Microkernels Meet Recursive Virtual Machines

Bryan Ford Mike Hibler
Jay Lepreau Patrick Tullmann
Godmar Back Stephen Clawson

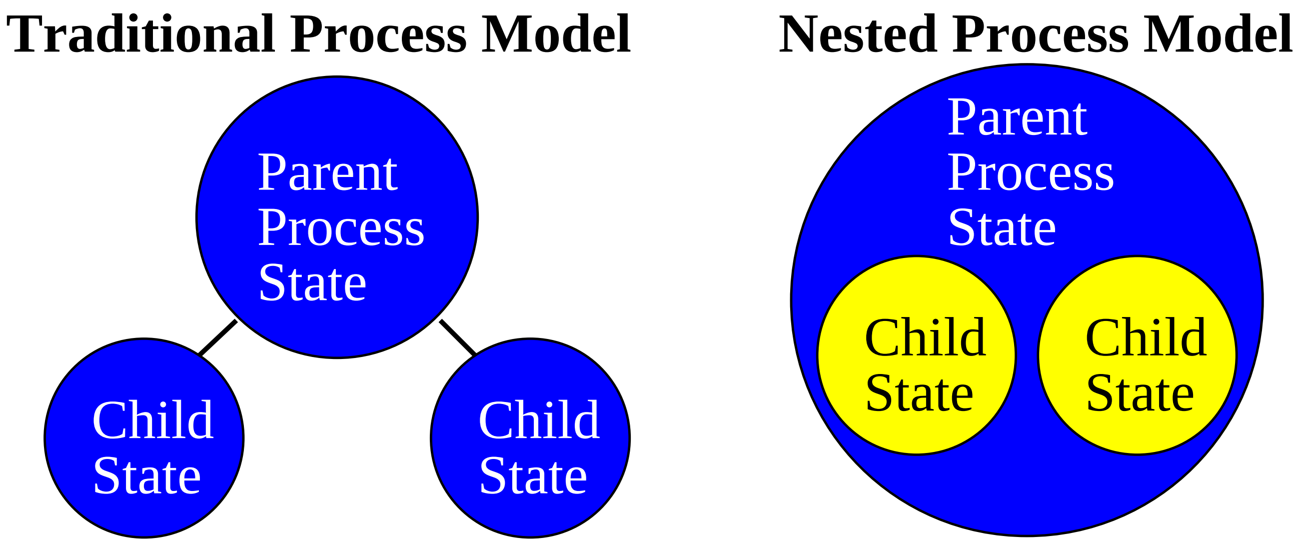
Department of Computer Science
University of Utah

