SUSE

SUSE Linux Enterprise for High-Performance Computing 15 SP4

Release Notes

SUSE Linux Enterprise for 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 for High-Performance Computing 15 SP4 and important product updates.

These release notes are updated periodically. The latest version of these release notes is always available at https://www.suse.com/releasenotes ↗. General documentation can be found at https://documentation.suse.com/sle-hpc/15-SP4 ↗.

Contents

1 About the release notes

These Release Notes are identical across all architectures, and the most recent version is always available online at https://www.suse.com/releasenotes ↗ .

Entries are only listed once but they can be referenced in several places if they are important and belong to more than one section.

Release notes usually only list changes that happened between two subsequent releases. Certain important entries from the release notes of previous product versions are repeated. To make these entries easier to identify, they contain a note to that effect.

However, repeated entries are provided as a courtesy only. Therefore, if you are skipping one or more service packs, check the release notes of the skipped service packs as well. If you are only reading the release notes of the current release, you could miss important changes.

2 SUSE Linux Enterprise for High-Performance Computing

SUSE Linux Enterprise for 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 for High-Performance Computing 15 SP4 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, LA-PACK, …

- Various MPI implementations

- Serial and parallel libraries for the HDF5 file format

2.1 Hardware Platform Support

SUSE Linux Enterprise for High-Performance Computing 15 SP4 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 for High-Performance Computing release, you should review at least the following sections:

- *Section 2.4, "Support statement for SUSE Linux Enterprise for High-Performance Computing"*

2.3 Support and life cycle

SUSE Linux Enterprise for 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 for 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 (SP4) will be fully maintained and supported until 6 months after the release of SUSE Linux Enterprise for High-Performance Computing 15 SP5.

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 for High-Performance Computing

To receive support, you need an appropriate subscription with SUSE. For more information, see https://www.suse.com/support/programs/subscriptions/?id=SUSE_Linux_Enterprise_Server ↗ .

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 for High-Performance Computing 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, see *Section 2.4.1, "Software requiring specific contracts"*

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

2.4.1 Software requiring specific contracts

Certain software delivered as part of SUSE Linux Enterprise for High-Performance Computing may require an external contract. Check the support status of individual packages using the RPM metadata that can be viewed with `rpm`, `zypper`, or YaST.

2.4.2 Software under GNU AGPL

SUSE Linux Enterprise for High-Performance Computing 15 SP4 (and the SUSE Linux Enterprise modules) includes the following software that is shipped *only* under a GNU AGPL software license:

- Ghostscript (including subpackages)

SUSE Linux Enterprise for High-Performance Computing 15 SP4 (and the SUSE Linux Enterprise modules) includes the following software that is shipped under multiple licenses that include a GNU AGPL software license:

- MySpell dictionaries and LightProof

- ArgyllCMS

2.5 Documentation and other information

2.5.1 Available on the product media

- Read the READMEs on the media.

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

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

- Check the `ChangeLog` file in the top level of the installation medium for a chronological log of all changes made to the updated packages.

- Find more information in the `docu` directory of the installation medium of SUSE Linux Enterprise for High-Performance Computing 15 SP4. This directory includes PDF versions of the SUSE Linux Enterprise for High-Performance Computing 15 SP4 Installation Quick Start Guide.

2.5.2 Online documentation

- For the most up-to-date version of the documentation for SUSE Linux Enterprise for High-Performance Computing 15 SP4, see https://documentation.suse.com/sle-hpc/15-SP4 ↗.

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

3 Modules, extensions, and related products

This section comprises information about modules and extensions for SUSE Linux Enterprise for High-Performance Computing 15 SP4 Modules and extensions add functionality to the system.

3.1 Modules in the SLE 15 SP4 product line

The SLE 15 SP4 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 SP4 at the release of SUSE Linux Enterprise for High-Performance Computing 15 SP4 are listed in the *Modules and Extensions Quick Start* at https://documentation.suse.com/sles/15-SP3/html/SLES-all/article-modules.html ↗.

Not all SLE modules are available with a subscription for SUSE Linux Enterprise for High-Performance Computing 15 SP4 itself (see the column *Available for*).

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

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://package-hub.suse.com/ .

3.3 Related products

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

- SUSE Linux Enterprise Server: https://www.suse.com/products/server

- SUSE Linux Enterprise JeOS: https://www.suse.com/products/server/jeos

- 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.

4.1 64K page size kernel flavor has been added

SUSE Linux Enterprise for High-Performance Computing for Arm 12 SP2 and later kernels have used a page size of 4K. This offers the widest compatibility also for small systems with little RAM, allowing to use Transparent Huge Pages (THP) where large pages make sense.

As a technology preview, SUSE Linux Enterprise for High-Performance Computing for Arm 15 SP4 adds a kernel flavor `64kb`, offering a page size of 64 KiB and physical/virtual address size of 52 bits. Same as the `default` kernel flavor, it does not use preemption.

Main purpose at this time is to allow for side-by-side benchmarking for High Performance Computing, Machine Learning and other Big Data use cases. Contact your SUSE representative if you notice performance gains for your specific workloads.

> **❗ Important: Swap needs to be re-initialized**
>
> After booting the 64K kernel, any swap partitions need to re-initialized to be usable. To do this, run the `swapon` command with the `--fixpgsz` parameter on the swap partition. Note that this process deletes data present in the swap partition (for example, suspend data). In this example, the swap partition is on `/dev/sdc1`:
>
> ```
> swapon --fixpgsz /dev/sdc1
> ```

> **❗ Important: Btrfs file system uses page size as block size**
>
> It is currently not possible to use Btrfs file systems across page sizes. Block sizes below page size are not yet supported and block sizes above page size might never be supported.
>
> During installation, change the default partitioning proposal and choose another file system, such as Ext4 or XFS, to allow rebooting from the default 4K page size kernel of the Installer into `kernel-64kb` and back.
>
> See the *Storage Guide* for a discussion of supported file systems.

> **✋ Warning: RAID 5 uses page size as stripe size**
>
> It is currently not yet possible to configure stripe size on volume creation. This will lead to sub-optimal performance if page size and block size differ.
>
> Avoid RAID 5 volumes when benchmarking 64K vs. 4K page size kernels.

See the *Storage Guide* for more information on software RAID.

> 🏷️ **Note: Cross-architecture compatibility considerations**
>
> The SUSE Linux Enterprise for High-Performance Computing 15 SP4 kernels on x86-64 use 4K page size.
>
> The SUSE Linux Enterprise for High-Performance Computing for POWER 15 SP4 kernel uses 64K page size.

5   Modules

5.1   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`, as well as numerous HPC libraries and frameworks.

This module is available with the SUSE Linux Enterprise for 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. Refer to the system administration guide for further details.

5.2   NVIDIA Compute Module

The NVIDIA Compute Module provides the NVIDIA CUDA repository for SUSE Linux Enterprise 15. Note that that any software within this repository is under a 3rd party EULA. For more information check https://docs.nvidia.com/cuda/eula/index.html ↗.

This module is not selected for addition by default when installing SUSE Linux Enterprise for High-Performance Computing. It may be selected manually during installation from the *Extension and Modules* screen. You may also select it on an installed system using YaST. To do so, run from a shell as root `yast registration`, select: `Select Extensions` and search for `NVIDIA Compute Module` and press `Next`.

> **ⓘ Important**
>
> Do not attempt to add this module with the `SUSEConnect` CLI tool. This tool is not yet capable of handling 3rd party repositories.

