# Slurm ~~24.08~~ 24.05, and Beyond

Tim Wickberg
*Chief Technology Officer*

slurm | SCHEDMD

# Development Cycle

# New Release Cycle

- Slurm has switched to a six month major release cycle
    - Previously nine months
- Major releases will be made every May and November going forward
    - Rather than... November, August, May, February, then November again...
- Goals with the switch:
    - Faster feature delivery for bleeding-edge systems
    - Alignment with annual maintenance windows for stable systems
        - "Predictable" timelines for major releases
        - We expect a lot of sites will stick with either May or November releases, and will skip every-other major release going forward

# Revised Support Cycle

- Slurm 23.02 will be supported until 24.11 is released
  - Window extended to 21 months, instead of 18 months, to help with transition period
- Direct upgrades from 3 prior major releases will be supported starting with 24.05
  - Prior to 24.05, upgrades are only supported from the previous 2 major releases
  - E.g., you can directly upgrade to Slurm 24.11 from 24.05, 23.11, or 23.02.

# Release Cycle

- Major releases are made every ~~nine~~ six months
- Major version numbers are the two digit year, two digit month:
  - 23.02 - February 2023
  - 23.11 - November 2023
  - 24.05 - May 2024
  - 24.11 - November 2024
- Major releases are supported for 18 months
  - Currently: 23.11 and 23.02
  - May - November 2024: 24.05, 23.11, 23.02
  - November - May 2024: 24.11, 24.05, 23.11
- Maintenance releases are made roughly monthly
  - Usually only made for the most recent major release
  - Security issues are fixed in all currently supported major releases

# Enhancement Requests

- SchedMD's Bugzilla installation catalogs outstanding enhancement requests under the "Sev 5 - Enhancement" severity level
  - Unless indicated through the "Target Release" field, SchedMD has not committed to delivering that enhancement on any specific time-frame (if ever)
    - Currently 548 open tickets... around 30 may make it into a release
- Customer enhancement requests are automatically re-routed to Sev 4 on submission
  - Allows for some initial triage and discussion
    - Will move to Sev 5 if we agree that's an interesting potential feature
  - Unless sponsored, most enhancements stay in Sev 5 indefinitely

# Slurm 23.11 - November 2023

# Slurm 23.11

- Please see the "Slurm 23.02, 23.11, and Beyond" presentation from SC'23 for details
- https://www.schedmd.com/publications/

# Highlights

- New auth/slurm authentication scheme
- SlurmDBD overhaul provides vastly increased performance for addition/removal of users
- Changes to 'scontrol reconfigure' to make it safe
- Reservations support more granular (TRES) requests
- Debian packaging support

# Slurm 24.05 - May 2024

# Isolated Job Step Management

- Offload job step management from the Slurm controller
  - Optional, and currently disabled by default
  - Step management moves to the "extern" step on the first allocated node
- Allows for **vastly** improved concurrency for systems with heavy step usage
  - Current reality
    - Step launch throughput is entangled with job launch throughput
    - Most Slurm installations peak at around 300 (jobs and steps) per second
  - Long-term goal is to be able to sustain 500 jobs per second
    - With each job - independently - able to launch 1000 steps per second

# Isolated Job Step Management

- Change in architecture provides flexibility to adopt other workflow tooling within the job
  - Currently, slurmctld is very performance sensitive
    - Adopting "heavier" tooling, such as workload management stacks, is risky
  - Moving the step management to the compute node isolates any performance or stability issues to the individual job
    - Looking at support for CWL, or similar workflow tooling, within the step management layer

# Isolated Job Step Management

- To enable, jobs use the --stepmgr option at submission time
  - Or can be managed through job_submit.lua
    - Could enable on a partition-by-partition or account-by-account basis
    - Allows for a phased roll-out
  - Or globally with SlurmctldParameters=step_mgr_enable
- We expect to make this the default behavior within a few releases
  - And eventually the only supported behavior

# Isolated Job Step Management

- Accounting
  - Step accounting details are proxied by slurmctld
    - Then forwarded to SlurmDBD
    - Forwarding does not require access to core locks (job write)
  - If a site doesn't require this data, setting AccountingStorageEnforce=nosteps can significantly reduce load on the SlurmDBD
- Display
  - "squeue -s" requires a specific JobID to query
    - Needs to connect to slurmctld to find the node that is managing the steps

