;login:
THE MAGAZINE OF USENIX & SAGE
August 2001 • Volume 26 • Number 5

Special Focus Issue:Clustering

Guest Editor: Joseph L. Kaiser

inside:

USENIX & SAGE
The Advanced Computing Systems Association &
The System Administrators Guild

the RHIC computing facility linux farms

by Tom Throwe

Throwe is the Deputy Head of RHIC Computing at Brookhaven National Laboratory (BNL) on Long Island New York. He received a Ph.D. in Nuclear Physics from Indiana University in 1984 and has been at BNL since 1987.

throwe@bnl.gov

The Relativistic Heavy Ion Collider (RHIC) at Brookhaven National Laboratory began colliding gold ions in June of 2000. The four experimental groups situated around the collider ring came online with the accelerator and collectively recorded 22 terabytes of data during that first run of the machine. RHIC began the second run on the machine in June of 2001.

The data from the RHIC experiments is sent via a gigabit Ethernet link to the RHIC Computing Facility (RCF), where it is stored, reconstructed, and analyzed. Currently, the computing facility consists of four StorageTek tape silos with a capacity of 1.2 petabytes of data, twelve Sun E450 servers serving 50 terabytes of RAID storage, and 422 dual and quad CPU farm machines with a total computing power of approximately 18,000 SPECint95. The facility is expected to grow to between 6 and 10 StorageTek silos, 100 terabytes of disk, and a steady state of approximately 600 dual CPU farm machines.

The Linux farms at RCF started out in 1996 as 10 4U quad CPU rack-mounted systems. Trying to exploit the "commodity" market, the farm was augmented with 24 "desktop" machines arranged on shelving in four racks in 1997. Another 24 "desktop" machines were purchased in 1998, but these had to be laid down on their sides to fit eight in a rack. As we approached our first major purchase, it was clear that we could not use "desktop" machines on shelves, so we only looked at rack-mounted systems and purchased 148 2U rack-mounted machines in 1999. The last purchase was made in late 2000 and consisted of 160 2U dual CPU machines. We are presently preparing for our next purchase, which was originally expected to be of comparable size to our 2000 purchase, but in a 1U form factor. The 1U size is attractive since we will soon be short of space, but our user base is looking toward a locally attached disk, which is limited in a 1U form factor. Purchases beyond our next purchase will involve the retiring of old machines to make space for the new higher performance machines. We are planning on a three- to four-year life cycle of the farm machines.

Figure 1: A picture of part of the RCF Linux computing farm at BNL. Two of the networking cabinets can be seen in the background.

The Linux machines at RCF form a "cluster" only in that they are configured and managed together. The machines do not share resources (other than the network), and no parallel jobs run on the machines. The code running on the machines may be the same, but the running jobs do not communicate with other jobs on the same or different nodes. Whether or not this style of computing continues into the future will depend on the needs and manpower of the RHIC experiments.

The machines are divided into two farms, namely a Central Reconstruction Server (CRS) farm and a Central Analysis Server (CAS) farm. Each of these farms is further divided by experimental group. The division by experiment is at the network and machine level such that one experiment cannot have an impact another experiment's resources. Each of the experiments has a large packet engine switch with a gigabit input link and 100 megabit switch ports to the individual machines. Each farm machine is on a separate switch port. Thus an experiment's server machines are connected to the switch via gigabit, and the server traffic is fanned out to the farm nodes on the 100 megabit network. Our division of resources by experiment leads to inefficiencies in usage of the network and machines, but it eliminates any impact of one experiment's possible poor usage of their resources on the other experiments' resources. So, in the end, the Linux machines are logically divided into eight farms where physically there is only one.

Data first flows into the CRS. The purpose of the reconstruction farm is to take the raw data collected by an experiment and convert it to "physics" quantities that the individual

physicists can then subject to analysis. General users have no direct access to the reconstruction farm machines. One or two members of each experimental collaboration have access to a job-submission account and software. The job-submission software for the reconstruction machines was written in-house mainly due to the need for a custom interface into the HPSS mass-storage system, which manages the data in the StorageTek robots. A job-control file is produced by the authorized user (since a very large number of jobs will be processed, the job-control file is not produced by hand, but by an automated method coupled with the sinking of the raw data) and submitted to the system. The input files necessary for the job are staged from tape and put into the HPSS disk cache. When all input files for a particular job are available, the system finds an available farm node and sends instructions to that node to initiate the transfer of the input files from the HPSS cache to the local disk of the farm node. When the transfer is complete the system sends a message to the farm machine's daemon, which determines if the job is allowed to start. Since the machines have dual CPUs, in the steady-state situation, each node will have two running jobs and one job waiting to run with the input files already resident on the disk. When a job is finished, depending on its exit status, the system will either transfer the output files back into HPSS and clean up the local disk, or on failure, it will just clean up the local disk. The user is then informed of the final status of the job. Database entries are made as the job passes through the system so that statistics on time spent in each stage of the process can be gathered as well as statistics on the final status of each job.

