ENEA GRID

Corso di introduzione all'uso ed all'amministrazione

Autore: Alessandro Secco
alessandro.secco@nice-italy.com

Lezione 2

- Riepilogo lezione 1
- Trasferire i file sul cluster: il client AFS.
- LSF Utente: concetti di base
- Lancio di un job

Riepilogo lezione 1

- Nella prima lezione si è parlato di *risorse* e di *interfaccia grafica* Enea.
- Risorse:
  - Definizione
  - Classificazione
  - Esempi
  - Esercizi
- Interfaccia grafica
  - Installazione ed uso

Trasferimento file: il client AFS.

Trasferimento file: il client AFS.

- Il link seguente contiene le informazioni su come installare il client afs su windows.
  - http://www.telegrid.enea.it/Openafs/Prova/installazione.html
- Per Linux, ogni distribuzione ha una procedura diversa, ma per quelle più comuni (OpenSuse, Fedora, Ubuntu, ...) esiste documentazione facilmente rintracciabile.
- Una volta installato il client, si mappa un disco (per esempio z:) sulla propria home; in seguito si possono copiare o editare i file come se fossero installati in locale.

LSF Utente: concetti di base

LSF Utente

- LSF (Load Sharing Facility) è un prodotto di gestione del carico di lavoro, sviluppato da Platform Computing, una società canadese:
  - http://www.platform.com.
- LSF esiste in Enea da circa 10 anni ed è usato da un numero sempre crescente di utenti.
- LSF svolge mansioni di gestione e monitoraggio delle risorse, dei job utente, accounting, ed altro ancora (si consiglia di visitare il sito).

LSF Utente: terminologia

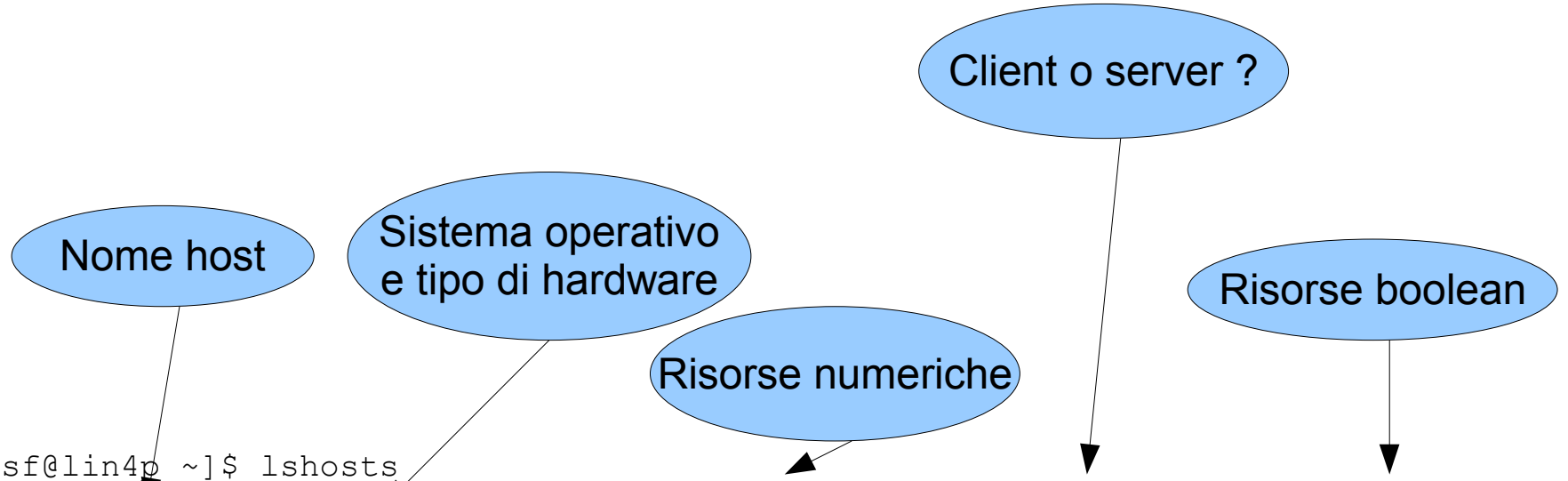
- CLUSTER: un gruppo di computer (host) in cui girano i demoni di LSF.
- JOB: è un comando batch lanciato dall'utente tramite LSF.
- TASK: è un comando interattivo lanciato tramite LSF; in Enea anche i task sono lanciati come job.
- CLIENT: è la macchina da cui si sottomettono i job ad LSF.
- SERVER: è il calcolatore sul quale gireranno i job di LSF. Una macchina può essere client e server allo stesso tempo.
- HOST di sottomissione: è il client usato per lanciare un job
- HOST di esecuzione: è il server su cui gira il job lanciato

