LiCO 7.1.0
Installation Guide (for EL8)

Fifteenth Edition (June 2023)

Reading instructions

- To ensure that commands can be copied and pasted from this document correctly, open this Guide with Adobe Acrobat Reader.
- Replace values in angle brackets with the actual values. For example, enter the actual username and password for <*_USERNAME> and <*_PASSWORD>.
- The annotations starting with “#” are the explanation for command lines.

Contents

Chapter 1. Overview

Introduction to LiCO

Lenovo Intelligent Computing Orchestration (LiCO) is an infrastructure management software for high-performance computing (HPC) and artificial intelligence (AI). It provides features like cluster management and monitoring, job scheduling and management, cluster user management, account management, and file system management.

With LiCO, users can centralize resource allocation in one supercomputing cluster and carry out HPC and AI jobs simultaneously. Users can perform operations by logging in to the management system interface with a browser, or by using command lines after logging in to a cluster login node with another Linux shell.

Typical cluster deployment

This Guide is based on the typical cluster deployment that contains management, login, and compute nodes.

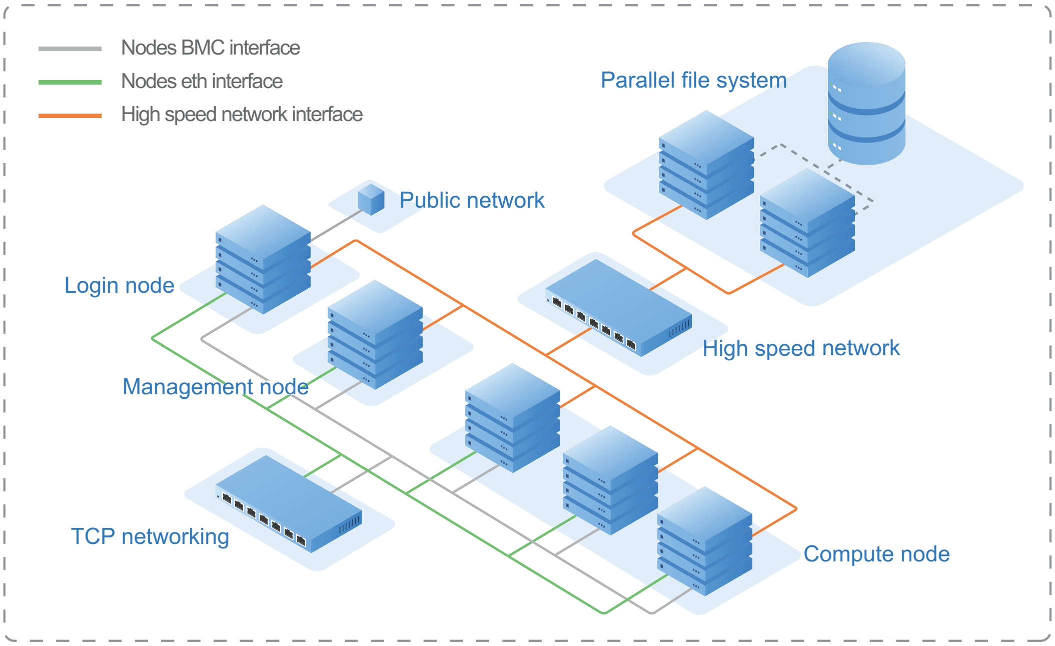


Figure 1. Typical cluster deployment

Elements in the cluster are described in the table below.

Table 1. Description of elements in the typical cluster

Element	Description
Management node	Core of the HPC/AI cluster, undertaking primary functions such as cluster management, monitoring, scheduling, strategy management, and user & account management.
Compute node	Completes computing tasks.
Login node	Connects the cluster to the external network or cluster. Users must use the login node to log in and upload application data, develop compilers, and submit scheduled tasks.
Parallel file system	A parallel file system provides a shared storage function. It is connected to the cluster nodes through a high-speed network. Parallel file system setup is beyond the scope of this Guide. A simple NFS setup is used instead.
Nodes BMC interface	Used to access the node BMC system.
Nodes eth interface	Used to manage nodes in cluster. It can also be used to transfer computing data.
High speed network interface	Optional. Used to support the parallel file system. It can also be used to transfer computing data.

Note: LiCO also supports the cluster deployment that only contains the management and compute nodes. In this case, all LiCO modules installed on the login node need to be installed on the management node.

Operating environment

Cluster server:

Lenovo ThinkSystem servers

Operating system:

Rocky 8.6/Red Hat Enterprise Linux(RHEL) 8.6

Client requirements:

- Hardware: CPU of 2.0 GHz or above, memory of 8 GB or above
- Browser: Chrome (V110.0 or higher) or Firefox (V110.0 or higher) recommended
- Display resolution: 1280 x 800 or above

Supported servers and chassis models

LiCO can be installed on certain servers, as listed in the table below.

Table 2. Supported servers

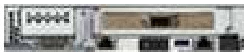
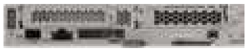
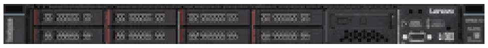
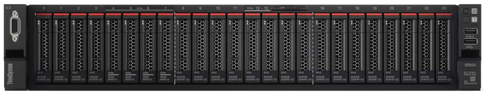
Product code	Machine type	Product name	Appearance
sd530	7X21	Lenovo ThinkSystem SD530 (0.5U)	
sd650	7X58	Lenovo ThinkSystem SD650 (2 nodes per 1U tray)	
sd650 v2	7D1M	ThinkSystem SD650 V2 (2 nodes per 1U tray)	
sd650 v3	7D7M	ThinkSystem SD650 V3 (2 nodes per 1U tray)	
sd650-i v3	7D7L	Lenovo ThinkSystem SD650-I V3 (1U)	
sd650-n v2	7D1N	Lenovo ThinkSystem SD650-N V2 (1U)	
sd665 v3	7D9P	ThinkSystem SD665 V3 (2 nodes per 1U tray)	
sr630	7X01, 7X02	Lenovo ThinkSystem SR630 (1U)	
sr630 v2	7Z70, 7Z71	Lenovo ThinkSystem SR630 V2 (1U)	
sr630 v3	7D72, 7D73, 7D74	Lenovo ThinkSystem SR630 V3 (1U)	
sr635	7Y98, 7Y99	Lenovo ThinkSystem SR635 (1U)	
sr635 v3	7D9H, 7D9G	Lenovo ThinkSystem SR635 V3 (1U)	
sr645	7D2X, 7D2Y	Lenovo ThinkSystem SR645 (1U)	
sr645 v3	7D9D, 7D9C	Lenovo ThinkSystem SR645 V3 (1U)	
sr650	7X05, 7X06	Lenovo ThinkSystem SR650 (2U)	

Table 2. Supported servers (continued)

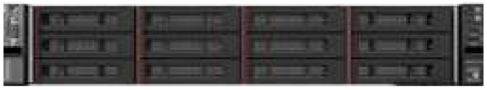
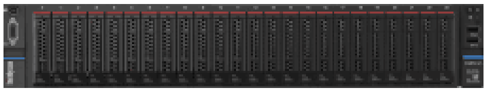
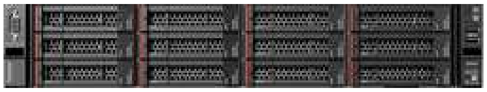
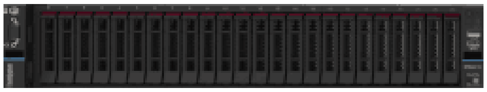
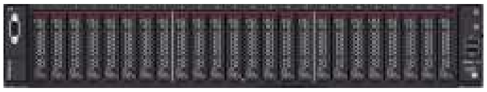
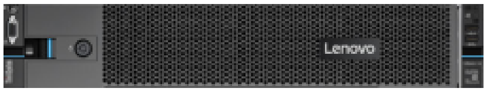
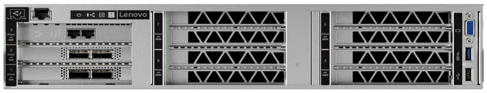
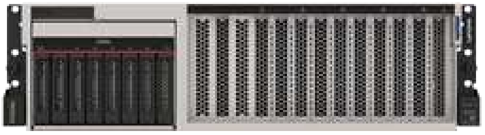
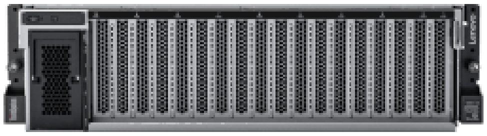
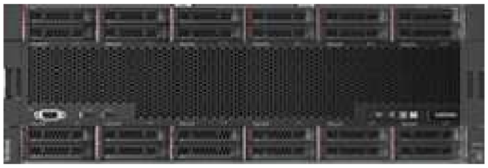
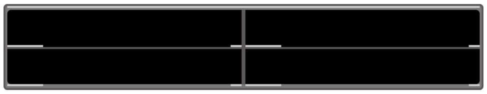
Product code	Machine type	Product name	Appearance
sr650 v2	7Z72,7Z73	Lenovo ThinkSystem SR650 V2 (2U)	
sr650 v3	7D75, 7D76, 7D77	Lenovo ThinkSystem SR650 V3 (2U)	
sr655	7Y00, 7Z01	Lenovo ThinkSystem SR655 (2U)	
sr655 v3	7D9F, 7D9E	Lenovo ThinkSystem SR655 V3 (2U)	
sr660 v2	7D6L	Lenovo ThinkServer SR660 V2 (2U)	
sr665	7D2V, 7D2W	Lenovo ThinkSystem SR665 (2U)	
sr665 v3	7D9B, 7D9A	Lenovo ThinkSystem SR665 V3(2U)	
sr670	7Y36, 7Y37, 7Y38	Lenovo ThinkSystem SR670 (2U)	
sr670 v2	7Z22, 7Z23, 7D47	Lenovo ThinkSystem SR670 V2 (3U)	
sr675 v3	7D9Q, 7D9R	Lenovo ThinkSystem SR675 V3 (3U)	
sr850	7X18, 7X19	Lenovo ThinkSystem SR850 (2U)	

Table 2. Supported servers (continued)

Product code	Machine type	Product name	Appearance
sr850p	7D2F, 7D2G, 7D2H	Lenovo ThinkSystem SR850P (2U)	
sr950	7X11, 7X12, 7X13	Lenovo ThinkSystem SR950 (4U)	

LiCO can be installed on certain chassis models, as listed in the table below.

Table 3. Supported chassis models

Product code	Machine type	Model name	Appearance
d2	7X20	D2 Enclosure (2U)	
n1200	5456, 5468, 5469	NeXtScale n1200 (6U)	
dw612s	7D1L	DW612S enclosure (12U)	

Prerequisites

- Refer to LiCO best recipe to ensure that the cluster hardware uses proper firmware levels, drivers, and settings:
  https://support.lenovo.com/us/en/solutions/ht507011
- Refer to the firmware levels part of LeSI 23A_SI best recipe to install the operating system security patch:
  https://support.lenovo.com/us/en/solutions/HT510136
- Unless otherwise stated in this Guide, all commands are run on the management node.
- To enable the firewall, modify the firewall rules according to "Firewall settings" on page 63.

- To prevent security vulnerabilities, it is recommended to regularly patch and update components and the operating system.
- Before setting up LiCO, it is recommended to apply the latest updates during or immediately after deploying operating system to the managed nodes.
- LiCO leverages OpenHPC packages which aggregate a number of common ingredients required to deploy and manage High Performance Computing (HPC) Linux clusters including provisioning tools, resource management, I/O clients, development tools, and a variety of scientific libraries. Lenovo provides a download of the most recent version of OpenHPC which is unmodified from what is distributed by OpenHPC. There are known open-source components within OpenHPC that have known, registered, vulnerabilities. None of these issues has been assessed as critical. However, it is recommended that the user update or remove such components using the native package management tools.
- To deploy LiCO in container, the management node and the login node share the same node. Therefore, it’s not recommended to set the login node. To deploy the management node and the login node separately, consult Lenovo Service.

Chapter 2. Deploy the cluster environment

If the cluster environment already exists, skip this chapter.

Install the operating system

Install an official version of Rocky 8.6 or RHEL 8.6. Users can select the minimum installation.

Configure the memory and restart the operating system:

```
echo '* soft memlock unlimited' >> /etc/security/limits.conf

echo '* hard memlock unlimited' >> /etc/security/limits.conf

echo '* soft stack unlimited' >> /etc/security/limits.conf

echo '* hard stack unlimited' >> /etc/security/limits.conf

reboot
```

Deploy the operating system on other nodes in the cluster

Configure environment variables

Step 1. Log in to the management node.

