, to perform configurations in OBS, for example, creating a bucket, uploading, downloading, and deleting files/folders. If you are familiar with command line interface (CLI), obsutil can provide you with better experience in batch processing and automated tasks.

- Step 1** Create a parallel file system on the OBS console. For details, see [Creating a Parallel File System](#).
- Step 2** Download the corresponding obsutil to the BMS based on your OS and install it. For details, see [Downloading and Installing obsutil](#).
- Step 3** Configure the OBS endpoint and AK/SK for obsutil to interconnect with OBS. You can use obsutil to perform operations on OBS buckets and objects only after obtaining the OBS authentication. For details, see [Performing the Initial Configuration](#).
- Step 4** Use obsutil to upload and download OBS files in the BMS. For details about obsutil, see [obsutil Introduction](#).

----End

Using EVS for Storage

Step 1 Purchase a disk on the EVS console, choose the AZ where the BMS is deployed, set **Attach to Server** to **Later**, choose a billing mode, and configure the disk size based on your needs. For details, see [Purchasing an EVS Disk](#).

Figure 3-4 Buying a disk

The screenshot shows the configuration interface for purchasing an EVS disk. It includes the following elements:

- Region:** A dropdown menu set to "CN Southwest-Guiyang1". Below it is a note: "Regions are geographic areas isolated from each other. Resources are region-specific and cannot be used across regions through internal network connections. For low network latency and quick resource access, select the nearest region."
- AZ:** Three buttons: "AZ4" (highlighted in blue), "AZ5 (1)", and "AZ1". Below them is a note: "No server is available in the current AZ. Select the AZ where your server resides. The AZ cannot be changed after the disk is created."
- Attach To Server:** Two buttons: "Now" and "Later" (highlighted in blue and enclosed in a red box).
- Billing Mode:** Two buttons: "Yearly/Monthly" and "Pay-per-use" (highlighted in blue and enclosed in a red box).

NOTE

Currently, an EVS disk cannot be mounted to the cloud server when it is being created. In this case, the system displays a message indicating that the Yearly/Monthly ECS has not been synchronized to the operation system. Please try again later. To address such problem, log in to the Huawei Cloud console, and choose **Billing** from the top menu bar. In the navigation pane on the left, choose **Orders > Renewals**. On the displayed page, check whether the ECS has been synchronized to the OS. If yes, set **Attach to Server** to **Later**.

Step 2 Go to the BMS details page of the server, click **Mount Disk**, and select the purchased EVS disk.

Figure 3-5 Mounting a disk

The screenshot shows the "Mount Disk" interface in the BMS console. It includes the following elements:

- Navigation tabs: "Disks", "RAID", "NICs", "Security Groups", "EIPs", "Monitoring", and "Tags".
- Attach Disk:** A button with a tooltip: "You can attach 59 more disks. If your BMS uses a private or shared image, ensure that the storage driver has been integrated into the image."
- System Disk:** A dropdown menu showing "System Disk | 200GiB".

NOTE

The mounted EVS disk will not be automatically deleted when you unsubscribe from a BMS. You can mount the disk to other BMSs or delete it as needed.

----End

3.4 Configuring the Software Environment

3.4.1 Configuring the Software Environment on the NPU Server

Scenario

This section describes how to configure the environment on an Snt9b BMS, including merging and mounting disks and installing Docker. Pay attention to the following before the configuration:

- During the the first installation, once you have configured the basic information such as storage, firmware, driver, and network access, try not to make any changes.
- For developers who need to develop on a BMS, start an independent Docker container as your personal development environment. The Snt9b BMS contains eight-card compute resources, which can be used by multiple users for development and debugging. To avoid usage conflicts, cards should be arranged to each user beforehand, and users should develop in their own Docker containers.
- ModelArts provides standard base container images, in which the basic MindSpore/PyTorch framework and the development and debugging tool chain are preset. You can use the image directly. Alternatively, you can use your own service images or images provided by AscendHub. If the software version preset in the image does not meet your requirements, you can install and replace it.
- Use the exposed SSH port to connect to the container in remote development mode (VSCode SSH Remote or Xshell) for development. You can mount your storage directory to the container to store code and data.

NOTE

Most operations in this guide have been preset in the latest Snt9b BMS environment. No further configuration is required. Skip the step if it has been preset.

Physical Machine Environment Configuration

Step 1 Configure timeout parameters.

Log in to the server using SSH and check the timeout configuration.

```
echo $TMOUT
```

If it is set to **300**, the server is disconnected after 5 minutes. You can configure the parameter to set a longer timeout interval. If it is set to **0**, skip this step. Run the following commands to configure the parameter:

```
vim /etc/profile
# Change the value of TMOUT from 300 to 0 at the end of the file. The value 0 indicates that the idle
connection is not disconnected.
export TMOUT=0
```

Run the following command for the configuration to take effect on the current terminal:

```
TMOUT=0
```

Step 2 Merge and mount disks.

After a BMS is purchased, there might be multiple unmounted nvme disks on the server. Before configuring the environment, you need to merge and mount the disks. This operation must be performed at the very first so that the content you stored will not be overwritten.

1. Run the **lsblk** command to check whether three 7-TB disks are not mounted. In the following figure, nvme0n1, nvme1n1, and nvme2n1 are not mounted.

Figure 3-6 Unmounted disks

```
[root@devserver-7354 ~]# lsblk
NAME        MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
sda          8:0    0  150G  0 disk
├─sda1       8:1    0    1G  0 part /boot/efi
└─sda2       8:2    0  149G  0 part /
nvme0n1     259:0    0    7T  0 disk
nvme1n1     259:1    0    7T  0 disk
nvme2n1     259:2    0    7T  0 disk
[root@devserver-7354 ~]#
```

2. The **MOUNTPOINT** column indicates the directory where the disk is mounted, as shown in the following figure. You can skip this step and create a directory in **/home**.

Figure 3-7 Mounted disks

```
[root@devserver-7354 ~]# lsblk
NAME        MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
sda          8:0    0  150G  0 disk
├─sda1       8:1    0    1G  0 part /boot/efi
└─sda2       8:2    0  149G  0 part /
nvme0n1     259:0    0    7T  0 disk /home
nvme1n1     259:1    0    7T  0 disk
├─nvme_group-docker_data 253:0    0   14T  0 lvm /docker
nvme2n1     259:2    0    7T  0 disk
└─nvme_group-docker_data 253:0    0   14T  0 lvm /docker
```

Run the automatic mounting script to mount **/dev/nvme0n1** to **/home** for each developer to create their own home directory. Mount the other two disks to **/docker** for containers to use. If **/docker** does not have sufficient space, the root directory may be fully occupied when multiple container instances are created.

```
cd /root/tools/
sh create_disk_partitions.sh
```

After the configuration, run the **df -h** command to view the information about the mounted disks.

Figure 3-8 Viewing mounted disks

```
[root@devserver-modelarts home]# df -h
Filesystem      Size  Used Avail Use% Mounted on
devtmpfs        756G     0  756G   0% /dev
tmpfs           756G     0  756G   0% /dev/shm
tmpfs           756G   28M  756G   1% /run
tmpfs           756G     0  756G   0% /sys/fs/cgroup
/dev/sda2       196G   2.4G  185G   2% /
tmpfs           756G   40K  756G   1% /tmp
/dev/sda1       1022M  8.3M  1014M   1% /boot/efi
/dev/mapper/nvme_group-docker_data 14T  121G   14T   1% /docker
/dev/nvme0n1    7.0T   50G   7.0T   1% /home
```

After the disks are merged and mounted, you can create a working directory and name it in **/home**.

Step 3 (Optional) Install the firmware and driver.

1. View the environment information. Run the following command to view the current firmware and driver versions:

```
npu-smi info -t board -i 1 | egrep -i "software|firmware"
```

Figure 3-9 Viewing the firmware and driver versions

```
[root@devserver-com ~]# npu-smi info -t board -i 1 | egrep -i "software|firmware"
Software Version      : 23.0.rc3
Firmware Version     : 6.4.0.4.220
```

The above figure shows an example of the latest Ascend commercial version. You can skip the step for installing the firmware and driver in this section.

If the current versions do not meet your requirements and need to be changed, see the subsequent operations.

2. View the OS version, check whether the architecture is AArch64 or x86_64, and obtain the firmware and driver packages from the Ascend official website. The firmware package is **Ascend-hdk-Model-npu-firmware_Version.run** and the driver package is **Ascend-hdk-Model-npu-driver_Version_linux-aarch64.run**. Only Huawei engineers and channel users have the permission to download the commercial version. For details, see [Ascend HDK 23.0.RC3](#).

```
arch
cat /etc/os-release
```

Figure 3-10 Viewing the OS version and architecture

```
[root@localhost ~]# arch
aarch64
[root@localhost ~]# cat /etc/os-release
NAME="EulerOS"
VERSION="2.0 (SP10)"
ID="euleros"
VERSION_ID="2.0"
PRETTY_NAME="EulerOS 2.0 (SP10)"
ANSI_COLOR="0;31"
```

The following uses the packages that adapt to EulerOS 2.0 (SP10) and AArch64 as an example.

3. Install the firmware and driver packages.
 - a. Check whether the npu-smi tool functions, which is vital for subsequent firmware and driver installation. Run the **npu-smi info** command. If the content in the following figure is displayed, npu-smi functions properly. If the content in the following figure is not displayed, for example, an error is reported or only part of the content is displayed, submit a service ticket to [contact Huawei technical support](#) to restore npu-smi. Then, you can install the firmware and driver of the new version.

Figure 3-11 Checking the npu-smi tool

```
[root@devserver-bms-fd775372-833351 ~]# npu-smi info
-----
npu-smi 23.0.rc3                               Version: 23.0.rc3
-----
| NPU  Name      | Health  | Power(W) | Temp(C) | Hugepages-Usage(page) |
| Chip  | Bus-Id  | AICore(%) | Memory-Usage(MB) | HBM-Usage(MB) |
-----+-----+-----+-----+-----+-----+
| 0     910B2    | OK      | 92.7     | 49      | 0 / 0                 |
| 0     0000:C1:00.0 |         | 0        | 0 / 0   | 4152 / 65536          |
-----+-----+-----+-----+-----+
| 1     910B2    | OK      | 87.0     | 52      | 0 / 0                 |
| 0     0000:01:00.0 |         | 0        | 0 / 0   | 4152 / 65536          |
-----+-----+-----+-----+-----+
| 2     910B2    | OK      | 94.5     | 53      | 0 / 0                 |
| 0     0000:C2:00.0 |         | 0        | 0 / 0   | 4152 / 65536          |
-----+-----+-----+-----+-----+
| 3     910B2    | OK      | 92.9     | 51      | 0 / 0                 |
| 0     0000:02:00.0 |         | 0        | 0 / 0   | 4152 / 65536          |
-----+-----+-----+-----+-----+
| 4     910B2    | OK      | 91.9     | 53      | 0 / 0                 |
| 0     0000:81:00.0 |         | 0        | 0 / 0   | 4152 / 65536          |
-----+-----+-----+-----+-----+
| 5     910B2    | OK      | 93.1     | 54      | 0 / 0                 |
| 0     0000:41:00.0 |         | 0        | 0 / 0   | 4153 / 65536          |
-----+-----+-----+-----+-----+
| 6     910B2    | OK      | 92.2     | 52      | 0 / 0                 |
| 0     0000:82:00.0 |         | 0        | 0 / 0   | 4153 / 65536          |
-----+-----+-----+-----+-----+
| 7     910B2    | OK      | 92.2     | 54      | 0 / 0                 |
| 0     0000:42:00.0 |         | 0        | 0 / 0   | 4153 / 65536          |
-----+-----+-----+-----+-----+
| NPU  Chip      | Process id | Process name | Process memory(MB) |
-----+-----+-----+-----+-----+
| No running processes found in NPU 0 |
-----+-----+-----+-----+
| No running processes found in NPU 1 |
-----+-----+-----+-----+
| No running processes found in NPU 2 |
-----+-----+-----+-----+
| No running processes found in NPU 3 |
-----+-----+-----+-----+
| No running processes found in NPU 4 |
-----+-----+-----+-----+
| No running processes found in NPU 5 |
-----+-----+-----+-----+
| No running processes found in NPU 6 |
-----+-----+-----+-----+
| No running processes found in NPU 7 |
-----+-----+-----+-----+

```

- b. After the tool is verified to be normal, install the firmware and driver.

 NOTE

1. Installation sequence is vital.
 1. Initial installation: In scenarios where no driver is installed on a hardware device before delivery, or the installed driver and firmware on the hardware device have been uninstalled, you need to install the driver and then firmware.
 2. Overwrite installation: In scenarios where the driver and firmware have been installed on a hardware device and you need to install them again, install the firmware first and then the driver.

Generally, the firmware and driver are pre-installed on the Snt9b server before delivery. So in this case, **overwrite installation** is used as an example.

- i. If the firmware and driver to be installed are of a lower version, ensure that npu-smi functions, and install the packages without the need to uninstall the existing versions.
- ii. If the firmware and driver fail to be installed, search for the solution in Developer Community based on the error message.

The installation commands are as follows.

- i. Install the firmware and then restart the server.

```
chmod 700 *.run
# Actual package name
```

- ```
./Ascend-hdk-Model-npu-firmware_Version.run --full
reboot
```
- ii. Install the driver and enter **y** as prompted. You do not need to restart the server.
 

```
Actual package name
./Ascend-hdk-Model-npu-driver_Version_linux-aarch64.run --full --install-for-all
```
  - iii. After the installation, check the firmware and driver versions. If the output is normal, the installation is successful.
 

```
npu-smi info -t board -i 1 | egrep -i "software|firmware"
```

**Figure 3-12** Checking the firmware and driver versions

```
[root@devserver-com ~]# npu-smi info -t board -i 1 | egrep -i "software|firmware"
Software Version : 23.0.rc3
Firmware Version : 6.4.0.4.220
```

#### Step 4 Install the Docker environment.

1. Run the **docker -v** command to check whether Docker has been installed. If yes, skip this step.

Run the following command to install Docker:

```
yum install -y docker-engine.aarch64 docker-engine-selinux.noarch docker-runc.aarch64
```

Run the **docker -v** command to check whether the installation is successful.

**Figure 3-13** Viewing the Docker version

```
[root@localhost ~]# docker -v
Docker version 18.09.0, build ba6df24
```

2. Configure IP forwarding for network access in containers.
 

Run the following command to check the value of **net.ipv4.ip\_forward**. Skip this step if the value is **1**.

```
sysctl -p | grep net.ipv4.ip_forward
```

If the value is not **1**, run the following command to configure IP forwarding:

```
sed -i 's/net.ipv4.ip_forward=0/net.ipv4.ip_forward=1/g' /etc/sysctl.conf
sysctl -p | grep net.ipv4.ip_forward
```

3. Check whether Ascend-docker-runtime has been installed and configured in the environment.

```
docker info |grep Runtime
```

If the runtime is **ascend** in the output, the installation and configuration are complete. In this case, skip this step.

**Figure 3-14** Querying Ascend-docker-runtime

```
[root@devserver-modelarts-demanager-0eaabe8f ~]# docker info |grep Runtime
Runtimes: ascend runc
Default Runtime: ascend
```

If Ascend-docker-runtime is not installed, click [here](#) to install it. The software package is a Docker plug-in provided by Ascend. During Docker runtime, paths of Ascend drivers can be automatically mounted to the container. You do not need to specify **--device** during container startup. After the package is downloaded, upload it to the server and install it.

```
chmod 700 *.run
./Ascend-hdk-Model-npu-driver_Version_linux-aarch64.run --install
```

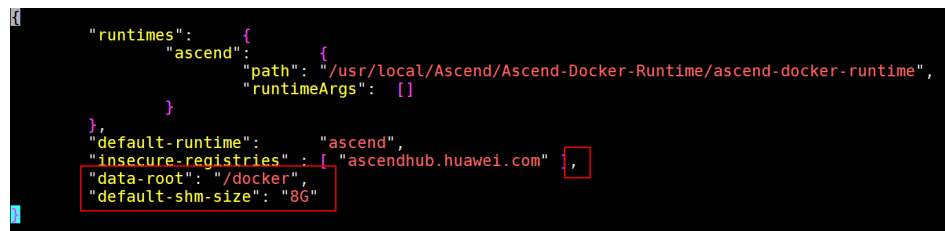
For details, see [Ascend Docker Runtime](#).

4. Set the newly mounted disk as the path used by Docker containers.

Edit the `/etc/docker/daemon.json` file. If the file does not exist, create it.  
`vim /etc/docker/daemon.json`

Add the two configurations as shown in the following figure. To ensure that the JSON format is correct, add a comma (,) at the end of the **insecure-registries** line. **data\_root** indicates the path where the Docker data is stored. **default-shm-size** indicates the default sharing size during container startup. The default value is **64 MB**. You can modify it in case the training fails due to insufficient sharing memory during distributed training.

