LOCKSS In The Cloud

David S. H. Rosenthal

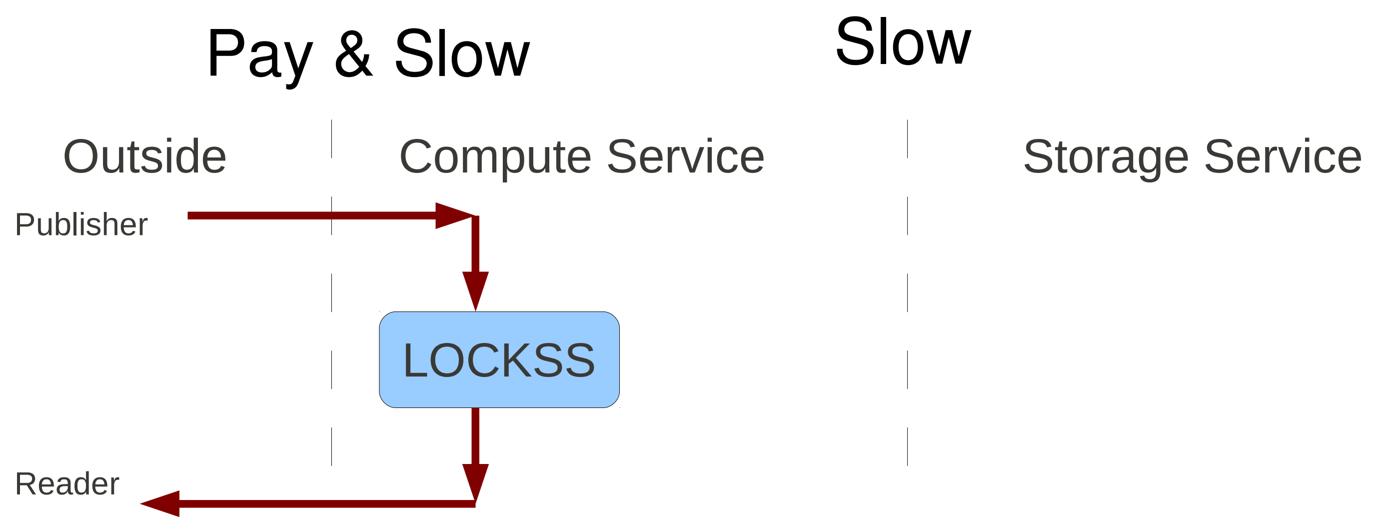
LOCKSS Program
Stanford University Libraries
http://www.lockss.org/
http://blog.dshr.org

LOTS OF COPIES KEEP STUFF SAFE

Overview

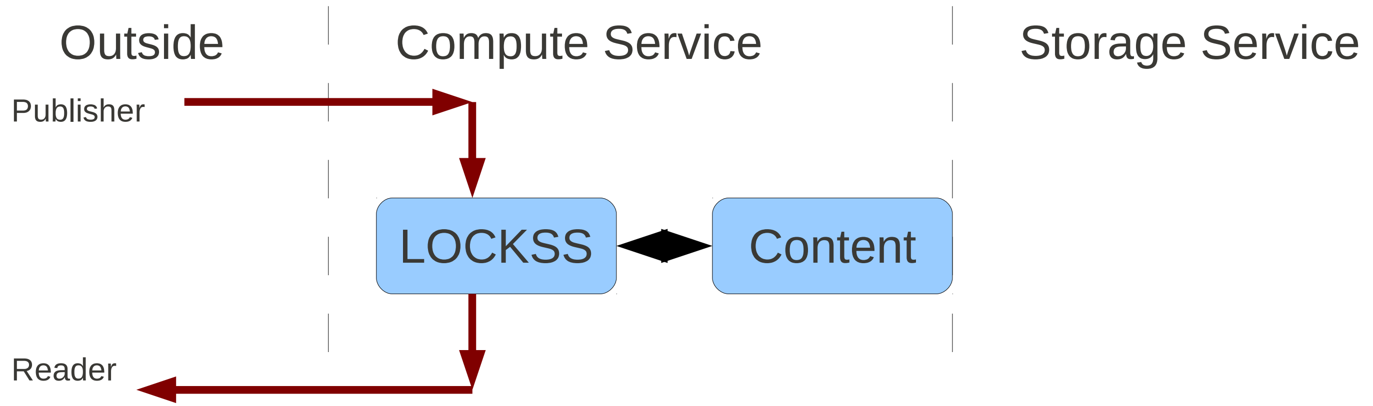
- 7 possible LOCKSS in cloud architectures
  - A look at each, with current status
- Remaining cloud issues
  - Service independence
  - Packaging
  - Scheduling to a budget
- Non-cloud developments
  - Bibliographic metadata support
  - Advanced ingest technology

