## introduction

previous kernels: the interface between the user applications and hardware resources, have fixed abstractions, those resources include processes, files, address spaces and interprocess communications.

exokernel: implement the IPC, virtual memory are implemented in user level applications. The exokernel only implements resource protections.

\tip: drawbacks or challenges for exokernel: hard for sharing between different library OSs.

thought: separate protection from management.

methods: export hardware resources rather emulating them

secure bindings: library OS bind of machine resources securely

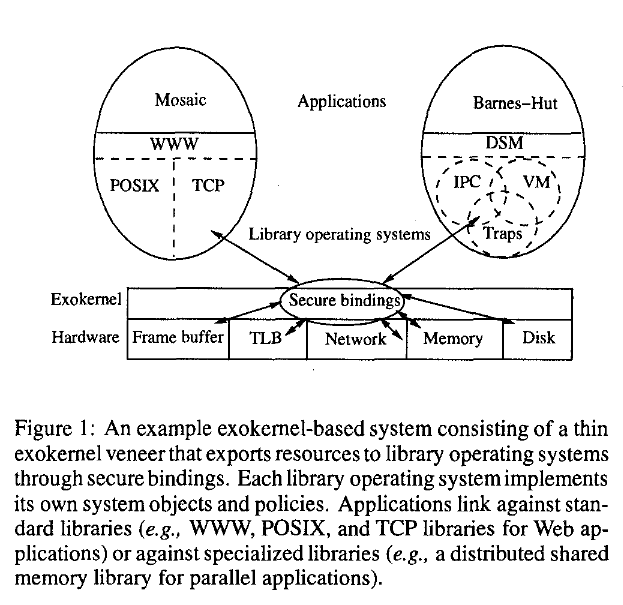
visible resource revocation: library OS revoke a resource

abort protocol: the exokernel can break the secure bindings

the drawbacks of previous methods:

* the abstraction is too general to all the applications that the performance hurt
* the user applications can not know every information
* the user applications feature kinds can suffer

## the design



portable library operating systems - no secure requirements - easier implementation

the functions of the exokernel: expose hardware resources securely, including the resource locations, resource names, resource revocation

### secure binding

exokernel does not need to understand the usage of a resource

three techniques in secure binding:

hardware mechanisms

software caching

downloading application codes

### visible resource revocation

two kinds of resource revocation

visible

invisible

revocation process: dialogue between an exokernel and the library operating systems.

if the revocation process fails -> abort protocol -> repossession vector to inform the corresponding library operating system.

## aegis

events

exception - dispatched to applications

interrupt

protected entry - protected control transfer with two kinds: synchronous and asynchronous(a kind of right amplification)

addressing - STLB / forward to application

## ExOS

manage fundamental OS abstractions at application levels. e.g. keep VM, process within the address space of the application that is using it.

IPC: implemented on the primitives offered by Aegis protected entry.

VM: the virtual memory system in ExOS is a linear table;

ASH: application-safe handler -> the handler was downloaded into the kernel

## extensibility

The extensibility lies for the different kinds of library operation systems.

* extensible RPC
  + lrpc, tlrpc
* page table algorithms e.g. linear page tables
* schedulers more options