CUG 2026 Slurm and Slinky Roadmap Update

Douglas Jacobsen

April 28, 2026

SchedMD Offerings

All products and services will continue and now be supported by NVIDIA

Workload Management

Leading workload manager in HPC community

Brings Slurm capabilities into Kubernetes

- Open-source
- Vendor-agnostic
- On-prem and cloud
- Support, training, and consultation available

Development Cycle

Release Cycle

- Major releases are made every six months
- Version is the two digit year, two digit month:
  - 25.05 - May 2025
  - 25.11 - November 2025
  - 26.06 - May 2026
- Major releases are supported for 18 months
  - Currently: 25.11, 25.05, 24.11
- Maintenance releases are made roughly monthly
  - Usually only for the most recent major release
  - Security releases are made for all supported major releases

Development Process

- Major projects are scoped and designed in collaboration with key customers
- Community contributions are welcomed as GitHub pull requests
  - https://github.com/SchedMD/slurm/pulls
- For issues, please open a support ticket at https://support.schedmd.com

Slurm 25.11 Release Highlights

Linux Namespace Plugin

- **Introduced new top-level namespace plugins**
  - namespace/linux is the target for future Slurm namespace development
    - replaces functionality of job_container/tmpfs
    - optionally enables user and pid namespaces
    - can configure different nodes differently
  - namespace/tmpfs is a migration of the legacy job_container/tmpfs
    - Preserves the same configuration file and format
    - Migrate to namespace/linux when feasible

Slurm's Advanced Topology Planning (topology/block)

- Node rank ordering based on topology
  - Slurm orders nodes based on the order in which they’re defined in slurm.conf
  - For dynamic nodes, this varies based on when the node is created or registered to the cluster
    - This causes issues for topology/block in particular, where the node ordering has traditionally reflected the block layout
  - Automatically order nodes based on the topological layout instead
- New CLI options to control how jobs are mapped to blocks

Expedited Requeue

Hero job support in the face of node instability

- Must be explicitly enabled on the system:
  - `SlurmctldParameters=enable_expedited_requeue`

- Jobs must opt-in to this behavior with the `--requeue=expedite` option
  - Systems should consider limiting access to this option through `job_submit.lua`
  - And implementing alongside Preemption mechanisms to ensure replacement nodes are made available immediately

Hierarchical Resources

Planning for resources adjacent to traditional compute

- Hierarchical resources provide scheduling for resources adjacent to compute, but that aren't exclusive to specific compute nodes
- Slurm has traditionally supported planning for "licenses"
  - But assumes every license is accessible from every node

Three modes of hierarchical resource operation, each with their own topology implications:

- **Mode 1** – Resource must be available in any adjacent layer to the compute node the job runs on
  - One mode of operation for interconnect-offloaded network resources
- **Mode 2** – Equivalent resources must be available at all layers
  - Second mode of operation for other network resources
- **Mode 3** – Provides for resource aggregation as you move up through the hierarchy
  - Can model power capping limits within the datacenter, from the PDU to rack, aisle, and DC level

Observability

How to tell what's running on your systems, and correlate with other systems

- Expanded Slurm's Kafka integration to provide job details at start as well as completion
- Internalized support for exporting OpenMetrics (Prometheus) metrics directly from slurmctld

- New identifier introduced for better traceability – SLUID
  - SLUID is an acronym – "Slurm Lexicographically-sortable Unique ID"
  - Unique to each run of a job
  - Never reused
    - JobId values wrap after 67,043,328
    - And repeat for requeued jobs – SLUID will change on requeue

Example SLUID: s8FXJCCS3F9Z00

Slurm 26.05 Release Highlights

Dynamic Memory Limits

- Job launches with the initial user-set limit on memory
- Later, the job can release unneeded memory which can then be used by other jobs
  - Can now update a running job to *lower* memory usage
    - `scontrol update job=1234 MinMemoryNode=1024m`
  - New option to sbatch command to automatically check memory usage at after a given time, and automatically update the memory limit to the current usage plus some configurable buffer
    - `sbatch --mem-update=10@00:15:00`

Slurm Scalability Improvements

- New threadpool capability - preallocates and preserves threads for reuse rather than constant dynamic re-creation
  - `SlurmctldParameters=threadpool=enabled,threadpool_preallocate=32,threadpool_preserve=128`
- Continuing to drive more communcation from legacy EIO system to conmgr
  - srun step launch
  - X11 connection forwarding
- Conmgr Improvements
  - Per-connection error detection, tuned slurmd defaults

HPE Slingshot Improvements

- Improved support for Slingshot Hardware Accelerated Collectives:
  - `switch/hpe_slingshot`, add `SwitchParameters=fm_authdir, fm_authdir_ctld`
- Streamline job step/stepmgr interactions with fabric manager
  - Prevent extern steps from trying to delete job references from the fabric manager (stepmgr only)

New Topology Plugins

- `topology/ring` – single-dimensional torus
- `topology/torus3d` - three-dimensional torus

New slurmrestd Endpoints

- Job Requeue:
  - GET /slurm/{version}/job/{job_id}/requeue[?{flags}]
    - Enables scontrol requeue and scontrol requeuehold functionality (depending on flags)
- Slurm Configuration
  - GET /slurm/{version}/conf
  - GET /slurmdb/{version}/conf

Mini Batch

Building on stepmgr capability, create queue of steps within a job allocation

- `srun –async` … to queue a step.
- Queue many steps and offload task management to stepmgr

