SUSE Linux Enterprise High Performance Computing 15 SP5

Administration Guide

Administration Guide

SUSE Linux Enterprise High Performance Computing 15 SP5

This guide covers system administration tasks such as remote administration, workload management, and monitoring.

Publication Date: December 05, 2024

https://documentation.suse.com

For SUSE trademarks, see https://www.suse.com/company/legal/. All third-party trademarks are the property of their respective owners. Trademark symbols (®, ™ etc.) denote trademarks of SUSE and its affiliates. Asterisks (*) denote third-party trademarks.

All information found in this book has been compiled with utmost attention to detail. However, this does not guarantee complete accuracy. Neither SUSE LLC, its affiliates, the authors nor the translators shall be held liable for possible errors or the consequences thereof.

Contents

Preface viii

1 Available documentation viii

2 Improving the documentation ix

3 Documentation conventions x

4 Support xii

Support statement for SUSE Linux Enterprise High Performance Computing xii • Technology previews xiii

1 Introduction 1

1.1 Components provided 1

1.2 Hardware platform support 2

1.3 Support and lifecycle 2

1.4 Documentation and other information 2

2 Installation and upgrade 3

2.1 System roles for SUSE Linux Enterprise High Performance Computing 15 SP5 3

2.2 Upgrading to SUSE Linux Enterprise High Performance Computing 15 SP5 4

3 Deploying compute nodes 5

3.1 About Warewulf 5

3.2 Deploying compute nodes with Warewulf 6

3.3 Advanced Warewulf tasks 9

Using Warewulf with UEFI Secure Boot 9

3.4 For more information 10

4 Remote administration 11

4.1 Genders — static cluster configuration database 11

Genders database format 11 • Nodeattr usage 12

4.2 pdsh — parallel remote shell program 12

4.3 PowerMan — centralized power control for clusters 13

4.4 MUNGE authentication 14

Setting up MUNGE authentication 14 • Enabling and starting MUNGE 15

4.5 mrsh/mrlogin — remote login using MUNGE authentication 16

5 Hardware 17

5.1 cpuid 17

5.2 hwloc — portable abstraction of hierarchical architectures for high-performance computing 17

6 Slurm — utility for HPC workload management 18

6.1 Installing Slurm 18

Minimal installation 18 • Installing the Slurm database 21

6.2 Slurm administration commands 24

scontrol 24 • sinfo 25 • sacctmgr and sacct 25 • sbatch, salloc, and srun 26

6.3 Upgrading Slurm 28

Slurm upgrade workflow 28 • Upgrading the `slurmdbd` database daemon 29 • Upgrading `slurmctld` and `slurmd` 31

6.4 Frequently asked questions 35

7 Monitoring and logging 37

7.1 ConMan — the console manager 37

7.2 Monitoring HPC clusters with Prometheus and Grafana 38

Installing Prometheus and Grafana 39 • Monitoring cluster workloads 40 • Monitoring compute node performance 42

7.3 rasdaemon — utility to log RAS error tracings 44

8 HPC user libraries 46

8.1 Lmod — Lua-based environment modules 46

Installation and basic usage 46 • Listing available modules 47 • Listing loaded modules 47 • Gathering information about a module 47 • Loading modules 47 • Environment variables 47 • For more information 48

8.2 GNU Compiler Toolchain Collection for HPC 48

Environment module 48 • Building High Performance Computing software with GNU Compiler Suite 48 • Later versions 49

8.3 High Performance Computing libraries 49

NumPy Python library 50 • SciPy Python Library 50 • memkind — heap manager for heterogeneous memory platforms and mixed memory policies 51 • Support for PMIx in Slurm and MPI libraries 52 • OpenBLAS library — optimized BLAS library 52

8.4 File format libraries 53

HDF5 HPC library — model, library, and file format for storing and managing data 53

8.5 MPI libraries 55

8.6 Profiling and benchmarking libraries and tools 56

IMB — Intel* MPI benchmarks 56 • PAPI HPC library — consistent interface for hardware performance counters 57 • mpiP — lightweight MPI profiling library 57

8.7 Creating environment containers with Apptainer 58

9 Spack package management tool 59

9.1 Installing Spack 59

9.2 Using Spack: simple example with netcdf-cxx4 60

9.3 Using Spack: complex example with mpich 65

9.4 Using a specific compiler 69

10 Dolly clone tool 71

10.1 Dolly cloning process 71

10.2 Using dolly 72

10.3 Dolly configuration file 75

10.4 Dolly limitations 76

A GNU licenses 78

Preface

1 Available documentation

Online documentation

Our documentation is available online at https://documentation.suse.com ↗. Browse or download the documentation in various formats.

> ### Note: Latest updates
>
> The latest updates are usually available in the English-language version of this documentation.

SUSE Knowledgebase

If you run into an issue, check out the Technical Information Documents (TIDs) that are available online at https://www.suse.com/support/kb/ ↗. Search the SUSE Knowledgebase for known solutions driven by customer need.

Release notes

For release notes, see https://www.suse.com/releasenotes/ ↗.

In your system

For offline use, the release notes are also available under `/usr/share/doc/release-notes` on your system. The documentation for individual packages is available at `/usr/share/doc/packages`.

Many commands are also described in their *manual pages*. To view them, run **`man`**, followed by a specific command name. If the **`man`** command is not installed on your system, install it with **`sudo zypper install man`**.

2 Improving the documentation

Your feedback and contributions to this documentation are welcome. The following channels for giving feedback are available:

Service requests and support

For services and support options available for your product, see https://www.suse.com/support/ ↗.

To open a service request, you need a SUSE subscription registered at SUSE Customer Center. Go to https://scc.suse.com/support/requests ↗, log in, and click *Create New*.

Bug reports

Report issues with the documentation at https://bugzilla.suse.com/ ↗.

To simplify this process, click the *Report an issue* icon next to a headline in the HTML version of this document. This preselects the right product and category in Bugzilla and adds a link to the current section. You can start typing your bug report right away.

A Bugzilla account is required.

Contributions

To contribute to this documentation, click the *Edit source document* icon next to a headline in the HTML version of this document. This will take you to the source code on GitHub, where you can open a pull request.

A GitHub account is required.

Note: *Edit source document* only available for English

The *Edit source document* icons are only available for the English version of each document. For all other languages, use the *Report an issue* icons instead.

For more information about the documentation environment used for this documentation, see the repository's README.

Mail

You can also report errors and send feedback concerning the documentation to `doc-team@suse.com`. Include the document title, the product version, and the publication date of the document. Additionally, include the relevant section number and title (or provide the URL) and provide a concise description of the problem.

3 Documentation conventions

The following notices and typographic conventions are used in this document:

- `/etc/passwd`: Directory names and file names
- *`PLACEHOLDER`*: Replace *`PLACEHOLDER`* with the actual value
- `PATH`: An environment variable
- **`ls`**, `--help`: Commands, options, and parameters
- `user`: The name of a user or group
- `package_name`: The name of a software package
- **Alt**, **Alt**–**F1**: A key to press or a key combination. Keys are shown in uppercase as on a keyboard.
- *File*, *File* › *Save As*: menu items, buttons
- **AMD/Intel** This paragraph is only relevant for the AMD64/Intel 64 architectures. The arrows mark the beginning and the end of the text block.
  IBM Z, POWER This paragraph is only relevant for the architectures `IBM Z` and `POWER`. The arrows mark the beginning and the end of the text block.
- *Chapter 1, "Example chapter"*: A cross-reference to another chapter in this guide.
- Commands that must be run with `root` privileges. You can also prefix these commands with the **`sudo`** command to run them as a non-privileged user:

```
# command
> sudo command
```

- Commands that can be run by non-privileged users:

```
> command
```

- Commands can be split into two or multiple lines by a backslash character (`\`) at the end of a line. The backslash informs the shell that the command invocation will continue after the end of the line:

```
> echo a b \
c d
```

- A code block that shows both the command (preceded by a prompt) and the respective output returned by the shell:

```
> command
output
```

- Notices

Warning: Warning notice

Vital information you must be aware of before proceeding. Warns you about security issues, potential loss of data, damage to hardware, or physical hazards.

Important: Important notice

Important information you should be aware of before proceeding.

Note: Note notice

Additional information, for example about differences in software versions.

Tip: Tip notice

Helpful information, like a guideline or a piece of practical advice.

- Compact Notices

Additional information, for example about differences in software versions.

Helpful information, like a guideline or a piece of practical advice.

4 Support

Find the support statement for SUSE Linux Enterprise High Performance Computing and general information about technology previews below. For details about the product lifecycle, see https://www.suse.com/lifecycle ↗.

If you are entitled to support, find details on how to collect information for a support ticket at https://documentation.suse.com/sles-15/html/SLES-all/cha-adm-support.html ↗.

4.1 Support statement for SUSE Linux Enterprise High Performance Computing

To receive support, you need an appropriate subscription with SUSE. To view the specific support offers available to you, go to https://www.suse.com/support/ ↗ and select your product.

The support levels are defined as follows:

L1

Problem determination, which means technical support designed to provide compatibility information, usage support, ongoing maintenance, information gathering and basic troubleshooting using available documentation.

L2

Problem isolation, which means technical support designed to analyze data, reproduce customer problems, isolate a problem area and provide a resolution for problems not resolved by Level 1 or prepare for Level 3.

L3

Problem resolution, which means technical support designed to resolve problems by engaging engineering to resolve product defects which have been identified by Level 2 Support.

For contracted customers and partners, SUSE Linux Enterprise High Performance Computing is delivered with L3 support for all packages, except for the following:

- Technology previews.
- Sound, graphics, fonts, and artwork.
- Packages that require an additional customer contract.

- Some packages shipped as part of the module *Workstation Extension* are L2-supported only.
- Packages with names ending in `-devel` (containing header files and similar developer resources) will only be supported together with their main packages.

SUSE will only support the usage of original packages. That is, packages that are unchanged and not recompiled.

4.2 Technology previews

Technology previews are packages, stacks, or features delivered by SUSE to provide glimpses into upcoming innovations. Technology previews are included for your convenience to give you a chance to test new technologies within your environment. We would appreciate your feedback. If you test a technology preview, please contact your SUSE representative and let them know about your experience and use cases. Your input is helpful for future development.

Technology previews have the following limitations:

- Technology previews are still in development. Therefore, they may be functionally incomplete, unstable, or otherwise *not* suitable for production use.
- Technology previews are *not* supported.
- Technology previews may only be available for specific hardware architectures.
- Details and functionality of technology previews are subject to change. As a result, upgrading to subsequent releases of a technology preview may be impossible and require a fresh installation.
- SUSE may discover that a preview does not meet customer or market needs, or does not comply with enterprise standards. Technology previews can be removed from a product at any time. SUSE does not commit to providing a supported version of such technologies in the future.

For an overview of technology previews shipped with your product, see the release notes at https://www.suse.com/releasenotes ↗.

1 Introduction

SUSE Linux Enterprise High Performance Computing is a highly scalable, high-performance parallel computing platform for modeling, simulation, and advanced analytics workloads.

1.1 Components provided

SUSE Linux Enterprise High Performance Computing 15 SP5 provides tools and libraries related to High Performance Computing. This includes:

- A workload manager
- Remote and parallel shells
- Performance monitoring and measuring tools
- A serial-console monitoring tool
- A cluster power management tool
- A tool for discovering the machine hardware topology
- System monitoring
- A tool for monitoring memory errors
- A tool for determining the CPU model and its capabilities (x86-64 only)
- A user-extensible heap manager capable of distinguishing between different kinds of memory (x86-64 only)
- Various MPI implementations
- Serial and parallel computational libraries providing common standards, such as BLAS, LAPACK, and others
- Serial and parallel libraries for the HDF5 file format

1.2 Hardware platform support

SUSE Linux Enterprise High Performance Computing 15 SP5 is available for the Intel 64/AMD64 (x86-64) and AArch64 platforms.

1.3 Support and lifecycle

SUSE Linux Enterprise High Performance Computing 15 SP5 is supported throughout the lifecycle of SUSE Linux Enterprise 15 SP5. Two lifecycle extensions, *Extended Service Overlap Support* (ESPOS) and *Long Term Support Service* (LTSS), are also available for this product. Any released package is fully maintained and supported until the availability of the next release.

For more information, see the Support Policy page at https://www.suse.com/support/policy.html ↗.

1.4 Documentation and other information

- Read the README files on the media.
- Get detailed change log information about a particular package from the RPM (where `FILENAME.rpm` is the name of the RPM):

  ```
  rpm --changelog -qp FILENAME.rpm
  ```

- Check the `ChangeLog` file in the top level of the media for a chronological log of all changes made to the updated packages.
- The most recent version of the release notes is always available at https://www.suse.com/releasenotes ↗.
- The most recent version of this documentation is always available at https://documentation.suse.com/ ↗.

2 Installation and upgrade

SUSE Linux Enterprise High Performance Computing comes with preconfigured system roles for HPC. These roles provide a set of preselected packages typical for the specific role, and an installation workflow that configures the system to make the best use of system resources based on a typical use case for the role.

2.1 System roles for SUSE Linux Enterprise High Performance Computing 15 SP5

You can choose specific roles for the system based on modules selected during the installation process. When the HPC Module is enabled, the following roles are available:

HPC management server (head node)

This role includes the following features:

- Uses Btrfs as the default root file system
- Includes HPC-enabled libraries
- Disables the firewall and Kdump services
- Installs a controller for the Slurm workload manager
- Mounts a large scratch partition to `/var/tmp`

HPC compute node

This role includes the following features:

- Based on the minimal setup configuration
- Uses XFS as the default root file system
- Includes HPC-enabled libraries
- Disables firewall and Kdump services
- Installs a client for the Slurm workload manager

- Does not create a separate `/home` partition
- Mounts a large scratch partition to `/var/tmp`

HPC development node

This role includes the following features:

- Includes HPC-enabled libraries
- Adds compilers and development toolchains

The scratch partition `/var/tmp/` is only created if there is sufficient space available on the installation medium (minimum 32 GB).

The environment module `Lmod` is installed for all roles. It is required at build time and runtime of the system. For more information, see *Section 8.1, “Lmod — Lua-based environment modules”*.

All libraries specifically built for HPC are installed under `/usr/lib/hpc`. They are not part of the standard search path, so the `Lmod` environment module system is required.

`MUNGE` authentication is installed for all roles. MUNGE keys are generated and must be copied to all nodes in the cluster. For more information, see *Section 4.4, “MUNGE authentication”*.

The system roles are only available for new installations of SUSE Linux Enterprise High Performance Computing.

2.2 Upgrading to SUSE Linux Enterprise High Performance Computing 15 SP5

You can upgrade to SLE HPC 15 SP5 from SLE HPC 15 SP4.

3 Deploying compute nodes

High Performance Computing clusters consist of one or more sets of identical compute nodes. In large clusters, each set could contain thousands of machines. To help deploy so many compute nodes as clusters scale up, the HPC module provides the deployment tool *Warewulf*.

3.1 About Warewulf

Warewulf is a deployment system for compute nodes in High Performance Computing clusters. Compute nodes are booted and deployed over the network with a kernel and node image provided by Warewulf. To generate the node image, Warewulf uses a *Warewulf container*, which is a base operating system container with a kernel and an `init` implementation installed. Warewulf configures images for the individual compute nodes using *node profiles* and *Warewulf overlays*.

Node profiles are used to apply the same configuration to multiple nodes. Node profiles can include settings such as the container to use, overlays to apply, and IPMI details. New nodes automatically use the `default` node profile. You can also create additional node profiles, for example, if two groups of nodes require different containers.

Warewulf overlays are compiled for each individual compute node:

- System (or `wwinit`) overlays are applied to nodes at boot time by the `wwinit` process, before `systemd` starts. These overlays are required to start the compute nodes, and contain basic node-specific configuration to start the first network interface. System overlays are not updated during runtime.

- Runtime (or generic) overlays are updated periodically at runtime by the `wwclient` service. The default is once per minute. These overlays are used to apply configuration changes to the nodes.

- The Host overlay is used for configuration that applies to the Warewulf server itself, such as adding entries to `/etc/hosts` or setting up the DHCP service and NFS exports.

System and runtime overlays can be overlayed on top of each other. For example, instead of altering a configuration setting in an overlay, you can override it with a new overlay. You can set a list of system and runtime overlays to apply to individual nodes, or to multiple nodes via profiles.

3.2 Deploying compute nodes with Warewulf

REQUIREMENTS

- The Warewulf server has a static IP address.
- The compute nodes are set to PXE boot.
- The Warewulf server is accessible from an external network, but is connected to the compute nodes via an internal cluster network used for deployment. This is important because Warewulf configures DHCP and TFTP on the Warewulf server, which might conflict with DHCP on the external network.

PROCEDURE 3.1: DEPLOYING COMPUTE NODES WITH WAREWULF

1. On the Warewulf server, install Warewulf:

   ```
   > sudo zypper install warewulf4
   ```

2. The installation creates a basic configuration for the Warewulf server in the file `/etc/warewulf/warewulf.conf`. Review this file to make sure the details are correct. In particular, check the following settings:

   ```
   ipaddr: 192.168.1.250
   netmask: 255.255.255.0
   network: 192.168.1.0
   ```

   `ipaddr` is the IP address of the Warewulf server on the internal cluster network to be used for node deployment. `netmask` and `network` must match this network.
   Additionally, check that the DHCP range is in the cluster network:

   ```
   dhcp:
     range start: 192.168.1.21
     range end: 192.168.1.50
   ```

3. In the file `/etc/sysconfig/dhcpd`, check that `DHCPD_INTERFACE` has the correct value. This must be the interface on which the cluster network is running.

