07 Lite Server User Guide 7.5.0

Issue 01

Date 2026-06-01

HUAWEI

HUAWEI CLOUD COMPUTING TECHNOLOGIES CO., LTD.

Trademarks and Permissions


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 End-to-End Process of Lite Servers....1
2 Lite Server High-Risk Operations....4
3 Mappings Between Lite Compute Nodes and OS Images....9
4 Enabling Lite Servers (New UI)....20
5 Configuring Lite Server Resources....33
5.1 Resource Configuration Workflow....33
5.2 Configuring the Lite Server Network....34
5.3 Configuring the Lite Server Storage....37
5.4 (Optional) Configuring the Software Environment for the Lite Servers....41
5.4.1 Configuring the Resource Software Environment for NPU-based Lite Servers....41
5.4.2 Creating the OS of a Lite Server....59
5.4.3 Configuring the NPU Driver Firmware Consistency and UDP Port Hash....61
6 Using Lite Server Resources....72
6.1 NPU Training and Inference Guide for LLM and AIGC Models....72
6.2 Adapting GPT-2 to GPU-based Training and Inference Based on Lite Servers....72
7 Managing Lite Server Resources....81
7.1 Viewing Details About a Lite Server....81
7.2 Starting or Stopping a Lite Server....83
7.3 Synchronizing the Status of a Lite Server....85
7.4 Changing or Resetting the OS of a Lite Server....86
7.5 Managing Lite Server Hot Standby Nodes....90
7.6 Changing the Name of a Lite Server....92
7.7 Authorizing Repair of a Lite Server....94
7.8 Restarting a Lite Server....98
8 Managing Lite Server Plugins....100
8.1 Installing the AI Plugin for a Lite Server....100
8.2 Upgrading the Ascend Driver and Firmware Version on a Lite Server....103
8.3 Upgrading the GPU Driver on a Lite Server....106
8.4 Installing or Upgrading the Cloud Eye Agent Plugin on a Lite Server....109
8.5 Diagnosing Faults on Lite Servers....111

8.6 Fixing Lite Server Vulnerabilities....115
8.7 Performing a One-Click Stress Test on a Lite Server....117
8.8 Configuring the Parameter Plane Network for Lite Servers....119
8.9 Performing Health Check on Lite Servers....122
9 Managing Lite Server Supernodes....125
9.1 Scaling a Lite Server Supernode....125
9.2 Expanding the System Disk Capacity of a Lite Server Supernode....128
9.3 Performing Periodic Stress Tests on Lite Server Supernodes....129
9.4 Enabling HCCL Communication Operator-Level Re-execution for Supernodes....154
10 Collecting Logs of Lite Servers....160
10.1 Collecting and Uploading NPU Logs....160
10.2 Collecting and Uploading GPU Logs....166
11 Monitoring Lite Servers....171
11.1 Using Cloud Eye to Monitor NPU Resources of Lite Servers....171
11.2 Using Cloud Eye to Monitor NPU Events of Lite Servers....209
11.3 Using Cloud Eye to Monitor Lite Servers and Generate Event Alarms....227
11.4 Using Cloud Eye to Automatically Detect Server Environments and Provide Event Notifications....230
11.5 Using DCGM to Monitor GPU Resources of Lite Servers....232
12 Managing CloudPond NPU Resources Using Lite Servers....236
13 Using CTS to Audit Operations on Lite Servers....242
14 Unsubscribing from a Lite Server....246

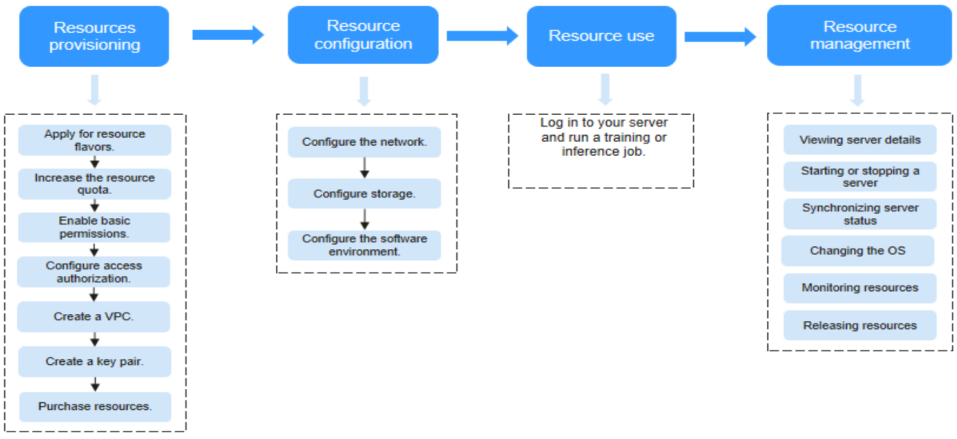
1 End-to-End Process of Lite Servers

For algorithm engineers, daily training and inference tasks require a flexible, powerful development environment on the cloud. However, existing cloud services often limit user customization, making it difficult to meet specific software installation and configuration requirements.

ModelArts Lite Servers provide various resources, allowing you to install and deploy third-party software such as AI frameworks and applications as user **root**, as well as building a dedicated cloud server environment. To create a Lite Server, you only need to configure the specifications, image, network, and key pairs. This provides you on-cloud physical resources for training and inference.

This section describes how to get started with Lite Servers effectively. Below is the workflow.

Figure 1-1 Workflow



1. Purchasing resources

   Use Lite Servers only after purchase.

   a. Contact your account manager to confirm the resource plan for Lite Servers. As some specifications are restricted, you will need to apply for access to your required specifications.

   b. The resources required by Lite Servers may exceed the default quotas provided by cloud services (such as ECS, EIP, and SFS). In this case,

submit a service ticket to increase resource quotas. For details about the default quotas, see **Viewing Quotas**.

c. Grant the required basic permissions to the IAM user.

d. Create an agency for ModelArts to enable it to access its dependency services.

e. Enable Lite Servers 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 Servers allow you to start, stop, and switch OSs. You can manage them on the ModelArts console.

Table 1-1 Terms

Term	Description
Common node	A single physical or virtual host providing basic independent compute, storage, and network resources. These include both Bare Metal Servers (BMSs) and Elastic Cloud Servers (ECSs).
ECS	An ECS is a fundamental computing unit consisting of CPUs, memory, an OS, and Elastic Volume Service (EVS) disks. Once an ECS is created, you can use it on the cloud just like a local PC or physical server. Lite Servers support various server types, including ECS. For details about ECSs, see **Elastic Cloud Server**.
BMS	BMS is a computing service that combines the elasticity of an ECS with the high performance of a physical machine. It provides dedicated physical servers on the cloud for core databases, critical applications, high-performance computing (HPC), and big data, ensuring superior performance and data security. Lite Servers support various server types, including BMS. For more information, see **Bare Metal Server**.
Supernode	A supernode is a high-performance computing resource provided by Huawei Cloud, primarily used for large AI model training and inference. A supernode consists of multiple nodes where internal NPUs are interconnected via a specific networking architecture to form a hyper-plane network, delivering ultra-fast transmission rates. Supernode servers support Snt9b23 resources and are currently available only in the CN South-Guiyang1, CN North3, and CN East2 regions.

Term	Description
Key pair	Lite Servers support SSH key pair authentication for login. You can access Lite Servers without a password, preventing credential leakage caused by interception or brute-force attacks, thereby enhancing the security of Lite Servers. **NOTE** To ensure server security, if a private key is not hosted on the platform, it can only be downloaded once. Store it securely.
Virtual Private Cloud (VPC)	A VPC provides an isolated, configurable, and manageable virtual network environment for Lite Servers. It enhances resource security and simplifies network deployment. Within a VPC, you can define security groups, VPNs, IP address ranges, and bandwidth. VPCs allow for easy management and configuration of internal networks for secure and rapid changes. Additionally, you can customize inbound and outbound rules for security groups to strengthen the protection of your Lite Servers. For details, see **Virtual Private Cloud**.

2 Lite Server High-Risk Operations

High-risk operations involved in the daily O&M of ModelArts Lite Servers must be performed in strict accordance with the operation guide. Failure to do so may affect normal service continuity.

Risk levels of high-risk operations:

- 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 2-1 High-risk operations

Object	Operation	Risk	Severity	Solution
OS	Upgrade or modify the OS kernel.	The driver and kernel versions may not be compatible. As a result, the OS cannot be started or basic functions are unavailable. High-risk commands, such as **apt-get upgrade** (upgrading all software in the system, including the kernel), are involved. Run the **uname -a** command to view the current kernel.	High	To perform upgrade or modification, **contact Huawei Cloud 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.	Low	Attach an EVS or SFS disk for capacity expansion after you **switch or reset the OS**.
	When the cloud server service is running properly, the user deletes the NIC route in the system or performs network destruction operations, such as running **ifconfig down** and **ifconfig up**, on the NIC.	The network service will be restarted and DHCP will be triggered to obtain the IP address and route again. As a result, the NIC route may be lost and the node may be unavailable.	High	Reset the OS. Ensure that your data has been backed up.
	Modify kernel parameters such as **net.ipv4.ip_forward**.	Cloud server route forwarding may be affected, causing network disconnection.	Medium	Set **net.ipv4.ip_forward** to **1**.
	Enable the system firewall.	The performance of HCCL, NCCL, and multi-node multi-PU training tasks may be affected.	Low	Disable the firewall.
	Change the time zone.	The node time changes, which will affect services.	Medium	Restore the time zone.
Driver and firmware	Upgrade the NPU driver or firmware.	The driver and firmware may not match, causing unavailable servers and affecting services.	Medium	**Reset the OS**. Ensure that your data has been backed up.
	Change the GPU driver.	The driver and firmware may not match, causing unavailable servers and affecting services.	Medium	**Reset the OS**. Ensure that your data has been backed up.

Object	Operation	Risk	Severity	Solution
	Change the SDI PU driver.	The NIC may be unavailable, causing unavailable servers and affecting services.	Medium	**Reset the OS**. Ensure that your data has been backed up.
Network	Change the NIC MAC address or IP address.	If misoperations are performed, the VM communication and services are interrupted, and other services are affected.	High	Roll back the modification. If the rollback fails, **reset the OS**. Ensure that your data has been backed up.
	Add, delete, or edit iptables rules, or restart the iptables service.	Service access requests are rejected.	High	Roll back the modification. If the rollback fails, **reset the OS**. Ensure that your data has been backed up.
Pre-installed software	Upgrade, downgrade, or uninstall pre-installed software such as Python 3.	This may cause anomalies in network configuration software, resulting in NIC configuration failures and rendering the node unavailable.	High	Roll back the modification. If the rollback fails, **reset the OS**. Ensure that your data has been backed up.
Directory/File	Modify key system directories and files of **root** or **opt**, such as **/etc/hccn.conf** and **/etc/netplan/roce.yaml**.	The system functions may be affected, and the cloud server may be unavailable.	High	Roll back the modification. If the rollback fails, **reset the OS**. Ensure that your data has been backed up.
	Modify the permissions of directories and files.	The service may be abnormal.	High	Roll back the modification.

Object	Operation	Risk	Severity	Solution
Server	Do not perform non-query operations on the server, such as stopping or starting the server, when the server instance is being provisioned, initialized, or when disks are being added, deleted, or the instance is being deleted.	Operations on the cloud server may fail.	Medium	**Reset the OS**. Ensure that your data has been backed up.
	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.	Low	Attach an EVS or SFS disk for capacity expansion. For details, see **Configuring the Lite Server Storage**.
Process	Run the **service network restart** command. Stop key system processes, such as sshd ces-agent.	Services may fail to be provisioned, the remote access to the cloud server may fail. Moreover, data may fail to be collected, affecting the reporting of monitoring indicators.	High	Restart the closed service.
Data disk	Modify the data disk attaching mode and mount point.	Services that are being used may become abnormal.	Low	Ensure that the data disk is not used by any service.

Object	Operation	Risk	Severity	Solution
Security group	Modify the port communication protocol. Allow high-risk ports such as port 22. IP address whitelist not configured.	The network may be attacked, affecting services of the server.	Medium	Restore the original content.

3 Mappings Between Lite Compute Nodes and OS Images

ModelArts Lite Servers support multiple OS images. You can view the currently supported images and their details, which help you select the appropriate option in **Table 4-4**.

Images Supported by NPU Snt9b23 Supernode Servers

Image name: HCE2.0-Arm-64bit-for-Snt9b2x-with-25.5.1-7.8.0.6.201-CANN8.5.2-v2

Table 3-1 Image details

Software Type	Version Details
OS	HCE 2.0
Kernel version	5.10.0-182.0.0.95.r2220_157.hce2.aarch64
Architecture	aarch64
Firmware version	7.8.0.6.201
npu-driver	25.5.1
Ascend-cann-toolkit	8.5.2
cann-ops	8.5.2
Ascend-mindx-toolbox	7.3.0
Docker	27.2.0
Ascend-docker-runtime	7.3.0
Mpich	4.3.0
MCU	25.55.23
Cloud Eye Agent	2.8.2.2

Image name: HCE2.0-Arm-64bit-for-Snt9b23-with-25.2.1-7.7.0.9.220-CANN8.1.RC2-v4

Table 3-2 Image details

Software Type	Version Details
OS	HCE 2.0
Kernel version	5.10.0-182.0.0.95.r2220_157.hce2.aarch64
Architecture	aarch64
Firmware version	7.7.0.9.220
npu-driver	25.2.1
Ascend-cann-toolkit	8.1.RC2
cann-kernels	8.1.RC2
Ascend-mindx-toolbox	7.0.RC1
Docker	27.2.0
Ascend-docker-runtime	7.0.RC1
Mpich	4.3.0
MCU	25.52.29
Cloud Eye Agent	2.8.2.2

Images Supported by NPU 560T Single-Node Servers

Image name: HCE2.0-Arm-64bit-for-Snt9b2x-with-25.5.1-7.8.0.6.201-CANN8.5.2-v2

Table 3-3 Image details

Software Type	Version Details
OS	HCE 2.0
Kernel version	5.10.0-182.0.0.95.r2220_157.hce2.aarch64
Architecture	aarch64
Firmware version	7.8.0.6.201
npu-driver	25.5.1
Ascend-cann-toolkit	8.5.2
cann-ops	8.5.2
Ascend-mindx-toolbox	7.3.0

Software Type	Version Details
Docker	27.2.0
Ascend-docker-runtime	7.3.0
Mpich	4.3.0
MCU	25.55.23
Cloud Eye Agent	2.8.2.2

Images Supported by NPU Snt9b BMSs

- Image name: Ubuntu22.04-Arm-64bit-for-Snt9A2-BareMetal-with-25.5.1-7.8.0.6.201-CANN8.5.2-v1

  Table 3-4 Image details

Software Type	Version Details
OS	Ubuntu 22.04
Kernel version	5.15.0-91-generic
Architecture	aarch64
Firmware version	7.8.0.6.201
npu-driver	25.5.1
Ascend-cann-toolkit	8.5.2
cann-ops	8.5.2
Ascend-mindx-toolbox	7.3.0
Docker	26.0.0
Ascend-docker-runtime	7.3.0
Mpich	3.2.1
MCU	25.55.28
Cloud Eye Agent	2.8.2.1

- Image name: HCE2.0-Arm-64bit-for-Snt9A2-BareMetal-with-25.5.1-7.8.0.6.201-CANN8.5.2-v1

Table 3-5 Image details

Software Type	Version Details
OS	HCE 2.0
Kernel version	5.10.0-136.12.0.86.r1526_92.hce2.aarch64
Architecture	aarch64
Firmware version	7.8.0.6.201
npu-driver	25.5.1
Ascend-cann-toolkit	8.2.1
cann-kernels	8.2.1
Ascend-mindx-toolbox	7.3.0
Docker	27.2.0
Ascend-docker-runtime	7.3.0
Mpich	4.3.0
MCU	25.55.28
Cloud Eye Agent	2.8.2.1

Images Supported by NPU Snt9b ECSs

- Image name: HCE2.0-Arm-64bit-for-Snt9A2-ECS-BareMetal-with-25.5.1-7.8.0.6.201-CANN8.5.2-v1

Table 3-6 Image details

Software Type	Version Details
OS	HCE 2.0
Kernel version	5.10.0-136.12.0.86.r1526_92.hce2.aarch64
Architecture	aarch64
Firmware version	7.8.0.6.201
npu-driver	25.5.1
Ascend-cann-toolkit	8.2.1
cann-kernels	8.2.1
Ascend-mindx-toolbox	7.3.0

Software Type	Version Details
Docker	27.2.0
Ascend-docker-runtime	7.3.0
Mpich	4.3.0
MCU	25.55.28
Cloud Eye Agent	2.8.2.1

- Image name: Ubuntu22.04-Arm-64bit-for-Snt9A2-ECS-BareMetal-with-25.5.1-7.8.0.6.201-CANN8.5.2-v1

  Table 3-7 Image details

Software Type	Version Details
OS	Ubuntu 22.04
Kernel version	5.15.0-91-generic
Architecture	aarch64
Firmware version	7.8.0.6.201
npu-driver	25.5.1
Ascend-cann-toolkit	8.5.2
cann-ops	8.5.2
Ascend-mindx-toolbox	7.3.0
Docker	26.0.0
Ascend-docker-runtime	7.3.0
Mpich	3.2.1
MCU	25.55.28
Cloud Eye Agent	2.8.2.1

Images Supported by NPU Snt3PD ECSs

Image name: Huawei-Cloud-EulerOS-2.0-64bit-for-kAi2p-with-HDK-24.1.0.1-and-CANN-8.0.1

Software Type	Version Details
OS	Huawei Cloud EulerOS 2.0
Kernel version	5.10.0-182.0.0.95.r2762_220.hce2.aarch64
Architecture	aarch64
npu-driver	24.1.0.1
Ascend-cann-toolkit	8.0.1
cann-kernels	8.0.1
Ascend-mindx-toolbox	6.0.0
Docker	18.09.0
Ascend-docker-runtime	v6.0.0
Mpich	3.2.1
MCU	24.5.8

Images Supported by GP Ant8 BMSs

Image name: Ubuntu-22.04-x86-for-Ant1-Ant8-BareMetal-with-RoCE-and-NVIDIA-550.90.07-CUDA-12.4-v2

Table 3-8 Image details

Software Type	Version Details
OS	Ubuntu 22.04
Kernel version	5.15.0-25-generic
Architecture	x86
Driver version	550.90.07
cuda	12.4
nv-fabricmanager	550.90.07-1
nv-container-toolkit	1.17.5-1
libnccl2	2.26.2-1+cuda12.4
libnccl-dev	2.26.2-1+cuda12.4
Docker	20.10.23
Mpich	4.1.5a1

- Image name: HCE2.0-x86-for-Ant8-with-RoCE-and-NVIDIA-535-CUDA-12.2-v1

Table 3-9 Image details

Software Type	Version Details
OS	Huawei Cloud EulerOS 2.0 (x86_64)
Kernel version	5.10.0-182.0.0.95.r3008_246.hce2.x86_64
Architecture	x86
Driver version	535.183.06
CUDA	12.2
nv-fabricmanager	535.183.06
nv-container-toolkit	1.18.0-1
libnccl2	libnccl-2.27.7-1+cuda12.2
libnccl-dev	libnccl-2.27.7-1+cuda12.2
Docker	27.2.0

Images Supported by GP Vnt1 BMSs

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-22.04-for-BareMetal-Vnt1-p3-with-NV-535-CUDA-12.2 (only for CN North-Beijing1, CN North-Beijing4, and CN South-Guangzhou)

Table 3-10 Image details

Software Type	Version Details
OS	Ubuntu 22.04
Kernel version	5.15.0-25-generic
Architecture	x86
Driver version	535.54.03
CUDA	12.2
container-toolkit	1.16.1-1
libnccl2	2.21.5-1+cuda12.2
libnccl-dev	2.21.5-1+cuda12.2
Docker	24.0.5

Software Type	Version Details
Cloud Eye Agent	2.7.2.1

- Image name: Ubuntu-22.04-for-BareMetal-Vnt1-p6-with-NVIDIA-545-CUDA-12.3 (only for CN East-Shanghai1)

  Table 3-11 Image details

Software Type	Version Details
OS	Ubuntu 22.04
Kernel version	5.15.0-25-generic
Architecture	x86
Driver version	545.23.08
CUDA	12.3
nv-container-toolkit	1.17.4.1
libnccl-dev	2.20.3-1+cuda12.3
libnccl2	2.20.3-1+cuda12.3
Docker	28.0.0
Cloud Eye Agent	2.8.2.2

Images Supported by GP Ant1 BMSs

Image name: Ubuntu-22.04-x86-for-Ant1-Ant8-BareMetal-with-RoCE-and-NVIDIA-550.90.07-CUDA-12.4-v2

Table 3-12 Image details

Software Type	Version Details
OS	Ubuntu 22.04
Kernel version	5.15.0-25-generic
Architecture	x86
Driver version	550.90.07
cuda	12.4
nv-fabricmanager	550.90.07-1
nv-container-toolkit	1.17.5-1
libnccl2	2.26.2-1+cuda12.4

Software Type	Version Details
libnccl-dev	2.26.2-1+cuda12.4
Docker	20.10.23
Mpich	4.1.5a1

Images Supported by GP Hnt02 ECSs

- Image name: HCE2.0-x86-for-H20-NV-535-CUDA-12.2-v2 (only for CN North-Ulanqab 1 and CN East 2)

Software Type	Version Details
OS	HCE 2.0
Kernel version	5.10.0-182.0.0.95.r1941_123.hce2.x86_64
Architecture	x86_64
Driver version	535.183.01
CUDA	12.2
nv-fabricmanager	535.183.01
libnccl	2.18.5-1+cuda12.2
libnccl-dev	2.18.5-1+cuda12.2
Cloud Eye Agent	2.7.2.1

- Image name: Ubuntu22.04_x86_for_h20_Driver-535-and-CUDA-12.2-v2 (only for CN North-Ulanqab 1 and CN East 2)

Software Type	Version Details
OS	Ubuntu 20.04 server 64-bit
Kernel version	5.15.0-107-generic
Architecture	x86_64
Driver version	535.183.01
CUDA	12.2
nv-fabricmanager	535.183.01
libnccl2	2.18.5-1+cuda12.2
libnccl-dev	2.18.5-1+cuda12.2
Docker	28.0.1

Images Supported by GP Lnt002 ECSs

Image name: Ubuntu-22.04-server-64bit-with-Tesla-Driver-535.183.01-and-CUDA-12.2 (only for CN North-Beijing4, CN East-Shanghai1, ME-Riyadh, and AP-Jakarta)

Software Type	Version Details
OS	Ubuntu 20.04 server 64-bit
Kernel version	5.15.0-92-generic
Architecture	x86
Driver version	535.183.01
CUDA	12.2
nv-container-toolkit	1.17.1
Docker	27.3.1
Cloud Eye Agent	2.7.3.t2

Image name: HCE2.0-x86-for-L2-NVIDIA-535-CUDA-12.2 (only for CN North-Beijing4, CN East-Shanghai1, and AP-Singapore)

Software Type	Version Details
OS	Huawei Cloud EulerOS 2.0(x86_64)
Kernel version	5.10.0-60.18.0.50.r865_35
Architecture	x86
Driver version	535.183.01
CUDA	12.2
nv-container-toolkit	1.13.5-1
Docker	18.09.0
Cloud Eye Agent	2.6.7.1

Images Supported by GP Ant03 ECSs

Image name: Ubuntu 22.04 server 64bit with Tesla Driver 470.182.03 and CUDA 11.4 (only for CN South-Guangzhou and CN East-Shanghai1)

Software Type	Version Details
OS	Ubuntu 22.04 server 64bit

Software Type	Version Details
Kernel version	5.15.0-60-generic
Architecture	x86
Driver version	470.182.03
cuda	11.4
nv-fabricmanager	470.182.03-1

4 Enabling Lite Servers (New UI)

Description

This section describes how to purchase Lite Servers on the ModelArts console and outlines the necessary preparations.

Before purchasing a Lite Server, ensure that you have completed the required preparations, including requesting a quota increase, configuring basic permissions, and setting up ModelArts agency authorization. When purchasing resources, you will create an instance and pay for the order. Provisioning typically takes 20 to 60 minutes. Once the resources are created, you can configure an EIP for access and begin your AI development tasks.

Constraints

Lite Server supernodes only support the yearly/monthly billing mode.

All resource specifications for Lite Server common nodes (ECS or BMS) support the yearly/monthly billing mode.

Enabling Resources

Figure 4-1 Lite Server resource provisioning process

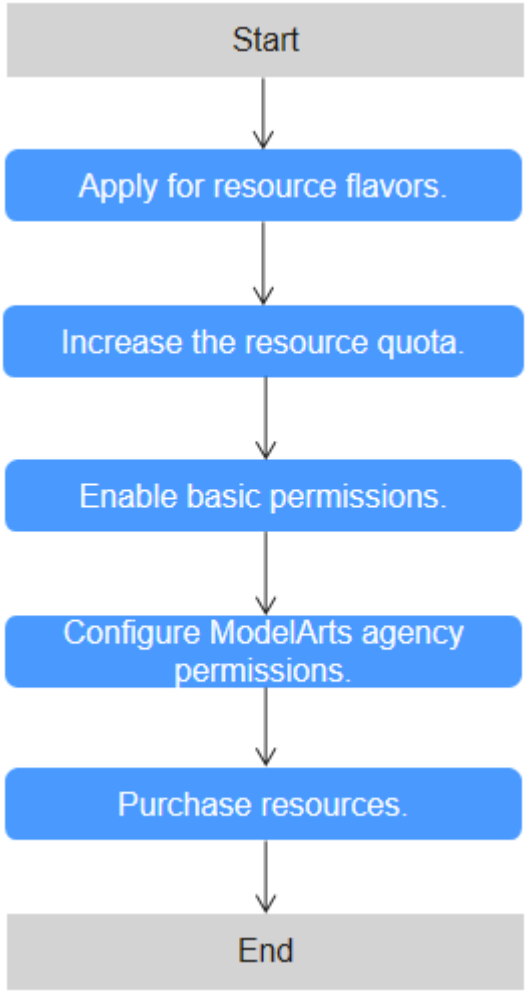


Table 4-1 Lite Server resource provisioning process

Phase	Task
Making preparations	1. Apply for resource specifications.
	2. Increase your resource quota.
	3. Enable basic permissions.
	4. Configure an agency authorization for ModelArts.
Purchasing Lite Server resources	5. Purchase Lite Server resources on the ModelArts console.

Step 1: Applying for Resource Specifications

Contact your account manager to confirm the resource plan and request access to the required specifications. If you do not have an account manager, you can submit a service ticket.

Step 2: Increasing Your Resource Quota

The resources required by Lite Servers may exceed the default resources (such as ECS, EIP, SFS, memory, and CPUs) provided by cloud services. In this case, you need to increase the resource quota.

1. Log in to the **Huawei Cloud console**.
2. Hover over **Resources** from the top menu bar and choose **My Quotas**.
3. Click **Increase Quota** in the upper right corner, fill in the materials, and submit a service ticket.

   NOTE

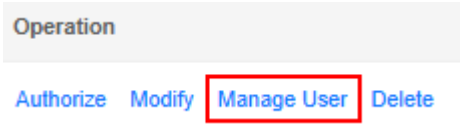
   The resource quota must exceed the amount you intend to provision and must be increased before you start the purchase; otherwise, resource provisioning will fail.

Step 3: Enabling Basic Permissions

Log in to the administrator account and grant the target IAM user the required basic permissions, including ModelArts FullAccess, BMS FullAccess, ECS FullAccess, VPC FullAccess, VPC Administrator, and VPCEndpoint Administrator. This allows the IAM user to use these cloud services.

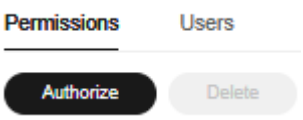
1. Log in to the **IAM console**.
2. In the navigation pane on the left, choose **User Groups** and click **Create User Group** in the upper right corner.
3. Enter a group name and click **OK**.
4. On the **User Groups** page, locate the target user group and click **Manage User** in the **Operation** column, and add the user to the user group.

   Figure 4-2 User group management

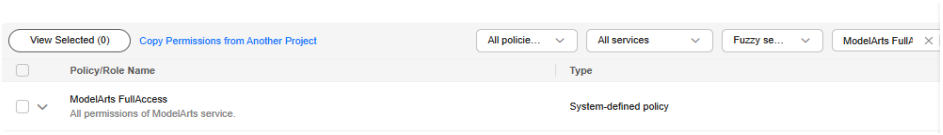
5. Click the user group name to access its details page.
6. In the **Permissions Assigned** tab, click **Assign**.

   Figure 4-3 Assigning permissions

7. Search for **ModelArts FullAccess** in the search box and select it.

   Figure 4-4 ModelArts FullAccess

   Use the same method to select **BMS FullAccess**, **ECS FullAccess**, **VPC FullAccess**, **VPC Administrator**, **VPCEndpoint Administrator**, and **CloudMatrixFullAccessPolicy (supernode)**. **Server Administrator** and **DNS Administrator** are dependency policies and are automatically selected.
8. Click **Next** and set **Scope** to **All resources**.
9. Click **OK**.

Step 4: Creating an Agency Authorization on ModelArts

During task execution, ModelArts Lite Servers must access other services. This includes pulling images from SWR when using containers. In such cases, ModelArts accesses other cloud services on behalf of you. To ensure security, ModelArts requires your authorization before accessing any cloud services, which is the agency process. Once authorized, you can run AI computing tasks on ModelArts.

- Creating an agency

  Create an agency on ModelArts to authorize access to other cloud services. To do so, log in to the ModelArts console. In the navigation pane on the left, choose **Permission Management** under **System Management**. On the displayed page, click **Add Authorization**.
- Updating an agency

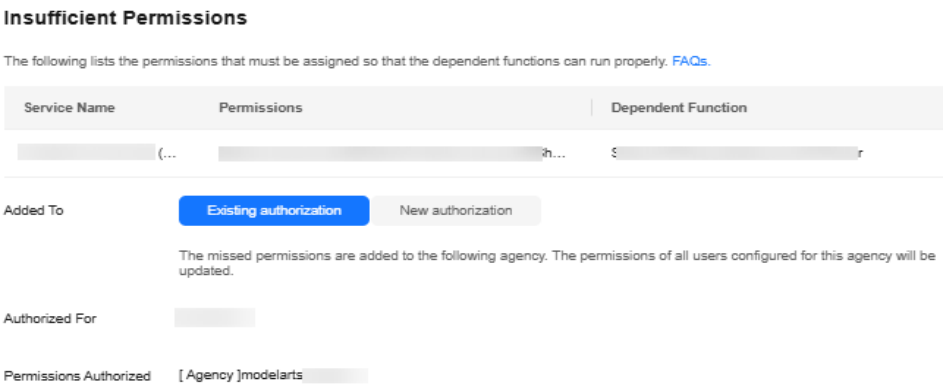
  Update the permissions for your existing ModelArts agency.

  a. Log in to the **ModelArts console**. In the navigation pane on the left, choose **Resource Management** > **Lite Servers**. Check whether a message is displayed indicating that the authorization is missing.

     - New console: In the navigation pane, choose **Resource Management** > **Lite Compute Resources** > **Lite Servers**.
     - Old console: In the navigation pane, choose **Resource Management** > **Lite Servers**.

  b. Click **View missing permissions** to update the agency if needed. Set **Added To** to **Existing authorization** and click **OK**.

Figure 4-5 Adding an authorization



Step 5: Purchasing Lite Server Resources

When you purchase Lite Servers, resources are created.

1. Log in to the **ModelArts console**. In the navigation pane, choose **Resource Management** > **Lite Servers**.
   - New console: In the navigation pane, choose **Resource Management** > **Lite Compute Resources** > **Lite Servers**.
   - Old console: In the navigation pane, choose **Resource Management** > **Lite Servers**.
2. Click **Buy Lite Server** in the upper right corner. Configure the parameters on the displayed page.

   NOTE

   The purchase page offers two versions: New Version and Old Version. The parameter order shown below matches the layout on the New Version's purchase page. While the Old Version's purchase page displays parameters in a different order, their descriptions remain the same.

Figure 4-6 Basic configurations

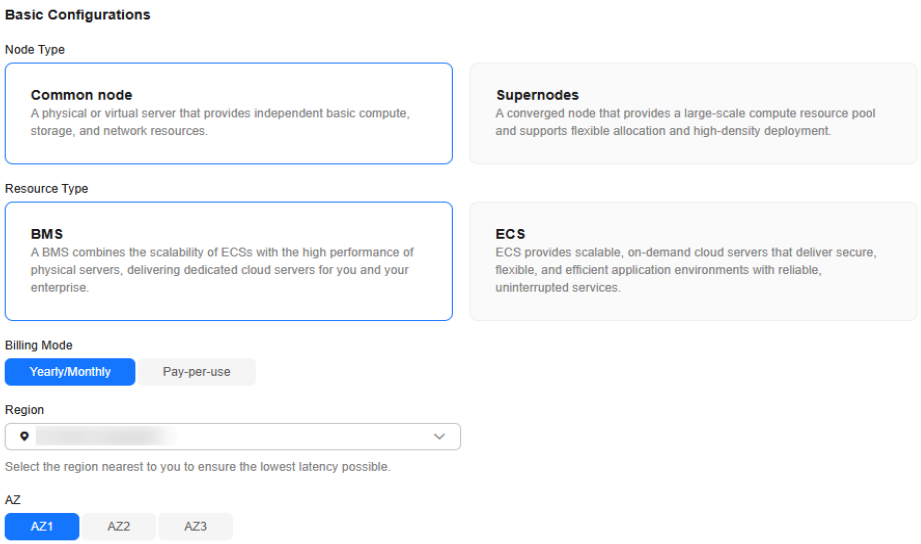


Table 4-2 Basic configurations

Parameter	Description
Node Type	• Common node: A single physical or virtual host providing basic independent compute, storage, and network resources. These include both Bare Metal Servers (BMSs) and Elastic Cloud Servers (ECSs). • Supernode: A converged node that provides a large-scale compute resource pool and supports flexible allocation and high-density deployment Supernodes are specifically designed for large-scale model training and inference tasks. These servers are typically equipped with multiple accelerators (such as Ascend NPUs) to provide robust compute for high-load requirements. Supernode resources use Snt9b23 specifications and are currently available only in the CN Southwest-Guiyang1, CN North3, and CN East2 regions.
Resource Type	This parameter appears when **Node Type** is set to **Common node**. Select **BMS** or **ECS** as required. • **BMS**: A BMS combines the scalability of ECSs with the high performance of physical servers, delivering dedicated cloud servers for you and your enterprise. • **ECS**: ECS provides secure, scalable, on–demand compute resources, enabling you to flexibly deploy applications and workloads.
Billing Mode	• Yearly/Monthly is a prepaid billing mode in which your subscription is billed based on the required duration. This mode is more cost–effective when the usage duration is predictable. • Lite Server supernodes only support the yearly/monthly billing mode. All resource specifications for Lite Server common nodes (ECS or BMS) support the yearly/monthly billing mode.
Region	Resources in different regions cannot communicate with each other over an internal network. For low network latency and quick resource access, select the nearest region to your services.

Parameter	Description
AZ	A physically isolated zone where resources have their own independent power supply and internal networks. When deploying resources, consider your application requirements for disaster recovery (DR) and network latency. • To get higher DR capability, deploy resources in different AZs in the same region. • For lower network latency, deploy resources in the same AZ. When **Node Type** is set to **Supernodes** or **Common node** > **ECS**, **Random AZ** is supported, which displays available resources across all AZs in the current region. **Common node** > **BMS** does not support **Random AZ**. For details about how to provision edge station resources, see **Managing CloudPond NPU Resources Using Lite Servers**.

Table 4-3 Specification configurations

Parameter	Description
CPU Architecture	CPU architecture of the resource type, which can be **x86** or **Arm**. • **x86**: Select this if GPU resources are used. • **Arm**: Select this if NPU resources are used. Select a CPU architecture and then select instance specifications as required. The flavors vary by region. The actual flavors are displayed on the console. Sold-out resources are displayed in gray and cannot be purchased. When you select Snt9b23 supernode specifications, the hyperplane networks are only available for single-instance VXpods. To ensure all compute resources can use the hyperplane, choose a single-instance VXpod with a larger node size. **NOTE** If no specifications are available, contact **Huawei technical support**.

Table 4-4 OS configurations

Parameter	Description
Image	OS image of the Lite Server. • **Public image** Public images are available for all users. All users can read the image by image ID. ModelArts allows you to perform development and training directly without additional configuration as it provides multiple public OS images, supports multiple OSs, and has built–in AI drivers and software in the images. For details about the supported public images, see **Mappings Between Lite Compute Nodes and OS Images**. The server needs compatibly drivers to run normally and an OS kernel change may cause some drivers to be incompatible, so you are advised not to change or upgrade the drivers or kernels. For details about how to switch or reset the OS image, see **Changing or Resetting the OS of a Lite Server**. • **Private image** Only the image creator can use the image. Use a private image to set up the Lite Server OS quickly, avoiding repetitive server configurations. Private images must be created in IMS. For details, see **Creating a Private Image**.

Table 4-5 Storage configurations

Parameter	Description
Storage	The storage configuration parameters apply to each common node. Actual storage configuration = Storage configuration of a single node x Number of purchased nodes.
Node System Disk Type	This parameter is only displayed when you select an instance flavor that supports attaching. The node system disk stores the OS of a server, and is automatically created and initialized upon Lite Server creation. Select a node system disk type and set the disk size. The system disk size ranges from 100 GiB to 1,024 GiB. You can also expand the system disk capacity after a Lite Server is created. Currently, only the system disk capacity can be expanded. For details, see **Expanding the System Disk Capacity of a Lite Server Supernode**. The system disk is automatically attached to each compute node.

Parameter	Description
(Optional) Node Data Disk Type	Click **Add Data Disk** to attach an EVS data disk to the Lite Server. Currently, local disks cannot be attached. You can select **Node Data Disk Type** and set **Size** and **Quantity**. The data disk size ranges from 100 GiB to 32,768 GiB. For BMSs and ECSs, there can be a maximum of 59 data disks. For supernodes, there can be a maximum of 8 data disks. You can also expand the data disk capacity after the Lite Server is created. The data disk is automatically attached to each compute node. For details about data disk attaching and detaching, see **Using EVS for Storage**. • For supernodes, data disks can be attached and detached on the Lite Server details page, or via APIs of the Lite Server. • For BMSs or ECSs, data disks can be attached and detached on the Lite Server details page or the BMS/ECS console.

Table 4-6 Network configurations

Parameter	Description
VPC	A VPC ensures the security, isolation, and network flexibility of Lite Server resources. Choose the VPC associated with your Lite Server from the drop–down list. You are advised to choose the same VPC for all related cloud services to simplify network connections. If no VPC is available, click **Create VPC** on the right and create one in the displayed dialog box. To create a VPC, you need to log in to the management console as the administrator.
Subnet	Select a subnet of the current VPC. If no subnet is available, click **Create Subnet** on the right and create one in the displayed dialog box. You cannot manually assign subnet IP addresses for a Lite Server. Only automatic assignment is supported.

Parameter	Description
Security Group	A security group is a collection of access control rules for Lite Servers that have the same security requirements and that are mutually trusted within a VPC. If no security group is available, click **Create Security Group** on the right and create one in the displayed dialog box. Ensure that the selected security group allows the access to port 22 (for Linux SSH login), port 3389 (for Windows remote login), and ICMP (Ping). Disable other irrelevant ports or IP addresses. In the security group inbound rules, you are advised to set sources of high-risk ports to trusted IP addresses, security groups, or IP address groups. This can prevent service interruptions, data breaches, or data ransomware caused by network intrusions.
IPv6	IPv6 is available when it is supported by the subnet, specifications, and image configured for the network. Ensure that IPv6 has been enabled. To enable IPv6, see **Creating a Subnet for an Existing VPC**. This parameter is only displayed for certain specifications and images.
RoCE Network	This parameter is available only when **Node Type** is set to **Common node**. When A/H series GPUs, Snt9, or Snt9b resources are used for distributed training, you need to configure the RoCE network to use the RoCE NICs on the hardware. The parameter is only displayed if you have selected one specification that supports RoCE networks. If you have not created a RoCE network, click **Create RoCE**. If you have created a RoCE network, select it directly.
Supernode Network	This parameter is displayed only when **Node Type** is set to **Supernodes**. You can click **Add Super Node Network** on the right to create one. Supernode networks are mandatory for distributed scenarios.

Table 4-7 Node management configurations

Parameter	Description
Server Name	Lite Server name. which can contain 1 to 64 characters. Only digit, letters, underscores (_), and hyphens (-) are allowed. **CAUTION** The server name in the order will not be changed. If you change the name after placing the order, the new name will not be synchronized to the order.

Parameter	Description
Login Mode	**Key pair** is recommended as it features higher security than **Password**. If you select **Password**, ensure that the password meets complexity requirements to prevent malicious attacks. • **Key pair** Use a key pair to log in to the Lite Server. You can select an existing key pair, or click **Create Key Pair** to create one. If you use an existing key pair, ensure that you have saved the key file locally. Otherwise, logging in to the Lite Server will fail. • **Password** A username and its initial password are used for authentication and logging in to the Lite Server. For Linux, use the initial password of user **root**. For Windows, use the initial password of user **Administrator**. The password must: – Contain 8 to 26 characters. – Contain at least three types of the following characters: uppercase letters, lowercase letters, digits, and special characters (!@$%^-_=+[{}]:,./?). – Cannot be the same as the username or the username spelled backwards. – Cannot contain root, administrator, or their reverse.

Table 4-8 Advanced configurations

Parameter	Description
Cloud Eye host monitoring	Once this function enabled, you can configure Cloud Eye host monitoring agency in one–click mode. Cloud Eye agency allows you to monitor various metrics of the Lite Server, including CPU, memory, network, disk, and process at an interval of 1 minute. For details, see **Using Cloud Eye to Monitor NPU Resources of Lite Servers**.

Parameter	Description
NodeTaskHub	The NodeTaskHub plug–in is preset for certain public images. This parameter is displayed when the corresponding image is selected. Once enabled, the system automatically installs the NodeTaskHub plug–in for the task center to deliver software upgrade, pressure test, and fault diagnosis tasks. For details, see **Installing the AI Plugin for a Lite Server**. When this function is enabled, the system calls the AOM agency and installs the NodeTaskHub plugin on the Lite Server. This allows the task center to handle Ascend software upgrades, stress tests, and fault diagnoses. By calling the AOM agency, ModelArts Lite Server can call AOM APIs to interconnect with the monitoring and alarm systems.
Custom Instance Data Injection	Use this function to configure Lite Servers if you need to: • Simplify node configuration using scripts. • Initialize the system via scripts. • Upload existing scripts to the server during server creation. • Use scripts for other purposes. Currently, **As text** and **As file** are supported. For details, see **Injecting User Data into BMSs** or **Injecting User Data into ECSs**.

Table 4-9 Purchase configurations

Parameter	Description
Required Duration	Set the required duration and select auto–renewal as needed.
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. When purchasing 48 supernodes, reserve a portion as standby nodes based on your service requirements. These standby nodes will automatically take over if a primary node fails.

3. View the configuration fee in the lower left corner of the page and click **Buy Now**. Then, pay for the order on the payment page.

   The detailed fees will be displayed. You can click to view discount details if there is any. The configuration fee is the final discounted fee. To view the actual fee deduction, see the bill.

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

4. Once the payment is complete, wait patiently while the Lite Server resources are being provisioned. This process typically takes 20 to 60 minutes. If the resource fails to be created, see **Handling Resource Purchase Failures**.

Handling Resource Purchase Failures

If the ModelArts Lite Server fails to be created, there may be multiple causes:

- Insufficient resources: Switch to the BMS or ECS 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 if your account's resource quota (cores and RAM capacity) meets the requirements. If your quota is too low, you will need to increase it before proceeding. Apply for a quota adjustment before purchasing resources.
- Internal supernode, BMS, or ECS error: Check whether there is an internal BMS or ECS error. If yes, submit a service ticket to BMS or ECS to locate and rectify the fault.

5 Configuring Lite Server Resources

5.1 Resource Configuration Workflow

After enabling Lite Server resources, you need to complete the configurations by referring to the following flowchart.

Figure 5-1 Resource configuration workflow

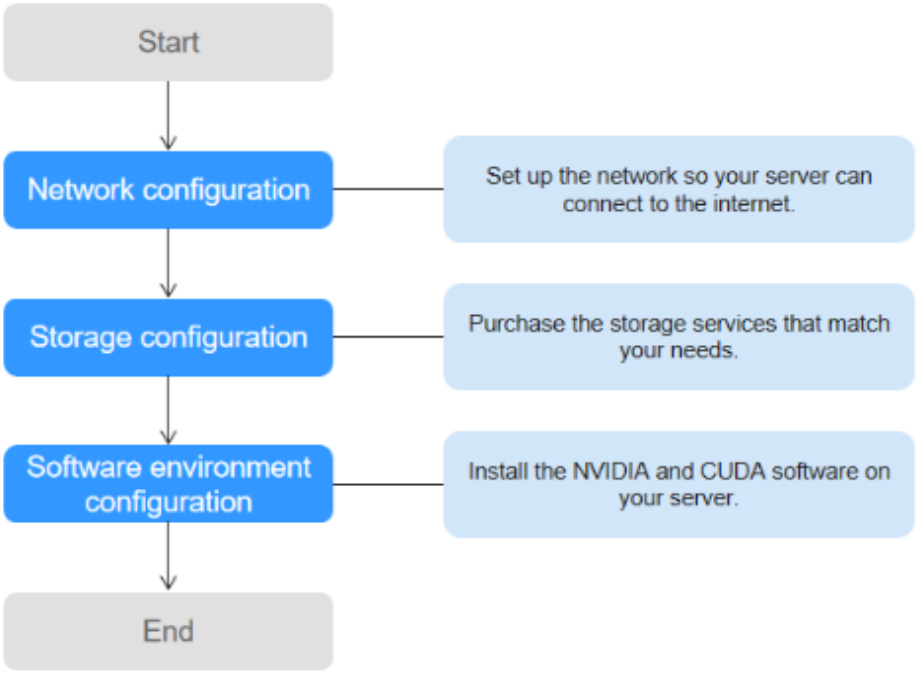


Table 5-1 Resource configuration workflow

Step	Task	Description
1	**Configuring the Lite Server Network**	After the Lite Server resource is activated, network configuration is required to enable communication with the internet. Since the Lite Server must be able to access the network during subsequent storage and software environment setup, network configuration should be completed first.
2	**Configuring the Lite Server Storage**	Data disks need to be mounted to the Lite Server to store data files. Currently, SFS, OBS, and EVS are supported.
3	**(Optional) Configuring the Software Environment for the Lite Servers**	The pre-installed software varies among images. For details about installed software, see **Mappings Between Lite Compute Nodes and OS Images**. If the software pre-installed in the Lite Server cannot meet service requirements, you can configure the required software environment on the Lite Server.