Lsf utente

- Ecco alcuni comandi essenziali per l'uso di LSF. I comandi hanno help e man pages.
  - lshosts: mostra le risorse statiche per host
  - lsload: risorse dinamiche per host
  - lsinfo: mostra l'elenco dei nomi di risorsa
  - bhosts: mostra il numero di job sui vari host
  - bqueues: informazioni sulle code
  - bsub: lancia un job
  - bjobs: controlla lo stato di un job
  - bkill: termina un job in esecuzione

Comandi: lshosts

- Per vedere le risorse statiche a disposizione si utilizza lshosts.



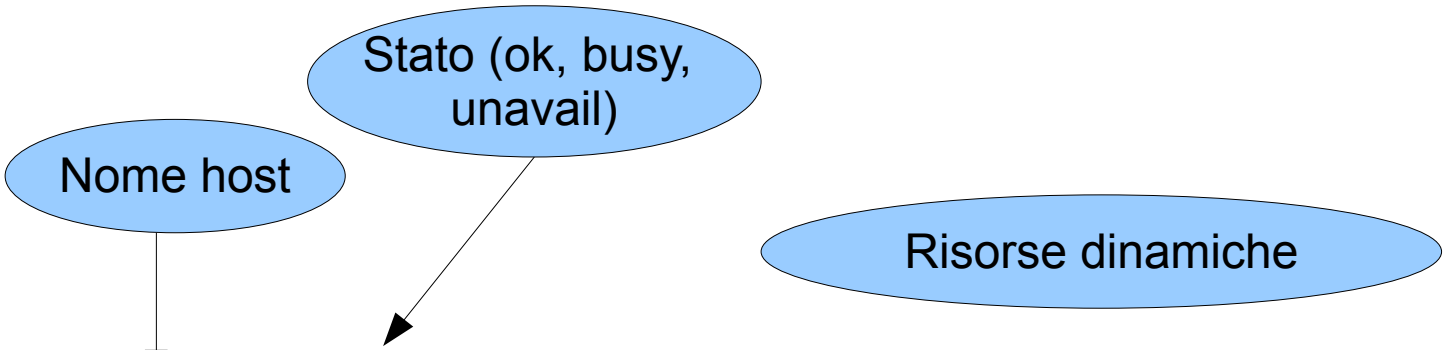

```
[lsf@lin4p ~]$ lshosts
HOST_NAME      type    model   cpuf ncpus maxmem maxswp server RESOURCES
graphbri.br     SGI6     ONYX3  30.0     1  1024M  4096M    Yes (sgi telnet)
ercules.bri IBMAIX5    POWER4  60.0     2  8192M  8704M    Yes (sp4 telnet abaqus)
campus03.br   LINUX   PIV3400  55.0     4  4027M 24003M    Yes (linux telnet abaqus)
spartacus.b UNKNOWN UNKNOWN_   1.0     -      -      -     No ()
```

Comandi: lshosts. Esercizio 1

- Provare a lanciare i seguenti comandi
  - lshosts portici **<Tutti gli host di un cluster>**
  - lshosts -l sp5-1 **<Dettaglio di un host>**
  - lshosts -R linux **<Filtro su risorsa boolean linux>**
  - lshosts -R type==linux **<Sistema operativo linux>**
- Nota: -R è usato in molti comandi di LSF e serve a richiedere delle risorse. Usa una sintassi simile alle condizioni del C ed impareremo come si usa tramite vari esempi.

