Improving the Reliability of Commodity Operating Systems

Mike Swift, Brian Bershad, Hank Levy

University of Washington

Outline

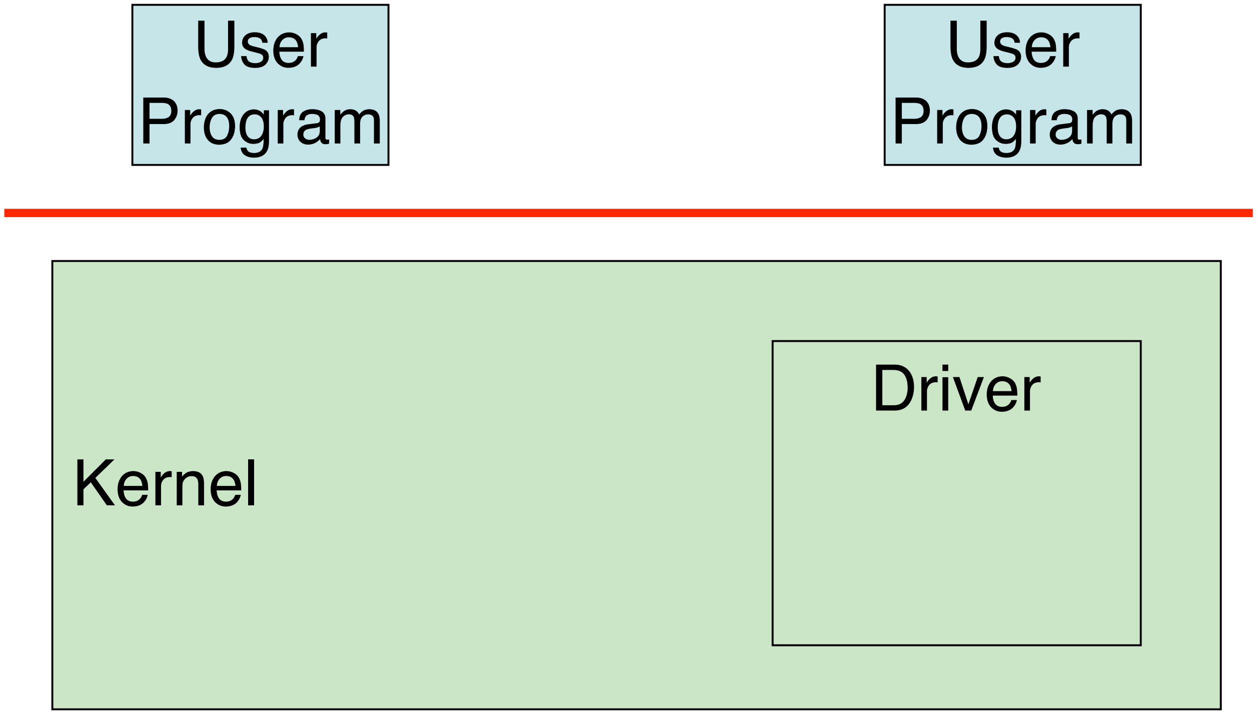
- Introduction
- Vision
- Design
- Evaluation
- Summary

The Problem

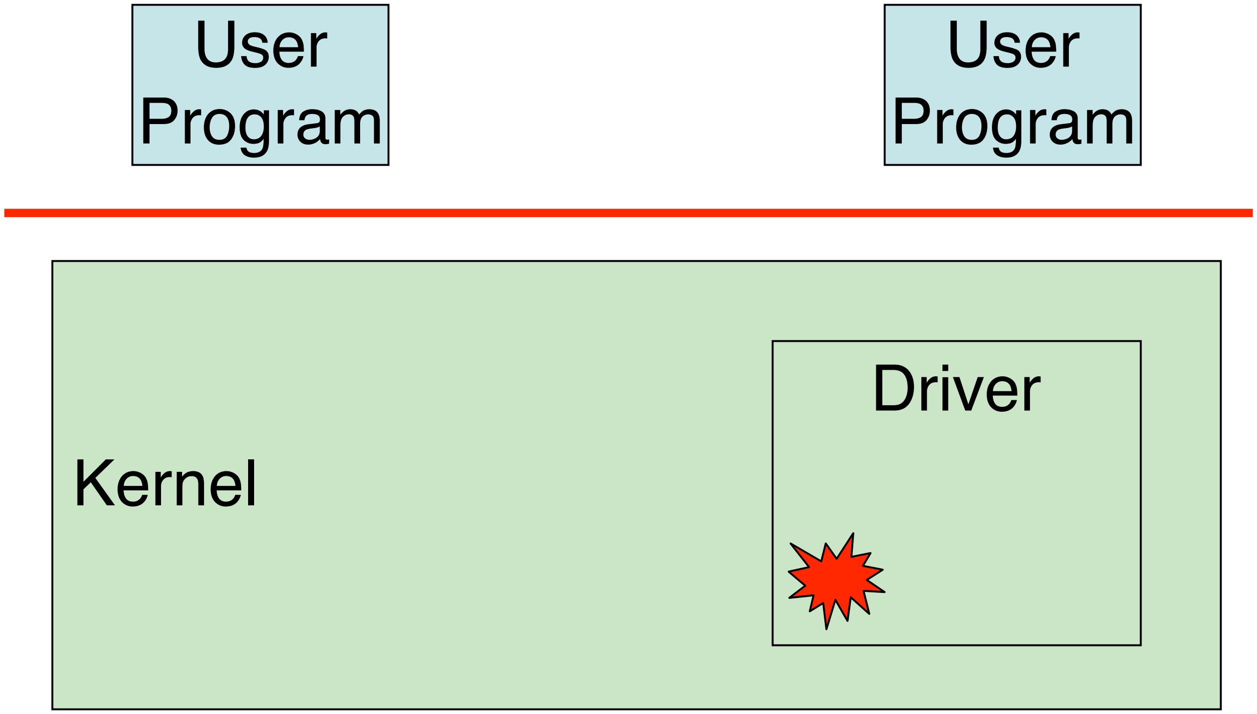
- Operating system crashes are a huge problem today
  - 5% of Windows systems crash every day
- Device drivers are the biggest cause of crashes
  - Drivers cause 85% of Windows XP crashes
  - Drivers are 7 times buggier than the kernel in Linux

- We built Nooks, a system that prevents drivers from crashing the OS
  - We can prevent 99% of faults in our tests that crash native Linux

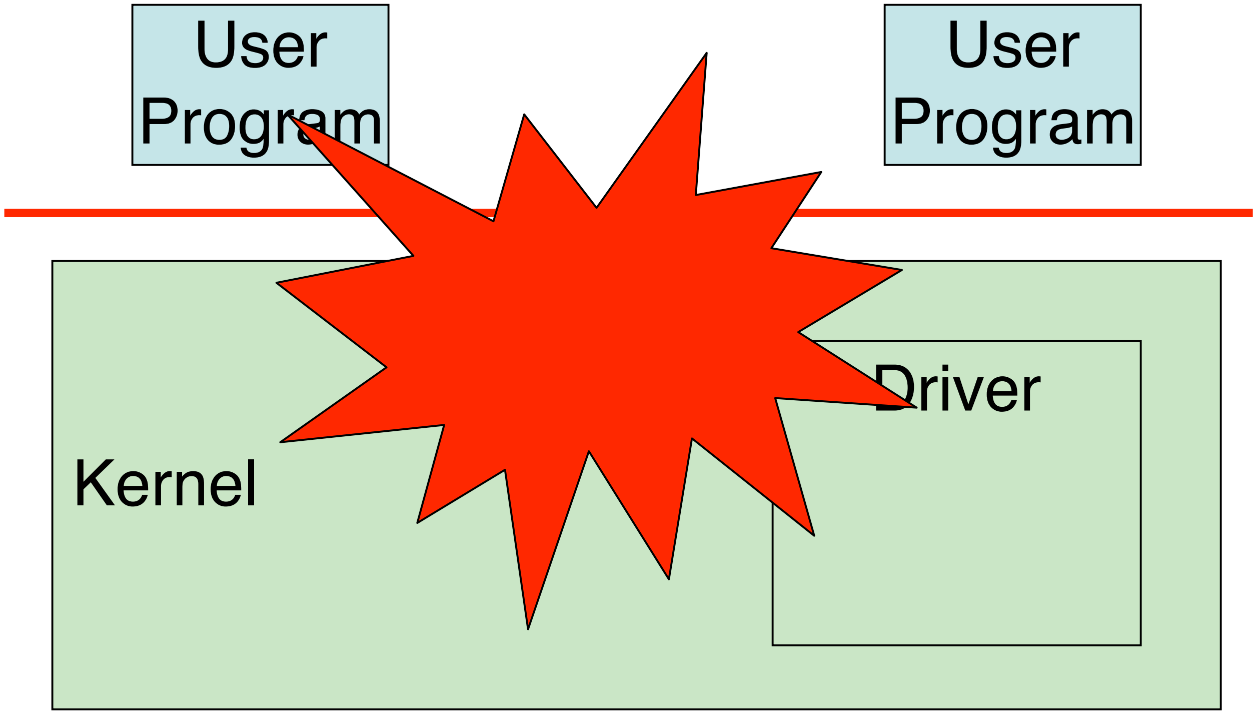
Crashes Today



Crashes Today



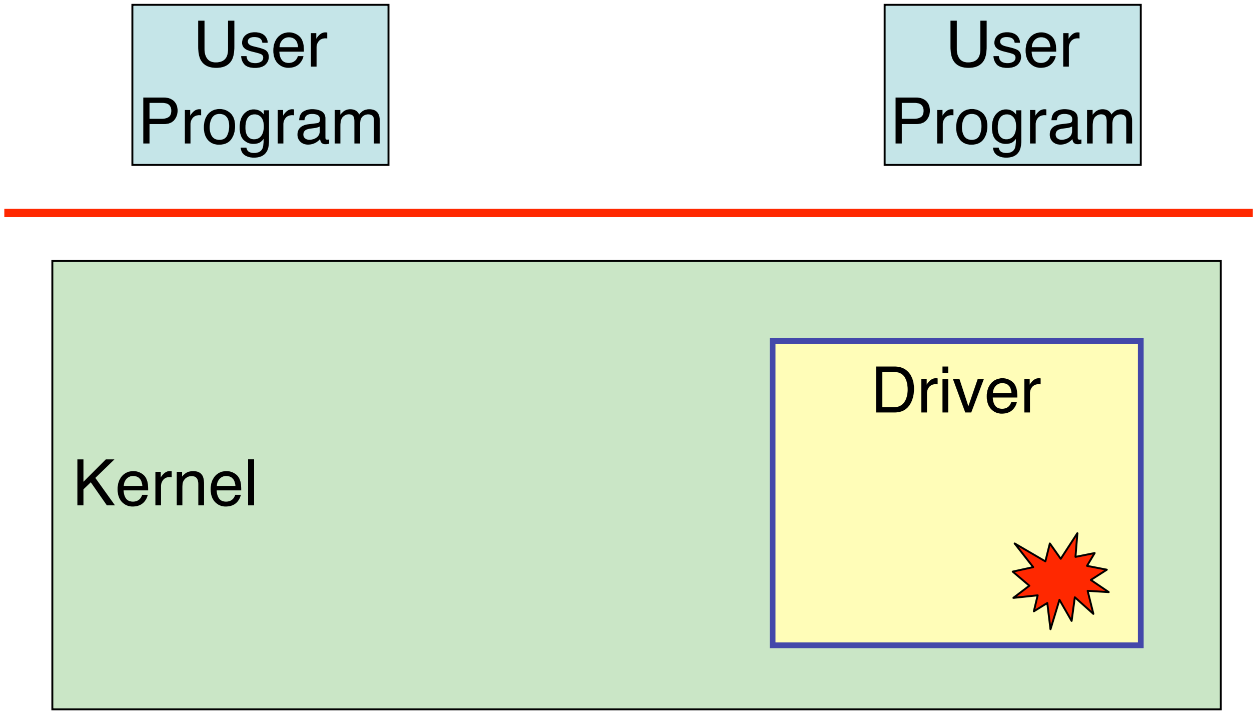
Crashes Today



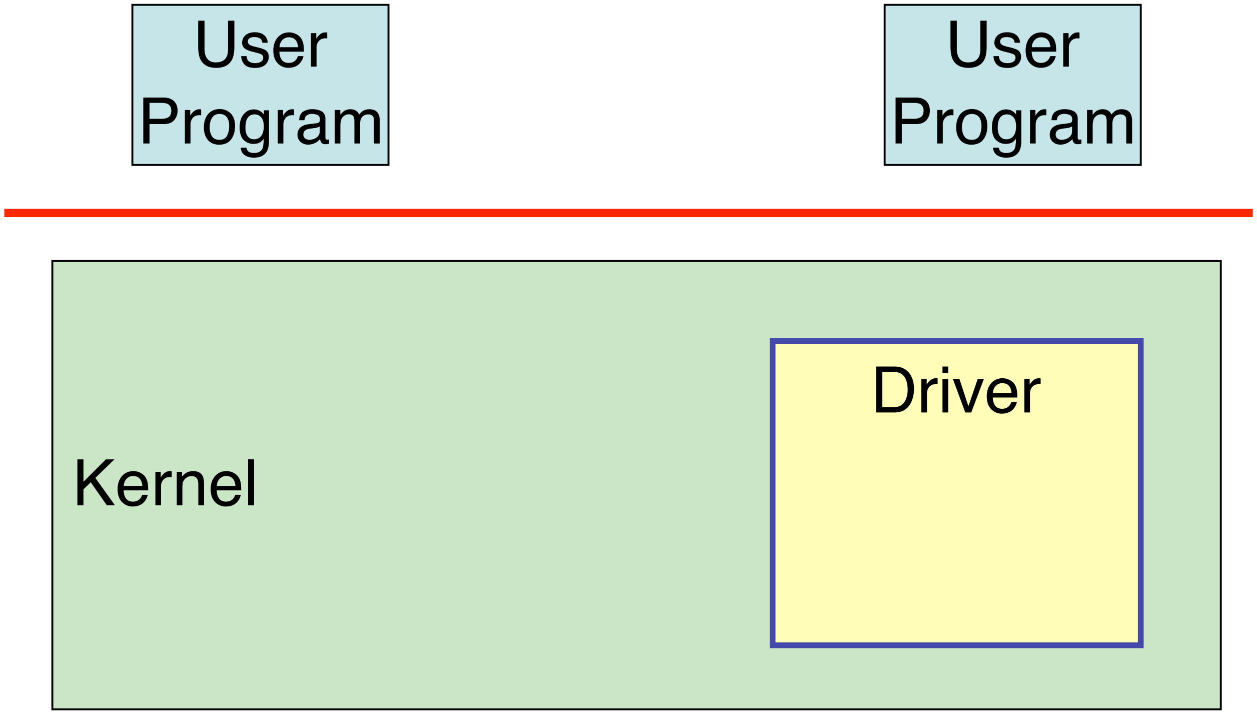
Outline

- Introduction
- Vision
- Design
- Evaluation
- Summary

Vision



Vision



Reality

- Windows XP
  - 113 million copies sold in 2002
  - 40 million lines of code
  - $1 billion development cost
  - 35,000 drivers available
- Linux:
  - 18 million users
  - 30 million lines of code
  - Equivalent $1 billion development cost

