Kunpeng BoostKit for HPC

Installation Guide

Issue 06

Date 2022-01-30

HUAWEI TECHNOLOGIES CO., LTD.

Trademarks and Permissions


and other Huawei trademarks are trademarks 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 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.

Contents

1 Basic Environment Setup Guide

1.1 Introduction

1.1.1 Purpose

This document describes how to prepare the basic environment for porting HPC solution industry applications on the Kunpeng computing platform. All operations in the document are performed after the OS of the server is installed.

1.1.2 Scenarios

This document applies to the following high performance computing (HPC) scenarios:

1. InfiniBand (IB) network cluster scenario. The IB network cluster supports Message Passing Interface (MPI) parallelism of most HPC applications and is the mainstream HPC network scenario.
2. Single-node scenario. This scenario does not support multi-node parallelism of applications such as most applications in the gene industry.

1.2 Environment Requirements

Hardware Requirements

Table 1-1 lists the hardware requirements.

Table 1-1 Hardware requirements

Item	Description
CPU	Kunpeng 920

OS Requirements

Table 1-2 lists the OS requirements.

Table 1-2 OS requirements

Item	Version	How to Obtain
CentOS	7.6	**https://www.centos.org/download/**
Kernel	4.14.0-115	Included in the OS image.

Other Software

Table 1-3 lists other required software.

Table 1-3 Other software

Item	Installation Package	How to Obtain
GNU	gcc-9.3.0.tar.xz	**https://ftp.gnu.org/gnu/gcc/gcc-9.3.0/**
GMP	gmp-6.1.0.tar.bz2	**http://gcc.gnu.org/pub/gcc/infrastructure/**
MPC	mpc-1.0.3.tar.gz	**http://gcc.gnu.org/pub/gcc/infrastructure/**
MPFR	mpfr-3.1.4.tar.bz2	**http://gcc.gnu.org/pub/gcc/infrastructure/**
Open MPI	openmpi-4.0.3.tar.gz	**https://download.open-mpi.org/release/open-mpi/v4.0/openmpi-4.0.3.tar.gz**
InfiniBand driver	MLNX_OFED_LINUX-4.6-1.0.1.1-rhel7.6alternate-aarch64.tgz	**https://content.mellanox.com/ofed/MLNX_OFED-4.6-1.0.1.1/MLNX_OFED_LINUX-4.6-1.0.1.1-rhel7.6alternate-aarch64.tgz**

1.3 Planning the Installation Paths

This chapter lists the software installation paths involved in the setup of the HPC solution basic environment.

Table 1-4 Installation paths

No.	Software Installation Path	Description	Remarks
1	/path/to/GMP	Installation path of GMP	The installation paths listed in this table are only examples. Shared paths are recommended. Replace the paths used in commands in this document with the actual paths planned during the installation process.
2	/path/to/MPC	Installation path of MPC	
3	/path/to/MPFR	Installation path of MPFR	
4	/path/to/GNU	Installation path of GNU	
5	/path/to/OPENMPI	Installation path of Open MPI	
6	/path/to/INFINIBAND	Path for storing the InfiniBand NIC driver package	
7	/path/to/ISO	Path for storing the OS image package	

1.4 Setting Up the Environment for the Cluster Scenario

1.4.1 Prerequisites

The required installation packages have been uploaded to the planned installation directories on the server using an SFTP tool.

1.4.2 Configuring the Local Yum Source

Procedure

Step 1 Use PuTTY to log in to the server as the **root** user.

Step 2 Run the following command to create the **/cdrom** directory:

```
mkdir /cdrom
```

Step 3 Run the following command to mount the OS image package to the **/cdrom** directory:

```
mount /path/to/ISO/CentOS-7-aarch64-Everything-1810.iso /cdrom
```

Step 4 Run the following commands to remove all files in the **/etc/yum.repos.d** directory:


```
cd /etc/yum.repos.d
mkdir bak
mv *.repo bak
```

Step 5 Run the following command to create a Yum source configuration file:

1. Create a Yum source configuration file.
   ```
   vi /etc/yum.repos.d/CentOS-base.repo
   ```
2. Press **i** to enter the insert mode and add the following content:
   ```
   [CentOS7.6-source]
   Name=CentOS 7.6 Repo
   baseurl=file:///cdrom
   enabled=1
   gpgcheck=0
   ```
3. Press **Esc**, type **:wq!**, and press **Enter** to save the file and exit.

----End

1.4.3 Installing GMP

Procedure

Step 1 Use PuTTY to log in to the server as the **root** user.

Step 2 Run the following commands to decompress the GMP installation package:

```
tar -xvf gmp-6.1.0.tar.bz2
```

Step 3 Run the following command to go to GMP source code directory:

```
cd gmp-6.1.0
```

Step 4 Run the following commands to perform compilation and installation:

```
./configure --prefix=/path/to/GMP
make
make install
```

Step 5 Run the following command to set the environment variable:

```
export LD_LIBRARY_PATH=/path/to/GMP/lib:$LD_LIBRARY_PATH
```

----End

1.4.4 Installing MPFR

Procedure

Step 1 Use PuTTY to log in to the server as the **root** user.

Step 2 Run the following commands to decompress the MPFR installation package:

```
tar -xvf mpfr-3.1.4.tar.bz2
```

Step 3 Run the following command to go to MPFR source code directory:

```
cd mpfr-3.1.4
```

Step 4 Run the following commands to perform compilation and installation:

```
./configure --prefix=/path/to/MPFR --with-gmp=/path/to/GMP
make
make install
```

Step 5 Run the following command to set the environment variable:


```
export LD_LIBRARY_PATH=/path/to/MPFR/lib:$LD_LIBRARY_PATH
```

----End

1.4.5 Installing MPC

Procedure

Step 1 Use PuTTY to log in to the server as the **root** user.

Step 2 Run the following commands to decompress the MPC installation package:

```
tar -xvf mpc-1.0.3.tar.gz
```

Step 3 Run the following command to go to MPC source code directory:

```
cd mpc-1.0.3
```

Step 4 Run the following commands to perform compilation and installation:

```
./configure --prefix=/path/to/MPC --with-gmp=/path/to/GMP --with-mpfr=/path/to/MPFR
make
make install
```

Step 5 Run the following command to set the environment variable:

```
export LD_LIBRARY_PATH=/path/to/MPC/lib:$LD_LIBRARY_PATH
```

----End

1.4.6 Installing GNU

Procedure

Step 1 Use PuTTY to log in to the server as the **root** user.

Step 2 Run the following command to decompress the GNU installation package:

```
tar -vxf gcc-9.3.0.tar.xz
```

Step 3 Run the following command to go to GNU source code directory:

```
cd gcc-9.3.0
```

Step 4 Run the following commands to perform compilation and installation:

```
./configure --disable-multilib --enable-languages="c,c++,fortran"  --prefix=/path/to/GNU --disable-static --enable-shared --with-gmp=/path/to/GMP --with-mpfr=/path/to/MPFR --with-mpc=/path/to/MPC
make
make install
```

Step 5 Run the following commands to load the environment variables:

```
export PATH=/path/to/GNU/bin:$PATH
export LD_LIBRARY_PATH=/path/to/GNU/lib64:$LD_LIBRARY_PATH
```

----End

1.4.7 Installing the InfiniBand NIC Driver

Procedure

Step 1 Use PuTTY to log in to the server as the **root** user.

Step 2 Run the following commands to install the system dependency package:

```
yum install tcsh tcl lsof tk -y
```


Step 3 Run the following commands to recompile the InfiniBand driver:

```
tar -xzvf MLNX_OFED_LINUX-4.6-1.0.1.1-rhel7.6alternate-aarch64.tgz
RPM_BUILD_NCPUS=16 ./mlnxofedinstall --add-kernel-support-build-only --without-depcheck --skip-distro-check
```

After the installation is complete, a new installation package **MLNX_OFED_LINUX-4.6-1.0.1.1-rhel7.6alternate-ext.tgz** is generated in the **/tmp** directory.

Step 4 Run the following commands to decompress the installation package:

```
cd /tmp/MLNX_OFED_LINUX-4.6-1.0.1.1-4.14.0-115.el7a.0.1.aarch64
tar -xvf MLNX_OFED_LINUX-4.6-1.0.1.1-rhel7.6alternate-ext.tgz
```

Step 5 Run the following command to install the InfiniBand driver:

```
cd MLNX_OFED_LINUX-4.6-1.0.1.1-rhel7.6alternate-ext
./mlnxofedinstall
```

Step 6 Run the following commands to restart the server:

```
reboot
```

----End

1.4.8 Installing Open MPI

Procedure

Step 1 Use PuTTY to log in to the server as the **root** user.

Step 2 Run the following command to install the system dependency package:

```
yum install libxml2* systemd-devel.aarch64 numa* -y
```

Step 3 Run the following command to decompress the Open MPI installation package:

```
tar -zxvf openmpi-4.0.3.tar.gz
```

Step 4 Run the following command to go to source code directory:

```
cd openmpi-4.0.3
```

Step 5 Run the following command to compile and configure Open MPI:

```
./configure --prefix=/path/to/OPENMPI --enable-pretty-print-stacktrace --enable-orterun-prefix-by-default --with-knem=/opt/knem-1.1.3.90mlnx1/ --with-hcoll=/opt/mellanox/hcoll/ --with-cma  --with-ucx  --enable-mpi1-compatibility
```

Step 6 Run the following command to compile and install Open MPI:

```
make -j 16
make  install
```

Step 7 Run the following commands to load the environment variables:

```
export PATH=/path/to/OPENMPI/bin:$PATH
export LD_LIBRARY_PATH=/path/to/OPENMPI/lib:$LD_LIBRARY_PATH
```

----End

1.5 Setting Up the Environment for the Single-Node Scenario

Prerequisites

The required installation packages have been uploaded to the planned installation directories on the server using an SFTP tool.

Procedure

Step 1 Configure the local Yum source. See **1.4.2 Configuring the Local Yum Source**.

Step 2 Install GMP. See **1.4.3 Installing GMP**.

Step 3 Install MPFR. See **1.4.4 Installing MPFR**.

Step 4 Install MPC. See **1.4.5 Installing MPC**.

Step 5 Install GNU. See **1.4.6 Installing GNU**.

----End

2 GNU 9.1 Installation Guide

2.1 Introduction

Overview

GNU is an open source development toolchain, including the GCC compiler, assembler, linker, and other open-source tools. The GNU compiler supports TaiShan servers since version 4.9. In addition, core optimization functions are added to GNU 9 and later versions. Therefore, GNU 9 is more user-friendly and more efficient on TaiShan servers.

Recommended Software Version

GNU 9.1

2.2 Environment Requirements

Hardware Requirements

Table 2-1 lists the hardware requirements.

Table 2-1 Hardware requirements

Item	Description
CPU	Kunpeng 920

Software Requirements

Table 2-2 lists the software requirements.

Table 2-2 Software requirements

Item	Version	Download URL
GMP	6.1.0	http://gcc.gnu.org/pub/gcc/infrastructure/
MPFR	3.1.4	
MPC	1.0.3	
GNU	9.1.0	https://ftp.gnu.org/gnu/gcc/gcc-9.1.0/
OpenMPI	4.0.1	https://www.open-mpi.org/software/ompi/v4.0/

OS Requirements

Table 2-3 lists the OS requirements.

Table 2-3 OS requirements

Item	Version	How to Obtain
CentOS	7.6	https://www.centos.org/download/

2.3 Planning Data

This chapter describes the software installation paths involved in the GNU installation.

