Distributed Data Integrity Assurance and Repair Using the LOCKSS Content Audit Protocol (LCAP)

Thib Guicherd-Callin
LOCKSS Program, Stanford University
Outline

1. Context and Use Cases
2. Threat Models
3. LCAP in Action
4. Unlocking LOCKSS for Developers
5. Q&A
Outline

1. Context and Use Cases
2. Threat Models
3. LCAP in Action
4. Unlocking LOCKSS for Developers
5. Q&A
Traditional Research Libraries

- Ownership model
- Many independent replicas

Features
- Disaster resistance
- Disaster recovery
- Tamper evident
- Permanent access
Research Libraries in the Web Era

- Leasing model
- One master copy
- Misfeatures
  - Disaster resistance?
  - Disaster recovery?
  - Tamper evident?
  - Permanent access?
LOCKSS Technology Use Cases

- "Lots Of Copies Keep Stuff Safe"
- Global LOCKSS Network (GLN)
- CLOCKSS Archive
- Government documents networks
- Regional and national networks
Key Publications

- "Founding paper"
Key Publications

- "Protocol paper"

Key Publications

- "Threat models paper"
Outline

1. Context and Use Cases
2. Threat Models
3. LCAP in Action
4. Unlocking LOCKSS for Developers
5. Q&A
Goal of Digital Preservation

The goal of a digital preservation system is that the information it contains remains accessible to users over a period of time much longer than the lifetime of individual storage media, hardware and software components.
Key Properties

- No single point of failure
- Media, hardware and software flow through as they fail or are replaced
- Regular audits frequent enough to keep probability of irrecoverable failure acceptable
Threat Taxonomy (1)

- Media failure
- Hardware failure
- Software failure
- Communication errors
- Failure of network services
- Natural disaster
Threat Taxonomy (2)

- Media and hardware obsolescence
- Software obsolescence
Threat Taxonomy (3)

- Operator error
- Economic failure
- Organizational failure
Threat Taxonomy (4)

- External attack
- Internal attack
Outline

1. Context and Use Cases
2. Threat Models
3. LCAP in Action
4. Unlocking LOCKSS for Developers
5. Q&A
Basic Principle
What is hash(X)?

The peers hold identical replicas of X
Peer P1 calls a poll on content X
P2, P3, P4, P5, P6 agreed with me on X

hash(X) = h1

Landslide agreement

hash(X) = h1
What is hash(X)?

Peer P2 calls a poll on content X
hash(X) = h1

P1, P3, P4, P5, P6 agreed with me on X

hash(X) = h1

hash(X) = h1

P2

hash(X) = h1

P3

hash(X) = h1

P5

hash(X) = h1

P4

Landslide agreement

P1, P3, P4, P5, P6 agreed with me on X

hash(X) = h1

P2

hash(X) = h1

P3

hash(X) = h1

P4

hash(X) = h1

P5

Landslide agreement
What is hash(X)?

Peer P1 incurs damage on content X
Peer P1 later calls a poll on content X
Landslide disagreement

hash(X) = h1

hash(X) = h1

hash(X) = h1

hash(X) = h1

hash(X) = h1

hash(X) = h1

hash(X) = h1

hash(X) = h2

P1

P2

P3

P4

P5

P6
Help me repair X

Repair request
P1 agreed with me on X

P2

Repair

P3

P4

P1

P6

P5
The peers hold identical replicas of X
Stealth Modification Gap

- Landslide agreement: take no action (high confidence in outcome)
- Inconclusive agreement: take no action and raise alarm (low confidence in outcome)
- Landslide disagreement: seek repair and notify (high confidence in outcome)
Nonces

- For each voter in a poll over X, the poller supplies a poller nonce P and the voter a voter nonce V
- Rather than hash(X), it is the value of hash(PVX) that is computed
- Nonces must be fresh
Repair Verification

- Byzantine fault
- Bait and switch
Physical Fixity vs. Logical Fixity

- What if the peers hold the same content even though not all of, or even none of, the replicas are byte-identical?
LCAP vs. Threats

- Media failure
- Hardware failure
- Software failure
- Communication errors
- Failure of network services
- Natural disaster

- Media and hardware obsolescence
- Software obsolescence
- Operator error
- Economic failure
- Organizational failure
- External attack
- Internal attack
Outline

1. Context and Use Cases
2. Threat Models
3. LCAP in Action
4. Unlocking LOCKSS for Developers
5. Q&A
Narrow Origins

- Audience: research libraries
- Target: Web content
- Context: appliance model
- "Monolithic stack"
LAAWS Initiative

- "LOCKSS Architected As Web Services"
- Two year Mellon Foundation grant
- Modernization effort
Re-Architecture
REST APIs and Software Assets

- Repository service
  - https://github.com/lockss/laaws-repository-service
- Configuration service
  - https://github.com/lockss/laaws-configuration-service
- Poller service
  - https://github.com/lockss/laaws-poller
- Metadata extraction service
  - https://github.com/lockss/laaws-metadataextractor
- Metadata service
  - https://github.com/lockss/laaws-metadataservice
- develop branch → docs/swagger.yaml
Dev/Demo Environment

- [ ] https://github.com/lockss/laaws-demo
- [ ] feature-mgdemo branch
- [ ] Contains:
  - Docker support infrastructure
  - Docker flavor
  - JAR flavor
Outline

1. Context and Use Cases
2. Threat Models
3. LCAP in Action
4. Unlocking LOCKSS for Developers
5. Q&A
Thank you