Vision Requirements

1. Isolation
2. Recovery
3. Compatibility
    - No code changes
    - No new languages
    - No new OS
    - No new hardware
    - *No new perspective*

Outline

- Introduction
- Vision
- Design
- Evaluation
- Summary

Assumptions and Principles

- Assumptions:
  - Drivers are generally well behaved
  - Don’t need to prevent every crash to be useful
- Principles:
  - Design for fault resistance (not fault tolerance)
  - Design for mistakes (not abuse)

Goal

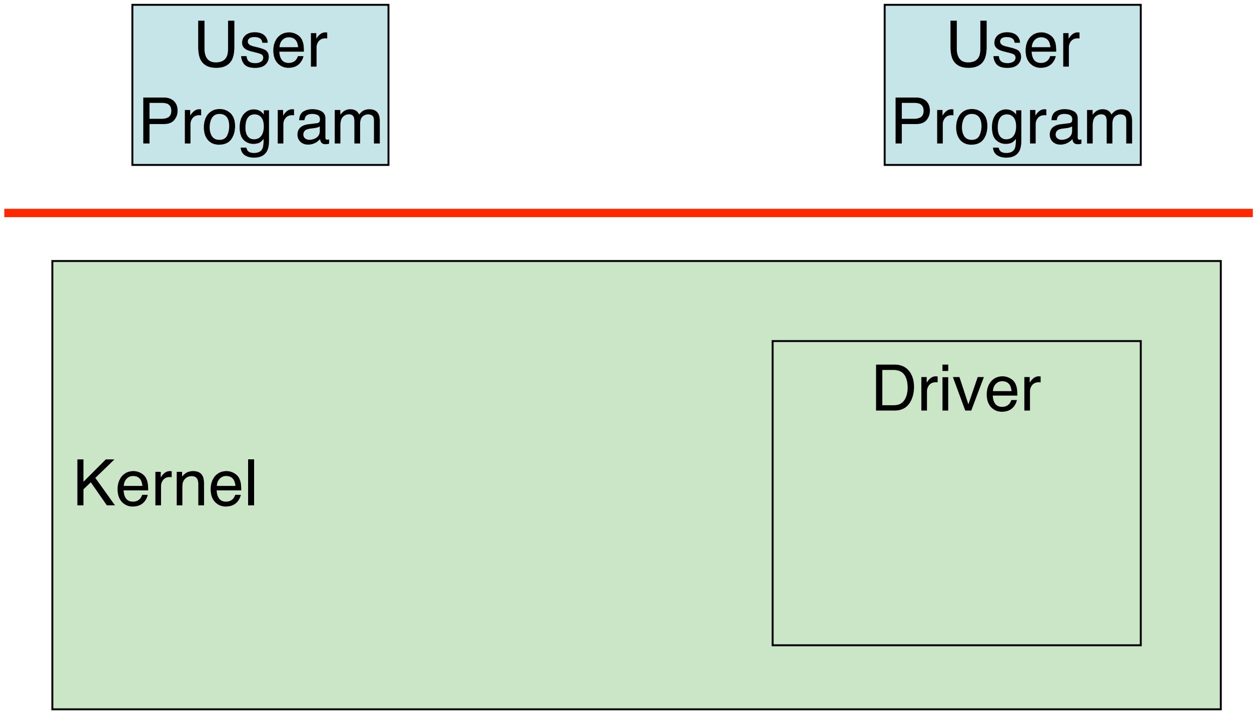
We want a practical, “best-effort” solution

- Prevents many crashes
- Good performance
- Works with today’s operating systems and drivers

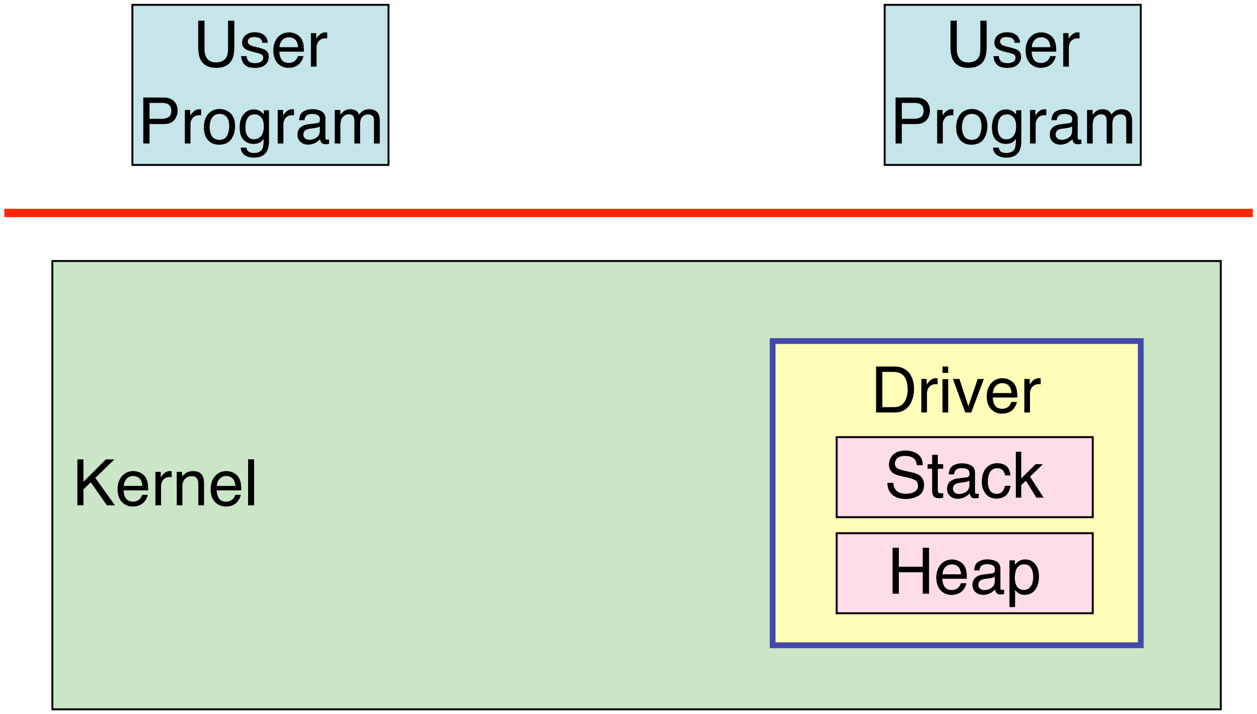
Design of Nooks

- Standard Linux kernel and drivers
- Plus:
  - Isolation
  - Recovery
- Compatible with existing code