Step 2. Edit `/root/lico_env.local` and update the environment variables listed in that file::

```
# Management node hostname

sms_name="head"

# IP address of management node in the cluster intranet

sms_ip="192.168.0.2"

# Network interface card MAC address corresponding to the management node IP

sms_mac='b8:59:9f:2b:a2:e2'

# Management node BMC address.

sms_bmc='192.168.1.2'

# set the dns server

dns_server="192.168.10.10"

# set the ipv4 gateway

ipv4_gateway="192.168.0.1"

# Set the domain name

domain_name="hpc.com"

# Set OpenLDAP domain name

lico_ldap_domain_name="dc=hpc,dc=com"
```

```
# set OpenLDAP domain component
lico_ldap_domain_component="hpc"
# original OS repository directory
repo_backup_dir="/install/custom/backup"
# OS image pathway
iso_path="/isos"
# Local repository directory for OS
os_repo_dir="/install/custom/server"
sdk_repo_dir="/install/custom/sdk"
# Local repository directory for confluent
confluent_repo_dir="/install/custom/confluent"
# link name of repository directory for Lenovo OpenHPC
link_ohpc_repo_dir="/install/custom/ohpc"
# link name of repository directory for LiCO
link_lico_repo_dir="/install/custom/lico"
# link name of repository directory for LiCO-dep
link_lico_dep_repo_dir="/install/custom/lico-dep"
# link name of directory for lico moniotr
link_lico_monitor_dir="/install/custom/lico-monitor"
# Local directory for for lico monitor, please change it according to this version.
lico_monitor_dir="/install/custom/lico-monitor-1.0.0"
# Local repository directory for Lenovo OpenHPC, please change it
# according to this version.
ohpc_repo_dir="/install/custom/ohpc-2.6.1"
# LiCO repository directory for LiCO, please change it according to this version.
lico_repo_dir="/install/custom/lico-7.1.0"
# LiCO repository directory for LiCO-dep, please change it according to this version.
lico_dep_repo_dir="/install/custom/lico-dep-7.1.0"
# When the GPU vendor is not NVIDIA, need to specify the GPU vendor.
# Example: gpu_vendor="intel"
gpu_vendor=""
# icinga api listener port
icinga_api_port=5665
```

```
# If the confluent automatic discovery mode is enabled, skip the following configurations.
# Total compute nodes
num_computes="2"
# Prefix of compute node hostname.
# Change the configuration according to actual conditions.
compute_prefix="c"
# Compute node hostname list.
# Change the configuration according to actual conditions.
c_name[0]=c1
c_name[1]=c2
# Compute node IP list.
# Change the configuration according to actual conditions.
c_ip[0]=192.168.0.6
c_ip[1]=192.168.0.16
# Network interface card MAC address corresponding to the compute node IP.
# Change the configuration according to actual conditions.
c_mac[0]=fa:16:3e:73:ec:50
c_mac[1]=fa:16:3e:27:32:c6
# Compute node BMC address list.
c_bmc[0]=192.168.1.6
c_bmc[1]=192.168.1.16

# Total login nodes. If there is no login node in the cluster, or the management node
# and the login node is the same node, the number of logins must be "0".
# And the 'l_name', 'l_ip', 'l_mac', and 'l_bmc' lines need to be removed.
num_logins="1"
# Login node hostname list.
# Change the configuration according to actual conditions.
l_name[0]=l1
# Login node IP list.
# Change the configuration according to actual conditions.
l_ip[0]=192.168.0.15
# Network interface card MAC address corresponding to the login node IP.
# Change the configuration according to actual conditions.
```

```
l_mac[0]=fa:16:3e:2c:7a:47

# Login node BMC address list.

l_bmc[0]=192.168.1.15
```

Step 3. Save the changes to `lico_env.local`, and reload the environment variables:

```
chmod 600 lico_env.local

source lico_env.local
```

Note: This Guide assumes that the node's BMC username and password are consistent. If they are inconsistent, they need to be modified during the installation.

After the cluster environment is set up, configure the IP address of the public network on the login or management node, and then users can log in to the LiCO Web portal from an external network.

Create a local repository

Create a local repository to install operating system.

For Rocky

Step 1. Create a directory to store the ISO storage:

```
mkdir -p ${iso_path}
```

Step 2. Download the `Rocky-8.6-x86_64-dvd1.iso` and `CHECKSUM` file from the following Web site: https://rockylinux.org/download

Step 3. Copy the file to `${iso_path}`.

Step 4. Validate that the verification code of the ISO file matches the code listed in `CHECKSUM`.

```
cd ${iso_path}

sha256sum Rocky-8.6-x86_64-dvd1.iso

cd ~
```

Step 5. Mount the ISO image:

```
mkdir -p ${os_repo_dir}

mount -o loop ${iso_path}/Rocky-8.6-x86_64-dvd1.iso ${os_repo_dir}
```

Step 6. Configure the local repository and copy it to `/etc/yum.repos.d/`:

```
cat << eof > ${iso_path}/EL8-OS.repo

[AppStream]

name=appstream

baseurl=file://${os_repo_dir}/AppStream/

enabled=1

gpgcheck=1

gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY-rockyofficial

[BaseOS]

name=baseos

baseurl=file://${os_repo_dir}/BaseOS/
```

```
enabled=1

gpgcheck=1

gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY-rockyofficial

eof

cp -a ${iso_path}/EL8-OS.repo /etc/yum.repos.d/
```

Step 7. Make a backup of the repository:

```
mkdir -p ${repo_backup_dir}

mv /etc/yum.repos.d/Rocky* ${repo_backup_dir}

dnf clean all

dnf makecache
```

Step 8. Enable the NGINX web server:

```
dnf module reset nginx

dnf module enable -y nginx:1.20
```

For RHEL

Step 1. Create a directory to store the ISO storage:

```
mkdir -p ${iso_path}
```

Step 2. Copy the `RHEL-8.6.0-20220420.3-x86_64-dvd1.iso` file to the `${iso_path}` directory.

Step 3. Verify that the ISO file is valid:

```
cd ${iso_path}

md5sum RHEL-8.6.0-20220420.3-x86_64-dvd1.iso

cd ~
```

Step 4. Mount the ISO image:

```
mkdir -p ${os_repo_dir}

mount -o loop ${iso_path}/RHEL-8.6.0-20220420.3-x86_64-dvd1.iso ${os_repo_dir}
```

Step 5. Configure the local repository and copy it to `/etc/yum.repos.d/`:

```
cat << eof > ${iso_path}/EL8-OS.repo

[AppStream]

name=appstream

baseurl=file://${os_repo_dir}/AppStream/

enabled=1

gpgcheck=1

gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY-redhat-release

[BaseOS]

name=baseos

baseurl=file://${os_repo_dir}/BaseOS/
```

```
enabled=1
gpgcheck=1
gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY-redhat-release
eof
cp -a ${iso_path}/EL8-OS.repo /etc/yum.repos.d/
```

Step 6. Enable the NGINX web server:

```
dnf module reset nginx
dnf module enable -y nginx:1.20
```

Install Lenovo confluent

Step 1. Download the following package:
https://hpc.lenovo.com/downloads/23a.1/confluent-3.7.1-el8.tar.xz

Step 2. Upload the package to the `/root` directory.

Step 3. Create confluent local repository:

```
dnf install -y bzip2 tar
mkdir -p $confluent_repo_dir
cd /root
tar -xvf confluent-3.7.1-el8.tar.xz -C $confluent_repo_dir
cd $confluent_repo_dir/lenovo-hpc-el8
./mklocalrepo.sh
cd ~
```

Step 4. Install the Lenovo confluent:

```
dnf install -y lenovo-confluent tftp-server
systemctl enable confluent --now
systemctl enable tftp.socket --now
systemctl disable firewalld --now
systemctl enable httpd --now
```

Step 5. Create confluent account:

```
source /etc/profile.d/confluent_env.sh
confetty create /users/<CONFLUENT_USERNAME> password=<CONFLUENT_PASSWORD> role=admin
```

Step 6. Close SELinux:

```
sed -i 's/enforcing/disabled/' /etc/selinux/config
setenforce 0
```

Deploy the operating system through confluent

Attention: When performing cluster deployment under confluent automatic discovery mode, follow the guidance in the following web sites:

- https://hpc.lenovo.com/users/documentation/confluentdisco.html

- https://hpc.lenovo.com/users/documentation/confluentquickstart_el8.html

It is recommended to create the groups named “all”, “login”, “compute”, and so on in confluent, and bind the nodes with the specific group; otherwise, the commands mentioned in the rest of chapters in this guide might not be used.

Specify global behaviors

Note: Before specifying global behaviors, ensure that the BMC user name and password are consistent in node; if not, they should be modified.

In confluent, most of the configurations are node oriented and can be derived from a group. The default group “everything” providing a method to indicate global settings is automatically added to each node.

```
nodegroupattrib everything deployment.useinsecureprotocols=firmware \

console.method=ipmi dns.servers=$dns_server dns.domain=$domain_name \

net.ipv4_gateway=$ipv4_gateway net.ipv4_method="static"
```

The `deployment.useinsecureprotocols=firmware` enables PXE support (HTTPS only mode is by default the only allowed mode), `console.method=ipmi` may be skipped but if specified instructs confluennt to use IPMI to access the text console to enable the `nodeconsole` command.

While passwords and similar may be specified the same way, it is recommended to use the `-p` argument to prompt for values, to keep them out of the command history. Note that if unspecified, default root password behavior is to disable password based login:

```
nodegroupattrib everything -p bmcuser bmcpass crypted.rootpassword
```

Define nodes in confluent

Step 1. Define the management node in the `lico_env.local` file to confluent:

```
nodegroupdefine all

nodegroupdefine login

nodegroupdefine compute

nodedefine $sms_name

nodeattrib $sms_name net.hwaddr=$sms_mac

nodeattrib $sms_name net.ipv4_address=$sms_ip

nodeattrib $sms_name hardwaremanagement.manager=$sms_bmc
```

Step 2. Define the compute node configuration to confluent:

```
for ((i=0; i<$num_computes; i++)); do

nodedefine ${c_name[$i]};

nodeattrib ${c_name[$i]} net.hwaddr=${c_mac[$i]};

nodeattrib ${c_name[$i]} net.ipv4_address=${c_ip[$i]};

nodeattrib ${c_name[$i]} hardwaremanagement.manager=${c_bmc[$i]};

nodedefine ${c_name[$i]} groups=all,compute;

done
```

Step 3. Define the login node configuration to confluent:

```
for ((i=0; i<$num_logins; i++)); do

nodedefine ${l_name[$i]};

nodeattrib ${l_name[$i]} net.hwaddr=${l_mac[$i]};

nodeattrib ${l_name[$i]} net.ipv4_address=${l_ip[$i]};

nodeattrib ${l_name[$i]} hardwaremanagement.manager=${l_bmc[$i]};

nodedefine ${l_name[$i]} groups=all,login;

done
```

Prepare name resolution

Note: The particular name resolution solution is not mandatory, but the following steps provide a basic strategy if no strategy is already in place.

Step 1. Append node information to `/etc/hosts`:

```
for node_name in $(nodelist); do

noderun -n $node_name echo {net.ipv4_address} {node} {node}.{dns.domain} >> /etc/hosts

done
```

Step 2. Install and start to dnsmasq, making `/etc/hosts` available through dns:

```
dnf install -y dnsmasq

systemctl enable dnsmasq --now
```

Initialize confluent operating system deployment

Users can set up requirements for operating system deployment through the initialized sub-command of the `osdeploy` command. The `-i` parameter is used to interactively prompt the options that are available:

```
ssh-keygen -t ed25519

cp ~/.ssh/id_ed25519.pub ~/.ssh/authorized_keys

chown confluent /var/lib/confluent

osdeploy initialize -i
```

Perform operating system deployment

For Rocky

Step 1. Import install media:

```
osdeploy import ${iso_path}/Rocky-8.6-x86_64-dvd1.iso
```

Step 2. Start deployment:

```
nodedeploy all -n rocky-8.6-x86_64-default
```

Step 3. (Optional) Check the deployment progress:

```
nodedeploy all
```

For RHEL

Step 1. Import install media:

```
osdeploy import ${iso_path}/RHEL-8.6.0-20220420.3-x86_64-dvd1.iso
```

Step 2. Start deployment:

```
nodedeploy all -n rhel-8.6-x86_64-default
```

Step 3. Check the deployment process:

```
nodedeploy all
```

Enable NGINX for other nodes

Attention: If the operating systems of other nodes are Rocky, close the system repo:

```
nodeshell all "mkdir -p ${repo_backup_dir}"

nodeshell all "mv /etc/yum.repos.d/Rocky* ${repo_backup_dir}"

nodeshell all "dnf clean all"

nodeshell all "dnf makecache"
```

Enable the NGINX service for other nodes:

```
nodeshell all dnf module reset nginx

nodeshell all dnf module enable -y nginx:1.20
```

Disable firewall for other nodes

Attention: To enable firewall, refer to "Firewall settings" on page 63.

```
nodeshell all "systemctl disable firewalld --now"

nodeshell all "sed -i 's/enforcing/disabled/' /etc/selinux/config"

nodeshell all "setenforce 0"
```

Checkpoint A

Check and ensure that the installation is completed:

```
nodeshell all uptime
```

Note: The output should be as follows:

```
c1: 05:03am up 0:02, 0 users, load average: 0.20, 0.13, 0.05

c2: 05:03am up 0:02, 0 users, load average: 0.20, 0.14, 0.06

l1: 05:03am up 0:02, 0 users, load average: 0.17, 0.13, 0.05

......
```

Install infrastructure software for nodes

Note: In the **Installation node** column, M stands for "Management node", L stands for "Login node", and C stands for "Compute node".

Table 4. Infrastructure software to be installed

Software name	Component name	Version	Service name	Installation node	Notes
nfs	nfs-utils	2.3.3	nfs-server	M, C, L	/
chrony	chrony	4.1	chronyd	M, C, L	/
slurm	ohpc-slurm-server	2.6.1	munge, slurmctld	M	/
	ohpc-slurm-client	2.6.1	munge, slurmd	C, L	/
icinga2	icinga2	2.13.7	icinga2	M, C, L	/
singularity	singularity-ohpc	3.7.1	/	M	/
mpi	openmpi4-gnu12-ohpc	4.1.4	/	M	At least one MPI type required
	mpich-ucx-gnu12-ohpc	3.4.3	/	M	
	mvapich2-gnu12-ohpc	2.3.7	/	M	
openldap	slapd-ssl-config	1.0.4	slapd	M	/
	nss-pam-ldapd	0.9.9	nslcd	M, C, L	/

Define a shared directory for installer

The following steps describe how to define a shared directory for installer by taking `/install/installer` as an example:

Step 1. Manage node sharing `/install/installer`:

```
dnf install -y nfs-utils

systemctl enable nfs-server --now


share_installer_dir="/install/installer"

mkdir -p $share_installer_dir


echo "/install/installer *(rw,async,no_subtree_check,no_root_squash)" >> /etc/exports

exportfs -a
```

Step 2. Install NFS for cluster nodes:

```
nodeshell all dnf install -y nfs-utils
```

Step 3. Configure the shared directory for cluster nodes:

```
nodeshell all mkdir -p $share_installer_dir

nodeshell all "echo '${sms_ip}:/install/installer /install/installer \

nfs nfsvers=4.0,nodev,nosuid,noatime 0 0' >> /etc/fstab"
```

Step 4. Mount shared directory:

```
nodeshell all mount /install/installer
```

Enable repository for other nodes

Step 1. Distribute `/etc/hosts`:

```
cp /etc/hosts $share_installer_dir

nodeshell all cp $share_installer_dir/hosts /etc/hosts
```

Step 2. Enable httpd services:

```
cat << eof > /etc/httpd/conf.d/installer.conf

Alias /install /install

<Directory /install>

AllowOverride None

Require all granted

Options +Indexes +FollowSymLinks

</Directory>

eof


systemctl restart httpd
```

Note: `/install` is the basic directory for repository which configured in the `lico_env.local` file.

