Features [2]

This chapter describes the general capabilities of the UNICOS under UNICOS feature. Figure 1 illustrates the basic concept of the feature.

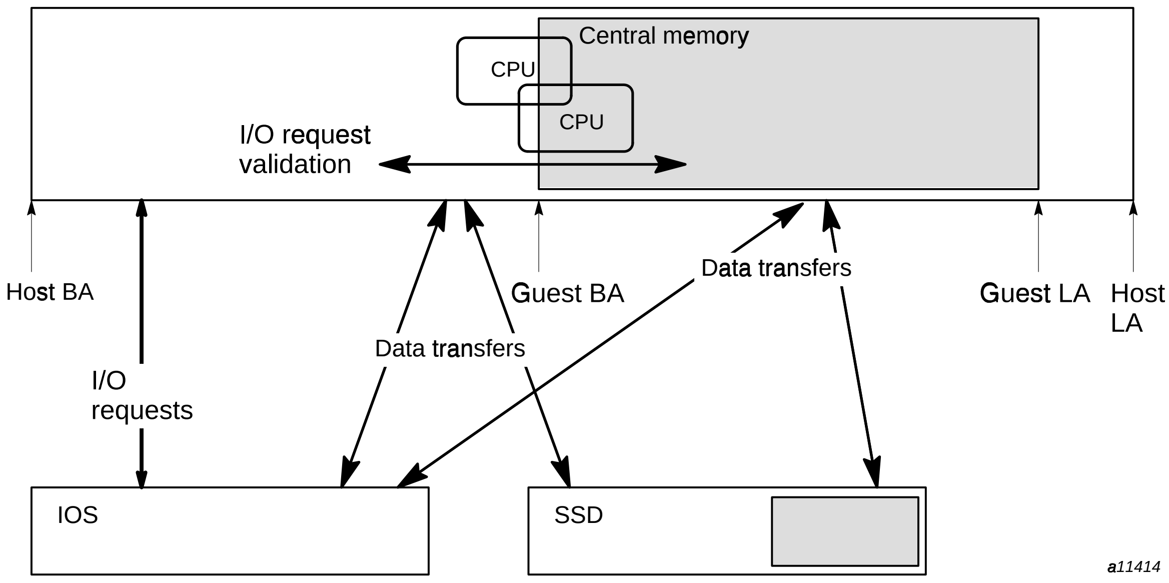

Figure 1. UNICOS under UNICOS concept

At guest boot time, the guest kernel and associated user space are allocated contiguously from the high address end of host user memory. Although the host kernel may access all of the physical memory, the guest kernel is limited to its own system and user space.

Guest central memory (shaded box) allocation is static in nature. That is, guest system memory is allocated dynamically from host memory at guest boot time, and is returned at guest termination. Guest memory is static for the life of the guest.

The administrator can configure minimum host CPUs and the guest user can set the maximum number of CPUs.

The SSD solid-state storage device is also statically allocated. SSD space that you plan to use on the guest must not be in use by the host at guest boot time.

This is also true for guest SSD memory (shaded box). The SSD allocation cannot be altered while the guest is running.

Guest CPU allocation, on the other hand, is dynamic. By default, there are no minimum or maximum number of CPUs that must (or can) be assigned to a guest system. Rather, the host is aware of the user workloads on both host and guest systems, and it will allocate CPUs to satisfy each of the kernel's requirements (within the constraints of CPU load balancing which is specified by the system administrator using the `guest`(1) command).

The IOS is shared by the host and guest systems and is unaware of the guest's presence. IOS code was not needed to support the UNICOS under UNICOS feature. SSD and IOS data transfers occur directly to and from guest memory, rather than being buffered through the host.

The major functional aspects of the UNICOS under UNICOS feature are as follows:

- A system containing support for the guest feature can run as either a guest or a host without recompiling.
- Guest and host kernels can be of different release and revision levels (for example, you can boot a UNICOS 10.0 guest from a UNICOS 9.0.2 host).

  Support will be provided for running a revision (such as UNICOS 9.0. *n*+1) to a particular level as a guest to the preceding revision (UNICOS 9.0.*n*) of that level. Support also will be provided for running one major release level (such as UNICOS 10.0) as a guest to the preceding release level (UNICOS 9.0).

  Major system architectural changes may prevent a new release level from running as a guest to the predecessor system. For a list of releases supporting your kernel as a guest, see the current *Cray Research Service Bulletin* (CRSB) .
- Guest kernels run in monitor mode.
- UNICOS under UNICOS safeguards :
  - Guest kernels can access only the memory allocated to them.
  - The host detects physical disk and SSD slice overlap.
  - The host detects I/O path overlaps.
- The host validates all SSD and IOS requests. Memory transfers are then made directly to and from the guest system's memory.

- The host performs the following functions on behalf of the guest:
  - User exchanges
  - Memory error handling
  - Register parity error handling
  - Physical I/O
- Guest panics typically do not cause a panic in the host.
- CPUs are scheduled dynamically across systems within specified percentages.
- Many major resources (such as SSD, disk, BMR, and network adapters) can be shared .
- The guest and host kernels share the I/O subsystem (IOS). The IOS draws no distinction between guest and host I/O requests.
- The `guest`(1) command is installed in the `/etc` directory. It is the user interface of the UNICOS under UNICOS feature and performs the following functions:
  - Starts a guest system by using the specified amount of memory
  - Stops a guest system
  - Releases guest system memory
  - Changes CPU percentages and ownership of running guests
  - Dumps running or stopped guest system memory
  - Returns the status of all active systems to any user