Existing Kernels

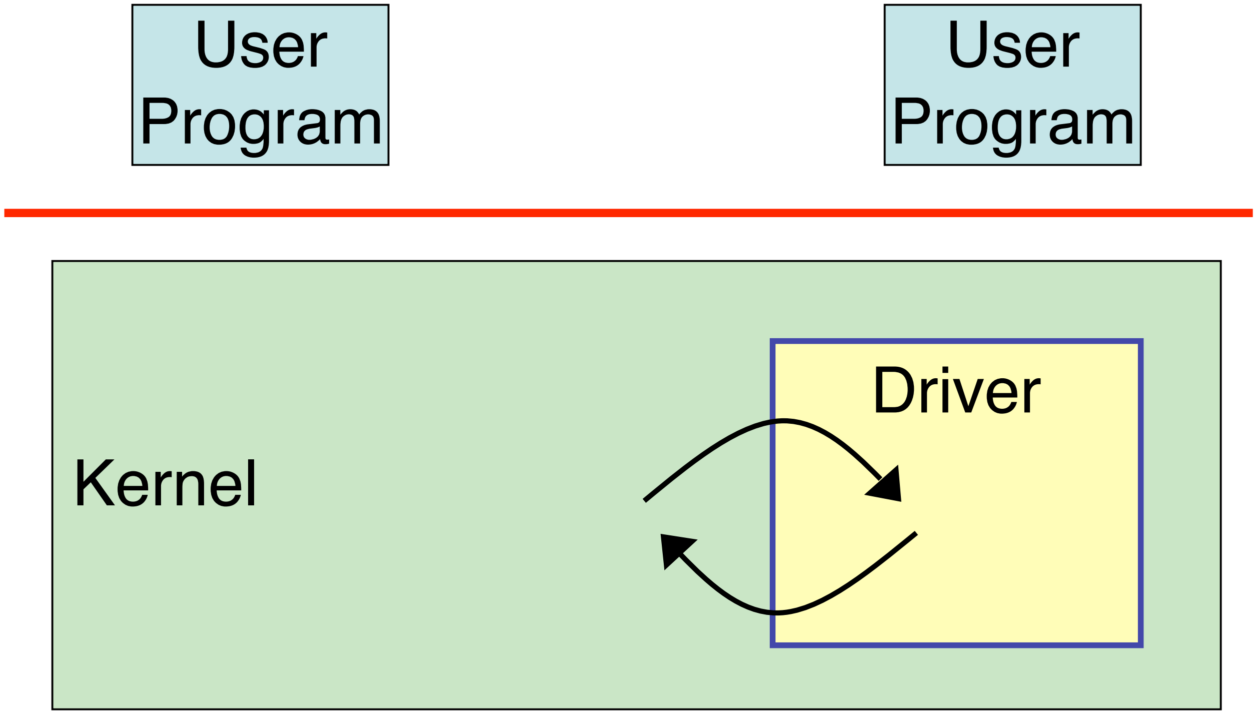


Isolation - Memory

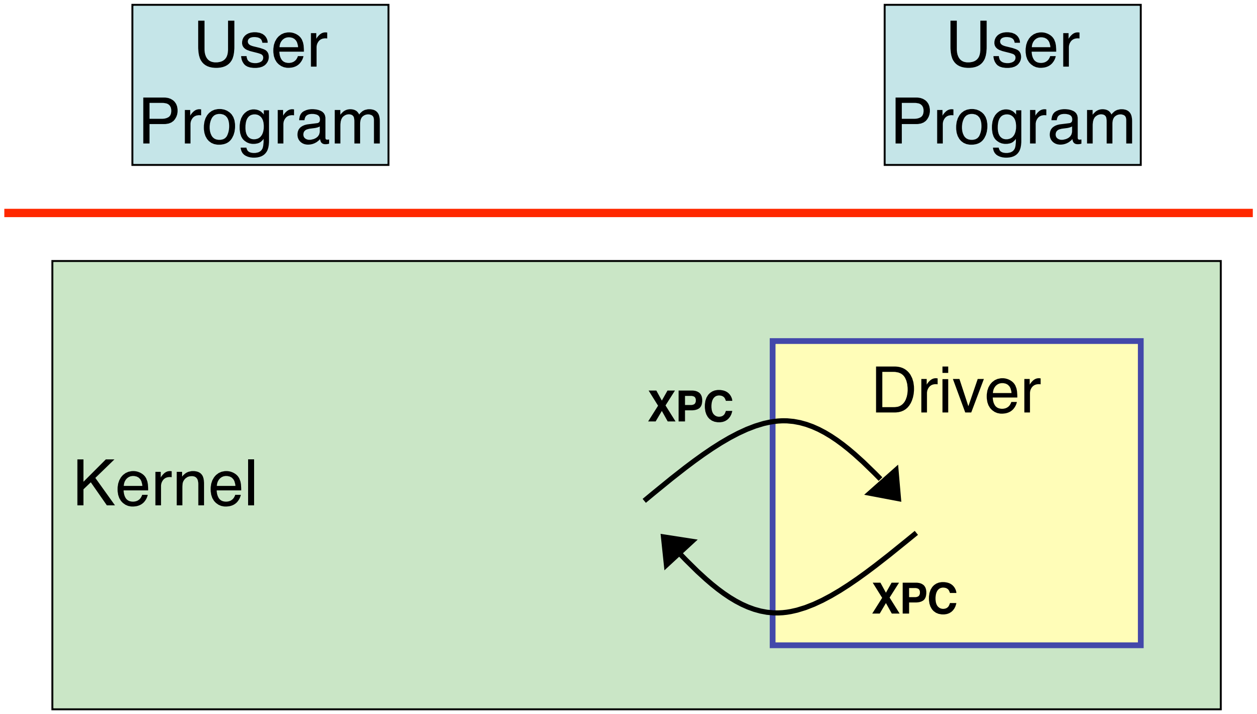


Lightweight Kernel Protection Domains

Isolation - Control Transfer

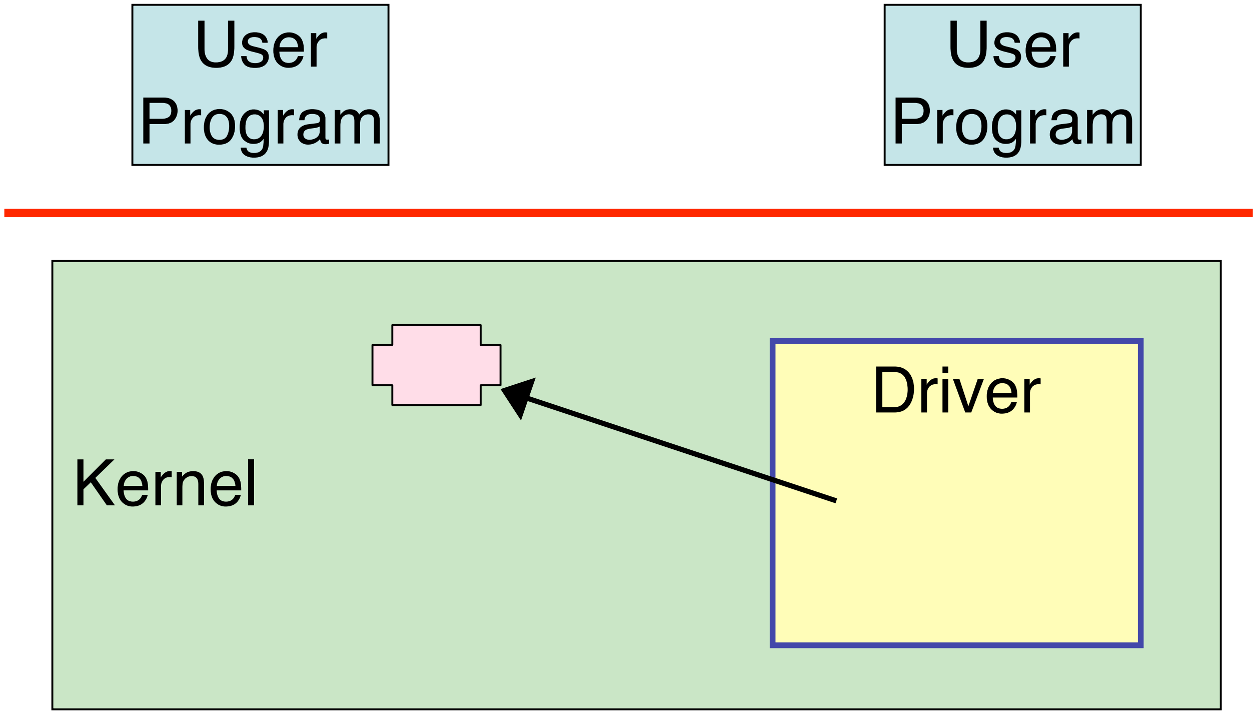


Isolation - Control Transfer

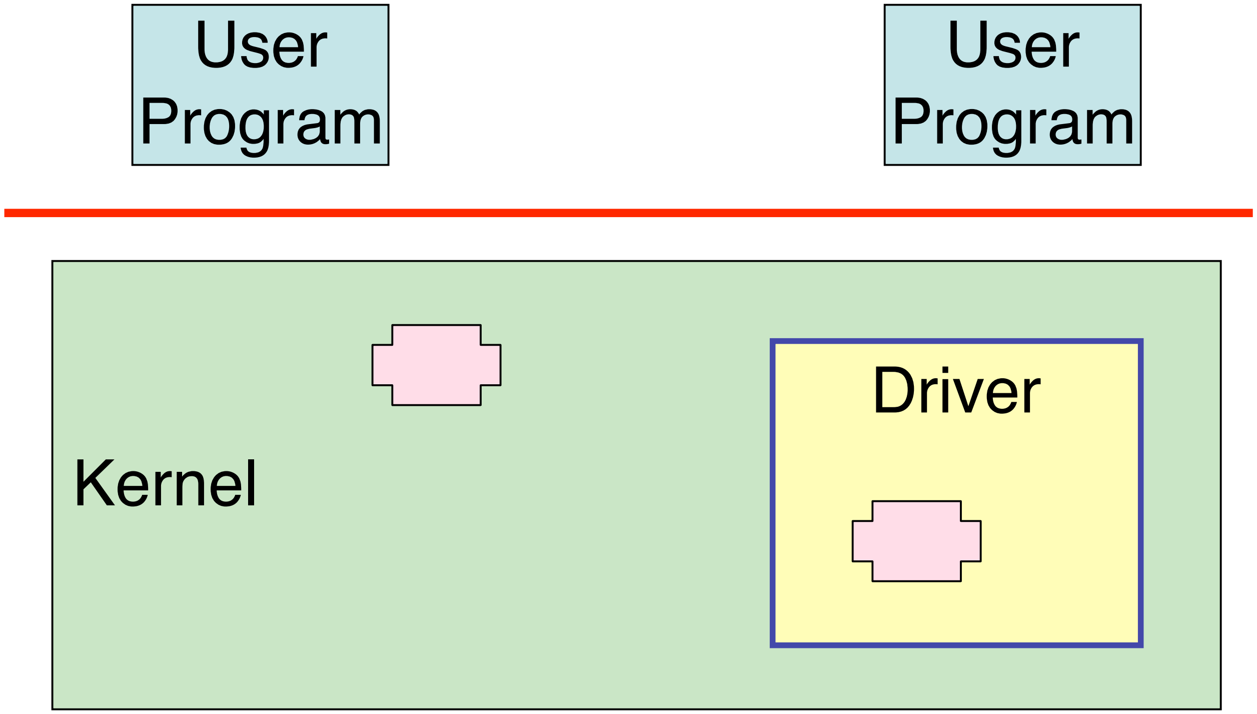


eXtension Procedure Call

Isolation - Data Access

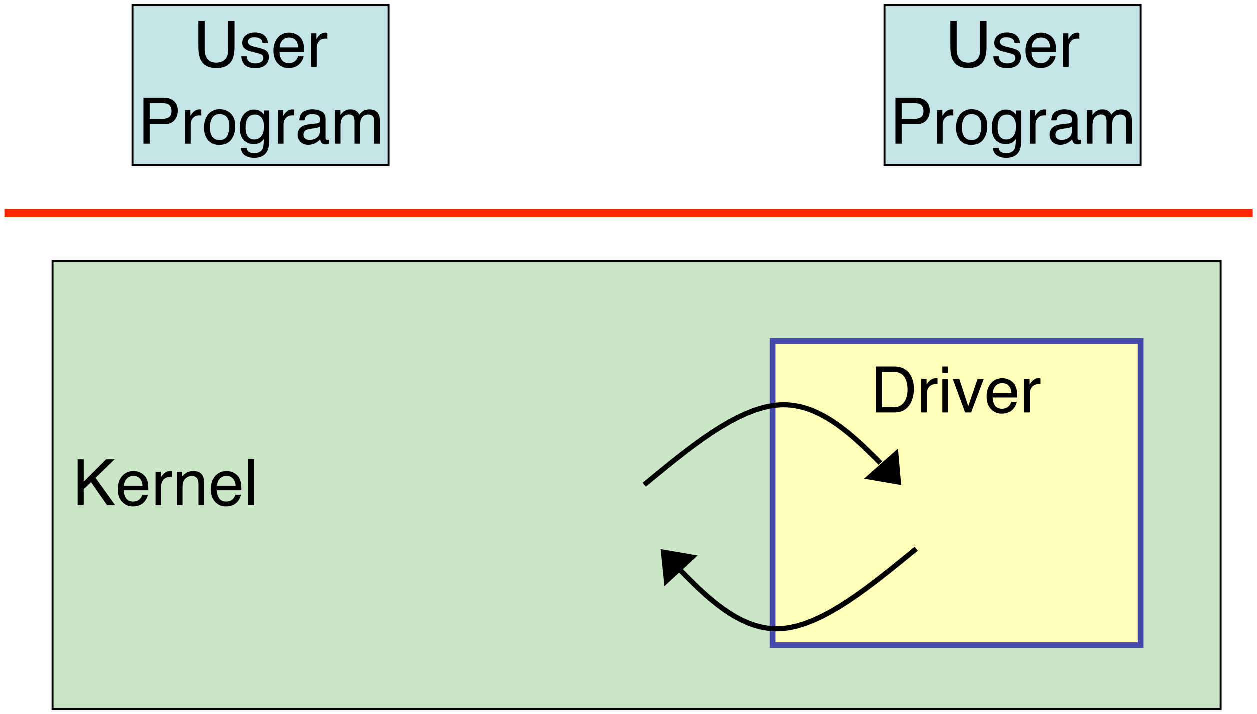