4. Start and enable the Warewulf service:

   ```
   > sudo systemctl enable --now warewulfd
   ```

5. Configure the services required by Warewulf:

   ```
   > sudo wwctl configure --all
   ```

This command performs the following tasks:

- Configures DHCP and enables the DHCP service.
- Writes the required PXE files to the TFTP root directory and enables the TFTP service.
- Updates the `/etc/hosts` file.
- Configures an NFS server on the Warewulf server and enables the NFS service.
- Creates host keys and user keys for passwordless SSH access to the nodes.

6. When the configuration is finished, log out of the Warewulf server and back into it. This creates an SSH key pair to allow passwordless login to the deployed compute nodes. If you require a password to secure the private key, set it now:

   ```
   > ssh-keygen -p -f $HOME/.ssh/cluster
   ```

7. Importing the Warewulf container from the SUSE registry requires SUSE Customer Center credentials. Set your SCC credentials as environment variables before you import the container:

   ```
   > export WAREWULF_OCI_USERNAME=USER@EXAMPLE.COM
   > export WAREWULF_OCI_PASSWORD=REGISTRATION_CODE
   ```

8. Import the Warewulf container from the SUSE registry:

   ```
   > sudo wwctl container import \
   docker://registry.suse.com/suse/hpc/warewulf4-x86_64/sle-hpc-node:15.5 \
   hpcnode15.5 --setdefault
   ```

   The `--setdefault` argument sets this as the default container in the `default` node profile.

9. Configure the networking details for the `default` profile:

   ```
   > sudo wwctl profile set -y default --netname default \
   --netmask 255.255.255.0 --gateway 192.168.1.250
   ```

   To see the details of this profile, run the following command:

   ```
   > sudo wwctl profile list -a default
   ```

10. Add compute nodes to Warewulf. For example, to add ten discoverable nodes with preconfigured IP addresses, run the following command:

    ```
    > sudo wwctl node add node[01-10] \ ❶
    --netdev eth0 -I 192.168.1.100 \ ❷
    --discoverable=true ❸
    ```

    ❶ One or more node names. Node names must be unique. If you have node groups or multiple clusters, add descriptors to the node names, for example `node01.cluster01`.

    ❷ The IP address for the first node. Subsequent nodes are given incremental IP addresses.

    ❸ Allows Warewulf to assign a MAC address to the nodes when they boot for the first time.

    To view the settings for these nodes, run the following command:

    ```
    > sudo wwctl node list -a node[01-10]
    ```

11. Add the nodes to the `/etc/hosts` file:

    ```
    > sudo wwctl configure hostfile
    ```

12. Rebuild the container image to make sure it is ready to use:

    ```
    > sudo wwctl container build hpcnode15.5
    ```

13. Build the default system and runtime overlays:

    ```
    > sudo wwctl overlay build
    ```

    This command compiles overlays for all the nodes.

You can now boot the compute nodes with PXE. Warewulf provides all the required information.

3.3 Advanced Warewulf tasks

3.3.1 Using Warewulf with UEFI Secure Boot

To boot compute nodes with UEFI Secure Boot enabled, the packages `shim` and `grub2-x86_64-efi` must be installed in the Warewulf container. For the container you imported in *Procedure 3.1, "Deploying compute nodes with Warewulf"*, this should already be the default. Use the following procedure to verify that the packages are installed:

PROCEDURE 3.2: VERIFYING PACKAGES IN A WAREWULF CONTAINER

1. Open a shell in the Warewulf container:

   ```
   > sudo wwctl container shell hpcnode15.5
   ```

2. Search for the packages and check their installation status in the `S` column:

   ```
   [hpcnode15.5] Warewulf> zypper search shim grub2
   ```

   If `shim` and `grub2-x86_64-efi` are not installed, install them now:

   ```
   [hpcnode15.5] Warewulf> zypper install shim grub2-x86_64-efi
   ```

3. Exit the container's shell:

   ```
   [hpcnode15.5] Warewulf> exit
   ```

   If any changes were made, Warewulf automatically rebuilds the container.

4. We recommend rebuilding the container again manually to make sure the changes are applied:

   ```
   > sudo wwctl container build hpcnode15.5
   ```

By default, Warewulf boots nodes via iPXE, which cannot be used when UEFI Secure Boot is enabled. Use the following procedure to switch to GRUB 2 as the boot method:

PROCEDURE 3.3: CONFIGURING WAREWULF TO BOOT VIA GRUB 2

1. Open the file `/etc/warewulf/warewulf.conf` and change the value of `grubboot` to `true`:

   ```
   warewulf:
   ```

```
[...]
grubboot: true
```

2. Reconfigure DHCP and TFTP to recognize the configuration change:

   ```
   > sudo wwctl configure dhcp
   > sudo wwctl configure tftp
   ```

3. Rebuild the system and runtime overlays:

   ```
   > sudo wwctl overlay build
   ```

3.4 For more information

- Warewulf: https://warewulf.org/docs/development/index.html ↗
- Node profiles: https://warewulf.org/docs/development/contents/profiles.html ↗
- Warewulf overlays: https://warewulf.org/docs/development/contents/overlays.html ↗
- The node provisioning process: https://warewulf.org/docs/development/contents/provisioning.html ↗

4 Remote administration

High Performance Computing clusters usually consist of a small set of identical compute nodes. However, large clusters could consist of thousands of machines. This chapter describes tools to help manage the compute nodes in a cluster.

4.1 Genders — static cluster configuration database

Genders is a static cluster configuration database used for configuration management. It allows grouping and addressing sets of nodes by attributes, and is used by a variety of tools. The Genders database is a text file that is usually replicated on each node in a cluster.

Perl, Python, Lua, C, and C++ bindings are supplied with Genders. Each package provides **`man`** pages or other documentation which describes the APIs.

4.1.1 Genders database format

The Genders database in SUSE Linux Enterprise High Performance Computing is a plain-text file called `/etc/genders`. It contains a list of node names with their attributes. Each line of the database can have one of the following formats.

```
nodename                 attr[=value],attr[=value],...
nodename1,nodename2,...  attr[=value],attr[=value],...
nodenames[A-B]           attr[=value],attr[=value],...
```

Node names are listed without their domain, and are followed by any number of spaces or tabs, then the comma-separated list of attributes. Every attribute can optionally have a value. The substitution string `%n` can be used in an attribute value to represent the node name. Node names can be listed on multiple lines, so a node's attributes can be specified on multiple lines. However, no single node can have duplicate attributes.

The attribute list must not contain spaces, and there is no provision for continuation lines. Commas and equals characters (`=`) are special, and cannot appear in attribute names or values. Comments are prefixed with the hash character (`#`) and can appear anywhere in the file.

Ranges for node names can be specified in the form `prefix[a-c,n-p,...]` as an alternative to explicit lists of node names. For example, `node[01-03,06]` would specify `node01`, `node02`, `node03`, and `node06`.

4.1.2 Nodeattr usage

The command line utility **nodeattr** can be used to query data in the genders file. When the genders file is replicated on all nodes, a query can be done without network access. The genders file can be called as follows:

```
> nodeattr [-q | -n | -s] [-r] attr[=val]
```

`-q` is the default option and prints a list of nodes with `attr[=val]`.

The `-c` or `-s` options give a comma-separated or space-separated list of nodes with `attr[=val]`.

If none of the formatting options are specified, **nodeattr** returns a zero value if the local node has the specified attribute, and non-zero otherwise. The `-v` option causes any value associated with the attribute to go to `stdout`. If a node name is specified before the attribute, the specified node is queried instead of the local node.

To print all attributes for a particular node, run the following command:

```
> nodeattr -l [node]
```

If no node parameter is given, all attributes of the local node are printed.

To perform a syntax check of the genders database, run the following command:

```
> nodeattr [-f genders] -k
```

To specify an alternative database location, use the option `-f`.

4.2 pdsh — parallel remote shell program

pdsh is a parallel remote shell that can be used with multiple back-ends for remote connections. It can run a command on multiple machines in parallel.

To install `pdsh`, run the command **zypper in pdsh**.

In SUSE Linux Enterprise High Performance Computing, the back-ends `ssh`, `mrsh`, and `exec` are supported. The `ssh` back-end is the default. Non-default login methods can be used by setting the `PDSH_RCMD_TYPE` environment variable, or by using the `-R` command argument.

When using the `ssh` back-end, you must use a non-interactive (passwordless) login method.

The `mrsh` back-end requires the `mrshd` daemon to be running on the client. The `mrsh` back-end does not require the use of reserved sockets, so it does not suffer from port exhaustion when running commands on many machines in parallel. For information about setting up the system to use this back-end, see *Section 4.5, “mrsh/mrlogin — remote login using MUNGE authentication”*.

Remote machines can be specified on the command line, or **`pdsh`** can use a `machines` file (`/etc/pdsh/machines`), **`dsh`** (Dancer's shell)-style groups or netgroups. It can also target nodes based on the currently running Slurm jobs.

The different ways to select target hosts are realized by modules. Some of these modules provide identical options to **`pdsh`**. The module loaded first will win and handle the option. Therefore, we recommended using a single method and specifying this with the `-M` option.

The `machines` file lists all target hosts, one per line. The appropriate netgroup can be selected with the `-g` command line option.

The following host-list plugins for **`pdsh`** are supported: `machines`, `slurm`, `netgroup` and `dshgroup`. Each host-list plugin is provided in a separate package. This avoids conflicts between command line options for different plugins which happen to be identical, and helps to keep installations small and free of unneeded dependencies. Package dependencies have been set to prevent the installation of plugins with conflicting command options. To install one of the plugins, run:

```
> sudo zypper in pdsh-PLUGIN_NAME
```

For more information, see the **`man`** page **`pdsh`**.

4.3 PowerMan — centralized power control for clusters

PowerMan can control the following remote power control devices (RPC) from a central location:

- local devices connected to a serial port
- RPCs listening on a TCP socket
- RPCs that are accessed through an external program

The communication to RPCs is controlled by “expect”-like scripts. For a list of currently supported devices, see the configuration file `/etc/powerman/powerman.conf`.

To install PowerMan, run **`zypper in powerman`**.

To configure PowerMan, include the appropriate device file for your RPC (`/etc/powerman/*.dev`) in `/etc/powerman/powerman.conf` and add devices and nodes. The device "type" needs to match the "specification" name in one of the included device files. The list of "plugs" used for nodes needs to match an entry in the "plug name" list.

After configuring PowerMan, start its service:

```
> sudo systemctl start powerman.service
```

To start PowerMan automatically after every boot, run the following command:

```
> sudo systemctl enable powerman.service
```

Optionally, PowerMan can connect to a remote PowerMan instance. To enable this, add the option `listen` to `/etc/powerman/powerman.conf`.

Important: Unencrypted transfer

When connecting to a remote PowerMan instance, data is transferred unencrypted. Therefore, use this feature only if the network is appropriately secured.

4.4 MUNGE authentication

MUNGE allows for secure communications between different machines that share the same secret key. The most common use case is the Slurm workload manager, which uses MUNGE for the encryption of its messages. Another use case is authentication for the parallel shell mrsh.

4.4.1 Setting up MUNGE authentication

MUNGE uses UID/GID values to uniquely identify and authenticate users, so you must ensure that users who will authenticate across a network have matching UIDs and GIDs across all nodes.

MUNGE credentials have a limited time-to-live, so you must ensure that the time is synchronized across the entire cluster.

MUNGE is installed with the command **`zypper in munge`**. This also installs further required packages. A separate `munge-devel` package is available to build applications that require MUNGE authentication.

When installing the `munge` package, a new key is generated on every system. However, the entire cluster needs to use the same MUNGE key. Therefore, you must securely copy the MUNGE key from one system to all the other nodes in the cluster. You can accomplish this by using **`pdsh`** with SSH. Ensure that the key is only readable by the `munge` user (permissions mask `0400`).

PROCEDURE 4.1: SETTING UP MUNGE AUTHENTICATION

1. On the server where MUNGE is installed, check the permissions, owner, and file type of the key file `/etc/munge/munge.key`:

   ```
   > sudo stat --format "%F %a %G %U %n" /etc/munge/munge.key
   ```

   The settings should be as follows:

   ```
   400 regular file munge munge /etc/munge/munge.key
   ```

2. Calculate the MD5 sum of `munge.key`:

   ```
   > sudo md5sum /etc/munge/munge.key
   ```

3. Copy the key to the listed nodes using **`pdcp`**:

   ```
   > pdcp -R ssh -w NODELIST /etc/munge/munge.key /etc/munge/munge.key
   ```

4. Check the key settings on the remote nodes:

   ```
   > pdsh -R ssh -w HOSTLIST stat --format \"%F %a %G %U %n\" /etc/munge/munge.key
   > pdsh -R ssh -w HOSTLIST md5sum /etc/munge/munge.key
   ```

   Ensure that they match the settings on the MUNGE server.

4.4.2 Enabling and starting MUNGE

`munged` must be running on all nodes that use MUNGE authentication. If MUNGE is used for authentication across the network, it needs to run on each side of the communications link.

To start the service and ensure it is started after every reboot, run the following command on each node:

```
> sudo systemctl enable --now munge.service
```

You can also use **`pdsh`** to run this command on multiple nodes at once.

4.5 mrsh/mrlogin — remote login using MUNGE authentication

mrsh is a set of remote shell programs using the *MUNGE* authentication system instead of reserved ports for security.

It can be used as a drop-in replacement for `rsh` and `rlogin`.

To install mrsh, do the following:

- If only the mrsh client is required (without allowing remote login to this machine), use: **`zypper in mrsh`**.
- To allow logging in to a machine, the server must be installed: **`zypper in mrsh-server`**.
- To get a drop-in replacement for **`rsh`** and **`rlogin`**, run: **`zypper in mrsh-rsh-server-compat`** or **`zypper in mrsh-rsh-compat`**.

To set up a cluster of machines allowing remote login from each other, first follow the instructions for setting up and starting MUNGE authentication in *Section 4.4, "MUNGE authentication"*. After the MUNGE service successfully starts, enable and start **`mrlogin`** on each machine on which the user will log in:

```
> sudo systemctl enable mrlogind.socket mrshd.socket
> sudo systemctl start mrlogind.socket mrshd.socket
```

To start mrsh support at boot, run the following command:

```
> sudo systemctl enable munge.service
> sudo systemctl enable mrlogin.service
```

We do not recommend using mrsh when logged in as the user `root`. This is disabled by default. To enable it anyway, run the following command:

```
> sudo echo "mrsh" >> /etc/securetty
> sudo echo "mrlogin" >> /etc/securetty
```

5 Hardware

This chapter describes tools that can be used to obtain hardware infrastructure information for HPC applications.

5.1 cpuid

`cpuid` executes the x86 CPUID instruction and decodes and prints the results to `stdout`. Its knowledge of Intel, AMD and Cyrix CPUs is fairly complete. It specifically targets the Intel Xeon Phi architecture.

To install **`cpuid`**, run **`zypper in cpuid`**.

For information about runtime options for **`cpuid`**, see the man page `cpuid(1)`.

Note that this tool is only available for x86-64.

5.2 hwloc — portable abstraction of hierarchical architectures for high-performance computing

`hwloc` provides CLI tools and a C API to obtain the hierarchical map of key computing elements, such as NUMA memory nodes, shared caches, processor packages, processor cores, processing units (logical processors or “threads”), and I/O devices. `hwloc` also gathers various attributes such as cache and memory information, and is portable across a variety of different operating systems and platforms. It can also assemble the topologies of multiple machines into a single one, so that applications can read the topology of an entire fabric or cluster at once.

`lstopo` allows the user to obtain the topology of a machine or convert topology information obtained on a remote machine into one of several output formats. In graphical mode (X11), it displays the topology in a window. Other available formats include plain text, PDF, PNG, SVG and FIG. For more information, see the man pages provided by `hwloc` and `lstopo`.

`hwloc` features full support for import and export of XML-formatted topology files via the `libxml2` library.

The package `hwloc-devel` offers a library that can be directly included into external programs. This requires that the `libxml2` development library (package `libxml2-devel`) is available when compiling `hwloc`.

6 Slurm — utility for HPC workload management

Slurm is a workload manager for managing compute jobs on High Performance Computing clusters. It can start multiple jobs on a single node, or a single job on multiple nodes. Additional components can be used for advanced scheduling and accounting.

The mandatory components of Slurm are the control daemon *slurmctld*, which handles job scheduling, and the Slurm daemon *slurmd*, responsible for launching compute jobs. Nodes running **`slurmctld`** are called *management servers* and nodes running **`slurmd`** are called *compute nodes*.

Additional components are a secondary *slurmctld* acting as a standby server for a failover, and the Slurm database daemon *slurmdbd*, which stores the job history and user hierarchy.

For further documentation, see the Quick Start Administrator Guide (https://slurm.schedmd.com/quickstart_admin.html) ↗ and Quick Start User Guide (https://slurm.schedmd.com/quickstart.html) ↗. There is further in-depth documentation on the Slurm documentation page (https://slurm.schedmd.com/documentation.html) ↗.

6.1 Installing Slurm

These instructions describe a minimal installation of Slurm with one management server and multiple compute nodes.

