Slurm License Management

Slurm 2013 User Group  Bill Brophy, Bull
Background

• **Software and the associated Licenses are EXPENSIVE**
  Nearly $160 billion will be spent by American companies on software purchases this year
  Companies overspend by nearly 30% on software license agreements and maintenance (embedded licenses, etc)

• **Software vendors are concerned about lost revenue**
  Commercial value of pirated & overused software rose to $63.4 billion in 2011
  Increasing the number & frequency of software audits
Background

• License Managers were developed to “guard” ISVs resources
  Prevent usage of unlicensed software
  Prevent overuse of software
  Manage licenses from various vendors

• Problems with integrating Resource Managers with License Managers
  License Managers do not provide an open interface
  Solutions often introduce Race Conditions
  Often involve a great deal of overhead
  ISVs, the License Manager’s clients, prefer to sell more licenses

• FlexNet Publisher (formerly Flexlm) is the major license manager
Current Slurm License Management

• License information parameter in slurm.conf
  License names can optionally be followed by a colon and count
Multiple license names should be comma separated
  e.g. Licenses=Intel_Compiler:4,TotalView

• salloc, srun & sbatch support for licenses
  -L, --licenses=<license>
  e.g. --licenses=Intel_Compiler:2,TotalView

• Reservations can be used to restrict license usage
  Licenses=<license>
  LICENSE_ONLY flag

• No integration with License Managers
  Acceptable if licenses are uses exclusively within Slurm
  Potential problems & inefficiency if usage external to Slurm
Development Plan

• Bull has initiated a Slurm License Management project
  Based on consultation with SchedMD
  Input from the development community welcomed

• Project will consist of multiple phase
  First phase will introduce new License Structures
  Second phase will integrate Slurm with License Manager(s)
    e.g. FlexNet Publisher
  Hongjia Cao has begun work on a plugin
Phase 1

- Two new license structures are defined
  - System license structure
  - Cluster license structure
- Include structures in database
- Lays the groundwork for including in Associations
- Populate new structures using sacctmgr interface
- slurmctld notified of license changes
- slurmctld to use existing structures
Phase 2

• Integration of Slurm with License Manager
  Initially only FlexNet Publisher
• Still in discussion stage
• Involves considering the interface with the licenses manager
• Providing the communication protocols between Slurm and the license manager
• Solution must be efficient & scalable
License Acquisition

1 Request Vendor Daemon Info

2 License Manager returns information

3 Request for License

4 Grant or Deny request
Slurm in License Acquisition

- License Manager
- Vendor Daemon
- Slurm
- Application
Slurm License Management Issues

- License request occurs AFTER Slurm has started the job
- Slurm has no knowledge of non-Slurm license users
- License Usage Counts are very Dynamic
- License Managers cater to Software Vendors (primary focus is license limit enforcement)

  FlexNet has a command called lmstat to retrieve information, which can be very slow if FlexNet is handling many applications.

  A possible alternative could be parsing the FlexNet log file (lmgr).
Possible Solution

- LSF Network Floating License Management
- Use an external program (ELIM) to obtain the number of licenses currently available
  Configure an external load index containing the number of free licenses on each host
- ELIM periodically informs LSF of the number of available licenses
- Configure a dedicated queue to run jobs requiring a floating software license
- Queue definitions REQUEUE_EXIT_VALUES parameter related to license denial codes
- For each job in the queue, LSF reserves a software license before dispatching a job, and releases the license when the job finishes.
- A batch job may fail to allocate a license due to an interactive job (race condition)
- If a job exits with one of the values in the REQUEUE_EXIT_VALUES, LSF will requeue the job.

David Bigagli was one of the designers & developers of LSF the License Scheduler

http://www.ccs.miami.edu/hpc/lsf/7.0.6/admin/licensemgt.html
Supplemental Information

• Following are several slides which give brief descriptions of ideas for integrating a resource manager, such as Slurm, with a license manager.

• Several of these ideas came from the Slurm developer’s forum.

• Other ideas were located in web searches on this topic.

• Additional suggestions welcome:

  Bill Brophy <bill.brophy@bull.com>
  Danny Auble <da@schedmd.com>
  David Bigagli <david@schedmd.com>
Approach 1

- Gary Brown Reservation & Commit model

- Slurm "reserves" licenses through external license manager (FlexNet)

- The running job actually "checks out" the reserved licenses

- Issues
  Race condition between running job checkout & external job checkout
  Requires new model for license managers
  Changes required to ALL Independent Software Vendors (ISVs) software
Approach 2

• Mark Olsen method for the Gridengine

• Sophisticated Perl script observes license manager using lmstat

• A load sensor adjusts “complex values” (licenses available to the system) for jobs external jobs using licenses

• Resource manager calculates “internal count”

• “Available complex values” = “complex values” – “internal count”

• Issues
  Delay in reports
  Race Conditions between external & internal license allocation

http://gridengine.info/files/Mark_Olesen-HowTo-Licenses-n1ge.html
Approach 3

- Hongjia Cao Approach

- Modify the vendor option file
  Reserve the number of licenses (features, in term of FlexNet) configured in Slurm
  Use a randomly generated project name

- On job resource allocation reserve licenses in resource manager
  Set the environment LM_PROJECT to the project name to checkout licenses

- On job resource deallocation, the licenses reserved to project of the job are taken back
  Reservation in vendor option file is deleted
  Imreread executed

- Issues
  Race condition on vendor option file updates
  Scalability
  Possible scheduling performance impact