**Figure 3-15** Docker configuration



Save the configuration and run the following command to restart Docker for the configuration to take effect:

```
systemctl daemon-reload && systemctl restart docker
```

#### Step 5 (Optional) Install pip.

1. Check whether pip has been installed and whether the access to the pip source is normal. If yes, skip this step.

```
pip install numpy
```

2. If pip is not installed, run the following commands:

```
python -m ensurepip --upgrade
ln -s /usr/bin/pip3 /usr/bin/pip
```

3. Configure the pip source.

```
mkdir -p ~/.pip
vim ~/.pip/pip.conf
```

Add the following information to the `~/.pip/pip.conf` file:

```
[global]
index-url = http://mirrors.myhuaweicloud.com/pypi/web/simple
format = columns
[install]
trusted-host=mirrors.myhuaweicloud.com
```

#### Step 6 Test the RoCE network.

1. Install CANN Toolkit.

Check whether CANN Toolkit has been installed on the server. If the version number is displayed, it has been installed.

```
cat /usr/local/Ascend/ascend-toolkit/latest/aarch64-linux/ascend_toolkit_install.info
```

If it is not installed, obtain the software package from the official website. For common users, [download the community edition](#). For Huawei engineers and channel users, [download the commercial edition](#).

Install CANN Toolkit. Replace the package name.

```
chmod 700 *.run
./Ascend-cann-toolkit_6.3.RC2_linux-aarch64.run --full --install-for-all
```

2. Install mpich-3.2.1.tar.gz.

Click [here](#) to download the package and run the following commands to install it:

```
mkdir -p /home/mpich
mv /root/mpich-3.2.1.tar.gz /home/
cd /home/;tar -zxvf mpich-3.2.1.tar.gz
cd /home/mpich-3.2.1
./configure --prefix=/home/mpich --disable-fortran
make && make install
```

3. Set environment variables and compile the HCCL operator.

```
export PATH=/home/mpich/bin:$PATH
cd /usr/local/Ascend/ascend-toolkit/latest/tools/hccl_test
export LD_LIBRARY_PATH=/home/mpich/lib:/usr/local/Ascend/ascend-toolkit/latest/lib64:$LD_LIBRARY_PATH
make MPI_HOME=/home/mpich ASCEND_DIR=/usr/local/Ascend/ascend-toolkit/latest
```

After the operator is compiled, the following information is displayed.

**Figure 3-16** Compiled operator

```
[root@devserver-com hccl_test]# make MPI_HOME=/home/mpich ASCEND_DIR=/usr/local/Ascend/ascend-toolkit/latest
g++ -std=c++11 -Werror -fstack-protector-strong -fPIE -pie -O2 -s -Wl,-z,relro -Wl,-z,now -Wl,-z,noexecstack -Wl,--copy-dt-needed-entries ./common/src/hccl_check_buf_init.cc ./common/src/hccl_check_common.cc ./common/src/hccl_opbase_rootinfo_base.cc ./common/src/hccl_test_common.cc ./common/src/hccl_test_main.cc ./opbase_test/hccl_allgather_rootinfo_test.cc -I./common/src -I/usr/local/Ascend/ascend-toolkit/latest/include -I/usr/local/Ascend/ascend-toolkit/latest/include -I/home/mpich/include -I./opbase_test -o all_gather_test -L/usr/local/Ascend/ascend-toolkit/latest/lib64 -lhccl -L/usr/local/Ascend/ascend-toolkit/latest/lib64 -lascendcl -L/home/mpich/lib -lmpi
all_gather_test compile completed
g++ -std=c++11 -Werror -fstack-protector-strong -fPIE -pie -O2 -s -Wl,-z,relro -Wl,-z,now -Wl,-z,noexecstack -Wl,--copy-dt-needed-entries ./common/src/hccl_check_buf_init.cc ./common/src/hccl_check_common.cc ./common/src/hccl_opbase_rootinfo_base.cc ./common/src/hccl_test_common.cc ./common/src/hccl_test_main.cc ./opbase_test/hccl_allreduce_rootinfo_test.cc -I./common/src -I/usr/local/Ascend/ascend-toolkit/latest/include -I/usr/local/Ascend/ascend-toolkit/latest/include -I/home/mpich/include -I./opbase_test -o all_reduce_test -L/usr/local/Ascend/ascend-toolkit/latest/lib64 -lhccl -L/usr/local/Ascend/ascend-toolkit/latest/lib64 -lascendcl -L/home/mpich/lib -lmpi
all_reduce_test compile completed
```

4. Perform all\_reduce\_test in the single-node scenario.

Go to the **hccl\_test** directory.

```
cd /usr/local/Ascend/ascend-toolkit/latest/tools/hccl_test
```

For single-node single-card, run the following command:

```
mpirun -n 1 ./bin/all_reduce_test -b 8 -e 1024M -f 2 -p 8
```

For single-node multi-card, run the following command:

```
mpirun -n 8 ./bin/all_reduce_test -b 8 -e 1024M -f 2 -p 8
```

Figure 3-17 all\_reduce\_test

```
[root@devserver-com hccl_test]# mpirun -n 8 ./bin/all_reduce_test -b 8 -e 1024M
the minbytes is 8, maxbytes is 1073741824, iters is 20, warmup_iters is 5
data_size(Bytes): | aveg_time(us): | alg_bandwidth(GB/s): | check_result:
8 | 1323.66 | 0.00001 | success
16 | 1537.41 | 0.00001 | success
32 | 1567.12 | 0.00002 | success
64 | 1530.88 | 0.00004 | success
128 | 1567.90 | 0.00008 | success
256 | 1544.79 | 0.00017 | success
512 | 1534.98 | 0.00033 | success
1024 | 1771.28 | 0.00058 | success
2048 | 1457.74 | 0.00140 | success
4096 | 1619.05 | 0.00253 | success
8192 | 1570.33 | 0.00522 | success
16384 | 1575.37 | 0.01040 | success
32768 | 1542.54 | 0.02124 | success
65536 | 1568.91 | 0.04177 | success
131072 | 1554.22 | 0.08433 | success
262144 | 1552.85 | 0.16881 | success
524288 | 1573.59 | 0.33318 | success
1048576 | 1540.16 | 0.68082 | success
2097152 | 1544.21 | 1.35807 | success
4194304 | 1555.34 | 2.69671 | success
8388608 | 1558.78 | 5.38153 | success
16777216 | 1556.50 | 10.77880 | success
33554432 | 1425.38 | 23.54074 | success
67108864 | 1349.46 | 49.72998 | success
134217728 | 2460.50 | 54.54894 | success
268435456 | 4623.78 | 58.05536 | success
536870912 | 9194.49 | 58.39050 | success
1073741824 | 18450.20 | 58.19677 | success
```

5. Test the bandwidth of multi-node RoCE NICs.
  - a. Check the Ascend RoCE IP address.

```
cat /etc/hccn.conf
```

Figure 3-18 Viewing Ascend RoCE IP address

```
[root@devserver-com hcc_test]# cat /etc/hccn.conf
address_0=29.89.132.13
netmask_0=255.255.0.0
netdetect_0=29.89.0.1
gateway_0=29.89.0.1
send_arp_status_0=1
address_1=29.89.20.64
netmask_1=255.255.0.0
netdetect_1=29.89.0.1
gateway_1=29.89.0.1
send_arp_status_1=1
address_2=29.89.155.174
netmask_2=255.255.0.0
netdetect_2=29.89.0.1
gateway_2=29.89.0.1
send_arp_status_2=1
address_3=29.89.148.38
netmask_3=255.255.0.0
netdetect_3=29.89.0.1
gateway_3=29.89.0.1
send_arp_status_3=1
address_4=29.89.134.236
netmask_4=255.255.0.0
netdetect_4=29.89.0.1
gateway_4=29.89.0.1
send_arp_status_4=1
address_5=29.89.133.119
netmask_5=255.255.0.0
netdetect_5=29.89.0.1
gateway_5=29.89.0.1
send_arp_status_5=1
address_6=29.89.51.253
netmask_6=255.255.0.0
netdetect_6=29.89.0.1
gateway_6=29.89.0.1
send_arp_status_6=1
address_7=29.89.96.167
netmask_7=255.255.0.0
netdetect_7=29.89.0.1
gateway_7=29.89.0.1
```

- b. Perform the RoCE test.

In session 1, run the `-iCard ID` command on the receive end.

```
hccn_tool -i 7 -roce_test reset
hccn_tool -i 7 -roce_test ib_send_bw -s 4096000 -n 1000 -tcp
```

In session 2, run the `-iCard ID` command on the sending end. The IP address of the receive end is at the end.

```
cd /usr/local/Ascend/ascend-toolkit/latest/tools/hccl_test
hccl_tool -i 0 -roce_test reset
hccl_tool -i 0 -roce_test ib_send_bw -s 4096000 -n 1000 address 192.168.100.18 -tcp
```

The following figure shows the RoCE test result.

**Figure 3-19** RoCE test result (receive end)

```
[root@devserver-com hccl_test]# hccl_tool -i 7 -roce_test ib_send_bw -s 4096000 -n 1000 -tcp
Dsmi get perftest status end. (status=1)
Dsmi start roce perftest end. (out=1)
Dsmi get perftest status end. (status=2)
Dsmi get perftest status end. (status=2)
Dsmi get perftest status end. (status=2)
Dsmi get perftest status end. (status=2)
Dsmi get perftest status end. (status=2)
Dsmi get perftest status end. (status=2)
Dsmi get perftest status end. (status=2)
Dsmi get perftest status end. (status=2)
Dsmi get perftest status end. (status=2)
Dsmi get perftest status end. (status=2)
Dsmi get perftest status end. (status=2)
Dsmi get perftest status end. (status=2)
Dsmi get perftest status end. (status=1)
roce_report:

* Waiting for client to connect.. *

 Send BW Test
Dual-port : OFF Device : hns_0
Number of qps : 1 Transport type : IB
Connection type : RC Using SRQ : OFF
RX depth : 512
CQ Moderation : 100
Mtu : 4096[B]
Link type : Ethernet
GID index : 3
Max inline data : 0[B]
rdma_cm QPs : OFF
Data ex. method : Ethernet

local address: LID 0000 QPN 0x000a PSN 0xf97ccb
GID: 00:00:00:00:00:00:00:00:00:00:255:255:29:89:96:167
remote address: LID 0000 QPN 0x001a PSN 0x3a835e
GID: 00:00:00:00:00:00:00:00:00:00:255:255:29:89:132:13

#bytes #iterations BW peak[MB/sec] BW average[MB/sec] MsgRate[Mpps]
4096000 1000 0.00 23395.00 0.005989

```

**Figure 3-20** RoCE test result (server)

```
[root@devserver-com hccl_test]# hccl_tool -i 0 -roce_test ib_send_bw -s 4096000 -n 1000 address 29.89.96.167 -tcp
Dsmi get perftest status end. (status=1)
Dsmi start roce perftest end. (out=1)
Dsmi get perftest status end. (status=1)
roce_report:

 Send BW Test
Dual-port : OFF Device : hns_0
Number of qps : 1 Transport type : IB
Connection type : RC Using SRQ : OFF
TX depth : 128
CQ Moderation : 100
Mtu : 4096[B]
Link type : Ethernet
GID index : 3
Max inline data : 0[B]
rdma_cm QPs : OFF
Data ex. method : Ethernet

local address: LID 0000 QPN 0x001a PSN 0x3a835e
GID: 00:00:00:00:00:00:00:00:00:00:255:255:29:89:132:13
remote address: LID 0000 QPN 0x000a PSN 0xf97ccb
GID: 00:00:00:00:00:00:00:00:00:00:255:255:29:89:96:167

#bytes #iterations BW peak[MB/sec] BW average[MB/sec] MsgRate[Mpps]
4096000 1000 23372.40 23369.61 0.005983

```



 NOTE

- If the RoCE bandwidth test has been started for a NIC, the following error message is displayed when the task is started again.

Figure 3-21 Error

```
[root@devserver-com hcc1_test]# hccn_tool -i 7 -roce_test ib_send_bw -s 4096 -n
1000 -tcp
Dsmi get perfest status end. (status=2)
Roce perfest is doing, please try later.
Cmd execute failed!
```

Run the following command to stop the **roce\_test** task and then start the task:

```
hccn_tool -i 7 -roce_test reset
```

- Run the following command to query the NIC status:  
for i in {0..7};do hccn\_tool -i \${i} -link -g;done
- Run the following command to check the IP address connectivity of the NIC on a single node:  
for i in \$(seq 0 7);do hccn\_tool -i \$i -net\_health -g;done

----End

## Creating a Containerized Custom Debugging Environment

### Step 1 Prepare a service base image.

You could start your Docker container on the physical machine (PM) for development. You can use your service images or base images provided by ModelArts, including Ascend+PyTorch and Ascend+MindSpore.

1. Choose an image based on your environment.  
# Container image matching Snt9b. The following shows an example.  
docker pull swr.<region-code>.myhuaweicloud.com/atelier/<image-name>:<image-tag>
2. Start the container image. If multiple users and containers are sharing a machine, allocate the cards beforehand. Do not use cards occupied by other containers.  
# Start the container. Specify the container name and image information.  
**ASCEND\_VISIBLE\_DEVICES** indicates the cards to be used by the container, for example, **0-1,3** indicates cards 0, 1, and 3 are used. The hyphens (-) specify the range.  
# -v /home:/home\_host indicates mounting the home directory of the host to the **home\_host** directory of the container. Use this mounting directory in the container to store code and data for persistent storage.  
docker run -itd --cap-add=SYS\_PTRACE -e ASCEND\_VISIBLE\_DEVICES=0 -v /home:/home\_host -p 51234:22 -u=0 --name *Custom container name* *SWR address of the image pulled in the preceding step* /bin/bash
3. Access the container.  
docker exec -ti *Custom container name in the last command* bash
4. Access the Conda environment.  
source /home/ma-user/.bashrc  
cd ~
5. View the information of available cards in the container.  
npu-smi info

If the following error message is displayed, the card specified by **ASCEND\_VISIBLE\_DEVICES** during container startup is occupied by another container. In this case, select another card and restart the new container.



Figure 3-22 Error

```
(PyTorch-1.11.0) [root@8e2a7f7f9f7a ma-user]# npu-smi info
DrvMngGetConsoleLogLevel failed. (g_conLogLevel=3)
dcmi model initialized failed, because the device is used. ret is -8020
(PyTorch-1.11.0) [root@8e2a7f7f9f7a ma-user]#
```

- After you run the **npu-smi info** command and the output is normal, run the following commands to test the container environment. If the output is normal, the container environment is available.

- PyTorch image test:  
python3 -c "import torch;import torch\_npu; a = torch.randn(3, 4).npu(); print(a + a);"
- MindSpore image test:  
# The run\_check program of MindSpore does not adapt to Snt9b. Configure two environment variables first.  
unset MS\_GE\_TRAIN  
unset MS\_ENABLE\_GE  
python -c "import mindspore;mindspore.set\_context(device\_target='Ascend');mindspore.run\_check()"  
# Restore the environment variables after the test for actual training.  
export MS\_GE\_TRAIN=1  
export MS\_ENABLE\_GE=1

Figure 3-23 Accessing the Conda environment and performing a test

```
[root@devserver-modelarts-demanager-0eaabe8f ~]# docker run -itd --cap-add=SYS_PTRACE -e ASCEND_VISIBLE_DEVICES=3 -v /home:/host_home -u=0 --name pytorch_test swr.cn-southwest-2.myhuaweicloud.com/atelier/pytorch_1_11_ascend:pytorch_1.11.0-cann_6.3.2-py_3.7-euler_2.10.7-aarch64-d910b-20230815141604-3685231 /bin/bash
0292be41aclef03a37b7c78adcff4fc999a967e1163e5f6e565edbe6a638c69b
[root@devserver-modelarts-demanager-0eaabe8f ~]# docker exec -ti 0292be41a bash
The environment has been set
[root@0292be41acle ma-user]# source .bashrc
The environment has been set
The environment has been set
(PyTorch-1.11.0) [root@0292be41acle ma-user]# python3 -c "import torch;import torch_npu; a = torch.randn(3, 4).npu(); print(a + a);"
tensor([[-1.0911, -0.4146, 1.6027, 1.8585],
 [3.2549, 0.7026, 2.9356, 0.9544],
 [5.1409, -0.8820, -0.3400, 0.0257]]) device='npu:0')
(PyTorch-1.11.0) [root@0292be41acle ma-user]#
```

## Step 2 (Optional) Configure SSH access for the container.

If you need to use the VS Code or SSH tool to directly connect to the container for development, perform the following operations:

- After accessing the container, run the SSH startup command to start the SSH service.  
ssh-keygen -A  
/usr/sbin/sshd  
# Check whether SSH is started.  
ps -ef |grep ssh
- Set the passwd for user **root** and enter the password as prompted.  
passwd

Figure 3-24 Setting a password for user root

```
[root@9f4f3b6794f7 ~]$ passwd
Changing password for user root.
New password:
Retype new password:
passwd: all authentication tokens updated successfully.
[root@9f4f3b6794f7 ~]#
```

- Run the **exit** command to exit the container and perform the SSH test on the host.  
ssh root@Host IP address -p 51234 (Mapped port number)

**Figure 3-25** Perform the SSH test.

```
[root@localhost ~]# ssh root@90.90.3.71 -p 51234
root@90.90.3.71's password:
Authorized users only. All activities may be monitored and reported.
[root@9f4f3b6794f7 ~]#
```

If the error message "Host key verification failed" is displayed when you perform the SSH container test on the host machine, delete the `~/.ssh/known_host` file from the host machine and try again.