6.1.1 Minimal installation

! **Important: Make sure of consistent UIDs and GIDs for Slurm's accounts**

For security reasons, Slurm does not run as the user `root`, but under its own user. It is important that the user `slurm` has the same UID/GID across all nodes of the cluster.

If this user/group does not exist, the package `slurm` creates this user and group when it is installed. However, this does not guarantee that the generated UIDs/GIDs will be identical on all systems.

Therefore, we strongly advise you to create the user/group `slurm` before installing `slurm`. If you are using a network directory service such as LDAP for user and group management, you can use it to provide the `slurm` user/group as well.

It is strongly recommended that all compute nodes share common user home directories. These should be provided through network storage.

PROCEDURE 6.1: INSTALLING THE SLURM PACKAGES

1. On the management server, install the `slurm` package with the command **`zypper in slurm`**.

2. On the compute nodes, install the `slurm-node` package with the command **`zypper in slurm-node`**.

3. On the management server and the compute nodes, the package `munge` is installed automatically. Configure, enable and start MUNGE on the management server and compute nodes as described in *Section 4.4, "MUNGE authentication"*. Ensure that the same `munge` key is shared across all nodes.

Note: Automatically opened ports

Installing the `slurm` package automatically opens the TCP ports 6817, 6818, and 6819.

PROCEDURE 6.2: CONFIGURING SLURM

1. On the management server, edit the main configuration file `/etc/slurm/slurm.conf`:

   a. Configure the parameter `SlurmctldHost=`*`SLURMCTLD_HOST`* with the host name of the management server.
   To find the correct host name, run **`hostname -s`** on the management server.

   b. Under the `COMPUTE NODES` section, add the following lines to define the compute nodes:

   ```
   NodeName=NODE_LIST State=UNKNOWN
   PartitionName=normal Nodes=NODE_LIST Default=YES MaxTime=24:00:00 State=UP
   ```

Replace *NODE_LIST* with the host names of the compute nodes, either comma-separated or as a range (for example: node[1-100]).
The NodeName line also allows specifying additional parameters for the nodes, such as Boards , SocketsPerBoard CoresPerSocket , ThreadsPerCore , or CPU . The actual values of these can be obtained by running the following command on the compute nodes:

```
node1# slurmd -C
```

2. Copy the modified configuration file /etc/slurm/slurm.conf from the management server to all compute nodes:

```
management# scp /etc/slurm/slurm.conf COMPUTE_NODE:/etc/slurm/
```

3. On the management server, start slurmctld and enable it so that it starts on every boot:

```
management# systemctl enable --now slurmctld.service
```

4. On each compute node, start slurmd and enable it so that it starts on every boot:

```
node1# systemctl enable --now slurmd.service
```

PROCEDURE 6.3: TESTING THE SLURM INSTALLATION

1. Check the status and availability of the compute nodes by running the **sinfo** command. You should see output similar to the following:

```
PARTITION AVAIL  TIMELIMIT  NODES  STATE NODELIST
normal*      up 1-00:00:00      2   idle node[01-02]
```

If the node state is not idle , see *Section 6.4, “Frequently asked questions”*.

2. Test the Slurm installation by running the following command:

```
management# srun sleep 30
```

This runs the **sleep** command on a free compute node for 30 seconds.
In another shell, run the **squeue** command during the 30 seconds that the compute node is asleep. You should see output similar to the following:

```
JOBID PARTITION     NAME     USER ST       TIME  NODES NODELIST(REASON)
    1     normal    sleep     root  R       0:05      1 node02
```

3. Create the following shell script and save it as `sleeper.sh`:

   ```
   #!/bin/bash
   echo "started at $(date)"
   sleep 30
   echo "finished at $(date)"
   ```

   Run the shell script in the queue:

   ```
   management# sbatch sleeper.sh
   ```

   The shell script is executed when enough resources are available, and the output is stored in the file `slurm-${JOBNR}.out`.

6.1.2 Installing the Slurm database

In a minimal installation, Slurm only stores pending and running jobs. To store finished and failed job data, the storage plugin must be installed and enabled. You can also enable *completely fair scheduling*, which replaces FIFO (first in, first out) scheduling with algorithms that calculate the job priority in a queue in dependence of the job which a user has run in the history.

The Slurm database has two components: the `slurmdbd` daemon itself, and an SQL database. MariaDB is recommended. The database can be installed on the same node that runs `slurmdbd`, or on a separate node. For a minimal setup, all these services run on the management server.

Before you begin, make sure Slurm is installed as described in *Section 6.1.1, “Minimal installation”*.

PROCEDURE 6.4: INSTALL `slurmdbd`

Note: MariaDB

If you want to use an external SQL database (or you already have a database installed on the management server), you can skip *Step 1* and *Step 2*.

1. Install the MariaDB SQL database:

   ```
   management# zypper install mariadb
   ```

2. Start and enable MariaDB:

   ```
   management# systemctl enable --now mariadb
   ```

3. Secure the database:

   ```
   management# mysql_secure_installation
   ```

4. Connect to the SQL database:

   ```
   management# mysql -u root -p
   ```

5. Create the Slurm database user and grant it permissions for the Slurm database, which will be created later:

   ```
   mysql> create user 'slurm'@'localhost' identified by 'PASSWORD';
   mysql> grant all on slurm_acct_db.* TO 'slurm'@'localhost';
   ```

   You can choose to use a different user name or database name. In this case, you must also change the corresponding values in the `/etc/slurm/slurmdbd.conf` file later.

6. Exit the database:

   ```
   mysql> exit
   ```

7. Install the `slurmdbd` package:

   ```
   management# zypper in slurm-slurmdbd
   ```

8. Edit the `/etc/slurm/slurmdbd.conf` file so that the daemon can access the database. Change the following line to the password that you used in *Step 5*:

   ```
   StoragePass=password
   ```

   If the database is on a different node, or if you chose a different user name or database name, you must also modify the following lines:

   ```
   StorageUser=slurm
   StorageLoc=slurm_acct_db
   DbdAddr=localhost
   DbdHost=localhost
   ```

9. Start and enable `slurmdbd`:

   ```
   management# systemctl enable --now slurmdbd
   ```

   The first start of `slurmdbd` might take some time.

10. To enable accounting, edit the `/etc/slurm/slurm.conf` file to add the connection between `slurmctld` and the `slurmdbd` daemon. Ensure that the following lines appear as shown:

    ```
    JobAcctGatherType=jobacct_gather/linux
    JobAcctGatherFrequency=30
    AccountingStorageType=accounting_storage/slurmdbd
    AccountingStorageHost=localhost
    ```

    This example assumes that `slurmdbd` is running on the same node as `slurmctld`. If not, change `localhost` to the host name or IP address of the node where `slurmdbd` is running.

11. Make sure `slurmdbd` is running before you continue:

    ```
    management# systemctl status slurmdbd
    ```

    If you restart `slurmctld` before `slurmdbd` is running, `slurmctld` fails because it cannot connect to the database.

12. Restart `slurmctld`:

    ```
    management# systemctl restart slurmctld
    ```

    This creates the Slurm database and adds the cluster to the database (using the `ClusterName` from `/etc/slurm/slurm.conf`).

13. *(Optional)* By default, Slurm does not take any group membership into account, and the system groups cannot be mapped to Slurm. However, you can mimic system groups with *accounts*. In Slurm, accounts are usually entities billed for cluster usage, while *users* identify individual cluster users. Multiple users can be associated with a single account.
    The following example creates an umbrella group `bavaria` for two subgroups called `nuremberg` and `munich`:

    ```
    management# sacctmgr add account bavaria \
    Description="umbrella group for subgroups" Organization=bavaria
    ```

    ```
    management# sacctmgr add account nuremberg,munich parent=bavaria \
    Description="subgroup" Organization=bavaria
    ```

The following example adds a user called `tux` to the subgroup `nuremberg`:

```
management# sacctmgr add user tux Account=nuremberg
```

6.2 Slurm administration commands

This section lists some useful options for common Slurm commands. For more information and a full list of options, see the **`man`** page for each command. For more Slurm commands, see https://slurm.schedmd.com/man_index.html ↗.

6.2.1 scontrol

The command **`scontrol`** is used to show and update the entities of Slurm, such as the state of the compute nodes or compute jobs. It can also be used to reboot or to propagate configuration changes to the compute nodes.

Useful options for this command are `--details`, which prints more verbose output, and `--oneliner`, which forces the output onto a single line, which is more useful in shell scripts.

For more information, see **`man scontrol`**.

`scontrol show` *`ENTITY`*

Displays the state of the specified *`ENTITY`*.

`scontrol update` *`SPECIFICATION`*

Updates the *`SPECIFICATION`* like the compute node or compute node state. Useful *`SPECIFICATION`* states that can be set for compute nodes include:

`nodename=`*`NODE`* **`state=down reason=`***`REASON`*

Removes all jobs from the compute node, and aborts any jobs already running on the node.

`nodename=`*`NODE`* **`state=drain reason=`***`REASON`*

Drains the compute node so that no *new* jobs can be scheduled on it, but does not end compute jobs already running on the compute node. *`REASON`* can be any string. The compute node stays in the `drained` state and must be returned to the `idle` state manually.

`nodename=`*`NODE`* `state=resume`

Marks the compute node as ready to return to the `idle` state.

`jobid=`*`JOBID REQUIREMENT=VALUE`*

Updates the given requirement, such as `NumNodes`, with a new value. This command can also be run as a non-privileged user.

`scontrol reconfigure`

Triggers a reload of the configuration file `slurm.conf` on all compute nodes.

`scontrol reboot` *`NODELIST`*

Reboots a compute node, or group of compute nodes, when the jobs on it finish. To use this command, the option `RebootProgram="/sbin/reboot"` must be set in `slurm.conf`. When the reboot of a compute node takes more than 60 seconds, you can set a higher value in `slurm.conf`, such as `ResumeTimeout=300`.

6.2.2 sinfo

The command **`sinfo`** retrieves information about the state of the compute nodes, and can be used for a fast overview of the cluster health. For more information, see **`man sinfo`**.

`--dead`

Displays information about unresponsive nodes.

`--long`

Shows more detailed information.

`--reservation`

Prints information about advanced reservations.

`-R`

Displays the reason a node is in the `down`, `drained`, or `failing` state.

`--state=`*`STATE`*

Limits the output only to nodes with the specified *`STATE`*.

6.2.3 sacctmgr and sacct

These commands are used for managing accounting. For more information, see **`man sacctmgr`** and **`man sacct`**.

`sacctmgr`

Used for job accounting in Slurm. To use this command, the service `slurmdbd` must be set up. See *Section 6.1.2, “Installing the Slurm database”*.

`sacct`

Displays the accounting data if accounting is enabled.

`--allusers`

Shows accounting data for all users.

`--accounts`=*`NAME`*

Shows only the specified user(s).

`--starttime`=*`MM/DD[/YY]-HH:MM[:SS]`*

Shows only jobs after the specified start time. You can use just *`MM/DD`* or *`HH:MM`*. If no time is given, the command defaults to `00:00`, which means that only jobs from today are shown.

`--endtime`=*`MM/DD[/YY]-HH:MM[:SS]`*

Accepts the same options as **`--starttime`**. If no time is given, the time when the command was issued is used.

`--name`=*`NAME`*

Limits output to jobs with the given *`NAME`*.

`--partition`=*`PARTITION`*

Shows only jobs that run in the specified *`PARTITION`*.

6.2.4 sbatch, salloc, and srun

These commands are used to schedule *compute jobs*, which means batch scripts for the **`sbatch`** command, interactive sessions for the **`salloc`** command, or binaries for the **`srun`** command. If the job cannot be scheduled immediately, only **`sbatch`** places it into the queue.

For more information, see **`man sbatch`**, **`man salloc`**, and **`man srun`**.

`-n` *`COUNT_THREADS`*

Specifies the number of threads needed by the job. The threads can be allocated on different nodes.

`-N` *`MINCOUNT_NODES[-MAXCOUNT_NODES]`*

Sets the number of compute nodes required for a job. The *`MAXCOUNT_NODES`* number can be omitted.

`--time` *`TIME`*

Specifies the maximum clock time (runtime) after which a job is terminated. The format of *`TIME`* is either seconds or *`[HH:]MM:SS`*. Not to be confused with **`walltime`**, which is `clocktime × threads`.

`--signal` *`[B:]NUMBER[@TIME]`*

Sends the signal specified by *`NUMBER`* 60 seconds before the end of the job, unless *`TIME`* is specified. The signal is sent to every process on every node. If a signal should only be sent to the controlling batch job, you must specify the **`B:`** flag.

`--job-name` *`NAME`*

Sets the name of the job to *`NAME`* in the queue.

`--array=`*`RANGEINDEX`*

Executes the given script via **`sbatch`** for indexes given by *`RANGEINDEX`* with the same parameters.

`--dependency=`*`STATE:JOBID`*

Defers the job until the specified *`STATE`* of the job *`JOBID`* is reached.

`--gres=`*`GRES`*

Runs a job only on nodes with the specified *generic resource* (GRes), for example a GPU, specified by the value of *`GRES`*.

`--licenses=`*`NAME[:COUNT]`*

The job must have the specified number (*`COUNT`*) of licenses with the name *`NAME`*. A license is the opposite of a generic resource: it is not tied to a computer, but is a cluster-wide variable.

`--mem=`*`MEMORY`*

Sets the real *`MEMORY`* required by a job per node. To use this option, memory control must be enabled. The default unit for the *`MEMORY`* value is megabytes, but you can also use `K` for kilobyte, `M` for megabyte, `G` for gigabyte, or `T` for terabyte.

`--mem-per-cpu=`*`MEMORY`*

This option takes the same values as **`--mem`**, but defines memory on a per-CPU basis rather than a per-node basis.

6.3 Upgrading Slurm

For existing products under general support, version upgrades of Slurm are provided regularly. Unlike maintenance updates, these upgrades are not installed automatically using `zypper patch` but require you to request their installation explicitly. This ensures that these upgrades are not installed unintentionally and gives you the opportunity to plan version upgrades beforehand.

Important: **`zypper up`** is not recommended

On systems running Slurm, updating packages with **`zypper up`** is not recommended. **`zypper up`** attempts to update all installed packages to the latest version, so might install a new major version of Slurm outside of planned Slurm upgrades.

Use **`zypper patch`** instead, which only updates packages to the latest bug fix version.

6.3.1 Slurm upgrade workflow

Interoperability is guaranteed between three consecutive versions of Slurm, with the following restrictions:

1. The version of `slurmdbd` must be identical to or higher than the version of `slurmctld`.
2. The version of `slurmctld` must the identical to or higher than the version of `slurmd`.
3. The version of `slurmd` must be identical to or higher than the version of the `slurm` user applications.

Or in short: version(`slurmdbd`) > = version(`slurmctld`) > = version(`slurmd`) > = version (Slurm user CLIs).

Slurm uses a segmented version number: the first two segments denote the major version, and the final segment denotes the patch level. Upgrade packages (that is, packages that were not initially supplied with the module or service pack) have their major version encoded in the package name (with periods `.` replaced by underscores `_`). For example, for version 23.02, this would be `slurm_23_02-*.rpm`. To find out the latest version of Slurm, you can check *My Tools* › *Packages* in the SUSE Customer Center, or run **`zypper search -v slurm`** on a node.

With each version, configuration options for `slurmctld`, `slurmd`, or `slurmdbd` might be deprecated. While deprecated, they remain valid for this version and the two consecutive versions, but they might be removed later. Therefore, it is advisable to update the configuration files after the upgrade and replace deprecated configuration options before the final restart of a service.

A new major version of Slurm introduces a new version of `libslurm`. Older versions of this library might not work with an upgraded Slurm. An upgrade is provided for all SUSE Linux Enterprise software that depends on `libslurm` . It is strongly recommended to rebuild local applications using `libslurm`, such as MPI libraries with Slurm support, as early as possible. This might require updating the user applications, as new arguments might be introduced to existing functions.

Warning: Upgrade `slurmdbd` databases before other Slurm components

If `slurmdbd` is used, always upgrade the `slurmdbd` database *before* starting the upgrade of any other Slurm component. The same database can be connected to multiple clusters and must be upgraded before all of them.

Upgrading other Slurm components before the database can lead to data loss.

6.3.2 Upgrading the `slurmdbd` database daemon

When upgrading `slurmdbd`, the database is converted when the new version of `slurmdbd` starts for the first time. If the database is big, the conversion could take several tens of minutes. During this time, the database is inaccessible.

It is highly recommended to create a backup of the database in case an error occurs during or after the upgrade process. Without a backup, all accounting data collected in the database might be lost if an error occurs or the upgrade is rolled back. A database converted to a newer version cannot be converted back to an older version, and older versions of `slurmdbd` do not recognize the newer formats.

Note: Convert primary `slurmdbd` first

If you are using a backup `slurmdbd`, the conversion must be performed on the primary `slurmdbd` first. The backup `slurmdbd` only starts after the conversion is complete.

PROCEDURE 6.5: UPGRADING THE slurmdbd DATABASE DAEMON

1. Stop the `slurmdbd` service:

   ```
   DBnode# rcslurmdbd stop
   ```

   Ensure that `slurmdbd` is not running anymore:

   ```
   DBnode# rcslurmdbd status
   ```

   `slurmctld` might remain running while the database daemon is down. During this time, requests intended for `slurmdbd` are queued internally. The DBD Agent Queue size is limited, however, and should therefore be monitored with `sdiag`.

