OS Structure

Conventional Operating Systems

- System resources managed by trusted servers and the kernel
- Applications are restricted to OS interfaces
- Handle diverse application needs $\Rightarrow$ design an interface to handle all possible needs
- OS research over the past few decades has shown that this difficult

Exokernel Principles

- **Key idea:** Provide applications with control over resources
- Conventional OS: combine protection and management
- Exokernel: separate protection from management
  - Protect resources but delegate management to applications
- **Example 1:** Buffer management
  - Application manages its own buffer cache
  - Cached pages are securely shared across all applications
- **Example 2:** Application device channels (ADCs)

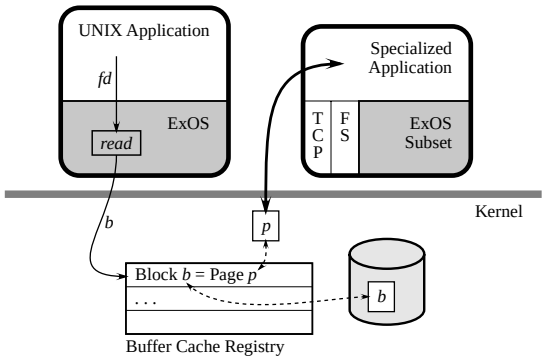
Exokernel Principles (contd)

- Separate protection from management
  - Use low level primitives for protection (disk blocks, TLB, ...)
  - Resource management functions: allocation, revocation...
- Expose allocation: application allocate resources explicitly
- Expose names: use physical names whenever possible
- Expose revocation: application choose when to give up resources
- Expose information: system information and data structures are accessible to applications

Architectural Issues

- Not all applications require customizes resource management
- Legacy applications should coexist with new applications
- **Solution:**
  - Use library operating systems for traditional applications
  - LibOSes handle interactions with the exokernel
  - LibOSes are unprivileged $\Rightarrow$ anyone can implement, improve or extend an LibOS
- **Issues**
  - Can new applications get benefits of exkernel's flexibility?
  - Do legacy applications pay a performance penalty?

Example


UNIX Application
fd
read
ExOS
Specialized
Application
T
C
P
F
S
ExOS
Subset
Kernel
b
p
Block b = Page p
. . .
b
Buffer Cache Registry

Xok/ExOS

- Xok: an Intel x86-based exokernel
- Features:
  - Physical memory: expose hardware capabilities (e.g., MMU bits) to applications
  - Wakeup predicates: application-specified predicates used to wakeup the application
  - Access control: xok provides protection through access control

Xok/ExOS (contd)

- ExOS: library operating system with 4.4 BSD interfaces
- Features:
  - Library-based implementation: applications can override any feature
  - Easy customizability
  - Supports UNIX abstractions like processes, IPC, files...

Experimental Results

- Testbed: 200MHz Pentium Pro with SCSI disks
- Compare Xok/EXOS with FreeBSD and OpenBSD
- Main results:
  - Legacy application and benchmarks have comparable performance
  - New applications (XCP, Cheetah http server) run 3-8 times faster