4. Use VS Code SSH to connect to the container environment.

If you have not used VS Code SSH, install the VS Code environment and Remote-SSH plug-in by referring to [Step1 Manually Connecting to a Notebook Instance Through VS Code](#).

Open VSCode Terminal and run the following command to generate a key pair on the local computer. If you already have a key pair, skip this step.

```
ssh-keygen -t rsa
```

Add the public key to the authorization file of the remote server. Replace the server IP address and container port number.

```
cat ~/.ssh/id_rsa.pub | ssh root@Server IP address -p Container port number "mkdir -p ~/.ssh && cat >> ~/.ssh/authorized_keys"
```

Open the Remote-SSH configuration file of VSCode and add SSH configuration items. Replace the server IP address and container port number.

```
Host Snt9b-dev
 HostName Server IP address
 User root
 port SSH port number of the container
 identityFile ~/.ssh/id_rsa
 StrictHostKeyChecking no
 UserKnownHostsFile /dev/null
 ForwardAgent yes
```

**Note: Use the key to log in. If you want to use the password, delete the identityFile configuration and enter the password as prompted during the connection.**

After the connection, install the Python plug-in. For details, see [Install the Python Plug-in in the Cloud Development Environment](#).

### Step 3 (Optional) Install CANN Toolkit.

CANN Toolkit has been installed in the preset images provided by ModelArts. If you need to use another version or use your own image that is not preset with CANN Toolkit, see the following operations.

1. Check whether CANN Toolkit has been installed in the container. If the version number is displayed, it has been installed.

```
cat /usr/local/Ascend/ascend-toolkit/latest/aarch64-linux/ascend_toolkit_install.info
```

2. If it is not installed or needs to be upgraded, obtain the software package from the official website. For common users, [download the community edition](#). For Huawei engineers and channel users, [download the commercial edition](#).

Install CANN Toolkit. Replace the package name.

```
chmod 700 *.run
./Ascend-cann-toolkit_6.3.RC2_linux-aarch64.run --full --install-for-all
```

3. If it has been installed but needs to be upgraded, run the following command. Replace the package name.

```
chmod 700 *.run
./Ascend-cann-toolkit_6.3.RC2_linux-aarch64.run --upgrade --install-for-all
```

#### Step 4 (Optional) Install MindSpore Lite.

MindSpore Lite has been installed in the preset image. If you need to use another version or use your own image that is not preset with MindSpore Lite, see the following operations.

1. Check whether MindSpore Lite has been installed in the container. If the software information and version are displayed, it has been installed.

```
pip show mindspore-lite
```

2. If it is not installed, download the .whl and .tar.gz packages from the [official website](#) and download them. Replace the package names.

```
pip install mindspore_lite-2.1.0-cp37-cp37m-linux_aarch64.whl
mkdir -p /usr/local/mindspore-lite
tar -zxvf mindspore-lite-2.1.0-linux-aarch64.tar.gz -C /usr/local/mindspore-lite --strip-components 1
```

#### Step 5 Configure the pip source and Yum repository.

- Configure the pip source.

The pip source has been configured in the preset image provided by ModelArts. If you need to use your own service images, configure it by referring to [Step 5](#).

- Configure the Yum repository.

Run the following commands to configure the Yum repository:

```
Automatically configure the Yum repository.
wget http://mirrors.myhuaweicloud.com/repo/mirrors_source.sh && bash mirrors_source.sh

Test
yum update --allowerase --skip-broken --nobest
```

#### Step 6 Install **git-lfs** and run the **git clone** command to download the code.

To use **git clone** and **git lfs** commands to download large models, see the following operations:

1. The Euler source does not have the **git-lfs** package. Therefore, you need to decompress the package. To do so, enter the following address in the address box of the browser, download the **git-lfs** package, and upload it to the **/home** directory on the server. This directory is mounted to the **/home\_host** directory of the container when the container is started. In this way, the **git-lfs** package can be directly used in the container.

```
https://github.com/git-lfs/git-lfs/releases/download/v3.2.0/git-lfs-linux-arm64-v3.2.0.tar.gz
```

2. Go to the container and run the **git-lfs** installation commands.

```
cd /home_host
tar -zxvf git-lfs-linux-arm64-v3.2.0.tar.gz
cd git-lfs-3.2.0
sh install.sh
```

3. Disable SSL verification for Git configuration.

```
git config --global http.sslVerify false
```

4. The following commands use code in diffusers as an example. Replace the development directory.

```
git clone diffusers source code, -b You can specify a branch for this parameter. Replace the
development directory.
cd /home_host/User directory
mkdir sd
cd sd
git clone https://github.com/huggingface/diffusers.git -b v0.11.1-patch
```

Run the **git clone** command to download the model on Hugging Face. The following uses a stable-diffusion (SD) model as an example. If the **SSL\_ERROR\_SYSCALL** error is reported during the download, try again. The download may take several hours due to network restrictions and large file size. If the download still fails after multiple retries, download the large file from the website and upload it to the personal development directory in / **home** on the server. To skip the large files during download, set **GIT\_LFS\_SKIP\_SMUDGE** to 1.

```
git lfs install
git clone https://huggingface.co/runwayml/stable-diffusion-v1-5 -b onnx
```

Figure 3-26 Downloaded code

```
[root@38a757e4636a sd]# git clone https://github.com/huggingface/diffusers.git -b v0.11.1-patch
Cloning into 'diffusers'...
remote: Enumerating objects: 34118, done.
remote: Counting objects: 100% (10965/10965), done.
remote: Compressing objects: 100% (765/765), done.
remote: Total 34118 (delta 10639), reused 10273 (delta 10190), pack-reused 23153
Receiving objects: 100% (34118/34118), 21.44 MiB | 9.58 MiB/s, done.
Resolving deltas: 100% (25313/25313), done.
[root@38a757e4636a sd]# cd diffusers/
[root@38a757e4636a diffusers]# git branch
* v0.11.1-patch
[root@38a757e4636a diffusers]#
```

**Step 7** Save the image in the container environment.

After the environment is configured, you can develop and debug the service code. To prevent the environment from being lost after the host is restarted, run the following commands to save the configured environment as a new image:

```
Check the ID of the container to be saved as an image.
docker ps
Save the image.
docker commit Container ID Custom image name:Custom image tag
View the saved image.
docker images
If you need to share the image with others in other environments, save the image as a local file. This
command takes a long time. You can view the file after it is saved.
docker save -o Custom name.tar Image name:Image tag
Load the file on other hosts. After the file is loaded, you can view the image.
docker load --input Custom name.tar
```

For details about how to migrate services to Ascend for development and debugging, see the related documents.

----End

### 3.4.2 Configuring the Software Environment on the GPU Server

#### Scenario

This section describes how to configure the environment on a GPU BMS, including installing NVIDIA and CUDA drivers. For different GPU preset images, the pre-installed software varies. You can view the installed software by referring to [Mapping Between Compute Resources and Image Versions](#). The following describes the common software installation procedure. You can view the content based on the software to be installed.

- [Installing a NVIDIA Driver](#)

- [Installing a CUDA Toolkit](#)
- [Installing Docker](#)
- [Installing nvidia-fabricmanager](#)

The following are the typical configuration scenarios. View the related documents for quick configuration.

- [Installing NVIDIA 515 and CUDA 11.7 on a GP Vnt1 BMS in EulerOS 2.9](#)
- [Installing NVIDIA 470 and CUDA 11.4 on a GP Vnt1 BMS in Ubuntu 18.04](#)
- [Installing NVIDIA 515 and CUDA 11.7 on a GP Vnt1 BMS in Ubuntu18.04](#)
- [Installing NVIDIA 515 and CUDA 11.7 on a GP Ant8 BMS in Ubuntu 20.04](#)

## Installing a NVIDIA Driver

**Step 1** Visit the [NVIDIA official website](#).

**Step 2** The Ant8 specifications are used as an example. Select a driver based on the Ant8 details and the required CUDA version.

**Figure 3-27** Selecting a driver

### Manual Driver Search

Search by product, product type or series

Data Center / Tesla  ⓘ

A-Series

NVIDIA A800

Linux 64-bit

12.0

English (US)

The driver version is automatically displayed and downloaded.

```
wget https://cn.download.nvidia.com/tesla/470.182.03/NVIDIA-Linux-x86_64-470.182.03.run
```

**Step 3** Assign permissions.

```
chmod +x NVIDIA-Linux-x86_64-470.182.03.run
```

**Step 4** Run the installation file.

```
./NVIDIA-Linux-x86_64-470.182.03.run
```

The NVIDIA-DRIVER driver is installed.

----End

## Installing a CUDA Toolkit

The NVIDIA driver is installed based on CUDA 12.0. In this case, CUDA 12.0 is installed by default.

**Step 1** Visit [CUDA Toolkit](#).

**Step 2** After you set the OS, architecture, distribution, version, and installation type, an installation command is generated. Copy the command and run it.

**Figure 3-28** Settings

|                  |             |               |                 |                |          |      |       |      |
|------------------|-------------|---------------|-----------------|----------------|----------|------|-------|------|
| Operating System | Linux       | Windows       |                 |                |          |      |       |      |
| Architecture     | x86_64      | ppc64le       | arm64-sbsa      | aarch64-jetson |          |      |       |      |
| Distribution     | CentOS      | Debian        | Fedora          | KylinOS        | OpenSUSE | RHEL | Rocky | SLES |
|                  | Ubuntu      | WSL-Ubuntu    |                 |                |          |      |       |      |
| Version          | 18.04       | 20.04         | 22.04           |                |          |      |       |      |
| Installer Type   | deb (local) | deb (network) | runfile (local) |                |          |      |       |      |

The generated installation commands are as follows:

```
wget https://developer.download.nvidia.com/compute/cuda/repos/ubuntu2004/x86_64/cuda-ubuntu2004.pin
sudo mv cuda-ubuntu2004.pin /etc/apt/preferences.d/cuda-repository-pin-600
wget https://developer.download.nvidia.com/compute/cuda/12.1.1/local_installers/cuda-repo-ubuntu2004-12-1-local_12.1.1-530.30.02-1_amd64.deb
sudo dpkg -i cuda-repo-ubuntu2004-12-1-local_12.1.1-530.30.02-1_amd64.deb
sudo cp /var/cuda-repo-ubuntu2004-12-1-local/cuda-*-keyring.gpg /usr/share/keyrings/
sudo apt-get update
sudo apt-get -y install cuda
```

### NOTE

To obtain CUDA of earlier versions, see [CUDA Toolkit Archive](#).

----End

## Installing Docker

Docker is not installed in some preset images of Vnt1 BMS. To install Docker, see the following operations:

**Step 1** Install Docker.

```
curl https://get.docker.com | sh && sudo systemctl --now enable docker
```

**Step 2** Install the NVIDIA container plug-in.

```
distribution=$(. /etc/os-release;echo ${ID}${VERSION_ID})
&& curl -fsSL https://nvidia.github.io/libnvidia-container/gpgkey | sudo gpg --dearmor -o /usr/share/
```

```
keyrings/nvidia-container-toolkit-keyring.gpg
&& curl -s -L https://nvidia.github.io/libnvidia-container/$distribution/libnvidia-container.list |
sed 's#deb https://#deb [signed-by=/usr/share/keyrings/nvidia-container-toolkit-keyring.gpg] https://#g' |
sudo tee /etc/apt/sources.list.d/nvidia-container-toolkit.list
apt-get update
apt-get install -y nvidia-container-toolkit
nvidia-ctl runtime configure --runtime=docker
systemctl restart docker
```

**Step 3** Check whether the Docker environment has been installed.

The following uses PyTorch 2.0 as an example. The image used in this case is large and it may take a while to pull the image.

```
docker run -ti --runtime=nvidia --gpus all pytorch/pytorch:2.0.0-cuda11.7-cudnn8-devel bash
```

**Figure 3-29** Image pulled

```
root@bms-8e98:~/miniconda3/bin# docker run -ti --runtime=nvidia --gpus all pytorch/pytorch:2.0.0-cuda11.7-cudnn8-devel bash
=====
== CUDA ==
=====
CUDA Version 11.7.0

Container image Copyright (c) 2016-2022, NVIDIA CORPORATION & AFFILIATES. All rights reserved.

This container image and its contents are governed by the NVIDIA Deep Learning Container License.
By pulling and using the container, you accept the terms and conditions of this license:
https://developer.nvidia.com/ngc/nvidia-deep-learning-container-license

A copy of this license is made available in this container at /NGC-DL-CONTAINER-LICENSE for your convenience.

** DEPRECATION NOTICE! **

THIS IMAGE IS DEPRECATED and is scheduled for DELETION.
https://gitlab.com/nvidia/container-images/cuda/blob/master/doc/support-policy.md

root@70e4729175c:/workspace# python
Python 3.10.9 (main, Mar 8 2023, 10:47:38) [GCC 11.2.0] on linux
Type "help", "copyright", "credits" or "license()" for more information.
>>> import torch
>>> torch.__version__
'2.0.0'
>>> torch.cuda.device_count()
8
>>> torch.cuda.is_available()
True
>>>
```

----End

## Installing nvidia-fabricmanager

NVLink and NVSwitch are supported for Ant GPUs. If you use a node with multiple GPUs, install nvidia-fabricmanager matching your driver version to enable interconnection between GPUs. Otherwise, GPU pods may fail to be used.

 **NOTE**

The nvidia-fabricmanager version must be the same as the nvidia driver version.

The following uses version 515.105.01 as an example.

```
version=515.105.01
main_version=$(echo $version | awk -F '.' '{print $1}')
apt-get update
apt-get -y install nvidia-fabricmanager-${main_version}-${version}-*
```

Verify the driver installation result. Start the fabricmanager service and check whether the status is **RUNNING**.

```
nvidia-smi -pm 1
nvidia-smi
systemctl enable nvidia-fabricmanager
systemctl start nvidia-fabricmanager
systemctl status nvidia-fabricmanager
```

## Installing NVIDIA 515 and CUDA 11.7 on a GP Vnt1 BMS in EulerOS 2.9

This section describes how to install NVIDIA 515.105.01 and CUDA 11.7.1 on a GP Vnt1 BMS in EulerOS 2.9.

### Step 1 Install the NVIDIA driver.

```
wget https://us.download.nvidia.com/tesla/515.105.01/NVIDIA-Linux-x86_64-515.105.01.run
chmod 700 NVIDIA-Linux-x86_64-515.105.01.run
```

```
yum install -y elfutils-libelf-devel
./NVIDIA-Linux-x86_64-515.105.01.run --kernel-source-path=/usr/src/kernels/
4.18.0-147.5.1.6.h998.eulerosv2r9.x86_64
```

#### NOTE

By default, the Yum repository used by the Vnt1 BMS is <http://repo.huaweicloud.com>, which is available. If an error message is displayed when you run the **yum update** command, indicating that a software package conflict occurs, run the **yum remove xxx software package** command.

The NVIDIA driver is a binary file and requires the **libelf** library in the **elfutils-libelf-devel** development package in the system. It provides a set of C functions for reading, modifying, and creating ELF files. NVIDIA drivers need these functions to parse the currently running kernel and other related information.