Isolation - Data Access

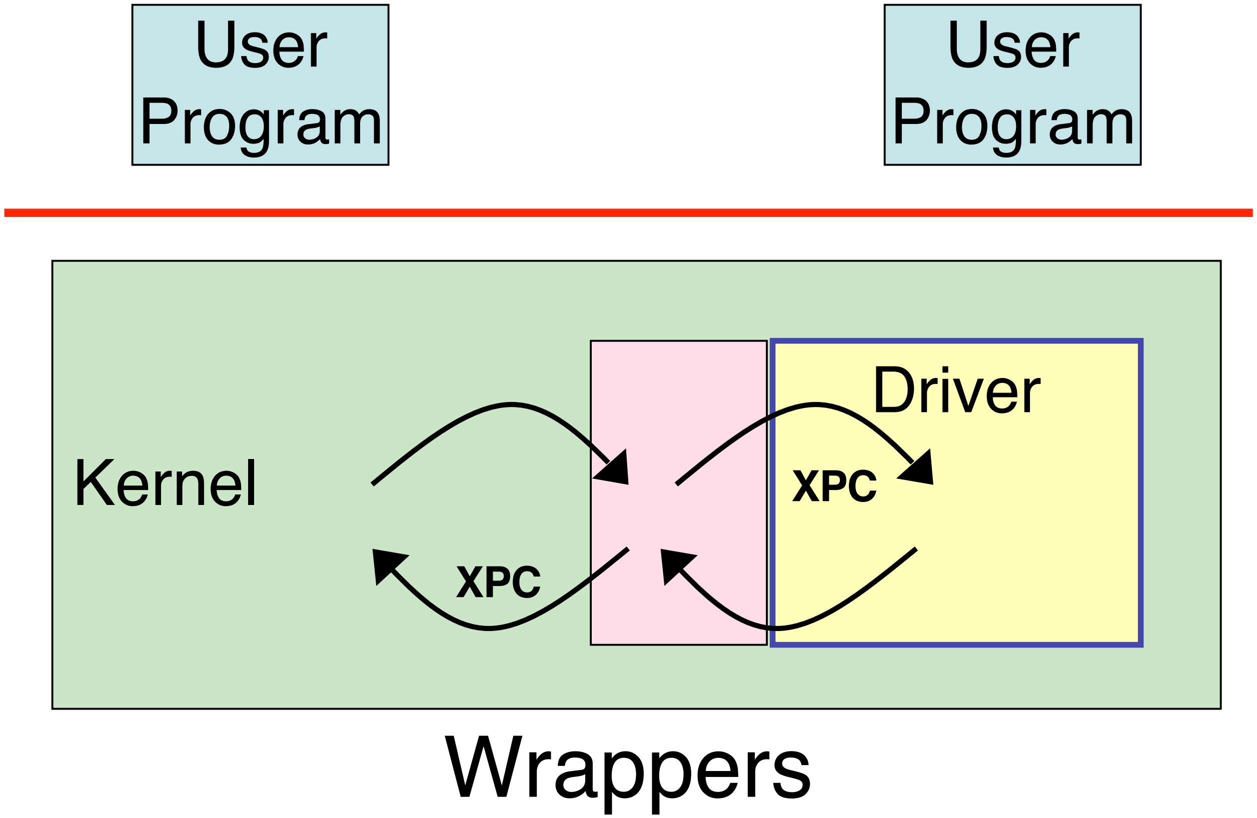


Copy-in / Copy-out

Isolation - Interposition



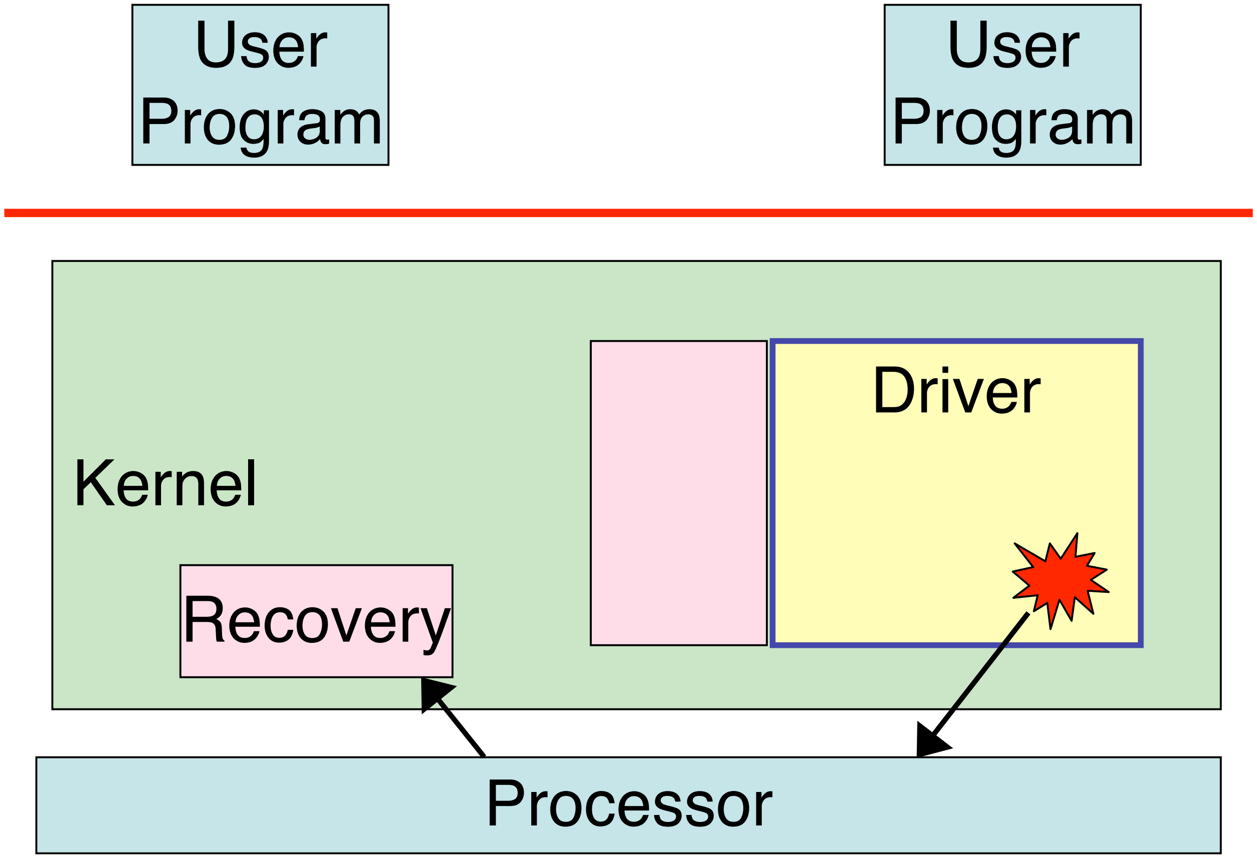
Isolation - Interposition



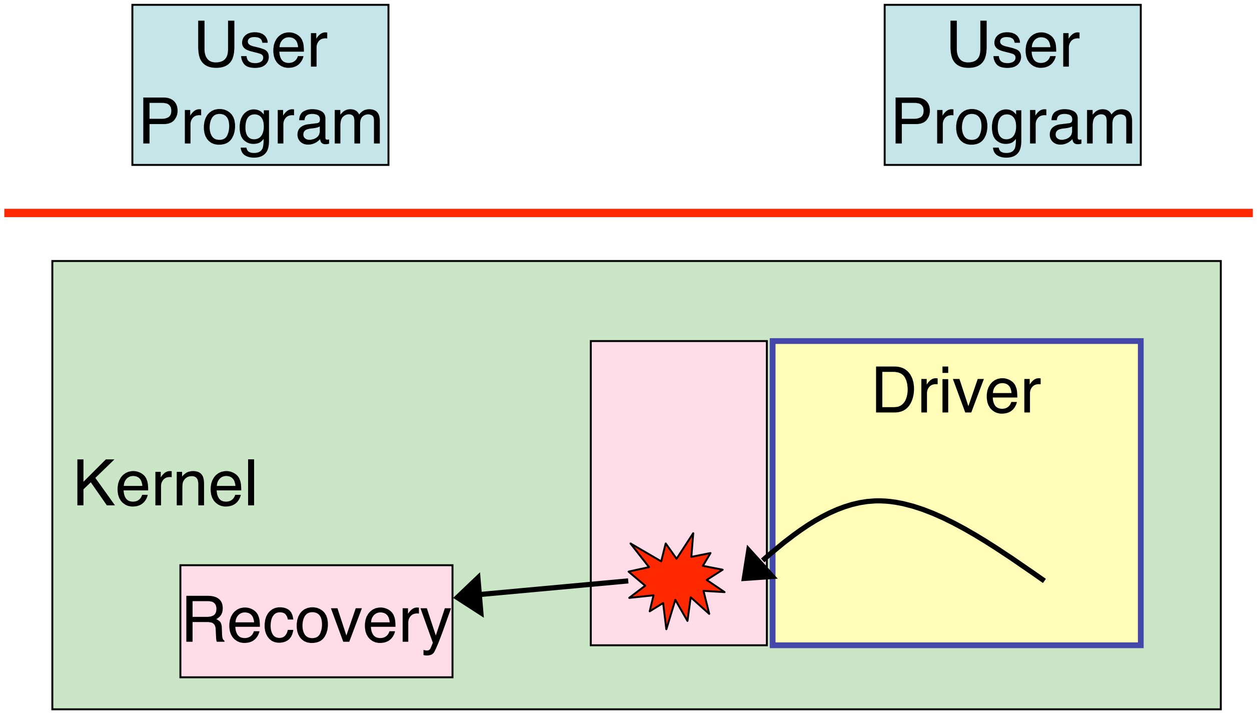
Design Summary

- Isolation
  - Lightweight Kernel Protection Domains
  - eXtension Procedure Call (XPC)
  - Copy-in/Copy-out
  - Wrappers