Table 2-4 Data plan

No.	Software Installation Path	Description	Remarks
1	/path/to/GMP	Installation path of GMP	The installation paths listed in this table are only examples. Shared paths are recommended. Replace the paths used in commands in this document with the actual paths planned during the installation process.
2	/path/to/MPFR	Installation path of MPFR	
3	/path/to/MPC	Installation path of MPC	

No.	Software Installation Path	Description	Remarks
4	/path/to/GNU	Installation path of GNU	
5	/path/to/OPENMPI	Installation path of OpenMPI	

2.4 Deploying GNU 9.1

2.4.1 Downloading Installation Packages

Procedure

Step 1 Download the following installation packages:

- GMP installation package: **gmp-6.1.0.tar.bz2**
- MPFR installation package: **mpfr-3.1.4.tar.bz2**
- MPC installation package: **mpc-1.0.3.tar.gz**

Download address: **http://gcc.gnu.org/pub/gcc/infrastructure/**

Step 2 Download the GNU installation package **gcc-9.1.0.tar.xz**.

Download address: **https://ftp.gnu.org/gnu/gcc/gcc-9.1.0/**.

Step 3 Download the OpenMPI installation package **openmpi-4.0.1.tar.gz**.

Download address: **https://www.open-mpi.org/software/ompi/v4.0/**

Step 4 Use the SFTP tool.

- Upload the GMP installation package to the */path/to/GMP* directory on the server.
- Upload the MPFR installation package to the */path/to/MPFR* directory on the server.
- Upload the MPC installation package to the */path/to/MPC* directory on the server.
- Upload the GNU installation package to the */path/to/GNU* directory on the server.
- Upload the OpenMPI installation package to the */path/to/OPENMPI* directory on the server.

----End

2.4.2 Compiling and Installing GMP

Procedure

Step 1 Use PuTTY to log in to the server as the **root** user.

Step 2 Run the following commands to decompress the installation package:

tar -vxf gmp-6.1.0.tar.bz2

Step 3 Run the following command to switch to the directory in which the decompressed files are stored:

cd gmp-6.1.0/

Step 4 Run the following command to perform compilation and installation:

./configure --prefix=*/path/to/GMP/gmp*-6.1.0

make

make install

Step 5 Run the following command to load the environment variable:

export LD_LIBRARY_PATH=*/path/to/GMP/gmp*-6.1.0**/lib:$LD_LIBRARY_PATH**

The following figure shows directory contents when the installation is successful.

----End

2.4.3 Compiling and Installing MPFR

Procedure

Step 1 Use PuTTY to log in to the server as the **root** user.

Step 2 Run the following commands to decompress the installation package:

tar -xvf mpfr-3.1.4.tar.bz2

Step 3 Run the following command to switch to the directory in which the decompressed files are stored:

cd mpfr-3.1.4/

Step 4 Run the following command to perform compilation and installation:

./configure --prefix=*/path/to/MPFR*/**mpfr-3.1.4 --with-gmp=***/path/to/GMP*/**gmp-6.1.0**

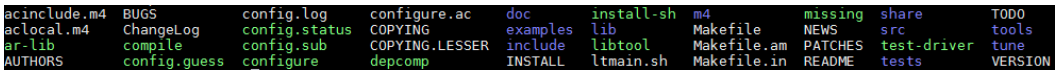
make

make install

Step 5 Run the following command to load environment variables:

export LD_LIBRARY_PATH=*/path/to/MPFR***/mpfr-3.1.4/lib:$LD_LIBRARY_PATH**

The following figure shows directory contents when the installation is successful.



----End

2.4.4 Compiling and Installing MPC

Procedure

Step 1 Use PuTTY to log in to the server as the **root** user.

Step 2 Run the following commands to decompress the installation package:

tar -zvxf mpc-1.0.3.tar.gz

Step 3 Run the following command to switch to the directory in which the decompressed files are stored:

cd mpc-1.0.3/

Step 4 Run the following command to perform compilation and installation:

./configure --prefix=*/path/to/MPC***/mpc-1.0.3 --with-gmp=***/path/to/GMP***/gmp-6.1.0 --with-mpfr=***/path/to/MPFR***/mpfr-3.1.4**

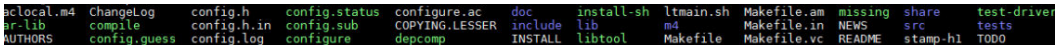
make

make install

Step 5 Run the following command to load environment variables:

export LD_LIBRARY_PATH=*/path/to/MPC***/mpc-1.0.3/lib:$LD_LIBRARY_PATH**

The following figure shows directory contents when the installation is successful.



----End

2.4.5 Compiling and Installing GNU 9.1

Procedure

Step 1 Use PuTTY to log in to the server as the **root** user.

Step 2 Run the following commands to decompress the installation package:

tar -vxf gcc-9.1.0.tar.xz

Step 3 Run the following command to switch to the directory in which the decompressed files are stored:

cd gcc-9.1.0/

Step 4 Run the following command to create the **obj** directory:

mkdir obj

Step 5 Run the following command to go to the **obj** directory:

cd obj

Step 6 Run the following command to perform compilation and installation:

../configure --disable-multilib --enable-languages="c,c++,fortran" --prefix=*/path/to/GNU* **--disable-static --enable-shared --with-gmp=***/path/to/GMP***/gmp-6.1.0 --with-mpfr=***/path/to/MPFR***/mpfr-3.1.4 --with-mpc=***/path/to/MPC***/mpc-1.0.3**

make

make install

Step 7 Run the following command to load environment variables:

export PATH*=/path/to/GNU***/bin:$PATH**

export LD_LIBRARY_PATH*=/path/to/GNU***/lib64:$LD_LIBRARY_PATH**

The following figure shows directory contents when the installation is successful.

```
[root@          gcc-9.1.0]# ls
ABOUT-NLS           config.rpath  COPYING.RUNTIME  intl         libgcc         libphobos       ltmain.sh          Makefile.in     obj
ar-lib              config.sub    depcomp          LAST_UPDATED libgfortran    libquadmath     lt~obsolete.m4     Makefile.tpl    README
ChangeLog           configure     fixincludes      libada       libgo          libsanitizer    lto-plugin         MD5SUMS         symlink-tree
ChangeLog.jit       configure.ac  gcc              libatomic    libgomp        libssp          ltoptions.m4       missing         test-driver
ChangeLog.tree-ssa  contrib       gnattools        libbacktrace libhsail-rt    libstdc++-v3    ltsugar.m4         mkdep           ylwrap
compile             COPYING       gotools          libcc1       libiberty      libtool-ldflags ltversion.m4       mkinstalldirs   zlib
config              COPYING3      include          libcpp       libitm         libtool.m4      MAINTAINERS        move-if-change
config.guess        COPYING3.LIB  INSTALL          libdecnumber libobjc        libvtv          maintainer-scripts multilib.am
config-ml.in        COPYING.LIB   install-sh       libffi       liboffloadmic  ltgcc.m4        Makefile.def       NEWS
[root@          gcc-9.1.0]# cd obj/
[root@          obj]# ls
aarch64-unknown-linux-gnu       lib          prev-aarch64-unknown-linux-gnu  serdep.tmp                      stage1-lto-plugin
bin                             lib64        prev-fixincludes                share                           stage1-zlib
build-aarch64-unknown-linux-gnu libbacktrace prev-gcc                        stage1-aarch64-unknown-linux-gnu stage_current
compare                         libcc1       prev-intl                       stage1-fixincludes              stage_final
config.log                      libcpp       prev-libbacktrace               stage1-gcc                      stage_last
config.status                   libdecnumber prev-libcpp                     stage1-intl                     zlib
fixincludes                     libexec      prev-libdecnumber               stage1-libbacktrace
gcc                             libiberty    prev-libiberty                  stage1-libcpp
include                         lto-plugin   prev-lto-plugin                 stage1-libdecnumber
intl                            Makefile     prev-zlib                       stage1-libiberty
[root@          obj]# cd bin
[root@          bin]# ls
aarch64-unknown-linux-gnu-c++  aarch64-unknown-linux-gnu-gcc-9.1.0  aarch64-unknown-linux-gnu-gcc-ranlib  cpp  gcc-ar      gcov       gfortran
aarch64-unknown-linux-gnu-g++  aarch64-unknown-linux-gnu-gcc-ar     aarch64-unknown-linux-gnu-gfortran    g++  gcc-nm      gcov-dump
aarch64-unknown-linux-gnu-gcc  aarch64-unknown-linux-gnu-gcc-nm     c++                                   gcc  gcc-ranlib  gcov-tool
[root@          bin]# cd ..
[root@          obj]# cd lib
lib/         lib64/       libbacktrace/ libcc1/      libcpp/      libdecnumber/ libexec/     libiberty/
[root@          obj]# cd lib64
[root@          lib64]# ls
libasan.la          libcc1.la         libgfortran.so.5.0.0  libitm.so.1       libssp.la           libstdc++.so               libtsan.so.0
libasan_preinit.o   libcc1.so         libgfortran.spec      libitm.so.1.0.0   libssp_nonshared.a  libstdc++.so.6             libtsan.so.0.0.0
libasan.so          libcc1.so.0       libgomp.la            libitm.spec       libssp_nonshared.la libstdc++.so.6.0.26        libubsan.la
libasan.so.5        libcc1.so.0.0.0   libgomp.so            liblsan.la        libssp.so           libstdc++.so.6.0.26-gdb.py libubsan.so
libasan.so.5.0.0    libgcc_s.so       libgomp.so.1          liblsan_preinit.o libssp.so.0         libsupc++.a                libubsan.so.1
libatomic.la        libgcc_s.so.1     libgomp.so.1.0.0      liblsan.so        libssp.so.0.0.0     libsupc++.la               libubsan.so.1.0.0
libatomic.so        libgfortran.la    libgomp.spec          liblsan.so.0      libstdc++fs.a       libtsan.la
libatomic.so.1      libgfortran.so    libitm.la             liblsan.so.0.0.0  libstdc++fs.la      libtsan_preinit.o
libatomic.so.1.2.0  libgfortran.so.5  libitm.so             libsanitizer.spec  libstdc++.la       libtsan.so
```

----End

2.5 Verifying GNU 9.1

Procedure

Step 1 Use PuTTY to log in to the server as the **root** user.

Step 2 Run the following command to load environment variables:

export PATH=*/path/to/GNU***/bin:$ PATH**

export LD_LIBRARY_PATH=*/path/to/GNU***/lib64:$LD_LIBRARY_PATH**

Step 3 Run the following command to check whether the GCC version is 9.1.0:

gcc -v

```
Using built-in specs.
COLLECT_GCC=gcc
COLLECT_LTO_WRAPPER=/path/to/GNU/bin/../libexec/gcc/aarch64-unknown-linux-gnu/9.1.0/lto-wrapper
Target: aarch64-unknown-linux-gnu
Configured with: ../configure --disable-multilib --enable-languages=c,c++,fortran --prefix=/path/to/GNU --
disable-static --enable-shared --with-gmp=/path/to/GMP --with-mpfr=/path/to/MPFR --with-mpc=/path/to/
MPC
Thread model: posix
gcc version 9.1.0 (GCC)
```

----End

3 HTCondor 8.9.2 Installation Guide

3.1 Introduction

HTCondor Overview

HTCondor is an open-source high-throughput computing software framework for coarse-grained distributed parallelization of computationally intensive tasks. It manages workload on a dedicated cluster of computers or farms out work to idle desktop computers, which is called cycle scavenging. HTCondor can run on Linux, UNIX, Mac OS X, FreeBSD, and Microsoft Windows. It can integrate dedicated resources (rack-mounted clusters) and non-dedicated desktop machines (cycle scavenging) into a computing environment.

Recommended Software Version

HTCondor 8.9.2

3.2 Environment Requirements

Hardware Requirements

Table 3-1 lists the hardware requirements.

Table 3-1 Hardware requirements

Item	Description
CPU	Kunpeng 920

Software Requirements

Table 3-2 lists the software requirements.

Table 3-2 Software requirements

Item	Version	How to Obtain
HTCondor	8.9.2	https://github.com/htcondor/htcondor/releases/tag/V8_9_2
munge	0.5.13	https://github.com/dun/munge/releases/tag/munge-0.5.13
SQLite	3.34.1	https://www.sqlite.org/2021/sqlite-autoconf-3340100.tar.gz

OS Requirements

Table 3-3 lists the OS requirements.