5.2 Configuring the Lite Server Network

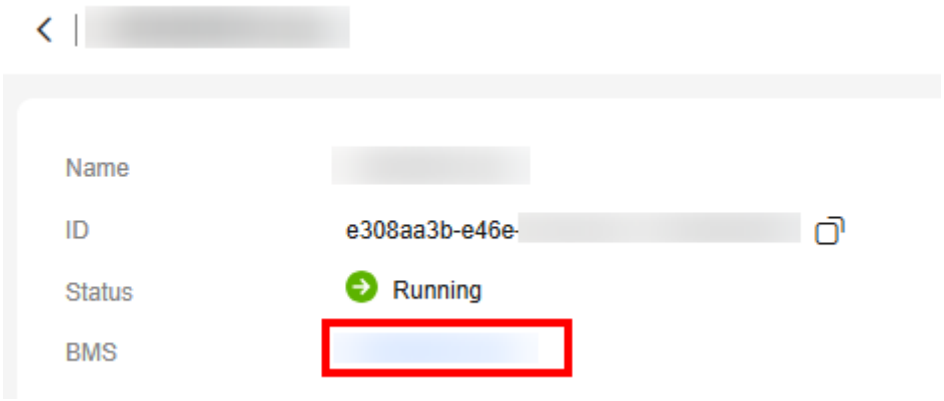
Configure the network so that the Lite Server can access the internet. Network configuration involves the following two scenarios:

- **Binding an EIP to a Single Lite Server**: Bind an EIP to a single Lite Server. This Lite Server will have exclusive access to the network bandwidth.
- **Binding an EIP to Multiple Lite Server**: Configure an EIP for a VPC and share it across all Lite Servers within that VPC via a NAT Gateway. This allows all nodes in the VPC to access the internet using the same EIP through shared network resources.

Binding an EIP to a Single Lite Server

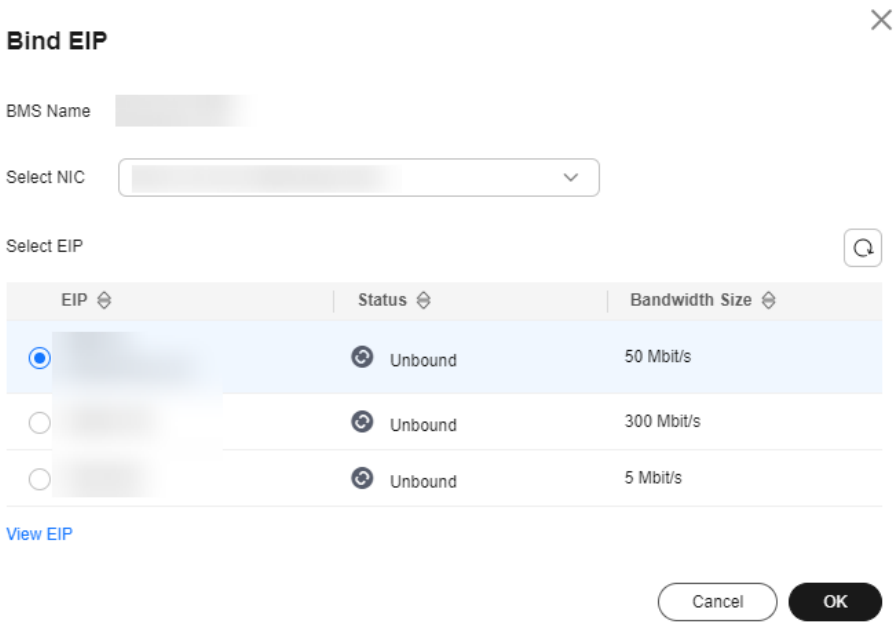
1. Log in to the **ModelArts console**. In the navigation pane, choose **Resource Management** > **Lite Servers**.
   - New console: In the navigation pane, choose **Resource Management** > **Lite Compute Resources** > **Lite Servers**.
   - Old console: In the navigation pane, choose **Resource Management** > **Lite Servers**.
2. In the **Common node** tab of the resource list page, click the name of the target Lite Server. The details page of the Lite Server is displayed.

Figure 5-2 BMS



3. Click the **EIPs** tab and then click **Bind EIP**. In the **Bind EIP** dialog box that appears, select the EIP to bind and click **OK**.

Figure 5-3 Binding an EIP



NOTE

Only one EIP can be bound to a NIC.

Binding an EIP to Multiple Lite Server

NOTE

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

1. **Buy an EIP**.

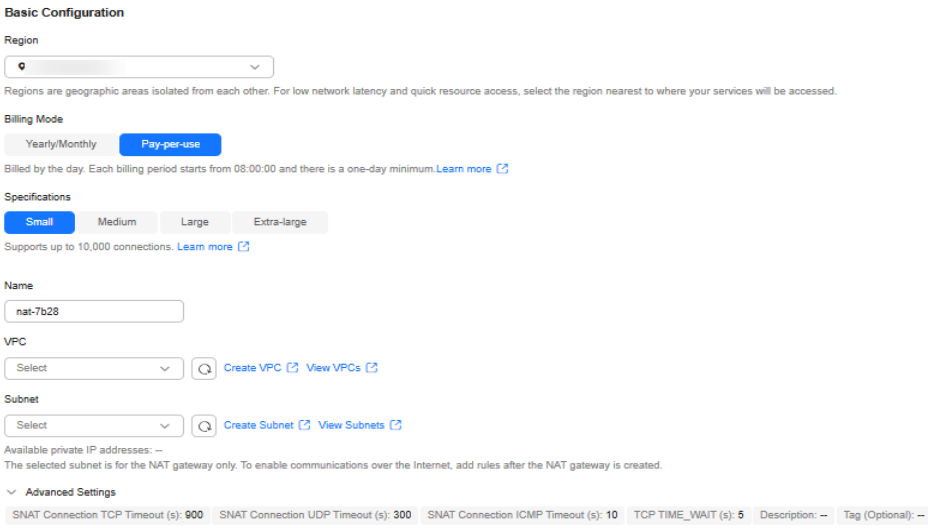
a. Log in to the **Huawei Cloud console**.
b. In the service list on the left, choose **Networking** > **Elastic IP**.
c. In the upper right corner of the EIP page, click **Buy EIP**.
d. Retain the default settings and click **Next**.
e. Confirm the configurations and select "I have read and agree to 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**.

2. **Buy a public NAT gateway**.

a. Log in to the **Huawei Cloud console**.
b. In the service list on the left, choose **Network** > **NAT Gateway**.
c. In the upper right corner of the Public NAT Gateway page, click **Buy Public NAT Gateway**.
d. Set **VPC**, **Subnet**, and **Billing Mode**. Retain the default settings for other parameters and click **Next**.

Figure 5-4 Buying a public NAT gateway



e. On the displayed page, confirm the NAT gateway information.

- 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 Lite Server.

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.

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

4. **Configure a DNAT rule.**

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

   a. In the **DNAT Rules** tab, click **Add DNAT Rule**.
   b. In the displayed dialog box, configure the DNAT rule as follows:
      - **Scenario**: Select **VPC**.
      - **Port Type**: Select **Specific port**.
      - **Protocol**: Select **TCP**.
      - **EIP**: Select the created EIP.
      - **Outside Port**: Ensure that the port number is unique. Recommended range: 20000–30000.
      - **Instance Type**: Click **Server** and select a server.
      - **NIC**: Select a NIC.
      - **Inside Port**: Choose **22**.
   c. Click **OK**.

5.3 Configuring the Lite Server Storage

Currently, SFS, OBS, and EVS are supported for Lite Servers. The table below describes the differences of storage solutions. For details about how to configure local disks, see **Merging and Mounting Disks**.

Table 5-2 Comparison of 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.
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 only be accessed after being mounted to Lite Servers 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 only be used and accessed from applications after being attached to Lite Servers and initialized.
Scenario	High-performance computing (HPC), media processing, file sharing, content management, and web services 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 HPC: High-speed and high-IOPS storage is required, such as industrial design and energy exploration.

Dimension	SFS	OBS	EVS
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
Online editing	Yes	No	Yes

Using SFS for Storage

If you use SFS for storage, SFS Turbo file systems are recommended. 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.

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.
2. Mount the created file system to the server. For details, see **Mounting an SFS Turbo File System to Linux ECSs**.
3. To avoid losing mount data when cloud servers restart, configure automatic mounting on restart. For details, see **Mounting an SFS Turbo File System to Linux ECSs**

Using OBS for Storage

Use OBS's parallel file system with obsutil for optimal performance. This system provides fast access to files, with low latency (milliseconds), high bandwidth (TB/s), and millions of IOPS. You can use obsutil, a command line tool for accessing OBS, to perform configurations in OBS, for example, creating a bucket, as well as 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.

1. Create a parallel file system on the OBS console. For details, see **Creating a Parallel File System**.
2. Download the corresponding obsutil to the Lite Server based on your OS and install it. For details, see **Downloading and Installing obsutil**.

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**.
4. Use obsutil to upload and download OBS files in the server. For details about obsutil, see **obsutil Introduction**.

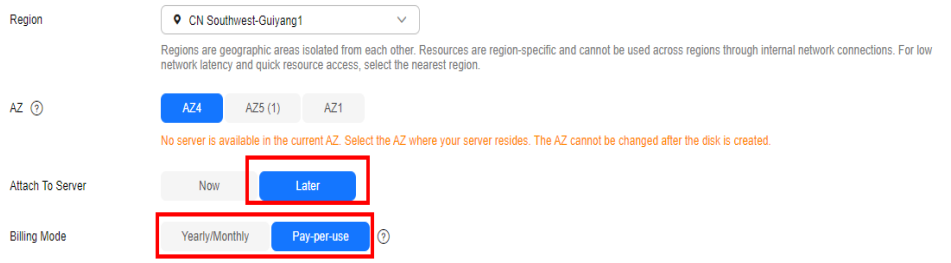
Using EVS for Storage

When purchasing Lite Server resources, you can attach EVS data disks. If the data disk space becomes insufficient after the Lite Server runs for a period of time, you can attach more data disks, which is known as post-attachment. In this case, you need to purchase EVS data disks and then attach them to the Lite Server. The following steps describe how to attach EVS disks in post-attachment mode.

1. Purchase an EVS data disk. Purchase a disk on the EVS console, select the AZ where the Lite Server is located, set **Attach to Server** to **Later**, set **Billing Mode** to **Yearly/Monthly** or **Pay-per-use**, and set the disk size as needed.
   - If the Lite Server resource type is set to BMS or supernode, the supported EVS disk types are subject to those displayed on the EVS console in the current AZ, and SCSI must be enabled in the advanced EVS configuration. A SCSI disk allows the server OS to directly access the underlying storage media and send SCSI commands to the disk.
   - If the Lite Server resource type is set to ECS, the supported EVS disk types are subject to those displayed on the EVS console in the current AZ. VBD and SCSI EVS disk types are supported. You can choose whether to enable SCSI.

   For details about EVS purchase parameters, see **Purchasing an EVS Disk**.

   Figure 5-5 Buying a disk

   NOTE

   Currently, an EVS disk cannot be attached 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 operations system. 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**.
2. After purchasing an EVS data disk, attach it to an existing Lite Server. You can choose a disk attaching mode.

   Attaching and detaching disks on the Lite Server details page:

Log in to the ModelArts console. In the navigation pane on the left, choose **Lite Servers**. Click a server name to access its details page. In the **Disk** tab, select an EVS data disk and set the mount point.

To detach a disk, click **Detach** on the details page of any Lite Server type to remove the corresponding data disk. System disks cannot be detached.

NOTE

- The attached EVS data disk will not be automatically deleted when you unsubscribe from a BMS. You can attach the disk to other BMSs or delete it as needed.
- When you purchase a yearly/monthly lightweight compute node on an ECS, BMS, or HPS instance, the data disk that comes with the node cannot be unsubscribed from separately.

If the Lite Server type is BMS, you can attach a disk on the BMS details page. For details, see **Attaching Data Disks to a BMS**.

If the Lite Server type is ECS, you can attach a disk on the ECS details page. For details, see **Attaching a Disk to an ECS**.

If the Lite Server type is supernode, you can also use the attaching and detaching APIs of the Lite Server.

3. After an EVS data disk is attached, it needs to be initialized. For details, see **Initialization Overview**.

5.4 (Optional) Configuring the Software Environment for the Lite Servers

5.4.1 Configuring the Resource Software Environment for NPU-based Lite Servers

Scenario

This section describes how to configure environments for NPU-based Lite Servers, including merging and mounting disks, and installing Docker. **Table 5-3** lists the configuration items.

The latest Lite Servers come with most settings preconfigured. You can skip these steps.

Table 5-3 Physical machine environment configuration items

Configuration Item	Application Scope
Configuring Server SSH Connection Timeout	All server models
Merging and Mounting Disks	All server models
Installing the Driver and Firmware	NPU servers only
Install the Docker environment.	All server models

Configuration Item	Application Scope
Installing the pip Source	All server models
Testing the RoCE Network	Snt9b servers only
Creating a Containerized Custom Debugging Environment	NPU servers only

Configuration Precautions

Before the configuration, pay attention to the following:

- During 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-PU compute resources, which can be used by multiple users for development and debugging. To avoid usage conflicts, PUs 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.

Configuring Server SSH Connection Timeout

1. Log in to the Lite Server using SSH and check the timeout configuration.
   ```
   echo $TMOUT
   ```
2. If it is set to **300**, the server will be 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
   ```
3. Run the following command for the configuration to take effect on the current terminal:
   ```
   TMOUT=0
   ```

   By running the **export TMOUT=0** command, the idle timeout of the session is set to 0 during SSH connection to the Linux server, that is, the connection will not be automatically disconnected due to idleness. For security purposes, SSH connections may be automatically disconnected if no operation is performed for a while. However, if you are performing a task that requires long-time connection, run this command to prevent disconnection caused by idleness.

You can run the **TMOUT=0** command in the current terminal session or add **export TMOUT=0** to the **/etc/profile** file. In this way, new sessions of all users will not be disconnected due to idleness.

Do not configure **TMOUT=0** in the production environment or on a public server, as it will bring certain security risks.

Merging and Mounting Disks

After provisioning Lite Server resources, the server may contain multiple unmounted NVMe disks. Therefore, you must merge and mount these disks before configuring the environment for the first time. This operation must be performed at the very first so that the content you stored will not be overwritten.

The local non-volatile memory express (NVMe) disks may get damaged. To avoid **training task interruption or compute waste** caused by local disk faults, you are advised to:

1. Purchase and use cloud storage to periodically synchronize data and model files.
2. Add periodic checkpointing or auto-save mechanisms into your training or development code to save model parameters and key intermediate results in a timely manner.

These can effectively reduce the impact of local disk faults on task stability and data security.

The following describes how to merge and mount disks.

1. Run **lsblk** to check whether there are three 7 TB disks that are unmounted.

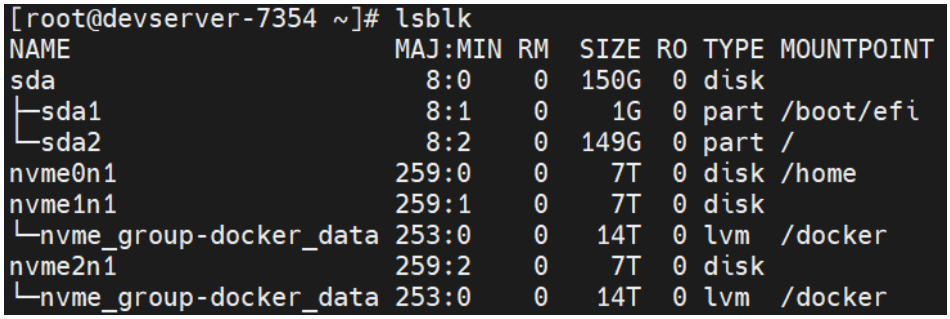
   As shown in **Figure 5-6**, **nvme0n1**, **nvme1n1**, and **nvme2n1** are unmounted.

   Figure 5-6 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 ~]#
   ```

   As shown in **Figure 5-7**, the **MOUNTPOINT** column indicates the directory where the disk is mounted. You can skip this step and create a directory in **/home**.

Figure 5-7 Mounted disks



2. Edit the disk mounting script **create_disk_partitions.sh**. This script mounts **/dev/nvme0n1** to **/home**, allowing each developer to create their own home directory. It also merges and mounts the **nvme1n1** and **nvme2n1** local disks to **/docker** for container use. If a large enough space is not specifically allocated to **/docker**, the root directory can easily become full when multiple users share the same Lite Server and create multiple container instances.

```
vim create_disk_partitions.sh
```

The following content shows the **create_disk_partitions.sh** script, which can be directly used without modification:

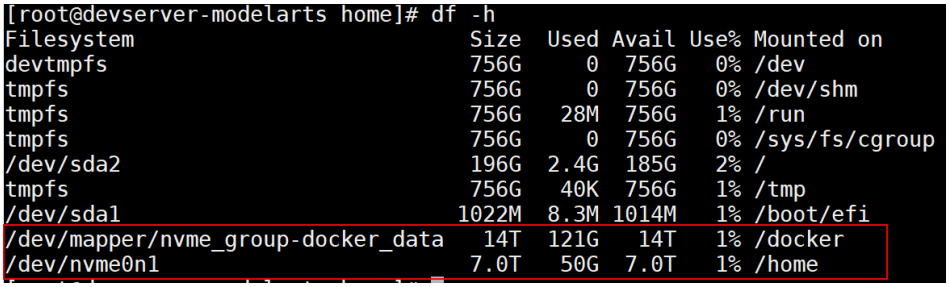
```
# ============================================================================
# Mount the nvme0n1 local disk to the /home directory.
# Merge the nvme1n1 and nvme2n1 local disks as a logical volume and mount them to the /docker directory. Set automatic attaching upon system startup.
# ============================================================================
set -e
# Mount nvme0n1 to the user directory.
mkfs -t xfs /dev/nvme0n1
mkdir -p /tmp/home
cp -r /home/* /tmp/home/
mount /dev/nvme0n1 /home
mv /tmp/home/* /home/
rm -rf  /tmp/home
# Mount nvme1n1 and nvme2n1 to the /docker directory.
pvcreate /dev/nvme1n1
pvcreate /dev/nvme2n1
vgcreate nvme_group  /dev/nvme1n1 /dev/nvme2n1
lvcreate -l 100%VG -n docker_data nvme_group
mkfs -t xfs /dev/nvme_group/docker_data
mkdir /docker
mount /dev/nvme_group/docker_data /docker
# Migrate Docker files to the new /docker directory.
systemctl stop docker
mv /var/lib/docker/* /docker
sed -i '/"default-runtime"/i\        "data-root":     "/docker"," /etc/docker/daemon.json
systemctl start docker
# Enable automatic attaching upon system startup.
uuid=`blkid -o value -s UUID /dev/nvme_group/docker_data` && echo UUID=${uuid} /docker xfs defaults,nofail 0 0 >> /etc/fstab
uuid=`blkid -o value -s UUID /dev/nvme0n1` && echo UUID=${uuid} /home xfs defaults,nofail 0 0 >> /etc/fstab
mount -a
df -h
```

3. Run the **create_disk_partitions.sh** script.

```
sh create_disk_partitions.sh
```

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

Figure 5-8 Viewing mounted disks



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

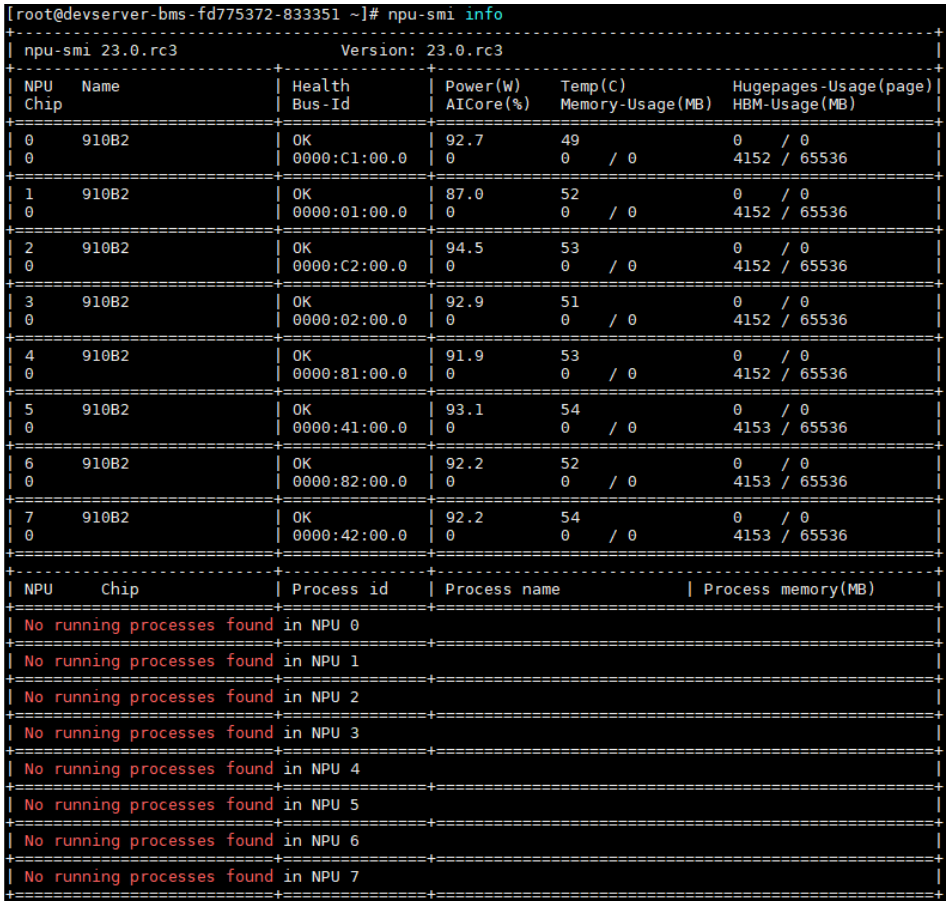
Installing the Driver and Firmware

1. Check whether the npu-smi tool can be used properly. The firmware and driver installation can continue only when the npu-smi tool can be used properly. Run the command below. If the output is the same as that shown in **Figure 5-9**, the npu-smi tool is normal.

   ```
   npu-smi info
   ```

   If the command output is not complete as shown in the figure below (for example, an error is reported or only the upper part of the command output is displayed without the following process information), restore the npu-smi tool and install the firmware and driver of the new version. To do so, submit a **service ticket** to contact Huawei Cloud technical support.

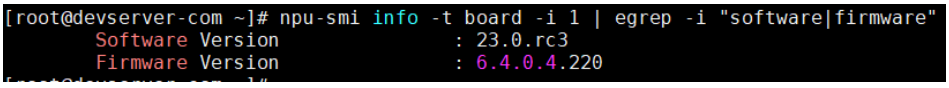
Figure 5-9 Checking the npu-smi tool



2. 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 5-10 Viewing the firmware and driver versions



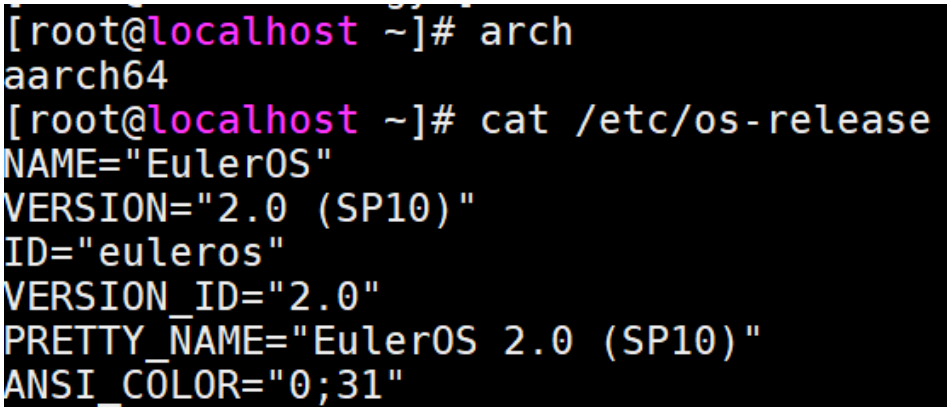
firmware indicates the firmware version, and **software** indicates the driver version.

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

3. View the OS version, check whether the architecture is AArch64 or x86_64, and obtain the firmware and driver packages from the 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 **the download link**.

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

Figure 5-11 Viewing the OS version and architecture



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

4. Install the driver and firmware.

CAUTION

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 Snt9b and Snt9b23 servers before delivery. So in this case, overwrite installation is used.

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.

Installation commands:

a. Install the firmware and then restart the server.

```
chmod 700 *.run
#  Actual package name
./Ascend-hdk-<model>-npu-firmware_<version>.run  --full
reboot
```

b. Install the driver. Enter **y** as prompted.

```
# Actual package name
./Ascend-hdk-<model>-npu-driver_<version>_linux-aarch64.run --full --install-for-all
```

c. **(Optional)** Determine whether to restart the system as prompted. If yes, run the following command. If not, skip this step.

```
reboot
```

d. 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 5-12 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
```

Install the Docker environment.

ModelArts Lite Server's public images already include Docker. To install Docker manually, follow these steps:

1. Run the command below to check whether Docker has been installed. If Docker has been installed, skip this step. **Figure 5-13** shows that Docker has been installed.
   ```
   docker -v
   ```

   Figure 5-13 Viewing the Docker version

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

   If Docker is not installed, run this command to install it:
   ```
   yum install -y docker-engine.aarch64 docker-engine-selinux.noarch docker-runc.aarch64
   ```
   After the installation is complete, run the **docker -v** command again to check whether Docker is installed.
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 5-14 Querying Ascend-docker-runtime

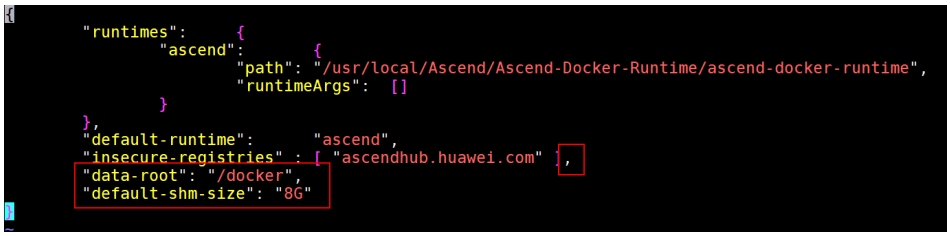
   ```
   [root@devserver-modelarts-demanager-0eaabe8f ~]# docker info |grep Runtime
   Runtimes: ascend runc
   Default Runtime: ascend
   ```

   If Ascend-docker-runtime is not installed, click **Ascend Docker Runtime** to obtain it. The software package is a Docker plugin 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 Lite Server and install it.
   ```
   chmod 700 *.run
   ./Ascend-hdk-<model>-npu-driver_<version>_linux-aarch64.run --install
   ```
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 5-15 Configuring Docker



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

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

Installing the pip Source

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
   ```

Testing the RoCE Network

The following RoCE network test only applies to Snt9b servers. For details about how to test the RoCE network of Snt9b23 servers, see **Diagnosing Faults on Lite Servers**.

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 has not been installed, download the software package from the official website. The following uses **Ascend-cann-toolkit_8.0.1_linux-aarch64.run** as an example.

   Install CANN Toolkit. Replace the package name.

   ```
   chmod 700 *.run
   ./Ascend-cann-toolkit_8.0.1_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 5-16 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_ch
   eck_buf_init.cc ./common/src/hccl_check_common.cc ./common/src/hccl_opbase_root
   info_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/a
   scend-toolkit/latest/lib64 -lhccl -L/usr/local/Ascend/ascend-toolkit/latest/lib
   64 -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_ch
   eck_buf_init.cc ./common/src/hccl_check_common.cc ./common/src/hccl_opbase_root
   info_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/a
   scend-toolkit/latest/lib64 -lhccl -L/usr/local/Ascend/ascend-toolkit/latest/lib
   64 -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-PU, run the following command:

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

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

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


Figure 5-17 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 5-18 Viewing Ascend RoCE IP address

```
[root@devserver-com hccl_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 **-i***<PU-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 **-i***<PU-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
hccn_tool -i 0 -roce_test reset
hccn_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 5-19 RoCE test result (receive end)

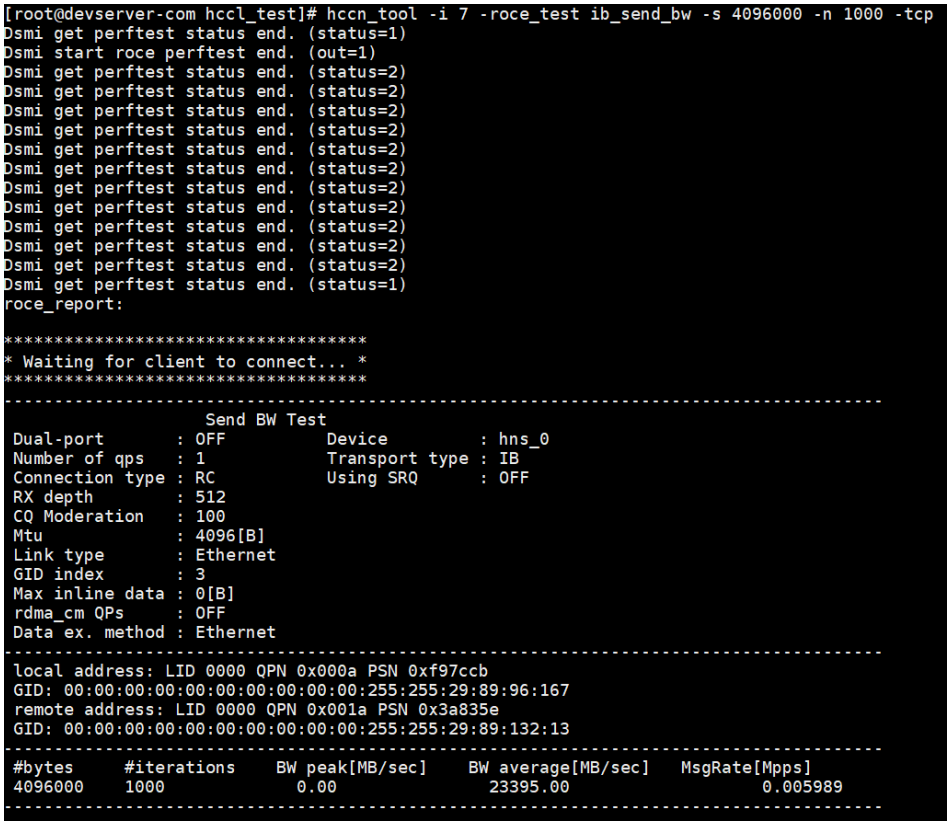
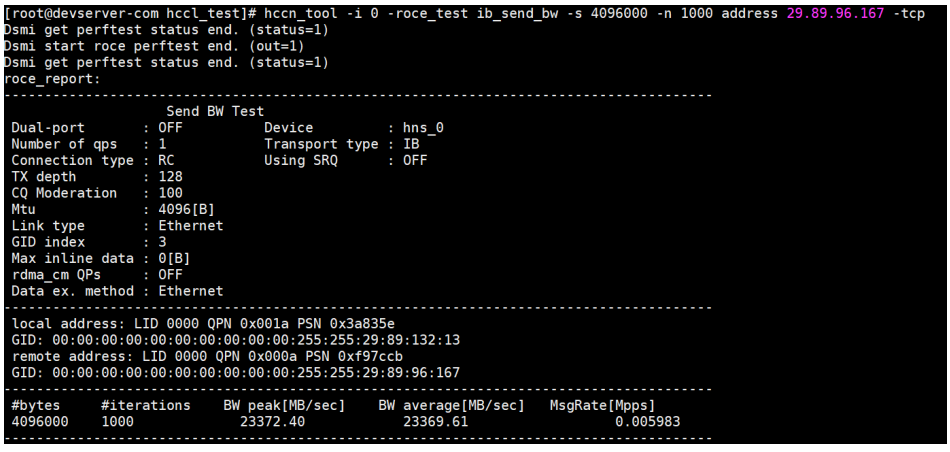


Figure 5-20 RoCE test result (server)



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

Figure 5-21 Error

```
[root@devserver-com hccl_test]# hccn_tool -i 7 -roce_test ib_send_bw -s 4096 -n
 1000 -tcp
Dsmi get perftest status end. (status=2)
Roce perftest 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 common node:

```
for i in $(seq 0 7);do hccn_tool -i $i -net_health -g;done
```

Creating a Containerized Custom Debugging Environment

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. Prepare a service base image.
   a. 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>
      ```

   b. Start the container image. If multiple users and containers are sharing a machine, allocate the PUs beforehand. Do not use PUs occupied by other containers.

      ```
      #  Start the container. Specify the container name and image information.
      ASCEND_VISIBLE_DEVICES indicates the PUs to be used by the container, for example, 0-1,3 indicates PUs 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
      ```

   c. Access the container.

      ```
      docker exec -ti <custom-container-name-in-the-last-command> bash
      ```

   d. Access the Conda environment.

      ```
      source /home/ma-user/.bashrc
      cd ~
      ```

   e. View the information of available PUs in the container.

      ```
      npu-smi info
      ```

      If the following error message is displayed, the PU specified by **ASCEND_VISIBLE_DEVICES** during container startup is occupied by another container. In this case, select another PU and restart the new container.

      Figure 5-22 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
      ```


f. After you run **npu-smi info** and the output is normal, run the commands below 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 5-23 Accessing the Conda environment and performing a test

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:

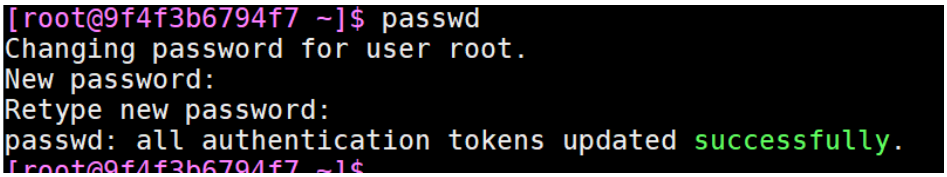
   a. After accessing the container, run the SSH startup command to start the SSH service.

```
ssh-keygen  -A
/usr/sbin/sshd
#  Check whether SSH is started.
ps -ef |grep ssh
```

   b. Set a password for user **root** as prompted.

```
passwd
```

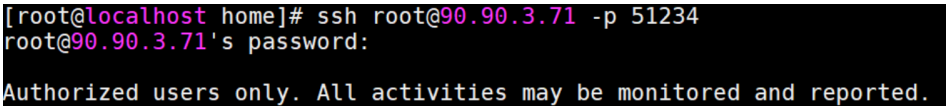
Figure 5-24 Setting a password for user **root**



   c. 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 5-25 Perform the SSH test.



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.

d. 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 plugin 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 plugin. For details, see **Install the Python Plug-in in the Cloud Development Environment**.

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.

a. 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
```

b. If it has not been installed or needs to be upgraded, download the software package from the official website. The following uses **Ascend-cann-toolkit_8.0.1_linux-aarch64.run** as an example.

Install CANN Toolkit. Replace the package name.

```
chmod 700 *.run
./Ascend-cann-toolkit_8.0.1_linux-aarch64.run --full --install-for-all
```

c. 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
```

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.

a. 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
```

b. 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
```

5. Configure the pip source.

   The pip source has been configured in the preset image provided by ModelArts. To use your own service images, configure it by referring to **Installing the pip Source**.

6. Configure a Yum repository.

   – Configure the Yum repository in Huawei EulerOS.
```
# Create the EulerOS.repo file in the /etc/yum.repos.d/ directory,
cd /etc/yum.repos.d/
mv EulerOS.repo EulerOS.repo.bak
vim EulerOS.repo
# Configure the EulerOS.repo file based on the EulerOS version and system architecture. EulerOS 2.10 is used as an example.
[base]
name=EulerOS-2.0SP10 base
baseurl=https://mirrors.huaweicloud.com/euler/2.10/os/aarch64/
enabled=1
gpgcheck=1
gpgkey=https://mirrors.huaweicloud.com/euler/2.10/os/RPM-GPG-KEY-EulerOS
# Clear the existing Yum cache.
yum clean all
# Generate a new Yum cache.
yum makecache
# Perform a test.
yum update --allowerasing --skip-broken --nobest
```

   – Configure a Yum repository in Huawei Cloud EulerOS.
```
# Download the new hce.repo file to the /etc/yum.repos.d/ directory.
wget -O /etc/yum.repos.d/hce.repo https://mirrors.huaweicloud.com/artifactory/os-conf/hce/hce.repo
# Clear the existing Yum cache.
yum clean all
# Generate a new Yum cache.
yum makecache
# Perform a test.
yum update --allowerasing --skip-broken --nobest
```

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

   a. 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
```

   b. 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
```

   c. Disable SSL verification for Git configuration.
```
git config --global http.sslVerify false
```


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

```
#  git clone diffusers source code. You can specify a branch for the -b 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 **git clone** to download the model on Hugging Face. The following uses a stable-diffusion (SD) model as an example.

If error "SSL_ERROR_SYSCALL" 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 5-26 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]#
```

8. 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**.
9. 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 TAR file. This command takes a long time. You can view the file by running the ls command 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.

5.4.2 Creating the OS of a Lite Server

Description

If the current Lite Server OS does not meet your requirements, you can use BMS or ECS to create an image and save the current OS as a new image for other Lite Servers.

Constraints

Before creating an image, ensure that the Lite Server is stopped.

The created image can only be created based on the current OS of the Lite Server.

Procedure

1. Delete temporary files before creating an OS image to prevent potential issues with running the image. To do so, log in to the Lite Server. Run the commands below separately or create a script to run the commands all at once. For details about the clearance script, see **Script for Deleting Temporary Files**.
   a. Clear user login records:
   ```
   echo > /var/log/wtmp
   echo > /var/log/btmp
   ```
   b. Delete temporary files:
   ```
   rm -rf /var/log/cloud-init*
   rm -rf /var/lib/cloud/*
   rm -rf /var/log/network-config.log
   rm -rf /opt/huawei/network_config/network_config.json
   rm -rf /opt/huawei/port_config/uplink_hash_config.log
   rm -rf /opt/huawei/firmware_check/firmware_check.log
   ```
   c. Delete residual configurations.
      - CentOS/EulerOS/HCE OS: Check the files whose names start with **ifcfg** in the **/etc/sysconfig/network-scripts/** folder and delete them except **ifcfg-lo**.

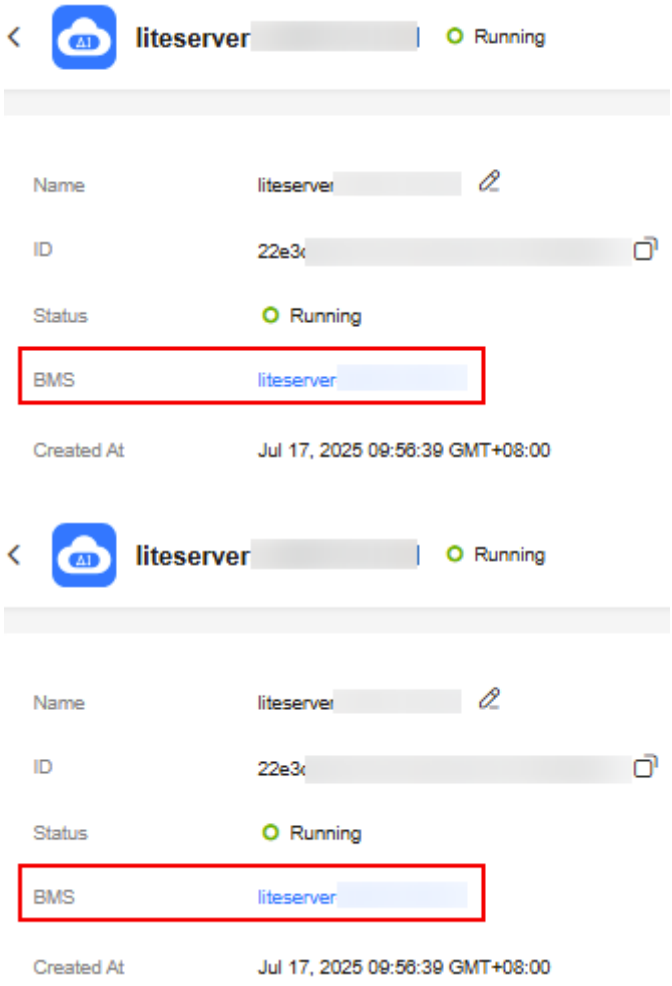
        Check files:
        ```
        ll /etc/sysconfig/network-scripts/
        ```
        Delete files:
        ```
        rm -rf /etc/sysconfig/network-scripts/ifcfg<i>xxx</i>
        ```
      - Ubuntu:
        ```
        rm -rf /etc/network/interfaces.d/50-cloud-init.cfg
        ```
   d. Clear the network information of the parameter plane:
   ```
   echo >/etc/netplan/roce.yaml
   echo > /etc/hccn.conf
   ```
   e. Clear operation records:
   ```
   history -w;echo > /root/.bash_history;history -c;history -c;history -c;
   ```
2. Shut down the Lite Server.
   a. Log in to the **ModelArts console**. In the navigation pane, choose **Resource Management** > **Lite Servers**.
      i. New console: In the navigation pane, choose **Resource Management** > **Lite Compute Resources** > **Lite Servers**.

ii. Old console: In the navigation pane, choose **Resource Management** > **Lite Servers**.

b. Choose **Resources** > **Common node** and click **Shut Down** in the **Operation** column to stop the target Lite Server.

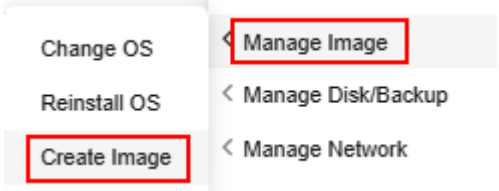
3. On the Lite Server details page, click the BMS address or ECS address to access its details page.

Figure 5-27 Lite Server details page



4. On the BMS or ECS console, return to the server list page, and choose **Manage Image** > **Create Image** in the **Operation** column.

Figure 5-28 Creating an image on an ECS



On the image creation page, enter the image name and enterprise project, select the agreement, and confirm the creation. The created image is saved in

the private image list of the IMS service. For details, see **Creating an Image** and **Creating a Private Image from a BMS**.

The created image can be used for other Lite Servers. For details, see **Changing or Resetting the OS of a Lite Server**.

Script for Deleting Temporary Files

```
#!/bin/bash

# Clear user login records:
echo > /var/log/wtmp
echo > /var/log/btmp

# Delete temporary files:
rm -rf /var/log/cloud-init*
rm -rf /var/lib/cloud/*
rm -rf /var/log/network-config.log
rm -rf /opt/huawei/network_config/network_config.json
rm -rf /opt/huawei/port_config/uplink_hash_config.log
rm -rf /opt/huawei/firmware_check/firmware_check.log

# Delete residual configurations:
## CentOS/EulerOS/HCE OS:
## Check the files whose names start with ifcfg in the /etc/sysconfig/network-scripts/ folder and delete them except ifcfg-lo.
find /etc/sysconfig/network-scripts/ -name 'ifcfg*' ! -name 'ifcfg-lo' -type f | xargs rm -f
## Ubuntu:
rm -rf /etc/network/interfaces.d/50-cloud-init.cfg

# Clear the network information of the parameter plane:
echo > /etc/netplan/roce.yaml
echo > /etc/hccn.conf

# Clear operation records:
history -w;echo > /root/.bash_history;history -c;history -c;history -c;
echo > ~/.bash_history
exec bash
```

5.4.3 Configuring the NPU Driver Firmware Consistency and UDP Port Hash

Description

In the public OSs of the Lite Server Snt9b and supernode Snt9b23, configurations for driver-firmware consistency and UDP port hash have been implemented. The **driver-firmware consistency** configuration ensures that the firmware is automatically refreshed after the server is shut down and restarted, maintaining the consistency between the OS HDK driver and firmware. The **UDP port hash** configuration ensures that the uplink port configuration of the parameter plane network is automatically refreshed after the server is shut down and restarted, preventing congestion in the parameter plane network.

When using a privately built OS, if you do not configure driver-firmware consistency and UDP port hash, the system will lack the ability to automatically refresh after the server is powered off and restarted. Therefore, it is recommended that you configure driver-firmware consistency and UDP port hash on your private OS.

Constraints

You can only configure driver firmware consistency and port hash for Lite Server Snt9b and supernode Snt9b23.

The driver firmware consistency configuration and UDP port hash configuration depend on bms-network-config. You need to first complete the cloud-based adaptation of the OS and install the software package. For details, see the **BMS image creation documentation**.

Configuring Driver Firmware Consistency

Obtain the required NPU driver and firmware packages based on your model and chip architecture from **Huawei Support**. HDK25.2.1 is used as an example to describe how to configure driver firmware consistency on an Arm64-powered Snt9b23 supernode.

Table 5-4 Driver and firmware version mapping requirements

Type	Package	Version
Driver package	Atlas-A3-hdk-npu-driver_25.2.1_linux-aarch64.run	25.2.1
Firmware package	Atlas-A3-hdk-npu-firmware_7.7.0.9.220.run	7.7.0.9.220

1. Upload the downloaded NPU driver package and firmware package to any directory on the Lite Server. You can use Xftp or an OBS bucket.
2. Log in to the Lite Server and go to the directory where the software package is to be uploaded.
3. Install the NPU driver.

   For details, see "Installing the Driver and Firmware" in **Configuring the Resource Software Environment for NPU-based Lite Servers**.
4. Create a driver firmware consistency configuration directory:
   ```
   mkdir -p /opt/huawei/firmware_check
   ```
5. Create a driver firmware consistency configuration script:
   ```
   cd /opt/huawei/firmware_check
   touch firmware_check.sh
   ```
6. Move the firmware package to the driver firmware consistency directory and modify the software package name:
   ```
   mv Atlas-A3-hdk-npu-firmware_7.7.0.9.220.run /opt/huawei/firmware_check/Ascend-hdk-npu-firmware_7.7.0.9.220.run
   ```

   Note: If the firmware version is different from the example or the version is changed later, modify the software package version number in the preceding commands accordingly.
7. Open the **firmware_check.sh** script and add the content below.

   Open the script file.