Once you have selected this module you will be asked to confirm the 3rd party license and verify the repository signing key.

6 Changes affecting all architectures

Information in this section applies to all architectures supported by SUSE Linux Enterprise for High-Performance Computing 15 SP4.

6.1 Enriched system visibility in the SUSE Customer Center (SCC)

SUSE is committed to helping provide better insights into the consumption of SUSE subscriptions regardless of where they are running or how they are managed; physical or virtual, on-prem or in the cloud, connected to SCC or Repository Mirroring Tool (RMT), or managed by SUSE Manager. To help you identify or filter out systems in SCC that are no longer running or decommissioned, SUSEConnect now features a daily "ping", which will update system information automatically.

For more details see the documentation at https://documentation.suse.com/subscription/suseconnect/single-html/SLE-suseconnect-visibility/ ↗ .

6.2 Automatically opened ports

Installing the following packages automatically opens the following ports:

- `dolly` - TCP ports 9997 and 9998

- `slurm` - TCP ports 6817, 6818, and 6819

> **! Important**
>
> These release notes only document changes in SUSE Linux Enterprise for High-Performance Computing compared to the immediate previous service pack of SUSE Linux Enterprise for High-Performance Computing. The full changes and fixes can be found on the respective web site of the packages.

6.3 `dolly`

`dolly` has been updated to version 0.63.6. It includes some fixes for hostname resolution, a better documentation and now provides a default configuration for firewall.

6.4 `memkind`

`memkind` has been updated to version 1.12.0. The full list of changes is available at http://memkind.github.io/memkind/ ↗.

6.5 `openblas`

`openblas` has been updated to version 0.3.17. It contains performance regression fixes and optimization. For more information see https://github.com/xianyi/OpenBLAS/releases/tag/v0.3.17 ↗.

6.6 `spack`

6.6.1 v0.19.1

6.6.1.1 Spack Bugfixes

- `buildcache create`: make `file exists` less verbose
- `spack mirror create`: don't change paths to urls
- Improve error message for requirements
- uninstall: fix accidental cubic complexity

- scons: fix signature for `install_args`

- Fix `combine_phase_logs` text encoding issues

- Use a module-like object to propagate changes in the MRO, when setting build env

- PackageBase should not define builder legacy attributes

- Forward lookup of the `run_tests` attribute

- Bugfix for timers

- Fix path handling in prefix inspections

- Fix libtool filter for Fujitsu compilers

- FileCache: delete the new cache file on exception

- Propagate exceptions from Spack python console

- Tests: Fix a bug/typo in a `config_values.py` fixture

- Various CI fixes

- Docs: remove monitors and analyzers, typos

- bump release version for tutorial command

6.6.2   v0.19.0

`v0.19.0` is a major feature release.

6.6.2.1   Major features in this release

1. **Package requirements**
   Spack's traditional package preferences (https://spack.readthedocs.io/en/latest/build_settings.html#package-preferences) ↗ are soft, but we've added hard requriements to `packages.yaml` and `spack.yaml`. Package requirements use the same syntax as specs:

```
packages:
  libfabric:
    require: "@1.13.2"
  mpich:
    require:
```

```
    - one_of: ["+cuda", "+rocm"]
```

More details in the docs (https://spack.readthedocs.io/en/latest/build_settings.html#package-requirements) ↗ .

2. **Environment UI Improvements**

   - Fewer surprising modifications to `spack.yaml` :

     - `spack install` in an environment will no longer add to the `specs:` list; you'll need to either use `spack add <spec>` or `spack install --add <spec>`.

     - Similarly, `spack uninstall` will not remove from your environment's `specs:` list; you'll need to use `spack remove` or `spack uninstall --remove`.
       This will make it easier to manage an environment, as there is clear separation between the stack to be installed (`spack.yaml` / `spack.lock`) and which parts of it should be installed (`spack install` / `spack uninstall`).

   - `concretizer:unify:true` is now the default mode for new environments
     We see more users creating `unify:true` environments now. Users who need `unify:false` can add it to their environment to get the old behavior. This will concretize every spec in the environment independently.

   - Include environment configuration from URLs (docs (https://spack.readthedocs.io/en/latest/environments.html#included-configurations) ↗)
     You can now include configuration in your environment directly from a URL:

     ```
     spack:
       include:
       - https://github.com/path/to/raw/config/compilers.yaml
     ```

3. **Compiler and variant propagation**
   Currently, compiler flags and variants are inconsistent: compiler flags set for a package are inherited by its dependencies, while variants are not. We should have these be consistent by allowing for inheritance to be enabled or disabled for both variants and compiler flags. Example syntax: * `package ++variant` : enabled variant that will be propagated to dependencies * `package +variant` : enabled variant that will NOT be propagated to dependencies * `package ~~variant` : disabled variant that will be propagated to dependencies * `package ~variant` : disabled variant that will NOT be propagated to dependencies * `package cflags==-g` : cflags will be propagated to dependencies * `package cflags=-g` : cflags will NOT be propagated to dependencies

+ Syntax for non-boolan variants is similar to compiler flags. More in the docs for variants (https://spack.readthedocs.io/en/latest/basic_usage.html#variants) ↗ and compiler flags (https://spack.readthedocs.io/en/latest/basic_usage.html#compiler-flags) ↗.

4. **Enhancements to git version specifiers**

   - `v0.18.0` added the ability to use git commits as versions. You can now use the `git.` prefix to specify git tags or branches as versions. All of these are valid git versions in `v0.19` :

     ```
     foo@abcdef1234abcdef1234abcdef1234abcdef1234      # raw commit
     foo@git.abcdef1234abcdef1234abcdef1234abcdef1234  # commit with git prefix
     foo@git.develop                                   # the develop branch
     foo@git.0.19                                       # use the 0.19 tag
     ```

   - `v0.19` also gives you more control over how Spack interprets git versions, in case Spack cannot detect the version from the git repository. You can suffix a git version with `=<version>` to force Spack to concretize it as a particular version :

     ```
     # use mybranch, but treat it as version 3.2 for version comparison
     foo@git.mybranch=3.2

     # use the given commit, but treat it as develop for version comparison
     foo@git.abcdef1234abcdef1234abcdef1234abcdef1234=develop
     ```

     More in the docs (https://spack.readthedocs.io/en/latest/basic_usage.html#version-specifier) ↗

5. **Changes to Cray EX Support**

   Cray machines have historically had their own platform within Spack, because we needed to go through the module system to leverage compilers and MPI installations on these machines. The Cray EX programming environment now provides standalone `craycc` executables and proper `mpicc` wrappers, so Spack can treat EX machines like Linux with extra packages .

   We expect this to greatly reduce bugs, as external packages and compilers can now be used by prefix instead of through modules. We will also no longer be subject to reproducibility issues when modules change from Cray PE release to release and from site to site. This also simplifies dealing with the underlying Linux OS on cray systems, as Spack will properly model the machine's OS as either SuSE or RHEL.

6. **Improvements to tests and testing in CI**

- `spack ci generate --tests` will generate a `.gitlab-ci.yml` file that not only does builds but also runs tests for built packages . Public GitHub pipelines now also run tests in CI.

- `spack test run --explicit` will only run tests for packages that are explicitly installed, instead of all packages.

