MCS Plugin
Multi Category Security
MCS Plugin
Agenda

Introduction

Implementation

Planned features
MCS Plugin

Introduction
Motivations

- Ensure populations confinement
  - Job confinement: no sharing of nodes for jobs from different populations of users
  - Information confinement: users can only see jobs/nodes of their population
  - A population is associated to a category. The term MCS comes from SELinux: MCS is an enhancement to SELinux, and allows users to label files with categories. A lot of informations can be a category: users, uid, UNIX groups...
Existing options for job confinement

- Exclusive nodes for sbatch/srun/salloc commands (-x option)
  - No risk for a job to share a node with a user of another population
  - But waste of resources if nodes are not used entirely

- Exclusive nodes per user for sbatch/srun/salloc commands
  (--exclusive=user)
  - No risk to share a node with another user, but can't share nodes between users of the same population
  - But waste of resources if nodes are not used entirely
Existing options for information confinement

- **Slurm.conf option: privatedata**
  - privatedata=jobs
    - Prevents users from viewing jobs or job steps belonging to other users.
  - privatedata=nodes
    - Prevents users from viewing node state information.
Goals

- Add a generic/extensible way to include a new logic for confinement.

  - The use of the notion of plugin in slurm was an evidence.

  - With a plugin, possibility to have many levels of logic:
    - 1 to 0: users have no MCS-label: only one population; identical to no plugin.
    - 1 to 1: a user is a population: A plugin for an equivalence between user and population (user name or uid for example). The MCS-label is deducted.
    - N to 1: a user has an unique MCS-label and a MCS-label has many users. For example: primary group. The MCS-label is deducted.
    - N to N: a user has a choice between different MCS-label and a MCS-label is associated to many users. There is a set of populations and every user could be in more than one population. Examples: a slurm account, a unix secondary group. This plugin needs an algorithm to choose the MCS-label if none is requested.
**Goals**

**Overview**:

- **1 to 0**
  - Users → MCS-label=N/A  
    No choice

- **1 to 1**
  - User1 → MCS-label=mcs1  
  - User2 → MCS-label=mcs2  
    No choice

- **1 to N**
  - User1 → MCS-label=mcs1
  - User2
  - User3 → MCS-label=mcs2  
    No choice
  - User4

- **N to N**
  - User1 → MCS-label=mcs1
  - User2
  - User3 → MCS-label=mcs2  
    Choices...
Our specific goal

- Nodes confinement with unix groups:
  - For a user in groupE and groupF:
    - If --mcs-label is specified, only empty nodes or nodes already tagged with this MCS-label are filtered.
    - If --mcs-label is not specified, only empty nodes or nodes already tagged with the default MCS-label are filtered (default is the first found in the list of possible MCS-labels).
  - For a user in groupE: only empty nodes or nodes already tagged with groupE MCS-label are filtered.
  - ...

![Diagram of sets E, EF, F, EFG, EG, FG, and G, illustrating the relationships and intersections between user groups.](image)
Our specific goal

- Information confinement: `squeue → shows only jobs with authorized MCS-label`
  - For a user in groupE and groupF: `squeue -O jobid,username,mcslabel`
    - JOBID  USER  MCSLABEL
    - 1       user1  groupE
    - 2       user2  groupE
    - 3       user1  groupE
    - 4       user3  groupF
  - For a user in groupF: `squeue -O jobid,username,mcslabel`
    - JOBID  USER  MCSLABEL
    - 4       user3  groupF
MCS Plugin

Implementation
Configuration choices

- MCS-label is a category label for jobs and/or nodes

- MCS-label for jobs
  - MCS-label for jobs can be optional or mandatory (slurm.conf option)
  - Users can choose (if possible) their MCS-label for their jobs (in a closed list)

- MCS-label for nodes
  - The selection of nodes can be (or not) filtered on MCS-label depending on slurm.conf options.

