第 2 讲：OS Architecture & Structure

第五节：Exokernel – Xok+ExOS

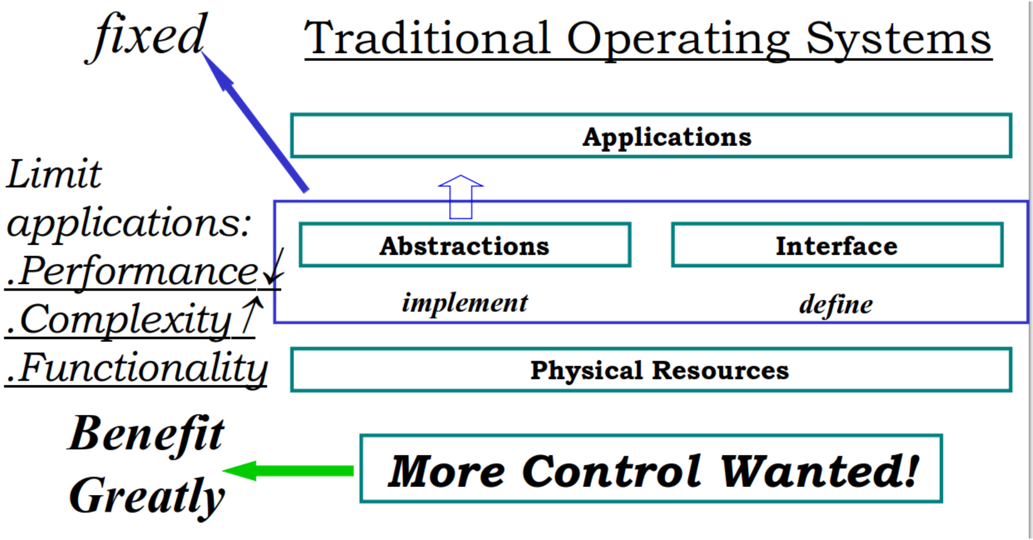
陈渝

清华大学计算机系

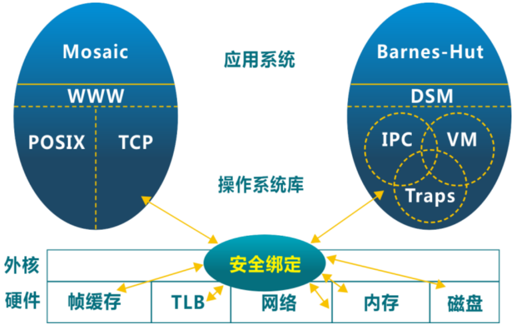
yuchen@tsinghua.edu.cn

2020 年 2 月 23 日

Problem

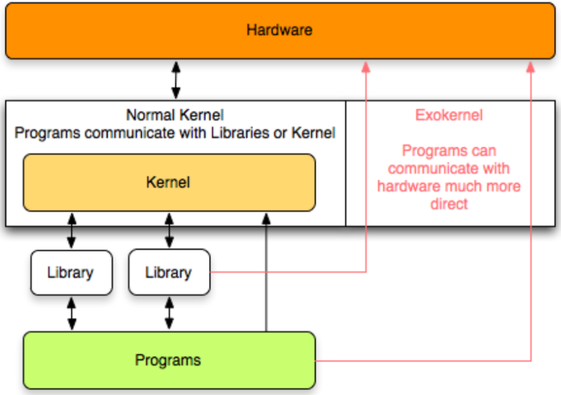

fixed
Traditional Operating Systems
Applications
Limit applications:
.Performance↓
.Complexity↑
.Functionality
Abstractions
Interface
implement
define
Physical Resources
Benefit Greatly
More Control Wanted!

Insight



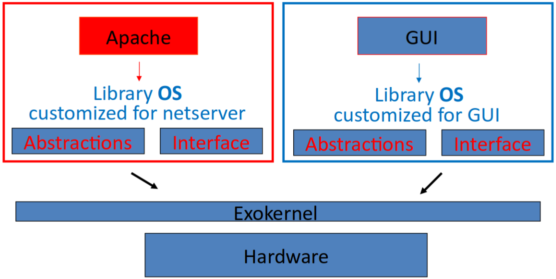
- Applications Know Better Than OS
- Application demands vary widely

