What's Happening Now

Linux 1.3.53:

Extremely fast context switch
(<50 µseconds, independent of number of processes)

Much better TCP performance (50% higher BW)

FreeBSD 2.1R

Ordered asynchronous file metadata writes (unverified)

Solaris 2.5

Faster context switching (unverified)
Faster networking

Conclusion

Performance (generally) doesn't matter!

Qualitative factors make the difference:

Linux, FreeBSD:	Freely distributable kernel source
Linux:	Vast user community
Solaris:	Support for multiprocessing

Results

Linux:	fast:	system call, small file performance
	slow:	networking
FreeBSD:	fast:	networking
	slow:	small file performance
Solaris:	fast: some other benchmark?	
	slow:	system call, context switching

Modified Andrew Benchmark

Across NFS to a SunOS server:

OS	Time (seconds)	NFS overhead	Normalized to best
FreeBSD	53.24	+12.20%	1.00
Linux	57.73	+33.20%	0.92
Solaris	58.38	+7.49%	0.91

Linux: Poor networking performance, untuned.

Modified Andrew Benchmark

On the local disk:

OS	Time (seconds)	Normalized to best
Linux	43.12	1.00
FreeBSD	47.45	0.91
Solaris	54.31	0.80

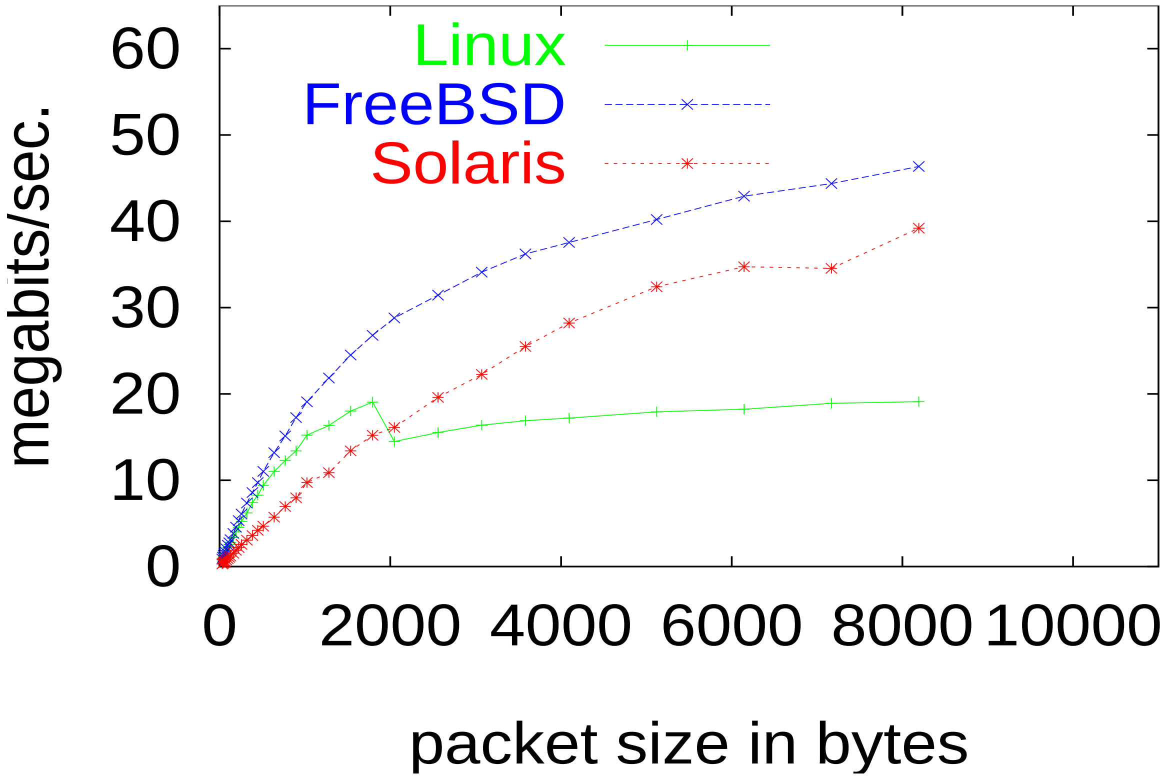
Linux: Uses asynchronous file metadata write.

Networking Performance (TCP)

OS	Bandwidth (megabits/ second)	Normalized to best
FreeBSD	65.95	1.00
Solaris	60.11	0.91
Linux	25.03	0.38

Linux: Does too many memory copies,
TCP window of 1 packet.

