



Release Notes

SUSE Linux Enterprise High-Performance Computing is a highly scalable, high-performance open-source operating system designed to utilize the power of parallel computing. This document provides an overview of high-level general features, capabilities, and limitations of SUSE Linux Enterprise High-Performance Computing 15 SP2 and important product updates.

These release notes are updated periodically. The latest version is always available at <https://www.suse.com/releasenotes> .


Publication Date: 2021-02-22 , Version: 15.200000000.20210222

Contents

1	About the Release Notes	3
2	SUSE Linux Enterprise High-Performance Computing	3
3	Modules, Extensions, and Related Products	6
4	Technology Previews	9
5	Installation and Upgrade	9
6	Software Changes	12
7	HPC Tools	27
8	MPI Libraries	50
9	General HPC Libraries	52
10	HPC File Format Libraries	63
11	Profiling and Benchmarking Libraries and Tools	69

- 12 Obtaining Source Code **71**
- 13 Legal Notices **71**

1 About the Release Notes

The most recent version of the Release Notes is available online at <https://www.suse.com/releasesnotes> .

Entries can be listed multiple times if they are important and belong to multiple sections.

Release notes only list changes that happened between two subsequent releases. Always review all release notes documents that apply in your upgrade scenario.

2 SUSE Linux Enterprise High-Performance Computing

SUSE Linux Enterprise High-Performance Computing is a highly scalable, high performance open-source operating system designed to utilize the power of parallel computing for modeling, simulation and advanced analytics workloads.

SUSE Linux Enterprise High-Performance Computing 15 SP2 provides tools and libraries related to High Performance Computing. This includes:

- Workload manager
- Remote and parallel shells
- Performance monitoring and measuring tools
- Serial console monitoring tool
- 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)
- User-extensible heap manager capable of distinguishing between different kinds of memory (x86-64 only)
- Serial and parallel computational libraries providing the common standards BLAS, LAPACK, ...

- Various MPI implementations
- Serial and parallel libraries for the HDF5 file format

2.1 Hardware Platform Support

SUSE Linux Enterprise High-Performance Computing 15 SP2 is available for the Intel 64/AMD64 (x86-64) and AArch64 platforms.

2.2 Important Sections of This Document

If you are upgrading from a previous SUSE Linux Enterprise High-Performance Computing release, you should review at least the following sections:

- *Section 2.4, “Support Statement for SUSE Linux Enterprise High-Performance Computing”*
- *Section 5.2, “Upgrade-Related Notes”*

2.3 Support and Life Cycle

SUSE Linux Enterprise High-Performance Computing is backed by award-winning support from SUSE, an established technology leader with a proven history of delivering enterprise-quality support services.

SUSE Linux Enterprise High-Performance Computing 15 has a 13-year life cycle, with 10 years of General Support and 3 years of Extended Support. The current version (SP2) will be fully maintained and supported until 6 months after the release of SUSE Linux Enterprise High-Performance Computing 15 SP3.

Any release package is fully maintained and supported until the availability of the next release. Extended Service Pack Overlay Support (ESPOS) and Long Term Service Pack Support (LTSS) are also available for this product. If you need additional time to design, validate and test your upgrade plans, Long Term Service Pack Support (LTSS) can extend the support you get by an additional 12 to 36 months in 12-month increments, providing a total of 3 to 5 years of support on any given Service Pack.

For more information, see:

- The support policy at <https://www.suse.com/support/policy.html> 
- Long Term Service Pack Support page at <https://www.suse.com/support/programs/long-term-service-pack-support.html> 

2.4 Support Statement for SUSE Linux Enterprise High-Performance Computing

To receive support, you need an appropriate subscription with SUSE. For more information, see <https://www.suse.com/support/programs/subscriptions/> .

The following definitions apply:

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 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 15 SP2 is delivered with L3 support for all packages, except for the following:

- Technology Previews, see *Section 4, "Technology Previews"*
- Sound, graphics, fonts and artwork
- Packages that require an additional customer contract

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

2.5 Documentation and Other Information

2.5.1 On the Product Medium

- For general product information, see the file README in the top level of the product medium.
- For a chronological log of all changes made to updated packages, see the file ChangeLog in the top level of the product medium.
- Detailed change log information about a particular package is available using RPM:

```
rpm --changelog -qp FILE_NAME.rpm
```

(Replace *FILE_NAME.rpm* with the name of the RPM.)

- For more information, see the directory docu of the product medium of SUSE Linux Enterprise High-Performance Computing 15 SP2.

2.5.2 Online Documentation


- Find a collection of White Papers in the SUSE Linux Enterprise High-Performance Computing Resource Library at <https://www.suse.com/products/server/hpc#resources> .

3 Modules, Extensions, and Related Products


This section comprises information about modules and extensions for SUSE Linux Enterprise High-Performance Computing 15 SP2. Modules and extensions add functionality to the system.

3.1 Modules in the SLE 15 SP2 Product Line

The SLE 15 SP2 product line is made up of modules that contain software packages. Each module has a clearly defined scope. Modules differ in their life cycles and update timelines.

The modules available within the product line based on SUSE Linux Enterprise 15 SP2 at the release of SUSE Linux Enterprise High-Performance Computing 15 SP2 are listed in the *Modules and Extensions Quick Start* at <https://documentation.suse.com/sles/15-SP2/html/SLES-all/art-modules.html> .

Not all SLE modules are available with a subscription for SUSE Linux Enterprise High-Performance Computing 15 SP2 itself (see the column *Available for*).


For information about the availability of individual packages within modules, see <https://scc.suse.com/packages> .

3.1.1 HPC Module

The HPC module contains HPC-specific packages. These include the workload manager Slurm, the node deployment tool **clustduct**, MUNGE for user authentication, the remote shell **mrsh**, the parallel shell **pdsh**, and numerous HPC libraries and frameworks.

This module is available with SUSE Linux Enterprise High-Performance Computing only. It is selected by default during the installation. It can be added or removed using the YaST UI or the **SUSEConnect** CLI tool. For more information, see the system administration guide.

3.1.2 NVIDIA Compute Module

The repositories for NVIDIA CUDA are available as the *NVIDIA Compute* module for x86-64 and AArch64. These repositories are provided by NVIDIA and the software in them is not supported by SUSE. All software in these repositories is licensed under the third-party [NVIDIA CUDA EULA](https://docs.nvidia.com/cuda/eula/index.html) (<https://docs.nvidia.com/cuda/eula/index.html>) .

The *NVIDIA Compute* module is not enabled by default when installing SUSE Linux Enterprise Server. During installation, the module can be selected manually from the *Extension and Modules* screen in YaST. Within an installed system, you can add it as follows: Run **yast registration** from a shell as root, select *Select Extensions*, search for *NVIDIA Compute Module* and continue with *Next*. Verify and accept the NVIDIA repository GPG key.



Important: Do Not Use the **SUSEConnect** Tool to Add this Repository


Do not try to add this module with the **SUSEConnect** CLI tool. **SUSEConnect** is not yet capable of handling third-party repositories.

! Important: Combining *Workstation Extension* and *NVIDIA Compute Module* Is Unsupported


The *Workstation Extension* module includes some of the same drivers for NVIDIA graphics cards as the *NVIDIA Compute* module. However, their package versions may differ. As SUSE package management installs the latest package versions by default, enabling both modules at the same time can lead to a system with a mixture of packages from both modules.








Such a setup can result in drivers not working as expected and is not supported by SUSE.

3.2 Available Extensions

The following extension is not covered by SUSE support agreements, available at no additional cost and without an extra registration key: SUSE Package Hub, see <https://packagehub.suse.com/> .

3.3 Related Products

This section lists related products. Usually, these products have their own release notes documents that are available from <https://www.suse.com/releasesnotes> .

- SUSE Linux Enterprise Server: <https://www.suse.com/products/server> 
- SUSE Linux Enterprise JeOS: <https://www.suse.com/products/server/jeos> 
- SUSE Enterprise Storage: <https://www.suse.com/products/suse-enterprise-storage> 
- SUSE Linux Enterprise Desktop: <https://www.suse.com/products/desktop> 
- SUSE Linux Enterprise Server for SAP Applications: <https://www.suse.com/products/sles-for-sap> 
- SUSE Linux Enterprise Real Time: <https://www.suse.com/products/realtime> 
- SUSE Manager: <https://www.suse.com/products/suse-manager> 

4 Technology Previews

Technology previews are packages, stacks, or features delivered by SUSE which are not supported. They may be functionally incomplete, unstable or in other ways not suitable for production use. They are included for your convenience and give you a chance to test new technologies within an enterprise environment.

Whether a technology preview becomes a fully supported technology later depends on customer and market feedback. Technology previews can be dropped at any time and SUSE does not commit to providing a supported version of such technologies in the future.

Give your SUSE representative feedback about technology previews, including your experience and use case.

5 Installation and Upgrade

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

5.1 System Roles for SUSE Linux Enterprise High-Performance Computing 15 SP2

With SUSE Linux Enterprise High-Performance Computing 15 SP2, it is possible to choose specific roles for the system based on modules selected during the installation process. When the HPC Module is enabled, these three 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 firewall and Kdump services
- Installs controller for the Slurm workload manager
- Mounts a large scratch partition to /var/tmp

HPC Compute Node

This role includes the following features:

- Uses XFS as the default root file system
- Includes HPC-enabled libraries
- Disables firewall and Kdump services
- Based from minimal setup configuration
- Installs 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 toolchain

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

The Environment Module Lmod will be installed for all roles. It is required at build time and run time of the system. For more information, see [Section 7.7, “Lmod — Lua-based Environment Modules”](#). All libraries specifically build for HPC will be installed under /usr/lib/hpc. They are not part of the standard search path, thus the Lmod environment module system is required.

munge authentication is installed for all roles. This requires copying the same generated munge keys to all nodes of a cluster. For more information, see [Section 7.15, “mrsh/mrlogin — Remote Login Using MUNGE Authentication”](#) and [Section 7.14, “MUNGE Authentication”](#).

From the Ganglia monitoring system, the data collector ganglia-gmod is installed for every role, while the data aggregator ganglia-gmetad needs to be installed manually on the system which is expected to collect the data. For more information, see [Section 7.3, “Ganglia — System Monitoring”](#).

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

5.2 Upgrade-Related Notes

This section includes upgrade-related information for the SUSE Linux Enterprise High-Performance Computing 15 SP2.