Comandi: lsload

- Il comando lsload è l'analogo del lshosts per le risorse dinamiche.




```
HOST_NAME       status  r15s   r1m  r15m   ut     pg  ls     it   tmp   swp   mem
eurofel16.frasc     ok   0.0   0.0   0.0   0%    0.0   0 10400 6412M 2037M 2634M
eurofel15.frasc     ok   0.0   0.0   0.0   0%    0.0   0   176 5624M 2047M 3294M
sp5-3.frascati.     ok  14.4  14.5  13.5  88%    9.9   0   382  255M   30G 1133M
info-eva.bologn     ok  26.8  26.4  26.5  84%    0.0   1   113  435M   16G 24G
lin4p.frascati.   busy   0.1   0.9   0.1   5%*245.6   6     0 5904M 7148M 7192M
aix42w.frascati unavail
```

Comandi: lsload. Esercizio 2

- Provare a lanciare lsload con le stesse opzioni che abbiamo usato con lshosts. Cosa cambia ?
- Le opzioni dei due comandi, in questo caso sono intercambiabili. L'output dei due comandi rispecchia due aspetti diversi. Il comando lshosts mostra risorse statiche, mentre lsload è focalizzato sulle risorse dinamiche.

Risorse incluse in LSF

- Risorse statiche:
  - type, model: sistema operativo e modello di macchina
  - cpuf: è un indicatore della potenza di CPU. L'amministratore può cambiare questo parametro se non fosse accurato
  - ncpus: indica il numero di cpu (nei processori multicore, dipende dalla configurazione)
  - maxmem, maxswp: memoria e swap installati
  - RESOURCES: risorse boolean, definite dall'amministratore

Risorse incluse in LSF

- Risorse dinamiche:
  - status: unavail indica indisponibilità. busy indica macchina carica. ok, macchina scarica.
  - r15s, r1m, r15m: è la run queue length mediata in 15 secondi, 1 minuto, 15 minuti (vedere man uptime).
  - ut: utilizzo in % di tutte le cpu
  - pg, ls, it: paging, login interattivi, idle time
  - tmp, swp, mem: rispettivamente lo spazio in tmp, lo swap e la memoria liberi.

Esempi di uso di risorse

- Per allenarsi con le risorse senza causare problemi si possono usare lsload ed lshosts.
  - lsload -R “r1m<1”
    - Minore, maggiore, uguale: <, >, ==
    - Qui l'uso delle virgolette è necessario
    - r1m si misura in processi
  - lsload -R “mem>3000”
    - La memoria si misura in MegaByte
  - lsload -R “swp>3000 && mem>4000”
    - Operatori. AND: &&, OR: ||, NOT: !

Esempi di uso di risorse

- lsload -R type==LINUX
  - type è statica, ma si può usare con lsload
- lsload -R "(status==busy || ut>50) && type==LINUX"
  - L'uso di parentesi è consentito e molto utile
- lshosts -R '!server'
  - “Not server” elenca le macchine solo client
  - L'uso dell'apice singolo è utile se la shell cercasse di interpretare il punto esclamativo.
- lsinfo
  - Vedo l'elenco di tutte le risorse.

Lancio di un job

Il comando bsub

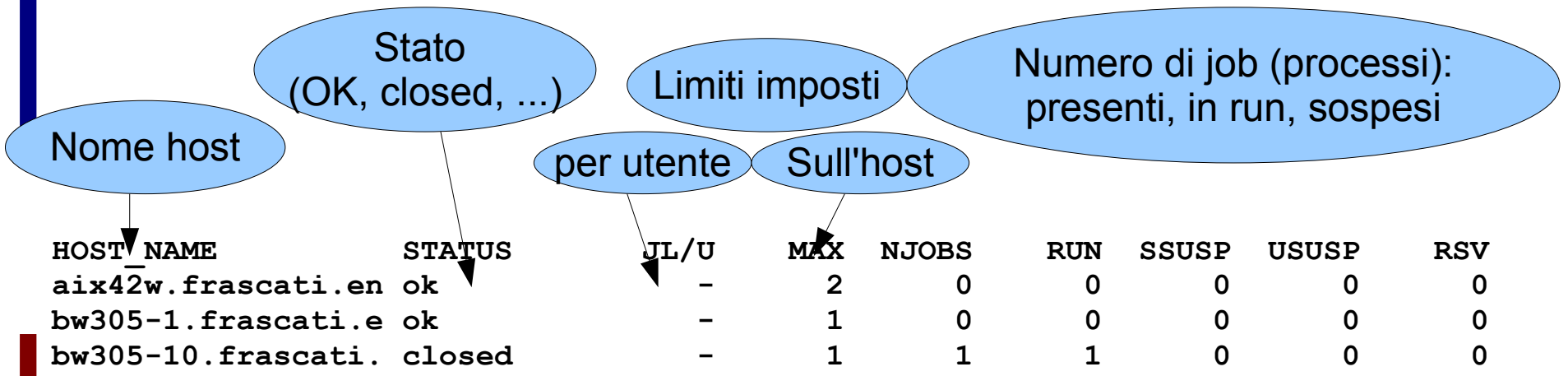
- Per lanciare un job in LSF bisogna trovarsi su un client o server LSF e lanciare:
  - bsub <para_bsub> comando <para_cmd>
- Primo lancio:
  - bsub sleep 10000
- Che informazioni mi dà il comando bsub in output ?
  - Il jobid, utile per far riferimento al job sottomesso
  - La coda di sottomissione

