-

Operating System Feature Comparison: Memory Management

Rhea Mae V. Edwards

CS 444
Spring 2017

The Basis of Memory

The memory subsystem for an operating system is very important and essential for the extensional use and capabilities that truly makes an operating system in what it is.

Different types of memory can be store in two different location within an operating system. One location is the heap, which tends to store dynamically allocated memory or variables, and the other location being the stack, which holds local variables used within a file. Usually, anything that is related to memory that is not the stack, bss (block started by symbol), text or data, is located within the heap. In further detail, information that is persistent and non-local, and that will have to be cleaned up at some point within a program will be on the heap, and on the other hand, any pieces of information that is non-persistent and local to a program, and that is usually meant to be cleaned up automatically by the system, is located within the stack. With the inclusion of address binding, helps the operating system actually know where every variable lives within memory, which is set to each variable when the program is executed.

There are two different types of memory, logical or virtual memory and physical memory. In simple terms, on a modern system, a logical process is able to be done because of the physical connections of a device. For example, logical address space is backed up with physical address space. Logical addresses are different from physical addresses within a system. There is memory isolation with virtual addresses, because it is unwanted to share memory to the same physical addresses.

The structure that is used to organize these memory addresses are page tables. From a virtual page to a physical page, there is some sort of mapping that is done in order to have such memory addresses to be transferred. Each physical address within a table contains a number of virtual addresses. These virtual addresses are split between the reserved space to either hold the page number or to be the offset of the address. Page tables are organized in a tree structure.

The typical organization of these page tables is a tree structure consisting of three levels, even though theoretically there can be numerous levels to such a tree structure. The first level being the page global directory, then moving o the second level being the page middle directory, and then the third level actually being the level that contains the the page table, which each page table contains physical addresses of pages that each lead to a physical page. There are also different sizing names depending on the size of a tree structure and its pages overall.

So knowing that page tables process address space for memory, the actual location for page tables in a system goes the every back-end to the control register 3 of a device. This register holds the pointer of the current process, being the base address for memory.

Memory Management

Managing memory within an operating system can be done in a variety of ways. Within the x86 paging modes, there are mainly four different modes that can be practiced:

1) No Paging:

   By not applying a paging system with memory, is a management mode, which does have a limit in how much memory an operating system can store overall.

2) 32-bit Mode:

   Such a mode has a 32-bit law (logical address width) and at the most a 40-bit paw (physical address width).

3) PAE Mode:

   This mode is similar to the 32-bit mode of page management, where it has a 32-bit law (logical address width), but can go up to having a 52-bit paw (physical address width).

4) IA-32e:

   For this mode of page management, it has a 48-bit law (logical address width) and up to a 52-bit paw (physical address width). IA-32e is also considered the official standard, the mode that is mostly used throughout the industry.

And with each paging modes, there are different paging structures that manage and organize each practice in its own way.

An operating system can hold up to so much memory, but it does has its limits. When an operating system runs out of space to hold its memory, a method is used called page swapping, which uses other pages (that are already in use or have been used) to remove and write the newest memory to. This process temporary removes and stores usually the least used pages within memory to use. There is also the idea of thrashing with deciding which pages to remove and use within the system again, which takes the least-wanted page behavior, but such method usually slows down the hard drive at about 90% when in use.

WINDOWS' MEMORY MANAGEMENT

The Windows operating system's memory management organization has its similarities and differences when its compared to Linux's memory management system.

Overall, Windows' memory management, by default, contains 32-bit processes 2GB in size. This size can be increased to 3GB or 4GB in size on a 32-bit or 64-bit Windows operating system, respectively. 64-bit processes of a Windows operating system have 8192 GB (8TB) of virtual memory space.

Windows' memory management also offers multiple services, such as:

- Address Mapping
- Paging
- Memory Mapped Files
- Copy-On-Write Memory
- Physical Memory Allocation and Use

Windows' also has the idea of a working set, which is a set of pages that are physically present. There are also five (technically six, which has been found not to be all that useful) key parts, that can be used:

1) Working Set Manager
2) Process/Stack Swapper
3) Modified Page Writer
4) Mapped Page Writer
5) Zero Page Thread

Windows' memory management system is also fully re-entrant, with a finely grained locking structure. This provides:

- Allocation and Freeing Virtual Memory
- Shared Mappings
- Map Files
- Flush Page

There are two differentiating page sizes for withing Windows' pages, being large (2MB) and small (4kB) that it supports.

Also with Windows' memory management, there are a variety of page states that can exist, being:

- Free
- Reserved

  A state at which when a page is requested but not currently in use, which is also private.
- Committed

  A private state, that offers valid mapping, resident, and demand-zero.
- Shared

  A state that has valid mapping and resident.

When you create a process in Windows', it creates a container of threads. A user can have as many of processes as it wants, and when a thread is created, it is consisted in the heap, stack, or any other data structure it is held in.

Similarities to Linux

Similarities to Linux and why these similarities exist...

Differences from Linux

Differences from Linux and why these differences exist...

Linux calls there larger sized pages huge pages. When a process is created in Linux, it creates a container of groups.

FreeBSD's Memory Management

The FreeBSD's operating system's memory management organization also has its similarities and differences when its compared to Linux's memory management system (and possibly with its compared to the Window's memory management system as well).

Overall, FreeBSD's memory management...

Similarities to Linux

Similarities to Linux and why these similarities exist...

Handles its BM structure differently that Linux. FreeBSD's memory management system also does its licensing differently when compared to Linux.

Differences from Linux

Differences from Linux and why these differences exist...

Similarly to Linux, the FreeBSD's memory management system also has page tables and more along those lines, and open source implementation.

REFERENCES

[1] Information referenced within this document has been collected during CS 444 (Operating Systems 2) lectures, taught by D. Kevin McGrath during the term of Spring 2017, or from any prior knowledge that has already been integrated within my memory learned from elsewhere in the past.