Future Directions for the Scyld Beowulf Management System

Cluster 2001

Scyld Computing Corporation
http://www.scyld.com

The Scyld Beowulf System

- A standard, supported Beowulf cluster operating system
- Targeting deployment of complex applications
- Convenient development environment
- Supported tools, applications and training available

Scyld Beowulf Essential Elements

- Operating system designed for clustering
- Front-end "Master" and compute "slave" nodes
- Easy installation and administration
- Broad device support
- Multiple booting systems

Scyld Beowulf Feature Overview

- "Install once, execute everywhere"
  - Only one system to install and update
  - Single point administration
  - Software version skew has been eliminated.
- Scalability
  - Dynamically adding compute nodes is fast and automatic
  - Scalable to hundreds of compute nodes
- Administration and User Interaction
  - Very similar to a single machine
  - Diskless administration model

Scyld Beowulf Software

Open Source software infrastructure

Based on Linux

Runs most existing Linux applications

Growing tool and application base

Design Philosophy and Goals

Users

- Application users should not need to know they are on a cluster
- Sophisticated users should require little new knowledge

Administrators

- Simplicity
- Minimal new cluster-specific tools
- GUI guides understanding

Developers

-

System Model

- "Master" front-end and "Slave" compute nodes
- Booting and configuration controlled from a master
- Single System Image model

- Front End "Master" system
  - A full operating system installation
  - Supports user login
- Compute "Slave" systems
  - Have full kernel
  - Only a minimal file system
  - No required executables!
  - No user, or hacker;-), login
- Processes are started with a remote execution

Scyld Kernel Component: BProc

- BProc is the Beowulf distributed PROCess space
- The mechanism to start processes on a compute node
- BProc has elements of
  - process migration
  - remote fork
  - checkpoint/restart
- Connection maintained to the Master node

How can this be Fast?

- Cached libraries ("VMA regions") on target machine
- Copy only changed pages in known VMA regions
- Copy unknown VMA regions

... the system performance depends heavily on caching.

Evolution of Library Caching

Current Scyld Beowulf system

- Static library cache
- Library file set is created and known by the master
- File set is transferred once at boot time

Dynamic Library Caching

- Move to general file object caching
- Allows caching of executables as well as libraries
- Copy pure pages as well as modified pages
- Using swap or filesystem for caching

Loading Dynamic Libraries

Some applications load dynamic libraries

- Examples: Perl, X11 v4
- This currently require local files

Approach

- Replace dlopen() implementation
- Check for cached or mounted version
- Fall back to contacting the master

Security and Access Model Improvements

Existing system (approximately)

- Single master
- Compute nodes will accept all jobs

Evolution of Scyld Beowulf

- Capable of supporting multiple masters
- User/group node permissions
- Masters may have disjoint users
- Implementation semantics tested by multiple masters

Other Caching Features for Scyld Beowulf

Caching name lookups

- Avoid network name lookups
  - Cost in delay, and serialization
- Uses the Linux NS-switch mechanism
- Slave nodes only need know the current users/groups
  - Pass the user/group/cwd with each job
- Cluster nodes can use simplified hostnames
  - ".0" is the first node
  - ".23" is the 24th node
  - ".-1" is the master

Future work? ("Teasers")

Kernel Checkpoint / roll-off / restart

- BProc is a good start

Tree-structured process control for large-scale systems

Self-similar reporting of activity

Additional cluster filesystems

Summary

The Scyld Beowulf system depends heavily on caching
The system can be improved by better caching strategies

We are continuing to evolve the system to
- minimize the tuning burden
- maximize the scalability

Visit our demos in vendor exhibit area