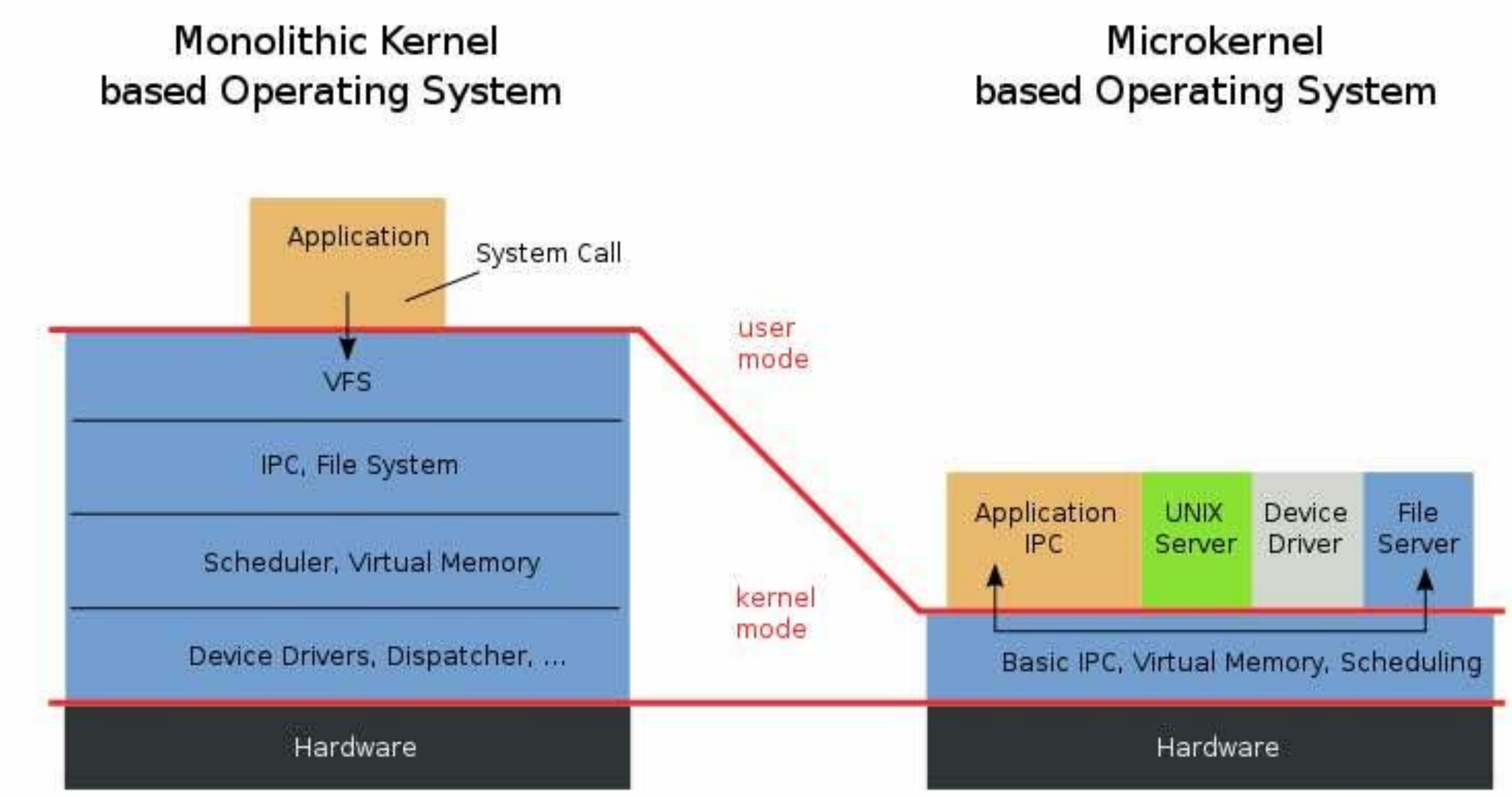
MICROKERNELS

KISHAN VARMA
VAISHALI CHANANA

AGENDA

- MONOLITHIC KERNELS VS MICROKERNELS
- FIRST GENERATION MICROKERNELS
- SECOND GENERATION MICROKERNELS
- EXOKERNEL
- AEGIS – AN EXOKERNEL
- ExOS

MONOLITHIC VS MICROKERNELS



Source : Wikipedia for Microkernels

Smaller Kernel : Easily maintained and less error prone.

Highly modular structure

More Flexible and Extensible

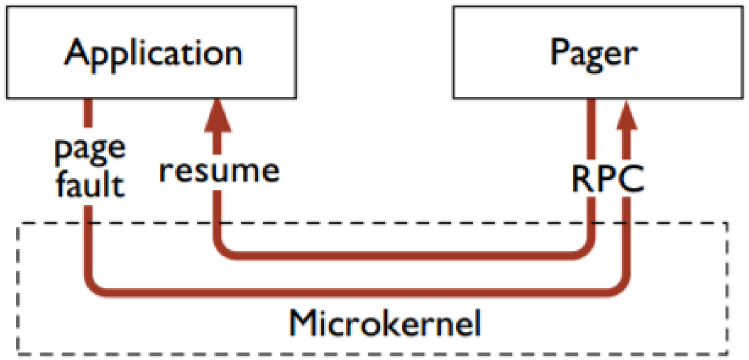
Isolation of Server Malfunctions

Device drivers can be run as servers

Different file systems, different APIs coexist in one system

FIRST GENERATION MICROKERNELS

- objects and mechanisms were lower-level with more general abstractions than UNIX
- become widely accepted
  - General flexibility
  - Preserving UNIX compatibility
- <u>Mach's external pager</u> :
  - Kernel manages physical and virtual memory but forwards page faults to specific user-level tasks
  - After page fault, pagers return the page image to kernel
- <u>Handling h/w interrupts</u> :
  - H/W interrupt as IPC messages for user-level process