Step 3. Enable repository:

```
nodeshell all dnf clean all
```

Configure the memory for other nodes

Step 1. Run the following commands:

```
cp /etc/security/limits.conf $share_installer_dir

nodeshell all cp $share_installer_dir/limits.conf /etc/security/limits.conf

nodeshell all reboot
```

Step 2. Check and ensure that the installation is completed:

```
nodeshell all uptime
```

Create the local repository for other nodes

Create the local repository for other nodes:

```
cp /etc/yum.repos.d/EL8-OS.repo $share_installer_dir


sed -i '/^baseurl=/d' $share_installer_dir/EL8-OS.repo

sed -i "/name=appstream/a\baseurl=http://${sms_name}${os_repo_dir}/AppStream/" \

$share_installer_dir/EL8-OS.repo

sed -i "/name=baseos/a\baseurl=http://${sms_name}${os_repo_dir}/BaseOS/" \

$share_installer_dir/EL8-OS.repo
```

```
nodeshell all cp $share_installer_dir/EL8-OS.repo /etc/yum.repos.d/
```

Configure Lenovo OpenHPC repositories

Step 1. Download the following package: https://hpc.lenovo.com/lico/downloads/7.1/Lenovo-OpenHPC-2.6.1.EL8.x86_64.tar

Step 2. Upload the package to the `/root` directory.

Step 3. Configure the local Lenovo OpenHPC repository:

```
mkdir -p $ohpc_repo_dir

cd /root

tar xvf Lenovo-OpenHPC-2.6.1.EL8.x86_64.tar -C $ohpc_repo_dir

rm -rf $link_ohpc_repo_dir

ln -s $ohpc_repo_dir $link_ohpc_repo_dir

$link_ohpc_repo_dir/make_repo.sh
```

Step 4. Configure the repository for login and compute nodes:

```
cp /etc/yum.repos.d/Lenovo.OpenHPC.local.repo $share_installer_dir

sed -i '/^baseurl=/d' $share_installer_dir/Lenovo.OpenHPC.local.repo

sed -i '/^gpgkey=/d' $share_installer_dir/Lenovo.OpenHPC.local.repo


echo "baseurl=http://${sms_name}${link_ohpc_repo_dir}/EL_8" \

>> $share_installer_dir/Lenovo.OpenHPC.local.repo


echo "gpgkey=http://${sms_name}${link_ohpc_repo_dir}/EL_8\

/repodata/repomd.xml.key" >> $share_installer_dir/Lenovo.OpenHPC.local.repo
```

Step 5. Distribute files for login and compute nodes:

```
nodeshell all cp $share_installer_dir/Lenovo.OpenHPC.local.repo \

/etc/yum.repos.d/

nodeshell all "echo -e %_excludedocs 1 >> ~/.rpmmacros"
```

Note: Sufficient packages should be installed on the operating system; otherwise, the subsequent installation steps might fail.

Configure the LiCO dependencies repositories

Step 1. Download the following package: https://hpc.lenovo.com/lico/downloads/7.1/lico-dep-7.1.0.el8.x86_64.tgz

Step 2. Upload the package to the `/root directory`.

Step 3. Configure the repository for the management node:

```
mkdir -p $lico_dep_repo_dir

cd /root
```

```
tar -xvf lico-dep-7.1.0.el8.x86_64.tgz -C $lico_dep_repo_dir

rm -rf $link_lico_dep_repo_dir

ln -s $lico_dep_repo_dir $link_lico_dep_repo_dir

$link_lico_dep_repo_dir/mklocalrepo.sh
```

Attention: Before running the commands, ensure that the management node has configured a local operating system repository for the above and the subsequent actions.

Step 4. (Optional) If the cluster already exists, check to ensure that the version is consistent with "Install the LiCO dependencies" on page 41.

Step 5. Configure the repository for other nodes:

```
cp /etc/yum.repos.d/lico-dep.repo $share_installer_dir

sed -i '/^baseurl=/d' $share_installer_dir/lico-dep.repo

sed -i '/^gpgkey=/d' $share_installer_dir/lico-dep.repo


sed -i "/name=lico-dep-local-library/a\baseurl=http://${sms_name}\

${link_lico_dep_repo_dir}/library/" $share_installer_dir/lico-dep.repo


sed -i "/name=lico-dep-local-library/a\gpgkey=http://${sms_name}\

${link_lico_dep_repo_dir}/RPM-GPG-KEY-LICO-DEP-EL8" $share_installer_dir/lico-dep.repo


sed -i "/name=lico-dep-local-standalone/a\baseurl=http://${sms_name}\

${link_lico_dep_repo_dir}/standalone/" $share_installer_dir/lico-dep.repo


sed -i "/name=lico-dep-local-standalone/a\gpgkey=http://${sms_name}\

${link_lico_dep_repo_dir}/RPM-GPG-KEY-LICO-DEP-EL8" $share_installer_dir/lico-dep.repo


nodeshell all cp $share_installer_dir/lico-dep.repo /etc/yum.repos.d
```

Obtain the LiCO installation package

Step 1. Obtain the LiCO 7.1.0 release package for EL8 `lico-release-7.1.0.el8.x86_64.tar.gz` and the LiCO license file from:
https://commercial.lenovo.com/cn/en/signin

Step 2. Obtain the LiCO monitor package `openlico-monitor-1.0.0.x86_64.tgz` from:
https://commercial.lenovo.com/cn/en/signin

Step 3. Upload the LiCO release and monitor packages to the management node.

Configure the local repository for LiCO

Step 1. Configure the local repository for the management node:

```
mkdir -p $lico_repo_dir
```

```
tar zxvf lico-release-7.1.0.el8.x86_64.tar.gz -C $lico_repo_dir --strip-components 1
rm -rf $link_lico_repo_dir
ln -s $lico_repo_dir $link_lico_repo_dir
$link_lico_repo_dir/mklocalrepo.sh
```

Step 2. Configure the local yum repository for the other nodes:

```
cp /etc/yum.repos.d/lico-release.repo $share_installer_dir
sed -i '/baseurl=/d' $share_installer_dir/lico-release.repo

sed -i "/name=lico-release/a\baseurl=http://${sms_name}\
${link_lico_repo_dir}" $share_installer_dir/lico-release.repo
```

Step 3. Distribute repo files:

```
nodeshell login cp $share_installer_dir/lico-release.repo /etc/yum.repos.d/
```

Configure the confluent local repository

Step 1. Configure the local repository for the other nodes:

```
cp /etc/yum.repos.d/lenovo-hpc.repo $share_installer_dir
sed -i '/^baseurl=/d' $share_installer_dir/lenovo-hpc.repo
sed -i '/^gpgkey=/d' $share_installer_dir/lenovo-hpc.repo

echo "baseurl=http://${sms_name}${confluent_repo_dir}/lenovo-hpc-el8" \
>> $share_installer_dir/lenovo-hpc.repo

echo "gpgkey=http://${sms_name}${confluent_repo_dir}/lenovo-hpc-el8\
/lenovohpckey.pub" >> $share_installer_dir/lenovo-hpc.repo
```

Step 2. Distribute the repo files:

```
nodeshell all cp $share_installer_dir/lenovo-hpc.repo /etc/yum.repos.d/
```

Install Slurm

Step 1. Install the base package:

```
dnf install -y lenovo-ohpc-base
```

Step 2. Install Slurm:

```
dnf install -y ohpc-slurm-server hwloc-ohpc
```

Step 3. Install the Slurm client:

```
nodeshell compute dnf install -y ohpc-base-compute ohpc-slurm-client lmod-ohpc
```

Step 4. (Optional) To save the previous job information and use memory accounting function, install and configure slurm accounting function referring to: https://slurm.schedmd.com/accounting.html

Configure NFS

Configure user shared directory

The following steps describes how to create the user shared directory by taking `/home` as an example.

Step 1. Manage node sharing `/home`:

```
echo "/home *(rw,async,no_subtree_check,no_root_squash)" >> /etc/exports
```

```
exportfs -a
```

Step 2. Unmount the mounted `/home`:

```
nodeshell all "sed -i '/ \/home /d' /etc/fstab"
```

```
nodeshell all umount /home
```

Step 3. Configure the shared directory for cluster nodes:

```
nodeshell all "echo '${sms_ip}:/home /home nfs nfsvers=4.0,nodev,nosuid,noatime \
```

```
0 0' >> /etc/fstab"
```

Step 4. Mount the shared directory:

```
nodeshell all mount /home
```

Configure shared directory for OpenHPC

Step 1. Manage node sharing `/opt/ohpc/pub` for OpenHPC:

```
echo "/opt/ohpc/pub *(ro,no_subtree_check)" >> /etc/exports
```

```
exportfs -a
```

Step 2. Configure shared directory for cluster nodes:

```
nodeshell all mkdir -p /opt/ohpc/pub
```

```
nodeshell all "echo '${sms_ip}:/opt/ohpc/pub /opt/ohpc/pub nfs \
```

```
nfsvers=4.0,nodev,noatime 0 0' >> /etc/fstab"
```

Step 3. Mount the shared directory:

```
nodeshell all mount /opt/ohpc/pub
```

Attention: This directory is mandatory. If this directory is shared from the management node and mounted on all other nodes, skip this step.

Configure shared directory for LiCO components

Step 1. Manage node sharing `/opt/lico/pub`:

```
mkdir -p /opt/lico/pub
```

```
touch /opt/lico/pub/DO_NOT_DELETE
```

```
echo "The file is required by lico monitor." >> /opt/lico/pub/DO_NOT_DELETE
```

```
echo "/opt/lico/pub *(ro,sync,no_subtree_check,no_root_squash)" >> /etc/exports
```

```
exportfs -a
```

Step 2. Configure shared directory for cluster nodes:

```
nodeshell all mkdir -p /opt/lico/pub
nodeshell all "echo '${sms_ip}:/opt/lico/pub /opt/lico/pub nfs nfsvers=4.0,nodev,noatime \

0 0' >> /etc/fstab"
```

Step 3. Mount the shared directory:

```
nodeshell all mount /opt/lico/pub
```

Configure Chrony

Note: If the chrony service has been configured for nodes in the cluster, skip this section.

Step 1. Install Chrony:

```
dnf install -y chrony
```

Step 2. Unsynchronized cluster time might cause unexpected problems. Configure chronyd service referring to:
https://chrony.tuxfamily.org/documentation.html

GPU driver installation

The GPU driver should be installed on each GPU compute node. If only a subset of nodes is installed with GPUs, replace the **compute** argument in `nodeshell` commands with the corresponding node range of GPU nodes.

Attention: Intel GPU is supported in LiCO 7.1.0 and higher versions. To use Intel GPU, skip the steps mentioned in this chapter and install Intel GPU Driver following https://hpc.lenovo.com/lico/downloads/7.1/how-to-install-intel-gpu.html.

Disable the nouveau drivers

To install the Display Driver, disable the Nouveau drivers first.

Step 1. Configure the operating system to start on the text console and then restart the system:

Note: This step is required only when the operating system is configured to start on a graphical desktop.

```
nodeshell compute systemctl set-default multi-user.target
```

Step 2. Add the configuration file:

```
cat << eof > $share_installer_dir/blacklist-nouveau.conf

blacklist nouveau

options nouveau modeset=0

eof
```

Step 3. Distribute the configuration file:

```
nodeshell compute cp $share_installer_dir/blacklist-nouveau.conf \

/usr/lib/modprobe.d/blacklist-nouveau.conf
```

Step 4. Regenerate the kernel initramfs:

```
nodeshell compute dracut --force
```

Step 5. Make the configuration take effect:

```
nodeshell compute reboot
```

Install the GPU driver

Step 1. Download the NVIDIA driver from https://us.download.nvidia.com/tesla/520.61.07/NVIDIA-Linux-x86_64-520.61.07.run, and copy it to the shared directory `$share_installer_dir`.