2. Create a backup of the `slurm_acct_db` database:

   ```
   DBnode# mysqldump -p slurm_acct_db > slurm_acct_db.sql
   ```

   If needed, this can be restored with the following command:

   ```
   DBnode# mysql -p slurm_acct_db < slurm_acct_db.sql
   ```

3. During the database conversion, the variable `innodb_buffer_pool_size` must be set to a value of 128 MB or more. Check the current size:

   ```
   DBnode# echo  'SELECT @@innodb_buffer_pool_size/1024/1024;' | \
   mysql --password --batch
   ```

4. If the value of `innodb_buffer_pool_size` is less than 128 MB, you can change it for the duration of the current session (on `mariadb`):

   ```
   DBnode# echo 'set GLOBAL innodb_buffer_pool_size = 134217728;' | \
   mysql --password --batch
   ```

   Alternatively, to permanently change the size, edit the `/etc/my.cnf` file, set `innodb_buffer_pool_size` to 128 MB, then restart the database:

   ```
   DBnode# rcmysql restart
   ```

5. If you need to update MariaDB, run the following command:

   ```
   DBnode# zypper update mariadb
   ```

Convert the database tables to the new version of MariaDB:

```
DBnode# mysql_upgrade --user=root --password=ROOT_DB_PASSWORD;
```

6. Install the new version of `slurmdbd`:

```
DBnode# zypper install --force-resolution slurm_VERSION-slurmdbd
```

7. Rebuild the database. If you are using a backup `slurmdbd`, perform this step on the primary `slurmdbd` first.

   Because a conversion might take a considerable amount of time, the `systemd` service might time out during the conversion. Therefore, we recommend performing the migration manually by running `slurmdbd` from the command line in the foreground:

```
DBnode# /usr/sbin/slurmdbd -D -v
```

   When you see the following message, you can shut down `slurmdbd` by pressing **Ctrl** – **C** :

```
Conversion done:
success!
```

8. Before restarting the service, remove or replace any deprecated configuration options. Check the deprecated options in the *Release Notes* (https://www.suse.com/releasenotes/x86_64/SLE-HPC/15-SP5).

9. Restart `slurmdbd`:

```
DBnode# systemctl start slurmdbd
```

Note: No daemonization during rebuild

During the rebuild of the Slurm database, the database daemon does not daemonize.

6.3.3 Upgrading `slurmctld` and `slurmd`

After the Slurm database is upgraded, the `slurmctld` and `slurmd` instances can be upgraded. It is recommended to update the management servers and compute nodes all at once. If this is not feasible, the compute nodes (`slurmd`) can be updated on a node-by-node basis. However, the management servers (`slurmctld`) must be updated first.

PREREQUISITES

- *Section 6.3.2, “Upgrading the `slurmdbd` database daemon”*. Upgrading other Slurm components before the database can lead to data loss.
- This procedure assumes that MUNGE authentication is used and that `pdsh`, the `pdsh` Slurm plugin, and `mrsh` can access all of the machines in the cluster. If this is not the case, install `pdsh` by running **`zypper in pdsh-slurm`**.
  If `mrsh` is not used in the cluster, the `ssh` back-end for `pdsh` can be used instead. Replace the option `-R mrsh` with `-R ssh` in the `pdsh` commands below. This is less scalable and you might run out of usable ports.

PROCEDURE 6.6: UPGRADING `slurmctld` AND `slurmd`

1. Back up the configuration file `/etc/slurm/slurm.conf`. Because this file should be identical across the entire cluster, it is sufficient to do so only on the main management server.

2. On the main management server, edit `/etc/slurm/slurm.conf` and set `SlurmdTimeout` and `SlurmctldTimeout` to sufficiently high values to avoid timeouts while `slurmctld` and `slurmd` are down:

   ```
   SlurmctldTimeout=3600
   SlurmdTimeout=3600
   ```

   We recommend at least 60 minutes (`3600`), and more for larger clusters.

3. Copy the updated `/etc/slurm/slurm.conf` from the management server to all nodes:

   a. Obtain the list of partitions in `/etc/slurm/slurm.conf`.

   b. Copy the updated configuration to the compute nodes:

   ```
   management# cp /etc/slurm/slurm.conf /etc/slurm/slurm.conf.update
   management# sudo -u slurm /bin/bash -c 'cat /etc/slurm/slurm.conf.update | \
   pdsh -R mrsh -P PARTITIONS "cat > /etc/slurm/slurm.conf"'
   management# rm /etc/slurm/slurm.conf.update
   ```

   c. Reload the configuration file on all compute nodes:

   ```
   management# scontrol reconfigure
   ```

   d. Verify that the reconfiguration took effect:

   ```
   management# scontrol show config | grep Timeout
   ```

4. Shut down all running `slurmctld` instances, first on any backup management servers, and then on the main management server:

   ```
   management# systemctl stop slurmctld
   ```

5. Back up the `slurmctld` state files. `slurmctld` maintains persistent state information. Almost every major version involves changes to the `slurmctld` state files. This state information is upgraded if the upgrade remains within the supported version range and no data is lost.
   However, if a downgrade is necessary, state information from newer versions is not recognized by an older version of `slurmctld` and is discarded, resulting in a loss of all running and pending jobs. Therefore, back up the old state in case an update needs to be rolled back.

   a. Determine the `StateSaveLocation` directory:

      ```
      management# scontrol show config | grep StateSaveLocation
      ```

   b. Create a backup of the content of this directory. If a downgrade is required, restore the content of the `StateSaveLocation` directory from this backup.

6. Shut down `slurmd` on the compute nodes:

   ```
   management# pdsh -R ssh -P PARTITIONS systemctl stop slurmd
   ```

7. Upgrade `slurmctld` on the main and backup management servers:

   ```
   management# zypper install --force-resolution slurm_VERSION
   ```

Important: Upgrade all Slurm packages at the same time

If any additional Slurm packages are installed, you must upgrade those as well. This includes:

- slurm-pam_slurm
- slurm-sview
- perl-slurm
- slurm-lua

- slurm-torque
- slurm-config-man
- slurm-doc
- slurm-webdoc
- slurm-auth-none
- pdsh-slurm

All Slurm packages must be upgraded at the same time to avoid conflicts between packages of different versions. This can be done by adding them to the `zypper install` command line described above.

8. Upgrade `slurmd` on the compute nodes:

```
management# pdsh -R ssh -P PARTITIONS \
zypper install --force-resolution slurm_VERSION-node
```

Note: Memory size seen by `slurmd` might change on update

Under certain circumstances, the amount of memory seen by `slurmd` might change after an update. If this happens, `slurmctld` puts the nodes in a `drained` state. To check whether the amount of memory seem by `slurmd` changed after the update, run the following command on a single compute node:

```
node1# slurmd -C
```

Compare the output with the settings in `slurm.conf`. If required, correct the setting.

9. Before restarting the service, remove or replace any deprecated configuration options. Check the deprecated options in the *Release Notes* (https://www.suse.com/releasenotes/x86_64/SLE-HPC/15-SP5) ↗.
If you replace deprecated options in the configuration files, these configuration files can be distributed to all management servers and compute nodes in the cluster by using the method described in *Step 3*.

10. Restart `slurmd` on all compute nodes:

    ```
    management# pdsh -R ssh -P PARTITIONS systemctl start slurmd
    ```

11. Restart `slurmctld` on the main and backup management servers:

    ```
    management# systemctl start slurmctld
    ```

12. Check the status of the management servers. On the main and backup management servers, run the following command:

    ```
    management# systemctl status slurmctld
    ```

13. Verify that the services are running without errors. Run the following command to check whether there are any `down`, `drained`, `failing`, or `failed` nodes:

    ```
    management# sinfo -R
    ```

14. Restore the original values of `SlurmdTimeout` and `SlurmctldTimeout` in `/etc/slurm/slurm.conf`, then copy the restored configuration to all nodes by using the method described in *Step 3*.

6.4 Frequently asked questions

1. *How do I change the state of a node from* `down` *to* `up`*?*

   When the `slurmd` daemon on a node does not reboot in the time specified in the `ResumeTimeout` parameter, or the `ReturnToService` was not changed in the configuration file `slurm.conf`, compute nodes stay in the `down` state and must be set back to the `up` state manually. This can be done for the *`NODE`* with the following command:

   ```
   management# scontrol update state=resume NodeName=NODE
   ```

2. *What is the difference between the states* `down` *and* `down*`*?*

   A `*` shown after a status code means that the node is not responding.

   When a node is marked as `down*`, it means that the node is not reachable because of network issues, or that `slurmd` is not running on that node.

   In the `down` state, the node is reachable, but either the node was rebooted unexpectedly, the hardware does not match the description in `slurm.conf`, or a health check was configured with the `HealthCheckProgram`.

3. *How do I get the exact core count, socket number, and number of CPUs for a node?*

   To find the node values that go into the configuration file `slurm.conf`, run the following command:

   ```
   node1# slurmd -C
   ```

7 Monitoring and logging

Obtaining and maintaining an overview over the status and health of a cluster's compute nodes helps to ensure a smooth operation. This chapter describes tools that give an administrator an overview of the current cluster status, collect system logs, and gather information on certain system failure conditions.

7.1 ConMan — the console manager

ConMan is a serial console management program designed to support many console devices and simultaneous users. It supports:

- local serial devices
- remote terminal servers (via the telnet protocol)
- IPMI Serial-Over-LAN (via FreeIPMI)
- Unix domain sockets
- external processes (for example, using **`expect`** scripts for **`telnet`**, **`ssh`**, or **`ipmi-sol`** connections)

ConMan can be used for monitoring, logging, and optionally timestamping console device output.

To install ConMan, run **`zypper in conman`**.

Important: conmand sends unencrypted data

The daemon `conmand` sends unencrypted data over the network and its connections are not authenticated. Therefore, it should be used locally only, listening to the port `localhost`. However, the IPMI console does offer encryption. This makes `conman` a good tool for monitoring many such consoles.

ConMan provides expect-scripts in the directory `/usr/lib/conman/exec`.

Input to `conman` is not echoed in interactive mode. This can be changed by entering the escape sequence `&E`.

When pressing `Enter` in interactive mode, no line feed is generated. To generate a line feed, press `Ctrl`–`L`.

For more information about options, see the ConMan man page.

7.2 Monitoring HPC clusters with Prometheus and Grafana

Monitor the performance of HPC clusters using Prometheus and Grafana.

Prometheus collects metrics from exporters running on cluster nodes and stores the data in a time series database. Grafana provides data visualization dashboards for the metrics collected by Prometheus. Preconfigured dashboards are available on the Grafana website.

The following Prometheus exporters are useful for High Performance Computing:

Slurm exporter

Extracts job and job queue status metrics from the Slurm workload manager. Install this exporter on a node that has access to the Slurm command line interface.

Node exporter

Extracts hardware and kernel performance metrics directly from each compute node. Install this exporter on every compute node you want to monitor.

Important: Restrict access to monitoring data

It is recommended that the monitoring data only be accessible from within a trusted environment (for example, using a login node or VPN). It should not be accessible from the internet without additional security hardening measures for access restriction, access control, and encryption.

MORE INFORMATION

- Grafana: https://grafana.com/docs/grafana/latest/getting-started/ ↗
- Grafana dashboards: https://grafana.com/grafana/dashboards ↗
- Prometheus: https://prometheus.io/docs/introduction/overview/ ↗
- Prometheus exporters: https://prometheus.io/docs/instrumenting/exporters/ ↗

- Slurm exporter: https://github.com/vpenso/prometheus-slurm-exporter ↗
- Node exporter: https://github.com/prometheus/node_exporter ↗

7.2.1 Installing Prometheus and Grafana

Install Prometheus and Grafana on a management server, or on a separate monitoring node.

PREREQUISITES

- You have an installation source for Prometheus and Grafana:
  - The packages are available from SUSE Package Hub. To install SUSE Package Hub, see https://packagehub.suse.com/how-to-use/ ↗.
  - If you have a subscription for SUSE Manager, the packages are available from the SUSE Manager Client Tools repository.

PROCEDURE 7.1: INSTALLING PROMETHEUS AND GRAFANA

In this procedure, replace *MNTRNODE* with the host name or IP address of the server where Prometheus and Grafana are installed.

1. Install the Prometheus and Grafana packages:

```
monitor# zypper in golang-github-prometheus-prometheus grafana
```

2. Enable and start Prometheus:

```
monitor# systemctl enable --now prometheus
```

3. Verify that Prometheus works:

   - In a browser, navigate to `MNTRNODE:9090/config`, or:
   - In a terminal, run the following command:

```
> wget MNTRNODE:9090/config --output-document=-
```

   Either of these methods should show the default contents of the `/etc/prometheus/prometheus.yml` file.

4. Enable and start Grafana:

```
monitor# systemctl enable --now grafana-server
```

5. Log in to the Grafana web server at `MNTRNODE:3000`.
   Use `admin` for both the user name and password, then change the password when prompted.

6. On the left panel, select the gear icon (⚙) and click *Data Sources*.

7. Click *Add data source*.

8. Find Prometheus and click *Select*.

9. In the *URL* field, enter `http://localhost:9090`. The default settings for the other fields can remain unchanged.

   > ! If Prometheus and Grafana are installed on different servers, replace `localhost` with the host name or IP address of the server where Prometheus is installed.

10. Click *Save & Test*.

You can now configure Prometheus to collect metrics from the cluster, and add dashboards to Grafana to visualize those metrics.

7.2.2 Monitoring cluster workloads

To monitor the status of the nodes and jobs in an HPC cluster, install the Prometheus Slurm exporter to collect workload data, then import a custom Slurm dashboard from the Grafana website to visualize the data. For more information about this dashboard, see https://grafana.com/grafana/dashboards/4323.

You must install the Slurm exporter on a node that has access to the Slurm command line interface. In the following procedure, the Slurm exporter will be installed on a management server.

PREREQUISITES

- *Section 7.2.1, “Installing Prometheus and Grafana”* is complete.

- The Slurm workload manager is fully configured.

- You have internet access and policies that allow you to download the dashboard from the Grafana website.

PROCEDURE 7.2: MONITORING CLUSTER WORKLOADS

In this procedure, replace *MGMTSERVER* with the host name or IP address of the server where the Slurm exporter is installed, and replace *MNTRNODE* with the host name or IP address of the server where Grafana is installed.

1. Install the Slurm exporter:

```
management# zypper in golang-github-vpenso-prometheus_slurm_exporter
```

2. Enable and start the Slurm exporter:

```
management# systemctl enable --now prometheus-slurm_exporter
```

Important: Slurm exporter fails when GPU monitoring is enabled

In Slurm 20.11, the Slurm exporter fails when GPU monitoring is enabled.

This feature is disabled by default. Do not enable it for this version of Slurm.

3. Verify that the Slurm exporter works:

- In a browser, navigate to `MNGMTSERVER:8080/metrics`, or:

- In a terminal, run the following command:

```
> wget MGMTSERVER:8080/metrics --output-document=-
```

Either of these methods should show output similar to the following:

```
# HELP go_gc_duration_seconds A summary of the GC invocation durations.
# TYPE go_gc_duration_seconds summary
go_gc_duration_seconds{quantile="0"} 1.9521e-05
go_gc_duration_seconds{quantile="0.25"} 4.5717e-05
go_gc_duration_seconds{quantile="0.5"} 7.8573e-05
...
```

4. On the server where Prometheus is installed, edit the `scrape_configs` section of the `/etc/prometheus/prometheus.yml` file to add a job for the Slurm exporter:

```
  - job_name: slurm-exporter
     scrape_interval: 30s
     scrape_timeout: 30s
     static_configs:
        - targets: ['MGMTSERVER:8080']
```

Set the `scrape_interval` and `scrape_timeout` to `30s` to avoid overloading the server.

5. Restart the Prometheus service:

```
monitor# systemctl restart prometheus
```

6. Log in to the Grafana web server at `MNTRNODE:3000`.

7. On the left panel, select the plus icon (⊞) and click *Import*.

8. In the *Import via grafana.com* field, enter the dashboard ID `4323`, then click *Load*.

9. From the *Select a Prometheus data source* drop-down box, select the Prometheus data source added in *Procedure 7.1, “Installing Prometheus and Grafana”*, then click *Import*.

10. Review the Slurm dashboard. The data might take some time to appear.

11. If you made any changes, click *Save dashboard* when prompted, optionally describe your changes, then click *Save*.

The Slurm dashboard is now available from the *Home* screen in Grafana.

7.2.3 Monitoring compute node performance

To monitor the performance and health of each compute node, install the Prometheus node exporter to collect performance data, then import a custom node dashboard from the Grafana website to visualize the data. For more information about this dashboard, see https://grafana.com/grafana/dashboards/405 ↗.

PREREQUISITES

- *Section 7.2.1, “Installing Prometheus and Grafana”* is complete.
- You have internet access and policies that allow you to download the dashboard from the Grafana website.
- To run commands on multiple nodes at once, **`pdsh`** must be installed on the system your shell is running on, and SSH key authentication must be configured for all of the nodes. For more information, see *Section 4.2, “pdsh — parallel remote shell program”*.

PROCEDURE 7.3: MONITORING COMPUTE NODE PERFORMANCE

In this procedure, replace the example node names with the host names or IP addresses of the nodes, and replace *MNTRNODE* with the host name or IP address of the server where Grafana is installed.

