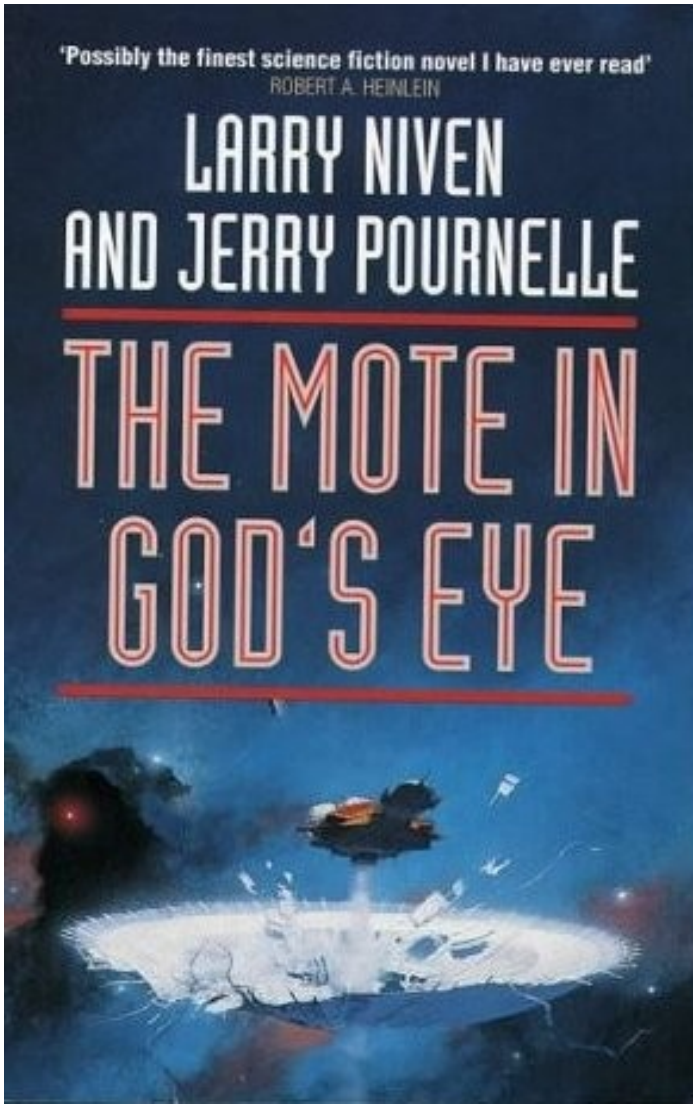
LOCKSS: Distributed Web Preservation Architecture

David S. H. Rosenthal
Vicky Reich

LOCKSS Program
Stanford University Libraries
http://www.lockss.org/

LOTS OF COPIES KEEP STUFF SAFE

Preserving Society's Knowledge



LOTS OF COPIES KEEP STUFF SAFE

Libraries: Robust Record

- Massively replicated, highly distributed
  - Collection policies mean more important more replicas
- Durable, write-once, tamper-evident media
  - Convincing fake printed books are hard & expensive
- Loosely coupled, independently administered
  - Failure of one library unlikely to affect others
- Failed slowly and gradually
  - Market in replicas = early warning of shortage

Libraries: Web Threat

- Web threatens library's role as memory
  - Many libraries used to own a copy, keep it in the stacks
  - Now they lease access to publisher's single copy

LOCKSS: Goals

- Restore libraries role as memory
  - Provide tools to collect & preserve Web content
- Revert to purchase model for content
  - Libraries continue to own copies of copyright content
- DMCA means need copyright permission
  - System must compromise between library & publisher
- What do publishers need to give permission?
  - Don't leak content, steal hits on content, re-brand content
- What do libraries need to build collections?
  - Local control of local copies of web content
  - Low cost-of-ownership collection & preservation

Practicalities

- Goal: minimal per-replica not per-byte cost
  - Don't ask “how few replicas do we need to be safe?”
  - Ask “how can many replicas reduce per-replica cost?”
- Goal: minimal barrier to entry
  - Librarians are risk-averse & impoverished
  - Enable learn-by-doing with cast-off hardware
- Media only a small part of storage cost
  - .36 media, .23 admin, .15 capital, .15 maint, .11 facility
- Storage only small part of preservation cost
  - 1hr of lawyer > 1TB of disk

Minimize Per-Replica Cost

- Hardware: consumer disks + generic PCs
  - Most library collections of published content not huge
  - E.g. all academic journals = 30-40TB
- Software: free, open-source
  - Re-use existing code as much as possible
- Sysadmin: de-skill via automation
  - Be fanatical about security of LOCKSS system
  - Package as network appliance – LOCKSS box
  - No backups – use replicas at other libraries
- User education: transparent content access
  - No need to educate users

What Are 100s Of Replicas Good For?