Compute Instance



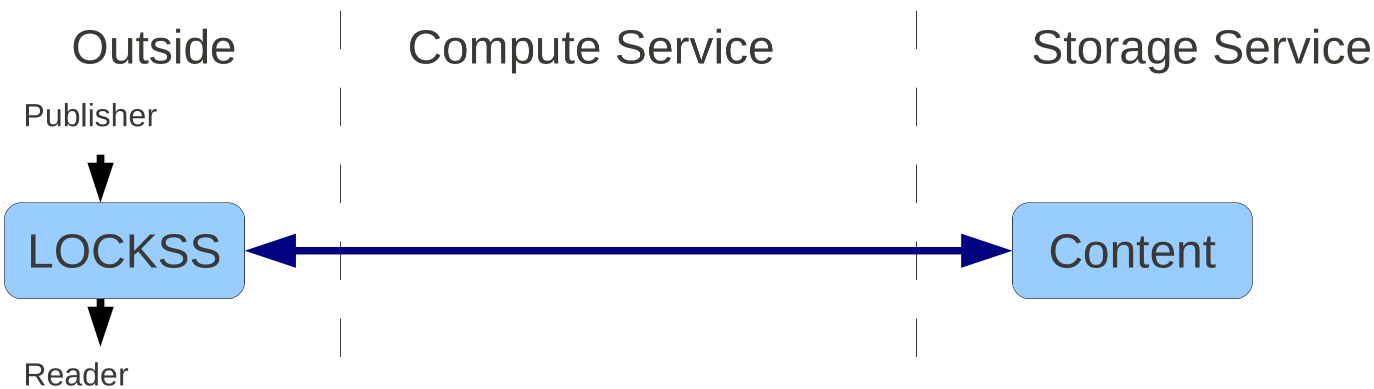
- This just works
- But it isn't useful, even in a large instance
  - <2TB total storage
  - Storage goes away with instance

Instance + EBS



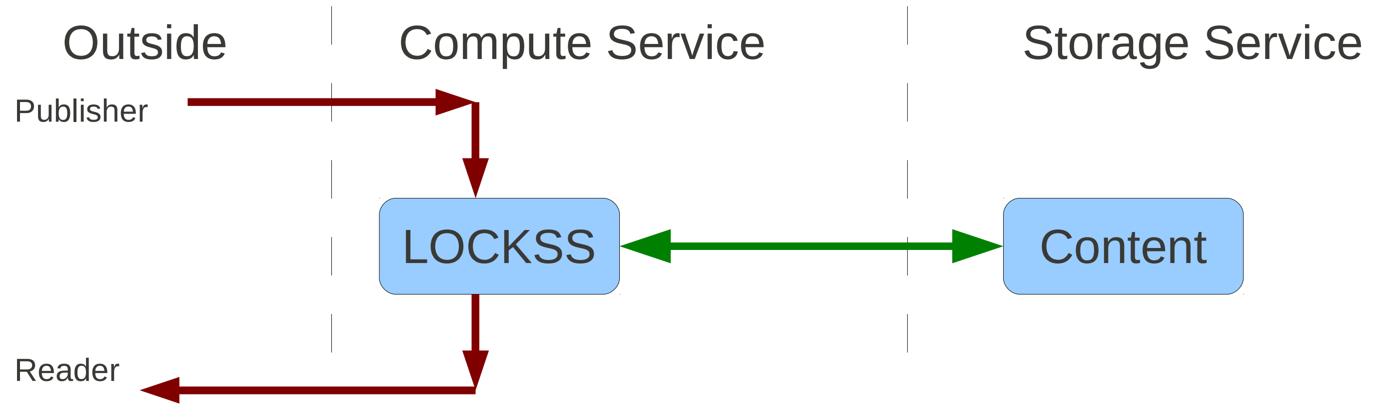
- This works for Private LOCKSS networks
  - Can get many TB of persistent, fairly reliable storage
- Problems for Global LOCKSS network
  - Amazon IP address – use VPN for subscription content?

