Node Control System

A Cluster Management Tool for Teaching and Research

Alex Fedosov

Keck Cluster Group

Department of Computer Science

University of San Francisco

Motivation

- Most existing cluster tools are batch-oriented
  - Users submit jobs
  - Scheduler runs a job when nodes are available
  - Results returned to the user
  - e.g. PBS (OpenPBS)
- Our main goal is interactive development
  - Batch system have very poor support for interactive tasks
  - OpenPBS only allows one user at a time to use nodes interactively
  - The goal of NCS is to provide excellent support for interactive development and primitive batching

Basic Usage

- NCS keeps track of node allocations to users
  - User checks out some nodes
  - Logs into nodes
  - Tests, debugs, runs MPI code
  - When done, user logs out, and checks nodes back in
- NCS maintains a record of which nodes are checked to which user and for how long
- Nodes "expire" after a certain period of time

Features

- CVS-style interface
  - checkout
  - checkin
  - login (checkout with automatic login into a node)
- Node Selection
  - all nodes
  - any n nodes
  - specific list of nodes

Features (cont.)

- Automatic .machines file generation
  - MPI jobs require a .machines file to know on which nodes to run
  - NCS automatically generates a proper .machines file when nodes are checked out and updates it as needed
  - A different .machines file is generated for every node group, node domain, or login session
- Node Groups
  - A user may want to run two different jobs at a time
  - NCS can separate nodes into separate logical groups
  - Each group gets its own .machines file

Features (cont.)

- Node Domains
  - Users may want to have a specific operating system or node configuration for their research
  - NCS allows separation of nodes based on their configuration (heterogenous setup)
  - e.g. we can have a domain of nodes running a different distribution of Linux, or a different OS altogether, but we don’t want these nodes to be allocated for normal MPI jobs

Features for Kernel Hackers

- Easy installation of custom kernels on nodes
  - Booting done via DHCP and network GRUB
  - Kernel needs to be copied into proper location
  - GRUB configuration for the node(s) needs to be modified
  - NCS does all this automatically
  - Configuration restored to default at checkin
- Serial Logging and Access
  - If kernels are built with serial console enabled, all kernel messages are captured and saved in a rotating log
  - Any node can be accessed over serial line

NCS Plugins

- New functionality can be added to NCS via plugins
  - Allows site customization
  - Loaded at runtime (no need to modify NCS itself)
  - Simple API allowing easy custom extensions

Our Site Customizations

- Node rebooting (using BayTech HW)
- Automatic .rhosts file generation (for authentication)
- Process cleanup (kill user’s processes at checkin)
- Access to GRUB configuration for checked out nodes

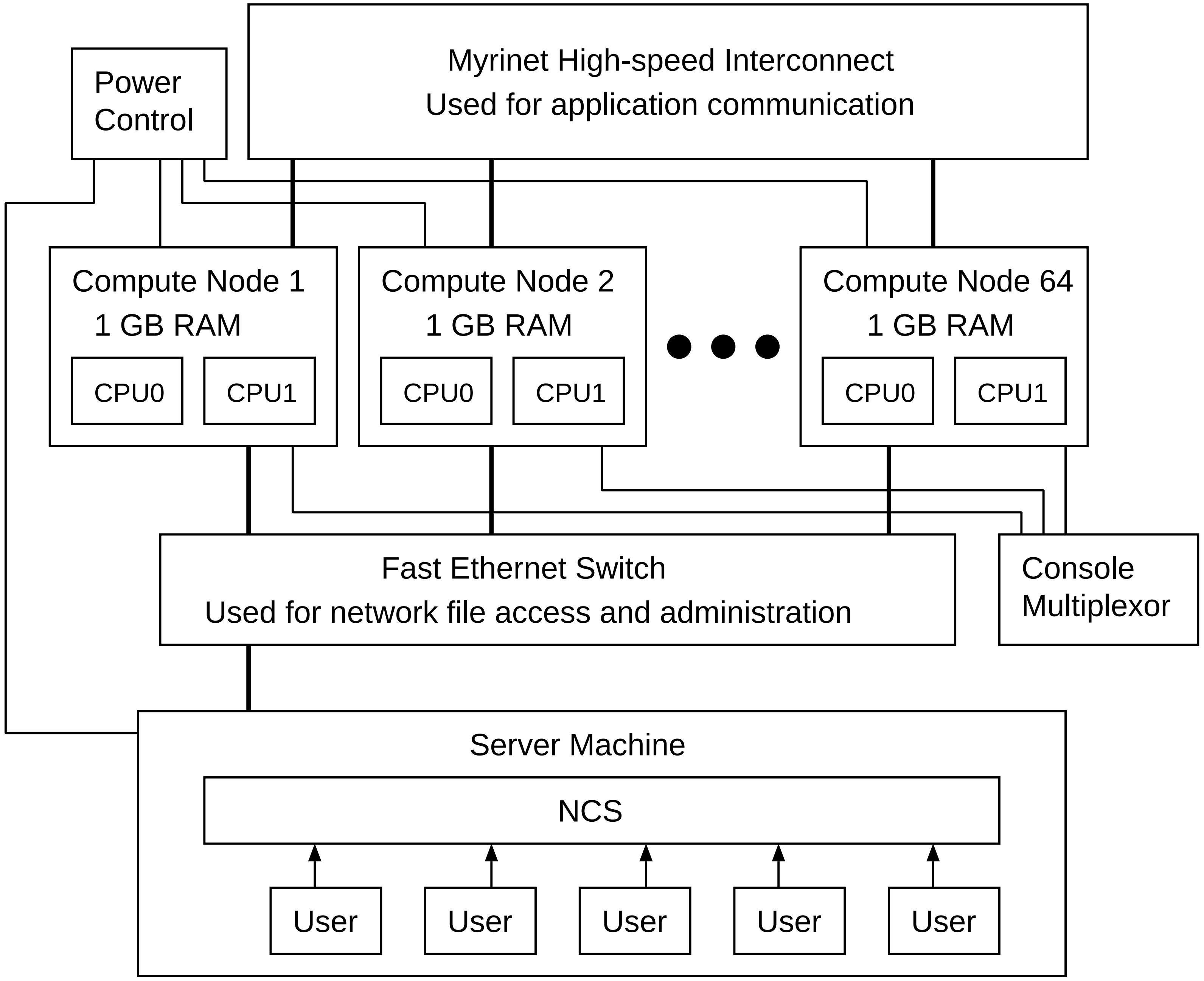
NCS daemon

- A daemon that runs in the background checks for expired nodes
- If a node is expired, e-mail is sent to the node owner, asking to renew or check in the node
- If no action taken, node is checked back in

Implementation

- Currently written in C (4000 lines of code)
- Uses a flat file to save node information
- Plugins implemented as dynamically-loaded libraries

Cluster Diagram



Conclusion

- Currently in use on the Keck Cluster
  - Research / Teaching Environment
  - 64 dual-processor nodes
  - 2 years of use
- Future Work
  - Rewrite in Python
  - Use a SQL database backend
  - Add primitive batching/queueing
  - Web Status Interface