Slurm Community BoF

Tim Wickberg, SchedMD
Danny Auble, SchedMD
Slurm Community
Birds-of-a-Feather
Agenda
Agenda

● Audience Survey
● Development Cycle Overview
● Slurm 23.02 Release
● Slurm 23.11 Release
● Future Releases
● Open Community Forum
Slides

- Slides from today - and from the booth talks - will be posted online shortly:
  - [https://slurm.schedmd.com/publications.html](https://slurm.schedmd.com/publications.html)
- The BoFs are also being livestreamed/recorded by SC
Questions?

- Feel free to ask throughout
- But - please use one of the microphones
  - Allows the folks in the room, as well as folks tuning in online, to hear your questions
  - We will respectfully decline questions if they're not asked through a mic
Two random rants
What is "Slurm"?

- Slurm is Slurm
  - Capital "S". Lowercase "lurm"
- Slurm is no longer "SLURM"
  - Historically, all-caps was an acronym...
    - ... but we moved away from it in 2012
      - And have been struggling to convey the switch
        - Please humor our branding efforts
          - Writing it in all-caps sounds like you're shouting :(
Versions

● "Slurm 23.11" or "Slurm 23.02", "22.05", "21.08"...
● There is no such thing as "Slurm 23"...
  ○ There are two major releases this year - 23.11 and 23.02
    ■ There are considerable differences between them
       ● Especially as 23.11 hasn't been released yet :(
... plus one request
Academic Citations

- We have a new peer-reviewed article.
  - Presented at JSSPP’23
    - Even featured as a keynote
    - 20 years after the first paper in JSSPP’03
- Please cite this new article instead:


- Citation, DOI link (which has the BibTeX): https://slurm.schedmd.com/faq.html#cite
Audience Survey
Who here is running Slurm?

- And who isn't?
  - I'll caution this BoF is **not** an introduction, but assumes a certain degree of familiarity
    - Please see the publication archive for introduction presentations
      - Or stop by the booth if you have any questions
What version are you running?

- 23.11
- 23.02
- 22.05
- 21.08
  - ... missing security fixes
- ... even older?
  - ... missing even more security fixes
How do you manage your installation?

- Build RPMs from official releases
- Build DEBs from official releases
- "make install" into a central directory
  - "make install" as part of the node image
- Spack
- RPMs from EPEL
  - Note: these are not recommended, and are not officially supported
- DEBs from Debian/Ubuntu
  - Note: these are not recommended, and are not officially supported
- ... other?
External Libraries

- Do you build Slurm with...
  - PMIx support
  - Nvidia
  - AMD (rsmi)
  - Intel (oneAPI)
  - HDF5
    - Does anyone here use acct_gather_profile/hdf5?
Feature Adoption

- Who here is running “configless”?
  - Who places configs on a central filesystem?
  - Manages them through Ansible/Chef/Salt/Puppet?
- Is anyone using the Perl API for their own scripts?
  - Not counting the openlava / torque wrapper scripts
- Is anyone brave enough to develop against libslurm directly?
- Who is developing against the REST API?
- Has anyone started using the native container (--container) support?
Do you have support?

- Or are you self-supported?
Slurm 23.02, 23.11, and Beyond

Tim Wickberg
Chief Technology Officer
Development Cycle
Release Cycle

- Major releases are currently made every nine months
- Version is the two digit year, two digit month:
  - 23.02 - February 2023
  - 23.11 - November 2023
  - 24.08 - August 2024
- Major releases are supported for 18 months
  - Currently: 22.05 and 23.02
  - After November: 23.02 and 23.11
- Maintenance releases are made roughly monthly
  - Usually only for the most recent major release
    - One main exception - security releases will be made for all supported major releases
Development Process

- Most larger work is handled through sponsored projects
  - SchedMD support only covers maintenance
- Some projects - those of wider community interest - may be handled internally on a best-effort basis
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 will stay in Sev 5 indefinitely
New scrun command

- Proxy to launch OCI-compliant container images on the cluster
- Slurm's version of crun / runc
- Refer to the "Containers in Slurm" talk from SLUG'23 for more details
  - [https://slurm.schedmd.com/publications.html](https://slurm.schedmd.com/publications.html)
New --tres-per-task option