1. Install the node exporter on each compute node. You can do this on multiple nodes at once by running the following command:

```
management# pdsh -R ssh -u root -w "NODE1,NODE2" \
"zypper in -y golang-github-prometheus-node_exporter"
```

2. Enable and start the node exporter. You can do this on multiple nodes at once by running the following command:

```
management# pdsh -R ssh -u root -w "NODE1,NODE2" \
"systemctl enable --now prometheus-node_exporter"
```

3. Verify that the node exporter works:

   - In a browser, navigate to `NODE1:9100/metrics`, or:

   - In a terminal, run the following command:

   ```
   > wget NODE1:9100/metrics --output-document=-
   ```

   Either of these methods should show output similar to the following:

```
# HELP go_gc_duration_seconds A summary of the pause duration of garbage collection
 cycles.
# TYPE go_gc_duration_seconds summary
go_gc_duration_seconds{quantile="0"} 2.3937e-05
go_gc_duration_seconds{quantile="0.25"} 3.5456e-05
go_gc_duration_seconds{quantile="0.5"} 8.1436e-05
...
```

4. On the server where Prometheus is installed, edit the `scrape_configs` section of the `/etc/prometheus/prometheus.yml` file to add a job for the node exporter:

```
  - job_name: node-exporter
    static_configs:
      - targets: ['NODE1:9100']
      - targets: ['NODE2:9100']
```

   Add a target for every node that has the node exporter installed.

5. Restart the Prometheus service:

   ```
   monitor# systemctl restart prometheus
   ```

6. Log in to the Grafana web server at `MNTRNODE:3000`.

7. On the left panel, select the plus icon (⊞) and click *Import*.

8. In the *Import via grafana.com* field, enter the dashboard ID `405`, then click *Load*.

9. From the *Select a Prometheus data source* drop-down box, select the Prometheus data source added in *Procedure 7.1, “Installing Prometheus and Grafana”*, then click *Import*.

10. Review the node dashboard. Click the *node* drop-down box to select the nodes you want to view. The data might take some time to appear.

11. If you made any changes, click *Save dashboard* when prompted. To keep the currently selected nodes next time you access the dashboard, activate *Save current variable values as dashboard default*. Optionally describe your changes, then click *Save*.

The node dashboard is now available from the *Home* screen in Grafana.

7.3 rasdaemon — utility to log RAS error tracings

`rasdaemon` is an RAS (Reliability, Availability and Serviceability) logging tool. It records memory errors using EDAC (Error Detection and Correction) tracing events. EDAC drivers in the Linux kernel handle detection of ECC (Error Correction Code) errors from memory controllers.

`rasdaemon` can be used on large memory systems to track, record, and localize memory errors and how they evolve over time to detect hardware degradation. Furthermore, it can be used to localize a faulty DIMM on the mainboard.

To check whether the EDAC drivers are loaded, run the following command:

```
# ras-mc-ctl --status
```

The command should return `ras-mc-ctl: drivers are loaded`. If it indicates that the drivers are not loaded, EDAC may not be supported on your board.

To start `rasdaemon`, run **`systemctl start rasdaemon.service`**. To start `rasdaemon` automatically at boot time, run **`systemctl enable rasdaemon.service`**. The daemon logs information to `/var/log/messages` and to an internal database. A summary of the stored errors can be obtained with the following command:

```
# ras-mc-ctl --summary
```

The errors stored in the database can be viewed with:

```
# ras-mc-ctl --errors
```

Optionally, you can load the DIMM labels silk-screened on the system board to more easily identify the faulty DIMM. To do so, before starting `rasdaemon`, run:

```
# systemctl start ras-mc-ctl start
```

For this to work, you need to set up a layout description for the board. There are no descriptions supplied by default. To add a layout description, create a file with an arbitrary name in the directory `/etc/ras/dimm_labels.d/`. The format is:

```
Vendor: MOTHERBOARD-VENDOR-NAME
Model: MOTHERBOARD-MODEL-NAME
  LABEL: MC.TOP.MID.LOW
```

8 HPC user libraries

Many HPC clusters need to accommodate multiple compute applications, each of which has its own very specific library dependencies. Multiple instances of the same libraries might exist, differing in version, build configuration, compiler, and MPI implementation. To manage these dependencies, you can use an environment module system. Most HPC libraries provided with SUSE Linux Enterprise High Performance Computing are built with support for environment modules. This chapter describes the environment module system *Lmod*, and a set of HPC compute libraries shipped with SLE HPC.

8.1 Lmod — Lua-based environment modules

Lmod is an advanced environment module system that allows the installation of multiple versions of a program or shared library, and helps configure the system environment for the use of a specific version. It supports hierarchical library dependencies and makes sure that the correct versions of dependent libraries are selected. Environment module-enabled library packages supplied with the HPC module support parallel installation of different versions and flavors of the same library or binary and are supplied with appropriate `lmod` module files.

8.1.1 Installation and basic usage

To install Lmod, run **`zypper in lua-lmod`**.

Before you can use Lmod, you must **`source`** an `init` file into the initialization file of your interactive shell. The following init files are available for various common shells:

```
/usr/share/lmod/lmod/init/bash
/usr/share/lmod/lmod/init/ksh
/usr/share/lmod/lmod/init/tcsh
/usr/share/lmod/lmod/init/zsh
/usr/share/lmod/lmod/init/sh
```

Pick the appropriate file for your shell, then add the following line into your shell's init file:

```
source /usr/share/lmod/lmod/init/INIT-FILE
```

The init script adds the command **`module`**.

8.1.2 Listing available modules

To list all the available modules, run **`module spider`**. To show all modules which can be loaded with the currently loaded modules, run **`module avail`**. A module name consists of a name and a version string, separated by a `/` character. If more than one version is available for a certain module name, the default version is marked by a `*` character. If there is no default, the module with the highest version number is loaded. To reference a specific module version, you can use the full string *`NAME/VERSION`*.

8.1.3 Listing loaded modules

`module list` shows all currently loaded modules. Refer to **`module help`** for some short help on the module command, and **`module help`** *`MODULE-NAME`* for help on the particular module. The **`module`** command is only available when you log in after installing `lua-lmod`.

8.1.4 Gathering information about a module

To get information about a particular module, run **`module whatis`** *`MODULE-NAME`*. To load a module, run **`module load`** *`MODULE-NAME`*. This will ensure that your environment is modified (that is, the `PATH` and `LD_LIBRARY_PATH` and other environment variables are prepended) so that binaries and libraries provided by the respective modules are found. To run a program compiled against this library, the appropriate **`module load`** commands must be issued beforehand.

8.1.5 Loading modules

The **`module load`** *`MODULE`* command must be run in the shell from which the module is to be used. Some modules require a compiler toolchain or MPI flavor module to be loaded before they are available for loading.

8.1.6 Environment variables

If the respective development packages are installed, build-time environment variables like `LIBRARY_PATH`, `CPATH`, `C_INCLUDE_PATH`, and `CPLUS_INCLUDE_PATH` are set up to include the directories containing the appropriate header and library files. However, some compiler and linker commands might not honor these. In this case, use the appropriate options together with the environment variables `-I PACKAGE_NAME_INC` and `-L PACKAGE_NAME_LIB` to add the include and library paths to the command lines of the compiler and linker.

8.1.7 For more information

For more information on Lmod, see https://lmod.readthedocs.org.

8.2 GNU Compiler Toolchain Collection for HPC

In SUSE Linux Enterprise High Performance Computing, the GNU compiler collection version 7 is provided as the base compiler toolchain. The `gnu-compilers-hpc` package provides the environment module for the base version of the GNU compiler suite. This package must be installed when using any of the HPC libraries enabled for environment modules.

8.2.1 Environment module

This package requires `lua-lmod` to supply environment module support.

To install `gnu-compilers-hpc`, run the following command:

```
> sudo zypper in gnu-compilers-hpc
```

To make libraries built with the base compilers available, you must set up the environment appropriately and select the GNU toolchain. To do so, run the following command:

```
> module load gnu
```

8.2.2 Building High Performance Computing software with GNU Compiler Suite

To use the GNU compiler collection to build your own libraries and applications, `gnu-compilers-hpc-devel` must be installed. It ensures that all compiler components required for HPC (that is, C, C++, and Fortran compilers) are installed.

The environment variables `CC`, `CXX`, `FC` and `F77` will be set correctly and the path will be adjusted so that the correct compiler version can be found.

8.2.3 Later versions

The Development Tools Module might provide later versions of the GNU compiler suite. To determine the available compiler suites, run the following command:

```
> zypper search '*-compilers-hpc'
```

If you have more than one version of the compiler suite installed, *Lmod* picks the latest one by default. If you require an older version, or the base version, append the version number:

```
> module load gnu/7
```

For more information, see *Section 8.1, "Lmod — Lua-based environment modules"*.

8.3 High Performance Computing libraries

Library packages that support environment modules follow a distinctive naming scheme. All packages have the compiler suite and, if built with MPI support, the MPI flavor included in their name: `*-[`*`MPI_FLAVOR`*`-]`*`COMPILER`*`-hpc*`. To allow the parallel installation of multiple versions of a library, the package name contains the version number (with dots `.` replaced by underscores `_`). `master-` packages are supplied to ensure that the latest version of a package is installed. When these master packages are updated, the latest version of the respective packages is installed, while leaving previous versions installed. Library packages are split between runtime and compile-time packages. The compile-time packages typically supply `include` files and `.so` files for shared libraries. Compile-time package names end with `-devel`. For some libraries, static (`.a`) libraries are supplied as well. Package names for these end with `-devel-static`.

As an example, these are the package names of the ADIOS library version 1.13.1, built with GCC for Open MPI v4:

- library master package: `adios-gnu-openmpi4-hpc`
- development master package: `adios-gnu-openmpi4-hpc-devel`
- library package: `adios_1_13_1-gnu-openmpi4-hpc`
- development package: `adios_1_13_1-gnu-openmpi4-hpc-devel`
- static library package: `adios_1_13_1-gnu-openmpi4-hpc-devel-static`

To install a library package, run **`zypper in`** *`LIBRARY-MASTER-PACKAGE`*. To install a development file, run **`zypper in`** *`LIBRARY-DEVEL-MASTER-PACKAGE`*.

The GNU compiler collection version 7 as provided with SLE HPC and the MPI flavors Open MPI v.3, Open MPI v.4, MPICH, and MVAPICH2 are currently supported.

The Development Tools Module might provide later versions of the GNU compiler suite. To view available compilers, run the following command:

```
> zypper search '*-compilers-hpc'
```

8.3.1 NumPy Python library

NumPy is a general-purpose array-processing package designed to efficiently manipulate large multi-dimensional arrays of arbitrary records without sacrificing too much speed for small multi-dimensional arrays.

NumPy is built on the Numeric code base and adds features introduced by the discontinued *NumArray* project, as well as an extended C API, and the ability to create arrays of arbitrary type, which also makes NumPy suitable for interfacing with general-purpose database applications.

There are also basic facilities for discrete Fourier transform, basic linear algebra, and random number generation.

This package is available both for Python 2 and 3. The specific compiler toolchain module must be loaded for this library. The correct library module for the Python version used needs to be specified when loading this module. To load this module, run the following command:

```
> module load TOOLCHAIN pythonVERSION-numpy
```

For information about the toolchain to load see: *Section 8.2, “GNU Compiler Toolchain Collection for HPC”*.

List of master packages:

- python*VERSION*-numpy-gnu-hpc
- python*VERSION*-numpy-gnu-hpc-devel

8.3.2 SciPy Python Library

SciPy is a collection of mathematical algorithms and convenience functions built on the NumPy extension of Python. It provides high-level commands and classes for manipulating and visualizing data. With SciPy, an interactive Python session becomes a data-processing and system-prototyping environment.

This package is available both for Python 2 (up to version 1.2.0 only) and 3. The specific compiler toolchain modules must be loaded for this library. The correct library module for the Python version used must be specified when loading this module. To load this module, run the following command:

```
> module load TOOLCHAIN pythonVERSION-scipy
```

For information about the toolchain to load, see *Section 8.2, “GNU Compiler Toolchain Collection for HPC”*.

List of master packages:

- `pythonPYTHON_VERSION-scipy-gnu-hpc`
- `pythonPYTHON_VERSION-scipy-gnu-hpc-devel`

8.3.3 memkind — heap manager for heterogeneous memory platforms and mixed memory policies

The *memkind* library is a user-extensible heap manager built on top of *jemalloc*. It enables control over memory characteristics and a partitioning of the heap between kinds of memory. The kinds of memory are defined by operating system memory policies that have been applied to virtual address ranges. Memory characteristics supported by `memkind` without user extension include control of NUMA and page size features.

For more information, see:

- the man pages `memkind` and `hbwallow`
- https://github.com/memkind/memkind ↗
- https://memkind.github.io/memkind/ ↗

This tool is only available for AMD64/Intel 64.

8.3.4 Support for PMIx in Slurm and MPI libraries

PMIx abstracts the internals of MPI implementations for workload managers and unifies the way MPI jobs are started by the workload manager. With PMIx, there is no need to use the individual MPI launchers on Slurm, because **`srun`** will take care of this. In addition, the workload manager can determine the topology of the cluster, so you do not need to specify topologies manually.

8.3.5 OpenBLAS library — optimized BLAS library

OpenBLAS is an optimized BLAS (Basic Linear Algebra Subprograms) library based on GotoBLAS2 1.3, BSD version. It provides the BLAS API. It is shipped as a package enabled for environment modules, so it requires using Lmod to select a version. There are two variants of this library: an OpenMP-enabled variant, and a `pthreads` variant.

OpenMP-Enabled Variant

The OpenMP variant covers the following use cases:

- **Programs using OpenMP.** This requires the OpenMP-enabled library version to function correctly.
- **Programs using pthreads.** This requires an OpenBLAS library without pthread support. This can be achieved with the OpenMP-version. We recommend limiting the number of threads that are used to 1 by setting the environment variable `OMP_NUM_THREADS=1`.
- **Programs without pthreads and without OpenMP.** Such programs can still take advantage of the OpenMP optimization in the library by linking against the OpenMP variant of the library.

When linking statically, ensure that `libgomp.a` is included by adding the linker flag `-lgomp`.

pthreads Variant

The pthreads variant of the OpenBLAS library can improve the performance of single-threaded programs. The number of threads used can be controlled with the environment variable `OPENBLAS_NUM_THREADS`.

Installation and Usage

This module requires loading a compiler toolchain beforehand. To select the latest version of this module provided, run the following command:

- Standard version:

  ```
  > module load TOOLCHAIN openblas
  ```

- OpenMP/pthreads version:

  ```
  > module load TOOLCHAIN openblas-pthreads
  ```

For information about the toolchain to load, see *Section 8.2, “GNU Compiler Toolchain Collection for HPC”*.

List of master packages:

- `libopenblas-gnu-hpc`
- `libopenblas-gnu-hpc-devel`
- `libopenblas-pthreads-gnu-hpc`
- `libopenblas-pthreads-gnu-hpc-devel`

8.4 File format libraries

8.4.1 HDF5 HPC library — model, library, and file format for storing and managing data

HDF5 is a data model, library, and file format for storing and managing data. It supports an unlimited variety of data types, and is designed for flexible and efficient I/O and for high-volume and complex data. HDF5 is portable and extensible, allowing applications to evolve in their use of HDF5.

There are serial and MPI variants of this library available. All flavors require loading a compiler toolchain module beforehand. The MPI variants also require loading the correct MPI flavor module.

To load the highest available serial version of this module, run the following command:

```
> module load TOOLCHAIN hdf5
```

When an MPI flavor is loaded, you can load the MPI version of this module by running the following command:

```
> module load TOOLCHAIN MPI_FLAVOR phdf5
```

For information about the toolchain to load, see *Section 8.2, "GNU Compiler Toolchain Collection for HPC"*. For information about available MPI flavors, see *Section 8.5, "MPI libraries"*.

List of master packages:

- `hdf5-hpc-examples`
- `hdf5-gnu-hpc-devel`
- `libhdf5-gnu-hpc`
- `libhdf5_cpp-gnu-hpc`
- `libhdf5_fortran-gnu-hpc`
- `libhdf5_hl_cpp-gnu-hpc`
- `libhdf5_hl_fortran-gnu-hpc`
- `hdf5-gnu-MPI_FLAVOR-hpc-devel`
- `libhdf5-gnu-MPI_FLAVOR-hpc`
- `libhdf5_fortran-gnu-MPI_FLAVOR-hpc`
- `libhdf5_hl_fortran-MPI_FLAVOR-hpc`

`MPI_FLAVOR` must be one of the supported MPI flavors described in *Section 8.5, "MPI libraries"*.

For general information about Lmod and modules, see *Section 8.1, "Lmod — Lua-based environment modules"*.

8.5 MPI libraries

Three different implementation of the Message Passing Interface (MPI) standard are provided standard with the HPC module:

- Open MPI (version 3 and version 4)
- MVAPICH2
- MPICH

These packages have been built with full environment module support (LMOD).