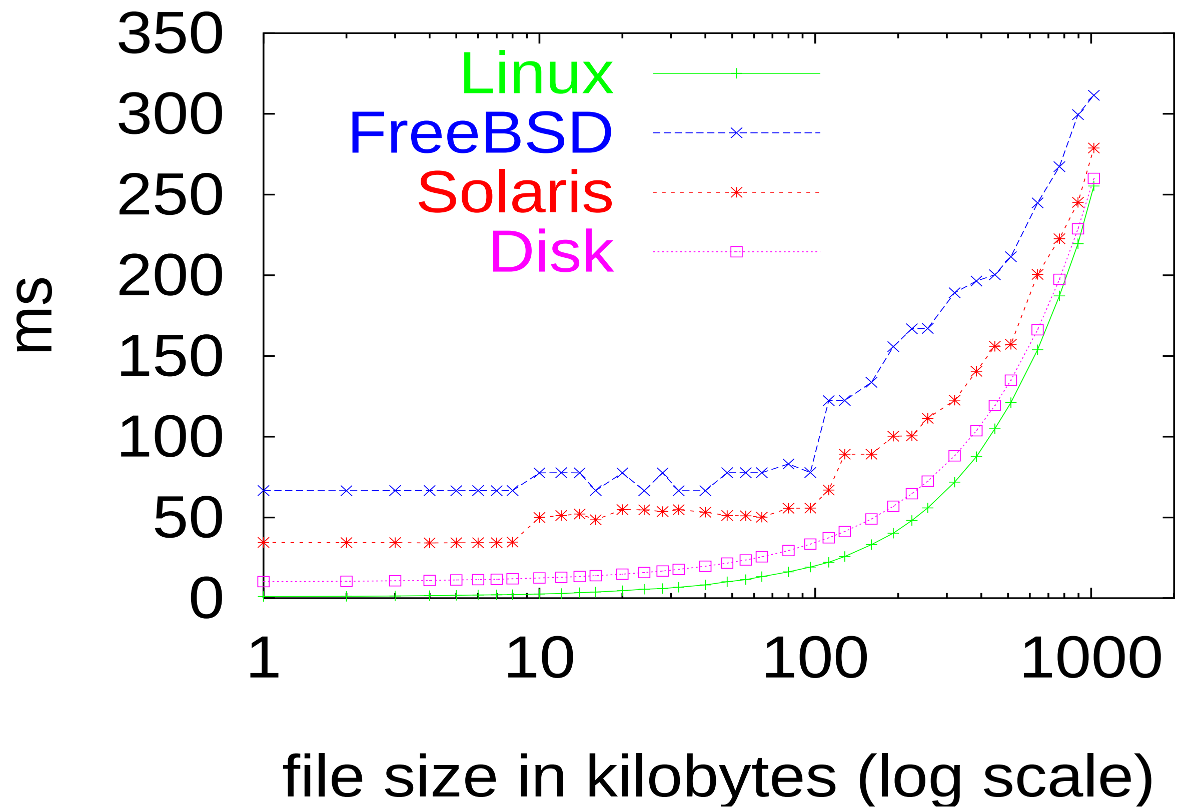
Networking Performance (UDP)



