The Scalable System Administrator: via C3 & M3C Tools

Ray Flanery, Al Geist, Brian Luethke[∀], Jens Schwidder[*], and Stephen Scott[Δ]
Computer Science & Mathematics Division
Oak Ridge National Laboratory
Oak Ridge, TN 37830-6367 USA
Author contact: scottsl@ornl.gov

Abstract

The computation power of PC clusters running the Linux operating system rivals that of supercomputers of just a few years ago at a fraction of the purchase price. However, the lack of good administration and user level application management tools represents a hidden operation cost that is often overlooked. This work attempts to reduce the total cost of cluster ownership and cluster use by providing a set of tools that permits the cluster to be treated as a single computer. This paper will highlight the integration of the C3 and M3C tools into a single unified cluster management solution.

Keywords: cluster tools, cluster administration

Introduction

The Monitoring and Managing Multiple Clusters (M3C) tool [1] is a web based interface application that enables a single system administrator to monitor not just one but several clusters at the same time. Furthermore, it provides a web based interface in which individual users may perform their own cluster based administrative tasks. The Cluster Command and Control (C3) tool suite [2] was developed for use in operating the HighTORC Linux cluster at Oak Ridge National Laboratory. This suite implements a number of command line based tools that have been shown to increase system manager scalability by reducing time and effort to operate and manage the cluster. The current release, v2.0, is multi-threaded and designed to improve the scalability of the C3 suite to larger clusters. The combination of the M3C interface and the C3 tools is a start to an end-to-end solution for the system administration and user administration tasks on a cluster. Due to the brevity of this paper we will not discuss implementation details of either the M3C or the C3 tools as this information may be found elsewhere. First we will look at the functionality provided by the command line based C3 tool suite. This is then followed by the discussion of how the C3 functionality is used and provided through the M3C web interface tool.

C3: Command Line Tool Suite

Eight general use tools have been developed in this effort thus far. Cl_pushimage provides a single point *push* of a system's disk image across the cluster. Cl_shutdown will shutdown nodes as specified in command arguments. Cl_pushimage and cl_shutdown are both root user system administrator tools. The other six tools, cl_push, cl_rm, cl_get, cl_ps, cl_kill, and cl_exec are tools that may be employed by any cluster user both at the system and application level. Cl_push will let you push individual files or directories across the cluster. Cl_rm will permit the deletion of files or directories on the cluster. Cl_get copies cluster based files to a user specified location. Cl_ps returns the aggregate result of the ps command run on each cluster node. Cl_kill is used to terminate a given task across the cluster. Cl_exec is the C3 general "catch all" utility that enables the execution of any command across the cluster nodes.

Cl_pushimage enables a system administrator logged in as root to *push* a cluster node image across a specified set of cluster nodes and optionally reboot those systems. This tool leverages the capabilities of Systemimager [3]. While Systemimager provides much of the functionality in this area, we felt that it fell short in that it did not enable a single point *push* for an image transfer. Cl_pushimage essentially *pushes* a request to each participating cluster node to *pull* an image from the image server. Each node then invokes the *pull* of the image from the outside cluster image server. As in all the C3 commands, the default cluster configuration file /etc/c3.conf is used by cl_pushimage if the user does not specify their own configuration file for the command. Cl_pushimage uses openssh to call the Systemimage tool updateimage on the cluster machine. Thus, effectively *pushing* an image to the specified nodes. Openssh is employed to provide secure root access to each of the cluster machines from the outside cluster image server. Of course this description assumes that Systemimager has already been employed to capture and relocate a cluster node image to the outside cluster image server machine.

Although most clusters are not frequently shutdown in their entirety, clusters that multi-boot various operating systems will most definitely benefit from such a command as will all clusters after updating the operating system kernel. Also, on those rare occasions where a cluster must be brought down quickly, such as when on auxiliary power due to a power outage, the **cl_shutdown** is much appreciated. Thus, the cl_shutdown was developed to avoid the problem of manually talking to

---

∀ Spring 2000 participant in the Office of Science, DOE Energy Research Undergraduate Laboratory Fellowships (ERULF).
* This research sponsored by the Department of Energy MICS office.
Δ Research sponsored by the Laboratory Directed Research and Development Program of Oak Ridge National Laboratory (ORNL), managed by UT-Battelle, LLC for the U. S. Department of Energy under Contract No. DE-AC05-00OR22725.

each of the cluster nodes during a shutdown process. As an added benefit, many motherboards now support an automatic power down after a halt - resulting in an "issue one command and walk away" administration for cluster shutdown.