- Allow jobs to be modeled as a number of tasks, with all appropriate resource types scaled directly by the number of tasks requested
  - Task can request licenses, GRES, CPUs, memory
  - Note - can't automatically propagate to srun within a batch script in 23.02
    - Can starting in 23.11
AllowAccounts - automatic recursion

- Update the "AllowAccounts" access control to automatically extend access to all child accounts
License Preemption

- When running with preemption, license usage is not considered by default, and jobs will not be preempted to free up licenses.
- This is an issue especially when using licenses to represent cluster-wide resources, as they won't be reclaimed to allow higher-priority work to preempt.
- Enable with PreemptParameters=reclaim_licenses.
Licenses

- https://slurm.schedmd.com/licenses.html#remote_licenses
- Remote licenses can now be set with "flags=absolute"
  - Means the per-cluster assignments are by explicit license count, instead of percent
  - slurmdbd.conf option of AllResourcesAbsolute=yes to enable this by default
- New "LastConsumed" value, designed to be frequently updated with current license server utilization values
  - Propagated to slurmctld automatically
  - Controller automatically factors that current status in when deciding how many licenses can be used for new jobs

LicenseName=foobar44@licsrv42
  Total=0 Used=0 Free=0 Reserved=0 Remote=yes
  LastConsumed=0 LastDeficit=0 LastUpdate=2023-02-02T18:20:57
Cloud nodes enhancements

- Pass list of requested features to ResumeProgram
- Reset active features on CLOUD nodes
- Allow for Node Weight to be considered on CLOUD nodes
- New flag to automatically power down “Exclusive” nodes once jobs are completed
Reservation Enhancements

- Add a Comment field to reservations
- Show active reservations on each node in 'scontrol show node'
- Support node addition and removal from a reservation through scontrol with += and -= on the node list
Accounting Tweaks

- New FailedNode field
  - Set for jobs that have been terminated due to a node failure
  - Help triage hardware issues
New job completion plugin

- New jobcomp/kafka plugin
Performance Improvements

- **Halved** the number of MUNGE interactions by slurmctld
Flexible Node Counts

- In addition to min and max node counts, allows the user to specify acceptable node counts
  - E.g., --nodes=20,40,80,160
- Also allows for a step function specification
  - E.g., --nodes=10-30:5 is equivalent to --nodes=10,15,20,25,30
"Explicit" GRES Flag

- Currently, all GRES are allocated to a job when --exclusive is set
- New GRES Flag "Explicit" avoids allocating that GRES by default for --exclusive jobs
  - Will only allocate it when explicitly requested
Debug option handling

- New `scontrol setdebug <level> nodes=node[1-10]` sub-command
  - Allows dynamic changes to debug level on specified nodes
- `scontrol setdebugflags flag,flag2,flag3 nodes=node[1-10]` also added
Greatly extended support for JSON and YAML output from user commands
Now allows many command filtering options to be used as well
RPC Rate Limiting

- New optional per-user RPC rate limiting mechanism
  - Backs off client commands if they're being too chatty
  - Sends new dedicated response code telling the command to sleep for a second before retrying, rather than crashing the user command
  - Can avoid having 'while true; do squeue; done' overload slurmctld
SlurmDBD Overhaul

- The "right-left tree" data structure was used to represent the association hierarchy in a flat row-oriented fashion
  - Unfortunately, insertion and deletion is $O(n)$
    - And can trigger $O(n)$ row updates in the database
    - Which cause $O(n)$ updates to slurmctld
  - New "lineage" approach significantly improves performance
    - Especially when heavily scripting against external accounting systems
    - Must move slurmctld to 23.11 alongside slurmdbd to see benefits
      - Otherwise slurmdbd must maintain both structures for backwards-compatibility
srun --external-launcher