- MCS-label of jobs is seen with sview/squeue

- MCS-label of nodes is seen with scontrol show nodes command

- Accordingly with privatedata, jobs and nodes informations can be filtered on the MCS-labels
New slurm options in slurm.conf

MCSPlugin

- 3 implementations mcs/none, mcs/user and mcs/group.
  - mcs/none: Default. No category associated to jobs.
  - mcs/user: Use user name as the category to associate jobs to. This option is equivalent to use --exclusive=user.
  - mcs/group: Use a user group as the category to associate jobs to. The list of available groups is defined in the mcs_plugin_parameters.
New slurm options in slurm.conf

- MCSPluginParameters is a string of the form:
  "[ondemand|enforced][,noselect|select,ondemandselect]
  [,privatedata]:[mcs_plugin_parameters]"

- [ondemand|enforced]: set MCS label on jobs on demand (with --msc-label=) or always

- [,noselect|select,ondemandselect]: select nodes with filter on MCS label: never, always or on demand (with --exclusive=mcs)

- [,privatedata]: accordingly with privatedata option:
  - if privatedata and privatedata=jobs: jobs informations are filtered based on their MCS labels
  - if privatedata and privatedata=nodes: nodes informations are filtered based on their MCS labels

The defaults are ondemand, ondemandselect and no privatedata.
New slurm options in slurm.conf

MCSPlugin is a string of the form:
"[ondemand|enforced][,noselect|select|ondemandselect]
[,privatedata]:[mcs_plugin_parameters]"

- [mcs_plugin_parameters]: Only mcs/group is currently supporting the mcs_plugin_parameters option. It can be used to specify the list of user groups (separated by |) that can be mapped to MCS labels by the mcs/group plugin.

- If no specific MCS label is requested (no --mcs-label option), the algorithm search the first group of the user in the groups list of mcs_plugin_parameters. If no valid group is found:
  - If ondemand is set, the job has no MCS-label,
  - If enforced is set, the job is failed.
# New slurm options in slurm.conf

<table>
<thead>
<tr>
<th>Nodes:</th>
<th>Jobs: On demand</th>
<th>Jobs: enforced</th>
</tr>
</thead>
<tbody>
<tr>
<td>No select</td>
<td>MCS-label is optional on jobs (option <code>--mcs-label</code>). No filter on nodes.</td>
<td>MCS-label is mandatory on jobs only. No filter on nodes even if option <code>--exclusive=mcs</code> is set.</td>
</tr>
<tr>
<td>select</td>
<td>MCS-label is optional on jobs (option <code>--mcs-label</code>). Filter on nodes only if MCS-label is set on job.</td>
<td>MCS-label is mandatory on jobs and nodes. Always filter on nodes.</td>
</tr>
<tr>
<td>ondemandselect</td>
<td>MCS-label is optional on jobs (option <code>--mcs-label</code>). Filter on nodes only if options <code>--exclusive=mcs</code> and <code>--mcs-label</code> are set.</td>
<td>MCS-label is mandatory on jobs only. Filter on nodes only if option <code>--exclusive=mcs</code> is set.</td>
</tr>
</tbody>
</table>
New slurm options in slurm.conf

Examples:

- MCSPlugin=mcs/none
- MCSPlugin=mcs/user
  - MCSParameters=enforced,select,privatedata
- MCSPlugin=mcs/user
  - MCSParameters=enforced,noselect
- MCSPlugin=mcs/group
  - MCSParameters=enforced,select,privatedata:groupA|groupB|groupC
- MCSPlugin=mcs/group
  - MCSParameters=ondemand,ondemandselect,privatedata:groupA|groupB|groupC
New options in salloc/sbatch/srun

- **--exclusive=mcs**
  - User can force the filter with this option (except if noselect mode)
  - With mcs/user and mcs/group

- **--mcs-label=groupD**
  - User can change default mcs-label
  - Only with mcs/group
  - GroupD must be in the list of user's group and in the list of possible MCS (in parameter mcs_plugin_parameters in slurm.conf)
New options in salloc/sbatch/srun

Examples

- `srun -n2 --exclusive=mcs a.out`
  - Use default MCS-label,
  - Selection of nodes is filtered on MCS-labels

- `srun -n2 --mcs-label=groupD --exclusive=mcs a.out`
  - Use specified valid MCS-label,
  - Selection of nodes is filtered on MCS-labels

- `srun -n2 --mcs-label=groupD a.out`
  - Use specified valid MCS-label,
  - Selection of nodes is not filtered on MCS-labels (if no select).
New options in salloc/sbatch/srun

- Examples with errors
  - Test to use a specific mcs-label with mcs/none plugin
    ```bash
    srun -n2 --mcs-label=foo  a.out
    ```
    - srun: error: --mcs-label=foo can't be used with mcs/none plugin
  