```
vim firmware_check.sh
```

Add the following content:

```
#!/bin/bash
HOME_DIR="/opt/huawei/firmware_check"
MAX_LOG_LINES=500
LOG_FILE="${HOME_DIR}/firmware_check.log"
ASCEND_INSTALL_INFO="/etc/ascend_install.info"
ASCEND_PATH="/usr/local/Ascend"
MIN_DURATION=3600
# ***If the driver version is changed during image creation, update the content below.***
PRESET_SOFTWARE_VERSION="25.2.1"
PRESET_FIRMWARE_VERSION="7.7.0.9.220"
function init_log() {
    if [ ! -f ${LOG_FILE} ]; then
        cat /dev/null > $LOG_FILE
        return
    fi
    line_count=$(wc -l < "${LOG_FILE}")
    if [ $line_count -gt $MAX_LOG_LINES ]; then
        tail -n "${MAX_LOG_LINES}" "${LOG_FILE}" > "${LOG_FILE}.tmp"
        mv "${LOG_FILE}.tmp" "${LOG_FILE}"
    fi
}
function log_info() {
    echo -e "$(date +%Y-%m-%d" "%H:%M:%S):[Info]$@" >> $LOG_FILE && echo -e "\033[32m[INFO]
\033[0m: $@" > /dev/tty
}
function log_warning() {
    echo -e "$(date +%Y-%m-%d" "%H:%M:%S):[Warnning]$@" >> $LOG_FILE && echo -e
"\033[33m[WARN]\033[0m: $@" > /dev/tty
}
function log_error() {
    echo -e "$(date +%Y-%m-%d" "%H:%M:%S):[Error]$@" >> $LOG_FILE && echo -e
"\033[31m[ERROR]\033[0m: $@" > /dev/tty
}
function get_param_from_config() {
    file=$1
    wanted=$2
    if [ ! -e $file ]; then
        log_error "File ${file} does not exist."
        return 1
    fi
    while IFS="=" read -r key val; do
        key=$(echo "$key" | tr -d '[:space:]')
        if [[ "$key" == "$wanted" ]]; then
            echo $val
            return 0
        fi
    done < "$file"
    return 1
}
function get_object_version() {
    install_info=$1
    install_path_key=$2
    default_install_path=$3
    object_type=$4
    install_path=$(get_param_from_config "${ASCEND_INSTALL_INFO}" "${install_path_key}")
    if [ $? -ne 0 ]; then
        log_warning "Failed to get value of ${install_path_key} from ${install_info}, use $
{default_install_path}."
        install_path=${default_install_path}
    fi
    version_info_file="${install_path}/${object_type}/version.info"
    version=$(get_param_from_config "${version_info_file}" "Version")
    if [ $? -ne 0 ]; then
        log_error "Failed to get Version value from ${version_info_file}."
        return 1
    fi
    echo $version
```



```
    return 0
}
function get_versions_with_tool() {
    output=$(npu-smi info -t board -i ${1})
    if [ $? -ne 0 ]; then
        log_error "Run command 'npu-smi info -t board -i ${i}' error:\n$output"
        return 1
    fi
    software_version=$(echo "$output" | awk -F':' '/Software Version/ {print $2}' | tr -d '[:space:]')
    firmware_version=$(echo "$output" | awk -F':' '/Firmware Version/ {print $2}' | tr -d '[:space:]')
    if [ -n "${software_version}" ] && [ -n "${firmware_version}" ]; then
        echo "${software_version}|${firmware_version}"
        return 0
    fi
    return 1
}
function check_versoins() {
    software_version=$1
    firmware_version=$2
    if [ "${software_version}" != "${PRESET_SOFTWARE_VERSION}" ]; then
        log_warning "The current software version is not preset version '$
{PRESET_SOFTWARE_VERSION}', does not need to upgrade the firmware."
        echo "false"
        return
    fi
    if [ -n "${firmware_version}" ] && [ "${firmware_version}" == "${PRESET_FIRMWARE_VERSION}" ];
then
        log_info "The current firmware version match preset version '${PRESET_FIRMWARE_VERSION}',
does not need to be upgraded."
        echo "false"
        return
    fi
    log_info "The current firmware '${firmware_version}' does not match preset version '$
{PRESET_FIRMWARE_VERSION}', need to be upgraded."
    echo "true"
    return
}
function upgrade_firmware() {
    firmware_version=$1
    firmware_package="${HOME_DIR}/Ascend-hdk-npu-firmware_${firmware_version}.run"
    if [ ! -e ${firmware_package} ]; then
        log_error "Firmware package ${firmware_package} does not exist."
        return 1
    fi
    bash ${firmware_package} --full --quiet >> $LOG_FILE 2>&1
    return $?
}
function upgrade_mcu() {
    bash ${HOME_DIR}/upgrade_mcu.sh ${HOME_DIR}/Ascend-hdk-mcu_$
{PRESET_MCU_VERSION}.hpm
}
function permit_check() {
    if [ ! -e ${TIME_TAG} ]; then
        log_info "First check of this machine, permit."
        return 0
    fi
    this_time=$(date +%s)
    if [ $? -ne 0 ]; then
        log_error "Failed to get current time, error code $?."
        return 1
    fi
    last_time=$(stat -c %Y ${TIME_TAG})
    if [ $? -ne 0 ]; then
        log_error "Failed to get the timestamp of last check, error code $?."
        return 1
    fi
    duration=$(expr $this_time - $last_time)
    if [ $? -ne 0 ]; then
        log_error "Failed to caculate duration, forbidden."
```

```
        return 1
    fi
    if [ $duration -lt ${MIN_DURATION} ]; then
        log_info "Duration less than ${MIN_DURATION} sec, forbidden."
        return 1
    fi
    log_info "Duration from this time to last time is ${duration} sec."
    return 0
}
function main() {
    log_info "Start to check the firmware."
    init_log
    permit_check
    if [ $? -ne 0 ]; then
        log_error "This check is too frequent, try later."
        return
    fi
    firmware_reboot="false"
    need_upgrade="false"
    for i in $npu_ids; do
        result=$(get_versions_with_tool ${i})
        if [ $? -eq 0 ]; then
            software_version=$(echo $result | awk -F'|' '{print $1}')
            firmware_version=$(echo $result | awk -F'|' '{print $2}')
            log_info "npu ${i} software version and firmware version from npu-smi are $
{software_version} and ${firmware_version}."
        else
            log_warning "npu ${i} failed to get versions using npu-smi, try to read the software version
from config file."
            if [ -z "${config_software_version}" ]; then
                log_error "failed to get software version from config file."
                return
            fi
            software_version=${config_software_version}
            log_info "npu ${i} software version from config is ${software_version}."
        fi
        need_upgrade=$(check_versoins "${software_version}" "${firmware_version}")
        if [ "${need_upgrade}" == "true" ]; then
            log_info "npu ${i} firmware need upgrade."
            break
        fi
    done
    if [ "${need_upgrade}" == "true" ]; then
        log_info "Start upgrading the firmware."
        upgrade_firmware ${PRESET_FIRMWARE_VERSION}
        if [ $? -ne 0 ]; then
            log_error "Failed to upgrade firmware."
        else
            log_info "The firmware upgrade to '${PRESET_FIRMWARE_VERSION}' succeeded, reboot now."
            firmware_reboot="true"
        fi
    fi
    if [[ "${firmware_reboot}" == "true" ]]; then
        log_info "Start to reboot."
        touch ${TIME_TAG}
        reboot
    fi
    log_info "The firmware check completed."
}
metadata=$(/usr/bin/curl http://169.254.169.254/openstack/latest/meta_data.json)
if [ $? -ne 0 ]; then
    log_error "Failed to get metadata, error code $?."
    exit
fi
if [ -z "${metadata}" ]; then
    log_error "Metadata is empty, abort."
    exit
fi
uuid=$(echo ${metadata} | grep -o '"uuid": "[^"]*' | sed 's/"uuid": "//')
```

```
TIME_TAG="${HOME_DIR}/timetag${uuid}"
config_software_version=$(get_object_version "${ASCEND_INSTALL_INFO}"
"Driver_Install_Path_Param" "${ASCEND_PATH}" "driver")
if [ $? -ne 0 ]; then
    log_error "npu ${i} failed to get software version from config file."
fi
npu_ids=$(npu-smi info -l | awk '/NPU ID/{print $NF}')
npu_count=$(echo "$npu_ids" | wc -l)
main
echo "Check $LOG_FILE for details."
```

Note: If the driver and firmware versions are different from the example or the version is changed later, modify the version settings in the script configuration.

```
# ***制作镜像时如果驱动版本有变化，需要更新这里***
PRESET_SOFTWARE_VERSION="25.2.1"
PRESET_FIRMWARE_VERSION="7.7.0.9.220"
```

8. Add the execute permission for the firmware package and configuration script:

```
cd /opt/huawei/firmware_check
chmod 700 firmware_check.sh
chmod 700 Ascend-hdk-npu-firmware_7.7.0.9.220.run
```

9. Add the automatic startup service for driver firmware consistency:

```
cd /etc/systemd/system
vim firmware_check.service
```

Add the following content:

```
[Unit]
Description=check and upgrade firmware
After=config-hash.service
Requires=config-hash.service
[Service]
Type=oneshot
ExecStart=/opt/huawei/firmware_check/firmware_check.sh
RemainAfterExit=yes
User=root
[Install]
WantedBy=cloud-init.target
```

Configure the service to automatically start upon system startup.

```
systemctl daemon-reload
systemctl enable firmware_check.service
```

Configuring UDP Port Hash

1. Create a UDP port hash configuration directory:

```
mkdir -p /opt/huawei/port_config
```

2. Add the hash configuration script:

```
touch port_config.json
touch uplink_hash_config.py
```

3. Write the hash configuration script:

```
vim uplink_hash_config.py
```

Add the following content:

```
# -*- coding: UTF-8 -*-
import base64
import logging
import time
import json
import shutil
import os
import requests
```



```
try:
    import commands
except ImportError:
    import subprocess as commands
A2_NPU_COUNT = 8
A3_NPU_COUNT = 16
METADATA_URL = 'http://169.254.169.254/openstack/latest/meta_data.json'
A3_32K_CLUSTER = "cluster4"
A3_64K_CLUSTER = "cluster5"
A3_128K_CLUSTER = "cluster6"
A3_9866_CLUSTER = "cluster8"
A3_9866_REGION_ID = "cn-guian02"
log = logging.getLogger(__name__)
class UplinkHashConfig:
    def __init__(self, config_dir="/opt/huawei/port_config/",
                 log_file="/opt/huawei/port_config/uplink_hash_config.log"):
        self.DIR = config_dir
        self.setup_log(log_file)
    def setup_log(self, log_file):
        handler = logging.FileHandler(log_file)
        formatter = logging.Formatter('%(asctime)s - %(filename)s[%(levelname)s]: %(message)s')
        handler.setFormatter(formatter)
        log.addHandler(handler)
        log.setLevel(logging.INFO)
    def read_url(self, url, timeout=None, retries=0, sec_between=1, check_status=True):
        manual_tries = 1
        if retries:
            manual_tries = max(int(retries) + 1, 1)
        if sec_between is None:
            sec_between = -1
        for i in range(0, manual_tries):
            try:
                r = requests.get(url, timeout=timeout)
                if check_status:
                    r.raise_for_status()
                return r
            except requests.exceptions.RequestException as e:
                if i + 1 < manual_tries and sec_between > 0:
                    log.warning("Get metadata failed, wait %s seconds to try again", sec_between)
                    time.sleep(sec_between)
        log.error("Get metadata failed, please check network and security group.")
        return None
    def get_meta_json(self, timeout=5, retries=5):
        try:
            resp = self.read_url(url=METADATA_URL,
                timeout=timeout,
                retries=retries)
            metadata = resp.json()
        except Exception as e:
            log.error(
                "Get metadata failed. Error: %s", e)
            return False, None
        return True, metadata
    def get_npu_count(self, meta_json):
        hyperinstance_type = meta_json.get("meta", {}).get("_sys_hyperinstance_type")
        if hyperinstance_type:
            log.info("Node type is hyperinstance.")
            return A3_NPU_COUNT
        return A2_NPU_COUNT
    def get_config_url(self, region):
        if region == "cn-north-7":
            url = "https://cnnorth7-modelarts-sdk.obs.cn-north-7.ulanqab.huawei.com"
        elif region == "cn-north-9":
            url = "https://cnnorth9-modelarts-sdk.obs.cn-north-9.myhuaweicloud.com"
        elif region == "cn-south-1":
            url = "https://cnsouth1-modelarts-sdk.obs.cn-south-1.myhuaweicloud.com"
        elif region == "cn-east-3":
            url = "https://cneast3-modelarts-sdk.obs.cn-east-3.myhuaweicloud.com"
        elif region == "ap-southeast-1":
```

```
            url = "https://ap-southeast1-modelarts-sdk.obs.ap-southeast-1.myhuaweicloud.com"
        elif region == "cn-north-11":
            url = "https://cnnorth11-modelarts-sdk.obs.cn-north-11.myhuaweicloud.com"
        elif region == "cn-southwest-2":
            url = "https://cn-southwest-2-modelarts-sdk.obs.cn-southwest-2.myhuaweicloud.com"
        elif region == "cn-east-4":
            url = "https://cneast4-modelarts-sdk.obs.dualstack.cn-east-4.myhuaweicloud.com"
        elif region == "la-south-2":
            url = "https://la-south2-modelarts-sdk.obs.la-south-2.myhuaweicloud.com"
        elif region == "ap-southeast-3":
            url = "https://ap-southeast3-modelarts-sdk.obs.ap-southeast-3.myhuaweicloud.com"
        elif region == "me-east-1":
            url = "https://me-east-1-modelarts-sdk.obs.me-east-1.myhuaweicloud.com"
        else:
            url = "https://{0}-modelarts-sdk.obs.{1}.myhuaweicloud.com".format(region.replace("-", ""),
region)
        return url + "/devserver/port_config.json"
    def download_file(self, url, destination):
        log.info("Downloaded file from %s to %s", url, destination)
        try:
            response = requests.get(url, timeout=10)
            response.raise_for_status()
            with open(destination, "wb") as f:
                f.write(response.content)
            return True
        except requests.exceptions.RequestException as e:
            log.error("Failed to download file from %s: %s", url, str(e))
            return False
    def get_port_config(self, url):
        config_file = os.path.join(self.DIR, "port_config.json")
        backup_file = os.path.join(self.DIR, "port_config.json.bak")
        if url is None:
            log.warning("Url is none, use local config file.")
            try:
                with open(config_file, "r") as f:
                    hash_config = json.load(f)
                log.info("Loaded config from %s", config_file)
                return hash_config
            except FileNotFoundError:
                log.error("Config file %s not found.", config_file)
                return None
            except json.JSONDecodeError:
                log.error("Failed to decode JSON from %s", config_file)
                return None
        try:
            shutil.copy2(config_file, backup_file)
            log.info("Backed up %s to %s", config_file, backup_file)
        except Exception as e:
            log.error("Failed to backup %s: %s", config_file, str(e))
            return None
        if self.download_file(url, config_file):
            try:
                with open(config_file, "r") as f:
                    hash_config = json.load(f)
                log.info("Loaded new config from %s", config_file)
                os.remove(backup_file)
                return hash_config
            except json.JSONDecodeError:
                log.error("Failed to decode json from downloaded file %s, restored backup file.",
config_file)
                shutil.copy2(backup_file, config_file)
                os.remove(backup_file)
                with open(config_file, "r") as f:
                    hash_config = json.load(f)
                return hash_config
        else:
            log.warning("Download failed, restored local backup config file.")
            shutil.copy2(backup_file, config_file)
            os.remove(backup_file)
```

```
        with open(config_file, "r") as f:
            hash_config = json.load(f)
        return hash_config
    def wait_until_npu_ready(self, npu_count):
        for _ in range(0, 10):
            count = 0
            for i in range(0, npu_count):
                cmd = "hccn_tool -i {} -lldp -g | grep Ifname | awk -F ': ' '{{print $2}}'".format(i)
                (status, output) = commands.getstatusoutput(cmd)
                if not output:
                    log.error("result: get ifname failed, try again after 30s, id:%s", i)
                    time.sleep(30)
                    break
                count += 1
            if count == npu_count:
                log.info("result: get all ifname success.")
                return
        log.warning("Failed to get ifname after 10 attempts.")
    def print_current_port(self, npu_count):
        for i in range(0, npu_count):
            cmd = "hccn_tool -i {} -udp -g".format(i)
            (status, output) = commands.getstatusoutput(cmd)
            log.info("port %s: %s", i, output)
    def config_udp_port_auto(self, npu_count):
        log.info("Configuring ports in auto mode.")
        for i in range(0, npu_count):
            cmd = "hccn_tool -i {} -udp -s auto".format(i)
            (status, _) = commands.getstatusoutput(cmd)
    def config_udp_port(self, port_config, flavor, npu_count, region_id):
        if port_config is None:
            self.config_udp_port_auto(npu_count)
            return
        cmd = "echo -n {} | sha256sum | awk '{{print $1}}'".format(flavor)
        (status, sha256_flavor) = commands.getstatusoutput(cmd)
        log.info("Sha256 flavor is %s", sha256_flavor)
        cur_cluster = self.get_cluster(port_config, sha256_flavor, region_id)
        if not cur_cluster:
            if A3_NPU_COUNT == npu_count:
                log.warning("Get cluster from port_config failed, flavor is %s, set default value %s", flavor,
A3_32K_CLUSTER)
                cur_cluster = A3_32K_CLUSTER
            else:
                log.error("Get cluster from port_config failed, flavor is %s", flavor)
                self.config_udp_port_auto(npu_count)
                return
        log.info("Cluster is %s", cur_cluster)
        desire_port_map = self.get_desire_port_map(cur_cluster, port_config)
        for i in range(0, npu_count):
            cmd = "hccn_tool -i {} -lldp -g | grep Ifname | awk -F ': ' '{{print $2}}'".format(i)
            (status, ifname_tmp) = commands.getstatusoutput(cmd)
            key = "{}|{}".format(i, ifname_tmp)
            desire_port = desire_port_map.get(key, None)
            if not desire_port:
                log.error("Get desire port from port_config failed, id %s, ifname %s", i, ifname_tmp)
                self.config_udp_port_auto(npu_count)
                return
            cmd = "hccn_tool -i {} -udp -g | awk -F ':' '{{print $2}}' | grep -v auto".format(i)
            (status, current_port) = commands.getstatusoutput(cmd)
            if desire_port != current_port:
                cmd = "hccn_tool -i {} -udp -s port {}".format(i, desire_port)
                (status, _) = commands.getstatusoutput(cmd)
    def get_cluster(self, port_config, sha256_flavor, region_id):
        if region_id == A3_9866_REGION_ID:
            return A3_9866_CLUSTER
        return port_config.get("flavors", {}).get(sha256_flavor, None)
    def get_desire_port_map(self, cur_cluster, port_config):
        if cur_cluster == A3_32K_CLUSTER:
            return self.get_a3_desire_port_map(5120, 5183, 32)
        elif cur_cluster == A3_64K_CLUSTER:
```



```
            return self.get_a3_desire_port_map(5184, 5327, 36)
        elif cur_cluster == A3_128K_CLUSTER:
            return self.get_a3_desire_port_map(5184, 5471, 36)
        else:
            return port_config.get("clusters", {}).get(cur_cluster, {})
    def get_a3_desire_port_map(self, start, end, max_tor_port):
        udp_port_map = {}
        tor_ge_id = 1
        tor_port = 0
        udp_port = start
        while udp_port <= end:
            udp_port_map["6|400GE{0}/0/{1}:1".format(tor_ge_id, tor_port)] = udp_port
            udp_port_map["14|400GE{0}/0/{1}:2".format(tor_ge_id, tor_port)] = udp_port
            udp_port_map["4|400GE{0}/0/{1}:1".format(tor_ge_id, tor_port + 1)] = udp_port + 1
            udp_port_map["12|400GE{0}/0/{1}:2".format(tor_ge_id, tor_port + 1)] = udp_port + 1
            udp_port_map["2|400GE{0}/0/{1}:1".format(tor_ge_id, tor_port + 2)] = udp_port + 2
            udp_port_map["10|400GE{0}/0/{1}:2".format(tor_ge_id, tor_port + 2)] = udp_port + 2
            udp_port_map["0|400GE{0}/0/{1}:1".format(tor_ge_id, tor_port + 3)] = udp_port + 3
            udp_port_map["8|400GE{0}/0/{1}:2".format(tor_ge_id, tor_port + 3)] = udp_port + 3
            udp_port_map["7|400GE{0}/0/{1}:1".format(tor_ge_id, tor_port)] = udp_port
            udp_port_map["15|400GE{0}/0/{1}:2".format(tor_ge_id, tor_port)] = udp_port
            udp_port_map["5|400GE{0}/0/{1}:1".format(tor_ge_id, tor_port + 1)] = udp_port + 1
            udp_port_map["13|400GE{0}/0/{1}:2".format(tor_ge_id, tor_port + 1)] = udp_port + 1
            udp_port_map["3|400GE{0}/0/{1}:1".format(tor_ge_id, tor_port + 2)] = udp_port + 2
            udp_port_map["11|400GE{0}/0/{1}:2".format(tor_ge_id, tor_port + 2)] = udp_port + 2
            udp_port_map["1|400GE{0}/0/{1}:1".format(tor_ge_id, tor_port + 3)] = udp_port + 3
            udp_port_map["9|400GE{0}/0/{1}:2".format(tor_ge_id, tor_port + 3)] = udp_port + 3
            udp_port += 4
            tor_port += 4
            if tor_port >= max_tor_port:
                tor_port = 0
                tor_ge_id += 1
        return udp_port_map
    def run(self):
        ret, meta_json = self.get_meta_json()
        if not ret:
            log.error("Get meta_json failed, metadata json is %s", meta_json)
            return
        log.info("Get meta_json success, metadata json is %s", meta_json)
        region_id = meta_json.get('region_id', None)
        if not region_id:
            log.error("Get region_id from metadata failed.")
            return
        log.info("Region is %s", region_id)
        flavor = meta_json.get('instance_type', None)
        if not flavor:
            log.error("Get instance_type from metadata failed.")
            return
        log.info("Flavor is %s", flavor)
        npu_count = self.get_npu_count(meta_json)
        log.info("Npu count is %s", npu_count)
        url = self.get_config_url(region_id)
        port_config = self.get_port_config(url)
        self.wait_until_npu_ready(npu_count)
        log.info("Before config uplink udp hash, Port is:")
        self.print_current_port(npu_count)
        log.info("=====================================================")
        self.config_udp_port(port_config, flavor, npu_count, region_id)
        log.info("Config uplink udp hash done. Port is:")
        self.print_current_port(npu_count)
if __name__ == "__main__":
    configurator = UplinkHashConfig()
    configurator.run()
```

4. Add the script execution permission:

```
cd /opt/huawei/port_config
chmod +x uplink_hash_config.py
```

5. Obtain the latest configuration file of the current site:

```
python uplink_hash_config.py
```

6. Add the automatic startup service for hash configuration:

```
cd /etc/systemd/system
vim config-hash.service
```

Add the following content:

```
[Unit]
Description=Run uplink hash config
After=bms-network-config.service
Requires=bms-network-config.service
[Service]
Type=oneshot
ExecStart=python3 /opt/huawei/port_config/uplink_hash_config.py
RemainAfterExit=yes
User=root
[Install]
WantedBy=cloud-init.target
```

Configure the service to automatically start upon system startup.

```
cd /etc/systemd/system
systemctl daemon-reload
systemctl enable config-hash.service
```

Verification

To verify whether driver firmware consistency and port hash are configured, **restart the server**. For details, see **Restarting a Lite Server**.

Note: The server restarts automatically for the firmware upgrade to take effect.

1. Verify the driver firmware version.

   View the NPU driver and firmware versions:

   ```
   ascend-dmi -c
   ```

   If the firmware version is the same as configured, the settings are successful.

2. Verify the UDP port hash configuration.

   View the uplink port on the parameter plane network:

   ```
   npu_ids=$(npu-smi info -l | awk '/NPU ID/{print $NF}')
   for i in $npu_ids; do hccn_tool -i $i -udp -g;done
   ```

   In the command output, if **udp_port** is not **Unknown** and **status** is **custom**, the configuration is successful.

Clearing and Saving the Image

1. Clear the logs about driver firmware consistency and UDP port hash.

```
rm -rf /opt/huawei/port_config/uplink_hash_config.log
rm -rf /opt/huawei/firmware_check/firmware_check.log
rm -rf /opt/huawei/firmware_check/time*
```

2. Clear the traces and create a new image. For details, see **Creating the OS of a Lite Server**.

6 Using Lite Server Resources

6.1 NPU Training and Inference Guide for LLM and AIGC Models

This section describes how to use NPUs for training and inference, including LLM and AIGC image and video generation. You can view the detailed guide.

LLM

- **Adapting Mainstream Open-Source Models to Ascend-VLLM PyTorch NPU Inference Based on Lite Server**

AIGC Model

- **AIGC Image Generation Model Training and Inference**
- **AIGC Image Generation Model Training and Inference**

6.2 Adapting GPT-2 to GPU-based Training and Inference Based on Lite Servers

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

  Megatron-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

  Generative Pre-trained Transformer 2 (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 using Megatron-DeepSpeed. The following images and specifications are used as example:

- Image: Ubuntu 20.04 x86 64bit SDI3 for Ant8 BareMetal with RoCE and NV-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 a GPU-powered BMS as user **root** in SSH mode. For details, see **Logging In to a BMS Using an SSH Key Pair**.
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 **git clone** failed, download the framework to a local path, copy it to the server, and run **docker cp** 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
```

2. Decompress the dataset.

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

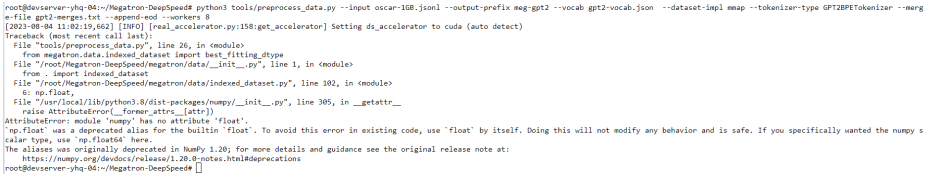
3. Preprocess data.

```
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

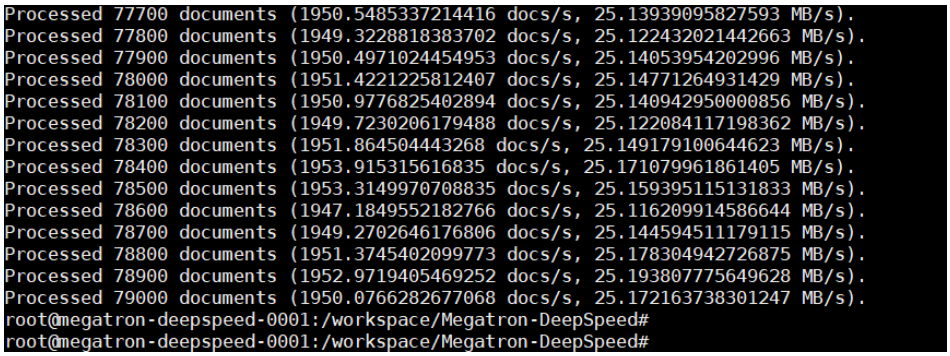
If the error message "np.float" is displayed, change it to "float" as prompted.

Figure 6-1 Data preprocessing error



4. Complete the preprocessing.

Figure 6-2 Data preprocessed



5. 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.

1. Create the script.

```
vim pretrain_gpt2.sh
```

2. 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 6-3 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 6-4 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.

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 6-5 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 following parameter in the pre-training script:

```
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
```

----End

Step4 Using the GPT-2 Model to Generate Text

Step 1 Generate text automatically.

1. Create a text generation script.

```
vim generate_text.sh
```

Add the following content:

```
#!/bin/bash

CHECKPOINT_PATH=checkpoints/gpt2
VOCAB_FILE=data/gpt2-vocab.json
MERGE_FILE=data/gpt2-merges.txt

python tools/generate_samples_gpt.py \
       --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 2 \
       --top_p 0.9 \
       --recompute
```

2. Generate text.

```
sh ./generate_text.sh
```

If the following information is displayed, the text is generated.

Figure 6-6 Text generated

```
Loading extension module fused_mix_prec_layer_norm_cuda...
>>> done with compiling and loading fused kernels. Compilation time: 1.622 seconds
building GPT model ...
 loading checkpoint from checkpoints/gpt2 at iteration 5000
 checkpoint version 3.0
  successfully loaded checkpoint from checkpoints/gpt2 at iteration 5000
Avg s/batch: 20.582671880722046
root@megatron-deepspeed-0001:/workspace/Megatron-DeepSpeed#
root@megatron-deepspeed-0001:/workspace/Megatron-DeepSpeed#
```

3. View the generated text.

```
cat unconditional_samples.json
```

The following figure shows the output.

Figure 6-7 File information

Step 2 Enable interactive dialog.

1. Create a text generation script.
   vim interactive_text.sh

   Enter the following information:

   ```
   #!/bin/bash

   CHECKPOINT_PATH=/workspace/Megatron-DeepSpeed/checkpoints/gpt2_345m
   VOCAB_FILE=/workspace/Megatron-DeepSpeed/data/gpt2-vocab.json
   MERGE_FILE=/workspace/Megatron-DeepSpeed/data/gpt2-merges.txt

   deepspeed /workspace/Megatron-DeepSpeed/tools/generate_samples_gpt.py \
       --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
   ```

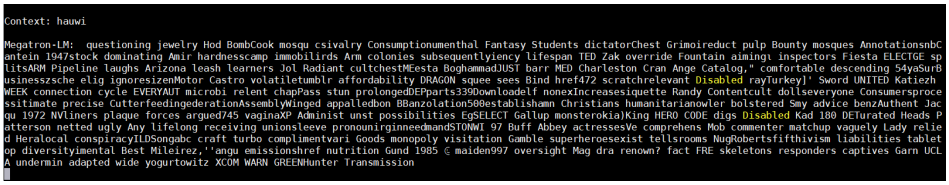
2. Start interactive dialog.
   bash interactive_text.sh

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

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

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

Figure 6-8 Model output



----End

7 Managing Lite Server Resources

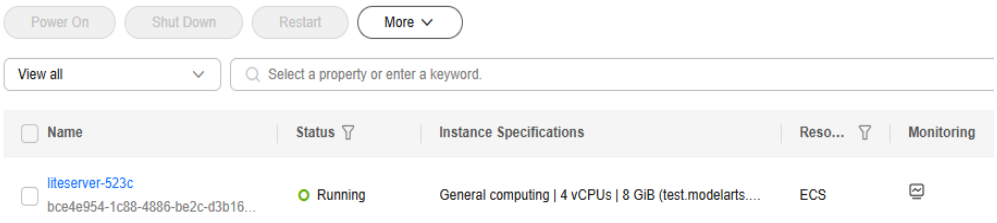
7.1 Viewing Details About a Lite Server

After you create a Lite Server, you can view and manage it on the management console. This section describes how to view Lite Server node details, including name, ID, specifications, and image.

Viewing Details About a Lite Server Common Node

In the common node list of Lite Servers, you can view the node details, including the node name, status, instance specifications, resource type, monitoring, RoCE ID, VPC, IP address, billing mode, creation time, and operation.

Figure 7-1 Viewing details about a Lite Server common node



Click a common node name to view its details. **Table 7-1** lists the parameters.

Table 7-1 Parameters on the details page

Parameter	Description
Name	Name of the Lite Server common node. You can modify it by referring to **Changing the Name of a Lite Server**.
Specifications	Specifications of the Lite Server common node.

Parameter	Description
ID	ID of the Lite Server common node, which can be queried in the Billing Center.
Billing Mode	Billing mode of the Lite Server common node.
Status	Status of the Lite Server common node.
RoCE ID	RoCE network ID of the Lite Server common node. In multi-node multi-PU training scenarios, the nodes must be on the same RoCE network. To view the nodes on the same RoCE network, copy the RoCE ID on the Lite Server details page, and search for the nodes on the Lite Server common node list page.
VPC	VPC bound to the Lite Server common node during creation. You can click the link to go to the VPC details page.
BMS/ECS	The Lite Server common node is a BMS or ECS. You can click the link to access the corresponding BMS or ECS details page.
Image	OS image of the Lite Server common node. You can switch or reset the OS image by referring to **Changing or Resetting the OS of a Lite Server**.
Created	Time when the Lite Server common node was created.
Updated At	Time when the Lite Server common node was updated.
Order	Order corresponding to the Lite Server common node. You can click the link to go to the Billing Center.
Access Key	AK name.
Disk	Disks attached to the Lite Server common node. To attach a disk, click **Mount Disk**. For details, see **Configuring the Lite Server Storage**.
Monitoring	Monitoring information about the Lite Server common node. To view the monitoring details, click **Monitoring Details** to go to the Cloud Eye console.

Viewing Details About a Lite Server Supernode

In the supernode list of Lite Servers, you can view the supernode details, including the name, status, creation time, billing mode, instance specifications, core hardware configuration, VPC, IP address, and operations.

Click a supernode name to access its details page. **Table 7-2** lists the parameters.

Table 7-2 Parameters on the details page

Parameter	Description
Name	Lite Server supernode name.
Specifications	Lite Server specifications.
ID	ID of the Lite Server supernode, which can be queried in the Billing Center.
Billing Mode	Billing mode of the Lite Server supernode.
Status	Status of the Lite Server supernode.
VPC	VPC bound to the Lite Server supernode when the Lite Server supernode is created. You can click the link to go to the VPC details page.
Image	OS image of the Lite Server supernode. You can switch or reset the OS image by referring to **Changing or Resetting the OS of a Lite Server**.
Created	Time when the Lite Server supernode was created.
Updated At	Time when the Lite Server supernode was updated.
Order	Order corresponding to the Lite Server supernode. You can click the link to go to the Billing Center.
Access Key	AK name.

7.2 Starting or Stopping a Lite Server

Description

You can power off a running Lite Server if it is no longer needed, avoiding resource consumption. To use a stopped Lite Server, power it on.

Constraints

- To power on a Lite Server, the Lite Server status must be **Stopped** or **Start Failed**.
- Only Lite Servers in the **Running** or **Stop Failed** state can be powered off.
- Common nodes and child nodes of supernodes can be batch powered on or off. However, batch operations are not supported for supernodes.

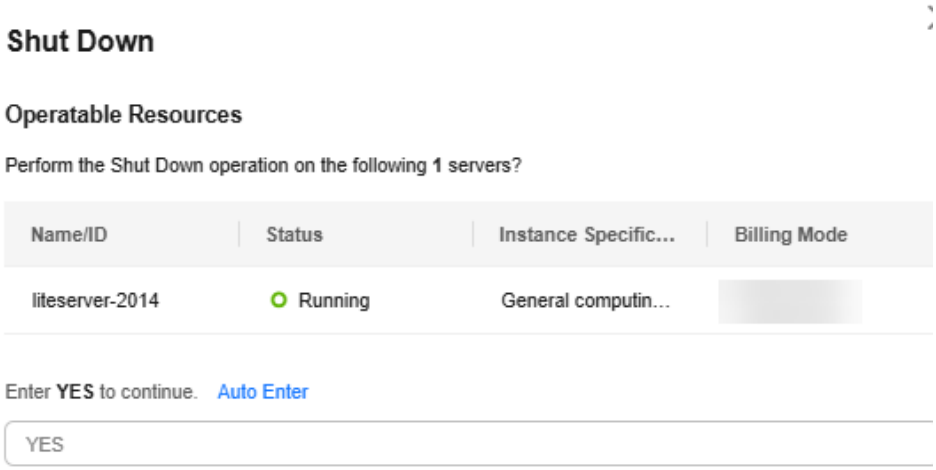
Powering On or Off a Lite Server

1. Log in to the **ModelArts console**. In the navigation pane, choose **Resource Management** > **Lite Servers**.
   - New console: In the navigation pane, choose **Resource Management** > **Lite Compute Resources** > **Lite Servers**.

- Old console: In the navigation pane, choose **Resource Management** > **Lite Servers**.

2. Power on or off one or multiple Lite Server common nodes at once.
   - To power on a Lite Server, locate it in the list and click **Power On** in the **Operation** column. The Lite Server must be in the **Stopped**, **Stop Failed**, or **Start Failed**.
   - To power off a Lite Server, locate it in the list and click **Shut Down** in the **Operation** column. In the displayed dialog box, confirm the information, enter **Yes**, and click **OK**. Only Lite Servers in the **Running** or **Stop Failed** state can be powered off.

     Figure 7-2 Shutdown

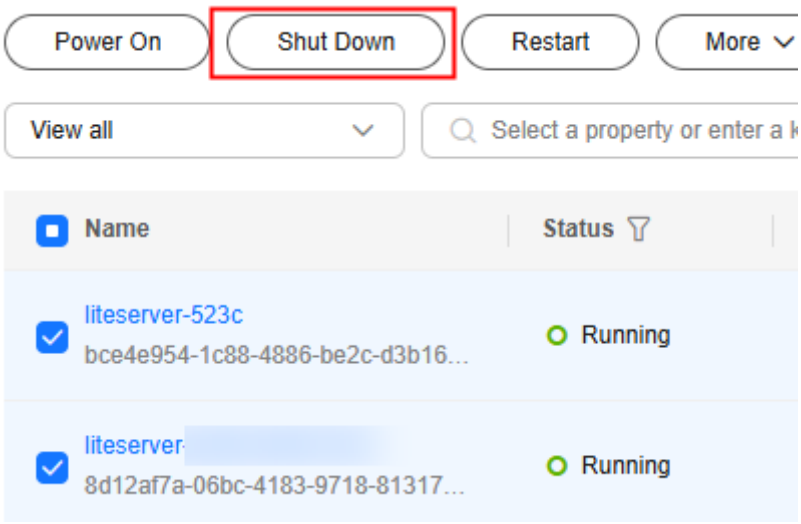
     NOTE

     Forcibly shutting down a server will interrupt your services. Ensure that the files on the server have been saved.

     In the common node list, select multiple nodes to power on or off them in batches.

     In the supernode list, select multiple child nodes to batch power on or off them.

Figure 7-3 Batch operation



7.3 Synchronizing the Status of a Lite Server

Description

A Lite Server common node is an ECS or BMS. After modifying the server status or disk information on the ECS or BMS, you can click **Synchronize** to synchronize the server status and disk information to ModelArts Lite Server.

The child node of a Lite Server supernode is an ECS. If you modify the disk information of the child node on ECS, you can click **Synchronize** to synchronize the disk information to ModelArts Lite Server.

Constraints

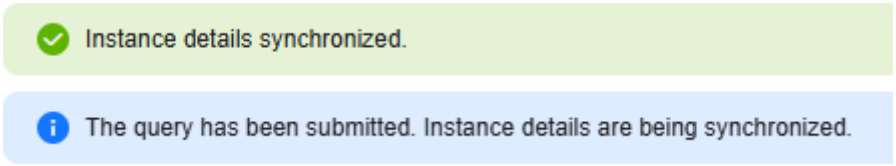
- The server status and disk information can be synchronized for common nodes (ECSs or BMSs).
- For a supernode, the parent node server status cannot be synchronized.
- For a supernode, only the disk information of its child nodes can be synchronized.
- For child nodes of a supernode, modifying the server status (such as restarting or powering off) via the ECS console is not supported.

Procedure

1. Log in to the **ModelArts console**. In the navigation pane, choose **Resource Management** > **Lite Servers**.
   - New console: In the navigation pane, choose **Resource Management** > **Lite Compute Resources** > **Lite Servers**.
   - Old console: In the navigation pane, choose **Resource Management** > **Lite Servers**.
2. In the resource list, locate the target common node or supernode and choose **More** > **Synchronize** in the **Operation** column. A message will be displayed,

indicating that the query task has been delivered and the instance details are being synchronized. Alternatively, go to the Lite Server details page, and click **Synchronize**.

If "Instance details synchronized" is displayed, the synchronization is complete.



7.4 Changing or Resetting the OS of a Lite Server

Description

If the OS image of the Lite Server does not meet requirements, you can change or reset the OS. Change the OS in any of the following ways:

- (Recommended) Change or reset the OS on the Lite Server page of the ModelArts console.
- Change the OS on the BMS console.
- Change the BMS OS using the BMS Go SDK.
- Change the BMS OS by encapsulating APIs using Python.

Constraints

- Node status: The Lite Server must be in the **Stopped**, **Resetting OS failed**, or **Changing OS failed**. Otherwise, the operation may fail as the system cannot be uninstalled and the disk will be uninstalled repeatedly.
- OS: The target OS must be an IMS public image or private shared image in the region.
- Common nodes and child nodes of supernodes support both changing the OS and resetting the OS. Batch operations are also supported.
- Supernodes allow you to change the OS, but you cannot reset it. Supernodes do not support batch OS change.

Impact

There will be the following impacts when you reset or change the Lite Server OS:

1. System disk ID change: The EVS system disk ID will be changed, which is different from that in the order. As a result, the EVS system disk capacity cannot be scaled out. When you scale out the system disk, a message is displayed, indicating that the current order has expired, the capacity cannot be expanded, and the order needs to be renewed.
2. userdata configuration: When you change the OS, userdata injection may not take effect, especially in configdriver mode. Ensure that the **userdata** parameter is imported during node creation, or manually configure necessary settings after the change. Therefore, attach an EVS disk or an SFS disk to expand the storage capacity after changing or resetting the OS.

3. Application and model: The deployed model or application may be impacted after you change the OS, as the dependent software package or library may need to be reinstalled or configured. Reconfigure necessary dependencies to ensure proper running of applications.
4. BMS risks: For BMS, upgrading OS kernel or driver may cause incompatibility, affecting system startup or basic functions. To perform an upgrade, contact the cloud service provider.

Before changing or resetting an OS, you need to ensure that the node is shut down, check the current settings, back up important data, and contact technical support if necessary.

Changing or Resetting the OS on the Server Page of the ModelArts Console

1. Log in to the **ModelArts console**. In the navigation pane, choose **Resource Management** > **Lite Servers**.
   - New console: In the navigation pane, choose **Resource Management** > **Lite Compute Resources** > **Lite Servers**.
   - Old console: In the navigation pane, choose **Resource Management** > **Lite Servers**.
2. In the resource list, locate the target common node or supernode and choose **More** > **Change OS** or **More** > **Reset OS** in the **Operation** column. In the displayed dialog box, confirm the information and click **OK**.

   The Lite Server common node or supernode is in the **Changing OS** or **Resetting OS** state.

   To change or reset the OS in batches, select multiple nodes in the list, and choose **More** > **Change OS** or **Reset OS** above the list.

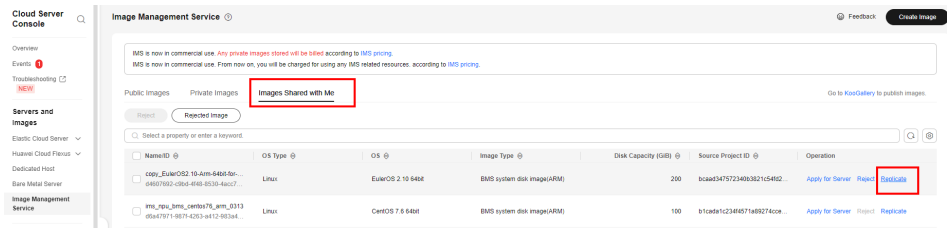
   To change or reset the OS for sub-nodes in batches, select multiple nodes in the supernode list, and choose **More** > **Change OS** or **Reset OS** above the list.

Changing the OS on the BMS Console

1. Obtain the OS image.

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

   Figure 7-4 Shared image

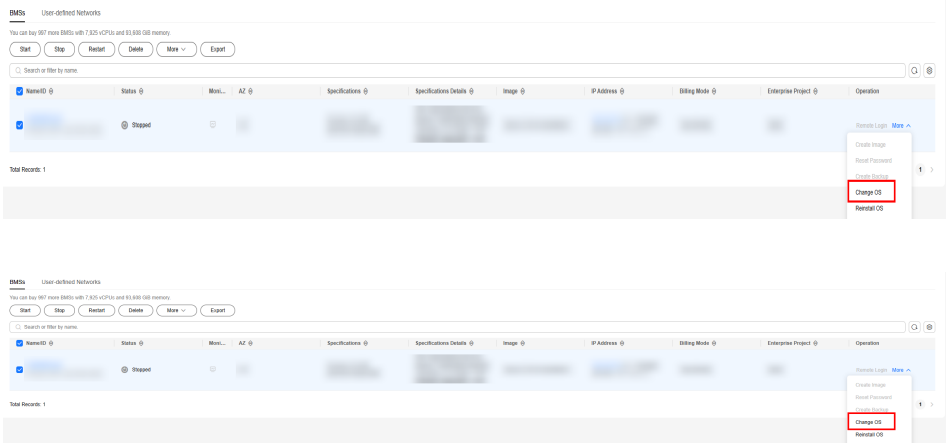


2. Change OS.

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

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

Figure 7-5 Changing OS



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

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 := "aGVsbG8gd29ybGQsIHdlbGNvbWUgdG8gam9pbiB0aGUgY29uZmVyZW5jZQ=="
    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 BMS 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")
```

7.5 Managing Lite Server Hot Standby Nodes

Description

To enable Lite Server hot standby, create custom K8s clusters. These clusters mark specific nodes as hot standbys by applying taints, so that service pods will not be scheduled to these nodes.

Constraints

Prepare required hot standby nodes by referring to the following table.

Table 7-3 Required hot standby nodes

Resource Type/ Required Nodes	< 10	10–49	50–99	100–249	250–499	500–749	750–1,000	> 1,000
Snt9a	0	1	2	3	5	7	10	12
Snt9b	0	1	2	3	4	5	6	10

Resource Type/ Required Nodes	< 10	10–49	50–99	100–249	250–499	500–749	750–1,000	> 1,000
GP Ant8	0	1	2	3	5	6	8	12
GP Vnt1	0	1	2	3	5	8	10	12

Example 1:

If you have purchased six Snt9b nodes, which is less than 10. According to the preceding table, you do not need to prepare hot standby nodes.

Example 2:

If you have purchased 600 Snt9b nodes, which is between 500 and 749. According to the preceding table, you need to prepare five more nodes as hot standbys. In this case, you need to purchase 605 nodes in total.

Prerequisites

You have created a K8s cluster using Lite Server resources.