You can upgrade to SUSE Linux Enterprise High-Performance Computing 15 SP2 from SLES 12 SP3 or SUSE Linux Enterprise High-Performance Computing 12 SP3. When upgrading from SLES 12 SP3, the upgrade will only be performed if the SUSE Linux Enterprise High-Performance Computing module has been registered prior to upgrading. Otherwise, the system will instead be upgraded to SLES 15.

To upgrade from SLES 12 to SLES 15, make sure to deregister the SUSE Linux Enterprise High-Performance Computing module prior to upgrading. To do so, open a root shell and execute:

```
SUSEConnect -d -p sle-module-hpc/12/ARCH
```

Replace *ARCH* with the architecture used (*x86_64*, *aarch64*).

When migrating to SUSE Linux Enterprise High-Performance Computing 15 SP2, all modules not supported by the migration target need to be deregistered. This can be done by executing:

```
SUSEConnect -d -p sle-module-MODULE_NAME/12/ARCH
```

Replace *MODULE_NAME* by the name of the module and *ARCH* with the architecture used (*x86_64*, *aarch64*).

When migrating from SUSE Linux Enterprise High-Performance Computing 15 GA to SUSE Linux Enterprise High-Performance Computing 15 SP2, make sure to remove the Legacy Module if it is installed. Otherwise, the migration will fail with the error message *No migration product found*. To remove the legacy module, execute:

```
SUSEConnect -d -p sle-module-legacy/15/ARCH
```

Replace *ARCH* with the architecture used.


SUSE Linux Enterprise 15 uses Python 3 by default. Starting with SLE 15 SP1, the Python 2 runtime and modules have been moved to the *Python 2* module. As SUSE Linux Enterprise High-Performance Computing 15 SP2 uses Python 2, you need to enable this module when upgrading from earlier versions.

6 Software Changes

6.1 New Packages


6.1.1 Adaptable IO System (ADIOS) 1.13.1 Has Been Added

The Adaptable IO System (ADIOS) provides a simple, flexible way for scientists to describe the data in their code that may need to be written, read, or processed outside of the running simulation.

For more information about this release, see the change log at <https://www.olcf.ornl.gov/center-projects/adios/> . For more information about using ADIOS, see *Section 10.1, “Adaptable IO System (ADIOS)”*.

6.1.2 HPC Flavor of Boost Has Been Added

Boost is a set of portable C++ libraries which provide a reference implementation of “existing practices”.

See the full release notes for Boost 1.71 at https://www.boost.org/users/history/version_1_71_0.html . For information about using Boost, see *Section 9.1, “boost — Boost C++ Libraries”*.

6.1.3 HPC Flavor of pnetcdf Has Been Added

NetCDF is a set of software libraries and self-describing, machine-independent data formats that support the creation, access, and sharing of array-oriented scientific data. Parallel netCDF (PnetCDF) is a library providing high-performance I/O while still maintaining file-format compatibility with NetCDF by Unidata.

For information about using pnetcdf, see *Section 10.4, “HPC Flavor of pnetcdf”*.

6.2 Updated Packages

6.2.1 `netcdf-cxx4` Has Been Updated to Version 4.3.1

The update introduced the following enhancement:

- Added `ncFile::create()`, also added a new open function and constructor to allow for more flexibility when opening a file.

For full release notes, see https://github.com/Unidata/netcdf-cxx4/blob/master/RELEASE_NOTES.md#netcdf-cxx4-v431-september-12-2019.

6.2.2 `scalapack` Has Been Updated to Version 2.1

Version 2.1 of `scalapack` contains:

- Added robust ScaLAPACK routine for computing the QR factorization with column pivoting.
- Improved accuracy of Frobenius norm by making intermediate column sums.
- Miscellaneous minor bug fixes.

For further information, see the release notes of ScaLAPACK 2.1 <http://netlib.org/scalapack/scalapack-2.1.0.html>.



Note: BLACS API

`libblacs` is no longer packaged separately. The BLACS ABI is part of `libscalapack`.

6.2.3 `cpuid` Has Been Updated to Version 20180519

`cpuid` 20180519 adds support for various new CPUs and makes available a number of new bit fields.

6.2.4 Support for EFA in AWS

Amazon Web Services (AWS) has introduced the EFA driver to support HPC workloads. The Linux kernel of SUSE Linux Enterprise High-Performance Computing now supports this driver.

6.2.4.1 libfabric Has been Updated to Version 1.9.0

For more information about the updates in libfabric 1.9.0, see <https://github.com/ofiwg/libfabric/blob/e2956e/NEWS.md> ↗.

6.2.4.2 rdma-core Has Been Updated to Version 27.1

For more information about the updates in rdma-core 27.1, see <https://github.com/linux-rdma/rdma-core/releases/tag/v27.1> ↗.

6.2.4.3 openmpi3 Has Been Updated to Version 3.1.6

For more information about the bug fixes in openmpi3 3.1.6, see <https://github.com/open-mpi/ompi/blob/ea34872/NEWS> ↗.

6.2.5 fftw3 Has Been Updated to Version 3.3.8

fftw3 3.3.8 includes official support for the AArch64 architecture.

6.2.6 gsl Has Been Updated to Version 2.6

For more information about the updates in gsl 2.6, see <http://git.savannah.gnu.org/cgit/gsl.git/tree/NEWS> ↗.

6.2.7 hdf5 Has Been Updated to Version 1.10.5

For more information about the updates in hdf5 1.10.5, see <https://support.hdfgroup.org/ftp/HDF5/releases/hdf5-1.10/hdf5-1.10.5/src/hdf5-1.10.5-RELEASE.txt> ↗.

6.2.8 hwloc Has Been Updated to Version 2.1.0

Version 2.1.0 of hwloc brings numerous improvements:

- There are two new object types: “Die” for Xeon 9200 processors, and “MemorySideCache” (used when DDR is a cache in front of NVDIMMs).
- There is better support for discovering the locality of memory.

- `lstopo` received a range of improvements: better graphical output, factorization of identical objects, and many keyboard shortcuts to configure interactive outputs.
- There are now Bash completions that ease command-line invocations.

For more information, see <https://www.mail-archive.com/hwloc-announce@lists.open-mpi.org/msg00127.html>.

Other changes include:

- New library ABI, with the following major changes:
 - Memory, I/O and Misc objects now have their own children list.
 - Topologies always have at least one NUMA object.
 - `HWLOC_OBJ_CACHE` object disambiguated to `HWLOC_OBJ_L1-5CACHE` and `HWLOC_OBJ_I1-5CACHE`.
 - Reworked ignoring/filtering API.
 - The distance API has been completely reworked.
 - Most functions in `hwloc/bitmap.h` now return an integer informing about success or or allocation failure.
 - Add `hwloc/shmem.h` for sharing topologies between processes running on the same machine.
 - Add the experimental Portable Network Locality (`netloc`) subproject.
- Support for the new sysfs structure of Linux 5.3.
- New objects `HWLOC_OBJ_DIE`, and distance-related features: `HWLOC_DISTANCES_KIND_HETEROGENEOUS_TYPES`.
- New topologies: `HWLOC_OBJ_MEMCACHE`, `HWLOC_OBJ_MEMCACHE`, `HWLOC_HWLOC_OBJ_MEMCACHE`.

Details can be found at <https://github.com/open-mpi/hwloc/blob/f1a2e22/NEWS>.

6.2.9 `imb` Has Been Updated to Version 2019.3

Among the changes are:

- Added the `warm_up` option that enables additional cycles before running benchmark (for all sizes).
- Added the `Reduce_local` benchmark for IMB-MPI1.
- Added IMB-P2P benchmarks.
- Bug fixes.

For details, see <https://github.com/intel/mpi-benchmarks/blob/IMB-v2019.3/README.md>.

6.2.10 `HYPRE` Has Been Updated to Version 2.18.2

`HYPRE` 2.18.2 includes bug fixes and some improvements, such as GPU optimizations. For more information, see <https://github.com/hypre-space/hypre/blob/master/CHANGELOG>.

6.2.11 `Lmod` Has Been Updated to Version 8.2.5

This update includes the following major changes:

- `extension()` function - allow modules to specify extensions that provide `import`, that is a list of other packages <https://sourceforge.net/p/lmod/mailman/message/36796576/>.
- Extended Defaults: Allows partial matching of the version to pick the most appropriate module. Users of a non-hierarchical module setup can disable this feature by setting the environment variable `LMOD_EXTENDED_DEFAULT=no`. For details, see <https://sourceforge.net/p/lmod/mailman/message/36660135/>.
- Support for Lua-based `modulerc` files. For details, see <https://sourceforge.net/p/lmod/mailman/message/36374560/>.
- Fix for `sh_to_modulefile.sh` which used to fail when unknown options were passed. For details, see <https://sourceforge.net/p/lmod/mailman/message/36118404/>.
- Supports reference counting for `PATH`-like variables. For details, see <https://sourceforge.net/p/lmod/mailman/message/36045411/>.

For other changes, see <https://github.com/TACC/Lmod/blob/8.2.5/README.md>.

6.2.12 memkind Has Been Updated to Version 1.9.0

memkind 1.9.0 includes a number of fixes and improvements over the 1.6.x version. Among these:

- Fixed issue during dynamic loading.
- Added C++ allocator for file-backed memory (PMEM kind).
- Extended memkind API with `memkind_malloc_usable_size()`.
- Added support to create `kind` without maximum size limit of PMEM kind (`max_size=0`).
- Provided the `memkind_detect_kind()` function to recognize a kind from memory allocation.
- Provided support for Dockerfiles.
- Unified the `memkind_realloc()` behavior for all kinds.

For more information, see <https://github.com/memkind/memkind/releases/tag/v1.9.0> and <https://github.com/memkind/memkind/blob/v1.9.0/ChangeLog>.

6.2.13 MPICH Has Been Updated to Version 3.3.2

MPICH 3.3.2 adds a number of improvements and bug fixes, highlights are:

- Support for the struct `sockaddr` in MPICH, Hydra, and PMI socket code.
- A new CH4 device layer implementation designed for low software overhead.
- Support for the PMIx client library in CH4 netmods. See also [Section 6.2.23, “Support for PMIx Has Been Added to Slurm and MPI libraries”](#) and <https://pmix.github.io/pmix/>.

For more information, see the `README` file in the package `mpich`.

