File System Tracing

The xFS Crew:
Tom Anderson, Mike Dahlin, Jeanna Neefe,
Drew Roselli, Randy Wang

Abstract

Collect long-term, comprehensive traces of file system activity for use in xFS design simulation.

Goals

- Update Ouster85, Baker91
- Unique Contributions
  long-term, metadata, pathnames, idle periods

- Re-usable tracing tool
  portable, collect other workloads
  low overhead

Status

- Tracing on NOW machines for 1 month

Outline

Method

- Auditing system

Some studies these traces enable

- Disk layout
- Distributed caching
- Idle periods
- Metadata

Summary

Methodology

Auditing subsystem

- Hp-ux 9.05, Solaris 2.4/2.5
- Catch syscalls on client, log on local disk

Problems

- Hp-ux: old source code, no command source
- Kernel changes: add calls, full pathnames
  Getting pathname in kernel daunting
  Solution: snoop *cd*'s
  start auditing earlier in init
  - cannot restart auditing
- Does not catch reads/writes to *mmap*'d files
- Client/server doubly catch calls

Logistics

Disk space

- ~2MB per node/day

  400 nodes, 6 months -> 144 GB

Slow-down

Solaris: 5%

Hp-ux, original: 11%

buffered output: 1%

Portability

- Redo with SLIC?