Table 3-3 OS requirements

Item	Version	How to Obtain
CentOS	7.6	https://www.centos.org/download/

3.3 Planning Data

This chapter lists the software installation paths involved in the HTCondor installation.

Table 3-4 Data plan

No.	Software Installation Path	Description	Remarks
1	/path/to/HTCONDOR	Installation path of HTCondor	The installation paths listed in this table are only examples. Shared paths are recommended. Replace the paths used in commands in this

No.	Software Installation Path	Description	Remarks
2	/path/to/MUNGE	Installation path of munge	document with the actual paths planned during the installation process.
3	/path/to/SQLITE	Installation path of SQLite	

3.4 Deploying HTCondor

3.4.1 Obtaining Installation Packages

Procedure

Step 1 Download the HTCondor installation package **htcondor-8_9_2.tar.gz**.

URL: **https://github.com/htcondor/htcondor/archive/V8_9_2.tar.gz**

Step 2 Download the munge installation package **munge-0.5.13.tar.xz**.

URL: **https://github.com/dun/munge/releases/tag/munge-0.5.13**

Step 3 Download the SQLite installation package **sqlite-autoconf-3340100.tar.gz** from the following address:

https://www.sqlite.org/2021/sqlite-autoconf-3340100.tar.gz

Step 4 Use an SFTP tool to upload the installation packages to the server.

- Upload the HTCondor installation package to the **/path/to/HTCONDOR** directory on the server.
- Upload the munge installation package to the **/path/to/MUNGE** directory on the server.
- Upload the SQLite installation package to the */path/to/SQLITE* directory on the server.

----End

3.4.2 Installing Dependencies

3.4.2.1 Installing munge

Step 1 Use PuTTY to log in to the server as the root user.

Step 2 Run the following command to install the system dependency package:

yum install bzip2-devel boost-devel-1.53.0-28.el7.aarch64 libuuid-devel.aarch64 libX11-devel.aarch64 -y

Step 3 Run the following command to build the munge RPM package:

rpmbuild -tb --clean munge-0.5.13.tar.xz

Step 4 Check whether the RPM package is successfully created.

ls /root/rpmbuild/RPMS/aarch64/ | grep munge

munge-0.5.13-1.el7.aarch64.rpm

munge-debuginfo-0.5.13-1.el7.aarch64.rpm

munge-devel-0.5.13-1.el7.aarch64.rpm

munge-libs-0.5.13-1.el7.aarch64.rpm

Step 5 Create a **mungerpm** directory in **/path/to/MUNGE** and copy the munge RPM package from **/root/rpmbuild/RPMS/aarch64/** to **/path/to/MUNGE/mungerpm**.

mkdir -p mungerpm

cp /root/rpmbuild/RPMS/aarch64/munge* */path/to/MUNGE/***mungerpm -f**

Step 6 Run the following command to install the munge RPM package:

cd */path/to/MUNGE/***mungerpm**

yum install -y munge-*

----End

3.4.2.2 Installing SQLite

Step 1 Use PuTTY to log in to the server as the root user.

Step 2 Run the following command to decompress the package:

tar -xvf sqlite-autoconf-3340100.tar.gz

Step 3 Run the following command to switch to the directory that contains decompressed files:

cd */path/to/SQLITE/***sqlite-autoconf-3340100**

Step 4 Run the following commands to configure and install the software:

./configure --prefix=/path/to/SQLITE

make -j 64

make install

----End

3.4.3 Compiling HTCondor

Step 1 Use PuTTY to log in to the server as the root user.

Step 2 Run the following commands to decompress the installation package:

cd */path/to/HTCONDOR*

tar -xvf htcondor-8_9_2.tar.gz

Step 3 Switch to the **condor-8.9.2** directory:

cd htcondor-8_9_2

ls

```
build  builder.sh  build-on-linux.sh  CITATION.cff  CMakeLists.txt  configure_redhat  configure_uw  doc
docs  externals  LICENSE-2.0.txt  msconfig  nmi_tools  NOTICE.txt  src  view
```

Step 4 Run the following command to edit the **config** file of HTCondor:

cp configure_redhat buildarm.sh

1. Open the **config** file of HTCondor.

   vi buildarm.sh

2. Press **i** to enter the insert mode and modify the file as follows:

```
#!/bin/sh

echo "-------------------------------------------------------------------"
echo "* NOTE: Attempting to configure a Red Hat-esk build"
echo "* which builds against system libs and selectively "
echo "* enables and disables portions of condor"
echo "* If you are unsure, you should run \"cmake .\""
echo "*"
echo "* add -D_DEBUG:BOOL=FALSE to get non-optimized code for debugging"
echo "* Another option would be to run ccmake or cmake-gui"
echo "* and select the options you care to build with"
echo "-------------------------------------------------------------------"
cmake  \
  -D_DEBUG:BOOL=TRUE \
  -DWITH_CREAM:BOOL=FALSE \
  -DNO_PHONE_HOME:BOOL=TRUE \
  -DHAVE_BACKFILL:BOOL=FALSE \
  -DHAVE_BOINC:BOOL=FALSE \
  -DHAVE_KBDD:BOOL=TRUE \
  -DHAVE_HIBERNATION:BOOL=TRUE \
  -DWANT_CONTRIB:BOOL=ON \
  -DWANT_MAN_PAGES:BOOL=TRUE \
  -DWANT_FULL_DEPLOYMENT:BOOL=FALSE \
  -DWANT_GLEXEC:BOOL=FALSE \
  -D_VERBOSE:BOOL=TRUE \
  -DBUILDID:STRING=RH_development \
  -DWITH_GLOBUS:BOOL=FALSE \
  -DWITH_VOMS:BOOL=FALSE \
  -DSQLITE3_LIB:FILEPATH=/path/to/SQLITE/lib/libsqlite3.so \
  -DHAVE_SQLITE3_H:FILEPATH=/path/to/SQLITE/include \
  -DCMAKE_INSTALL_PREFIX:PATH=${PWD}/release_dir "$@"
```

3. Press **Esc**, type **:wq!**, and press **Enter** to save the file and exit.

Step 5 Run the following command to perform the configuration:

./buildarm.sh

```
...
-- Configuring done
-- Generating done
-- Build files have been written to: /home/htcondor/condor-8.9.2
```

Step 6 Run the following commands to perform the compilation and installation:

make -j 64

make install

----End

3.4.4 Configuring HTCondor

> **NOTE**
>
> This section describes how to configure HTCondor on a node, that is, perform configuration, and submit and execute tasks on a node.

Procedure

Step 1 Switch to the **release_dir** directory.

```
cd /path/to/HTCONDOR/htcondor-8_9_2/release_dir
```

Step 2 Create a **condor.sh** file.

1. Create **condor.sh**.
   ```
   vi condor.sh
   ```
2. Press **i** to enter the insert mode and add the following content:
   ```
   export CONDOR_CONFIG=/path/to/HTCONDOR/htcondor-8_9_2/release_dir/etc/condor_config
   export PATH=/path/to/HTCONDOR/htcondor-8_9_2/release_dir/bin:/path/to/HTCONDOR/condor-8.9.2/release_dir/sbin:$PATH
   ```
3. Press **Esc**, type **:wq!**, and press **Enter** to save the file and exit.
4. Run the following command to make the file take effect:
   ```
   source condor.sh
   ```

Step 3 Switch to the **release_dir/etc** directory.

```
cd /path/to/HTCONDOR/htcondor-8_9_2/release_dir/etc
```

Step 4 Create and edit the **condor_config** file.

1. Create a **condor_config** configuration file.
   ```
   vi condor_config
   ```
2. Press **i** to enter the insert mode and modify the file as follows:
   ```
   CONDOR_HOST             = 192.168.47.111
   RELEASE_DIR             = /path/to/HTCONDOR/htcondor-8_9_2/release_dir
   LOCAL_DIR               = /data/
   LOCAL_CONFIG_DIR        = $(LOCAL_DIR)/config
   LOCAL_CONFIG_FILE       = $(LOCAL_DIR)/condor_config.local
   CONDOR_ADMIN            = root@192.168.47.111
   MAIL                    = /usr/bin/mail
   ALLOW_ADMINISTRATOR     = $(CONDOR_HOST)
   ALLOW_NEGOTITATOR       = $(CONDOR_HOST)
   LOCK                    = $(LOG)
   CONDOR_IDS              = 2001.2001

   use SECURITY : HOST_BASED

   LOG                     = $(LOCAL_DIR)/log
   SPOOL                   = $(LOCAL_DIR)/spool
   BIN                     = $(RELEASE_DIR)/bin
   LIB                     = $(RELEASE_DIR)/lib
   SBIN                    = $(RELEASE_DIR)/sbin
   LIBEXEC                 = $(RELEASE_DIR)/libexec
   HISTORY                 = $(RELEASE_DIR)/history


   MASTER_LOG              = $(LOG)/MasterLog
   SCHEDD_LOG              = $(LOG)/SchedLog
   SHADOW_LOG              = $(LOG)/ShadowLog

   SHADOW_LOCK             = $(LOCK)/ShadowLock

   DAEMON_LIST = COLLECTOR MASTER NEGOTIATOR SCHEDD STARTD
   ```



```
CONDOR_HOST = $(CONDOR_HOST)
USE_CLONE_TO_CREATE_PROCESSES = False
```

3. Press **Esc**, type **:wq!**, and press **Enter** to save the file and exit.

Step 5 Create the **condor** user and **condor** user group.

```
groupadd -g 2001 condor
useradd -u 2001 -g 2001 condor
```

Step 6 Create the directories and files of HTCondor.

```
mkdir -p /data
cd /data
mkdir -p config  examples  execute  log  spool
touch condor_config.local
touch log/MasterLog log/SchedLog log/ShadowLog log/ShadowLock
chown -R condor.condor *
```

Step 7 Configure the **init.d** service.

```
cp /path/to/HTCONDOR/htcondor-8_9_2/release_dir/etc/init.d/condor /etc/init.d/ -f
```

1. Open the **/etc/init.d/condor** file.
   ```
   vi /etc/init.d/condor
   ```
2. Press **i** to enter the insert mode and modify the file as follows:
   ```
   ...
   # Path to your primary condor configuration file.
   CONDOR_CONFIG="/path/to/HTCONDOR/htcondor-8_9_2/release_dir/etc/condor_config"

   # Path to condor_config_val
   CONDOR_CONFIG_VAL="/path/to/HTCONDOR/htcondor-8_9_2/release_dir/bin/condor_config_val"
   ...
   ```
3. Press **Esc**, type **:wq!**, and press **Enter** to save the file and exit.

----End

3.4.5 Starting HTCondor

Procedure

Step 1 Use PuTTY to log in to the server as the **root** user.

Step 2 Run the following command to start HTCondor:

```
/etc/init.d/condor start
```

----End

3.5 Configuring HTCondor Clusters

3.5.1 Before You Start

Procedure

Step 1 Check that communication between nodes is normal and the SSH trust relationship is established (broadcast communication is recommended in the same network segment).

Step 2 Check that HTCondor has been installed on each node. For details, see **3.4 Deploying HTCondor**.

Step 3 Determine the role of each node (specified by **DAEMON_LIST** in the **condor_config** file).

Step 4 Check that the settings in the **condor_config** file are correct.