6.2.14 MUMPS Has Been Updated to Version 5.2.1

MULTifrontal Massively Parallel sparse direct Solver (MUMPS) 5.2.1 includes bug fixes, improved distributed memory usage and MPI granularity, and reduced memory usage due to low-rank factorization. For more information, see <http://mumps.enseeiht.fr/index.php?page=dwnld#cl>.

6.2.15 mvapich2 Has Been Updated to Version 2.3.3

For more information about the updates in mvapich2 2.3.3, see <http://mvapich.cse.ohio-state.edu/overview/>.

6.2.16 netcdf Has Been Updated to Version 4.7.3

- Provide byte-range reading of remote data sets.
- Add a function for changing the ncid of an open file.
- Corrected assignment of anonymous (that is, phony) dimensions in an HDF5 file.
- Add a dispatch version number to `netcdf_meta.h` and `libnetcdf.settings`.
- Now testing that endianness can only be set on atomic `ints` and `floats`.
- Various bug fixes.

For the full release notes, see <https://github.com/Unidata/netcdf-c/releases/v4.7.3>.

6.2.17 Trilinos Has Been Updated to Version 12.14.1

For details on the changes to the individual component (packages) in Trilinos itself, see the respective packages.

6.2.18 petsc Has Been Updated to Version 3.12

Various subsystems have received updates. For the list of changes, see:

- Release notes version 3.12 (<https://www.mcs.anl.gov/petsc/documentation/changes/312.html>)
- Release notes version 3.11 (<https://www.mcs.anl.gov/petsc/documentation/changes/311.html>)
- Release notes version 3.10 (<https://www.mcs.anl.gov/petsc/documentation/changes/310.html>)
- Release nodes version 3.9 (<https://www.mcs.anl.gov/petsc/documentation/changes/39.html>)

6.2.19 netcdf-cxx4 Has Been Updated to Version 4.3.1

The netcdf-cxx4 4.3.1 adds ncFile::create() and a new open function and constructor to allow for more flexibility when opening a file. For more information, see <https://www.unidata.ucar.edu/blogs/news/entry/netcdf-c-4-3-1>.

6.2.20 netcdf-fortran Has Been Updated to Version 4.5.2

For more about updates in netcdf-fortran 4.5.2, see the change logs at:

- <https://github.com/Unidata/netcdf-fortran/releases/tag/v4.5.2>
- <https://github.com/Unidata/netcdf-fortran/releases/tag/v4.5.1>
- <https://github.com/Unidata/netcdf-fortran/releases/tag/v4.5.0>

6.2.21 OpenBLAS Has Been Updated to Version 0.3.7

For more information about the updates in OpenBLAS 0.3.7, see <https://www.openblas.net/Changelog.txt>.

6.2.22 PAPI Has Been Updated to Version 5.7.0

PAPI 5.7.0 includes a new component, called pcp which interfaces to the Performance Co-Pilot (PCP). This release also upgrades the PAPI nvml component with write access to the information and controls exposed via the NVIDIA Management Library. For more information, see <https://icl.cs.utk.edu/papi/news/news.html?id=381>.

6.2.23 Support for PMIx Has Been Added to 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 utilize the individual MPI launchers on Slurm anymore, srun will take care of this. In addition, the workload manager can determine the topology of the cluster. This removes the need for users to specify topologies manually.

6.2.24 NumPy Has Been Updated to Version 1.16

NumPy 1.16 ([python3-numpy](#)) includes several new features and bug fixes, including:

- Added an extensible random module along with four selectable random number generators and improved seeding designed for use in parallel processes.
- Added radix sort and Timsort sorting methods.

For more information, see <https://numpy.org/doc/1.16/release.html>.

6.2.25 python3-scipy Has Been Updated to Version 1.33

SciPy 1.33 ([python3-scipy](#)) includes several new features and bug fixes, including:

- Added a new extensible random module along with four selectable random number generators and improved seeding designed for use in parallel processes.
- For the Attribute-Relation File Format (ARFF), added several relational attributes to `scipy.io.arff.loadarff`.
- Removed several functions from `scipy.interpolate` (`spleval`, `spline`, `splmake`, and `spltopp`) and from `scipy.misc` (`bytescale`, `fromimage`, `imfilter`, `imread`, `imresize`, `imrotate`, `imsave`, `imshow`, `toimage`).
- Added a new function to compute the Epps-Singleton test statistic, `scipy.stats.epps_singleton_2samp`, which can be applied to continuous and discrete distributions.
- Added new functions `scipy.stats.median_absolute_deviation` and `scipy.stats.gstd` (geometric standard deviation). The `scipy.stats.combine_pvalues` method now supports `pearson`, `tippett` and `mudholkar_george` pvalue combination methods.
- `scipy.spatial.ConvexHull` now has a `good` attribute that can be used alongside the `Qn` `qhull` options to determine which external facets of a convex hull are visible from an external query point.
- Modernized `scipy.spatial.cKDTree.query_ball_point` to use certain newer Cython features, including GIL handling and exception translation.
- Added new keyword to `csgraph.dijkstra` that allows users to query the shortest path to any of the passed-in indices, as opposed to the shortest path to every passed index.

- Added the keyword `workers` to the `scipy.optimize.brute` minimizer, which can be used to parallelize computation.
- Removed a dependency on `np.polynomial`: Gaussian filter improves performance by up to an order of magnitude. This may impact `scipy.ndimage.gaussian_filter`, for example.
- Added wrappers for `?syconv` routines which convert a symmetric matrix given by a triangular matrix factorization into two matrices and vice versa.
- `scipy.linalg.clarkson_woodruff_transform` now uses an algorithm that leverages sparsity. This provides a speedup of up to 90 % for dense input matrices.

For more information, see:

- Release Notes SciPy 1.3.3 (<https://docs.scipy.org/doc/scipy/reference/release.1.3.3.html>) ↗
- Release Notes SciPy 1.3.2 (<https://docs.scipy.org/doc/scipy/reference/release.1.3.2.html>) ↗.
- Release Notes SciPy 1.3.1 (<https://docs.scipy.org/doc/scipy/reference/release.1.3.1.html>) ↗
- Release Notes SciPy 1.3.0 (<https://docs.scipy.org/doc/scipy/reference/release.1.3.0.html>) ↗
- Release Notes SciPy 1.2.3 (<https://docs.scipy.org/doc/scipy/reference/release.1.2.3.html>) ↗
- Release Notes SciPy 1.2.2 (<https://docs.scipy.org/doc/scipy/reference/release.1.2.2.html>) ↗
- Release Notes SciPy 1.2.1 (<https://docs.scipy.org/doc/scipy/reference/release.1.2.1.html>) ↗

`python-scipy 1.3.3` is available for Python 3 only. The last `python-scipy` version shipped with Python 2 support was 1.2.2 which is provided for backward compatibility.

6.2.26 Slurm Has Been Updated to Version 20.02

Slurm is shipped in version 20.02, meaning the same version of Slurm is now used across SUSE Linux Enterprise High-Performance Computing versions. Before upgrading your Slurm installation to a new major version, make sure to read *Section 7.11.2, “Slurm Upgrade Compatibility”* and *Section 7.11.3, “Upgrading Slurm”*.

6.2.26.1 New Subpackage `slurm-webdoc`

`slurm-webdoc` installs a web server and the `slurm-doc` package, so you can view Slurm documentation. It also contains a tool to help you generate a `slurm.conf`.

6.2.26.2 Adding a Cluster to the Database

When using `slurmdbd` make sure a table for a cluster is added to the database before `slurmctld` is started (or restart it afterwards). Otherwise, no accounting information may be written to the database.

To add a cluster table, run: `sacctmgr -i add cluster CLUSTERNAME`.

6.2.26.3 Important Slurm Configuration Changes

Version 17.11:

- `slurm.conf`:
 - Added `SchedulerParameters` configuration option `disable_hetero_steps` to disable job steps that span multiple components of a heterogeneous job.
 - Added `SchedulerParameters` configuration option `enable_hetero_steps` to enable job steps that span multiple components of a heterogeneous job.
 - Added `PrivateData=events` configuration parameter.
 - Added `SlurmctldSyslogDebug` and `SlurmdSyslogDebug` to control which messages from the `slurmctld` and `slurmd` daemons get written to syslog.
- `slurmdbd.conf`: Added `DebugLevelSyslog` to control which `slurmdbd` messages get written to syslog.
- `cgroup.conf`: Added `MemorySwappiness` value.
- `cgroup.conf`: Added `MemorySwappiness` value.
- Plugins: Removed obsolete MPI plugins, checkpoint/poe plugin.
- `srun`: remove `--mpi-combine` option.
- `scontrol`: `top` disabled for regular user. Set `SchedulerParameters=enable_user_top` to override.
- `scancel`: Added `--hurry` option to avoid staging out any burst buffer data.
- `queue`: Added `--local` and `--sibling` options to modify filtering of jobs on federated clusters.

Version 18.08:

- slurm.conf:
 - Removed NoOverMemoryKill from JobAcctGatherParams, as this is now the default. To change this to the old default, set JobAcctGatherParams NoOverMemoryKill.
 - ControlMachine, ControlAddr, BackupAddr, BackupController These options have been deprecated and replaced by SlurmctldHost. You can configure multiple SlurmctldHosts. The first one will be the used as master controller, subsequent ones are used as fallbacks in the order specified. ControlAddr and BackupAddr can be specified enclosed in parentheses after the host name. For details, see slurm.conf.
 - SelectType: The argument CR_ALLOCATE_FULL_SOCKET is now the default for CR_SOCKET*. It should be removed.
- Deprecated arguments to utilities:
 - srun, sbatch, salloc:
 - The option -P has been deprecated, use -d instead
 - The option --minicores has been deprecated, use --cores-per-socket
 - The option --minisockets has been deprecated, use --sockets-per-node
 - The option --minithreads has been deprecated, use --threads-per-core
 - salloc: The option -W has been deprecated, use --immediate instead
 - sacct: The option -C has been deprecated, use -M instead
 - slurmctld: The option -t has been removed, use -X instead.