http://www.cs.utah.edu/projects/flux/

October 30, 1996

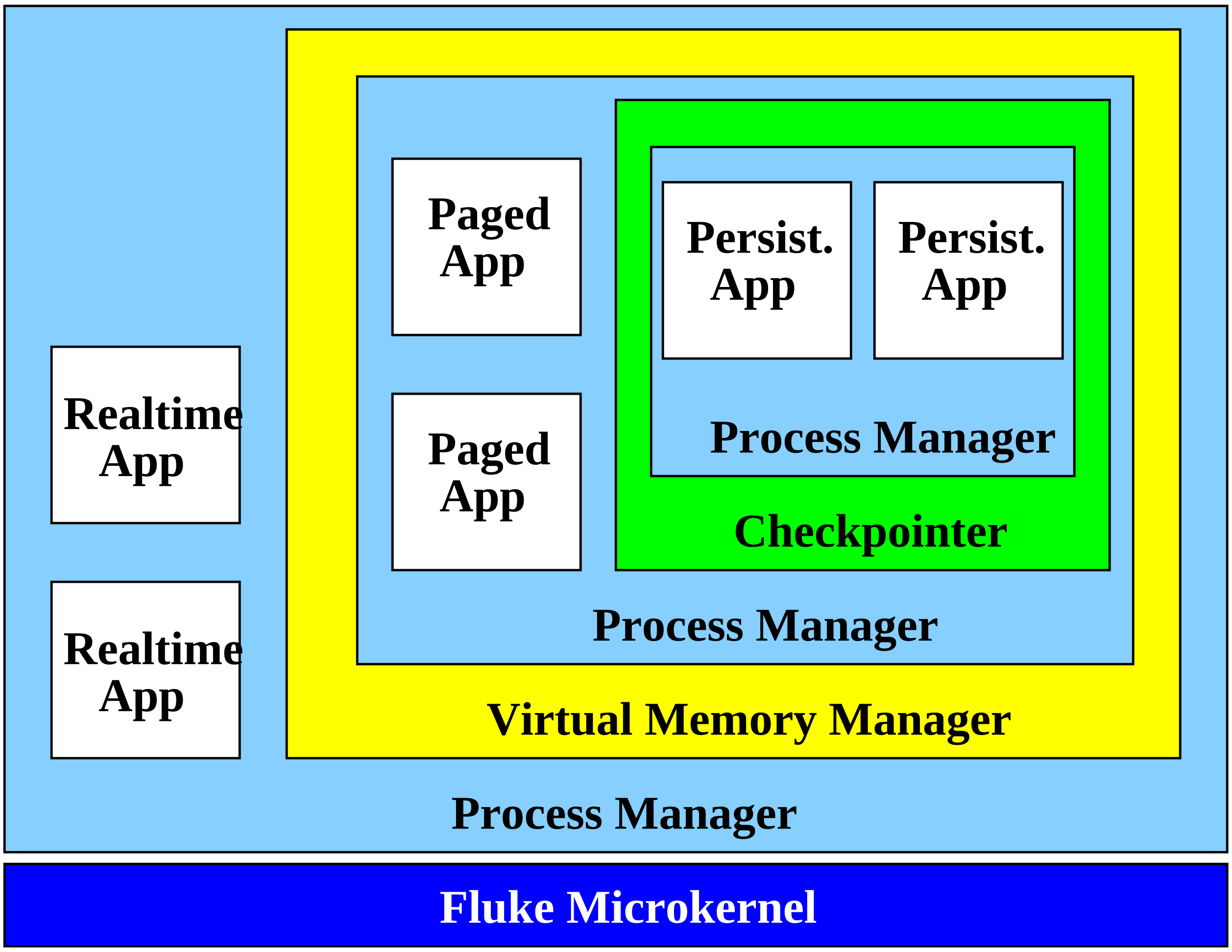
The Nested Process Model

Child process is **encapsulated** in its parent.



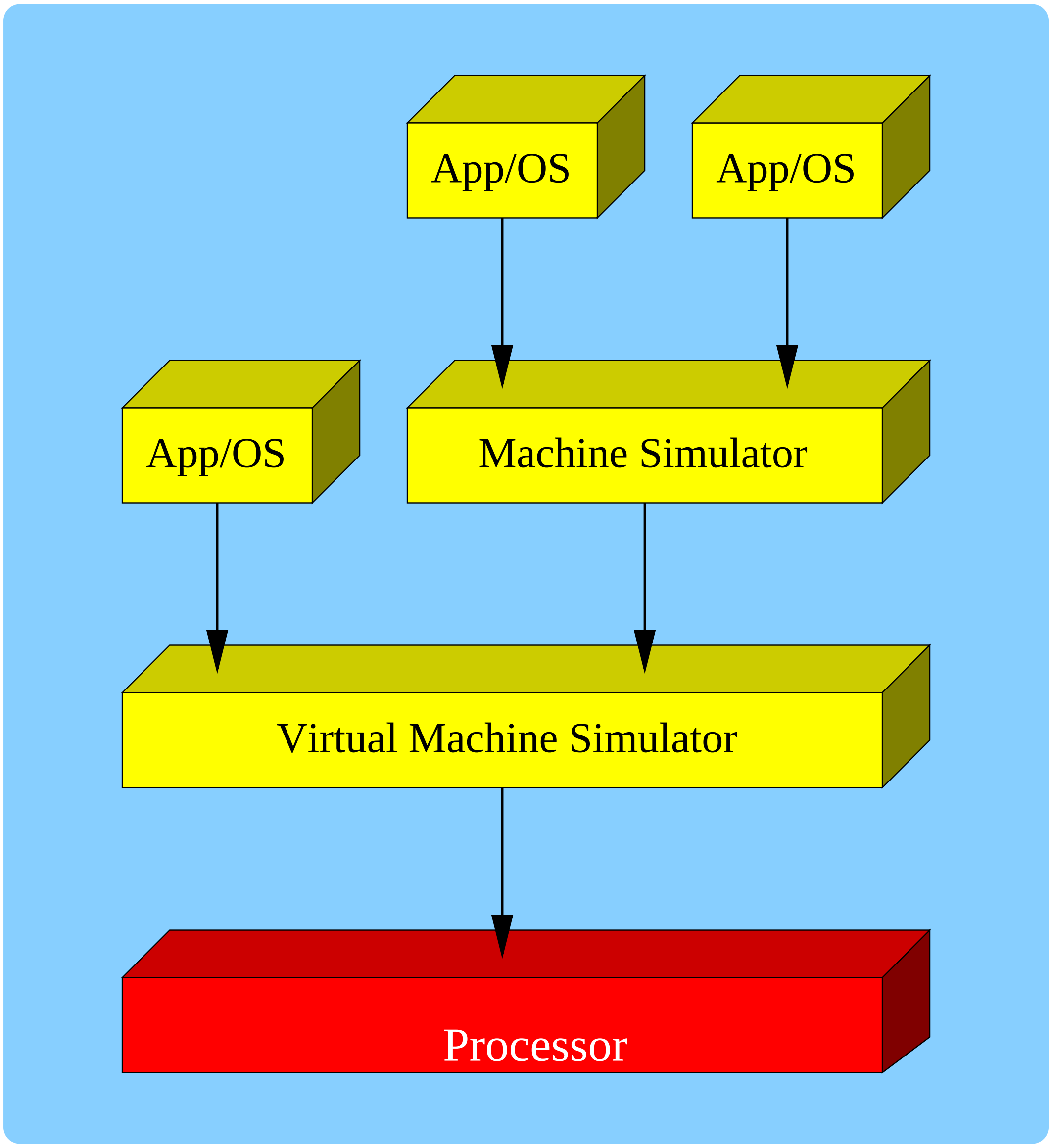
Parent has **complete control** over the child.