Hot Standby Replacement

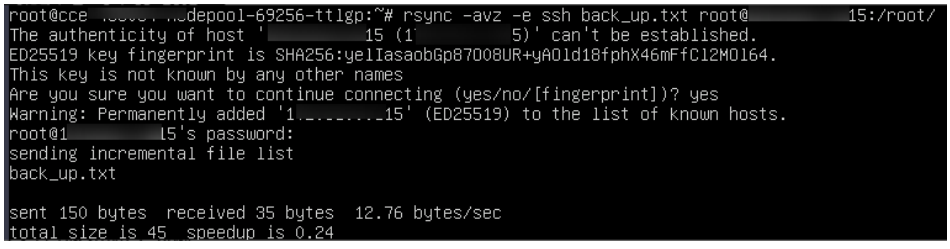
If a hardware fault occurs on a service node in the cluster and hot standby replacement is required, back up data first, and then taint the faulty node and remove the hot standby node's taint.

1. **Data backup**

   Use rsync, a powerful file synchronization tool that supports local and remote synchronization, to flexibly and efficiently back up data.

   ```
   rsync -avz -e ssh /source/ user@remote:/destination/
   ```

   Back up the faulty node file to the hot standby node. The **backup.txt** file is used as an example.

   Complete the backup using SSH. Then, you can view the file on the new hot standby node.

2. **Hot standby node replacement**

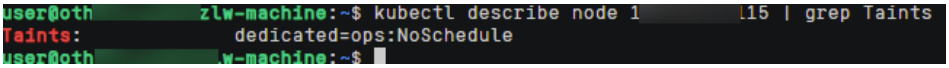
   Taint the faulty node.

   ```
   kubectl taint nodes <node-name> dedicated=ops:NoSchedule
   ```

   - **<node-name>**: Replace it with the actual node name.
   - **dedicated=ops**: key–value pair of the taint.
   - **NoSchedule**: taint effect. This indicates that kube-scheduler will not schedule pods to the node.

   Check whether the node is tainted.

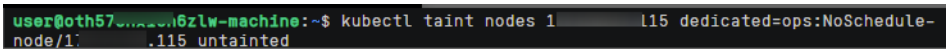
   ```
   kubectl describe node <node-name> | grep Taints
   ```

   The node is labeled as unschedulable.

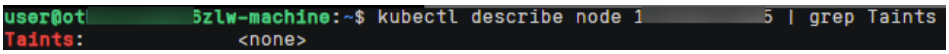
   Remove the taint from the hot standby node.

   ```
   kubectl taint node <node-name> dedicated=ops:NoSchedule-
   ```

   Ensure that the taint is removed.

   ```
   kubectl describe node <node-name> | grep Taints
   ```

   The hot standby node replacement is complete. Repair the faulty node.

7.6 Changing the Name of a Lite Server

Description

During routine cloud service management, you may need to name or rename ECSs to better manage and identify instances. To improve user experience, you can change the name of a Lite Server.

- You can change the names of common nodes in the **Common node** tab on the **Lite Servers** page or the node details page. Duplicate names are allowed.
- You can change the names of child nodes in the **Supernodes** tab on the **Lite Servers** page or the node details page. Duplicate names are allowed.

Constraints

- The server name in the order will not be changed. If you change the name after placing the order, the new name will not be synchronized to the order.
- When the resource type of a Lite Server is a BMS or an ECS, modifying the server name on the BMS or ECS console will not automatically sync to the Lite Server. You must perform a manual synchronization. For details, see **Synchronizing the Status of a Lite Server**.
- If a supernode is used to create a Lite Server, you can only change the name of its child nodes. The supernode name cannot be changed.
- You can modify the server name only when the Lite Server status is **Running** or **Stopped**.

- The Lite Server name cannot be left blank.
- If duplicate names are now allowed, and you enter an existing name, a message will be displayed, indicating that the cloud service has been used.

Changing the Name of a Lite Server

1. Log in to the **ModelArts console**. In the navigation pane, choose **Resource Management** > **Lite Servers**.
   - New console: In the navigation pane, choose **Resource Management** > **Lite Compute Resources** > **Lite Servers**.
   - Old console: In the navigation pane, choose **Resource Management** > **Lite Servers**.
2. In the resource list, locate the target common node or supernode and click on the right to chang the node name. After the name is changed, click **OK**. The change applies immediately.

Figure 7-6 Changing the node name

Alternatively, go to the node details page, click next to the node name, and click **OK**.

Figure 7-7 Changing the node name

When modifying the name, you can set the parameter to allow duplicate node names.

7.7 Authorizing Repair of a Lite Server

Description

When a Lite Server requires hardware maintenance due to an unrecoverable fault, a scheduled event will be pushed to the **Event Center** on the console. In the event center, you can view the event information, type, status, and description. You can also authorize Huawei technical support to perform O&M on the faulty node or redeploy the node.

Table 7-4 Event operation execution conditions

Event Type	Event Status	Supported Operations	Applicable Resource Type	Description
System maintenance	Authorization Pending	Authorization and redeployment	Snt9b	System maintenance is to authorize Huawei technical support to systematically maintain the faulty node.
Local disk recovery	Authorization Pending	Authorization and redeployment	Snt9b	Local disk recovery is to authorize Huawei technical support to maintain the faulty local disk. After the local disk is recovered, **reset the nodes** or log in to the nodes to restore the node partition. **WARNING** After authorization, recovering the local disk will cause local supernode disk loss. Therefore, migrate services and back up data before authorization.
Supernode maintenance	Authorization Pending	Authorization	Snt9b23	Supernode maintenance is to authorize Huawei technical support to recover faulty nodes by manually repairing or replacing components.

Event Type	Event Status	Supported Operations	Applicable Resource Type	Description
Supernode redeployment	Authorization Pending	Authorization	Snt9b23	Supernode redeployment is to authorize the Huawei O&M system to recover faulty nodes by automatically replacing nodes. After the recovery, the node name, node ID, and IP address remain unchanged except the physical device information.
Supernode local disk recovery	Authorization Pending	Authorization	Snt9b23	Supernode local disk recovery is to authorize Huawei technical support to restore the local disk of the supernode. After the local disk is recovered, **reset the nodes** or log in to the nodes to restore the node partition. **WARNING** After authorization, recovering the local disk will cause local supernode disk loss. Therefore, migrate services and back up data before authorization.

- Authorization: Authorize Huawei technical support to repair the hardware of the faulty node one by one, which takes a long time.
- Redeployment: Authorize Huawei technical support to replace the faulty node with a new one, which is fast, but local disk data will be lost after the redeployment. Exercise caution. Moreover, migrate services and back up data before redeployment.

Constraints

- Only Snt9b and Snt9b23 supernodes support hardware maintenance through scheduled events.
- Redeployment of supernodes must be performed within physical supernodes. If there are 48 supernodes, redeployment is not supported and the authorization button becomes unavailable.
- If the planned event does not meet the requirements listed in **Table 7-4**, the **Authorize** button becomes unavailable.
- Before authorizing a supernode redeployment event, you need to stop the Lite Server instance on the **Lite Servers** page. Otherwise, the authorization fails. After the event is executed, restart the server instance.

- Authorizing a node will affect services running on it. The authorization operation can be performed only when the event type is **Supernode Redeployment** and the node is shut down.
- After the local node disk and supernode disk are restored, the local disk data will be lost. Therefore, migrate services and backup data before authorization. After the local disk is restored, log in to the Lite Server to partition the local disk.

Viewing Scheduled Events

Log in to the **ModelArts console**. In the navigation pane on the left, choose **Resource Management** > **Auxiliary Tools** > **Event Center** (old console: **Resource Management** > **Event Center**). On the **Event Center** page, you can view event details. By default, events in the **Authorization Pending**, **Authorized**, and **Executing** states are displayed. You can remove the filter criteria to view events in all states.

Table 7-5 Scheduled event description

Attribute	Description	Example
Event ID	Unique event ID.	5ad1df12-e3d2-4f36-b367-xxxxxxxxxxxx
Node Name/ID	Name and ID of the server node that initiates the event.	devserver-dd50 1e0d95ad-5a9f-46e3-9ba6-c5f8fcxxxx
Event Type	For details about the event types, see **Table 7-4**.	Supernode Redeployment
Event Status	• **Authorization Pending**: Querying. The event is to be authorized. After authorization, the status changes to **Authorized**. • **Authorized**: The O&M task is planned to be executed but has not started. After the task starts, the task enters the **Executing** state. • **Executing**: The O&M task is being executed. • **Completed**: The O&M task has been executed. • **Failed**: The O&M task fails to be executed. • **Canceled**: The system cancels the O&M task.	Authorization Pending

Attribute	Description	Example
Event Description	Cause of the event.	Underlying hardware fault. alarmName=XXXX,bmcip=2409:27ff:1003:0103:0011:0000:0000:xxxx,componentName=XXXX is automatically connected through CAR.
Obtained At	Event creation time	2025/02/19 16:05:32 GMT +08:00
Executed	Time when an event enters the scheduling and execution phase	2025/03/03 16:23:16 GMT +08:00
Operation	**Authorize**: Authorizing a node will affect services running on it. The authorization operation can be performed only when the event type is **Supernode Redeployment** and the node is shut down. **NOTE** Redeployment of supernodes must be performed within physical supernodes. If there are 48 supernodes, redeployment is not supported and the authorization button becomes unavailable.	--

Authorization Operations

If the faulty nodes meet the requirements listed in **Table 7-4**, you can authorize Huawei technical support to perform O&M on the faulty nodes.

To do so, log in to the ModelArts console. In the navigation pane on the left, choose **Resource Management** > **Auxiliary Tools** > **Event Center** (old console: **Resource Management** > **Event Center**). Locate the target node and click Authorize in the Operation column. In the displayed dialog box, click OK. The following steps describe how to authorize Huawei technical support to perform O&M on a supernode.

1. Log in to the **ModelArts console**. In the navigation pane on the left, choose **Resource Management** > **Auxiliary Tools** > **Event Center** (old console: **Resource Management** > **Event Center**). On the displayed page, view the supernode maintenance event and perform authorization.
2. The supernode maintenance event enters the **Authorized** state.
3. After the supernode is repaired, the event status is **Completed**. In this case, the node is available.

After the O&M, Huawei technical support will disable the authorization. You do not need to perform further operations.

For local disk or supernode disk restoration, you need to log in to the Lite Server and partition the local disk.

Redeployment Operations

After the redeployment, the data on the local disk will be lost. Exercise caution. Before redeployment, migrate services and back up data, and preprocess the local disk before instance redeployment. For details, see **Preprocessing for Instance Redeployment**.

If the faulty node meets the redeployment conditions listed in **Table 7-4**, you can go to the **Resource Management** > **Auxiliary Tools** > **Event Center** (old console: **Resource Management** > **Event Center**) page, locate the target node, and click **Redeploy** in the **Operation** column. In the dialog box that is displayed, enter **YES** and click **OK** to complete the redeployment authorization.

If the planned event does not meet the requirements listed in **Table 7-5**, the **Redeploy** button becomes unavailable.

After the O&M, Huawei technical support will disable the authorization. You do not need to perform further operations.

7.8 Restarting a Lite Server

Description

The **Lite Servers** page allows you to restart servers.

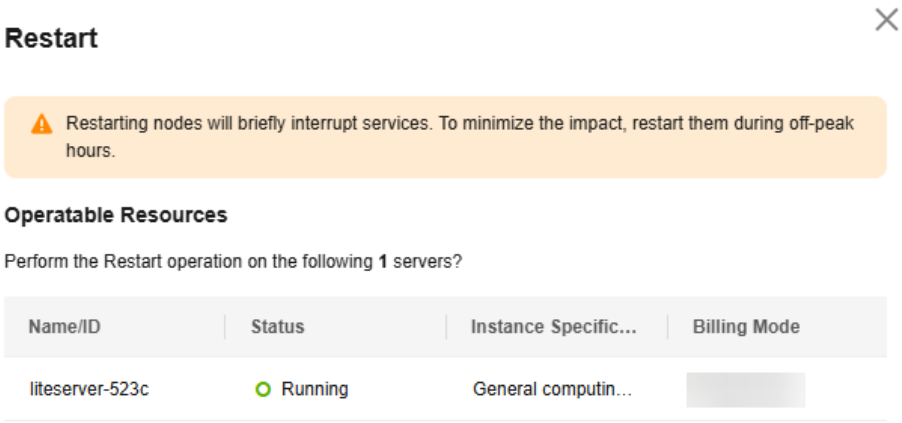
Constraints

- The server must be in the running or restart failed state.
- Restart is supported for both common nodes and supernodes.
- Common nodes and child nodes of supernodes can be batch restarted. However, batch operations are not supported for supernodes.

Procedure

1. Log in to the **ModelArts console**. In the navigation pane, choose **Resource Management** > **Lite Servers**.
   - New console: In the navigation pane, choose **Resource Management** > **Lite Compute Resources** > **Lite Servers**.
   - Old console: In the navigation pane, choose **Resource Management** > **Lite Servers**.
2. Restart the server in either of the following modes:
   - Method 1: In the resource list, locate the target common node or supernode and choose **More** > **Restart** in the **Operation** column. In the dialog box that is displayed, confirm the information and click **OK**.

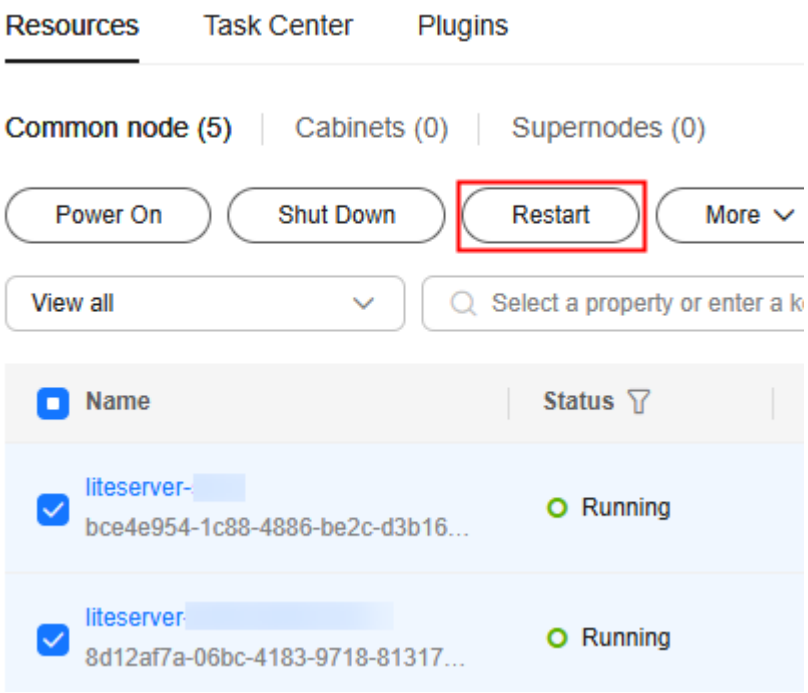
Figure 7-8 Restarting a server node



In the common node list, select multiple nodes to restart them in batches.

In the supernode list, select multiple nodes to batch restart them.

Figure 7-9 Batch restart



- Method 2: Click the name of a Lite Server. On the details page that is displayed, click **Restart** in the upper right corner. In the dialog box that is displayed, confirm the information and click **OK**.

8 Managing Lite Server Plugins

8.1 Installing the AI Plugin for a Lite Server

Description

NodeTaskHub is a plugin that manages elastic nodes efficiently. It sends tasks and enables automated O&M for ModelArts Lite Servers. It supports high-frequency operations such as Ascend software upgrade, real-time detection, and fault diagnosis, reducing manual intervention risks and ensuring stable and efficient AI service processes.

The Lite Server task center provides you with multiple task templates for creating tasks. Task delivery depends on the NodeTaskHub plugin installed on the Lite Server. For some public images of the Lite Server, the NodeTaskHub plugin is preset. You can choose to automatically install the plugin when purchasing the Lite Server. If it is not installed, manually install it by referring to the following content.

Constraints

- Currently, only Snt9b and Snt9b21 common nodes and Snt9b23 supernodes are supported.
- The node must be in the running state.
- The Docker service is required for the plug-in. Ensure that the Docker environment has been installed on the node. The Docker environment has been installed in the public OS image of the Lite Server. However, if it is not installed in your custom image, perform operations in **Install the Docker environment.**.
- The plug-in occupies port 25317 of the node.

Checking the Plug-in Installation

1. Log in to the **ModelArts console**. In the navigation pane, choose **Resource Management** > **Lite Servers**.
   - New console: In the navigation pane, choose **Resource Management** > **Lite Compute Resources** > **Lite Servers**.

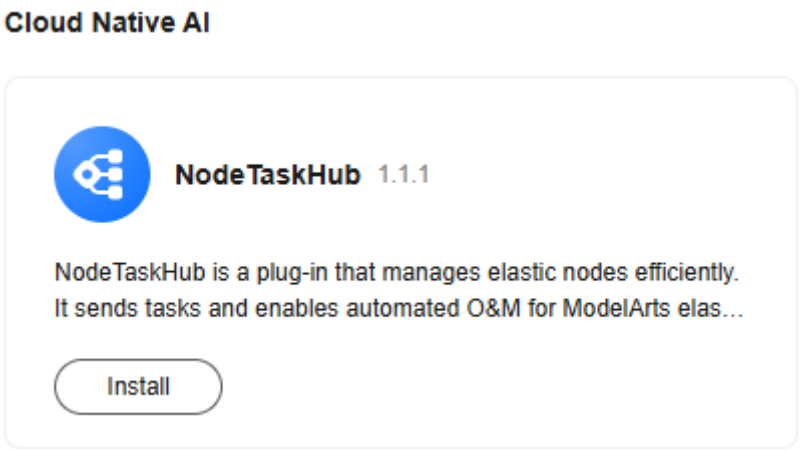
- Old console: In the navigation pane, choose **Resource Management** > **Lite Servers**.

2. On the plugin management page, check whether the plugin has been installed. If not, click **Install** in the **Operation** column on the right and install the plugin as prompted. Alternatively, manually install it by referring to **Installing a Plug-in Manually**.

Installing a Plug-in Manually

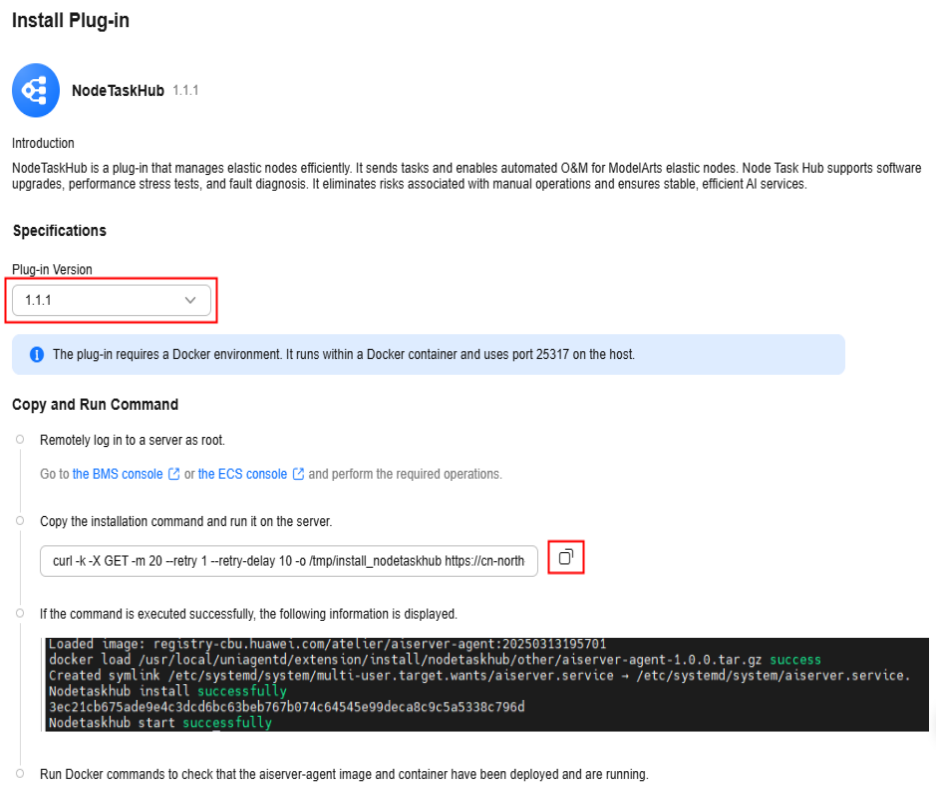
1. Log in to the **ModelArts console**.
2. In the navigation pane on the left, choose **Resource Management** > **Auxiliary Tools** > **Plugin Marketplace** (old console: **Plugin Marketplace**). Search for the NodeTaskHub plugin and click **Install**.

   Figure 8-1 Searching for the plug-in

3. Select the plugin version as prompted and copy the one-click installation command and execute it on the Lite Server to install NodeTaskHub.

Figure 8-2 Installing a plug-in



4. After the plug-in is installed, return to the **Plugins** tab. The plug-in is in the **Running** state.

Related Operations

In the navigation pane on the left, choose **Resource Management** > **Lite Servers**. Click the **Plugins** tab. You can restart, uninstall, and upgrade plugins.

Follow-Up Operations

After the plugin is installed, you can perform the following tasks in the Lite Server task center:

- **Upgrading the Ascend Driver and Firmware Version on a Lite Server**
- **Upgrading the GPU Driver on a Lite Server**
- **Installing or Upgrading the Cloud Eye Agent Plugin on a Lite Server**
- **Diagnosing Faults on Lite Servers**
- **Fixing Lite Server Vulnerabilities**
- **Performing a One-Click Stress Test on a Lite Server**
- **Configuring the Parameter Plane Network for Lite Servers**

8.2 Upgrading the Ascend Driver and Firmware Version on a Lite Server

Description

This section describes how to deliver an Ascend driver and firmware upgrade task on the Lite Server task center. In this way, you can upgrade the driver and firmware on the Snt9b and Snt9b23 servers with one click.

Constraints

- Currently, only Snt9b and Snt9b21 common nodes and Snt9b23 supernodes are supported.
- The upgrade may cause service interruption. Ensure that there is no running service on the node before the upgrade. After the upgrade, restart the node.
- If the official driver and firmware versions are used on the node and the upgrade fails, they can be rolled back to the original versions. If the driver and firmware on the node are damaged or the official versions are not used, the query will fail. In this case, the upgrade task can still be delivered. However, the driver and firmware cannot be rolled back once the upgrade fails, and you need to contact Huawei O&M engineers.
- The driver and firmware are compatible with the Ascend software package, such as CANN and MindSpore. Ensure that the upgraded versions are compatible with the Ascend software package used in the service. For details, see **Table 8-1**.

Table 8-1 Component compatibility

CANN Version	Ascend HDK Version
CANN 8.0.RC3	Ascend HDK 24.1.RC3 Ascend HDK 24.1.RC2 Ascend HDK 24.1.RC1 Ascend HDK 23.0.0/23.0.X
CANN 8.0.0	Ascend HDK 24.1.0 Ascend HDK 24.1.RC3 Ascend HDK 24.1.RC2 Ascend HDK 24.1.RC1 Ascend HDK 23.0.0/23.0.X

CANN Version	Ascend HDK Version
CANN 8.1.RC1	Ascend HDK 25.0.RC1 Ascend HDK 24.1.0 Ascend HDK 24.1.RC3 Ascend HDK 24.1.RC2 Ascend HDK 24.1.RC1 Ascend HDK 23.0.X
CANN 8.2.RC1	Ascend HDK 25.2.0 Ascend HDK 25.0.RC1 Ascend HDK 24.1.0 Ascend HDK 24.1.RC3 Ascend HDK 24.1.RC2

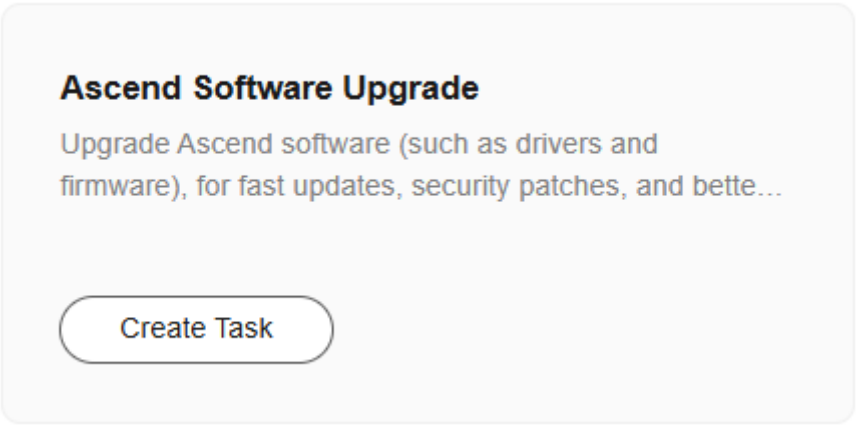
Prerequisites

This operation depends on the Lite Server AI plugin pre-installed on the node. Install the plugin by referring to **Installing the AI Plugin for a Lite Server**.

Procedure

1. Log in to the **ModelArts console**. In the navigation pane, choose **Resource Management** > **Lite Servers**. Click the **Task Center** tab.
   - New console: In the navigation pane, choose **Resource Management** > **Lite Compute Resources** > **Lite Servers**.
   - Old console: In the navigation pane, choose **Resource Management** > **Lite Servers**.
2. Click **Create Task** in the upper right corner. On the displayed **Job Templates** page, locate **Ascend Software Upgrade**, and click **Create Task**.

Figure 8-3 Task template



3. On the displayed page, set **Name**, **Description**, **Task**, and **Server Model**. Click **Select Node**. In the displayed node list on the right, select the target nodes and click **OK**. The driver firmware version query task will be delivered to the corresponding nodes, which takes about 1 minute.

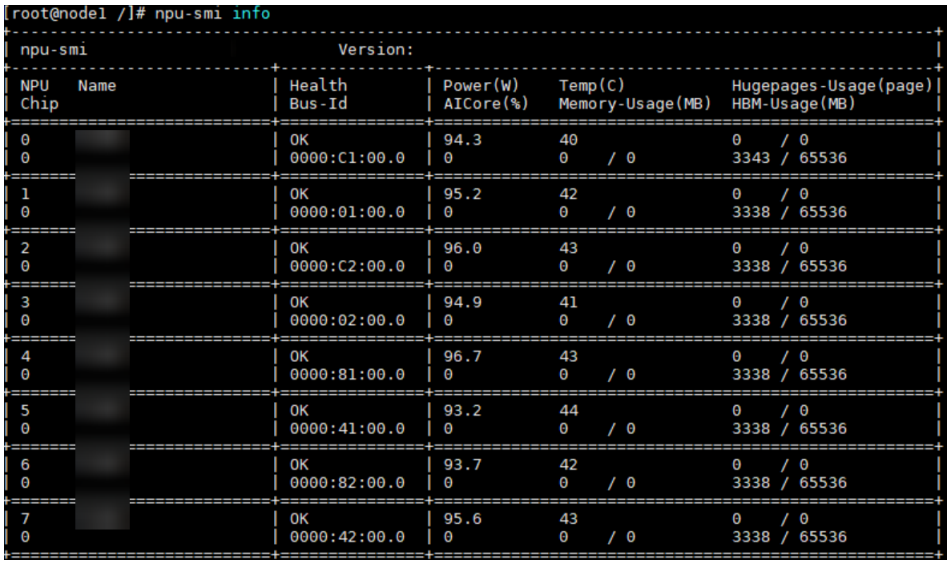
   Table 8-2 Parameters for creating a task

Parameter	Description
Name	The system automatically generates a task name. You can change the name as required.
Description	Enter the task description for quick search.
Task	Select **HDK upgrade**.
Server Model	Snt9b and Snt9b21 common nodes and Snt9b23 supernodes are supported.
Select Node	Click **Select Node**. In the node list displayed on the right, select the nodes where the driver and firmware need to be upgraded. You can select nodes in batches or search for nodes by keyword and click **OK**. This will deliver a query task to the selected nodes about the driver firmware version and CANN information. Wait for the query result, which takes about one minute.
Select Driver and Firmware Versions	Select the target driver and firmware versions from the drop-down list. Check the component compatibility by referring to **Table 8-1** to prevent service interruption caused by upgrade failure. You can also deliver a diagnosis task for Ascend devices by referring to **Diagnosing Faults on Lite Servers**. The compatibility between the driver & firmware and CANN will be automatically diagnosed.

4. Select the driver firmware version to be upgraded, click **Next**, confirm the upgrade information, select automatic or manual restart upon upgrade, and click **OK** to deliver the upgrade task. It takes about 10 minutes to complete the upgrade after the task is delivered.
5. During the upgrade, you can view the task execution status in the **Task Center** tab. Click the task name to access its details page, where you can view the task details and logs.
6. Restart is required upon upgrade. If manual restart is selected, run the **reboot** command on the node. The process takes about 10 minutes.
7. Run the command on the node to check whether the driver is loaded. If the command below is returned, the load is successful. Otherwise, contact Huawei technical support.

```
npu-smi info
```


Figure 8-4 Checking whether the driver is loaded



8.3 Upgrading the GPU Driver on a Lite Server

Description

In high-performance computing and deep learning, users often require the latest GPU drivers and related software to optimize computational performance. However, many GPU models currently on the market come pre-installed with outdated driver and software versions, which can lead to compatibility issues when users attempt to use the latest versions of CUDA.

To enhance user experience, Lite Servers provide a one-click software upgrade feature. This supports the automated upgrading of GPU drivers, CUDA, nvidia-fabricmanager, nv_peer_mem, and NCCL. Users can query supported software versions via commands and dispatch upgrade tasks, eliminating the tedious process of manually logging into different servers for software downloads, installation, and verification. Furthermore, the upgrade process automatically handles the deprecation of nv_peer_mem and the enablement of nvidia-peermem, ensuring version consistency across all components while improving system stability and reliability.

Table 8-3 Supported software for upgrade

Software	Description	Version
GPU driver	GPU driver, which has a specific compatibility relationship with CUDA.	550.90.07
CUDA	A parallel computing platform and programming model used to develop GPU-accelerated applications.	12.4

Software	Description	Version
nvidia-fabricmanager	Resource management and scheduling; manages NVLink, GPU, and network resources in multi-GPU and multi-node environments.	Matches the NVIDIA driver version. 550.90.07
nv_peer_mem	Data transfer acceleration; enables GPU Direct RDMA to optimize the data path between GPUs and NICs.	nv_peer_mem was deprecated in CUDA 11.5; its replacement (nvidia_peermem) is now integrated into the driver.
NCCL	A distributed communication library used to optimize data transfer efficiency in multi-GPU or multi-node environments.	2.27.6

Constraints

- Do not reset or power off the host or device during the software upgrade process. Doing so may cause the device to fail to boot or lead to an upgrade failure.
- Before upgrading the software package, ensure that no processes are occupying the node, including container mapping.
- Use the driver and fabricmanager versions from the same software version list to ensure version compatibility.
- Currently, only version 550 drivers are supported. Since there is no unified official version for existing user drivers, rollbacks are not supported.
- Supported models: Ant1, Ant8, Hnt02, Lnt002, and Vnt1.

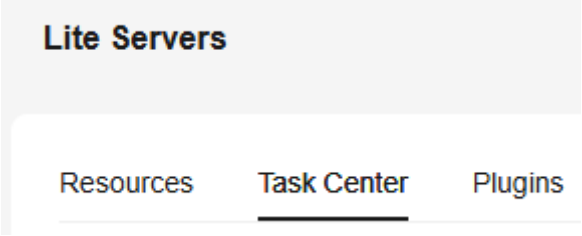
Prerequisites

This operation depends on the Lite Server AI plugin pre-installed on the node. Install the plugin by referring to **Installing the AI Plugin for a Lite Server**.

Procedure

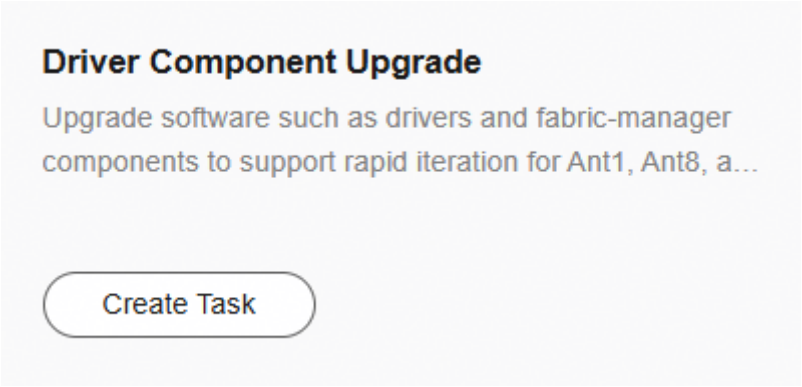
1. Log in to the **ModelArts console**. In the navigation pane, choose **Resource Management** > **Lite Servers**. Click the **Task Center** tab.
   - New console: In the navigation pane, choose **Resource Management** > **Lite Compute Resources** > **Lite Servers**.
   - Old console: In the navigation pane, choose **Resource Management** > **Lite Servers**.

Figure 8-5 Task center



2. Click **Create Task** in the upper left corner. On the displayed **Job Templates** page, locate **Driver Component Upgrade**, and click **Create Task**.

Figure 8-6 Task template



3. On the displayed page, enter the **Name** and **Description**, select the **Task** and **Server Model**, and click **Select Node**. After selecting the nodes from the node list and clicking **OK**, the system will dispatch a driver and firmware version query task to the corresponding nodes. This process takes approximately one minute to retrieve the actual driver and firmware information.

Table 8-4 Parameters for creating a task

Parameter	Description
Name	Modify the auto-generated task name as required.
Description	Enter a description for the task to help with quick identification and tracking.
Task	Select **Driver Upgrade**.
Server Model	Select Ant1, Ant8, Hnt02, Lnt002 or Vnt1.
Select Node	Click **Select Node** to choose the nodes requiring driver or firmware upgrades from the list. You can batch select nodes or search for specific nodes using keywords. Click **OK** to confirm.

Parameter	Description
Select Driver and Firmware Versions	Select the target version for the driver components from the drop-down list. Ensure that the target version is compatible with your software to prevent upgrade failures or service interruptions. Rollbacks are not supported for this upgrade. Perform a thorough risk assessment and back up your data before proceeding. Verify the driver version: `nvidia-smi`

4. After selecting the target driver version, click **Next** to verify the upgrade details. Click **OK** to dispatch the upgrade task. Once the task is initiated, the entire upgrade process takes approximately one hour for Ant1 models and about 30 minutes for other models.
5. During the upgrade, you can return to the **Task Center** page to monitor the execution status. Click a specific task name to access the task details page, where you can view detailed progress and logs.
6. After the process completes, run the relevant commands on the node to verify whether the driver has been loaded.
   ```
   nvidia-smi
   ```

8.4 Installing or Upgrading the Cloud Eye Agent Plugin on a Lite Server

Description

If the Cloud Eye Agent version on your Lite Server's OS image is outdated and disrupts Cloud Eye services, you cannot upgrade it using existing methods. Rebuilding the OS image or manually running commands is complex, slow, and error-prone, making these options inefficient. A quick, bulk upgrade solution for Cloud Eye Agent without interrupting operations was needed.

To address this, Lite Servers now feature Cloud Eye Agent installation and upgrade capabilities. Through the Lite Server platform, you can easily initiate batch installation or upgrade tasks for the Cloud Eye Agent. This feature supports target version installation and upgrade, significantly reduces the risks associated with manual intervention, and improves O&M efficiency and system stability.

Constraints

Currently, only Cloud Eye Agent on Snt9b and Snt9b21 common nodes and Snt9b23 supernodes can be installed or upgraded in batches.

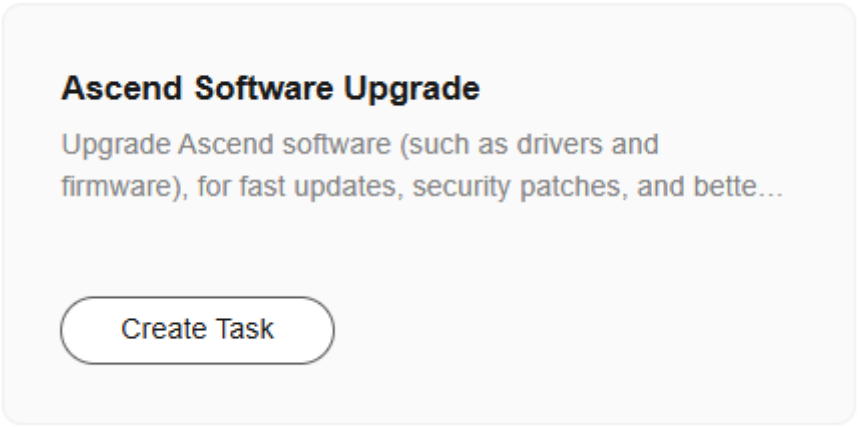
Prerequisites

This operation depends on the Lite Server AI plugin pre-installed on the node. Install the plugin by referring to **Installing the AI Plugin for a Lite Server**.

Procedure

1. Log in to the **ModelArts console**. In the navigation pane, choose **Resource Management** > **Lite Servers**. Click the **Task Center** tab.
   - New console: In the navigation pane, choose **Resource Management** > **Lite Compute Resources** > **Lite Servers**.
   - Old console: In the navigation pane, choose **Resource Management** > **Lite Servers**.
2. Click **Create Task** in the upper right corner. On the displayed **Job Templates** page, locate **Ascend Software Upgrade**, and click **Create Task**.

   Figure 8-7 Task template

3. On the displayed page, set **Name**, **Description**, **Task**, and **Server Model**, and click **Select Node**. In the displayed dialog box, select nodes and click **OK**. The version query task is delivered to the corresponding nodes to obtain the actual Cloud Eye Agent version information.

   Table 8-5 Parameters for creating a task

Parameter	Description
Name	The system automatically generates a task name. You can change the name as required.
Description	Enter the task description for quick search.
Task	Select **Cloud Eye-Agent upgrade**.
Server Model	Snt9b and Snt9b21 common nodes and Snt9b23 supernodes are supported.

Parameter	Description
Select Node	Click **Select Node**. In the node list displayed on the right, select the nodes where Cloud Eye Agent needs to be upgraded. You can select nodes in batches or search for nodes by keyword. Click **OK**. This will submit a query task to the selected nodes about the Cloud Eye Agent version. Wait for the query result, which takes about 1 to 2 minutes.
Select Cloud Eye Agent Version	Select the target Cloud Eye Agent version from the drop-down list.

4. Select the target Cloud Eye Agent version, click **Next**, confirm the upgrade information, and click **OK**. It takes about five minutes to complete the upgrade after the task is created.
5. During the upgrade, you can view the task execution status in the **Task Center** tab. Click the task name to access its details page, where you can view the task details and logs.

   If the task execution status is Successful, the Cloud Eye Agent is installed or upgraded.

Related Operations

After the Cloud Eye Agent is installed or upgraded, you can monitor Lite Server resources. For details, see **Using Cloud Eye to Monitor NPU Resources of Lite Servers**.

8.5 Diagnosing Faults on Lite Servers

Description

The **Task Center** for Lite Servers provides one-click fault diagnosis capabilities, covering both parameter plane network diagnosis and Ascend device diagnosis. You can quickly perform network and Ascend device health checks directly from the product console without needing in-depth knowledge of specific diagnosis command-line operations.

For the parameter plane network diagnosis, you can query the network status, IP address, and mask information of the PU. For the Ascend device diagnosis, you can check the driver firmware version compatibility and implement automatic in-band check. You can batch start diagnosis tasks on multiple servers at the same time, improving efficiency to a large extent.

Constraints

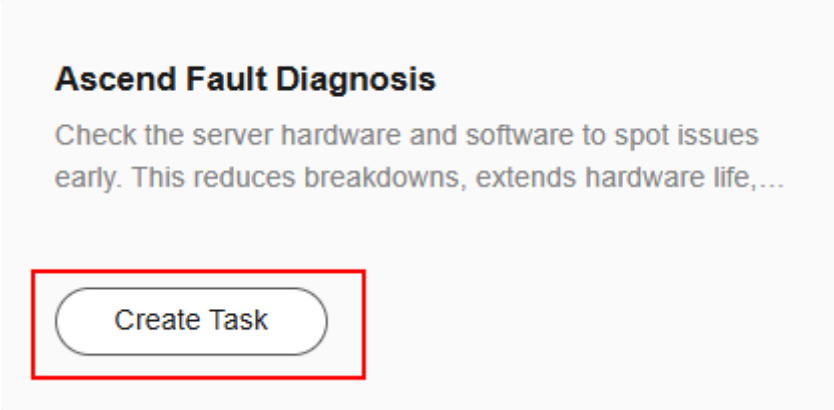
- Currently, only Snt9b and Snt9b21 common nodes and Snt9b23 supernodes are supported.
- You can select a maximum of 50 common nodes or supernode child nodes for the same task.

- The NodeTaskHub plug-in is required for the node where the task is to be created. Ensure that the plug-in is installed before task creation. For details, see **Installing the AI Plugin for a Lite Server**.
- Only one diagnosis task can be executed on a node at the same time. The task cannot be interrupted once started. Plan the task priority.
- Ensure that no services are running on the nodes you are going to diagnose. Running commands during diagnosis can cause service interruptions or errors.
- Install the MCU, driver, and firmware for Ascend HDK 23.0.0 or later before starting the diagnosis. A preconfigured OS is already installed. If you use a custom OS, ensure that the software has been installed correctly.
- The diagnosis requires the Ascend-docker-runtime development kit. This software is pre-installed on the default OS. If you use a custom OS, ensure the software has been installed correctly.

Procedure

1. Log in to the **ModelArts console**. In the navigation pane, choose **Resource Management** > **Lite Servers**. Click the **Task Center** tab.
   - New console: In the navigation pane, choose **Resource Management** > **Lite Compute Resources** > **Lite Servers**.
   - Old console: In the navigation pane, choose **Resource Management** > **Lite Servers**.
2. Click **Create Task** in the upper right corner. On the displayed **Job Templates** page, locate **Ascend Fault Diagnosis**, and click **Create Task**.

   Figure 8-8 Task templates

3. On the **Ascend Fault Diagnosis** page, enter the task name and description. Set server model and type, select a diagnosis item, select the notice, and click **Create now**.

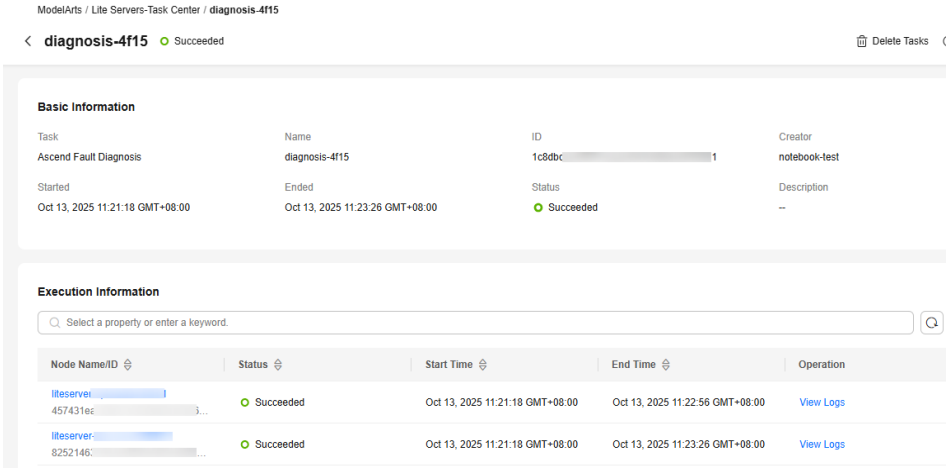
   Table 8-6 Parameters for creating a task

Parameter	Description
Name	The system automatically generates a task name. You can change the name as required.

Parameter	Description
Description	Enter the task description for quick search.
Server Model	Select a server model and select nodes in the node list. You can search for node information using keywords. Snt9b and Snt9b21 common nodes and Snt9b23 supernodes are supported.
Diagnosis Item	You can select **Parameter Plane Network Diagnosis**, **Ascend Device Diagnosis**, or both. • **Parameter Plane Network Diagnosis**: Check and record parameter-plane network metrics and information. • **Ascend Device Diagnosis**: Check the health and compatibility of Ascend software and chip metrics.

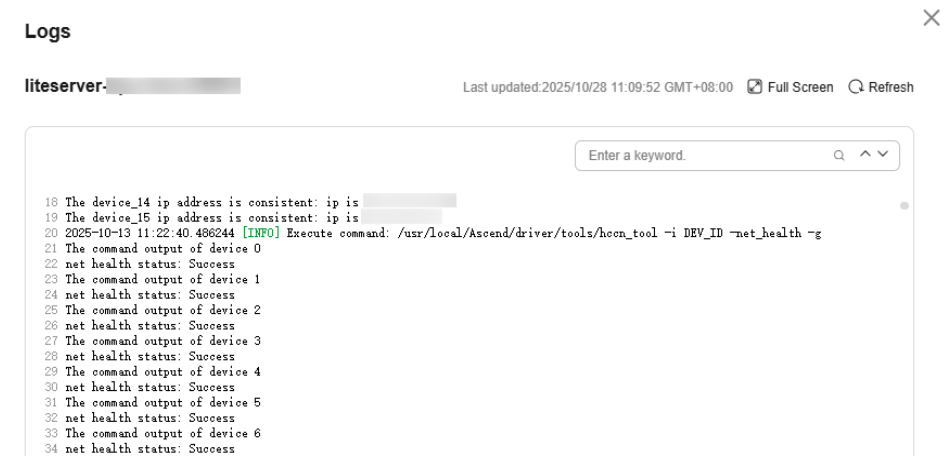
4. View the task execution status in the **Task Center** tab.
5. Click the task name to access its details page, where you can view the task details.

Figure 8-9 Checking the task details



6. On the task details page, locate the target node and click **View Logs** in the **Operation** column. In the displayed window on the right, view the detailed log about task execution. All check results are displayed in the task logs, and basic log analysis is provided.

Figure 8-10 Viewing logs



In-band Automatic Check Items

The table below lists the in-band automatic check items in the Ascend device check task.