During the running of reconstruction jobs, the user has access to machine and job status through a Graphical User Interface (GUI) written in Perl/Tk. The user can query the status of an individual machine or groups of machines, and he or she can disable and enable machines to stop or allow jobs to be queued to those machines. Similarly, the user can query the status of jobs running on the system and can kill, suspend, or resume those jobs. Nonprivileged users can query information about machines and jobs in the reconstruction farms, but they cannot affect the running of the system.

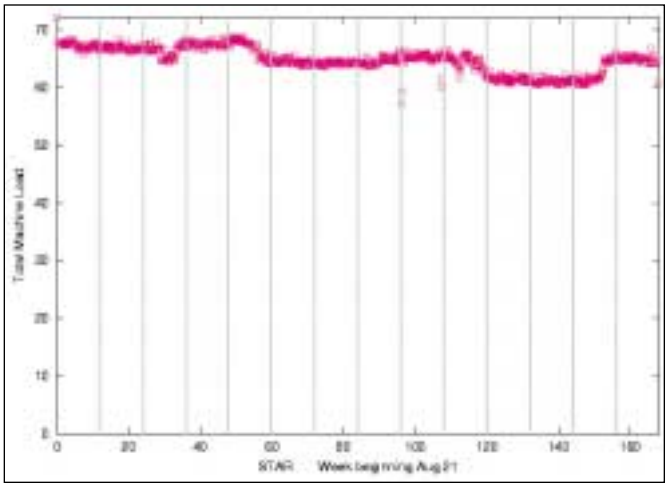

Figure 2: A plot showing the type of steady-state load achieved on the Central Reconstruction Server (CRS) farm at RCF. The STAR experiment's portion of the farm at the time of the plot was 33 machines, so a fully loaded system would have a load of 66 plus the load due to pre-staging a job on each node.

The experiment's code that is run on the reconstruction machines is written by a number of collaborators within the experiment. After being certified by the experiment, the code is integrated into the experiment's reconstruction package and a schedule for reconstructing the data is agreed upon within the collaboration. Due to the amount of time needed to do a reconstruction pass of the current data set, the experiment is unlikely to be able to make more than one such pass on the bulk of the data, so each experiment has a quality-control mechanism in place to try to ensure that the code is both robust and producing accurate results.

Since the RCF reconstruction farm has such restricted access, the load on the machines is fairly steady, as can be seen in Figure 2, and the average uptime of the machines in the CRS farm is rather high and measured in "hundreds" of days rather than "tens" of days. In general, the reconstruction farm machines are only brought down for deliberate maintenance.

The Central Analysis Server farms at the RCF are used somewhat differently than the reconstruction farms. The analysis machines allow batch-only access or batch and interactive access. Since the hardware is segregated by experiment, each experiment group is free to decide on a usage policy. Most groups have decided to allow both batch and

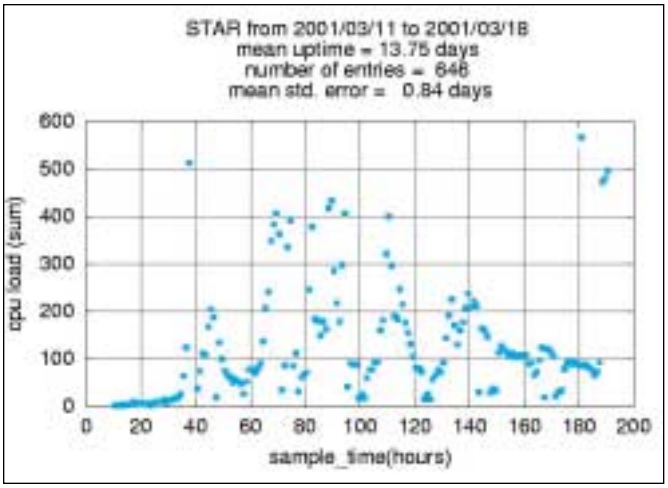

Figure 3: A plot showing the load on the STAR experiment's portion of the Central Analysis Server (CAS) farm at RCF. The load is produced by interactive and batch jobs and shows much more variation than the load on the CRS farms.

interactive access to their CAS machines, but some have designated a subset of their CAS machines to be batch only. This arrangement requires somewhat more work on the part of the RCF staff, but the configuration is fairly static and usually only has to be done once.

The batch system used in the RCF on the CAS farms is the Load Sharing Facility (LSF) from Platform Computing (*http://www.platform.com/platform/platform.nfs/webpage/lsf/*). The system has been found to be quite flexible and allows for the experiments' batch queues to be kept separate and individually configured. The LSF system is quite powerful and allows for the management of a number of system resources, but we essentially only use its batch queueing system.

