Chapter 20 Paging: Smaller Tables

- Page tables can be too big and consume too much memory.
- Linear page tables get pretty big
  - Assume a 32-bit address space.
  - 4 KB pages.
  - 4 byte page table entry.
  - An address space has roughly one million virtual pages in it.
  - Multiply by the page table entry size and we get 4MB in size.
- There is usually one page table for every process in the system.
  - 100+ processes in a system is not unusual.

20.1 Simple Solution: Bigger Pages

- We can reduce the size of the page table by using bigger pages.
  - If we assume 16KB pages in the example above we would reduce the size to 1MB page table.
- Using this solution leads to wasting spcae within the page table.
  - Internal fragmentation.
  - Memory fills up quickly with large pages of small amounts of information.

20.2 Hybrid Approach: Paging and Segments

- Instead of having a single page table for the entire address space of the process have one per logical segment.
  - The base register and bound registers are in the MMU.
  - The base pointer points to the physical address of the page table of that segment.

20.3 Multi-level Page Tables

- How to get rid of invalid regions in the page table instead of keeping them all in memory.
- Multi-level page table
  - turns the linar page table into something like a tree
  - Many modern systems employ it.
- The idea behind multi-level page tables
  - Chop up the page table into page-sized units.
  - If an entire page of a page-table is invalid don't allocate that page of the page table.
  - To track if a page is valid use a structure called a page directory.
- The page directory contains one entry per page of the page table.
  - Consists of a number of page directory entries.
    - A PDE has a valid bit and a page frame number.
- Multi-level page tables have advantages over previous approaches.
  - Multi-level table only allocates page-table space in proportion to the amount of address space you are using.
  - Compact and supports sparse address spaces.
  - If carefully constructed each portion of the page table fits within a page.
    - Makes it easier to manage memory.
    - The os can grab the next free page when it needs to allocate or grow a page table.
- There is a cost to multi-level tables
  - On a TLB miss two loads from memory will be required to get the right translation information.
    - Example of a time-space trade-off
  - Compleixy is another drawback.

20.4 Inverted Page Tables

- Instead of having many page tables there is a single page table that has an entry for each physical page.
- The entry tells which process is using the page.
- Also which virtual page of that process maps to the physical page.

- Finding the correct entry is done by searching through the data structure.

  - A hash table is often built over the base structure to speed lookups.

- **Swapping the Page Tables to Disk**

  - So far the assumption is that page tables reside in kernel-owned physical memory.
  - It is possible that page tables may be too large to fit in memory.
    - Some systems store these in kernel virtual memory.
    - Allows swapping memory to disk when memory gets tight.