During the installation, select **OK** or **YES** as prompted. After the installation, run the **reboot** command to restart the server. Log in to the server again and run the following command to view the GPU information:

```
nvidia-smi -pm 1 #This command will be executed for a while. Wait patiently. The persistent mode is
enabled to optimize the GPU performance on the Linux instance.
nvidia-smi
```

### Step 2 Install CUDA.

```
wget https://developer.download.nvidia.com/compute/cuda/11.7.1/local_installers/
cuda_11.7.1_515.65.01_linux.run
chmod 700 cuda_11.7.1_515.65.01_linux.run
./cuda_11.7.1_515.65.01_linux.run --toolkit --samples --silent
```

Check the installation result.

```
/usr/local/cuda/bin/nvcc -V
```

### Step 3 Install PyTorch 2.0 and verify CUDA.

To install PyTorch 2.0, Python 3.10 is required, and the miniconda environment needs to be installed and configured.

#### 1. Install miniconda and create the alpha environment.

```
wget https://repo.anaconda.com/miniconda/Miniconda3-py310_23.1.0-1-Linux-x86_64.sh
chmod 750 Miniconda3-py310_23.1.0-1-Linux-x86_64.sh
bash Miniconda3-py310_23.1.0-1-Linux-x86_64.sh -b -p /home/miniconda
export PATH=/home/miniconda/bin:$PATH
conda create --quiet --yes -n alpha python=3.10
```

#### 2. Install PyTorch 2.0 and verify the CUDA status.

Install PyTorch 2.0 in the alpha environment and use the Tsinghua PIP source.

```
source activate alpha
pip install torch==2.0 -i https://pypi.tuna.tsinghua.edu.cn/simple
python
```

Verify the installation status of PyTorch and CUDA. If the output is **True**, the installation is successful.



```
import torch
print(torch.cuda.is_available())
```

----End

## Installing NVIDIA 470 and CUDA 11.4 on a GP Vnt1 BMS in Ubuntu 18.04

This section describes how to install NVIDIA 470 and CUDA 11.4 on a GP Vnt1 BMS in Ubuntu 18.04.

### Step 1 Install the NVIDIA driver.

```
apt-get update
sudo apt-get install nvidia-driver-470
```

### Step 2 Install CUDA.

```
wget https://developer.download.nvidia.com/compute/cuda/11.4.4/local_installers/
cuda_11.4.4_470.82.01_linux.run
chmod +x cuda_11.4.4_470.82.01_linux.run
./cuda_11.4.4_470.82.01_linux.run --toolkit --samples --silent
```

### Step 3 Verify the NVIDIA installation result.

```
nvidia-smi -pm 1
nvidia-smi
/usr/local/cuda/bin/nvcc -V
```

### Step 4 Install PyTorch 2.0 and verify CUDA.

To install PyTorch 2.0, Python 3.10 is required, and the miniconda environment needs to be installed and configured.

#### 1. Install miniconda and create the alpha environment.

```
wget https://repo.anaconda.com/miniconda/Miniconda3-py310_23.1.0-1-Linux-x86_64.sh
chmod 750 Miniconda3-py310_23.1.0-1-Linux-x86_64.sh
bash Miniconda3-py310_23.1.0-1-Linux-x86_64.sh -b -p /home/miniconda
export PATH=/home/miniconda/bin:$PATH
conda create --quiet --yes -n alpha python=3.10
```

#### 2. Install PyTorch 2.0 and verify the CUDA status.

Install PyTorch 2.0 in the alpha environment and use the Tsinghua PIP source.  
source activate alpha  
conda install pytorch torchvision torchaudio pytorch-cuda=11.7 -c pytorch -c nvidia  
python

Verify the installation status of PyTorch and CUDA. If the output is **True**, the installation is successful.

```
import torch
print(torch.cuda.is_available())
```

----End

## Installing NVIDIA 515 and CUDA 11.7 on a GP Vnt1 BMS in Ubuntu18.04

This section describes how to install NVIDIA 515 and CUDA 11.7 on a GP Vnt1 BMS in Ubuntu18.04.

### Step 1 Install the NVIDIA driver.

```
wget https://us.download.nvidia.com/tesla/515.105.01/NVIDIA-Linux-x86_64-515.105.01.run
chmod +x NVIDIA-Linux-x86_64-515.105.01.run
./NVIDIA-Linux-x86_64-515.105.01.run
```

### Step 2 Install CUDA.

```
wget https://developer.download.nvidia.com/compute/cuda/11.7.1/local_installers/
cuda_11.7.1_515.65.01_linux.run
```

```
chmod +x cuda_11.7.1_515.65.01_linux.run
./cuda_11.7.1_515.65.01_linux.run --toolkit --samples --silent
```

**Step 3** Install Docker.

```
curl https://get.docker.com | sh && sudo systemctl --now enable docker
```

**Step 4** Install the NVIDIA container plug-in.

```
distribution=$(. /etc/os-release;echo IDVERSION_ID)
&& curl -fsSL https://nvidia.github.io/libnvidia-container/gpgkey | sudo gpg --dearmor -o /usr/share/
keyrings/nvidia-container-toolkit-keyring.gpg
&& curl -s -L https://nvidia.github.io/libnvidia-container/$distribution/libnvidia-container.list |
sed 's#deb https://#deb [signed-by=/usr/share/keyrings/nvidia-container-toolkit-keyring.gpg] https://#g' |
sudo tee /etc/apt/sources.list.d/nvidia-container-toolkit.list
apt-get update
apt-get install -y nvidia-container-toolkit
nvidia-ctl runtime configure --runtime=docker
systemctl restart docker
```

**Step 5** Check whether the Docker environment has been installed.

The following uses PyTorch 2.0 as an example. The image used in this case is large and it may take a while to pull the image.

```
docker run -ti --runtime=nvidia --gpus all pytorch/pytorch:2.0.0-cuda11.7-cudnn8-devel bash
```

**Figure 3-30** Image pulled

```
root@bms-8e98:~/miniconda3/bin# docker run -ti --runtime=nvidia --gpus all pytorch/pytorch:2.0.0-cuda11.7-cudnn8-devel bash
=====
== CUDA ==
=====
CUDA Version 11.7.0
Container image Copyright (c) 2016-2022, NVIDIA CORPORATION & AFFILIATES. All rights reserved.
This container image and its contents are governed by the NVIDIA Deep Learning Container License.
By pulling and using the container, you accept the terms and conditions of this license:
https://developer.nvidia.com/ngc/nvidia-deep-learning-container-license
A copy of this license is made available in this container at /NGC-DL-CONTAINER-LICENSE for your convenience.

** DEPRECATION NOTICE! **

THIS IMAGE IS DEPRECATED and is scheduled for DELETION.
https://gitlab.com/nvidia/container-images/cuda/blob/master/doc/support-policy.md
root@e78e4729175c:/workspace# python
Python 3.10.9 (main, Mar 8 2023, 10:47:38) [GCC 11.2.0] on linux
Type "help", "copyright", "credits" or "license()" for more information.
>>> import torch
>>> torch.__version__
'2.0.0'
>>> torch.cuda.device_count()
8
>>> torch.cuda.is_available()
True
>>>
```

----End

## Installing NVIDIA 515 and CUDA 11.7 on a GP Ant8 BMS in Ubuntu 20.04

This section describes how to install NVIDIA driver 515, CUDA 11.7, and nvidia-fabricmanager 515 on GP Ant8 BMS (Ubuntu 20.04) and perform the nccl-test test.

**Step 1** Replace the APT source.

```
sudo sed -i "s@http://.*archive.ubuntu.com@http://repo.huaweicloud.com@g" /etc/apt/sources.list
sudo sed -i "s@http://.*security.ubuntu.com@http://repo.huaweicloud.com@g" /etc/apt/sources.list
sudo apt update
```

**Step 2** Install the NVIDIA driver.

```
wget https://us.download.nvidia.com/tesla/515.105.01/NVIDIA-Linux-x86_64-515.105.01.run
chmod +x NVIDIA-Linux-x86_64-515.105.01.run
./NVIDIA-Linux-x86_64-515.105.01.run
```

### Step 3 Install CUDA.

```
Install the .run package.
wget https://developer.download.nvidia.com/compute/cuda/11.7.0/local_installers/
cuda_11.7.0_515.43.04_linux.run
chmod +x cuda_11.7.0_515.43.04_linux.run
./cuda_11.7.0_515.43.04_linux.run --toolkit --samples --silent
```

### Step 4 Install NCCL.

#### NOTE

- For details about how to install NCCL, see [NVIDIA Deep Learning NCCL Documentation](#).
- For details about the mapping between NCCL and CUDA versions and the installation method, see [NCL Developer](#).

The following uses CUDA 11.7 as an example. Install NCCL.

```
wget https://developer.download.nvidia.com/compute/cuda/repos/ubuntu2004/x86_64/cuda-
keyring_1.0-1_all.deb
sudo dpkg -i cuda-keyring_1.0-1_all.deb
sudo apt update
sudo apt install libnccl2=2.14.3-1+cuda11.7 libnccl-dev=2.14.3-1+cuda11.7
```

The following is displayed after NCCL is installed.

#### Figure 3-31 Viewing NCCL

```
root@wangjianfeng:~#
root@wangjianfeng:~#
root@wangjianfeng:~# dpkg -l | grep nccl
ii libnccl-dev 2.14.3-1+cuda11.7 amd64 NVIDIA Collective Communication Library (NCCL) Development Files
ii libnccl2 2.14.3-1+cuda11.7 amd64 NVIDIA Collective Communication Library (NCCL) Runtime
root@wangjianfeng:~#
root@wangjianfeng:~#
```

### Step 5 Install nvidia-fabricmanager.

#### NOTE

The nvidia-fabricmanager version must be the same as the nvidia driver version.

```
version=515.105.01
main_version=$(echo $version | awk -F '.' '{print $1}')
apt-get update
apt-get -y install nvidia-fabricmanager-${main_version}=${version}-*
```

Verify the driver installation result. Start the fabricmanager service and check whether the status is **RUNNING**.

```
nvidia-smi -pm 1
nvidia-smi
systemctl enable nvidia-fabricmanager
systemctl start nvidia-fabricmanager
systemctl status nvidia-fabricmanager
```

### Step 6 Install nv-peer-memory.

```
git clone https://github.com/Mellanox/nv_peer_memory.git
cd ./nv_peer_memory
./build_module.sh
cd /tmp
tar xzf /tmp/nvidia-peer-memory_1.3.orig.tar.gz
cd nvidia-peer-memory-1.3
dpkg-buildpackage -us -uc
dpkg -i ../nvidia-peer-memory-dkms_1.2-0_all.deb
```

nv\_peer\_mem works in Linux kernel mode. Run the **lsmod | grep peer** command to check whether nv\_peer\_mem is loaded to the kernel.

 NOTE

- If the code cannot be pulled by running the **git clone** command, configure git.

```
git config --global core.compression -1
export GIT_SSL_NO_VERIFY=1
git config --global http.sslVerify false
git config --global http.postBuffer 10524288000
git config --global http.lowSpeedLimit 1000
git config --global http.lowSpeedTime 1800
```
- If `nv-peer-memory` is not displayed after the installation, the InfiniBand driver version may be too early. In this case, upgrade InfiniBand.

```
wget https://content.mellanox.com/ofed/MLNX_OFED-5.4-3.6.8.1/MLNX_OFED_LINUX-5.4-3.6.8.1-ubuntu20.04-x86_64.tgz
tar -zxvf MLNX_OFED_LINUX-5.4-3.6.8.1-ubuntu20.04-x86_64.tgz
cd MLNX_OFED_LINUX-5.4-3.6.8.1-ubuntu20.04-x86_64
apt-get install -y python3 gcc quilt build-essential bzip2 dh-python pkg-config dh-autoreconf python3-distutils debhelper make
./mlnxofedinstall --add-kernel-support
```
- For details about how to install a later version, see [Linux InfiniBand Drivers](#). For example, install the latest version, which is `MLNX_OFED-5.8-2.0.3.0`.

```
wget https://content.mellanox.com/ofed/MLNX_OFED-5.8-2.0.3.0/MLNX_OFED_LINUX-5.8-2.0.3.0-ubuntu20.04-x86_64.tgz
tar -zxvf MLNX_OFED_LINUX-5.8-2.0.3.0-ubuntu20.04-x86_64.tgz
cd MLNX_OFED_LINUX-5.8-2.0.3.0-ubuntu20.04-x86_64
apt-get install -y python3 gcc quilt build-essential bzip2 dh-python pkg-config dh-autoreconf python3-distutils debhelper make
./mlnxofedinstall --add-kernel-support
```
- After `nv_peer_mem` is installed, view its status.

```
/etc/init.d/nv_peer_mem/ status
```

If the file does not exist, the file may not be copied by default during the installation. In this case, you need to copy the file.

```
cp /tmp/nvidia-peer-memory-1.3/nv_peer_mem.conf /etc/infiniband/
cp /tmp/nvidia-peer-memory-1.3/debian/tmp/etc/init.d/nv_peer_mem /etc/init.d/
```

**Step 7** Configure environment variables. NOTE

The MPI path version must match. You can run the **ls /usr/mpi/gcc/** command to view the Open MPI version.

```
Add to ~/.bashrc.
export LD_LIBRARY_PATH=/usr/local/cuda/lib:/usr/local/cuda/lib64:/usr/include/nccl.h:/usr/mpi/gcc/
openmpi-4.1.2a1/lib:$LD_LIBRARY_PATH
export PATH=$PATH:/usr/local/cuda/bin:/usr/mpi/gcc/openmpi-4.1.2a1/bin
```

**Step 8** Install and compile `nccl-test`.

```
cd /root
git clone https://github.com/NVIDIA/nccl-tests.git
cd ./nccl-tests
make MPI=1 MPI_HOME=/usr/mpi/gcc/openmpi-4.1.2a1 -j 8
```

 NOTE

The parameter **MPI=1** must be added during compilation. Otherwise, the test between multiple devices cannot be performed.

The MPI path version must match. You can run the **ls /usr/mpi/gcc/** command to view the Open MPI version.

**Step 9** Perform the `nccl-test` test.

- Single-server test:

```
/root/nccl-tests/build/all_reduce_perf -b 8 -e 1024M -f 2 -g 8
```
- Multi-server test (replace the content comes after **btl\_tcp\_if\_include** with the active NIC name):

```
mpirun --allow-run-as-root --hostfile hostfile -mca btl_tcp_if_include eth0 -mca btl_openib_allow_ib true -x NCCL_DEBUG=INFO -x NCCL_IB_GID_INDEX=3 -x NCCL_IB_TC=128 -x NCCL_ALGO=RING -x NCCL_IB_HCA=^mlx5_bond_0 -x LD_LIBRARY_PATH /root/nccl-tests/build/all_reduce_perf -b 8 -e 11g -f 2 -g 8
```

hostfile format:

```
#Private IP address of the host Number of processes on a single node
192.168.20.1 slots=1
192.168.20.2 slots=1
```

NCCL environment variables:

- **NCCL\_IB\_GID\_INDEX=3**: enables data packets to be transmitted through the queue 4 of switches, which is RoCE-compliant.
- **NCCL\_IB\_TC=128**: enables RoCEv2. RoCEv1 is enabled by default. However, RoCEv1 does not support congestion control on switches, which may lead to packet loss. In addition, later-version switches do not support RoCEv1, leading to a RoCEv1 failure.
- **NCCL\_ALGO=RING**: The bus bandwidth of `nccl_test` is calculated based on the ring algorithm.

The calculation formulas are as follows: Bus bandwidth = Algorithm bandwidth x 2(N-1)/N, Algorithm bandwidth = Data volume/Time

The ring algorithm must be used. The formulas are different for the tree algorithm.

The bus bandwidth calculated by the tree algorithm is equivalent to the performance acceleration compared with the ring algorithm. The total time required for algorithm calculation is reduced. Therefore, the bus bandwidth calculated using the formula is also increased. Theoretically, the tree algorithm is better than the ring algorithm. However, the tree algorithm has higher requirements on the network than the ring algorithm, and the calculation may be unstable. The tree algorithm can complete the all reduce calculation with less data traffic, but it is not suitable for testing performance. Therefore, the actual bandwidth of two nodes is 100 GB/s, but the tested speed is 110 GB/s or even 130 GB/s. After this parameter is added, the speed is stable in the case of two or more nodes.

 NOTE

During the test, password-free login is required between the node where the **mpirun** command is executed and the node in the hostfile. To set SSH password-free login, perform the following steps:

1. Generate a pair of public and private keys on the local client.

```
ssh-keygen
```

After the preceding command is executed, **id\_rsa.pub** (public key) and **id\_rsa** (private key) are created in the **.ssh** folder in the user directory. View the public key and private key.

```
cd ~/.ssh
```

2. Upload the public key to the server.

For example, if the username is **root** and the server address is 192.168.222.213.

```
ssh-copy-id -i ~/.ssh/id_rsa.pub root@192.168.222.213
```

View the **id\_rsa.pub** (public key) content.

```
cd ~/.ssh
```

```
vim authorized_keys
```

3. Test password-free login.

The client connects to the remote server through SSH. You can log in to the server without entering a password.

```
ssh root@192.168.222.213
```

----End

# 4 Using Lite Server Resources

---

## 4.1 PyTorch GPU Training and Inference Guide for GPT-2

### Scenario

This section describes how to use the DeepSpeed framework to train GPT-2 in a GP Ant8 BMS (single-node single-card and single-node multi-card). After the training, the automatically generated content and interactive dialog box mode are provided.

