Greetings!
From A File System User

Jim Gray
Microsoft Research
4th USENIX Conference on
File and Storage Technologies (FAST 2005)
12/14/2005, San Francisco, CA

My Goal: to Confuse You.

- Architectural Breaking points:
  Infinite Capacity Disks
  - Overhead IOs?
- File System Guru Gap
  - Managing a petabyte (Harvey Newman story)
- Schema?
- To blob or not to blob
- Smart disks
- MTBF (copying a petabyte)

Architectural Breaking Points

- Processors are infinitely fast

  but always waiting for memory: CPI_∞
- RAM is infinite: 8M pages in 64GB
  5-minute rule is now 30 minute rule!
- Processors need 100 disks/core
  400 disks/cpu chip (!)
  Forces SAN interconnect ($$$!)
- Disks are infinitely large
  – just can't fill them
  - can't backup/restore them.

Disk Is Infinite Capacity
So, what?

- Disk heading for 10TB/drive
  - 1.3 days to read/write sequential
  - 5 months to read/write random
  - 10 billion 1kb files (a lot of inodes)
- How do utilities work?
  - Backup/restore
  - Check-checksum
  - Content index
  - Reorganize

Infinite Disk Capacity Consequences

- Tape is expensive storage:
  Use “extra disk space” creatively
  - Copy-on-write snapshots
  - Many versions
  - Cold storage (write protected).
  - No “backup/restore” just use another copy
- JBOD triplicate
  or 4-plex geo-plex
- RAID5 expensive
  - Extra IOs (disk accesses (not space) are precious)
  - Must geo-plex anyway

Infinite Disk Capacity Consequences

- Current approach to Format/Check/Reorg/… takes forever and uses precious accesses
- Aggregate Housekeeping IOs: one pass to
  - check checksums
    - among the triplex
    - Or on each disk (needs hardware support or…)
  - Reorganize the data
  - Make a snapshot/backup copy?
- Or piggyback on “nearby” activity
- Use change log to drive content indexing

Infinite Disk Capacity Consequences

- Optimize disk accesses
  Revisit Log Structured File system
- Optimize for management
  - Auto placement
  - Auto repair
- Do it all automatically

Infinite Disk Capacity Consequences

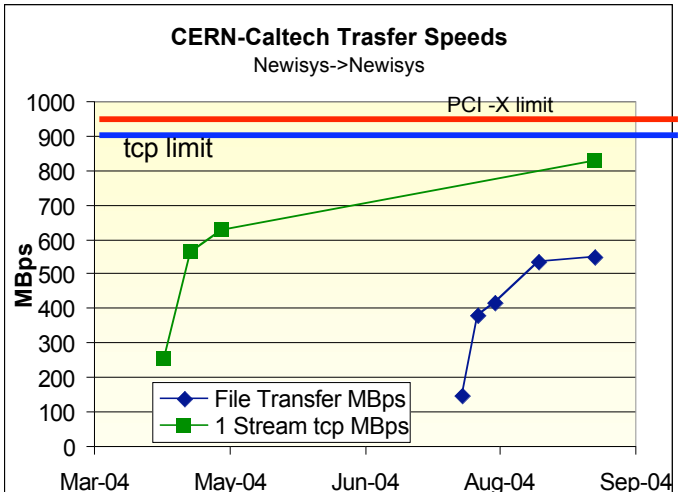
- Need SAFE disk storage.
- Design software and hardware for disk ARCHIVE / INTERCHANGE
- Should be faster/more automatic than tape
- Disk modules that can be pulled (probably a NAS)
- Fence part of a “online” drive so virus can’t hurt it.

The Guru Gap

- Caltech group could move 5GB/s (40 Gbps) from Phoenix via internet to Chicago..
- But 10MBps from disk to disk.
  - File layouts
  - File create
  - Multiple data copies
  - Window sizes
  - Array mistakes
  - Synchronous file IO

CALTECH

CERN



Harvey Newman, Yang Xai Caltech
Peter Kukol, Ahmed Talat, Inder Sethi,
Bruce Worthington: Microsoft
Brent Kelley: AMD
Rich Oehler, John Jean, Dave Radditz: NewIsys

OK, so data is Arriving at 1GB/s Now what?