Supports efficient **decomposition** of system services

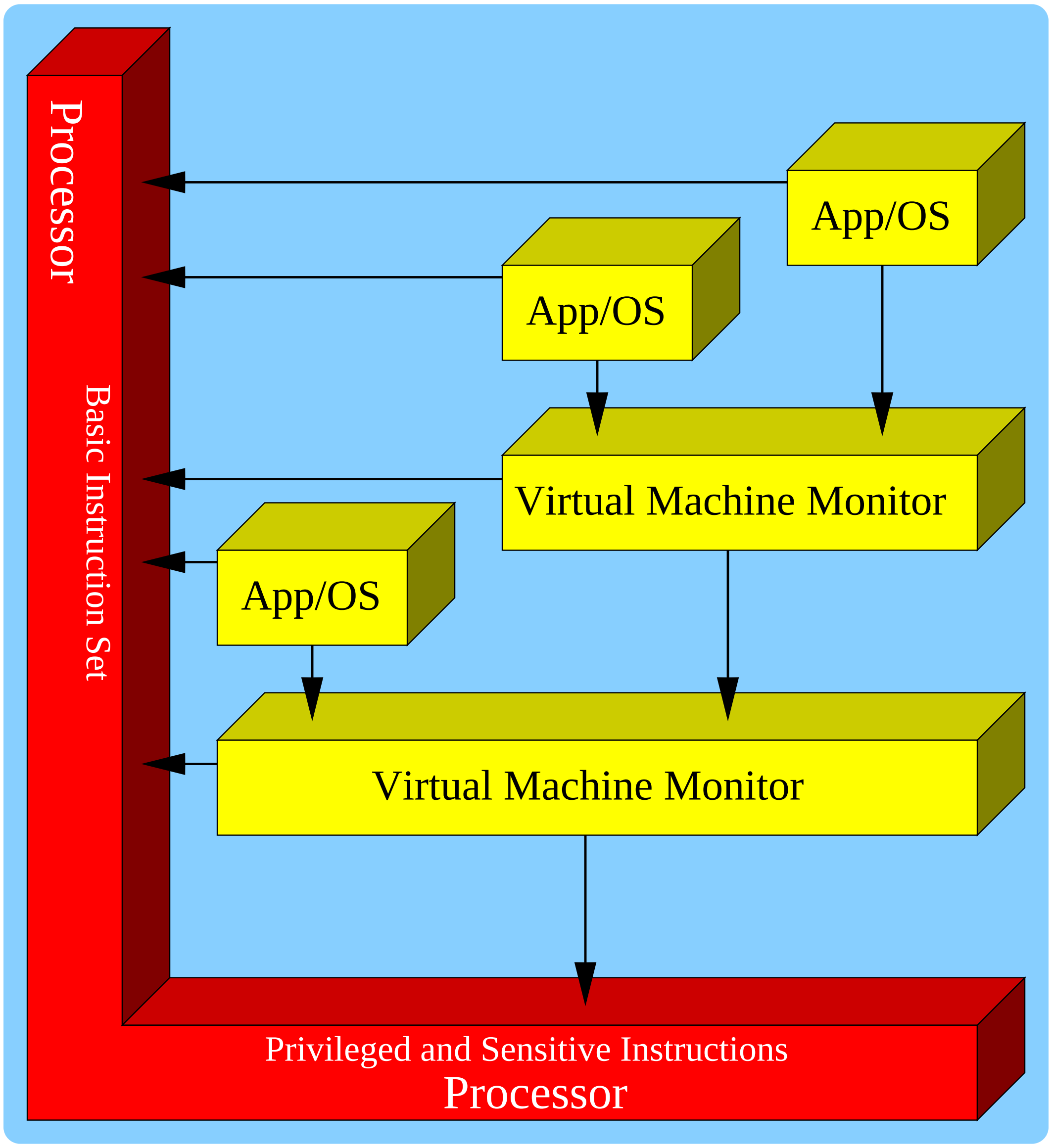


$\Rightarrow$ **modularity**, **extensibility**, **security**, ...