----End

3.5.2 Configuring the condor_config File

Procedure

Step 1 Configure the **condor_config** file on the Manager node as follows:

```
CONDOR_HOST            = 192.168.47.111
RELEASE_DIR            = /path/to/HTCONDOR/htcondor-8_9_2/release_dir
LOCAL_DIR              = /data
LOCAL_CONFIG_DIR       = $(LOCAL_DIR)/config
LOCAL_CONFIG_FILE      = $(LOCAL_DIR)/condor_config.local
CONDOR_ADMIN           = root@192.168.47.111
MAIL                   = /usr/bin/mail
CONDOR_IDS             = 2001.2001

ALLOW_WRITE            = 192.168.47.42
ALLOW_READ             = 192.168.47.42
ALLOW_ADVERTISE_SCHEDD = 192.168.47.*
ALLOW_ADVERTISE_MASTER = 192.168.47.42
ALLOW_ADVERTISE_STARTD = 192.168.47.42

use SECURITY : HOST_BASED

LOG                    = $(LOCAL_DIR)/log
ALL_DEBUG              = D_ALL
SPOOL                  = $(LOCAL_DIR)/spool
LOCK                   = $(LOG)
BIN                    = $(RELEASE_DIR)/bin
LIB                    = $(RELEASE_DIR)/lib
SBIN                   = $(RELEASE_DIR)/sbin
LIBEXEC                = $(RELEASE_DIR)/libexec
HISTORY                = $(RELEASE_DIR)/history

MASTER_LOG             = $(LOG)/MasterLog
SCHEDD_LOG             = $(LOG)/SchedLog
SHADOW_LOG             = $(LOG)/ShadowLog

SHADOW_LOCK            = $(LOCK)/ShadowLock

DAEMON_LIST = COLLECTOR MASTER NEGOTIATOR SCHEDD STARTD
USE_CLONE_TO_CREATE_PROCESSES = False
```

Step 2 Configure the **condor_config** file on the submit or execute node as follows:

```
CONDOR_HOST            = 192.168.47.111
RELEASE_DIR            = /path/to/HTCONDOR/htcondor-8_9_2/release_dir
LOCAL_DIR              = /data
LOCAL_CONFIG_DIR       = $(LOCAL_DIR)/config
LOCAL_CONFIG_FILE      = $(LOCAL_DIR)/condor_config.local
CONDOR_ADMIN           = root@192.168.47.111
MAIL                   = /usr/bin/mail
CONDOR_IDS             = 2008.2008

ALLOW_WRITE            = 192.168.47.*
ALLOW_READ             = 192.168.47.*

use SECURITY : HOST_BASED
LOG                    = $(LOCAL_DIR)/log
SPOOL                  = $(LOCAL_DIR)/spool
LOCK                   = $(LOCAL_DIR)/lock
BIN                    = $(RELEASE_DIR)/bin
```



```
LIB                 = $(RELEASE_DIR)/lib
SBIN                 = $(RELEASE_DIR)/sbin
LIBEXEC               = $(RELEASE_DIR)/libexec
HISTORY               = $(RELEASE_DIR)/history

MASTER_LOG            = $(LOG)/MasterLog
SCHEDD_LOG            = $(LOG)/SchedLog
SHADOW_LOG             = $(LOG)/ShadowLog
SHADOW_LOCK            = $(LOCK)/ShadowLock

DAEMON_LIST = MASTER STARTD SCHEDD
USE_CLONE_TO_CREATE_PROCESSES = False
```

Step 3 Run the following command to insert the **condor_config** configuration again for all the manager and worker nodes:

```
condor_reconfig
```

Step 4 Check the condor queue status. It should be the total number of cores on the node.

```
condor_status
```

Example:

```
                     Machines Owner Claimed Unclaimed Matched Preempting  Drain

aarch64/LINUX             224     0       0       224       0          0      0

       Total              224     0       0       224       0          0      0
```

----End

3.6 Verifying HTCondor

3.6.1 Common Commands

Table 3-5 Common commands

Command	Description
condor_q	Queries the task queue.
condor_status	Queries resource status.
condor_history	Queries historical tasks.
condor_submit	Submits a job.
congdor_rm	Deletes a job.

3.6.2 Submitting Script Jobs

Procedure

Step 1 Use PuTTY to log in to the server as the root user.

Step 2 Switch to the **condor** user.

```
su - condor
```

Step 3 Switch to the **/data/examples** directory.

```
cd /data/examples
```

Step 4 Edit the **test.sh** file.

1. Open **test.sh**.
   ```
   vi test.sh
   ```
2. Press **i** to enter the insert mode and add the following content:
   ```
   #!/bin/bash
   for x in {0..10}
   do
   echo "This is a test"
   sleep 5s
   done
   ```
3. Press **Esc**, type **:wq!**, and press **Enter** to save the file and exit.

Step 5 Edit the **test.sub** file.

1. Open **test.sub**.
   ```
   vi test.sub
   ```
2. Press **i** to enter the insert mode and add the following content:
   ```
   universe        = vanilla
   executable      = ./test.sh
   output          = test.o
   error           = test.e
   log             = test.log
   should_transfer_files   = YES
   when_to_transfer_output = ON_EXIT
   Notification            = never
   queue
   ```
3. Press **Esc**, type **:wq!**, and press **Enter** to save the file and exit.

Step 6 Run the following command to submit a job:

```
condor_submit test.sub
```

Step 7 Run the following command to submit a multi-queue job and verify the cluster:

```
cat test.sub
universe        = vanilla
executable      = ./test.sh
output          = test.o
error           = test.e
log             = test.log
should_transfer_files   = YES
when_to_transfer_output = ON_EXIT
Notification            = never
queue 150
```

Step 8 Run the following command to verify the configuration:

```
condor_submit test.sub
```



```
slot74@TsXA320V2 LINUX      aarch64 Claimed   Busy      0.000 2710  0+00:00:19
slot75@TsXA320V2 LINUX      aarch64 Claimed   Busy      0.000 2710  0+00:00:19
slot76@TsXA320V2 LINUX      aarch64 Claimed   Busy      0.000 2710  0+00:00:20
slot77@TsXA320V2 LINUX      aarch64 Claimed   Busy      0.000 2710  0+00:00:19
slot78@TsXA320V2 LINUX      aarch64 Claimed   Busy      0.000 2710  0+00:00:19
slot79@TsXA320V2 LINUX      aarch64 Claimed   Busy      0.000 2710  0+00:00:19
slot80@TsXA320V2 LINUX      aarch64 Claimed   Busy      0.010 2710  0+00:00:09
slot81@TsXA320V2 LINUX      aarch64 Claimed   Busy      0.010 2710  0+00:00:09
slot82@TsXA320V2 LINUX      aarch64 Claimed   Busy      0.020 2710  0+00:00:09
slot83@TsXA320V2 LINUX      aarch64 Unclaimed Idle      0.000 2710  0+00:00:05
slot84@TsXA320V2 LINUX      aarch64 Unclaimed Idle      0.000 2710  0+00:00:05
slot85@TsXA320V2 LINUX      aarch64 Unclaimed Idle      0.000 2710  0+00:00:04
slot86@TsXA320V2 LINUX      aarch64 Unclaimed Idle      0.000 2710  0+00:00:04
slot87@TsXA320V2 LINUX      aarch64 Unclaimed Idle      0.000 2710  0+00:00:04
slot88@TsXA320V2 LINUX      aarch64 Unclaimed Idle      0.000 2710  0+00:00:03
slot89@TsXA320V2 LINUX      aarch64 Unclaimed Idle      0.000 2710  0+00:00:03
slot90@TsXA320V2 LINUX      aarch64 Unclaimed Idle      0.000 2710  0+00:00:03
slot91@TsXA320V2 LINUX      aarch64 Unclaimed Idle      0.000 2710  0+00:00:02
slot92@TsXA320V2 LINUX      aarch64 Unclaimed Idle      0.000 2710  0+00:00:02
slot93@TsXA320V2 LINUX      aarch64 Unclaimed Idle      0.000 2710  0+00:00:01
slot94@TsXA320V2 LINUX      aarch64 Unclaimed Idle      0.000 2710  0+00:00:01
slot95@TsXA320V2 LINUX      aarch64 Unclaimed Idle      0.000 2710  0+00:00:01
slot96@TsXA320V2 LINUX      aarch64 Unclaimed Idle      0.000 2710  0+00:00:00
slot1@Ts2280V2   LINUX      aarch64 Claimed   Busy      0.010 4078  0+00:00:02
slot2@Ts2280V2   LINUX      aarch64 Claimed   Busy      0.000 4078  0+00:00:02
slot3@Ts2280V2   LINUX      aarch64 Claimed   Busy      0.000 4078  0+00:00:02
slot4@Ts2280V2   LINUX      aarch64 Claimed   Busy      0.000 4078  0+00:00:03
slot5@Ts2280V2   LINUX      aarch64 Claimed   Busy      0.000 4078  0+00:00:03
slot6@Ts2280V2   LINUX      aarch64 Claimed   Busy      0.000 4078  0+00:00:02
slot7@Ts2280V2   LINUX      aarch64 Claimed   Busy      0.000 4078  0+00:00:04
slot8@Ts2280V2   LINUX      aarch64 Claimed   Busy      0.000 4078  0+00:00:04
```

```
               Machines Owner Claimed Unclaimed Matched Preempting  Drain

 aarch64/LINUX      224     0     136        88       0          0      0

        Total       224     0     136        88       0          0      0
```

----End

3.7 Troubleshooting

3.7.1 Failed to Start the HTCondor Service on the Kunpeng 920

Symptom

HTCondor service failed to start on the Kunpeng 920.

Procedure

Step 1 Modify the **daemon_core.cpp** file.

1. Open **daemon_core.cpp**.
   ```
   vi condor-8.9.2/src/condor_daemon_core.V6/daemon_core.cpp
   ```
2. Press **i** to enter the insert mode and modify lines 5856 to 5880 as follows:
   ```
   if( daemonCore->UseCloneToCreateProcesses() ) {
   dprintf(D_FULLDEBUG,"Create_Process: using fast clone() "
   "to create child process.\n");

   // The stack size must be big enough for everything that
   // happens in CreateProcessForkit::clone_fn().  In some
   // environments, some extra steps may need to be taken to
   // make a stack on the heap (to mark it as executable), so
   // we just do it using the parent's stack space and we use
   // CLONE_VFORK to ensure the child is done with the stack
   // before the parent throws it away.
   //const int stack_size = 16384;
   const int stack_size = 64*1024*2;
   ```



```
//const int stack_size = 64*1024*16;
char child_stack[stack_size]  ;

// Beginning of stack is at end on all processors that run
// Linux, except for HP PA.  Here we just detect at run-time
// which way it goes.
char *child_stack_ptr = child_stack;
if( stack_direction() == STACK_GROWS_DOWN ) {
child_stack_ptr += stack_size;
}
child_stack_ptr = (char *)((std::uintptr_t)child_stack_ptr & ~(std::uintptr_t)0<<4);
```

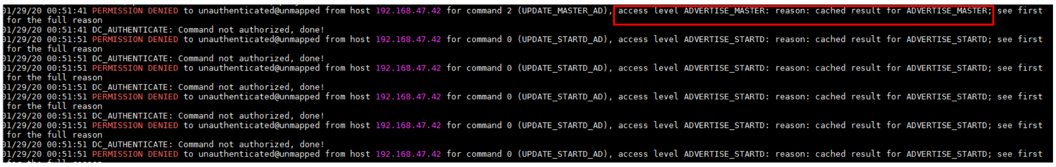
3. Press **Esc**, type **:wq!**, and press **Enter** to save the file and exit.

----End

3.7.2 Connection Rejected Due to Permission

Symptom

The access request is rejected because the permission is denied.



Possible Causes

The ADVERTISE_MASTER daemon does not have the permission. You need to configure "ALLOW_ADVERTISE_MASTER = 192.168.47.42" in the **condor_config** file. This parameter supports subnet matching, for example, you can enter **192.168.47.***.