FIRST GENERATION MICROKERNELS (CONTD)

- Weakness
  - Higher cost of RPC
  - Higher cost of memory references
    - Worst locality properties of combined microkernel code
    - High cache miss rate due to more modularity
  - Main memory is still managed by the microkernel

SECOND GENERATION MICROKERNELS

- Designing microkernel architecture from scratch
- Believes that efficiency and flexibility require minimal set of abstractions
- EXOKERNEL
  - Developed at MIT in 1994-95, works on idea that abstractions are costly and restrict flexibility
  - Multiplex hardware resources by minimal set of primitives
  - More details later
- L4
  - Processor dependent
  - No hardwiring policy inside kernel; kernel offers the basic mechanisms

SECOND GENERATION MICROKENELS (CONTD.)

ADDRESS SPACE ABSTRACTION:

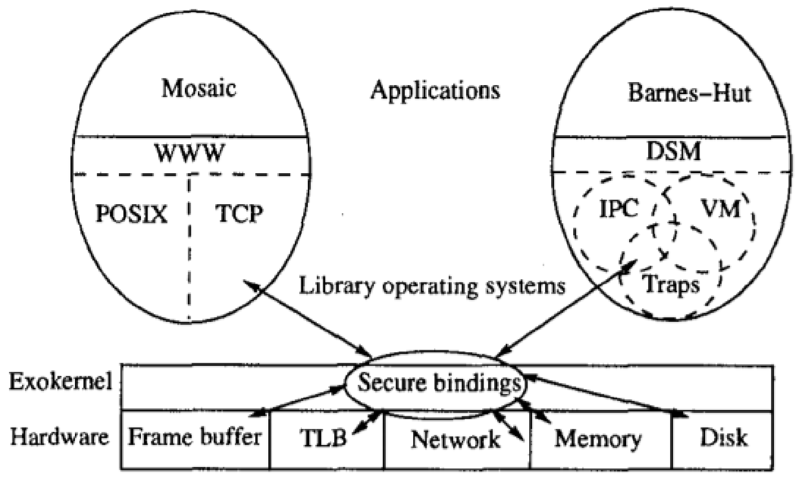
- Protection schemes and physical memory management on top of microkernel
- Supports recursive construction of address spaces outside the kernel
- Three operations – grant, map and demap – implemented by IPC
  - Granted page is removed from granter's address space and included in grantee's address space
  - After mapping, page can be accessed in multiple address spaces
  - Demapped page remains in demapper's address space but is removed from all other address spaces
- Memory management and paging outside the kernel and these operations inside the kernel

IPC ABSTRACTION:

- Passing shorter messages
- Single copy transfer by temporarily sharing the target region

EXOKERNELS : MOTIVATION

- Centralized resource management via a set of abstractions that cannot be specialized, extended, or replaced.
- Fixed high-level abstractions limit the functionality of applications.
- Applications know better what the goals of their decisions should be
- The lower the level of a primitive, the more efficiently it can be implemented, and the more latitude it grants to implementers of higher-level abstractions.
- Library Operating Systems : Portable & Compatible as is desirable.



EXOKERNELS : DESIGN

- Exokernel hands over resource policy decisions to Library Operating Systems.
- Techniques employed by Exokernel :
    - Secure Binding
    - Visible Revocation
    - Abort Protocol
- A secure binding is a protection mechanism that decouples authorization from the actual use of a resource.
- Secure Binding : Hardware Mechanisms, Software Caching and Downloading Application Code into the kernel.
- Resource Revocation: An Exokernel uses Visible Revocation.
- Abort Protocol: Significance of *Repossession vector.*

AEGIS : AN EXOKERNEL

- Aegis comes with a system call interface and primitive operations.
- Scheduling : Round Robin
- Aegis represents the CPU as a linear vector, where each element corresponds to a time slice.
- Aegis Processor Environment : Exceptions, Interrupts, Protected Control transfers, and Address translations.
- Handling of Exceptions
- Handling TLB Miss.
- Protected Control Transfers : Synchronous and Asynchronous.
- Aegis uses Dynamic Packet Filter (DPF), a new packet filter system that is over an order of magnitude more efficient than previous systems.

System call	Description
`Yield`	Yield processor to named process
`Scall`	Synchronous protected control transfer
`Acall`	Asynchronous protected control transfer
`Alloc`	Allocation of resources (*e.g.*, physical page)
`Dealloc`	Deallocation of resources

Source : Exokernel's Paper

Primitive operations	Description
`TLBwr`	Insert mapping into TLB
`FPUmod`	Enable/disable FPU
`CIDswitch`	Install context identifier
`TLBvadelete`	Delete virtual address from TLB

Source : Exokernel's Paper

EXOS : A LIBRARY OPERATING SYSTEM

- ExOS : manages fundamental operating system abstractions (e.g., virtual memory and process) at application level, completely within the address space of the application that is using it.
- IPC Abstractions : pipe, Shared Memory and LRPC .
- Application-Level Virtual Memory : Page tables implemented as Linear Vector.

  No support to handle swapping .
- Application Specific Safe Handlers : Untrusted application-level message-handlers downloaded into the kernel but made safe by a combination of code inspection and sandboxing , and executed upon message arrival.


ANY
QUESTIONS?