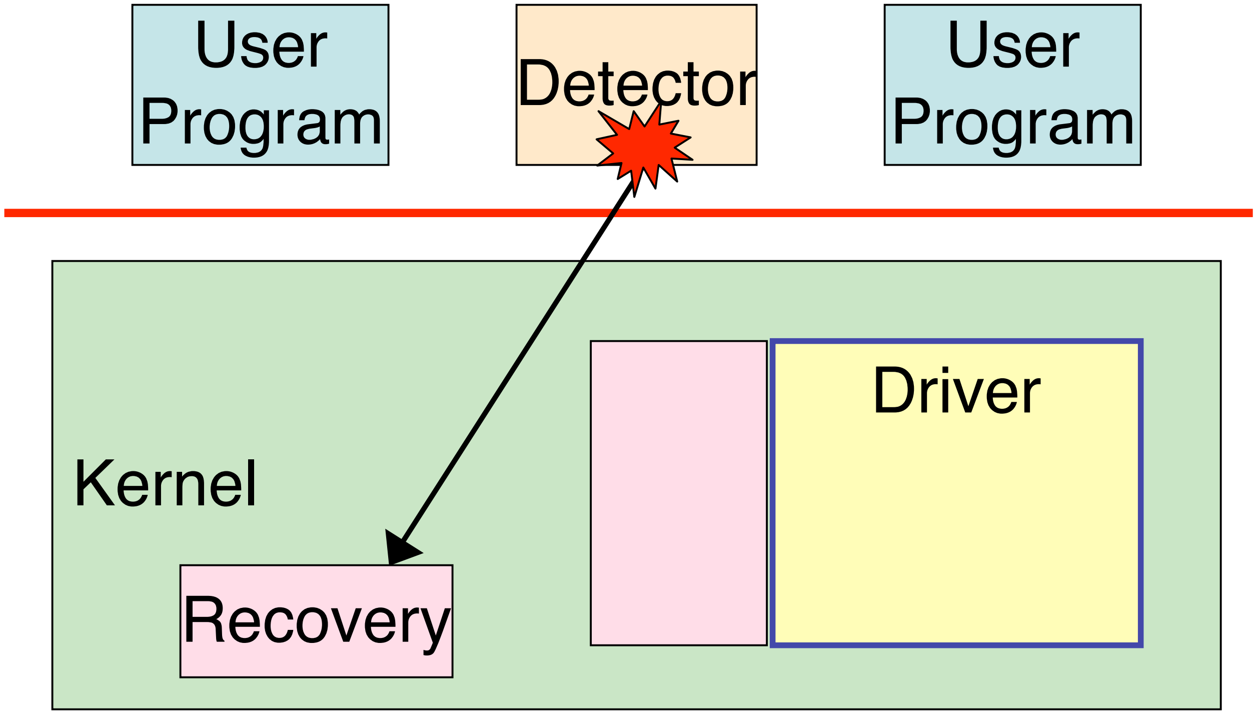
Recovery - Fault Detection



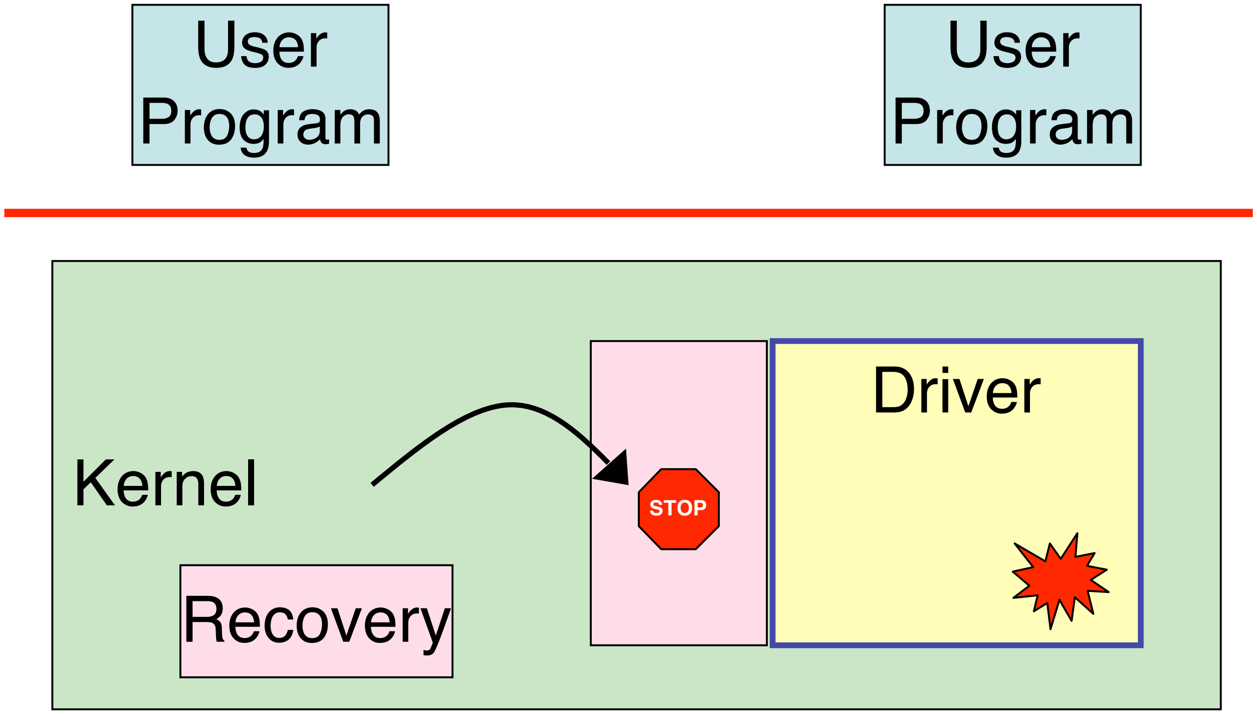
Recovery - Fault Detection



Recovery - Fault Detection

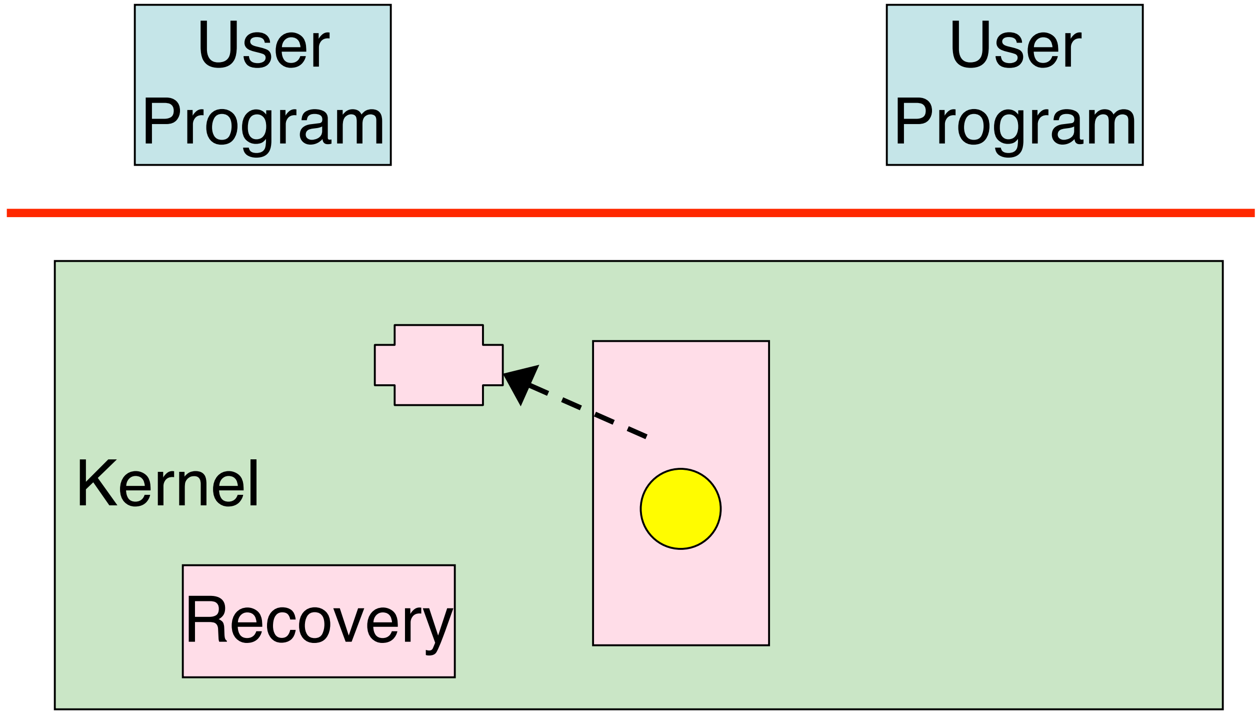


Recovery



Stop

Recovery



User Program

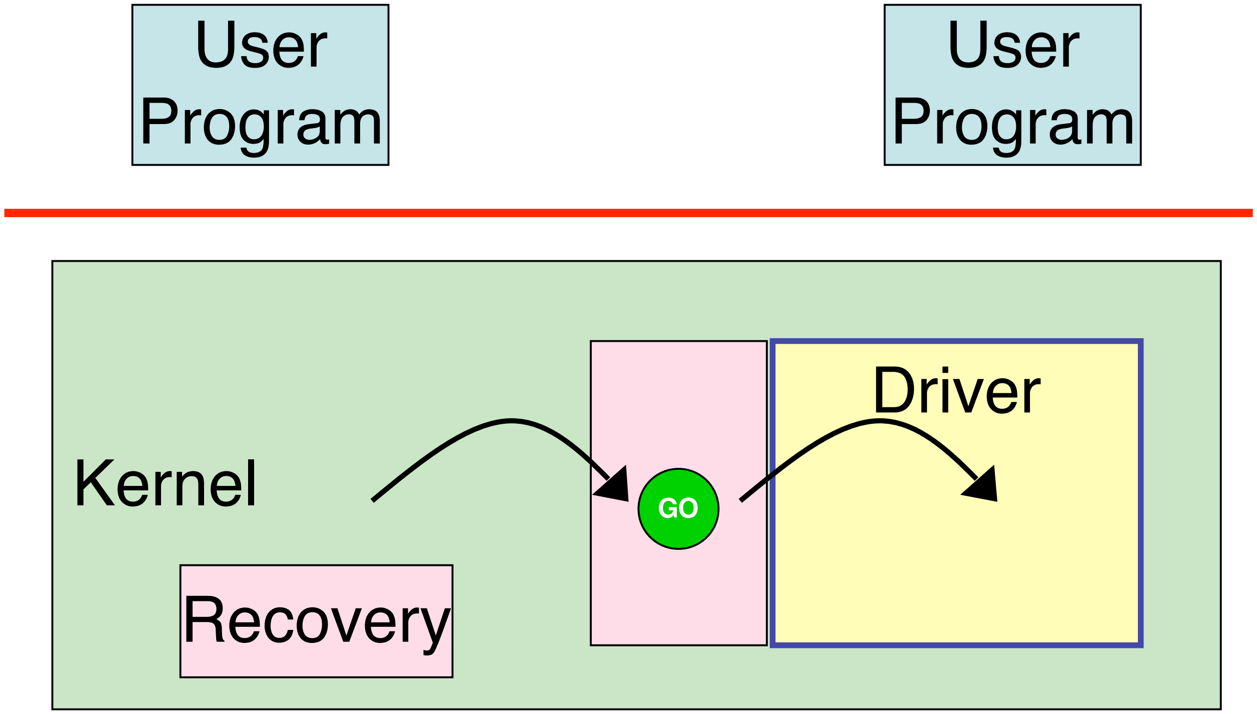
User Program

Kernel

Recovery

Stop / Unload

Recovery



Stop / Unload / Reload

Design Summary

- Isolation
  - Lightweight Kernel Protection Domains
  - eXtension Procedure Call (XPC)
  - Copy-in/Copy-out
  - Wrappers
- Recovery
  - Hardware and software checks
  - Stop / Unload and GC / Reload

Some Limitations

- Blame the processor
- Blame the operating system
- Blame us

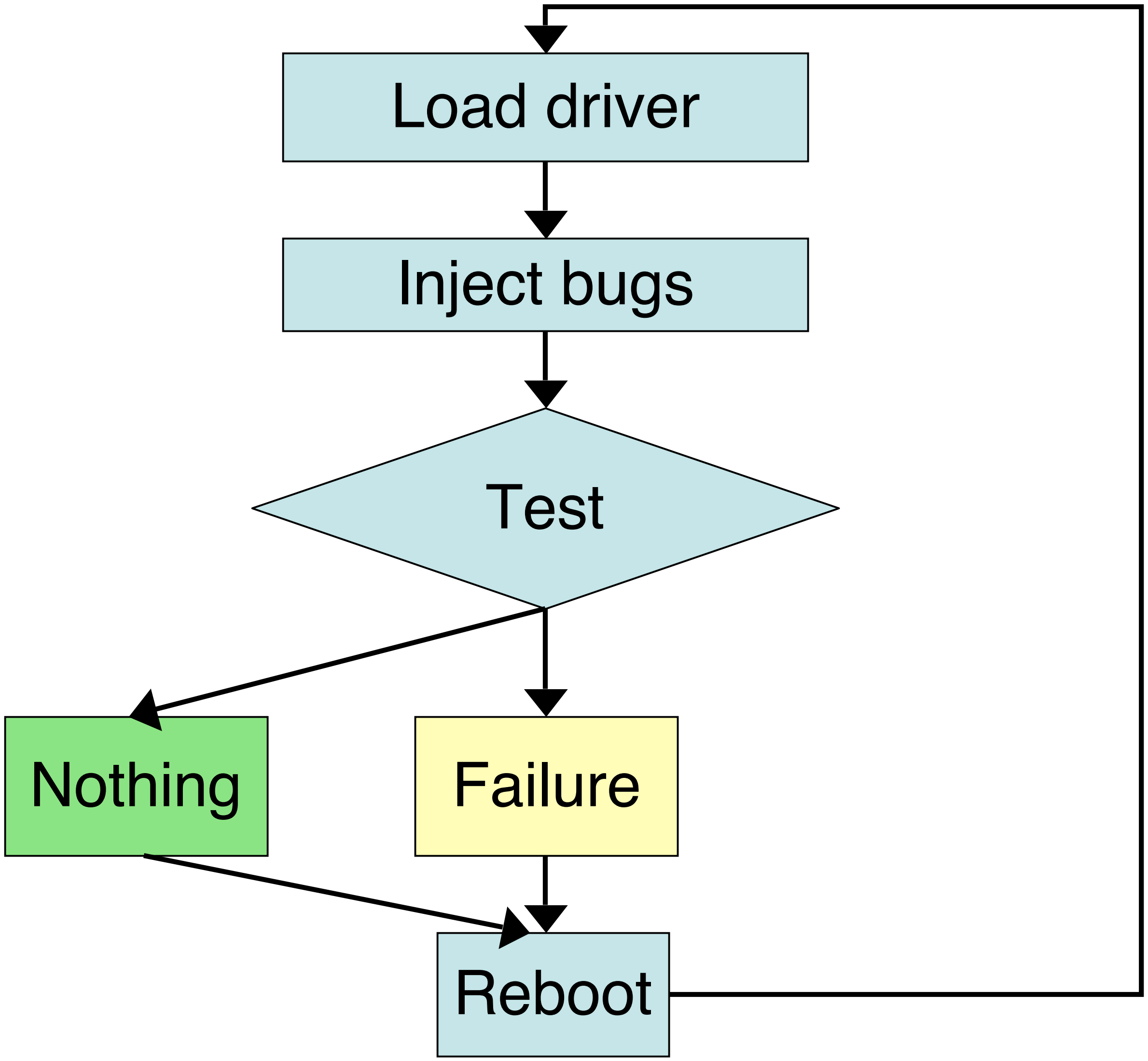
Outline

- Vision
- Design
- Evaluation
  - Reliability
  - Performance
  - Implementation Cost
- Summary

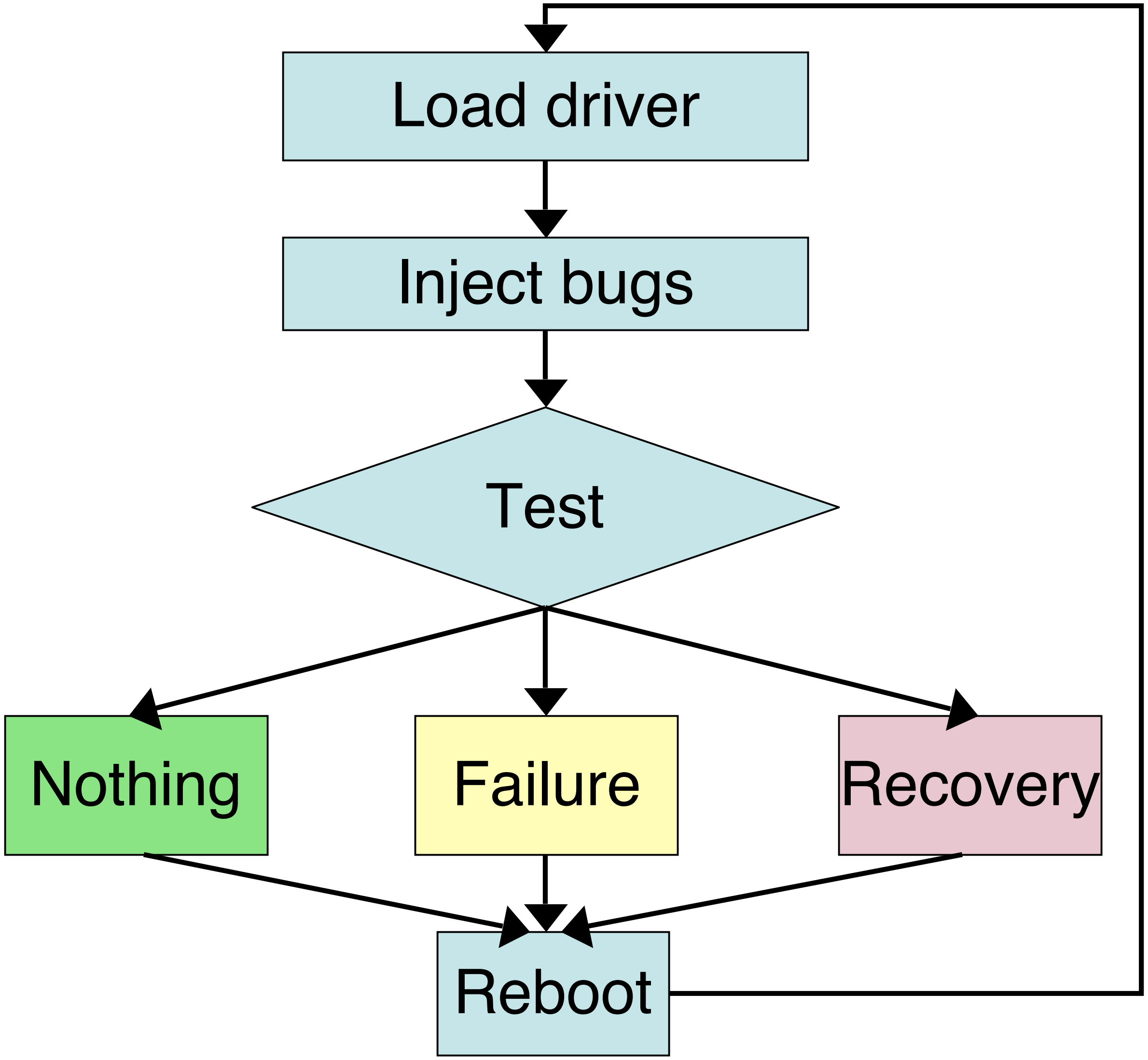
Tested Drivers

- Sound card drivers
  - SoundBlaster 16 (sb)
  - Ensoniq 1371
- Network drivers
  - Intel Pro/1000 Gigabit Ethernet (e1000)
  - AMD PCnet32 10/100 Mb Ethernet (pcnet32)
  - 3COM 3c90x 10/100 Mb Ethernet
  - 3Com 3c59x 10/100 Mb Ethernet
- Filesystems
  - VFAT Windows-compatible filesystem (vfat)
- Other
  - kHTTPd in-kernel web server (khttpd)