Storage Service Client



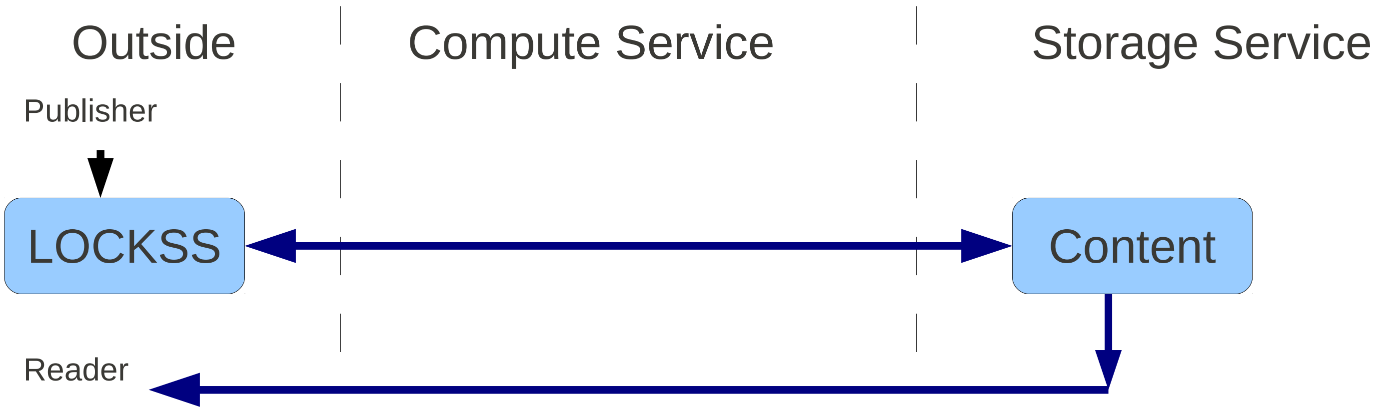
- Prototype working with S3, Walrus, IAS3
  - But with miserable cost & performance
- Need to re-architect LOCKSS repository
  - Minimize interactions with storage

Instance + Storage Service



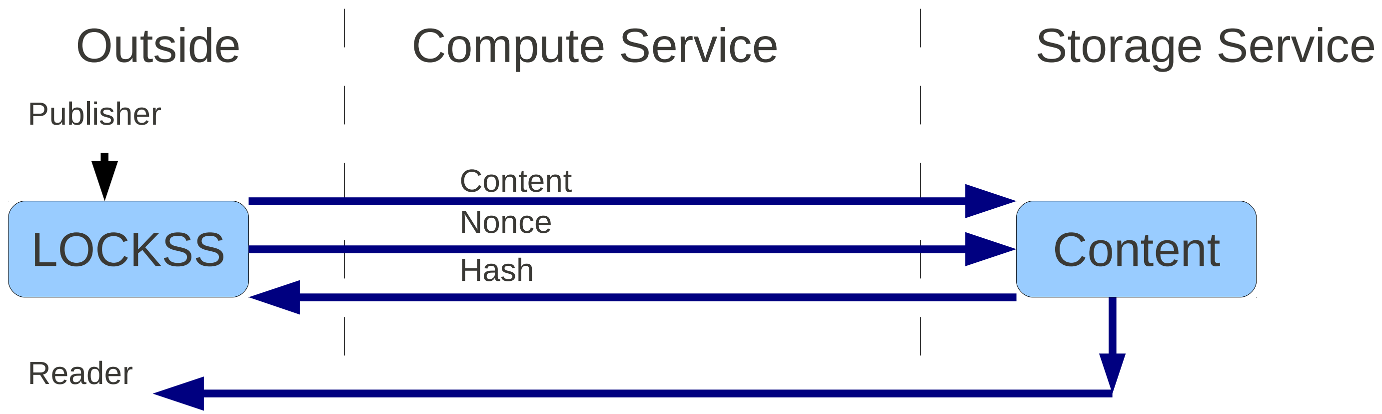
- This should now work
  - Haven't tried it yet
  - Better performance but still impractically slow