- Common MPI stacks use `srun` internally to launch their own launch processes
  - `orted`, `hydra`, ...
- Newer `sbatch` options - such as `--tres-per-task` - cannot be inherited by `srun` without causing layout issues for `mpirun/mpiexec`
- New internal `--external-launcher` flag is automatically propagated back to `srun` through `mpirun/mpiexec`, and indicates `srun` is being used to bootstrap an external MPI stack
  - Provides all resources on each node to process, does not try to interpret other Slurm layout options
- Automatically injects four environment variables into job, all set to "--external-launcher":
  - `OMPI_MCA_plm_slurm_args`
  - `PRTE_MCA_plm_slurm_args`
  - `HYDRA_LAUNCHER_EXTRA_ARGS`
  - `I_MPI_HYDRA_BOOTSTRAP_EXEC_EXTRA_ARGS`
Fixing 'scontrol reconfigure'

- In 23.11, 'scontrol reconfigure', SIGHUP, and restarting slurmd processes all provide equivalent changes.
- Previously, certain changes cannot take effect within the process through 'scontrol reconfigure', and required an explicit restart of the daemon.
  - Which changes could be safely applied through "scontrol reconfigure" were... intuitive... and mostly undocumented.
- Note - need to use newer systemd service files to take advantage.
  - They now use a new --systemd option to slurmd / slurmd.
    - And switch to Type=notify instead of Type=simple to accommodate the new process model that is required.
Fixing 'scontrol reconfigure'

- "scontrol reconfigure" can now catch configuration mistakes, and continue execution on the prior configuration instead of fatal()'ing
  - scontrol client command also receives an error code
    - Rather than timing out on error if the reconfigure failed and the slurmctld stopped
- Reconfigure now allows for almost any (supported) configuration changes to take place
  - Notable exceptions:
    - Can't change between select/cons_tres and select/linear
      - Requires complete shutdown and restart as the queue will be lost
    - Won't change network listening ports
      - Avoids various communication problems if the ports were closed and reopened constantly
        - But prevents changes to SlurmctldPort / SlurmdPort from taking immediate effect
Change SlurmctldHost settings without breaking running jobs

- In Slurm 23.02 and older, changes to SlurmctldHost are not possible with jobs running on the system
  - The slurmstepd processes load their configuration when the step is launched, and have no mechanism permitting updates
  - Once a job/step completes, the slurmstepd needs to communicate directly with slurmctld... if you change the IP address of the SlurmctldHost this will fail, and running jobs will never complete
  - Change allows for slurmstepd processes to be pushed updates by slurmd automatically
Additional HA Sanity Checks

- The "Field Notes" presentation mentioned a hypothetical issue that can happen if the StateSaveLocation is not mounted on your backup controller:
  - Backup asserts control, has no job state available, and will start killing jobs off when the slurmd processes on the compute node re-register.
- Backup will now check on the heartbeat file, refuse to take control if it is missing:
  - Primary controller frequently updates a timestamp in the heartbeat file:
    - Used to prevent backups from asserting control too aggressively in a network partition event.
  - Protects against misconfiguration of StateSaveLocation, as well as an array of potential filesystem problems.
New auth/slurm and cred/slurm plugins

- New internal authentication and job credential plugins
  - Alternative to MUNGE
  - Builds off existing capabilities - unix socket authentication through SO_PEERCRED (used by slurmstepd to authenticate RPCs), plus auth/jwt authentication plugin
- Simple HMAC scheme (SHA-256) built off JWT
  - Separate from existing auth/jwt plugin
  - Will require a shared key that is shared throughout the cluster
    - /etc/slurm/slurm.key
    - Similar security posture to MUNGE
- Client commands use a local socket, automatically managed by slurmctld / slurmdb / slurmd, or new sackd daemon on the login node
- Will allow for future extension and flexibility...
LDAP-less control plane

- Support running the slurmd without LDAP
  - Optional capability enabled through auth/slurm's credential format extensibility
  - Username, uid, gid, groups will be captured alongside the job submission
  - auth/slurm permits the login node to securely provide these details, which auth/munge cannot due to protocol limitations
  - Set AuthInfo=use_client_ids in slurm.conf and slurmd.conf to enable
New login node process - sackd

- For sites running auth/slurm, a new daemon - sackd - provides authentication for client commands
- This daemon can also integrate into a "configless" environment, and manage the locally cached set of configuration files for the login node
  - Updates will be received automatically through "scontrol reconfigure"
    - Similar mechanism already exists to update slurmd processes
TRES Reservations