  - Test to use a bad specific mcs-label with mcs/group plugin
    ```bash
    srun -n2 --mcs-label=foo  a.out
    ```
    - srun: error: Failed to create job : invalid mcs-label : foo

  - Test to use default mcs-label with mcs/group plugin and user has no group in the list of possible mcs-labels
    ```bash
    srun -n2  a.out
    ```
    - srun: error: Failed to create job : no valid mcs-label found
New output option in squeue/sview

- Output option mcslabel in squeue
  
  Example: `squeue -O jobid,username,mcslabel,nodelist`

<table>
<thead>
<tr>
<th>JOBID</th>
<th>USER</th>
<th>MCSLABEL</th>
<th>NODELIST</th>
</tr>
</thead>
<tbody>
<tr>
<td>1300955</td>
<td>user1</td>
<td>groupA</td>
<td>node[1002-1005]</td>
</tr>
<tr>
<td>1300982</td>
<td>user2</td>
<td>groupB</td>
<td>node[1049,1051,1053]</td>
</tr>
<tr>
<td>1300996</td>
<td>user3</td>
<td>groupB</td>
<td>node[1001,1012-1013]</td>
</tr>
</tbody>
</table>

- Output option mcslabel in sview
New output in scontrol show conf

Example

```
scontrol show conf | grep -i mcs
MCSPlugin = mcs/none
MCSParameters = (null)
```
New output in scontrol show nodes

Example
scontrol show nodes
  NodeName=node0 Arch=x86_64 CoresPerSocket=4
  CPUAlloc=0 CPUErr=0 CPUTot=8 CPULoad=0.10
  Features=unshare,fs_scratch,fs_store
    Gres=(null)
  NodeAddr=node0 NodeHostName=node0 Version=15.08
  OS=Linux RealMemory=48000 AllocMem=0 FreeMem=43692 Sockets=2 Boards=1
  State=DOWN+DRAIN ThreadsPerCore=1 TmpDisk=0 Weight=1 Owner=N/A
  MCS_label=N/A
  BootTime=2016-08-22T15:04:00 SlurmdStartTime=2016-08-22T16:49:13
  CapWatts=n/a
  CurrentWatts=0 LowestJoules=0 ConsumedJoules=0
  ExtSensorsJoules=n/s ExtSensorsWatts=0 ExtSensorsTemp=n/s
  Reason=foo
Availability in Slurm

- First developments in 2015
- In slurm 16.05.0-pre1 version
MCS Plugin

Planned features
MCS Plugin
Planned features

MCS-label stored in database

- MCS-label is not stored in the database.
- Should be stored in cluster_job_table table (tinytext type)
- Add a new format option McsLabel in sacct
MCS Plugin
Planned features

Use a hash table for MCS

- Current mcs/group plugin asks the operating system for groups membership of users whenever it is necessary
  - → putting the pressure on the OS groups caching logic,
  - → and thus introducing an heavy load for large systems with a high number of pending and running jobs.

So:
- Reusing and/or enhancing the group caching logic of Slurm in the mcs/group plugin is planned to reduce that effect.
Thank you for your attention

Questions ?
API Functions in MCS plugin

- extern int slurm_mcs_init(void);
- extern int slurm_mcs_fini(void);
- extern int mcs_p_set_mcs_label(struct job_record *job_ptr, char *label);
  - Verify and set or calculate MCS-label for a job.
  - Called by _job_create to get the mcs_label for a job.
- extern int mcs_p_check_mcs_label(uint32_t user_id, char *mcs_label);
  - For squeue/scontrol show nodes in case of option privatedata.
  - Check the compatibility between MCS-label of user and MCS-label of jobs/nodes.
Internal functions in MCS plugin

- extern int slurm_mcs_reconfig(void);
- extern char *slurm_mcs_get_params_specific(void);
- extern int slurm_mcs_reset_params(void);
- extern int slurm_mcs_get_select(struct job_record *job_ptr);
- extern int slurm_mcs_get_enforced(void);
- extern int slurm_mcs_get_privedata(void);
- extern char *slurm_mcs_get_params_specific(void);
- extern int mcs_g_set_mcs_label(struct job_record *job_ptr, char *label);
- extern int mcs_g_check_mcs_label(uint32_t user_id, char *mcs_label);