# Isolated Job Step Management - Cray Specifics

- Cray / Slingshot systems have a few caveats related to their plugins
    - Depending on site-configuration, --stepmgr may not be available
        - But we want it to be, and are working to ensure SchedMD-recommended plugin configurations are supported
- mpi/cray_shasta plugin
    - Cray MPI requires applications to use the same TCP port across all nodes
    - Uses "MpiParams=ports=xxxx-yyyy" to centrally schedule TCP ports in slurmctld
    - For single-node jobs this isn't an issue
    - But needs to happen on a step-by-step basis for multi-node jobs
        - Initial 24.05.0 release might not support --stepmgr for these jobs

# Isolated Job Step Management - Cray Specifics

- switch/hpe_slingshot
    - Per-job VNI assignment is fine
    - Per-step VNI assignment does not work with --stepmgr
        - Assumes centralized management of VNIs
        - Fabric manager integration also problematic, as it assumes REST calls originate at the slurmctld
    - "Instant On" capability assumes REST calls originate from slurmctld
        - HPE does not recommend enabling this mode
        - SchedMD looking at removing this entirely in a future release
            - Not recommended due to significant performance impact to slurmctld
            - Zero customer systems using this, as far as we know

# Federation

- Allow client command operation when SlurmDBD is unavailable
  - Commands will instead receive remote cluster information from the local slurmctld

# New QOS Limits

- MaxTRESRunMinsPerAccount
  - Automatically establishes a "bucket" (TRESRunMin) of resources for each Slurm account running under the QOS
- MaxTRESRunMinsPerUser
  - Automatically establishes a "bucket" (TRESRunMin) of resources for each user running under the QOS

# Relative QOS limits

- New QOS Flag - "Relative" - treats limits as a percentage of the cluster's total resources
  - Or an relative to the individual partition, if used as a PartitionQOS
  - Instead of an absolute resource count

# Reservations

- New Reservation flag - USER_DELETE - allows users permitted to run under the reservation to delete it
    - More granular form of the existing SlurmctldParameters=user_resv_delete flag

# Node Features Changes Without Reboot

- New option - Flags=rebootless - in helpers.conf
    - Indicates Slurm does not need to reboot the node to effect changes
    - Useful if you're using these features to manage GPU states

# Cloud Node Management

- SlurmctldParameters=max_powered_nodes=N serves as a hard limit on all powered-up nodes
- Allow NodeSet names to be used in SuspendExcNodes.
- SuspendExcNodes=<nodes>:N prevents N nodes out of <nodes> from being suspended

# Accounting for StdIn, StdErr, StdOut

- Now stored in SlurmDBD and available in accounting data
- Values are stored verbatim, without any expansion patterns substituted
- New --expand-patterns option attempts to replace these expansion patterns
  - Note that some patterns - %t %n %N - result in multiple output files
    - Only a single result will be returned, corresponding to the first output file

# TLS Encryption

- Available between slurmctld and slurmdbd
  - Also between slurmctld when running in a Federation
- Requires Amazon's s2n-tls library, and a pre-shared certificate
- First - of three - phases to encrypt all Slurm traffic

# switch/nvidia_imex

- Manages NVIDIA IMEX channels on a per-job basis
    - IMEX channels provide isolation on NVIDIA interconnects

# ReservedCoresPerGPU

- Dedicate cores on nodes exclusively to GPU work
    - Cores only assigned to the job when the corresponding GPU is allocated
    - Allows for CPU-based workloads to better run alongside other work on GPU nodes
        - Avoid starving the GPU workloads and risk idling the ($expensive) GPUs

# auth/slurm - Key rotation

- By default, auth/slurm uses slurm.key as the shared-secret within the cluster
- New work allows for slurm.jwks to be provided as an alternative
    - Multiple keys can be defined
        - Default key can be specified, alongside an expiration
        - Allows for seamless key rotation within the cluster
    - Token are still HS256, shared-secrets must be available throughout the Slurm cluster
        - Future work may add support for RS256 tokens, which would allow for keys to be limited in scope
            - E.g., "cloud-bursted" parts of the cluster may use one private key, and the corresponding public key revoked on the rest of the cluster in response to a security incident