Linux: Does too many memory copies

Small File Performance

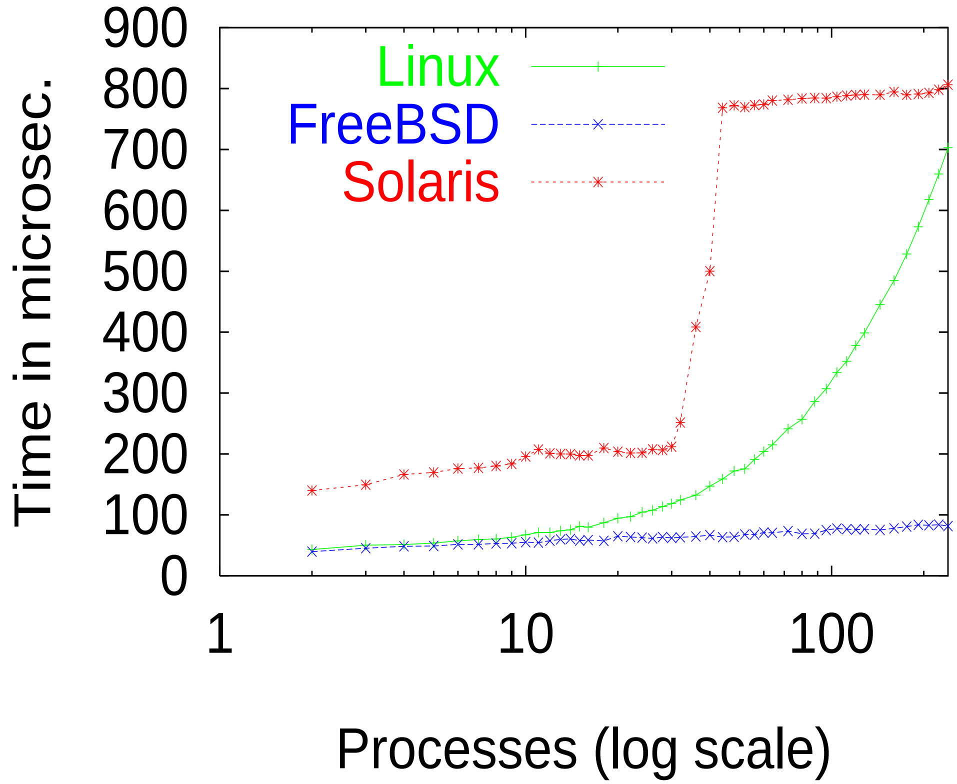
- Create file
- Write n bytes
- Read n bytes
- Delete file



Linux: Uses asynchronous writes of file metadata.

FreeBSD: Performance worse than Solaris on similar FS.

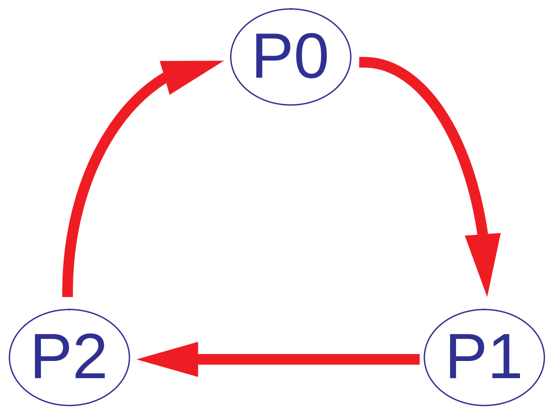
Context Switch (mostly)



Linux: Searching long linked list during context switch

Solaris: Multi-threaded kernel has complex scheduler

Context Switch Benchmark Design



What is measured:

- Pipe latency
- Scheduling
- Context switching

System Call

OS	Time (μseconds)	Normalized to best
Linux	2.31	1.00
FreeBSD	2.62	0.88
Solaris	3.52	0.66

Table 1. Results averaged over 1000 iterations of calling `getpid()` in a loop

Linux: Slightly more optimized kernel entry assembly

Solaris: Multi-threaded, fully preemptive kernel

Benchmarks

- Microbenchmarks
  System Call
  Context Switch
  File System
  Networking
- Application Benchmark
  Modified Andrew Benchmark

Methodology

Benchmarking vs. Other Techniques (e.g. kernel counters)

Advantages:

- Portable
- Most Important Metric = Wall Clock Time
- Comparable results

How we benchmarked:

- Black box
- No optimizations

Operating Systems Tested

Non-development release version of the OS in June 1995:
(Bug fixes until October 1995)

OS	**Version**
Linux	1.2.8
FreeBSD	2.0.5R
Solaris x86	2.4

1995 Platform

How we went shopping for an OS:

- Runs on our hardware:

  100 MHZ Pentium, 32MB, NCR 53c810 SCSI controller, 2GB internal disk, 2GB external disk, 17” monitor 10Mb/sec. Ethernet

- Easily installable
- Easily available

Which Brand?

- Homemade OS
- Commercial UNIX
- Free UNIX

    A Toy?

Desirable *Research* Operating System Features:

performance,
reliability,
kernel source code,
technical support,
driver support,
application software,
large user base

Why PC?

Mainframe → Minicomputer → Workstation → PC

Market trend:

Increase in PC hardware performance/price ratio

100 MHZ Pentium, 32MB, 2GB disk, 17” monitor
10Mb/sec. Ethernet

~ 100 SpecInt92/$5000 (June 1995)

A Performance Comparison of UNIX Operating Systems on the Pentium

Kevin Lai, Mary Baker
{laik,mgbaker}@plastique.stanford.edu
http://mosquitonet.stanford.edu

Department of Computer Science
Stanford University