7. **Experimental binding link model**

    You can add a new option to `config.yaml` to make Spack embed absolute paths to needed shared libraries in ELF executables and shared libraries on Linux (docs (https://spack.readthedocs.io/en/latest/config_yaml.html#shared-linking-bind) ↗):

    ```
    config:
      shared_linking:
        type: rpath
        bind: true
    ```

    This can improve launch time at scale for parallel applications, and it can make installations less susceptible to environment variables like `LD_LIBRARY_PATH`, even especially when dealing with external libraries that use `RUNPATH`. You can think of this as a faster, even higher-precedence version of `RPATH`.

6.6.2.2   Other new features of note

- `spack spec` prints dependencies more legibly. Dependencies in the output now appear at the *earliest* level of indentation possible

- You can override `package.py` attributes like `url`, directly in `packages.yaml` (docs (https://spack.readthedocs.io/en/latest/build_settings.html#assigning-package-attributes) ↗)

- There are a number of new architecture-related format strings you can use in Spack configuration files to specify paths (docs (https://spack.readthedocs.io/en/latest/configuration.html#config-file-variables) ↗)

6.6.2.3   Performance Improvements

- Major performance improvements for installation from binary caches

- Test suite can now be parallelized using `xdist` (used in GitHub Actions)

- Reduce lock contention for parallel builds in environments

6.6.2.4   New binary caches and stacks

- We now build nearly all of E4S with `oneapi` in our buildcache

- Added 3 new machine learning-centric stacks to binary cache: `x86_64_v3`, CUDA, ROCm

6.6.2.5   Removals and Deprecations

- Support for Python 3.5 is dropped . Only Python 2.7 and 3.6+ are officially supported.

- This is the last Spack release that will support Python 2 . Spack `v0.19` will emit a deprecation warning if you run it with Python 2, and Python 2 support will soon be removed from the `develop` branch.

- `LD_LIBRARY_PATH` is no longer set by default by `spack load` or module loads. Setting `LD_LIBRARY_PATH` in Spack environments/modules can cause binaries from outside of Spack to crash, and Spack's own builds use `RPATH` and do not need `LD_LIBRARY_PATH` set in order to run. If you still want the old behavior, you can run these commands to configure Spack to set `LD_LIBRARY_PATH`:

  ```
  spack config add modules:prefix_inspections:lib64:[LD_LIBRARY_PATH]
  spack config add modules:prefix_inspections:lib:[LD_LIBRARY_PATH]
  ```

- The `spack:concretization:[together|separately]` has been removed after being deprecated in `v0.18`. Use `concretizer:unify:[true|false]`.

- `config:module_roots` is no longer supported after being deprecated in `v0.18`. Use configuration in module sets instead (docs (https://spack.readthedocs.io/en/latest/module_file_support.html) ↗).

- `spack activate` and `spack deactivate` are no longer supported, having been deprecated in `v0.18`. Use an environment with a view instead of activating/deactivating (docs (https://spack.readthedocs.io/en/latest/environments.html#configuration-in-spack-yaml) ↗).

- The old YAML format for buildcaches is now deprecated . If you are using an old buildcache with YAML metadata you will need to regenerate it with JSON metadata.

- `spack bootstrap trust` and `spack bootstrap untrust` are deprecated in favor of `spack bootstrap enable` and `spack bootstrap disable` and will be removed in `v0.20`.

- The `graviton2` architecture has been renamed to `neoverse_n1`, and `graviton3` is now `neoverse_v1`. Buildcaches using the old architecture names will need to be rebuilt.

- The terms `blacklist` and `whitelist` have been replaced with `include` and `exclude` in all configuration files . You can use `spack config update` to automatically fix your configuration files.

6.6.2.6 Notable Bugfixes

- Permission setting on installation now handles effective uid properly

- `buildable:true` for an MPI implementation now overrides `buildable:false` for `mpi`

- Improved error messages when attempting to use an unconfigured compiler

- Do not punish explicitly requested compiler mismatches in the solver

- `spack stage`: add missing –fresh and –reuse

- Fixes for adding build system executables like `cmake` to package scope

- Bugfix for binary relocation with aliased strings produced by newer `binutils`

6.6.3   v0.18.1

6.6.3.1 Spack Bugfixes

- Fix several bugs related to bootstrapping

- Fix a regression that was causing spec hashes to differ between Python 2 and Python 3

- Fixed compiler flags for oneAPI and DPC++

- Fixed several issues related to concretization

- Improved support for Cray manifest file and `spack external find`

- Assign a version to openSUSE Tumbleweed according to the GLIBC version in the system

- Improved Dockerfile generation for `spack containerize`

- Fixed a few bugs related to concurrent execution of commands

6.6.3.2   Package updates

- WarpX: add v22.06, fixed libs property

- openPMD: add v0.14.5, update recipe for @develop

6.6.4   v0.18.0

`v0.18.0` is a major feature release.

6.6.4.1   Major features in this release

1. **Concretizer now reuses by default**

   `spack install --reuse` was introduced in `v0.17.0`, and `--reuse` is now the default concretization mode. Spack will try hard to resolve dependencies using installed packages or binaries .

   To avoid reuse and to use the latest package configurations, (the old default), you can use `spack install --fresh`, or add configuration like this to your environment or `concretizer.yaml`:

   ```
   concretizer:
       reuse: false
   ```

2. **Finer-grained hashes**

   Spack hashes now include `link`, `run`, *and* `build` dependencies, as well as a canonical hash of package recipes. Previously, hashes only included `link` and `run` dependencies (though `build` dependencies were stored by environments). We coarsened the hash to reduce churn in user installations, but the new default concretizer behavior mitigates this concern and gets us reuse *and* provenance. You will be able to see the build dependencies of new installations with `spack find`. Old installations will not change and their hashes will not be affected.

3. **Improved error messages**

   Error handling with the new concretizer is now done with optimization criteria rather than with unsatisfiable cores, and Spack reports many more details about conflicting constraints.

4. **Unify environments when possible**

   Environments have thus far supported `concretization: together` or `concretization: separately`. These have been replaced by a new preference in `concretizer.yaml`:

   ```
   concretizer:
       unify: [true|false|when_possible]
   ```

   `concretizer:unify:when_possible` will *try* to resolve a fully unified environment, but if it cannot, it will create multiple configurations of some packages where it has to. For large environments that previously had to be concretized separately, this can result in a huge speedup (40-50x).

5. **Automatically find externals on Cray machines**

   Spack can now automatically discover installed packages in the Cray Programming Environment by running `spack external find` (or `spack external read-cray-manifest` to *only* query the PE). Packages from the PE (e.g., `cray-mpich` are added to the database with full dependency information, and compilers from the PE are added to `compiler-s.yaml`. Available with the June 2022 release of the Cray Programming Environment.

6. **New binary format and hardened signing**

   Spack now has an updated binary format, with improvements for security. The new format has a detached signature file, and Spack verifies the signature before untarring or decompressing the binary package. The previous format embedded the signature in a `tar` file, which required the client to run `tar` *before* verifying . Spack can still install from build caches using the old format, but we encourage users to switch to the new format going forward.

   Production GitLab pipelines have been hardened to securely sign binaries. There is now a separate signing stage so that signing keys are never exposed to build system code, and signing keys are ephemeral and only live as long as the signing pipeline stage.

7. **Bootstrap mirror generation**

   The `spack bootstrap mirror` command can automatically create a mirror for bootstrapping the concretizer and other needed dependencies in an air-gapped environment.

8. **Makefile generation**

`spack env depfile` can be used to generate a `Makefile` from an environment, which can be used to build packages the environment in parallel on a single node. e.g.:

```
spack -e myenv env depfile > Makefile
make
```

Spack propagates `gmake` jobserver information to builds so that their jobs can share cores.