Version 19.05:

- slurm.conf:
 - The parameter FastSchedule has been deprecated.
 - The parameter MemLimitEnforce has been removed. Functionality has been moved into JobAcctGatherParam=OverMemoryKill.
 - The option CryptoType has been changed to CredType, crypto/munge plugin has been renamed to cred/munge.
- Deprecated arguments to utilities:
 - sreport: Default behavior of SizesByAccount and SizesByAccountAndWckey has been changed, a new AcctAsParent option has been added.
 - gres.conf: CPUs parameter is deprecated, use Cores instead.
 - srun, sbatch:
 - The Slurm_NPROCS environment variable has been deprecated, use Slurm_NTASKS instead.
 - salloc, sbatch, srun:
 - The argument -U has been removed—it was deprecated when -A was made the single character option before the Slurm 2.1 release—as an alternative to --account

Version 20.02:

- slurm.conf:
 - slurmctld will terminate with a fatal error if compute nodes are configured with CPUs == #Sockets. CPUs has to be either the total number of cores or threads.
 - Option FastSchedule has been removed.
 - New parameter AccountingStorageExternalHost.
 - Option kill_invalid_depend and max_depend_depth have been moved to the new DependencyParameters option. Use in SchedulerParameters is now deprecated.

- Removed deprecated `max_job_bf` `SchedulerParameters` and replaced it with `bf_max_job_test`.
- `MaxDBDMsgs` allows to specify how many messages will be stored in `slurmctld` when `slurmdbd` is down.
- New `NodeSet` configuration option to simplify partition configuration sections for heterogeneous/condo-style clusters.
- `smap`: `smap` has been removed.
- `sbatch`, `salloc`, `srun`: `--reboot` disabled for non-admins.
- `sbcast`: Allow alternative path for `sbcast` traffic using the `BcastAddr` option to `NodeName` lines to allow `sbcast` traffic.

6.2.27 MUNGE Has Been Updated to Version 0.5.14

The update includes:

- Added negative caching of user lookups for processing supplementary groups.
- Added `--origin` and `--stop` command-line options to `munged`.
- Added `--numeric` command-line option to `unmunge`.
- Added several configuration options.
- Improved logging: Non-existing users are only logged once.
- Bug fixes.

6.2.28 conman Has Been Updated to Version 0.3.0

This update includes a bug fix that has been previously released in a maintenance update.

This update fixes slow connects to Unix socket consoles triggered from `inotify`.

6.2.29 genders Has Been Updated to Version 1.27.3

This update adds the following features:

- Improve file parsing speed.
- Support `--compress-hosts` option in `nodeattr`, to compress genders database by hosts rather than `attrs`.
- Support `--compress-attrs` option as an alias of `--compress` for consistency.
- Various bug fixes.

6.2.30 clustduct Has Been Updated to Version 0.0.5

- Integrated support for SLE 15 SP2.
- Numerous bug fixes.

6.2.31 ganglia-web Has Been Updated to Version 3.7.4

- Added graph for disk space (Used/Avail/Total).
- Added graph for I/O read and write, from `mod_io`.
- Added time fields to the aggregate graphs.
- Added div for metric, as an anchor for metric search.
- Allowed adding default graphs on host view only.
- Improved performance of the host view by only refreshing graphs that are visible in the browser viewport.
- Improvements to the dialog content for adding items to views. For example, only display warning/critical fields for metric graphs.
- Add support for specifying the timezone to use for displaying performance data.
- Improve user experience for selecting metrics as part of building aggregate graphs.
- Improve and fix `cpu_guest` and `cpu_gnice` (`guest_nice`) display.
- Multiple bug fixes.

6.2.32 `scotch` Has Been Update to Version 6.0.9

This update brings a number of bug fixes. For more information, see https://gforge.inria.fr/frs/?group_id=248.

6.3 Removed Packages and Functionality

6.3.1 `ohpc` Has Been Removed

The package `ohpc` provided RPM macros for compatibility with the OpenHPC project. It is no longer needed.

7 HPC Tools

7.1 `cpuid` — x86 CPU Identification Tool

`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 also supports Intel Knights Mill CPUs (x86-64).

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.

7.2 `ConMan` — The Console Manager

`ConMan` is a serial console management program designed to support a large number of 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 a large number of such consoles.

Usage:

- ConMan comes with a number of expect-scripts: check /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 in interactive mode, no line feed is generated. To generate a line feed, press .

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

7.3 Ganglia — System Monitoring

Ganglia is a scalable distributed monitoring system for high-performance computing systems, such as clusters and grids. It is based on a hierarchical design targeted at federations of clusters.

Using Ganglia

To use Ganglia, make sure to install ganglia-gmetad on the management server. Then start the Ganglia meta-daemon: rcgmead start. To make sure the service is started after a reboot, run: systemctl enable gmetad. On each cluster node which you want to monitor, install

ganglia-gmond, start the service rcgmond start and make sure it is enabled to be started automatically after a reboot: systemctl enable gmond. To test whether the gmond daemon has connected to the meta-daemon, run gstat -a and check that each node to be monitored is present in the output.

Ganglia on Btrfs

When using the Btrfs file system, the monitoring data will be lost after a rollback and the service gmetad will not start again. To fix this issue, either install the package ganglia-gmetad-skip-bcheck or create the file /etc/ganglia/no_btrfs_check.

Using the Ganglia Web Interface

Install ganglia-web on the management server. Depending on which PHP version is used (default is PHP 7), enable it in Apache2: a2enmod php7.

Then start Apache2 on this machine: rcapache2 start and make sure it is started automatically after a reboot: systemctl enable apache2. The Ganglia Web interface should be accessible from http://MANAGEMENT_SERVER/ganglia-web.

7.4 Genders — Static Cluster Configuration Database

Support for Genders has been added to the HPC module.

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

Perl, Python, Lua, C, and C++ bindings are supplied with Genders, the respective packages provide man pages or other documentation describing the APIs.

To create the Genders database, follow the instructions and examples in /etc/genders and check /usr/share/doc/packages/genders-base/TUTORIAL. Testing a configuration can be done with nodeattr (for more information, see man 1 nodeattr).

List of packages:

- genders
- genders-base

- [genders-devel](#)
- [python-genders](#)
- [genders-perl-compat](#)
- [libgenders0](#)
- [libgendersplusplus2](#)

7.5 GNU Compiler Collection for HPC

On SLE-HPC the GNU compiler collection version 7 is provided as the base compiler. The [gnu-compilers-hpc](#) 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.

7.5.1 Environment Module

This package requires [lua-lmod](#) to supply environment module support.

To install [gnu-compilers-hpc](#), run:

```
zypper in gnu-compilers-hpc
```

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

```
module load gnu
```

7.5.2 Building HPC Software with GNU Compiler Suite

To use the GNU compiler collection to build your own libraries and applications, [gnu-compilers-hpc-devel](#) needs to be installed. It makes sure 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 will be found.

7.5.3 Later Versions

The Development Tools module may provide later versions of the GNU compiler suite. To determine the available compiler suites, run:

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

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

```
module load gnu/7
```

For more information, see [Section 7.7, “Lmod — Lua-based Environment Modules”](#).

7.6 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 even 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. Additionally it can assemble the topologies of multiple machines into a single one, to let applications consult 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, several other output formats are available as well, including plain text, PDF, PNG, SVG and FIG. For more information, see the man pages provided by hwloc and lstopo.

It also 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.

7.7 Lmod — Lua-based Environment Modules

Lmod is an advanced environment module system which 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 version of dependent libraries are selected. Environment Modules-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.

Installation and Basic Usage

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

Before Lmod can be used, an init file needs to be sourced from the initialization file of your interactive shell. The following init files are available:

```
/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 to the init file of your shell:

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

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

Listing Available Modules

To list 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 (marked by *). 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.

Listing Loaded Modules

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

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) such 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.

Loading Modules

The `module load MODULE` command needs to 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.

Environment Variables

If the respective development packages are installed, build-time environment variables like `LIBRARY_PATH`, `CPATH`, `C_INCLUDE_PATH` and `CPLUS_INCLUDE_PATH` will be set up to include the directories containing the appropriate header and library files. However, some compiler and linker commands may 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.

For More Information

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

7.8 `pdsh` — Parallel Remote Shell Program

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

To install `pdsh`, run `zypper in pdsh`.

On SLE-HPC, the back-ends `ssh`, `mrsh`, and `exec` are supported. The `ssh` back-end is the default. Non-default login methods can be used by either setting the `PDSH_RCMD_TYPE` environment variable or by using the `-R` command argument.

When using the `ssh` back-end, it is important that a non-interactive (that is, passwordless) login method is used.

The `mrsh` back-end requires the `mrshd` to be running on the client. The `mrsh` back-end does not require the use of reserved sockets. Therefore, it does not suffer from port exhaustion when executing commands on many machines in parallel. For information about setting up the system to use this back-end, see [Section 7.15, “mrsh/mrlogin — Remote Login Using MUNGE Authentication”](#).

Remote machines can be specified on the command line. Alternatively, `pdsh` can use a `machines` file (`/etc/pdsh/machines`), `dsh`-style (Dancer's shell) groups, or netgroups. It can target also 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 recommend limiting yourself to 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 unnecessary dependencies. Package dependencies have been set up to prevent the installation of plugins with conflicting command options. To install one of the plugins, run:

```
zypper in pdsh-PLUGIN_NAME
```

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

7.9 PowerMan — Centralized Power Control for Clusters

PowerMan allows manipulating remote power control devices (RPC) from a central location. It can control:

- local devices connected to a serial port
- RPCs listening on a TCP socket
- RPCs which 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 it, 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 need to match an entry in the “plug name” list.

After configuring PowerMan, start its service by:

```
systemctl start powerman.service
```

To start PowerMan automatically after every boot, do:

```
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

Data is transferred unencrypted, therefore this is not recommended unless the network is appropriately secured.

7.10 rasdaemon — Utility to Log RAS Error Tracings

rasdaemon is a RAS (Reliability, Availability and Serviceability) logging tool. It records memory errors using the EDAC tracing events. EDAC drivers in the Linux kernel handle detection of ECC 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 board.