### Background

- Megatron-DeepSpeed

Deepspeed is a PyTorch-based deep learning model training framework. It combines Megatron-LM and DeepSpeed to train on systems with distributed computing, and takes full advantage of the parallel processing capabilities of multiple GPUs and deep learning accelerators. Large-scale language models can be efficiently trained.

Megatron-LM is a model for large-scale language modeling. Based on the Generative Pre-trained Transformer (GPT) architecture, it is a neural network model based on the self-attention mechanism and is widely used in natural language processing tasks, such as text generation, machine translation, and dialog systems.

DeepSpeed is an open-source library for accelerating deep learning training. It is optimized for large-scale model and distributed training, which can significantly improve the training speed and efficiency. DeepSpeed provides various technologies and optimization policies, including distributed gradient descent, model parallelization, gradient accumulation, and dynamic precision scaling. It also supports optimizing the memory usage and compute resource allocation of foundation models.

- GPT2

GPT2 is a new pre-trained model released by OpenAI based on the GPT model in 2018. It is a large language model based on Transformer. It is trained on a

large number of datasets and directly runs a pre-trained GPT-2 model. Given a preset start word or sentence, it can randomly generate subsequent text.

## Preparing the Environment

Purchase a GPU-powered BMS on ModelArts and select a general-purpose image for AIGC to train a GPT2 model. The following images and specifications are used as example:

- Image: Ubuntu 20.04 x86 64bit SDI3 for Ant8 BareMetal with RoCE and NVIDIA-525 CUDA-12.0
- BMS specifications: GP Ant8, including eight GPU cards and eight RoCE NICs

To purchase an Ant8 BMS, submit a service ticket on the Huawei Cloud official website.

## Step 1 Installing a Model

**Step 1** Install the Megatron-DeepSpeed framework.

1. Log in to the GPU BMS using SSH as user **root**.
2. Pull the PyTorch image. You can download a common image source.  

```
docker pull nvcr.io/nvidia/pytorch:21.10-py3
```
3. Start the container.  

```
docker run -d -t --network=host --gpus all --privileged --ipc=host --ulimit memlock=-1 --ulimit stack=67108864 --name megatron-deepspeed -v /etc/localtime:/etc/localtime -v /root/.ssh:/root/.ssh nvcr.io/nvidia/pytorch:21.10-py3
```
4. Access the container terminal.  

```
docker exec -it megatron-deepspeed bash
```
5. Download the Megatron-DeepSpeed framework.  

```
git clone https://github.com/bigscience-workshop/Megatron-DeepSpeed
```

### NOTE

If the **git clone** fails to be executed, download the model to the local host, copy it to the server, and run the **docker cp** command to copy it to the container.

6. Install the Megatron-DeepSpeed framework.  

```
cd Megatron-DeepSpeed
pip install -r requirements.txt -i http://mirrors.myhuaweicloud.com/pypi/web/simple --trusted-host mirrors.myhuaweicloud.com
pip install mpi4py -i http://mirrors.myhuaweicloud.com/pypi/web/simple --trusted-host mirrors.myhuaweicloud.com
```
7. Modify the test code and comment out the line where the assertion is located.  

```
vim /workspace/Megatron-DeepSpeed/megatron/model/fused_softmax.py +191
```

Add **#** before **assert mask is None, "Mask is silently ignored due to the use of a custom kernel"**.

```
assert mask is None, "Mask is silently ignored due to the use of a custom kernel"
```

**Step 2** Download and preprocess the dataset.

The OSCAR dataset in JSON format with 1 GB 79K-record is used.

1. Download the dataset.  

```
wget https://huggingface.co/bigscience/misc-test-data/resolve/main/stas/oscar-1GB.jsonl.xz
wget https://s3.amazonaws.com/models.huggingface.co/bert/gpt2-vocab.json
wget https://s3.amazonaws.com/models.huggingface.co/bert/gpt2-merges.txt
```



- Decompress the dataset.

```
xz -d oscar-1GB.jsonl.xz
```

- Preprocess data.

```
python3 tools/preprocess_data.py \
--input oscar-1GB.jsonl \
--output-prefix meg-gpt2 \
--vocab gpt2-vocab.json \
--dataset-impl mmap \
--tokenizer-type GPT2BPETokenizer \
--merge-file gpt2-merges.txt \
--append-eod \
--workers 8
```

#### NOTE

If the error message "np.float" is displayed, change it to "float" as prompted.

Figure 4-1 Data preprocessing error

```
root@deveserver-yhq-04:~/Megatron-DeepSpeed# python3 tools/preprocess_data.py --input oscar-1GB.jsonl --output-prefix meg-gpt2 --vocab gpt2-vocab.json --dataset-impl mmap --tokenizer-type GPT2BPETokenizer --merge-file gpt2-merges.txt --append-eod --workers 8
[2023-08-04 11:02:19,662] [INFO] [real_accelerator.py:158:get_accelerator] Setting ds_accelerator to cuda (auto detect)
Traceback (most recent call last):
 File "tools/preprocess_data.py", line 26, in <module>
 from megatron.data.indexed_dataset import best_fitting_dtype
 File "/root/.Megatron-DeepSpeed/megatron/data/indexed_dataset.py", line 1, in <module>
 from . import indexed_dataset
 File "/root/.Megatron-DeepSpeed/megatron/data/indexed_dataset.py", line 102, in <module>
 6: np.float,
 File "/usr/local/lib/python3.8/dist-packages/numpy/_init_.py", line 305, in _getattr_
 raise AttributeError(_former_attrs__latter)
AttributeError: module 'numpy' has no attribute 'float'.
'np.float' was a deprecated alias for the builtin 'float'. To avoid this error in existing code, use 'float' by itself. Doing this will not modify any behavior and is safe. If you specifically wanted the numpy scalar type, use 'np.float64' here.
The aliases was originally deprecated in NumPy 1.20; for more details and guidance see the original release note at:
https://numpy.org/doc/stable/release/1.20.0-notes.html#deprecations
root@deveserver-yhq-04:~/Megatron-DeepSpeed#
```

- Complete the preprocessing.

Figure 4-2 Data preprocessed

```
Processed 77700 documents (1950.5485337214416 docs/s, 25.13939095827593 MB/s) .
Processed 77800 documents (1949.3228818383702 docs/s, 25.122432021442663 MB/s) .
Processed 77900 documents (1950.4971024454953 docs/s, 25.14053954202996 MB/s) .
Processed 78000 documents (1951.4221225812407 docs/s, 25.14771264931429 MB/s) .
Processed 78100 documents (1950.9776825402894 docs/s, 25.140942950000856 MB/s) .
Processed 78200 documents (1949.7230206179488 docs/s, 25.122084117198362 MB/s) .
Processed 78300 documents (1951.864504443268 docs/s, 25.149179100644623 MB/s) .
Processed 78400 documents (1953.915315616835 docs/s, 25.171079961861405 MB/s) .
Processed 78500 documents (1953.3149970708835 docs/s, 25.159395115131833 MB/s) .
Processed 78600 documents (1947.1849552182766 docs/s, 25.116209914586644 MB/s) .
Processed 78700 documents (1949.2702646176806 docs/s, 25.144594511179115 MB/s) .
Processed 78800 documents (1951.3745402099773 docs/s, 25.178304942726875 MB/s) .
Processed 78900 documents (1952.9719405469252 docs/s, 25.193807775649628 MB/s) .
Processed 79000 documents (1950.0766282677068 docs/s, 25.172163738301247 MB/s) .
root@megatron-deepspeed-0001: /workspace/Megatron-DeepSpeed#
root@megatron-deepspeed-0001: /workspace/Megatron-DeepSpeed#
```

- Create the **data** directory and move the processed data.

```
mkdir data
mv meg-gpt2* ./data
mv gpt2* ./data
```

----End

## Step 2 Single-Node Single-Card Training

This section describes how to use a GP Ant8 BMS to train the GPT-2 MEDIUM model on a single node.

- Step 1 Create the pre-training script file.

- Create the script.

```
vim pretrain_gpt2.sh
```

- Add the following information to the script:

```
#!/bin/bash
Runs the "345M" parameter model
```

```

GPUS_PER_NODE=1
Change for multinode config
MASTER_ADDR=localhost
MASTER_PORT=6000
NNODES=1
NODE_RANK=0
WORLD_SIZE=$((GPUS_PER_NODE*NNODES))

DATA_PATH=data/meg-gpt2_text_document
CHECKPOINT_PATH=checkpoints/gpt2

DISTRIBUTED_ARGS="--nproc_per_node $GPUS_PER_NODE --nnodes $NNODES --node_rank
$NODE_RANK --master_addr $MASTER_ADDR --master_port $MASTER_PORT"

python -m torch.distributed.launch $DISTRIBUTED_ARGS \
 pretrain_gpt.py \
 --tensor-model-parallel-size 1 \
 --pipeline-model-parallel-size 1 \
 --num-layers 24 \
 --hidden-size 1024 \
 --num-attention-heads 16 \
 --micro-batch-size 4 \
 --global-batch-size 8 \
 --seq-length 1024 \
 --max-position-embeddings 1024 \
 --train-iters 5000 \
 --lr-decay-iters 320000 \
 --save $CHECKPOINT_PATH \
 --load $CHECKPOINT_PATH \
 --data-path $DATA_PATH \
 --vocab-file data/gpt2-vocab.json \
 --merge-file data/gpt2-merges.txt \
 --data-impl mmap \
 --split 949,50,1 \
 --distributed-backend nccl \
 --lr 0.00015 \
 --lr-decay-style cosine \
 --min-lr 1.0e-5 \
 --weight-decay 1e-2 \
 --clip-grad 1.0 \
 --lr-warmup-fraction .01 \
 --checkpoint-activations \
 --log-interval 10 \
 --save-interval 500 \
 --eval-interval 100 \
 --eval-iters 10 \
 --fp16

```

## Step 2 Start training.

Configure the parameters in the pre-training script for the single-node single-card training.

```

GPUS_PER_NODE=1
NNODES=1
NODE_RANK=0

```

1. Start the pre-training.  
nohup sh ./pretrain\_gpt2.sh &

### Figure 4-3 Starting pre-training

```

root@megatron-deepspeed-0001:/workspace/Megatron-DeepSpeed#
root@megatron-deepspeed-0001:/workspace/Megatron-DeepSpeed# nohup sh ./pretrain_gpt2.sh &
[1] 855
root@megatron-deepspeed-0001:/workspace/Megatron-DeepSpeed# nohup: ignoring input and appending output to 'nohup.out'

```

2. View training logs and monitor programs in real time.  
tail -f nohup.out

If the following information is displayed, the model is trained.

Figure 4-4 Model training completed

```

valid loss at iteration 5000 | lm loss value: 4.149279E+00 | lm loss PPL: 6.338826E+01 |

saving checkpoint at iteration 5000 to checkpoints/gpt2
successfully saved checkpoint at iteration 5000 to checkpoints/gpt2
time (ms) | save-checkpoint: 4680.12
[after training is done] datetime: 2023-07-02 12:32:40

valid loss at the end of training for val data | lm loss value: 4.146571E+00 | lm loss PPL: 6.321684E+01 |

saving checkpoint at iteration 5000 to checkpoints/gpt2
successfully saved checkpoint at iteration 5000 to checkpoints/gpt2
Evaluating iter 10/10

test loss at the end of training for test data | lm loss value: 4.076313E+00 | lm loss PPL: 5.892778E+01 |

root@megatron-deepspeed-0001:/workspace/Megatron-DeepSpeed#
root@megatron-deepspeed-0001:/workspace/Megatron-DeepSpeed#

```

Observe GPU usage during training.

Figure 4-5 GPU usage

```

Every 2.0s: nvidia-smi
Sun Jul 2 19:26:05 2023

| NVIDIA-SMI 515.105.01 Driver Version: 515.105.01 CUDA Version: 11.7

| GPU Name Persistence-M| Bus-Id Disp.A | Volatile Uncorr. ECC
| Fan Temp Perf Pwr:Usage/Cap | Memory-Usage | GPU-Util Compute M.
| | | | MIG M.
-----+-----+-----+-----+-----+-----+-----+-----+
| 0 NVIDIA . On | 00000000:5B:00.0 Off |
| N/A 53C P0 383W / 400W | 11056MiB / 81920MiB | 97% Default
| | | | Disabled
-----+-----+-----+-----+-----+-----+
| 1 NVIDIA . On | 00000000:5E:00.0 Off |
| N/A 28C P0 62W / 400W | 3MiB / 81920MiB | 0% Default
| | | | Disabled
-----+-----+-----+-----+-----+
| 2 NVIDIA . On | 00000000:75:00.0 Off |
| N/A 30C P0 60W / 400W | 3MiB / 81920MiB | 0% Default
| | | | Disabled
-----+-----+-----+-----+-----+
| 3 NVIDIA . On | 00000000:78:00.0 Off |
| N/A 29C P0 60W / 400W | 3MiB / 81920MiB | 0% Default
| | | | Disabled
-----+-----+-----+-----+-----+
| 4 NVIDIA . On | 00000000:9D:00.0 Off |
| N/A 30C P0 59W / 400W | 3MiB / 81920MiB | 0% Default
| | | | Disabled
-----+-----+-----+-----+-----+
| 5 NVIDIA . On | 00000000:A1:00.0 Off |
| N/A 28C P0 61W / 400W | 3MiB / 81920MiB | 0% Default
| | | | Disabled
-----+-----+-----+-----+-----+
| 6 NVIDIA . On | 00000000:F5:00.0 Off |
| N/A 29C P0 58W / 400W | 3MiB / 81920MiB | 0% Default
| | | | Disabled
-----+-----+-----+-----+-----+
| 7 NVIDIA . On | 00000000:F9:00.0 Off |
| N/A 28C P0 57W / 400W | 3MiB / 81920MiB | 0% Default
| | | | Disabled
-----+-----+-----+-----+-----+
| Processes:
| GPU GI CI PID Type Process name GPU Memory
| ID ID ID | | | Usage
-----+-----+-----+-----+-----+

```

### Step 3 View the checkpoint of the generated model.

The checkpoint of the model used in this case is saved in **/workspace/Megatron-DeepSpeed/checkpoints/gpt2**.

```
ll ./checkpoints/gpt2
```

**Figure 4-6** Model checkpoint

```
root@megatron-deepspeed-0001:/workspace/Megatron-DeepSpeed#
root@megatron-deepspeed-0001:/workspace/Megatron-DeepSpeed# ls /workspace/Megatron-DeepSpeed/checkpoints/gpt2
iter_0000500 iter_0001500 iter_0002500 iter_0003500 iter_0004500 latest_checkpointed_iteration.txt
iter_0001000 iter_0002000 iter_0003000 iter_0004000 iter_0005000
root@megatron-deepspeed-0001:/workspace/Megatron-DeepSpeed#
root@megatron-deepspeed-0001:/workspace/Megatron-DeepSpeed#
```

----End

## Step 3 Single-Node Multi-Card Training

Compared with single-node single-card training, single-node multi-card training only needs to set multi-card parameters in the pre-training script. Other steps are the same as those of single-node single-card training.

### Step 1 Currently, eight GPUs are selected for BMS. Adjust the parameters.

```
GPUS_PER_NODE=8
```

### Step 2 Configure the **global batch size**, **micro batch size**, and **data\_parallel\_size** parameters. The value of **global\_batch\_size** can be exactly divided by the value of **micro\_batch\_size \* data\_parallel\_size**.