The following packages are available:

- For Open MPI:
  - user programs: `openmpi3-gnu-hpc` and `openmpi4-gnu-hpc`
  - shared libraries: `libopenmpi3-gnu-hpc` and `libopenmpi4-gnu-hpc`
  - development libraries, headers and tools required for building: `openmpi3-gnu-hpc-devel` and `openmpi4-gnu-hpc-devel`
  - documentation: `openmpi3-gnu-hpc-docs` and `openmpi4-gnu-hpc-docs`.
- For MVAPICH2
  - user programs and libraries: `mvapich2-gnu-hpc`
  - development libraries, headers and tools for building: `mvapich2-gnu-hpc-devel`
  - documentation: `mvapich2-gnu-hpc-doc`

  For MPICH:

  - user programs and libraries: `mpich-gnu-hpc`
  - development libraries, headers and tools for building: `mpich-gnu-hpc-devel`

The different MPI implementations and versions are independent of each other, and can be installed in parallel.

Use environment modules to pick the version to use:

- For Open MPI v.3:

  ```
  > module load TOOLCHAIN openmpi/3
  ```

- For Open MPI v.4:

  ```
  > module load TOOLCHAIN openmpi/4
  ```

- For MVAPICH2:

  ```
  > module load TOOLCHAIN mvapich2
  ```

- For MPICH:

  ```
  > module load TOOLCHAIN mpich
  ```

For information about the toolchain to load, see *Section 8.2, “GNU Compiler Toolchain Collection for HPC”*.

8.6 Profiling and benchmarking libraries and tools

SUSE Linux Enterprise High Performance Computing provides tools for profiling MPI applications and benchmarking MPI performance.

8.6.1 IMB — Intel* MPI benchmarks

The Intel* MPI Benchmarks package provides a set of elementary benchmarks that conform to the MPI-1, MPI-2, and MPI-3 standards. You can run all of the supported benchmarks, or a subset specified in the command line, using a single executable file. Use command line parameters to specify various settings, such as time measurement, message lengths, and selection of communicators. For details, see the Intel* MPI Benchmarks User's Guide: https://software.intel.com/en-us/imb-user-guide.

For the IMB binaries to be found, a compiler toolchain and an MPI flavor must be loaded beforehand. To load this module, run the following command:

```
> module load TOOLCHAIN MPI_FLAVOR imb
```

For information about the toolchain to load, see *Section 8.2, "GNU Compiler Toolchain Collection for HPC"*. For information on available MPI flavors, see *Section 8.5, "MPI libraries"*.

- `imb-gnu-MPI_FLAVOR-hpc`

8.6.2 PAPI HPC library — consistent interface for hardware performance counters

PAPI provides a tool with a consistent interface and methodology for the performance counter hardware found in most major microprocessors.

This package works with all compiler toolchains and does not require a compiler toolchain to be selected. Load the latest version provided by running the following command:

```
> module load TOOLCHAIN papi
```

For information about the toolchain to load, see *Section 8.2, "GNU Compiler Toolchain Collection for HPC"*.

List of master packages:

- `papi-hpc`
- `papi-hpc-devel`

For general information about Lmod and modules, see *Section 8.1, "Lmod — Lua-based environment modules"*.

8.6.3 mpiP — lightweight MPI profiling library

mpiP is a lightweight profiling library for MPI applications. Because it only collects statistical information about MPI functions, mpiP generates considerably less overhead and much less data than tracing tools. All the information captured by mpiP is task-local. It only uses communication during report generation, typically at the end of the experiment, to merge results from all of the tasks into one output file.

For this library a compiler toolchain and MPI flavor must be loaded beforehand. To load this module, run the following command:

```
> module load TOOLCHAIN MPI_FLAVOR mpip
```

For information about the toolchain to load, see *Section 8.2, “GNU Compiler Toolchain Collection for HPC”*. For information on available MPI flavors, see *Section 8.5, “MPI libraries”*.

List of master packages:

- `mpiP-gnu-MPI_FLAVOR-hpc`
- `mpiP-gnu-MPI_FLAVOR-hpc-devel`
- `mpiP-gnu-MPI_FLAVOR-hpc-doc`

`MPI_FLAVOR` must be one of the supported MPI flavors described in *Section 8.5, “MPI libraries”*.

8.7 Creating environment containers with Apptainer

You can deploy environments with preconfigured environment variables by using *environment containers*. Environment containers include only the components that are part of the environment, plus any required user applications. To create a container from the current HPC environment, use the container platform Apptainer (formerly called Singularity). Apptainer is available from SUSE Package Hub. You can also use Spack to configure the environment to use with Apptainer.

For more information, see the following documentation:

- Enabling the SUSE Package Hub extension: https://packagehub.suse.com/how-to-use/ ↗.
- Using Spack to configure the environment: https://spack.readthedocs.io/en/latest/containers.html# ↗.
- Apptainer documentation: https://apptainer.org/docs/ ↗.

To migrate from Singularity to Apptainer, see https://apptainer.org/docs/admin/latest/singularity_migration.html ↗.

9 Spack package management tool

Spack is a configurable Python-based package manager, automating the installation and fine-tuning of simulations and libraries. Spack can install many variants of the same build using different compilers, options, and MPI implementations. For more information, see the Spack Documentation (https://spack-tutorial.readthedocs.io/en/latest/) ↗.

9.1 Installing Spack

Use this procedure to install Spack on any node in the cluster.

PROCEDURE 9.1: INSTALLING AND CONFIGURING SPACK

1. Install Spack:

   ```
   # zypper in spack
   ```

2. Set up your environment with the appropriate script for your shell:

   - For bash/zsh/sh:

     ```
     # . /usr/share/spack/setup-env.sh
     ```

   - For tcsh/csh:

     ```
     # source /usr/share/spack/setup-env.csh
     ```

   - For fish:

     ```
     # . /usr/share/spack/setup-env.fish
     ```

3. It is recommended to install `bash-completion` so you can use TAB key auto-completion for Spack commands:

   ```
   # zypper in bash-completion
   ```

   ```
   # spack TAB
   activate      clone         dependencies  fetch         list          providers
    solve         url
   ```

```
add           commands      dependents    find          load          pydoc
 spec          verify
arch          compiler      deprecate     flake9        location      python
 stage         versions
blame         compilers     dev-build     gc            log-parse     reindex
 test          view
buildcache    concretize    develop       gpg           maintainers   remove
 test-env
build-env     config        docs          graph         mark          repo
 tutorial
cd            containerize  edit          help          mirror        resource
 undevelop
checksum      create        env           info          module        restage
 uninstall
ci            deactivate    extensions    install       patch         rm
 unit-test
clean         debug         external      license       pkg           setup
 unload
```

9.2 Using Spack: simple example with netcdf-cxx4

This example procedure shows you different ways to build `netcdf-cxx4` with Spack.

PROCEDURE 9.2: BUILDING `netcdf-cxx4` WITH SPACK

1. Show detailed information on `netcdf-cxx4`:

```
# spack info netcdf-cxx4
AutotoolsPackage:   netcdf-cxx4

Description:
    NetCDF (network Common Data Form) is a set of software libraries and
    machine-independent data formats that support the creation, access, and
    sharing of array-oriented scientific data. This is the C++ distribution.

Homepage: https://www.unidata.ucar.edu/software/netcdf

Maintainers: @WardF

Tags:
    None

Preferred version: ❶
    4.3.1    ftp://ftp.unidata.ucar.edu/pub/netcdf/netcdf-cxx4-4.3.1.tar.gz
```

```
Safe versions:
    4.3.1    ftp://ftp.unidata.ucar.edu/pub/netcdf/netcdf-cxx4-4.3.1.tar.gz
    4.3.0    ftp://ftp.unidata.ucar.edu/pub/netcdf/netcdf-cxx4-4.3.0.tar.gz

Variants: ❷
    Name [Default]    Allowed values    Description
    ==============    ==============
 ===================================================

    doxygen [on]      on, off           Enable doxygen docs
    pic [on]          on, off           Produce position-independent code (for
 shared libs)
    shared [on]       on, off           Enable shared library
    static [on]       on, off           Enable building static libraries

Installation Phases:
    autoreconf    configure    build    install

Build Dependencies: ❸
    autoconf  automake  doxygen  libtool  m4  netcdf-c

Link Dependencies:
    netcdf-c

Run Dependencies:
    None

Virtual Packages:
    None
```

❶ Spack uses the latest version by default, unless you specify otherwise with `@VERSION`.

❷ You can disable the variant options using `-VARIANT` or enable them using `+VARIANT`.

❸ The packages required by `netcdf-cxx4` must already be built.

2. Build `netcdf-cxx4` with the variants `static` and `doxygen` disabled:

```
# spack install netcdf-cxx4 -static -doxygen
==> netcdf-c: Executing phase: 'autoreconf'
==> netcdf-c: Executing phase: 'configure'
==> netcdf-c: Executing phase: 'build'
==> netcdf-c: Executing phase: 'install'
[+] /usr/opt/spack/linux-sle_hpc15-skylake/gcc-7.5.0/netcdf-c-4.7.4-
vry3tfp6kpq364gyxrj6fali4kqhirx7
==> Installing netcdf-cxx4-4.3.1-msiysdrdua3vv6izluhaeos4nyo5gslq
==> No binary for netcdf-cxx4-4.3.1-msiysdrdua3vv6izluhaeos4nyo5gslq found:
 installing from source
```

```
==> Fetching https://spack-llnl-mirror.s3-us-west-2.amazonaws.com/_source-cache/
archive/6a/6a1189a181eed043b5859e15d5c080c30d0e107406fbb212c8fb9814e90f3445.tar.gz
##################################################################################################
 100.0%
==> netcdf-cxx4: Executing phase: 'autoreconf'
==> netcdf-cxx4: Executing phase: 'configure'
==> netcdf-cxx4: Executing phase: 'build'
==> netcdf-cxx4: Executing phase: 'install'
[+] /usr/opt/spack/linux-sle_hpc15-skylake/gcc-7.5.0/netcdf-cxx4-4.3.1-
msiysdrdua3vv6izluhaeos4nyo5gslq
```

3. Rebuild `netcdf-cxx4` with the default variants, and specify the version:

```
# spack install netcdf-cxx4@4.3.1
==> Warning: Missing a source id for python@3.6.13
==> Warning: Missing a source id for pkgconf@1.5.3
[+] /usr (external autoconf-2.69-tatq2aqbhboxbyjt2fsraoapgqwf3y5x)
[+] /usr (external automake-1.15.1-3d7wkh42v52c6n77t4p7l2i7nguryisl)
[+] /usr (external bison-3.0.4-y6ckc7e7mqnnkgmkbgcfbw5vgqzg5b6m)
[+] /usr (external cmake-3.17.0-jr4evnjsgd7uh5stt33woclti37743kg)
[+] /usr (external flex-2.6.4-vea2lhgajmeyjm6ei5d2bqvpss4ipors)
[+] /usr/opt/spack/linux-sle_hpc15-skylake/gcc-7.5.0/libiconv-1.16-
itovpc5jssshcgpeejrro6l7jn4ynaq7
[+] /usr (external python-3.6.13-rpf47wa6wfn7h3rnydpxijoczc6opno2)
[+] /usr (external libtool-2.4.6-ddch2qlie7t4ypbqg6kmf3uswqg2uylp)
[+] /usr (external m4-1.4.18-tloh56qj47ahddst5g2xqsawffuz5ew6)
[+] /usr (external pkgconf-1.5.3-gmxadsjg6q3xqwjwws5a4v4b4ugvi6p4)
[+] /usr/opt/spack/linux-sle_hpc15-skylake/gcc-7.5.0/util-macros-1.19.1-
rpnlbst6v3oqjm7tfoxasmn7wlilpqut
[+] /usr (external xz-5.2.3-x3glm5yp2ixldbe7n557evglhygvlkqh)
[+] /usr/opt/spack/linux-sle_hpc15-skylake/gcc-7.5.0/zlib-1.2.11-
z6y74kgd73yc23kr5252slbydmk4posh
[+] /usr/opt/spack/linux-sle_hpc15-skylake/gcc-7.5.0/
doxygen-1.8.20-3griwieblqgb6ykc5avzkzrxmtaw4s2g
[+] /usr/opt/spack/linux-sle_hpc15-skylake/gcc-7.5.0/numactl-2.0.14-
tcuvjjtkhnyf5ijrazenjra5h5dbj4in
[+] /usr/opt/spack/linux-sle_hpc15-skylake/gcc-7.5.0/libpciaccess-0.16-
y3w7dlktz22lmdj6fei4aj2f4t2rqu6l
[+] /usr/opt/spack/linux-sle_hpc15-skylake/gcc-7.5.0/libxml2-2.9.10-
d4c7cskvhn7qwzzb2wiq7rl67vbl44je
[+] /usr/opt/spack/linux-sle_hpc15-skylake/gcc-7.5.0/hwloc-1.11.11-
rjdqchtk6i27lqxxwi4cvfyvrxxgwq7k
[+] /usr/opt/spack/linux-sle_hpc15-skylake/gcc-7.5.0/openmpi-3.1.6-
ks5elgg25bbnzwa7fmv7lewbkrcp2qsx
[+] /usr/opt/spack/linux-sle_hpc15-skylake/gcc-7.5.0/
hdf5-1.10.7-3cyidn4yvikyyuxehak7ftey2l57ku37
```

```
[+] /usr/opt/spack/linux-sle_hpc15-skylake/gcc-7.5.0/netcdf-c-4.7.4-
vry3tfp6kpq364gyxrj6fali4kqhirx7
==> Installing netcdf-cxx4-4.3.1-tiyqyxb3eqpptrlcll6rlf27aisekluy
==> No binary for netcdf-cxx4-4.3.1-tiyqyxb3eqpptrlcll6rlf27aisekluy found:
 installing from source
==> Using cached archive: /var/spack/cache/_source-cache/
archive/6a/6a1189a181eed043b5859e15d5c080c30d0e107406fbb212c8fb9814e90f3445.tar.gz
==> netcdf-cxx4: Executing phase: 'autoreconf'
==> netcdf-cxx4: Executing phase: 'configure'
==> netcdf-cxx4: Executing phase: 'build'
==> netcdf-cxx4: Executing phase: 'install'
[+] /usr/opt/spack/linux-sle_hpc15-skylake/gcc-7.5.0/netcdf-cxx4-4.3.1-
tiyqyxb3eqpptrlcll6rlf27aisekluy
```

4. Check which packages are now available with Spack:

```
# spack find
==> 14 installed packages
-- linux-sle_hpc15-skylake / gcc@7.5.0 -------------------------
doxygen@1.8.20  hwloc@1.11.11  libpciaccess@0.16  netcdf-c@4.7.4     netcdf-
cxx4@4.3.1  openmpi@3.1.6       xz@5.2.3
hdf5@1.10.7     libiconv@1.16  libxml2@2.9.10     netcdf-cxx4@4.3.1  numactl@2.0.14
    util-macros@1.19.1  zlib@1.2.11
```

In this example, there are now two versions of `netcdf-cxx4`. All of the build requirements for `netcdf-cxx4` are also present. If you want to show dependency hashes as well as versions, use the `-l` option:

```
# spack find -l
==> 14 installed packages
-- linux-sle_hpc15-skylake / gcc@7.5.0 -------------------------
3griwie doxygen@1.8.20  y3w7dlk libpciaccess@0.16  tiyqyxb netcdf-cxx4@4.3.1
 x3glm5y xz@5.2.3
3cyidn4 hdf5@1.10.7     d4c7csk libxml2@2.9.10     tcuvjjt numactl@2.0.14
 z6y74kg zlib@1.2.11
rjdqcht hwloc@1.11.11   vry3tfp netcdf-c@4.7.4     ks5elgg openmpi@3.1.6
itovpc5 libiconv@1.16   msiysdr netcdf-cxx4@4.3.1  rpnlbst util-macros@1.19.1
```

5. Show the differences between the two versions of `netcdf-cxx4`:

   a. Find the paths to the `netcdf-cxx4` packages:

   ```
   # spack find --paths
   ==> 15 installed packages
   -- linux-sle_hpc15-skylake / gcc@7.5.0 -------------------------
   ```

```
doxygen@1.8.20      /usr/opt/spack/linux-sle_hpc15-skylake/gcc-7.5.0/
doxygen-1.8.20-3griwieblqgb6ykc5avzkzrxmtaw4s2g
hdf5@1.10.7         /usr/opt/spack/linux-sle_hpc15-skylake/gcc-7.5.0/
hdf5-1.10.7-3cyidn4yvikyyuxehak7ftey2l57ku37
hwloc@1.11.11       /usr/opt/spack/linux-sle_hpc15-skylake/gcc-7.5.0/
hwloc-1.11.11-rjdqchtk6i27lqxxwi4cvfyvrxxgwq7k
hwloc@2.2.0         /usr/opt/spack/linux-sle_hpc15-skylake/gcc-7.5.0/
hwloc-2.2.0-4lxxw65tzjeqhyxelowclnwqfb3m3rmk
libiconv@1.16       /usr/opt/spack/linux-sle_hpc15-skylake/gcc-7.5.0/
libiconv-1.16-itovpc5jssshcgpeejrro6l7jn4ynaq7
libpciaccess@0.16   /usr/opt/spack/linux-sle_hpc15-skylake/gcc-7.5.0/
libpciaccess-0.16-y3w7dlktz22lmdj6fei4aj2f4t2rqu6l
libxml2@2.9.10      /usr/opt/spack/linux-sle_hpc15-skylake/gcc-7.5.0/
libxml2-2.9.10-d4c7cskvhn7qwzzb2wiq7rl67vbl44je
netcdf-c@4.7.4      /usr/opt/spack/linux-sle_hpc15-skylake/gcc-7.5.0/netcdf-
c-4.7.4-vry3tfp6kpq364gyxrj6fali4kqhirx7
netcdf-cxx4@4.3.1   /usr/opt/spack/linux-sle_hpc15-skylake/gcc-7.5.0/netcdf-
cxx4-4.3.1-msiysdrdua3vv6izluhaeos4nyo5gslq
netcdf-cxx4@4.3.1   /usr/opt/spack/linux-sle_hpc15-skylake/gcc-7.5.0/netcdf-
cxx4-4.3.1-tiyqyxb3eqpptrlcll6rlf27aisekluy
numactl@2.0.14      /usr/opt/spack/linux-sle_hpc15-skylake/gcc-7.5.0/
numactl-2.0.14-tcuvjjtkhnyf5ijrazenjra5h5dbj4in
openmpi@3.1.6       /usr/opt/spack/linux-sle_hpc15-skylake/gcc-7.5.0/
openmpi-3.1.6-ks5elgg25bbnzwa7fmv7lewbkrcp2qsx
util-macros@1.19.1  /usr/opt/spack/linux-sle_hpc15-skylake/gcc-7.5.0/util-
macros-1.19.1-rpnlbst6v3oqjm7tfoxasmn7wlilpqut
xz@5.2.3            /usr
zlib@1.2.11         /usr/opt/spack/linux-sle_hpc15-skylake/gcc-7.5.0/
zlib-1.2.11-z6y74kgd73yc23kr5252slbydmk4posh
```

b. Move into the parent directory:

```
# cd /usr/opt/spack/linux-sle_hpc15-skylake/gcc-7.5.0
```

c. Run a `diff` between each version's `spec.yaml` file. This file describes how the package was built.

```
# diff -ru netcdf-cxx4-4.3.1-msiysdrdua3vv6izluhaeos4nyo5gslq/.spack/spec.yaml
 netcdf-cxx4-4.3.1-tiyqyxb3eqpptrlcll6rlf27aisekluy/.spack/spec.yaml
--- netcdf-cxx4-4.3.1-msiysdrdua3vv6izluhaeos4nyo5gslq/.spack/spec.yaml
 2021-10-04 11:42:23.444000000 +0200
+++ netcdf-cxx4-4.3.1-tiyqyxb3eqpptrlcll6rlf27aisekluy/.spack/spec.yaml
 2021-10-04 11:51:49.880000000 +0200
@@ -38,10 +38,10 @@
       version: 7.5.0
     namespace: builtin
     parameters:
-      doxygen: false
+      doxygen: true
       pic: true
       shared: true
-      static: false
+      static: true
       cflags: []
       cppflags: []
       cxxflags: []
@@ -54,8 +54,8 @@
         type:
         - build
         - link
-    hash: msiysdrdua3vv6izluhaeos4nyo5gslq
-    full_hash: oeylqzvergh3ckviaqyhy3idn7vyk3hi
+    hash: tiyqyxb3eqpptrlcll6rlf27aisekluy
+    full_hash: i6btamzuyoo343n63f3rqopkz7ymkapq
 - netcdf-c:
     version: 4.7.4
arch:
```

This example shows that one version of `netcdf-cxx4` was built with `doxygen` and `static` enabled, and the other version of `netcdf-cxx4` was built with `doxygen` and `static` disabled.

9.3 Using Spack: complex example with mpich

This example procedure shows you different ways to build `mpich` with Spack.

PROCEDURE 9.3: BUILDING `mpich` WITH SPACK

1. List the available versions of `mpich`:

```
# spack versions mpich
==> Safe versions (already checksummed):
  develop  3.3.2  3.3.1  3.3  3.2.1  3.2  3.1.4  3.1.3  3.1.2  3.1.1  3.1  3.0.4
==> Remote versions (not yet checksummed):
==> Warning: Found no unchecksummed versions for mpich
```

2. Show detailed information on `mpich`:

```
# spack info mpich
AutotoolsPackage:   mpich
```

```
Description:
    MPICH is a high performance and widely portable implementation of the
    Message Passing Interface (MPI) standard.

Homepage: http://www.mpich.org

Maintainers: @raffenet @yfguo

Tags:
    None

Preferred version:
    3.3.2      http://www.mpich.org/static/downloads/3.3.2/mpich-3.3.2.tar.gz

Safe versions:
    develop    [git] https://github.com/pmodels/mpich.git
    3.3.2      http://www.mpich.org/static/downloads/3.3.2/mpich-3.3.2.tar.gz
    3.3.1      http://www.mpich.org/static/downloads/3.3.1/mpich-3.3.1.tar.gz
    3.3        http://www.mpich.org/static/downloads/3.3/mpich-3.3.tar.gz
    3.2.1      http://www.mpich.org/static/downloads/3.2.1/mpich-3.2.1.tar.gz
    3.2        http://www.mpich.org/static/downloads/3.2/mpich-3.2.tar.gz
    3.1.4      http://www.mpich.org/static/downloads/3.1.4/mpich-3.1.4.tar.gz
    3.1.3      http://www.mpich.org/static/downloads/3.1.3/mpich-3.1.3.tar.gz
    3.1.2      http://www.mpich.org/static/downloads/3.1.2/mpich-3.1.2.tar.gz
    3.1.1      http://www.mpich.org/static/downloads/3.1.1/mpich-3.1.1.tar.gz
    3.1        http://www.mpich.org/static/downloads/3.1/mpich-3.1.tar.gz
    3.0.4      http://www.mpich.org/static/downloads/3.0.4/mpich-3.0.4.tar.gz

Variants:
    Name [Default]       Allowed values          Description
    =================    ====================
 ========================================================================

    argobots [off]       on, off                 Enable Argobots support
    device [ch3]         ch3, ch4                Abstract Device Interface (ADI)
 implementation. The ch4 device is
                                                 currently in experimental state
    fortran [on]         on, off                 Enable Fortran support
    hwloc [on]           on, off                 Use external hwloc package
    hydra [on]❶         on, off                 Build the hydra process manager
    libxml2 [on]         on, off                 Use libxml2 for XML support instead
 of the custom minimalistic
                                                 implementation
    netmod [tcp]❷      tcp, mxm, ofi, ucx      Network module. Only single netmod
 builds are supported. For ch3 device
                                                 configurations, this presumes the
 ch3:nemesis communication channel.
```

```
                                                 ch3:sock is not supported by this
 spack package at this time.
    pci [on]             on, off                 Support analyzing devices on PCI
 bus
    pmi [pmi]            off, pmi, pmi2, pmix    PMI interface.
    romio [on]           on, off                 Enable ROMIO MPI I/O implementation
    slurm [off]          on, off                 Enable SLURM support
    verbs [off]          on, off                 Build support for OpenFabrics
 verbs.
    wrapperrpath [on]    on, off                 Enable wrapper rpath

Installation Phases:
    autoreconf    configure    build    install

Build Dependencies:
    argobots  automake   hwloc      libpciaccess  libxml2  pkgconfig  python  ucx
    autoconf  findutils  libfabric  libtool       m4       pmix       slurm

Link Dependencies:
    argobots  hwloc  libfabric  libpciaccess  libxml2  pmix  slurm  ucx

Run Dependencies:
    None

Virtual Packages:
    mpich@3: provides mpi@:3.0
    mpich@1: provides mpi@:1.3
    mpich provides mpi
```

1 `NAME [on]` is a boolean variant.

2 `NAME=VALUE` is a non-boolean variant.

3. Build `mpich` with some variants disabled:

```
# spack install mpich@3.3.2 -romio -libxml2 -hydra -fortran
==> mpich: Executing phase: 'autoreconf'
==> mpich: Executing phase: 'configure'
==> mpich: Executing phase: 'build'
==> mpich: Executing phase: 'install'
[+] /usr/opt/spack/linux-sle_hpc15-skylake/gcc-7.5.0/
mpich-3.3.2-3nexqqx2r4m7p2y7lmovuwvvobi2bygw
```

4. Rebuild `mpich` with the default variants:

```
# spack install mpich@3.3.2
.....
```

```
[+] /usr/opt/spack/linux-sle_hpc15-skylake/gcc-7.5.0/mpich-3.3.2-
sahjm2uhsoc3bi2cljtypwuqaflhamnx
```

5. Rebuild `mpich` with the compiler flag `cppflags`, a specific `libxml` version of `2.9.4`, an x86_64 `target`, and a non-boolean `netmod` option:

```
# spack install mpich@3.3.2❶ -static❷ cppflags="-O3 -fPIC"❸ target=x86_64❹
 ^libxml2@2.9.4❺ netmod=tcp❻
```

❶ `@` is an optional version specifier.

❷ `+` or `-` specifies the boolean variant (on or off).

❸ `cflags, cxxflags, fflags, cppflags, ldflags`, and `ldlibs` are variants for the compiler flag.

❹ `target=*value*` and `os=*value*` specify the architecture and the operating system. You can list all of the available targets by running the command **`spack arch --known-targets`**.

❺ `^` specifies a dependency.

❻ `*name*=*value*` specifies a non-boolean variant.

6. List all of the available packages and their dependency hashes:

```
# spack find -l
==> 33 installed packages
-- linux-sle_hpc15-skylake / gcc@7.5.0 --------------------------
3griwie doxygen@1.8.20  y3w7dlk libpciaccess@0.16  vry3tfp netcdf-c@4.7.4
 x3glm5y xz@5.2.3
3cyidn4 hdf5@1.10.7     to2atk6 libpciaccess@0.16  msiysdr netcdf-cxx4@4.3.1
 bzeebyp xz@5.2.3
rjdqcht hwloc@1.11.11   4cyoxau libxml2@2.9.4      tiyqyxb netcdf-cxx4@4.3.1
 z6y74kg zlib@1.2.11
4lxxw65 hwloc@2.2.0     d4c7csk libxml2@2.9.10     tcuvjjt numactl@2.0.14
 m4zmub6 zlib@1.2.11
yof3lps hwloc@2.2.0     3nexqqx mpich@3.3.2        ks5elgg openmpi@3.1.6
itovpc5 libiconv@1.16   sahjm2u mpich@3.3.2        rpnlbst util-macros@1.19.1
5mvdlgl libiconv@1.16   jxbspwk mpich@3.3.2        boclx6a util-macros@1.19.1

-- linux-sle_hpc15-x86_64 / gcc@7.5.0 ---------------------------
yzxibyd hwloc@2.2.0    7wmqik2 libpciaccess@0.16  6xjiutf mpich@3.3.2
 zmfrpzf xz@5.2.3
mm7at5h libiconv@1.16  jrtvvdj libxml2@2.9.4      jjxbukq util-macros@1.19.1
 jxqtsne zlib@1.2.11
```

7. Show the specifications and output dependencies of `/6xjiutf` (`mpich`):

   ```
   # spack find --deps /6xjiutf
   ==> 1 installed package
   -- linux-sle_hpc15-x86_64 / gcc@7.5.0 ---------------------------
   mpich@3.3.2
       hwloc@2.2.0
           libpciaccess@0.16
           libxml2@2.9.4
               libiconv@1.16
               xz@5.2.3
               zlib@1.2.11
   ```

9.4 Using a specific compiler

In this example procedure, the goal is to build `mpich` with `gcc-10.2.0`.

PROCEDURE 9.4: USING A SPECIFIC COMPILER WITH SPACK

1. Install `gcc-10.2.0`:

   ```
   # spack install gcc@10.2.0
   .....
   [+] /usr/opt/spack/linux-sle_hpc15-skylake/gcc-7.5.0/gcc-10.2.0-
   tkcq6d6xnyfgyqsnpzq36dlk2etylgp7
   ```

2. Test whether `gcc` is usable by loading it and attempting to build `mpich`, using the `%` option to specify that you want to use `gcc-10.2.0`:

   ```
   # module load gcc-10.2.0-gcc-7.5.0-tkcq6d6 ❶
   # gcc --version
   gcc (Spack GCC) 10.2.0
   # spack install mpich@3.2.1 %gcc@10.2.0
   ==> Error: No compilers with spec gcc@10.2.0 found for operating system sle_hpc15
    and target skylake.
   Run 'spack compiler find' to add compilers or 'spack compilers' to see which
    compilers are already recognized by spack.
   ```

   ❶ **`module load gcc-10.2.0`** also works if you have only one version of `gcc-10.2.0` available in Spack.

   In this example, the compiler cannot be found yet.

3. Update the list of available compilers:

```
# spack compiler find
==> Added 1 new compiler to /home/tux/.spack/linux/compilers.yaml
    gcc@10.2.0
==> Compilers are defined in the following files:
    /home/tux/.spack/linux/compilers.yaml
```

4. Rerun the command to build `mpich` with `gcc-10.2.0`:

```
# spack install mpich@3.2.1 %gcc@10.2.0
.....
[+] /usr/opt/spack/linux-sle_hpc15-skylake/gcc-10.2.0/mpich-3.2.1-
e56rcwtqofh2xytep5pjgq6wrxwmsy25
```

10 Dolly clone tool

Dolly is used to send data from a management server to many other nodes. It can distribute files, container images, partitions, or whole storage devices.

10.1 Dolly cloning process

One machine is the management server and distributes the data to the nodes. The management server stores the image, partition, disk, or data to be cloned. This machine runs dolly as a server. All other nodes are dolly clients. They receive the data from the ring, store it locally, and send it to the next node in the ring. All of this happens at the same time, so transferring data to one node or to hundreds of nodes takes the same amount of time.

Dolly creates a virtual TCP ring to distribute data.

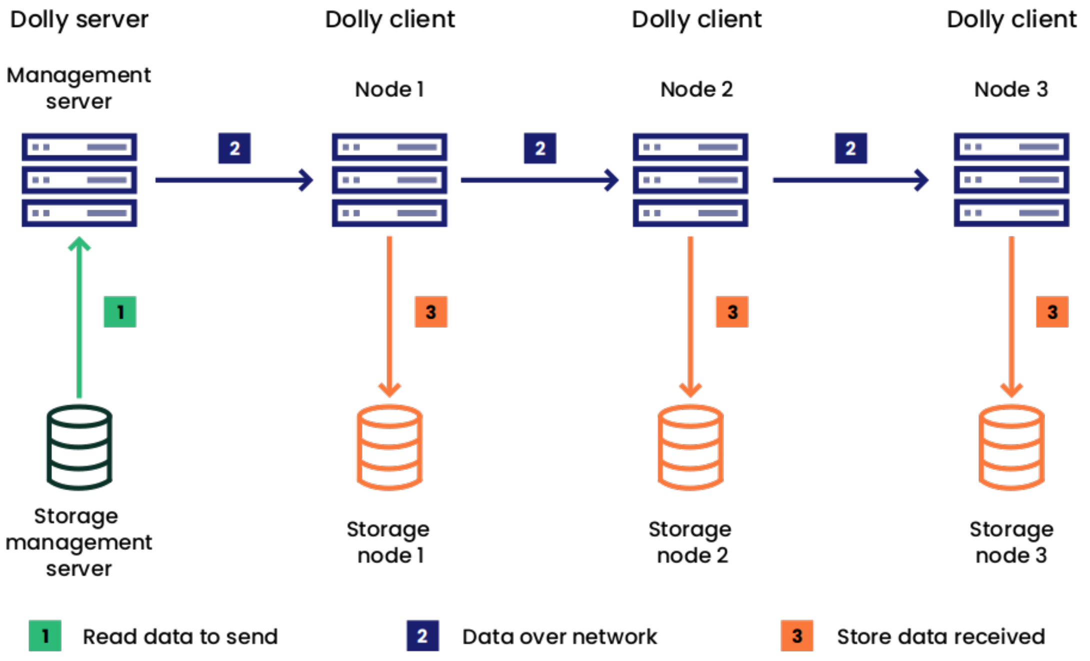

FIGURE 10.1: DOLLY CLONING PROCESS

10.2 Using dolly

Install dolly on the management server and all dolly client nodes:

```
# zypper in dolly
```

Note: Automatically opened ports

Installing the `dolly` package automatically opens the TCP ports 9997 and 9998.

The **`dolly`** command requires the following information, either directly on the command line or from a configuration file:

- Data to send over the network from the dolly server. This could be a storage device, a file (`gzip` or other file types), or an image (containers or other data).
- A target on the dolly clients. The target must be the same data type as the input from the dolly server. For example, dolly cannot send a `gzip` file to a storage device.
- A list of the dolly client nodes where you want to send the data.

Any other parameters are optional. For more information, see **`man dolly`** and *Section 10.4, “Dolly limitations”*.

PROCEDURE 10.1: CLONING DATA FROM A DOLLY SERVER TO DOLLY CLIENTS

1. On the dolly server, run the following command:

```
# dolly -s❶ -v❷ -o LOGFILE❸ -I INPUT❹ -O OUTPUT❺ -H NODE1,NODE2,..❻
```

❶ Specifies that this node is the dolly server (the node that sends the data).

❷ Switches dolly to verbose mode, which is helpful for debugging.

❸ The file that statistical information is written to.

❹ The data to clone.

❺ The target that will store the data on each dolly client.

❻ A comma-separated list of dolly clients to receive the data.