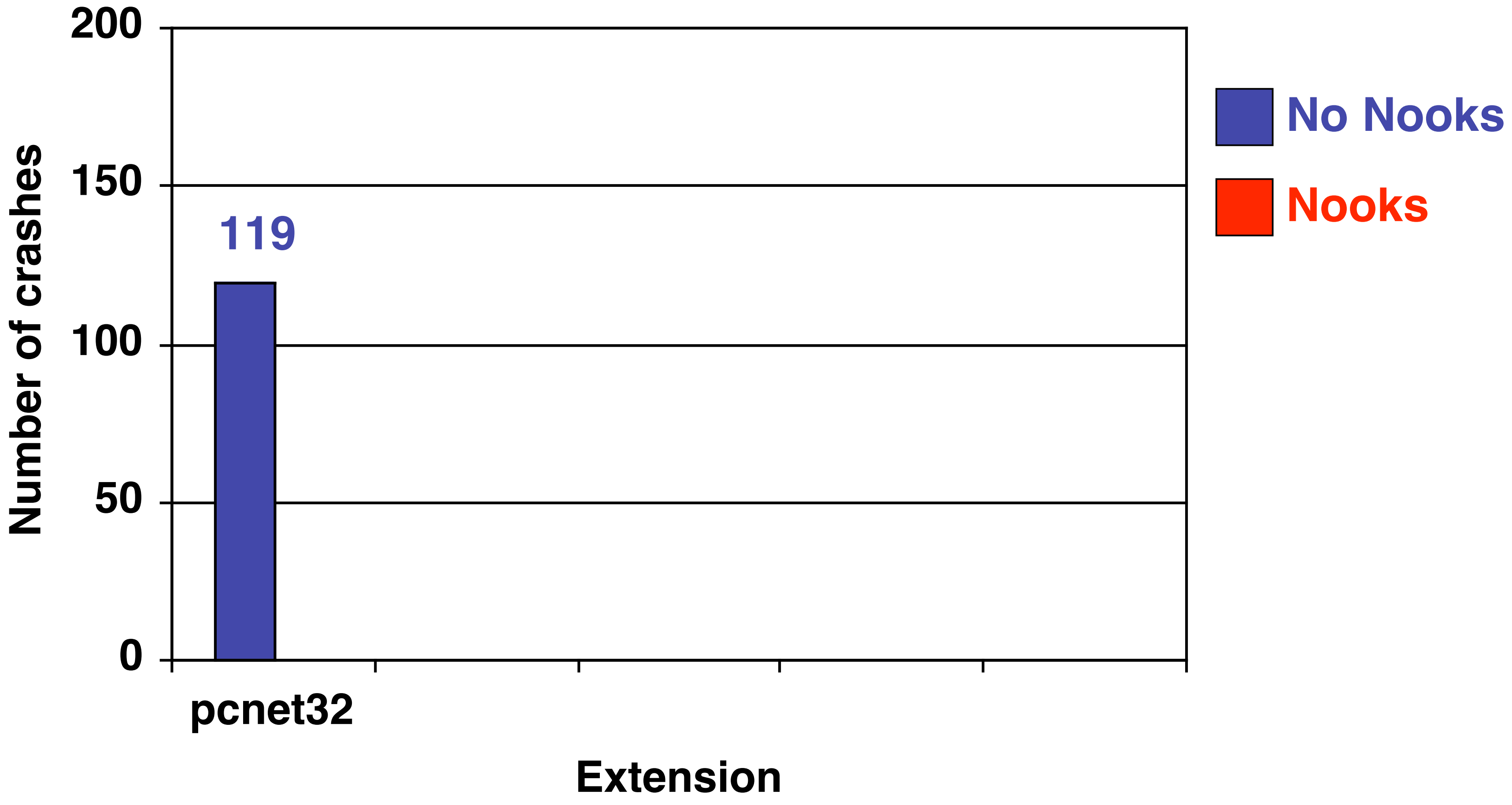
Reliability Test Methodology



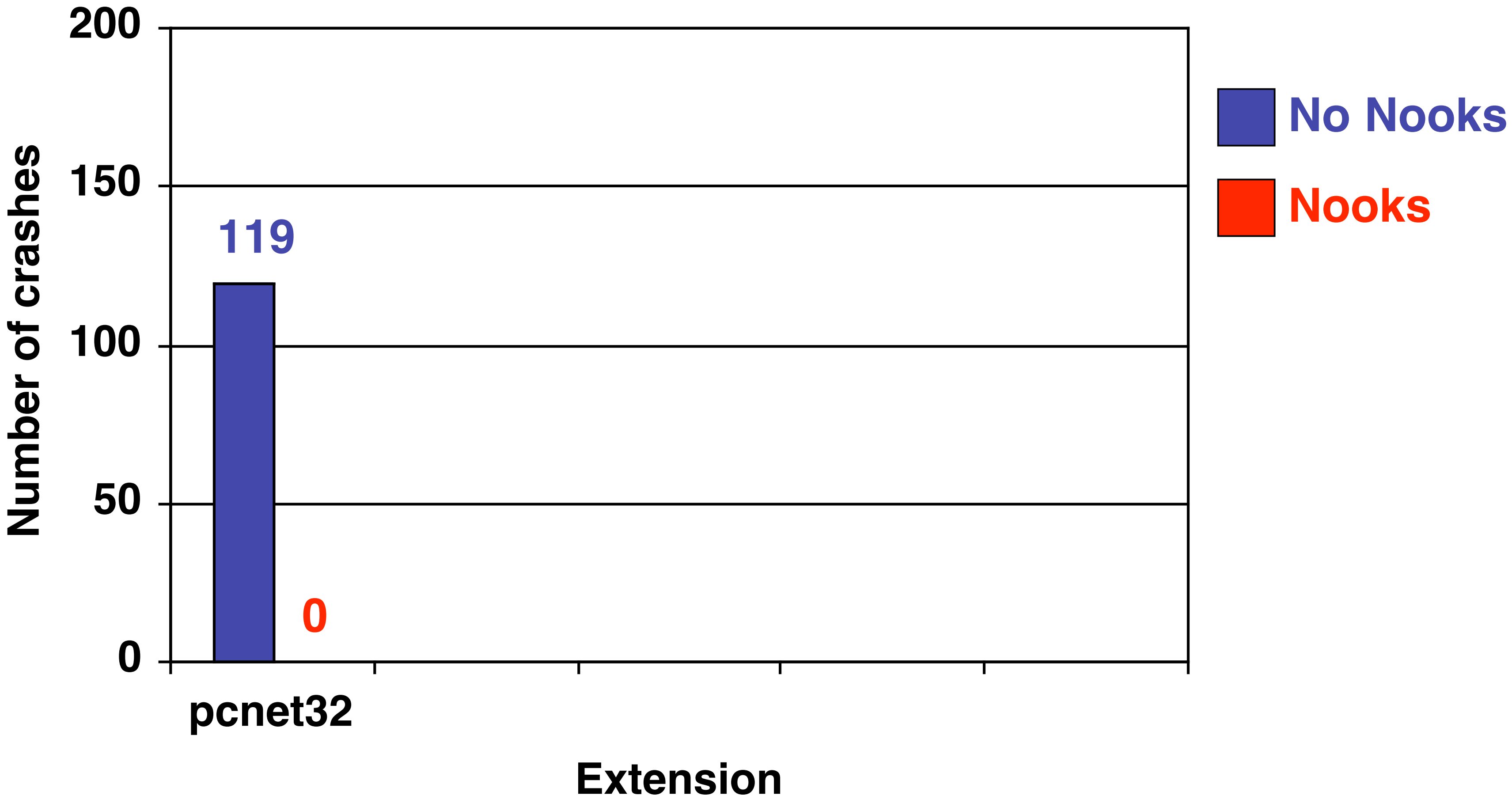
Reliability Test Methodology



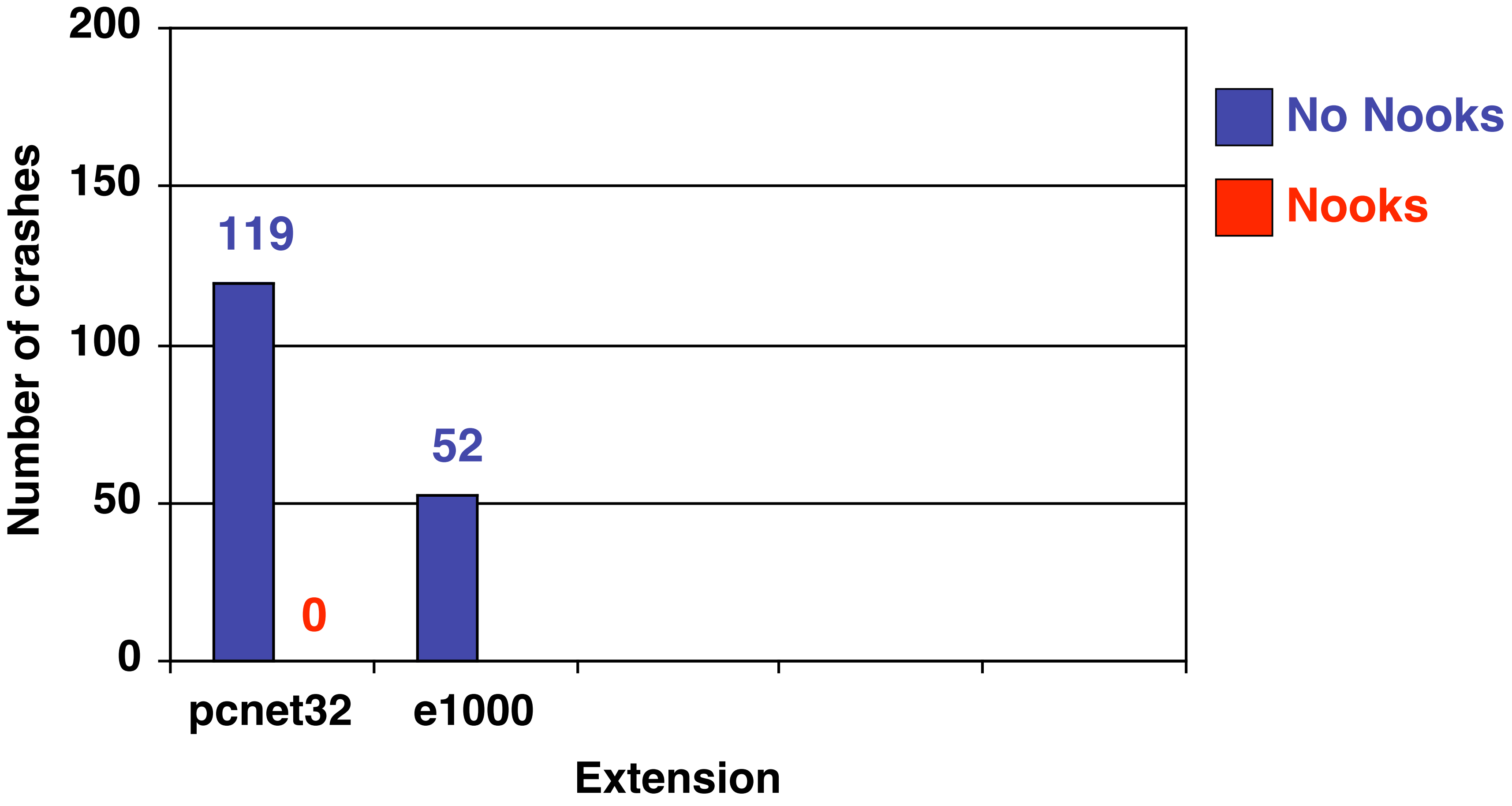
Nooks Stops Crashes



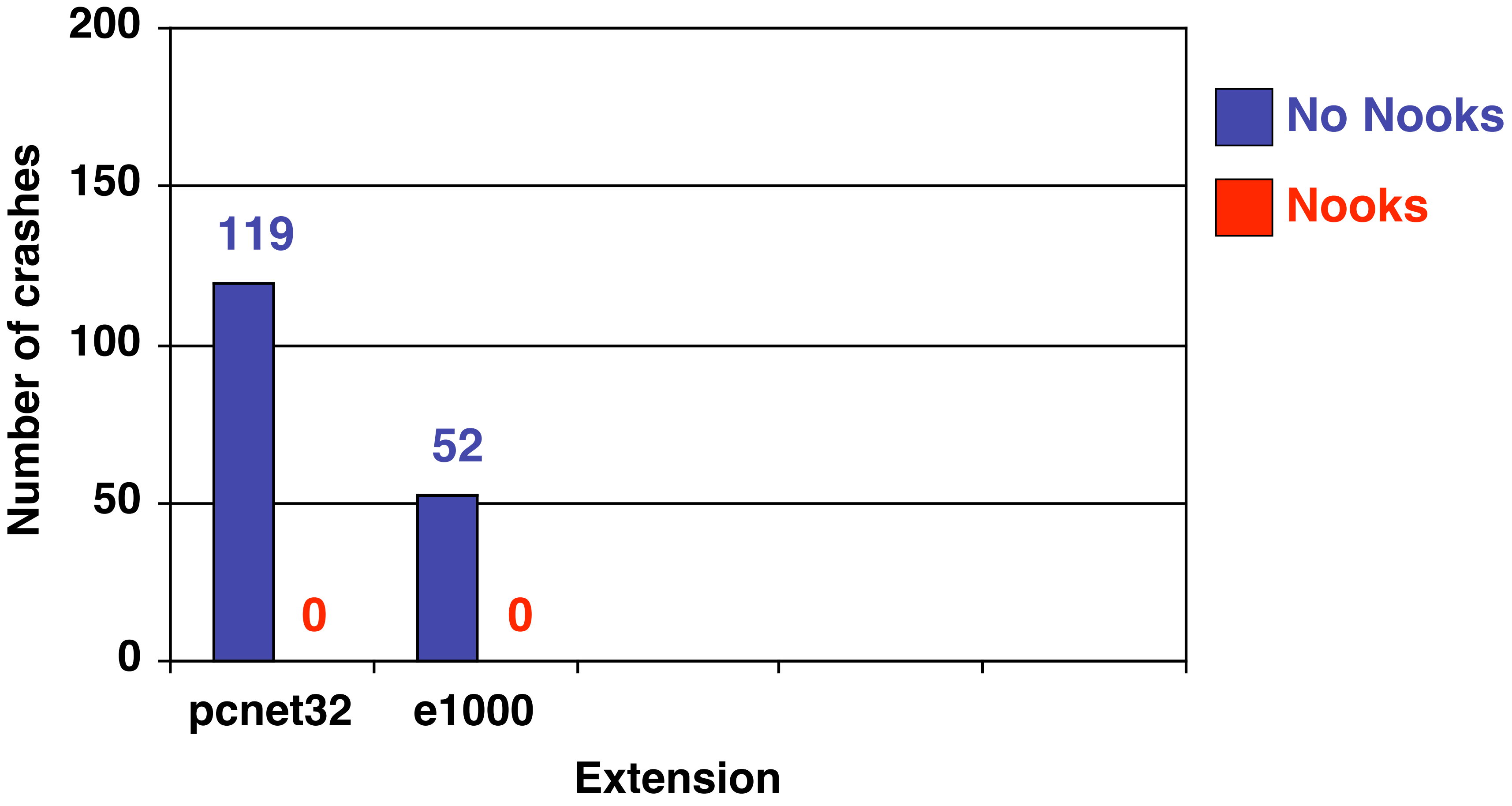
Nooks Stops Crashes



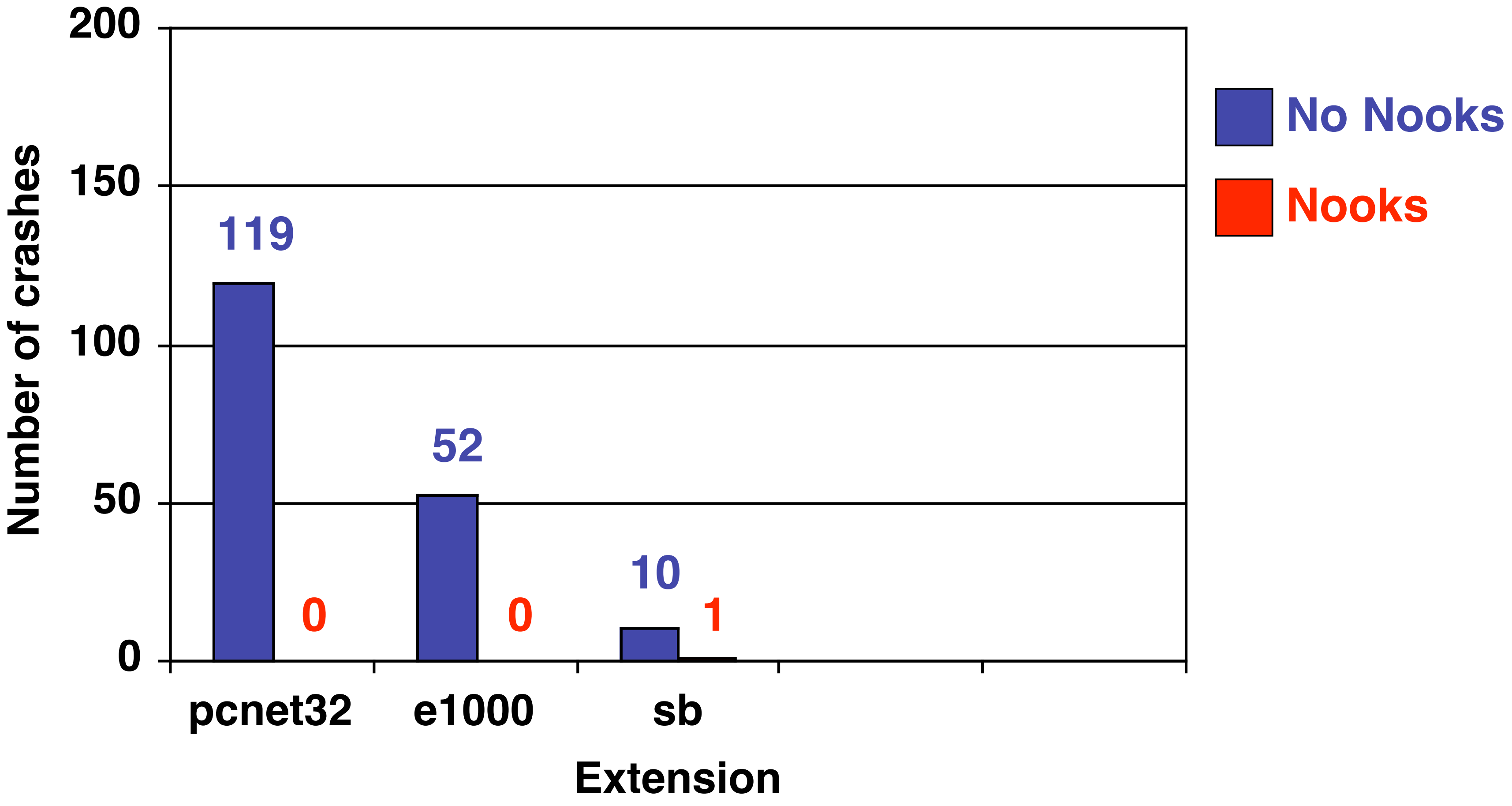