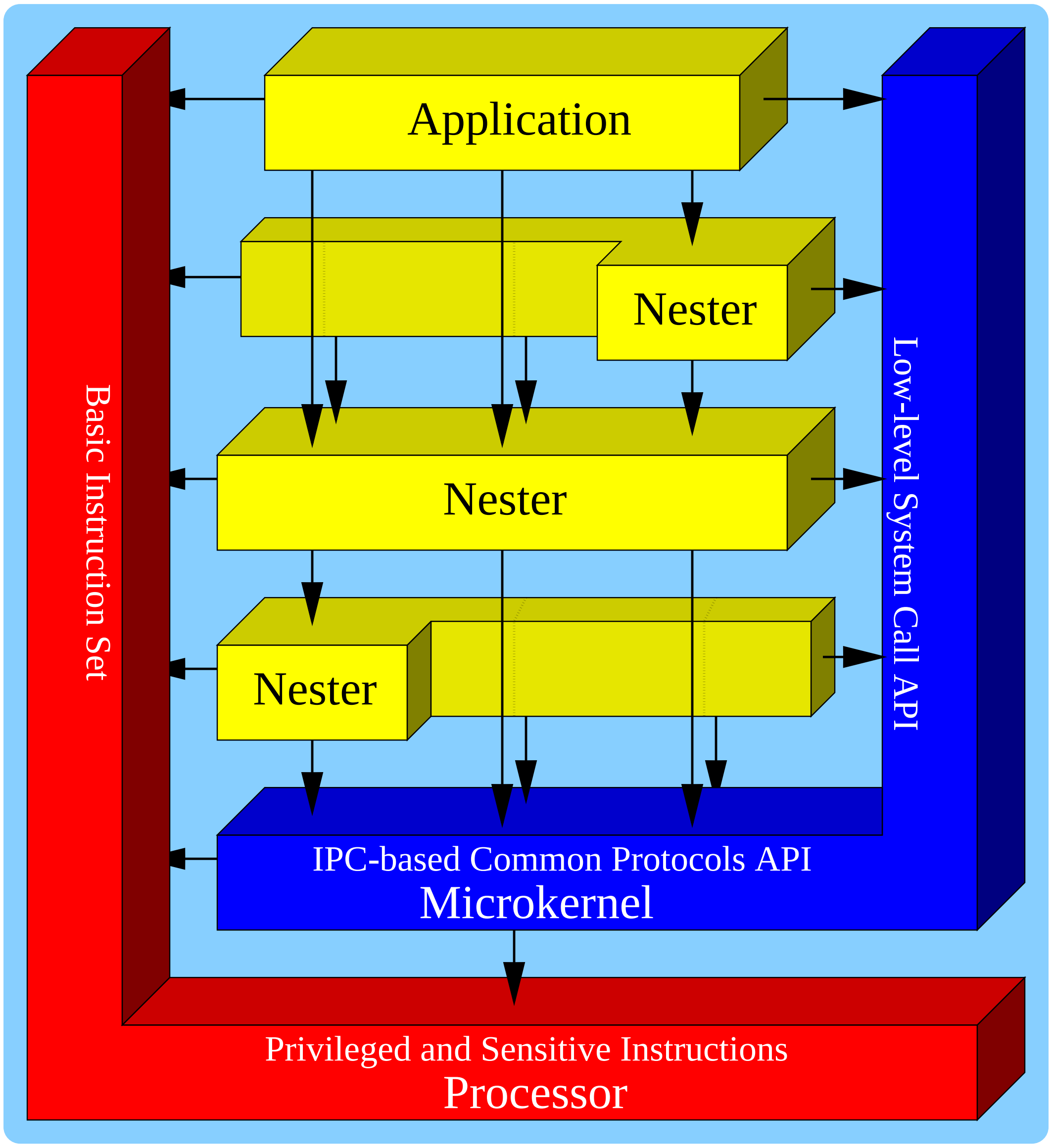
Virtual Machine Simulators



Virtual Machine Monitors



The Fluke Nested Process Architecture



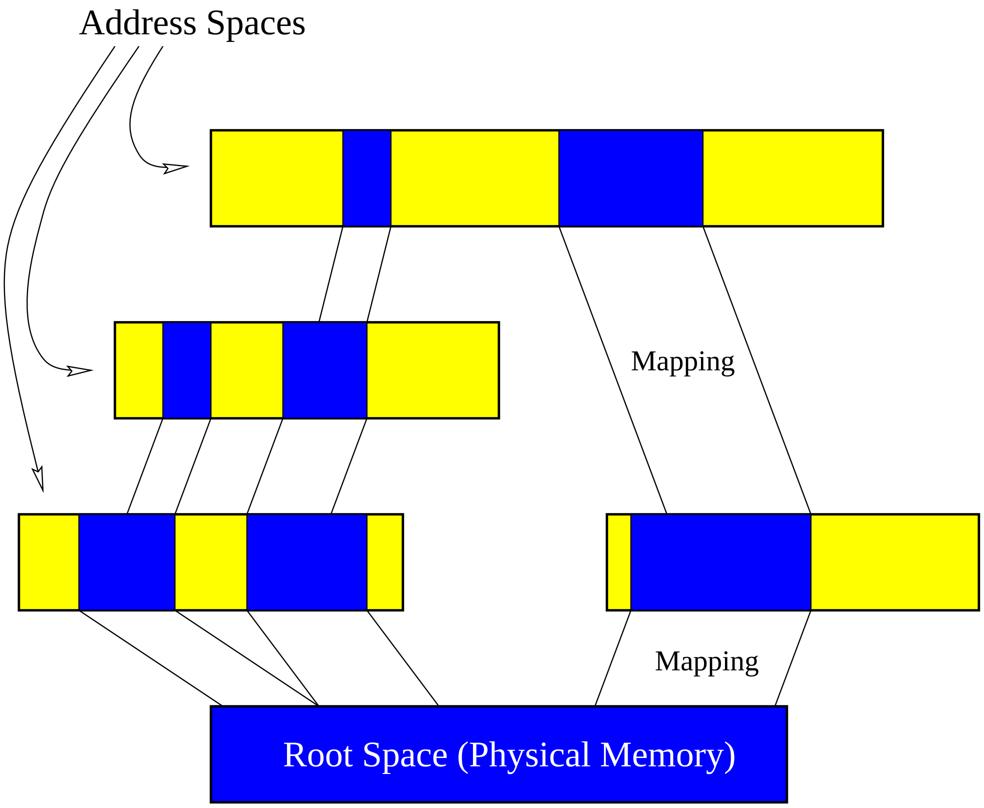
Hierarchical Resource Management

A child can *obtain* resources only through its parent.