9. **New variant features**

In addition to being conditional themselves, variants can now have conditional *values* (https://spack.readthedocs.io/en/latest/packaging_guide.html#conditional-possible-values) ↗ that are only possible for certain configurations of a package.

Variants can be declared sticky (https://spack.readthedocs.io/en/latest/packaging_guide.html#sticky-variants) ↗, which prevents them from being enabled or disabled by the concretizer. Sticky variants must be set explicitly by users on the command line or in `packages.yaml`.

- Allow conditional possible values in variants

- Add a sticky property to variants

6.6.4.2   Other new features of note

- Environment views can optionally link only `run` dependencies with `link:run`

- `spack external find --all` finds library-only packages in addition to build dependencies

- Customizable `config:license_dir` option

- `spack external find --path PATH` takes a custom search path

- `spack spec` has a new `--format` argument like `spack find`

- `spack concretize --quiet` skips printing concretized specs

- `spack info` now has cleaner output and displays test info

- Package-level submodule option for git commit versions

- Using `/hash` syntax to refer to concrete specs in an environment now works even if `/hash` is not installed.

6.6.4.3 Major internal refactors

- full hash (see above)
- new develop versioning scheme `0.19.0-dev0`
- Allow for multiple dependencies/dependents from the same package
- Splice differing virtual packages

6.6.4.4 Performance Improvements

- Concretization of large environments with `unify: when_possible` is much faster than concretizing separately (see above)
- Single-pass view generation algorithm is 2.6x faster

6.6.4.5 Archspec improvements

- `oneapi` and `dpcpp` flag support
- better support for `M1` and `a64fx`

6.6.4.6 Removals and Deprecations

- Spack no longer supports Python `2.6`
- Removed deprecated `--run-tests` option of `spack install`; use `spack test`
- Removed deprecated `spack flake8`; use `spack style`
- Deprecate `spack:concretization` config option; use `concretizer:unify`
- Deprecate top-level module configuration; use module sets
- `spack activate` and `spack deactivate` are deprecated in favor of environments; will be removed in `0.19.0`

6.6.4.7 Notable Bugfixes

- Fix bug that broke locks with many parallel builds
- Many bugfixes and consistency improvements for the new concretizer and `--reuse`

6.6.4.8   Packages

- `CMakePackage` uses `CMAKE_INSTALL_RPATH_USE_LINK_PATH`

- Refactored `lua` support: `lua-lang` virtual supports both `lua` and `luajit` via new `LuaPackage` build system

- PythonPackage: now installs packages with `pip`

- Python: improve site_packages_dir handling

- Extends: support spec, not just package name

- Use stable URLs and `?full_index=1` for all github patches

6.7 `mpich`

`mpich` has been updated to version 3.4.2. For more information see https://www.mpich.org/2021/05/28/mpich-3-4-2-released/ ↗ .

6.8   Slurm

6.8.1   Important Notes for Upgrading Slurm Releases:

If using the slurmdbd (Slurm DataBase Daemon) you must update this first. If using a backup DBD you must start the primary first to do any database conversion, the backup will not start until this has happened.

6.8.2   Slurm version 22.05

An update to `Slurm` version 22.05 is available.

6.8.2.1   Important notes for upgrading to version 22.05

Slurmdbd version 22.05 will work Slurm daemons of version 20.11. You will not need to update all clusters at the same time, but it is very important to update slurmdbd first and having it running before updating any other clusters making use of it.

Slurm can be upgraded from version 20.11 to version 22.05 without loss of jobs or other state information. Upgrading directly from an earlier version of Slurm will result in loss of state information.

For more information and a recommended upgrade procedure, see the section "Upgrading Slurm" in the chapter "Slurm — utility for HPC workload management" of the in the SLE HPC 15 "Administration Guide".

All SPANK plugins must be recompiled when upgrading from any Slurm version prior to 22.05.

If you are using the Slurm plugin for pdsh you must make sure, `pdsh_slurm_22_05` is installed together with slurm_22_05.

6.8.2.2   Highlights of version 20.11

- The template `slurmrestd.service` unit file now defaults to listen on both the Unix socket and the `slurmrestd` port.

- The template `slurmrestd.service` unit file now defaults to enable auth/jwt and the munge unit is no longer a dependency by default.

- Add extra "EnvironmentFile=-/etc/default/$service" setting to service files.

- Allow jobs to pack onto nodes already rebooting with the desired features.

- Reset job start time after nodes are rebooted, previously only done for cloud/power save boots.

- Node features (if any) are passed to `RebootProgram` if run from `slurmctld`.

- Fail srun when using invalid `--cpu-bind options` (e.g. `--cpu-bind=map_cpu:99` when only 10 CPUs are allocated).

- Storing batch scripts and env vars are now in indexed tables using substantially less disk space. Those storing scripts in 21.08 will all be moved and indexed automatically.

- Run `MailProg` through `slurmscriptd` instead of directly fork+exec()'ing from `slurmctld`.

- Add `acct_gather_interconnect/sysfs` plugin.

- Future and Cloud nodes are treated as "Planned Down" in usage reports.

- Add new shard plugin for sharing GPUs but not with mps.

- Add support for Lenovo SD650 V2 in `acct_gather_energy/xcc` plugin.

- Remove `cgroup_allowed_devices_file.conf`, since the default policy in modern kernels is to whitelist by default. Denying specific devices must be done through `gres.conf`.

- Node state flags (`DRAIN`, `FAILED`, `POWERING UP`, etc.) will be cleared now if node state is updated to `FUTURE`.

- `srun` will no longer read in `SLURM_CPUS_PER_TASK`. This means you will implicitly have to specify `--cpus-per-task` on your `srun` calls, or set the new `SRUN_CPUS_PER_TASK` environment variable to accomplish the same thing.

- Remove `connect_timeout` and timeout options from `JobCompParams` as there's no longer a connectivity check happening in the `jobcomp/elasticsearch` plugin when setting the location off of `JobCompLoc`.

- Add support for hourly reoccurring reservations.

- Allow nodes to be dynamically added and removed from the system. Configure `MaxNode-Count` to accomodate nodes created with dynamic node registrations (`slurmd -Z --conf=""`) and `scontrol`.

- Added support for Cgroup Version 2.

- `sacct` - allocations made by `srun` will now always display the allocation and step(s). Previously, the allocation and step were combined when possible.

- `cons_tres` - change definition of the "least loaded node" (LLN) to the node with the greatest ratio of available CPUs to total CPUs.

- Add support to ship `Include` configuration files with configless.

- Provide a detailed reason in the job log as to why it has been terminated when hitting a resource limit.

- Pass and use `alias_list` through credential instead of environment variable.

- Add ability to get host addresses from `nss_slurm`.

- Enable reverse fanout for `cloud+alias_list jobs`.

- Add support to delete/update nodes by specifying nodesets or the 'ALL' keyword alongside the delete/update node message nodelist expression (i.e. `scontrol delete/update NodeName=ALL` or `scontrol delete/update NodeName=ns1,nodes[1-3]`).

- Expanded the set of environment variables accessible through Prolog/Epilog and `PrologSlurmctld`/`EpilogSlurmctld` to include `SLURM_JOB_COMMENT`, `SLURM_JOB_ST-DERR`, `SLURM_JOB_STDIN`, `SLURM_JOB_STDOUT`, `SLURM_JOB_PARTITION`, `SLURM_JOB_AC-COUNT`, `SLURM_JOB_RESERVATION`, `SLURM_JOB_CONSTRAINTS`, `SLURM_JOB_NUM_HOSTS`, `SLURM_JOB_CPUS_PER_NODE`, `SLURM_JOB_NTASKS`, and `SLURM_JOB_RESTART_COUNT`.