Disk Layout Study

Long-term

- Can see beyond cache
- "Warm" the disk

  [Seltzer 94] fragmentation changes over time

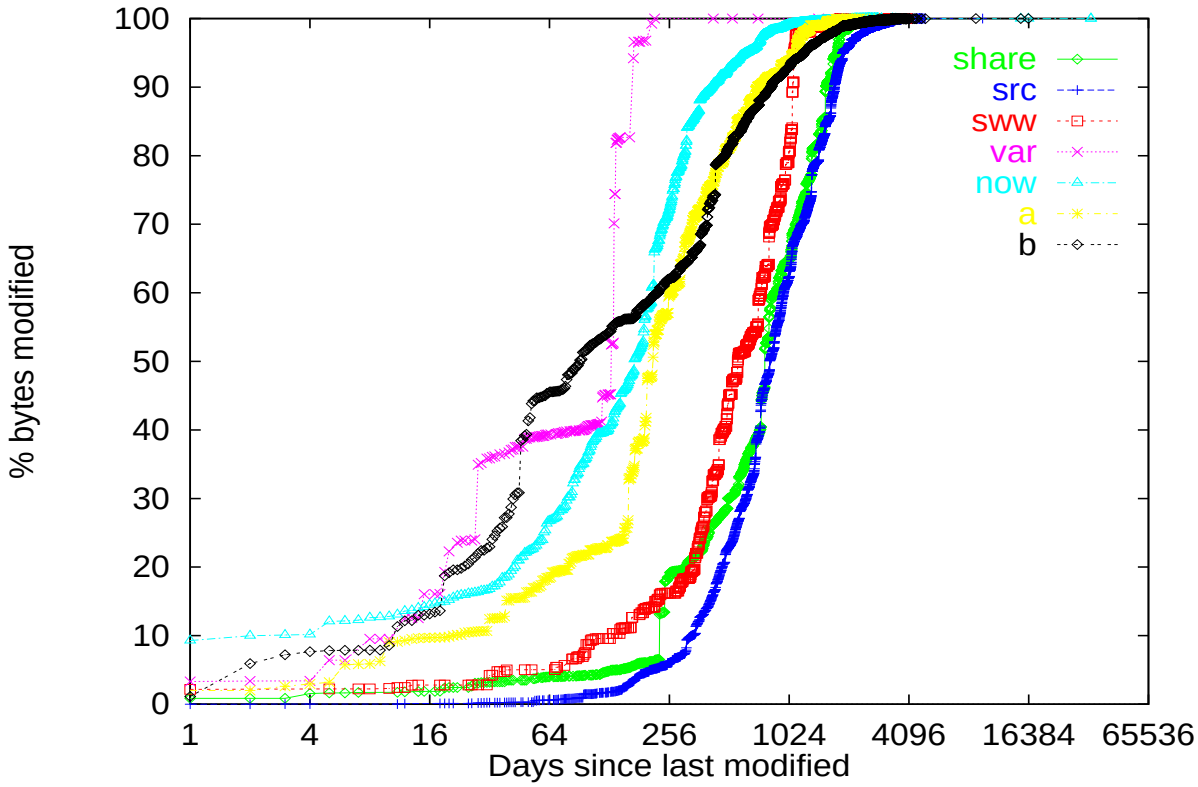
Cache hits

- Dynamic read re-organization
- Cleaning from cache

Pathnames

- Pre-fetching, re-organizing policies
- Compare with update-in-place file systems

