Cluster Research

LOBOS and LinuxBIOS

Clusters are receiving wide attention in all areas of computing, especially the traditional supercomputing community. A modest number of inexpensive desktop machines connected via a high speed network is both affordable and sufficient for many applications. Even the largest supercomputers in the world are and will continue to be clusters of smaller supercomputers. For example, the ASCI "Q" system to be built at Los Alamos National Laboratory is a cluster of 374 machines each consisting of 64 processors. Using smaller machines as building blocks for a supercomputer helps amortize the development costs of the hardware, thus making it more reliable and affordable. But the hidden caveat of clusters is that maintenance cannot be amortized.

Why are clusters difficult to maintain?

Cluster nodes were designed to be stand-alone systems and as a result are more complicated than they need to be. For example, system administration is performed on a per-node basis. To upgrade a 128 node cluster of workstations, the administrator must walk to each of the machines with a keyboard, mouse, and monitor and perform the exact same upgrade 128 times. Contrast this with only one upgrade needed for a traditional supercomputer of 128 processors. Failures and rebooting may result in a similar expedition.

What is especially troublesome for Linux-based clusters is the existence of the legacy start-up mechanisms embedded in the BIOS. Current BIOS software performs machine initialization that is slow and often erroneous and redundant. Moreover, the BIOS and other machine-unique information such as configuration files cripple system administration by requiring physical interaction when changes need to be made to the system.

Where do we start?

The ultimate goal of our work is to reduce the maintenance cost of a cluster to 4 person-hours per week per 1024 nodes. Current clusters of this size require at least 10 times as many person-hours.

Linux Boots Linux

LOBOS (Linux OS Boots OS) is a system call added to the Linux kernel itself that allows a running Linux kernel to boot a new kernel. LOBOS has been used by two other groups as a reference implementation; one of these versions will be included in the next Linux kernel release.

LOBOS first reads the new kernel into memory and performs some hardware setup that prepares the machine for rebooting. Next, the actual bootstrapping code is copied to a "safe" location that will not be overwritten when the machine reboots. Finally, the code jumps to the bootstrap code which copies the new kernel and jumps to the kernel start. The key to LOBOS is that the machine is rebooted without using the existing BIOS in any way. This capability allows Linux to be used as a network bootstrap program and even as a BIOS.

Boot the BIOS

Since Linux can reboot Linux without using the BIOS, the next step is to eliminate the BIOS completely. LinuxBIOS replaces the normal BIOS found on Intel-based PCs, Alphas, and other machines with a Linux kernel that can boot Linux from a cold start. The LinuxBIOS is primarily Linux—about 10 lines of patches to the current Linux kernel. Additionally, the startup code—about 500 lines of assembly and 1500 lines of C—executes 16 instructions to get into 32-bit mode and then performs RAM and other hardware initialization required before Linux can take over.

LinuxBIOS is currently working on several mainboards from SiS and VIA, and work on boards from Acer, Dell, and Compaq is in progress. The LinuxBIOS project has also received software contributions from numerous vendors who use these boards.

Using a real operating system to boot another operating system provides much greater flexibility than using a simple netboot program or the BIOS. Because Linux is the boot mechanism, it can boot over standard Ethernet or over other interconnects such as Myrinet, Quadrics, or Scaleable Coherent Interface. It can use SSH connections to load the kernel, or it can use the InterMezzo caching file system or traditional NFS. Cluster nodes can be as simple as they need to be—perhaps as simple as a CPU and memory, no disk, no floppy, and no file system. The nodes will be much less autonomous, thus making them easier to maintain.

What next?

LOBOS and LinuxBIOS give system builders a new way of looking at things. The following are some interesting ideas we are looking at.

Real netboot: Linux can use any kind of network and any protocol and, therefore, so can LinuxBIOS. For example, an entire cluster can be loaded in constant time using IP multicast.

Secure netboot: Current netboot systems trust anybody. An authentication mechanism for kernels is sorely needed. We are currently extending the work in secure OS loading by our colleagues at the University of Maryland to work under LinuxBIOS.

Reload the OS with every application: Different applications tickle different kernel bugs that often propagate and eventually lead to a system crash. If the entire cluster can be rebooted in 3 seconds using LinuxBIOS, rebooting for each new application is a very reasonable solution. Moreover, we can load different versions of the kernel with each application.

Los Alamos

LALP-00-134 November 2000

More information about LOBOS and LinuxBIOS

contacts: Ron Minnich
e-mail: rminnich@lanl.gov
web: http://www.linuxbios.org

More information about the Advanced Computing Laboratory…

web: www.acl.lanl.gov

This work supported by the US Department of Energy.