To check whether the EDAC drivers are loaded, execute:

```
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, execute `systemctl enable rasdaemon.service`. The daemon will log information to `/var/log/messages` and to an internal database. A summary of the stored errors can be obtained with:

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

7.11 Slurm — Utility for HPC Workload Management

Slurm is an open-source, fault-tolerant, and highly scalable cluster management and job scheduling system for Linux clusters containing up to 65,536 nodes. Components include machine status, partition management, job management, scheduling and accounting modules.

7.11.1 Installing Slurm

For a minimal setup to run Slurm with MUNGE support on one compute node and multiple control nodes, follow these instructions:

1. Before installing Slurm, create a user and a group called `slurm`.

! 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.

2. Install `slurm-munge` on the control and compute nodes: **`zypper in slurm-munge`**.
3. Configure, enable and start `munge` on the control and compute nodes as described in [Section 7.15, “mrsh/mrlogin — Remote Login Using MUNGE Authentication”](#).
4. On the compute node, edit `/etc/slurm/slurm.conf`:

- a. Configure the parameter `ControlMachine=CONTROL_MACHINE` with the host name of the control node.

To find out the correct host name, run `hostname -s` on the control node.

- b. Additionally add:

```
NodeName=NODE_LIST Sockets=SOCKETS \  
CoresPerSocket=CORES_PER_SOCKET \  
ThreadsPerCore=THREADS_PER_CORE \  
State=UNKNOWN
```

and

```
PartitionName=normal Nodes=NODE_LIST \  

```

```
Default=YES MaxTime=24:00:00 State=UP
```

Replace the following parameter values:

- NODE_LIST denotes the list of compute nodes. That is, it should contain the output of `hostname -s` run on each compute node, either comma-separated or as ranges (for example, `foo[1-100]`).
- SOCKETS denotes the number of sockets.
- CORES_PER_SOCKET denotes the number of cores per socket.
- THREADS_PER_CORE denotes the number of threads for CPUs which can execute more than one thread at a time.

Make sure that SOCKETS * CORES_PER_SOCKET * THREADS_PER_CORE does not exceed the number of system cores on the compute node.

- c. On the control node, copy `/etc/slurm/slurm.conf` to all compute nodes:

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

- d. On the control node, start `slurmctld`:

```
systemctl start slurmctld.service
```

Also enable it so that it starts on every boot:

```
systemctl enable slurmctld.service
```

- e. On the compute nodes, start and enable `slurmd`:

```
systemctl start slurmd.service  
systemctl enable slurmd.service
```

The last line causes `slurmd` to be started on every boot automatically.

7.11.2 Slurm Upgrade Compatibility

New major versions of Slurm are released in regular intervals. With some restrictions (see below), interoperability is guaranteed between 3 consecutive versions. However, unlike updates to maintenance releases (that is, releases which differ in the last version number), upgrades to newer major versions may require more careful planning.

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

On new installations, we recommend installing the latest available version.

Slurm uses a segmented version number: The first two segments denote the major version, the final segment denotes the patch level.

Check the list below for available major versions.

Upgrade packages (that is, packages that were not a part of the module or service pack initially) have their major version encoded in the package name (with periods . replaced by underscores _). For example, for version 18.08, this would be: slurm_18_08-*.rpm.

To upgrade the package slurm to 18.08, run the command:

```
zypper install --force-resolution slurm_18_08
```

To upgrade Slurm subpackages, proceed analogously.

In addition to the “three-major version rule” mentioned at the beginning of this section, obey the following rules regarding the order of updates:

1. The version of slurmdbd must be identical to or higher than the version of slurmctld
2. The version of slurmctld must be 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:

$\text{version}(\text{slurmdbd}) \geq \text{version}(\text{slurmctld}) \geq \text{version}(\text{slurmd}) \geq \text{version}(\text{Slurm user CLIs})$.

With each version, configuration options for `slurmctld`/`slurmd` or `slurmdbd` may be deprecated. While deprecated, they will remain valid for this version and the two subsequent versions but they may be removed later.

Therefore, it is advisable to update the configuration files after the upgrade and replace deprecated configuration options before finally restarting a service.

7.11.3 Upgrading Slurm

For this workflow it is assumed that MUNGE authentication is used and that `pdsh`, the `pdsh` Slurm plugin and `mrsh` can be used to access all machines of the cluster. That means, `mrshd` is running on all nodes in the cluster.

If this is not the case, install `pdsh`:

```
# zypper in pdsh-slurm
```

If `mrsh` is not used in the cluster, the SSH back-end for `pdsh` can be used as well for this: Replace the option `-R mrsh` with `-R ssh` in the `pdsh` commands below. This is less scalable and you may run out of usable ports.

PROCEDURE 1: UPGRADING SLURM

1. Upgrade `slurmdbd` Database Daemon

If the database daemon `slurmdbd` is used, it must be upgraded first. If the same database is used for multiple clusters, be careful. The database needs to be updated before any cluster is updated.

Upon the first start of `slurmdbd` after a `slurmdbd` upgrade, it will convert its database. If the database is large, the conversion will take several 10s of minutes. During this time, the database is not accessible.

We strongly recommend creating 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 in such an event. A database converted to a newer version cannot be converted back to an older one and older versions of `slurmdbd` will not recognize the newer formats. To back up and upgrade `slurmdbd`, follow this procedure:

- a. Stop the `slurmdbd` service:

```
# rcslurmdbd stop
```


Make sure that slurmdbd is not running anymore:

```
# rcslurmdbd status
```

- b. Create a backup of the slurm_acct_db database:

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



If needed, this can be restored by running:

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

- c. In preparation of the conversion, make sure the variable innodb_buffer_size is set to a value ≥ 128 Mb:

On the database server, run:

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

If the size is less than 128 Mb, it can be changed on the fly for the current session (on mariadb):

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

Alternatively, the size can be changed permanently by editing /etc/my.cnf and setting it to 128 Mb. Then restart the database:

```
# rcmysql restart
```

- d. Install the upgrade of slurmdbd:

```
zypper install --force-resolution slurm_version-slurmdbd
```



Note: Update MariaDB Separately

If you also need to update mariadb, it is recommended to perform this step separately, *before* performing [Step 1.d](#).

i. Upgrade MariaDB:

```
# zypper update mariadb
```

ii. Run the conversion of the database tables to the new version of MariaDB:

```
mysql_upgrade --user=root --password=root_db_password;
```

e. Rebuild database

Because a conversion can take a considerable amount of time, the systemd service can run into a timeout during the conversion. Thus we recommend to perform the migration manually by running `slurmdbd` from the command line in the foreground:

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

When you see the below message, `slurmdbd` can be shut down:

```
Conversion done:
success!
```

To do so, use signal `SIGTERM` (that is, press `Ctrl-C`).

f. Before restarting the service, remove or replace deprecated configuration options. For a list of deprecated options, see [Section 6.2.26.3, “Important Slurm Configuration Changes”](#).

When this has been completed, restart `slurmdbd`. During the database rebuild that `slurmdbd` performs upon the first start, it will not daemonize.



Note: Convert Primary slurmdbd First

If a backup database daemon is used, the primary one needs to be converted first. The backup will not start until this has happened.

Only after the conversion has been completed, the backup will start.

2. Update slurmd and slurmd

When the Slurm database has been updated, the `slurmctld` and `slurmd` instances can be updated. We recommend updating the head and compute nodes all in a single pass. If this is not feasible, the compute nodes (`slurmd`) can be updated on a node-by-node basis. However, this requires that the master nodes (`slurmctld`) have been updated successfully.

a. Back up the `slurmctld`/`slurmd` configuration

It is advisable to create a backup copy of the Slurm configuration before starting the upgrade process. Since the configuration file `/etc/slurm/slurm.conf` should be identical across the entire cluster, it is sufficient to do so on the master controller node.

b. Increase Timeouts

Set `SlurmdTimeout` and `SlurmctldTimeout` in `/etc/slurm/slurm.conf` to sufficiently high values to avoid timeouts while `slurmctld` and `slurmd` are down. We recommend at least 60 minutes, more on larger clusters.

- i. Edit `/etc/slurm/slurm.conf` on the master controller node and set the values for this variable to at least `3600` (1 hour).

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

- ii. Copy `/etc/slurm/slurm.conf` to all nodes. If MUNGE authentication is used in the cluster as recommended, use these steps:

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

B. Execute:

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

C. Verify that the reconfiguration took effect:

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

c. Shut down any running slurmctld instances

- i. If applicable, shut down any backup controllers on the backup head nodes:

```
backup: # systemctl stop slurmctld
```

- ii. Shut down the master controller:

```
master: # systemctl stop slurmctld
```

d. Back up the slurmctld state files

Almost every major version involves changes to the slurmctld state files. The format of older releases is understood if it is not older than 2 major versions.

Old versions will not understand a newer format, thus it is useful to back up the old state in case an update needs to be rolled back. Otherwise, the content will not be understood on downgrade and will be discarded. This will in turn lead to the loss of all pending and running jobs.

- i. Determine the StateSaveLocation directory:

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

- ii. Create a backup of the content of this directory to be able to roll back the update if an issue arises.

Should a downgrade be required, make sure to restore the content of the StateSaveLocation directory from this backup.

e. Shut down slurmd on the nodes

```
# pdsh -R ssh -P partitions systemctl stop slurmd
```

f. Update slurmctld on the master and backup nodes as well as slurmd on the compute nodes

- i. On the master/backup node(s), run:

```
master: # zypper install \  
--force-resolution slurm_version
```

- ii. On the master node, run:

```
master: # pdsh -R ssh -P partitions \  
zypper install --force-resolution \  
slurm_version-node
```

g. Replace deprecated options

If deprecated options need to be replaced in the configuration files (see the list in [Section 6.2.26.3, “Important Slurm Configuration Changes”](#)), this can be performed before updating the services. These configuration files can be distributed to all controllers and nodes of the cluster by using the method described in [Step 2.b.ii](#).



Note: Memory Size Seen by `slurmd` Can Change on Update

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

```
node1: # slurmd -C
```

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

h. Restart `slurmd` on all compute nodes

On the master controller, run:

```
master: # pdsh -R ssh -P partitions \  
systemctl start slurmd
```

On the master, run:

```
master: # systemctl start slurmctld
```

Then execute the same on the backup controller(s).

i. Verify whether the system operates properly

- i. Check the status of the controller(s).

On the master and backup controllers, run:

```
# systemctl status slurmctld
```

- ii. Verify that the services are running without errors using:

```
sinfo -R
```

You will see whether there are any down, drained, failing, or failed nodes after the restart.

j. Clean up

Restore the `SlurmdTimeout` and `SlurmctldTimeout` values in `/etc/slurm/slurm.conf` on all nodes (see [Step 2.b](#)).

Each service pack of SUSE Linux Enterprise High-Performance Computing includes a new major version of `libslurm`. The old version will not be uninstalled on upgrade. User-provided applications will work, as long as the library version used for building is no more than two major versions behind. We strongly recommend rebuilding local applications using `libslurm`—such as MPI libraries with Slurm support—as early as possible. This can require updating user application if new arguments were introduced to existing functions.

7.11.4 For More Information

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

7.12 Enabling the `pam_slurm_adopt` Module

The `pam_slurm_adopt` module allows restricting access to compute nodes to those users that have jobs running on them. It can also take care of *run-away processes* from user's jobs and end these processes when the job has finished.

`pam_slurm_adopt` works by binding the login process of a user and all its child processes to the `cgroup` of a running job.

It can be enabled with following steps:

1. In the configuration file `slurm.conf`, set the option `PrologFlags=contain`.
Make sure the option `ProctrackType=proctrack/cgroup` is also set.
2. Restart the services `slurmctld` and `slurmd`.
For this change to take effect, it is not sufficient to issue the command `scontrol reconfigure`.
3. Decide whether to limit resources:
 - If resources are not limited, user processes can continue running on a node even after the job to which they were bound has finished.
 - If resources are limited using a `cgroup`, user processes will be killed when the job finishes, and the controlling `cgroup` is deactivated.
To activate resource limits via a `cgroup`, in the file `/etc/slurm/cgroup.conf`, set the option `ConstrainCores=yes`.

Due to the complexity of accurately determining RAM requirements of jobs, limiting the RAM space is not recommended.

4. Install the package `slurm-pam_slurm`:

```
zypper install slurm-pam_slurm
```

5. (Optional) You can disallow logins by users who have no running job in the machine:

- **Disabling SSH Logins Only:** In the file `/etc/pam.d/ssh`, add the option:

```
account    required pam_slurm_adopt.so
```

- **Disabling All Types of Logins:** In the file `/etc/pam.d/common-account`, add the option:

```
account    required pam_slurm_adopt.so
```

7.13 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` which enables control of 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 x86-64.

7.14 MUNGE Authentication

MUNGE allows for secure communication between different machines which 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*.

MUNGE uses the UID/GID to uniquely identify and authenticate users. This care must be taken that the users who are to be authenticated across a network have unified UIDs and GIDs.

MUNGE credentials have a limited time-to-live. Therefore it must be ensured that the time is synchronized across the entire cluster.

Install MUNGE using `zypper in munge`. This will install all packages required at runtime. The package `munge-devel` can be used to build applications that require MUNGE authentication.

When installing MUNGE, a new key is generated on every system. However, the entire cluster needs to use the same MUNGE key. Therefore the key from one system needs to be copied to all other nodes in the cluster in a secure way. Make sure that the key is only readable by the `munge` user (permissions mask `0400`).

pdsh (with SSH) can be used to do this:

Check permissions, owner, and file type of the file located under `/etc/munge/munge.key` key on the local system:

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

The settings should be:

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

Calculate the MD5 sum of `munge.key`:

```
# md5sum /etc/munge/munge.key
```

Copy the key to the listed nodes:

```
# pdcp -R ssh -w HOSTLIST /etc/munge/munge.key \  
/etc/munge/munge.key
```

Check the key settings in the remote host:

```
# 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
```

Make sure they match.

munged needs to be run on all nodes where MUNGE authentication is to take place. If MUNGE is used for authentication across the network, it needs to run on each side of the communication.

To start the service and make sure it is started after every reboot, on each node, run:

```
systemctl enable munge.service  
systemctl start munge.service
```

To perform this on multiple nodes, you can also use *pdsh*.

7.15 *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 needs to 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 7.14, “Munge Authentication”](#). After Munge has been successfully started, enable and start *mrlogin* on each machine on which the user will log in:

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

To start *mrsh* support at boot, run:

```
systemctl enable munge.service
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:

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

8 MPI Libraries

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

- Open MPI (version 2 and 3)
- MVAPICH2
- MPICH

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

The following packages are available:

- for Open MPI:
 - user programs: [openmpi2-gnu-hpc](#) and [openmpi3-gnu-hpc](#)
 - shared libraries: [libopenmpi2-gnu-hpc](#) and [libopenmpi3-gnu-hpc](#)
 - development libraries, headers and tools required for building: [openmpi2-gnu-hpc-devel](#) and [openmpi3-gnu-hpc-devel](#)
 - documentation: [openmpi2-gnu-hpc-docs](#) and [openmpi3-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 can be installed in parallel.

Use environment modules to pick the version to use:

- for Open MPI version 2:

```
module load TOOLCHAIN openmpi/2
```

- for Open MPI version 3:

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

- for MVAPICH2:

```
module load TOOLCHAIN mvapich2
```

- for MPICH:

```
module load TOOLCHAIN mpich
```

For information about the toolchain to load, see [Section 7.5, “GNU Compiler Collection for HPC”](#).

9 General HPC Libraries

Library packages which 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 facilitate the parallel installation of multiple versions of a library, the package name contains the version number (with dots `.` replaced by underscores `_`). To simplify the installation of a library, `master` packages are supplied which will ensure that the latest version of a package is installed. When these `master` packages are updated, the latest version of the respective packages will be 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 libraries (`.a`) are supplied as well, package names for these end with `-devel-static`.

As an example, package names of the ScaLAPACK library version 2.0.2 built with GCC for Open MPI v1:

- library package: `libscalapack2_2_0_2-gnu-openmpi2-hpc`
- library master package: `libscalapack2-gnu-openmpi2-hpc`
- development package: `libscalapack2_2_0_2-gnu-openmpi2-hpc-devel`
- development master package: `libscalapack2-gnu-openmpi2-hpc-devel`
- static library package: `libscalapack2_2_0_2-gnu-openmpi2-hpc-devel-static`

The digit `2` appended to the library name denotes the `.so` version of the library.

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

For GNU compiler collection, see [Section 7.5, “GNU Compiler Collection for HPC”](#). Supported MPI flavors are described in [Section 8, “MPI Libraries”](#).

The Development Tools module may provide later versions of the GNU compiler suite. To view available compilers, run:

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

9.1 boost — Boost C++ Libraries

Boost is a set of portable C++ libraries which provide a reference implementation of “existing practices”. See the full release notes for Boost 1.71 at https://www.boost.org/users/history/version_1_71_0.html.

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

```
module load TOOLCHAIN boost
```

To use the MPI-specific Boost libraries, add an argument for the MPI flavor to the command:

```
module load TOOLCHAIN MPI_FLAVOR boost
```

For information about the toolchain to load, see [Section 7.5, “GNU Compiler Collection for HPC”](#). For information about available MPI flavors, see [Section 8, “MPI Libraries”](#).

List of master packages:

- [boost-gnu-hpc](#)
- [boost-gnu-hpc-devel](#)

Most Boost libraries do not depend on MPI flavors. However, Boost contains a set of libraries to abstract interactions with MPI. These libraries depend on the MPI flavor used.

List of MPI master packages:

- [boost-gnu-MPI_FLAVOR-hpc](#)
- [boost-gnu-MPI_FLAVOR-hpc-devel](#)
- [boost-gnu-MPI_FLAVOR-hpc-python3](#)

Replace [MPI_FLAVOR](#) with one of the MPI flavors described in [Section 8, “MPI Libraries”](#).

9.2 FFTW HPC Library — Discrete Fourier Transforms

FFTW is a C subroutine library for computing the Discrete Fourier Transform (DFT) in one or more dimensions, of both real and complex data, and of arbitrary input size.

This library is available as both a serial and an MPI-enabled variant. This module requires a compiler toolchain module loaded. To select an MPI variant, the respective MPI module needs to be loaded beforehand. To load this module, run:

```
module load TOOLCHAIN fftw3
```

or

```
module load TOOLCHAIN MPI_FLAVOR fftw3
```

For information about the toolchain to load, see [Section 7.5, “GNU Compiler Collection for HPC”](#). For information about available MPI flavors, see [Section 8, “MPI Libraries”](#).

List of master packages:

- [libfftw3-gnu-hpc](#)
- [fftw3-gnu-hpc-devel](#)
- [libfftw3-gnu-MPI_FLAVOR-hpc](#)
- [fftw3-gnu-MPI_FLAVOR-hpc-devel](#)

Replace MPI_FLAVOR with one of the MPI flavors described in [Section 8, “MPI Libraries”](#).

9.3 GSL — GNU Scientific Library

The GNU Scientific Library (GSL) is a numerical library for C and C++ programmers.

The library provides a wide range of mathematical routines such as random number generators, special functions and least-squares fitting. There are over 1000 functions in total with an extensive test suite.

It is free software under the GNU General Public License.

For this library, a compiler toolchain needs to be loaded beforehand. To load gsl, run:

```
module load TOOLCHAIN gsl
```

For information about the toolchain to load, see [Section 7.5, “GNU Compiler Collection for HPC”](#).

List of master packages:

- [gsl-gnu-hpc](#)
- [gsl-gnu-hpc-devel](#)
- [gsl-gnu-hpc-doc](#)
- [libgsl-gnu-hpc](#)
- [libgslcblas-gnu-hpc](#)

9.4 HYPRE — Scalable Linear Solvers and Multigrid Methods

HYPRE is a library of linear solvers which aim to solve large and detailed simulations faster than traditional methods at large scales.

The library offers a comprehensive suite of scalable solvers for large-scale scientific simulation, featuring parallel multigrid methods for both structured and unstructured grid problems. HYPRE is highly portable, and supports a number of languages. It is developed at the Lawrence Livermore National Laboratory.

For this library, a compiler toolchain and an MPI flavor needs to be loaded beforehand. To load this module, run:

```
module load TOOLCHAIN MPI_FLAVOR hypre
```

For information about the toolchain to load, see [*Section 7.5, “GNU Compiler Collection for HPC”*](#). For information about available MPI flavors, see [*Section 8, “MPI Libraries”*](#).

List of master packages:

- [hypre-gnu-MPI_FLAVOR-hpc-devel](#)
- [libHYPRE-gnu-MPI_FLAVOR-hpc](#)

9.5 METIS — Serial Graph Partitioning and Fill-reducing Matrix Ordering Library

METIS is a set of serial programs for partitioning graphs, partitioning finite element meshes, and producing fill reducing orderings for sparse matrices. The algorithms implemented in METIS are based on the multilevel recursive-bisection, multilevel k-way, and multi-constraint partitioning schemes.

Experiments on a wide range of graphs have shown that METIS is one to two orders of magnitude faster than other widely used partitioning algorithms. Graphs with several millions vertices can be partitioned in 256 parts in a few seconds on current generation systems.

