Retour d'exprience sur le cluster du CDS

Gilles Landais (CDS)

6 mai 2011

Sommaire

Généralités

Types de cluster

- Haute disponibilite
- Load balancing
- Cluster de calculs
  - avec interaction entre noeud *(Beowulf)*
  - sans interactions

Utilisation add-hoc

- MPI pour le code
- Software *(authentification, balancing, prises en charge des jobs, files d'attetnes)*

Composant du cluster du CDS

Couche	**CDS**
Haute disponibilité	Heartbeat
Systeme d'images	DCC *(Debian Cluster Components)*
Monitoring	Ganglia
software UI	ALI

Composant système du cluster

DCC

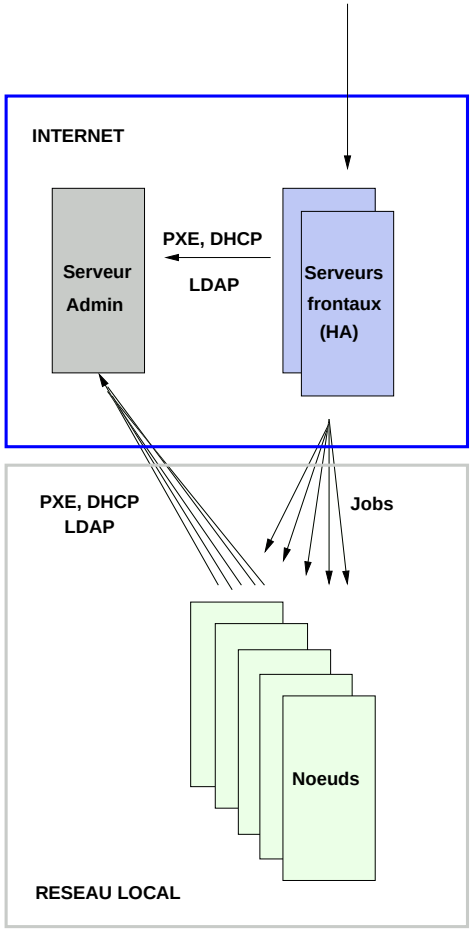
- Développé en croatie par le centre de Recherche **IRB** *(Institut de Recherche de Boskovic)*.
- Package debian s'appuyant sur des softwares linux

Compostants de DCC

- Système d'image, PXE au boot

  (SSI « Single System Image »)
- Administration des noeuds *(C3)*
- Shorewall
- LDAP, DHCP
- TORQUE

  (files d'attente, sheduling, monitoring mais non utilisé)



Le cluster ALI

- Développé en interne de 2004 *(programmation C)*
- Load balancing
- Interractions entre les noeuds *(non utilisé)*
- Authentification par IP
- Liste statique de jobs configurable sur chaque worker

Possibilités d'utilisation

- MPI *(non utilisé)*
- envoi de fichiers
- time-out
- réexécution en cas d'echec
- chaînage de jobs

Le fichier de requêtes

```
%HEADER Requete exemple
%FIFILES fich1 fich2
%FOFILES fich.tar
%SEPARATOR
# Compression du premier fichier
%ID 1
%CMD gzip fich1
%IFILES fich1
%OFILES fich1.gz
%LFILES
%REQUIRE gzip
%DEPS
# Compression du second fichier
%ID 2
%CMD gzip fich2.zip
%IFILES fich2
%OFILES fich2.gz
%LFILES
%REQUIRE gzip
%DEPS
# Archivage des deux fichiers compresses
%ID 3
%CMD tar -cf fich.tar fich1.gz fich2.gz
%IFILES fich1.gz fich2.gz
%OFILES fich.tar
%LFILES
%REQUIRE tar
%DEPS 1 2
```

