Alma Mater Studiorum – CdS in Ingegneria Informatica

Tesi di Laurea in Amministrazione di Sistemi T

Sviluppo di un'infrastruttura virtuale per l'erogazione di servizi di calcolo con SLURM

Relatore:
Prof. Marco Prandini

Correlatore:
Ing. Andrea Giovine

Presentata da:
Massimo Valerio Zerbini

Obiettivi del progetto

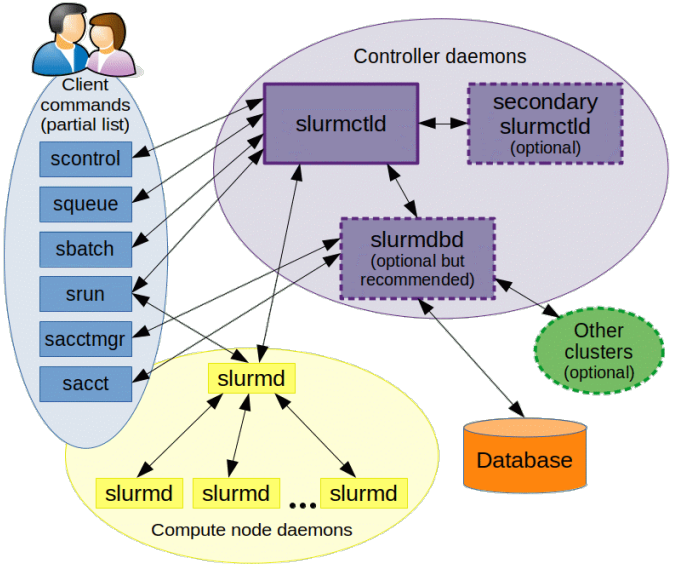
- *(Impostazione dell'infrastruttura virtuale)*
- **Condivisione delle risorse di computazione:** assegnamento dinamico di risorse in base alle necessità dei job da eseguire
- **Priorità di *scheduling*:** configurazione di una partizione SLURM prioritaria (su una specifica risorsa), accessibile esclusivamente da un determinato utente
- **Federazione di 2 *cluster*:** coordinamento tra più *cluster* SLURM per l'esecuzione di calcoli

Ambiente di sviluppo

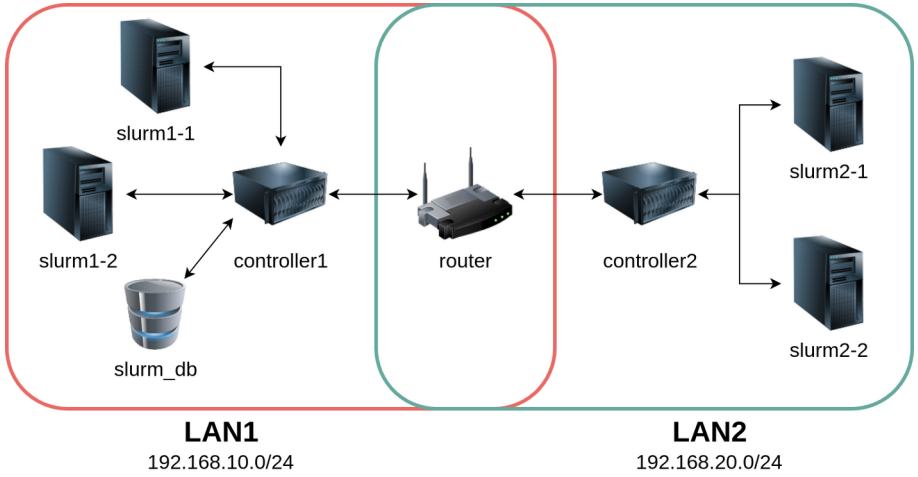
- **Vagrant & Ansible** per *Infrastructure as Code* (IaC) virtuale:
  - gestione semplificata di molteplici VM, a partire da una *box* di base
  - configurazione automatizzata delle macchine virtuali (*provisioning)*

- **Repository Git** per il controllo delle versioni:
  - cronologia delle modifiche
  - progettazione non lineare delle funzionalità (*branching* & *merging*)

Topologia SLURM (Simple Linux Utility for Resource Management)



Architettura SLURM



Infrastruttura virtuale desiderata

DHCP, DNS & Accounting

`dnsmasq`: DHCP *server* + DNS *resolver* locale

- i nodi della LAN eseguono una richiesta DHCP a `dnsmasq`, che concede il *leasing* di un indirizzo IP all'interno di un *range* configurabile, **registrando allo stesso tempo l'*hostname* corrispondente**
- in questo modo, i nodi richiedono a `dnsmasq` la risoluzione di un *hostname* nella LAN, ricevendo l'indirizzo IP in risposta

`controller1` —*DNS resolution*→ `192.168.10.23`

MariaDB: *Database Management System* (DBMS) relazionale, basato su MySQL

- utilizzato dal demone **`slurmdbd`** per la registrazione delle attività SLURM (*accounting*)
- necessario introdurre un utente apposito con i **privilegi** appropriati (`slurm_db.*:ALL`)
- **essenziale** per la configurazione della priorità di *scheduling* e per l'impostazione di una federazione

Condivisione delle risorse (CPU, RAM, GPU)