Le code

- La coda è un contenitore astratto di job che serve a scegliere in modo accurato il server di esecuzione.
- Il comando **bqueues** elenca tutte le code a disposizione. **bqueues -l <coda>**: mostra i dettagli di una coda.
- Usare **bsub -q <coda>** per lanciare il proprio job su una specifica coda. Regole:
  - Tutte le code sono limitate
  - Le code maggiormente limitate hanno maggior priorità.
  - E' molto utile stimare la durata del proprio job
- E' compito dell'amministratore gestire le code

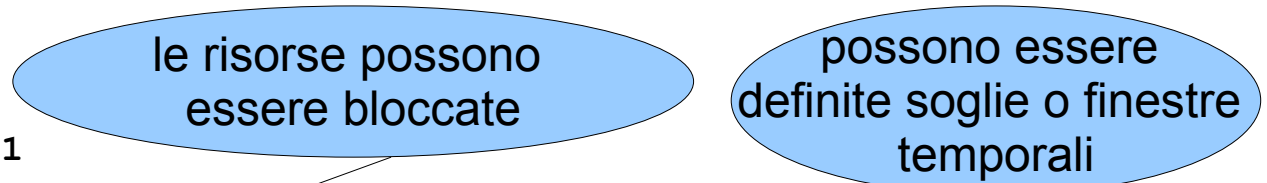
Gli host

- Il comando **bhosts** mostra la situazione dei job distribuiti su tutti gli host.
- A differenza di lsload e lshosts, che mostrano la situazione monitorata, bhosts vede il cluster con gli occhi di LSF



bhosts -l

- Per schedulare i job, LSF tiene effettivamente conto di questi parametri, a prescindere dall'output di lsload o lshosts.




```
[lsf@lin4p ~]$ bhosts -l sp5-1
HOST  sp5-1.frascati.enea.it
STATUS           CPUF  JL/U    MAX  NJOBS    RUN  SSUSP  USUSP    RSV DISPATCH_WINDOW
ok              75.00     -     16     13     13      0      0      0      -

 CURRENT LOAD USED FOR SCHEDULING:
                r15s   r1m  r15m    ut    pg    io   ls    it   tmp   swp   mem
 Total           1.0   1.0   1.0   59%  82.7   312    5    54  554M   30G 3200M
 Reserved        0.0   0.0   0.0    0%   0.0     0    0     0    0M    0M    0M

 LOAD THRESHOLD USED FOR SCHEDULING:
           r15s   r1m  r15m    ut      pg    io   ls    it   tmp   swp   mem
 loadSched   -     -     -     -       -     -    -     -     -     -     -
 loadStop    -     -     -     -       -     -    -     -     -     -     -
```

Il job

- Un job è identificato dal jobid
- Per vedere tutti i job attivi sull'utente corrente, uso il comando bjobs senza argomenti.
- bjobs -l <jobid> dà informazioni utili su un certo job
- Per uccidere un job, si usa bkill <jobid>
- I file prodotti dal job rimangono nella directory corrente, ma attenzione a <stdout> e <stderr>.

Stato del job

- bjobs fa vedere gli stati del job.
  - PEND: il job è in “pending”, ovvero attende risorse dal sistema. bjobs -pl <jobid> fa vedere le risorse mancanti.
  - RUN: il job sta girando. bjobs dice su che server.
  - DONE, EXIT: Si vedono solo con bjobs -a e per un periodo limitato di tempo. Significano “job finito” e “job uscito con errore”.
  - PSUSP, USUSP, SSUSP: Job sospeso (in pending, da utente, da sistema).