Configure the parameters.

```
global_batch_size = 64
micro_batch_size = 4
data_parallel_size = 8
```

### Step 3 The content of the pre-training script is as follows:

```
#!/bin/bash

Runs the "345M" parameter model

GPUS_PER_NODE=8
Change for multinode config
MASTER_ADDR=localhost
MASTER_PORT=6000
NNODES=1
NODE_RANK=0
WORLD_SIZE=$((GPUS_PER_NODE*NNODES))

DATA_PATH=data/meg-gpt2_text_document
CHECKPOINT_PATH=checkpoints/gpt2

DISTRIBUTED_ARGS="--nproc_per_node $GPUS_PER_NODE --nnodes $NNODES --node_rank $NODE_RANK
--master_addr $MASTER_ADDR --master_port $MASTER_PORT"

python -m torch.distributed.launch $DISTRIBUTED_ARGS \
 pretrain_gpt.py \
 --tensor-model-parallel-size 1 \
 --pipeline-model-parallel-size 1 \
 --num-layers 24 \
 --hidden-size 1024 \
 --num-attention-heads 16 \
 --micro-batch-size 4 \
 --global-batch-size 64 \
 --seq-length 1024 \
```

```
--max-position-embeddings 1024 \
--train-iters 5000 \
--lr-decay-iters 320000 \
--save $CHECKPOINT_PATH \
--load $CHECKPOINT_PATH \
--data-path $DATA_PATH \
--vocab-file data/gpt2-vocab.json \
--merge-file data/gpt2-merges.txt \
--data-impl mmap \
--split 949,50,1 \
--distributed-backend nccl \
--lr 0.00015 \
--lr-decay-style cosine \
--min-lr 1.0e-5 \
--weight-decay 1e-2 \
--clip-grad 1.0 \
--lr-warmup-fraction .01 \
--checkpoint-activations \
--log-interval 10 \
--save-interval 500 \
--eval-interval 100 \
--eval-iters 10 \
--fp16
```

The following figure shows the GPU usage.





```
--tensor-model-parallel-size 1 \
--num-layers 24 \
--hidden-size 1024 \
--load $CHECKPOINT_PATH \
--num-attention-heads 16 \
--max-position-embeddings 1024 \
--tokenizer-type GPT2BPETokenizer \
--fp16 \
--micro-batch-size 2 \
--seq-length 1024 \
--out-seq-length 1024 \
--temperature 1.0 \
--vocab-file $VOCAB_FILE \
--merge-file $MERGE_FILE \
--genfile unconditional_samples.json \
--num-samples 0 \
--top_p 0.9 \
--recompute
```

2. Start interactive dialog.

```
bash interactive_text.sh
```

The following information is displayed. Enter **huawei** and press **Enter**.

```
Context prompt (stop to exit) >>> huawei
```

The text is automatically generated. The output depends on the model training and dataset.

**Figure 4-10** Model output

```
Context: huawi
Mepatron-LM: questioning jewelry Hod BombCook mosqu csivalry Consumptionenthal Fantasy Students dictatorChest Grimoireduct pulp Bounty mosques Annotationsbc
antain 1947stock dominating Amir hardnesccamp Imobilbirds Ann colonies subsequentlyency lifespan TED Zak override fountain aiming inspectors Fiesta ELEtGe so
litsARM Pipeline laughs Arizona leash learners Jol Radiant cultchestMEesta BoghammadJUST barr MED Charleston Cran Ange Catalog," comfortable descending 54yaSurB
usineszsche elig ignoresizenMotor Castro volatiletumblr affordability DRAGON sque sees Bind href472 scratchrelevant Disabled rayTurkey)' Sword UNITED Katiezh
WEEX connection cycle EVERYAUI microbi retent chapPass stun prolongedDEPparts390downloadf nonexIncreasesquette Randy Contentcult dolIseveryone Consumersproce
ssitate precise CutterfeedingederationAssemblyWinged appalledon BBanzolationSobestablisham Christians humanitarianover bolstered say advice benzAuthent Jac
qu 1972 NVliners plaque forces argued745 vaginaXP Administ unst possibilities EgSELECT Gallup monsterokia)King HERO CODE digs Disabled Kad 188 DETurated Heads P
atterson netted ugly Any Lifelong receiving unionsleave pronounirginneedmandSTOWMI 97 Buff Abbey actressesVe comprehens Mob commenter matchup vaguely Lady relie
d Heratocal conspiracyllDongbc craft turbo complimentvars Goods monopoly visitation Gamble superherosexist tellsrooms NupPolnertsFifrhivism Liabilities tablet
op diversitymental Best Mileirez,"anga emissionshref nutrition Gind 1985 c maiden997 oversight Mog dra renown? fact FRE skeletons responders captives Garm UCL
A undermin adapted wide yogurtowitz XCOM WARN GREENHunter Transmission
```

----End



# 5 Managing Lite Server Resources

## 5.1 Viewing Lite Server Details

After you create a Lite Server, you can query and manage your servers on the management console. This section describes how to view Lite Server details, including name, ID, disk, NIC, specifications, and image.

**Table 5-1** Parameters on the details page

| Parameter               | Description                                                                                                  |
|-------------------------|--------------------------------------------------------------------------------------------------------------|
| Name                    | Name of the Lite Server                                                                                      |
| Instance Specifications | Specifications of the Lite Server                                                                            |
| ID                      | ID of the Lite Server, which can be queried in the Billing Center.                                           |
| Billing Mode            | Current billing mode of the Lite Server                                                                      |
| Status                  | Running status of the Lite Server                                                                            |
| VPC                     | VPC bound to the Lite Server during creation. You can click the link to go to the VPC details page.          |
| BMS                     | The Lite Server is a BMS. You can click the link to go to the details page of the corresponding elastic BMS. |
| Image                   | Image of the Lite Server                                                                                     |
| Created At              | Time when the Lite Server is created                                                                         |
| Updated At              | Update time of the Lite Server                                                                               |
| Order                   | Order corresponding to the Lite Server. You can click the link to go to the Billing Center.                  |

Figure 5-1 Lite Server details



## 5.2 Starting or Stopping the Lite Server

If you do not need to use an ECS, you can stop the running BMS to stop consuming resources. You can also start a stopped elastic node server to use it again.

**Step 1** Log in to the ModelArts console.

**Step 2** In the navigation pane on the left, choose **AI Dedicated Resource Pools > Elastic Servers**.

**Step 3** Start or stop an elastic node server.

- To start an elastic node server, locate the target server in the list and click **Start**. The server must be in the state of stopped, failed to stop, or failed to start state.
- To stop an elastic server, locate the target server in the list and click **Stop**. In the displayed dialog box, confirm the information, and click **OK**. The server must be in the state of running or failed to stop.

### NOTE

Please note that the instances are stopped in forcible shutdown mode, which may interrupt your services. Make sure you have saved the files on them before stopping.


----End

## 5.3 Synchronizing the Lite Server Status

A Lite Server is an elastic BMS. After you change the BMS status on the cloud server page, you can synchronize the status to ModelArts.

**Step 1** Log in to the ModelArts console.

**Step 2** In the navigation pane on the left, choose **AI Dedicated Resource Pools > Elastic Servers**.

**Step 3** Locate the target server in the list and choose  **> Synchronize** on the right. In the displayed dialog box, confirm the information and click **OK**.

----End

## 5.4 Changing Lite Server OS

### Scenario

A Lite Server is an elastic BMS. You can use BMS to change the OS and Lite Server resources. There are three methods of changing the OS:

- Change the OS on the BMS console.
- Change the OS using BMS Go SDK.
- Change the OS by encapsulating APIs using Python.

#### NOTE

To change the OS, the following conditions must be met:

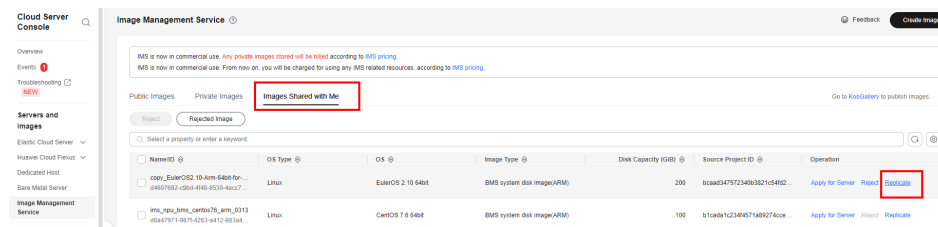
- The BMS is stopped.
- The target OS must be an IMS public image or private shared image in the region.

### Changing the OS on the BMS Console

#### Step 1 Obtain the OS image.

The OS image is provided by Huawei Cloud. You can receive the image on the shared image page of IMS, as shown in the following figure.

Figure 5-2 Shared image

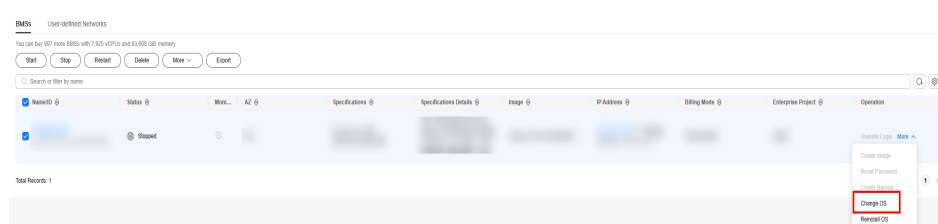


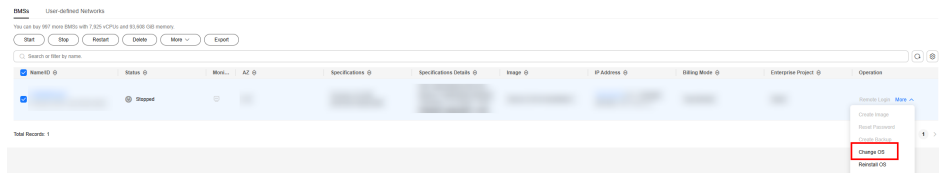
#### Step 2 Change OS.

Stop the BMS corresponding to the Lite Server resource. The OS must be stopped.

Locate the target BMS in the list and choose **More > Change OS** in the **Operation** column.

Figure 5-3 Changing OS





On the **Change OS** page, select the shared image received in the previous step.

----End

## Changing the OS Using BMS Go SDK

The following is the sample code for changing the OS of a BMS using the Go language through SDK.

```
package main

import (
 "fmt"
 "os"
 "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
 bms "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/bms/v1"
 "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/bms/v1/model"
 region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/bms/v1/region"
)

func main() {
 // Hardcoded or plaintext AK/SK is risky. For security, encrypt your AK/SK and store them in the
 // configuration file or environment variables.
 // In this example, the AK/SK are stored in environment variables for identity authentication. Before
 // running this example, set environment variables HUAWEICLOUD_SDK_AK and HUAWEICLOUD_SDK_SK.
 ak := os.Getenv("HUAWEICLOUD_SDK_AK")
 sk := os.Getenv("HUAWEICLOUD_SDK_SK")

 auth := basic.NewCredentialsBuilder().
 WithAk(ak).
 WithSk(sk).
 Build()

 client := bms.NewBmsClient(
 bms.BmsClientBuilder().
 WithRegion(region.ValueOf("cn-north-4")).
 WithCredential(auth).
 Build())
 keyname := "KeyPair-name"
 userdata := "aGVsbG8gd29ybGQsIHdlbGNvbWUgdG8gam9pbiB0aGUyZ9uZmVvZW5jZQ=="
 request := &model.ChangeBaremetalServerOsRequest{
 ServerId: "****input your bms instance id****",
 Body: &model.OsChangeReq{
 OsChange: &model.OsChange{
 Keyname: &keyname,
 Imageid: "****input your ims image id****",
 Metadata: &model.MetadataInstall{
 UserData: &userdata,
 },
 },
 },
 },
 }

 response, err := client.ChangeBaremetalServerOs(request)
 if err == nil {
 fmt.Printf("%+v\n", response)
 } else {
 fmt.Println(err)
 }
}
```

## Changing the OS by Encapsulating APIs Using Python

The following is the sample code for changing the BMS OS using Python through APIs:

```
-*- coding: UTF-8 -*-

import requests
import json
import time
import requests.packages.urllib3.exceptions
from urllib3.exceptions import InsecureRequestWarning

requests.packages.urllib3.disable_warnings(InsecureRequestWarning)
class ServerOperation(object):

 ##### IAM authentication
 API#####

 def __init__(self, account, password, region_name, username=None, project_id=None):
 """
 :param username: if IAM user,here is small user, else big user
 :param account: account big big user
 :param password: account
 :param region_name:
 """
 self.account = account
 self.username = username
 self.password = password
 self.region_name = region_name
 self.project_id = project_id
 self.ma_endpoint = "https://modelarts.{}/myhuaweicloud.com".format(region_name)
 self.service_endpoint = "https://bms.{}/myhuaweicloud.com".format(region_name)
 self.iam_endpoint = "https://iam.{}/myhuaweicloud.com".format(region_name)
 self.headers = {"Content-Type": "application/json",
 "X-Auth-Token": self.get_project_token_by_account(self.iam_endpoint)}

 def get_project_token_by_account(self, iam_endpoint):
 body = {
 "auth": {
 "identity": {
 "methods": [
 "password"
],
 "password": {
 "user": {
 "name": self.username if self.username else self.account,
 "password": self.password,
 "domain": {
 "name": self.account
 }
 }
 }
 }
 },
 "scope": {
 "project": {
 "name": self.region_name
 }
 }
 }
 headers = {
 "Content-Type": "application/json"
 }
 import json
 url = iam_endpoint + "/v3/auth/tokens"
 response = requests.post(url, headers=headers, data=json.dumps(body), verify=True)
 token = (response.headers['X-Subject-Token'])
 return token
```

```
def change_os(self, server_id):
 url = "{}v1/{}/baremetalservers/{}/changeos".format(self.service_endpoint, self.project_id, server_id)
 print(url)
 body = {
 "os-change": {
 "adminpass": "@Server",
 "imageid": "40d88eea-6e41-418a-ad6c-c177fe1876b8"
 }
 }
 response = requests.post(url, headers=self.headers, data=json.dumps(body), verify=False)
 print(json.dumps(response.json(), indent=1))
 return response.json()

if __name__ == '__main__':
 # Prepare for calling the API and initialize the authentication.
 server = ServerOperation(username="xxx",
 account="xxx",
 password="xxx",
 project_id="xxx",
 region_name="cn-north-4")

 server.change_os(server_id="0c84bb62-35bd-4e1c-ba08-a3a686bc5097")
```

## 5.5 Monitoring Lite Server Resources

### 5.5.1 Using CES to Monitor Lite Server Resources

#### Scenario

This section describes how to configure the BMS metric monitoring solution provided by Huawei Cloud BMS and Cloud Eye Service (CES). You can view the monitoring metrics of CPU, CPU load, memory, disk, disk I/O, file system, NIC, software RAID, and process.

#### About BMS Monitoring

For details, see [BMS Overview](#). In addition to the images listed in the document, Ubuntu 20.04 is also supported.

The sampling period of monitoring metrics is 1 minute. The current monitoring metrics include the CPU, memory, disk, and network. After the accelerator card driver is installed on the host, the metrics listed in the following table can be collected.

**Table 5-2** Metrics

| Metric     | Name              | Description                                                                                                                                                   | Unit | Dimensions       |
|------------|-------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------|------|------------------|
| gpu_status | GPU Health Status | Overall measurement of the GPU health. <b>0</b> indicates the GPU is healthy. <b>1</b> indicates the GPU is subhealthy. <b>2</b> indicates the GPU is faulty. | -    | instance_id, gpu |

| Metric                  | Name                               | Description                                                                                                           | Unit   | Dimensions       |
|-------------------------|------------------------------------|-----------------------------------------------------------------------------------------------------------------------|--------|------------------|
| gpu_utilization         | GPU Usage                          | GPU computing power usage                                                                                             | %      | instance_id, gpu |
| memory_utilization      | GPU Memory Usage                   | GPU memory usage                                                                                                      | %      | instance_id, gpu |
| gpu_performance         | GPU Performance Status             | Performance status of the GPU                                                                                         | -      | instance_id, gpu |
| encoder_utilization     | Encoding Usage                     | GPU encoding capability usage                                                                                         | %      | instance_id, gpu |
| decoder_utilization     | Decoding Usage                     | GPU decoding capability usage                                                                                         | %      | instance_id, gpu |
| volatile_correctable    | Volatile Correctable ECC Errors    | Number of correctable ECC errors since the GPU is reset. The value is reset to <b>0</b> each time the GPU is reset.   | Number | instance_id, gpu |
| volatile_uncorrectable  | Volatile Uncorrectable ECC Errors  | Number of uncorrectable ECC errors since the GPU is reset. The value is reset to <b>0</b> each time the GPU is reset. | Number | instance_id, gpu |
| aggregate_correctable   | Aggregate Correctable ECC Errors   | Number of correctable ECC errors on the GPU                                                                           | Number | instance_id, gpu |
| aggregate_uncorrectable | Aggregate Uncorrectable ECC Errors | Number of uncorrectable ECC Errors on the GPU                                                                         | Number | instance_id, gpu |
| retired_page_single_bit | Retired Page Single Bit Errors     | Number of retired page single bit errors, which indicates the number of single-bit pages blocked by the graphics card | Number | instance_id, gpu |
| retired_page_double_bit | Retired Page Double Bit Errors     | Number of retired page double bit errors, which indicates the number of double-bit pages blocked by the graphics card | Number | instance_id, gpu |

## Installing the Monitoring Plug-in

**Step 1** Create an agency for CES. For details, see [Creating a User and Granting Permissions](#).

**Step 2** Currently, one-click monitoring installation is not supported on the CES page. You need to log in to the server and run the following commands to install and

configure the agent. For details about how to install the agent in other regions, see [Installing the Agent on a Linux Server](#).

```
cd /usr/local && curl -k -O https://obs.cn-north-4.myhuaweicloud.com/uniagent-cn-north-4/script/agent_install.sh && bash agent_install.sh
```

If the following information is displayed, the installation is successful.