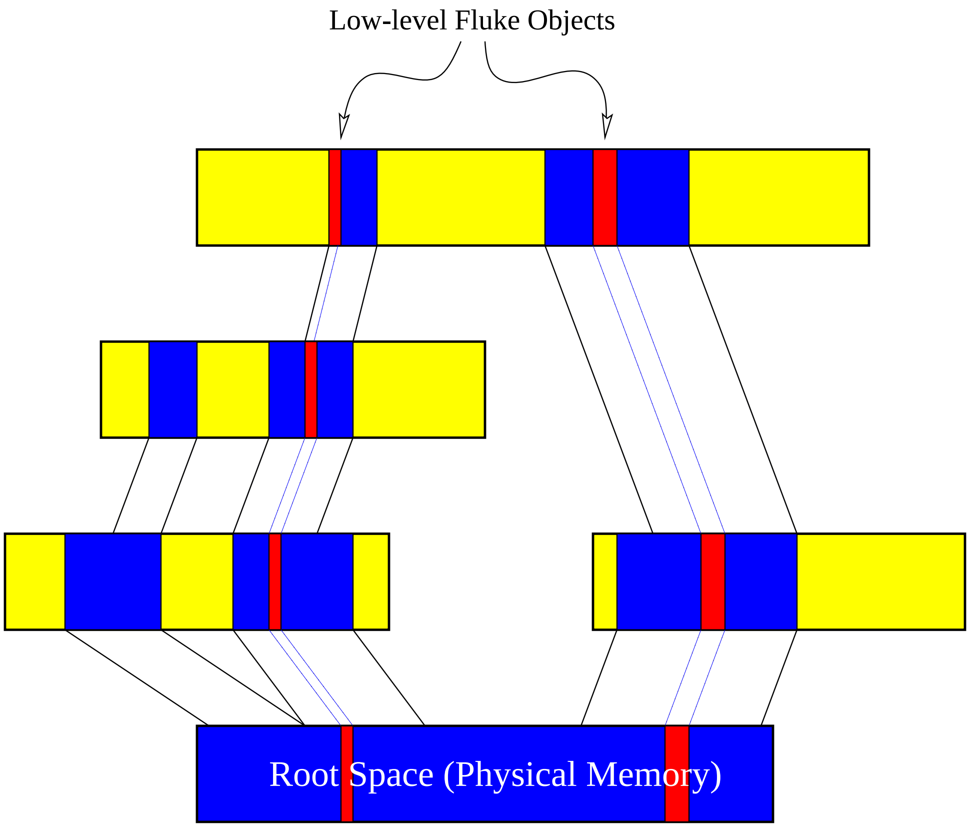
These resources are *managed* directly by the microkernel:

- **CPU time**: Hierarchical scheduling, e.g., CPU Inheritance Scheduling
- **Memory**: Relative Address Spaces
- **Kernel objects** (threads, ports, etc.): Relative Address Spaces