Nooks Stops Crashes



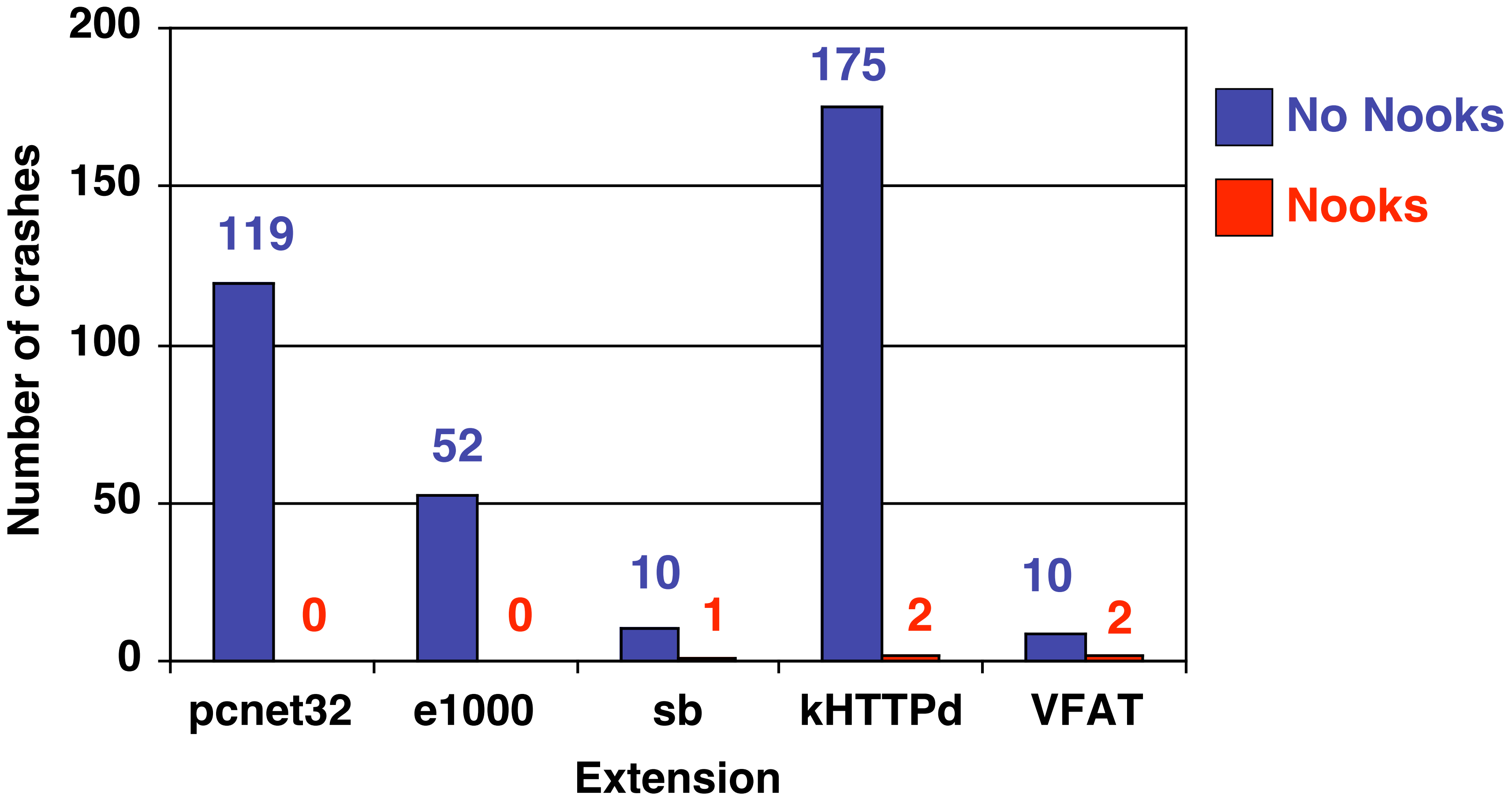
Nooks Stops Crashes



Nooks Stops Crashes



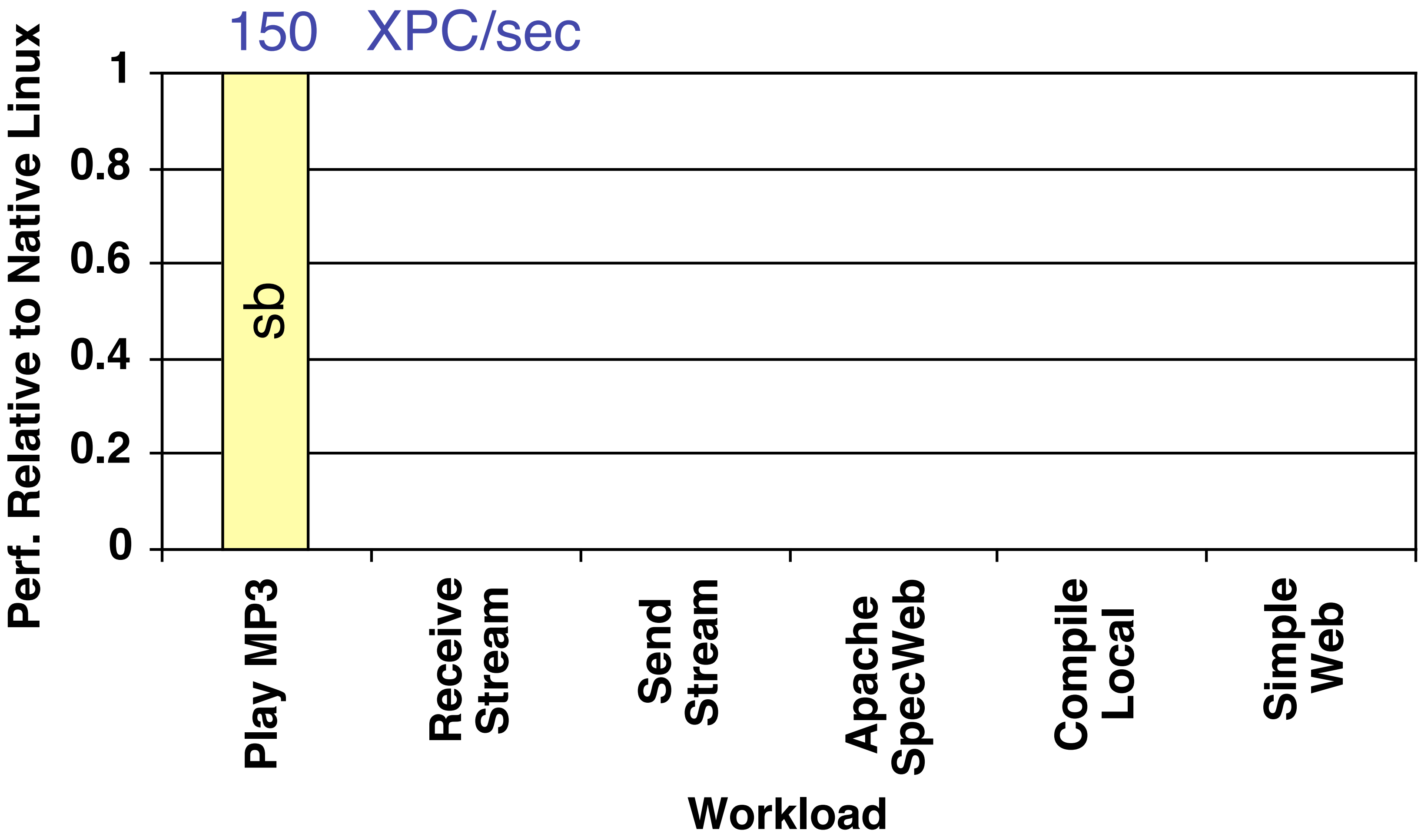
Nooks Stops Crashes



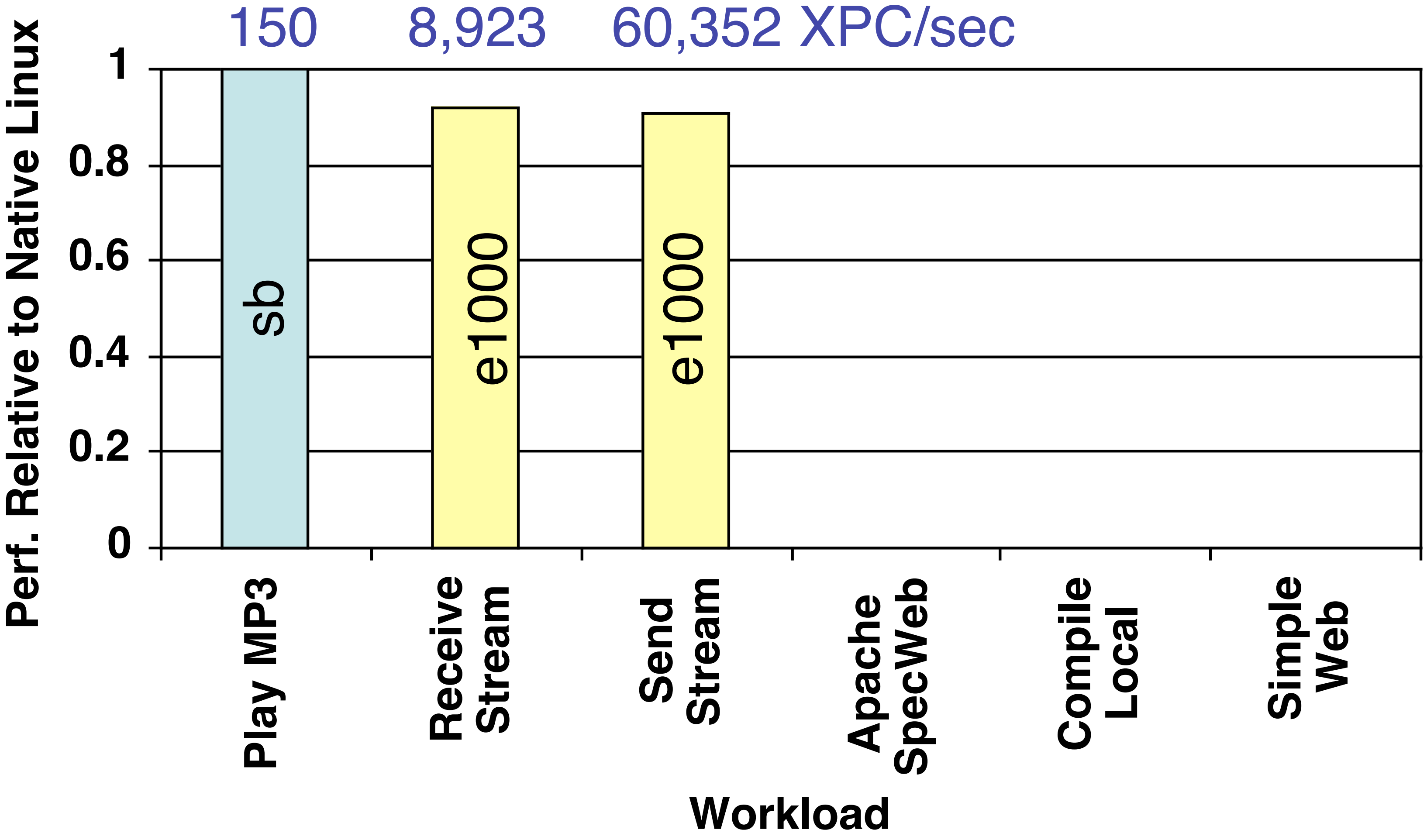
Performance

- Dominant cost is XPC
  - Performance depends frequency of interaction with kernel

