How to Install and Setup Slurm

Status	🏗️ In Progress
Type	👷 How To
Created	@April 3, 2023 1:01 PM
Last Edited Time	@April 5, 2023 11:17 AM

Overview
- Control Node
- Compute Nodes

Installation
- Packages
  - All Nodes
  - Control Node
  - Compute Nodes

Enabling SSH
- Control Node
- Compute Nodes

Configuring Munge
- Moving the Munge Key
- Testing Munge
  - Test On One Node
  - Testing Between Two Nodes

Configuring Slurm
- Slurm.conf
  - Adding Compute Nodes:
  - Adding Partitions:
- Cgroup.conf
- Moving the Files

Setting up NFS:
- Creating a Directory and Exporting it
- Mounting Mirrored Directory
- Creating Users Folder
  - Symbolic Link

OpenMPI

Sample Scripts
- Show Hostnames
- MPI Hello World

Notes

Overview

For this guide I will be using Ubuntu Live Server version 22.04 LTS.

In order for this to go smoothly, every node must have a static IP as they will be communicating with each other through SSH.

Control Node

This is the main node that all users will log into to create and run their programs. Talks to all the other nodes and is where the NFS file share is off. Can only have one control node.

Compute Nodes

Simply runs the jobs given to it by the control node. Can have as many as you like.

Installation

Packages

The following packages will need to be installed on their respective nodes.

All Nodes

- libopenmpi3
- libopenmpi-dev
- libglib2.0-0
- munge
- net-tools
- openssh-client

Control Node

- nfs-kernel-server
- slurmctld
- krb5-client
- libpam-krb5

Compute Nodes

- slurmd
- nfs-common

Enabling SSH

SSH is used in the cluster by slurm from the control node to communicate with the compute nodes. Hostbased authentication will be used.

Control Node

In the the file `/etc/ssh/ssh_config` edit the following lines:

`EnableSSHKeySign yes`

`HostbasedAuthentication yes`

These lines will need to be uncommented as well.

After editing the file the service will need to be restarted with the command: `sudo systemctl restart sshd`

Compute Nodes

In the file `/etc/ssh/ssh_config` edit the following lines:

`HostbasedAuthentication yes`

`IgnoreRhosts no`

These lines will need to be uncommented as well.

Create the `shosts.equiv` and save the IP of the control node into it. It will be saved at `/etc/ssh/shosts.equiv`

Create a file called `ssh_known_hosts` in `/etc/ssh/` Hostbased Authentication will use this file. To get it to work run the command: `ssh-keyscan <Control-Node-IP-Addr> >> /etc/ssh/ssh_know_hosts` This will save the control nodes IP to the file and allow any user on the control node to passwordlessly ssh into the compute nodes.

Configuring Munge

On installation munge creates a `munge.key` file in `/etc/munge`. However, all nodes need the same munge key for slurm authentication to work between all the nodes.

Moving the Munge Key

From the control node in the directory `/etc/munge` copy the munge key to compute nodes using the following scp command:

`scp munge.key <username>@<compute-node-ip>:/home/username`

Then move the munge key from for the users home directory to `/etc/munge`. Munge may require that the key is owned by root and then all the nodes will need to be rebooted.

Testing Munge

Test On One Node

`munge -n | unmunge`

Testing Between Two Nodes

From the control node to compute node

`munge -n | ssh <compute-node> unmunge`

Configuring Slurm

You can find examples of the files that need to be edited in `/usr/share/doc/slurm-client/examples/`.

There are two config files that need to be created in `/etc/slurm/`.

Slurm.conf

Add the following lines to the file:

`SlurmctldHost=<control-node-hostname>` or IP

Adding Compute Nodes:

`NodeName=<compute-node-hostname> CPUS=32 RealMemory=30000 Sockets=1 CoresPerSocket=8 ThreadsPerCore=2 State=UNKNOWN` All the information filled into the CPUS, memory etc. is all from the nodes physical hardware specs. Information about the CPU can be found with the command: `lscpu` It is recommended to not pull the full amount of memory available to slurm. In my case I have 32 GB of ram but am putting 30 GB down for slurm to have access to.

Adding Partitions:

`PartitionName=all NOdes=<Control-Node>,<Compute-Nodes> Default=YES MaxTime=INFINITE State=UP`

MaxTime can be edited to allow jobs to timeout after a certain amount of time.

Cgroup.conf

Add the following lines:

`CgroupAutomount=yes`

`CgrouptReleaseAgentDir="/etc/slurm/cgroup"`

`ConstrainCores=yes`

`ConstrainDevices=yes`

`ConstrainRAMSpace=yes`

These lines will be uncommented as well

Moving the Files

Copy both of the config files, `Slurm.conf` and `Cgroup.conf` and move them into `/etc/slurm/`

On the control node restart slurm with the following command: `sudo systemctl restart slurmctld`

On the compute nodes restart slurm with the following command: `sudo systemctl restart slurmd` (If the control node is also a compute run this command as well)

Now on the control node run the command: `sudo scontol reconfigure` and then the command: `sinfo` to show all the compute nodes and partitions.

```
root@iccsl-2021:~# sinfo
PARTITION AVAIL  TIMELIMIT  NODES  STATE NODELIST
all*         up   infinite      1  down* iccsl-node2
all*         up   infinite      4   idle iccsl-2021,iccsl-node[3-5]
root@iccsl-2021:~#
```

If you are running into issues, check the status of `slurmd` on the compute nodes and `slurmctld` on the control node. Also check the munge and slurm logs in directory `/var/log/`

Setting up NFS:

NFS allows the control node to share the users program with the compute nodes.