Unlike the CRS farms, where access is restricted and the jobs are strongly controlled, the CAS farm machines can often get overloaded with users. Figure 3 shows a fairly typical fluctuation of the load on an experiment's part of the CAS farm with time. These large fluctuation loads make for a much less robust farm. The average uptime of these machines then varies by usage pattern among the experiments and by allowed access methods. The average uptimes range from days or weeks for some interactive nodes to months for some of the batch only nodes. In general, the average uptime of the CAS nodes is much less than the average uptime of the CRS nodes, and the downtime is more likely to be caused by user activity on the CAS machine than on the CRS machines.

Monitoring of the RCF Linux farms exists at a number of levels. General connectivity monitoring, farm-specific monitoring, service monitoring, and hardware monitoring are all done. The Mon program (*http://www.kernel.org/software/mon/*), because of its ease of use and extensibility, is presently used for connectivity monitoring, with any detected machine failures going to the Linux farm group's pagers and monitoring Web pages. A heartbeat from the server daemons on the CRS machines is coupled to the CRS control software to keep jobs from trying to be queued to machines with problems. Load, NFS, AFS, and LSF monitors on the CAS machines also send alarms to the Linux farm group's pagers and another monitoring Web page. The CRS and CAS-specific monitoring software was written in-house, mainly in Perl and Python. Hardware status, including temperature, voltages, fan status, and disk errors, is monitored with the VACM software from VA Linux (*http://www.valinux.com/software/vacm/*). The VACM software also allows for the remote reboot and power cycle of any of the farm machines and has greatly enhanced the manageability of the machines.

Up to now, the operating system on the farm machines has been installed using a BOOTP network boot. In this process, the OS is first built on a development machine and transferred to an installation directory tree. A compressed tar image is made of the directory tree, and this image is copied to the BOOTP servers. In the case of a new machine, the network boot procedure is initiated from a floppy disk, while a rebuild of a system can be initiated from a swap of the kernel by replacing `lilo.conf` followed by a reboot. The initial BOOTP connection provided the network parameters for the booting machine and the NFS file system from which to mount the machine's initial root file system in memory. The local disk of the machine is then partitioned, and the compressed tar file of the OS image is untarred onto the local disk. The local network files are customized with the network parameters obtained during the boot process, LILO is run on the local disk, and the machine is rebooted. This first local boot of the machine

runs a customization script that configures the machine as either a CRS or CAS node and then does the final reboot of the machine. When the machine comes up this third time, it either automatically registers itself as a CRS node, or comes up as a CAS node with LSF running. The entire process from initial BOOTP start to final configuration takes about 10 minutes plus the time to build the AFS cache. With the distributed BOOTP servers in place, we have brought up an entire rack of machines at the same time without a problem.

Major upgrades to the system are currently done by producing a new system image and redoing the installation. Minor upgrades are done by distributing the file or rpm via scp, NFS, or AFS and initiating the upgrade by distributing the appropriate command to the nodes via SSH. However, both the initial installation method and the upgrade methods currently used are proving to be inadequate. The installation method has evolved over time and is much more flexible now than when we started, but the hardware is becoming more diverse and the users are demanding more customization, so we are looking at other methods. The "KickStart" system is actively being tested and the "SystemImager" of VA Linux is also being investigated. The goal is to move to a more customizable installation and upgrade system than we currently have.

As noted previously, we are not at the full capacity of the facility. It is expected that we will reach a steady state of approximately 600 dual CPU machines with a replacement cycle of three to four years. All farm CPU purchases to date were of machines with minimal local storage (enough for two running jobs plus a staged waiting job on the CRS machines and a comparable or somewhat larger amount on the CAS machines). Unfortunately, the experiments' need for disk space is outstripping the budget we have for centralized RAID. The experiment groups therefore want to embark on a study of using disks local to the farm nodes to augment the centralized disk. Such investigations of distributed disks are also going on at other institutions. The argument is that by diverting a relatively small amount of the centralized disk budget, a larger amount of non-centralized IDE disks can be attached to the farm CPU than the amount of centralized disks the diverted funds would have purchased. The problem is then shifted from managing and serving data from a large centralized disk to one of managing and tracking the data on a large number of distributed disks. Everyone agrees that keeping track of where the data is in a distributed system and managing the loss and replacement of data files due to disk failures is a difficult problem, but enough people see this as the only way even to approach the amount of disk space they feel they need.

The Linux farms at the RCF have performed well to date. We have had a total of one disk failure, one CPU failure, three motherboard failures, and eight power supply failures in the entire farm. The rack-mounted machines have proven to be quite reliable and maintainable. There is some apprehension – as we add IDE disk drives to some of the existing machines and anticipate future purchases of machines fully loaded with IDE disks — that the increased number of disks and the accompanying increased heat load will increase the failure rates of the machines. Everyone agrees that the machines should work at their maximum configuration, but since we have not run any machines in that configuration, we will reserve judgment until we do have such experience. Our goal is to provide a robust and manageable facility that meets the needs of our user community, and we will work with our users as we evolve the facility to meet that goal.

Keeping track of where the data is in a distributed system and managing the loss and replacement of data files due to disk failures is a difficult problem