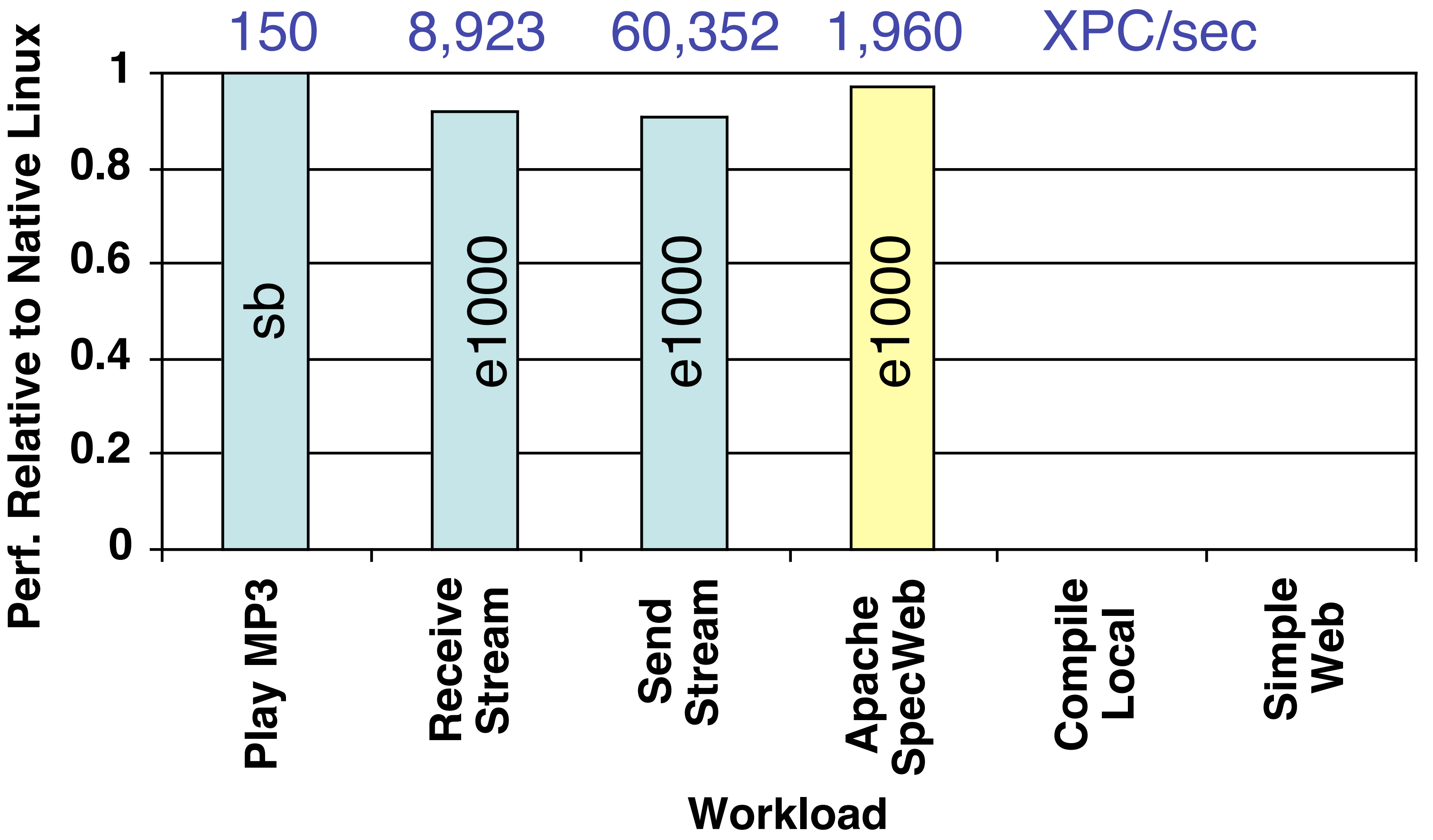
Relative Performance



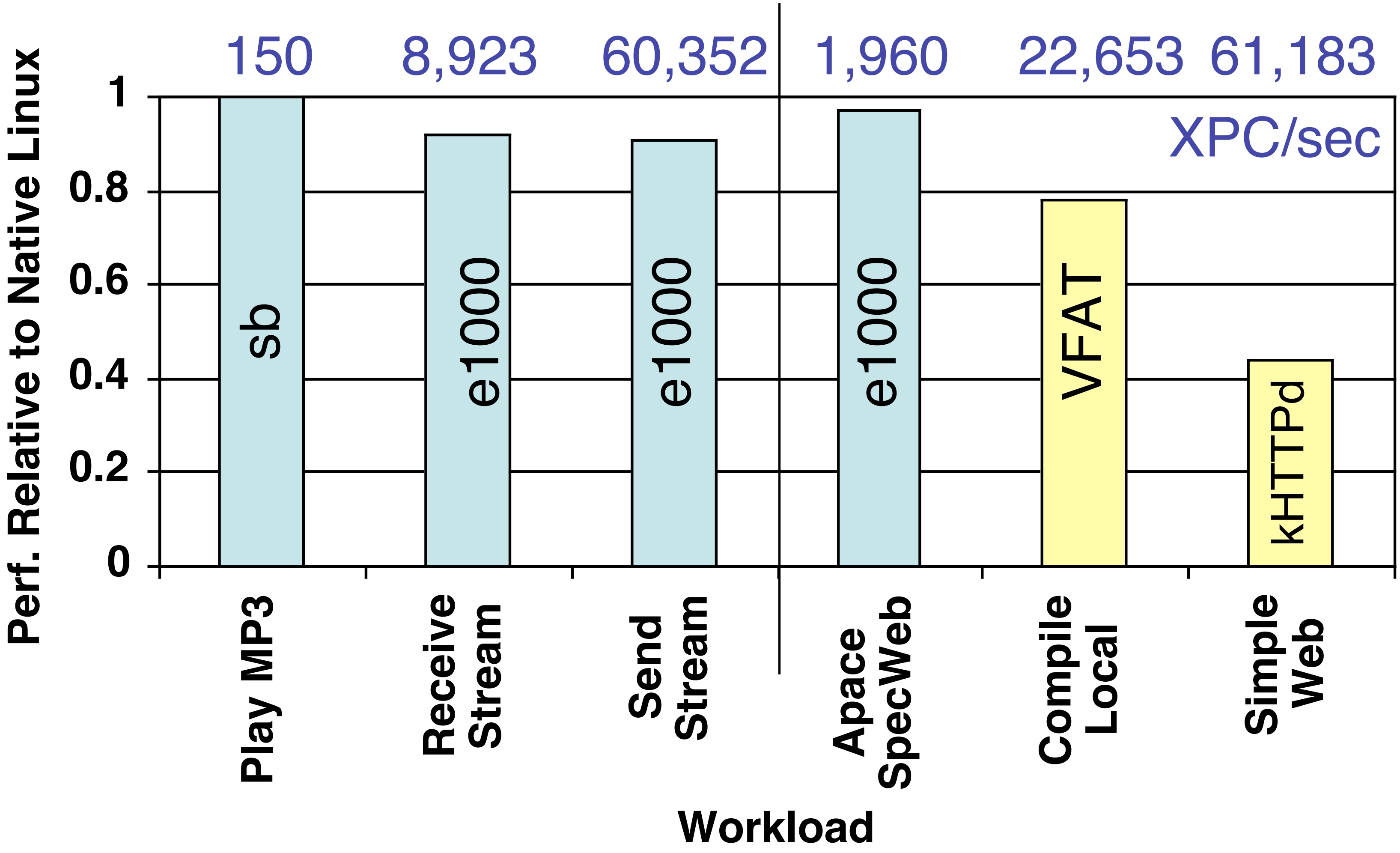
Relative Performance



Relative Performance



Relative Performance



Implementation Cost

- Changes to old code
  - Kernel: 924 out of 1.1 million lines
  - Device drivers+VFAT: 0 out of 33,000 lines
  - kHTTPd: 13 out of 2,000 lines
- New code
  - Nooks reliability layer: 22,266 lines

Summary

- Nooks provides a new reliability layer between drivers and the OS
- Nooks prevents 99% of tested faults that cause Linux to crash
- Nooks imposes a modest performance cost

Questions?

Thanks to

Doug Buxton, Steve Martin, Christophe Augier

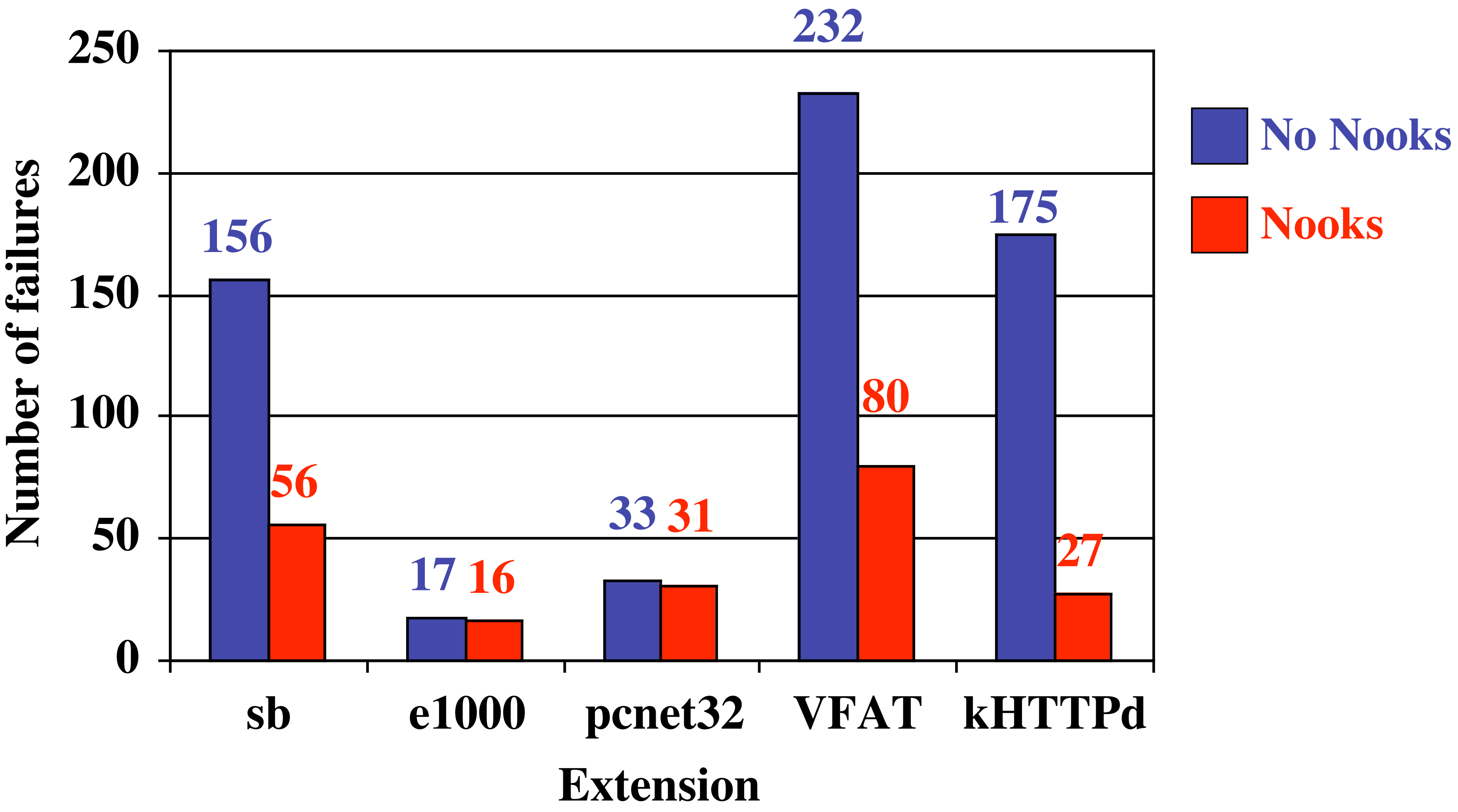
Microsoft

www.cs.washington.edu/homes/mikesw/nooks

Why didn’t we use a microkernel?

- Doesn’t address our limitations
  - Isolation not much better
  - Fault detection not much better
  - Recovery not much better
  - Doesn’t improve performance
- Requires more changes to the kernel
- Makes compatibility more difficult

Nooks Catches Bugs



Future work

- Improve performance
- Better recovery
- Automate wrapper generation