Procedure

- Generally, only the information of the D_ALWAYS level is recorded in the log file. To enable logs of higher levels to be recorded, add the following information to the **condor_config** file:
  ```
  ALL_DEBUG   =  D_ALL
  ```
- In condor-8.9.* and later versions, it is not enough to configure only **ALLOW_WRITE** and **ALLOW_READ** in the **condor_config** file. You need to add "**use SECURITY: HOST_BASED**".
- If the permission is incorrect, configure the **condor_config** parameter based on the complete log information. If the **ADVERTISE_ SCHEDD** permission is missing, add the following information to the **condor_config** file:
  ```
  ALLOW_ADVERTISE_ SCHEDD = 192.168.47.*
  ```
- A space is required before and after the equal sign (=) in the **condor_config** parameter. Otherwise, the parameter does not take effect.

  Example:
  ```
  ALLOW_WRITE  =  192.168.47.42, 192.168.47.111
  ```

3.8 More Information

1. Installation guide at the official HTCondor website:

   http://research.cs.wisc.edu/htcondor/

2. Getting started: Creating a multiple node Condor pool:

   https://spinningmatt.wordpress.com/2011/06/12/getting-started-creating-a-multiple-node-condor-pool/

3. Need host-based security at least for HTCondor 8.9+:

   https://github.com/opensciencegrid/condor-cron/pull/10/files#diff-9fab5b60a4551556ee9b100bb56030ee

4. Condor problem location mailing list archives:

   https://www-auth.cs.wisc.edu/lists/htcondor-users/2018-October/threads.shtml

4 Lustre 2.12.2 Installation Guide

4.1 Introduction

Lustre Overview

Lustre is a parallel file system, which is usually used in large computer clusters and supercomputers. The name "Lustre" is a portmanteau word derived from Linux and cluster. As early as 1999, the cluster file system company created by Pete Brahms started to develop Lustre and released the Lustre 1.0 in 2003. The Lustre software is available under the GNU GPLv2 license.

In the HPC application, the storage solution is one of the key solutions, and the Lustre parallel file system is a common solution in the HPC storage solution. With the launch of the Kunpeng 920 platform, the availability and performance of Lustre clients need to be verified as soon as possible. This document describes how to recompile the client source code on the Kunpeng 920 platform.

Recommended Software Version

Lustre 2.12.2

4.2 Environment Requirements

Hardware Requirements

Table 4-1 lists the hardware requirements.

Table 4-1 Hardware requirements

Item	Description
CPU	Kunpeng 920

Software Requirements

Table 4-2 lists the software requirements.

Table 4-2 Software requirements

Item	Version	How to Obtain
Open-source Lustre client source code package	2.12.2	**https://downloads.whamcloud.com/public/lustre/lustre-2.12.2/el7.6.1810/client/SRPMS/**

OS Requirements

Table 4-3 lists the OS requirements.

Table 4-3 OS requirements

Item	Version	How to Obtain
CentOS	7.6	**https://www.centos.org/download/**

4.3 Deploying Lustre

4.3.1 Compiling the Client

Procedure

Step 1 Use PuTTY to log in to the server as the root user.

Step 2 Install the source code package.

cd /tmp

rpm -ivh lustre-2.12.2-1.src.rpm lustre-client-dkms-2.12.2-1.el7.src.rpm

Step 3 Go to the installation directory.

cd /root/rpmbuild/SOURCES

Step 4 Decompress the source code package.

tar -xvf lustre-2.12.2.tar.gz

Step 5 Go to the installation directory.

cd lustre-2.12.2

Step 6 Perform the compilation and installation.

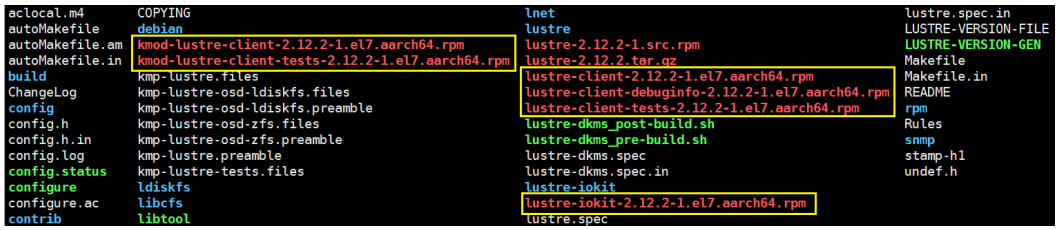
./configure --with-o2ib=/usr/src/ofa_kernel/default/

make rpms

Step 7 After the compilation is successful, run the following command to query the RPM packages generated in the directory:

ls

Example:



----End

4.3.2 Installing the Client

Procedure

Step 1 Use PuTTY to log in to the server as the root user.

Step 2 Go to the installation directory.

cd /root/rpmbuild/SOURCES/lustre-2.12.2

Step 3 Install the RPM packages of the client.

rpm -ivh --nodeps kmod-lustre-client*.rpm lustre-client*.rpm lustre-iokit*.rpm

----End

4.4 Configuring Lustre

4.4.1 Configuring the LNet for Clients

Procedure

Step 1 Use PuTTY to log in to the server as the root user.

Step 2 Configure the Lustre Networking (LNet) for clients.

lustre_rmmod

modprobe lnet

lctl network down

Step 3 Modify the configuration file.

1. Open the **iml_lnet_module_parameters.conf** file.

   vi /etc/modprobe.d/iml_lnet_module_parameters.conf

2. Press **i** to enter the insert mode and add the following content:

   ```
   options lnet networks=o2ib0(ib0)
   ```

3. Press **Esc**, type **:wq!**, and press **Enter** to save the file and exit.

Step 4 Configure the LNet.

modprobe lustre

lctl network up

Step 5 Check the LNet configuration.

lctl list_nids

```
10.10.10.101@o2ib
```

Step 6 Use the LNet ping tool to check the communication between the servers and clients.

lctl ping *10.10.10.105@o2ib0*

```
12345-0@lo
12345-10.10.10.105@o2ib
```

----End

4.4.2 Mounting the File System

Procedure

Step 1 Use PuTTY to log in to the server as the **root** user.

Step 2 Create a directory and mount the file system.

mkdir /mnt/lustre

mount -t lustre 10.10.10.105@o2ib0 :/lustre /mnt/lustre

----End

4.4.3 Verifying Lustre Functions

Procedure

Step 1 Use PuTTY to log in to the server as the **root** user.

Step 2 Run the following command to verify the basic write operation:

echo "This is the basic write test for lustre" >> /mnt/lustre/test.log

Step 3 Run the following command to verify the basic read operation:

cat /mnt/lustre/test.log

```
This is the basic write test for lustre
```

----End

5 OpenHPC 1.3.8 Installation Guide

5.1 Introduction

OpenHPC Overview

OpenHPC is a community-driven Free and Open Source Software (FOSS) tool for Linux based on HPC. OpenHPC has no special requirements for hardware.

OpenHPC provides an integrated and tested set of software components and a supported standard Linux distribution, which can be used to implement a full-featured computing cluster. The components span the entire HPC software ecosystem, including provisioning and system administration tools, resource management, I/O services, development tools, numerical libraries, and performance analysis tools.

For more details, visit the official OpenHPC website **https://openhpc.community**. The source code is hosted on GitHub at **https://github.com/openhpc/ohpc**.

Recommended Software Version

OpenHPC 1.3.8

5.2 Environment Requirements

Hardware Requirements

Table 5-1 lists the hardware requirements.

Table 5-1 Hardware requirements

Item	Description
CPU	Kunpeng 920

Software requirements

Table 5-2 lists the software requirements.

Table 5-2 Software requirements

Item	Version	How to Obtain
OpenHPC suite	1.3.8	**http://build.openhpc.community/dist/1.3.8/OpenHPC-1.3.8.CentOS_7.aarch64.tar**

OS Requirements

Table 5-3 lists the OS requirements.

Table 5-3 OS requirements

Item	Version	How to Obtain
CentOS	7.6	**https://www.centos.org/download/**

Basic Software Stack

Category	Supported Software	Version	How to Obtain
Compiler	GNU	4.9	Included in CentOS
	GNU	5	**https://mirrors.ustc.edu.cn/gnu/**
			Included in OpenHPC 1.3.8
	GNU	6	**https://mirrors.ustc.edu.cn/gnu/**
	GNU	7	**https://mirrors.ustc.edu.cn/gnu/**
			Included in OpenHPC 1.3.8
	GNU	8	**https://mirrors.ustc.edu.cn/gnu/**
			Included in OpenHPC 1.3.8
	GNU	9	**https://mirrors.ustc.edu.cn/gnu/**
	LLVM	4	Included in OpenHPC 1.3.8

Category	Supported Software	Version	How to Obtain
	LLVM	5	Included in OpenHPC 1.3.8
	R	3.3	Included in OpenHPC 1.3.8
MPI	HPC-X	2.4	**http://www.mellanox.com/downloads/hpc/hpc-x/v2.4/hpcx-v2.4.0-gcc-MLNX_OFED_LINUX-4.6-1.0.1.1-redhat7.6-aarch64.tbz**
	Open MPI	3.1	**https://www.open-mpi.org/software/ompi/v3.1**
			Included in OpenHPC 1.3.8
		4.0.1	**https://www.open-mpi.org/software/ompi/v4.0**
	MPICH	3.3	Included in OpenHPC 1.3.8
	MVAPICH	2.3	Included in CentOS
Math & IO libraries	OpenBLAS	0.2.19	Included in OpenHPC 1.3.8
	ScaLAPACK	2	Included in OpenHPC 1.3.8
	FFTW	3.3	Included in OpenHPC 1.3.8
	SuperLU Dist	4.2	Included in OpenHPC 1.3.8
	MUMPS	5	Included in OpenHPC 1.3.8
	HDF5	1.8.17	Included in OpenHPC 1.3.8
	NetCDF	4.4	Included in OpenHPC 1.3.8
	PnetCDF	1.8	Included in OpenHPC 1.3.8
		1.9	
		1.11	
	PETSc	3.7	Included in OpenHPC 1.3.8
	Boost	1.61	Included in OpenHPC 1.3.8
		1.63	
	GSL	2.4	Included in OpenHPC 1.3.8
		2.5	
	hypre	2.13	Included in OpenHPC 1.3.8
		2.14	
		2.15	

Category	Supported Software	Version	How to Obtain
Scheduling software	Slurm	18.08.7	Included in OpenHPC 1.3.8
	Open PBS Pro	19.1.1	Included in OpenHPC 1.3.8
Management software	ClusterTech CEHSS	5.3	**https://www.clustertech.com/hk/welcome** high-performance computing
	Warewulf	1.3.8	Included in OpenHPC 1.3.8
Storage software	Lustre Client	2.1.2	**https://downloads.whamcloud.com/public/lustre/lustre-2.12.2/el7.6.1810/client/SRPMS/**
	BeeGFS	7.1.3	**https://git.beegfs.io/pub/v7/tree/7.1.3**
	NFS	N/A	Included in CentOS
Benchmark	HPL	2.3	**https://www.netlib.org/benchmark/hpl/hpl-2.3.tar.gz**
	HPCG	N/A	**https://github.com/hpcg-benchmark/hpcg**
	Stream	N/A	**https://www.cs.virginia.edu/stream/FTP/Code/**
	IOR	3.1	**https://github.com/hpc/ior**
	MDTest	3.1	**https://github.com/hpc/ior**
	IMB	4.1	Included in OpenHPC 1.3.8
	OSU MPI Benchmark	5.6.1	Included in OpenHPC 1.3.8
Monitor	Ganglia	3.7.2	Included in OpenHPC 1.3.8
	Nagios	2.1.1	Included in OpenHPC 1.3.8

5.3 Planning Data

This chapter lists the software installation paths involved in the OpenHPC installation.

Table 5-4 Data plan

No.	Software Installation Path	Description	Remarks
1	/path/to/OPENHPC	Installation path of OpenHPC.	The installation path listed in this table is only examples. Shared paths are recommended. All the paths used in the commands in this document are examples only. Use the actual paths planned during the installation process.

5.4 Deploying OpenHPC

5.4.1 Obtaining the Software Package

Procedure

Step 1 Download the OpenHPC installation package **OpenHPC-1.3.8.CentOS_7.aarch64.tar**.