While cl_pushimage has the ability to push an entire disk image to a cluster node, it is too cumbersome as an application support tool when one simply desires to push files or directories across the cluster. Furthermore, cl_pushimage is only available to system administrators with root level access. From these restrictions grew the desire for a simplified cluster *push* tool, **cl_push**, providing the ability for any user to *push* files and entire directories across cluster nodes. Cl_push uses rsync to push files from server to cluster node. ***Caution* –** do not use cl_push to push the root file system across nodes. Systemimager provides a number of special operations to enable cl_pushimage to operate properly.

The converse of cl_push is the **cl_get** command. This command will retrieve specified files from each node and deposit them in a specified directory location. Since all files originally have the same name, only from different source nodes, each file name is differentiated by an underscore and node IP or domain name appended to its tail. Whether IP or domain name depends on that specified in the cluster specification file. Note that cl_get operates only on files and ignores subdirectories and links.

cl_rm is the cluster version of the rm delete file/directory command. This command will go out across the cluster and attempt to delete the file(s) or directory target in a given location across all specified cluster nodes. By default, no error is returned in the case of not finding the target. Unlike rm, there is no interactive mode for cl_rm due to the obvious problem associated with cluster nodes asking for delete confirmation.

The **cl_ps** utility runs the ps command on each node of the cluster with the options specified by the user. Locally, each node stores output in /$HOME/ps_output. A cl_get is then issued for the ps_output file returning each of these to the caller with the node ID appended per the cl_get command. The cl_rm is then issued to purge the ps_output files from each of the cluster nodes. Note that this is an example of how multiple C3 commands may be strung together to perform a group of operations.

The **cl_kill** utility runs the kill command on each of the cluster nodes for a specified process name. Unlike the kill command, the cl_kill must use process name as the process ID (PID) will likely differ across the cluster nodes. The root user may also indicate a user name so that it may effect a kill for the specified user/process. Signals may also be used by root to effectively do a broad based kill command.

The **cl_exec** is the utility tool of the C3 suite in that it enables the execution of any command on each cluster node. As such, cl_exec may be considered the cluster version of rsh. A string passed to cl_exec is executed "as is" on each node. This provides a great deal of flexibility in both the format of command output and arguments passed to each instruction.

M3C: Integrated Web Tool

M3C presently consists of six integrated tool interfaces. The web based GUI has a middle layer composed of CGI scripts that read the data collected by - and delivers information requests to - backend tools. While all six tools are designed for remote cluster administration, four of the tools may also be used by a regular cluster user to monitor and run their jobs. The M3C suite includes: a monitor view that allows multiple users to see the real-time load and other metrics on specified nodes; a reservation tool that allows multiple users to easily see existing reservations on clusters and to reserve a set of nodes for any of the available times; a job submission tool that allows users to specify the number of nodes and other job requirements and submit this to a cluster's batch queue system; a software install tool that allows a user to install his data or programs on sets of nodes and allows system administrators to easily install upgrades or patches across multiple clusters; an administrative tool that allows the system administrator to reboot or shutdown selected nodes; and finally a partition tool for dividing a large cluster into sets of interactive and batch nodes.

The monitor view, as presently implemented, displays the load, temperature, and list of the top tasks running on the selected clusters. These three properties were chosen in order to illustrate/investigate how the monitor tool could display data graphically (load), by value (temperature), and by a text string (tasks). See Figure 1. The M3C user can specify the update frequency for the monitor view. By default the view is only updated when the "refresh" button is pressed. This provides the minimum intrusion on the web system and the cluster. A pull down menu allows the user to set an update frequency in a range between 1 second and 1 minute. The update frequency set in the M3C tool just defines how frequently the user views are updated; it is independent of the frequency that the cluster updates the monitor file. One reason they are independent is because multiple M3C tools can be running at the same time, each with different update frequencies set. Another reason they are independent is to prevent a M3C user from adversely impacting other users' cluster performance by setting a high update frequency. Backend scripts and daemons running on the clusters collect the information displayed in the M3C monitor tool. Load, temperature, and running processes are presently displayed but other node properties can easily be added to the M3C monitor view.

