

# Extensibility: SPIN and exokernels

Robert Grimm

# OS Abstraction Barrier

---

- Fixed high-level abstractions
  - Hurt application performance
  - Hide information
  - Limit functionality
- Examples
  - Buffer cache management
  - Persistent storage

# Goals

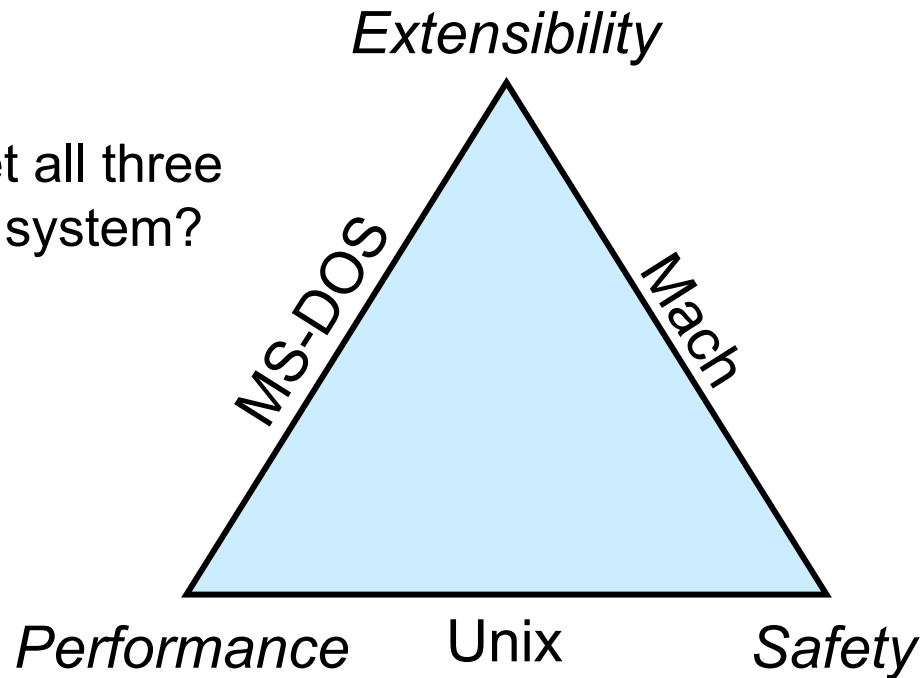
---

- Extensibility
  - Applications introduce specialized services
- Safety
  - Kernel, applications, services are protected
- Performance
  - Extensibility and safety have low cost

# Why is this hard?

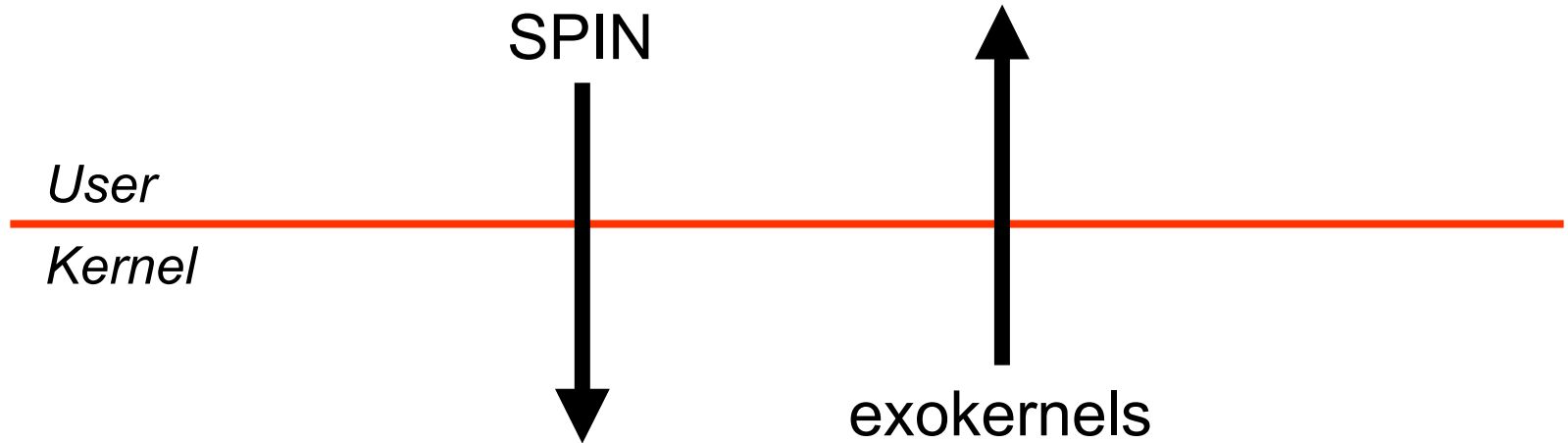
---

Can we get all three  
in a single system?