Output del job

- bsub supporta la ridirezione dell'output tramite i parametri:
  - -i <nomefile>: passa al un file sullo <stdin>
  - -o <nomefile>: salva lo <stdout+stderr> del comando dentro ad un file
  - -e <nomefile>: separa <stderr> da <stdout> e lo salva dentro ad un file
- Se l'utente non specifica la loro destinazione, <stdout> ed <stderr> vengono persi.
- Nel nome file si può usare la macro %J per avere in automatico il jobid nel nome del file.
  - bsub -o output.%J ls -l

Come preparare un job

- LSF accetta ogni comando unix.
- In realtà il comando bsub legge il percorso del comando che viene sottomesso; il comando deve quindi avere lo stesso path su tutte le macchine
  - **NOTA BENE**: non si può usare tmp come directory per lanciare i propri job, ma bisogna usare un file system centralizzato come AFS, a meno di casi molto particolari.
- E' consigliato creare una sottodirectory della propria home dove mettere i file del proprio job

Come preparare un job

- Se il job è presente nel path, basta lanciarlo dentro bsub:
  - bsub -R fluent -q large_72h fluent <params>
- Se il job è uno script della home, esso deve essere eseguibile, e va richiamato con il path
  - bsub -R linux ./script1.sh
- bsub legge anche lo <stdin>
  - bsub < script2.sh
- Sperimentare le differenze tra i due lanci
- Provare a creare script2.sh, mettendo **#BSUB -q large_72h**. Cosa succede ?

Come preparare un job

- Se il vostro job va compilato, leggersi:
  - **http://www.afs.enea.it/project/eneagrid/Resources/Working.html**
- Job multipiattaforma in 5 punti (grazie AFS):
  1. Compilare il job su tutte le piattaforme.
  2. Lanciare fs sysname su ogni piattaforma
  3. Creare delle sottodirectory con i nomi ottenuti dal comando precedente.
  4. Ricopiare eseguibili e librerie nelle sottodirectory create
  5. Lancio bsub @sys/nome_job <parametri>
- Il sistema sostituisce @sys con la piattaforma

Esercizio 3: preparare un job

```
bash-3.00$ hostname
sp4-1.frascati.enea.it

bash-3.00$ cat test.c
#include <stdio.h>

main()
{
        printf("hello from here\n");
}

bash-3.00$ fs sysname
Current sysname is 'rs_aix52'

bash-3.00$ mkdir rs_aix52
bash-3.00$ cc test.c -o @sys/test

bash-3.00$ ssh bw305-1

[lsf@bw305-1 ~/esempio]$ fs sysname
Current sysname is 'i386_linux26'
[lsf@bw305-1 ~/esempio]$ mkdir i386_linux26
[lsf@bw305-1 ~/esempio]$ cc test.c -o @sys/test

bash-3.00$ bsub -R "sp4 || linux" -o result @sys/test
```

Interazione con il job

- Un'opzione spesso utile, soprattutto in fase di debug, di bsub è -I (i maiuscolo)
- **bsub -I <comando>** ridireziona lo <stdout> del comando a video, dando l'impressione di aver lanciato il comando interattivamente.
  - NOTA: è bene non abusare di questa opzione (può aggiungere lavoro ai demoni di LSF).
- Per vedere l'output a video di un job batch lanciato senza -I si può usare **bpeek <jobid>**.
- **bpeek -f <jobid>** fa vedere l'output del job mentre viene prodotto.

Comando bsub: riepilogo

- bsub <para_bsub> comando <para_cmd>
- -o, -e, -i: ridirezione stdout, stderr stdin
- -q: scelta coda
- -R: restringe su gruppo risorse
- -I: simula lancio interattivo
- -h: elenca tutti i parametri

Lancio e controllo di un job

- **bqueues**
  - **Scelta della coda di lancio del proprio job**
- **bsub -q <coda> [-o <output>] <nomejob>**
  - **Il sistema risponde con il <jobid>**
  - **Il job si trova in stato PEND**
- **bjobs -l <jobid>**
  - **Il sistema dice in che stato si trova il job**
  - **Se il job è RUN, sta girando**
  - **Se il job rimanesse in PEND per lungo tempo, si usi bjobs -pl <nomejob> per avere più informazioni sulle cause, ed eventualmente comunicarle all'amministratore.**
  - **Se il sistema non dà informazioni sul job, significa che esso è terminato. Si guardino i file di output.**
- **bpeek [-f] <jobid>**
  - **Il sistema fa vedere l'output del job in fase di creazione**
- **bkill <jobid>**
  - **Un job PEND o RUN può essere terminato**