Creating a Directory and Exporting it

Run the command: `sudo mkdir /mirror` on all the nodes to create a directory in the root of the system

Then for the control node run the command: `echo "/mirror *(rw,sync)" | sudo tee -a /etc/exports` This allows us to export the mirror directory using NFS for the compute nodes to find. Finally restart the NFS service on the control node with the command: `sudo service nfs-kernel-server restart`

Mounting Mirrored Directory

On all the compute nodes run the command: `sudo mount <control-node>:/mirror /mirror`

This will mount the control nodes mirror directory to the compute nodes mirror directory and allow the sharing of files.

Creating Users Folder

Inside the mirror directory on the control node create a user directory for every user that is using slurm.

Symbolic Link

To allow the users mirror folder to be more accessible create a symbolic link to it from their home directory using the command: `ln -s /mirrir/<username> /home/<username>/slurm`

If you do this for one user it must be done on all the compute nodes to for that user otherwise you will encounter an error. The user must be on all the nodes too.

OpenMPI

There is no configuration that must be done with OpenMPI but in order to run sbatch shell files that execute C files that have MPI commands in them the shell files must be run with `mpiexec` not srun or the job will fail.

Sample Scripts

Show Hostnames

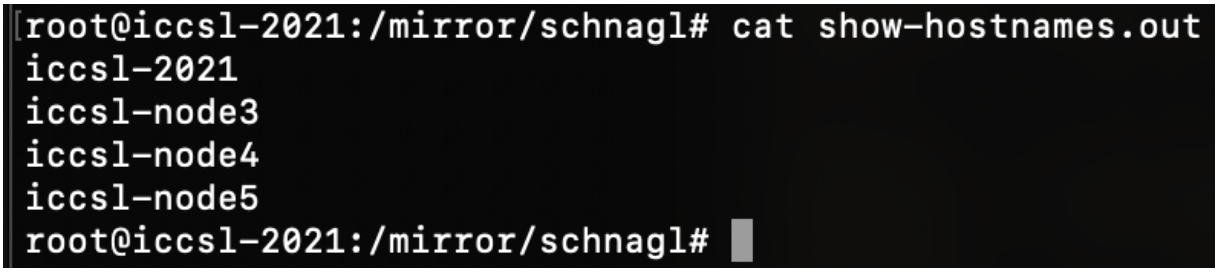
```
#!/bin/bash
#
#SBATCH --job-name=show-hostnames
```

```
#SBATCH --output=show-hostnames.out
#
#SBATCH --ntasks=4
#SBATCH --time=10:00
#SBATCH --mem-per-cpu=100
#SBATCH --ntasks-per-node=1

srun hostname
```

Most of the SBATCH configuration for this file is fairly easy, but in order for it to display all hostnames you will want to set `--ntasks=` to the number of compute nodes you have.

To execute the script run the command: `sbatch show-hostnames.sh` and then cat the output file to see the results.



MPI Hello World

```
#include <stdio.h>
#include <unistd.h>
#include <mpi.h>

int main(int argc, char** argv)
{
  // Init the MPI environment
  MPI_Init(NULL, NULL);

  // Get the number of processes
  int world_size;
  MPI_Comm_size(MPI_COMM_WORLD, &world_size);

  // Get the rank of the process
  int world_rank;
  MPI_Comm_rank(MPI_COMM_WORLD, &world_rank);

  // Get the name of the processor
  char processor_name[MPI_MAX_PROCESSOR_NAME];
  int name_len;
  MPI_Get_processor_name(processor_name, &name_len);

  // Print a hello world message
  printf("Hello, World! from from processor %s, rank %d out of %d processors\n", processor_name, world_rank, world_size);

  // Finalize the MPI environment
  MPI_Finalize();
}
```

Compile the code with the command: `mpicc -o mpi-helloworld mpi-helloworld.c`

The create a SBATCH script to run it

```
#!/bin/sh
#SBATCH -o mpi-helloworld.out
#SBATCH --nodes=4
#SBATCH --ntasks-per-node=1

mpiexec ./mpi-helloworld
```

To execute the script run the command: `sbatch mpi-helloworld.sh`

Check the results of the script with cat on the output file

```
root@iccsl-2021:/mirror/schnagl# cat mpi-helloworld.out
Hello, World! from the processor iccsl-2021, rank 0, out of 3 processors
Hello, World! from the processor iccsl-node4, rank 2, out of 3 processors
Hello, World! from the processor iccsl-node3, rank 1, out of 3 processors
root@iccsl-2021:/mirror/schnagl#
```

Notes

Make sure all of the IP’s for all nodes are static instead of DHCP.

If a server is brought down, ethernet disconnected, etc. On the control node run `sudo scontrol update nodename=<compute-node-name> state=idle` to bring the nodes back up.

If the control node keeps going down, the issue for me seemed to be in the munge authentication for slurm. Which means the munge key was not copied over correctly or the servers where not rebooted after the key was copied over.

All users using slurm must be added manually to each node `sudo adduser <username>`