543

AT890-0648 RMC
K. A. Bohrer and J. S. Peek

92A062541

IBM Technical Disclosure Bulletin Vol. 35 No. 4A September 1992

UNIQUENESS OF THE AIX STORAGE MANAGEMENT DESIGN

The AIX* Storage Management design protects the kernel from data overruns, validates input data, allows for multiple heaps and allocates from a static area. The Storage Management Design rises above the ordinary allocation scheme by creating a very protective and closely monitored environment for the allocation of data areas to kernel processes. The elements of the design that make the AIX scheme unique from all other UNIX** based operating systems will be addressed in this article.

Definitions:

Kernel:
The base Operating System responsible for the control, access and allocation of all hardware resources is the kernel. The kernel is a collection of routines and extensions that include runtime services used only by other parts of the kernel, and SVCs (supervisor calls) which are made from user mode and requests the kernel to perform a particular task for the user level program.

Virtual Memory Segment:
In a virtual memory system, the memory is divided up into segments. Each such segment is 256 MB. Each segment in the system will have a unique segment identifier. The system has 16 segment registers to handle addressability to the various segments.

Address Space:
The kernel can access any segment it wishes, but the primary segments of the kernel's address space are the kernel segment in which the code for the kernel is located and the kernel extension segment which contains mostly large data structures used to maintain the system.

Storage Management:
Storage Management is the collection of policies, resources and routines that control the allocation and deallocation of memory.

A Heap:
A heap is contiguous memory from which storage is allocated and freed. There must be a way to describe the storage within a heap and to determine what storage is allocated and what storage should be allocated upon the next request for memory. Heaps are of a known and static size and begin on page boundaries.

Descriptor:
A descriptor is a structure used by the allocation routine to describe areas of memory. The descriptor will have the size of the area it describes along with the starting address for the area. A bit in the size field is used to determine if the area has been allocated or is free and can be allocated to another process.

Heap Anchor:
The heap anchor is a structure which describes the heap. Each heap has one anchor. The anchor describes where the heap begins (this points to the first page of descriptors), a pointer to the last block accessed by the last request for this heap, and a pointer the available pool of descriptors. The structure also contains a lock which is used to serialize requests.

A Page:
A page is 4096 bytes (4 kilobytes).

For a Storage Management policy to work and meet the diverse needs of processes and extensions that are part of the kernel, the policy must maintain dependability of the area to be allocated. Most Unix systems have data areas and descriptors mixed together. This allows descriptors to be overwritten by the errant program, thus wreaking havoc and damaging the integrity of the system. Another problem with having descriptors in-line with the data areas, is that to allocate or free storage you usually have to follow this chain through out the data areas. This could cause numerous page faults as the pages may not have been in memory. So dependability and protection of the descriptors is a very desired quality.

Other storage management policies use a dynamic area from which to allocate data. This is intrinsically slower if you are constantly having to expand the segment that you are using for allocation. Also, it is desirable to be able to release pages that have been freed. Otherwise, all the paging space will be used up and the system will spend too much time paging and may eventually run out of paging space.

A very important part of the AIX version 3 Storage Management is the protection offered to the system. All the descriptors used will be separate the data areas allocated. When a heap is initialized, the first page of the heap will be set aside for descriptors. These are the primary descriptors and are always used. After this page is a read-only page. This means that any attempt to write to this page will cause an exception to the process that was attempting to write. Any additional descriptors used will be outside the heap and protected by a read-only page also. This protection prevents any process from overwriting the descriptors. The only two routines that know the structure of the descriptors is xmalloc and xmfree, the two runtime services that allocate and free the storage. The descriptors will only be accessed by these two routines.

* Trademark of IBM Corp.
** Trademark of UNIX System Laboratories, Inc.