Ideas



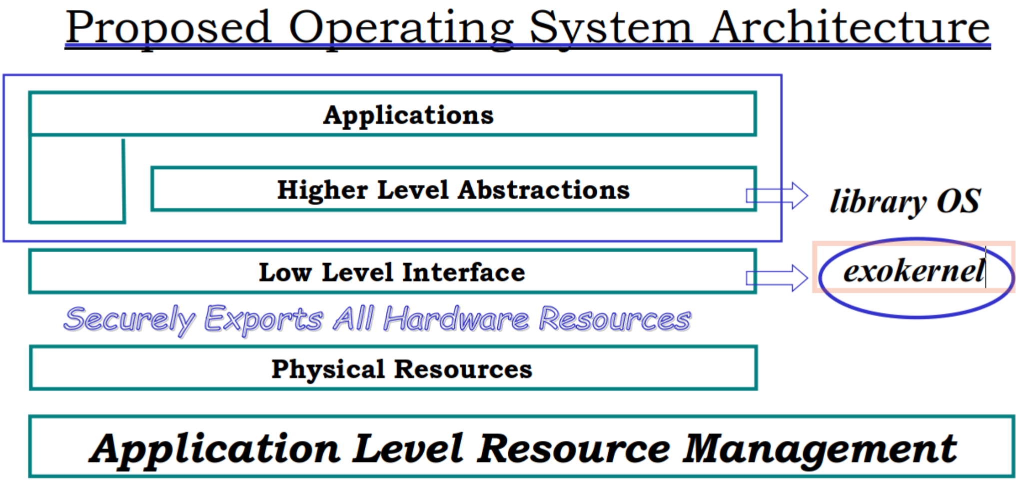
- Give un-trusted applications as much control over physical resources as possible
- To force as few abstraction as possible on developers
- separate protection from management

Ideas

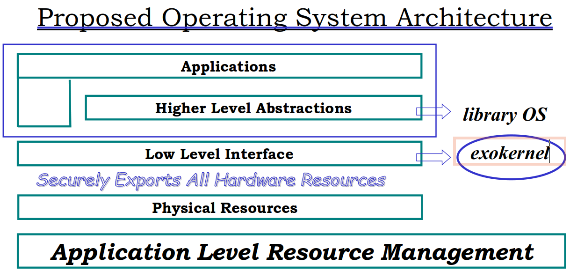


- Give un-trusted applications as much control over physical resources as possible
- To force as few abstraction as possible on developers
- separate protection from management

Solution

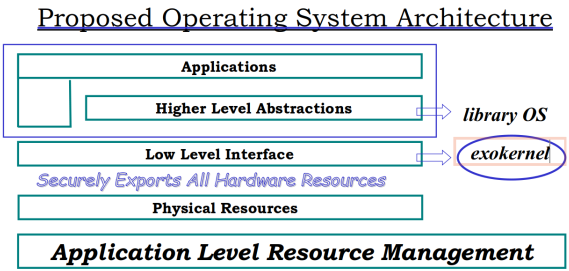

Proposed Operating System Architecture
Applications
Higher Level Abstractions
library OS
Low Level Interface
exokernel
Securely Exports All Hardware Resources
Physical Resources
Application Level Resource Management

Challenges



- Tracking ownership of resources
- Ensuring resource protection
- Revoking resource access

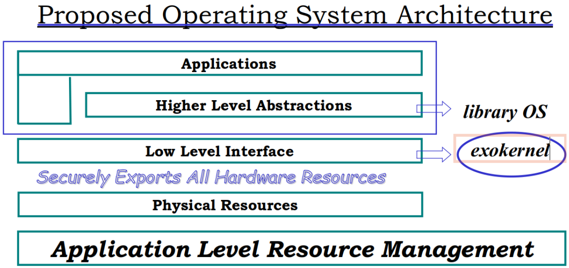
Techniques



- Secure binding
- Visible revocation
- Abort protocol

Techniques – secure binding

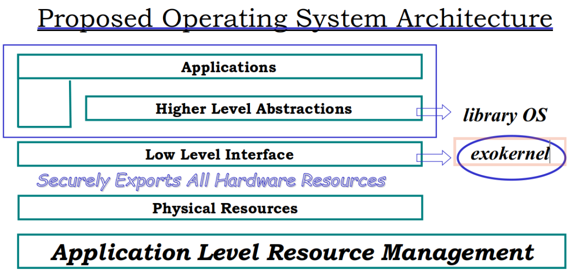
It is a protection mechanism that decouples authorization from actual use of a resource



- Secure binding techniques
  - Hardware mechanism
  - Software caching
  - Downloading application code

Techniques – visible resource revocation

A way to reclaim resources and break their(application & resources) secure binding

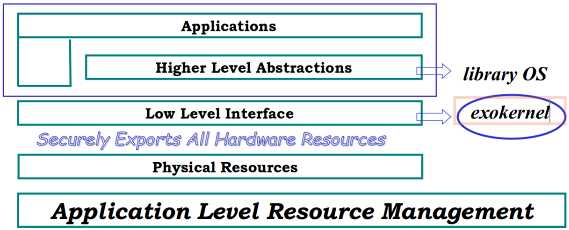


- An exokernel uses visible revocation for most resources
  - traditional OS have performed revocation invisibly.
- dialogue between an exokernel and a library OS
- library OS should organize resource lists

Techniques – the abort protocol

If a library OS fails to respond quickly, the secure bindings need to be broken "by force"