Step 2. Run the following commands:

```
dnf install -y tar bzip2 make automake gcc gcc-c++ pciutils \

elfutils-libelf-devel libglvnd-devel


dnf install -y kernel-devel-$(uname -r) kernel-headers-$(uname -r)


chmod +x $share_installer_dir/NVIDIA-Linux-x86_64-520.61.07.run


$share_installer_dir/NVIDIA-Linux-x86_64-520.61.07.run --add-this-kernel -s

nodeshell compute $share_installer_dir/NVIDIA-Linux-x86_64-520.61.07-custom.run -s
```

Step 3. Run the following command on the GPU nodes to determine if GPU can be identified:

```
nodeshell compute nvidia-smi
```

Note: If the GPU information cannot be identified by running the command, restart all GPU nodes. Then re-run the command.

```
nodeshell compute reboot
```

Configure automatic start for the GPU driver

Step 1. Add the configuration file:

```
cat << eof > $share_installer_dir/nvidia-persistenced.service

[Unit]

Description=NVIDIA Persistence Daemon

After=syslog.target

[Service]

Type=forking

PIDFile=/var/run/nvidia-persistenced/nvidia-persistenced.pid

Restart=always

ExecStart=/usr/bin/nvidia-persistenced --verbose

ExecStopPost=/bin/rm -rf /var/run/nvidia-persistenced/*

TimeoutSec=300

[Install]

WantedBy=multi-user.target

eof


cat << eof > $share_installer_dir/nvidia-modprobe-loader.service
```

```
[Unit]

Description=NVIDIA ModProbe Service

After=syslog.target

Before=slurmd.service

[Service]

Type=oneshot

ExecStart=/usr/bin/nvidia-modprobe -u -c=0

RemainAfterExit=yes

[Install]

WantedBy=multi-user.target

eof
```

Step 2. Distribute the configuration file:

```
nodeshell compute cp $share_installer_dir/nvidia-persistenced.service \

/usr/lib/systemd/system/nvidia-persistenced.service


nodeshell compute cp $share_installer_dir/nvidia-modprobe-loader.service \

/usr/lib/systemd/system/nvidia-modprobe-loader.service


nodeshell compute mkdir -p /var/run/nvidia-persistenced
```

Step 3. Restart service:

```
nodeshell compute systemctl daemon-reload

nodeshell compute systemctl enable nvidia-persistenced --now

nodeshell compute systemctl enable nvidia-modprobe-loader.service --now
```

(Optional) Configure automatic start for the GPU MIG

GPU MIG instances will be destroyed after restarting the compute node. In this case, users can automatically create GPU MIG instances through the following configuration.

Note: Skip this section if GPU MIG function is not used.

Step 1. Add the script file:

Note: For creating MIG for other specifications, run `nvidia-smi mig -h` and refer to the results.

Following is an example for creating seven GPU MIG instances of type 1g.5gb (each GPU MIG instance contains compute instance of type 1g.5gb):

```
cat << eof > $share_installer_dir/nvidia-mig-create.sh

#!/bin/bash

set -e

nvidia-smi -mig 1
```

```
nvidia-smi mig -cgi 19,19,19,19,19,19,19 -C
eof
chmod a+x $share_installer_dir/nvidia-mig-create.sh
```

Step 2. Add the configuration file:

```
cat << eof > $share_installer_dir/nvidia-mig-persistenced.service
[Unit]
Description=NVIDIA MIG Create
After=nvidia-persistenced.service
[Service]
Type=oneshot
Restart=never
ExecStart=/usr/bin/nvidia-mig-create.sh
TimeoutSec=300
RemainAfterExit=yes
User=root
[Install]
WantedBy=multi-user.target
eof
```

Step 3. Distribute the script and configuration file:

```
nodeshell compute cp $share_installer_dir/nvidia-mig-create.sh \
/usr/bin/nvidia-mig-create.sh
nodeshell compute cp $share_installer_dir/nvidia-mig-persistenced.service \
/usr/lib/systemd/system/nvidia-mig-persistenced.service
```

Step 4. Restart service:

```
nodeshell compute systemctl daemon-reload
nodeshell compute systemctl enable nvidia-mig-persistenced --now
```

Configure Slurm

Step 1. Download `slurm.conf` from the following web site: https://hpc.lenovo.com/lico/downloads/7.1/examples/conf/

Step 2. Upload `slurm.conf`to `$share_installer_dir`, and modify this file according to the instructions in "slurm.conf" on page 26.

Step 3. Download `cgroup.conf` from the following web site: https://hpc.lenovo.com/lico/downloads/7.1/examples/conf/

Step 4. Upload `cgroup.conf` to `$share_installer_dir`.

Step 5. Distribute the configuration:

```
cp $share_installer_dir/slurm.conf /etc/slurm/slurm.conf
```

```
nodeshell compute cp $share_installer_dir/slurm.conf /etc/slurm/slurm.conf

cp $share_installer_dir/cgroup.conf /etc/slurm/cgroup.conf

nodeshell compute cp $share_installer_dir/cgroup.conf /etc/slurm/cgroup.conf

cp /etc/munge/munge.key $share_installer_dir

nodeshell compute cp $share_installer_dir/munge.key /etc/munge/munge.key

mkdir -p /var/spool/slurm/ctld

nodeshell compute mkdir -p /var/spool/slurm/d
```

Step 6. (Optional) For GPU nodes only:

- If Nvidia MIG is enabled in the GPU node, configure GPU nodes. For more information, refer to: https://gitlab.com/nvidia/hpc/slurm-mig-discovery
- If Intel GPU is used in the GPU node, configure GPU nodes. download the `gres.conf` file from https://hpc.lenovo.com/lico/downloads/7.1/examples/conf/intel/gres.conf and upload it to `/etc/slurm` on the GPU node.
- If Nvidia MIG is disabled or not supported in the GPU node, download the sample file `gres.conf` from https://hpc.lenovo.com/lico/downloads/7.1/examples/conf/, edit the sample file, and upload it to `/etc/slurm` on the GPU node. For more information, refer to "gres.conf" on page 27.

Step 7. Start service:

```
systemctl enable munge

systemctl enable slurmctld

systemctl restart munge

systemctl restart slurmctld
```

Step 8. Start other node service:

```
nodeshell compute systemctl enable munge

nodeshell compute systemctl restart munge

nodeshell compute systemctl enable slurmd

nodeshell compute systemctl restart slurmd
```

slurm.conf

The following typical fields need to be configured:

- Cluster name:

```
ClusterName=mycluster
```

- Management node name:

```
SlurmctldHost=c031
```

- GPU scheduling:

```
GresTypes=gpu
```

Note: In the cluster, this entry is used when a GPU node is included. If the cluster includes no GPU node, delete this entry.

- Cluster node definitions:

```
NodeName=c031 Gres=gpu:4 CPUs=28 RealMemory=200000 State=UNKNOWN

NodeName=c032 Gres=gpu:4 CPUs=28 RealMemory=200000 State=UNKNOWN
```

  - **Gres**: Number of GPUs
  - **CPUs**: Number of CPUs on a node.

  - **RealMemory**: Memory size of a node (Unit: M).
- Partition definitions:

  ```
  PartitionName=compute Nodes=c0[31-32] Default=YES MaxTime=INFINITE State=UP
  ```

  ```
  PartitionName=compute1 Nodes=c0[31-32] Default=NO MaxTime=INFINITE State=UP
  ```

  Notes:
  - **Default**: identifies whether this partition is the default partition. To submit a job, select a partition. If the partition is not selected, the default partition is used.
  - **Nodes**: the NodeName list. If NodeName is irregular, Nodes=[nodename1,nodename2,...] is allowed.
- Enforced part limit definitions:

  ```
  EnforcePartLimits=ALL
  ```

  Attention: Use this configuration for submitting a direct error response when a job requests resources that exceed the cluster resource amount. Otherwise, the job remains in the queue.

For more details about how to configure `slurm.conf`, refer to the official Slurm site: https://slurm.schedmd.com/slurm.conf.html

gres.conf

This configuration file describes the GPUs installed on the GPU nodes and the GPU memory. The content of this file may vary based on the GPU node.

Modify the following content:

```
Name=gpu File=/dev/nvidia[0-3]
```

Note: In `/dev/nvidia[0-3]`, `[0–3]` should be changed to the actual GPU configuration. For example, `/dev/nvidia0` means one GPU card, whereas `/dev/nvidia[0-1]` means two GPU cards.

Enable Slurm Accounting

Step 1. Install MariaDB:

```
dnf install -y mariadb-server mariadb-devel
```

Step 2. Start the MariaDB service:

```
systemctl enable mariadb --now
```

Step 3. Configure MariaDB for slurmdbd:

```
mysql

create database slurm_acct_db;

create user '<SLURMDBD_USERNAME>'@'%' identified by '<SLURMDBD_PASSWORD>';

grant ALL on slurm_acct_db.* to '<SLURMDBD_USERNAME>'@'%';

create user '<SLURMDBD_USERNAME>'@'localhost' identified by '<SLURMDBD_PASSWORD>';

grant ALL on slurm_acct_db.* to '<SLURMDBD_USERNAME>'@'localhost';

exit
```

Step 4. Download `slurmdbd.conf` from: https://hpc.lenovo.com/lico/downloads/7.1/examples/conf/

Step 5. Upload `slurmdbd.conf` to `/etc/slurm/`, and modify the slurmdbd configuration file according to the configuration commands in Step 3.

Note: LiCO configures `slurmdbd.conf` based on the slurmdbd installed on the LiCO management node. To configure `slurmdbd.conf` for other purposes, following the instructions in: https://slurm.schedmd.com/slurmdbd.conf.html

Step 6. Modify the permission for the slurmdbd configuration file and enable the slurmdbd service:

```
mkdir -p /var/log/slurm/

chmod 600 /etc/slurm/slurmdbd.conf

systemctl enable slurmdbd --now

systemctl status slurmdbd
```

Step 7. Modify the file `/etc/slurm/slurm.conf` for slurmdbd.

Attention:

- Add or modify the following commands in the file `/etc/slurm/slurm.conf` on the LiCO management node.
- Replace the variables in angle brackets with the actual values.

```
# ACCOUNTING

AccountingStorageTRES=cpu,mem,energy,node,billing,fs/disk,vmem,pages,gres/gpu

AccountingStorageHost=<SLURMCTLDHOST> # same with the SlurmctldHost

# AccountingStoragePort=6819

AccountingStorageType=accounting_storage/slurmdbd
```

Step 8. (Optional) Modify the file `/etc/slurm/slurm.conf` for slurm QOS.

Attention: Do not modify the file if the slurm QOS function is not used. Otherwise, the job might failed to be submitted.

```
AccountingStorageEnforce=associations,limits,qos
```

Step 9. Distribute the configuration:

```
\cp /etc/slurm/slurm.conf $share_installer_dir

nodeshell compute cp $share_installer_dir/slurm.conf /etc/slurm/slurm.conf
```

Step 10. Restart the service:

```
nodeshell compute systemctl restart munge

nodeshell compute systemctl restart slurmd


systemctl restart munge

systemctl restart slurmctld

systemctl restart slurmdbd
```

Step 11. create slurm sacct:

Note: Replace the variables in angle brackets with the actual values.

```
sacctmgr add cluster <SLURM_CLUSTER_NAME>

sacctmgr list cluster
```

(Optional) Install Icinga2

Note: If LiCO is not used for cluster monitoring, skip this section. If IB device is prepared and IB driver installation is required, install IB driver in the operating system referring to LeSI 22B_SI best recipe before installing Icinga2. USB network card influences IB network card invoked by MPI. Therefore, it is recommended to add “rmmod cdc_ether” in power on procedure to remove USB network card.

Step 1. Install InfluxDB:

```
dnf install -y influxdb

systemctl enable influxdb --now
```

Step 2. Create InfluxDB users for icinga2:

```
# To enter the InfluxDB shell:

influx

# To create database:

create database icinga

# To use database:

use icinga

# To create retention policy

create retention policy "three_hour_only" on "icinga" duration 3h replication 1 default

# To create an administrator user, ensure that the password is a string:

create user <INFLUX_USERNAME> with password '<INFLUX_PASSWORD>' with all privileges

# To exit the influxDB shell:

exit

# To do configuration:

sed -i '/# index-version = "inmem"/a\ index-version = "tsi1"' /etc/influxdb/config.toml

# To restart InfluxDB:

systemctl restart influxdb
```

Step 3. Install icinga2:

```
dnf install -y icinga2

nodeshell all dnf install -y icinga2
```

Step 4. Install LiCO icinga2 plugin:

```
dnf install -y nagios-plugins-ping python3-virtualenv


mkdir -p $lico_monitor_dir

tar -xvzf openlico-monitor-1.0.0.x86_64.tgz -C $lico_monitor_dir

rm -rf $link_lico_monitor_dir

ln -s $lico_monitor_dir $link_lico_monitor_dir
```

```
python3 $link_lico_monitor_dir/install.py
```

Step 5. Open API function:

```
icinga2 api setup
```

Step 6. Configure the icinga2:

```
icinga2 node setup --master --disable-confd

echo -e "LANG=en_US.UTF-8" >> /etc/sysconfig/icinga2

systemctl restart icinga2
```

Step 7. Configure icinga2 agent for other nodes:

```
nodeshell all icinga2 pki save-cert --trustedcert \

/var/lib/icinga2/certs/trusted-parent.crt --host ${sms_name}


for ((i=0;i<$num_computes;i++));do

ticket=`icinga2 pki ticket --cn ${c_name[${i}]}`

nodeshell ${c_name[${i}]} icinga2 node setup --ticket ${ticket} --cn ${c_name[${i}]} \

--endpoint ${sms_name} --zone ${c_name[${i}]} --parent_zone master --parent_host \

${sms_name} --trustedcert /var/lib/icinga2/certs/trusted-parent.crt \

--accept-commands --accept-config --disable-confd

done


for ((i=0;i<$num_logins;i++));do

ticket=`icinga2 pki ticket --cn ${l_name[${i}]}`

nodeshell ${l_name[${i}]} icinga2 node setup --ticket ${ticket} --cn ${l_name[${i}]} \

--endpoint ${sms_name} --zone ${l_name[${i}]} --parent_zone master --parent_host \

${sms_name} --trustedcert /var/lib/icinga2/certs/trusted-parent.crt \

--accept-commands --accept-config --disable-confd

done


nodeshell all "echo -e 'LANG=en_US.UTF-8' >> /etc/sysconfig/icinga2"

nodeshell all systemctl restart icinga2
```

Step 8. Configure global template file on the management node:

a. Create a folder:

```
mkdir -p /etc/icinga2/zones.d/global-templates
```

b. Download `commands.conf` from https://hpc.lenovo.com/lico/downloads/7.1/examples/conf/icinga2/commands/slurm/ and upload it to `/etc/icinga2/zones.d/global-templates`.

c. Modify the directory privilege:

```
chown -R icinga:icinga /etc/icinga2/zones.d/global-templates
```