Most data on disk is old

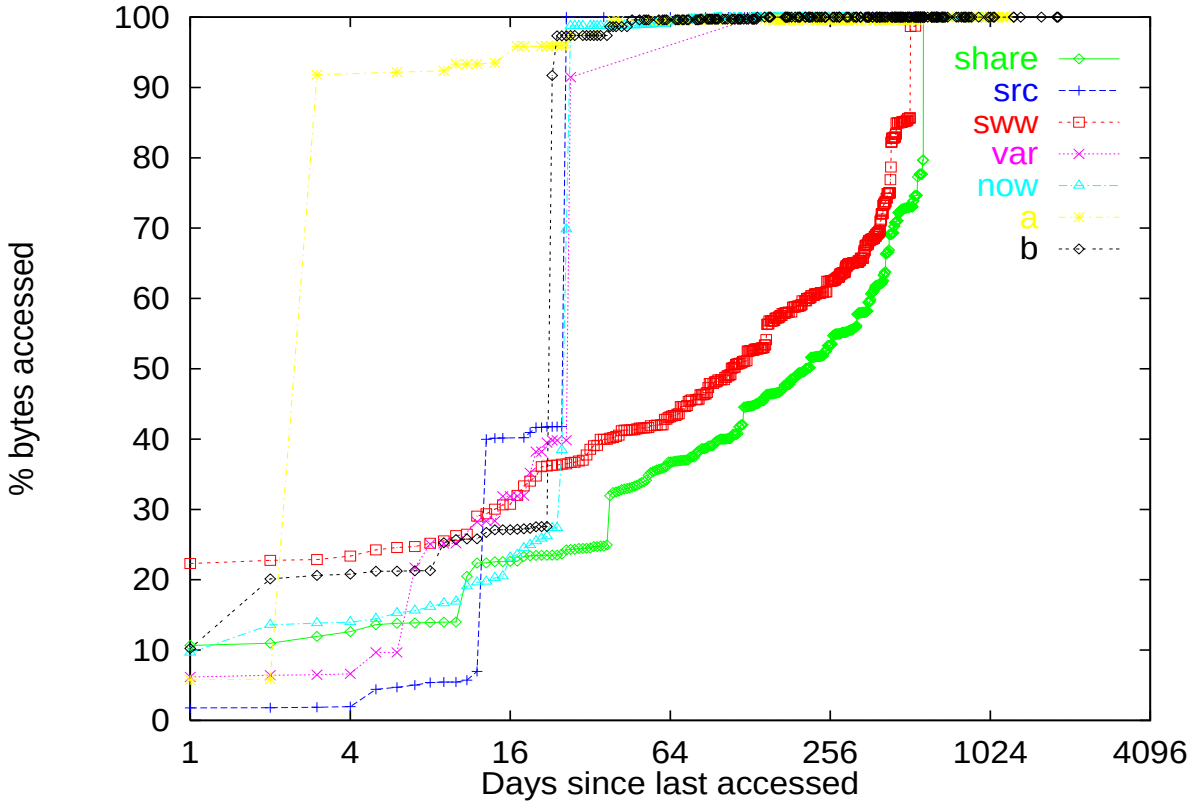


Within 1 month

Share (sww/src/share/var): 0 - 25% written

Home (now/a/b): 10 - 20% written

Most data has been read recently

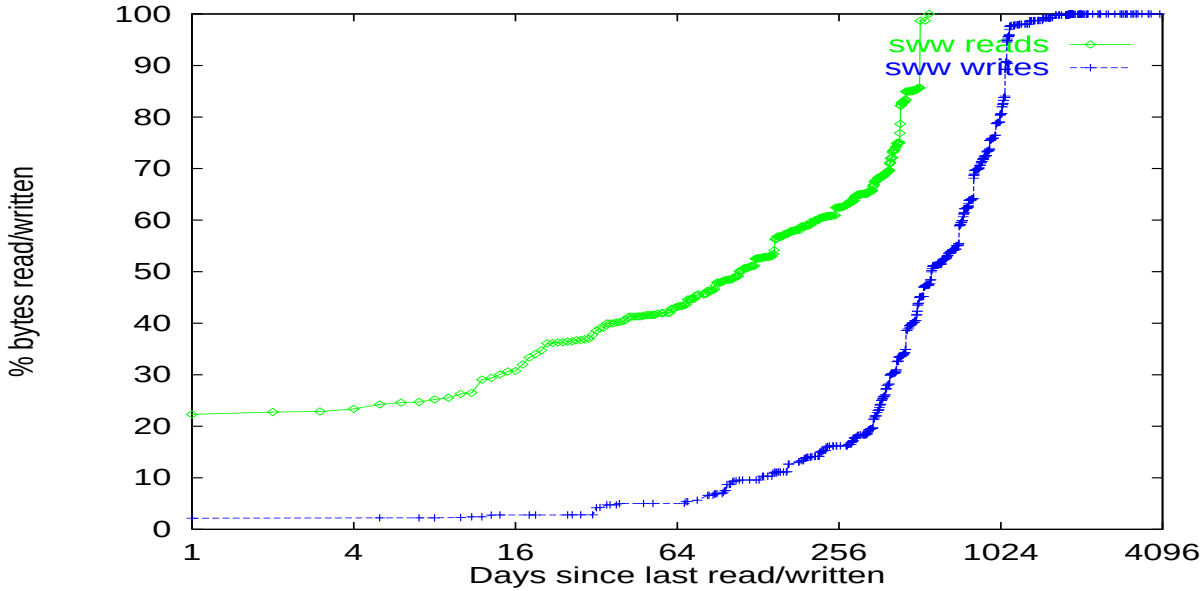


Within 1 month

Share (sww/src/share/var): 25 - 40% read

Home (now/a/b): 100% read

Update/access Gap



In 2 weeks	sww/src/share	now/a/b
modified	< 5%	10 - 15%
read only	25 - 35%	5 - 85%
untouched	60 - 80%	5 - 85%

Re-organize:

[Blaze 92] more read -> less written

Possible disk layout strategy

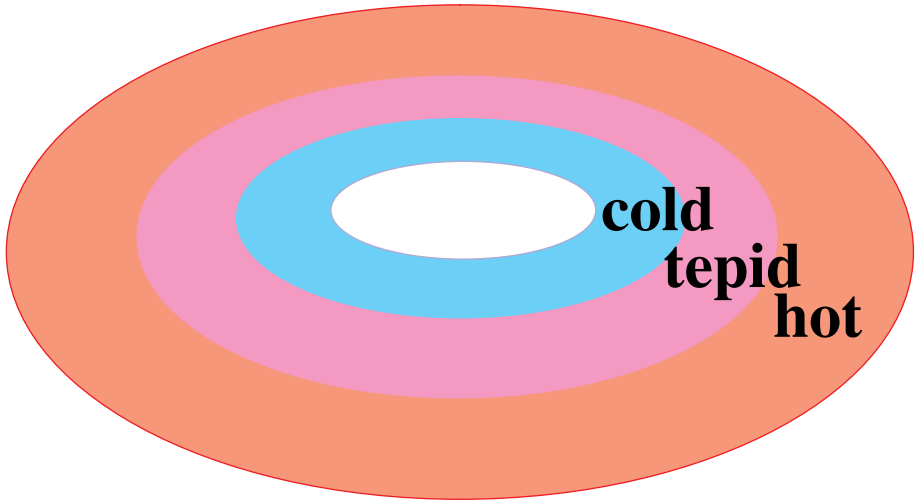
Separate regions on disk

- *Hot* - frequently accessed
- *Tepid* - less frequently accessed
- *Cold* - rarely accessed

How to split the disk into regions?

- Hot data: optimize for bandwidth or latency
- Migrate cold data to “corner” of disk

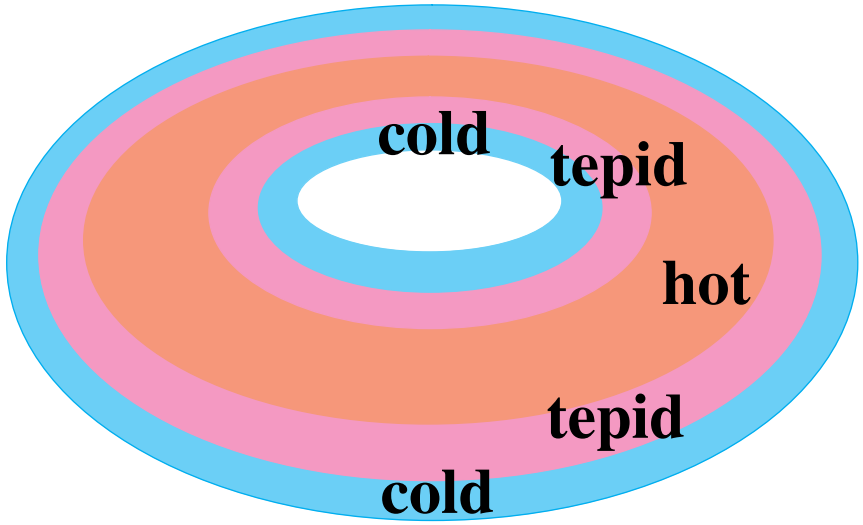
Write-dominated workload



Optimize for bandwidth

- *Hot* - written, not read (in cache)
- *Tepid* - not written, read infrequently
- *Cold* - unaccessed data

Read-dominated workload



Minimize latency

- *Hot* - frequently read
- *Tepid* - less frequently read
- *Cold* - unaccessed data

Test out with traces.

Distributed caching study

Scalability

- [Baker91] lower hit rate than expected

Long-term

- Short traces exaggerate compulsory misses

Cache hits

- Network/server-based traces only catch misses

Large-scale

- Efficacy of cooperative caching on large workload (singlet forwarding vs. global LRU [Feely 95])

Lots of caching policies; which is best?

Idle time study

Exact read/write times -> idle time

- Clean/ prefetching/ re-organizing policies
  caveat: does not include VM demands

Long-term, large-scale

- [Seltzer 95] short-term study
- Daily/ weekly activity cycle

Lots of plans for idle time.
Do they concur with reality?

Metadata study

Catch all file system calls

- Caching policy for metadata
  [Wilkes 93] 67-78% disk writes are metadata

- Where to store metadata
  directory entry, inode, separate file
  [Oust critique Selt 93] access time in inode map

Systematic approach to metadata issues.

Milestones

Develop

Tracing tool	(4/96)	☑
Decrease overhead	(6/96)	☑

Install

NOW cluster	(3/96)	☑
NOW desktop	(5/96)	☑
All instructional machines	(7/96)	
The world	(12/96)	

Summary

Traces are an invaluable resource
vs. benchmarks, synthetic workloads

- Long-term
- Realistic

Previous traces are too short, limited, old

Lots of grist!