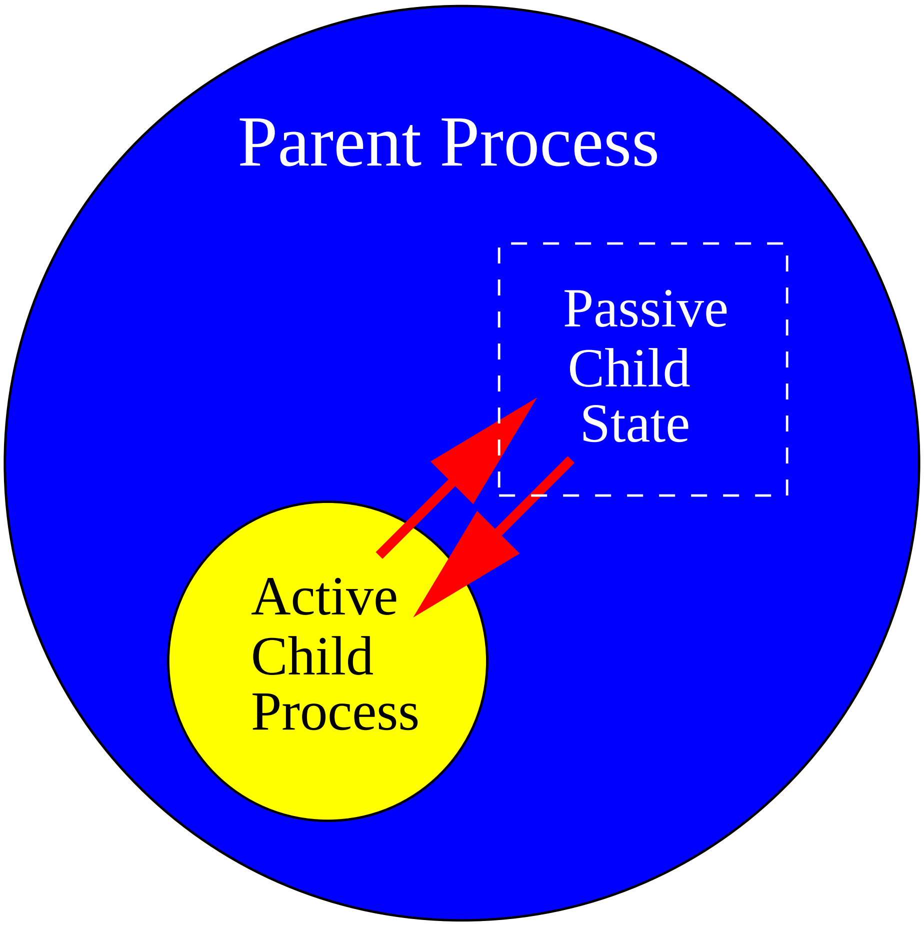
Relative Address Spaces



Fluke Low-level Objects



State Visibility

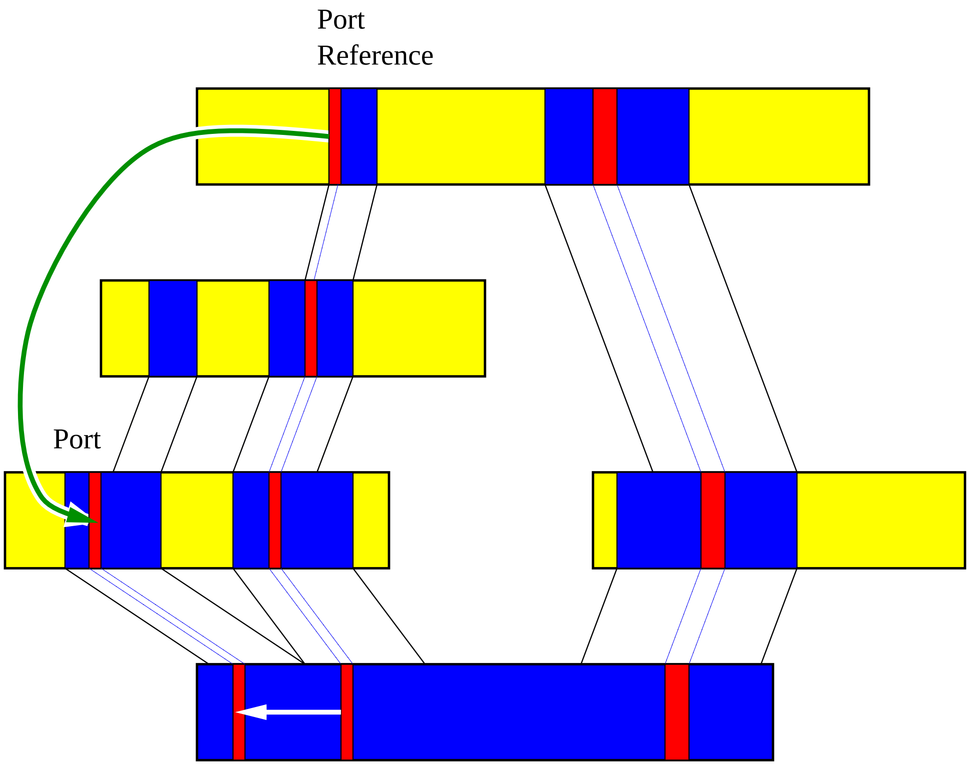


Relativity of Reference

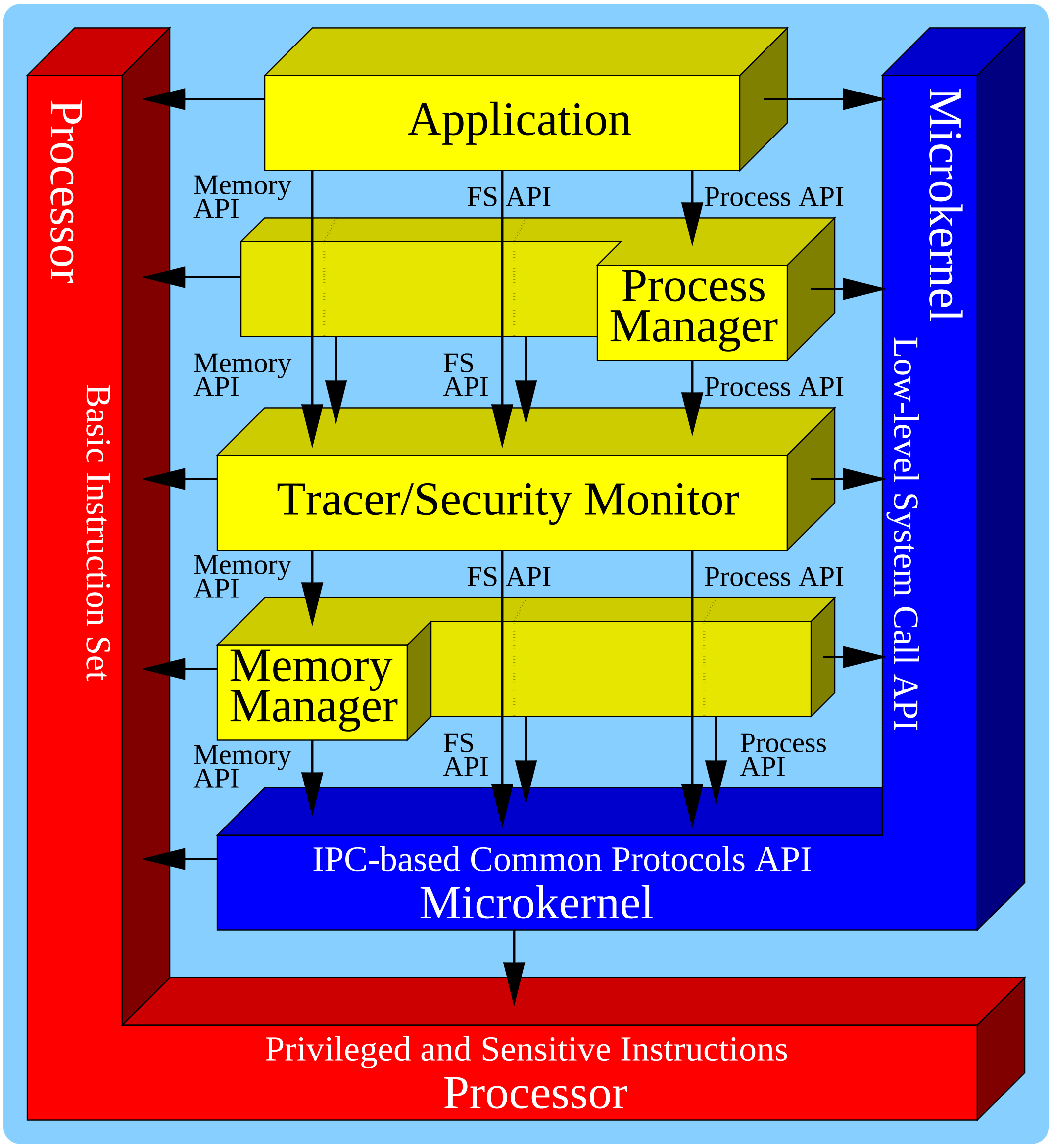
- Low-level API includes no *absolute* names, privileges, or resources.

- Classic kernel-mediated capability model gives relativity of cross-domain references and "short-circuiting" of nesting layers.