Architecture ALI

Le serveur

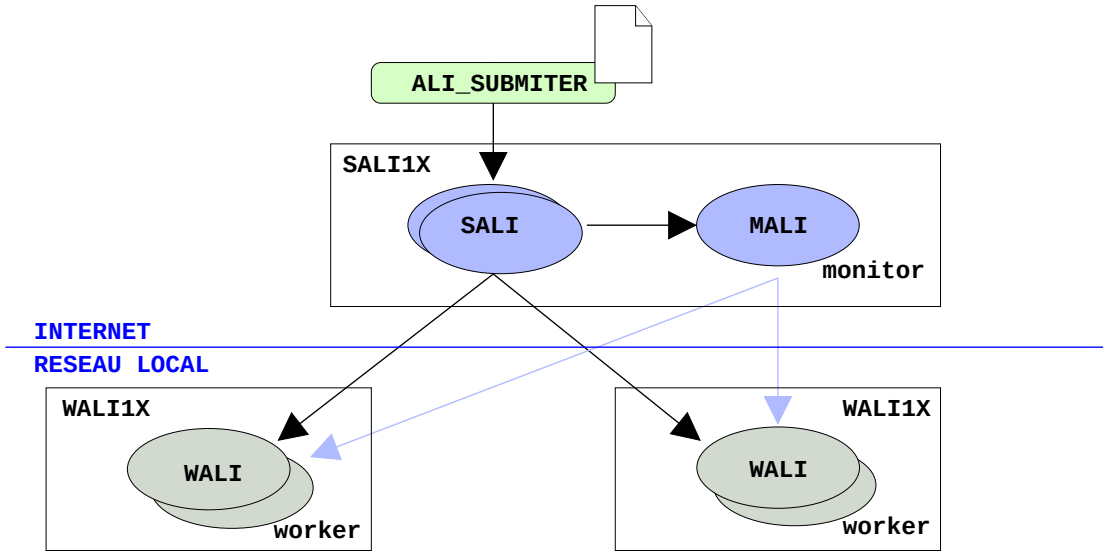
- authentification
- decomposition de la requête et récupération des fichiers
- ordonance l'exécution des jobs sur les workers donnés par le moniteur

Le moniteur

- Etat des workers
- Liste des jobs disponible par worker
- Calcul de la charge *(CPU, temps réseaux, score)*

Le worker

- exécution de la requête.
  les sorties standard et d'erreurs sont envoyés au fil de l'eau au server qui les retransmet au client



Utilisation et statistiques

Contexte

- **VizieR :** accès aux grands catalogues
  Utilisation : calculs leger, mais nombreux

  Nb de requetes : $\approx$ 100 000, pic $>$1 000 000

- **Aladin :** sextractor, aladinBatch (rgb)
  Utilisation : calculs plus lourd, peu nombreux

  Nb de requetes : $\approx$ 40

- **GDL :**
  Utilisation : 60 000 scripts concernant chacune une galaxie.
  calculs lèger 1min/job executé sur 4noeuds à raison de 2noeuds par jobs (*Rightarrow* 9000 galaxies traitées/jour)

  Nb de requetes : ? sur ? jours

Architectures testées

	+eurs serveurs	**cluster**	**serveur unique**
Années	2004(?)- juin 2007	juin 2007-2010	depuis oct 2010
Services	vizier	aladin,vizier,gdl	vizier
Software	**ALI**	**ALI**, Heartbeat, SSI, Ganglia	
Nb de noeuds	4	8	1+1

Maintenance du système

- **Problème hardware** avec les machines TRANSTEC :
  $\Rightarrow$ maintenance lourde
- **System DCC**
  - abandon LDAP : car probleme pour le lancement au boot
  - système d'image ok mais contraignant

    $\Rightarrow$ *script de mise a jour atomatique sur VizieR*
  - **DCC non maintenu**
- **heartbeat** OK
  - reglage necessaire pour eviter le ping-pong
  - Ajout de « mon » pour la survaillance ALI

    (autre soft :AHM=application heartbeat monitor)

Utilisation de ALI

Les points forts

- tient la charge (cas GDL)
- bien programme (fork, thread..)

  quelques developpement : 64bits, logs, bugs
- peut s'installer facilement sur des plates forme de même type

Les zones de doutes

- problème de répartition de charge :

  base sur 1 calcul de la charge asynchrone sur sched_yield (thread sur le pere : comment peut etre interprete cela dans un multi core ?)
- toutes les fonctionnalités non utilisées

Les points faibles

- fragilité au niveau du moniteur
- pas de files d'attentes
- pas d'API java satisfaisante

Conclusion

Maintenance

- maintenance accrue pour une puissance surestimée (cas VizieR). ⇒ retour à une architecture plus simple.
- non évolutivité du système d'image DCC :
  Si besoin, passer a une solution
  - utilisation de standard simples

    (doc ubuntu : http ://doc.ubuntu-fr.org/cluster), ou cluster Beowulf ?
  - systeme clef-en-main

Des possibilités non utilisés

- pour le chainage de jobs
- calculs longs et massifs

Des manques

Files d'attentes absentes : utilisation de la couche ? ?, TORQUE (?)