Step 9. Define the zone file:

a. Define the zone file:

```
mkdir -p /etc/icinga2/zones.d/master


echo -e "object Host \"${sms_name}\" {\n check_command = \"hostalive\"\n \
address = \"${sms_ip}\"\n vars.agent_endpoint = name\n}\n" >> \
/etc/icinga2/zones.d/master/hosts.conf


for ((i=0;i<$num_computes;i++));do
echo -e "object Endpoint \"${c_name[${i}]}\" {\n host = \"${c_name[${i}]}\"\n \
port = \"${icinga_api_port}\"\n log_duration = 0\n}\nobject \
Zone \"${c_name[${i}]}\" {\n endpoints = [ \"${c_name[${i}]}\" ]\n \
parent = \"master\"\n}\n" >> /etc/icinga2/zones.d/master/agent.conf
if [[ ${gpu_vendor} = "" ]];then
echo -e "object Host \"${c_name[${i}]}\" {\n check_command = \"hostalive\"\n \
address = \"${c_ip[${i}]}\"\n vars.agent_endpoint = name\n}\n" >> \
/etc/icinga2/zones.d/master/hosts.conf
else
echo -e "object Host \"${c_name[${i}]}\" {\n check_command = \"hostalive\"\n \
address = \"${c_ip[${i}]}\"\n vars.agent_endpoint = name\n vars.gpu.vendor = \
\"${gpu_vendor}\"\n}\n" >> /etc/icinga2/zones.d/master/hosts.conf
fi
done


for ((i=0;i<$num_logins;i++));do
echo -e "object Endpoint \"${l_name[${i}]}\" {\n host = \"${l_name[${i}]}\"\n \
port = \"${icinga_api_port}\"\n log_duration = 0\n}\nobject \
Zone \"${l_name[${i}]}\" {\n endpoints = [ \"${l_name[${i}]}\" ]\n \
parent = \"master\"\n}\n" >> /etc/icinga2/zones.d/master/agent.conf
echo -e "object Host \"${l_name[${i}]}\" {\n check_command = \"hostalive\"\n \
address = \"${l_ip[${i}]}\"\n vars.agent_endpoint = name\n}\n" >> \
/etc/icinga2/zones.d/master/hosts.conf
done
```

b. Download `service.conf` from https://hpc.lenovo.com/lico/downloads/7.1/examples/conf/icinga2/ and upload it to `/etc/icinga2/zones.d/master`.

c. Modify the directory privilege:

```
chown -R icinga:icinga /etc/icinga2/zones.d/master

systemctl restart icinga2
```

Step 10. Enable influxdb writer feature for icinga.

a. Enable influxdb writer feature for icinga:

```
icinga2 feature enable influxdb
```

b. Download `influxdb.conf` from https://hpc.lenovo.com/lico/downloads/7.1/examples/conf/icinga2/ , upload `influxdb.conf` to `/etc/icinga2/features-available/`, and modify the influxdb writer configuration file referring to Step 2 on page 29.

```
host = "<influxdb server address>"

port = 8086

database = "icinga"

username = "<username>"

password = "<password>"
```

c. Restart the icinga2 service to take effect.

```
systemctl restart icinga2
```

Step 11. Enable service:

```
nodeshell all modprobe ipmi_devintf

nodeshell all systemctl enable icinga2


modprobe ipmi_devintf

systemctl enable icinga2
```

Step 12. (Optional) Check the configuration:

```
icinga2 daemon -C
```

Install MPI

Step 1. Install three modules (OpenMPI, MPICH, and MVAPICH) to the system:

```
dnf install -y openmpi4-gnu12-ohpc mpich-ucx-gnu12-ohpc mvapich2-gnu12-ohpc ucx-ib-ohpc \

ucx-cma-ohpc ucx-rdmacm-ohpc
```

Step 2. Enable gnu12:

```
ln -sT gnu12 /opt/ohpc/pub/modulefiles/gnu
```

Step 3. (Optional) To call the RDMA protocol through the IB network card, create the configuration file `/opt/ohpc/pub/mpi/ucx-ohpc/<UCX-VERSION>/etc/ucx/ucx.conf` and replace <*UXC-VERSION*> with the existing ucx-ohpc version:

```
mkdir -p /opt/ohpc/pub/mpi/ucx-ohpc/<UCX-VERSION>/etc/ucx/

touch /opt/ohpc/pub/mpi/ucx-ohpc/<UCX-VERSION>/etc/ucx/ucx.conf
```

Then add the following content to the configuration file:

```
UCX_NET_DEVICES=<ibdev>
```

Following is the example for the results of running `ibdev2netdev`, which show the association information between Ethernet devices and IB devices:

```
i40iw0 port 1 ==eno1 (Up)

i40iw1 port 1 ==eno2 (Down)

mlx5_0 port 1 ==ib0 (Up)

mlx5_0 port 1 ==ib0 (Up)
```

Finally modify the corresponding configuration content to the following:

```
UCX_NET_DEVICES=mlx5_*
```

Step 4. Set the default module. Do one of the following:

- Set OpenMPI module as the default:

  ```
  dnf install -y lmod-defaults-gnu12-openmpi4-ohpc
  ```

  (Optional) To set the openmpi program to call the IB network card by default, edit the configuration file `/opt/ohpc/pub/mpi/openmpi4-gnu12/<OPENMPI_VERSION>/etc/openmpi-mca-params.conf`, replace `<OPENMPI_VERSION>` with the existing OpenMPI version, and add the following commands:

  ```
  pml = ucx

  pml_ucx_devices = <ibdev>
  ```

  The following example shows how to replace `<ibdev>`:

  Following are the results of running `ibdev2netdev`, which show the association information between Ethernet devices and IB devices:

  ```
  i40iw0 port 1 ==eno1 (Up)

  i40iw1 port 1 ==eno2 (Down)

  mlx5_0 port 1 ==ib0 (Up)

  mlx5_0 port 1 ==ib0 (Up)
  ```

  And the corresponding configuration commands should be modified to:

  ```
  pml = ucx

  pml_ucx_devices = mlx5_*
  ```

- Set the MPICH module as the default:

  ```
  dnf install -y lmod-defaults-gnu12-mpich-ucx-ohpc
  ```

- Set the MVAPICH module as the default:

  ```
  dnf install -y lmod-defaults-gnu12-mvapich2-ohpc
  ```

  Note: MVAPICH2 requires that Infiniband is present and working correctly.

Install Singularity

Singularity is an HPC-facing lightweight container framework.

Step 1. Install Singularity:

```
dnf install -y singularity-ohpc
```

Step 2. Edit the file /opt/ohpc/pub/modulefiles/ohpc by adding the following content to the end of the module try-add block:

```
module try-add singularity
```

Step 3. In the module del block, add the following content as the first line:

```
module del singularity
```

Step 4. Run the following command:

```
source /etc/profile.d/lmod.sh
```

Changes to /opt/ohpc/pub/modulefiles/ohpc may be lost when the default modules are changed with the installation of the lmod-defaults* package. In that case, either modify /opt/ohpc/pub/modulefiles/ohpc again, or add module try-add singularity to the end of /etc/profile.d/lmod.sh.

Install OpenLDAP

Note: If OpenLDAP is configured or other authentication services are used in the cluster, skip this section.

OpenLDAP is the open-source version of the lightweight directory access protocol. It is recommended to use OpenLDAP to manage users. However, LiCO also supports other authentication services that compatible with Linux-PAM.

Step 1. Install OpenLDAP:

```
dnf install -y slapd-ssl-config openldap-servers
```

Step 2. Modify the configuration file:

```
sed -i "s/dc=hpc,dc=com/${lico_ldap_domain_name}/" /usr/share/openldap-servers/lico.ldif

sed -i "/dc:/s/hpc/${lico_ldap_domain_component}/" /usr/share/openldap-servers/lico.ldif

sed -i "s/dc=hpc,dc=com/${lico_ldap_domain_name}/" /etc/openldap/slapd.conf

slapadd -v -l /usr/share/openldap-servers/lico.ldif -f /etc/openldap/slapd.conf -b \

${lico_ldap_domain_name}
```

Step 3. Obtain the OpenLDAP key:

```
slappasswd
```

Step 4. Edit /etc/openldap/slapd.conf to set the root password to the key that was obtained.

```
rootpw <ENCRYPT_LDAP_PASSWORD>
```

Step 5. Change the owner of the configuration file:

```
chown -R ldap:ldap /var/lib/ldap

chown ldap:ldap /etc/openldap/slapd.conf
```

Step 6. Start the OpenLDAP service:

```
systemctl enable slapd --now
```

Step 7. Verify that the service has been started:

```
systemctl status slapd
```

Install OpenLDAP-client

Install OpenLDAP-client:

```
echo "TLS_REQCERT never" >> /etc/openldap/ldap.conf
```

```
cp /etc/openldap/ldap.conf $share_installer_dir
```

```
nodeshell all cp $share_installer_dir/ldap.conf /etc/openldap/ldap.conf
```

Install nss-pam-ldapd

nss-pam-ldapd is a name service switching module and pluggable authentication module. LiCO uses this module for user authentication.

Step 1. Install nss-pam-ldapd:

```
dnf install -y nss-pam-ldapd
```

Step 2. Install nss-pam-ldapd on the other node:

```
nodeshell all dnf install -y nss-pam-ldapd
```

Step 3. Download `nslcd.conf` from: https://hpc.lenovo.com/lico/downloads/7.1/examples/conf/

Step 4. Upload the file to `$share_installer_dir`. Use the instructions in the file to modify the configuration.

Step 5. Distribute the configuration:

```
cp $share_installer_dir/nslcd.conf /etc/nslcd.conf
```

```
nodeshell all cp $share_installer_dir/nslcd.conf /etc/nslcd.conf
```

```
chmod 600 /etc/nslcd.conf
```

```
nodeshell all chmod 600 /etc/nslcd.conf
```

Step 6. Start the nslcd service:

```
systemctl enable nslcd --now
```

```
nodeshell all systemctl enable nslcd --now
```

Configure authselect-nslcd-config

Step 1. Create the path for the configuration file:

```
mkdir -p /usr/share/authselect/vendor/nslcd
```

```
nodeshell all mkdir -p /usr/share/authselect/vendor/nslcd
```

Step 2. Download configuration files from: https://hpc.lenovo.com/lico/downloads/7.1/examples/conf/authselect/authselect.tar.gz

Step 3. Upload the configuration files to `$share_installer_dir`.Then deploy the configuration files:

Step 4. Distribute the configuration:

```
tar -xzvf $share_installer_dir/authselect.tar.gz -C /usr/share/authselect/vendor/nslcd/
```

```
nodeshell all tar -xzvf $share_installer_dir/authselect.tar.gz -C \

/usr/share/authselect/vendor/nslcd/
```

Step 5. Enable the configuration:

```
authselect select nslcd with-mkhomedir --force
```

```
nodeshell all authselect select nslcd with-mkhomedir --force
```

Configure non-root login to compute nodes

Attention:

- To allow the non-root login to the compute nodes regardless of whether a Slurm job is running on these nodes, skip this section.
- The following commands are applicable for the scenario that the non-root login is allowed to the compute nodes when the Slurm job is running on the compute nodes under a particular username. In this case, the non-root ssh login only works on this particular username.

```
nodeshell compute "echo 'account required pam_slurm_adopt.so \

action_adopt_failure=deny action_generic_failure=deny' >> /etc/pam.d/sshd"

nodeshell compute authselect select nslcd without-systemd with-mkhomedir --force
```

Checkpoint B

Step 1. Verify if Slurm is properly installed:

```
sinfo
```

Notes:

- The output should be as follows:

```
PARTITION AVAIL TIMELIMIT NODES STATE NODELIST

normal* up 1-00:00:00 2 idle c[1-2]

……
```

- The status of all nodes should be **idle**; **idle*** is unacceptable. If the status is not **idle**, identify the causes by checking the logs in `/var/log/slurmctld.log` of management node and the logs in `/var/log/slurmd.log` of status error nodes.

Step 2. Add a **test** account:

```
useradd test -m --uid 65530

nodeshell all useradd test --uid 65530
```

Step 3. Log in to a compute node by using the test account and the test program disrubuted by Slurm:

```
su - test

mpicc -O3 /opt/ohpc/pub/examples/mpi/hello.c

srun -n 8 -N 1 -w <NODENAME> -p <PARTITIONNAME> --pty /bin/bash

prun ./a.out
```

Note: The output should be as follows:

```
Master compute host = c1

Resource manager = slurm

Launch cmd = mpiexec.hydra -bootstrap slurm ./a.out

Hello, world (8 procs total)

--> Process # 0 of 8 is alive. -> c1

--> Process # 4 of 8 is alive. -> c2

--> Process # 1 of 8 is alive. -> c1

--> Process # 5 of 8 is alive. -> c2
--> Process # 2 of 8 is alive. -> c1

--> Process # 6 of 8 is alive. -> c2

--> Process # 3 of 8 is alive. -> c1

--> Process # 7 of 8 is alive. -> c2
```

Step 4. End the test:

```
exit
```

Note: To leave from “test” user session, input “exit” again.

Step 5. Remove the test user:

```
nodeshell all userdel test

userdel test -r
```

After the command is completed, the account will be switched to the root user of the management node.

Chapter 3. Install the LiCO dependencies

Cluster check

If the steps in Chapter 2 "Deploy the cluster environment" on page 7 are skipped, follow this section to make sure that the cluster environment is ready. Otherwise, proceed to "Install the LiCO dependencies" on page 41.

Check environment variables

Check the environment variables *${sms_name}*, *${lico_ldap_domain_name}*, and *${lico_repo_dir}*:

```
echo $sms_name;echo $lico_repo_dir;echo $lico_ldap_domain_name
```

Notes:

- The output should be as follows:

  ```
  head
  ```

  ```
  /install/custom/lico-7.1.0
  ```

  ```
  dc=hpc,dc=com
  ```

- If there is no output, refer to "Configure environment variables" on page 7.