URL: **http://build.openhpc.community/dist/1.3.8/OpenHPC-1.3.8.CentOS_7.aarch64.tar**

Step 2 Use an SFTP tool to upload the OpenHPC installation package to the */path/to/OPENHPC* directory on the server.

----End

5.4.2 Deploying OpenHPC on a Singe Node

Procedure

Step 1 Use PuTTY to log in to the server as the **root** user.

Step 2 Run the following commands to decompress the installation package:

cd */path/to/OPENHPC*

tar -xvf OpenHPC-1.3.8.CentOS_7.aarch64.tar

Step 3 Display the files in the installation package.

ls

```
CentOS_7  make_repo.sh  OpenHPC-1.3.8.CentOS_7.aarch64.tar  OpenHPC.local.repo  READM
```

Step 4 Run the **make_repo.sh** file.

./make_repo.sh

The following is an example of the command output.


```
Creating OpenHPC.local.repo file in /etc/yum.repos.d
Local repodata stored in /opt/OpenHPC
```

----End

5.4.3 Deploying OpenHPC in a Cluster

Procedure

Step 1 Upload the original OpenHPC installation package to the */path/to/OPENHPC* directory on all nodes.

NOTE

Do not configure NFS sharing for the */path/to/OPENHPC* directory.

Step 2 Batch decompress the packages.

clush -a "tar -xvf */path/to/OPENHPC*/**OpenHPC-1.3.8.CentOS_7.aarch64.tar"**

Step 3 Run the **make_repo.sh** file on all the nodes.

clush -a "cd /*path/to/OPENHPC*/OpenHPC-1.3.8.CentOS_7.aarch64;./make_repo.sh"

----End

5.4.4 Loading the OpenHPC Basic Environment

Procedure

Step 1 Use PuTTY to log in to the server as the **root** user.

Step 2 Run the following command to install **lmod-ohpc**:

yum install lmod-ohpc

Step 3 Run the following command for the installation to take effect:

source /etc/profile.d/lmod.sh

Step 4 Modify the **/root/.bashrc** file.

1. Open **/root/.bashrc**.

   vi /root/.bashrc

2. Press **i** to enter the insert mode and modify the file as follows:

   ```
   module use /opt/ohpc/pub/modulefiles/
   module use /opt/ohpc/pub/moduledeps/gnu8/
   module use /opt/ohpc/pub/moduledeps/gnu8-openmpi3/
   ```

3. Press **Esc**, type **:wq!**, and press **Enter** to save the file and exit.

Step 5 Run the following command to make the environment variables take effect:

source /root/.bashrc

----End

5.5 Installing Components

5.5.1 Installing the Job Scheduling System

5.5.1.1 Installing Slurm

Procedure

Step 1 Use PuTTY to log in to the server as the **root** user.

Step 2 Run the following commands to install the Slurm server:

yum install -y slurm-ohpc slurm-slurmctld-ohpc slurm-slurmdbd-ohpc

Step 3 Start the Slurm service.

1. Open the **/etc/slrum.conf** file.

   vi /etc/slrum.conf

2. Press **i** to enter the insert mode and add the following content:

   ```
   systemctl start slurmctld
   systemctl enable slurmctld
   ```

3. Press **Esc**, type **:wq!**, and press **Enter** to save the file and exit.

Step 4 Install the Slurm client.

yum install -y slurm-slurmd-ohpc

Step 5 Start the Slurm service.

1. Open the **/etc/slrum.conf** file.

   vi /etc/slurm.conf

2. Press **i** to enter the insert mode and add the following content:

   ```
   systemctl enable slurmd
   systemctl start slurmd
   ```

3. Press **Esc**, type **:wq!**, and press **Enter** to save the file and exit.

Step 6 Configure Slurm. For details, see "Installing Slurm" in the **Slurm 18.08.7 Installation Guide**.

----End

5.5.2 Install the Compiler

5.5.2.1 Install the GNU Compiler

Procedure

Step 1 Use PuTTY to log in to the server as the **root** user.

Step 2 Install the GNU compiler provided by the OpenHPC suite on the management node.

yum install -y gnu8-compilers-ohpc

module add gnu8/8.3.0

----End

5.5.3 Installing Math Libraries

5.5.3.1 Installing NetCDF

Procedure

Step 1 Use PuTTY to log in to the server as the **root** user.

Step 2 Install NetCDF on the management node.

yum install -y netcdf-gnu8-openmpi3-ohpc

module add netcdf/4.6.3

----End

5.5.3.2 Installing NetCDF-Fortran

Procedure

Step 1 Use PuTTY to log in to the server as the **root** user.

Step 2 Install NetCDF-Fortran on the management node.

yum install -y netcdf-fortran-gnu8-openmpi3-ohpc

module add netcdf-fortran/4.4.5

----End

5.5.3.3 Installing PnetCDF

Procedure

Step 1 Use PuTTY to log in to the server as the **root** user.

Step 2 Install PnetCDF on the management node.

yum install -y pnetcdf-gnu8-openmpi3-ohpc

module add pnetcdf/1.11.1

----End

5.5.3.4 Installing OpenBLAS

Procedure

Step 1 Use PuTTY to log in to the server as the **root** user.

Step 2 Install OpenBLAS on the management node.

module add gnu8/8.3.0

yum install -y openblas-gnu8-ohpc.aarch64

module add openblas/0.3.5

----End

5.5.3.5 Installing FFTW

Procedure

Step 1 Use PuTTY to log in to the server as the **root** user.

Step 2 Install FFTW on the management node.

module add gnu8/8.3.0

yum install -y fftw-gnu8-openmpi3-ohpc.aarch64

module add fftw/3.3.8

----End

5.5.3.6 Installing Metis

Procedure

Step 1 Use PuTTY to log in to the server as the **root** user.

Step 2 Install Metis on the management node.

module add gnu8/8.3.0

yum install -y metis-gnu8-ohpc.aarch64

module add metis/5.1.0

----End

5.5.3.7 Installing SuperLU

Procedure

Step 1 Use PuTTY to log in to the server as the **root** user.

Step 2 Install SuperLU on the management node.

module add gnu8/8.3.0

yum install -y superlu-gnu8-ohpc.aarch64

module add superlu/5.2.1

----End

5.5.3.8 Installing SuperLU Dist

Procedure

Step 1 Use PuTTY to log in to the server as the **root** user.

Step 2 Install SuperLU Dist on the management node.

module add gnu8/8.3.0

yum install -y superlu_dist-gnu8-openmpi3-ohpc.aarch64

module add superlu_dist/6.1.1

module add openmpi3/3.1.4

----End

5.5.3.9 Installing ScaLAPACK

Procedure

Step 1 Use PuTTY to log in to the server as the **root** user.

Step 2 Install ScaLAPACK on the management node.

module add gnu8/8.3.0

yum install -y scalapack-gnu8-openmpi3-ohpc.aarch64

module add superlu_dist/6.1.1

module add scalapack/2.0.2

----End

5.5.3.10 Installing Hwloc

Procedure

Step 1 Use PuTTY to log in to the server as the **root** user.

Step 2 Install Hwloc on the management node.

yum install hwloc-ohpc.aarch64 -y

module add hwloc/2.0.3

export PATH=/opt/ohpc/pub/libs/hwloc/2.0.3/include:$PATH

----End

5.5.3.11 Installing Spack

Procedure

Step 1 Use PuTTY to log in to the server as the **root** user.

Step 2 Install Spack on the management node.

yum install -y spack-ohpc.noarch

module add spack/0.12.1

----End

5.5.4 Installing the Container

5.5.4.1 Installing the Singularity Container

Procedure

Step 1 Use PuTTY to log in to the server as the **root** user.

Step 2 Run the following commands to deploy the Singularity container and configure environment variables:

yum install -y singularity-ohpc.aarch64

module add singularity/3.2.1

Step 3 Run the following command to download the image:

singularity pull library://sylabsed/examples/lolcow

----End

5.5.5 Installing Resource Monitoring Software

5.5.5.1 Installing Ganglia

Procedure

Step 1 Use PuTTY to log in to the server as the **root** user.

Step 2 Run the following commands to download the dependencies from the external network:

wget http://www.rpmfind.net/linux/epel/7/aarch64/Packages/l/libconfuse-2.7-7.el7.aarch64.rpm

wget http://www.rpmfind.net/linux/remi/enterprise/7/remi/aarch64/php-ZendFramework-1.12.20-1.el7.remi.noarch.rpm

Step 3 Install the dependencies.

yum install -y libconfuse-2.7-7.el7.aarch64.rpm php-ZendFramework-1.12.20-1.el7.remi.noarch.rpm

Step 4 Install Ganglia on the server.

yum install -y ohpc-ganglia

Step 5 Install Ganglia on the client.

yum install -y ganglia-gmond-ohpc

Step 6 Configure Ganglia on the server.

perl -pi -e "s/<sms>/${sms_name}/" /etc/ganglia/gmond.conf

NOTE

In the **gmond.conf** file, change the value of **HOST** to the host name or IP address of the server.

Step 7 Configure Ganglia on the client.

scp SMS_IP:/etc/ganglia/gmond.conf /etc/ganglia

echo "gridname Mysite" >> /etc/ganglia/gmetad.conf

Step 8 Start the gmond process on the client.

systemctl start gmond

systemctl enable gmond

Step 9 Start the processes on the server.

systemctl start gmond

systemctl start gmetad

systemctl enable gmond

systemctl enable gmetad

systemctl try-restart httpd

systemctl enable httpd

----End

5.5.5.2 Installing Nagios and NRPE

Procedure

Step 1 Use PuTTY to log in to the server as the **root** user.

Step 2 Run the following commands to download the dependencies from the external network:

wget http://www.rpmfind.net/linux/fedora/linux/releases/32/Everything/aarch64/os/Packages/q/qstat-2.15-11.20200131gitd1469ab.fc32.aarch64.rpm

wget http://www.rpmfind.net/linux/epel/7/aarch64/Packages/f/fping-3.10-4.el7.aarch64.rpm

wget http://www.rpmfind.net/linux/epel/7/ppc64/Packages/p/python2-mock-1.0.1-10.el7.noarch.rpm

Step 3 Install the dependencies.

yum install qstat-2.15-7.20150619gita60436.fc29.aarch64.rpm fping-3.10-4.el7.aarch64.rpm python2-mock-1.0.1-10.el7.noarch.rp

Step 4 Install Nagios on the management node.

yum install ohpc-nagios -y

Step 5 Install NRPE on the compute node.

yum install nagios-plugins-all-ohpc nrpe-ohpc -y

Step 6 Enable and configure NRPE on the compute node.

systemctl enable nrpe

perl -pi -e "s/^allowed_hosts=/# allowed_hosts=/" /etc/nagios/nrpe.cfg

echo "nrpe 5666/tcp # NRPE" >> /etc/services

echo "nrpe : *192.168.47.111* **: ALLOW" >> /etc/hosts.allow**

echo "nrpe : ALL : DENY" >> /etc/hosts.allow

/usr/sbin/useradd -c "NRPE user for the NRPE service" -d /var/run/nrpe -r -g nrpe -s /sbin/nologin nrpe

/usr/sbin/groupadd -r nrpe

NOTE

In the command, *192.168.47.111* is the IP address of the Nagios server.

Step 7 Configure remote services on the compute node.

mv /etc/nagios/conf.d/services.cfg.example /etc/nagios/conf.d/services.cfg

Step 8 Configure the compute node on the server.

mv /etc/nagios/conf.d/hosts.cfg.example /etc/nagios/conf.d/hosts.cfg

Step 9 Update the alarm email notification information on the server.

perl -pi -e "s/ \/bin\/mail/ \/usr\/bin\/mailx/g" /etc/nagios/objects/commands.cfg

perl -pi -e "s/nagios\@localhost/root\@${sms_name}/" /etc/nagios/objects/contacts.cfg

Step 10 Set the password of the web service user on the server.

htpasswd -bc /etc/nagios/passwd nagiosadmin huawei

Step 11 Start the processes on the server.

systemctl enable nagios.service

systemctl start nagios.service

chmod u+s `which ping`

systemctl start httpd

systemctl enable httpd

----End

5.5.6 Installing Test Tools

5.5.6.1 Installing the Performance tools-imb Tool

Procedure

Step 1 Use PuTTY to log in to the server as the **root** user.