Exclusive Job Updates

- Clarified `srun --exclusive` behavior when srun is used to both allocate a job and run a step at the same time
  - Added new srun option `--exclusive=allocation` to get a node-exclusive allocation without implying `--exact`
- New OverSubscribe field in `sacct` output to show which options for `--exclusive/--oversubscribe` were used in job allocation

Expanded metrics support

- Add gpu statistics to the new built-in openmetrics support
- Support optional authentication for the openmetrics interfaces

Alternative to libhttpparser

- New httpparser and urlparser plugins based on libllhttp to replace the deprecated libhttpparser library

Changes to remote licenses / resources

- Option to preserve the case for licenses (remote resources) managed through SlurmDBD
- Allow remote resources to be requested without the “@server” suffix
- Allow a remote resource to be 100% accessed by multiple clusters, rather than requiring subdivision
  - Expected to be used alongside a cron job updating the “LastConsumed” value

And Beyond

- Slurm 26.11 will have many new larger features, but is still being scoped
  - Current ideas (not committed 100% yet)
    - HRES Lua-based Plugin Architecture
    - More flexible mount namespace options in namespace/linux
    - Improved dynamic management of topology/block
    - And many more

Slurm References

Share with your customers and partners

- Documentation:
  - https://slurm.schedmd.com/
- Repositories:
  - https://github.com/SchedMD/Slurm
- Slurm Roadmap from SC’25:
  - https://slurm.schedmd.com/SC25/Slurm_BoF_SC25.pdf
- GTC’26 Slurm/Slinky Sesson:
  - https://www.nvidia.com/en-us/on-demand/session/gtc26-S82420

Slinky

Slinky

Slinky is not an acronym

Because the best we could come up with was "**Sl**urm **in K**ubernetes, **Y**es"

- Slinky is the toolkit of components to enable Slurm operation in Kubernetes environments
  - Started with the Slurm Operator to manage Slurm cluster deployments on Kubernetes clusters
  - Expanded to the Slurm Bridge to plug Slurm directly into Kubernete's scheduling API
  - Includes additional tooling
    - Slurm Client for golang access to Slurm's REST API
    - As well as various container images

Slinky References

- Documentation:
  - https://slinky.schedmd.com/
- Repositories:
  - https://github.com/SlinkyProject
- More detailed presentations:
  - Slurm Bridge - https://slurm.schedmd.com/MISC25/Slurm_Bridge_KubeCon_25.pdf
  - Slurm Operator - https://slurm.schedmd.com/SLUG24/Slinky-Slurm-Operator.pdf

Slinky - Slurm Operator

Slurm Operator

Reference architectures for deploying Slurm in Kubernetes

- Deploy and manage Slurm clusters within a Kubernetes environment
  - Each Slurm compute node maps to a Kubernetes pod running the slurmd process
- Support autoscaling based on cluster utilization metrics
- Runs Slurm jobs natively within Slurm’s own resource management layer
  - Allows for fast deployment of multi-node workloads
- Users interact with Slurm through traditional CLI tools
  - Through one or more "login node" pods they can SSH into
  - Kubernetes is not involved in scheduling or managing compute jobs
- Slurm runs Slurm workloads directly
  - Allows for fine-grained resource limits
  - Backfill scheduling
  - Advanced topology planning

Slurm Operator - Updates for v1.1.0

- New daemonset scaling mode
  - Consistent mapping between Kubernetes node name and Slurm name
- Support for pod topology
  - Node annotation injected into the slurmd pod to trigger Slurm's automatic topology mode
  - `topology.slinky.slurm.net/spec: topo-switch:s0,topo-block:b0`
- Added PodDisruptionBudget support for slurm-operator and slurm-operator-webhook pods

Slinky - Slurm Bridge

Slurm Bridge

- Slurm Bridge enables Slurm as a first-class Kubernetes scheduler
  - Enable Slurm's scheduling capabilities – efficient multi-node planning, backfill, fair share, … – in Kubernetes environments
- For compute nodes, the Slurm Bridge assumes responsibility for workload planning
  - Initial focus has been on multi-node workloads
    - Work continues to provide more granular mechanisms for sub-node workloads as well

Slurm Bridge

Technical overview

- Slurm as a Kubernetes scheduler using the Scheduling Framework
  - Uses PreEnqueue, PreFilter, Filter, and PostFilter for placement decision
  - Uses PreBind to generate DRA ResourceClaims
- Slurm Bridge translates pod resource requirements into a Slurm “external job”
  - External jobs – compared to traditional batch jobs – are only a request for Slurm to allocate resources
    - No batch script, no job environment, no processes launched on the compute node
  - Scheduled by Slurm – launched by kubelet
- Slurm schedules to nodes running slurmd, or to “external nodes” without slurmd
  - Allows for “converged” system designs

Slurm Bridge - Updates for v1.1.0

v1.1, a.k.a. DRAapalooza

- Support for automatically creating Slurm “external” nodes from labelled Kubernetes nodes
  - `scheduler.slinky.slurm.net/external-node: true`
  - `scheduler.slinky.slurm.net/external-node-partitions: slurm-bridge`
- Support for emitting resource claims against the DRA GPU driver
- Support for emitting resource claims against the DRA CPU driver
- Beta support for NVLink Compute Domain setup
- New annotation to allow for sub-node resource allocation for a Pod:
  - `slurmjob.slinky.slurm.net/exclusive: false`

Upcoming Releases

What’s Next?

Roadmap

Upcoming releases

Slurm

- Slurm 26.05 in May
- Slurm 26.11 in November

Slinky

- Slinky v1.1.0 available now
  - New daemonset scaling mode for Slurm Operator
  - DRA support in Slurm Bridge
- Slinky v1.2.0 this summer


NVIDIA