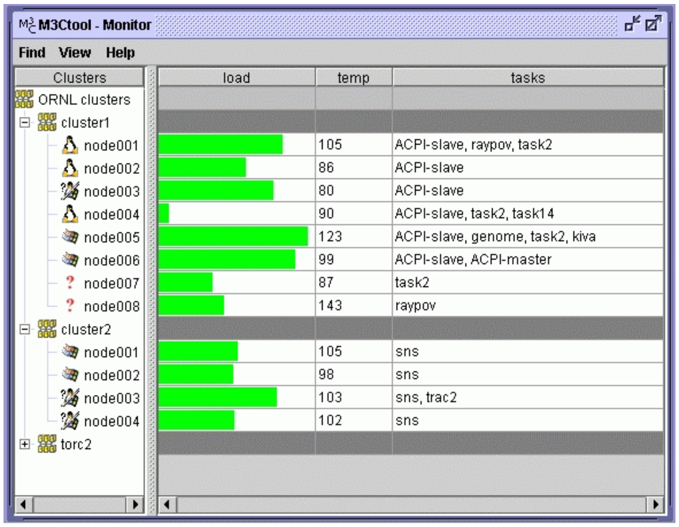

Figure 1. M3C Monitor GUI.

Presently if a user wants to run a simulation on a dedicated block of cluster nodes, they must contact the system administrator to coordinate the request with other users' requests for dedicated time. The reservation tool in M3C provides a way for multiple users to see what times and which nodes are available on the cluster and enables the user to make reservations for any times that are not already booked. Figure 2 shows a screenshot of the reservation tool. All reservations made by others appear in yellow, while all reservations made by this user are delineated in green. The tool allows nodes to be blocked out across multiple clusters if the user needs more nodes of a certain type than exist in any one cluster. System administrators can use this tool to block out and coordinate times for backups, system maintenance, or even replacing the operating system on one or more clusters.

In order to help the user find which and how many nodes have particular properties in the clusters managed by M3C, the tool suite includes a "Find" function that works with all six tools. In the case of reservations, the user's application may need nodes with a certain amount of memory or perhaps with a particular software package installed. System administrators may need to apply a patch to all nodes with a certain version of the OS. The Find function highlights (selects) all the nodes that match all the user specified requirements. M3C has no knowledge about the cluster nodes beyond what the cluster owner is willing to share either statically in the description file or dynamically from monitors.

To use the reservation tool, a set of nodes from the "tree" view is selected. Then the time block when these nodes are desired is specified by left and right clicking in the time view. When the reserve button is pressed, the request is sent to the CGI script(s), which checks that another user has not reserved the same nodes in the last few moments. If the nodes are still available, then the reservation tool colors the reservation green and places the user's name in the block. If the nodes are not available, an error is returned and the user can press the "refresh" button and see the latest reservation data and adjust the request accordingly. If simulation requirements change, the M3C tool allows the user who made the reservation to "unreserve" the selected nodes, which then go back into the interactive pool for other users.

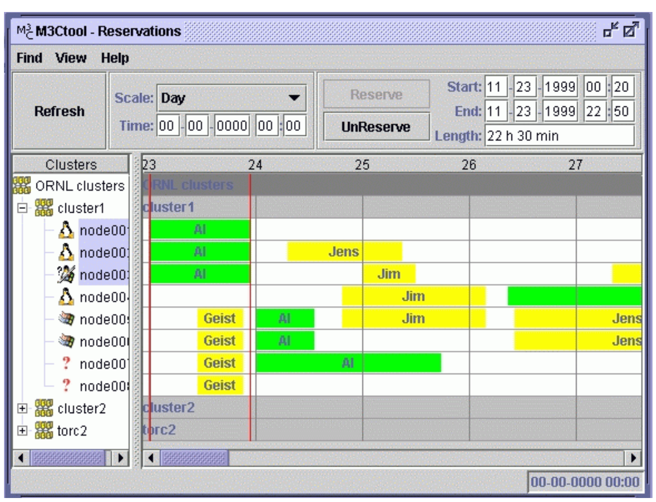

Figure 2. M3C Reservation GUI.