The fill-reducing orderings produced by METIS are significantly better than those produced by other widely used algorithms including multiple minimum degree. For many classes of problems arising in scientific computations and linear programming, METIS is able to reduce the storage and computational requirements of sparse matrix factorization, by up to an order of magnitude. Moreover, unlike multiple minimum degree, the elimination trees produced by METIS are suitable for parallel direct factorization. Furthermore, METIS is able to compute these orderings very fast. Matrices with millions of rows can be reordered in just a few seconds on current generation workstations and PCs.

For this library, a compiler toolchain needs to be loaded beforehand. To load METIS, run:

```
module load TOOLCHAIN metis
```

For information about the toolchain to load, see [Section 7.5, “GNU Compiler Collection for HPC”](#).

List of master packages:

- [metis-gnu-hpc](#)
- [metis-gnu-hpc-devel](#)
- [metis-gnu-hpc-doc](#)
- [metis-gnu-hpc-examples](#)
- [libmetis-gnu-hpc](#)

9.6 MUMPS — Multifrontal Massively Parallel Sparse Direct Solver

MUMPS (MULTifrontal Massively Parallel sparse direct Solver) solves a sparse system of linear equations $Ax = b$ using Gaussian elimination.

This library requires a compiler toolchain and an MPI flavor to be loaded beforehand. To load this module, run:

```
module load TOOLCHAIN MPI_FLAVOR mumps
```

For information about the toolchain to load, see [Section 7.5, “GNU Compiler Collection for HPC”](#). For information about available MPI flavors, see [Section 8, “MPI Libraries”](#).

List of master packages:

- [libmumps-gnu-MPI_FLAVOR-hpc](#)
- [mumps-gnu-MPI_FLAVOR-hpc-devel](#)
- [mumps-gnu-MPI_FLAVOR-hpc-doc](#)
- [mumps-gnu-MPI_FLAVOR-hpc-examples](#)

9.7 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 numarray 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 data-base 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:

```
module load TOOLCHAIN pythonPYTHON_VERSION-numpy
```

For information about the toolchain to load, see [*Section 7.5, “GNU Compiler Collection for HPC”*](#).

List of master packages:

- [pythonPYTHON_VERSION-numpy-gnu-hpc](#)
- [pythonPYTHON_VERSION-numpy-gnu-hpc-devel](#)

9.8 OCR — Open Community Runtime (OCR) for Shared Memory

The Open Community Runtime project is an application-building framework that explores methods for high-core-count programming with focus on HPC applications.

This first reference implementation is a functionally complete implementation of the OCR 1.0.0 specification, with extensive tools and demonstration examples, running on both single-node and clusters.

This library is available both without and with MPI support. For this library a compiler toolchain and if applicable, an MPI flavor needs to be loaded beforehand. To load ocr, run:

```
module load TOOLCHAIN ocr
```

or

```
module load TOOLCHAIN MPI_FLAVOR ocr
```

For information about the toolchain to load, see [Section 7.5, “GNU Compiler Collection for HPC”](#). For information about available MPI flavors, see [Section 8, “MPI Libraries”](#).

List of master packages:

- [ocr-gnu-hpc](#)
- [ocr-gnu-hpc-devel](#)
- [ocr-gnu-hpc-doc](#)
- [ocr-gnu-hpc-examples](#)
- [ocr-gnu-MPI_FLAVOR-hpc](#)
- [ocr-gnu-MPI_FLAVOR-hpc-devel](#)
- [ocr-gnu-MPI_FLAVOR-hpc-doc](#)
- [ocr-gnu-MPI_FLAVOR-hpc-examples](#)

9.9 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 and thus 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 all 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:

- standard version:

```
module load TOOLCHAIN openblas
```

- pthreads version:

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

For information about the toolchain to load, see [Section 7.5, “GNU Compiler Collection for HPC”](#).

List of master package:

- [`libopenblas-gnu-hpc`](#)
- [`libopenblas-gnu-hpc-devel`](#)

- [libopenblas-pthreads-gnu-hpc](#)
- [libopenblas-pthreads-gnu-hpc-devel](#)

9.10 PETSc HPC Library — Solver for Partial Differential Equations

PETSc is a suite of data structures and routines for the scalable (parallel) solution of scientific applications modeled by partial differential equations.

This module requires loading a compiler toolchain as well as an MPI library flavor beforehand. To load this module, run:

```
module load TOOLCHAIN MPI_FLAVOR petsc
```

For information about the toolchain to load, see [Section 7.5, “GNU Compiler Collection for HPC”](#). For information about available MPI flavors, see [Section 8, “MPI Libraries”](#).

List of master packages:

- [libpetsc-gnu-MPI_FLAVOR-hpc](#)
- [petsc-gnu-MPI_FLAVOR-hpc-devel](#)

[MPI_FLAVOR](#) must be one of the supported MPI flavors described in [Section 8, “MPI Libraries”](#).

9.11 ScaLAPACK HPC Library — LAPACK Routines

The library ScaLAPACK (short for *Scalable LAPACK*) includes a subset of LAPACK routines designed for distributed memory MIMD-parallel computers.

This library requires loading both a compiler toolchain and an MPI library flavor beforehand. To load this module, run:

```
module load TOOLCHAIN MPI_FLAVOR scalapack
```

For information about the toolchain to load, see [Section 7.5, “GNU Compiler Collection for HPC”](#). For information about available MPI flavors, see [Section 8, “MPI Libraries”](#).

List of master packages:

- [libscalapack2-gnu-MPI_FLAVOR-hpc](#)
- [libscalapack2-gnu-MPI_FLAVOR-hpc-devel](#)

`MPI_FLAVOR` must be one of the supported MPI flavors described in [Section 8, “MPI Libraries”](#).

9.12 SCOTCH — Static Mapping and Sparse Matrix Reordering Algorithms

SCOTCH is a set of programs and libraries which implement the static mapping and sparse matrix reordering algorithms developed within the SCOTCH project.

Its purpose is to apply graph theory, with a divide and conquer approach, to scientific computing problems such as graph and mesh partitioning, static mapping, and sparse matrix ordering, in application domains ranging from structural mechanics to operating systems or bio-chemistry.

For this library a compiler toolchain and an MPI flavor needs to be loaded beforehand. To load this module, run:

```
module load TOOLCHAIN MPI_FLAVOR scotch
```

For information about the toolchain to load, see [Section 7.5, “GNU Compiler Collection for HPC”](#). For information about available MPI flavors, see [Section 8, “MPI Libraries”](#).

List of master packages:

- [`libptscotch-gnu-MPI_FLAVOR-hpc`](#)
- [`ptscotch-gnu-MPI_FLAVOR-hpc`](#)
- [`ptscotch-gnu-MPI_FLAVOR-hpc-devel`](#)

9.13 SciPy Python Library

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

The additional benefit of basing SciPy on Python is that this also makes a powerful programming language available for use in developing sophisticated programs and specialized applications. Scientific applications using SciPy benefit from the development of additional modules in numerous niches of the software landscape by developers across the world. Everything from

parallel programming to web and data-base subroutines and classes have been made available to the Python programmer. All of this power is available in addition to the mathematical libraries in SciPy.

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 needs to be specified when loading this module. To load this module, run:

```
module load TOOLCHAIN pythonPYTHON_VERSION-scipy
```

For information about the toolchain to load, see [Section 7.5, “GNU Compiler Collection for HPC”](#).

List of master packages:

- [pythonPYTHON_VERSION-scipy-gnu-hpc](#)
- [pythonPYTHON_VERSION-scipy-gnu-hpc-devel](#)

9.14 SuperLU — Supernodal LU Decomposition of Sparse Matrices

SuperLU is a general purpose library for the direct solution of large, sparse, nonsymmetric systems of linear equations. The library is written in C and can be called from C and Fortran programs.

It uses MPI and OpenMP to support various forms of parallelism. It supports both real and complex data types, both single and double precision, and 64-bit integer indexing. The library routines performs an LU decomposition with partial pivoting and triangular system solves through forward and back substitution. The LU factorization routines can handle non-square matrices but the triangular solves are performed only for square matrices. The matrix columns can be preordered (before factorization) either through library or user-supplied routines. This reordering for sparsity is completely separate from the factorization. Working precision iterative refinement subroutines are provided for improved backward stability. Routines are also provided to equilibrate the system, estimate the condition number, calculate the relative backward error, and estimate error bounds for the refined solutions.

This library requires a compiler toolchain and an MPI flavor to be loaded beforehand. To load this module, run:

```
module load TOOLCHAIN superlu
```

For information about the toolchain to load, see [Section 7.5, “GNU Compiler Collection for HPC”](#).

List of master packages:

- [libsuperlu-gnu-hpc](#)
- [superlu-gnu-hpc-devel](#)
- [superlu-gnu-hpc-doc](#)
- [superlu-gnu-hpc-examples](#)

9.15 Trilinos — Object-oriented Software Framework

The Trilinos Project is an effort to develop algorithms and enabling technologies within an object-oriented software framework for the solution of large-scale, complex multi-physics engineering and scientific problems. A unique design feature of Trilinos is its focus on packages. This library needs a compiler toolchain and MPI flavor to be loaded beforehand. To load this module, run:

```
module load TOOLCHAIN MPI_FLAVOR trilinos
```


For information about the toolchain to load, see [Section 7.5, “GNU Compiler Collection for HPC”](#). For information about available MPI flavors, see [Section 8, “MPI Libraries”](#).

List of master packages:

- [libtrilinos-gnu-MPI_FLAVOR-hpc](#)
- [trilinos-gnu-MPI_FLAVOR-hpc-devel](#)

10 HPC File Format Libraries

10.1 Adaptable IO System (ADIOS)

The Adaptable IO System (ADIOS) provides a simple, flexible way for scientists to describe the data in their code that may need to be written, read, or processed outside of the running simulation. For more information, see <https://www.olcf.ornl.gov/center-projects/adios/> .

```
module load TOOLCHAIN MPI_FLAVOR adios
```

For information about the toolchain to load, see [Section 7.5, “GNU Compiler Collection for HPC”](#). For information about available MPI flavors, see [Section 8, “MPI Libraries”](#).

List of master packages:

- [adios-gnu-MPI_FLAVOR-hpc](#)
- [adios-gnu-MPI_FLAVOR-hpc-devel](#)

Replace [MPI_FLAVOR](#) with one of the MPI flavors described in [Section 8, “MPI Libraries”](#).