- Attempt to requeue jobs terminated by `slurm.conf` changes (node vanish, node socket/core change, etc). Processes may still be running on excised nodes. Admin should take precautions when removing nodes that have jobs on running on them.

- Add `switch/hpe_slingshot` plugin.

- Add new `SchedulerParameters` option `bf_licenses` to track licenses as within the backfill scheduler.

6.8.2.3   Configureation File changes (for details, see the appropriate man page)

- `AcctGatherEnergyType` `rsmi` is now `gpu`.

- `TaskAffinity` parameter was removed from `cgroup.conf`.

- Fatal if the mutually-exclusive `JobAcctGatherParams` options of `UsePss` and `NoShared` are both defined.

- `KeepAliveTime` has been moved into `CommunicationParameters`. The standalone option will be removed in a future version.

- `preempt/qos` - add support for WITHIN mode to allow for preemption between jobs within the same QOS.

- Fatal error if `CgroupReleaseAgentDir` is configured in `cgroup.conf`. The option has long been obsolete.

- Fatal if more than one burst buffer plugin is configured.

- Added `keepaliveinterval` and `keepaliveprobes` to `CommunicationParameters`.

- Added new `max_token_lifespan=<seconds>` to `AuthAltParameters` to allow sites to restrict the lifespan of any requested ticket by an unprivileged user.

- Disallow `slurm.conf` node configurations with `NodeName=ALL`.

6.8.2.4 Command Changes (for details, see the appropriate man page)

- Remove support for (non-functional) `--cpu-bind=boards`.

- Added `--prefer` option at job submission to allow for 'soft' constraints.

- Add `condflags=open` to sacctmgr show events to return open/currently down events.

- `sacct -f` flag implies `-c` flag.

- `srun --overlap` now allows the step to share all resources (CPUs, memory, and GRES), where previously `--overlap` only allowed the step to share CPUs with other steps.

6.8.2.5 API Changes

- `openapi/v0.0.35` - Plugin has been removed.

- `burst_buffer` plugins - `err_msg` added to `bb_p_job_validate()`.

- `openapi` - added flags to `slurm_openapi_p_get_specification()`. Existing plugins only need to update their prototype for the function as manipulating the flags pointer is optional.

- `openapi` - Added `OAS_FLAG_MANGLE_OPID` to allow plugins to request that the `operationId` of path methods be mangled with the full path to ensure uniqueness.

- `openapi/[db]v0.0.36` - Plugins have been marked as deprecated and will be removed in the next major release.

- switch plugins - add `switch_g_job_complete()` function.

6.8.3 Highlights of Slurm version 21.08

6.8.3.1 Highlights

- Removed `gres/mic` plugin used to support Xeon Phi coprocessors.

- Add `LimitFactor` to the QOS. A float that is factored into an associations `GrpTRES` limits. For example, if the LimitFactor is 2, then an association with a `GrpTRES` of 30 CPUs, would be allowed to allocate 60 CPUs when running under this QOS.

- A job's `next_step_id` counter now resets to 0 after being requeued. Previously, the step id's would continue from the job's last run.

- API change: Removed slurm_kill_job_msg and modified the function signature for `slurm_kill_job2`. `slurm_kill_job2` should be used instead of `slurm_kill_job_msg`.

- `AccountingStoreFlags=job_script` allows you to store the job's batch script.

- `AccountingStoreFlags=job_env` allows you to store the job's env vars.

- Removed `sched/hold` plugin.

- `cli_filter/lua`, `jobcomp/lua`, `job_submit/lua` now load their scripts from the same directory as the `slurm.conf` file (and thus now will respect changes to the `SLURM_CONF` environment variable).

- `SPANK` - call `slurm_spank_init` if defined without `slurm_spank_slurmd_exit` in `slurmd` context.

- Add new `PLANNED` state to a node to represent when the backfill scheduler has it planned to be used in the future instead of showing as `IDLE`. sreport also has changed it's cluster utilization report column name from 'Reserved' to 'Planned' to match this nomenclature.

- Put node into `INVAL` state upon registering with an invalid node configuration. Node must register with a valid configuration to continue.

- Remove `SLURM_DIST_LLLP` environment variable in favor of just `SLURM_DISTRIBUTION`.

- Make `--cpu-bind=threads` default for `--threads-per-core` — can be overridden by the CLI or an environment variable.

- `slurmd` - allow multiple comma-separated controllers to be specified in configless mode with `--conf-server`

- Manually powering down of nodes with `scontrol` now ignores `SuspendExc<Nodes|Parts>`.

- Distinguish queued reboot requests (REBOOT@) from issued reboots (REBOOT^).

- `auth/jwt` - add support for RS256 tokens. Also permit the username in the 'username' field in addition to the 'sun' (Slurm UserName) field.

- service files - change dependency to network-online rather than just network to ensure DNS and other services are available.

- Add "Extra" field to node to store extra information other than a comment.

- Add `ResumeTimeout`, `SuspendTimeout` and `SuspendTime` to Partitions.

- The `memory.force_empty` parameters is no longer set by `jobacct_gather/cgroup` when deleting the cgroup`. This previously caused a significant delay (~2s) when terminating a job, and is not believed to have provided any perceivable benefit. However, this may lead to slightly higher reported kernel mem page cache usage since the kernel cgroup memory is no longer freed immediately.

- `TaskPluginParam=verbose` is now treated as a default. Previously it would be applied regardless of the job specifying a `--cpu-bind`.

- Add `node_reg_mem_percent` `SlurmctldParameter` to define percentage of memory nodes are allowed to register with.

- Define and separate node power state transitions. Previously a powering down node was in both states, `POWERING_OFF` and `POWERED_OFF`. These are now separated. e.g. `IDLE+POWERED_OFF` (`IDLE~`) → `IDLE+POWERING_UP` (`IDLE#`) - Manual power up or allocation → `IDLE` → `IDLE+POWER_DOWN` (`IDLE!`) - Node waiting for power down → `IDLE+POWERING_DOWN` (`IDLE%`) - Node powering down → `IDLE+POWERED_OFF` (`IDLE~`) - Powered off

- Some node state flag names have changed. These would be noticeable for example if using a state flag to filter nodes with sinfo. e.g. `POWER_UP` → `POWERING_UP` `POWER_DOWN` → `POWERED_DOWN` `POWER_DOWN` now represents a node pending power down

- Create a new process called `slurmscriptd` which runs `PrologSlurmctld` and `EpilogSlurmctld`. This avoids `fork()` calls from `slurmctld`, and can avoid performance issues if the slurmctld has a large memory footprint.

- Pass JSON of job to node mappings to `ResumeProgram`.

- QOS accrue limits only apply to the job QOS, not partition QOS.

- Any return code from SPANK plugin or SPANK function that is not `SLURM_SUCCESS` (zero) will be considered to be an error. Previously, only negative return codes were considered an error.

- Add support for automatically detecting and broadcasting executable shared object dependencies for `sbcast` and `srun --bcast`.

- All SPANK error codes now start at 3000. Where previously SPANK would give a return code of 1, it will now return 3000. This change will break ABI compatibility with SPANK plugins compiled against older version of Slurm.