A common way to share resources on a large cluster is to run a batch scheduler. M3C includes a tool to collect information for parallel batch schedulers such as PBS and Condor. The submit job tool is shown in Figure 3. The user supplies the job name or path to the executable, and any requirements for the job. For example, does it have to run on a particular cluster? How many nodes does it require? Are there any special node requirements? In Figure 3 the climate simulation job PCCM3 is allowed to run on any cluster as long as the nodes are running Linux. Other requirements may be added. The tree view is not used in the submit job tool except as a means to find out particular properties that the user may want to specify in the job requirements. (Double clicking on a node in the tree view brings up a small information box listing all known properties of that node.) Which nodes that the job runs on is left to the discretion of the batch queue system – selecting nodes in the tree view is ignored.

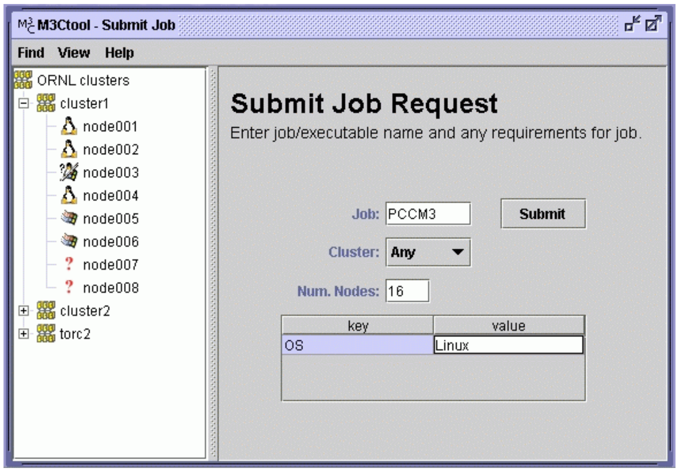

Figure 3. M3C Scheduler GUI

Software installation from system kernel to application to data set may be performed via the M3C GUI as shown for PVM in Figure 4. This is very useful to the system administrator who needs to install the latest security patch to the Linux nodes in all the clusters for which he is responsible. This tool is also useful to the scientist who wants to preload data files or executable on a set of cluster nodes before starting a simulation. A kernel installation may use the cl_pushimage to change operating system and programming environment for the entire cluster or even a partition of the cluster. The mirror button indicates to M3C that it is to perform an image push across the specified nodes – effectively mirroring the systems. A cl_shutdown with reboot option will also be performed in order to activate the newly installed operating system. The cl_push is employed for the simpler task of installing individual applications or data sets across the cluster. The M3C user will select nodes where the install is to take place, provide the installation destination path, and the local file or directory for installation. Optionally, the installed package may be executed via a GUI button.

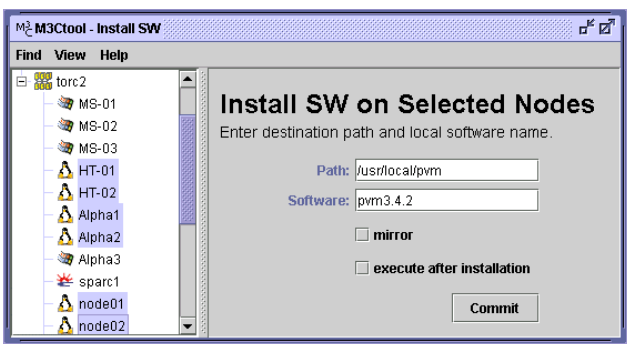

Figure 4. M3C Install Software GUI.

A number of system administration tasks must be performed multiple times across a cluster in order to manage the system and applications. For example, to simply shutdown a 64-node Linux cluster without special tools or hardware, a system administrator will have to login to each individual node and perform a controlled system shutdown on each. The cost of not doing so is at best suffering through the Linux system integrity check at the next startup and at worst it is the loss of data that was cached immediately prior to the uncontrolled shutdown. With M3C, system administration tasks such as reboot, shutdown, add user, and delete user may be performed with the click of a single mouse button via the system administration GUI – Figure 5. The use of this tool is similar to the other M3C tools in that the administrator selects a set of nodes and then presses the function to be applied to those nodes. This request is then handed off to the C3 tools to effect across the cluster. The "add user" option produces an additional dialogue box that must be filled before handing off to C3.