- Attivazione del plugin `select/cons_tres` (*Consumable Trackable Resources*)
- Configurazione aggiuntiva per le **GRES** (*Generic Resources*), sotto cui la GPU è catalogata

```
vagrant@controller1:~$ srun -N1 -c1 --mem=256 sleep 60 &
[1] 18254
vagrant@controller1:~$ srun -N1 -c1 --mem=256 sleep 60 &
[2] 18264
vagrant@controller1:~$ srun -N1 -c1 --mem=256 sleep 60 &
[3] 18274
vagrant@controller1:~$ srun -N1 -c1 --mem=256 sleep 60 &
[4] 18284
vagrant@controller1:~$ squeue
JOBID PARTITION     NAME     USER ST       TIME  NODES NODELIST(REASON)
    1     debug    sleep  vagrant  R       0:15      1 slurm1-1
    2     debug    sleep  vagrant  R       0:14      1 slurm1-1
    3     debug    sleep  vagrant  R       0:13      1 slurm1-1
    4     debug    sleep  vagrant  R       0:12      1 slurm1-1
```

Esecuzione parallela di 4 job (ciascuno **richiedente 1 CPU e 256 MB di RAM**) su un singolo nodo da 4 CPU e 1 GB di RAM

```
vagrant@controller1:~$ srun -N1 --gpus=1 --mem=256 sleep 60 &
[1] 18294
vagrant@controller1:~$ srun -N1 --gpus=1 --mem=256 sleep 60 &
[2] 18304
vagrant@controller1:~$ squeue
JOBID PARTITION     NAME     USER ST       TIME  NODES NODELIST(REASON)
    5     debug    sleep  vagrant  R       0:08      1 slurm1-1
    6     debug    sleep  vagrant  R       0:07      1 slurm1-1
```

Esecuzione parallela di 2 job (ciascuno **richiedente 1 GPU**) su un singolo nodo da 2 GPU

Priorità di scheduling

- Definizione di una **partizione SLURM aggiuntiva**, accessibile solo da un determinato utente
- Attivazione del plugin `priority/multifactor`, per impostare la priorità della partizione **su una GPU**
- **Registrazione** nel DB **della nuova partizione** e **delle varie associazioni utente**

```
vagrant@controller1:~$ sudo -u user2 srun -N1 --gpus=1 --mem=256 sleep 60 &
[1] 18314
vagrant@controller1:~$ sudo -u user3 srun -N1 --gpus=1 --mem=256 sleep 60 &
[2] 18324
vagrant@controller1:~$ sudo -u user4 srun -N1 --gpus=1 --mem=256 sleep 60 &
[3] 18334
vagrant@controller1:~$ sudo -u user1 srun -N1 --partition=gpupart --gpus=1 --mem=256 sleep 60 &
[4] 18344
vagrant@controller1:~$ squeue
JOBID PARTITION     NAME     USER ST       TIME  NODES NODELIST(REASON)
    9     debug    sleep    user4 PD       0:00      1 (Resources)
    8     debug    sleep    user3  R       0:03      1 slurm1-1
    7     debug    sleep    user2  R       0:17      1 slurm1-1
   10   gpupart    sleep    user1 PD       0:00      1 (QOSGrpGRES)
```

Job 9 e 10 in contesa di risorse (in particolare, di una GPU)

termine job 7 →

```
vagrant@controller1:~$ squeue
JOBID PARTITION     NAME     USER ST       TIME  NODES NODELIST(REASON)
    9     debug    sleep    user4 PD       0:00      1 (Resources)
    8     debug    sleep    user3  R       0:50      1 slurm1-1
   10   gpupart    sleep    user1  R       0:04      1 slurm1-1
```

Grazie alla sottomissione nella partizione `gpupart`, **il job dell'utente prioritario** (job 10) **viene eseguito prima**

Federazione di cluster SLURM

- Utilizzo del **medesimo DB** per entrambi i *cluster*
- Definizione e **registrazione della federazione** nel DB
- **Inoltro dei job** tra i *cluster*, a seconda delle necessità

```
vagrant@controller1:~$ sacctmgr show federation
Federation    Cluster ID             Features     FedState
---------- ---------- -- -------------------- ------------
testfeder+   cluster1  1                           ACTIVE
testfeder+   cluster2  2                           ACTIVE

vagrant@controller1:~$ scontrol show federation
Federation: testfederation
Self:       cluster1:127.0.0.1:6817 ID:1 FedState:ACTIVE Features:
Sibling:    cluster2:192.168.20.194:6817 ID:2 FedState:ACTIVE Features:
↪   PersistConnSend/Recv:Yes/Yes Synced:Yes
```

Verifica dello **stato** della federazione
(comandi `sacctmgr` e `scontrol`)

```
vagrant@controller1:~$ srun --clusters=cluster2 -N2 hostname
slurm2-1
slurm2-2

vagrant@controller2:~$ srun --clusters=cluster1 -N2 hostname
slurm1-1
slurm1-2
```

Verifica del corretto **funzionamento** della federazione
(inoltro ed esecuzione di `hostname`)

Sviluppi futuri

- Ampliamento delle **funzionalità di calcolo** (e.g. *preemption* & *gang scheduling*)
- Introduzione di **test automatici** provocati dalla modifica dell'infrastruttura, principio noto come *Continuous Integration & Continuous Delivery* (CI/CD)
- Applicazione in **contesti reali** di calcolo ad alte prestazioni (*High Performance Computing*)