10.2 HDF5 HPC Library — Model, Library, 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:

```
module load TOOLCHAIN hdf5
```

When an MPI flavor is loaded, the MPI version of this module can be loaded by:

```
module load TOOLCHAIN MPI_FLAVOR phpdf5
```

For information about the toolchain to load, see [Section 7.5, “GNU Compiler Collection for HPC”](#). For information about available MPI flavors, see [Section 8, “MPI Libraries”](#).

List of master packages:

- [hdf5-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, “MPI Libraries”](#).

10.3 NetCDF HPC Library — Implementation of Self-Describing Data Formats

The NetCDF software libraries for C, C++, FORTRAN, and Perl are a set of software libraries and self-describing, machine-independent data formats that support the creation, access, and sharing of array-oriented scientific data.

[netcdf Packages](#)

The packages with names starting with [`netcdf`](#) provide C bindings for the NetCDF API. These are available with and without MPI support.

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.

The MPI variant becomes available when the MPI module is loaded. Both variants require loading a compiler toolchain module beforehand. To load the highest version of the non-MPI [`netcdf`](#) module, run:

```
module load TOOLCHAIN MPI_FLAVOR netcdf
```

For information about the toolchain to load, see [Section 7.5, “GNU Compiler Collection for HPC”](#). For information about available MPI flavors, see [Section 8, “MPI Libraries”](#).

List of master packages:

- [netcdf-gnu-hpc](#)
- [netcdf-gnu-hpc-devel](#)
- [netcdf-gnu-hpc](#)
- [netcdf-gnu-hpc-devel](#)
- [netcdf-gnu-MPI_FLAVOR-hpc](#)
- [netcdf-gnu-MPI_FLAVOR-hpc-devel](#)
- [netcdf-gnu-MPI_FLAVOR-hpc](#)

[MPI_FLAVOR](#) must be one of the supported MPI flavors described in [Section 8, “MPI Libraries”](#).

[netcdf-cxx Packages](#)

[netcdf-cxx4](#) provides a C++ binding for the NetCDF API.

This module requires loading a compiler toolchain module beforehand. To load this module, run:

```
module load TOOLCHAIN netcdf-cxx4
```

For information about the toolchain to load, see [Section 7.5, “GNU Compiler Collection for HPC”](#). For information about available MPI flavors, see [Section 8, “MPI Libraries”](#).

List of master packages:

- [libnetcdf-cxx4-gnu-hpc](#)
- [libnetcdf-cxx4-gnu-hpc-devel](#)
- [netcdf-cxx4-gnu-hpc-tools](#)

[netcdf-fortran Packages](#)

The [netcdf-fortran](#) packages provide FORTRAN bindings for the NetCDF API, with and without MPI support.

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.

The MPI variant becomes available when the MPI module is loaded. Both variants require loading a compiler toolchain module beforehand. To load the highest version of the non-MPI netcdf module, run:

```
module load TOOLCHAIN MPI_FLAVOR netcdf
```

For information about the toolchain to load, see [Section 7.5, “GNU Compiler Collection for HPC”](#). For information about available MPI flavors, see [Section 8, “MPI Libraries”](#).

List of master packages:

- [netcdf-gnu-hpc](#)
- [netcdf-gnu-hpc-devel](#)
- [netcdf-gnu-hpc](#)
- [netcdf-gnu-hpc-devel](#)
- [netcdf-gnu-MPI_FLAVOR-hpc](#)
- [netcdf-gnu-MPI_FLAVOR-hpc-devel](#)
- [netcdf-gnu-MPI_FLAVOR-hpc](#)

MPI_FLAVOR is one of the supported MPI flavors [Section 8, “MPI Libraries”](#).

[netcdf-cxx](#) Packages

netcdf-cxx4 provides a C++ binding for the NetCDF API.

This module requires loading a compiler toolchain module beforehand. To load this module, run:

```
module load TOOLCHAIN netcdf-cxx4
```

For information about the toolchain to load, see [Section 7.5, “GNU Compiler Collection for HPC”](#).

List of master packages:

- [libnetcdf-cxx4-gnu-hpc](#)
- [libnetcdf-cxx4-gnu-hpc-devel](#)
- [netcdf-cxx4-gnu-hpc-tools](#)

[netcdf-fortran](#) Packages

The [netcdf-fortran](#) packages provide FORTRAN bindings for the NetCDF API, with and without MPI support.

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.

The MPI variant becomes available when the MPI module is loaded. Both variants require loading a compiler toolchain module beforehand. To load the highest version of the non-MPI [netcdf](#) module, run:

```
module load TOOLCHAIN netcdf-fortran
```

or

```
module load TOOLCHAIN MPI_FLAVOR netcdf-fortran
```

For information about the toolchain to load, see [Section 7.5, “GNU Compiler Collection for HPC”](#). For information about available MPI flavors, see [Section 8, “MPI Libraries”](#).

List of master packages:

- [libnetcdf-fortran-gnu-hpc](#)
- [libnetcdf-fortran-gnu-hpc-devel](#)
- [libnetcdf-fortran-gnu-hpc](#)
- [libnetcdf-fortran-gnu-MPI_FLAVOR-hpc](#)
- [libnetcdf-fortran-gnu-MPI_FLAVOR-hpc-devel](#)
- [libnetcdf-fortran-gnu-MPI_FLAVOR-hpc](#)

10.4 HPC Flavor of [pnetcdf](#)

NetCDF is a set of software libraries and self-describing, machine-independent data formats that support the creation, access, and sharing of array-oriented scientific data.

Parallel NetCDF ([PnetCDF](#)) is a library providing high-performance I/O while still maintaining file-format compatibility with NetCDF by Unidata.

The package is available for the MPI Flavors: Open MPI 2 and 3, MVAPICH2 and MPICH.

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

```
module load TOOLCHAIN MPI_FLAVOR pnetcdf
```

For information about the toolchain to load, see [Section 7.5, “GNU Compiler Collection for HPC”](#). For information about available MPI flavors, see [Section 8, “MPI Libraries”](#).

List of master packages:

- [`libpnetcdf-gnu-MPI_FLAVOR-hpc`](#)
- [`pnetcdf-gnu-MPI_FLAVOR-hpc`](#)
- [`pnetcdf-gnu-MPI_FLAVOR-hpc-devel`](#)

`MPI_FLAVOR` must be one of the supported MPI flavors described in [Section 8, “MPI Libraries”](#).

11 Profiling and Benchmarking Libraries and Tools

Performance optimization plays an important role in HPC. That applications are run across multiple cluster nodes poses an additional challenge. SLE-HPC provides a number of tools for profiling MPI applications and benchmarking MPI performance.

11.1 IMB — Intel* MPI Benchmarks

Intel* MPI Benchmarks provides a set of elementary benchmarks that conform to MPI-1, MPI-2, and MPI-3 standard. One can run all of the supported benchmarks, or a subset specified in the command line using one 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 located at: <https://software.intel.com/en-us/imb-user-guide>.

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

```
module load TOOLCHAIN MPI_FLAVOR imb
```

For information about the toolchain to load, see [Section 7.5, “GNU Compiler Collection for HPC”](#). For information about available MPI flavors, see [Section 8, “MPI Libraries”](#).

- [imb-gnu-MPI_FLAVOR-hpc](#)

11.2 PAPI HPC Library — Consistent Interface for Hardware Performance Counters

PAPI (package [papi](#)) provides a tool with a consistent interface and methodology for use of 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. The latest version provided can be loaded by running:

```
module load TOOLCHAIN papi
```

For information about the toolchain to load, see [*Section 7.5, “GNU Compiler Collection for HPC”*](#).

List of master packages:

- [papi-hpc](#)
- [papi-hpc-devel](#)

For general information about Lmod and modules, see [*Section 7.7, “Lmod — Lua-based Environment Modules”*](#).

11.3 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 needs to be loaded beforehand. To load this, run:

```
module load TOOLCHAIN MPI_FLAVOR mpiP
```

For information about the toolchain to load, see [*Section 7.5, “GNU Compiler Collection for HPC”*](#). For information about available MPI flavors, see [*Section 8, “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, “MPI Libraries”](#).

12 Obtaining Source Code


This SUSE product includes materials licensed to SUSE under the GNU General Public License (GPL). The GPL requires SUSE to provide the source code that corresponds to the GPL-licensed material. The source code is available for download at <https://www.suse.com/download/sle-hpc/> on Medium 2. For up to three years after distribution of the SUSE product, upon request, SUSE will mail a copy of the source code. Send requests by e-mail to mailto:sle_source_request@suse.com. SUSE may charge a reasonable fee to recover distribution costs.

13 Legal Notices


SUSE makes no representations or warranties with regard to the contents or use of this documentation, and specifically disclaims any express or implied warranties of merchantability or fitness for any particular purpose. Further, SUSE reserves the right to revise this publication and to make changes to its content, at any time, without the obligation to notify any person or entity of such revisions or changes.


Further, SUSE makes no representations or warranties with regard to any software, and specifically disclaims any express or implied warranties of merchantability or fitness for any particular purpose. Further, SUSE reserves the right to make changes to any and all parts of SUSE software, at any time, without any obligation to notify any person or entity of such changes.


Any products or technical information provided under this Agreement may be subject to U.S. export controls and the trade laws of other countries. You agree to comply with all export control regulations and to obtain any required licenses or classifications to export, re-export, or import deliverables. You agree not to export or re-export to entities on the current U.S. export exclusion lists or to any embargoed or terrorist countries as specified in U.S. export laws. You agree to not

use deliverables for prohibited nuclear, missile, or chemical/biological weaponry end uses. Refer to <https://www.suse.com/company/legal/>  for more information on exporting SUSE software. SUSE assumes no responsibility for your failure to obtain any necessary export approvals.

Copyright © 2010-2021 SUSE LLC.

This release notes document is licensed under a Creative Commons Attribution-NoDerivatives 4.0 International License (CC-BY-ND-4.0). You should have received a copy of the license along with this document. If not, see <https://creativecommons.org/licenses/by-nd/4.0/> .

SUSE has intellectual property rights relating to technology embodied in the product that is described in this document. In particular, and without limitation, these intellectual property rights may include one or more of the U.S. patents listed at <https://www.suse.com/company/legal/>  and one or more additional patents or pending patent applications in the U.S. and other countries.

For SUSE trademarks, see SUSE Trademark and Service Mark list (<https://www.suse.com/company/legal/> ). All third-party trademarks are the property of their respective owners.