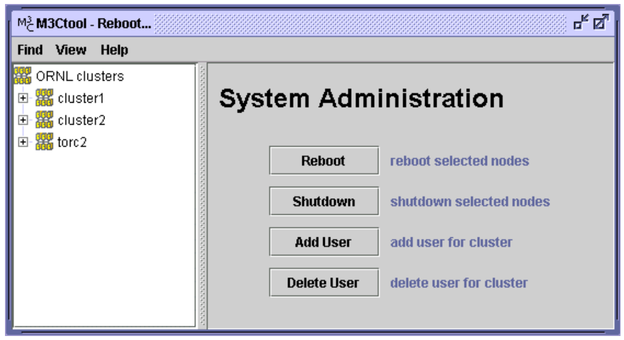

Figure 5. M3C System Administration GUI.

While small clusters are often used as a dedicated computation resource, larger clusters are often partitioned into sub-clusters, resulting in multiple computing resources. Some partitions may be dedicated to batch jobs, while others are grouped for interactive use, and still others may be grouped for a dedicated single-user application run. Figure 6 shows the M3C partition GUI where a cluster may be partitioned for single user, interactive, and batch runs. Where an interactive run is considered the default environment, the single user run will deactivate all other node users by locking their account access via a cl_push of a temporary password file excluding other user access. These partitions then appear in the M3C reservations time view so that users can easily see what parts of the cluster are available for interactive use.

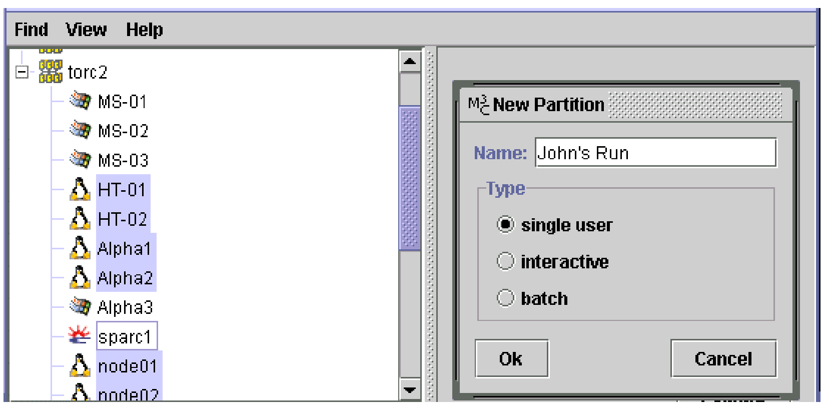

Figure 6. M3C Partition GUI.

Conclusion & Future Work

With the computation power of Linux based PC clusters rivaling that of supercomputers at a fraction of the purchase price, we have found that there is a significant hidden administration cost that is incurred when operating the requisite large number of cluster nodes. While individual users have made attempts to resolve this problem for their smaller internal clusters, we are now just seeing a concerted effort in the cluster community to cooperate in resolving this problem with open source solutions that will be made available to all cluster owners.

The combination of the M3C GUI and the backend C3 tool suite is our initial effort to produce a seamless scalable system administrator for cluster computing. We are presently using these tools to administer our TORC and HighTORC clusters. The C3 and M3C packages may be downloaded from http://www.epm.ornl.gov/torc. We plan to continue our effort in area of cluster tools as we gain experience in the administration and operation of our clusters. Another area to tackle in the future is to provide the same environment and functionality for NT & Windows based clusters. We also expect to deploy the M3C/C3 tool as a part of the Cluster Community Development Kit. CCDK is an open source effort spanning government research labs, academic institutions, and computer industry leaders with the intent to produce a "pre-packaged" cluster environment CD. Additional information on CCDK may be found at http://www.epm.ornl.gov/ccdk.

References

[1] A. Geist and J. Schwidder, Managing Multiple Multi-user PC Clusters, http://www.epm.ornl.gov/~geist/M3C.pdf
[2] R. Flanery, A. Geist, B. Luethke, and S. Scott, Cluster Command & Control (C3) Tools Suite, http://www.epm.ornl.gov/~sscott
[3] Systemimager source and documentation, http://www.systemimager.org