Table 8-7 In-band automatic check items

Check Item	Command Reference	Action
Checking the UDP port split configuration	hccn_tool -i $i -udp -g	Check whether the port number is 0/4791.
Checking the NPU health information	timeout 20s npu-smi info -t health -i "$i" | grep OK -c	Only for Snt9b23. Check whether the NPU health code is 3.
Checking whether the NPU driver versions are consistent	timeout 20s npu-smi info -t board -i "$i" | grep Version	Check whether the driver numbers of all NPUs are the same.
Checking the PCIE link status	lspci | grep d8 / lspci | grep d8 -c	Only for Snt9b23. Check whether the PCIE link is 16.
Checking whether the NPU NIC is UP	hccn_tool -i $i -link -g	Check whether the NIC is down.
Checking the NPU NIC health status	hccn_tool -i $i -net_health -g	Check whether the NIC is healthy.
Checking whether the NPU PFC meets the requirements	hccn_tool -i $i -pfc -g	Check whether the PFC meets the requirements. The PFC configuration is as follows:

Check Item	Command Reference	Action
Checking whether the TLS certificate meets the requirements	`hccn_tool -i $i -tls -g | grep switch`	Check whether **switch[0]** in the field meets the requirements.
Driver and firmware version compatibility test	`ascend-dmi -ci`	Check whether the compatibility meets the requirements.

8.6 Fixing Lite Server Vulnerabilities

Description

O&M teams often face security issues like kernel or OpenSSH vulnerabilities after updating operating systems. These problems threaten system stability and safety. Fixing them usually requires rebuilding the OS image, which takes too much time and effort. To solve this, Lite Servers offer a quick fix tool. It allows you to scan and resolve multiple security vulnerabilities with just one click. This helps O&M staff manage system security easily, quickly fixing vulnerabilities to keep services running smoothly and safely.

Constraints

The constraints on creating a custom script task on the O&M plane are as follows:

- Currently, only Snt9b and Snt9b21 common nodes and Snt9b23 supernodes are supported.
- Only vulnerabilities in the Huawei Cloud EulerOS 2.0 can be fixed.
- The NodeTaskHub plug-in is required for the node where the task is to be created. Ensure that the plug-in is installed before task creation. For details, see **Installing the AI Plugin for a Lite Server**.
- Only one task can be executed on a node at the same time. The task cannot be interrupted once started. Plan the task priority.
- Ensure that no services are running on the target node. Services may be interrupted or abnormal during task execution.
- Install the MCU, driver, and firmware for Ascend HDK 23.0.0 or later before starting the diagnosis. A preconfigured OS is already installed. If you use a custom OS, ensure that the software has been installed correctly.
- The diagnosis requires the Ascend-docker-runtime development kit. This software is pre-installed on the default OS. If you use a custom OS, ensure the software has been installed correctly.

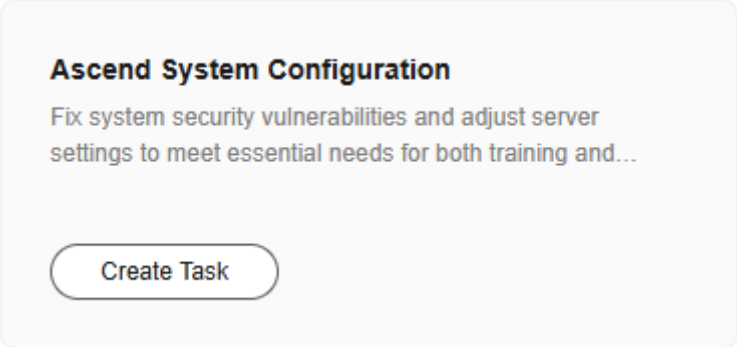
Procedure

1. Log in to the **ModelArts console**. In the navigation pane, choose **Resource Management** > **Lite Servers**. Click the **Task Center** tab.

- New console: In the navigation pane, choose **Resource Management** > **Lite Compute Resources** > **Lite Servers**.
- Old console: In the navigation pane, choose **Resource Management** > **Lite Servers**.

2. Click **Create Task** in the upper right corner. On the displayed **Job Templates** page, locate **Ascend System Configuration**, and click **Create Task**.

Figure 8-11 Task template



3. On the **Ascend System Configuration** page, set **Name**, **Description**, **Server Model**, and **Configuration Item**, agree to the terms of use, and click **Create now**.

Table 8-8 Parameters for creating a task

Parameter	Description
Name	The system automatically enters the name of the system configuration task. You can change the task name.
Description	Enter the task description for quick search.
Server Model	Select a server model and select nodes in the node list. You can search for node information using keywords. Snt9b and Snt9b21 common nodes and Snt9b23 supernodes are supported.
Configuration Item	**System Bug Fixes** identifies and fixes Huawei Cloud EulerOS 2.0 system security vulnerabilities.

4. View the task execution status in the **Task Center** tab.
5. Click the task name to access its details page, where you can view the task details.
6. On the task details page, locate the target node and click **View Logs** in the **Operation** column. In the displayed window on the right, view the detailed log about task execution. All check results are displayed in the task logs, and basic log analysis is provided.

Vulnerability Check Items

The table below describes how to fix system vulnerabilities in an Ascend system configuration task.

Vulnerability	OS	Solution
ipvs_fnat	HCE2.0	Checks whether the vulnerability exists. If yes, the system automatically fixes the vulnerability.
openssh	HCE2.0	Checks whether the vulnerability exists. If yes, the system automatically fixes the vulnerability.

8.7 Performing a One-Click Stress Test on a Lite Server

Description

The **Task Center** of Lite Servers provides one-click stress tests. You can quickly perform a stress test on a Lite Server without learning about software stacks such as AI Core and HBM. The task allows you to test the bandwidth, compute, power consumption, and diagnosis pressure of Ascend servers, providing hardware assurance for high-load scenarios such as AI training and inference. In addition, the task can be concurrently executed on multiple servers in batches, greatly improving efficiency.

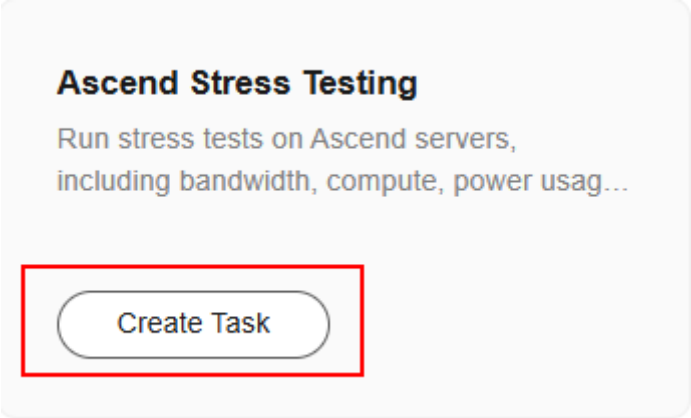
Constraints

- Currently, only Snt9b and Snt9b21 common nodes and Snt9b23 supernodes are supported.
- The NodeTaskHub plug-in is required for the node where the task is to be created. Ensure that the plug-in is installed before task creation. For details, see **Installing the AI Plugin for a Lite Server**.
- Only one pressure test task can be executed on a node at the same time. The task cannot be interrupted once started. Plan the task priority.
- Ensure that no services are running on the target nodes. Running commands during the pressure test can cause service interruptions or errors.
- Install the MCU, driver, and firmware for Ascend HDK 23.0.0 or later before starting the pressure test. A preconfigured OS is already installed. If you use a custom OS, ensure that the software has been installed correctly.
- The pressure test requires the Ascend-docker-runtime development kit. This software is pre-installed on the default OS. If you use a custom OS, ensure the software has been installed correctly.

Procedure

1. Log in to the **ModelArts console**. In the navigation pane, choose **Resource Management** > **Lite Servers**. Click the **Task Center** tab.
   - New console: In the navigation pane, choose **Resource Management** > **Lite Compute Resources** > **Lite Servers**.
   - Old console: In the navigation pane, choose **Resource Management** > **Lite Servers**.
2. Click **Create Task** in the upper right corner. On the displayed **Job Templates** page, locate **Ascend Stress Testing**, and click **Create Task**.

   Figure 8-12 Task templates

3. On the **Ascend Stress Testing** page, enter the task name and description. Set server model and type, select a test case, select the notice, and click **Create now**.

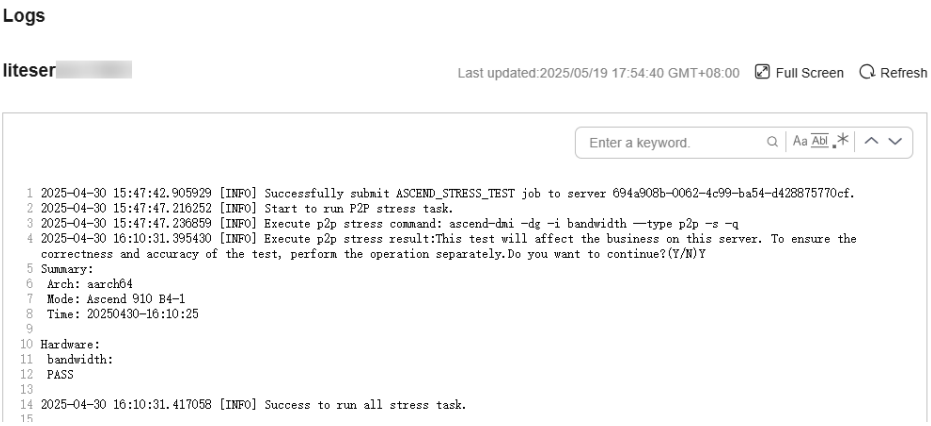
   Table 8-9 Parameters for creating a task

Parameter	Description
Name	The system automatically generates a task name. You can change the name as required.
Description	Enter the task description for quick search.
Server Model	Select a server model and select nodes in the node list. You can search for node information using keywords. Snt9b and Snt9b21 common nodes and Snt9b23 supernodes are supported.

Parameter	Description
Test Case	You can select any of the following pressure test cases. The pressure test cases can be executed one by one or at the same time. • **AI Core Stress Test**: Run a stress test on AI Core errors to diagnose issues. The test uses 20 to 40 GB of memory on the host server. Before you start, make sure there is enough memory or the test may fail. • **HBM Stress Test**: Run a stress test on the high-bandwidth memory to get results. • **P2P Stress Test**: Check for faults on the HCCS communication links between all devices on the test node.

4. View the task execution status in the **Task Center** tab.
5. Click the task name to access its details page, where you can view the task details.
6. On the task details page, locate the target node and click **View Logs** in the **Operation** column. In the displayed window on the right, view the detailed log about task execution.

Figure 8-13 Viewing logs



8.8 Configuring the Parameter Plane Network for Lite Servers

Description

The **Task Center** of Lite Servers provides one-click system configuration. You can quickly complete network configurations on the product page of Lite Servers. To meet basic requirements of training and inference scenarios and improve server performance, you can optimize server parameters, such as the RoCE network

uplink port and parameter plane network IP address. In addition, you can perform operations on multiple servers in batches for higher efficiency.

Constraints

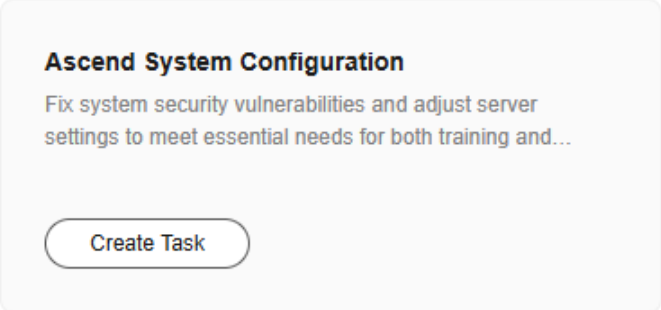
- Currently, only Snt9b and Snt9b21 common nodes and Snt9b23 supernodes are supported.
- The NodeTaskHub plug-in is required for the node where the task is to be created. Ensure that the plug-in is installed before task creation. For details, see **Installing the AI Plugin for a Lite Server**.
- Only one task can be executed on a node at the same time. The task cannot be interrupted once started. Plan the task priority.
- Ensure that no services are running on the target node. Services may be interrupted or abnormal during task execution.
- Install the MCU, driver, and firmware for Ascend HDK 23.0.0 or later before starting the task. A preconfigured OS is already installed. If you use a custom OS, ensure that the software has been installed correctly.
- The task requires the Ascend-docker-runtime development kit. This software is pre-installed on the default OS. If you use a custom OS, ensure the software has been installed correctly.

Procedure

1. Log in to the **ModelArts console**. In the navigation pane, choose **Resource Management** > **Lite Servers**. Click the **Task Center** tab.
   - New console: In the navigation pane, choose **Resource Management** > **Lite Compute Resources** > **Lite Servers**.
   - Old console: In the navigation pane, choose **Resource Management** > **Lite Servers**.

2. Click **Create Task** in the upper right corner. On the displayed **Job Templates** page, locate **Ascend System Configuration**, and click **Create Task**.

   Figure 8-14 Task template

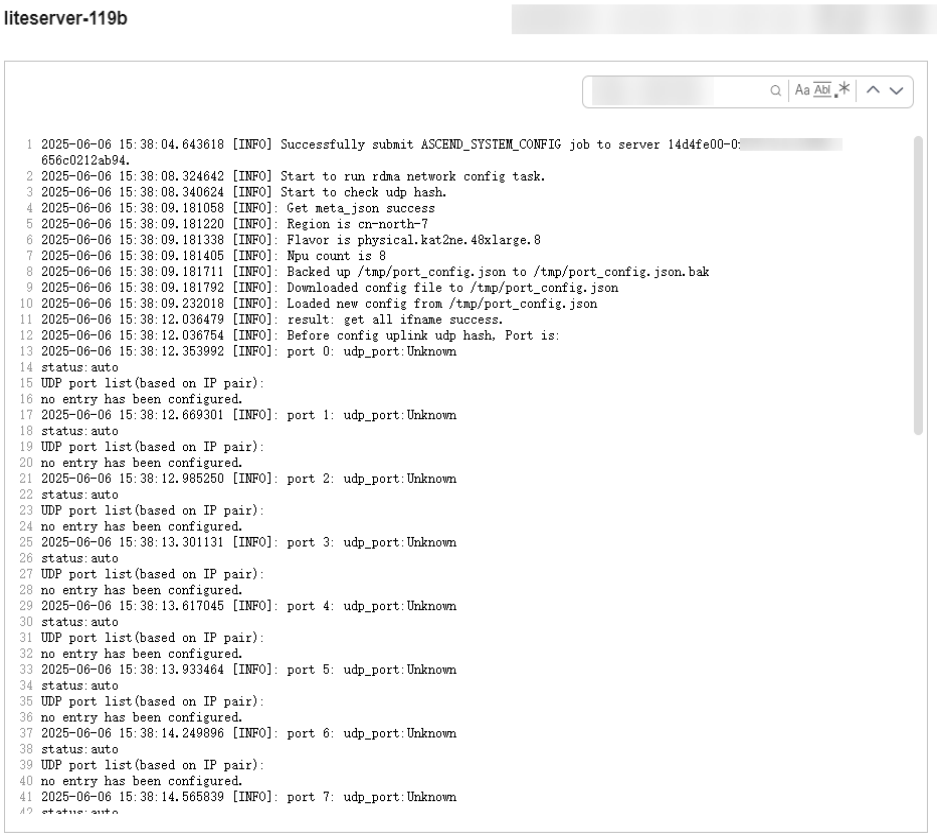
3. On the **Ascend System Configuration** page, set **Name**, **Description**, **Server Model**, and **Configuration Item**, agree to the terms of use, and click **Create now**.

Table 8-10 Parameters for creating a task

Parameter	Description
Name	The system automatically generates a task name. You can change the name as required.
Description	Enter the task description for quick search.
Server Model	Select a server model and select nodes in the node list. You can search for node information using keywords. Snt9b and Snt9b21 common nodes and Snt9b23 supernodes are supported.
Configuration Item	**Parameter Plane Network Configuration**: Configure the RoCE network uplink ports properly and set up the parameter plane network with correct IP addresses, subnet masks, and gateways.

4. View the task execution status in the **Task Center** tab.
5. Click the task name to access its details page, where you can view the task details.
6. On the task details page, locate the target node and click **View Logs** in the **Operation** column. In the displayed window on the right, view the detailed log about task execution.

Figure 8-15 Viewing logs



8.9 Performing Health Check on Lite Servers

Description

During routine O&M, O&M engineers may face many challenges in pressure tests, health checks, fault detection, and log analysis. To improve O&M efficiency, Lite Servers interconnect with AI Compute Service Brain to comprehensively inspect and manage Lite Servers. You can directly create a health check job on the **Lite Servers** page. In this way, O&M engineers can access the AI Compute Service Brain page, and create and execute a health check job, improving system O&M capabilities and reliability.

Constraints

- Currently, only Snt9b and Snt9b21 common nodes and Snt9b23 supernodes are supported.
- The Lite Server node is in the **Running** state.

Prerequisites

The NodeTaskHub plugin has been installed for the Lite Server on which you want to create a health check task. For details, see **Installing the AI Plugin for a Lite Server**.

Creating an Inspection Job

1. Log in to the **ModelArts console**. In the navigation pane, choose **Resource Management** > **Lite Servers**.
   - New console: In the navigation pane, choose **Resource Management** > **Lite Compute Resources** > **Lite Servers**.
   - Old console: In the navigation pane, choose **Resource Management** > **Lite Servers**.
2. In the **Common node** list, choose **More** > **Create Health Check Task** in the **Operation** column on the right. On the displayed page, configure parameters.

Table 8-11 Parameters for creating a health check task

Configuration Item	Parameter	Description
Basic Configuration	Assignment Name	Enter a custom inspection task name.
Inspection Object	Select Object	You can select common nodes, supernodes, or an entire rack of nodes. Select the target nodes in the list. You can select at most 48 nodes.
Inspection Type	Standard Inspection	Minute-scale check that does not affect jobs on nodes.
	Deep Inspection	Hour-scale check that affects services on nodes. This will occupy NPUs for a long time. Ensure that no service is running in the cluster during the inspection.
Load Test Case Configuration	NPU Performance Diagnosis	Perform performance diagnosis by bandwidth, AI FLOPs, or eye pattern test. You can select one or more options for diagnosis. • **BandWidth**: Diagnose the local bandwidth. • **Aiflops**: Diagnose the compute of chips. • **Eye Diagram Test**: Query the detailed data of the signal quality.
	NPU stress testing	Perform the stress test on the AI Core, HBM, and P2P. • AI Core stress test: Perform a pressure test on AI Core errors. • HBM stress test: Perform a stress test on high-bandwidth memory. • P2P stress test: Check whether a hardware fault occurs on the HCCS communication link from the source device to the target device.

Configuration Item	Parameter	Description
	Network Load Testing	Perform the single-node HCCL communication bandwidth test, multi-node HCCL bandwidth test, and RDMA communication bandwidth test. • Single-node HCCL communication bandwidth test: Perform the collective communication performance pressure test between a single compute node. • Multi-node HCCL bandwidth test: Perform the baseline collective communication performance pressure test between multiple compute nodes. • RoCE network bandwidth test: The system tests the RoCE network's bandwidth between two nodes. • Hyperplane test: tests the collective communication bandwidth of the hyperplane network.

3. Read the term of use, enter **YES**, and click **Create now**.
4. After the inspection job is submitted, go to the ModelArts console. In the navigation pane on the left, choose **Health Inspection** under **O&M Management**. View the inspection job status and details.

9 Managing Lite Server Supernodes

9.1 Scaling a Lite Server Supernode

Description

After using Lite Server supernodes for a period of time, you may need to increase or decrease the number of supernodes due to service changes.

Constraints

Scaling out

- This applies only to the scale–out of Snt9b23 supernodes.
- There can be no more than 48 child nodes during the supernode scale–out.
- Only available nodes can be scaled out.
- The scale–out does not affect services on original nodes.

Scaling in

- This applies only to the scale-in of Snt9b23 supernodes.
- If the supernode has only one child node, scale-in is not supported.
- If the node to be scaled in is in an intermediate state, such as switching OS, restarting, starting, or stopping, scale-in is not supported.
- Scaling in affects services on original nodes and the EVS disks of the nodes are released. Ensure that services have been migrated away from the nodes to be scaled in to prevent data loss.

Billing

Yearly/monthly is supported for both node scale-out and scale-in. A billing order will be generated for the new nodes.

Scaling out the Supernodes

1. Log in to the **ModelArts console**. In the navigation pane, choose **Resource Management** > **Lite Servers**. The **Resources** tab is displayed.

- New console: In the navigation pane, choose **Resource Management** > **Lite Compute Resources** > **Lite Servers**.
- Old console: In the navigation pane, choose **Resource Management** > **Lite Servers**.

2. Click the **Supernodes** tab, locate the target supernode in the list, and choose ••• > **Change Supernode Specifications** on the right.
3. On the displayed page, view the resource information to be changed and select the target specifications. Sold-out resources are displayed in gray and cannot be selected.
4. Set the parameters.

Table 9-1 Parameters

Parameter	Description
System Disk	Select **System Disk Type** and set **Size**. A system disk is automatically created when you create a Lite Server. Set the system disk size to at least 200 GB.
Add Data Disk	Click **Add Data Disk** to attach a data disk to the Lite Server. You can also attach data disks on ECS after the Lite Server resource is created. For details, see **Using EVS for Storage**.
Image	• **Public image** Public images are available for all users. All users can read the image by image ID. ModelArts allows you to perform development and training directly without additional configuration as it provides multiple public images, supports multiple OSs, and has built–in AI drivers and software. For details about the supported public images, see **Mappings Between Lite Compute Nodes and OS Images**. • **Private image** Only the image creator can use the image. You can select a private image to save your time from repeatedly configuring servers.
Login Credential	**Key pair** is recommended as it features higher security than **Password**. Only key pairs can be used as the login credential for expanding the capacity of supernodes. • **Key pair** Use a key pair to log in to the Server node. You can select an existing key pair, or click **Create Key Pair** to create one. **NOTE** If you use an existing key pair, ensure that you have saved the key file locally. Otherwise, logging in to the Server node will fail.

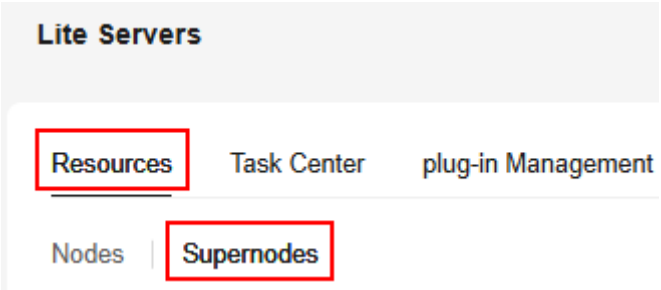
Parameter	Description
(Optional) Custom Instance Injection	Use this function to configure Server nodes if you want to: • Use scripts to simplify the Server node configuration. • Use scripts to initialize OSs. • Use existing scripts and upload them to the server when creating the Server node. • Use scripts for other purposes. Currently, **As text** and **As file** are supported. For details, see **Injecting User Data**.

5. Click **Next** to preview the change information, view the node configuration and fee, and click **Submit**.
6. After the scale-out is complete, expand the supernode that has been scaled out on the resource list page and check the status of the new child nodes under the supernode.

Scaling in the Supernodes

1. Log in to the **ModelArts console**. In the navigation pane, choose **Resource Management** > **Lite Servers**. The **Resources** tab is displayed.
   - New console: In the navigation pane, choose **Resource Management** > **Lite Compute Resources** > **Lite Servers**.
   - Old console: In the navigation pane, choose **Resource Management** > **Lite Servers**.

   Figure 9-1 Resource list

2. Click the **Supernodes** tab, locate the target supernode in the list, and choose ... > **Change Supernode Specifications** on the right.
3. On the displayed page, view the resource information to be changed and select the target specifications. In the list of target reserved nodes, select the nodes to be reserved.

   Target specifications specify specifications of the nodes to be reserved.

   The number of reserved nodes must be the same as the number of compute nodes in the target specifications. Otherwise, the scale-in task fails to be submitted.

4. Select **I acknowledge the potential risks and agree to proceed with the scale-in** and enter **YES** to confirm the scale-in.

   WARNING

   When scaling in, unreserved nodes and their associated EVS disks will be removed. Move your services off these nodes first to avoid losing data.

5. Click **Next** to preview the change information, view the node reservation and deletion information, confirm the information, and click **Submit**.

9.2 Expanding the System Disk Capacity of a Lite Server Supernode

Description

When you use a supernode, the OS fails to be switched due to insufficient system disk space. Additionally, during routine service runtime, data is continuously increasing. As a result, the remaining space of the system disk decreases and finally reaches the upper limit, affecting service running.

To address these problems and ensure service continuity and stability, Lite Server allows you to expand the system disk capacity on the node details page. You can easily switch to the EVS page to expand the capacity as required, preventing operation failures or service interruption caused by insufficient space.

Constraints

- Expanding a supernode system disk depends on EVS capacity expansion. Go to the EVS console to expand the system disk capacity.
- A supernode's system disk can expand but cannot shrink.
- Common nodes cannot expand or shrink their system disk size. Attempting these changes through the EVS console could cause errors.

Billing

After the disk capacity expansion, the billing information will be displayed in the billing order generated during supernode creation. No new billing order is generated.

Expanding the System Disk Capacity of a Supernode

1. Log in to the **ModelArts console**. In the navigation pane, choose **Resource Management** > **Lite Servers**. The **Resources** tab is displayed.
   - New console: In the navigation pane, choose **Resource Management** > **Lite Compute Resources** > **Lite Servers**.
   - Old console: In the navigation pane, choose **Resource Management** > **Lite Servers**.

2. In the **Supernodes** tab, click the target supernode to access its child node details page.
3. On the child node details page, locate the system disk to be expanded and click **Expand** on the right. Click **Expand** as prompted.
4. On the displayed EVS console, expand the disk capacity by referring to **Expand Disk Capacity**.
5. After the system disk capacity is expanded, manually synchronize the information on the ModelArts Lite Server page by referring to **Synchronizing the Status of a Lite Server**.

Follow-Up Operations

Synchronizing the Status of a Lite Server

9.3 Performing Periodic Stress Tests on Lite Server Supernodes

Scenario

For Snt9b23 supernodes, you can perform periodic performance tests and fault diagnosis on Huawei Cloud AI servers to detect NPU faults in a timely manner and reduce impact on services.

Table 9-2 Performance test

Scenario	Description
Bandwidth test	Test the bus bandwidth, memory bandwidth, and total time consumption.
Compute test	Construct a matrix multiplication **A(m,k)*B(k,n)** and execute it a certain number of times. Calculate the AI Core compute and the real-time power at full computational load of the entire PU or processor based on the computation workload and the time taken to execute the matrix multiplication.
Power consumption test	Detect the power consumption of the entire PU by running the single-operator model.
Eye pattern test	Test the network and query the current signal quality.
Stream test	One-click traffic testing and custom traffic testing are supported.
Software and hardware version compatibility test	The software and hardware compatibility tool obtains the hardware information, architecture, driver version, firmware version, and software version.

Table 9-3 Fault diagnosis

Scenario	Description
Network diagnosis	Diagnose the health status of the network and output the diagnosis result.
Signal quality diagnosis	Diagnose the signal quality and output the diagnosis result.
On-chip memory diagnosis	Diagnose the high bandwidth memory and output the diagnosis result.
On-chip memory stress test	Run a stress test on the high-bandwidth memory and output the diagnosis result.
On-chip memory high-risk address stress test	Run a stress test on the high-risk addresses of the high bandwidth memory and output the diagnosis result.
AI Core diagnosis	Diagnose the AI Core error and output the diagnosis result.
AI FLOPs diagnosis	Diagnose the chip compute and output the test result.
Bandwidth diagnosis	Diagnose the local bandwidth and output the diagnosis result.
P2P stress test	Check whether the HCCS communication link from the source device to the target device has hardware faults and output the test result.
Power consumption stress test	Perform the EDP/TDP power consumption stress test and output the diagnosis result.

Constraints

- Only Snt9b23 supernodes are supported.
- Ascend DMI is used for stress test. Starting multiple processes on the same device to test performance data is not supported. If multi-process test is performed, the test result may be inaccurate or unpredictable.
- Performance test and fault diagnosis will affect training and inference services. Ensure that no service is running first.
- To ensure the correctness and accuracy of the test result, run each detection command separately.
- Ascend DMI can check only the NPUs that are properly installed. To ensure the accuracy of the test result, run the **npu-smi info** command first.

Performance Test 1: Bandwidth Test

Test the bus bandwidth, memory bandwidth, and total time consumption. **Table 9-4** describes the parameters of the bandwidth test command.

```
ascend-dmi --bw -h
```


Table 9-4 Parameters for the bandwidth test

Parameter	Description	Mandatory
[-bw, --bw, --bandwidth]	This parameter is used to test the bandwidth of the chip. **-bw** is supported, but **--bw** and **--bandwidth** are recommended.	Yes
[-t, --type]	Specify the data flows to be tested. When the bandwidth test function is used, the tested data flows can be classified into the directions listed below. If this parameter is not specified, **h2d**, **d2h**, and **d2d** are returned by default. Bandwidth and total duration in the three directions. • **h2d**: Data is transferred from the host memory to the device memory through the PCIe bus. The overall bandwidth and total duration are tested. • **d2h**: Data is transferred from the device memory to the host memory through the PCIe bus. The overall bandwidth and total duration are tested. • **d2d**: Data is transferred from the device memory to the same device memory. (This mode is used to test the device memory bandwidth.) The overall bandwidth and total duration are tested. • **p2p**: Data is transferred from the specified source device to the target device. The transfer rate and total duration are tested.	No
[-s, --size]	Specify the data size to be transferred and the test result display mode. For supernodes, the maximum transmission value ranges from 1 byte to 4 GB in d2h, h2d, and p2p modes. • The **-s** parameter must be followed by a number, which specifies the size of the data to be transferred. If no number is specified, the syntax is incorrect. – In h2d, d2h, d2d, and p2p modes where **-ds** and **-dd** are specified, **-s** specifies the fixed-length mode. If **-s** is not specified, the step mode is used. The default data transfer range ranges from 2 bytes to 32 MB.	No
[-et, --et, --execute-times]	Specify the number of iterations, that is, the number of memory copy times. The value ranges from 1 to 1000. If this parameter is not specified, the default value **5** is used in step mode, and the default value **40** is used in fixed-length mode.	No

Parameter	Description	Mandatory
[-d, --device]	Specify the ID of the device whose bandwidth needs to be tested. The device ID is the logical ID of the Huawei Cloud AI processor. If the device ID is not specified, the bandwidth information of device 0 is returned by default.	No
[-ds, --ds, --device-src]	Specify the ID of the source device for a P2P test. This parameter must be specified together with the **-dd**, **--dd**, and **--device-dst** parameters. Otherwise, all Huawei Cloud AI NPU chips are tested.	No
[-dd, --dd, --device-dst]	Specify the ID of the target device for a P2P test. This parameter must be specified together with the **-ds**, **--ds**, and **--device-src** parameters. Otherwise, all Ascend NPU chips are tested.	No
[-fmt, --fmt, --format]	Specify the output format. The value can be **normal** or **json**. If not specified, the default value **normal** is used.	No
[-q, --quiet]	If this parameter is specified, no foolproof message is displayed. By default, this operation is allowed.	No

The following uses data transmission from a device to the same device as an example to describe how to test the bandwidth and total duration.

```
ascend-dmi --bw -t d2d -d 0
```


Figure 9-2 Bandwidth test example

```
[root                          002 ~]# ascend-dmi --bw -t d2d -d 0 -q
Device to Device Test
Device  0: As       ).
-----------------------------------------------------------------
    Size(GB)  Execute Times  Bandwidth(GB/s)   Elapsed Time(us)
-----------------------------------------------------------------
   26.21            1     1525.590088          17183.12
   26.21            1     1544.480225          16972.96
   26.21            1     1539.688599          17025.78
   26.21            1     1541.082642          17010.38
   26.21            1     1541.443359          17006.40
   26.21            1     1543.283447          16986.12
   26.21            1     1540.625977          17015.42
   26.21            1     1542.132446          16998.80
   26.21            1     1539.639893          17026.32
   26.21            1     1541.294678          17008.04
   26.21            1     1541.660767          17004.00
   26.21            1     1544.651123          16971.08
   26.21            1     1535.153198          17076.08
   26.21            1     1544.551270          16972.18
   26.21            1     1540.394287          17017.98
   26.21            1     1541.617310          17004.48
   26.21            1     1538.237793          17041.84
   26.21            1     1540.818115          17013.30
   26.21            1     1539.348877          17029.54
   26.21            1     1542.096069          16999.20
   26.21            1     1540.656982          17015.08
   26.21            1     1540.705933          17014.54
   26.21            1     1541.807739          17002.38
   26.21            1     1542.384766          16996.02
   26.21            1     1539.410156          17028.86
-----------------------------------------------------------------
```

Table 9-5 Parameters for the bandwidth test output

Parameter	Description
Host to Device Test	Data flow direction of the bandwidth. The possible outputs are as follows: • Host to Device Test • Device to Host Test • Device to Device Test • Unidirectional Peer to Peer Test • Bidirectional Peer to Peer Test
Device X: Ascend *XXX*	*X* indicates the ID of the device to be tested, and *XXX* indicates the processor type. **0** indicates the source device, and **1** indicates the target device.
ID	**0**→**1** indicates the unidirectional P2P bandwidth from device 0 to device 1. **0**↔**1** indicates the bidirectional P2P bandwidth between device 0 and device 1.
Size(Bytes)	Size of the data to be transferred, in bytes.

Parameter	Description
Execute Times	Number of iterations.
Bandwidth(GB/s)	Bandwidth of the chip.
Elapsed Time(us)	Total execution duration.

Performance Test 2: Compute Test

Construct a matrix multiplication **A(m,k)*B(k,n)** and execute it a certain number of times. Calculate the AI Core compute and the real-time power at full computational load of the entire PU or processor based on the computation workload and the time taken to execute the matrix multiplication. For details about the parameters for compute test, see **Table 9-6**.

Table 9-6 Parameters for the compute test

Parameter	Description	Mandatory
[-f, --flops]	This parameter is used to test the compute capability of the entire PU or chip.	Yes
[-t, --type]	Specify the operator type. The value can be **fp16**, **fp32**, **hf32**, **bf16**, or **int8**. The default value is **fp16**.	No
[-d, --device]	Specify the device ID. The entire PU where the device ID is located is tested. The device ID is the logical ID of the Huawei Cloud AI chip. If this parameter is not specified, the compute of device 0 is returned by default.	No
[-et, --et, --execute-times]	Specify the number of times that the matrix multiplication is executed on a single AI Core of the chip. • Training: If this parameter is not specified, the default value **60** is used. In this case, the unit is hundred thousand, and the value ranges from **10** to **80**. • Inference: If this parameter is not specified, the default value **10** is used. In this case, the unit is million, and the value ranges from **10** to **80**.	No

The following shows an example of running the int8 operator and 6 million executions of compute on device 7.

```
ascend-dmi -f -t int8 -d 7 -et 60 -q
```

Figure 9-3 Compute test example

```
[root@de                    02 ~]# ascend-dmi -f -t int8 -d 7 -et 60 -q
-------------------------------------------------------------------------------
   Device        Execute Times       Duration(ms)       TOPS@INT8          Power(W)
-------------------------------------------------------------------------------
   6/7           900,000,000         1657               1563.549           776.300049
-------------------------------------------------------------------------------
```


Table 9-7 Parameters for the compute test output

Parameter	Description
Device	Device ID.
Execute Times	Number of times that a single AI Core performs matrix multiplication multiplied by the number of AI Cores.
Duration(ms)	Time taken to perform matrix multiplication multiple times.
TFLOPS@*FP16*	Compute obtained through compute test. FP16 is the specified operator run type.
Power(W)	Real-time power at full compute.

Performance Test 3: Power Consumption Test

Detect the power consumption of the entire PU by running the single-operator model. For details about the parameters for power consumption test, see **Table 9-8**.

```
ascend-dmi -p -h
```

Table 9-8 Parameters for power consumption test

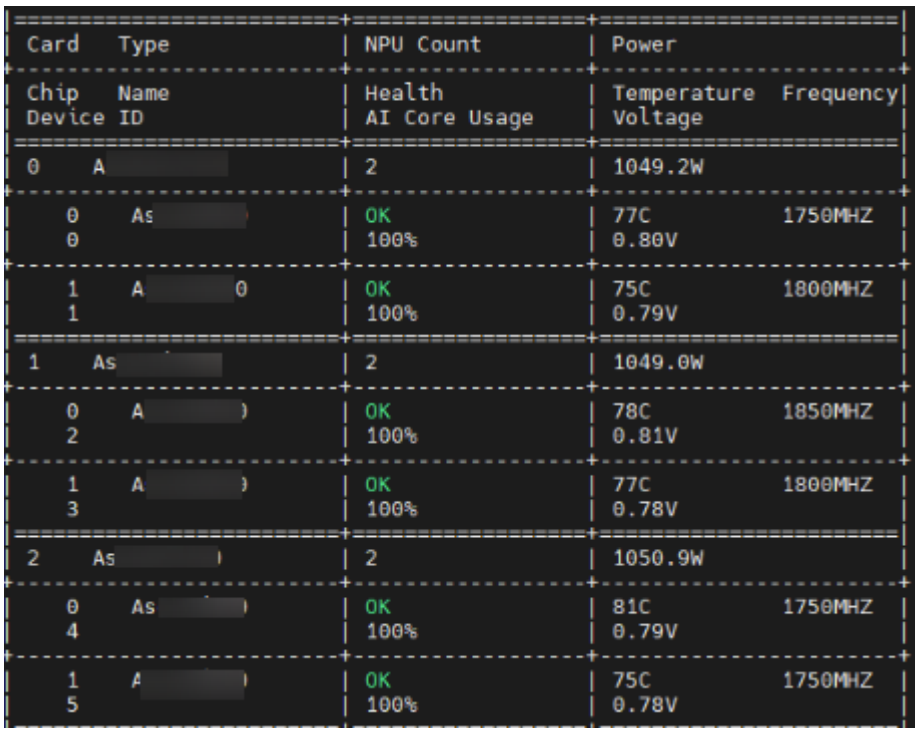
Parameter	Description	Mandatory
[-p, --power]	This parameter is used to test the power consumption of the entire PU.	Yes
[-t, --type]	Specify the operator type. The value can be **fp16** or **int8**. The default value is **fp16**.	No
[-pt, --pt, --pressure-type]	Specify the stress test type. • Currently, the following types are supported: – Estimated design power (**edp**): EDP power consumption stress test – Thermal design power (**tdp**): TDP power consumption stress test • It can be used together with **--dur**, **--it**, **--pm**, and **-q**. • It cannot be used together with **-t**. • If this parameter is not specified, the power consumption of the entire PU is tested by default.	No
[-dur, --dur, --duration]	Specify the running time. If this parameter is not specified, the default value **600** is used. The unit is second. The value ranges from **60** to **604800**.	No

Parameter	Description	Mandatory
[-it, --it, --interval-times]	Specify the interval for refreshing the screen information. If this parameter is not specified, the default value **5** is used. The unit is second. The value ranges from **1** to **5**.	No
[--skip-check]	If this parameter is specified, the device health check is skipped. Otherwise, the device health status is checked by default.	No
[-pm, --pm, --print-mode]	Specify the screen output mode. If this parameter is not specified, the default value **refresh** is used. Print modes: • **refresh**: Clear the historical information each time the information is printed. • **history**: Print the saved historical information. Note: In refresh mode, if there are a large number of chips, you are advised to reduce the font size so that all results are displayed on one screen. Otherwise, the display may be abnormal and some content may be repeatedly printed.	No

The following shows an example of executing for 60s at an interval of 5s in refresh mode.

```
ascend-dmi -p --dur 60 --it 5--pm refresh
```

Figure 9-4 Power consumption test example



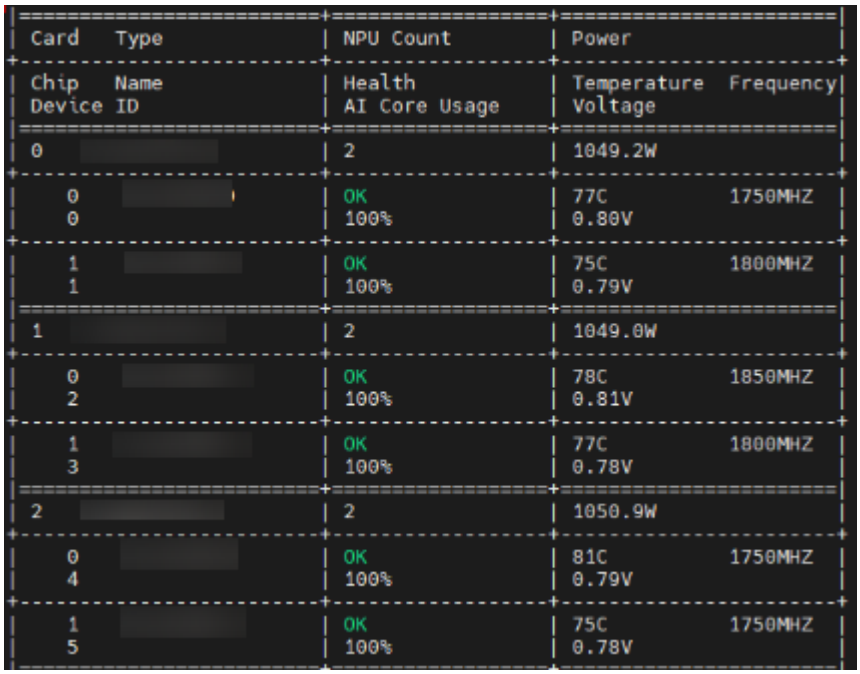


Table 9-9 Parameters for power consumption test output

Parameter	Description
Type	PU model
PU	PU ID
Chip	Processor No.
Name	Processor name
Type	Processor model
Chip Name	Chip name
NPU Count	Number of NPUs
Power	Actual power consumption of the entire PU or chip.
Health	Processor health status
Temperature	Current processor temperature
Device ID	Device logical ID of the processor
AI Core Usage	AI Core usage
Voltage	Current processor voltage
Frequency	Current processor frequency

Performance Test 4: Eye Pattern Test

Test the network and query the current signal quality. This function is used to query the specific signal quality. To check whether the signal quality of the current port is normal, perform signalQuality diagnosis. If CDR loopback has been configured for an NPU, disable the loopback before performing the eye pattern test. For details about the parameters for eye pattern test, see **Table 9-10**.

```
ascend-dmi --sq -h
```

Table 9-10 Parameters for the eye pattern test

Parameter	Description	Mandatory
[-sq, --sq, --signal-quality]	Query the signal quality of the PCIe, HCCS, and RoCE communication ports on the NPU.	Yes
[-d, --device]	Specify the device ID to be queried. If multiple device IDs are specified, use commas (,) to separate them. If this parameter is not specified, all NPUs on the device are queried by default.	No
[-t --type]	Specify the type of the communication port. Currently, HCCS and RoCE are supported. If multiple communication port types are specified, use commas (,) to separate them. If this parameter is not specified, the signal quality of the RoCE communication port is queried.	No

The following shows an example of querying the HCCS and RoCE signal quality of device 0 and device 1.

Figure 9-5 Eye pattern test example

```
[root                    2 ~]# ascend-dmi --sq -t hccs,roce -d 0,1
type: hccs
Prompt message: M*: macro port, L*: lane, S: SNR, H: HEH
Normal range: S(SNR) >= 400000 and H(HEH) >= 350
-------------------------------------------------------------------------------------------------------------
    device     signal-to-noise ratio
-------------------------------------------------------------------------------------------------------------
    0          M1:  L0: S:624980 H:404    L1: S:570601 H:405    L2: S:602362 H:406    L3: S:582752 H:395
               M2:  L0: S:625353 H:409    L1: S:580083 H:387    L2: S:628120 H:407    L3: S:596369 H:404
               M3:  L0: S:630788 H:393    L1: S:608124 H:397    L2: S:595847 H:381    L3: S:523765 H:389
               M4:  L0: S:544151 H:396    L1: S:609019 H:403    L2: S:561796 H:405    L3: S:513117 H:398
               M5:  L0: S:572574 H:399    L1: S:591882 H:391    L2: S:627122 H:397    L3: S:582290 H:401
               M6:  L0: S:568245 H:393    L1: S:553167 H:398    L2: S:599843 H:419    L3: S:533950 H:406
               M7:  L0: S:586303 H:402    L1: S:589030 H:380    L2: S:604690 H:390    L3: S:563846 H:384
-------------------------------------------------------------------------------------------------------------
    1          M1:  L0: S:645175 H:401    L1: S:599261 H:403    L2: S:589186 H:403    L3: S:569855 H:396
               M2:  L0: S:622159 H:400    L1: S:529333 H:396    L2: S:535708 H:398    L3: S:565269 H:406
               M3:  L0: S:618786 H:396    L1: S:575470 H:407    L2: S:610021 H:405    L3: S:558587 H:402
               M4:  L0: S:572800 H:401    L1: S:539484 H:405    L2: S:558558 H:387    L3: S:496511 H:378
               M5:  L0: S:578993 H:393    L1: S:570842 H:401    L2: S:628120 H:411    L3: S:575201 H:393
               M6:  L0: S:593619 H:403    L1: S:517064 H:386    L2: S:561252 H:396    L3: S:509361 H:399
               M7:  L0: S:592860 H:403    L1: S:555182 H:401    L2: S:582173 H:406    L3: S:528242 H:399
-------------------------------------------------------------------------------------------------------------


type: roce
Prompt message: M*: macro port, L*: lane, S: SNR, H: HEH
Normal range: S(SNR) >= 400000 and H(HEH) >= 350
-----------------------------------------------------------------------------------
    device                signal-to-noise ratio
-----------------------------------------------------------------------------------
    0                     M0:    L0: S:660394 H:410       L1: S:575225 H:404
                                 L2: S:674404 H:400       L3: S:576804 H:417
-----------------------------------------------------------------------------------
    1                     M0:    L0: S:683499 H:419       L1: S:659572 H:395
                                 L2: S:676381 H:414       L3: S:585714 H:402
-----------------------------------------------------------------------------------
```


Table 9-11 Parameters of the HCCS signal quality

Parameter	Description
type	Specify the type of the communication port.
device	Logical ID of an NPU.
M* (macro port)	Macro port, for example, **M0** and **M1** indicate macro ports 0 and 1, respectively.
L* (LANE)	Lane number of the HCCS link, for example, **L0** and **L1** indicate lane 0 and lane 1, respectively.
S (SNR)	Signal-to-noise ratio (SNR) of a lane.
H (HEH)	Half eye height of a lane.

