Gigabit File Servers: NFS as a Plausibility Study

David A. Eckhardt
Nectar Project
School of Computer Science
Carnegie Mellon University

David.Eckhardt@cs.cmu.edu

Outline

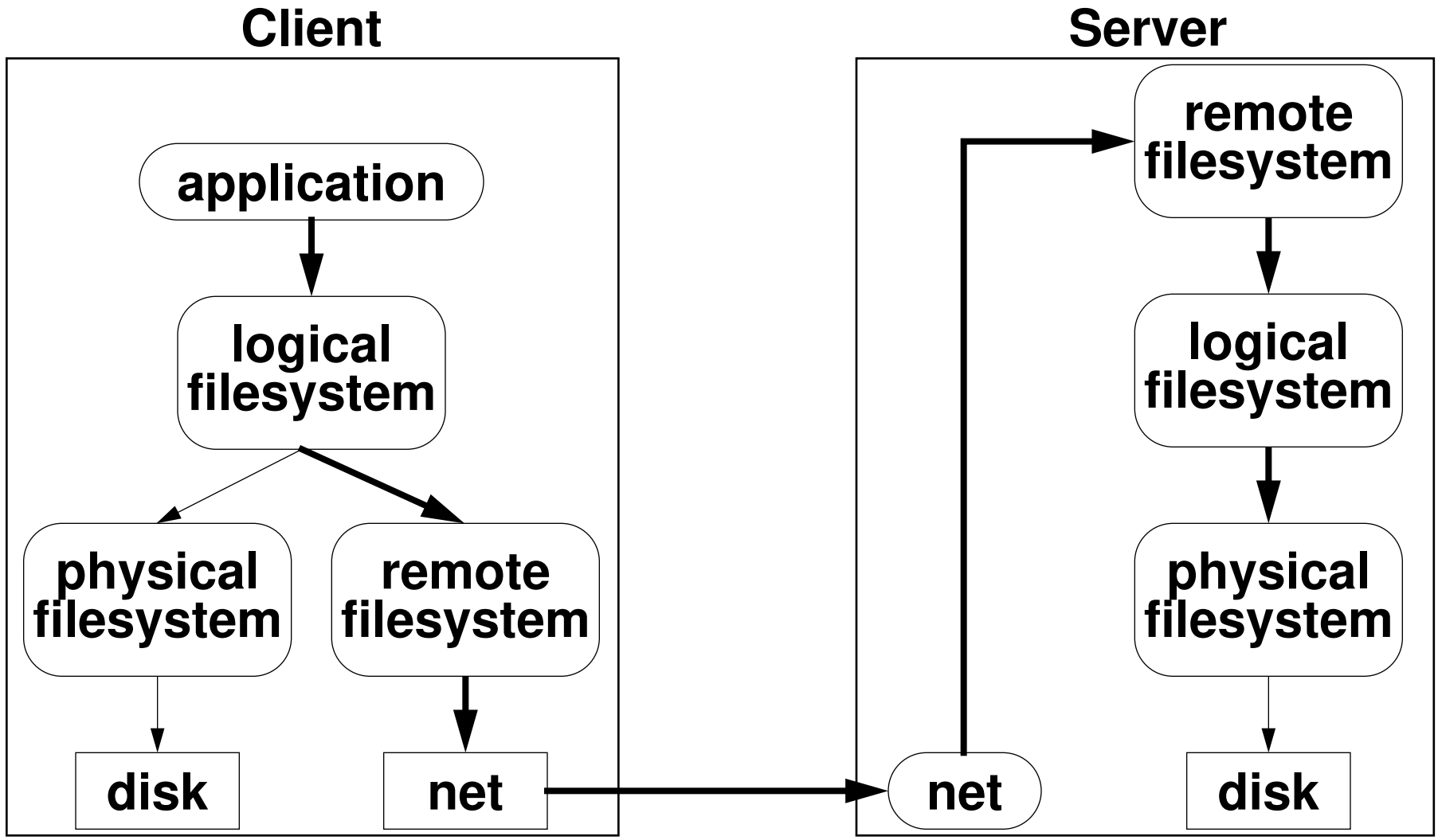
Remote Filesystem Overview

Approach

Results

Conclusion

Remote Filesystem Overview



Approach

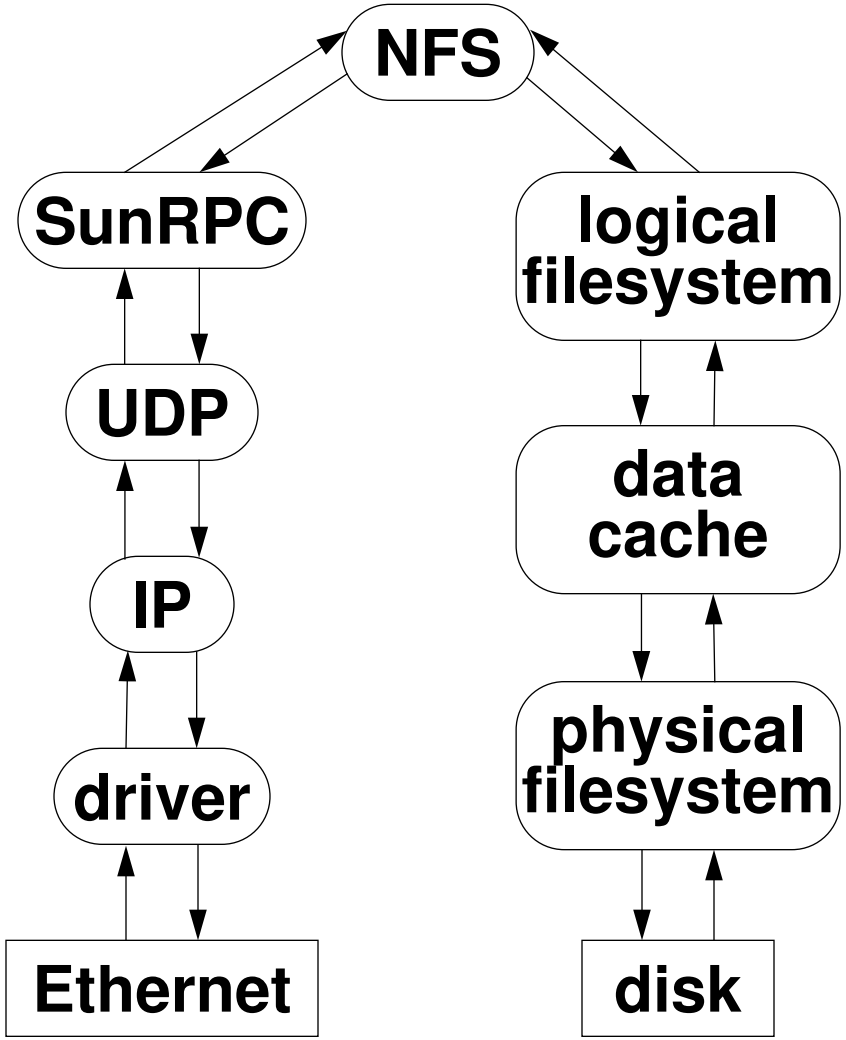
Measure time server spends handling a read request

- Assume read satisfied by buffer cache
- Exposes performance of network (not disk)
- Currently a common case
- Should be more common
  - Large server caches, aggressive prefetching

Results still applicable for writing data

- Writes currently “artificially” slow
- Strategies being pursued:
  - Delayed writeback
  - Non-volatile memory
  - Disk arrays
  - Log-structured filesystems

4.3 BSD NFS Server



NFS Time Expenditures

50Mhz HP9000/720, common case

file handle to vnode pointer	11μs
get vnode attributes	3μs
check access	1μs
read-and-pin buffer	32μs
translate attributes	3μs
release buffer	10μs
release vnode	33μs
	93μs

Fast and Slow Paths

Common case: 93μs

Slow case: 387μs