Check the shared directory for installer

Check the shared directory `$share_installer_dir`:

```
echo $share_installer_dir
```

Notes:

- The output should be as follows:

  ```
  /install/installer
  ```

- If there is no output, refer to "Define a shared directory for installer" on page 16.

Check the LiCO dependencies repository

Check the LiCO dependencies repository:

```
dnf repolist | grep lico-dep-local
```

Notes:

- The output should be as follows:

  ```
  lico-dep-local-library lico-dep-local-library
  ```

  ```
  lico-dep-local-standalone lico-dep-local-standalone
  ```

- If there is no output, see "Configure the LiCO dependencies repositories" on page 18.

Check the LiCO repository

Check the LiCO repository:

```
dnf repolist | grep lico-release
```

Notes:

- The output should be as follows:

  ```
  lico-release lico-release
  ```

- If there is no output, refer to "Configure the local repository for LiCO" on page 19.

Check the operating system installation

Go to "Checkpoint A" on page 15 to check the operating system installation for the cluster. If the operating system installation check fails, reconfigure the cluster operating system referring to "Deploy the operating system on other nodes in the cluster" on page 7.

Check NFS

Note: If the cluster does not use NFS as the distributed file system, skip this section.

Check the NFS service:

```
systemctl status nfs-server | grep Active && exportfs -v | grep -E '/home|/opt/ohpc/pub'
```

Notes:

- The output should be as follows：

  ```
  Active: active (exited) since Sat 2019-10-12 16:04:21 CST; 2 days ago
  ```

  ```
  /opt/ohpc/pub <world> (sync,wdelay,hide,no_subtree_check,sec=sys,ro,secure,root_squash,no_all_
  squash)
  ```

  ```
  /home <world>(async,wdelay,hide,no_subtree_check,sec=sys,rw,secure,no_root_squash,no_all_squash)
  ```
- If the status is not "active (exited)" or there is no output for `exportfs`, refer to "Configure NFS" on page 21 and "Configure shared directory for OpenHPC" on page 21.

Check the mounting points on all other nodes:

```
nodeshell all "df | grep -E '/home | /opt/ohpc/pub'"
```

Notes:

- The output should be as follows:

  ```
  c1: 10.1.1.31:/home 485642240 111060992 374581248 23% /home
  ```

  ```
  c1: 10.1.1.31:/opt/ohpc/pub 485642240 111060992 374581248 23% /opt/ohpc/pub
  ```
- If the status is not "active (exited)" or there is no output for `exportfs`, refer to "Configure NFS" on page 21 and "Configure shared directory for OpenHPC" on page 21.

Check Slurm

Check slurmctld:

```
systemctl status slurmctld | grep Active
```

Notes:

- The output should be as follows:

  ```
  Active: active (running) since Tue 2018-07-24 19:02:49 CST; 1 months 20 days ago
  ```
- If the status is not "active (running)", go to "Install Slurm" on page 20 and "Configure Slurm" on page 25.

Check slurmd on the compute nodes:

```
nodeshell compute "systemctl status slurmd | grep Active"
```

Notes:

- The output should be as follows:

  ```
  c1: Active: active (running) since Tue 2018-07-24 19:02:49 CST; 1 months 20 days ago
  ```

  ```
  c2: Active: active (running) since Sat 2018-07-21 17:16:59 CST; 1 months 23 days ago
  ```

- If the output does not contain all compute nodes, go to "Install Slurm" on page 20 and "Configure Slurm" on page 25.

Check MPI and Singularity

Check MPI and Singularity:

```
module list
```

Notes:

- The output should be as follows:

  ```
  Currently Loaded Modules:
  ```

  ```
  1) prun/2.2 2) gnu12/12.2.0 3) openmpi4/4.1.4 4) singularity/3.7.1 5) ohpc
  ```
- If the output does not contain one of the following: openmpi4, mpich, or mvapich2, refer to "Install MPI" on page 32.
- If the output does not contain "singularity", refer to "Install Singularity" on page 33.

Check OpenHPC installation

Check the OpenHPC installation for the cluster. If the OpenHPC installation check fails, reconfigure OpenHPC referring to "Install infrastructure software for nodes" on page 15.

Install the LiCO dependencies

Note: In the **Installation node** column, M stands for "Management node", L stands for "Login node", and C stands for "Compute node".

Table 5. LiCO dependencies to be installed

Software	Component	Version	Service	Installation node	Notes
rabbitmq	rabbitmq-server	3.9.10	rabbitmq-server	M	
mariadb	mariadb-server	10.3.32	mariadb		
influxdb	influxdb	1.8.10	influxdb	M	
libuser	libuser	0.62	/	M	/
	python3-libuser	0.62	/	M	/

Install MariaDB

LiCO uses MariaDB as an object-related database for data storage.

Step 1. Install MariaDB:

```
dnf install -y mariadb-server mariadb-devel
```

Step 2. Start the MariaDB service:

```
systemctl enable mariadb --now
```

Step 3. Configure MariaDB for LiCO:

Note: The username and password will be used in installing lico-passwd-tool. Therefore, keep a record of them when installing MariaDB.

```
mysql
```

```
create database lico character set utf8 collate utf8_bin;

create user '<USERNAME>'@'%' identified by '<PASSWORD>';

grant ALL on lico.* to '<USERNAME>'@'%';

exit
```

Step 4. Configure the MariaDB limits:

```
sed -i "/\[mysqld\]/a\max-connections=1024" /etc/my.cnf.d/mariadb-server.cnf

mkdir /usr/lib/systemd/system/mariadb.service.d

cat << eof > /usr/lib/systemd/system/mariadb.service.d/limits.conf

[Service]

LimitNOFILE=10000

eof

systemctl daemon-reload

systemctl restart mariadb
```

Install InfluxDB

LiCO uses InfluxDB as a time series database for storage monitoring.

Step 1. Iinstall InfluxDB:

```
dnf install -y influxdb

systemctl enable influxdb --now
```

Step 2. Create InfluxDB users:

- To enter the InfluxDB shell:

  ```
  influx
  ```

- To create database:

  ```
  create database lico
  ```

- To use database:

  ```
  use lico
  ```

- To create an administrator user, ensure that the password is a string:

  ```
  create user <INFLUX_USERNAME> with password '<INFLUX_PASSWORD>' with all privileges
  ```

- To exit the influxDB shell:

  ```
  exit
  ```

- To do configuration:

  ```
  sed -i '/# auth-enabled = false/a\ auth-enabled = true' /etc/influxdb/config.toml
  ```

- To restart InfluxDB:

  ```
  systemctl restart influxdb
  ```

Configure user authentication

Install libuser

The libuser module is a recommended toolkit for OpenLDAP. The installation of this module is optional.

Step 1. Install libuser:

```
dnf install -y libuser
```

Step 2. Download `libuser.conf` from https://hpc.lenovo.com/lico/downloads/7.1/examples/conf/ to `/etc` on the management node, and modify this file referring to the instructions in the file.

Chapter 4. Deploy LiCO in local cluster

This chapter describes how to deploy LiCO in local cluster.

Attention: To deploy LiCO in container, skip this chapter.

Install LiCO

Note: In the **Installation node** column, M stands for “Management node”, L stands for “Login node”, and C stands for “Compute node”.

Table 6. List of LiCO components to be installed

Software	Component	Version	Service	Installation node	Notes
lico-core	lico-core	7.1.0	lico	M	
lico-portal	lico-portal	7.1.0		M, L	
lico-core-extend	lico-confluent-proxy	1.3.0		M	
	lico-vnc-proxy	1.4.0	lico-vnc-proxy	M	
lico monitor	openlico_monitor	1.0.0		M	
lico alarm notification	lico-sms-agent	1.2.7	lico-sms-agent	M	Required if alerts should be sent by SMS.
	lico-wechat-agent	1.2.7	lico-wechat-agent	M	Required if alerts should be sent by WeChat.
	lico-mail-agent	1.4.0	lico-mail-agent	M	Required if alerts should be sent by e-mail.
lico manager	lico-file-manager	2.3.0	lico-file-manager	M	Essential components
lico-task	lico-async-task	2.0.0	lico-async-task	M, L	

Install LiCO Core

Step 1. Install RabbitMQ as a message broker.

```
dnf install -y rabbitmq-server
```

```
systemctl enable rabbitmq-server --now
```

Step 2. Install the following packages for LiCO module.

```
dnf install -y python3-cffi python3-libuser buildah
```

Step 3. Do one of the following:

- To use LiCO for cluster monitoring, install the LiCO module:

```
dnf install -y lico-core lico-file-manager lico-confluent-proxy \
lico-vnc-proxy lico-async-task lico-bash-profile
```

- For other purposes, install the LiCO module:

```
dnf install -y lico-core lico-file-manager lico-confluent-proxy \
lico-async-task lico-bash-profile
```

Step 4. (Optional) Provide e-mail, SMS, and WeChat services:

```
dnf install -y lico-mail-agent
dnf install -y lico-sms-agent
dnf install -y lico-wechat-agent
```

Step 5. Restart services:

```
systemctl restart confluent
```

Install LiCO on the login node

Install the LiCO module on the login node:

```
nodeshell login dnf install -y lico-bash-profile lico-portal
```

Configure LiCO

Configure the service account

Notes:

- The user name or password of MariaDB, InfluxDB, Confluent, LDAP are configured in this guide.
- Obtain the user name and password of icinga2 through the `/etc/icinga2/conf.d/api-users.conf` file.
- Obtain the user name and password of datasource through the `/etc/icinga2/features-available/influxdb.conf` file.
- Customize a cluster key for encrypting the input user name and password, and keep this cluster key in a safe place.

On the management node, use the tool `lico-password-tool`.

Input the cluster key and the user name or password for MariaDB, InfluxDB, Confluent, Icinga2, LDAP, and datasource following the prompt:

```
lico-password-tool
```

Configure service account for other nodes:

```
cp /var/lib/lico/tool/.db* $share_installer_dir
nodeshell login mkdir -p /var/lib/lico/tool/
nodeshell login cp $share_installer_dir/.db* /var/lib/lico/tool/
```

Configure the authorized key

Step 1. Modify the `/etc/ssh/sshd_config` configuration file and the following content:

```
AuthorizedKeysCommand /bin/bash -c 'exec /opt/lico/pub/ssh/lico-authorized-keys --home %h'
AuthorizedKeysCommandUser root
```

Step 2. Reload the sshd service.

```
systemctl reload sshd
```

Step 3. Configure the authorized key for other nodes:

```
cp /etc/ssh/sshd_config $share_installer_dir
```

```
cp /etc/ssh/ssh_config.d/10-lico.conf $share_installer_dir
```

```
nodeshell all cp $share_installer_dir/10-lico.conf /etc/ssh/ssh_config.d/
```

```
nodeshell all cp $share_installer_dir/sshd_config /etc/ssh/
```

Step 4. Reload the sshd service for other nodes:

```
nodeshell all systemctl reload sshd
```

Configure cluster nodes

Step 1. Import the cluster information to the system:

```
cp /etc/lico/nodes.csv.example /etc/lico/nodes.csv
```

Step 2. Edit the cluster information file:

```
vi /etc/lico/nodes.csv
```

Note: It is recommended to download the file to the local and edit it using EXCEL or other table editing software. Then, this file can be uploaded to the management node to overwrite the original file.

Room information

Below is an example of the room information table.

Table 7. Room information table

room	name	location_description
	Shanghai Solution Room	Shanghai Zhangjiang

Enter one entry of information for the fields **name** and **location_description**.

Logic group information

Managers can use logic groups to divide nodes in the cluster into groups. The logic groups do not impact the use of computer resources or permissions configurations.

Below is an example of the logic group information table.

Table 8. Logic group information table

group	name
	login

Input at least one logic group name in the **name** field.

Room row information

Room row refers to the rack order in the room. Enter the information about the rack row where the cluster node is located.

Below is an example of the room row information table.

Table 9. Room row information table

row	name	index	belonging_room
	row1	1	Shanghai Solution Room

Enter at least one entry of row information in the fields below:
- **name**: row name (must be unique in the same room)
- **index**: row order (must be a positive integer and be unique in the same room)
- **belonging_room**: name of the room where the row belongs

  Note: Add this information to the room information table.

Rack information

Below is an example of the rack information table.

Table 10. Rack information table

rack	name	column	belonging_row
	rack1	1	row1

Enter at least one entry of rack information in the fields below:
- **name**: rack name (must be unique in the same room)
- **column**: rack location column, also known as rack number (must be a positive integer and be unique in the same row)
- **belonging_row**: name of the row where the rack belongs

  Note: Add this information to the row information table.

Chassis information

If there is a chassis in the cluster, enter the chassis information.

Below is an example of the chassis information table.

Table 11. Chassis information table

chassis	name	belonging_rack	location_u_in_rack	machine_type
	chassis1	rack1	7	7X20

The fields in this table are described as follows:
- **name**: chassis name (must be unique in the same room)
- **belonging_rack**: rack location name (must use the name configured in the rack information table)
- **location_u_in_rack**: location of the chassis base in the rack (Unit: U). In a standard cabinet, the value should be between 1 and 42. For example, a chassis base is located at 5U.
- **machine_type**: chassis type (refer to "Supported servers and chassis models" on page 3)

Node information

Enter the information about all nodes in the cluster into the node information table. Due to its width, the example node information table is displayed in two split parts.