Table 9-12 Parameters of the RoCE signal quality

Parameter	Description
type	Specify the type of the communication port.
device	Logical ID of the NPU.
M* (macro port)	Macro port, for example, **M0** indicates macro port 0.
S (SNR)	Signal-to-noise ratio (SNR) of a lane.
H (HEH)	Half eye height of a lane.
L* (LANE)	Lane number of the RoCE link, for example, **L0** and **L1** indicate lane 0 and lane 1, respectively.

Performance Test 5: Stream Test

One-click traffic testing and custom traffic testing are supported.

Table 9-13 Stream test

Test Name	Supported Traffic Mode	Usage
One-click traffic testing	Stream test in a CDR loopback or fiber optic circulator (loopback device)	Execute the one-click traffic test command. The Ascend DMI tool automatically sends and receives the streams of all lanes of the specified device. After a period of time, the streams are disabled and the results are queried.

Test Name	Supported Traffic Mode	Usage
Custom traffic testing	Stream test in a CDR loopback, fiber optic circulator (loopback device), and traffic test using direct connection between NPUs	Custom traffic testing is to separate each step of one-click traffic testing. You can flexibly control the TX and RX directions and specify the lanes for traffic testing.

There are three traffic testing modes:

- CDR loopback traffic test: A single device sends and receives traffic at the same time. This test can be used to check the signal quality from the physical SerDes port of the NPU to the CDR unit. Before starting traffic testing, ensure that the optical module is in position. Then, run the commands below to configure or disable CDR loopback.

  Run the commands below to configure CDR loopback. In the commands, **t** uses values **3** and **0** in sequence and **i** indicates the NPU ID.

  hccn_tool -i 0 -scdr -t 3

  hccn_tool -i 0 -scdr -t 0

  Run the commands below to disable CDR loopback. In the commands, **t** uses values **2** and **1** in sequence and **i** indicates the NPU ID.

  hccn_tool -i 0 -scdr -t 2

  hccn_tool -i 0 -scdr -t 1
- Traffic testing using a fiber optic circulator (loopback device) connected to an optical module: A single device sends and receives traffic at the same time. This method can be used to check the signal quality of the physical SerDes port of the NPU to the optical module. No loopback needs to be set.
- Traffic testing using direct connection between NPUs: After traffic is sent in the TX direction of the SerDes port on NPU A, data flows reach the SerDes port on NPU B through the tested link. NPU B compares the received data with the code pattern in the RX direction and collects statistics on bit errors. This method can be used to check the signal quality of the link between two NPUs (only custom traffic testing is supported).

For details about parameters for stream tests, see **Table 9-14**.

```
ascend-dmi --prbs-check -h
```

Table 9-14 Parameters for stream test

Parameter	Description	Mandatory
[-pc, --pc, --prbs-check]	This parameter is used for PRBS stream tests.	Yes

Parameter	Description	Mandatory
[-d, --device]	Specify the ID of the device for which the stream test is to be performed. • The device ID is the logical ID of the Huawei Cloud AI processor. If this parameter is not specified, the stream of all NPUs is tested. • Multiple device IDs can be specified at the same time. Use commas (,) to separate them.	No
[-dur, --dur, --duration]	Specify the duration of the stream test. • The value ranges from **3** to **10**, in seconds. • If this parameter is not specified, the default value **3** is used.	No
[--prbs-mode]	Whether to switch the traffic testing status. **EN**: enabled **DS**: disabled • The value is case-sensitive. • If **--prbs-mode** is set to **EN** or **DS**, the configuration takes effect in both the signal TX and RX directions, regardless of whether **--generator-pattern**, **--generator-lanes**, **--checker-pattern**, or **--checker-lanes** is specified. • If **--prbs-mode** is set to **EN**, **-generator-pattern**, **--checker-pattern**, **--generator-lanes**, and **--checker-lanes** can be specified. • If **--prbs-mode** is set to **DS**, the traffic testing stops. In this case, **-generator-pattern**, **--checker-pattern**, **--generator-lanes**, and **--checker-lanes** cannot be specified. • This parameter cannot be specified together with **--show** or **--clear**.	Yes
[--generator-pattern]	Specify the stream type of the TX end. • Currently, the following stream types are supported: **prbs7**, **prbs9**, **prbs10**, **prbs11**, **prbs15**, **prbs20**, **prbs23**, and **prbs31**. • If this parameter is not specified, the default value **prbs31** is used. • The parameter value is case-insensitive. For example, **prbs7** and **PRBS7** both are supported. • This parameter cannot be specified together with **--show** or **--clear**.	No

Parameter	Description	Mandatory
[--generator-lanes]	Specify the lane of the TX end. ● You can specify one or more lanes at a time. Use commas (,) to separate multiple lanes. If multiple lanes are specified, the lanes must be consecutive, for example, **0, 1, 2** or **2, 1, 3**. Non-consecutive lanes are not supported. ● If this parameter is not specified, all lanes are tested by default. ● This parameter cannot be specified together with **--show** or **--clear**. ● The value can be **0**, **1**, **2**, or **3**.	No
[--checker-pattern]	Specify the stream type of the RX end. ● Currently, the following types are supported: **prbs7**, **prbs9**, **prbs10**, **prbs11**, **prbs15**, **prbs20**, **prbs23**, and **prbs31**. ● If this parameter is not specified, the default value **prbs31** is used. ● The parameter value is case-insensitive. For example, **prbs7** and **PRBS7** both are supported. ● This parameter cannot be specified together with **--show** or **--clear**.	No
[--checker-lanes]	Specify the lane of the RX end. ● You can specify one or more lanes at a time. Use commas (,) to separate multiple lanes. If multiple lanes are specified, the lanes must be consecutive, for example, 0, 1, 2 or 2, 1, 3. Non-consecutive lanes are not supported. ● If this parameter is not specified, all lanes are tested by default. ● This parameter cannot be specified together with **--show** or **--clear**. ● The value can be **0**, **1**, **2**, or **3**.	No
[-show, --show, --show-diagnostic-info]	Display the stream test result. ● This parameter cannot be specified together with **--clear**, **--prbs-mode**, **--generator-pattern**, **--generator-lanes**, **--checker-pattern**, and **--checker-lanes**. ● After the information is displayed, the result of the current stream test is cleared.	No

Parameter	Description	Mandatory
[-clear, --clear, --clear-diagnostic-info]	Clear the stream test result. • This parameter cannot be specified together with **--show**, **--prbs-mode**, **--generator-pattern**, **--generator-lanes**, **--checker-pattern**, and **--checker-lanes**. • You can specify other parameters at the same time.	No

The following is an example of one-click traffic testing:

```
ascend-dmi -pc -d 9 --pattern prbs15 -dur 5
```

Figure 9-6 One-click traffic testing example

```
[root@dev                          ~]# ascend-dmi -pc -d 5 --pattern prbs15 -dur 5
This operation will make network port on devices down, please make sure no business is running on devices.
 Do you want to continue?(Y/N)y
PRBS15 on device 5:
-------------------------------------------------------------------------------------------------------
lane              error count         error rate          alos              time(ms)
-------------------------------------------------------------------------------------------------------
0                 67092480            0.0251382604%       0                 5024
1                 67092480            0.0251319427%       0                 5026
2                 67092480            0.0251509054%       0                 5022
3                 67092480            0.0251382604%       0                 5025
-------------------------------------------------------------------------------------------------------
```

Table 9-15 Parameters of one-click traffic testing output

Parameter	Description
device	Logical ID of the NPU.
lane	Lane ID of the RoCE link.
error count	Number of bit errors. The maximum value is **67092480**, indicating full bit errors.
error rate	Bit error rate. If the bit error rate is less than 10-5, the signal quality is normal.
alos	The value **0** indicates normal, and the value **1** indicates that the input signal amplitude is too low.
times	Traffic testing duration.

The following shows an example of custom traffic testing:

```
# Enable the stream test on Device8 and Device9.
ascend-dmi -pc --clear --device 8,9 -q
# On Device8 and Device9, the TX ends are lane 0 and lane 1, and the code pattern is PRBS20. The RX ends are lane 2 and lane 3, and the code pattern is PRBS23.
ascend-dmi -pc --prbs-mode EN -q --device 8,9 --generator-pattern prbs20 --generator-lanes 0,1--checker-pattern prbs23 --checker-lanes 2,3
# Display the stream test results of Device8 and Device9.
```



```
ascend-dmi -pc --show-diagnostic-info -d 8,9 -q
# Disable traffic testing on Device8 and Device9.
ascend-dmi -pc --prbs-mode DS -d 8,9 -q
# Clear the traffic testing results on Device8 and Device9.
ascend-dmi -pc --clear-diagnostic-info -d 8,9 -q
```

Figure 9-7 Custom traffic testing example

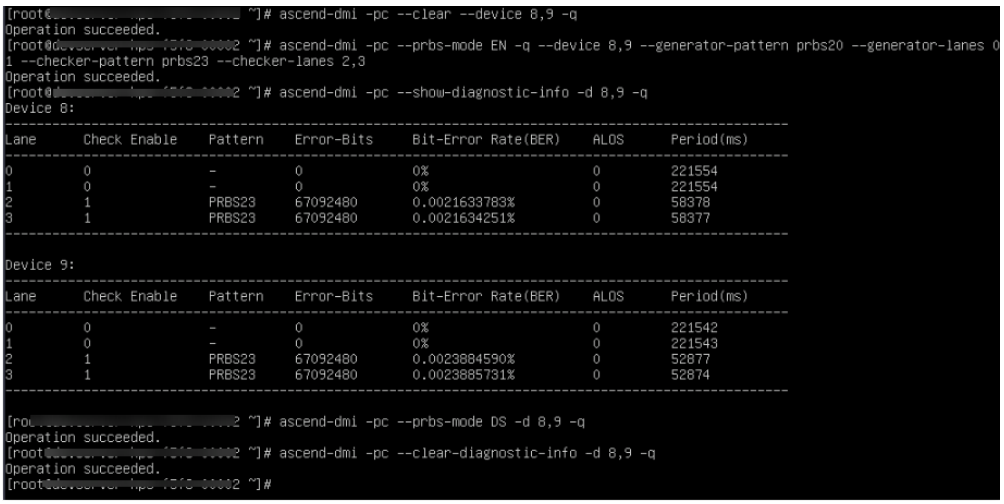


Table 9-16 Parameters of custom traffic testing output

Parameter	Description
Lane	Lane ID of the RoCE link.
Check Enable	Check status of the RX end. The value **0** indicates disabled and **1** indicates enabled.
Pattern	Code pattern of check in the RX direction.
Error-Bits	Number of bit errors. The upper limit is **67092480** (full bit errors).
Bit-Error Rate (BER)	Bit error rate, which is the number of bit errors divided by the total number of transmitted bits multiplied by 100%.
ALOS	The value must be **0** for normal traffic testing. If the value is **1**, the signal amplitude is too low. If no traffic is testing, this parameter is meaningless.
Period	Time when the traffic testing is controlled or the check result is read last time.

Performance Test 6: Software and Hardware Version Compatibility Test

The software and hardware compatibility tool obtains the hardware information, architecture, driver version, firmware version, and software version. For details about the parameters for software and hardware compatibility tests, see **Table 9-17**.

```
ascend-dmi -c -h
```


Table 9-17 Software and hardware version compatibility test

Parameter	Description	Mandatory
[-c, --compatible]	This parameter is used to check the software and hardware version compatibility. • If driver 22.0.0 or CANN 6.2.RC1 or later has been installed, the **-c** parameter is used to check the compatibility between the NPU firmware and driver, and between the driver and CANN. • If the driver version is earlier than 22.0.0 and the CANN version is earlier than 6.2.RC1, the **-c** parameter is used to check whether the corresponding driver, firmware, and software package are installed.	Yes
[-p, --path]	You can specify the installation path of the CANN software package to be tested. If the installation path is not specified, the default installation path is used. Example command for specifying the installation path of the software package: **ascend-dmi -c -p /home/xxx/Ascend**	No

The following shows an example of the hardware and software version compatibility test:

```
ascend-dmi -c
```

Figure 9-8 Software and hardware version compatibility test example

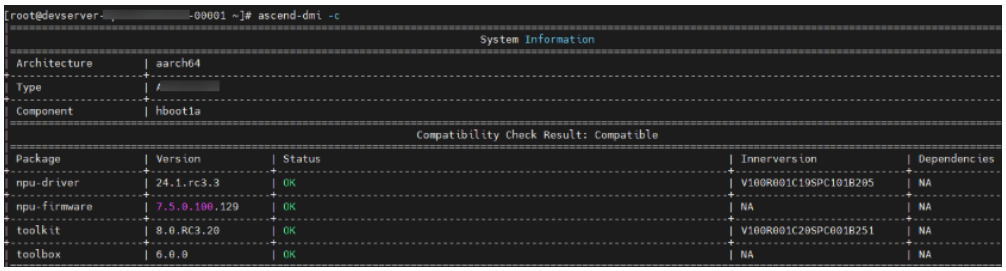


Table 9-18 Parameters for software and hardware version compatibility test output

Parameter	Description
System Information	System information
Architecture	Architecture
Type	PU or chip model
Compatibility Check Result	Compatibility check result
Package	Package name
Version	Version
Status	Status. Possible values are as follows: • **OK**: compatible • **INCOMPATIBLE PACKAGE**: incompatible • **NA**: Unknown status. The software version may fail to be obtained. Non-root users cannot query the firmware compatibility, and the status of NPU firmware is displayed as **NA**.
Innerversion	Internal version number
Dependencies	Dependencies

Fault Diagnosis

View the parameters of the fault diagnosis command.

```
ascend-dmi --dg -h
```

Table 9-19 Fault diagnosis parameters

Parameter	Description	Mandatory
[-dg, --dg, --diagnosis]	This parameter is used to perform a fault diagnosis test on the entire PU.	Yes
[-i, --items]	Specify the diagnosis check items. • You can specify one or more of driver, CANN, device, network, bandwidth, AI FLOPs, HBM, and signalQuality. Use commas (,) to separate multiple items. • If this parameter is not specified, check items except AI Core and PRBS are diagnosed by default.	No

Parameter	Description	Mandatory
[-d, --device]	Specify the ID of the device to be diagnosed. The device ID is the logical ID of the Huawei Cloud AI chip. ● You can specify one or more device IDs. Use commas (,) to separate them. ● If no device ID is specified, the diagnosis results of all devices are returned by default.	No
[-r, --result]	Specify the path for storing the stress test result and information collection result, for example, **/test**. The specified path must meet security requirements and cannot contain the wildcard (*). ● To specify a path for storing results, you need to create the **ascend_check** folder in the specified path. The path specified by user **root** will be created in the root directory, and the path specified by non-root users will be created in the **$HOME** directory. ● If the path is not specified, the results will be stored in the default path, which is **/var/log/ascend_check** for user **root** and **$HOME/var/log/ascend_check** for non-root users.	No
[-s, --stress]	This parameter is used to perform stress tests. Currently, the following stress tests are supported: on-chip memory stress test, AI Core stress test, P2P stress test, and power consumption stress test. ● If on-chip memory and power consumption are included, this parameter can be used together with the **-st** parameter. The time for performing the stress test is specified by **--st**. ● If AI Core check item is included, this parameter can be used together with the **-sc** parameter. The number of performed stress tests is specified by **--sc**. ● If bandwidth check is specified, this parameter can be used together with **-t** to perform a P2P stress test.	No
[-st, --st, --stress-time]	Specify the time for the EDP and TDP stress tests. ● The value ranges from **60** to **604800**, in seconds. ● This parameter must be used together with **[-s, --stress]** when the EDP and TDP stress tests are included. ● This parameter must be used together with **[-s, --stress]** when the on-chip memory diagnosis is included.	No

Parameter	Description	Mandatory
[-fmt, --fmt, --format]	Specify the output format. The value can be **normal** or **json**. • If not specified, the default value **normal** is used. • If this parameter is set to **json**, the stress test result will be stored in the **ascend_check/environment_check_before.txt** file. However, if **json** format is not specified, the fault diagnosis result will not be stored.	No
[-h, --help]	View the parameters of the fault diagnosis command.	No

Fault Diagnosis 1: Network Diagnosis

Diagnose the health status of the network and output the diagnosis result.

```
# Example of diagnosing the network health status of Device0
ascend-dmi -dg -i network -d 0
```

Figure 9-9 Network diagnosis example

The parameters in the command output are as follows:

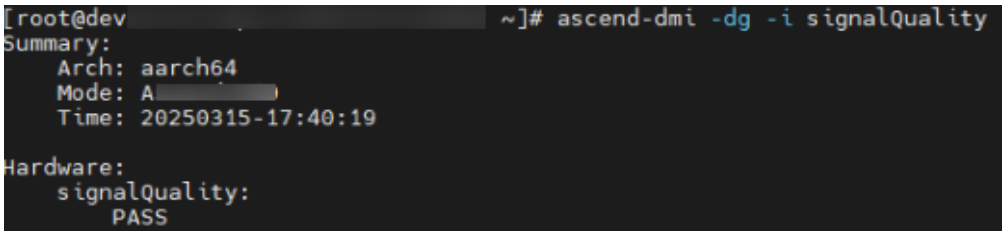
- **PASS**: The network is healthy.
- **SKIP**: The current product form does not support this check item.
- **INFO**: The network check result is informational.
- **WARN**: The network check result is an alarm.
- **FAIL**: The network check fails.

Fault Diagnosis 2: SignalQuality Diagnosis

Diagnose the signal quality and output the diagnosis result.

```
# Example of SignalQuality diagnosis
ascend-dmi -dg -i signalQuality -q
```


Figure 9-10 SignalQuality diagnosis example



The parameters in the command output are as follows:

- **PASS**: The HCCS and RoCE communication ports on the NPU pass the test, and the signal quality is normal.
- **SKIP**: The current device does not support eye pattern diagnosis.
- **IMPORTANT_WARN**: Important warning. The signal quality of one or more of the HCCS and RoCE ports is abnormal. Contact Huawei engineers.
- **FAIL**: The eye pattern detection fails.

Fault Diagnosis 3: On-Chip Memory Diagnosis

Diagnose the high bandwidth memory and output the diagnosis result.

```
# Example of on-chip memory diagnosis
ascend-dmi -dg -i hbm
```

Figure 9-11 On-chip memory diagnosis example

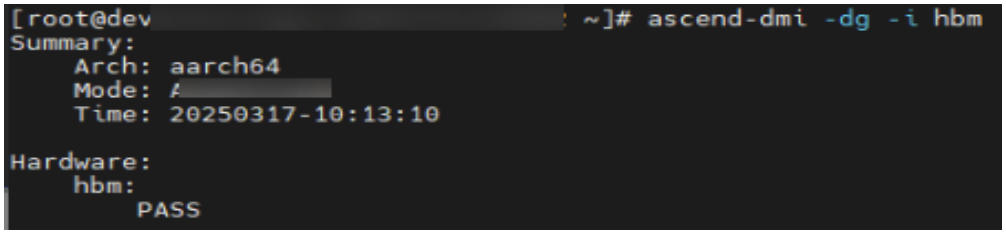


Table 9-20 Parameters for on-chip diagnosis output

Output Status	Description
PASS	The on-chip memory passes the check and is normal.
SKIP	The current hardware form does not support on-chip memory check.
GENERAL_WARN	There are isolated pages with historical multi-bit errors. The NPU chip triggering the alarm has a health management fault code of 0x80E18401. The NPU chip can still be used.
IMPORTANT_WARN	The number of isolated pages now differs from the previous count. Restart to reset the NPU chip.

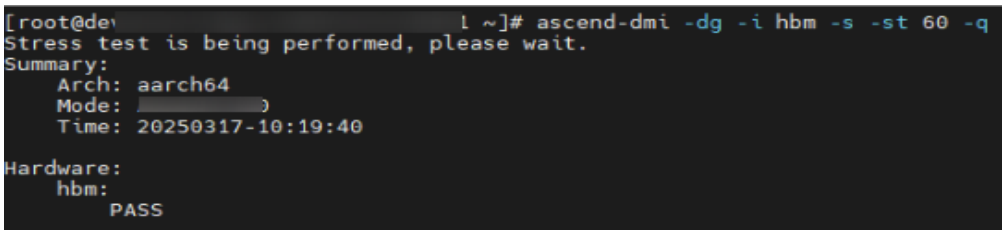
Output Status	Description
EMERGENCY_WARN	• There are too many isolated pages with historical multi-bit errors, and too many device isolation rows. The NPU chip has a health management fault code of 0x80E18402. Replace the faulty part. • If there are four or more isolation rows in the same stack but different banks, the device is at high risk. Replace the spare part. • If the number of isolation rows in the same stack, same SID, and different PCs is greater than or equal to four, the device is at high risk. Replace the spare part. • If the number of isolation rows in the same stack, SID, PC, and bank is greater than 16, the device is at high risk. Replace the spare part. • If the number of different addresses is greater than five (excluding adjacent error addresses of four bits or less) in the same stack, SID, PC, and bank, the device is at high risk. Replace the spare part.
FAIL	The on-chip memory detection fails. Contact Huawei engineers.

Fault Diagnosis 4: On-Chip Memory Stress Test

Run a stress test on the high-bandwidth memory and output the diagnosis result.

```
# Example
ascend-dmi -dg -i hbm -s -st 60 -q
```

Figure 9-12 On-chip memory diagnosis example



Output parameters:

- **PASS**: The on-chip memory stress test is passed.
- **SKIP**: The current device does not support the on-chip memory stress test.
- **FAIL**: The on-chip memory stress test fails because new multi-bit isolation pages are added. The software execution fails.

Fault Diagnosis 5: On-Chip Memory High-Risk Address Stress Test

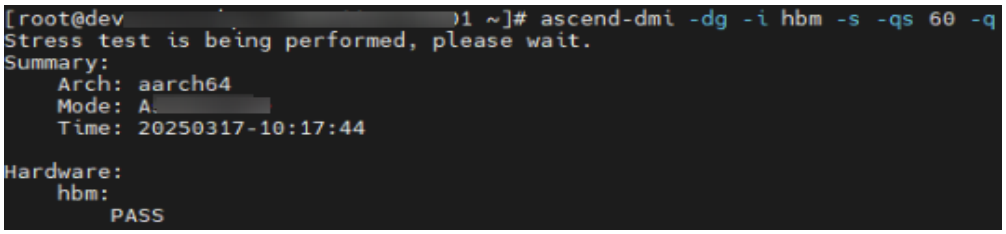
Run a stress test on the high-risk addresses of the high bandwidth memory and output the diagnosis result.

Table 9-21 Parameters for on-chip memory high-risk address stress tests

Parameter	Description	Mandatory
[-s, --stress]	This parameter is used to perform stress tests. Currently, the following stress tests are supported: on-chip memory stress test, AI Core stress test, P2P stress test, and power consumption stress test.	Yes
[-qs, --qs, --quick stress]	Specify the range of fast stress test for high-bandwidth memory high-risk addresses. • The value ranges from **0** to **100**. The recommended value is **100**. • If the value is **0**, the fast stress test is performed on all high-bandwidth memory addresses by default. • If HBM diagnosis is included, this parameter must be used together with **[-s, --stress]** and cannot be used together with **[-st, --st, --stress-time]** or **[--sc, --stress-count]**.	Yes

```
# Example of stress test on on-chip memory high-risk addresses
ascend-dmi -dg -i hbm -s -qs 60-q
```

Figure 9-13 On-chip memory high-risk address stress test example



Output parameters:

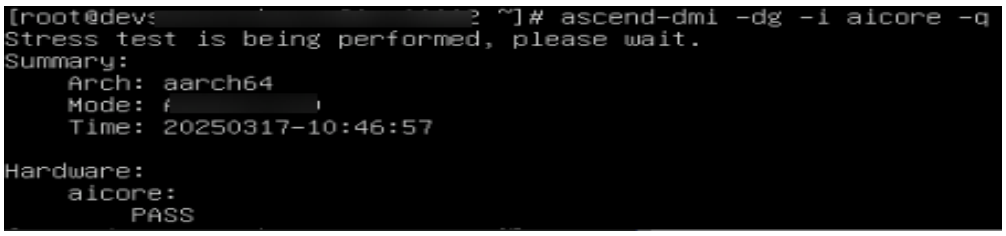
- **PASS**: The high-bandwidth memory high-risk address fast stress test is passed. No isolation page is added.
- **SKIP**: The current device does not support the on-chip memory high-risk address stress test.
- **FAIL**: The high-bandwidth memory high-risk address fast stress test fails. New isolation pages are added.

Fault Diagnosis 6: AI Core Diagnosis

Diagnose the AI Core error and output the diagnosis result.

```
# Example of AI Core diagnosis
ascend-dmi -dg -i aicore -q
```


Figure 9-14 AI Core diagnosis example



Output parameters:

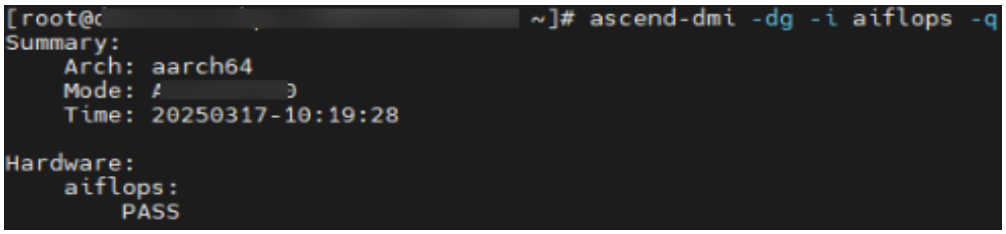
- **PASS**: The diagnosis result is normal.
- **SKIP**: The diagnosis is performed by a non-root user and AI Core diagnosis is not supported.
- **EMERGENCY_WARN**: emergency warning. Replace the hardware.
- **FAIL**: AI Core diagnosis fails. Contact Huawei engineers.

Fault Diagnosis 7: AI FLOPs Diagnosis

Diagnose the chip compute and output the test result.

```
# Example of AI FLOPs diagnosis
ascend-dmi -dg -i aiflops -q
```

Figure 9-15 AI FLOPs diagnosis example



Output parameters:

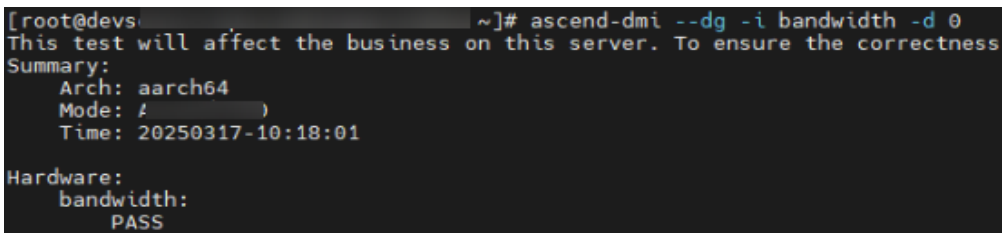
- **PASS**: The compute test result is normal (greater than the reference value).
- **WARN**: The chip overheats during the compute test.
- **FAIL**: The compute test fails. The test result is smaller than the reference value.

Fault Diagnosis 8: Bandwidth Diagnosis

Diagnose the local bandwidth and output the diagnosis result.

```
# Example of performing bandwidth diagnosis on Device0
ascend-dmi --dg -i bandwidth -d 0
```

Figure 9-16 Bandwidth diagnosis example



Output parameters:

- **PASS**: The bandwidth test result is normal.
- **FAIL**: The bandwidth test fails. The test result is smaller than the reference value. Contact Huawei engineers.

Fault Diagnosis 9: P2P Stress Test

Checks whether the HCCS communication link from the source device to the target device has hardware faults and outputs the test result.

Table 9-22 Parameters for P2P stress tests

Parameter	Description	Mandatory
[-s, --stress]	This parameter is used to perform stress tests. Currently, the following stress tests are supported: on-chip memory stress test, AI Core stress test, P2P stress test, and power consumption stress test. • If bandwidth check is specified, this parameter can be used together with **-s** to perform a P2P stress test.	Yes
[-t, --type]	Specify the data flows to be tested. • This parameter takes effect only when **item** is set to **bandwidth** and **-s** is specified, indicating that the P2P pressure test is performed. • Currently, only the P2P bandwidth type is supported. **p2p**: Data is transferred from the specified source device to the target device. The transfer rate and total duration are tested.	Yes

```
# Example of P2P stress test
ascend-dmi -dg -i bandwidth --type p2p -s
```

Figure 9-17 P2P stress test example

```
[root@                              ~]# ascend-dmi -dg -i bandwidth --type p2p -s
This test will affect the business on this server. To ensure the correctness and
Summary:
    Arch: aarch64
    Mode: (          )
    Time: 20250317-10:25:41

Hardware:
    bandwidth:
        PASS
```

Output parameters:

- **PASS**: The stress test is passed, and the result is normal.
- **SKIP**: The current device does not support the P2P stress test.
- **EMERGENCY_WARN**: emergency warning. The stress test fails. Contact Huawei engineers to replace the hardware.

- **FAIL**: P2P stress test fails. Contact Huawei engineers.

Fault Diagnosis 10: Power Consumption Stress Test

Perform the EDP/TDP power consumption stress test and output the diagnosis result.

```
# Example of power consumption stress test
ascend-dmi -dg -i edp -s -st 60 -q
ascend-dmi -dg -i tdp -s -st 60 -q
```

Figure 9-18 TDP power consumption stress test example

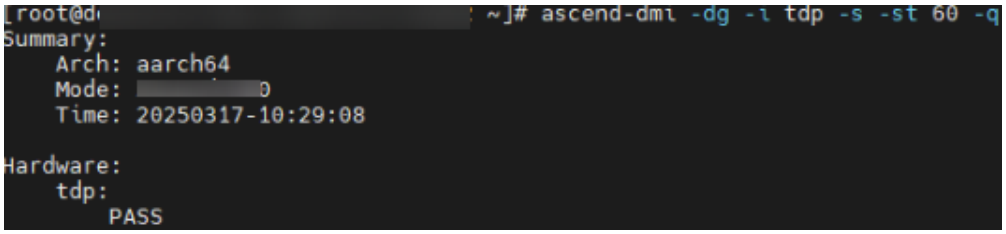
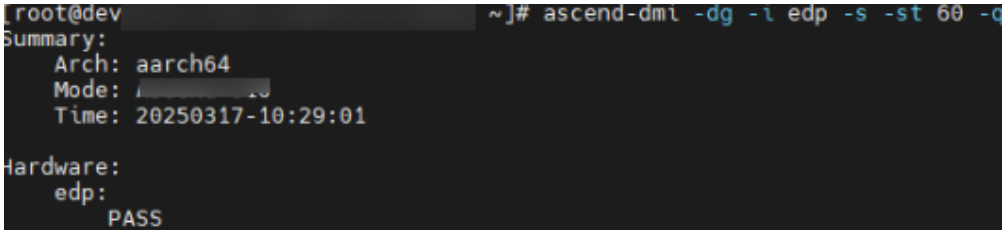


Figure 9-19 EDP power consumption stress test example



Output parameters:

- **PASS**: The power consumption stress test result is normal.
- **SKIP**: The current device does not support the power consumption stress test.
- **IMPORTANT_WARN**: A chip alarm is generated during the stress test. Handle the alarm based on the description. If the fault persists, contact Huawei engineers.
- **FAIL**: The power consumption stress test fails. Contact Huawei engineers.

9.4 Enabling HCCL Communication Operator-Level Re-execution for Supernodes

Scenario

To address the high failure rate of optical modules under Snt9b23 supernodes, the stability and reliability of the system are improved by introducing a re-execution mechanism at the Huawei Collective Communication Library (HCCL) communication operator level.

HCCL, a distributed communication library designed by Huawei for Ascend AI processors, aims to optimize efficient collaboration between multiple devices and accelerate distributed training of deep learning models, applicable to AI scenarios

where large-scale compute is required. In distributed training, HCCL is responsible for coordinating data synchronization (such as gradient aggregation and parameter update) between multiple Ascend processors, reducing communication overheads and improving training efficiency.

Constraints

- Only Snt9b23 supernodes are supported.
- Enabling operator re-execution slightly affects the performance.
- Re-execution depends on the VPC plane (non-parameter plane) network for status negotiation within the communication domain. If the VPC planes are different, re-execution cannot be performed.
- For the HCCS plane, if the link is not recovered and the route is not converged, re-execution cannot be performed.
- Re-execution depends on that all PUs in a communication domain stop at the same communication operator when a fault occurs. Otherwise, re-execution cannot be performed. The success rate is about 95%.
- Using the communication operator in inplace mode may cause UserIn data to be polluted, affecting the reliability of re-execution. Although 80% of communication operators can be re-executed in the inplace mode, there are exceptions, for example, for **all_reduce**, **all_gather**, and **reduce_scatter** operators in the Torch framework.
- For RoH/RoCE failover (lane borrowing) caused by intermittent disconnection or link disconnection, re-execution can be performed only once in the same communication domain, and switchback is not supported. During the failover, services can be continued. However, you should save checkpoints and rectify faults in a timely manner.
- The following table lists the supported HCCL re-execution scope for the current Ascend execution mode.

Table 9-23 HCCL re-execution scope

Mode	HCCL Communication Operator Unfolding Mode	Supported
Single-operator	Stars	Supported
	FFts+	Supported
	AI CPU unfolding	Supported
	Integrate communication and computing (mc2)	Not supported
Graph mode	Full POD mode, in which communication operators are integrated as expanded tasks.	Not supported Full POD mode, in which HCCL is not involved in the graph execution process and cannot be re-executed.
	AI CPU unfolding	Supported

Principles

The connection system of the Snt9b23 supernode mainly includes two transmission planes: HCCS plane and RoH/RoCE plane.

On the HCCS plane, the optical interconnection technology is used between L1-1520 and L2-1520. On the RoH/RoCE plane, optical interconnection is used for parts beyond the NPU range. The fault rate of the electrical interconnection domain is relatively low. Therefore, this mechanism is mainly used to handle optical module faults in the optical interconnection domain. Specifically:

- Faulty optical module between L1-1520 and L2-1520 on the HCCS plane
- Faulty optical module of the Snt9b23 out of the RoH/RoCE plane

HCCS plane

For the HCCS plane, if the optical module between L1 and L2 is intermittently disconnected or disconnected, the 1520 device automatically switches the path (provided that multiple paths exist). However, link disconnection may cause packet loss and further service interruption. In this case, the framework layer rolls back to the previous checkpoint for resumable training. By introducing the HCCL re-execution mechanism, returning to the checkpoint for resumable training may be effectively reduced after 1520 completes path switching, further improving service continuity and reliability.

RoH/RoCE plane

For the RoH/RoCE plane, the protocol has a built-in retransmission mechanism at the transport layer, which can rectify packet loss or intermittent disconnection. However, the reliability of this mechanism is still limited. To enhance the overall reliability, the re-execution mechanism is introduced at the HCCL layer. When an intermittent disconnection lasts for more than 30 seconds or a link disconnection occurs, the system establishes a new transmission path (lane borrowing) and starts the re-execution process at the operator level, ensuring service stability.

Parameter Configuration (HCCL_OP_RETRY_ENABLE)

The environment variable **HCCL_OP_RETRY_ENABLE** is used to configure whether to enable HCCL operator re-execution. Re-execution refers to the process in which HCCL attempts to re-execute the communication operator when the communication operator reports an SDMA or RDMA CQE error. This feature can effectively avoid communication interruption caused by hardware intermittent disconnection and improve communication stability.

The re-execution feature can be configured in the communication domains at the following physical layers:

- **L0**: communication domain within a server
- **L1**: communication domain between servers
- **L2**: communication domain between supernodes

Configuration:

Before running a training job, run the following command on the server node:


```
export HCCL_OP_RETRY_ENABLE="L0:0, L1:1, L2:1"
```

Table 9-24 Parameters

Parameter	Description	Value Range	Default Value	Recommended Value
L0	Communication domain within a server	• **0**: Re-execution is disabled for communication tasks in the communication domain within a server. • **1**: Re-execution is enabled for communication tasks in the communication domain within a server.	0	0
L1	Communication domain between servers	• **0**: Re-execution is disabled for communication tasks in the communication domain between servers. **0** is the default value. • **1**: Re-execution is enabled for communication tasks in the communication domain between servers.	0	1
L2	Communication domain between supernodes	• **0**: Re-execution is disabled for communication tasks in the communication domain between supernodes. **0** is the default value. • **1**: Re-execution is enabled for communication tasks in the communication domain between supernodes.	0	1

Note:

- When **L2** is set to **1**, the communication between supernodes can be performed using the standby device NIC when the device NIC is faulty. The standby NIC is the NIC of the other die in the same NPU.
- If the communication domain is created based on the ranktable, you need to configure the standby NIC using the **backup device ip** parameter in the ranktable file.
- If the communication domain is created based on the root broadcast, the two dies of the same NPU are automatically configured as the standby NICs of each other. No manual configuration is required.

Parameter Configuration (HCCL_OP_RETRY_PARAMS)

The environment variable **HCCL_OP_RETRY_ENABLE** is used to configure the parameters for re-executing the HCCL operator, including the maximum number of re-executions, the waiting time for the first re-execution, and the interval between two re-executions.

Configuration example

```
export HCCL_OP_RETRY_PARAMS="MaxCnt:3, HoldTime:5000, IntervalTime:1000"
```

Table 9-25 Parameters

Parameter	Description	Type	Value Range	Default Value	Unit	Recommended Value
MaxCnt	Maximum number of re-executions	uint32	[1, 10]	3	Count	Retain the default value **3**.
HoldTime	Waiting time from the time when a communication operator execution failure is detected to the time when the operator is re-executed for the first time	uint32	[0, 60000]	5000	ms	Retain the default value **5000**.

Parameter	Description	Type	Value Range	Default Value	Unit	Recommended Value
IntervalTime	Interval between two re-executions	uint32	[0, 60000]	1000	ms	Retain the default value **1000**.

Constraints:

This environment variable takes effect only when the HCCL re-execution feature is enabled (at any layer) using the **HCCL_OP_RETRY_ENABLE** environment variable.

10 Collecting Logs of Lite Servers

10.1 Collecting and Uploading NPU Logs

Description

When an NPU on a Lite Server is faulty, you can use this solution to quickly deliver an NPU log collection task. The collected logs are stored in a specified directory on Lite Server. Once authorized, your logs will be automatically uploaded to the OBS bucket provided by Huawei Cloud for fault locating and analysis.

The following logs can be collected:

- Device logs: system logs on the control CPU of the device, event-level system logs, system logs on non-control CPUs, and black box logs. When a device is abnormal, crashes, or has performance problems, you can accurately locate the root cause, shorten the troubleshooting time, and improve the O&M efficiency.
- Host logs: kernel message logs on the host, system monitoring files on the host, OS log files on the host, and kernel message log files saved on the host when the system crashes. O&M engineers can view the host monitoring data and quickly determine whether the problem is caused by the application or underlying host resources, such as full CPUs, used up memory, and full disks. Accurate locating can greatly improve troubleshooting efficiency.
- NPU environment logs: logs collected using tools such as npu-smi and hccn. By collecting such logs, you can improve O&M efficiency, ensure system stability, optimize resource usage, and facilitate root cause analysis.

Constraints

- One-click log collection is supported only for Snt9b and Snt9b21 common nodes and Snt9b23 supernodes. Manual log collection is supported for 300I Duo, Snt9b, Snt9b23, and Snt9b21.
- You can select a maximum of 50 common nodes or supernode child nodes for the same task.
- Only one task can be executed on a node at the same time. The task cannot be interrupted once started. Plan the task priority.

- Ensure that no services are running on the node. The command execution in the log collection task may interrupt or cause exceptions for the current services.
- Ensure that the space of the directory for collecting logs is greater than 1 GB.

Prerequisites

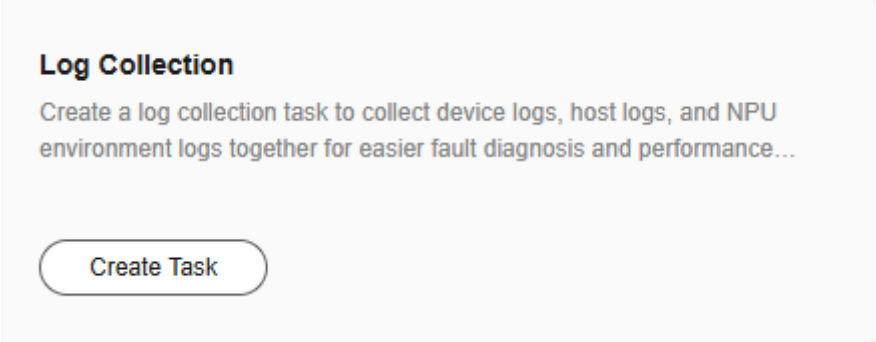
Quick log collection depends on the AI plugin preinstalled on the Lite Server. If the plugin is not installed, install it by referring to **Installing the AI Plugin for a Lite Server**.

One-Click Log Collection

1. Log in to the **ModelArts console**. In the navigation pane, choose **Resource Management** > **Lite Servers**. Click the **Task Center** tab.
   - New console: In the navigation pane, choose **Resource Management** > **Lite Compute Resources** > **Lite Servers**.
   - Old console: In the navigation pane, choose **Resource Management** > **Lite Servers**.

2. Click **Create Task** in the upper right corner. On the displayed **Job Templates** page, locate **Log Collection**, and click **Create Task**.

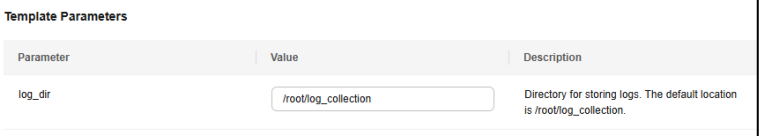
   Figure 10-1 Task template

3. On the **Log Collection** page, enter the task name and description. Set server model and node type, select collection items, select the term of use, and click **Create now**.

   Table 10-1 Parameters for creating a task

Parameter	Description
Name	The system automatically generates a task name. You can change the name as required.
Description	Enter the task description for quick search.

Parameter	Description
Template Parameters	Enter the node directory on the Lite Server for storing logs. The default value is **/root/log_collection**. 
Server Model	Snt9b and Snt9b21 common nodes and Snt9b23 supernodes are supported.
Collection Items	Select **Device side log**, **Host side log**, **NPU environment log**, or all of them. • Device logs: system logs on the control CPU of the device, event-level system logs, system logs on non-control CPUs, and black box logs. • Host logs: kernel message logs on the host, system monitoring files on the host, OS log files on the host, and kernel message log files saved on the host when the system crashes. • NPU environment logs: logs collected using tools such as npu-smi and hccn.

4. Enable **Log upload** to upload collected logs to OBS for Huawei engineers to analyze logs. Click **Create now**.
5. View the task execution status in the **Task Center** tab.
6. Click the task name to access its details page, where you can view the task details.
7. On the task details page, locate the target node and click **View Logs** in the **Operation** column. In the displayed window on the right, view the detailed log about task execution. In the **Task Center** tab, you can view all log collection results and check whether logs are collected and uploaded.

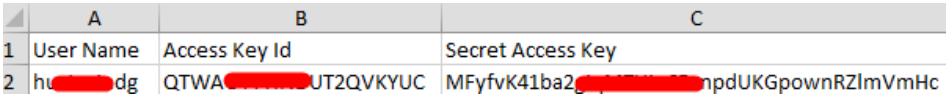
Manually Collecting Logs

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 10-2 credential.csv

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

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

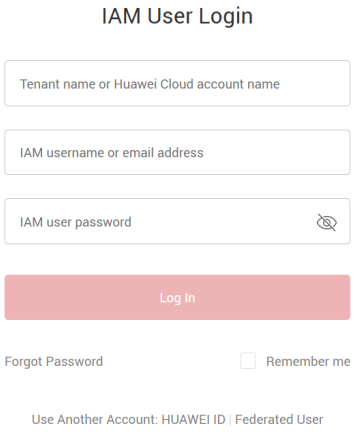
a. Log in to the **Huawei Cloud console**.
b. Hover the cursor over the username in the upper right corner and choose **My Credentials** from the drop-down list.
c. In the navigation pane on the left, choose **Access Keys**.
d. Click **Create Access Key**.
e. Download the key and keep it secure.

2. Prepare the tenant ID and IAM ID for OBS bucket configuration.

   Send the prepared information to technical support, who 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 10-3 Tenant and IAM accounts



3. Collect the logs and upload the script.

   Modify the **NpuLogCollection** parameter in the script below. Replace **ak**, **sk**, and **obs_dir** with the values obtained in the previous steps. For the 300IDuo model, set **is_300_iduo** to **True**. Upload the script to the node whose NPU logs need to be collected.