- That's ~ 30 PB/year
- That's 10,000 10TB disks (tri-plexed data)
- How do we find anything (just file names?)
  - How do you manage
    30 million files of 1GB each?
- Want a "big" database to point at the "things in the files"
- Google File System seems like the right idea, but does it generalize and what is the "database"?

File Systems and DB Systems

- DB systems are SCHEMA FIRST
  - Must define metadata then data
- File systems are SCHEMA NEVER
  - No support for anything but byte stream.
- Well, not quite:
  - Remember VSAM, RMS, Netware,..
  - Sleepycat?
- Huge battles raging over XML (jit schema)
- Huge battles raging over object stores.

Hierarchies Are Not Helping

- Filers & Pilers:
  - Even filers are having problems organizing their trees
  - Wiki, Sharepoint, Flicker, email,…are piles: or many hierarchies.
- Directories are queries
- File names are not the main dimension
- Often (photos, email,…) text is not main dimension (date, from/to, subject…) are
- Needs schema.

Why Not a FS/DB Détente?

- Let a file look like a DB table / XML doc
- Let a table/xml doc look like a file.
- Support both interfaces
  - With security
  - With transactional semantics (if desired)
- Extract schemas from objects
- Take a JIT Schema approach
- Build lots of indices
- Interesting idea, see WinFS (and others)

To Blob or Not To Blob

- For N less than 1MB

```
Select x from T where key = 123
```

  Faster than

```
h = open(T); read(h,x,n); close(h)
```

- Because DB is CISC and FS is RISC
- Most things are less than 1MB
- DB should work to make this 10MB
- File system should borrow ideas from DB.

Smart Disks are (not yet) Here

- Disc controllers are cpus with ram and software (lots of it).
- Why not put general purpose software in controller?
- Put applications close to the data.

- This has not happened for cost reasons but it seems like a natural evolution.

Filler: How Often do Disks Fail?

Observed failure rates.					
System	**Source**	**Type**	**Part Years**	**Fails**	**Fails /Year**
TerraServer SAN	Barclay	SCSI 10krpm	858	24	2.8%
		controllers	72	2	2.8%
		san switch	9	1	11.1%
TerraServer Brick	Barclay	SATA 7krpm	138	10	7.2%
Web Property 1	anon	SCSI 10krpm	**15,805**	972	6.0%
		controllers	900	139	15.4%
Web Property 2	anon	PATA 7krpm	**22,400**	740	3.3%
		motherboard	3,769	66	1.7%

What About Bit Error Rates

- Uncorrectable Errors on Read (UERs)
  - quoted rates10^{-13} to 10^{-15}
  - That's 1 error in 1TB to 1 error in 100TB
  - WOW!!!
- We moved 1.5 PB looking for errors
- Saw 5 UER events
  - 4 real
  - 2 of them were masked by retry
- Saw many controller fails and system security reboots
- Conclusion:
  - UER not a useful metric – want mean time to data loss
  - UER better than advertised.

So, You Want to Copy a Petabyte?

- Today, that's 4,000 disks (from 2k to 2k)
- Takes 4 hours if they run in parallel, but…
- Probably not one file.
- You will see a few UERs.
- What's the best strategy?

UER things I wish I knew

- Better statistics from larger farms, and more diversity.
- What is the UER on a LAN, WAN?
- What is the UER over time:
  for a file on disk
  for a disk

- What's the best replication strategy?
  - Symmetric (1+1)+(1+1) or triplex (1+1) + 1

The Elephant In the Room
Parallel IO

- No standard Parallel IO interface
- Much talk about multi-core and multi-thread but not comparable talk about multi-disk.
- DB and MPIO and MapReduce are “niche” solutions.
- Perhaps there is no general-purpose File-IO abstraction, but… we should try harder.

Did I Confuse You?

- Architectural breaking points.
- Tape is dead, so what? (how to live without backup)
- Disk is infinite, so what?
- Overhead IOs are significant
- Checksum pages (somehow)
- Triplex and Geoplex will be standard
- LFS is in our future
- The guru gap is still very real
- Smart Disks are coming
- Schematized storage is coming
- Our fault models are dreadful