# auth/slurm - SELinux support

- Capture and send client commands' SELinux context
  - Embed and sign as part of the auth token
  - Closes the awkward integration issue when using MUNGE + SELinux support
    - Previously, a site-provided job_submit plugin would need to validate the context, as there is no way to securely transmit it as part of a job submission

# topology/block

- Plugin is designed for systems where optimizing topology for individual jobs is considered paramount
  - This plugin may introduce significant drops in cluster utilization, with the understanding that the performance of individual jobs is likely to greatly increase, and thus the total system throughput in terms of jobs completed will be increased overall
  - Nvidia talked in their presentation this morning about the base unit of compute moving to the datacenter level
    - This plugin aligns with that vision - optimization of the system throughput is the goal, even at the expense of individual node utilization
- Significant changes to the configuration:
  - BlockLevels have been removed
  - BlockSizes must be explicitly configured instead

# topology/block

- Added --segment job option
  - Represents the underlying "segment" size of the compute processes
    - E.g., compute tasks may naturally decode to 12-node segments, and then run as some multiple of these segments
    - Topology plugin will respect this size when laying out the job
- Added --exclusive=topo option
  - Will disallow other jobs from sharing the blocks allocated to the job
  - Intended only for workloads that are very sensitive to communication jitter within the block
    - Will obviously cause even lower system utilization rates, and should be used with care

# Job Confinement for Prolog/Epilog and SPANK

- New SlurmdParameters=contain_spank forces SPANK hooks to run within job_container/tmpfs namespace
  - spank_user_init(), spank_task_post_fork(), and spank_task_exit()
  - Mainly intended for use by "pyxis" SPANK plugin
- Add PrologFlags=RunInJob option to launch Prolog/Epilog within the job's cgroup hierarchy
  - Avoid mistakes with these scripts from, e.g., modifying GPUs not assigned to the job

# Cleanup

- Removed support for Cray XC systems (the "cray_aries" plugins)
  - For customers still running these machines, support for Slurm 23.11 has been extended through your HPE support EOL
- Refactored the "topology" plugin interfaces
  - Plugins now actually integrate their own node selection code
- Removed openapi/dbv0.0.38 and openapi/v0.0.38 plugins
- Tagged openapi/dbv0.0.39 and openapi/v0.0.39 plugins as deprecated

# Random

- Added "elevenses" as an additional time specification
    - Other shorthands include "midnight", "noon", "fika", and "teatime"

# … and Beyond

# Slurm 24.11 - November 2024

# Expanded Backwards Compatibility

- Start supporting upgrades and mixed-version operation with three prior major releases
    - Instead of only the two prior major releases

# "Fall-back" QOS values

- Allow user to specify, in priority order, a number of QOS values
- Scheduler will iterate through the fall-back QOS options until a suitable one is found, or all options have been exhausted

# TLS Encryption

- Expand encryption support to slurmctld to/from slurmd
  - This may require an external certificate manager / certificate authority
- Second – of three – phases to encrypt all Slurm traffic

# Backfill tweaks

- topology/block can lead to throughput issues under high fragmentation
    - Backfill scheduler is "conservative" in existing implementation
        - Will never stall the launch of a higher priority job, will always plan for it to start ASAP, and only then plan other jobs around it
        - With the topology strictly enforced, fragmentation can lead to considerable delays… but launching large jobs on the first available fitting set of nodes may perpetuate high-level fragmentation
    - Exploring approaches to mitigate these issues, potentially develop a heuristic that is willing to delay larger job launches in favor of reduced fragmentation, and higher utilization rates

# Slurm 25.05 - May 2025

# TLS Encryption

- Expand encryption support to everything
- Final phase to encrypt all Slurm traffic
  - Including client commands, srun I/O forwarding

# Upcoming Events

Tim Wickberg
*CTO*

# SLUG'24
# September 2024

UNIVERSITY
OF OSLO

# Upcoming Events

- ISC'24
- SLUG'24 - September 12-13th
  - Call for presentations is open through May 31st
  - Tickets - https://slug24.splashthat.com/
- SC'24
  - Booth 2809
  - Slurm Community Birds-of-a-Feather session is submitted

# Thank You

SCHEDMD

The Slurm Company