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

class NpuLogCollection(object):
    NPU_LOG_PATH = "/var/log/npu_log_collect"
    SUPPORT_REGIONS = ['cn-southwest-2', 'cn-north-9', 'cn-east-4', 'cn-east-3', 'cn-north-4', 'cn-south-1']
    OPENSTACK_METADATA = "http://169.254.169.254/openstack/latest/meta_data.json"
    OBS_BUCKET_PREFIX = "npu-log-"

    def __init__(self, ak, sk, obs_dir, is_300_iduo=False):
        self.ak = ak
        self.sk = sk
```

```
        self.obs_dir = obs_dir
        self.is_300_iduo = is_300_iduo
        self.region_id = self.get_region_id()
        self.card_ids, self.chip_count = self.get_card_ids()

    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):
        # 300IDUO does not support
        hccn_tool_log_shell = "echo {npu_network_info}\n" \
                              "for i in {npu_card_ids}; do hccn_tool -i $i -net_health -g >> {npu_log_path}/npu-
smi_net-health.log ;done\n" \
                              "for i in {npu_card_ids}; do hccn_tool -i $i -link -g >> {npu_log_path}/npu-
smi_link.log ;done\n" \
                              "for i in {npu_card_ids}; do hccn_tool -i $i -tls -g |grep switch >> {npu_log_path}/
npu-smi_switch.log;done\n" \
                              "for i in {npu_card_ids}; do hccn_tool -i $i -optical -g | grep prese >>
{npu_log_path}/npu-smi_present.log ;done\n" \
                              "for i in {npu_card_ids}; do hccn_tool -i $i -link_stat -g >> {npu_log_path}/
npu_link_history.log ;done\n" \
                              "for i in {npu_card_ids}; do hccn_tool -i $i -ip -g >> {npu_log_path}/
npu_roce_ip_info.log ;done\n" \
                              "for i in {npu_card_ids}; do hccn_tool -i $i -lldp -g >> {npu_log_path}/
npu_nic_switch_info.log ;done\n" \
            .format(npu_log_path=self.NPU_LOG_PATH,
                    npu_card_ids=self.card_ids,
                    npu_network_info="collect npu network info")

        collect_npu_log_shell = "# !/bin/sh\n" \
                                "step=1\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 --
version > {npu_log_path}/npu-smi_firmware-version.log\n" \
                                "for i in {npu_card_ids}; do for ((j=0;j<{chip_count};j++)); do npu-smi info -t
health -i $i -c $j; done >> {npu_log_path}/npu-smi_health-code.log;done;\n" \
                                "for i in {npu_card_ids}; do npu-smi info -t board -i $i >> {npu_log_path}/npu-
smi_board.log; done;\n" \
                                "echo {echo_npu_ecc_info}\n" \
                                "for i in {npu_card_ids};do npu-smi info -t ecc -i $i >> {npu_log_path}/npu-
smi_ecc.log; done;\n" \
                                "lspci | 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/null\n" \
            .format(npu_log_path=self.NPU_LOG_PATH,
                    npu_card_ids=self.card_ids,
                    chip_count=self.chip_count,
                    echo_npu_driver_info="collect npu driver info.",
                    echo_npu_ecc_info="collect npu ecc info.",
                    echo_npu_device_log="collect npu device log.",
```



```
                echo_npu_tools_log="collect npu tools log.")
        if self.is_300_iduo:
            return collect_npu_log_shell
        return collect_npu_log_shell + hccn_tool_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, 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 = '\"' + "Date:" + date + '\"'
        header_auth = '\"' + "Authorization:" + auth + '\"'
        header_obs_acl = '\"' + canonicalized_headers + '\"'

        cmd = "curl -X PUT -T " + log_tar + " -w %{http_code} " + obs_url + " -H " + header_date + " -H
" + header_auth + " -H " + header_obs_acl
        result = subprocess.run(cmd, shell=True, capture_output=True, text=True)
        http_code = result.stdout.strip()
        if result.returncode == 0 and http_code == "200":
            print("success to upload {} to obs bucket {}".format(log_tar, obs_bucket))
        else:
            print("failed to upload {} to obs bucket {}".format(log_tar, obs_bucket))
            print(result)

    #  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

    # get NPU Id and Chip count
    def get_card_ids(self):
        card_ids = []
        cmd = "npu-smi info -l"
        result = subprocess.run(cmd, shell=True, capture_output=True, text=True)
        if result.returncode != 0:
            print("failed to execute command[{}]".format(cmd))
            return ""
        match = re.search(r'Chip Count\s*:\s*(\d+)', result.stdout)
        # default chip count is 1, 300IDUO or 910C is 2
        chip_count = 1
        if match and int(match.group(1)) > 0:
            chip_count=int(match.group(1))
```

```
        # filter NPU ID Regex
        pattern = re.compile(r'NPU ID(.*?): (.*?)\n', re.DOTALL)
        matches = pattern.findall(result.stdout)
        for match in matches:
            if len(match) != 2:
                continue
            id = int(match[1])
            # if drop card
            if id < 0:
                print("Card may not be found, NPU ID: {}".format(id))
                continue
            card_ids.append(id)
        print("success to get card id {}, Chip Count {}".format(card_ids, chip_count))
        return " ".join(str(x) for x in card_ids), chip_count

    def execute(self):
        if self.obs_dir == "":
            print("the obs_dir is null, please enter a correct dir")
        else:
            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',
                                          is_300_iduo=False)
    npu_log_collection.execute()
```

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 10-4 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
   ```

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

   Figure 10-5 Viewing the result

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

10.2 Collecting and Uploading GPU Logs

Scenario

When a GPU is faulty, collect GPU log information by referring to this exercise. The logs generated in this exercise are saved on the node and automatically uploaded to the OBS bucket provided by technical support. Logs are used only for fault

locating and analysis. You need to provide the AK/SK for authorization and authentication.

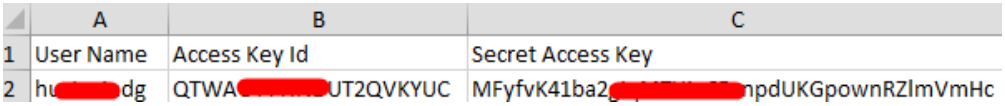
Procedure

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 10-6 credential.csv

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

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

   a. Log in to the **Huawei Cloud console**.

   b. Hover over the username in the upper right corner and choose **My Credentials** from the drop-down list.

   c. In the navigation pane on the left, choose **Access Keys**.

   d. Click **Create Access Key**.

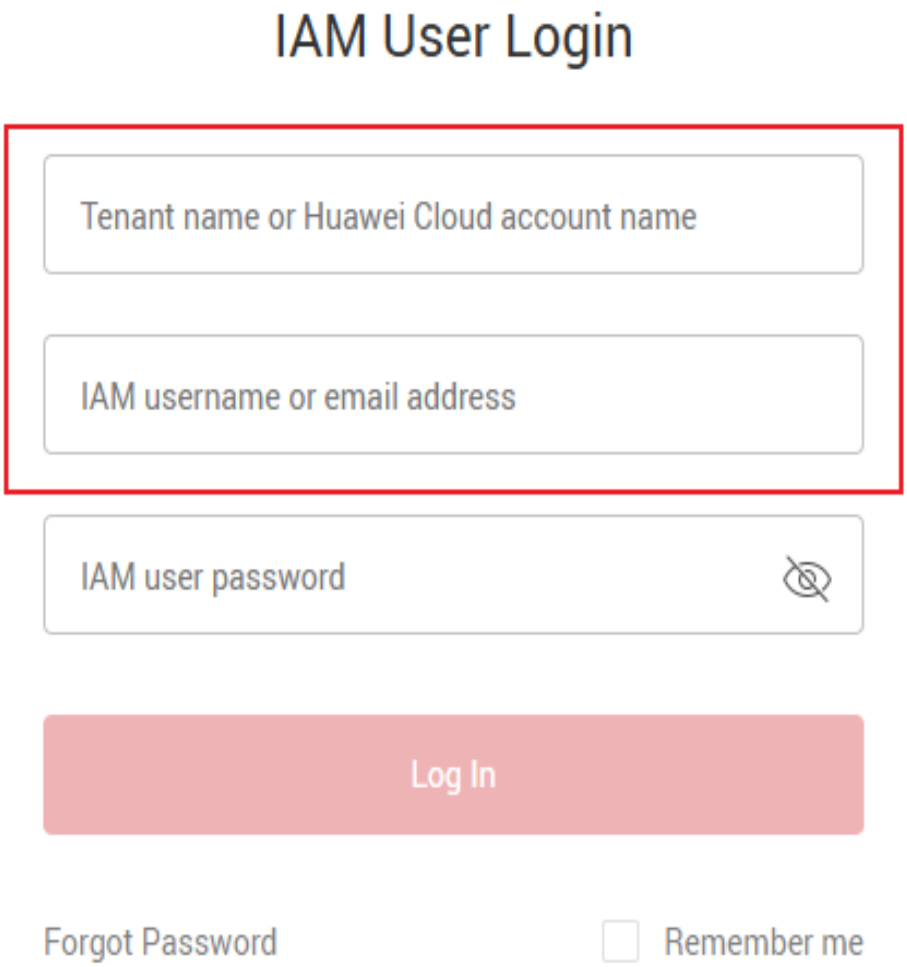
   e. Download the key and keep it secure.

2. Prepare the tenant ID and IAM ID for OBS bucket configuration.

   Send the prepared information to Huawei technical support, who 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 10-7 Tenant ID and IAM ID



3. Collect the logs and upload the script.

   Modify the **GpuLogCollection** parameter in the script below and replace **ak**, **sk**, and **obs_dir** with the values obtained in the previous steps. Upload the script to the node whose GPU logs need to be collected.

```
import json
import os
import sys
import hashlib
import hmac
import binascii
from datetime import datetime
class GpuLogCollection(object):
    GPU_LOG_PATH = "nvidia-bug-report.log.gz"
    SUPPORT_REGIONS = ['cn-north-4', '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_gpu_log_shell(self):
        collect_gpu_log_shell = "nvidia-bug-report.sh"
        return collect_gpu_log_shell
    def collect_gpu_log(self):
        print("begin to collect gpu log")
        os.system(self.gen_collect_gpu_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-gpu-log-%s.gz" % (instance_ip, date_collect)
        os.system("cp %s %s" % (self.GPU_LOG_PATH, log_tar))
        print("success to collect gpu 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, 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 = '\"' + "Date:" + date + '\"'
        header_auth = '\"' + "Authorization:" + auth + '\"'
        header_obs_acl = '\"' + canonicalized_headers + '\"'
        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_gpu_log()
        self.upload_log_to_obs(log_tar)
if __name__ == '__main__':
    gpu_log_collection = GpuLogCollection(ak='ak',
                                          sk='sk',
                                          obs_dir='xxx')
    gpu_log_collection.execute()
```

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 10-8 Log collected

```
root@dev[illegible]::~/test# python3 log.py
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100  1710  100  1710    0     0  63333      0 --:--:-- --:--:-- --:--:-- 63333
begin to collect gpu log

nvidia-bug-report.sh will now collect information about your
system and create the file 'nvidia-bug-report.log.gz' in the current
directory.  It may take several seconds to run.  In some
cases, it may hang trying to capture data generated dynamically
by the Linux kernel and/or the NVIDIA kernel module.  While
the bug report log file will be incomplete if this happens, it
may still contain enough data to diagnose your problem.

If nvidia-bug-report.sh hangs, consider running with the --safe-mode
and --extra-system-data command line arguments.

Please include the 'nvidia-bug-report.log.gz' log file when reporting
your bug via the NVIDIA Linux forum (see forums.developer.nvidia.com)
or by sending email to 'linux-bugs@nvidia.com'.

By delivering 'nvidia-bug-report.log.gz' to NVIDIA, you acknowledge
and agree that personal information may inadvertently be included in
the output.  Notwithstanding the foregoing, NVIDIA will use the
output only for the purpose of investigating your reported issue.

Running nvidia-bug-report.sh... complete.

  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100    12  100    12    0     0    521      0 --:--:-- --:--:-- --:--:--   521
success to collect gpu log with 192.168.0.23-gpu-log-20250206092139.gz
begin to upload 192.168.0.23-gpu-log-20250206092139.gz to obs bucket c30049938
success to upload 192.168.0.23-gpu-log-20250206092139.gz to obs bucket c30049938
```

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

Figure 10-9 Viewing the result

```
root@devser[illegible]c:~/test# ls
192.168.0.23-gpu-log-20250206092139.gz  log.py  nvidia-bug-report.log.gz
```

11 Monitoring Lite Servers

11.1 Using Cloud Eye to Monitor NPU Resources of Lite Servers

Description

The monitoring capabilities of Lite Servers rely on Cloud Eye. This section describes how to integrate with Cloud Eye to monitor resources and events on these nodes.

Constraints

- The Cloud Eye Agent plug-in, which has strict resource usage restrictions, is required for monitoring. When the resource usage exceeds the threshold, the Agent circuit breaker occurs. For details about the resource usage, see **Cloud Eye Server Monitoring**.
- If you run the NPU pressure test command using Ascend-dmi, some NPU metric data may be lost.
- You have fully tested the monitoring agent in the public image provided by Lite Servers. If you use your own image, perform the test before deploying the image in the production environment to prevent information errors.

Prerequisites

The Cloud Eye Agent has been installed on the Lite Server. For details about how to check whether the Cloud Eye Agent is installed and how to install it, see **Installing Cloud Eye Agent Monitoring Plug-ins**.

Monitoring Solution for Lite Servers

For details, see **Bare Metal Server (BMS) Server Monitoring**. In addition to the images listed in the document, Ubuntu 20.04 is also supported.

The sampling period of monitoring metrics is 1 minute. Do not change it. Otherwise, the function may be abnormal. The current monitoring metrics include the CPU, memory, disk, and network. After the accelerator PU driver is installed on the host, the related metrics can be collected.

The NPU metric collection function depends on the Linux system tool lspci. Some events depend on the blkid and grub2-editenv system tools. Ensure that these tools are normal.

Table 11-1 Monitoring tools

Tool	Check Method	Installation Method
lspci	Run **lspci** in the shell environment. The PCI device in the system can be queried. The following shows an example: $ sudo lspci 00:00.0 PCI bridge: Huawei Technologies Co., Ltd. HiSilicon PCIe Root Port with Gen4 (rev 21) 00:08.0 PCI bridge: Huawei Technologies Co., Ltd. HiSilicon PCIe Root Port with Gen4 (rev 21) 00:10.0 PCI bridge: Huawei Technologies Co., Ltd. HiSilicon PCIe Root Port with Gen4 (rev 21)	lspci is a tool used to display PCI device information. It is usually included in the **pciutils** software package. This software package is installed by default in most Linux versions. Generally, lspci is pre-installed. If lspci is not installed, you can use the package manager to install pciutils. Run the following commands in Debian/Ubuntu: sudo apt-get update sudo apt-get install pciutils Run the following command in Red Hat/CentOS/EulerOS: sudo yum install pciutils

Tool	Check Method	Installation Method
blkid	Run **blkid** in the shell environment. The block device in the system can be queried. The following shows an example: $ sudo blkid /dev/sda1: UUID="123e4567-e89b-12d3-a456-426614174000" TYPE="vfat" PARTUUID="56789abc-def0-1234-5678-9abcd3f2c0a1" /dev/sda2: UUID="a1b2c3d4-e5f6-789a-bcde-f0123456789a" TYPE="swap" PARTUUID="edcba98-7654-3210-fedc-ba9876543210" /dev/sda3: UUID="01234567-89ab-cdef-0123-456789abcdef" TYPE="ext4" PARTUUID="fedcba09-8765-4321-fedc-ba0987654321"	blkid is a tool used to display block device attributes in Linux. It is usually included in the **util-linux** software package. This software package is installed by default in most Linux versions. Generally, blkid is pre-installed. If blkid is not installed, you can use the package manager to install util-linux. Run the following commands in Debian/Ubuntu: sudo apt-get update sudo apt-get install util-linux Run the following command in Red Hat/CentOS/EulerOS: sudo yum install util-linux
grub2-editenv (required only for Red Hat, CentOS, and EulerOS)	Run **blkid** in the shell environment. The block device in the system can be queried. The following shows an example: 1 2 3 4 $ sudo grub2-editenv list timeout=5default=0saved_entry=Red Hat Enterprise Linux Server, with Linux 4.18.0-305.el8.x86_64	grub2-editenv is part of GRUB2 and is used to manage GRUB environment variables. GRUB2 is installed by default in most Linux versions. Generally, grub2-editenv is pre-installed. If grub2-editenv is not installed, you can use the package manager to install it. Run the following commands in Debian/Ubuntu: sudo apt-get update sudo apt-get install grub2 Run the following command in Red Hat/CentOS/EulerOS: sudo yum install grub2

Installing Cloud Eye Agent Monitoring Plug-ins

OS-level, proactive, and fine-grained server monitoring is provided after the Cloud Eye Agent is installed on the Lite Server (ECS or BMS).

Cloud Eye Agent is installed in the preset OS on the Lite Server by default. You can check the Agent status and version on the Cloud Eye console.

If Cloud Eye Agent is not installed or the Cloud Eye Agent version does not meet the requirements, use either of the following methods:

Method 1: Install Cloud Eye Agent on the console by referring to **Installing or Upgrading the Cloud Eye Agent Plugin on a Lite Server**.

Method 2: Manually install Cloud Eye Agent.

1. Create an agency for Cloud Eye. For details, see **Creating a User and Granting Permissions**. If you have enabled Cloud Eye host monitoring authorization during **Table 4-1**, skip this step.
2. Currently, one-click monitoring installation is not supported on the Cloud Eye 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
   ```

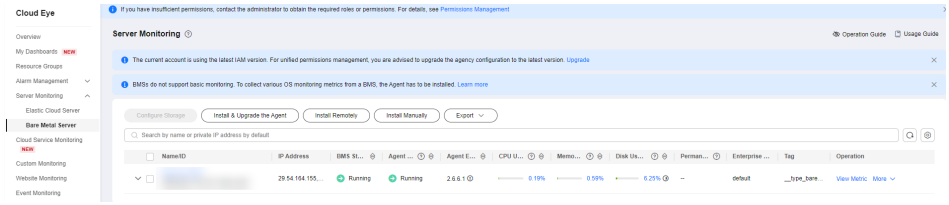
   If the following information is displayed, the installation is successful.

Figure 11-1 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.
```

3. View the monitoring items on Cloud Eye page. Accelerator PU monitoring items are available only after the accelerator PU driver is installed on the host.

Figure 11-2 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.

Metric Namespace

AGT.ECS and SERVICE.BMS

Key Metrics for Training and Inference

Table 11-2 lists the key metrics for training or inference on the Lite Server.

Table 11-2 Common monitoring metrics

No.	Category	Metric	Display Name	Description	Unit	Number System	Range	Applicable Model
1	Overall	npu_device_health	NPU Health Status	Health status of the NPU	-	N/A	**0**: normal **1**: minor alarm **2**: major alarm **3**: critical alarm	Snt3P 300IDuo Snt9b Snt9b23
2		npu_util_rate_general	Overall NPU Usage	Overall NPU usage, which covers AI Core and Vector Core statistics.	%	N/A	0%–100%	Snt9b Snt9b23

No.	Category	Metric	Display Name	Description	Unit	Number System	Range	Applicable Model
3	DDR	npu_util_rate_mem	NPU Memory Usage	NPU memory usage	%	N/A	0%–100%	Snt3P 300IDuo
4		npu_util_rate_mem_bandwidth	NPU Memory Bandwidth Usage	NPU memory bandwidth usage	%	N/A	0%–100%	
5	HBM	npu_hbm_bandwidth_util	HBM Bandwidth Usage	NPU HBM bandwidth usage (old metric)	%	N/A	0%–100%	Snt9b Snt9b23
6		npu_util_rate_hbm_bw	HBM Bandwidth Usage	NPU HBM bandwidth usage (new metric)	%	N/A	0%–100%	Snt9b Snt9b23
7	AICore	npu_util_rate_ai_core	AI Core Usage of the NPU	AI core usage of the NPU	%	N/A	0%–100%	Snt3P 300IDuo Snt9b Snt9b23

No.	Category	Metric	Display Name	Description	Unit	Number System	Range	Applicable Model
8	AI Vector	npu_util_rate_vector_core	NPU Vector Core Usage	NPU Vector core usage	%	N/A	0%–100%	Snt 3P 300 IDuo Snt 9b Snt 9b23

Lite Server NPU Metrics

The table below only lists NPU metrics. For details about other metrics, see **What Metrics Are Supported by the Agent?**

Table 11-3 NPU metrics (all)

No.	Metric	Display Name	Description	Unit	Conversion Rule	Value Range	Dimension	Supported Model	Supported Cloud Eye Agent Version
1	npu_device_health	NPU Health Status	Health status of the NPU	-	N/A	**0**: normal **1**: minor alarm **2**: major alarm **3**: critical alarm	instance_id, npu	Snt3P300IDuo Snt9b Snt9b23	telescope: 2.7.4.3 2.7.5.3 2.7.5.4 2.7.5.9 or later
2	npu_driver_health	NPU Driver Health Status	Health status of the NPU driver	-	N/A	**0**: normal **3**: critical alarm	instance_id, npu		
3	npu_power	NPU Power	NPU power	W	N/A	>0	instance_id, npu		
4	npu_temperature	NPU Temperature	NPU temperature	°C	N/A	Natural number	instance_id, npu		

No.	Metric	Display Name	Description	Unit	Conversion Rule	Value Range	Dimension	Supported Model	Supported Cloud Eye Agent Version
5	npu_voltage	NPU Voltage	NPU voltage	V	N/A	Natural number	instance_id, npu		
6	npu_util_rate_general	Overall NPU usage	Overall NPU usage, which covers AI Core and Vector Core statistics.	%	N/A	0%–100%	instance_id, npu	Snt9b Snt9b23	

Lite Server HBM Metrics

Table 11-4 NPU metrics (HBM)

No.	Metric	Display Name	Description	Unit	Conversion Rule	Value Range	Dimension	Supported Model	Supported Cloud Eye Agent Version
1	npu_util_rate_hbm	NPU HBM Usage	HBM usage of the NPU	%	N/A	0%–100%	instance_id, npu	Snt9b Snt9b23	telescope: 2.7.4.3 2.7.5.3 2.7.5.4 2.7.5.9 or later
2	npu_hbm_freq	HBM Frequency	NPU HBM frequency (old version)	MHz	N/A	>0	instance_id, npu		
3	npu_freq_hbm	HBM Frequency	NPU HBM frequency (new version)	MHz	N/A	>0	instance_id, npu		
4	npu_hbm_usage	HBM Usage	NPU HBM usage	MB	N/A	≥ 0	instance_id, npu		
5	npu_hbm_temperature	HBM Temperature	NPU HBM temperature	°C	N/A	Natural number	instance_id, npu		
6	npu_hbm_bandwidth_util	HBM Bandwidth Usage	NPU HBM bandwidth usage (old version)	%	N/A	0%–100%	instance_id, npu		
7	npu_util_rate_hbm_bw	HBM Bandwidth Usage	NPU HBM bandwidth usage (new version)	%	N/A	0%–100%	instance_id, npu		

No.	Metric	Display Name	Description	Unit	Conversion Rule	Value Range	Dimension	Supported Model	Supported Cloud Eye Agent Version
8	npu_hbm_mem_capacity	NPU HBM Memory Capacity	HBM memory capacity of the NPU	MB	N/A	≥ 0	instance_id, npu		
9	npu_hbm_ecc_enable	HBM ECC Status	NPU HBM ECC status	-	N/A	**0**: ECC detection is disabled. **1**: ECC detection is enabled.	instance_id, npu		
10	npu_hbm_single_bit_error_cnt	Single-bit Errors on HBM	Current number of single-bit errors on the NPU HBM	count	N/A	≥ 0	instance_id, npu		
11	npu_hbm_double_bit_error_cnt	Double-bit Errors on HBM	Current number of double-bit errors on the NPU HBM	count	N/A	≥ 0	instance_id, npu		

No.	Metric	Display Name	Description	Unit	Conversion Rule	Value Range	Dimension	Supported Model	Supported Cloud Eye Agent Version
12	npu_hbm_total_single_bit_error_cnt	Single-bit Errors in HBM Lifecycle	Number of single-bit errors in the NPU HBM lifecycle	count	N/A	≥ 0	instance_id, npu		
13	npu_hbm_total_double_bit_error_cnt	Double-bit Errors in HBM Lifecycle	Number of double-bit errors in the NPU HBM lifecycle	count	N/A	≥ 0	instance_id, npu		
14	npu_hbm_single_bit_isolated_pages_cnt	Isolated NPU Memory Pages with HBM Single-bit Errors	Number of isolated NPU memory pages with HBM single-bit errors	count	N/A	≥ 0	instance_id, npu		

No.	Metric	Display Name	Description	Unit	Conversion Rule	Value Range	Dimension	Supported Model	Supported Cloud Eye Agent Version
15	npu_hbm_double_bit_isolated_pages_cnt	Isolated NPU Memory Pages with HBM Multi-bit Errors	Number of isolated NPU memory pages with HBM double-bit errors	count	N/A	≥ 0	instance_id, npu		

Lite Server DDR Metrics

Table 11-5 NPU metrics (DDR)

No.	Metric	Display Name	Description	Unit	Conversion Rule	Value Range	Dimension	Supported Model	Supported Cloud Eye Agent Version
1	npu_usage_mem	Used NPU Memory	Used NPU memory	MB	N/A	≥ 0	instance_id, npu	Snt3P 300IDuo	telescope: 2.7.4.3 2.7.5.3 2.7.5.4 2.7.5.9 or later
2	npu_util_rate_mem	NPU Memory Usage	NPU memory usage	%	N/A	0%–100%	instance_id, npu		
3	npu_freq_mem	NPU Memory Frequency	NPU memory frequency	MHz	N/A	>0	instance_id, npu		
4	npu_util_rate_mem_bandwidth	NPU Memory Bandwidth Usage	NPU memory bandwidth usage	%	N/A	0%–100%	instance_id, npu		
5	npu_sbe	NPU Single-bit Errors	Number of single-bit errors on the NPU	count	N/A	≥ 0	instance_id, npu		
6	npu_dbe	NPU Double-bit Errors	Number of double-bit errors on the NPU	count	N/A	≥ 0	instance_id, npu		

Lite Server AI Core Metrics

Table 11-6 NPU metrics (AI Core)

No.	Metric	Display Name	Description	Unit	Conversion Rule	Value Range	Dimension	Supported Model	Supported Cloud Eye Agent Version
1	npu_freq_ai_core	AI Core Frequency of the NPU	AI core frequency of the NPU	MHz	N/A	>0	instance_id, npu	Snt3P300IDuo Snt9b Snt9b23	telescope: 2.7.4.3 2.7.5.3 2.7.5.4 2.7.5.9 or later
2	npu_freq_ai_core_rated	Rated Frequency of the NPU AI Core	Rated frequency of the NPU AI core	MHz	N/A	>0	instance_id, npu		
3	npu_util_rate_ai_core	AI Core Usage of the NPU	AI core usage of the NPU	%	N/A	0%–100%	instance_id, npu		

Lite Server AI Vector Metrics

Table 11-7 NPU metrics (AI Vector)

No.	Metric	Display Name	Description	Unit	Conversion Rule	Value Range	Dimension	Supported Model	Supported Cloud Eye Agent Version
1	npu_util_rate_vector_core	NPU Vector Core Usage	NPU Vector Core Usage	%	N/A	0%–100%	instance_id, npu	Snt3P300IDuo Snt9b Snt9b23	telescope: 2.7.5.9 or later

Lite Server AI CPU Metrics

Table 11-8 NPU metrics (AI CPU)

No.	Metric	Display Name	Description	Unit	Conversion Rule	Value Range	Dimension	Supported Model	Supported Cloud Eye Agent Version
1	npu_aicpu_num	Number of AI CPUs of the NPU	Number of AI CPUs of the NPU	count	N/A	≥ 0	instance_id, npu	Snt3P 300IDuo Snt9b Snt9b23	telescope: 2.7.4.3 2.7.5.3 2.7.5.4 2.7.5.9 or later
2	npu_util_rate_ai_cpu	NPU AI CPU Usage	AI CPU usage of the NPU	%	N/A	0%–100%	instance_id, npu		
3	npu_aicpu_avg_util_rate	Average AI CPU Usage of the NPU	Average AI CPU usage of the NPU	%	N/A	0%–100%	instance_id, npu		
4	npu_aicpu_max_freq	Maximum AI CPU Frequency of the NPU	Maximum AI CPU frequency of the NPU	MHz	N/A	>0	instance_id, npu		
5	npu_aicpu_cur_freq	AI CPU Frequency of the NPU	AI CPU frequency of the NPU	MHz	N/A	>0	instance_id, npu		

Lite Server CTRL CPU Metrics

Table 11-9 NPU metrics (CTRL CPU)

No.	Metric	Display Name	Description	Unit	Conversion Rule	Value Range	Dimension	Supported Model	Supported Cloud Eye Agent Version
1	npu_util_rate_ctrl_cpu	Control CPU Usage of the NPU	Control CPU usage of the NPU	%	N/A	0%–100%	instance_id, npu	Snt3P 300IDuo Snt9b Snt9b23	telescope: 2.7.4.3 2.7.5.3 2.7.5.4 2.7.5.9 or later
2	npu_freq_ctrl_cpu	Control CPU Frequency of the NPU	Control CPU frequency of the NPU	MHz	N/A	>0	instance_id, npu		

Lite Server PCIe Link Metrics

Table 11-10 NPU metrics (PCIE link)

No.	Metric	Display Name	Description	Unit	Conversion Rule	Value Range	Dimension	Supported Model	Supported Cloud Eye Agent Version
1	npu_link_cap_speed	Max. NPU Link Speed	Maximum link speed of the NPU	GT/s	N/A	≥ 0	instance_id, npu	310P 300IDuo Snt9b Snt9b23	telescope: 2.7.4.3 2.7.5.3 2.7.5.4 2.7.5.9 or later
2	npu_link_cap_width	Max. NPU Link Width	Maximum link width of the NPU	count	N/A	≥ 0	instance_id, npu		
3	npu_link_status_speed	NPU Link Speed	Link speed of the NPU	GT/s	N/A	≥ 0	instance_id, npu		
4	npu_link_status_width	NPU Link Width	Link width of the NPU	count	N/A	≥ 0	instance_id, npu		

Lite Server RoCE Network Metrics

Table 11-11 NPU metrics (RoCE network)

No.	Metric	Display Name	Description	Unit	Conversion Rule	Value Range	Dimension	Supported Model	Supported Cloud Eye Agent Version
1	npu_device_network_health	NPU Network Health Status	Connectivity of the IP address of the RoCE NIC on the NPU	-	N/A	**0**: The network health status is normal. Other values: The network status is abnormal.	instance_id, npu	Snt9b Snt9b23	telescope: 2.7.4.3 2.7.5.3 2.7.5.4 2.7.5.9 or later
2	npu_network_port_link_status	NPU Network Port Link Status	Link status of the NPU network port	-	N/A	**0**: up **1**: down	instance_id, npu		
3	npu_roce_tx_rate	NPU NIC Uplink Rate	Uplink rate of the NPU NIC	MB/s	N/A	≥ 0	instance_id, npu		

No.	Metric	Display Name	Description	Unit	Conversion Rule	Value Range	Dimension	Supported Model	Supported Cloud Eye Agent Version
4	npu_roce_rx_rate	NPU NIC Downlink Rate	Downlink rate of the NPU NIC	MB/s	N/A	≥ 0	instance_id, npu		
5	npu_mac_tx_mac_pause_num	PAUSE Frames Sent from MAC	Total number of PAUSE frames sent from the MAC address corresponding to the NPU	count	N/A	≥ 0	instance_id, npu		
6	npu_mac_rx_mac_pause_num	PAUSE Frames Received by MAC	Total number of PAUSE frames received by the MAC address corresponding to the NPU	count	N/A	≥ 0	instance_id, npu		
7	npu_mac_tx_pfc_pkt_num	PFC Frames Sent from MAC	Total number of PFC frames sent from the MAC address corresponding to the NPU	count	N/A	≥ 0	instance_id, npu		

No.	Metric	Display Name	Description	Unit	Conversion Rule	Value Range	Dimension	Supported Model	Supported Cloud Eye Agent Version
8	npu_mac_rx_pfc_pkt_num	PFC Frames Received by MAC	Total number of PFC frames received by the MAC address corresponding to the NPU	count	N/A	≥ 0	instance_id, npu		
9	npu_mac_tx_bad_pkt_num	Bad Packets Sent from MAC	Total number of bad packets sent from the MAC address corresponding to the NPU	count	N/A	≥ 0	instance_id, npu		
10	npu_mac_rx_bad_pkt_num	Bad Packets Received by MAC	Total number of bad packets received by the MAC address corresponding to the NPU	count	N/A	≥ 0	instance_id, npu		
11	npu_roce_tx_err_pkt_num	Bad Packets Sent by RoCE	Total number of bad packets sent by the RoCE NIC on the NPU	count	N/A	≥ 0	instance_id, npu		

No.	Metric	Display Name	Description	Unit	Conversion Rule	Value Range	Dimension	Supported Model	Supported Cloud Eye Agent Version
12	npu_roce_rx_err_pkt_num	Bad Packets Received by RoCE	Total number of bad packets received by the RoCE NIC on the NPU	count	N/A	≥ 0	instance_id, npu		
13	npu_roce_tx_all_pkt_num	Packets Transmitted by NPU RoCE	The number of packets transmitted by the NPU's RoCE.	count	N/A	≥ 0	instance_id, npu		telescope: 2.7.5.9 or later
14	npu_roce_rx_all_pkt_num	Packets Received by NPU RoCE	The number of packets received by the NPU's RoCE.	count	N/A	≥ 0	instance_id, npu		
15	npu_roce_new_pkt_rty_num	Packets Retransmitted by NPU RoCE	The number of packets retransmitted by the NPU's RoCE.	count	N/A	≥ 0	instance_id, npu		

No.	Metric	Display Name	Description	Unit	Conversion Rule	Value Range	Dimension	Supported Model	Supported Cloud Eye Agent Version
16	npu_roce_out_of_order_num	Abnormal PSN Packets Received by NPU RoCE	This metric indicates that number of PSN packets received by NPU RoCE is greater than that of expected or duplicate PSN packets. If packets are out of order or lost, retransmission is triggered.	count	N/A	≥ 0	instance_id, npu		
17	npu_roce_rx_cnp_pkt_num	CNP Packets Received by NPU RoCE	The number of CNP packets received by the NPU's RoCE.	count	N/A	≥ 0	instance_id, npu		
18	npu_roce_tx_cnp_pkt_num	CNP Packets Transmitted by NPU RoCE	The number of CNP packets transmitted by the NPU's RoCE.	count	N/A	≥ 0	instance_id, npu		

Lite Server RoCE Optical Module Metrics

Table 11-12 NPU metrics (RoCE optical module)

No.	Metric	Display Name	Description	Unit	Conversion Rule	Value Range	Dimension	Supported Model	Supported Cloud Eye Agent Version
1	npu_opt_temperature	NPU Optical Module Temperature	NPU optical module temperature	°C	N/A	Natural number	instance_id, npu	Snt9b Snt9b23	telescope: 2.7.4.3 2.7.5.3 2.7.5.4 2.7.5.9 or later
2	npu_opt_temperature_high_thres	Upper Limit of the NPU Optical Module Temperature	Upper limit of the NPU optical module temperature	°C	N/A	Natural number	instance_id, npu		
3	npu_opt_temperature_low_thres	Lower Limit of the NPU Optical Module Temperature	Lower limit of the NPU optical module temperature	°C	N/A	Natural number	instance_id, npu		
4	npu_opt_voltage	NPU Optical Module Voltage	NPU optical module voltage	mV	N/A	≥ 0	instance_id, npu		

No.	Metric	Display Name	Description	Unit	Conversion Rule	Value Range	Dimension	Supported Model	Supported Cloud Eye Agent Version
5	npu_opt_voltage_high_thres	Upper Limit of the NPU Optical Module Voltage	Upper limit of the NPU optical module voltage	mV	N/A	≥ 0	instance_id, npu		
6	npu_opt_voltage_low_thres	Lower Limit of the NPU Optical Module Voltage	Lower limit of the NPU optical module voltage	mV	N/A	≥ 0	instance_id, npu		
7	npu_opt_tx_power_lane0	TX Power of the NPU Optical Module in Channel 0	Transmit power of the NPU optical module in channel 0	mW	N/A	≥ 0	instance_id, npu		
8	npu_opt_tx_power_lane1	TX Power of the NPU Optical Module in Channel 1	Transmit power of the NPU optical module in channel 1	mW	N/A	≥ 0	instance_id, npu		

No.	Metric	Display Name	Description	Unit	Conversion Rule	Value Range	Dimension	Supported Model	Supported Cloud Eye Agent Version
9	npu_opt_tx_power_lane2	TX Power of the NPU Optical Module in Channel 2	Transmit power of the NPU optical module in channel 2	mW	N/A	≥ 0	instance_id, npu		
10	npu_opt_tx_power_lane3	TX Power of the NPU Optical Module in Channel 3	Transmit power of the NPU optical module in channel 3	mW	N/A	≥ 0	instance_id, npu		
11	npu_opt_rx_power_lane0	RX Power of the NPU Optical Module in Channel 0	Receive power of the NPU optical module in channel 0	mW	N/A	≥ 0	instance_id, npu		
12	npu_opt_rx_power_lane1	RX Power of the NPU Optical Module in Channel 1	Receive power of the NPU optical module in channel 1	mW	N/A	≥ 0	instance_id, npu		

No.	Metric	Display Name	Description	Unit	Conversion Rule	Value Range	Dimension	Supported Model	Supported Cloud Eye Agent Version
13	npu_opt_rx_power_lane2	RX Power of the NPU Optical Module in Channel 2	Receive power of the NPU optical module in channel 2	mW	N/A	≥ 0	instance_id, npu		
14	npu_opt_rx_power_lane3	RX Power of the NPU Optical Module in Channel 3	Receive power of the NPU optical module in channel 3	mW	N/A	≥ 0	instance_id, npu		
15	npu_opt_tx_bias_lane0	TX Bias Current of the NPU Optical Module in Channel 0	Transmitted bias current of the NPU optical module in channel 0	mA	N/A	≥ 0	instance_id, npu		
16	npu_opt_tx_bias_lane1	TX Bias Current of the NPU Optical Module in Channel 1	Transmitted bias current of the NPU optical module in channel 1	mA	N/A	≥ 0	instance_id, npu		

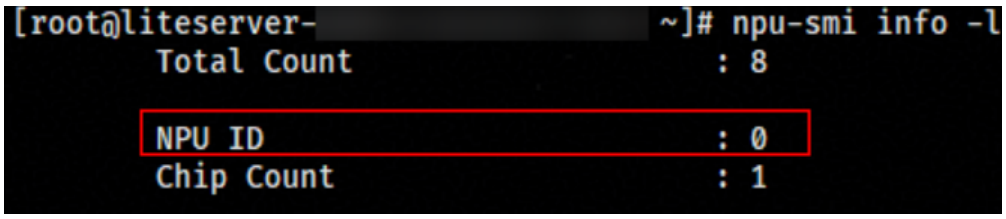
No.	Metric	Display Name	Description	Unit	Conversion Rule	Value Range	Dimension	Supported Model	Supported Cloud Eye Agent Version
17	npu_opt_tx_bias_lane2	TX Bias Current of the NPU Optical Module in Channel 2	Transmitted bias current of the NPU optical module in channel 2	mA	N/A	≥ 0	instance_id, npu		
18	npu_opt_tx_bias_lane3	TX Bias Current of the NPU Optical Module in Channel 3	Transmitted bias current of the NPU optical module in channel 3	mA	N/A	≥ 0	instance_id, npu		
19	npu_opt_tx_los	TX Los of the NPU Optical Module	TX Los flag of the NPU optical module	count	N/A	≥ 0	instance_id, npu		
20	npu_opt_rx_los	RX Los of the NPU Optical Module	RX Los flag of the NPU optical module	count	N/A	≥ 0	instance_id, npu		

No.	Metric	Display Name	Description	Unit	Conversion Rule	Value Range	Dimension	Supported Model	Supported Cloud Eye Agent Version
21	npu_opt_media_snr_lane0	NPU Optical Module Channel 0 Optical SNR	The signal-to-noise ratio (SNR) on the media (optical) side of channel 0 in the NPU optical module	dB	N/A	≥ 0	instance_id, npu		telescope: 2.7.5.9 or later
22	npu_opt_media_snr_lane1	NPU Optical Module Channel 1 Optical SNR	The signal-to-noise ratio (SNR) on the media (optical) side of channel 1 in the NPU optical module	dB	N/A	≥ 0	instance_id, npu		
23	npu_opt_media_snr_lane2	NPU Optical Module Channel 2 Optical SNR	The signal-to-noise ratio (SNR) on the media (optical) side of channel 2 in the NPU optical module	dB	N/A	≥ 0	instance_id, npu		

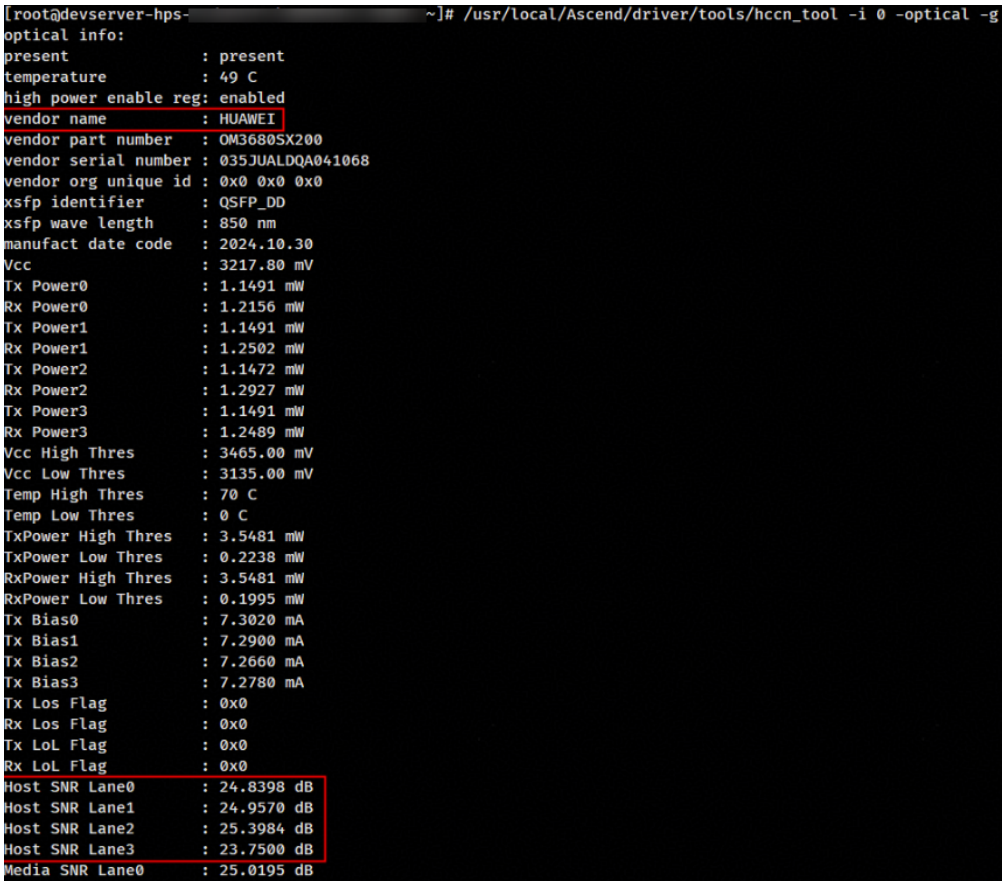
No.	Metric	Display Name	Description	Unit	Conversion Rule	Value Range	Dimension	Supported Model	Supported Cloud Eye Agent Version
24	npu_opt_media_snr_lane3	NPU Optical Module Channel 3 Optical SNR	The signal-to-noise ratio (SNR) on the media (optical) side of channel 3 in the NPU optical module	dB	N/A	≥ 0	instance_id, npu		

On the Lite Server, run the commands below to check whether vendor name is **HUAWEI**. If not, **npu_opt_host_snr_lanex** metrics of the RoCE optical module cannot be reported.

```
# Obtain the NPU ID.
npu-smi info -l
```




```
# Obtain the vendor name based on the NPU ID.
hccn_tool -i <NPU_ID> -optical -g
```



Lite Server HCCS Serdes SNR Metrics

Table 11-13 NPU Metrics (HCCS SerDes SNR)

No.	Metric	Display Name	Description	Unit	Conversion Rule	Value Range	Dimension	Supported Model	Supported Cloud Eye Agent Version
1	npu_macro1_serdes_lane0_snr	NPU Macro1 SerDes Lane 0 SNR	The SNR for SerDes Lane 0 in NPU Macro1	dB	N/A	Natural number	instance_id, npu	Snt9b Snt9b23	telescope: 2.7.5.9 or later

No.	Metric	Display Name	Description	Unit	Conversion Rule	Value Range	Dimension	Supported Model	Supported CloudEye Agent Version
2	npu_macro1_serdes_lane1_snr	NPU Macro1 SerDes Lane 1 SNR	The SNR for SerDes Lane 1 in NPU Macro1	dB	N/A	Natural number	instance_id, npu		
3	npu_macro1_serdes_lane2_snr	NPU Macro1 SerDes Lane 2 SNR	The SNR for SerDes Lane 2 in NPU Macro1	dB	N/A	Natural number	instance_id, npu		
4	npu_macro1_serdes_lane3_snr	NPU Macro1 SerDes Lane 3 SNR	The SNR for SerDes Lane 3 in NPU Macro1	dB	N/A	Natural number	instance_id, npu		
5	npu_macro2_serdes_lane0_snr	NPU Macro2 SerDes Lane 0 SNR	The SNR for SerDes Lane 0 in NPU Macro2	dB	N/A	Natural number	instance_id, npu		
6	npu_macro2_serdes_lane1_snr	NPU Macro2 SerDes Lane 1 SNR	The SNR for SerDes Lane 1 in NPU Macro2	dB	N/A	Natural number	instance_id, npu		

No.	Metric	Display Name	Description	Unit	Conversion Rule	Value Range	Dimension	Supported Model	Supported CloudEye Agent Version
7	npu_macro2_serdes_lane2_snr	NPU Macro2 SerDes Lane 2 SNR	The SNR for SerDes Lane 2 in NPU Macro2	dB	N/A	Natural number	instance_id, npu		
8	npu_macro2_serdes_lane3_snr	NPU Macro2 SerDes Lane 3 SNR	The SNR for SerDes Lane 3 in NPU Macro2	dB	N/A	Natural number	instance_id, npu		
9	npu_macro3_serdes_lane0_snr	NPU Macro3 SerDes Lane 0 SNR	The SNR for SerDes Lane 0 in NPU Macro3	dB	N/A	Natural number	instance_id, npu		
10	npu_macro3_serdes_lane1_snr	NPU Macro3 SerDes Lane 1 SNR	The SNR for SerDes Lane 1 in NPU Macro3	dB	N/A	Natural number	instance_id, npu		
11	npu_macro3_serdes_lane2_snr	NPU Macro3 SerDes Lane 2 SNR	The SNR for SerDes Lane 2 in NPU Macro3	dB	N/A	Natural number	instance_id, npu		

No.	Metric	Display Name	Description	Unit	Conversion Rule	Value Range	Dimension	Supported Model	Supported CloudEye Agent Version
12	npu_macro3_serdes_lane3_snr	NPU Macro3 SerDes Lane 3 SNR	The SNR for SerDes Lane 3 in NPU Macro3	dB	N/A	Natural number	instance_id, npu		
13	npu_macro4_serdes_lane0_snr	NPU Macro4 SerDes Lane 0 SNR	The SNR for SerDes Lane 0 in NPU Macro4	dB	N/A	Natural number	instance_id, npu		
14	npu_macro4_serdes_lane1_snr	NPU Macro4 SerDes Lane 1 SNR	The SNR for SerDes Lane 1 in NPU Macro4	dB	N/A	Natural number	instance_id, npu		
15	npu_macro4_serdes_lane2_snr	NPU Macro4 SerDes Lane 2 SNR	The SNR for SerDes Lane 2 in NPU Macro4	dB	N/A	Natural number	instance_id, npu		
16	npu_macro4_serdes_lane3_snr	NPU Macro4 SerDes Lane 3 SNR	The SNR for SerDes Lane 3 in NPU Macro4	dB	N/A	Natural number	instance_id, npu		

No.	Metric	Display Name	Description	Unit	Conversion Rule	Value Range	Dimension	Supported Model	Supported CloudEye Agent Version
17	npu_macro5_serdes_lane0_snr	NPU Macro5 SerDes Lane 0 SNR	The SNR for SerDes Lane 0 in NPU Macro5	dB	N/A	Natural number	instance_id, npu		
18	npu_macro5_serdes_lane1_snr	NPU Macro5 SerDes Lane 1 SNR	The SNR for SerDes Lane 1 in NPU Macro5	dB	N/A	Natural number	instance_id, npu		
19	npu_macro5_serdes_lane2_snr	NPU Macro5 SerDes Lane 2 SNR	The SNR for SerDes Lane 2 in NPU Macro5	dB	N/A	Natural number	instance_id, npu		
20	npu_macro5_serdes_lane3_snr	NPU Macro5 SerDes Lane 3 SNR	The SNR for SerDes Lane 3 in NPU Macro5	dB	N/A	Natural number	instance_id, npu		
21	npu_macro6_serdes_lane0_snr	NPU Macro6 SerDes Lane 0 SNR	The SNR for SerDes Lane 0 in NPU Macro6	dB	N/A	Natural number	instance_id, npu		

No.	Metric	Display Name	Description	Unit	Conversion Rule	Value Range	Dimension	Supported Model	Supported CloudEye Agent Version
22	npu_macro6_serdes_lane1_snr	NPU Macro6 SerDes Lane 1 SNR	The SNR for SerDes Lane 1 in NPU Macro6	dB	N/A	Natural number	instance_id, npu		
23	npu_macro6_serdes_lane2_snr	NPU Macro6 SerDes Lane 2 SNR	The SNR for SerDes Lane 2 in NPU Macro6	dB	N/A	Natural number	instance_id, npu		
24	npu_macro6_serdes_lane3_snr	NPU Macro6 SerDes Lane 3 SNR	The SNR for SerDes Lane 3 in NPU Macro6	dB	N/A	Natural number	instance_id, npu		
25	npu_macro7_serdes_lane0_snr	NPU Macro7 SerDes Lane 0 SNR	The SNR for SerDes Lane 0 in NPU Macro7	dB	N/A	Natural number	instance_id, npu		
26	npu_macro7_serdes_lane1_snr	NPU Macro7 SerDes Lane 1 SNR	The SNR for SerDes Lane 1 in NPU Macro7	dB	N/A	Natural number	instance_id, npu		

No.	Metric	Display Name	Description	Unit	Conversion Rule	Value Range	Dimension	Supported Model	Supported Cloud Eye Agent Version
27	npu_macro7_serdes_lane2_snr	NPU Macro7 SerDes Lane 2 SNR	The SNR for SerDes Lane 2 in NPU Macro7	dB	N/A	Natural number	instance_id, npu		
28	npu_macro7_serdes_lane3_snr	NPU Macro7 SerDes Lane 3 SNR	The SNR for SerDes Lane 3 in NPU Macro7	dB	N/A	Natural number	instance_id, npu		

11.2 Using Cloud Eye to Monitor NPU Events of Lite Servers

Description

You can use Cloud Eye to centrally collect key events and cloud resource operational events. When an event occurs, you will receive an alarm. Lite Servers mainly support BMS and ECS events. The table below lists NPU-related events. For details about other events, see **Events Supported by Event Monitoring**.