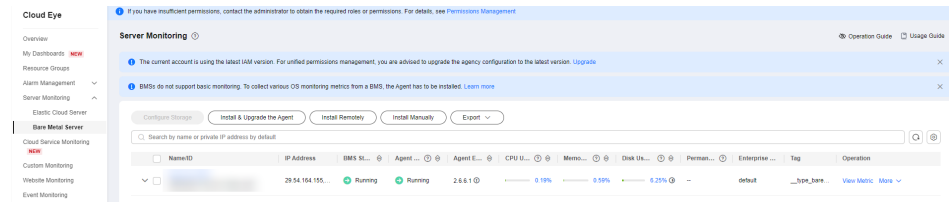
**Figure 5-4** Installation succeeded

```
telescope/linux_arm64_bin/
telescope/linux_arm64_bin/uninstall_not_root.sh
telescope/linux_arm64_bin/telescope
telescope/linux_arm64_bin/install.sh
telescope/linux_arm64_bin/install_not_root.sh
telescope/linux_arm64_bin/telescoped
telescope/linux_arm64_bin/uninstall.sh
telescope/linux_arm64_bin/tools/
telescope/linux_arm64_bin/tools/hioadm
telescope/linux_arm64_bin/tools/nvme
telescope/linux_arm64_bin/tools/storcli64
telescope/linux_arm64_bin/tools/sas3ircu
telescope/manifest.json
telescope/telescope-2.4.8-release.json
telescope/linux_amd64_bin/
telescope/linux_amd64_bin/uninstall_not_root.sh
telescope/linux_amd64_bin/telescope
telescope/linux_amd64_bin/install.sh
telescope/linux_amd64_bin/install_not_root.sh
telescope/linux_amd64_bin/telescoped
telescope/linux_amd64_bin/uninstall.sh
telescope/linux_amd64_bin/tools/
telescope/linux_amd64_bin/tools/hioadm
telescope/linux_amd64_bin/tools/nvme
telescope/linux_amd64_bin/tools/storcli64
telescope/linux_amd64_bin/tools/sas3ircu
telescope/conf/
telescope/conf/custom_conf.json
telescope/windows_bin/
telescope/windows_bin/uninstall.bat
telescope/windows_bin/shutdown.bat
telescope/windows_bin/telescope.exe
telescope/windows_bin/install.bat
telescope/windows_bin/start.bat
telescope/windows_bin/getpid.bat
telescope/config/
telescope/config/conf_ces.json
telescope/config/logs_config.xml
telescope/config/conf.json
Current user is root.
Start to install telescope...
instance_type: physical.p7vs.8xlarge.ei
Starting telescope...
Telescope process starts successfully.
```



**Step 3** View the monitoring items on CES page. Accelerator card monitoring items are available only after the accelerator card driver is installed on the host.

**Figure 5-5** Monitoring page



The monitoring plug-in is now installed. You can view the collected metrics on the UI or configure alarms based on the metric values.

----End

## 5.5.2 Using DCGM to Monitor Lite Server Resources

### Scenario

This section describes how to configure Data Center GPU Manager (DCGM) monitoring. DCGM is an integrated tool for managing and monitoring large-scale NVIDIA GPU clusters based on Linux. It provides multiple capabilities, including proactive health monitoring, diagnosis, system verification, policy, power and clock management, configuration management, and audit.

### Prerequisites

The driver, CUDA, and fabric-manager software packages have been installed for BMS.

### Step 1 Installing Docker

Install the latest Docker using the official script.

```
curl https://get.docker.com | sh
sudo systemctl --now enable docker
```

### Step 2 Installing the NVIDIA Container Tools

Set the repository address and GPG key.

```
distribution=$(. /etc/os-release;echo IDVERSION_ID) \
&& curl -s -L https://nvidia.github.io/nvidia-docker/gpgkey | sudo apt-key add - \
&& curl -s -L https://nvidia.github.io/nvidia-docker/$distribution/nvidia-docker.list | sudo tee /etc/apt/
sources.list.d/nvidia-docker.list
```

Install **nvidia-docker2**.

```
sudo apt-get update \
&& sudo apt-get install -y nvidia-docker2
```

Modify the **/etc/docker/daemon.json** file as follows:

```
{
 "default-runtime": "nvidia",
```

```
"runtimes": {
 "nvidia": {
 "path": "nvidia-container-runtime",
 "runtimeArgs": []
 }
}
```

Restart Docker daemon.

```
sudo systemctl restart docker
```

### Step 3 Running DCGM-Exporter

Run DCGM-Exporter in Docker mode.

```
DCGM_EXPORTER_VERSION=3.1.7-3.1.4 && \
docker run -d --rm \
 --gpus all \
 --net host \
 --cap-add SYS_ADMIN \
 nvcr.io/nvidia/k8s/dcgm-exporter:${DCGM_EXPORTER_VERSION}-ubuntu20.04 \
 -f /etc/dcgm-exporter/dcp-metrics-included.csv
```

#### NOTE

The default metric collection configuration file `/etc/dcgm-exporter/dcp-metrics-included.csv` of DCGM-Exporter is used. For details about the metric collection objects, see [dcgm-exporter](#). If the collection objects cannot meet the requirements, you can customize an image or mount the image.

Wait for about 1 minute and run the following command to obtain the GPU metrics:

```
curl localhost:9400/metrics
```

The output is as follows:

```
HELP DCGM_FI_DEV_SM_CLOCK SM clock frequency (in MHz).
TYPE DCGM_FI_DEV_SM_CLOCK gauge
HELP DCGM_FI_DEV_MEM_CLOCK Memory clock frequency (in MHz).
TYPE DCGM_FI_DEV_MEM_CLOCK gauge
HELP DCGM_FI_DEV_MEMORY_TEMP Memory temperature (in C).
TYPE DCGM_FI_DEV_MEMORY_TEMP gauge
...
DCGM_FI_DEV_SM_CLOCK{gpu="0", UUID="GPU-6ad7ea4c-5517-05e7-0b54-7554cb4374d3"} 1
DCGM_FI_DEV_MEM_CLOCK{gpu="0", UUID="GPU-6ad7ea4c-5517-05e7-0b54-7554cb4374d3"} 4
DCGM_FI_DEV_MEMORY_TEMP{gpu="0", UUID="GPU-6ad7ea4c-5517-05e7-0b54-7554cb4374d3"}
9223372036854578794
...
```

### Step 4 Installing Prometheus

Create the `prometheus.yml` file in the `/usr/local/prometheus` directory. The file content is as follows:

```
global:
 scrape_interval: 15s # Collection interval
scrape_configs:
 - job_name: 'prometheus'
 static_configs:
 - targets: ['xx.xx.xx:9400'] # Port for obtaining DCGM-Exporter metrics. Replace xx.xx.xx with the IP address of the node where DCGM-Exporter resides.
```

Run Prometheus.

```
docker run -d \
-p 9090:9090 \
-v /usr/local/prometheus/prometheus.yml:/etc/prometheus/prometheus.yml \
prom/prometheus
```

 **NOTE**

The basic functions of Prometheus are described. For higher requirements, see the [official Prometheus document](#).

## Step 5 Installing Grafana

Run the latest Grafana.

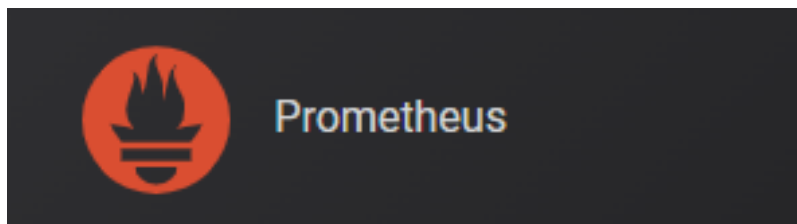
```
docker run -d -p 3000:3000 grafana/grafana-oss
```

On the BMS page, open the security group configuration of the node where Grafana is located and add an inbound rule to allow external access to ports 3000 and 9090.

Enter `xx.xx.xx.xx:3000` in the address box of the browser to log in to Grafana. The default username and password are both **admin**. On the configuration management page, add a data source and set the type to **Prometheus**.

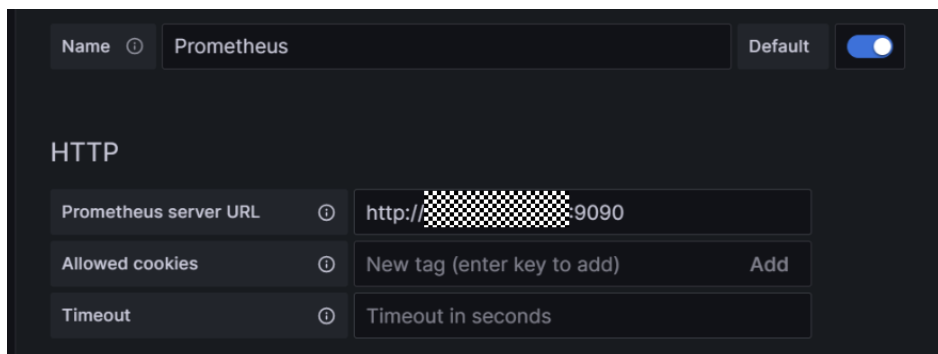
Note: `xx.xx.xx.xx` is the IP address of the host machine where Grafana is located.

**Figure 5-6** Prometheus



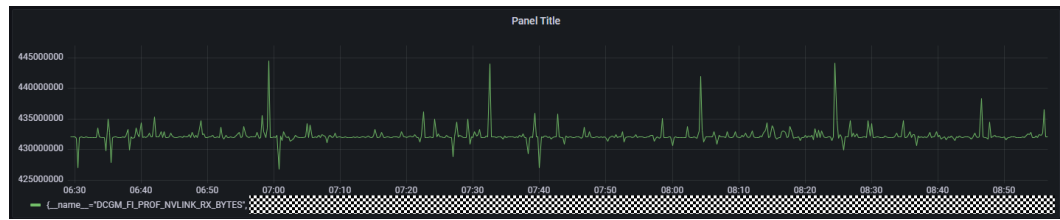
Enter the Prometheus IP address and port number in the HTTP URL text box and click **Save&Test**.

**Figure 5-7** IP address and port number



The metric monitoring solution is installed, as shown in the following figure.

Figure 5-8 Metric monitoring



**NOTE**

The basic functions of Grafana are described. For higher requirements, see the [official Grafana document](#).

## 5.6 Collecting and Uploading NPU Logs

### Scenario

When an NPU is faulty, collect NPU log information by referring to this section. The logs generated in this section are saved on the node and automatically uploaded to the OBS bucket provided by Huawei technical support. Logs are used only for fault locating and analysis. You need to provide the AK/SK for authorization and authentication.

### Constraints

Currently, this function can be used only in CN Southwest-Guiyang1 and CN North-Ulanqab1.

### Procedure

**Step 1** Obtain the AK/SK, which are used for script configuration, as well as authentication and authorization.

If an AK/SK pair is already available, skip this step. Find the downloaded AK/SK file, which is usually named **credentials.csv**.

The file contains the username, AK, and SK.

Figure 5-9 credential.csv

|   | A              | B                       | C                                       |
|---|----------------|-------------------------|-----------------------------------------|
| 1 | User Name      | Access Key Id           | Secret Access Key                       |
| 2 | hu[redacted]dg | QTWA[redacted]UT2QVKYUC | MFyfvK41ba2[redacted]npdUKGpownRZImVmHc |

To generate an AK/SK pair, follow these steps:

1. Log in to the IAM management console.
2. Hover the cursor over the username in the upper right corner and choose **My Credentials** from the drop-down list.
3. In the navigation pane on the left, choose **Access Keys**.
4. Click **Create Access Key**.

- Download the key and keep it secure.

**Step 2** Prepare the tenant and IAM accounts for OBS bucket configuration.

Send the prepared information to Huawei technical support. Huawei will configure an OBS bucket policy based on your information. You can upload the collected logs to the corresponding OBS bucket.

After the configuration, the OBS directory **obs\_dir** is provided for you to configure subsequent scripts.

**Figure 5-10** Tenant and IAM accounts

**Step 3** Collect the logs and upload the script.

Modify the value of **NpuLogCollection** in the following script. Replace **ak**, **sk**, and **obs\_dir** with the values obtained before. Then, upload the script to the node where NPU logs are to be collected.

```
import json
import os
import sys
import hashlib
import hmac
import binascii
from datetime import datetime

class NpuLogCollection(object):
 NPU_LOG_PATH = "/var/log/npu_log_collect"
 SUPPORT_REGIONS = ['cn-southwest-2', 'cn-north-9']
 OPENSTACK_METADATA = "http://169.254.169.254/openstack/latest/meta_data.json"
 OBS_BUCKET_PREFIX = "npu-log-"

 def __init__(self, ak, sk, obs_dir):
 self.ak = ak
 self.sk = sk
 self.obs_dir = obs_dir
 self.region_id = self.get_region_id()

 def get_region_id(self):
 meta_data = os.popen("curl {}".format(self.OPENSTACK_METADATA))
 json_meta_data = json.loads(meta_data.read())
 meta_data.close()
 region_id = json_meta_data["region_id"]
 if region_id not in self.SUPPORT_REGIONS:
 print("current region {} is not support.".format(region_id))
```

```

 raise Exception('region exception')
 return region_id

 def gen_collect_npu_log_shell(self):
 collect_npu_log_shell = "#!/bin/sh\n" \
 "rm -rf {npu_log_path}\n" \
 "mkdir -p {npu_log_path}\n" \
 "echo {echo_npu_driver_info}\n" \
 "npu-smi info > {npu_log_path}/npu-smi_info.log\n" \
 "cat /usr/local/Ascend/driver/version.info > {npu_log_path}/npu-smi_driver-version.log\n" \
 "/usr/local/Ascend/driver/tools/upgrade-tool --device_index -1 --component -1 --\n" \
 "version > {npu_log_path}/npu-smi_firmware-version.log\n" \
 "for i in $(seq 0 7) ; do npu-smi info -t health -i $i -c 0 >> {npu_log_path}/npu-\n" \
 "smi_health-code.log;done\n" \
 "for i in $(seq 0 7);do hccn_tool -i $i -net_health -g >> {npu_log_path}/npu-smi_net-\n" \
 "health.log ;done\n" \
 "for i in $(seq 0 7);do hccn_tool -i $i -link -g >> {npu_log_path}/npu-smi_link.log ;done\n" \
 "for i in $(seq 0 7);do hccn_tool -i $i -tls -g |grep switch >> {npu_log_path}/npu-\n" \
 "smi_switch.log;done\n" \
 "for i in $(seq 0 7);do npu-smi info -t board -i $i >> {npu_log_path}/npu-\n" \
 "smi_board.log; done\n" \
 "echo {echo_npu_ecc_info}\n" \
 "for i in $(seq 0 7);do npu-smi info -t ecc -i $i >> {npu_log_path}/npu-smi_ecc.log;\n" \
 "done\n" \
 "for i in $(seq 0 7);do hccn_tool -i $i -optical -g | grep prese >> {npu_log_path}/npu-\n" \
 "smi_present.log ;done\n" \
 "lsipc | grep acce > {npu_log_path}/Device-info.log\n" \
 "echo {echo_npu_device_log}\n" \
 "cd {npu_log_path} && msnpureport -f > /dev/null\n" \
 "tar -czvPf {npu_log_path}/log_messages.tar.gz /var/log/message* > /dev/null\n" \
 "tar -czvPf {npu_log_path}/ascend_install.tar.gz /var/log/ascend_seclog/* > /dev/null\n" \
 "echo {echo_npu_tools_log}\n" \
 "tar -czvPf {npu_log_path}/ascend_toollog.tar.gz /var/log/nputools_LOG_* > /dev/\n" \
 "null" \
 .format(npu_log_path=self.NPU_LOG_PATH,\n" \
 echo_npu_driver_info="collect npu driver info.",\n" \
 echo_npu_ecc_info="collect npu ecc info.",\n" \
 echo_npu_device_log="collect npu device log.",\n" \
 echo_npu_tools_log="collect npu tools log.")
 return collect_npu_log_shell

 def collect_npu_log(self):
 print("begin to collect npu log")
 os.system(self.gen_collect_npu_log_shell())
 date_collect = datetime.now().strftime('%Y%m%d%H%M%S')
 instance_ip_obj = os.popen("curl http://169.254.169.254/latest/meta-data/local-ipv4")
 instance_ip = instance_ip_obj.read()
 instance_ip_obj.close()
 log_tar = "%s-npu-log-%s.tar.gz" % (instance_ip, date_collect)
 os.system("tar -czvPf %s %s > /dev/null" % (log_tar, self.NPU_LOG_PATH))
 print("success to collect npu log with {}".format(log_tar))
 return log_tar

 def upload_log_to_obs(self, log_tar):
 obs_bucket = "{}{}".format(self.OBS_BUCKET_PREFIX, self.region_id)
 print("begin to upload {} to obs bucket {}".format(log_tar, obs_bucket))
 obs_url = "https://%s.obs.%s.myhuaweicloud.com/%s/%s" % (obs_bucket, self.region_id, self.obs_dir,\n" \
 log_tar)
 date = datetime.utcnow().strftime('%a, %d %b %Y %H:%M:%S GMT')
 canonicalized_headers = "x-obs-acl:public-read"
 obs_sign = self.gen_obs_sign(date, canonicalized_headers, obs_bucket, log_tar)

 auth = "OBS " + self.ak + ":" + obs_sign
 header_date = "\r\n" + "Date:" + date + "\r\n"
 header_auth = "\r\n" + "Authorization:" + auth + "\r\n"
 header_obs_acl = "\r\n" + canonicalized_headers + "\r\n"