Table 12. Node information table (Part 1)

node	name	nodetype	immip	hostip	machine_ type	ipmi_user
	head	head	10.240.212.13	127.0.0.1	7X58	<BMC_ USERNAME>

Table 13. Node information table (Part 2)

ipmi_pwd	belonging_rack	belonging_chassis	location_u	groups
<BMC_PASSWORD>	rack1		2	login

The fields are described as follows:

- **name**: node hostname (domain name not needed)
- **nodetype**: **head** means management node; **login** means login node; **compute** means compute node.
- **immip**: IP address of the node's BMC system
- **hostip**: IP address of the node on the host network
- **machine_type**: product name for the node (see "Supported servers and chassis models" on page 3)
- **ipmi_user**: XCC (BMC) account for the node
- **ipmi_pwd**: XCC (BMC) password for the node
- **belonging_rack**: name of the rack where the node is located (need to add the configured name to the rack information table). If the node belongs to a chassis, leave this field blank.
- **belonging_chassis**: name of the chassis where the node is located (need to add the configured name to the chassis information table). If the node belongs to a rack, leave this field blank.
- **location_u**: node location. If the node is located in the chassis, enter the slot in the chassis in which the node is located. If the node is located in a rack, enter the location of the node base in the rack (Unit: U).
- **groups**: name of the node location logic group. One node can belong to multiple logic groups. Group names should be separated by ";". Configure the logic group name in the logic group information table.

Configure generic resources

This module only executes when the scheduler is slurm. Do one of the following to configure generic resource:

- If no generic resources are configured by default, GPU resource is in cluster and accounting is required, run the following command：

  ```
  cp /etc/lico/gres.csv.example /etc/lico/gres.csv
  ```

- If Slurm is configured with other generic resource, and accounting is required for these resources, run the following command:

  ```
  vi /etc/lico/gres.csv
  ```

  Note: To ensure the historical billing information accuracy, the generic resource removed from `gres.csv` will still remain in the system database.

Gres information

Following is an example of the gres information table：

code	display_name	unit
gpu	GPU	card

Enter at least one entry of generic resource information in the fields below:

- **code**: Code should align with the generic resource type defined in the scheduler. If LiCO is installed following this document, input the code according to the configuration of GresTypes in `slurm.conf`. In addition, users can define the type for the specific resource by adding the colon, for example, <resource>:<type>. However, this format is not supported under the billing policy, and users can add the limitation on the specific resource type to the scheduler limitation instead.

- **display_name**: Name of generic resource displayed in the LiCO system. A meaningful display name is recommended.
- **unit**: Unit of resource.

Configure LiCO components

For more information about configuring LiCO, refer to:
https://hpc.lenovo.com/lico/downloads/7.1/configuration/host/configuration.html

Configure the timezone offset for accounting

Edit `/etc/lico/lico.ini.d/accounting.ini` to configure the timezone offset:

```
[ACCOUNTING.BILLING]

TIMEZONE_OFFSET = <timezone offset>
```

lico-portal

To prevent the conflict between https and the NGINX web server, modify some pathway files for nodes installed with the `lico-portal` module, which provides external Web services with different ports.

/etc/nginx/nginx.conf

Edit `/etc/nginx/nginx.conf` by changing the port to **8080**:

```
listen 8080 default_server;

listen [::]:8080 default_server;
```

To hide the server version information, modify `/etc/nginx/nginx.conf` by turning off **server_tokens**:

```
http{

......

sendfile on;

server_tokens off;

……

}
```

/etc/nginx/conf.d/lico.conf.d/00-bind.conf

Edit `/etc/nginx/conf.d/lico.conf.d/00-bind.conf` by changing the default https port 443 to another port:

```
listen <port> ssl http2;
```

Note: Ensure that the port is not used by other applications and is not blocked by the firewall.

/etc/nginx/conf.d/lico.conf.d/00-params.conf

Edit `/etc/nginx/conf.d/lico.conf.d/00-params.conf` by replacing the first line to the following content:

```
set $lico_host 127.0.0.1;
```

Note: If `lico-portal` does not run, change 127.0.0.1 to the IP address of the management node.

/etc/lico/portal.conf

Edit `/etc/lico/portal.conf` by adding custom shortcut links. Refer to `/etc/lico/portal.conf.example` for the configuration format.

Initialize the system

Initialize LiCO:

```
lico init
```

Initialize Cloud Tools

Initialize Cloud Tools:

```
lico cloudtool import -n 'CVAT' -c \

cvat -t cvat -p job_queue,cores_per_node,username,password,ram_size,share_dir


lico cloudtool import -n 'Jupyter Notebook' -c jupyter -t jupyter -p \

image_path,jupyter_cmd,password,job_queue,cores_per_node,gpu_per_node,check_timeout,run_time


lico cloudtool import -n 'RStudio Server' -c \

rstudio -t rstudio -p job_queue,cores_per_node,gpu_per_node,password,run_time


lico cloudtool import -n 'TigerVNC' -c \

tigervnc -t tigervnc -p job_queue,cores_per_node,gpu_per_node,runtime_id,password,run_time
```

Initialize users

Complete the following step to initialize LiCO users:

Step 1. (Optional) To use LDAP to manage user, find the following configuration in the LiCO configuration file `/etc/lico/lico.ini.d/user.ini` and change the value to “true”:

```
USE_LIBUSER = false
```

Step 2. (Optional) Add a new user to LDAP with administrator privileges:

```
luseradd <HPC_ADMIN_USERNAME> -P <HPC_ADMIN_PASSWORD>

nodeshell all "su - <HPC_ADMIN_USERNAME> -c whoami"
```

Note: Use `LDAP_PASSWORD` configured in “Install OpenLDAP” on page 34.

Step 3. Import the user into LiCO:

```
lico import_user -u <HPC_ADMIN_USERNAME> -r admin
```

Import system images

Generate and upload LiCO specified images based on the instructions on https://hpc.lenovo.com/lico/downloads/7.1/images/host/readme.html

Start and log in to LiCO

Start LiCO

Step 1. Start LiCO-related services on the login node:

```
nodeshell login systemctl enable nginx --now
```

Step 2. Start LiCO-related services on the management node:

```
systemctl enable lico-async-task --now

systemctl enable lico-confluent-proxy --now

systemctl enable lico-vnc-proxy --now

systemctl enable lico-file-manager --now

systemctl enable lico-mail-agent --now

systemctl enable lico-sms-agent --now

systemctl enable lico-wechat-agent --now

systemctl enable lico --now
```

(Optional) LiCO Unmonitor Configuration

When LiCO is not used for cluster monitoring, use the unmonitored version referring to the “Monitor configuration” section in https://hpc.lenovo.com/lico/downloads/7.1/configuration/host/configuration.html, and collect the cluster information.

```
lico collect_cluster_property
```

Note: If these steps are skipped when using unmonitored LiCO, the status of the imported license would be “unmatch”.

Log in to LiCO

After the LiCO service is started, point the browser to the following location:
`https://<ip of login node>:<port>/`

Note: Replace `port` with the port number set in `/etc/nginx/conf.d/lico.conf.d/00-bind.conf` in “lico-portal” on page 50.

If the installation is correct, the LiCO login page opens. Users can log in using the LDAP account set in “Initialize users” on page 51.

Configure the LiCO services

The LiCO service configuration files are under the `/etc/lico` directory. These configuration files control the operating parameters for various LiCO background service components. Users can modify this configuration file as needed.

If the configuration or the operating status of components mentioned in this document is changed while LiCO is running, restart LiCO:

```
systemctl restart lico
```

Note: Configurations not mentioned in the instructions in this section can be modified after consulting with service staff. Modifications made without a service consultation could result in a system failure.

Chapter 5. Deploy LiCO in container

This chapter describes how to deploy LiCO in container.

Attention: To deploy LiCO in local cluster, skip this chapter.

Install docker-ce

Step 1. Enable docker-ce repo:

```
dnf install -y yum-utils

yum-config-manager --add-repo https://download.docker.com/linux/centos/docker-ce.repo
```

Step 2. Install docker-ce:

```
dnf install -y docker-ce docker-ce-cli containerd.io \

docker-buildx-plugin docker-compose-plugin
```

Step 3. Enable docker services

```
systemctl enable docker --now
```

Step 4. Install buildah:

```
dnf install -y 'dnf-command(copr)'

dnf module disable -y container-tools

dnf copr enable -y rhcontainerbot/container-selinux


curl -L -o /etc/yum.repos.d/devel:kubic:libcontainers:stable.repo \

https://download.opensuse.org/repositories/devel:kubic:libcontainers:stable/CentOS_8/\

devel:kubic:libcontainers:stable.repo


dnf install -y buildah

dnf module enable -y container-tools
```

Build LiCO image

Step 1. Prepare packages. Download the following packages and upload all of them to the `/root` directory:

- `Lenovo-OpenHPC-2.6.1.EL8.x86_64.tar`:
  https://hpc.lenovo.com/lico/downloads/7.1/Lenovo-OpenHPC-2.6.1.EL8.x86_64.tar
- `lico-dep-7.1.0.el8.x86_64.tgz`:
  https://hpc.lenovo.com/lico/downloads/7.1/lico-dep-7.1.0.el8.x86_64.tgz
- `authselect.tar.gz`:
  https://hpc.lenovo.com/lico/downloads/7.1/examples/conf/authselect/authselect.tar.gz
- LiCO 7.1.0 release package `lico-release-7.1.0.el8.x86_64.tar.gz`:
  https://commercial.lenovo.com/cn/en/signin
- LiCO 7.1.0 docker package `lico-docker-7.1.0.x86_64.tar.gz`:
  https://commercial.lenovo.com/cn/en/signin

Step 2. Prepare build environment:

```
cd /root
lico_container_workspace="/root/lico-workspace/7.1.0"
mkdir -p $lico_container_workspace

tar -xzvf lico-docker-7.1.0.x86_64.tar.gz -C $lico_container_workspace
mv Lenovo-OpenHPC-2.6.1.EL8.x86_64.tar $lico_container_workspace
mv lico-dep-7.1.0.el8.x86_64.tgz $lico_container_workspace
mv lico-release-7.1.0.el8.x86_64.tar.gz $lico_container_workspace
mv authselect.tar.gz $lico_container_workspace

cp $lico_container_workspace/lico-docker/Dockerfile \
$lico_container_workspace/lico-docker/scripts/build.sh $lico_container_workspace

cp $lico_container_workspace/lico-docker/scripts/lico-control /usr/bin/

chmod +x $lico_container_workspace/build.sh
chmod +x /usr/bin/lico-control
```

Step 3. Build LiCO image:

```
cd $lico_container_workspace
./build.sh
```

Initialize container LiCO

Step 1. Prepare the mount folder and configure the file:

```
lico-control prepare
```

Step 2. Configure cluster nodes. Refer to "Configure cluster nodes" on page 47.

Step 3. Initial data for LiCO:

```
lico-control init --mode all
```

Attention:

- When the program starts running the `lico-password-tool` command, enter the user name and password as described in "Configure the service account" on page 46.
- LiCO will initialize an administrator account for logging in to the LiCO web page, remember to check the output of the program.
- Configure the authorized key. Refer to "Configure the authorized key" on page 46.

Configure container LiCO

Import system images

Generate and upload LiCO specified images based on the instructions on https://hpc.lenovo.com/lico/downloads/7.1/images/container/readme.html.

Configure LiCO components

To configuring LiCO components, refer to: https://hpc.lenovo.com/lico/downloads/7.1/configuration/container/configuration.html

Start and log in to container LiCO

Start container LiCO

Start container LiCO:

```
lico-control run
```

Log in to contianer LiCO

After the LiCO service is started, go to:
`https://<ip of lico node>:<port>/`

Configure the LiCO services

The LiCO service configuration files, controlling the operating parameters for various LiCO service components, are under the `/etc/lico` directory. Users can modify these files based on the actual needs. If the configuration or the operating status of components mentioned in the specific document is changed while LiCO is running, restart LiCO:

```
lico-control restart
```

Note: For the configurations not mentioned in the configuration files, it is recommended to consult Lenovo Service before making any changes; otherwise, it might cause the system failure.

Chapter 6. Appendix: Important information

List of cluster services

Note: In the **Installation node** column, M stands for “Management node”, L stands for “Login node”, and C stands for “Compute node”.

Table 14. List of cluster services

Software	Component	Service	Default port	Installation node
lico	lico-core	lico	18080/tcp	M
	lico-confluent-proxy		18081/tcp	M
	lico-vnc-proxy	lico-vnc-proxy	18082/tcp	M
	lico-sms-agent	lico-sms-agent	18092/tcp	M
	lico-wechat-agent	lico-wechat-agent	18090/tcp	M
	lico-mail-agent	lico-mail-agent	18091/tcp	M
	lico-file-manager	lico-file-manager	18085/tcp	M
	lico-async-task	lico-async-task	18086/tcp	M, L
lico dependencies	ngnix	ngnix	80/tcp, 443/tcp	L, M
	rabbitmq	rabbitmq-server	5672/tcp	M
	mariadb	mariadb	3306/tcp	
	confluent	confluent	4005/tcp	M
	influxdb	influxdb	8086/tcp, 8088/tcp	M
	ldap	slapd	389/tcp,636/tcp	M
		nslcd		M, C, L
cluster	nfs	nfs	111/tcp, 111/udp, 2049/tcp, 2049/udp	M, C, L
	chrony	chronyd		M, C, L
	slurm	munge		M, C
		slurmctld	6817/tcp	M
		slurmd	6818/tcp	C
	Icinga2	icinga2	5665/tcp, 5665/udp	M, C, L
	dns	named	53/udp	M
	dhcp	dhcpd	67/udp	M

(Optional) Install Intel oneAPI

Attention: This section is only for deploying LiCO in local cluster. To deploy LiCO in container, consult Lenovo Service.

To install Intel oneAPI, go to:
https://hpc.lenovo.com/lico/downloads/7.1/Install_Intel_oneAPI.html

To configure Intel oneAPI, go to:
https://hpc.lenovo.com/lico/downloads/7.1/configuration/host/configuration.html

(Optional) Initialize Hybird HPC

For RedHat

To use the functions of LiCO Hybird HPC, go to:
https://hpc.lenovo.com/lico/hybrid/hpc/en-us/initialize.html

(Optional) Configure VNC

Note: If LiCO is not used for cluster monitoring, skip this section.

Install the VNC module only on compute nodes that require the VNC functionality and GUI.