- Allow for TRES-oriented reservations
  - E.g., reserve 200 GPUs alongside 800 CPUs
- `scontrol create reservation=test start=now duration=5 account=foo tres=gres/gpu=1`
- Treated similarly to a job, and will use DefCPUPerGPU when constructing the reservation
"Extra" Constraints

- Set of key=value pairs, with the values provided by site-specific scripts
  - Can be integers, floats, or string types
  - Values intended to be refreshed periodically
    - Future work may build this into slurmd
    - For 23.11, sites are expected to use 'cron' to push periodic updates through
      'scontrol update nodename=foo extra=<updated payload>"

- New job submission flag, --extra, to allow users to filter the cluster nodes
  - Similar, but separate, from existing feature/constraint syntax

- Loosely functionally equivalent to LSF's ELIM feature
  - Not necessarily recommended for most sites
    - Hands a lot of responsibility for scheduling decisions to the end-user, and is
      much slower as each node has to be constantly and individually reassessed for
      suitability

- SchedulerParameters=extra_constraints to enable
Relative QOS limits

- Flag allows QOS to be specified as a percentage of the cluster's total resources
  - Or an individual partition, if used as a PartitionQOS
Debian Packaging Support

- Providing official Debian / Ubuntu package support
  - Packages will be under a common slurm-smd-* prefix
    - Avoids conflicts with the existing mix of slurm-wlm / slurm-llnl packages
      - (Which SchedMD does not support or recommend)
  - Package layout is roughly aligned with the RPM layout from slurm.spec
    - And not the existing unofficial Slurm debian packages
OpenAPI, --json/--yaml option updates

- Significant refactoring of the OpenAPI plugin code now allows for most --json/--yaml command-line options to use their filtering options
- New optional arguments allow CLI tools to provide output through a specific OpenAPI plugin version
  - Defaults to current OpenAPI schema
- See Nate's REST presentation for additional details
topology/block

- New topology/block plugin - and associated plumbing - that forcibly respects a "block" oriented topology on certain new hardware platforms
- Ensures jobs are always placed on optimal set of switches, rather than what is currently available
  - Existing topology/switch plugin is best-effort, and will launch jobs on available resources immediately rather than wait indefinitely for a better fit
  - Downside: system utilization can collapse if not kept in check
Soft Time Limits

- Allow a job to provide the **expected** run time in addition to the traditional hard time limit
  - Use this value for backfill planning, rather than the usual time limit
  - Increases system utilization, especially for systems with a few large jobs and a constant flow of higher-throughput
- Not recommended for most general-access systems, as users would be incentivized to submit all work with a very short soft limit to get it running immediately
  - Designed for more "cooperative" environments with a smaller user base
  - Optional, must be explicitly configured to enable
    - SchedulerParameters=time_min_as_soft_limit
Cloud / Dynamic Nodes

- Slurm's configuration files don't have network details for the dynamic nodes
  - But commands such as srun and sdiag need to communicate directly with those nodes
  - Initial dynamic node support relied on flattening all communication by disabling fanout, and passing network details through environment variables and other means
Cloud / Dynamic Nodes

- Network changes in 23.11
  - The `cloud_reg_addrs` option has been removed
    - Option told `slurmctld` to automatically update it's address cache with the inbound IP address when `slurmd` registered
    - Now the default for behavior for cloud / dynamic / dynamic_future nodes
  - `CommunicationParameters=NoAddrCache` option removed
    - No longer needed that `cloud_reg_addrs` is the default

- Message Fanout
  - Fanout now works with cloud and dynamic nodes
  - Passes node addresses through dynamic tree automatically
  - Allows offload of internal bookkeeping operations (node ping, reconfigure) to the `slurmd` processes again, reduces network load on `slurmctld`
Cloud / Dynamic Nodes

- Revamped networking - "Alias Addresses"
  - Client commands get alias addresses automatically through appropriate RPCs
    - Or through new dedicated RPC
  - Clients don’t rely on older "alias_list" approach now
    - Remove SLURM_NODE_ALIASES
    - For large-scale cloud node launch, this prevents the job environment from exploding, as that variable could be massive in practice
Cloud / Dynamic Nodes