- Many, not very reliable replicas are a given
  - Can they cooperate to increase reliability?
  - Without leaking content to non-subscribers?
- LOCKSS boxes continually audit each other
  - By voting in polls on the hashes of content items
  - Agree with majority? Content OK.
  - Disagree with majority? Request repair from majority
- Remember history of agreement with boxes
  - Give repair only if agreement in previous polls
  - Repair isn't a leak; it can only replace pre-existing copy

Threat Model

- Media failure
- Hardware failure
- Software failure
- Network failure
- Obsolescence
- Natural Disaster

Threat Model

- Media failure
- Hardware failure
- Software failure
- Network failure
- Obsolescence
- Natural Disaster
- Operator error
- External Attack
- Insider Attack
- Economic Failure
- Organization Failure

Example: Disks

- Manufacturers specifications:
  - 10^6 hours MTTF
  - 10^{-14} unrecoverable bit error rate
- Schroeder & Pinheiro FAST '07 papers:
  - Field replacement rate 2-20 time MTTF
  - No "bathtub curve" of early failures
  - Enterprise disks 10x expensive, no more reliable
  - No correlation between temperature & failure
  - Significant autocorrelation – very bad for RAID
  - Significant long-rage correlation
  - SMART data logging not useful for failure prediction

Example: Software

File system code is carefully written & tested:

- Iron File System (Prabhakaran 2005):
  - Fault injection using pseudo-driver below file system
  - Bugs and inconsistencies in ext3, JFS, ReiserFS, NTFS
- FiSC (Yang 2006):
  - Model checking of file system code
  - 33 severe bugs in ext3, JFS, ReiserFS, XFS
  - Could destroy / in each file system
- Take away message:
  - The more you look, the more you find

Example: Insider Attack

- E.g. alienated system administrator
  - Major cause of system compromise (Keeney 2005)
  - Despite being massively under-reported
- E.g. piper calling the tune
  - Suppression or rewriting by government or funder
  - Hansen testimony to Waxman committee
- Paper record was fairly tamper-evident
  - How do we make electronic record tamper-evident?

LOCKSS Overview

- Each library runs a “persistent web cache”
  - Cache is never flushed
  - Caches cooperate to detect and repair damage
- Preloaded by a crawler with selected content
  - Crawler must be very slow and careful
  - Natural overlap of library collections = replication
- Readers use LOCKSS box like cache
  - Box forwards request to publisher + IfModifiedSince
  - OK, no reply, error = return preserved content
  - Otherwise return publisher content

Audit & Repair via Polls

- Poller box decides content needs auditing
  - Chooses timeframe, sample of boxes with content
  - Sends invitations to potential voter boxes
- If voter box schedules hashes in timeframe
  - Accepts invitation, waits for nonce from poller
  - Choose voter nonce, hashes nonces + content
  - Sends vote to poller, waits for receipt
- Poller hashes nonces + content, tallies votes
  - If disagree with majority, request repairs
  - Send receipts to voters

Formats

- LOCKSS is *format agnostic*
  - Collect and preserve any format delivered via HTTP
  - Content must be quasi-static
    - I.e. all viewers see the same important parts
    - Ads, etc. filtered by plugin before hashing
- LOCKSS supports *format migration*
  - Preserves only the original bits from the publisher
  - HTTP format negotiation to identify obsolete format
  - Trigger format converter to get temporary access copy
  - Deliver to browser with appropriate mime-type
    - I.e. not the obsolete one it was collected with

Deployment

- Went live 2004, 50 libraries.
- Now about 200 libraries worldwide
  - 6-weekly daemon releases, 6-monthly platform releases
- Now about 200 publishers worldwide
  - Weekly content releases of 100s of volumes
- Most publishers OK to ingest back content
  - Startup transient load >> sustained load
- System is low-maintenance, transparent
  - Easy to support, but easy to take for granted

Other Genres Of Content

- Open-Access
  - Specially important in the humanities
  - Eg: *World Haiku Review* rescued from LOCKSS
- Federal & State Documents
  - Eg: *Secrecy News* & its FOIA'ed documents
- Special Collections
  - Eg: MetaArchive of Southern Culture (NDIIPP project)
- Blogs – basic Blogger plugin just released
  - Eg: blog.dshr.org

Measuring Performance

- Long-term storage is a big market
  - Without a performance benchmark!
  - Benchmarks drive mature tech markets
- My suggested benchmark: bit half-life
  - Look at a bit in a storage system
  - How long until 50% chance it has flipped?
- Technology cost/performance axes
  - Cost: $/bit/yr
  - Performance: bit half-life

Petabyte for a Century

- Suppose need to keep petabyte for century
  - With 50% chance of every bit surviving undamaged
  - Now that's big, in 100 years its 10^{-9} of a hard drive
- 0.8 exabit-year with 50% survival unimpaired
  - Consider possibility of *bit rot* affecting the system
  - Radioactivity analogy, small probability of bit flip
  - Bit half-life $0.8*10^{18}$ yr = ~100M times age of universe
- Can we test that systems are this reliable?
  - Watch exabyte for year, see ~5 bit flips? Not feasible.
  - Requirement is ~10,000 times our ability to test
  - CERN tests see ~10,000 times higher bit flip rate

Credits

- LOCKSS Engineering Team (since 1998)
  - Tom Lipkis, Tom Robertson, Seth Morabito, Thib G-C.
- LOCKSS Research Team (since 2001)
  - Best Paper @ SOSP2003
  - Mary Baker, Mehul Shah & colleagues @ HP Labs
  - Mema Roussopoulos & students @ Harvard CS
  - Petros Maniatis & interns @ Intel Research Berkeley
- Funding from
  - libraries, Sun, NSF, Mellon, LoC, publishers, ...