Fluke Capability Model



Fluke Architecture Components



High-level Common Protocols

Interfaces common to a set of cooperating nesters. For example,

`Parent::`	`get_process_service` `get_memory_service`
`Process::`	`create_child` `exec`
`Memory::`	`create_var_segment` `create_sub_pool`

`FileSystem::`	`open`
	`close`
	`mkdir`

`FileDescription::`	`read`
	`write`
	`map`

Nested Process Model Gives:

- **OS Modularity**: each service is implemented by a distinct process (a "nester")
- **Extensibility**: a modified service can be provided by a second nester or a modified nester
- **Composition**: nesters can be mixed and matched
- **Flexible scope**: can apply a service to a group of processes just as easily as to one
- **Security**: strong security mechanisms "for free"
- **Strong resource control**: provided by hierarchical structure
- **Flexibility**: strict hierarchy is *enabled*, not *enforced*

Prototype Implementation

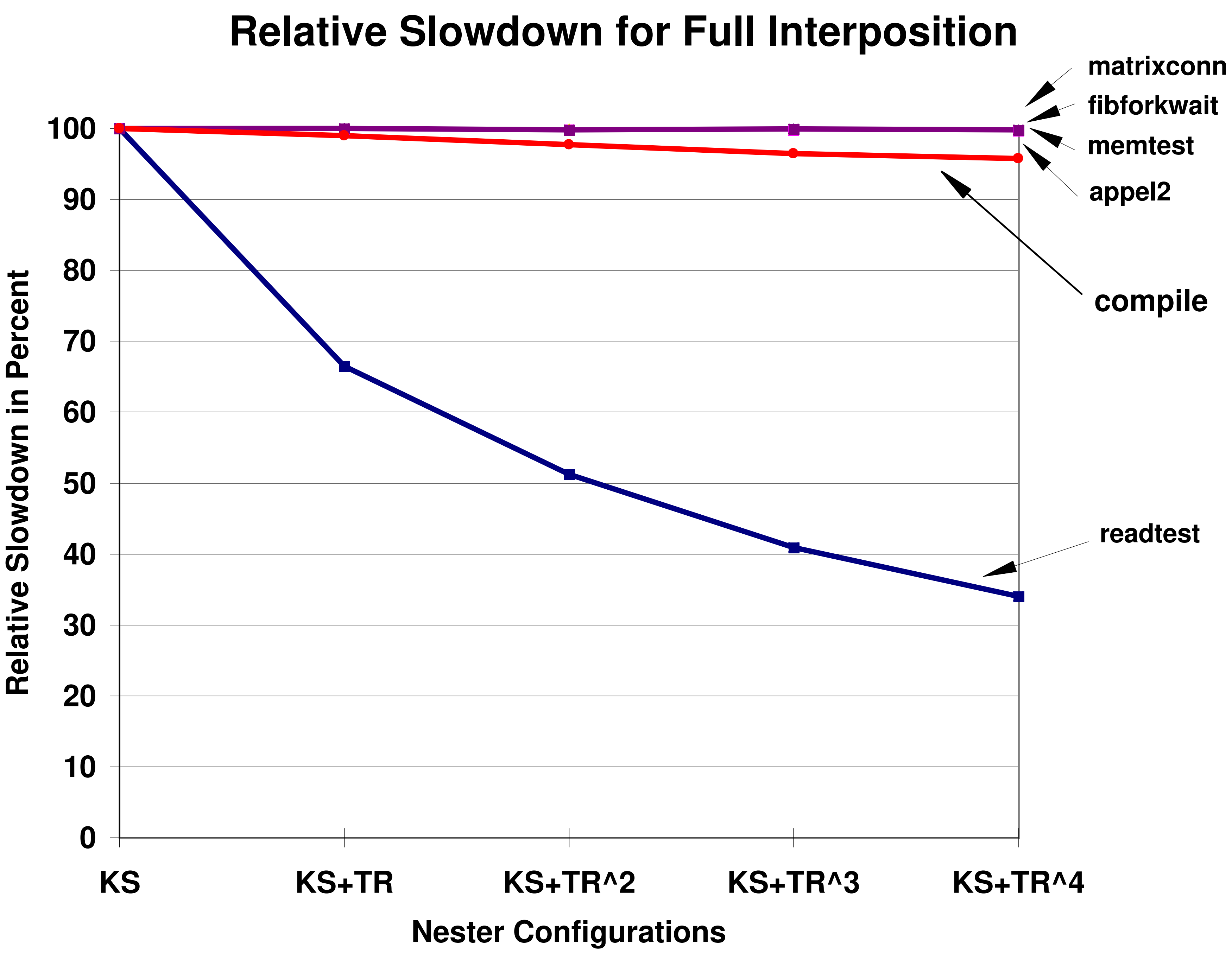
- Kernel: "portable," unoptimized, written in C
- Libraries
  - `libc`: provides client side of Common Protocols
  - `libnest`: provides server side of Common Protocols
- Nesters
  - Debugger
  - Tracer
  - Process Manager
  - Virtual Memory Manager
  - Checkpointer
  - Filesystem nester
- Applications:

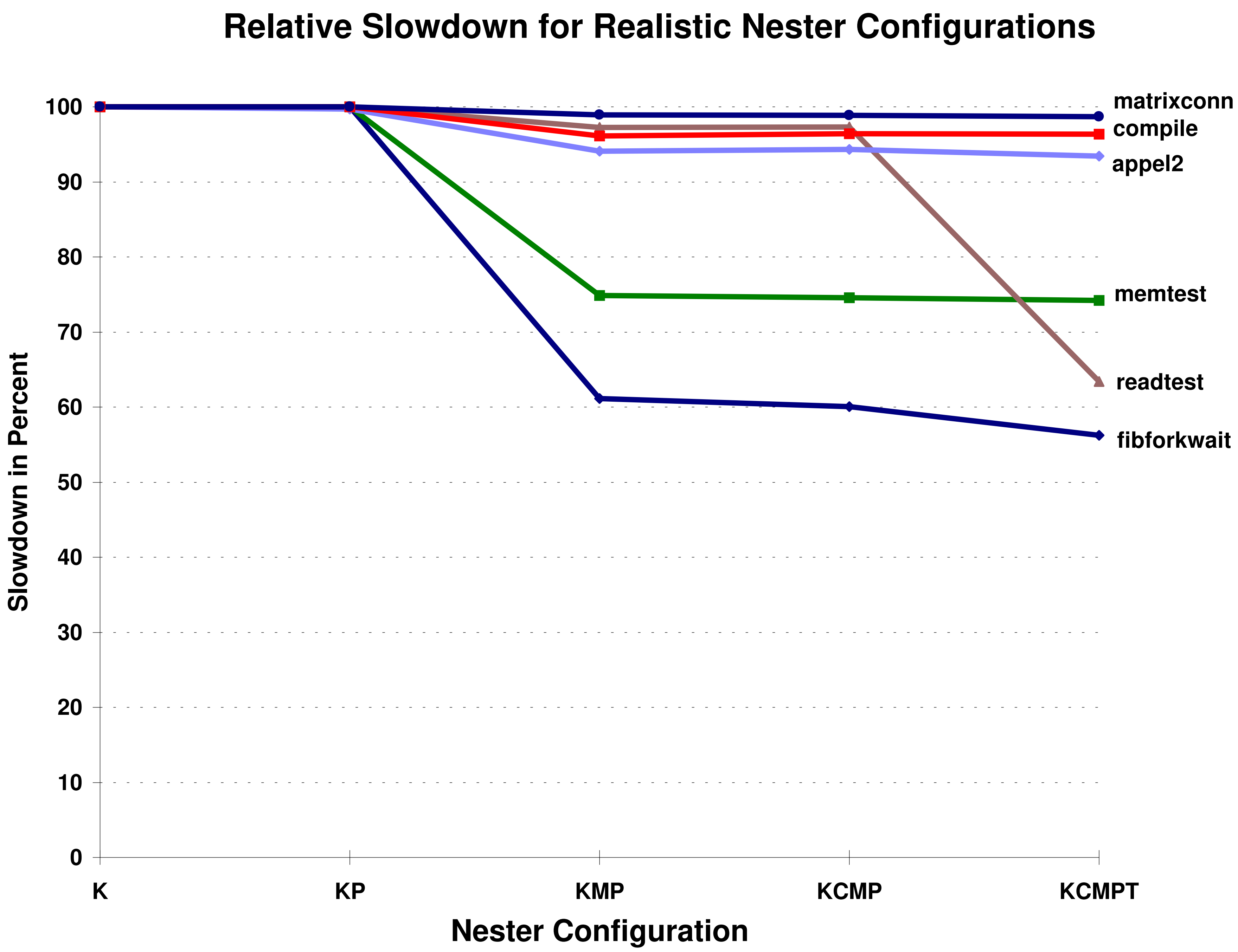
Results

- Absolute performance
- Relative slowdown

Test Programs

Test	Fluke	FreeBSD
`memtest`	929.1 ms	914.9 ms
`appel2`	5.4 ms	3.6 ms
`readtest`	125.8 ms	153.0 ms
`matconn`	102.9 ms	71.6 ms
`cc1`	3.83 sec	3.85 sec


Relative Slowdown for Full Interposition
matrixconn
fibforkwait
memtest
appel2
compile
readtest
Relative Slowdown in Percent
100
90
80
70
60
50
40
30
20
10
0
KS
KS+TR
KS+TR^2
KS+TR^3
KS+TR^4
Nester Configurations



Relative Slowdown for Realistic Nester Configurations
Slowdown in Percent
100
90
80
70
60
50
40
30
20
10
0
K
KP
KMP
KCMP
KCMPT
Nester Configuration
matrixconn
compile
appel2
memtest
readtest
fibforkwait

Related Work

- CAP: early nested process architecture
- L4, Grasshopper: memory remapping
- System 38, Intel i960XA:
  "tagged memory"
- Amoeba, Cache Kernel: state accessibility, with some constraints
- Stackable filesystems & network protocols: domain-specific stacking

Status

- Kernel, libraries, nesters as above: supports POSIX subset on x86

- Kernel API published

- Source release within a few months

- Portable prototype not fast; however...

GNU Apps Running on Fluke

- **compile**: gcc cpp cc1 make gawk bsdsed bash-batch
- **binutils**: gas ld ar objcopy objdump ranlib size strings nm strip gprof
- **fileutils**: chgrp chmod chown cp dd dir dircolors du ginstall ln ls mkdir mkfifo mknod mv rm rmdir sync touch vdir
- **textutils**: cat cksum comm csplit cut expand fmt fold head join md5sum nl od paste pr sort split sum tac tail tr unexpand uniq wc
- **diffutils**: cmp diff diff3 sdiff
- **shellutils**: basename date dirname echo env expr factor false groups hostname id logname pathchk printenv printf pwd seq sleep stty tee test true tty uname users who whoami yes

Conclusion

Fluke combines principles of microkernels and virtual machines to support:

- Complete state encapsulation and control
- Efficiently stackable OS services
- Modularized VM, process management, debugging, tracing, checkpointing.