CUG 2008 HELSINKI • MAY 5–8, 2008

CROSSING THE BOUNDARIES

Compute Node Linux (CNL)
The Evolution of a Compute OS

CRAY

THE SUPERCOMPUTER COMPANY

Overview

- CNL – The original scheme – plan, goals, requirements
- Status of CNL
- Plans – Features and directions
- Futures

CNL - The Original Statement of Purpose

- CNL is the name of the multi-year program to use an optimized Linux-based kernel for the computational nodes. It will be delivered in a phased approach starting with a limited availability in early 2007 which will demonstrate the concept and stability on smaller systems and continue with releases which show performance at scale

- CNL is a component of the Cray Linux Environment (CLE)

Basic Direction for CNL

- CNL is an initial implementation of an architectural component that can allow many varied and different compute OSes to run within a Cray system
  - The fundamental aim is to fit the compute node OS to the applications that run on scalable systems
  - There is no general purpose OS that is a single answer to all issues at the compute node level
- Linux is a great starting point for differentiation of Compute Node OSes. It is possible to provide several variants to support different application workloads and to allow customers or service to experiment with others
  - CNL variants will validate the direction and help Cray understand the interest in specialization long term

CNL is only a component of the system and relies on the rest of the system infrastructure

Basic Direction for CNL

- The initial focus of CNL has been – and will continue to be - light weight compute node OS for scalability
- The larger plan has components that are deliverable across a series of releases
- CNL
  - Improves support for a wider set of processor and node architectures
    - Lays a shared foundation for current and future Cray products
  - Improves support for a wider set of applications (including ISV)
  - Includes support for other programming models

Compute Node OS Objectives

- Provide OS environment that supports HPC applications
  - Scalable
  - High Performance
  - Reliable
  - Extensible
  - Robust
- Establish common software infrastructure for future Cray systems based on Linux and Linux derivatives
  - Commodity scalar processor based platforms
  - Proprietary vector processor systems
  - Hybrid systems
- Increase market flexibility
  - Adapt quickly to market requirements
  - Run (scalable) ISV applications

Compute Node Linux “Initial” Requirements

- The initial requirements for CNL are based on Catamount functionality and the need to scale
  - Scaling to 20K compute sockets
  - Application I/O equivalent to Catamount
  - Start applications as fast as Catamount
  - Boot compute nodes almost as fast as Catamount
  - Small memory footprint
- Support N-way cores (and then multiple sockets)
- Improved application portability
- Support for multiple programming models including:
  - MPI
  - SHMEM
  - OpenMP
  - Global Arrays/ ARMCI
  - PGAS Language Support (CAF and UPC) for future systems

Linux versus other compute node OSes

- Microkernels
  - “Lightweight”
    - Small memory and cpu cycle footprint
    - Relatively deterministic
  - Usually tuned to the workload
  - Track record of “scalable” for production environments and workloads
  - Isolated development communities and usually lagging support for hardware advances

- Linux
  - Full-featured for wide application space
    - Shared memory and threads for OpenMP
    - Python, Sockets (TCP), dynamic libraries, other file systems
  - Provides compatibility and portability
  - Community supported
    - Faster access to new hardware support and new features
    - The community goals are not always aligned with HPC’s goals
  - Limitations to scaling? File system based OS and limited scale services