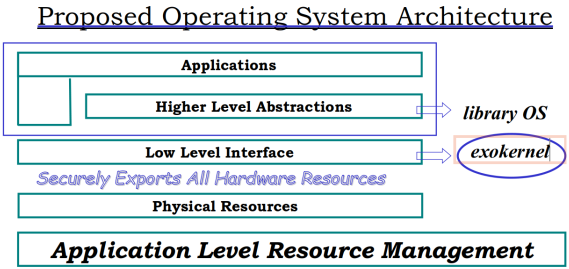
Proposed Operating System Architecture



- The abort protocol
  - An exokernel simply breaks all secure bindings to the resource and informs the library operating system

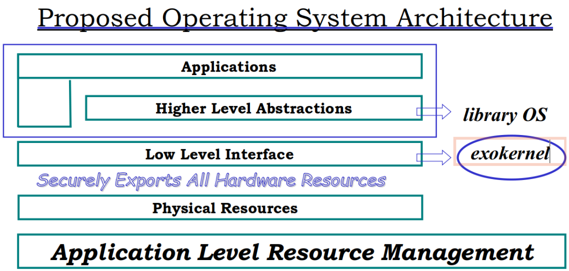
Techniques – library OS

Manage OS abstractions at application level



- IPC Abstractions
- Application-level Virtual Memory
- Remote Communication

Implementation



- Prototype (Xok / ExOS)
  - Exokernel: Xok on x86, Aegis runs on DEC
  - Library OS: ExOS, fundamental OS abstractions at application level

Aegis: Base Costs

Machine	OS	Procedure call	Syscall (getpid)
DEC2100	Ultrix	0.57	32.2
DEC2100	Aegis	0.56	3.2 / 4.7
DEC3100	Ultrix	0.42	33.7
DEC3100	Aegis	0.42	2.9 / 3.5
DEC5000	Ultrix	0.28	21.3
DEC5000	Aegis	0.28	1.6 / 2.3

Table 4: Time to perform null procedure and system calls. Two numbers are listed for Aegis's system calls: the first for system calls that do not use a stack, the second for those that do. Times are in microseconds.

Aegis: Exceptions

Machine	OS	unalign	overflow	coproc	prot
DEC2100	Ultrix	n/a	208.0	n/a	238.0
DEC2100	Aegis	2.8	2.8	2.8	3.0
DEC3100	Ultrix	n/a	151.0	n/a	177.0
DEC3100	Aegis	2.1	2.1	2.1	2.3
DEC5000	Ultrix	n/a	130.0	n/a	154.0
DEC5000	Aegis	1.5	1.5	1.5	1.5

Table 5: Time to dispatch an exception in Aegis and Ultrix; times are in microseconds.

Performance

***Aegis*: providing protected control transfer as substrate for efficient IPC implementation**

OS	Machine	MHz	Transfer cost
Aegis	DEC2100	12.5MHz	2.9
Aegis	DEC3100	16.67MHz	2.2
Aegis	DEC5000	25MHz	1.4
L3	486	50MHz	9.3 (normalized)

Table 6: Time to perform a (unidirectional) protected control transfer; times are in microseconds.

L3: the fastest published result.

Aegis: using Dynamic Packet Filter

Filter	Cold Cache	Warm Cache
MPF	71.0	35.0
PATHFINDER	39.0	19.0
DPF	7.5	1.5

Table 7: Time on a DEC5000/200 to classify TCP/IP headers destined for one of ten TCP/IP filters; times are in microseconds.

MPF: a widely used packet filter engine.

PATHFINDER: fastest packet filter engine.

ExOS: IPC Abstractions

Machine	OS	pipe	pipe'	shm	lrpc
DEC2100	Ultrix	326.0	n/a	187.0	n/a
DEC2100	ExOS	30.9	24.8	12.4	13.9
DEC3100	Ultrix	243.0	n/a	139.0	n/a
DEC3100	ExOS	22.6	18.6	9.3	10.4
DEC5000	Ultrix	199.0	n/a	118.0	n/a
DEC5000	ExOS	14.2	10.7	5.7	6.3

Table 8: Time for IPC using pipes, shared memory, and LRPC on ExOS and Ultrix; times are in microseconds. Pipe and shared memory are unidirectional, while LRPC is bidirectional.

ExOS: Virtual Memory measured by matrix multiplication

Machine	OS	matrix
DEC2100	Ultrix	7.1
DEC2100	ExOS	7.0
DEC3100	Ultrix	5.2
DEC3100	ExOS	5.2
DEC5000	Ultrix	3.8
DEC5000	ExOS	3.7

Table 9: Time to perform a 150x150 matrix multiplication; time in seconds.

ExOS: Remote Communication

Machine	OS	Roundtrip latency
DEC5000/125	ExOS/ASH	259
DEC5000/125	ExOS	320
DEC5000/125	Ultrix	3400
DEC5000/200	Ultrix/FRPC	340

Table 11: Roundtrip latency of a 60-byte packet over Ethernet using ExOS with ASHs, ExOS without ASHs, Ultrix, and FRPC; times are in microseconds.

FRPC: fastest RPC on comparable hardware.