# Two Approaches

---



# SPIN Approach

---

- Put extension code in the kernel
  - Cheap communication
- Use language protection features
  - Static safety
- Dynamically interpose on any service
  - Fine-grained extensibility

# The Big Picture

---

OSF/1 Unix  
Server

Unix  
App

Video  
Server

Web  
Server

*Applications*

*User*

---

*Kernel*

Mach API

Unix API

Net Video

HTTP

*Application  
Extensions*

Threads

---

Syscall

Process

Network

FS

*Shared  
Extensions*

---

Execution  
State

Memory

Devices

Extension  
Services

*Core  
Services*

# Modula-3

---

- Type-safe programming language
- Interfaces
- Garbage collection
- Other features: objects, generic interfaces, threads, exceptions
  
- Most of kernel written in Modula-3
- Extensions must be written in Modula-3

# Safety

---

- Capabilities
  - Simply a pointer
- Protection domains
  - Language-level
  - Limit visibility of names
  - Enforced at dynamic link time

# Extensibility

---

- Extension model
  - Events
  - Event handlers
  - Guards
- Mechanism
  - Dispatcher
  - Common case: procedure call

# Core Services

---

- Memory management
  - Physical addresses
  - Virtual addresses
  - Translations
- Thread management
  - Signals to scheduler
    - Block, unblock
  - Signals to thread manager
    - Checkpoint, resume

# Performance

---

- It works

# Exokernels Approach

---

- Make the application do it!

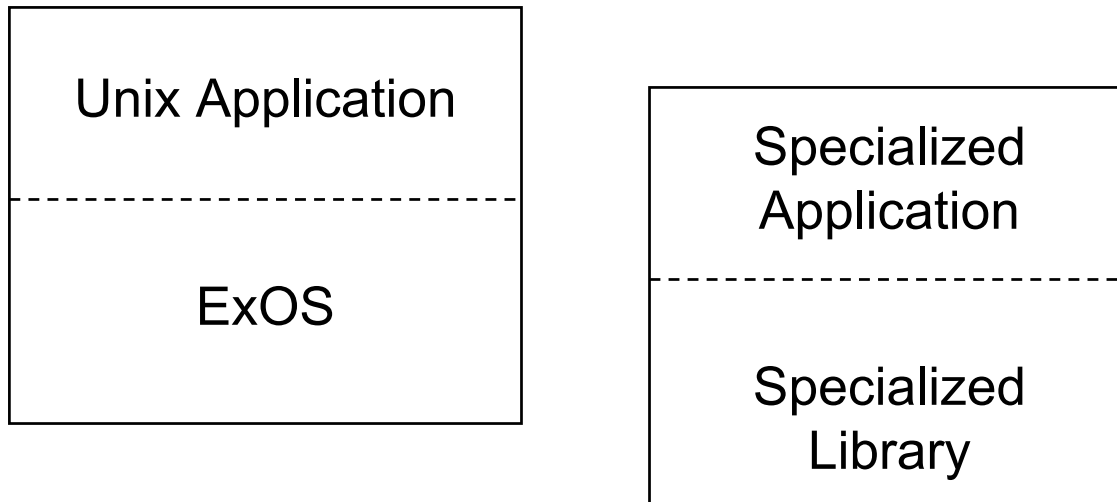
# Exokernels Approach (again)

---

- Separate protection and management
- Expose allocation
- Expose names
- Expose revocation
- Expose information

# The Big Picture

---



*User*

*Kernel*

---

Disk

Scheduling

Packet  
Dispatch

Control  
Transfer

Execution  
State

Memory

Devices

# At The Core

---

- Processor time slices
- Processor environments
  - Exceptions
  - Interrupts
  - Protected entries
  - Addressing
  - Hierarchical capabilities
- Book keeping

# Case Study: The Disk

---

- Problem
  - How to store meta-data?
    - Ownership of disk blocks
- Failed approaches
  - Simple capabilities
  - Self-descriptive meta-data
  - Template-based descriptions

# The Disk (continued)

---

- Untrusted deterministic functions
  - Programmatic templates
- Shared data
  - Buffer cache registry
- Ordered disk writes
  - Ensure consistency after crash

# Performance

---

- It works
- It scales

# Issues

---

- SPIN
  - Trusted compiler
  - Resource control

# Issues (continued)

---

- Exokernels
  - Extension model
  - Downloaded code
    - Wakeup predicates
    - Dynamic packet filters
    - Application-specific handlers
    - Untrusted deterministic functions
  - Complexity of disk management

# What do you think?

---