Step 2 Run the following commands to install OpenMPI and IMB on the compute node:

yum install openmpi3-gnu7-ohpc.aarch64 -y

yum install imb-gnu7-openmpi3-ohpc.aarch64 -y

Step 3 Configure environment variables in the **.bashrc** file of the compute node.

1. Open **.bashrc**.

   vi /root/.bashrc

2. Press **i** to enter the insert mode and modify the file as follows:

   ```
   module use /opt/ohpc/pub/moduledeps/gnu7-openmpi3/
   module use /opt/ohpc/pub/moduledeps/gnu7
   module use /opt/ohpc/pub/modulefiles/
   module add openmpi3/3.1.0
   module add imb/2018.1
   ```

3. Press **Esc**, type **:wq!**, and press **Enter** to save the file and exit.

----End

5.5.6.2 Installing the TAU Tool

Procedure

Step 1 Use PuTTY to log in to the server as the **root** user.

Step 2 Run the following command to install the TAU tool on the compute node.

yum install tau-gnu7-openmpi3-ohpc.aarch64 -y

Step 3 Configure environment variables on the compute node.

module use /opt/ohpc/pub/moduledeps/gnu7-openmpi3/

module use /opt/ohpc/pub/moduledeps/gnu7

module add openmpi3/3.1.0

module add tau/2.27.1

module add imb/2018.1

----End

5.5.6.3 Installing the PAPi Tool

Procedure

Step 1 Use PuTTY to log in to the server as the **root** user.

Step 2 Run the following commands to install PAPi and Automake on the server:

yum install -y papi-ohpc

yum install -y automake-ohpc

Step 3 Configure PAPi and Automake environment variables.

module add papi/5.7.0

export PATH= /opt/ohpc/pub/utils/autotools/bin:$PATH

----End

6 OpenMPI 4.0.1 Installation Guide

6.1 Introduction

Open MPI Overview

Open MPI is a high-performance message passing interface (MPI) library project combining technologies and resources from several other projects (FT-MPI, LA-MPI, LAM/MPI, and PACX-MPI). It is an open-source implementation of the MPI-2 standard, and developed and maintained by some scientific research institutions and enterprises. Therefore, Open MPI can obtain professional expertise, industrial technologies, and resources from the high-performance community to create the best MPI library for system and software vendors, program developers, and researchers.

Recommended Software Version

Open MPI 4.0.1

6.2 Environment Requirements

Hardware Requirements

Table 6-1 lists the hardware requirements.

Table 6-1 Hardware requirements

Item	Description
CPU	Kunpeng 920

Software requirements

Table 6-2 lists the software requirements.

Table 6-2 Software requirements

Item	Version	How to Obtain
Open MPI	4.0.1	**https://www.open-mpi.org/software/ompi/v4.0/**

OS Requirements

Table 6-3 lists the OS requirements.

Table 6-3 OS requirements

Item	Version	How to Obtain
CentOS	7.6	**https://www.centos.org/download/**

6.3 Planning Data

This chapter lists the software installation paths involved in the Open MPI installation.

Table 6-4 Data plan

No.	Software Installation Path	Description	Remarks
1	-	Installation path of each software package required for setting up the basic environment.	For details, see "Planning the Installation Paths" in **HPC Solution Basic Environment Setup Guide**.

No.	Software Installation Path	Description	Remarks
2	/path/to/OPENMPI	Installation path of Open MPI.	The installation paths listed in this table are only examples. Shared paths are recommended. All the paths used in the commands in this document are examples only. Use the actual paths planned during the installation process.

6.4 Configuring the Installation Environment

Prerequisites

The installation packages are uploaded to the destination directories on the server using an SFTP tool.

Configuration Process

Table 6-5 Configuration process

No.	Operation	Description
1	Configure the basic environment.	For details, see "Setting Up the Single-Node System Environment" in **HPC Solution Basic Environment Setup Guide**.

6.5 Deploying Open MPI

6.5.1 Obtaining the Software Package

Procedure

Step 1 Download the Open MPI installation package **openmpi-4.0.1.tar.gz**.

https://download.open-mpi.org/release/open-mpi/v4.0/openmpi-4.0.1.tar.gz

Step 2 Use an SFTP tool to upload the Open MPI installation package to the */path/to/OpenMPI* directory on the server.

----End

6.5.2 Compiling and Installing Open MPI

Procedure

Step 1 Use PuTTY to log in to the server as the **root** user.

Step 2 Run the following command to install dependencies:

yum install numactl-devel-* systemd-devel-*

Step 3 Load the compiler.

export PATH=*/path/to/GNU***/bin:$PATH**

export LD_LIBRARY_PATH=*/path/to/GNU***/lib64:$LD_LIBRARY_PATH**

Step 4 Decompress the Open MPI installation package.

cd */path/to/OPENMPI*

tar -xvf openmpi-4.0.1.tar.gz

Step 5 Run the following commands:

cd openmpi-4.0.1

./configure --prefix=*/path/to/OPENMPI* **--enable-pretty-print-stacktrace --enable-orterun-prefix-by-default --with-knem=/opt/knem-1.1.3.90mlnx1/ --with-hcoll=/opt/mellanox/hcoll/ --with-cma --with-ucx --enable-mpi1-compatibility CC=gcc CXX=g++ FC=gfortran**

NOTE

- **--with-ucx**: Use the built-in library **/usr/lib64/ucx**.
- **--with-knem** and **--with-hcoll**: Install the mellanox driver. For details, see "Installing the InfiniBand NIC Driver" in **HPC Solution Basic Environment Setup Guide**.

Step 6 Compile and install Open MPI.

make -j 16

make install

----End

6.6 Verifying Open MPI

Procedure

Step 1 Use PuTTY to log in to the server as the **root** user.

Step 2 Run the following commands to load the environment variables:

export PATH=*/path/to/GNU***/bin:*/path/to/OPENMPI***/bin:$PATH**

export LD_LIBRARY_PATH=*/path/to/GNU***/lib64:*/path/to/OPENMPI***/lib:$LD_LIBRARY_PATH**

Step 3 Check whether Open MPI is successfully installed.

mpirun --version

Open MPI is successfully installed if the following information is displayed:

```
mpirun (Open MPI) 4.0.1
Report bugs to http://www.open-mpi.org/community/help/
```

----End

7 Slurm 18.08.7 Installation Guide

7.1 Introduction

Slurm Overview

Slurm is an open-source, highly scalable cluster management tool and job scheduling system for Linux clusters of various scales. It provides the following key features:

Resource allocation

Allocates exclusive or non-exclusive resources of a certain period for users to run jobs.

Job management framework

Provides a framework for starting, executing, and monitoring parallel jobs on the allocated resources.

Queues

Places jobs in a queue when the submitted jobs require more resources than the available resources.

Abundant job scheduling policies

Provides advanced job scheduling policies, such as resource reservation, fair-share scheduling, and backfilling.

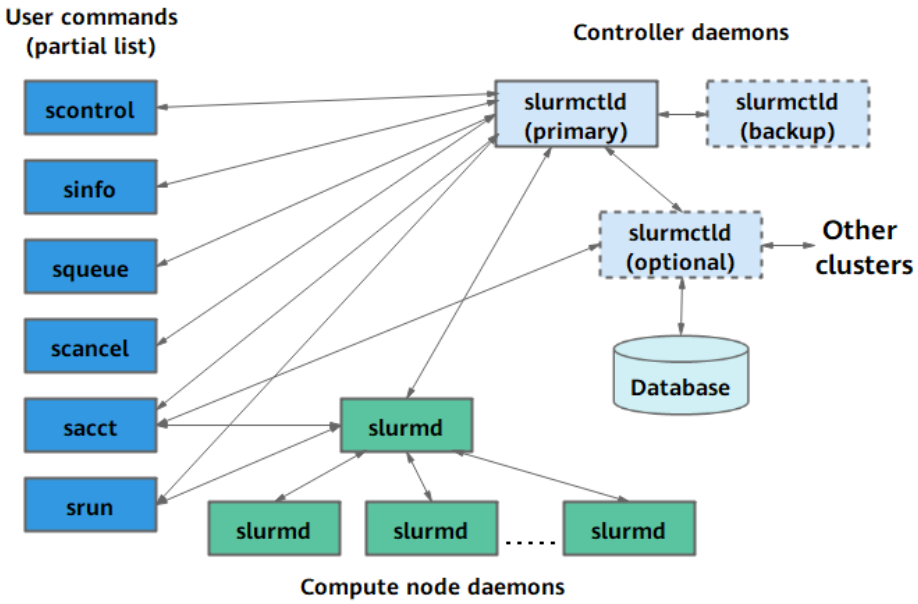
Other tools

Provide tools such as job information statistics and job status diagnosis.

Recommended Software Version

Slurm 18.08.7

Software Architecture



Slurm uses slurmctld, a centralized management process, to monitor resources and jobs.

Each compute node has a slurmd daemon, which waits for jobs, executes jobs, returns the result, and waits for more jobs.

slurmdbd is optional. It records job statistics of multiple clusters managed by Slurm in a database.

For more details, visit:

https://slurm.schedmd.com/overview.html

7.2 Environment Requirements

Hardware Requirements

Table 7-1 lists the hardware requirements.

Table 7-1 Hardware requirements

Item	Description
CPU	Kunpeng 920

Software Requirements

Table 7-2 lists the software required.

Table 7-2 Software requirements

Item	Version	How to Obtain
Slurm	18.08.7	**https://www.schedmd.com/downloads.php**
munge	0.5.13	**https://github.com/dun/munge/releases/tag/munge-0.5.13**

OS Requirements

Table 7-3 lists the OS requirements.

Table 7-3 OS requirements

Item	Version	How to Obtain
CentOS	7.6	**https://www.centos.org/download/**

Cluster Information

Table 7-4 lists the cluster information.

Table 7-4 Cluster Information

Node	IP Address	Function
master	192.168.40.11	A management node that runs slurmctld.
testnode1	192.168.40.111	A compute node that runs slurmd.
testnode2	192.168.40.112	A compute node that runs slurmd.

7.3 Planning Data

This chapter lists the software installation paths involved in the Slurm installation.

Table 7-5 Data plan

No.	Software Installation Path	Function	Description
1	/path/to/SLURM	Installation path of Slurm	The installation paths listed in this table are only examples. Shared paths are recommended. All the paths used in the commands in this document are examples only. Use the actual paths planned during the installation process.
2	/path/to/MUNGE	Installation path of munge	

7.4 Creating the Slurm RPM Package

7.4.1 Downloading Software Packages

Step 1 Download the munge installation package **munge-0.5.13.tar.xz**.

URL: **https://github.com/dun/munge/releases/download/munge-0.5.13/munge-0.5.13.tar.xz**

Step 2 Download the Slurm installation package **slurm-18.08.7.tar.bz2**.

Download address: **https://src.fedoraproject.org/lookaside/extras/slurm/slurm-18.08.7.tar.bz2/sha512/d0047086f1b716877cc5bb39539bf96a8fd08b1851c85fd85112c6432c1ce2a0f29fc9dd8803094c8fa44d063cec5f417e6bed231b6d338934ff4b48424a5a93/slurm-18.08.7.tar.bz2**

Step 3 Use the SFTP tool to upload the software packages to the server.

- Upload the munge installation package to the */path/to/MUNGE* directory on the server.
- Upload the Slurm installation package to the */path/to/SLURM* directory on the server.

----End

7.4.2 Installing Dependencies

Procedure

Step 1 Use PuTTY to log in to the server as the **root** user.