- SPANK plugins are now required to match the current Slurm release, and must be recompiled for each new Slurm major release. (They do not need to be recompiled when upgrading between maintenance releases.)

- `SLURM_NODE_ALIASES` now has brackets around the node's address to be able to distinguish IPv6 addresses. e.g. `<node_name>:[<node_addr>]:<node_hostname>`

- The `job_container/tmpfs` plugin now requires `PrologFlags=contain` to be set in `slurm.conf`.

- Limit `max_script_size` to 512 MB.

6.8.3.2 Configuration File Changes (for details, see the appropriate man page)

- Errors detected in the parser handlers due to invalid configurations are now propagated and can lead to fatal (and thus exit) the calling process.

- Enforce a valid configuration for `AccountingStorageEnforce` in `slurm.conf`. If the configuration is invalid, then an error message will be printed and the command or daemon (including `slurmctld`) will not run.

- Removed `AccountingStoreJobComment` option. Please update your config to use `AccountingStoreFlags=job_comment` instead.

- Removed `DefaultStorage{Host,Loc,Pass,Port,Type,User}` options.

- Removed `CacheGroups`, `CheckpointType`, `JobCheckpointDir`, `MemLimitEnforce`, `SchedulerPort`, `SchedulerRootFilter` options.

- Added `Script` to `DebugFlags` for debugging `slurmscriptd` (the process that runs `slurmctld` scripts such as `PrologSlurmctld` and `EpilogSlurmctld`).

- Rename `SbcastParameters` to `BcastParameters`.

- systemd service files - add new "-s" option to each daemon which will change the working directory even with the `-D` option. (Ensures any core files are placed in an accessible location, rather than `/`.)

- Added `BcastParameters=send_libs` and `BcastExclude` options.

- Remove the (incomplete) `burst_buffer/generic` plugin.

- Make `SelectTypeParameters=CR_Core_Memory` default for `cons_tres` and `cons_res`.

- Remove support for `TaskAffinity=yes` in `cgroup.conf`. Adding `task/affinity` to `TaskPlugins` in `slurm.conf` is strongly recommended instead.

6.8.3.3 Command Changes (for details, see the appropriate man page)

- Changed the `--format` handling for negative field widths (left justified) to apply to the column headers as well as the printed fields.

- Invalidate multiple partition requests when using partition based associations.

- `scrontab` - create the temporary file under the `TMPDIR` environment variable (if set), otherwise continue to use `TmpFS` as configured in `slurm.conf`.

- `sbcast / srun --bcast` - removed support for zlib compression. lz4 is vastly superior in performance, and (counter-intuitively) zlib could provide worse performance than no compression at all on many systems.

- `sacctmgr` - changed column headings to `ParentID` and `ParentName` instead of `Par ID` and "Par Name` respectively.

- `SALLOC_THREADS_PER_CORE` and `SBATCH_THREADS_PER_CORE` have been added as input environment variables for `salloc` and `sbatch`, respectively. They do the same thing as `--threads-per-core`.

- Don't display node's comment with `scontrol show nodes` unless set.

- Added `SLURM_GPUS_ON_NODE` environment variable within each job/step.

- `sreport` - change to sorting `TopUsage` by the `--tres` option.

- `slurmrestd` - do not run allow operation as `SlurmUser`/root by default.

- `scontrol show node` now shows `State` as base_state + flags instead of shortened state with flags appended. eg. `IDLE#` → `IDLE+POWERING_UP`. Also `POWER` state flag string is `POWERED_DOWN`.

- `scrontab` - add ability to update crontab from a file or standard input.

- `scrontab` - added ability to set and expand variables.

- Make `srun` sensitive to `BcastParameters`.

- Added `sbcast/srun --send-libs`, `sbcast --exclude` and `srun --bcast-exclude`.

- Changed `ReqMem` field in `sacct` to match memory from ReqTRES. It now shows the requested memory of the whole job with a letter appended indicating units (M for megabytes, G for gigabytes, etc.). `ReqMem` is only displayed for the job, since the step does not have requested TRES. Previously `ReqMem` was also displayed for the step but was just displaying `ReqMem` for the job.

6.8.3.4 API Changes

- `jobcomp` plugin: change plugin API to `jobcomp_p_*()`.

- `sched` plugin: change plugin API to `sched_p_*()` and remove `slurm_sched_p_initial_priority()` call.

- `step_ctx code` has been removed from the api.

- `slurm_stepd_get_info()`/`stepd_get_info()` has been removed from the api.

- The v0.0.35 OpenAPI plugin has now been marked as deprecated. Please convert your requests to the v0.0.37 OpenAPI plugin.

6.9 Creating containers from current HPC environment

Usually users use environment modules to adjust their environment (that is, environment variables like `PATH`, `LD_LIBRARY_PATH`, `MANPATH` etc.) to pick exactly the tools and libraries they need for their work. The same can be achieved with containers by including only those components in a container that are part of this environment. This functionality is now provided using the `spack` and `singularity` applications.

6.10 Slurm 23.02

6.10.1 Important Notes on Upgrading Slurm from a Previous Version

If using the `slurmdbd` (Slurm DataBase Daemon) you must update this first.

If using a backup DBD you must start the primary first to do any database conversion, the backup will not start until this has happened.

The 23.02 `slurmdbd` will work with Slurm daemons of version 21.08 and above. You will not need to update all clusters at the same time, but it is very important to update `slurmdbd` first and having it running before updating any other clusters making use of it.

Slurm can be upgraded from version 22.05 to version 23.02 without loss of jobs or other state information. Upgrading directly from an earlier version of Slurm will result in loss of state information.

All `SPANK` plugins must be recompiled when upgrading from any Slurm version prior to 23.02.

> **Note**
>
> PMIx v1.x is no longer supported.

6.10.2 Highlights

- `slurmctld` - Add new RPC rate limiting feature. This is enabled through `SlurmctldParameters=rl_enable`, otherwise disabled by default.

- Make `scontrol` reconfigure and sending a `SIGHUP` to the `slurmctld` behave the same. If you were using `SIGHUP` as a 'lighter' scontrol reconfigure to rotate logs please update your scripts to use SIGUSR2 instead.

- Change cloud nodes to show by default. `PrivateData=cloud` is no longer needed.

- `sreport` - Count planned (FKA reserved) time for jobs running in `IGNORE_JOBS` reservations. Previously was lumped into `IDLE` time.

- `job_container/tmpfs` - Support running with an arbitrary list of private mount points (`/tmp` and `/dev/shm` are the default, but not required).

- `job_container/tmpfs` - Set more environment variables in `InitScript`.

- Make all cgroup directories created by Slurm owned by root. This was the behavior in cgroup/v2 but not in cgroup/v1 where by default the step directories ownership were set to the user and group of the job.

- `accounting_storage/mysql` - change purge/archive to calculate record ages based on end time, rather than start or submission times.

- `job_submit/lua` - add support for `log_user()` from `slurm_job_modify()`.

- Run the following scripts in slurmscriptd instead of `slurmctld`: `ResumeProgram`, `ResumeFailProgram`, `SuspendProgram`, `ResvProlog`, `ResvEpilog`, and `RebootProgram` (only with `SlurmctldParameters=reboot_from_controller`).

- Only permit changing log levels with `srun --slurmd-debug` by root or `SlurmUser`.

- `slurmctld` will `fatal()` when reconfiguring the `job_submit` plugin fails.

- Add PowerDownOnIdle partition option to power down nodes after nodes become idle.