Constraints

- The Cloud Eye Agent plugin, which has strict resource usage restrictions, is required for event reporting to Cloud Eye. When the resource usage exceeds the threshold, the Agent circuit breaker occurs. For details about the resource usage, see **Cloud Eye Server Monitoring**.
- You have fully tested the monitoring agent in the public image provided by Lite Servers. If you use your own image, perform the test before deploying the image in the production environment to prevent information errors.

Prerequisites

The Cloud Eye Agent has been installed on the Lite Server. For details about how to check whether the Cloud Eye Agent is installed and how to install it, see **Installing Cloud Eye Agent Monitoring Plug-ins**.

Event Source

BMS/ECS

Event Namespace

SYS.BMS/SYS.ECS/service.ModelArts

Event List

Table 11-14 Fault events supported by Lite Servers (BMS/ECS)

Event	Event ID	Event Severity	Description	Solution	Impact	Supported Model	Supported Cloud Eye Agent Version
NPU: device not found by npu-smi info	NPUSMICardNotFound	Major	The Ascend driver is faulty or the NPU is disconnected.	Reference: *ModelArts Troubleshooting* > "Lite Server" > "Handling Suggestions for NPUSMICardNotFound Events"	The NPU cannot be used normally.	Snt3P Snt3PD Snt9b Snt9b23	telescope: 2.7.4.3 2.7.5.3 2.7.5.4 2.7.5.9 or later

Event	Event ID	Event Severity	Description	Solution	Impact	Supported Model	Supported Cloud Eye Agent Version
NPU: PCIe link error	PCIeErrorFound	Major	The **lspci** command output shows that the NPU is in the **rev ff** state.	Reference: *ModelArts Troubleshooting* > "Lite Server" > "Handling Suggestions for PCIeErrorFound Events"	The NPU cannot be used normally.	Snt3P Snt3PD Snt9b Snt9b23	telescope: 2.7.4.3 2.7.5.3 2.7.5.4 2.7.5.9 or later
NPU: device not found by lspci	LspciCardNotFound	Major	The NPU is disconnected.	Reference: *ModelArts Troubleshooting* > "Lite Server" > "Handling Suggestions for LspciCardNotFound Events"	The NPU cannot be used normally.	Snt3P Snt3PD Snt9b Snt9b23	telescope: 2.7.4.3 2.7.5.3 2.7.5.4 2.7.5.9 or later

Event	Event ID	Event Severity	Description	Solution	Impact	Supported Model	Supported Cloud Eye Agent Version
NPU: overtemperature	TemperatureOverUpperLimit	Major	The temperature of DDR or software is too high.	Reference: *ModelArts Troubleshooting* > "Lite Server" > "Handling Suggestions for NPUErrorCode Warning Events"	The instance may be powered off and devices may not be found.	Snt3P Snt3PD Snt9b Snt9b23	telescope: 2.7.4.3 2.7.5.3 2.7.5.4 2.7.5.9 or later
NPU: uncorrectable ECC error	UncorrectableEccErrorCount	Major	There are uncorrectable ECC errors on the NPU.	Reference: *ModelArts Troubleshooting* > "Lite Server" > "Handling Suggestions for NPUErrorCode Warning Events"	Services may be interrupted.	Snt3P Snt3PD	telescope: 2.7.4.3 2.7.5.3 2.7.5.4 2.7.5.9 or later

Event	Event ID	Event Severity	Description	Solution	Impact	Supported Model	Supported Cloud Eye Agent Version
NPU: request for instance restart	RebootVirtualMachine	Suggestion	A fault occurs and the instance needs to be restarted.	Reference: *ModelArts Troubleshooting* > "Lite Server" > "Handling Suggestions for NPUErrorCode Warning Events"	Services may be interrupted.	Snt3P Snt3PD Snt9b Snt9b23	telescope: 2.7.4.3 2.7.5.3 2.7.5.4 2.7.5.9 or later
NPU: request for SoC reset	ResetSOC	Suggestion	A fault occurs and the SoC needs to be reset.	Reference: *ModelArts Troubleshooting* > "Lite Server" > "Handling Suggestions for NPUErrorCode Warning Events"	Services may be interrupted.	Snt3P Snt3PD Snt9b Snt9b23	telescope: 2.7.4.3 2.7.5.3 2.7.5.4 2.7.5.9 or later

Event	Event ID	Event Severity	Description	Solution	Impact	Supported Model	Supported Cloud Eye Agent Version
NPU: request for restart AI process	RestartAIProcess	Suggestion	A fault occurs and the AI process needs to be restarted.	Reference: *ModelArts Troubleshooting* > "Lite Server" > "Handling Suggestions for NPUErrorCode Warning Events"	The current AI task will be interrupted.	Snt3P Snt3PD Snt9b Snt9b23	telescope: 2.7.4.3 2.7.5.3 2.7.5.4 2.7.5.9 or later

Event	Event ID	Event Severity	Description	Solution	Impact	Supported Model	Supported Cloud Eye Agent Version
NPU: error codes	NPUErrorCodeWarning	Major	A large number of NPU error codes indicating major or higher-level errors are returned. You can further locate the faults based on the error codes.	Reference: *ModelArts Troubleshooting* > "Lite Server" > "Handling Suggestions for NPUErrorCodeWarning Events"	Services may be interrupted.	Snt3P Snt3PD Snt9b Snt9b23	telescope: 2.7.4.3 2.7.5.3 2.7.5.4 2.7.5.9 or later
Multiple NPU HBM ECC errors	NpuHbmMultiEccInfo	Suggestion	There are NPU HBM ECC errors.	This event is only a reference for other events. You do not need to handle it separately.	This event is only a reference for other events. You do not need to handle it separately.	Snt9b Snt9b23	telescope: 2.7.5.9 or later

Event	Event ID	Event Severity	Description	Solution	Impact	Supported Model	Supported Cloud Eye Agent Version
GPU: invalid RoCE NIC configuration	GpuRoceNicConfigIncorrect	Major	GPU: invalid RoCE NIC configuration	Reference: *ModelArts Troubleshooting* > "Lite Server" > "Handling Suggestions for GpuRoceNicConfigIncorrect Events"	The parameter plane network is abnormal, preventing the execution of the multi-node task.	GPU	telescope: 2.7.5.9 or later
ReadOnly issues in OS	ReadOnlyFileSystem	Critical	The file system %s is read-only.	Reference: *ModelArts Troubleshooting* > "Lite Server" > "Handling Suggestions for ReadOnlyFileSystem Events"	The files cannot be written or operated.	-	telescope: 2.7.5.3 2.7.5.9 or later

Event	Event ID	Event Severity	Description	Solution	Impact	Supported Model	Supported Cloud Eye Agent Version
NPU: driver and firmware not matching	NpuDriverFirmwareMismatch	Major	The NPU's driver and firmware do not match.	Reference: *ModelArts Troubleshooting* > "Lite Server" > "Handling Suggestions for NpuDriverFirmwareMismatch Events"	NPUs cannot be used.	Snt3P Snt3PD Snt9b Snt9b23	telescope: 2.7.5.3 2.7.5.9 or later
NPU: RoCE NIC down	RoCELinkStatusDown	Major	The RoCE link of NPU %d is down.	Reference: *ModelArts Troubleshooting* > "Lite Server" > "Handling Suggestions for RoCELinkStatusDown Events"	The NPU NIC is unavailable.	Snt9b Snt9b23	telescope: 2.7.5.3 2.7.5.9 or later

Event	Event ID	Event Severity	Description	Solution	Impact	Supported Model	Supported Cloud Eye Agent Version
NPU: RoCE NIC health status abnormal	RoCEHealthStatusError	Major	The RoCE network health status of NPU %d is abnormal.	Reference: *ModelArts Troubleshooting* > "Lite Server" > "Handling Suggestions for RoCEHealthStatusError Events"	The NPU NIC is unavailable.	Snt9b Snt9b23	telescope: 2.7.5.3 2.7.5.9 or later
NPU: RoCE NIC configuration file **/etc/hccn.conf** not exist	HccnConfNotExisted	Major	The RoCE NIC configuration file **/etc/hccn.conf** does not exist.	Reference: *ModelArts Troubleshooting* > "Lite Server" > "Handling Suggestions for HccnConfNotExisted Events"	The RoCE NIC is unavailable.	Snt9b Snt9b23	telescope: 2.7.5.3 2.7.5.9 or later

Event	Event ID	Event Severity	Description	Solution	Impact	Supported Model	Supported Cloud Eye Agent Version
GPU: basic components abnormal	GpuEnvironmentSystem	Major	The **nvidia-smi** command is abnormal.	1. Check whether the driver version is compatible with the GPU. Access the NVIDIA official website to check the GPU model and driver version. **https://www.nvidia.cn/drivers/lookup/** 2. Run **lsmod | grep nvidia** to check whether the NVIDIA kernel module is loaded. If not, run	The GPU driver is unavailable.	GPU	telescope: 2.7.5.3 2.7.5.9 or later

Event	Event ID	Event Severity	Description	Solution	Impact	Supported Model	Supported Cloud Eye Agent Version
				sudo modprobe nvidia to manually load it. 3. Reinstall the GPU driver. 4. If the fault persists, submit a service ticket and contact the O&M engineer.			
		Major	The nvidia-fabricmanager version was inconsistent with the GPU driver version.	Check whether the GPU driver version matches the nvidia-fabricmanager version.	The nvidia-fabricmanager cannot work properly, affecting GPU usage.		

Event	Event ID	Event Severity	Description	Solution	Impact	Supported Model	Supported Cloud Eye Agent Version
		Major	The container plugin nvidia-container-toolkit is not installed.	Install and configure the NVIDIA Container Toolkit by referring to its installation guide. **https://docs.nvidia.com/datacenter/cloud-native/container-toolkit/latest/install-guide.html**	Docker cannot attach GPUs.		

Event	Event ID	Event Severity	Description	Solution	Impact	Supported Model	Supported Cloud Eye Agent Version
Local disk attachment inspection	MountDiskSystem	Major	The **/etc/fstab** file contains invalid UUIDs.	Reference: *ModelArts Troubleshooting* > "Lite Server" > "Handling Suggestions for MountDiskSystem Events"	The disk attachment error resulted in abnormal server restart.	-	telescope: 2.7.5.3 2.7.5.9 or later
GP: incorrectly configured dynamic route for Ant series server	GpuRouteConfigError	Major	The dynamic route of the NIC %s of an Ant series server is not configured or is incorrectly configured. CMD [ip route]: %s | CMD [ip route show table all]: %s.	Submit a service ticket and contact the O&M engineer.	The NPU network communication is abnormal.	GPU	telescope: 2.7.5.3 2.7.5.9 or later

Event	Event ID	Event Severity	Description	Solution	Impact	Supported Model	Supported Cloud Eye Agent Version
NPU: RoCE port not split	RoCEUdpConfigError	Major	The RoCE UDP port is not split.	Reference: *ModelArts Troubleshooting* > "Lite Server" > "Handling Suggestions for RoCEUdpConfigError Events"	The communication performance of NPUs is affected.	Snt9b Snt9b23	telescope: 2.7.5.9 or later
Warning of automatic system kernel upgrade	KernelUpgradeWarning	Major	Warning of automatic system kernel upgrade. Old version: %s; new version: %s.	Reference: *ModelArts Troubleshooting* > "Lite Server" > "Handling Suggestions for KernelUpgradeWarning Events"	The AI software may be unavailable.	Snt3P Snt3PD Snt9b Snt9b23	telescope: 2.7.5.3 2.7.5.9 or later

Event	Event ID	Event Severity	Description	Solution	Impact	Supported Model	Supported Cloud Eye Agent Version
NPU environment command detection	NpuToolsWarning	Major	The hccn_tool is unavailable.	Reference: *ModelArts Troubleshooting* > "Lite Server" > "Handling Suggestions for NpuToolsWarning Events"	The IP address and gateway of the RoCE NIC cannot be configured.	Snt9b Snt9b23	telescope: 2.7.5.3 2.7.5.9 or later
		Major	The npu-smi is unavailable.	Reference: *ModelArts Troubleshooting* > "Lite Server" > "Handling Suggestions for NpuToolsWarning Events"	NPUs cannot be used.	Snt3P Snt3PD Snt9b Snt9b23	telescope: 2.7.5.3 2.7.5.9 or later

Event	Event ID	Event Severity	Description	Solution	Impact	Supported Model	Supported Cloud Eye Agent Version
		Major	The ascend-dmi is unavailable.	Reference: *ModelArts Troubleshooting* > "Lite Server" > "Handling Suggestions for NpuToolsWarning Events"	The ascend-dmi cannot be used for performance analysis.	Snt9b Snt9b23	telescope: 2.7.5.3 2.7.5.9 or later
NPU: L1 switch port partial failure	NpuL1SwitchPortPartialFunctionFailure	Major	Some functions of the NPU's L1 1520 switch port fail.	Reference: *ModelArts Troubleshooting* > "Lite Server" > "Handling Suggestions for NpuL1SwitchPortPartialFunctionFailure Events"	Services may be interrupted.	Snt9b23	telescope: 2.7.5.9 or later lqdcmi: 2.1.0 and later

Event	Event ID	Event Severity	Description	Solution	Impact	Supported Model	Supported Cloud Eye Agent Version
NPU: L1 switch fault	NpuL1SwitchFault	Major	There are faults in the L1 1520 switch of the NPU.	Reference: *ModelArts Troubleshooting* > "Lite Server" > "Handling Suggestions for NpuL1SwitchFault Events"	Services may be interrupted.	Snt9b23	telescope: 2.7.5.9 or later lqdcmi: 2.1.0 and later
NPU: Unmatched RoCE IP address	NpuRoceIPAddressMismatch	Major	The actual IP address of the RoCE NIC is inconsistent with the IP address in the hccn.conf configuration file.	Reference: *ModelArts Troubleshooting* > "Lite Server" > "Handling Suggestions for NpuRoceIPAddressMismatch Events"	The parameter plane network is abnormal, preventing the execution of the multi-node task.	Snt9b Snt9b23	telescope: 2.7.5.9 or later

Event	Event ID	Event Severity	Description	Solution	Impact	Supported Model	Supported Cloud Eye Agent Version
VRD target version differs from the current version	InconsistentNpuVrdVersion	Major	VRD target version differs from the current version.	Upgrade VRD by powering off, waiting 5 minutes, then powering it back on.	The event has no short-term impacts, but it may affect NPU stability over time.	Snt9b Snt9b23	-

11.3 Using Cloud Eye to Monitor Lite Servers and Generate Event Alarms

Scenario

Cloud Eye allows you to set alarms for monitored metrics and events. You will receive notifications about faults via SMS or email. Fault records can also be accessed using APIs.

Constraints

- This solution is implemented based on Cloud Eye alarm rules. As an account can create a maximum of 100 alarm rules on Cloud Eye, at most 100 supernodes can be monitored.
- Cloud Eye host monitoring agency needs to be enabled as the alarms are generated based on Cloud Eye fault detection events. You can enable this agency when purchasing the supernode or create the agency on the Cloud Eye console after the purchase. For details, see **Permissions**.
- Alarm notifications are sent via SMS messages and emails using Simple Message Notification (SMN). There will be certain fees. For details, see **Product Pricing Details**.

Procedure

1. Log in to the **Cloud Eye console**.
2. Create an alarm rule template.

Table 11-15 Parameters

Parameter	Recommended Value
Name	You are advised to name the template after the fault severity, for example, **Subhealthy supernode**.
Alarm Type	Event
Method	Select **Configure manually**. The recommended settings for other parameters are as follows: • **Event Name**: Select the target events by referring to **Event List**. • **Alarm Policy**: Generate the alarm once if the event occurs four times within 5 minutes. **Note: Improper configurations may cause too many alarms or slow response.** • **Alarm Severity**: Select **Major**.

3. Create an alarm rule.

Table 11-16 Alarm rule parameters

Parameter	Recommended Value
Name	You are advised to name the alarm rule in the *<Supernode-name>_<Fault-level>* format, for example, **SuperPod_01_Subhealthy**.
Alarm Type	Event
Event Type	System event
Event Source	Elastic Cloud Server
Monitoring Scope	Specific resources
Instance	All subnodes in the supernode. Click **Select Specific Resources**, search for the supernode name, select all, and click **OK**.
Method	Configure manually
Alarm Policy	Enable **Use Template** and select the alarm template created in **2** from the drop-down list.

Parameter	Recommended Value
Alarm Notifications	(Optional) Enable this if you want to receive alarm notifications by SMS, email, HTTP, or HTTPS. **Note**: SMN charges you for SMS, email, HTTP, and HTTPS messages. For details, see **Product Pricing Details**.
Recipient	(Optional) This parameter is available only when **Alarm Notifications** is enabled. You are advised to create a topic.
Notification Window	(Optional) This parameter is available only when **Alarm Notifications** is enabled. The default value is recommended.
Trigger Condition	(Optional) This parameter is available only when **Alarm Notifications** is enabled. The default value is recommended.
Enterprise Project	Set this parameter based on the real-life situation.

4. (Optional) Create a topic.

   Table 11-17 Parameters for creating a topic

Parameter	Recommended Value
Topic Name	Enter a name in English, for example, **SuperPod-Sub-Health**.
Display Name	Name displayed in the email subject. Set it to the fault severity, for example, **Subhealthy supernode**.
Enterprise Project	Set this parameter based on the real-life situation.

5. (Optional) Add a subscription. After creating a topic, add subscriptions to receive alarm notifications.

   Then, the terminal will receive a subscription confirmation. Confirm the subscription to receive alarm notifications.

Email Alarm Notification Example

In an alarm notification email, the subject displays the alarm severity, the content displays key information such as the alarm object, alarm policy, and alarm time. The alarm rule contains the name of the supernode to which the fault object belongs. For details about how to handle alarms, see **Event List**.

Querying Alarm Records

You can query alarm records through APIs. For details, see **Alarm Records**.

11.4 Using Cloud Eye to Automatically Detect Server Environments and Provide Event Notifications

Description

In routine system O&M, administrators must frequently inspect machine environments to ensure system stability and security. However, traditional inspection methods often rely on manual operations, which are time-consuming, labor-intensive, and prone to overlooking latent issues. To address these challenges, Cloud Eye Agent now features automated machine environment inspection. With this capability, the Cloud Eye Agent automatically detects vulnerabilities and configuration issues within the environment. It then reports these findings to the Lite Server, where they are displayed as event notifications on the console. You are also provided with direct links to the corresponding repair tasks. This ensures that you can identify system issues in real time and take immediate corrective action, significantly enhancing both the security and stability of the system.

Constraints

Only Snt9b nodes and Snt9b23 supernodes are supported.

New events and recovery events can only be reported after Cloud Eye Agent of the latest version is installed.

Prerequisites

Cloud Eye Agent has been installed on the Lite Server by default. If it is not installed, install it by referring to **Installing or Upgrading the Cloud Eye Agent Plugin on a Lite Server**.

Viewing Events

Log in to the **ModelArts console**. In the navigation pane, choose **Resource Management** > **Lite Servers**. The **Resources** tab is displayed.

- New console: In the navigation pane, choose **Resource Management** > **Lite Compute Resources** > **Lite Servers**.
- Old console: In the navigation pane, choose **Resource Management** > **Lite Servers**.

In the upper left corner of the page, you can view the number of event notifications. Click the number to go to the event notification details page and view the details.

Figure 11-3 Event notifications

On the displayed page, view the event details and handling suggestions. Click **Repair** in the **Operation** column to rectify the fault.

Figure 11-4 Event notification details page

Table 11-18 Events that support automatic detection and notification

No.	Check Item	Description	Solution
1	System software OpenSSH vulnerability	This event is reported when the system software version is earlier than 8.8p1-2.r34.	Call the Ascend system configuration task in the Lite Server task center to upgrade the system software.
2	System software ipvs-fnat vulnerability	This event is reported when the system software version is earlier than 1.0.1-161.r5.	Call the Ascend system configuration task in the Lite Server task center to upgrade the system software.

No.	Check Item	Description	Solution
3	Abnormal MCU upgrade process in the **firmware_check.sh** script	This event is reported when **firmware_check.sh** or **upgrade_mcu.sh** does not exist in the **/opt/huawei/firmware_check** directory, or **firmware_check.sh** does not contain any identifier (the version is considered as an earlier version).	Call the Ascend system configuration task in the Lite Server task center to replace the script with the latest one.
4	No UDP hash configured for old HCE images (exist for images earlier than RC3.3)	This event is reported when **uplink_hash_config.py** or **uplink_hash_config.sh** does not exist in the **/opt/huawei/port_config** directory.	Call the Ascend system configuration task in the Lite Server task center to replace the script with the latest one.
5	New configuration for crash_kernel	This event is reported when the kernel.printk, crashkernel, and kernel.softlockup_panic configurations are not optimal.	Call the Ascend system configuration task in the Lite Server task center to modify the configuration to the latest specified one.
6	Cloud Eye Agent incompatible with HDK	This event is reported when the Cloud Eye Agent version is earlier than 2.8.2.2 and the HDK version is earlier than 25.*x*.	Call the Ascend software upgrade task in the Lite Server task center to upgrade Cloud Eye Agent to the specified version.

11.5 Using DCGM to Monitor GPU Resources of Lite Servers

Description

This section describes how to configure DCGM monitoring on Lite Servers to monitor GPU resources.

DCGM is an integrated tool for managing and monitoring large-scale 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.

Constraints

Only GPU resources can be monitored.

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 Container Toolkit

Set the repository address and GPG key.

```
distribution=$(. /etc/os-release;echo $ID$VERSION_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.xx:9400'] # Port for obtaining DCGM-Exporter metrics. Replace xx.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.

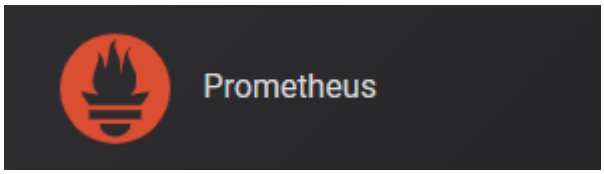
```
docker run -d -p 3000:3000 grafana/grafana-oss
```

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**.

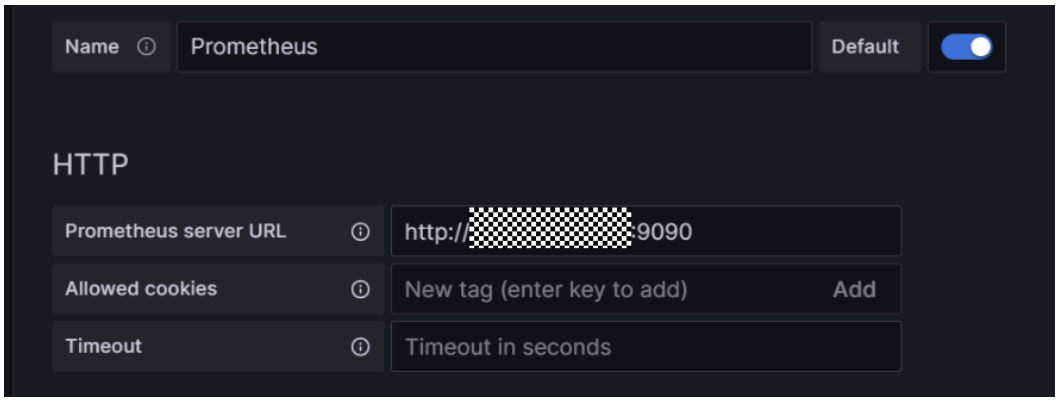
Note: *xx.xx.xx.xx* is the IP address of the host machine where Grafana is located.

Figure 11-5 Prometheus



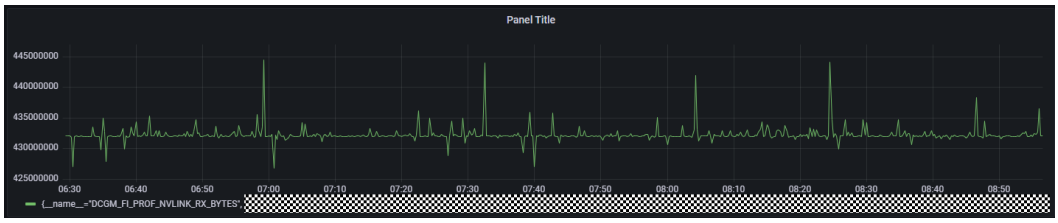
Enter the Prometheus IP address and port number in the HTTP URL text box and click **Save&Test**.

Figure 11-6 IP address and port number



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

Figure 11-7 Metric monitoring



NOTE

The basic functions of Grafana are described. For higher requirements, see the **official Grafana document**.

12 Managing CloudPond NPU Resources Using Lite Servers

Description

The CloudPond Ascend compute resources you have been purchased must be provisioned by Lite Servers. This section describes how to provision resources through Lite Servers.

CloudPond is similar to an edge AZ of Huawei Cloud. 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?**

Lite Servers only manage the lifecycle of Ascend compute resources on CloudPond, such as provisioning, starting, stopping, and deleting them.

Constraints

Lite Servers can only manage Snt9b resources on CloudPond. They cannot manage GPU or CPU resources.

Only the new Lite Server purchase page supports the provisioning of CloudPond resources.

Prerequisites

- You have purchased CloudPond. For details, see **Getting Started with CloudPond**.
- Contact the customer manager to determine a Lite Server resource solution. Then, apply for the required resource specifications. Alternatively, submit a service ticket.
- Increase the resource quota. For details, see **Step 2: Increasing Your Resource Quota**.
- Enable basic permissions. For details, see **Step 3: Enabling Basic Permissions**.
- Configure ModelArts agency authorization. For details, see **Step 4: Creating an Agency Authorization on ModelArts**.

Billing

Provisioning and managing CloudPond resources on a Lite Server are free of charge. The CloudPond resource fee is actually charged when you purchase CloudPond resources.

Provisioning CloudPond Resources

Creating a Lite Server is the process of provisioning CloudPond resources.

1. Log in to the **ModelArts console**. In the navigation pane, choose **Resource Management** > **Lite Servers**. The **Resources** tab is displayed.
   - New console: In the navigation pane, choose **Resource Management** > **Lite Compute Resources** > **Lite Servers**.
   - Old console: In the navigation pane, choose **Resource Management** > **Lite Servers**.
2. Click **Buy Lite Server** in the upper right corner. Configure the parameters on the displayed page.

   NOTE

   There are two versions of the page for purchasing Lite Servers. Only the new version supports the provisioning of CloudPond resources.

Table 12-1 Parameters for basic configurations

Parameter	Description
Type	Select **Common node**.
Resource Type	Select **ECS**.
Billing Mode	Select **Pay-per-use**. Currently, Lite Servers are created free of charge.
Region	Select the region where the purchased Lite Site resources are located.
AZ	Select **Lite Site** and select existing CloudPond resources from the drop-down list. If no CloudPond resource is available, create one by referring to **Getting Started with CloudPond**. **Lite Site** is only displayed after you have purchased CloudPond. For details about how to purchase CloudPond, see **CloudPond > Getting Started**.

Table 12-2 Parameters for resource configurations

Parameter	Description
CPU Architecture	CPU architecture of the resource type. Lite Servers can only manage Ascend Snt9b resources of CloudPond. Therefore, select **Arm**. Select a CPU architecture and then select instance specifications as required. The flavors vary by region. The actual flavors are displayed on the console. **NOTE** If no specifications are available, contact **Huawei technical support**.

Table 12-3 Parameters for configuring the OS

Parameter	Description
Image	• **Public image** Public images are available for all users. All users can read the image by image ID. ModelArts allows you to perform development and training directly without additional configuration as it provides multiple public images, supports multiple OSs, and has built–in AI drivers and software. For details about the supported public images, see **Mappings Between Lite Compute Nodes and OS Images**. • **Private image** Only the image creator can use the image. You can select a private image to save your time from repeatedly configuring servers.

Table 12-4 Parameters for configuring storage

Parameter	Description
Node System Disk Type	This parameter is only displayed when you select an instance flavor that supports attaching. The node data disk stores the OS of a server, and is automatically created and initialized upon Lite Server creation. Select a node system disk type and set the disk size. You can also expand the system disk capacity after the Lite Server is created. The system disk is automatically attached to each compute node.

Parameter	Description
(Optional) Node Data Disk Type	Click **Add Data Disk** to attach an EVS data disk to the Lite Server. Currently, local disks cannot be attached. You can select **Node Data Disk Type** and set **Size** and **Quantity**. The data disk size ranges from 100 GiB to 32,768 GiB. For ECSs, there can be a maximum of 59 data disks. You can also expand the data disk capacity after the server is created. The data disk is automatically attached to each compute node.

Table 12-5 Parameters for configuring the network

Parameter	Description
VPC	A Virtual Private Cloud (VPC) ensures the security, isolation, and network flexibility of server resources. Choose the VPC associated with your server from the drop–down list. You are advised to choose the same VPC for all related cloud services to simplify network connections. If no VPC is available in the drop–down list, click **Create VPC** on the right to create a VPC. To create a VPC, you need to log in to the management console as the administrator.
Subnet	Select the subnet under the CloudPond edge region of the VPC. If no subnet is available in the drop–down list, click **Create Subnet** on the right to create one.
Security group	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. If no security group is available in the drop–down list, click **Create Security Group** on the right to create one.
IPv6	IPv6 is available when it is supported by the subnet, specifications, and image configured for the network. Ensure that IPv6 has been enabled. To enable IPv6, see **Creating a Subnet for an Existing VPC**. This parameter is only displayed for certain specifications and images.

Parameter	Description
RoCE Network	When Ascend Snt9b resources are used for distributed training, you need to configure the RoCE network to use the RoCE NICs on the hardware. The parameter is only displayed if you have selected one specification that supports RoCE networks. If you have not created a RoCE network, click **Create RoCE**. If you have created a RoCE network, select it directly.

Table 12-6 Parameters for management configurations

Parameter	Description
Server Name	Lite Server name. which can contain 1 to 64 characters. Only digit, letters, underscores (_), and hyphens (-) are allowed.
Login Mode	**Key pair** is recommended as it features higher security than **Password**. If you select **Password**, ensure that the password meets complexity requirements to prevent malicious attacks. • **Key pair** Use a key pair to log in to the Server node. You can select an existing key pair, or click **Create Key Pair** to create one. **NOTE** If you use an existing key pair, ensure that you have saved the key file locally. Otherwise, logging in to the Server node will fail. • **Password** A username and its initial password are used for authentication and logging in to the Server node. For Linux, use the initial password of user **root**. For Windows, use the initial password of user **Administrator**. The password must: – Contain 8 to 26 characters. – Contain at least three types of the following characters: uppercase letters, lowercase letters, digits, and special characters (!@$%^-_=+[{}]:,./?). – Cannot be the same as the username or the username spelled backwards. – Cannot contain root, administrator, or their reverse.

Parameter	Description
Enterprise Project	This parameter is only available if you have enabled the enterprise project function, or if your account is an enterprise account. To enable this function, contact your customer manager. An enterprise project makes it easy to manage projects and groups of cloud resources and users. Use the default enterprise project or create one. Select an enterprise project from the drop–down list. For details about enterprise projects, see ***Enterprise Management User Guide***. **CAUTION** The enterprise project cannot be modified for a purchased server. Currently, the enterprise project information cannot be synchronized in the order.

Table 12-7 Parameters for advanced configurations

Parameter	Description
Cloud Eye host monitoring	Enable this function. Once this function enabled, you can configure Cloud Eye host monitoring agency in one–click mode. Cloud Eye agency allows you to monitor various metrics of the server, including CPU, memory, network, disk, and process at an interval of 1 minute. For details, see **Using Cloud Eye to Monitor NPU Resources of Lite Servers**.

Table 12-8 Parameters for purchase configurations

Parameter	Description
Quantity	You can purchase multiple instances simultaneously, with a value between **1** and **10**.

3. Click **Buy now**. Currently, Servers are provisioned free of charge.
4. The resource will be created in 20 to 60 minutes. If the resource fails to be created, see **Handling Resource Purchase Failures**.

13 Using CTS to Audit Operations on Lite Servers

ModelArts Lite Servers can connect to CTS. With CTS, you can obtain operations associated with ModelArts Lite Servers for later query, audit, and backtrack operations.

Constraints

CTS stores operation records of ModelArts Lite Servers from the last seven days.

Prerequisites

You have enabled CTS. For details, see **Cloud Trace Service User Guide**.

Key Lite Server Operations Supported by CTS

Table 13-1 Key Lite Server operations supported by CTS

Operation	Resource Type	Trace
Update	Server	batchactionServer
Update	ServerHyperinstance	scaledownhyperinstanceServerHyperinstance
Update	ServerHyperinstance	scaleuphyperinstanceServerHyperinstance
Create	Server	createServer
Update	Server	stopServer
Update	Server	startServer
Update	Server	rebootServer
Update	Server	updateServer
Delete	Server	deleteServer

Operation	Resource Type	Trace
Read	Server	getServer
Update	Server	syncServer
Read	Server	listbyuserServer
Read	Server	listServer
Update	Server	changeosServer
Update	Server	reinstallosServer
Update	Server	attachvolumeServer
Update	Server	detachvolumeServer
Read	Server	getoperationServer
Read	ServerHyperinstance	gethyperinstancescaleevaluations-ServerHyperinstance
Read	ServerHyperinstance	listhyperinstanceclusterscapacity-ServerHyperinstance
Update	Server	bindpublicipServer
Read	Server	listpublicipServer
Create	AIServerJob	createjobAIServerJob
Read	AIServerJob	getjobAIServerJob
Read	Server	gettopologiesServer
Read	AIServerJob	listjobsAIServerJob
Delete	AIServerJob	deletejobsAIServerJob
Read	AIServerJobTemplate	listjobtemplatesAIServerJobTem-plate
Read	AIServerJobTemplate	getjobtemplateAIServerJobTem-plate
Read	AIServerService	getjobserviceAIServerService
Read	Server	listflavorsServer
Create	ServerRoceNetwork	createrocenetworkServerRoceNet-work
Create	HyperCluster	createhyperclusterHyperCluster
Read	HyperCluster	gethyperclusterHyperCluster
Read	HyperCluster	listhyperclusterHyperCluster
Delete	HyperCluster	deletehyperclusterHyperCluster

Operation	Resource Type	Trace
Read	ServerImage	listimagesServerImage
Read	ServerImage	getimageServerImage
Read	ServerHyperinstance	listhyperinstancebyuserServerHyperinstance
Read	ServerHyperinstance	gethyperinstanceServerHyperinstance
Delete	ServerHyperinstance	deletehyperinstanceServerHyperinstance
Update	ServerHyperinstance	changehyperinstanceosServerHyperinstance
Read	ServerHyperinstance	gethyperinstanceoperationServerHyperinstance
Update	ServerHyperinstance	starthyperinstanceServerHyperinstance
Update	ServerHyperinstance	stophyperinstanceServerHyperinstance
Read	ServerHyperinstance	queryhyperinstancetagsServerHyperinstance
Create	ServerHyperinstance	createhyperinstancetagsServerHyperinstance
Delete	ServerHyperinstance	deletehyperinstancetagsServerHyperinstance
Read	ServerHyperinstance	listhyperinstanceServerHyperinstance
Update	Server	attachportServer
Update	Server	detachportServer
Read	AIServerAgentResponse	listagentsAIServerAgentResponse

Viewing Audit Logs on the CTS Console

1. Log in to the **CTS console**.
2. Click  in the upper left corner and select a region.
3. In the navigation pane on the left, choose **Trace List**.
4. Specify filters as needed. You can query traces using a combination of the following filters:

- **Trace Source**, **Resource Type**, and **Search By**:

  Select a filter from the drop-down list.

  When you select **Trace Name**, you need to enter a specific trace name.

  When you select **Resource ID**, you need to enter a specific resource ID.

  When you select **Resource Name**, you need to enter a specific resource name.
- **Operator**: Select a specific operator (a user rather than a tenant).
- **Trace Status**: Select **All trace statuses**, **Normal**, **Warning**, or **Incident**.
- **Time Range**: You can query traces generated during any time range of the last seven days.

5. Click ⌄ on the left of a trace to expand its details.
6. Locate the target trace and click **View Trace** in the **Operation** column. In the displayed **View Trace** dialog box, view the trace structure details.

   For details about the CTS trace structure, see **CTS User Guide**.

14 Unsubscribing from a Lite Server

Description

You can delete or unsubscribe from a Lite Server that is no longer used to release resources. For details about how to stop billing, see **Stopping Billing**.

Constraints

- If a yearly/monthly Lite Server common node, an entire rack of resources, or a supernode is in the **Create Failed** or **Error** state, you can directly delete it.
- During the grace period or frozen period, yearly/monthly Lite Server resources, including common nodes, full-rack resources, or supernode resources, cannot be released using the **Unsubscribe** function. In such cases, you can release these resources directly using the **Release** function.
- When you unsubscribe from or release a yearly/monthly Lite Server, the entire order is canceled. You cannot release or unsubscribe from a single child node under a rack or supernode.
- When you delete a yearly/monthly Lite Server, for ECS and BMS server types and supernode resources, the data disks set on the server creation page will be automatically deleted as well. Data disks that are attached after the server is created will not be deleted.
- When you delete a pay-per-use Lite Server, for ECS and BMS server types, the data disks set on the server creation page will not be deleted. Data disks that are attached after the server is created will not be deleted. For supernode resources, the data disks configured during the server creation process will be deleted upon resource deletion. However, any data disks manually attached after the server was created will not be deleted.
- For common nodes, you can perform operations such as batch unsubscription, batch deletion, and batch release. Such features are not supported for supernodes.

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 or multiple instances in batches on the ModelArts console.

- Method 2: Unsubscribe from a single or multiple instances in the Billing Center.

Method 1: Unsubscribe from the resource on the ModelArts console.

1. Log in to the **ModelArts console**. In the navigation pane, choose **Resource Management** > **Lite Servers**. The **Resources** tab is displayed.
   - New console: In the navigation pane, choose **Resource Management** > **Lite Compute Resources** > **Lite Servers**.
   - Old console: In the navigation pane, choose **Resource Management** > **Lite Servers**.
2. In the common node or supernode list, click **View all** to view all servers.

   NOTE

   If an agency needs to be configured, contact your account administrator. For details, see **Configuring an Agency for ModelArts**.
3. In the common node or supernode list, locate the target server and choose **More** > **Unsubscribe** in the **Operation** column.

   In the common node list, select multiple Lite Servers, and choose **More** > **Unsubscribe** above the list.
4. Confirm the resources to be unsubscribed from and select the reason for unsubscription.
5. 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**.
6. Click **Confirm** and confirm the resources to be unsubscribed from.
7. Click **Unsubscribe** again to unsubscribe from the yearly/monthly resources.

Method 2: Unsubscribe from a single instance in the Billing Center.

1. Log in to the **ModelArts console**. In the navigation pane, choose **Resource Management** > **Lite Servers**. The **Resources** tab is displayed.
   - New console: In the navigation pane, choose **Resource Management** > **Lite Compute Resources** > **Lite Servers**.
   - Old console: In the navigation pane, choose **Resource Management** > **Lite Servers**.
2. In the common node or supernode list, unsubscribe from the target server.
3. Hover the cursor over the node name and copy the ID of the instance to be unsubscribed from.

   NOTE

   The Resource ID bound to a yearly/monthly purchase order for a Lite Server is the specific Lite Server ID. This is distinct from the underlying BMS/ECS IDs encapsulated by the product. To unsubscribe from a Lite Server, you must locate the corresponding ID in the **Resources** tab on the ModelArts console.
4. Choose **Billing** from the top menu bar. In the navigation pane on the left, choose **Orders** > **Unsubscriptions**.
5. Search for the instance ID, confirm the information, and click **Unsubscribe from Resource** in the **Operation** column.

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

- **Unsubscribing from multiple instances in the Billing Center**

1. Log in to the **ModelArts console**. In the navigation pane, choose **Resource Management** > **Lite Servers**. The **Resources** tab is displayed.
   - New console: In the navigation pane, choose **Resource Management** > **Lite Compute Resources** > **Lite Servers**.
   - Old console: In the navigation pane, choose **Resource Management** > **Lite Servers**.
2. In the common node or supernode list, unsubscribe from the target 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**.
5. Select the target instances and click **Batch Unsubscribe**.
6. Confirm the resources to be unsubscribed from and select the reason for unsubscription.
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.

Releasing a Frozen Yearly/Monthly Lite Server

If yearly/monthly Lite Server common nodes, an entire rack of resources, or supernodes are in the grace period or frozen, you cannot unsubscribe from the resources to release them. Instead, use the release function to release them.

1. Log in to the **ModelArts console**. In the navigation pane, choose **Resource Management** > **Lite Servers**. The **Resources** tab is displayed.
   - New console: In the navigation pane, choose **Resource Management** > **Lite Compute Resources** > **Lite Servers**.
   - Old console: In the navigation pane, choose **Resource Management** > **Lite Servers**.
2. In the common node or supernode list, unsubscribe from the target server.

3. Locate the target server in the list and choose **More** > **Release** in the **Operation** column. In the displayed dialog box, confirm the information and click **OK**.

   Alternatively, access the server details page, and choose ••• > **Release** in the upper right corner. In the displayed dialog box, confirm the information and click **OK**.

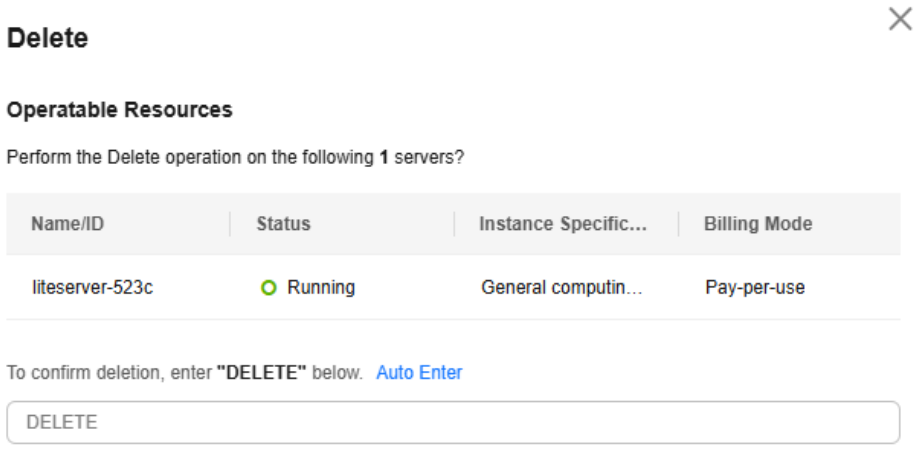
   In the common node list, select multiple Lite Servers, and choose **More** > **Release** above the list.

Deleting a Lite Server

Yearly/Monthly Lite Servers can only be deleted when they are in the **Create Failed** or **Error** state.

1. Log in to the **ModelArts console**. In the navigation pane, choose **Resource Management** > **Lite Servers**. The **Resources** tab is displayed.
   - New console: In the navigation pane, choose **Resource Management** > **Lite Compute Resources** > **Lite Servers**.
   - Old console: In the navigation pane, choose **Resource Management** > **Lite Servers**.
2. In the common node or supernode list, unsubscribe from the target server.
3. Locate the target server in the list and choose **More** > **Delete** in the **Operation** column. In the displayed dialog box, confirm the information, enter **DELETE**, and click **OK**.

Figure 14-1 Deleting a Server instance



In the common node list, select multiple Lite Servers, and choose **More** > **Delete** above the list.

Figure 14-2 Batch deletion