Differences:

- **Needless data copy (245μs)**
- **Testing for rare conditions**

Gigabit Budget

Time to provide 8K packet

Actual **93μs**

Time to transmit 8K packet

800 megabit (HiPPI) **80μs**

622 megabit (OC-12 SONET) **100μs**

We already meet our goal

Scalability

Application scales well as machines improve

- Compared times on SparcStation 1+, HP9000/720
- Most subtasks run four times faster

Faster machines already available

Conclusions

Application not the bottleneck

Still many opportunities for optimization

Real culprit: Extra copies

If network bandwidth = memory bandwidth, every copy is catastrophic

Summary of Measurements

	SS1+	50Mhz Snake (Slow)	(Fast)
Receive packet (UDP, RPC)	776μs	199μs	197μs
File handle to vnode pointer	40μs	12μs	11μs
Get vnode attributes	15μs	4μs	3μs
Check access (same user)	4μs	1μs	1μs
Read-and-lock buffer			32μs
Translate attributes			3μs
Allocate buffer	1800μs	15μs	
Generic read	210μs	75μs	
Copy (B=#bytes)	.13Bμs	.03Bμs	
Translate attributes	20μs	2μs	
unlock buffer+release vnode	0+6μs	0+33μs	10+33μs
wait + kfree + MFREE	0+3+0μs	0+1+0μs	6+0+3μs