- Add "[jobid.stepid]" prefix from `slurmstepd` and "slurmscriptd" prefix from `slurmscriptd` to Syslog logging. Previously was only happening when logging to a file.

- Add purge and archive functionality for job environment and job batch script records.

- Extend support for Include files to all "configless" client commands.

- Make node weight usable for powered down and rebooting nodes.

- Removed "launch" plugin.

- Add "Extra" field to job to store extra information other than a comment.

- Add usage gathering for AMD (requires ROCM 5.5+) and NVIDIA gpus.

- Add job's allocated nodes, features, oversubscribe, partition, and reservation to `SLURM_RESUME_FILE` output for power saving.

- Automatically create directories for stdout/stderr output files. Paths may use `%j` and related substitution characters as well.

- Add `--tres-per-task` to `salloc`/`sbatch`/`srun`.

- Allow `nodefeatures` plugin features to work with cloud nodes. e.g. - Powered down nodes have no active changeable features.

    - Nodes can't be changed to other active features until powered down.

    - Active changeable features are reset/cleared on power down.

- Make `slurmstepd` cgroups constrained by total configured memory from `slurm.conf` (`NodeName=<> RealMemory=#`) instead of total physical memory.

- `node_features/helpers` - add support for the OR and parentheses operators in a `--constraint` expression.

- `slurmctld` will `fatal()` when [Prolog|Epilog]Slurmctld are defined but are not executable.

- Validate node registered active features are a super set of node's currently active changeable features.

- On clusters without any `PrologFlags` options, batch jobs with failed prologs nolonger generate an output file.

- Add `SLURM_JOB_START_TIME` and `SLURM_JOB_END_TIME` environment variables.

- Add `SuspendExcStates` option to `slurm.conf` to avoid suspending/powering down specific node states.

- Add support for DCMI power readings in IPMI plugin.

- `slurmrestd` served `/slurm/v0.0.39` and `/slurmdb/v0.0.39` endpoints had major changes from prior versions. Almost all schemas have been renamed and modified. Sites using OpenAPI Generator clients are highly suggested to upgrade to to using atleast version 6.x due to limitations with prior versions.

- Allow for `--nodelist` to contain more nodes than required by `--nodes`.

- Rename "nodes" to "nodes_resume" in `SLURM_RESUME_FILE` job output.

- Rename "all_nodes" to "all_nodes_resume" in `SLURM_RESUME_FILE` output.

- Add `jobcomp/kafka` plugin.

- Add new `PreemptParameters=reclaim_licenses` option which will allow higher priority jobs to preempt jobs to free up used licenses. (This is only enabled for with `PreemptModes` of `CANCEL` and `REQUEUE`, as Slurm cannot guarantee suspended jobs will release licenses correctly.)

- `hpe/slingshot` - add support for the instant-on feature.

- Add ability to update `SuspendExc*` parameters with `scontrol`.

- Add ability to restore `SuspendExc*` parameters on restart with `slurmctld -R` option.

- Add ability to clear a GRES specification by setting it to "0" via "scontrol update job".

- Add `SLURM_JOB_OVERSUBSCRIBE` environment variable for `Epilog`, `Prolog`, `EpilogSlurmctld`, `PrologSlurmctld`, and mail ouput.

- System node down reasons are appended to existing reasons, separated by ':'.

- New command scrun has been added. scrun acts as an Open Container Initiative (OCI) runtime proxy to run containers seamlessly via Slurm.

- Fixed `GpuFreqDef` option. When set in `slurm.conf`, it will be used if `--gpu-freq` was not explicitly set by the job step.

6.10.3   Configuration File Changes (see appropriate man page for details)

- `job_container.conf` - Added "Dirs" option to list desired private mount points.

- `node_features` plugins - invalid users specified for `AllowUserBoot` will now result in `fatal()` rather than just an error.

- Deprecate `AllowedKmemSpace`, `ConstrainKmemSpace`, `MaxKmemPercent`, and `MinKmemSpace`.

- Allow jobs to queue even if the user is not in `AllowGroups` when `EnforcePartLimits=no` is set. This ensures consistency for all the Partition access controls, and matches the documented behavior for `EnforcePartLimits`.

- Add `InfluxDBTimeout` parameter to `acct_gather.conf`.

- `job_container/tmpfs` - add support for expanding `%h` and `%n` in `BasePath`.

- `slurm.conf` - Removed `SlurmctldPlugstack` option.

- Add new `SlurmctldParameters=validate_nodeaddr_threads=<number>` option to allow concurrent hostname resolution at `slurmctld` startup.

- Add new `AccountingStoreFlags=job_extra` option to store a job's extra field in the database.

- Add new "defer_batch" option to `SchedulerParameters` to only defer scheduling for batch jobs.

- Add new `DebugFlags` option "JobComp" to replace "Elasticsearch".

- Add configurable job requeue limit parameter - `MaxBatchRequeue` - in `slurm.conf` to permit changes from the old hard-coded value of 5.

- `helpers.conf` - Allow specification of node specific features.

- `helpers.conf` - Allow many features to one helper script.

- `job_container/tmpfs` - Add "Shared" option to support shared namespaces. This allows autofs to work with the `job_container/tmpfs` plugin when enabled.

- `acct_gather.conf` - Added `EnergyIPMIPowerSensors=Node=DCMI` and `Node=DCMI_ENHANCED`.

- Add new "getnameinfo_cache_timeout=<number>" option to CommunicationParameters to adjust or disable caching the results of `getnameinfo()`.

- Add new PrologFlags=ForceRequeueOnFail option to automatically requeue batch jobs on Prolog failures regardless of the job --requeue setting.

- Add `HealthCheckNodeState=NONDRAINED_IDLE` option.

- Add "explicit" to Flags in `gres.conf`. This makes it so the gres is not automatically added to a job's allocation when `--exclusive` is used. Note that this is a per-node flag.

- Moved the "preempt_" options from `SchedulerParameters` to `PreemptParameters`, and dropped the prefix from the option names. (The old options will still be parsed for backwards compatibility, but are now undocumented.)

- Add `LaunchParameters=ulimit_pam_adopt`, which enables setting `RLIMIT_RSS` in adopted processes.

- Update SwitchParameters=job_vni to enable/disable creating job VNIs for all jobs, or when a user requests them.

- Update `SwitchParameters=single_node_vni` to enable/disable creating single node VNIs for all jobs, or when a user requests them.

- Add ability to preserve `SuspendExc*` parameters on reconfig with `ReconfigFlags=KeepPowerSaveSettings`.

- `slurmdbd.conf` - Add new `AllResourcesAbsolute` to force all new resources to be created with the `Absolute` flag.

- `topology/tree` - Add new `TopologyParam=SwitchAsNodeRank` option to reorder nodes based on switch layout. This can be useful if the naming convention for the nodes does not natually map to the network topology.

- Removed the default setting for `GpuFreqDef`. If unset, no attempt to change the GPU frequency will be made if `--gpu-freq` is not set for the step.

6.10.4   Command Changes (see man pages for details)

- `sacctmgr` - no longer force updates to the `AdminComment`, `Comment`, or `SystemComment` to lower-case.

- `sinfo` - Add `-F/--future` option to sinfo to display future nodes.

- `sacct` - Rename "Reserved" field to "Planned" to match sreport and the nomenclature of the 'Planned' node.

- `scontrol` - advanced reservation flag `MAINT` will no longer replace nodes, similar to `STATIC_ALLOC`

- `sbatch` - add parsing for `#PBS -d` and `#PBS -w`.