For example, the following command sends `/dev/sdc1` to the nodes `sle152`, `sle153`, `sle154`, and `sle155`:

```
# dolly -s -v -o /tmp/dolly.log -I /dev/sdc1 -O /dev/sdc1 -H
 sle152,sle153,sle154,sle155
```

2. On each dolly client, start **`dolly`**:

```
# dolly -v
```

You can run this command on multiple nodes at once using **`pdsh`**. See *Section 4.2, “pdsh — parallel remote shell program”*.

EXAMPLE 10.1: DOLLY SERVER VERBOSE OUTPUT

This example shows typical verbose output from the **`dolly`** command on the dolly server:

```
'writing '192.168.255.2'
'writing '192.168.255.3'
'writing '192.168.255.4'
'writing '192.168.255.5'
'Parameter file:
infile = '/dev/sdc1'
outfile = '/dev/sdc1'
using data port 9998
using ctrl port 9997
myhostname = 'sle151'
fanout = 1
nr_childs = 1
server = 'sle151'
I'm the server.
I'm not the last host.
There are 4 hosts in the ring (excluding server):
        '192.168.255.2'
        '192.168.255.3'
        '192.168.255.4'
        '192.168.255.5'
Next hosts in ring:
        192.168.255.2 (0)
All parameters read successfully.
No compression used.
Using transfer size 4096 bytes.

Trying to build ring...
Connecting to host 192.168.255.2... Send buffer 0 is 131072 bytes
data control.
Waiting for ring to build...
```

```
Host got parameters '192.168.255.2'.
Machines left to wait for: 4
Host ready '192.168.255.2'.
Machines left to wait for: 3
Host got parameters '192.168.255.3'.
Machines left to wait for: 3
Host ready '192.168.255.3'.
Machines left to wait for: 2
Host got parameters '192.168.255.4'.
Machines left to wait for: 2
Host ready '192.168.255.4'.
Machines left to wait for: 1
Host got parameters '192.168.255.5'.
Machines left to wait for: 1
Host ready '192.168.255.5'.
Machines left to wait for: 0
Accepted.
Server: Sending data...
Sent MB: 15854, MB/s: 29.655, Current MB/s: 111.111
Read 15854469120 bytes from file(s).
Writing maxbytes = 15854469120 to ctrlout
Sent MB: 15854.
Synced.
Waiting for child 0.
Clients done.
Time: 534.627532
MBytes/s: 29.655
Aggregate MBytes/s: 118.621
Transmitted.
```

EXAMPLE 10.2: DOLLY CLIENT VERBOSE OUTPUT

This example shows typical verbose output from the **dolly** command on the dolly clients:

```
Trying to build ring...
Buffer size: 98304
Receive buffer is 196608 bytes
Accepting...control...
Trying to read parameters...done.
Parsing parameters...
done.
192.168.255.2 is number 0
Parameter file:
infile = '/dev/sdc1'
outfile = '/dev/sdc1'
using data port 9998
using ctrl port 9997
```

```
myhostname = '192.168.255.2'
fanout = 1
nr_childs = 1
server = 'sle151'
I'm not the server.
I'm not the last host.
There are 4 hosts in the ring (excluding server):
        '192.168.255.2'
        '192.168.255.3'
        '192.168.255.4'
        '192.168.255.5'
Next hosts in ring:
192.168.255.3 (1)
All parameters read successfully.
No compression used.
Using transfer size 4096 bytes.
Connected data...done.
Connecting to host 192.168.255.3...
data control.
Accepted.
Receiving...
Transfered MB: 15854, MB/s: 29.655, Current MB/s: 116.661
Max. bytes will be 15854469120 bytes. 49152 bytes left.
Transfered MB: 15854, MB/s: 29.655

Synced.
Transmitted.
```

10.3 Dolly configuration file

You can use a dolly configuration file with the `-f` parameter instead of providing the information manually on the command line. The following example shows a typical configuration file called `/etc/dolly.cfg`:

```
infile /tmp/sle15.sif❶
outfile /data/sle15.sif❷
server sle151❸
firstclient sle152❹
lastclient sle154❺
clients 3❻
sle152❼
sle153
sle154
endconfig❽
```

1. The data to send over the network from the dolly server.
2. The file or device that will store the data on the dolly clients.
3. The name of the dolly server.
4. The first dolly client in the ring.
5. The last dolly client in the ring.
6. Specifies how many dolly clients are in the ring.
7. The list of dolly clients, one per line.
8. Specifies the end of the configuration file.

To use this configuration file, run the following command on the dolly server:

```
# dolly -v -s -f /etc/dolly.cfg
```

10.4 Dolly limitations

Be aware of the following restrictions:

- The output data type must be the same as the input data type. Mixing the type of input and output can lead to data corruption.
- Only clone partitions that are identical in size on the dolly server and the client node.
- Only clone strictly identical storage devices, or corruption can occur.

The following command line parameters are not supported and are provided as a technology preview only:

- `-S`: Ignoring the FQDN is not supported.
- `-6`: Using IPv6 is not supported.
- `-n`: Not doing a sync before exiting is not supported as this can lead to data corruption.
- `-c`: Specifying the uncompressed size of a compressed file should only be used for performance statistics.

The following configuration file options are not supported and are provided as a technology preview only:

- `compressed`: Using the compression option is not supported.
- `split`: Splitting files is not supported (infile or outfile).
- `fanout`: This option must be set to 1 (a linear list). A binary tree or more is not supported.
- `segsize`: This benchmark switch is not supported.
- `add`: Using more than one interface to clone data is not supported.

A GNU licenses

This appendix contains the GNU Free Documentation License version 1.2.

GNU Free Documentation License

0. PREAMBLE

The purpose of this License is to make a manual, textbook, or other functional and useful document "free" in the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or non-commercially. Secondarily, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.

This License is a kind of "copyleft", which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.

We have designed this License to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

1. APPLICABILITY AND DEFINITIONS

This License applies to any manual or other work, in any medium, that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. Such a notice grants a world-wide, royalty-free license, unlimited in duration, to use that work under the conditions stated herein. The "Document", below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as "you". You accept the license if you copy, modify or distribute the work in a way requiring permission under copyright law.

A "Modified Version" of the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language.

A "Secondary Section" is a named appendix or a front-matter section of the Document that deals exclusively with the relationship of the publishers or authors of the Document to the Document's overall subject (or to related matters) and contains nothing that could fall directly within that overall subject. (Thus, if the Document is in part a textbook of mathematics, a Secondary Section may not explain any mathematics.) The relationship could be a matter of historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.

The "Invariant Sections" are certain Secondary Sections whose titles are designated, as being those of Invariant Sections, in the notice that says that the Document is released under this License. If a section does not fit the above definition of Secondary then it is not allowed to be designated as Invariant. The Document may contain zero Invariant Sections. If the Document does not identify any Invariant Sections then there are none.

The "Cover Texts" are certain short passages of text that are listed, as Front-Cover Texts or Back-Cover Texts, in the notice that says that the Document is released under this License. A Front-Cover Text may be at most 5 words, and a Back-Cover Text may be at most 25 words.

A "Transparent" copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, that is suitable for revising the document straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters. A copy made in an otherwise Transparent file format whose markup, or absence of markup, has been arranged to thwart or discourage subsequent modification by readers is not Transparent. An image format is not Transparent if used for any substantial amount of text. A copy that is not "Transparent" is called "Opaque".

Examples of suitable formats for Transparent copies include plain ASCII without markup, Texinfo input format, LaTeX input format, SGML or XML using a publicly available DTD, and standard-conforming simple HTML, PostScript or PDF designed for human modification. Examples of transparent image formats include PNG, XCF and JPG. Opaque formats include proprietary formats that can be read and edited only by proprietary word processors, SGML or XML for which the DTD and/or processing tools are not generally available, and the machine-generated HTML, PostScript or PDF produced by some word processors for output purposes only.

The "Title Page" means, for a printed book, the title page itself, plus such following pages as are needed to hold, legibly, the material this License requires to appear in the title page. For works in formats which do not have any title page as such, "Title Page" means the text near the most prominent appearance of the work's title, preceding the beginning of the body of the text.

A section "Entitled XYZ" means a named subunit of the Document whose title either is precisely XYZ or contains XYZ in parentheses following text that translates XYZ in another language. (Here XYZ stands for a specific section name mentioned below, such as "Acknowledgements", "Dedications", "Endorsements", or "History".) To "Preserve the Title" of such a section when you modify the Document means that it remains a section "Entitled XYZ" according to this definition.

The Document may include Warranty Disclaimers next to the notice which states that this License applies to the Document. These Warranty Disclaimers are considered to be included by reference in this License, but only as regards disclaiming warranties: any other implication that these Warranty Disclaimers may have is void and has no effect on the meaning of this License.

2. VERBATIM COPYING

You may copy and distribute the Document in any medium, either commercially or non-commercially, provided that this License, the copyright notices, and the license notice saying this License applies to the Document are reproduced in all copies, and that you add no other conditions whatsoever to those of this License. You may not use technical measures to obstruct or control the reading or further copying of the copies you make or distribute. However, you may accept compensation in exchange for copies. If you distribute a large enough number of copies you must also follow the conditions in section 3.

You may also lend copies, under the same conditions stated above, and you may publicly display copies.

3. COPYING IN QUANTITY

If you publish printed copies (or copies in media that commonly have printed covers) of the Document, numbering more than 100, and the Document's license notice requires Cover Texts, you must enclose the copies in covers that carry, clearly and legibly, all these Cover Texts: Front-Cover Texts on the front cover, and Back-Cover Texts on the back cover. Both covers must also clearly and legibly identify you as the publisher of these copies. The front cover must present the full title with all words of the title equally prominent and visible. You may add other material on the covers in addition. Copying with changes limited to the covers, as long as they preserve the title of the Document and satisfy these conditions, can be treated as verbatim copying in other respects.

If the required texts for either cover are too voluminous to fit legibly, you should put the first ones listed (as many as fit reasonably) on the actual cover, and continue the rest onto adjacent pages.

If you publish or distribute Opaque copies of the Document numbering more than 100, you must either include a machine-readable Transparent copy along with each Opaque copy, or state in or with each Opaque copy a computer-network location from which the general network-using public has access to download using public-standard network protocols a complete Transparent copy of the Document, free of added material. If you use the latter option, you must take reasonably prudent steps, when you begin distribution of Opaque copies in quantity, to ensure that this Transparent copy will remain thus accessible at the stated location until at least one year after the last time you distribute an Opaque copy (directly or through your agents or retailers) of that edition to the public.

It is requested, but not required, that you contact the authors of the Document well before redistributing any large number of copies, to give them a chance to provide you with an updated version of the Document.

4. MODIFICATIONS

You may copy and distribute a Modified Version of the Document under the conditions of sections 2 and 3 above, provided that you release the Modified Version under precisely this License, with the Modified Version filling the role of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of it. In addition, you must do these things in the Modified Version:

A. Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if there were any, be listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that version gives permission.

B. List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the Document (all of its principal authors, if it has fewer than five), unless they release you from this requirement.

C. State on the Title page the name of the publisher of the Modified Version, as the publisher.

D. Preserve all the copyright notices of the Document.

E. Add an appropriate copyright notice for your modifications adjacent to the other copyright notices.

F. Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below.

G. Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document's license notice.

H. Include an unaltered copy of this License.

I. Preserve the section Entitled "History", Preserve its Title, and add to it an item stating at least the title, year, new authors, and publisher of the Modified Version as given on the Title Page. If there is no section Entitled "History" in the Document, create one stating the title, year, authors, and publisher of the Document as given on its Title Page, then add an item describing the Modified Version as stated in the previous sentence.

J. Preserve the network location, if any, given in the Document for public access to a Transparent copy of the Document, and likewise the network locations given in the Document for previous versions it was based on. These may be placed in the "History" section. You may omit a network location for a work that was published at least four years before the Document itself, or if the original publisher of the version it refers to gives permission.

K. For any section Entitled "Acknowledgements" or "Dedications", Preserve the Title of the section, and preserve in the section all the substance and tone of each of the contributor acknowledgements and/or dedications given therein.

L. Preserve all the Invariant Sections of the Document, unaltered in their text and in their titles. Section numbers or the equivalent are not considered part of the section titles.

M. Delete any section Entitled "Endorsements". Such a section may not be included in the Modified Version.

N. Do not retitle any existing section to be Entitled "Endorsements" or to conflict in title with any Invariant Section.

O. Preserve any Warranty Disclaimers.

If the Modified Version includes new front-matter sections or appendices that qualify as Secondary Sections and contain no material copied from the Document, you may at your option designate some or all of these sections as invariant. To do this, add their titles to the list of Invariant Sections in the Modified Version's license notice. These titles must be distinct from any other section titles.

You may add a section Entitled "Endorsements", provided it contains nothing but endorsements of your Modified Version by various parties--for example, statements of peer review or that the text has been approved by an organization as the authoritative definition of a standard.

You may add a passage of up to five words as a Front-Cover Text, and a passage of up to 25 words as a Back-Cover Text, to the end of the list of Cover Texts in the Modified Version. Only one passage of Front-Cover Text and one of Back-Cover Text may be added by (or through arrangements made by) any one entity. If the Document already includes a cover text for the same cover, previously added by you or by arrangement made by the same entity you are acting on behalf of, you may not add another; but you may replace the old one, on explicit permission from the previous publisher that added the old one.

The author(s) and publisher(s) of the Document do not by this License give permission to use their names for publicity for or to assert or imply endorsement of any Modified Version.

5. COMBINING DOCUMENTS

You may combine the Document with other documents released under this License, under the terms defined in section 4 above for modified versions, provided that you include in the combination all of the Invariant Sections of all of the original documents, unmodified, and list them all as Invariant Sections of your combined work in its license notice, and that you preserve all their Warranty Disclaimers.

The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be replaced with a single copy. If there are multiple Invariant Sections with the same name but different contents, make the title of each such section unique by adding at the end of it, in parentheses, the name of the original author or publisher of that section if known, or else a unique number. Make the same adjustment to the section titles in the list of Invariant Sections in the license notice of the combined work.

In the combination, you must combine any sections Entitled "History" in the various original documents, forming one section Entitled "History"; likewise combine any sections Entitled "Acknowledgements", and any sections Entitled "Dedications". You must delete all sections Entitled "Endorsements".

6. COLLECTIONS OF DOCUMENTS

You may make a collection consisting of the Document and other documents released under this License, and replace the individual copies of this License in the various documents with a single copy that is included in the collection, provided that you follow the rules of this License for verbatim copying of each of the documents in all other respects.

You may extract a single document from such a collection, and distribute it individually under this License, provided you insert a copy of this License into the extracted document, and follow this License in all other respects regarding verbatim copying of that document.

7. AGGREGATION WITH INDEPENDENT WORKS

A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, is called an "aggregate" if the copyright resulting from the compilation is not used to limit the legal rights of the compilation's users beyond what the individual works permit. When the Document is included in an aggregate, this License does not apply to the other works in the aggregate which are not themselves derivative works of the Document.

If the Cover Text requirement of section 3 is applicable to these copies of the Document, then if the Document is less than one half of the entire aggregate, the Document's Cover Texts may be placed on covers that bracket the Document within the aggregate, or the electronic equivalent of covers if the Document is in electronic form. Otherwise they must appear on printed covers that bracket the whole aggregate.

8. TRANSLATION

Translation is considered a kind of modification, so you may distribute translations of the Document under the terms of section 4. Replacing Invariant Sections with translations requires special permission from their copyright holders, but you may include translations of some or all Invariant Sections in addition to the original versions of these Invariant Sections. You may include a translation of this License, and all the license notices in the Document, and any Warranty Disclaimers, provided that you also include the original English version of this License and the original versions of those notices and disclaimers. In case of a disagreement between the translation and the original version of this License or a notice or disclaimer, the original version will prevail.

If a section in the Document is Entitled "Acknowledgements", "Dedications", or "History", the requirement (section 4) to Preserve its Title (section 1) will typically require changing the actual title.

9. TERMINATION

You may not copy, modify, sublicense, or distribute the Document except as expressly provided for under this License. Any other attempt to copy, modify, sublicense or distribute the Document is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

10. FUTURE REVISIONS OF THIS LICENSE

The Free Software Foundation may publish new, revised versions of the GNU Free Documentation License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns. See https://www.gnu.org/copyleft/ ↗.

Each version of the License is given a distinguishing version number. If the Document specifies that a particular numbered version of this License "or any later version" applies to it, you have the option of following the terms and conditions either of that specified version or of any later version that has been published (not as a draft) by the Free Software Foundation. If the Document does not specify a version number of this License, you may choose any version ever published (not as a draft) by the Free Software Foundation.

ADDENDUM: How to use this License for your documents

```
Copyright (c) YEAR YOUR NAME.
Permission is granted to copy, distribute and/or modify this document
under the terms of the GNU Free Documentation License, Version 1.2
or any later version published by the Free Software Foundation;
with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts.
A copy of the license is included in the section entitled “GNU
Free Documentation License”.
```

If you have Invariant Sections, Front-Cover Texts and Back-Cover Texts, replace the “with...Texts.” line with this:

```
with the Invariant Sections being LIST THEIR TITLES, with the
Front-Cover Texts being LIST, and with the Back-Cover Texts being LIST.
```

If you have Invariant Sections without Cover Texts, or some other combination of the three, merge those two alternatives to suit the situation.

If your document contains nontrivial examples of program code, we recommend releasing these examples in parallel under your choice of free software license, such as the GNU General Public License, to permit their use in free software.