Client + Memento



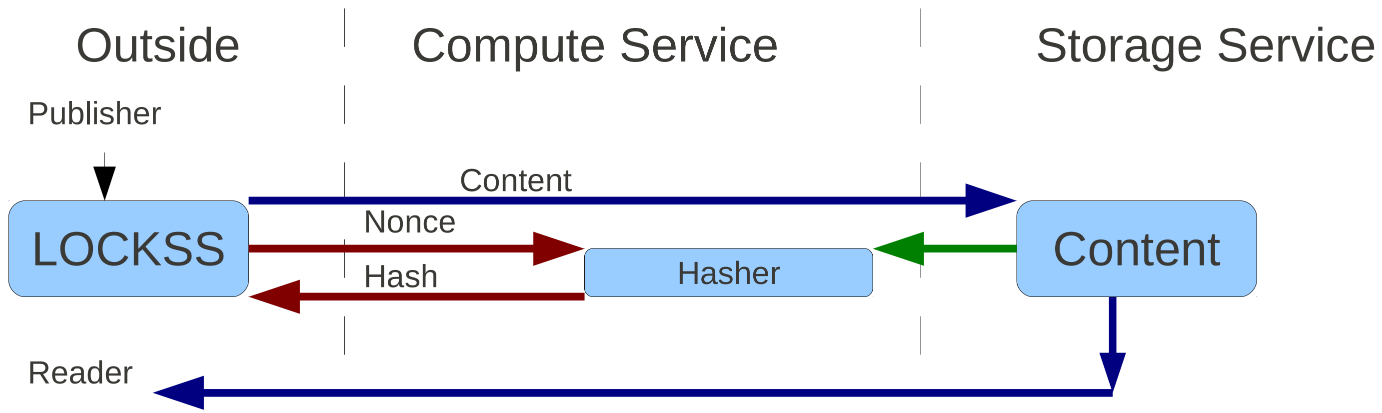
- LOCKSS acting as Memento Aggregator
  - Readers redirected to content in Storage Service
- Probably not big effect on cost/performance
  - But makes quite a bit of LOCKSS code redundant

Client + Memento + Nonce



- Service enhanced – adds nonce to hash
  - Easy for Walrus, IAS3, hard for S3
- Big improvement in cost/performance
  - Details once we've done the experiment

Split Client



- Worse cost/performance, more complex
  - Will work with vanilla S3
- This may well be where we end up
  - Only way to be service-independent for now

Issues

- Service-agnostic is tricky
  - Basic compatibility, many minor differences
- Packaging is tricky
  - E.g. setting up VPN for daemon in compute service
- Scheduling vs. budget
  - LOCKSS tried to keep CPU & I/O 100% busy
  - Cloud needs to keep within monthly budget

Other Developments

- Metdata, SFX, OpenURLs, etc
  - OpenURL resolution to article & table of contents
  - KBART support to list holdings for link resolvers
  - For plugins with metadata extractors
- Ingest with Carnegie-Mellon West
  - Prototype of robust approach to form-filling
    - Important for e.g. Government Documents
  - Proof-of-concept of ingesting AJAX sites
    - E.g new Royal Society of Chemistry site