- `scontrol` show `assoc_mgr` will show username(uid) instead of uid in QoS section.

- Add `strigger --draining` and `-R/--resume` options.

- Change `--oversubscribe` and `--exclusive` to be mutually exclusive for job submission. Job submission commands will now fatal if both are set. Previously, these options would override each other, with the last one in the job submission command taking effect.

- `scontrol` - Requested `TRES` and allocated `TRES` will now always be printed when showing jobs, instead of one `TRES` output that was either the requested or allocated.

- `srun --ntasks-per-core` now applies to job and step allocations. Now, use of `--ntasks-per-core=1` implies `--cpu-bind=cores` and `--ntasks-per-core>1` implies `--cpu-bind=threads`.

- `salloc`/`sbatch`/`srun` - Check and abort if ntasks-per-core > threads-per-core.

- `scontrol` - Add `ResumeAfter=<secs>` option to "scontrol update nodename=".

- Add a new "nodes=" argument to `scontrol setdebug` to allow the debug level on the `slurmd` processes to be temporarily altered.

- Add a new "nodes=" argument to "scontrol setdebugflags" as well.

- Make it so `scrontab` prints client-side the `job_submit()` error messsage (which can be set i.e. by using the `log_user()` function for the lua plugin).

- `scontrol` - Reservations will not be allowed to have `STATIC_ALLOC` or `MAINT` flags and `REPLACE[_DOWN]` flags simultaneously.

- `scontrol` - Reservations will only accept one reoccurring flag when being created or updated.

- `scontrol` - A reservation cannot be updated to be reoccurring if it is already a floating reservation.

- `squeue` - removed unused "%s" and "SelectJobInfo" formats.

- `squeue` - align print format for exit and derived codes with that of other components (`<exit_status>:<signal_number>`).

- `sacct` - Add `--array` option to expand job arrays and display array tasks on separate lines.

- Partial support for "--json" and "--yaml" formated outputs have been implemented for `sacctmgr`, `sdiag`, `sinfo`, `squeue`, and `scontrol`. The resultant data ouput will be filtered by normal command arguments. Formatting arguments will continue to be ignored.

- `salloc` / `sbatch` / `srun` - extended the `--nodes` syntax to allow for a list of valid node counts to be allocated to the job. This also supports a "step count" value (e.g., `--nodes=20-100:20` is equivalent to `--nodes=20,40,60,80,100`) which can simplify the syntax when the job needs to scale by a certain "chunk" size.

- `srun` - add user requestible vnis with "--network=job_vni" option.

- `srun` - add user requestible single node VNIs with the "--network=single_node_vni" option.

6.10.5   API Changes

- `job_container` plugins - `container_p_stepd_create()` function signature replaced `uint32_t` uid with `stepd_step_rec_t*` step.

- `gres` plugins - `gres_g_get_devices()` function signature replaced `pid_t` pid with `stepd_step_rec_t*` step.

- `cgroup` plugins - `task_cgroup_devices_constrain()` function signature removed `pid_t` pid.

- `task` plugins - replace `task_p_pre_set_affinity()`, `task_p_set_affinity()`, and `task_p_post_set_affinity()` with `task_p_pre_launch_priv()` like it was back in slurm 20.11.

- Allow for concurrent processing of `job_submit_g_submit()` and `job_submit_g_modify()` calls. If your plugin is not capable of concurrent operation you must add additional locking within your plugin.

- Removed return value from `slurm_list_append()`.

- The `List` and `ListIterator` types have been removed in favor of `list_t` and `list_itr_t` respectively.

- `burst buffer` plugins - add `bb_g_build_het_job_script()`. `bb_g_get_status()` - added authenticated UID and GID. `bb_g_run_script()` - added `job_info` argument.

- `burst_buffer.lua` - Pass UID and GID to most hooks. Pass `job_info` (detailed job information) to many hooks. See `etc/burst_buffer.lua.example` for a complete list of changes. WARNING: Backwards compatibility is broken for `slurm_bb_get_status`: UID and GID are passed before the variadic arguments. If UID and GID are not explicitly listed as arguments to `slurm_bb_get_status()`, then they will be included in the variadic arguments. Backwards compatibility is maintained for all other hooks because the new arguments are passed after the existing arguments.

- `node_features` plugins - `node_features_p_reboot_weight()` function removed. `node_features_p_job_valid()` - added parameter `feature_list`. `node_features_p_job_xlate()` - added parameters `feature_list` and `job_node_bitmap`.

- New `data_parser` interface with v0.0.39 plugin.

7   Removed and deprecated features and packages

This section lists features and packages that were removed from SUSE Linux Enterprise for High-Performance Computing or will be removed in upcoming versions.

7.1 Removed features and packages

The following features and packages have been removed in this release.

- Python 2 bindings for `genders` has been removed. These are now provided for Python 3.

- Ganglia is not supported anymore in 15 SP4. It has been replaced with Grafana (https://grafana.com/ ↗)

- Due to a lack of usage by customers, some library packages have been removed from the HPC module in SLE HPC 15 SP4. On SUSE Linux Enterprise you can build your own library using `spack`. These libraries will continue to be available through SUSE Package Hub. The following libraries have been removed:

  - `boost`

  - `adios`

  - `gsl`

  - `fftw3`

  - `hypre`

  - `metis`

  - `mumps`

  - `netcdf`

  - `ocr`

  - `petsc`

  - `ptscotch`

  - `scalapack`

- `superlu`

- `trilinos`

7.2 Deprecated features and packages

The following features and packages are deprecated and will be removed in a future version of SUSE Linux Enterprise for High-Performance Computing.

8 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 sle_source_request@suse.com (mailto:sle_source_request@suse.com)↗. SUSE may charge a reasonable fee to recover distribution costs.

9 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.

A Changelog for 15 SP4

A.1 Pre-release

A.1.1 New

- *Section 6.10, "Slurm 23.02"* (Jira (https://jira.suse.com/browse/PED-2802) ↗)

- *Section 6.6, "spack"* (Jira (https://jira.suse.com/browse/PED-2803) ↗)

A.2   2022-11-30

A.2.1   New

- Added *Section 6.1, "Enriched system visibility in the SUSE Customer Center (SCC)"* (Jira (https://jira.suse.com/browse/SLE-24988) ↗)

A.3   2022-08-31

A.3.1   New

- Added *Section 6.2, "Automatically opened ports"* (Jira (https://jira.suse.com/browse/SLE-22743) ↗)

A.4   2022-05-11

A.4.1   New

- Added this changelog

A.5   2022-03-23

A.5.1   New

- Added *Section 6.8, "Slurm"* (Jira (https://jira.suse.com/browse/SLE-12352) ↗)
- Added notes about `dolly`, `memkind`, `openblas`, `spack`, and `mpich` in *Section 6, "Changes affecting all architectures"*
- Added note about Ganglia being unsupported in *Section 7, "Removed and deprecated features and packages"* (Jira (https://jira.suse.com/browse/SLE-17777) ↗)
- Added note about removal of Python 2 bindings for `genders` (Jira (https://jira.suse.com/browse/SLE-23359) ↗)

A.5.2 Updates

- Added a note about building libraries using `spack` in *Section 7, "Removed and deprecated features and packages"* (Jira (https://jira.suse.com/browse/SLE-17776) ⬈)

- Added `adios` and `superlu` to the list of removed libraries in *Section 7, "Removed and deprecated features and packages"*

A.6 2021-11-03

- Initial SP4 release