Cray Decision Points

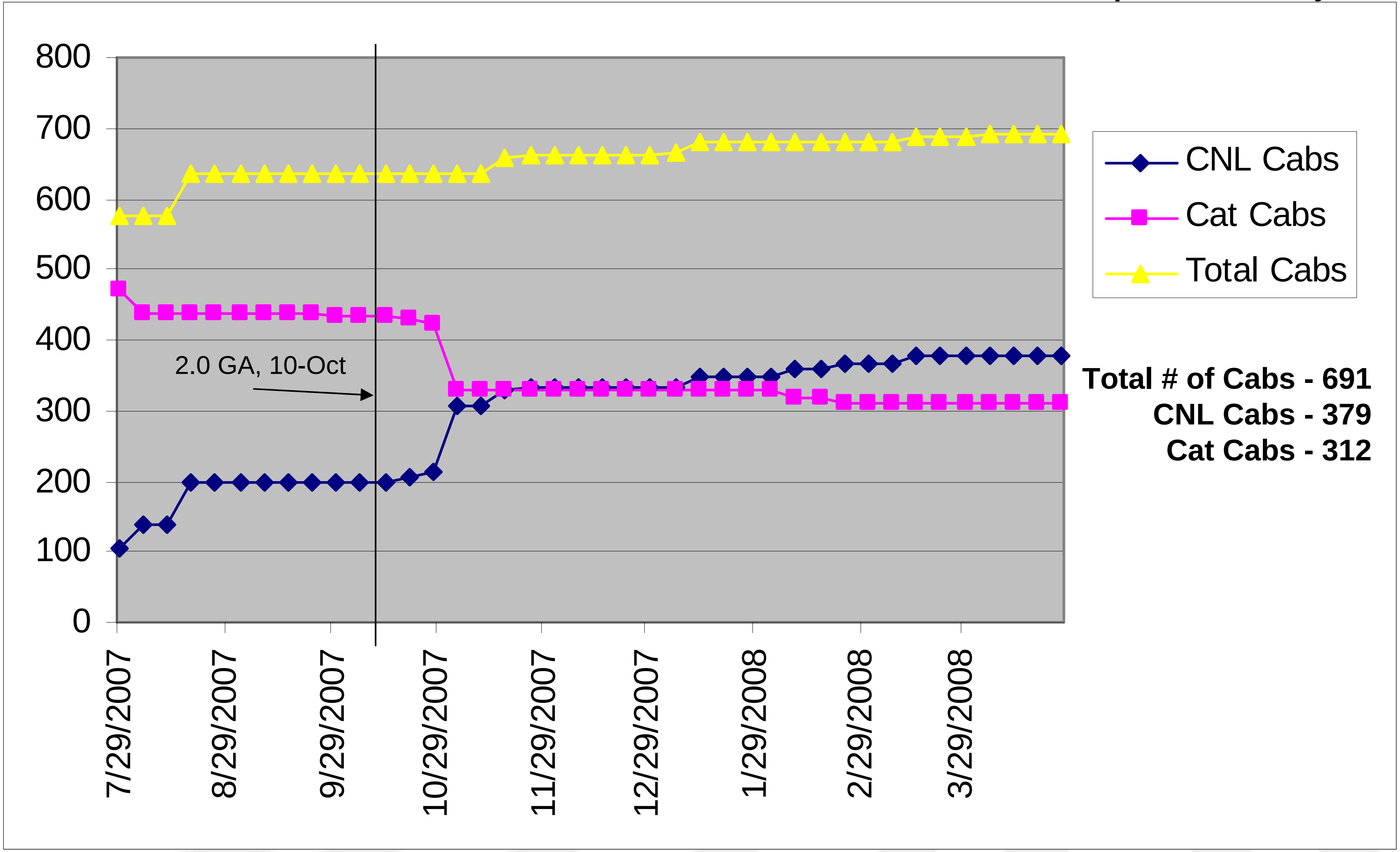
- Significant effort necessary to make a microkernel (or microkernels) able to provide the breadth of services our customers require
- Significant efforts would be required to keep up with the hardware changes in the next 3 to 5 years
- Initial Linux experiments indicated that it could be made to work
  - MPI applications still scaled or could be tuned to scale
  - Additional applications and programming environments worked
  - Support for scaling node core counts comparable to microkernels
- This is not to say that microkernels could not be made to serve the needs of a wider application base and more complicated node architecture, just that there seemed to be an easier way that met all of the requirements

CNL Functional Progress

- Meeting requirements
  - Scale at ~20K nodes
  - Quad Core
  - OpenMP
  - NUMA Node
- Running on more nodes than the alternative
  - Multiple customers
- Better support for ISV applications over time
  - Multi-phase project
  - This is mostly a configuration/image management issue
  - Requirements for ISV applications vary:
    - Different libraries, different OS services, etc
    - Developing infrastructure to allow custom Application Partition
    - Dynamic libraries*

The Rise of CNL

Last Updated: 03-May-08



Total # of Cabs - 691
CNL Cabs - 379
Cat Cabs - 312

CNL - Migration tipping points

- Why migrate? The rationale would be -
  - Move to QuadCore or XT5
  - Network access from compute nodes
  - Applications are making small I/O requests <.5MB
  - Applications require OpenMP
  - Applications are doing a lot of core to core communication

CNL - Features to be released and planned

- Hardware
  - Quad Core – tuning improvements
  - NUMA Nodes
  - Huge Pages
- External Network Access (RSIP)
  - Improve socket access to external network – load leveling and failover
- Fat Penguins – (ISV application support)
  - Configuration – provisioning support for compute nodes
  - Allocation – Attributes covers most of the requirements
  - Purpose was to allow more general purpose system (more functionality) at a smaller scale. So far this functionality is mostly covered by dynamic libraries…

CNL - Futures

- Scaling work continues, with a new goal of 1 million cores
  - Includes scaling both the cores per node and the number of nodes
- Jitter/Noise will need additional focus and understanding
- I/O alternatives and options to reach more file systems
- Future releases of the Cray Linux Environment will enable more services
  - Need to add those services, by using configurations
  - Must pay attention to the effect those services have on scaling

CNL - Futures

- New Hardware Support
  - OctCore and beyond
    - Could require software specialization of cores for NIC support..
  - NUMA Nodes
    - Multi tiered and multi NIC
  - Huge Pages
    - Coalesce pages?
    - 1GB pages?
  - New networks
    - Gemini – opens up more memory to memory communication options

CNL - Futures

- Other future possibilities include:
  - Reboot on job launch to use kernel (and other services) tuned to application
    - Provides more flexibility for handling the different configurations
    - Could also be used to allow microkernels to be booted
      - But Cray would likely not develop those kernels
      - Node scheduling and job launch services still required
  - Virtualization techniques
    - Need to be careful about overheads, especially NIC access
    - Eases the mechanics of the “reboot” strategy above
  - Advanced runtime libraries
    - Replaces some OS functionality

CNL - Wrapping up

- CNL evolution –
  - The compute OS is specialized (part of a larger species) sufficiently for now
  - A striped down Linux has proven to be a good Compute Node OS
  - Future scaling (disruptive events) may lead to further evolutionary changes, but for now the changes will be incremental and controlled