IBM
Version 1.0
9th June 1995

LoadLeveler at CERN

Tim Bell
IBM Scientific, Technical and Education
tbel@chvm1.vnet.ibm.com

Overview

LoadLeveler is a distributed, network-wide job management program for dynamically scheduling work on different processors.

This document describes the LoadLeveler installation at CERN, with particular focus on the new features added in the LoadLeveler 1.2.1 release.

- **System configuration** describes the CERN hardware configuration as it is related to LoadLeveler.

- **LoadLeveler configuration** describes the basic configuration parameters for the LoadLeveler installation at CERN.

- **Sysctl and LoadLeveler** describes a secure mechanism to allow users limited access to processes running their jobs.

- **New features** describes how the additional features in LoadLeveler 1.2.1 have been used at CERN.

System Configuration

CERN, the European Centre for High Energy Physics, provides experimental facilities for physicists from its 17 member states and other visiting scientists throughout the world.

The computing requirments of these users vary from mail and text processing to heavy compute and data intensive batch and parallel processing.

The CERN SP2 is a 65 processor Unix system using IBM RS/6000 nodes linked by an internal high speed network with a centrally managed software environment. The nodes are functionally divided into four groups of 16 each for different types of work namely interactive logins, sequential job batch processing, parallel job batch processing and data, tape and network services.

AFS is used very heavily in this environment. It provides the single system image for users' home directories and the files common to their experiments. Many software products are served directly out of AFS using symbolic links.

For more details on the CERN SP2, see http://wwwsp2.cern.ch on the World Wide Web.

Within this environment, LoadLeveler provides the following facilities

- Interactive load balancing of users across nodes in the SP2 and other Unix services on site

- Batch services for serial compute jobs

- Scheduling for parallel applications

LoadLeveler Configuration

Batch Configuration

The CERN batch configuration is designed to maximize short job turnaround while allowing the heavy CPU jobs to get good usage of the resources available.

The basic configuration uses a range of classes (short, medium, long and verylong) with a range of maximum job cpu times from 5 minutes to 6 days). An additional class, night, provides off-peak and weekend computing time on the interactive areas of the SP2 during periods of low demand. Access to this class is limited to specific users only.

Users in different experiments are defined in LoadLeveler groups which provide associated queue priorities. This allows these groups with a large computing budget to be given higher priorities. An automated procedure calculates each group's resource utilisation over the last month and adjusts their priorities accordingly. This ensures a fair allocation of CPU amongst the groups.

The processors are configured as follows

- **parallel** nodes supports a mixture of short, medium, long, verylong and parallel.
- **batch** nodes supports the same class mix as parallel. Additional paging space is available on these nodes to provide multiple jobs running per node.
- **interactive** nodes supports the night class only. The night class will only allows jobs to start after 7pm and before midnight during the week and anytime at weekends. A maximum cpu time of 8 hours ensures that the jobs are finished when the prime shift starts. This is configured using LoadLeveler's **START** expression.

```
Is_Weekend          = (tm_wday==0 || tm_wday==6)
Is_Start_Night_Time = (tm_hour>18)

START: $(Is_Start_Night_Time) || $(Is_Weekend)
```

Interactive Configuration

One of the roles of the CERN SP2 is to provide large scale interactive support for Unix users. The Interactive Session Support facility in LoadLeveler provides a name server which will return the least loaded node according to a site defined metric. This allows a user to run

```
$ telnet cernsp
```

and be given the least loaded operational node.

For CERN, the metric is based on the number of logged in users with some weighting for if they are using an Xterminal. Every few minutes, the system is scanned to evaluate

```
Xterminals*3 + Telnet*2 + Process
```

where

- **Xterminals** is the number of users logged in from an Xterminal
- **Telnet** is the number logged in via telnet or rlogin
- **Process** is the number of users who have processes running

This metric tries to balance users across the system while providing some factor for their likely future utilisation. A matric based on the CPU load average is too dependent on the current load for good balancing.

The metric also can be set to return a low priority if the file /etc/iss.nologin exists. This allows the administrator to 'drain' the interactive use of a node if there is scheduled system maintenance. When the maintenance is completed, the file can be removed and the metric will return the correct value for the node. Users will therefore see an improved availability as they will not be given a node which is just about to be shutdown.

Sysctl and LoadLeveler

For reliable operation of batch jobs, it is important to allow very limited access to the nodes which are executing jobs. However, many users want to be able to query detailed information about their job progress such as the process tree and output files other than stdout.

The sysctl program, provided with the SP2 parallel support programs, allows access to specific functions using kerberos security mechanisms. Procedures can be written which verify parameters and execute commands. The procedures are written using TCL which is a simple control language with additional facilities provided by the sysctl libraries.

The CERN users required the ability to execute the following commands:

- ps
- cat
- ls
- touch

Using sysctl, procedures were written which verified the access permissions for the user and then execute the command. To hide the complexity of the sysctl client from the user, a small wrapper was written which provides a similar interface to the standard Unix command. For example,

```
llps sp005.10401.0 -ef
```

will execute 'ps -ef' on the node where sp005.10401 is running.

This will allow all .rhosts and rlogin rights for a node to be revoked for normal users.

New Features

As part of the LoadLeveler 1.2.1 release, there are some new features which have been used at CERN during the early testing.

CPU Normalisation

Traditionally at CERN, the CPU time required for a batch job has been given in 'CERN units'. These correspond to the performance of a CPU on a set of typical high energy physics applications. They are based on the performance of an IBM 370/168 mainframe with an SP2 thin-66 node giving 25 CERNunits.

