Single Address Space OS (SASOS)

- Allocating address space at 1 GB/s (without reuse) will last for 500 years in a 64 bit architecture
  - Few seconds in a 32 bit architecture
- Now you don't have to create separate address spaces for each process - all processes can share a single address space
- Opal - separate address space from protection domain
  - Address space is set of virtual bindings
  - Protection domain defines what is accessible
  - Motivating example: Boeing which uses many parts to design a complex system

OS classification

- Unprotected OS - e.g. DOS
  - Cooperation is simple, efficient but fragile
- Private address space OS - e.g. UNIX, XP etc
  - Cooperation via communication primitives
- SASOS - e.g. Opal
  - Uniformly addresses global VM
  - Pure protection domain
  - Segment grain allocation, sharing and management of VM
  - Uniform naming and accessing control using capabilities

Issues

- Virtual contiguity and segment growth
  - How do you grow “arrays”
- Memory reclamation and address recycling
  - If no process has access to a segment then reclaim
  - Dangling pointers (to the old segment) are a problem
- Process creation by cloning address spaces (fork)
  - Cannot implement fork() abstraction
- Address remapping and copy on write

- Programs have to be careful in accessing private, temporary data or global data