Step 1. Run the following commands on the target compute node to install the VNC function:

```
dnf install -y gdm tigervnc tigervnc-server nautilus-open-terminal
```

Step 2. Edit `/etc/gdm/custom.conf` on the compute node, and make the following changes:

```
[xdmcp]

Enable=true
```

Step 3. Restart the compute node to make the changes take effect:

```
reboot
```

Step 4. Start Tigervnc server. Refer to:
https://github.com/TigerVNC/tigervnc/blob/master/unix/vncserver/HOWTO.md

Configure the Confluent Web console

To open the management node console from the LiCO web portal, configure and restart the management node before the configurations take effect.

For Rocky

Step 1. Edit the `/etc/default/grub` file by adding the following text to the end of `GRUB_CMDLINE_LINUX`:

For the ThinkSystem SR635/SR655 server, add:

```
console=ttyS1,115200
```

For other server models, add:

```
console=ttyS0,115200
```

Step 2. Start the UEFI mode or legacy mode.

To start the legacy mode:

```
grub2-mkconfig -o /boot/grub2/grub.cfg
```

To start the UEFI mode:

```
grub2-mkconfig -o /boot/efi/EFI/rocky/grub.cfg
```

For RHEL

Step 1. Edit the `/etc/default/grub` file by adding the following text to the end of `GRUB_CMDLINE_LINUX`:

For the ThinkSystem SR635/SR655 server, add:

```
console=ttyS1,115200
```

For other server models, add:

```
console=ttyS0,115200
```

Step 2. Start the UEFI mode or legacy mode.

To start the legacy mode:

```
grub2-mkconfig -o /boot/grub2/grub.cfg
```

To start the UEFI mode:

```
grub2-mkconfig -o /boot/efi/EFI/redhat/grub.cfg
```

(Optional) Configure EAR

Attention: This section is only for deploying LiCO in local cluster. To deploy LiCO in container and configure EAR through LiCO, consult Lenovo Service.

Energy Aware Runtime (EAR) library is designed to provide an energy efficient solution for MPI applications. It aims at finding the optimal frequency for a job according to its selected energy policy, being totally dynamic and transparent.

LiCO supports to configure EAR through LiCO. For more information about EAR, go to:

https://gitlab.bsc.es/ear_team/ear/-/wikis/Home

To use this function, run the command `dnf install -y lico-core-ear` to insall lico-core-ear component, and modify the configuration referring to:
https://hpc.lenovo.com/lico/downloads/7.1/configuration/host/configuration.html

LiCO supports EAR 3.3.

LiCO commands

Change a user role

- To change a user role in local cluster:

```
lico change_user_role -u <ROLE_USERNAME> -r <ROLE>
```

- To change a user role in container:

```
docker exec -it lico lico change_user_role -u <ROLE_USERNAME> -r <ROLE>
```

Table 15. Parameter description

Parameter	Description
-u	Specify the user name to be modified.
-r	Specify the role to be set, including administrator, operator, and user.

Resume a user

- To resume a user in local cluster:

```
lico resume_user -u <SUSPENDED_USERNAME>
```

- To resume a user in container:

```
docker exec -it lico lico resume_user -u <SUSPENDED_USERNAME>
```

Table 16. Parameter description

Parameter	Description
-u	Specify the user to be resumed.

Delete a user

- To delete a user in local cluster:

```
lico delete_user -u <DELETED_USERNAME>
```

- To delete a user in container:

```
docker exec -it lico lico delete_user -u <DELETED_USERNAME>
```

Table 17. Parameter description

Parameter	Description
-u	Specify the user to be deleted.

Import a user

- To import a user in local cluster:

```
lico import_user -u <IMPORT_USERNAME> -r <ROLE>
```

- To import a user in container:

```
docker exec -it lico lico import_user -u <IMPORT_USERNAME> -r <ROLE>
```

Table 18. Parameter description

Parameter	Description
-u	Specify the user name to be imported.
-r	Specify the role to be set, including administrator, operator, and user.

Generate nodes.csv in confluent

If the confluent is in the cluster with the cluster information configured, generate the cluster configuration file of LiCO:

- To deploy LiCO in local cluster:

```
lico export_nodes_from_confluent
```

- To deploy LiCO in container:

```
docker exec -it lico lico export_nodes_from_confluent
```

Notes:

- Before running this command, ensure that the confluent management node information is configured in LiCO.
- After running this command, `export_nodes.csv` will be generated in the current directory by default. Users should rename the file to `nodes.csv` and reedit it according to "Configure cluster nodes" on page 47.
- For more information about parameters of the command, refer to the help file of the command.

Firewall settings

Considering the security of the system, it is recommended to enable the firewall on the management node and the login node.

Run the following commands to install and enable the firewall:

```
dnf install -y firewalld
```

```
systemctl enable firewalld --now
```

Note: To set up the cluster and installed LiCO following this document, set up the firewall first referring to the official firewall setup document: https://access.redhat.com/documentation/en-us/red_hat_enterprise_linux/8/html/security_hardening/index

Set the firewall on the management node

Step 1. Add all ports:

a. Add RPC application port:

Note: All the following ports are default settings. Check the settings by running the `rpcinfo -p` command.

```
firewall-cmd --zone=public --add-port=111/tcp --permanent
```

```
firewall-cmd --zone=public --add-port=111/udp --permanent
firewall-cmd --zone=public --add-port=2049/tcp --permanent
firewall-cmd --zone=public --add-port=2049/udp --permanent
firewall-cmd --zone=public --add-port=20048/tcp --permanent
firewall-cmd --zone=public --add-port=20048/udp --permanent
firewall-cmd --zone=public --add-port=52891/udp --permanent
firewall-cmd --zone=public --add-port=33504/tcp --permanent
firewall-cmd --zone=public --add-port=39123/tcp --permanent
firewall-cmd --zone=public --add-port=52656/udp --permanent
```

b. Add SSH service port:

```
firewall-cmd --zone=public --add-service=ssh --permanent
```

c. Add nginx service port:

Note: Port 443 can be configured based on the actual conditions.

```
firewall-cmd --zone=public --add-port=443/tcp --permanent
```

d. Add httpd service port:

```
firewall-cmd --zone=public --add-port=80/tcp --permanent
```

e. Add Icinga2 gmond port:

```
firewall-cmd --zone=public --add-port=5665/udp --permanent
firewall-cmd --zone=public --add-port=5665/tcp --permanent
```

f. Add Slurm slurmctld port:

```
firewall-cmd --zone=public --add-port=6817/tcp --permanent
```

g. Add OpenLDAP slapd port:

```
firewall-cmd --zone=public --add-port=636/tcp --permanent
firewall-cmd --zone=public --add-port=389/tcp --permanent
```

h. Add MariaDB port:

```
firewall-cmd --zone=public --add-service=mysql --permanent
```

i. Add lico-core port:

```
firewall-cmd --zone=public --add-port=18080-18095/tcp --permanent
```

j. Add ports managed by LiCO:

```
firewall-cmd --zone=public --add-port=25000-27500/tcp --permanent
```

k. Add DNS service port:

```
firewall-cmd --zone=public --add-service=dns --permanent
```

l. Add DHCP service port:

```
firewall-cmd --zone=public --add-service=dhcp --permanent
```

Step 2. Add the internal network interface into the public zone:

Note: For eth0 and eth1, refer to the internal and external network interface.

```
firewall-cmd --zone=public --add-interface=eth0 --permanent
```

```
firewall-cmd --zone=public --add-interface=eth1 --permanent
```

Step 3. Enable roles:

```
firewall-cmd --complete-reload
```

Set the firewall on the login node

Step 1. Add roles to public zone:

```
firewall-cmd --zone=public --add-service=ssh --permanent
```

- Add nginx service port and adjust 443 based on the actual setting:

  ```
  firewall-cmd --zone=public --add-port=443/tcp --permanent
  ```

- Add ports managed by LiCO:

  ```
  firewall-cmd --zone=public --add-port=25000-27500/tcp --permanent
  ```

Step 2. Add the internal and external network interface into the public zone:

Note: eth0 and eth1 refer to the internal and external network interface.

```
firewall-cmd --zone=public --add-interface=eth0 --permanent
```

```
firewall-cmd --zone=public --add-interface=eth1 --permanent
```

Step 3. Enable roles:

```
firewall-cmd --complete-reload
```

Improve cryptographic policies

The system-wide cryptographic policies is a system component configuring the core cryptographic subsystems, covering the TLS, IPsec, SSH, DNSSec, and Kerberos protocols.

To improve policies, go to https://access.redhat.com/documentation/en-us/red_hat_enterprise_linux/8/html/security_hardening/using-the-system-wide-cryptographic-policies_security-hardening

Slurm issues troubleshooting

This section lists the solutions to some Slurm issues.

Node status check

Use the Slurm command `sinfo` to check the node status.

- If the status is **drain**, change the node status to normal:

  ```
  scontrol update NodeName=host1 State=RESUME
  ```

- If the node status is **down**, do the following:
  1. Use the Slurm command `scontrol show nodes` to view the detailed node information and view the **reason** in the output of this command.
  2. Ensure that all nodes have the same `slurm.conf` file under `/etc/slurm`.
  3. Ensure that the **slurmd** and **munge** services are active on all the nodes, and that the **slurmctld** service is active on the management node.
  4. Ensure that all nodes have the same date and that the **chronyd** service is active on all nodes.

Status setting error

When setting the slurm queue node status to **DOWN**, but the status is automatically changed to **IDLE**, users can edit `/etc/slurm/slurm.conf`, and set the **ReturnToService** to **0**: **ReturnToService=0**.

Confluent issues troubleshooting

- If the time-out message is displayed when pushing operating system to other nodes, log in to BMC of the node, and click **BMC Configuration → Network** to check whether **IPMI over LAN** is enabled.
- If failing to select the partition for the installed nodes when deploying cluster through confluent, log in to the node, run **dd if=/dev/zero of=/dev/sda bs=1M count=32** based on the actual conditions of the hard disk, and re-deploy the node.
- For any problems in using Confluent, go to:
  https://hpc.lenovo.com/users/documentation/

Running job issue troubleshooting

When running a GPU job, the following error message might be displayed:

```
failed call to cuInit: CUDA_ERROR_UNKNOWN: unknown error

retrieving CUDA diagnostic information for host: c1
```

In this case, run the following commands on the management node:

```
nodeshell compute modprobe nvidia-uvm

nodeshell compute nvidia-modprobe -u -c=0
```

MPI issues troubleshooting

When running an Open MPI program, the following error might be displayed:

```
WARNING: Open MPI accepted a TCP connection from what appears to be a another Open MPI process

but cannot find a corresponding process entry for that peer.
```

If the TCP connection is ignored, the Open MPI program might not be executed properly.

When the Open MPI program uses the unroutable USB NICs, whose name might be “enp0s20f0u1u6” or similar under RedHat/Rocky 8 , this warning might be displayed. Select one of the following workarounds to resolve this issue:

Note: In the following workarounds, change <USB_NIC_NAME> to the absolute name of the unroutable USB NICs based on the actual conditions.

- Disable the USB NICs on all nodes by running the following command:

  ```
  nodeshell all ifconfig <USB_NIC_NAME> down
  ```

  Note: This step might interrupt the running Lenovo management tools, such as OneCLI. To use OneCLI, re-enable the NICs for a while.
- Instruct Open MPI to ignore the NICs:

  ```
  mpirun --mca btl_tcp_if_exclude <USB_NIC_NAME>
  ```

  Note: It is recommended to create the custom system-wide MPI templates.
- Permanently disable USB NICs:

  ```
  rmmod cdc_ether
  ```

Note: This step might permanently disable OneCLI and other Lenovo managenet tools.

Notices and trademarks

Notices

Lenovo may not offer the products, services, or features discussed in this document in all countries. Consult your local Lenovo representative for information on the products and services currently available in your area. Any reference to a Lenovo product, program, or service is not intended to state or imply that only that Lenovo product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any Lenovo intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any other product, program, or service.

Lenovo may have patents or pending patent programs covering subject matter described in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:

Lenovo (United States), Inc.
8001 Development Drive
Morrisville, NC 27560
U.S.A.
Attention: Lenovo Director of Licensing

LENOVO PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some jurisdictions do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

Changes are made periodically to the information herein; these changes will be incorporated in new editions of the publication. To provide better service, Lenovo reserves the right to improve and/or modify the products and software programs described in the manuals included with your computer, and the content of the manual, at any time without additional notice.

The software interface and function and hardware configuration described in the manuals included with your computer might not match exactly the actual configuration of the computer that you purchase. For the configuration of the product, refer to the related contract (if any) or product packing list, or consult the distributor for the product sales. Lenovo may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

The products described in this document are not intended for use in implantation or other life support applications where malfunction may result in injury or death to persons. The information contained in this document does not affect or change Lenovo product specifications or warranties. Nothing in this document shall operate as an express or implied license or indemnity under the intellectual property rights of Lenovo or third parties. All information contained in this document was obtained in specific environments and is presented as an illustration. The result obtained in other operating environments may vary.

Lenovo may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Any references in this publication to non-Lenovo Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this Lenovo product, and use of those Web sites is at your own risk.

Any performance data contained herein was determined in a controlled environment. Therefore, the result obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurements may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

For the latest information or any questions or comments, contact or visit the Lenovo Web site:

https://support.lenovo.com

Trademarks

LENOVO, LENOVO logo, THINKPAD, THINKPAD logo, TRACKPOINT, ULTRACONNECT, and Yoga are trademarks of Lenovo. Microsoft, Windows, Direct3D, BitLocker, and Cortana are trademarks of the Microsoft group of companies. Ubuntu is a registered trademark of Canonical Ltd. The terms HDMI and HDMI High-Definition Multimedia Interface are trademarks or registered trademarks of HDMI Licensing LLC in the United States and other countries. Wi-Fi, Wi-Fi Alliance, and Miracast are registered trademarks of Wi-Fi Alliance. USB-C is a trademark of USB Implementers Forum. All other trademarks are the property of their respective owners. © 2023 Lenovo.