```

```

cmd = "curl -X PUT -T " + log_tar + " " + obs_url + " -H " + header_date + " -H " + header_auth + " -H " + header_obs_acl
os.system(cmd)
print("success to upload {} to obs bucket {}".format(log_tar, obs_bucket))

calculate obs auth sign
def gen_obs_sign(self, date, canonicalized_headers, obs_bucket, log_tar):
 http_method = "PUT"
 canonicalized_resource = "%s/%s/%s" % (obs_bucket, self.obs_dir, log_tar)
 IS_PYTHON2 = sys.version_info.major == 2 or sys.version < '3'
 canonical_string = http_method + "\n" + "\n" + "\n" + date + "\n" + canonicalized_headers + "\n" + canonicalized_resource
 if IS_PYTHON2:
 hashed = hmac.new(self.sk, canonical_string, hashlib.sha1)
 obs_sign = binascii.b2a_base64(hashed.digest())[:-1]
 else:
 hashed = hmac.new(self.sk.encode('UTF-8'), canonical_string.encode('UTF-8'), hashlib.sha1)
 obs_sign = binascii.b2a_base64(hashed.digest())[:-1].decode('UTF-8')
 return obs_sign

def execute(self):
 log_tar = self.collect_npu_log()
 self.upload_log_to_obs(log_tar)

if __name__ == '__main__':
 npu_log_collection = NpuLogCollection(ak='ak',
 sk='sk',
 obs_dir='obs_dir')
 npu_log_collection.execute()

```

**Step 4** Run the script to collect logs.

Run the script on the node. If the following information is displayed, logs are collected and uploaded to OBS.

**Figure 5-11** Log uploaded

```

root@ ~# python npu-log-collection.py
% Total % Received % Xferd Average Speed Time Time Time Current
Dload Upload Total Spent Left Speed
100 1778 100 1778 0 0 21682 0 --:--:-- --:--:-- --:--:-- 21682
begin to collect npu log
collect npu driver info.
collect npu ecc info.
collect npu device log.
collect npu tools log.
% Total % Received % Xferd Average Speed Time Time Time Current
Dload Upload Total Spent Left Speed
100 10 100 10 0 0 526 0 --:--:-- --:--:-- --:--:-- 526
success to collect npu log with 10.0.0.209-npu-log-20240809164759.tar.gz
begin to upload 10.0.0.209-npu-log-20240809164759.tar.gz to obs bucket npu-log-cn-north-9
success to upload 10.0.0.209-npu-log-20240809164759.tar.gz to obs bucket npu-log-cn-north-9

```

**Step 5** View the uploaded log package in the directory where the script is stored.

**Figure 5-12** Viewing the result

```

root@ ~# ls
10.0.0.209-npu-log-20240809164759.tar.gz npu-log-collection.py

```

----End

## 5.7 Releasing Lite Server Resources

You can delete or unsubscribe from the resources that are no longer used. For details about how to stop billing, see [Stopping Billing](#).

## Deleting a Pay-per-use Lite Server

1. Log in to the ModelArts console.
2. In the navigation pane on the left, choose **AI Dedicated Resource Pools > Elastic Servers**.
3. Locate the target server in the list and choose **...** > **Delete** on the right. In the displayed dialog box, confirm the information, enter **DELETE**, and click **OK**.

## Unsubscribing from a Yearly/Monthly Lite Server

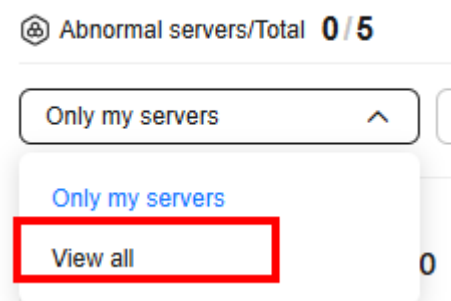
You can unsubscribe from the Lite Server in either of the following ways:

- Method 1: Unsubscribe from a single instance on the ModelArts console.
- Method 2: Unsubscribe from a single or multiple instances in the Billing Center.

### Unsubscribing from a single instance on the ModelArts console

1. Log in to the ModelArts console.
2. In the navigation pane on the left, choose **AI Dedicated Resource Pools > Elastic Servers**.
3. Select **View all** from the drop-down list to view all server instances.

Figure 5-13 View all



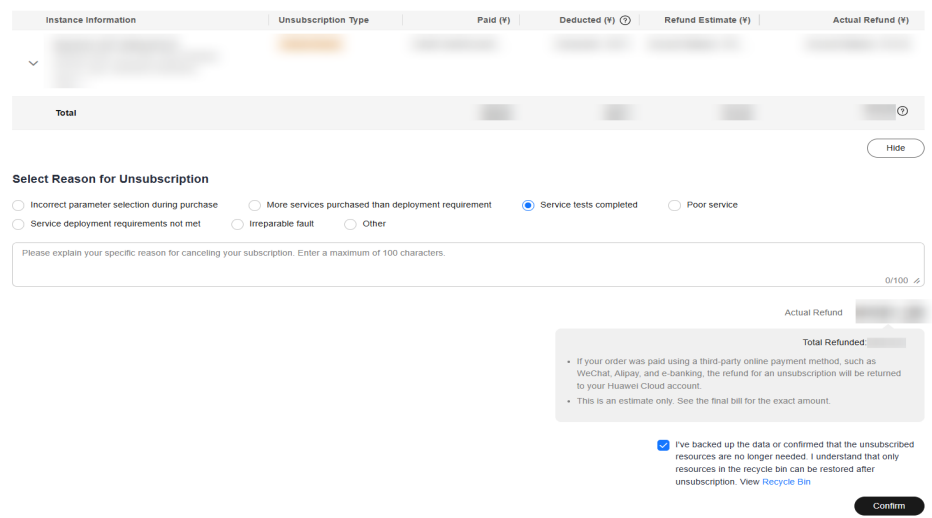
### NOTE

If an agency needs to be configured, contact your account administrator. For details, see [Configuring an Agency for ModelArts](#).

4. Locate the target BMS in the list and choose **...** > **Unsubscribe**.
5. Confirm the resources to be unsubscribed from and select the reason for unsubscription.



Figure 5-14 Unsubscription page



6. Confirm the information and select **I understand a handling fee will be charged for this unsubscription and After being unsubscribed from, the resources not in the recycle bin will be deleted immediately and cannot be restored. I've backed up data or no longer need the data.**
7. Click **Confirm** and confirm the resources to be unsubscribed from.
8. Click **Unsubscribe** again to unsubscribe from the yearly/monthly resources.

### Unsubscribing from a single instance in the Billing Center

1. Log in to the ModelArts console.
2. In the navigation pane on the left, choose **AI Dedicated Resource Pools > Elastic Servers**.
3. Select **View all** from the drop-down list to view all server instances.

Figure 5-15 View all

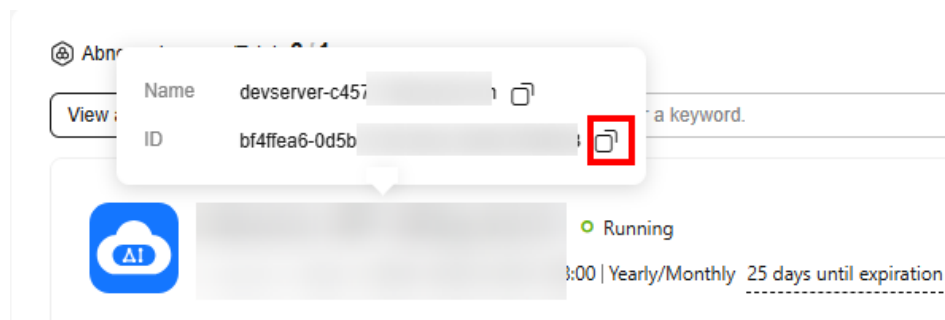


#### NOTE

If an agency needs to be configured, contact your account administrator. For details, see [Configuring an Agency for ModelArts](#).

4. Copy the ID of the instance to be unsubscribed from.

**Figure 5-16** Copying the instance ID

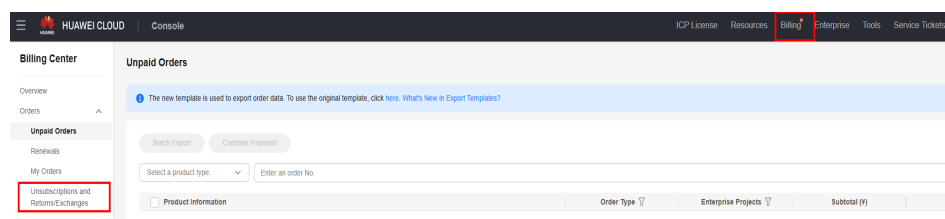


**NOTE**

The resource ID bound to the DevServer purchase order is the DevServer ID, which is different from the BMS/ECS ID encapsulated in the DevServer product. To unsubscribe from DevServer, log in to the ModelArts console and choose **AI Dedicated Resource Pools > Elastic Servers** on the left.

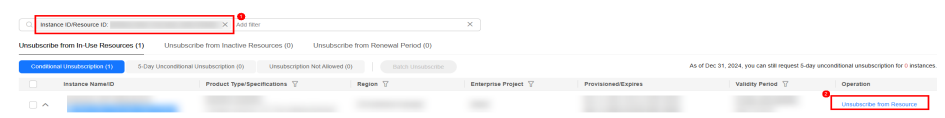
5. Choose **Billing** from the top menu bar. In the navigation pane on the left, choose **Orders > Unsubscriptions and Returns/Exchanges**.

**Figure 5-17** Unsubscriptions and Returns/Exchanges



6. Search for the instance ID, confirm the information, and click **Unsubscribe from Resource** in the **Operation** column.

**Figure 5-18** Searching for the instance ID



7. Confirm the resources to be unsubscribed from and select the reason for unsubscription.
  8. Confirm the information and select **I understand a handling fee will be charged for this unsubscription and After being unsubscribed from, the resources not in the recycle bin will be deleted immediately and cannot be restored. I've backed up data or no longer need the data.**
  9. Click **Confirm** and confirm the resources to be unsubscribed from.
  10. Click **Unsubscribe** again to unsubscribe from the yearly/monthly resources.
- **Unsubscribing from multiple instances in the Billing Center**
1. Log in to the ModelArts console.

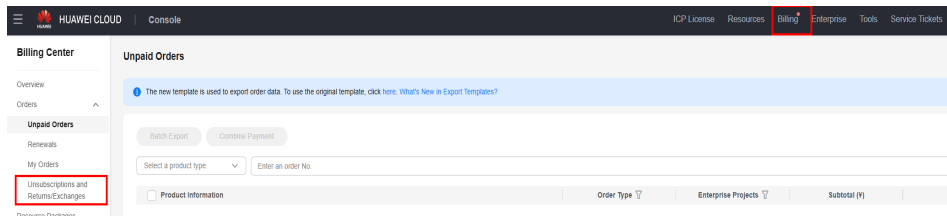
2. In the navigation pane on the left, choose **Dedicated Resource Pools > Elastic Server**.
3. Record the IDs of instances to be unsubscribed from.

 **NOTE**

If an agency needs to be configured, contact your account administrator. For details, see [Configuring an Agency for ModelArts](#).

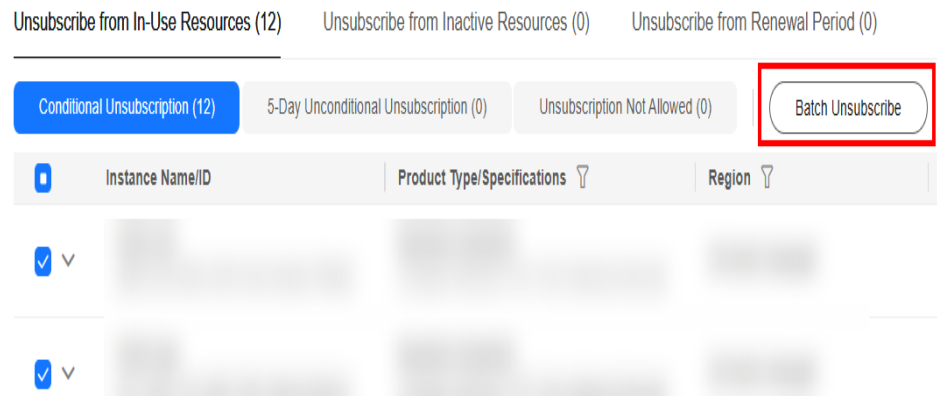
4. Choose **Billing** from the top menu bar. In the navigation pane on the left, choose **Orders > Unsubscriptions and Returns/Exchanges**.

**Figure 5-19** Unsubscriptions and Returns/Exchanges



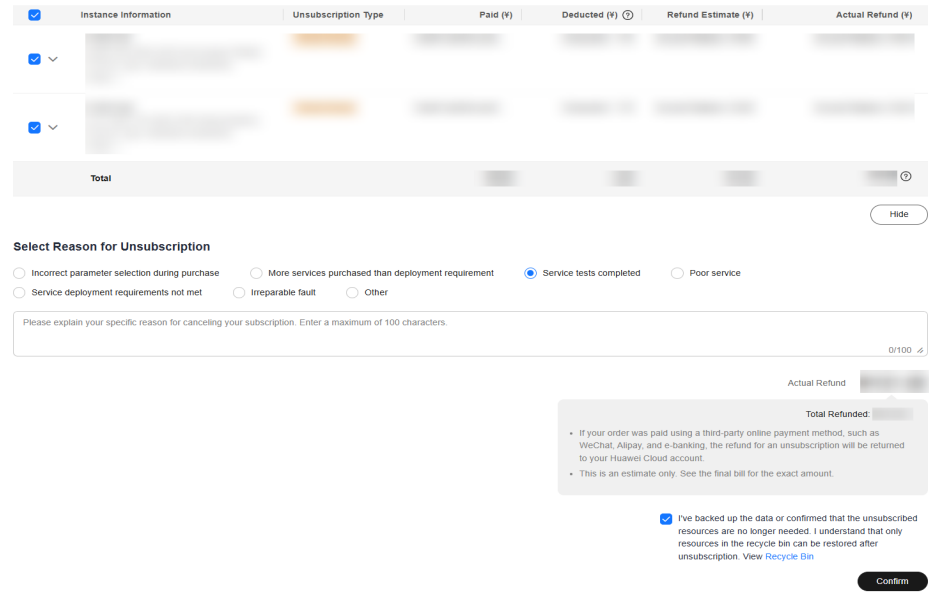
5. Select the target instances and click **Batch Unsubscribe**.

**Figure 5-20** Batch Unsubscribe



6. Confirm the resources to be unsubscribed from and select the reason for unsubscription.

Figure 5-21 Unsubscription page



7. Confirm the information and select **I understand a handling fee will be charged for this unsubscription and After being unsubscribed from, the resources not in the recycle bin will be deleted immediately and cannot be restored. I've backed up data or no longer need the data.**
8. Click **Confirm** and confirm the resources to be unsubscribed from.
9. Click **Unsubscribe** again to unsubscribe from the yearly/monthly resources.