Step 2 Install the dependent library.

yum install -y rpm-build rpmdevtools bzip2-devel openssl-devel zlib-devel readline-devel pam-devel perl-DBI perl-ExtUtils-MakeMaker mariadb*

----End

7.4.3 Compiling munge

Procedure

Step 1 Use PuTTY to log in to the server as the **root** user.

Step 2 Build the munge RPM package.

cd */path/to/MUNGE*

rpmbuild -tb --clean munge-0.5.13.tar.xz

Step 3 Check whether the munge RPM package is successfully created.

ls /root/rpmbuild/RPMS/aarch64/ | grep munge

```
munge-0.5.13-1.el7.aarch64.rpm
munge-debuginfo-0.5.13-1.el7.aarch64.rpm
munge-devel-0.5.13-1.el7.aarch64.rpm
munge-libs-0.5.13-1.el7.aarch64.rpm
```

Step 4 Create a **mungerpm** folder in */path/to/MUNGE* and copy the munge RPM package from **/root/rpmbuild/RPMS/aarch64** to */path/to/MUNGE***/mungerpm**.

mkdir -p */path/to/MUNGE/***mungerpm**

cp /root/rpmbuild/RPMS/aarch64/munge* */path/to/MUNGE/***mungerpm –f**

----End

7.4.4 Compiling Slurm

Procedure

Step 1 Use PuTTY to log in to the server as the **root** user.

Step 2 Switch to the munge directory.

cd /home/mungerpm

Step 3 Install munge on **testnode1** and **testnode2**.

yum install -y munge*

Step 4 Switch to the Slurm directory.

cd */path/to/SLURM*

Step 5 Build the Slurm RPM package.

rpmbuild -ta --clean slurm-18.08.7.tar.bz2

Step 6 Check whether the Slurm RPM package is successfully created.

ls /root/rpmbuild/RPMS/aarch64/ | grep slurm

```
slurm-18.08.7-1.el7.aarch64.rpm
slurm-contribs-18.08.7-1.el7.aarch64.rpm
slurm-devel-18.08.7-1.el7.aarch64.rpm
slurm-example-configs-18.08.7-1.el7.aarch64.rpm
slurm-libpmi-18.08.7-1.el7.aarch64.rpm
slurm-openlava-18.08.7-1.el7.aarch64.rpm
```

```
slurm-pam_slurm-18.08.7-1.el7.aarch64.rpm
slurm-perlapi-18.08.7-1.el7.aarch64.rpm
slurm-slurmctld-18.08.7-1.el7.aarch64.rpm
slurm-slurmd-18.08.7-1.el7.aarch64.rpm
slurm-slurmdbd-18.08.7-1.el7.aarch64.rpm
slurm-torque-18.08.7-1.el7.aarch64.rpm
```

Step 7 Create a **slurmrpm** folder in */path/to/SLURM* and copy the Slurm RPM package from **/root/rpmbuild/RPMS/aarch64** to */path/to/SLURM***/slurmrpm**.

mkdir -p */path/to/SLURM/***slurmrpm**

cp /root/rpmbuild/RPMS/aarch64/slurm* */path/to/SLURM/***slurmrpm -f**

----End

7.5 Installing and Configuring Slurm

7.5.1 Installing munge

Procedure

Step 1 Use PuTTY to log in to the server as the **root** user.

Step 2 Run the following command to mount the **/home** directory of the master node to testnode1 and testnode2:

mount master:/home /home

Step 3 Install the munge RPM package on testnode1 and testnode2.

cd /home/mungerpm

yum localinstall -y munge*

Step 4 On the master, testnode1, and testnode2 nodes, change the permissions for the **munge** directory.

chmod -Rf 700 /etc/munge

chmod -Rf 711 /var/lib/munge

chmod -Rf 700 /var/log/munge

chmod -Rf 0755 /var/run/munge

Step 5 Start the ntpd service on the master node.

yum install -y ntp

systemctl start ntpd

Step 6 On testnode1 and tesnode2, synchronize the system time with the master node.

ntpdate master

Step 7 Copy **/etc/munge/munge.key** from the master node to testnode1 and testnode2.

scp /etc/munge/munge.key testnode1:/etc/munge/

scp /etc/munge/munge.key testnode2:/etc/munge/

Step 8 On testnode1 and tesnode2, change the permissions for the **/etc/munge/munge.key** file.

chown munge.munge /etc/munge/munge.key

Step 9 Start munge on the master node, testnode1, and testnode2.

systemctl start munge

systemctl enable munge

----End

7.5.2 Installing Slurm

Procedure

Step 1 Use PuTTY to log in to the server as the **root** user.

Step 2 Install the Slurm RPM package on the master, testnode1, and testnode2 nodes.

cd /home/slurmrpm

yum install -y slurm*

Step 3 Check whether the **slurm** user is created on all nodes.

- If yes, run the following command to query user information:

  grep "slurm" /etc/group

  ```
  slurm:x:202:
  ```

- If no, create the **slurm** user on the master, testnode1, and testnode2 nodes.

  groupadd -g 202 slurm

  useradd -u 202 -g 202 slurm

Step 4 Create the **/var/spool/slurm/ssl**, **/var/spool/slurm/d**, and **/var/log/slurm** directories on the master, testnode1, and testnode2 nodes.

mkdir -p /var/spool/slurm/ssl

mkdir -p /var/spool/slurm/d

mkdir -p /var/log/slurm

Step 5 On the master, testnode1, and testnode2 nodes, set permissions for **/var/spool/slurm**.

chown -R slurm.slurm /var/spool/slurm

Step 6 Modify the **/etc/slurm/slurm.conf** file on the master node.

1. Open **/etc/slurm/slurm.conf**.

   vi /etc/slurm/slurm.conf

2. Press **i** to enter the insert mode and add the following content:

   ```
   ControlMachine=master
   ControlAddr=192.168.40.11
   MpiDefault=none
   ProctrackType=proctrack/pgid
   ReturnToService=1
   SlurmctldPidFile=/var/run/slurmctld.pid
   ```

```
SlurmdPidFile=/var/run/slurmd.pid
SlurmdSpoolDir=/var/spool/slurm/d
SlurmUser=slurm
#SlurmdUser=root
StateSaveLocation=/var/spool/slurm/ssl
SwitchType=switch/none
TaskPlugin=task/none
FastSchedule=1
SchedulerType=sched/backfill
SelectType=select/linear
AccountingStorageType=accounting_storage/none
ClusterName=cluster
JobAcctGatherType=jobacct_gather/none
SlurmctldDebug=3
SlurmctldLogFile=/var/log/slurm/slurmctld.log
SlurmdDebug=3
SlurmdLogFile=/var/log/slurm/slurmd.log

NodeName=testnode1 CPUs=96 Sockets=4 CoresPerSocket=24 State=UNKNOWN
NodeName=testnode2 CPUs=40 Sockets=4 CoresPerSocket=10 State=UNKNOWN

PartitionName=ARM Nodes=testnode1 Default=YES MaxTime=INFINITE State=UP
PartitionName=X86 Nodes=testnode1 Default=YES MaxTime=INFINITE State=UP
```

3. Press **Esc**, type **:wq!**, and press **Enter** to save the changes and exit.

Step 7 On the master node, run the following commands to copy the **/etc/slurm/slurm.conf** file to the testnode1 and testnode2 nodes:

scp /etc/slurm/slurm.conf testnode1:/etc/slurm

scp /etc/slurm/slurm.conf testnode2:/etc/slurm

Step 8 Start the slurmctld service on the master node.

systemctl start slurmctld

systemctl enable slurmctld

Step 9 Start the slurmd service on the testnode1 and testnode2 nodes.

systemctl start slurmd

systemctl enable slurmd

----End

7.6 Using Slurm

7.6.1 Common Commands

The following table describes the common commands.

Table 7-6 Common commands

Command	Description
sinfo	Checks node and partition status.
squeue	Checks the task queue status.

Command	Description
scontrol show nodes	Displays detailed node information.
scontrol show jobs	Displays detailed job information.
srun	Executes a Job.
salloc	Allocates resources.
sbatch	Submits a script job.
scancel	Cancels a job.

7.6.2 Common Script Operations

7.6.2.1 Writing a Script

Procedure

Step 1 Use PuTTY to log in to the server as the **root** user.

Step 2 Write a script.

1. Open the script.

   vi test.slurm

2. Press **i** to enter the insert mode and add the following content:

```
#!/bin/bash
#SBATCH -J slurmtest
#SBATCH -o %J.out
#SBATCH -e %J.err
#SBATCH -N 1
#SBATCH --exclusive
#SBATCH -p X86

for x in {0..10}
do
echo "This is a slurmtest for `arch`!"
sleep 5
done
```

3. Press **Esc**, type **:wq!**, and press **Enter** to save the script and exit.

----End

7.6.2.2 Submitting a Job

Procedure

Step 1 Use PuTTY to log in to the server as the **root** user.

Step 2 Run the following command to submit a job:

sbatch test.slurm

----End

7.6.2.3 Canceling a Job

Procedure

Step 1 Use PuTTY to log in to the server as the **root** user.

Step 2 Query the job ID.

squeue

Step 3 Run the following command to cancel the job:

scancel +*Job ID*

Example: **scancel 123**

----End

7.6.2.4 Common Parameters

Table 7-7 describes the common parameters.

Table 7-7 Common parameters

Parameter	Description
-J	Specifies the job name.
-o	Specifies the file that stores the job output.
-e	Specifies the file that stores error information.
--nodes / -N	Specifies the number of nodes.
--ntasks / -n	Specifies the total number of processes.
--exclude	Specifies the nodes that are excluded.
--nodelist	Specifies the list of nodes to be used.
--time	Specifies the maximum running time of the job.
--exclusive	Specifies the exclusive use of the assigned nodes.
--partition/-P	Specifies the partitions in which the nodes can be allocated to the job.
-d singleton	Runs only one of the files with the same name.

7.6.3 Querying Slurm Node Status

Procedure

Step 1 Use PuTTY to log in to the server as the **root** user.

Step 2 Run the **sinfo** command on the master node to query node status.

sinfo

```
PARTITION    AVAIL  TIMELIMIT  NODES  STATE NODELIST
ARM          up   infinite1 idletestnode1
X86*  up    infinite1 idletestnode2
```

Table 7-8 Description

Status	Description
unk	The node status is unknown.
idle	The node is in idle state.
alloc	The node is allocated.
down	The node is faulty.

----End

7.6.4 More information

For more information, visit **https://slurm.schedmd.com/documentation.html**.

7.7 Troubleshooting

7.7.1 Abnormal Compute Node Status

Symptom

The compute node is **down** and cannot be recovered automatically.

Procedure

Step 1 Use PuTTY to log in to the server as the **root** user.

Step 2 Run the following command to manually restore the node:

scontrol update nodename=testnode1 state=resume

Step 3 Check the node status.

sinfo

```
PARTITION    AVAIL  TIMELIMIT  NODES  STATE  NODELIST
ARM          up     infinite   1      idle   testnode1
X86*         up     infinite   1      idle   testnode2
```

----End

8 Change History

Date	Description
2022-01-30	This issue is the sixth official release. Modified **7.5.2 Installing Slurm** in the *Slurm 18.08.7 Installation Guide* and updated the address for downloading the munge installation package.
2021-10-26	This issue is the fifth official release. Modified the compilation and installation path in **2.4.2 Compiling and Installing GMP** in the *GNU 9.1 Installation Guide*.
2021-08-20	This issue is the fourth official release. Deleted redundant spaces from **Step 4** in the *Basic Environment Setup Guide*.
2021-07-23	This issue is the third official release. Modified the address for downloading the Slurm installation package in the *Slurm 18.08.7 Installation Guide*.
2021-04-29	This issue is the second official release. Modified **1.4.8 Installing Open MPI** in the *Basic Environment Setup Guide*.
2020-03-20	This issue is the first official release.