- TopologyParam=RoutePart
  - Use Partitions as the boundary for message fan-out
    - Acts independently of the topology/tree plugin, which can still be used for scheduling if desired
  - Useful for multi-zone / multi-network clusters to limit potential failure propagation
Cloud / Dynamic Nodes

- Cloud InstanceId and InstanceType
  - Visible through sacctmgr show instances
  - Useful to track what class of cloud hardware was used in the accounting database
SelectTypeParameters=LL_SHARED_GRES

- Similar to CR_LLNN... but favor nodes with least-loaded shared GRES
  - Shared GRES types are MPS or Shard
  - CR_LLNN only considers CPU occupancy
  - This allows you to steer jobs to nodes have the least occupied GPUs instead
Shards

- Shards allow for GRES (e.g., GPUs) to be cooperatively split
  - See https://slurm.schedmd.com/gres.html#Sharding for further details
  - Similar to NVIDIA’s MPS, but without any specific hardware cooperation
  - No enforcement of cooperation - not recommended for most systems

- Enhancements focus on allowing a job to have shards across multiple GPUs within a single node, as well as enabling --tres-per-task to work seamlessly with these shards
... and Beyond
Slurm 24.08
ReservedCoresPerGPU

- Dedicate cores on node to GPU work
  - Cores only assigned if the corresponding GPU has been allocated to the job
  - Allows for CPU-based workloads to better overlap into GPU nodes, without threatening to starve the GPU workloads and risk idling the (expensive) GPUs
- Currently, the same use case can be partially covered by using the MaxCPUsPerNode setting on a Partition
  - But that doesn’t easily scale with a heterogeneous mix of nodes, and requires splitting work across multiple partitions
Node Features

- Allow for Node Features changes without rebooting the node
  - The node_features plugin interface was originally designed for CPU NUMA/memory layout changes for the Intel KNL chips, and assumed any changes would require the node to reboot to take effect
  - But most, e.g., GPU mode changes can be done live
    - Inconvenient to need a node reboot for all changes
  - Currently required by the node_features stack, although can be faked by using the "-b" option to slurmd with some careful scripting in your RebootProgram
Further auth/slurm extensions

- Capture and send client commands' SELinux context as part of the auth token
  - Closes the awkward integration hole when using MUNGE where the client requested context needs to be validated by your job_submit plugin somehow
Independent Step Scheduling

- Allow the step scheduler to be run on a compute node instead of inside the slurmctld process
  - Greater throughput for the job, less RPC load on the slurmctld
    - Win-win in many respects
  - In future, should allow greater flexibility and expanded capabilities without detriment to system throughput or responsiveness
    - E.g., potential to add native support for workflow languages like CWL
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
... and even further beyond*
Scope Limit for MPI Plugins

- Refactor the mpi plugin interface to run most hooks as the user, rather than uid 0
- CVE-2023-41915 implies we cannot always trust the MPI libraries we build against...
Standalone Step Management Layer

- Build on the isolation of the step management code (see Brian's talk from earlier)
  - Potentially allow a lightweight independent step management process to run underneath a Slurm (or other WLM) allocation
- Extend the step management layer with support for CWL or other workflow standards
Converged Computing

- Allow for Slurm to cooperatively schedule alongside other cloud orchestration layers
  - Such as Kubernetes
- Extend official support for projects like CoreWeave's SUNK
- CoreWeave is working to open-source SUNK in early 2024
SUNK Implementation Overview

Services containerized in Kubernetes
SUNK
Implementation Overview

Services **containerized in Kubernetes**
Slurm components as **Pods**
Configuration as **ConfigMaps** and **Secrets**
**Nodesets** maintaining compute
**Slurm Syncer** reconciling state
Staying consistent with the **Operators**
Schedule from both sides
Expose prometheus **Metrics**
Upcoming Events

Tim Wickberg
CTO
Upcoming Events

- SLUG’24 Pre-Sale tickets are available for SLUG’23 attendees
  - Discounted rate available through January 12th
  - [https://slug24.splashthat.com/](https://slug24.splashthat.com/)
Open Forum

Tim Wickberg
CTO