By turning on this feature, all information related to CPU time, such as job cpu limits, cpu time reported to the user and accounting, is normalised according to the speed of the CPU as configured in the LoadLeveler administration file.

If additional nodes are added which have different CPU characteristics, the speed parameters can be set differently from the rest of the nodes. The users, however, can submit using normalised cpu time limits and LoadLeveler will scale the cpu times appropriately.

Priority of jobs based on current usage

Two additional parameters have been added to the options for the job priority. These are

- **UserRunningJobs** the number of jobs a user is running
- **GroupRunningJobs** the number of jobs the group that the user belongs to is running

These expressions can be used to re-order the queue depending on current usage in the pool. The current CERN SYSPRIO expression is

```
SYSPRIO: 0 - ClassSysprio*10000 - UserRunningJobs*10000 -
             GroupRunningJobs*1000 - (QDate)
```

This will cause the priority of a job to be based mainly on the number of running jobs for a user with a factor for the running jobs for their group and the time submitted.

The priorities are recalculated according a site defined frequency which is set to 5 minutes at CERN.

Improved accounting reports

The llsummary command can now report the history of previous CPU usage based on a large variety of filters such as for a given user, group, host that the job executed on etc. In addition, the reports can be generated indexed by week, month, user, group or account. An extra option has also been provided to show throughput information. This allows calculation of queue time, wall clock time and cpu utilisation which can be used to tune the configuration to improve the service.

List of default classes

LoadLeveler now supports a list of default classes.

When a job is submitted, this list is searched until a class is found which matches the resource requirements of the jobs. For example

```
default_class = short medium long
```

will try to use the class short but if the cpu time (or any other parameter) is too high, it will try medium and then long. Users are encouraged to submit jobs giving resource limits only rather than class names. This allows some flexibility to change the definitions of the classes without the users having to change their job scripts.

Notification program

LoadLeveler now provides the ability to define the notification program for job start, completion or administrative mails.

At CERN, Zephyr from Project Athena is used to exchange messages and notify users of news. LoadLeveler can now support a site defined program which is executed to deliver the notification via Zephyr if it can or to send mail if not.

Associate LoadLeveler group with Unix group

LoadLeveler now provides an option to define a LoadLeveler group membership in terms of Unix groups. This allows the LoadLeveler group to be set to the Unix group when a job is submitted and to use the Unix group membership to limit users of groups.

CERN use this to define the LoadLeveler group membership in terms of the Unix group information. This avoids having to maintain two sets of information about groups.

llclass command

An 'llclass' command has been added which queries the class configuration parameters and current usage.

Using the llclass command, a user can quickly determine what classes are available and how many jobs could be run in a particular class. Amongst other information, this command reports the maximum number of jobs allowed to run in a particular class and the number of free slots for new jobs.

Group include/exclude for class

A class can now be configured to be only usable by a certain group.

Nice support

LoadLeveler will now set the nice priority of processes.

This is used to adjust the amount of CPU allocated when there are multiple jobs running on the same system. The Unix nice value can be used so that a short job gets a larger percentage of the running CPU than a long one. This is also used to set the priority of the 'night' class jobs very low so that they do not disturb interactive users who log in during off-peak periods.

For AIX, the importance of the nice value in scheduling decisions can be increased using the 'schedtune' command from /usr/lpp/bos/samples. This allows the relative importance of the nice value in scheduling decisions to be increased. The default parameters for the scheduling values 'D' and 'R' are both 16. Using these settings, the nice value allows up to 60% of the cpu to be allocated to the highest priority job. If the 'D' and 'R' values are both set to 2, the higher priority job can get 80% of the cpu.

As the system has become busier, this feature has been used to start very long jobs with a low priority. Short jobs are also allowed to run on the same nodes and they are allocated most of the cpu during their run. Thus, the turnaround time for short jobs is nearly as good as on an idle node and the cpu utilisation of the system is high because the very long jobs are running when there are no short jobs to be processed.

Group administrator support

Group administrators can be defined who are allowed to cancel, hold and release jobs for users in a specific group.

Multiple pool support

An environment variable can now be set which allows a user to submit, query, etc. to a different pool to the one configured for that machine.

This can be used when there are local and central clusters to allow a user to operate on both clusters without needing to log in to the central systems.

Job submission exit

At submit time, LoadLeveler will execute a filter which can perform local site specific verification and customisation. If this filter returns a non-zero value, the job will not be submitted.

This is used within the CERN environment to override the LoadLeveler defaults for parameters such as output, error, jobname etc.

User environment prolog and epilog

LoadLeveler provides two additional exits for the job which run under the user's id rather than 'loadl'. In addition, the AFS tokens, if appropriate, is valid during the execution of these steps. The ability to set environment variables for the job is provided from the prolog.

In order to assist in the management of temporary work space, the CERN user environment prologs create a scratch directory for the user to work on, copy over any files as requested in the job and return any requested files to the ouput directory when the job has completed. In addition, it will erase any left-over files in the scratch area.

A feature of the user environment prolog is that if the program prints a string of the format

```
env variable=value
```

the variable is set in the running environment of the job. This allows dynamically determined variables to be automatically initialised for the job.

AFS token extension

LoadLeveler provides a mechanism for requesting a long token when a job is started.

AFS uses the Kerberos security facilities to control access to files. When a user logs in, a token is provided which proves the user's identity to a kerberised application. After 25 hours, this token expires and the user has to re-enter their password. While this provides good security, it can cause problems in a batch environment when jobs run for a long period or remain in the queue. This new feature of LoadLeveler is used at CERN to 'revive' a token using a special secure server. A user is granted a token with a lifetime according to the length of job they submit.