

CS 518

Virtual Memory in VAX/VMS

Randy Wang
Fall, 2002
Princeton University

Outline

- Theme: h/w s/w interactions
- Hardware features
- Software features
- BSD
- Closing remarks on virtual memory

CS518

1

Randy Wang

HW/SW Interactions

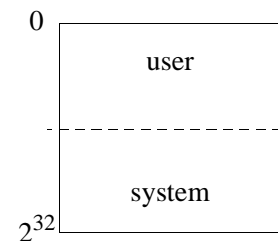
- OS designs intimately tied to technology advances
- VMS designed ~1975-1978, paper written in 1982
- What happened?
 - Physical memory got a lot bigger
 - Virtual address space got a lot bigger
 - (16 bit PDPs to 32 bit VAXes)
 - Paging devices got slower
 - (drums replaced by disks)

CS518

2

Randy Wang

Hardware Features



- Single address space shared by OS and user
 - Easy access of user addresses by OS
 - System space visible to user but subject to protection
 - OS calls become protected procedure calls
 - *An example convenience of a comfortably large virtual address space*

CS518

3

Randy Wang

Hardware Features (cont.)

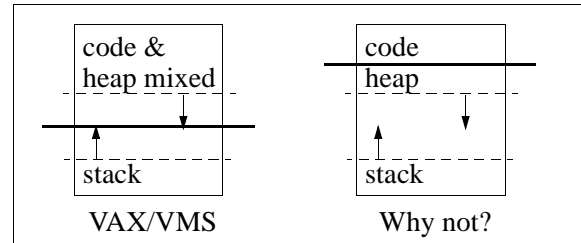
- Page table organization
 - User page tables in system address space
 - Can page user page tables
 - *Because both virtual and physical memories are big*
- Page size too small (512 bytes)
 - *Hangovers from PDP-11 (and drums)*
- No hardware support for use bits
 - *1st step towards the trend of minimal hardware support for virtual memory*

CS518

4

Randy Wang

Hardware Features (cont.)



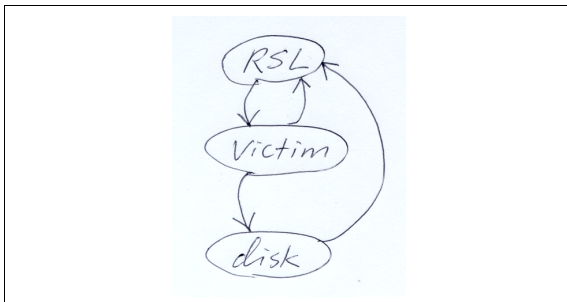
- Not enough user segments
 - User needs 3 segments, VAX/VMS has 2
 - Mix code and heap into one segment
 - Consequence: code page tables are not shareable (code pages can still be shared)
 - Why not put heap and stack into same segment?
 - limits size of code segment?
 - increases user page table size (but it's pageable)
 - increases system virtual address space, so it increases system page table

CS518

5

Randy Wang

Software Features



- Per-process replacement
 - Resident set limit (RSL)
 - No use-bits — FIFO replacement (with a twist)
 - Use free (and dirty) list as victim buffer
- $RSL \leq \text{process} \leq RSL + \text{victim buffer}$

CS518

6

Randy Wang

Software Features (cont.)

- Cluster I/Os
 - Reads:
 - readahead for pages that are contiguous both virtually and physically
 - Where do you put the read-ahead pages?
 - Writes
 - Write cluster of pages (between high/low thresholds)
 - Sort based on virtual addresses for later readahead
 - Swapping
 - Under high load, toss out whole processes
 - When swapping back in, bring in the entire resident set

CS518

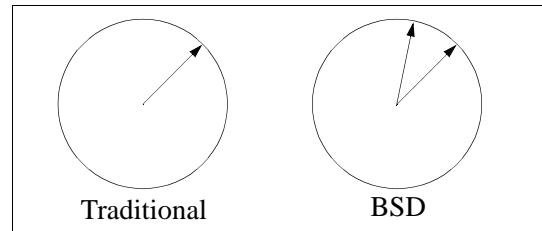
7

Randy Wang

Software Features (cont.)

- Voodoo constants
 - Resident size limit (RSL)
 - Victim buffer list length
 - Clustered write threshold
 - ...
 - Knobs that you have no ideas how to turn

BSD



- Simulated use bits
 - Use protection bits to simulate use bits on fault
 - Demonstrated minimum overhead
- 2-handed clock
 - Works well with large number of pages without high rate of use bit scanning

BSD (cont.)

- Global replacement
- “Disfavor” large processes when swapping
- Demand-loading (after swapped in)
- More voodoo constants

Virtual Memory Trends

- Minimum hardware support for virtual memory
 - Just TLBs
 - Everything else done in software
 - Incredible flexibility and power (stay tuned)
- Page size implications
 - Paging overhead
 - TLB coverage
 - Application needs (e.g. false sharing?)

Virtual Memory Trends (cont.)

- Is paging obsolete?
 - Unlike file system access, paging I/O tends to be random
 - Disk latency isn't improving
 - Physical memory is getting very large
 - The amount of time spent on paging is becoming less tolerable
 - You can't afford to have a "working set" that's substantially larger than physical memory
- Distributed memory
 - Paging over the network continues to receive interest
 - Distributed shared memory continues to receive interest

Project Ideas